SAMA5D2 Series

Atmel | SMART ARM-based Embedded MPU

DATASHEET

Introduction

The Atmel® | SMART SAMA5D2 series is a high-performance, ultra-low-power ARM® Cortex®-A5 processor-based MPU running up to 500 MHz, with support for multiple memories such as DDR2, DDR3, DDR3L, LPDDR1, LPDDR2, LPDDR3, and QSPI Flash. The devices integrate powerful peripherals for connectivity and user interface applications, and offer advanced security functions (ARM TrustZone®, tamper detection, secure data storage, etc.), as well as high-performance crypto-processors AES, SHA and TRNG.

The SAMA5D2 series is delivered with a free Atmel Linux distribution and bare metal C examples.

Features

- ARM Cortex-A5 core
  - ARMv7-A architecture
  - ARM TrustZone
  - NEON™ Media Processing Engine
  - Up to 500 MHz
  - ETM/ETB 8 Kbytes
- Memory Architecture
  - Memory Management Unit
  - 32-Kbyte L1 data cache, 32-Kbyte L1 instruction cache
  - 128-Kbyte L2 cache configurable to be used as an internal SRAM
  - One 128-Kbyte scrambled internal SRAM
  - One 160-Kbyte internal ROM
    - 64-Kbyte scrambled and maskable ROM embedding Atmel boot loader/Atmel Secure boot loader
    - 96-Kbyte unscrambled, unmaskable ROM for NAND Flash BCH ECC table
  - High-bandwidth scramblable 16-bit or 32-bit Double Data Rate (DDR) multiport dynamic RAM controller supporting up to 512 Mbyte 8-bank DDR2/DDR3 (DLL off only)/DDR3L (DLL off only)/LPDDR1/LPDDR2/LPDDR3, including “on-the-fly” encryption/decryption path
  - 8-bit SLC/MLC NAND controller, with up to 32-bit Error Correcting Code (PMECC)

- System running up to 166 MHz in typical conditions
  - Reset controller, shutdown controller, periodic interval timer, independent watchdog timer and secure Real-Time Clock (RTC) with clock calibration
  - One 600 to 1200 MHz PLL for the system and one 480 MHz PLL optimized for USB high speed
  - Digital fractional PLL for audio (11.2896 MHz and 12.288 MHz)
  - Internal low-power 12 MHz RC and 32 KHz typical RC
  - Selectable 32.768-Hz low-power oscillator and 8 to 24 MHz oscillator
  - 51 DMA Channels including two 16-channel 64-bit Central DMA Controllers
  - 64-bit Advanced Interrupt Controller (AIC)
  - 64-bit Secure Advanced Interrupt Controller (SAIC)
  - Three programmable external clock signals
- Low-Power Modes
  - Ultra Low-power mode with fast wakeup capability
  - Low-power Backup mode with 5-Kbyte SRAM and SleepWalking™ features
    - Wakeup from up to nine wakeup pins, UART reception, analog comparison
    - Fast wakeup capability
    - Extended Backup mode with DDR in Self-Refresh mode
- Peripherals
  - LCD TFT controller up to 1024x768, with four overlays, rotation, post-processing and alpha blending, 24-bit parallel RGB
  - ITU-R BT. 601/656/1120 Image Sensor Controller (ISC) supporting up to 5 M-pixel sensors with a parallel 12-bit interface for Raw Bayer, YCbCr, Monochrome and JPEG-compressed sensor interface
  - Two Synchronous Serial Controllers (SSC), two Inter-IC Sound Controllers (I^2SC), and one Stereo Class D amplifier
  - One Pulse Density Modulation Interface Controller (PDMIC)
  - One USB high-speed device port (UDPHS) and one USB high-speed host port or two USB high-speed host ports (UHPHS)
  - One USB high-speed host port with a High-Speed Inter-Chip (HSIC) interface
  - One 10/100 Ethernet MAC (GMAC)
    - Energy efficiency support (IEEE 802.3az standard)
    - Ethernet AVB support with IEEE802.1AS time stamping
    - IEEE802.1Qav credit-based traffic-shaping hardware support
    - IEEE1588 Precision Time Protocol (PTP)
  - Two high-speed memory card hosts:
    - SDMMC0: SD 3.0, eMMC 4.51, 8 bits
    - SDMMC1: SD 2.0, eMMC 4.41, 4 bits only
  - Two master/slave Serial Peripheral Interfaces (SPI)
  - Two Quad Serial Peripheral Interfaces (QSPI)
  - Five FLEXCOMs (USART, SPI and TWI)
  - Five UARTs
  - Two master CAN-FD (MCAN) controllers with SRAM-based mailboxes, and time- and event-triggered transmission
  - One Rx only UART in backup area (RXLP)
  - One analog comparator (ACC) in backup area
  - Two 2-wire interfaces (TWIHS) up to 400 Kbits/s supporting the I^2C protocol and SMBUS (TWIHS)
  - Two 3-channel 32-bit Timer/Counters (TC), supporting basic PWM modes
  - One full-featured 4-channel 16-bit Pulse Width Modulation (PWM) controller
  - One 12-channel, 12-bit, Analog-to-Digital Converter (ADC) with Resistive TouchScreen capability

- Safety
  - Zero-power Power-On Reset (POR) cells
  - Main crystal clock failure detector
  - Write-protected registers
  - Integrity Check Monitor (ICM) based on SHA256
  - Memory Management Unit
  - Independent watchdog
- Security
  - 5 Kbytes of internal scrambled SRAM:
    - 1 Kbyte non-erasable on tamper detection
    - 4 Kbytes erasable on tamper detection
  - 256 bits of scrambled and erasable registers
  - Eight tamper pins for static or dynamic intrusion detections
  - Environmental monitors on secured versions: temperature, voltage, frequency and active die shield

Note: For environmental monitors, refer to datasheet "SAMA5D2 Security Module" (Atmel literature No. 44036), available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for details.

  - Atmel Secure Boot

Note: For secure boot strategies, refer to application note "SAMA5D2x Secure Boot Strategy" (Atmel literature No. 44040), available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for details.

  - On-the-fly AES encryption/decryption on DDR and QSPI memories (AESB)
  - RTC including time-stamping on security intrusions
  - Programmable fuse box with 544 fuse bits (including JTAG protection and BMS)
- Hardware cryptography
  - SHA (SHA1, SHA224, SHA256, SHA384, SHA512): compliant with FIPS PUB 180-2
  - AES: 256-, 192-, 128-bit key algorithm, compliant with FIPS PUB 197
  - TDES: two-key or three-key algorithms, compliant with FIPS PUB 46-3
  - True Random Number Generator (TRNG) compliant with NIST Special Publication 800-22 Test Suite and FIPS PUBs 140-2 and 140-3
- Up to 128 I/Os
  - Fully programmable through set/clear registers
  - Multiplexing of up to eight peripheral functions per I/O line
  - Each I/O line can be assigned to a peripheral or used as a general purpose I/O
  - The PIO controller features a synchronous output providing up to 32 bits of data output in one write operation
- Packages
  - 289-ball LFBGA, 14 x 14 mm body, 0.8 mm pitch
  - 256-ball TFBGA, 8 x 8 mm body, 0.4 mm pitch
  - 196-ball TFBGA, 11 x 11 mm body, 0.75 mm pitch

1. Description

The Atmel | SMART SAMA5D2 Series is a high-performance, power-efficient embedded MPU based on the ARM Cortex-A5 processor. It integrates the ARM NEON SIMD engine for accelerated multimedia and signal processing, a configurable 128-Kbyte L2 cache, a floating point unit for high-precision computing and reliable performance, as well as high data bandwidth architecture. The device features an advanced user interface and connectivity peripherals. Advanced security is provided by powerful cryptographic accelerators, by the ARM TrustZone technology securing access to memories and sensitive peripherals, and by several hardware features that safeguard memory content, authenticate software reliability, detect physical attacks and prevent information leakage during code execution.

The SAMA5D2 features an internal multilayer bus architecture associated with 2 x 16 DMA channels and dedicated DMAs for the communication and interface peripherals required to ensure uninterrupted data transfers with minimal processor overhead. The device supports DDR2, DDR3, DDR3L, LPDDR1, LPDDR2, LPDDR3, and SLC/MLC NAND Flash memory up to 32-bit ECC.

The comprehensive peripheral set includes an LCD TFT controller with overlays for hardware-accelerated image composition, an image sensor controller, audio support through I^2S, SSC, a stereo Class D amplifier and a digital microphone. Connectivity peripherals include a 10/100 EMAC, USBs, CAN-FDs, FLEXCOMs, UARTs, SPIs and two QSPIs, SDIO/SD/e.MMCs, and TWIs/I^2C.

Protection of code and data is provided by automatic scrambling of memories and an Integrity Check Monitor (ICM) to detect any modification of the memory contents. The SAMA5D2 also supports execution of encrypted code (QSPI or one portion of the DDR) with an “on-the-fly” encryption-decryption process.

With its secure design architecture, cryptographic acceleration engines, and secure boot loader, the SAMA5D2 is the ideal solution for point-of-sale (POS), IoT and industrial applications requiring anti-cloning, data protection and secure communication transfer.

SAMA5D2 devices feature three software-selectable low-power modes: Idle, Ultra-low-power and Backup.

In Idle mode, the processor is stopped while all other functions can be kept running.

In Ultra-low-power-mode 0, the processor is stopped while all other functions are clocked at 512 Hz and interrupts or peripherals can be configured to wake up the system based on events, including partial asynchronous wakeup (SleepWalking).

In Ultra-low-power mode 1, all clocks and functions are stopped but some peripherals can be configured to wake up the system based on events, including partial asynchronous wakeup (SleepWalking).

In Backup mode, RTC and wakeup logic are active. The Backup mode can be extended to feature DDR in Self-refresh mode.

SAMA5D2 devices also include an Event System that allows peripherals to receive, react to and send events in Active and Idle modes without processor intervention.

2. Configuration Summary

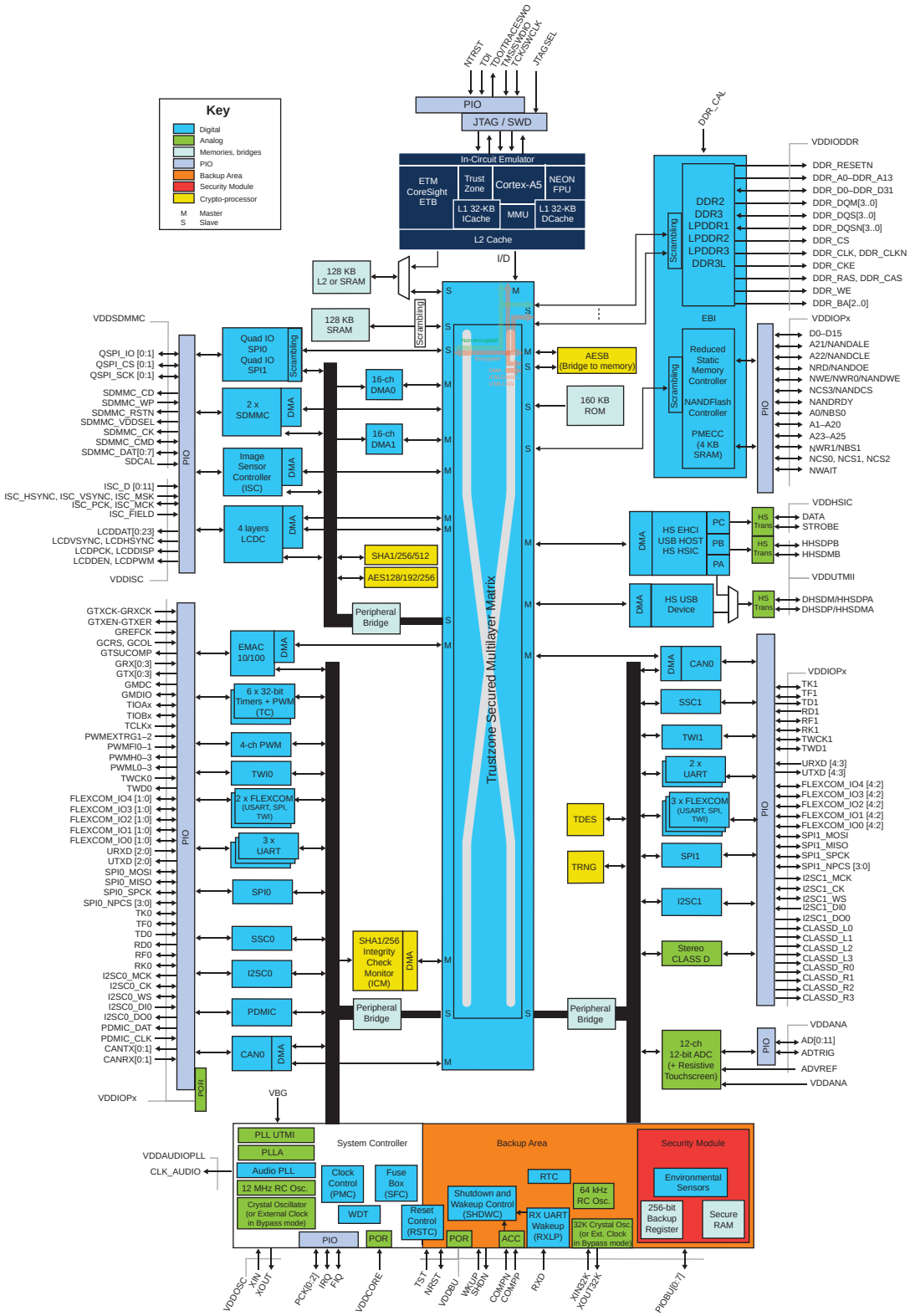
Table 2-1. SAMA5D2 Configuration Summary

Feature	SAMA5D21	SAMA5D22	SAMA5D23	SAMA5D24	SAMA5D26	SAMA5D27	SAMA5D28
Package	TFBGA196			TFBGA256	LFBGA289		
PIOs	72			105	128		
DDR Bus	16-bit			16/32-bit			
SRAM	128 Kbytes						
QSPI	2						
LCD	24-bit RGB						
Camera Interface (ISC)	1						
EMAC	1						
CAN	–	1		–		2	
USB	2 (2 Hosts or 1 Host/1 Device)			3 (2 Hosts/ 1 HSIC, or 1 Host/ 1 Device/ 1 HSIC)	2 (2 Hosts or 1 Host/ 1 Device)	3 (2 Hosts/1 HSIC or 1 Host/1 Device/1 HSIC)	
UART/SPI/I^2C	9 / 6 / 6			10 / 7 / 7			
SDIO/SD/MMC	1			2			
I^2S/SSC/ Class D/PDM	2 / 2 / 1 / 1						
ADC Inputs	5			12			
Timers	6						
PWM	4 (PWM) + 5 (TC)			4 (PWM) + 6 (TC)			
Tamper Pins	6			2	8		
AESB	–	Yes			–	Yes	
Environmental Monitors, Die Shield	–		Yes	–			Yes

For information on device pin compatibility, refer to Section 5.2 “Pinouts”.

3. Block Diagram

Figure 3-1. SAMA5D2 Series Block Diagram



Note: See Section 35. “DMA Controller (XDMAC)” for peripheral connections to DMA.

4. Signal Description

Table 4-1 gives details on signal names classified by peripheral.

Table 4-1. Signal Description List

Signal Name	Function	Type	Comments	Active Level
Clocks, Oscillators and PLLs				
XIN	Main Oscillator Input	Input	–	–
XOUT	Main Oscillator Output	Output	–	–
XIN32	Slow Clock Oscillator Input	Input	–	–
XOUT32	Slow Clock Oscillator Output	Output	–	–
CLK_AUDIO	Audio Clock	Output	–	–
VBG	Bias Voltage Reference for USB	Analog	–	–
PCK 0–2	Programmable Clock Output	Output	Reset State: - PIO Input - Internal Pull-up enabled - Schmitt Trigger enabled	–
Shutdown, Wakeup Logic				
SHDN	Shutdown Control	Output	–	–
PIOBU 0–7	Tamper or Wakeup Inputs	Input	–	–
WKUP	Wakeup Input	Input	–	–
ICE and JTAG				
TCK/SWCLK	Test Clock/Serial Wire Clock	Input	–	–
TDI	Test Data In	Input	–	–
TDO	Test Data Out	Output	–	–
TMS/SWDIO	Test Mode Select/Serial Wire Input/Output	I/O	–	–
JTAGSEL	JTAG Selection	Input	–	–
Reset/Test				
NRST	Microprocessor Reset	Input	–	Low
TST	Test Mode Select	Input	–	–
NTRST	Test Reset Signal	Input	–	–
Advanced Interrupt Controller - AIC				
IRQ	External Interrupt Input	Input	–	–
Secured Advanced Interrupt Controller - SAIC				
FIQ	Fast Interrupt Input	Input	–	–
PIO Controller				
PA0–PAxx	Parallel IO Controller	I/O	–	–
PB0–PBxx	Parallel IO Controller	I/O	–	–
PC0–PCxx	Parallel IO Controller	I/O	–	–
PD0–PDxx	Parallel IO Controller	I/O	–	–

Table 4-1. Signal Description List (Continued)

Signal Name	Function	Type	Comments	Active Level
External Bus Interface - EBI				
D[15:0]	Data Bus	I/O	–	–
A[25:0]	Address Bus	Output	–	–
NWAIT	External Wait Signal	Input	–	Low
Static Memory Controller - HSMC				
NCS0–NCS3	Chip Select Lines	Output	–	Low
NWR0–NWR1	Write Signal	Output	–	Low
NRD	Read Signal	Output	–	Low
NWE	Write Enable	Output	–	Low
NBS0–NBS1	Byte Mask Signal	Output	–	Low
NANDOE	NAND Flash Output Enable	Output	–	Low
NANDWE	NAND Flash Write Enable	Output	–	Low
DDR2/DDR3/LPDDR1/LPDDR2/LPDDR3 Controller				
DDR_CK, DDR_CKN	DDR differential clock	Output	–	–
DDR_CKE	DDR Clock Enable	Output	When Backup Self-refresh mode is used, should be tied to GND using 100 KΩ pull-down	High
DDR_CS	DDR Controller Chip Select	Output	–	Low
DDR_BA[2:0]	Bank Select	Output	–	Low
DDR_WE	DDR Write Enable	Output	–	Low
DDR_RAS, DDR_CAS	Row and Column Signal	Output	–	Low
DDR_A[13:0]	DDR Address Bus	Output	–	–
DDR_D[31:0]	DDR Data Bus	I/O/-PD	–	–
DDR_DQS[3:0], DDR_DQSN[3:0]	Differential Data Strobe	I/O- PD	–	–
DDR_DQM[3:0]	Write Data Mask	Output	–	–
DDR_CAL	DDR/LPDDR calibration	Input	–	–
DDR_VREF	DDR/LPDDR reference	Input	–	–
DDR_RESETN	DDR3 Active Low Asynchronous Reset	Output	When Backup Self-refresh mode is used, should be tied to VDDIODDR using 100 KΩ pull-up	–
Secure Data Memory Card - SDMMCx [1:0]				
SDMMCx_CD	SDcard / e.MMC Card Detect	Input	–	–
SDMMCx_CMD	SDcard / e.MMC Command line	I/O	–	–
SDMMCx_WP	SDcard connector write protect signal	Input	–	–
SDMMCx_RSTN	e.MMC reset signal	Output	–	–
SDMMCx_VDDSEL	SDcard signal voltage selection	Output	–	–
SDMMCx_CK	SDcard / e.MMC clock signal	Output	–	–

Table 4-1. Signal Description List (Continued)

Signal Name	Function	Type	Comments	Active Level
SDMMCx_DAT[7:0]	SDcard / e.MMC data lines	I/O	–	–
Flexible Serial Communication Controller - FLEXCOMx [4:0]				
FLEXCOMx_IO0	FLEXCOMx Transmit Data	I/O	–	–
FLEXCOMx_IO1	FLEXCOMx Receive Data	I/O	–	–
FLEXCOMx_IO2	FLEXCOMx Serial Clock	I/O	–	–
FLEXCOMx_IO3	FLEXCOMx Clear To Send / Peripheral Chip Select	I/O	–	–
FLEXCOMx_IO4	FLEXCOMx Request To Send / Peripheral Chip Select	Output	–	–
Universal Asynchronous Receiver Transmitter - UARTx [4..0]				
UTXDx	UARTx Transmit Data	Output	–	–
URXDx	UARTx Receive Data	Input	–	–
Inter-IC Sound Controller - I2SCx [1..0]				
I2SCx_MCK	Master Clock	Output	–	–
I2SCx_CK	Serial Clock	I/O	–	–
I2SCx_WS	I^2S Word Select	I/O	–	–
I2SCx_DI0	Serial Data Input	Input	–	–
I2SCx_DO0	Serial Data Output	Output	–	–
Synchronous Serial Controller - SSCx [1..0]				
TDx	SSC Transmit Data	Output	–	–
RDx	SSC Receive Data	Input	–	–
TKx	SSC Transmit Clock	I/O	–	–
RKx	SSC Receive Clock	I/O	–	–
TFx	SSC Transmit Frame Sync	I/O	–	–
RFx	SSC Receive Frame Sync	I/O	–	–
Timer/Counter - TCx [5..0]				
TCLKx	TC Channel x External Clock Input	Input	–	–
TIOAx	TC Channel x I/O Line A	I/O	–	–
TIOBx	TC Channel x I/O Line B	I/O	–	–
Quad IO SPI - QSPIx [1..0]				
QSPIx_SCK	QSPI serial Clock	Output	–	–
QSPIx_CS	QSPI Chip Select	Output	–	–
QSPIx_IO[0..3]	QSPI I/O QIO0 is QMOSI Master Out - Slave In QIO1 is QMISO Master In - Slave Out	I/O	–	–

Table 4-1. Signal Description List (Continued)

Signal Name	Function	Type	Comments	Active Level
Serial Peripheral Interface - SPIx [1..0]				
SPIx_MISO	Master In Slave Out	I/O	–	–
SPIx_MOSI	Master Out Slave In	I/O	–	–
SPIx_SPCK	SPI Serial Clock	I/O	–	–
SPIx_NPCS0	SPI Peripheral Chip Select 0	I/O	–	Low
SPIx_NPCS[3..1]	SPI Peripheral Chip Select	Output	–	Low
Two-wire Interface - TWIx [1..0]				
TWDx	Two-wire Serial Data	I/O	–	–
TWCKx	Two-wire Serial Clock	I/O	–	–
Pulse Width Modulation Controller - PWM				
PWMH0–3	PWM Waveform Output High	Output	–	–
PWML0–3	PWM Waveform Output Low	Output	–	–
PWMFI0–1	PWM Fault Inputs	Input	–	–
PWMEXTRG1–2	PWM External Trigger	Input	–	–
USB Host High Speed Port - UHPHS				
HHSDPA	USB Host Port A High Speed Data +	Analog	–	–
HHSDMA	USB Host Port A High Speed Data -	Analog	–	–
HHSDPB	USB Host Port B High Speed Data +	Analog	–	–
HHSDMB	USB Host Port B High Speed Data -	Analog	–	–
USB Device High Speed Port - UDPHS				
DHSDP	USB Device High Speed Data +	Analog	–	–
DHSDM	USB Device High Speed Data -	Analog	–	–
USB High-Speed Inter-Chip Port - HSIC				
HHSTROBE	USB High-Speed Inter-Chip Strobe	I/O	–	–
HHDATA	USB High-Speed Inter-Chip Data	I/O	–	–
Ethernet 10/100 - GMAC				
GREFCK	Reference Clock	Input	–	–
GTXCK	Transmit Clock	Input	–	–
GRXCK	Receive Clock	Input	–	–
GTXEN	Transmit Enable	Output	–	–
GTX0–GTX3	Transmit Data	Output	–	–
GTXER	Transmit Coding Error	Output	–	–
GRXDV	Receive Data Valid	Input	–	–
GRX0–GRX3	Receive Data	Input	–	–
GRXER	Receive Error	Input	–	–
GCRS	Carrier Sense	Input	–	–

Table 4-1. Signal Description List (Continued)

Signal Name	Function	Type	Comments	Active Level
GCOL	Collision Detected	Input	–	–
GMDC	Management Data Clock	Output	–	–
GMDIO	Management Data Input/Output	I/O	–	–
GTSUCOMP	TSU timer comparison valid	Output	–	–
LCD Controller - LCDC				
LCDDAT0–23	LCD Data Bus	Output	–	–
LCDVSYNC	LCD Vertical Synchronization	Output	–	–
LCDHSYNC	LCD Horizontal Synchronization	Output	–	–
LCDPCK	LCD pixel Clock	Output	–	–
LCDDEN	LCD Data Enable	Output	–	–
LCDPWM	LCDPWM for Contrast Control	Output	–	–
LCDDISP	LCD Display ON/OFF	Output	–	–
Touchscreen Analog-to-Digital Converter - ADC				
AD0–11	12 Analog Inputs	Analog	–	–
ADTRG	ADC Trigger	Input	–	–
ADVREF	ADC Reference	Analog	–	–
Secure Box Module - SBM				
PIOBU0–7	Tamper I/Os	I/O	–	–
Image Sensor Controller - ISC				
ISC_D0–ISC_D11	Image Sensor Data	Input	–	–
ISC_HSYNC	Image Sensor Horizontal Synchro	Input	–	–
ISC_VSYNC	Image Sensor Vertical Synchro	Input	–	–
ISC_PCK	Image Sensor Pixel clock	Input	–	–
ISC_MCK	Image Sensor Main clock	Output	–	–
ISC_FIELD	Field identification signal	Input	–	–
Audio Class Amplifier - CLASSD				
CLASSD_L0	CLASSD Left Output L0	Output	–	–
CLASSD_L1	CLASSD Left Output L1	Output	–	–
CLASSD_L2	CLASSD Left Output L2	Output	–	–
CLASSD_L3	CLASSD Left Output L3	Output	–	–
CLASSD_R0	CLASSD Right Output R0	Output	–	–
CLASSD_R1	CLASSD Right Output R1	Output	–	–
CLASSD_R2	CLASSD Right Output R2	Output	–	–
CLASSD_R3	CLASSD Right Output R3	Output	–	–

Table 4-1. Signal Description List (Continued)

Signal Name	Function	Type	Comments	Active Level
Control Area Network - CAN				
CANRXx	CAN Receive	Input	–	–
CANTXx	CAN Transmit	Output	–	–
Pulse Density Modulation Interface Controller - PDMIC				
PDMIC_DAT	PDM Data	Input	–	–
PDMIC_CLK	PDM Clock	Output	–	–

5. Package and Pinout

5.1 Packages

The SAMA5D2 is available in the packages listed in Table 5-1.

Table 5-1. SAMA5D2 Packages

Package Name	Pin Count	Ball Pitch
LFBGA289	289	0.8 mm
TFBGA256	256	0.4 mm
TFBGA196	196	0.75 mm

The package mechanical characteristics are described in Section 63. “Mechanical Characteristics”.

5.2 Pinouts

Pinouts are provided in Table 5-2 Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A), Table 5-3 Pin Description (SAMA5D23 pins different from those in SAMA5D21/SAMA5D22) and Table 5-4 Pin Description (SAMA5D28B pins different from those in SAMA5D28A).

Note: Devices ATSAMA5D21 and ATSAMA5D22 are pin-to-pin compatible.
Devices ATSAMA5D26 and ATSAMA5D27 are pin-to-pin compatible.
The SAMA5D23 has a different pinout than the SAMA5D21 and SAMA5D22 (SAMA5D22A-CU, SAMA5D22B-CU) (see Table 5-3 Pin Description (SAMA5D23 pins different from those in SAMA5D21/SAMA5D22).
For the SAMA5D28, only the SAMA5D28A-CU is pin-to-pin compatible with SAMA5D27 and SAMA5D26 (SAMA5D27A-CU, SAMA5D27B-CU, SAMA5D26B-CU). The SAMA5D28B-CU pinout is different from that of the SAMA5D28A-CU (see Table 5-4 Pin Description (SAMA5D28B pins different from those in SAMA5D28A).
For further information please contact Atmel Marketing.

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
U11	R10	–	VDDSDMMC	GPIO_EMMC	PA0	I/O	–	–	A	SDMMC0_CK	I/O	1	PIO, I, PU, ST
									B	QSPI0_SCK	O	1	
									F	D0	I/O	2	
P10	R9	–	VDDSDMMC	GPIO_EMMC	PA1	I/O	–	–	A	SDMMC0_CMD	I/O	1	PIO, I, PU, ST
									B	QSPI0_CS	O	1	
									F	D1	I/O	2	
T11	U11	–	VDDSDMMC	GPIO_EMMC	PA2	I/O	–	–	A	SDMMC0_DAT0	I/O	1	PIO, I, PU, ST
									B	QSPI0_IO0	I/O	1	
									F	D2	I/O	2	
R10	P10	–	VDDSDMMC	GPIO_EMMC	PA3	I/O	–	–	A	SDMMC0_DAT1	I/O	1	PIO, I, PU, ST
									B	QSPI0_IO1	I/O	1	
									F	D3	I/O	2	
U12	P11	–	VDDSDMMC	GPIO_EMMC	PA4	I/O	–	–	A	SDMMC0_DAT2	I/O	1	PIO, I, PU, ST
									B	QSPI0_IO2	I/O	1	
									F	D4	I/O	2	
T12	V11	–	VDDSDMMC	GGPIO_EMMC	PA5	I/O	–	–	A	SDMMC0_DAT3	I/O	1	PIO, I, PU, ST
									B	QSPI0_IO3	I/O	1	
									F	D5	I/O	2	
R12	U12	–	VDDSDMMC	GPIO_EMMC	PA6	I/O	–	–	A	SDMMC0_DAT4	I/O	1	PIO, I, PU, ST
									B	QSPI1_SCK	O	1	
									D	TIOA5	I/O	1	
									E	FLEXCOM2_IO0	I/O	1	
									F	D6	I/O	2	
T13	V12	–	VDDSDMMC	GPIO_EMMC	PA7	I/O	–	–	A	SDMMC0_DAT5	I/O	1	PIO, I, PU, ST
									B	QSPI1_IO0	I/O	1	
									D	TIOB5	I/O	1	
									E	FLEXCOM2_IO1	I/O	1	
									F	D7	I/O	2	
N10	N11	–	VDDSDMMC	GPIO_EMMC	PA8	I/O	–	–	A	SDMMC0_DAT6	I/O	1	PIO, I, PU, ST
									B	QSPI1_IO1	I/O	1	
									D	TCLK5	I	1	
									E	FLEXCOM2_IO2	I/O	1	
									F	NWE/NANDWE	O	2	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
N11	P12	–	VDDSDMMC	GPIO_EMMC	PA9	I/O	–	–	A	SDMMC0_DAT7	I/O	1	PIO, I, PU, ST
									B	QSPI1_IO2	I/O	1	
									D	TIOA4	I/O	1	
									E	FLEXCOM2_IO3	O	1	
									F	NCS3	O	2	
U13	U13	–	VDDSDMMC	GPIO_EMMC	PA10	I/O	–	–	A	SDMMC0_RSTN	O	1	PIO, I, PU, ST
									B	QSPI1_IO3	I/O	1	
									D	TIOB4	I/O	1	
									E	FLEXCOM2_IO4	O	1	
									F	A21/NANDALE	O	2	
P15	R14	–	VDDIOP1	GPIO	PA11	I/O	–	–	A	SDMMC0_VDDSEL	O	1	PIO, I, PU, ST
									B	QSPI1_CS	O	1	
									D	TCLK4	I	1	
									F	A22/NANDCLE	O	2	
N15	N13	–	VDDIOP1	GPIO	PA12	I/O	–	–	A	SDMMC0_WP	I	1	PIO, I, PU, ST
									B	IRQ	I	1	
									F	NRD/NANDOE	O	2	
P16	P14	–	VDDIOP1	GPIO	PA13	I/O	–	–	A	SDMMC0_CD	I	1	PIO, I, PU, ST
									E	FLEXCOM3_IO1	I/O	1	
									F	D8	I/O	2	
M14	P17	–	VDDIOP1	GPIO_QSPI	PA14	I/O	–	–	A	SPI0_SPCK	I/O	1	PIO, I, PU, ST
									B	TK1	I/O	1	
									C	QSPI0_SCK	O	2	
									D	I2SC1_MCK	O	2	
									E	FLEXCOM3_IO2	I/O	1	
									F	D9	I/O	2	
N16	R18	–	VDDIOP1	GPIO	PA15	I/O	–	–	A	SPI0_MOSI	I/O	1	PIO, I, PU, ST
									B	TF1	I/O	1	
									C	QSPI0_CS	O	2	
									D	I2SC1_CK	I/O	2	
									E	FLEXCOM3_IO0	I/O	1	
									F	D10	I/O	2	
M10	N15	–	VDDIOP1	GPIO_IO	PA16	I/O	–	–	A	SPI0_MISO	I/O	1	PIO, I, PU, ST
									B	TD1	O	1	
									C	QSPI0_IO0	I/O	2	
									D	I2SC1_WS	I/O	2	
									E	FLEXCOM3_IO3	O	1	
									F	D11	I/O	2	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
N17	P18	–	VDDIOP1	GPIO_IO	PA17	I/O	–	–	A	SPI0_NPCS0	I/O	1	PIO, I, PU, ST
									B	RD1	I	1	
									C	QSPI0_IO1	I/O	2	
									D	I2SC1_DI0	I	2	
									E	FLEXCOM3_IO4	O	1	
									F	D12	I/O	2	
U14	M9	L9	VDDIOP1	GPIO_IO	PA18	I/O	–	–	A	SPI0_NPCS1	O	1	PIO, I, PU, ST
									B	RK1	I/O	1	
									C	QSPI0_IO2	I/O	2	
									D	I2SC1_DO0	O	2	
									E	SDMMC1_DAT0	I/O	1	
									F	D13	I/O	2	
T14	V13	N9	VDDIOP1	GPIO_IO	PA19	I/O	–	–	A	SPI0_NPCS2	O	1	PIO, I, PU, ST
									B	RF1	I/O	1	
									C	QSPI0_IO3	I/O	2	
									D	TIOA0	I/O	1	
									E	SDMMC1_DAT1	I/O	1	
									F	D14	I/O	2	
P12	L9	M9	VDDIOP1	GPIO_IO	PA20	I/O	–	–	A	SPI0_NPCS3	O	1	PIO, I, PU, ST
									D	TIOB0	I/O	1	
									E	SDMMC1_DAT2	I/O	1	
									F	D15	I/O	2	
R13	M10	M10	VDDIOP1	GPIO_IO	PA21	I/O	–	–	A	IRQ	I	2	PIO, I, PU, ST
									B	PCK2	O	3	
									D	TCLK0	I	1	
									E	SDMMC1_DAT3	I/O	1	
									F	NANDRDY	I	2	
U15	V14	P9	VDDIOP1	GPIO_QSPI	PA22	I/O	–	–	A	FLEXCOM1_IO2	I/O	1	PIO, I, PU, ST
									B	D0	I/O	1	
									C	TCK	I	4	
									D	SPI1_SPCK	I/O	2	
									E	SDMMC1_CK	I/O	1	
									F	QSPI0_SCK	O	3	
U16	U14	P10	VDDIOP1	GPIO	PA23	I/O	–	–	A	FLEXCOM1_IO1	I/O	1	PIO, I, PU, ST
									B	D1	I/O	1	
									C	TDI	I	4	
									D	SPI1_MOSI	I/O	2	
									F	QSPI0_CS	O	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
T15	R13	N10	VDDIOP1	GPIO_IO	PA24	I/O	–	–	A	FLEXCOM1_IO0	I/O	1	PIO, I, PU, ST
									B	D2	I/O	1	
									C	TDO	O	4	
									D	SPI1_MISO	I/O	2	
									F	QSPI0_IO0	I/O	3	
U17	U15	L10	VDDIOP1	GPIO_IO	PA25	I/O	–	–	A	FLEXCOM1_IO3	O	1	PIO, I, PU, ST
									B	D3	I/O	1	
									C	TMS	I	4	
									D	SPI1_NPCS0	I/O	2	
									F	QSPI0_IO1	I/O	3	
P13	L10	P11	VDDIOP1	GPIO_IO	PA26	I/O	–	–	A	FLEXCOM1_IO4	O	1	PIO, I, PU, ST
									B	D4	I/O	1	
									C	NTRST	I	4	
									D	SPI1_NPCS1	O	2	
									F	QSPI0_IO2	I/O	3	
T16	V17	P12	VDDIOP1	GPIO_IO	PA27	I/O	–	–	A	TIOA1	I/O	2	PIO, I, PU, ST
									B	D5	I/O	1	
									C	SPI0_NPCS2	O	2	
									D	SPI1_NPCS2	O	2	
									E	SDMMC1_RSTN	O	1	
									F	QSPI0_IO3	I/O	3	
R16	U16	M11	VDDIOP1	GPIO	PA28	I/O	–	–	A	TIOB1	I/O	2	PIO, I, PU, ST
									B	D6	I/O	1	
									C	SPI0_NPCS3	O	2	
									D	SPI1_NPCS3	O	2	
									E	SDMMC1_CMD	I/O	1	
									F	CLASSD_L0	O	1	
T17	U17	N11	VDDIOP1	GPIO	PA29	I/O	–	–	A	TCLK1	I	2	PIO, I, PU, ST
									B	D7	I/O	1	
									C	SPI0_NPCS1	O	2	
									E	SDMMC1_WP	I	1	
									F	CLASSD_L1	O	1	
R15	V18	N12	VDDIOP1	GPIO	PA30	I/O	–	–	B	NWE/NANDWE	O	1	PIO, I, PU, ST
									C	SPI0_NPCS0	I/O	2	
									D	PWMH0	O	1	
									E	SDMMC1_CD	I	1	
									F	CLASSD_L2	O	1	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
R17	U18	M12	VDDIOP1	GPIO	PA31	I/O	–	–	B	NCS3	O	1	PIO, I, PU, ST
									C	SPI0_MISO	I/O	2	
									D	PWML0	O	1	
									F	CLASSD_L3	O	1	
J8	G9	A6	VDDIOP0	GPIO	PB0	I/O	–	–	B	A21/NANDALE	O	1	PIO, I, PU, ST
									C	SPI0_MOSI	I/O	2	
									D	PWMH1	O	1	
A8	A7	A5	VDDIOP0	GPIO	PB1	I/O	–	–	B	A22/NANDCLE	O	1	PIO, I, PU, ST
									C	SPI0_SPCK	I/O	2	
									D	PWML1	O	1	
									F	CLASSD_R0	O	1	
A7	B7	B6	VDDIOP0	GPIO	PB2	I/O	–	–	B	NRD/NANDOE	O	1	PIO, I, PU, ST
									D	PWMFI0	I	1	
									F	CLASSD_R1	O	1	
A6	B6	B5	VDDIOP0	GPIO	PB3	I/O	–	–	A	URXD4	I	1	PIO, I, PU, ST
									B	D8	I/O	1	
									C	IRQ	I	3	
									D	PWMEXTRG1	I	1	
									F	CLASSD_R2	O	1	
B6	A6	A4	VDDIOP0	GPIO	PB4	I/O	–	–	A	UTXD4	O	1	PIO, I, PU, ST
									B	D9	I/O	1	
									C	FIQ	I	4	
									F	CLASSD_R3	O	1	
B7	D7	D6	VDDIOP0	GPIO_QSPI	PB5	I/O	–	–	A	TCLK2	I	1	PIO, I, PU, ST
									B	D10	I/O	1	
									C	PWMH2	O	1	
									D	QSPI1_SCK	O	2	
									F	GTSUCOMP	O	3	
C7	B5	A3	VDDIOP0	GPIO	PB6	I/O	–	–	A	TIOA2	I/O	1	PIO, I, PU, ST
									B	D11	I/O	1	
									C	PWML2	O	1	
									D	QSPI1_CS	O	2	
									F	GTXER	O	3	
C6	A5	B4	VDDIOP0	GPIO_IO	PB7	I/O	–	–	A	TIOB2	I/O	1	PIO, I, PU, ST
									B	D12	I/O	1	
									C	PWMH3	O	1	
									D	QSPI1_IO0	I/O	2	
									F	GRXCK	I	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
A5	E7	A2	VDDIOP0	GPIO_IO	PB8	I/O	–	–	A	TCLK3	I	1	PIO, I, PU, ST
									B	D13	I/O	1	
									C	PWML3	O	1	
									D	QSPI1_IO1	I/O	2	
									F	GCRS	I	3	
A4	F6	B3	VDDIOP0	GPIO_IO	PB9	I/O	–	–	A	TIOA3	I/O	1	PIO, I, PU, ST
									B	D14	I/O	1	
									C	PWMFI1	I	1	
									D	QSPI1_IO2	I/O	2	
									F	GCOL	I	3	
H8	D6	A1	VDDIOP0	GPIO_IO	PB10	I/O	–	–	A	TIOB3	I/O	1	PIO, I, PU, ST
									B	D15	I/O	1	
									C	PWMEXTRG2	I	1	
									D	QSPI1_IO3	I/O	2	
									F	GRX2	I	3	
B5	A4	B1	VDDIOP0	GPIO	PB11	I/O	–	–	A	LCDDAT0	O	1	PIO, I, PU, ST
									B	A0/NBS0	O	1	
									C	URXD3	I	3	
									D	PDMIC_DAT		2	
									F	GRX3	I	3	
D6	B3	B2	VDDIOP0	GPIO	PB12	I/O	–	–	A	LCDDAT1	O	1	PIO, I, PU, ST
									B	A1	O	1	
									C	UTXD3	O	3	
									D	PDMIC_CLK		2	
									F	GTX2	O	3	
B4	A3	C1	VDDIOP0	GPIO	PB13	I/O	–	–	A	LCDDAT2	O	1	PIO, I, PU, ST
									B	A2	O	1	
									C	PCK1	O	3	
									F	GTX3	O	3	
C5	B4	D5	VDDIOP0	GPIO_QSPI	PB14	I/O	–	–	A	LCDDAT3	O	1	PIO, I, PU, ST
									B	A3	O	1	
									C	TK1	I/O	2	
									D	I2SC1_MCK	O	1	
									E	QSPI1_SCK	O	3	
									F	GTXCK	I/O	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
H7	G8	E5	VDDIOP0	GPIO	PB15	I/O	–	–	A	LCDDAT4	O	1	PIO, I, PU, ST
									B	A4	O	1	
									C	TF1	I/O	2	
									D	I2SC1_CK	I/O	1	
									E	QSPI1_CS	O	3	
									F	GTXEN	O	3	
D5	E5	C5	VDDIOP0	GPIO_IO	PB16	I/O	–	–	A	LCDDAT5	O	1	PIO, I, PU, ST
									B	A5	O	1	
									C	TD1	O	2	
									D	I2SC1_WS	I/O	1	
									E	QSPI1_IO0	I/O	3	
									F	GRXDV	I	3	
C4	G7	C2	VDDIOP0	GPIO_IO	PB17	I/O	–	–	A	LCDDAT6	O	1	PIO, I, PU, ST
									B	A6	O	1	
									C	RD1	I	2	
									D	I2SC1_DI0	I	1	
									E	QSPI1_IO1	I/O	3	
									F	GRXER	I	3	
A3	A2	D4	VDDIOP0	GPIO_IO	PB18	I/O	–	–	A	LCDDAT7	O	1	PIO, I, PU, ST
									B	A7	O	1	
									C	RK1	I/O	2	
									D	I2SC1_DO0	O	1	
									E	QSPI1_IO2	I/O	3	
									F	GRX0	I	3	
D4	H7	C4	VDDIOP0	GPIO_IO	PB19	I/O	–	–	A	LCDDAT8	O	1	PIO, I, PU, ST
									B	A8	O	1	
									C	RF1	I/O	2	
									D	TIOA3	I/O	2	
									E	QSPI1_IO3	I/O	3	
									F	GRX1	I	3	
B3	A1	C3	VDDIOP0	GPIO	PB20	I/O	–	–	A	LCDDAT9	O	1	PIO, I, PU, ST
									B	A9	O	1	
									C	TK0	I/O	1	
									D	TIOB3	I/O	2	
									E	PCK1	O	4	
									F	GTX0	O	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
A2	D2	D1	VDDIOP0	GPIO	PB21	I/O	–	–	A	LCDDAT10	O	1	PIO, I, PU, ST
									B	A10	O	1	
									C	TF0	I/O	1	
									D	TCLK3	I	2	
									E	FLEXCOM3_IO2	I/O	3	
									F	GTX1	O	3	
C3	G5	D2	VDDIOP0	GPIO	PB22	I/O	–	–	A	LCDDAT11	O	1	PIO, I, PU, ST
									B	A11	O	1	
									C	TD0	O	1	
									D	TIOA2	I/O	2	
									E	FLEXCOM3_IO1	I/O	3	
									F	GMDC	O	3	
A1	C2	E1	VDDIOP0	GPIO	PB23	I/O	–	–	A	LCDDAT12	O	1	PIO, I, PU, ST
									B	A12	O	1	
									C	RD0	I	1	
									D	TIOB2	I/O	2	
									E	FLEXCOM3_IO0	I/O	3	
									F	GMDIO	I/O	3	
E5	F4	D3	VDDIOP0	GPIO	PB24	I/O	–	–	A	LCDDAT13	O	1	PIO, I, PU, ST
									B	A13	O	1	
									C	RK0	I/O	1	
									D	TCLK2	I	2	
									E	FLEXCOM3_IO3	O	3	
									F	ISC_D10	I	3	
B2	C1	E3	VDDIOP0	GPIO	PB25	I/O	–	–	A	LCDDAT14	O	1	PIO, I, PU, ST
									B	A14	O	1	
									C	RF0	I/O	1	
									E	FLEXCOM3_IO4	O	3	
									F	ISC_D11	I	3	
E4	E4	E2	VDDIOP0	GPIO	PB26	I/O	–	–	A	LCDDAT15	O	1	PIO, I, PU, ST
									B	A15	O	1	
									C	URXD0	I	1	
									D	PDMIC_DAT		1	
									F	ISC_D0	I	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
B1	F1	E6	VDDIOP0	GPIO	PB27	I/O	–	–	A	LCDDAT16	O	1	PIO, I, PU, ST
									B	A16	O	1	
									C	UTXD0	O	1	
									D	PDMIC_CLK		1	
									F	ISC_D1	I	3	
C2	D1	F1	VDDIOP0	GPIO	PB28	I/O	–	–	A	LCDDAT17	O	1	PIO, I, PU, ST
									B	A17	O	1	
									C	FLEXCOM0_IO0	I/O	1	
									D	TIOA5	I/O	2	
									F	ISC_D2	I	3	
D3	F2	F6	VDDIOP0	GPIO	PB29	I/O	–	–	A	LCDDAT18	O	1	PIO, I, PU, ST
									B	A18	O	1	
									C	FLEXCOM0_IO1	I/O	1	
									D	TIOB5	I/O	2	
									F	ISC_D3	I	3	
D2	E2	F2	VDDIOP0	GPIO	PB30	I/O	–	–	A	LCDDAT19	O	1	PIO, I, PU, ST
									B	A19	O	1	
									C	FLEXCOM0_IO2	I/O	1	
									D	TCLK5	I	2	
									F	ISC_D4	I	3	
C1	E1	F7	VDDIOP0	GPIO	PB31	I/O	–	–	A	LCDDAT20	O	1	PIO, I, PU, ST
									B	A20	O	1	
									C	FLEXCOM0_IO3	O	1	
									D	TWD0	I/O	1	
									F	ISC_D5	I	3	
P17	R15	M13	VDDIOP1	GPIO	PC0	I/O	–	–	A	LCDDAT21	O	1	PIO, I, PU, ST
									B	A23	O	1	
									C	FLEXCOM0_IO4	O	1	
									D	TWCK0	I/O	1	
									F	ISC_D6	I	3	
N12	M11	P13	VDDIOP1	GPIO	PC1	I/O	–	–	A	LCDDAT22	O	1	PIO, I, PU, ST
									B	A24	O	1	
									C	CANTX0	O	1	
									D	SPI1_SPCK	I/O	1	
									E	I2SC0_CK	I/O	1	
									F	ISC_D7	I	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
N14	P15	N13	VDDIOP1	GPIO	PC2	I/O	–	–	A	LCDDAT23	O	1	PIO, I, PU, ST
									B	A25	O	1	
									C	CANRX0	I	1	
									D	SPI1_MOSI	I/O	1	
									E	I2SC0_MCK	O	1	
									F	ISC_D8	I	3	
M15	K9	K10	VDDIOP1	GPIO	PC3	I/O	–	–	A	LCDPWM	O	1	PIO, I, PU, ST
									B	NWAIT	I	1	
									C	TIOA1	I/O	1	
									D	SPI1_MISO	I/O	1	
									E	I2SC0_WS	I/O	1	
									F	ISC_D9	I	3	
M11	K10	P14	VDDIOP1	GPIO	PC4	I/O	–	–	A	LCDDISP	O	1	PIO, I, PU, ST
									B	NWR1/NBS1	O	1	
									C	TIOB1	I/O	1	
									D	SPI1_NPCS0	I/O	1	
									E	I2SC0_DI0	I	1	
									F	ISC_PCK	I	3	
L10	L11	J8	VDDIOP1	GPIO	PC5	I/O	–	–	A	LCDVSYNC	O	1	PIO, I, PU, ST
									B	NCS0	O	1	
									C	TCLK1	I	1	
									D	SPI1_NPCS1	O	1	
									E	I2SC0_DO0	O	1	
									F	ISC_VSYNC	I	3	
K10	L12	N14	VDDIOP1	GPIO	PC6	I/O	–	–	A	LCDHSYNC	O	1	PIO, I, PU, ST
									B	NCS1	O	1	
									C	TWD1	I/O	1	
									D	SPI1_NPCS2	O	1	
									F	ISC_HSYNC	I	3	
M16	M12	M14	VDDIOP1	GPIO_CLK	PC7	I/O	–	–	A	LCDPCK	O	1	PIO, I, PU, ST
									B	NCS2	O	1	
									C	TWCK1	I/O	1	
									D	SPI1_NPCS3	O	1	
									E	URXD1	I	2	
									F	ISC_MCK	O	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
J10	K11	J9	VDDIOP1	GPIO	PC8	I/O	–	–	A	LCDDEN	O	1	PIO, I, PU, ST
									B	NANDRDY	I	1	
									C	FIQ	I	1	
									D	PCK0	O	3	
									E	UTXD1	O	2	
									F	ISC_FIELD	I	3	
D1	–	–	VDDISC	GPIO	PC9	I/O	–	–	A	FIQ	I	3	PIO, I, PU, ST
									B	GTSUCOMP	O	1	
									C	ISC_D0	I	1	
									D	TIOA4	I/O	2	
E3	–	–	VDDISC	GPIO	PC10	I/O	–	–	A	LCDDAT2	O	2	PIO, I, PU, ST
									B	GTXCK	I/O	1	
									C	ISC_D1	I	1	
									D	TIOB4	I/O	2	
									E	CANTX0	O	2	
E2	–	–	VDDISC	GPIO	PC11	I/O	–	–	A	LCDDAT3	O	2	PIO, I, PU, ST
									B	GTXEN	O	1	
									C	ISC_D2	I	1	
									D	TCLK4	I	2	
									E	CANRX0	I	2	
									F	A0/NBS0	O	2	
E1	–	–	VDDISC	GPIO	PC12	I/O	–	–	A	LCDDAT4	O	2	PIO, I, PU, ST
									B	GRXDV	I	1	
									C	ISC_D3	I	1	
									D	URXD3	I	1	
									E	TK0	I/O	2	
									F	A1	O	2	
F3	–	–	VDDISC	GPIO	PC13	I/O	–	–	A	LCDDAT5	O	2	PIO, I, PU, ST
									B	GRXER	I	1	
									C	ISC_D4	I	1	
									D	UTXD3	O	1	
									E	TF0	I/O	2	
									F	A2	O	2	
F5	–	–	VDDISC	GPIO	PC14	I/O	–	–	A	LCDDAT6	O	2	PIO, I, PU, ST
									B	GRX0	I	1	
									C	ISC_D5	I	1	
									E	TD0	O	2	
									F	A3	O	2	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
F2	–	–	VDDISC	GPIO	PC15	I/O	–	–	A	LCDDAT7	O	2	PIO, I, PU, ST
									B	GRX1	I	1	
									C	ISC_D6	I	1	
									E	RD0	I	2	
									F	A4	O	2	
G6	–	–	VDDISC	GPIO	PC16	I/O	–	–	A	LCDDAT10	O	2	PIO, I, PU, ST
									B	GTX0	O	1	
									C	ISC_D7	I	1	
									E	RK0	I/O	2	
									F	A5	O	2	
F1	–	–	VDDISC	GPIO	PC17	I/O	–	–	A	LCDDAT11	O	2	PIO, I, PU, ST
									B	GTX1	O	1	
									C	ISC_D8	I	1	
									E	RF0	I/O	2	
									F	A6	O	2	
H6	–	–	VDDISC	GPIO	PC18	I/O	–	–	A	LCDDAT12	O	2	PIO, I, PU, ST
									B	GMDC	O	1	
									C	ISC_D9	I	1	
									E	FLEXCOM3_IO2	I/O	2	
									F	A7	O	2	
G2	–	–	VDDISC	GPIO	PC19	I/O	–	–	A	LCDDAT13	O	2	PIO, I, PU, ST
									B	GMDIO	I/O	1	
									C	ISC_D10	I	1	
									E	FLEXCOM3_IO1	I/O	2	
									F	A8	O	2	
G3	–	–	VDDISC	GPIO	PC20	I/O	–	–	A	LCDDAT14	O	2	PIO, I, PU, ST
									B	GRXCK	I	1	
									C	ISC_D11	I	1	
									E	FLEXCOM3_IO0	I/O	2	
									F	A9	O	2	
G1	–	–	VDDISC	GPIO	PC21	I/O	–	–	A	LCDDAT15	O	2	PIO, I, PU, ST
									B	GTXER	O	1	
									C	ISC_PCK	I	1	
									E	FLEXCOM3_IO3	O	2	
									F	A10	O	2	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
H2	–	–	VDDISC	GPIO	PC22	I/O	–	–	A	LCDDAT18	O	2	PIO, I, PU, ST
									B	GCRS	I	1	
									C	ISC_VSYNC	I	1	
									E	FLEXCOM3_IO4	O	2	
									F	A11	O	2	
G5	–	–	VDDISC	GPIO	PC23	I/O	–	–	A	LCDDAT19	O	2	PIO, I, PU, ST
									B	GCOL	I	1	
									C	ISC_HSYNC	I	1	
									F	A12	O	2	
H1	–	–	VDDISC	GPIO_CLK	PC24	I/O	–	–	A	LCDDAT20	O	2	PIO, I, PU, ST
									B	GRX2	I	1	
									C	ISC_MCK	O	1	
									F	A13	O	2	
H5	–	–	VDDISC	GPIO	PC25	I/O	–	–	A	LCDDAT21	O	2	PIO, I, PU, ST
									B	GRX3	I	1	
									C	ISC_FIELD	I	1	
									F	A14	O	2	
J9	–	–	VDDIOP2	GPIO	PC26	I/O	–	–	A	LCDDAT22	O	2	PIO, I, PU, ST
									B	GTX2	O	1	
									D	CANTX1	O	1	
									F	A15	O	2	
H9	–	–	VDDIOP2	GPIO	PC27	I/O	–	–	A	LCDDAT23	O	2	PIO, I, PU, ST
									B	GTX3	O	1	
									C	PCK1	O	2	
									D	CANRX1	I	1	
									E	TWD0	I/O	2	
									F	A16	O	2	
E8	–	–	VDDIOP2	GPIO	PC28	I/O	–	–	A	LCDPWM	O	2	PIO, I, PU, ST
									B	FLEXCOM4_IO0	I/O	1	
									C	PCK2	O	1	
									E	TWCK0	I/O	2	
									F	A17	O	2	
G8	–	–	VDDIOP2	GPIO	PC29	I/O	–	–	A	LCDDISP	O	2	PIO, I, PU, ST
									B	FLEXCOM4_IO1	I/O	1	
									F	A18	O	2	
F8	–	–	VDDIOP2	GPIO	PC30	I/O	–	–	A	LCDVSYNC	O	2	PIO, I, PU, ST
									B	FLEXCOM4_IO2	I/O	1	
									F	A19	O	2	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
D8	–	–	VDDIOP2	GPIO	PC31	I/O	–	–	A	LCDHSYNC	O	2	PIO, I, PU, ST
									B	FLEXCOM4_IO3	O	1	
									C	URXD3	I	2	
									F	A20	O	2	
G10	E9	–	VDDIOP2	GPIO_CLK	PD0	I/O	–	–	A	LCDPCK	O	2	PIO, I, PU, ST
									B	FLEXCOM4_IO4	O	1	
									C	UTXD3	O	2	
									D	GTSUCOMP	O	2	
									F	A23	O	2	
E10	F8	–	VDDIOP2	GPIO	PD1	I/O	–	–	A	LCDDEN	O	2	PIO, I, PU, ST
									D	GRXCK	I	2	
									F	A24	O	2	
G9	F9	–	VDDIOP2	GPIO_CLK	PD2	I/O	–	–	A	URXD1	I	1	PIO, I, PU, ST
									D	GTXER	O	2	
									E	ISC_MCK	O	2	
									F	A25	O	2	
K1	J4	–	VDDANA	GPIO_AD	PD3	I/O	–	–	A	UTXD1	O	1	PIO, I, PU, ST
									B	FIQ	I	2	
									D	GCRS	I	2	
									E	ISC_D11	I	2	
									F	NWAIT	I	2	
J6	H6	–	VDDANA	GPIO_AD	PD4	I/O	–	–	A	TWD1	I/O	2	PIO, I, PU, ST
									B	URXD2	I	1	
									D	GCOL	I	2	
									E	ISC_D10	I	2	
									F	NCS0	O	2	
J4	H1	–	VDDANA	GPIO_AD	PD5	I/O	–	–	A	TWCK1	I/O	2	PIO, I, PU, ST
									B	UTXD2	O	1	
									D	GRX2	I	2	
									E	ISC_D9	I	2	
									F	NCS1	O	2	
J2	G4	–	VDDANA	GPIO_AD	PD6	I/O	–	–	A	TCK	I	2	PIO, I, PU, ST
									B	PCK1	O	1	
									D	GRX3	I	2	
									E	ISC_D8	I	2	
									F	NCS2	O	2	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
J7	H5	F5	VDDANA	GPIO_AD	PD7	I/O	–	–	A	TDI	I	2	PIO, I, PU, ST
									C	UTMI_RXVAL	O	1	
									D	GTX2	O	2	
									E	ISC_D0	I	2	
									F	NWR1/NBS1	O	2	
J1	G1	F3	VDDANA	GPIO_AD	PD8	I/O	–	–	A	TDO	O	2	PIO, I, PU, ST
									C	UTMI_RXERR	O	1	
									D	GTX3	O	2	
									E	ISC_D1	I	2	
									F	NANDRDY	I	2	
K9	H4	G5	VDDANA	GPIO_AD	PD9	I/O	–	–	A	TMS	I	2	PIO, I, PU, ST
									C	UTMI_RXACT	O	1	
									D	GTXCK	I/O	2	
									E	ISC_D2	I	2	
J3	G2	G4	VDDANA	GPIO_AD	PD10	I/O	–	–	A	NTRST	I	2	PIO, I, PU, ST
									C	UTMI_HDIS	O	1	
									D	GTXEN	O	2	
									E	ISC_D3	I	2	
M1	H2	H1	VDDANA	GPIO_AD	PD11	I/O	–	–	A	TIOA1	I/O	3	PIO, I, PU, ST
									B	PCK2	O	2	
									C	UTMI_LS0	O	1	
									D	GRXDV	I	2	
									E	ISC_D4	I	2	
									F	ISC_MCK	O	4	
K8	K5	H6	VDDANA	GPIO_AD	PD12	I/O	–	–	A	TIOB1	I/O	3	PIO, I, PU, ST
									B	FLEXCOM4_IO0	I/O	2	
									C	UTMI_LS1	O	1	
									D	GRXER	I	2	
									E	ISC_D5	I	2	
									F	ISC_D4	I	4	
L2	J5	H3	VDDANA	GPIO_AD	PD13	I/O	–	–	A	TCLK1	I	3	PIO, I, PU, ST
									B	FLEXCOM4_IO1	I/O	2	
									C	UTMI_CDRCPSEL0	I	1	
									D	GRX0	I	2	
									E	ISC_D6	I	2	
									F	ISC_D5	I	4	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
K4	K6	G6	VDDANA	GPIO_AD	PD14	I/O	–	–	A	TCK	I	1	A, PU, ST
									B	FLEXCOM4_IO2	I/O	2	
									C	UTMI_CDRCPSEL1	I	1	
									D	GRX1	I	2	
									E	ISC_D7	I	2	
									F	ISC_D6	I	4	
K7	K4	H5	VDDANA	GPIO_AD	PD15	I/O	–	–	A	TDI	I	1	PIO, I, PU, ST
									B	FLEXCOM4_IO3	O	2	
									C	UTMI_CDRCPDIVEN	I	1	
									D	GTX0	O	2	
									E	ISC_PCK	I	2	
									F	ISC_D7	I	4	
L1	K1	G1	VDDANA	GPIO_AD	PD16	I/O	–	–	A	TDO	O	1	PIO, I, PU, ST
									B	FLEXCOM4_IO4	O	2	
									C	UTMI_CDRBISTEN	I	1	
									D	GTX1	O	2	
									E	ISC_VSYNC	I	2	
									F	ISC_D8	I	4	
K2	K2	G2	VDDANA	GPIO_AD	PD17	I/O	–	–	A	TMS	I	1	A, PU, ST
									C	UTMI_CDRCPSELDIV	O	1	
									D	GMDC	O	2	
									E	ISC_HSYNC	I	2	
									F	ISC_D9	I	4	
J5	L5	G3	VDDANA	GPIO_AD	PD18	I/O	–	–	A	NTRST	I	1	PIO, I, PU, ST
									D	GMDIO	I/O	2	
									E	ISC_FIELD	I	2	
									F	ISC_D10	I	4	
K6	L4	H4	VDDANA	GPIO_AD	PD19	I/O	AD0	–	A	PCK0	O	1	PIO, I, PU, ST
									B	TWD1	I/O	3	
									C	URXD2	I	3	
									E	I2SC0_CK	I/O	2	
									F	ISC_D11	I	4	
M2	M1	J1	VDDANA	GPIO_AD	PD20	I/O	AD1	–	A	TIOA2	I/O	3	PIO, I, PU, ST
									B	TWCK1	I/O	3	
									C	UTXD2	O	3	
									E	I2SC0_MCK	O	2	
									F	ISC_PCK	I	4	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
N1	M2	K1	VDDANA	GPIO_AD	PD21	I/O	AD2	–	A	TIOB2	I/O	3	PIO, I, PU, ST
									B	TWD0	I/O	4	
									C	FLEXCOM4_IO0	I/O	3	
									E	I2SC0_WS	I/O	2	
									F	ISC_VSYNC	I	4	
L4	M4	J3	VDDANA	GPIO_AD	PD22	I/O	AD3	–	A	TCLK2	I	3	PIO, I, PU, ST
									B	TWCK0	I/O	4	
									C	FLEXCOM4_IO1	I/O	3	
									E	I2SC0_DI0	I	2	
									F	ISC_HSYNC	I	4	
M3	P1	K2	VDDANA	GPIO_AD	PD23	I/O	AD4	–	A	URXD2	I	2	PIO, I, PU, ST
									C	FLEXCOM4_IO2	I/O	3	
									E	I2SC0_DO0	O	2	
									F	ISC_FIELD	I	4	
L7	L6	–	VDDANA	GPIO_AD	PD24	I/O	AD5	–	A	UTXD2	O	2	PIO, I, PU, ST
									C	FLEXCOM4_IO3	O	3	
L6	M5	–	VDDANA	GPIO_AD	PD25	I/O	AD6	–	A	SPI1_SPCK	I/O	3	PIO, I, PU, ST
									C	FLEXCOM4_IO4	O	3	
N2	N1	–	VDDANA	GPIO_AD	PD26	I/O	AD7	–	A	SPI1_MOSI	I/O	3	PIO, I, PU, ST
									C	FLEXCOM2_IO0	I/O	2	
L8	N2	–	VDDANA	GPIO_AD	PD27	I/O	AD8	–	A	SPI1_MISO	I/O	3	PIO, I, PU, ST
									B	TCK	I	3	
									C	FLEXCOM2_IO1	I/O	2	
M4	P2	–	VDDANA	GPIO_AD	PD28	I/O	AD9	–	A	SPI1_NPCS0	I/O	3	PIO, I, PU, ST
									B	TDI	I	3	
									C	FLEXCOM2_IO2	I/O	2	
N3	R1	–	VDDANA	GPIO_AD	PD29	I/O	AD10	–	A	SPI1_NPCS1	O	3	PIO, I, PU, ST
									B	TDO	O	3	
									C	FLEXCOM2_IO3	O	2	
									D	TIOA3	I/O	3	
									E	TWD0	I/O	3	
L9	N4	–	VDDANA	GPIO_AD	PD30	I/O	AD11	–	A	SPI1_NPCS2	O	3	PIO, I, PU, ST
									B	TMS	I	3	
									C	FLEXCOM2_IO4	O	2	
									D	TIOB3	I/O	3	
									E	TWCK0	I/O	3	

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
M7	T1	–	VDDANA	GPIO	PD31	I/O	–	–	A	ADTRG	I	1	PIO, I, PU, ST
									B	NTRST	I	3	
									C	IRQ	I	4	
									D	TCLK3	I	3	
									E	PCK0	O	2	
L5	L1	K3	VDDANA	power	VDDANA	I	–	–	–	–	–	–	–
K5	L2	K4	GNDANA	ground	GNDANA	I	–	–	–	–	–	–	–
M6	P5	L2	VDDANA	–	ADVREF	I	–	–	–	–	–	–	–
K3	J1	H2	VDDANA	power	VDDANA	I	–	–	–	–	–	–	–
L3	J2	J2	GNDANA	ground	GNDANA	I	–	–	–	–	–	–	–
H16, D16	J17, D12	H12, C12	VDDIODDR	DDR	DDR_VREF	–	–	–	–	–	–	–	–
B12	B12	B7	VDDIODDR	DDR	DDR_D0	–	–	–	–	–	–	–	–
A12	B13	A7	VDDIODDR	DDR	DDR_D1	–	–	–	–	–	–	–	–
C12	D13	C8	VDDIODDR	DDR	DDR_D2	–	–	–	–	–	–	–	–
A13	A13	B9	VDDIODDR	DDR	DDR_D3	–	–	–	–	–	–	–	–
A14	A15	A9	VDDIODDR	DDR	DDR_D4	–	–	–	–	–	–	–	–
C13	D14	C9	VDDIODDR	DDR	DDR_D5	–	–	–	–	–	–	–	–
A15	B15	A10	VDDIODDR	DDR	DDR_D6	–	–	–	–	–	–	–	–
B15	B16	B10	VDDIODDR	DDR	DDR_D7	–	–	–	–	–	–	–	–
G17	G18	H13	VDDIODDR	DDR	DDR_D8	–	–	–	–	–	–	–	–
G16	K17	H14	VDDIODDR	DDR	DDR_D9	–	–	–	–	–	–	–	–
H17	J13	J13	VDDIODDR	DDR	DDR_D10	–	–	–	–	–	–	–	–
K17	H15	J14	VDDIODDR	DDR	DDR_D11	–	–	–	–	–	–	–	–
K16	J15	L13	VDDIODDR	DDR	DDR_D12	–	–	–	–	–	–	–	–
J13	J14	L14	VDDIODDR	DDR	DDR_D13	–	–	–	–	–	–	–	–
K14	K13	J12	VDDIODDR	DDR	DDR_D14	–	–	–	–	–	–	–	–
K15	K18	K12	VDDIODDR	DDR	DDR_D15	–	–	–	–	–	–	–	–
B8	A8	–	VDDIODDR	DDR	DDR_D16	–	–	–	–	–	–	–	–
B9	B9	–	VDDIODDR	DDR	DDR_D17	–	–	–	–	–	–	–	–
C9	D9	–	VDDIODDR	DDR	DDR_D18	–	–	–	–	–	–	–	–
A9	A9	–	VDDIODDR	DDR	DDR_D19	–	–	–	–	–	–	–	–
A10	B11	–	VDDIODDR	DDR	DDR_D20	–	–	–	–	–	–	–	–
D10	D10	–	VDDIODDR	DDR	DDR_D21	–	–	–	–	–	–	–	–
B11	A11	–	VDDIODDR	DDR	DDR_D22	–	–	–	–	–	–	–	–
A11	A12	–	VDDIODDR	DDR	DDR_D23	–	–	–	–	–	–	–	–
J12	L18	–	VDDIODDR	DDR	DDR_D24	–	–	–	–	–	–	–	–
H10	K15	–	VDDIODDR	DDR	DDR_D25	–	–	–	–	–	–	–	–

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
J11	K14	–	VDDIODDR	DDR	DDR_D26	–	–	–	–	–	–	–	–
K11	M18	–	VDDIODDR	DDR	DDR_D27	–	–	–	–	–	–	–	–
L13	N17	–	VDDIODDR	DDR	DDR_D28	–	–	–	–	–	–	–	–
L11	M14	–	VDDIODDR	DDR	DDR_D29	–	–	–	–	–	–	–	–
L12	M15	–	VDDIODDR	DDR	DDR_D30	–	–	–	–	–	–	–	–
M17	N18	–	VDDIODDR	DDR	DDR_D31	–	–	–	–	–	–	–	–
F12	D17	E11	VDDIODDR	DDR	DDR_A0	–	–	–	–	–	–	–	–
C17	A17	C11	VDDIODDR	DDR	DDR_A1	–	–	–	–	–	–	–	–
B17	A18	B12	VDDIODDR	DDR	DDR_A2	–	–	–	–	–	–	–	–
B16	F15	A12	VDDIODDR	DDR	DDR_A3	–	–	–	–	–	–	–	–
C16	G12	D11	VDDIODDR	DDR	DDR_A4	–	–	–	–	–	–	–	–
G14	H12	D14	VDDIODDR	DDR	DDR_A5	–	–	–	–	–	–	–	–
F14	F13	B14	VDDIODDR	DDR	DDR_A6	–	–	–	–	–	–	–	–
F11	H10	D9	VDDIODDR	DDR	DDR_A7	–	–	–	–	–	–	–	–
C14	A16	C10	VDDIODDR	DDR	DDR_A8	–	–	–	–	–	–	–	–
D13	E12	D10	VDDIODDR	DDR	DDR_A9	–	–	–	–	–	–	–	–
C15	H11	F9	VDDIODDR	DDR	DDR_A10	–	–	–	–	–	–	–	–
A16	J10	A11	VDDIODDR	DDR	DDR_A11	–	–	–	–	–	–	–	–
A17	D15	B11	VDDIODDR	DDR	DDR_A12	–	–	–	–	–	–	–	–
G11	J11	E13	VDDIODDR	DDR	DDR_A13	–	–	–	–	–	–	–	–
E17	C18	A13	VDDIODDR	DDR	DDR_CLK	–	–	–	–	–	–	–	–
D17	C17	B13	VDDIODDR	DDR	DDR_CLKN	–	–	–	–	–	–	–	–
F16	F18	E14	VDDIODDR	DDR	DDR_CKE	–	–	–	–	–	–	–	–
E16	F17	D13	VDDIODDR	DDR	DDR_RESETN	–	–	–	–	–	–	–	–
G13	J12	F11	VDDIODDR	DDR	DDR_CS	–	–	–	–	–	–	–	–
F15	D18	A14	VDDIODDR	DDR	DDR_WE	–	–	–	–	–	–	–	–
F13	E18	C14	VDDIODDR	DDR	DDR_RAS	–	–	–	–	–	–	–	–
G12	E17	C13	VDDIODDR	DDR	DDR_CAS	–	–	–	–	–	–	–	–
C11	D11	D8	VDDIODDR	DDR	DDR_DQM0	–	–	–	–	–	–	–	–
G15	H14	G14	VDDIODDR	DDR	DDR_DQM1	–	–	–	–	–	–	–	–
C8	B8	–	VDDIODDR	DDR	DDR_DQM2	–	–	–	–	–	–	–	–
H11	L13	–	VDDIODDR	DDR	DDR_DQM3	–	–	–	–	–	–	–	–
B13	A14	B8	VDDIODDR	DDR	DDR_DQS0	–	–	–	–	–	–	–	–
J17	H18	K14	VDDIODDR	DDR	DDR_DQS1	–	–	–	–	–	–	–	–
C10	A10	–	VDDIODDR	DDR	DDR_DQS2	–	–	–	–	–	–	–	–
L17	M17	–	VDDIODDR	DDR	DDR_DQS3	–	–	–	–	–	–	–	–
B14	B14	A8	VDDIODDR	DDR	DDR_DQSN0	–	–	–	–	–	–	–	–
J16	J18	K13	VDDIODDR	DDR	DDR_DQSN1	–	–	–	–	–	–	–	–

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
B10	B10	–	VDDIODDR	DDR	DDR_DQSN2	–	–	–	–	–	–	–	–
L16	L17	–	VDDIODDR	DDR	DDR_DQSN3	–	–	–	–	–	–	–	–
H12	H13	F13	VDDIODDR	DDR	DDR_BA0	–	–	–	–	–	–	–	–
H13	K12	G13	VDDIODDR	DDR	DDR_BA1	–	–	–	–	–	–	–	–
F17	H17	F14	VDDIODDR	DDR	DDR_BA2	–	–	–	–	–	–	–	–
E13	G17	F10	VDDIODDR	DDR	DDR_CAL	–	–	–	–	–	–	–	–
L15, J15, H15, E15, D15, D12, D11	B17, E11, E14, F10, G11, G15, L14	C6, E10, E12, G10, G12, H11, J10	VDDIODDR	power	VDDIODDR	I	–	–	–	–	–	–	–
L14, J14, H14, E14, D14, E12, E11	B18, E10, E15, F11, G10, G14, L15	C7, D12, E9, F12, G11, H10, J11	GNDIODDR	power	GNDIODDR	I	–	–	–	–	–	–	–
H3, N5, N9, K13, D9, D7	H8, J6, J9, K8, L8	E8, G8, H8, H9, J5	VDDCORE	power	VDDCORE	I	–	–	–	–	–	–	–
H4, M5, M9, K12, E9, E7	H9, J7, J8, K7, L7	F8, G7, G9, H7, J4	GNDCORE	ground	GNDCORE	I	–	–	–	–	–	–	–
E6, F7	B1, D5	D7, F4	VDDIOP0	power	VDDIOP0	I	–	–	–	–	–	–	–
F6, G7	B2, D4	E4, E7	GNDIOP0	ground	GNDIOP0	I	–	–	–	–	–	–	–
R14, N13	T18, V16	K8, L11	VDDIOP1	power	VDDIOP1	I	–	–	–	–	–	–	–
M13, P14	T17, V15	K9, L12	GNDIOP1	ground	GNDIOP1	I	–	–	–	–	–	–	–
F10	D8	–	VDDIOP2	power	VDDIOP2	I	–	–	–	–	–	–	–
F9	E8	–	GNDIOP2	ground	GNDIOP2	I	–	–	–	–	–	–	–
P11	R11	–	VDDSDMMC	power	VDDSDMMC	I	–	–	–	–	–	–	–
R11	R12	–	GNDSDMMC	ground	GNDSDMMC	I	–	–	–	–	–	–	–
F4	–	–	VDDISC	power	VDDISC	I	–	–	–	–	–	–	–
G4	–	–	GNDISC	ground	GNDISC	I	–	–	–	–	–	–	–
M12	R17	K11	VDDFUSE	power	VDDFUSE	I	–	–	–	–	–	–	–

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
U4	V5	P3	VDDPLLA	power	VDDPLLA	I	–	–	–	–	–	–	–
U5	U6	P4	GNDPLLA	ground	GNDPLLA	I	–	–	–	–	–	–	–
T3	M7	K6	VDDAUDIOPLL	power	VDDAUDIOPLL	I	–	–	–	–	–	–	–
T5	P7	L6	GNDDPLL	ground	GNDDPLL	I	–	–	–	–	–	–	–
T4	N6	J6	GNDAUDIOPLL	ground	GNDAUDIOPLL	I	–	–	–	–	–	–	–
U3	M8	J7	VDDAUDIOPLL	–	CLK_AUDIO	–	–	–	–	–	–	–	–
U7	V7	P5	VDDOSC	–	XIN	–	–	–	–	–	–	–	–
U6	V6	P6	VDDOSC	–	XOUT	–	–	–	–	–	–	–	–
T7	R8	N5	VDDOSC	–	VDDOSC	–	–	–	–	–	–	–	–
T6	U5	N6	GNDOSC	power	GNDOSC	I	–	–	–	–	–	–	–
P8	N8	K7	VDDUTMII	power	VDDUTMII	I	–	–	–	–	–	–	–
R9	P9	–	VDDHSIC	power	VDDHSIC	I	–	–	–	–	–	–	–
P9	N9	L8	GNDUTMII	power	GNDUTMII	I	–	–	–	–	–	–	–
T8	U8	N7	VDDUTMII	–	HHSDPA	I	–	–	–	–	–	–	–
R8	V8	P7	VDDUTMII	–	HHSDMA	–	–	–	–	–	–	–	–
U8	U9	N8	VDDUTMII	–	HHSDPB	–	–	–	–	–	–	–	–
U9	V9	P8	VDDUTMII	–	HHSDMB	–	–	–	–	–	–	–	–
T9	U10	–	VDDHSIC	–	HHSDPDATC	–	–	–	–	–	–	–	–
U10	V10	–	VDDHSIC	–	HHSDMSTRC		–	–	–	–	–	–	–
P7	P8	M7	VDDUTMIC	power	VDDUTMIC	I	–	–	–	–	–	–	–
R7	U7	M8	GNDUTMIC	power	GNDUTMIC	I	–	–	–	–	–	–	–
T10	N10	–	VDDSDMMC	–	SDCAL	–	–	–	–	–	–	–	–
R6	R7	L7	VDDUTMIC	–	VBG	–	–	–	–	–	–	–	–
P3	P4	M2	VDDBU	–	TST	–	–	–	–	–	–	–	–
U2	V1	N3	VDDBU	–	NRST	–	–	–	–	–	–	–	–
T2	V2	L4	VDDBU	–	JTAGSEL	–	–	–	–	–	–	–	–
P4	R5	P1	VDDBU	–	WKUP	–	–	–	–	–	–	–	–
N4	U2	–	VDDBU	–	RXD	–	–	–	–	–	–	–	–
R1	U1	N1	VDDBU	–	SHDN	–	–	–	–	–	–	–	–
R3	R6	K5	VDDBU	–	PIOBU0	–	–	–	–	–	–	–	–
N8	R4	L3	VDDBU	–	PIOBU1	–	–	–	–	–	–	–	–
R2	–	M3	VDDBU	–	PIOBU2	–	–	–	–	–	–	–	–
R5	–	N4	VDDBU	–	PIOBU3	–	–	–	–	–	–	–	–
R4	–	L5	VDDBU	–	PIOBU4	–	–	–	–	–	–	–	–
P5	–	M6	VDDBU	–	PIOBU5	–	–	–	–	–	–	–	–
P6	–	–	VDDBU	–	PIOBU6	–	–	–	–	–	–	–	–
M8	–	–	VDDBU	–	PIOBU7	–	–	–	–	–	–	–	–
N7	U3	M4	VDDBU	power	VDDBU	I	–	–	–	–	–	–	–

Table 5-2. Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) (Continued)

289-pin BGA	256-pin BGA	196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
					Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
N6	U4	M5	GNDBU	ground	GNDBU	I	–	–	–	–	–	–	–
P1	T2	M1	VDDBU	–	XIN32	–	–	–	–	–	–	–	–
P2	R2	L1	VDDBU	–	XOUT32	–	–	–	–	–	–	–	–
T1	V3	N2	VDDBU	–	COMPP	I	–	–	–	–	–	–	–
U1	V4	P2	VDDBU	–	COMPN	I	–	–	–	–	–	–	–

Note: 1. Signal = 'PIO' if GPIO; Dir = Direction; PU = Pull-up; PD = Pull-down; HiZ = High impedance; ST = Schmitt Trigger

The SAMA5D23 pin description is identical to the SAMA5D21/SAMA5D2 pin description with the exception of the pins listed in Table 5-3.

Table 5-3. Pin Description (SAMA5D23 pins different from those in SAMA5D21/SAMA5D22)

196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
			Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
N4	GNDBU	ground	GNDBU	I	–	–	–	–	–	–	–
M6	GNDDPLL	ground	GNDDPLL	I	–	–	–	–	–	–	–
M3	JTAGSEL	–	JTAGSEL	–	–	–	–	–	–	–	–
K11	VDDIOP1	GPIO	PA31	I/O	–	–	B	NCS3	O	1	PIO, I, PU, ST
							C	SPI0_MISO	I/O	2	
							D	PWML0	O	1	
							F	CLASSD_L3	O	1	
D6	VDDIOP0	GPIO	PB0	I/O	–	–	B	A21/NANDALE	O	1	PIO, I, PU, ST
							C	SPI0_MOSI	I/O	2	
							D	PWMH1	O	1	
A6	VDDIOP0	GPIO	PB2	I/O	–	–	B	NRD/NANDOE	O	1	PIO, I, PU, ST
							D	PWMFI0	I	1	
							F	CLASSD_R1	O	1	
B6	VDDIOP0	GPIO	PB3	I/O	–	–	A	URXD4	I	1	PIO, I, PU, ST
							B	D8	I/O	1	
							C	IRQ	I	3	
							D	PWMEXTRG1	I	1	
							F	CLASSD_R2	O	1	
B5	VDDIOP0	GPIO_QSPI	PB5	I/O	–	–	A	TCLK2	I	1	PIO, I, PU, ST
							B	D10	I/O	1	
							C	PWMH2	O	1	
							D	QSPI1_SCK	O	2	
							F	GTSUCOMP	O	3	

Table 5-3. Pin Description (SAMA5D23 pins different from those in SAMA5D21/SAMA5D22)

196-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
			Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
M12	VDDIOP1	GPIO	PC0	I/O	–	–	A	LCDDAT21	O	1	PIO, I, PU, ST
							B	A23	O	1	
							C	FLEXCOM0_IO4	O	1	
							D	TWCK0	I/O	1	
							F	ISC_D6	I	3	
M13	VDDIOP1	GPIO	PC1	I/O	–	–	A	LCDDAT22	O	1	PIO, I, PU, ST
							B	A24	O	1	
							C	CANTX0	O	1	
							D	SPI1_SPCK	I/O	1	
							E	I2SC0_CK	I/O	1	
							F	ISC_D7	I	3	
L4	VDDBU	–	PIOBU1	–	–	–	–	–	–	–	–
L3	VDDBU	–	PIOBU2	–	–	–	–	–	–	–	–
M5	VDDBU	–	PIOBU3	–	–	–	–	–	–	–	–
L6	VDDBU	–	PIOBU5	–	–	–	–	–	–	–	–
P13	VDDFUSE	power	VDDFUSE	I	–	–	–	–	–	–	–

Note: 1. Signal = ‘PIO’ if GPIO; Dir = Direction; PU = Pull-up; PD = Pull-down; HiZ = High impedance; ST = Schmitt Trigger

The SAMA5D28B pin description is identical to the SAMA5D28A pin description with the exception of the pins listed in Table 5-4.

Table 5-4. Pin Description (SAMA5D28B pins different from those in SAMA5D28A)

289-pin BGA	Power Rail	I/O Type	Primary		Alternate		PIO peripheral				Reset State (Signal, Dir, PU, PD, HiZ, ST)[1]
			Signal	Dir	Signal	Dir	Func	Signal	Dir	IO Set	
P4	VDDCORE	power	VDDCORE	I	–	–	–	–	–	–	–
N5	GNDCORE	ground	GNDCORE	I	–	–	–	–	–	–	–
R2	VDDBU	–	WKUP	–	–	–	–	–	–	–	–
N6	VDDBU	–	PIOBU0	–	–	–	–	–	–	–	–
M8	VDDBU	–	PIOBU2	–	–	–	–	–	–	–	–
P6	VDDBU	–	PIOBU3	–	–	–	–	–	–	–	–
P5	VDDBU	–	PIOBU4	–	–	–	–	–	–	–	–
R5	VDDBU	–	PIOBU5	–	–	–	–	–	–	–	–
N7	VDDBU	–	PIOBU6	–	–	–	–	–	–	–	–
M5	VDDBU	–	PIOBU7	–	–	–	–	–	–	–	–
R3	VDDBU	power	VDDBU	I	–	–	–	–	–	–	–
R4	GNDBU	ground	GNDBU	I	–	–	–	–	–	–	–

Note: 1. Signal = ‘PIO’ if GPIO; Dir = Direction; PU = Pull-up; PD = Pull-down; HiZ = High impedance; ST = Schmitt Trigger

6. Power Considerations

6.1 Power Supplies

Table 6-1. SAMA5D2 Power Supplies

Name	Voltage Range, Nominal	Associated Ground	Powers
VDDCORE	1.10V – 1.32V, 1.20V	GNDCORE	Core, including the processor, the embedded memories and the peripherals
VDDPLLA	1.10V – 1.32V, 1.20V	GNDPLLA	PLLA Cell
VDDUTMIC	1.10V – 1.32V, 1.20V	GNDUTMII	USB device and host UTMI+ core
VDDHSIC	1.10V – 1.30V, 1.20V	GNDUTMII	USB High-Speed Inter-Chip
VDDIODDR	1.70V – 1.90V, 1.80V 1.14V – 1.30V, 1.20V 1.29V – 1.45V, 1.35V 1.43V – 1.57V, 1.50V	GNDIODDR	LPDDR1 / DDR2 Interface I/O lines LPDDR2 / LPDDR3 Interface I/O lines DDR3L Interface I/O lines DDR3 Interface I/O lines
VDDIOP0	1.65V – 3.60V	GNDIOP0	Peripheral I/O lines
VDDIOP1	1.65V – 3.60V	GNDIOP1	Peripheral I/O lines
VDDIOP2	1.65V – 3.60V	GNDIOP2	Peripheral I/O lines
VDDISC	1.65V – 3.60V	GNDISC	Image Sensor I/O lines
VDDSDMMC	1.65V – 3.60V	GNDSDMMC	SDMMC I/O lines
VDDUTMII	3.00V – 3.60V, 3.30V	GNDUTMII	USB device and host UTMI+ interface
VDDOSC	1.65V – 3.60V	GNDOSC	Main Oscillator Cell and PLL UTMI. If PLL UTMI or USB is used, the range is restricted to 3.00V–3.60V
VDDAUDIOPLL	3.00V – 3.60V, 3.30V	GNDAUDIOPLL GNDDPLL	Audio PLL
VDDANA	1.65V – 3.60V, 3.30V	GNDANA	VDD Analog
VDDFUSE	2.25V – 2.75V, 2.50V	GNDFUSE	Fuse box for programming. It can be tied to ground with a 100 Ω resistor for fuse reading only. It must be powered for fuse programming and to switch to Secure Mode.
VDDBU	1.65V – 3.60V	GNDBU	Slow Clock Oscillator, the internal 32-kHz RC Oscillator and a part of the System Controller

6.2 Powerup Considerations

At powerup, from a supply sequencing perspective, the SAMA5D2 power supply inputs are categorized into two groups:

- Group 1 (core group) contains VDDCORE, VDDUTMIC, VDDHSIC and VDDPLLA.
- Group 2 (periphery group) contains all other power supply inputs except VDDFUSE.

Figure 6-1 shows the recommended powerup sequence. Note that:

- VDDBU, when supplied from a battery, is an always-on supply input and is therefore not part of the power supply sequencing. When no backup battery is present in the application, VDDBU is part of Group 2.
- VDDFUSE is the only power supply that may be left unpowered during operation. This is possible if and only if the application does not access the Customer Fuse Matrix in Write mode. It is good practice to turn on

VDDFUSE only when the Customer Fuse Matrix is accessed in Write mode, and to turn off VDDFUSE otherwise.

- VDDIODDR may be nominally supplied at 1.2V when the SAMA5D2 device is equipped with an LPDDR2 or LPDDR3 memory. In this case, VDDIODDR can be considered as part of Group 1.

Figure 6-1. Recommended Powerup Sequence

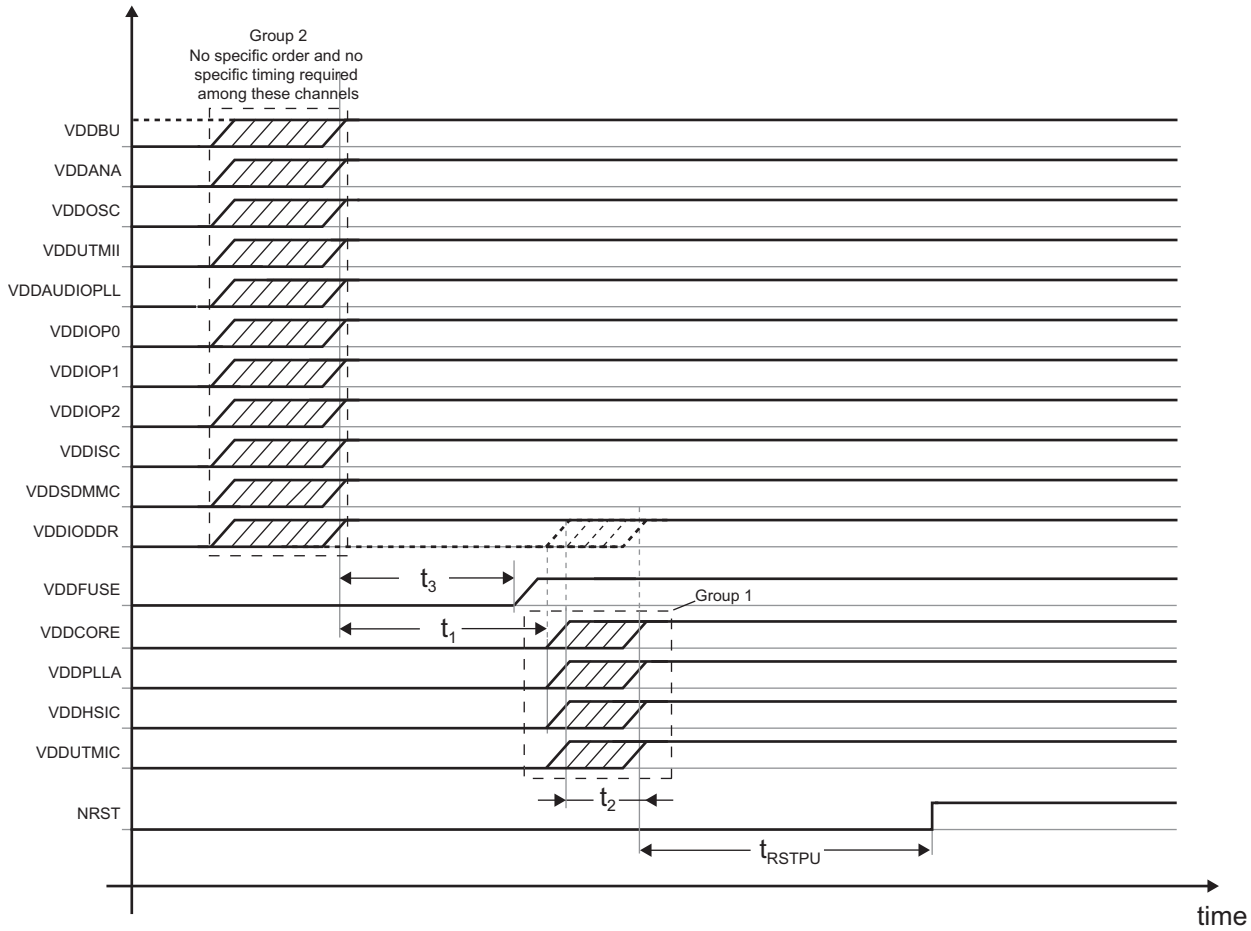


Table 6-2. Powerup Timing Specification

Symbol	Parameter	Conditions	Min	Max	Unit
t_1	Group 2 to Group 1 delay	Delay from the last Group 2 established[1] supply to the first Group1 supply turn-on	1	–	ms
t_2	Group 1 delay[2]	Delay from the first group 1 established supply to the last Group 1 established supply	–	1	
t_3	VDDFUSE to Group 1 delay	Delay from the last Group 2 established supply to VDDFUSE turn-on	1	–	
t_{RSTPU}	Reset delay at powerup	From the last established supply to NRST high	1	–	

Notes: 1. An “established” supply refers to a power supply established at 90% of its final value.
2. Also applies to VDDIODDR when considered as part of Group 1.

6.3 Powerdown Considerations

Figure 6-2 shows the SAMA5D2 powerdown sequence that starts by asserting the NRST line to 0. Once NRST is asserted, the supply inputs can be immediately shutdown without any specific timing or order. VDDBU may not be shutdown if the application uses a backup battery on this supply input. In applications where VDDFUSE is powered, it is mandatory to shutdown VDDFUSE prior to removing any other supply. VDDFUSE can be removed before or after asserting the NRST signal.

Figure 6-2. Recommended Powerdown Sequence

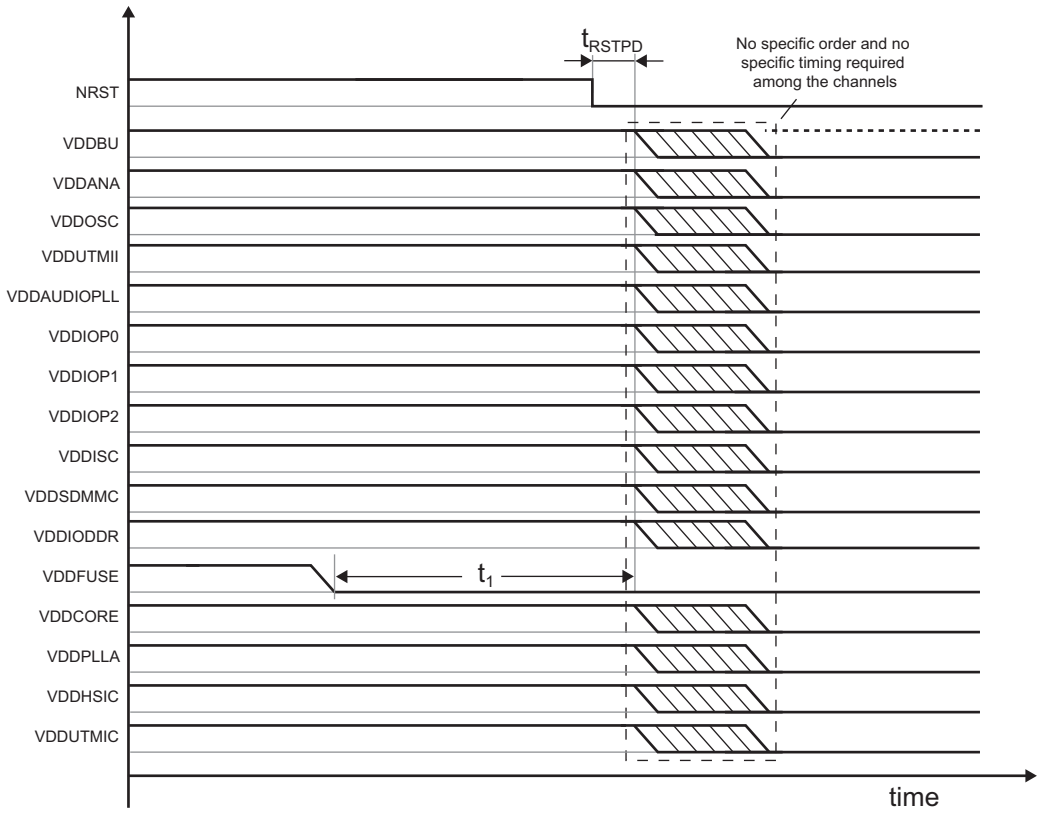


Table 6-3. Powerdown Timing Specification

Symbol	Parameter	Conditions	Min	Max	Unit
t_{RSTPD}	Reset delay at powerdown	From NRST low to the first supply turn-off	0	–	ms
t_1	VDDFUSE delay at shutdown	From VDDFUSE < 1V to the first supply turn-off	0	–	

6.4 Power Supply Sequencing at Backup Mode Entry and Exit

6.4.1 VDDBU Power Architecture

The backup power switch aims at optimizing the power consumption on VDDBU source by switching the supply of the backup digital part (BUREG memories + 64-kHz RC oscillator) to VDDANA.

When enabled, the backup power source can be automatically switched to VDDANA, which reduces power consumption on VDDBU. Then, VDDBU powers the pads, VDDBU POR, 32-kHz crystal and, on secure products SAMA5D23 and SAMA5D28, the temperature sensor and the backup supply monitor.

The power source (VDDANA or VDDBU) can be selected manually or can be set to work automatically by programming an SFRBU register (see SFRBU_PSWBUCTRL in Section 19. “Special Function Registers Backup (SFRBU)”).

6.4.2 Backup Mode Entry

Figure 6-3 shows the recommended power down sequence to place the SAMA5D2 either in Backup mode or in Backup mode with its DDR in self-refresh. The SHDN signal, output of Shutdown Controller (SHDWC), signals the shutdown request to the power supply. This output is supplied by VDDBU that is present in Backup mode. Placing the external DDR memory in self-refresh while in Backup mode, requires to maintain also VDDIODDR. One possible way to signal this additional need to the power supply is to position one of the general purpose I/Os supplied by VDDBU (PIOBUx) in a predefined state.

Figure 6-3. Recommended Backup Mode Entry

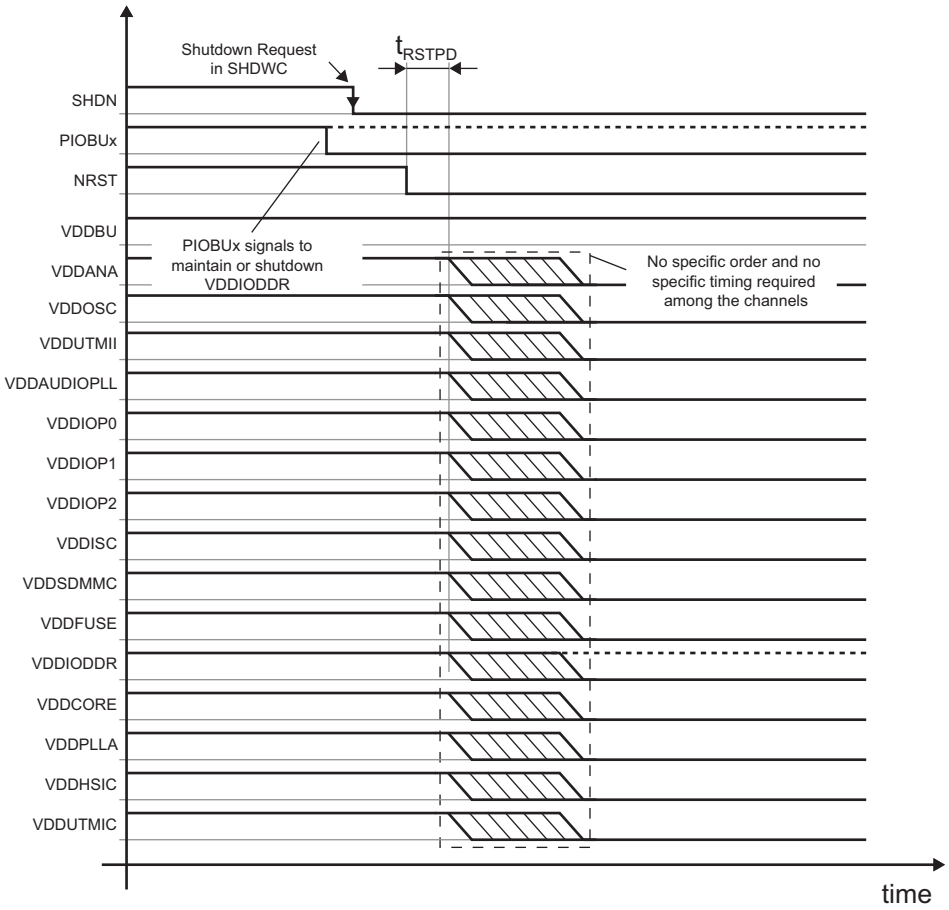


Table 6-4. Powerdown Timing Specification

Symbol	Parameter	Conditions	Min	Max	Unit
t_{RSTPD}	Reset delay at powerdown	From NRST low to the first supply turn-off	0	–	ms

6.4.3 Backup Mode Exit (Wakeup)

Figure 6-4 shows the recommended powerup sequence to wake up SAMA5D2 from Backup mode. Upon a wakeup event, the Shutdown Controller toggles its SHDN output back to VDDBU to request the power supply to restart. Except VDDIODDR which may already be present if the external DDR memory was placed in Self-refresh mode, this powerup sequence is the same as the one of Figure 6-1. In particular, the definitions of Group 1 and Group 2 are the same.

Figure 6-4. Recommended Power Supply Sequencing at Wakeup

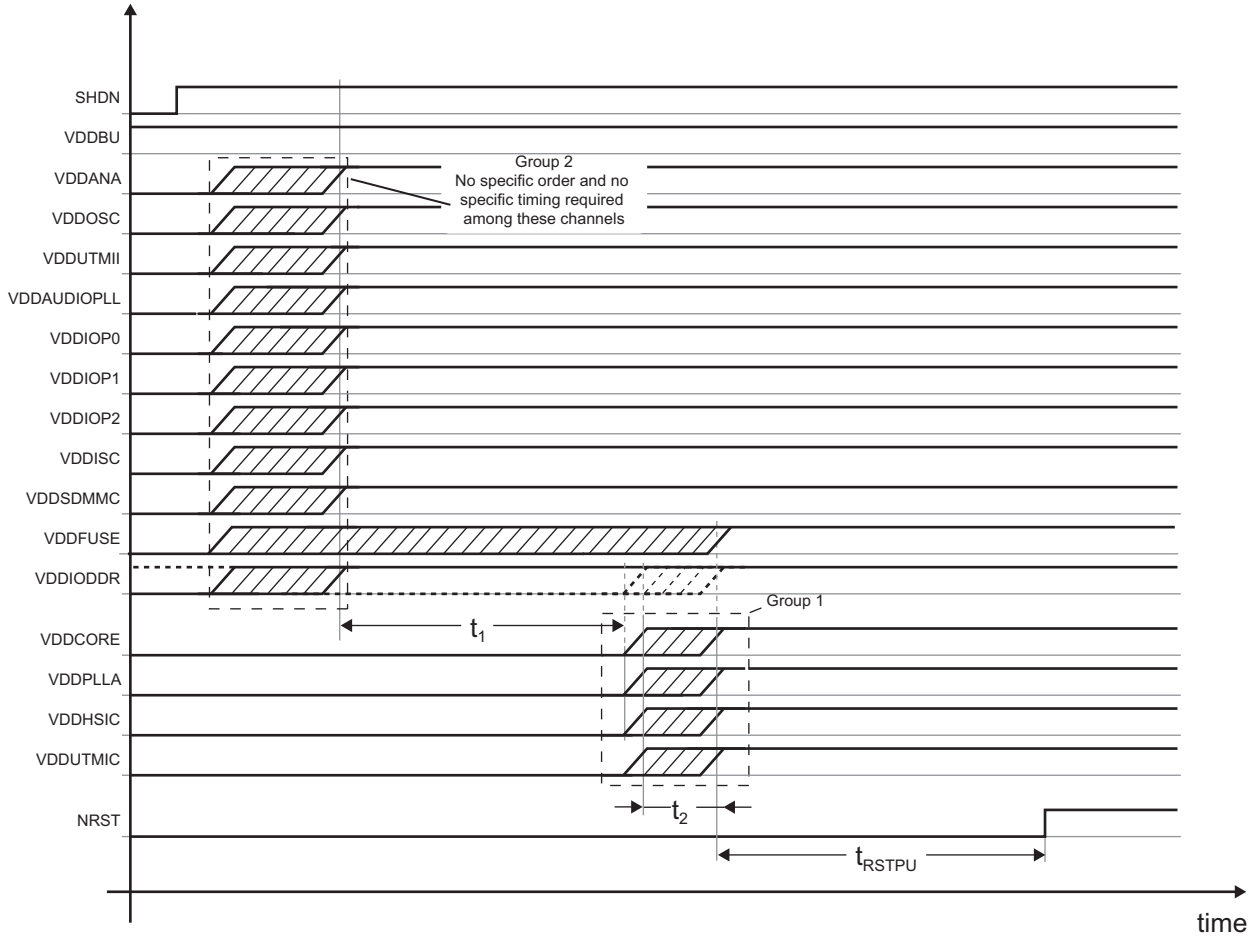


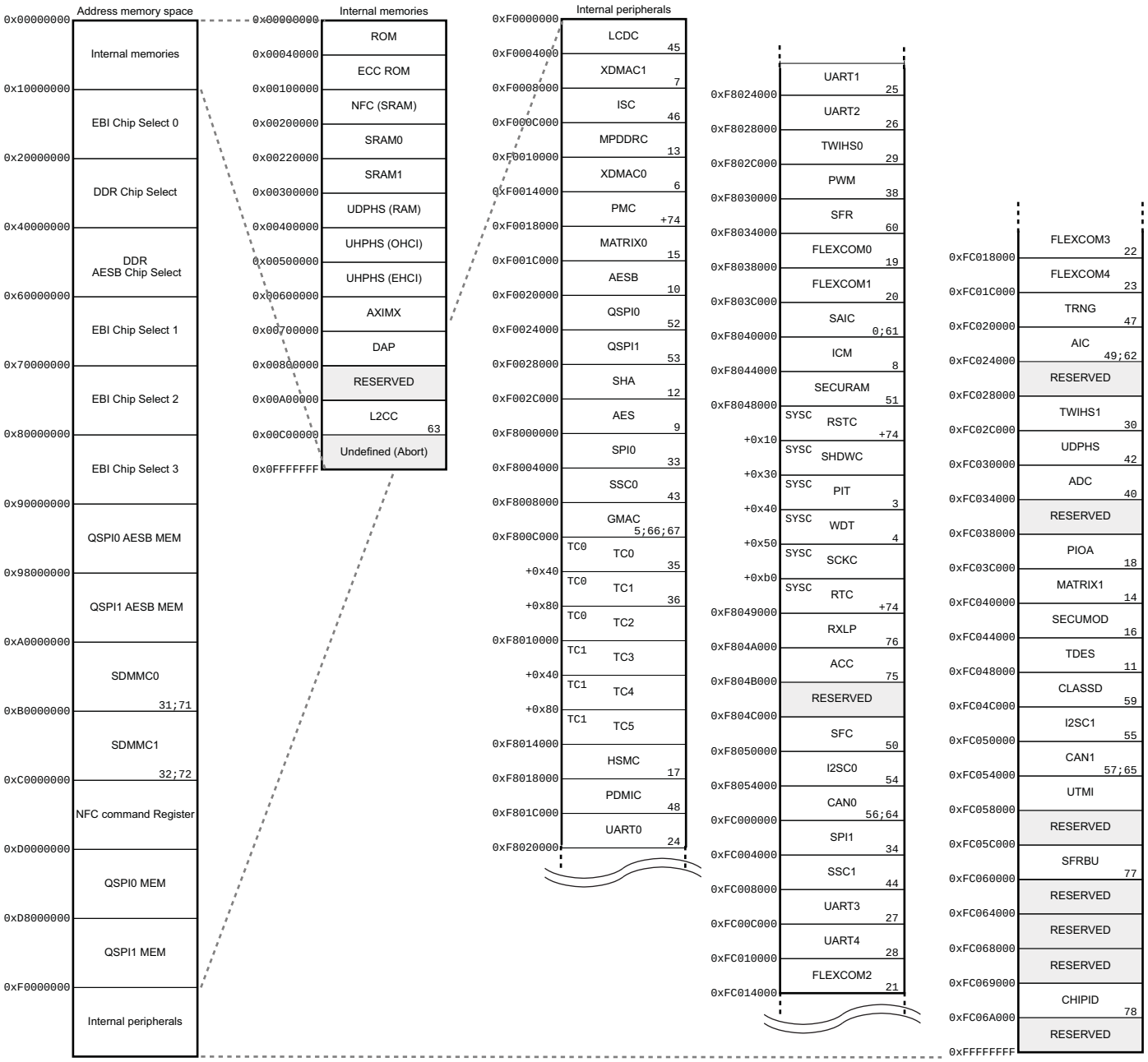
Table 6-5. Powerup Timing Specification

Symbol	Parameter	Conditions	Min	Max	Unit
t_1	Group 2 to Group 1 delay	Delay from the last Group 2 established(1) supply to the first Group1 supply turn-on	1	–	ms
t_2	Group 1 delay(2)	Delay from the first group 1 established supply to the last Group 1 established supply	–	1	
t_{RSTPU}	Reset delay at powerup	From the last established supply to NRST high.	1	–	

Notes: 1. An “established” supply refers to a power supply established at 90% of its final value.
2. Also applies to VDDIODDR when considered as part of Group 1.

7. Memories

Figure 7-1. Memory Mapping



7.1 Embedded Memories

7.1.1 Internal SRAM

The SAMA5D2 embeds a total of 128 Kbytes of high-speed SRAM. After reset, and until the Remap command is performed, the SRAM is accessible at address 0x0020 0000. When the AXI Bus Matrix is remapped, the SRAM is also available at address 0x0.

The device features a second 128-Kbyte SRAM that can be allocated either to the L2 cache controller or used as an internal SRAM. After reset, this block is connected to the system SRAM, making the two 128-Kbyte RAMs contiguous. The SRAM_SEL bit, located in the SFR_L2CC_HRAMC register, is used to reassign this memory as a L2 cache memory.

7.1.2 Internal ROM

The product embeds one 160-Kbyte secured internal ROM mapped at address 0 after reset. The ROM contains a standard and secure bootloader as well as the BCH (Bose, Chaudhuri and Hocquenghem) code tables for NAND Flash ECC correction. The memory area containing the secure boot is automatically hidden after the execution of the secure boot while the one containing the code tables for ECC remains visible.

7.1.3 Boot Strategies

For standard boot strategies, refer to Section 15. “Standard Boot Strategies” of this datasheet.
For secure boot strategies, refer to Application Note “SAMA5D2x Secure Boot Strategy”, literature No. 44040 (Non-Disclosure Agreement required).

7.2 External Memory

The SAMA5D2 offers connections to a wide range of external memories or to parallel peripherals.

7.2.1 External Bus Interface

The External Bus Interface (EBI) is a 16-bit wide interface working at MCK/2.

The EBI supports:

- Static memories
- 8-bit NAND Flash with 32-bit BCH ECC
- 16-bit NAND Flash

EBI I/Os accept three drive levels (Low, Medium, High) to avoid overshoots and provide the best performances according to the bus load and external memories voltage.

The drive levels are configured with the DRVSTR field in the PIO Configuration Register (PIO_CFGRx) if the corresponding line is nonsecure or the Secure PIO Configuration Register (S_PIO_CFGRx) if the I/O line is secure.

At reset, the selected drive is low. The user must make sure to program the correct drive according to the device load. The I/O embeds serial resistors for impedance matching.

7.2.2 Supported Memories on DDR2/DDR3/LPDDR1/LPDDR2/LPDDR3 Interface

- 16-bit or 32-bit external interface
- 512 Mbytes of address space on DDR CS and DDR/AES CS in 32-bit mode
- 256 Mbytes of address space on DDR CS and DDR/AES CS in 16-bit mode
- Supports 16-bit or 32-bit 8-bank DDR2, DDR3, LPDDR1, LPDDR2 and LPDDR3 memories
- Automatic drive level control
- Multiport

- Scramblable data path
- Port 0 of this interface has an embedded automatic AES encryption and decryption mechanism (see Section 56. “Advanced Encryption Standard Bridge (AESB)”). Writing to or reading from the address 0x40000000 may trigger the encryption and decryption mechanism depending on the AESB on External Memories configuration.
- TrustZone: The multiport feature of this interface implies TrustZone configuration constraints. See Section 17.12 “TrustZone Extension to AHB and APB” for more details.

7.2.3 Supported Memories on Static Memories and NAND Flash Interfaces

The Static Memory Controller is dedicated to interfacing external memory devices:

- Asynchronous SRAM-like memories and parallel peripherals
- NAND Flash (MLC and SLC) 8-bit datapath

The Static Memory Controller is able to drive up to four chip select. NCS3 is dedicated to the NAND Flash control.

The HSMC embeds a NAND Flash Controller (NFC). The NFC can handle automatic transfers, sending the commands and address cycles to the NAND Flash and transferring the contents of the page (for read and write) to the NFC SRAM. It minimizes the processor overhead.

In order to improve overall system performance, the DATA phase of the transfer can be DMA-assisted. The static memory embeds the NAND Flash Error Correcting Code controller with the following features:

- Algorithm based on BCH codes
- Supports also SLC 1-bit (BCH 2-bit), SLC 4-bit (BCH 4-bit)
- Programmable Error Correcting Capability
  - 2-bit, 4-bit, 8-bit and 16-bit errors for 512 bytes/sector (4-Kbyte page)
  - 24-bit error for 1024 bytes/sector (8-Kbyte page)
- Programmable sector size: 512 bytes or 1024 bytes
- Programmable number of sectors per page: 1, 2, 4 or 8 blocks of data per page
- Programmable spare area size
- Supports spare area ECC protection
- Supports 8-Kbyte page size using 1024 bytes/sector and 4-Kbyte page size using 512 bytes/sector
- Error detection is interrupt-driven
- Provides hardware acceleration for error location
- Finds roots of error-locator polynomial
- Programmable number of roots

7.2.4 DDR and SDMMC I/Os Calibration

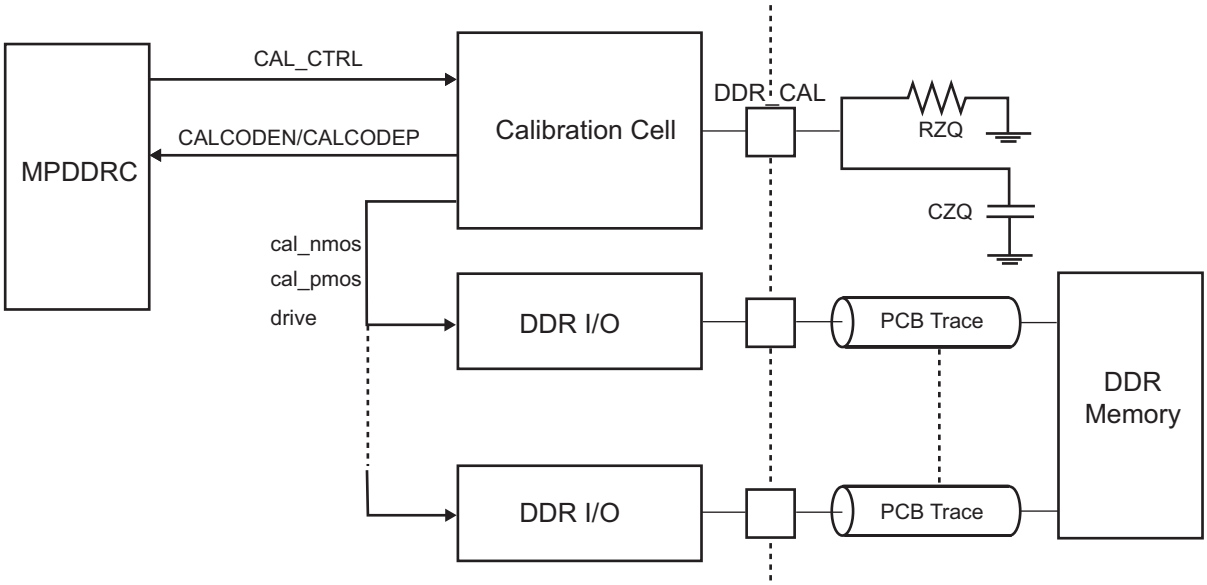
7.2.4.1 DDR I/O Calibration

The DDR2/DDR3/LPDDR1/LPDDR2/LPDDR3/DDR3L I/Os embed an automatic impedance matching control to avoid overshoots and reach the best performances according to the bus load and external memories. A serial termination connection scheme, where the driver has an output impedance matched to the characteristic impedance of the line, is used to improve signal quality and reduce EMI.

One specific analog input, DDR_CAL, is used to calibrate all DDR / IOs.

The MPDDRC supports the ZQ calibration procedure used to calibrate the SAMA5D2 DDR I/O drive strength and the commands to setup the external DDR device drive strength (refer to Section 33. “Multiport DDR-SDRAM Controller (MPDDRC)”). The calibration cell supports all the memory types listed above.

Figure 7-2. DDR Calibration Cell



The calibration cell provides an input pin, DDR_CAL, loaded with one of the following resistor RZQ values:

- 24 KΩ for LPDDR2/LPDDR3
- 23 KΩ for DDR3L
- 22 KΩ for DDR3
- 21 KΩ for DDR2/LPDDR1

The typical value for CZQ is 22 pF.

LPDDR2 Power Fail Management

The DDR controller (MPDDRC) is used to manage the LPDDR memory when an uncontrolled power off occurs.

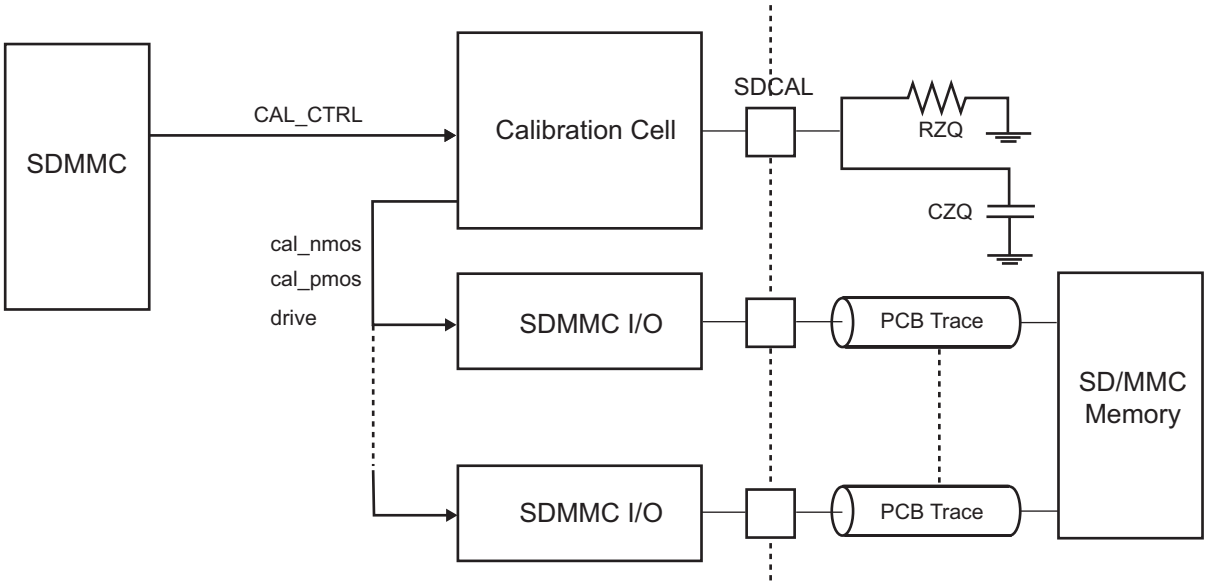
The DDR power rail must be monitored externally and generate an interrupt when a power fail condition is triggered. The interrupt handler must apply the sequence defined in the MPDDRC Low-power Register by setting bit LPDDR2_PWOFF (LPDDR2 Power Off bit).

7.2.4.2 SDMMC I/O Calibration

The SAMA5D2 embeds as well an SDMMC I/O calibration cell. The purpose of this block is to provide to e.MMC/SD I/Os an output impedance reference to limit the impact of process, voltage and temperature on the drivers output impedance. The impedance control is required at high frequency in order to improve signal quality.

The control and procedure to setup the SDMMC calibration cell is described in Section 48. “Secure Digital Multimedia Card Controller (SDMMC)”.

Figure 7-3. SDMMC I/O Calibration Cell



The calibration cell provides an input pin SDCAL loaded with a 20 KΩ resistor for 1.8V memories and a 16.9 KΩ resistor for 3.3V memories.

According to the e.MMC specification, the output impedance calibration is mandatory for HS200 mode (1.8V) when it is not for other modes (3.3V).

In the same way, according to the SD specification, the output impedance calibration is mandatory for 1.8V signaling when it is not for 3.3V signaling.

Thus, the calibration cell design is oriented to get the highest accuracy under 1.8V.

In case of interfacing which would need to operate under both 1.8V and 3.3V, external devices RZQ and CZQ must get values related to the 1.8V mode. The typical value for CZQ is 22 pF.

8. Event System

The events generated by peripherals are designed to be directly routed to peripherals managing/using these events without processor intervention. Peripherals receiving events contain logic by which to select the one required.

8.1 Real-time Event List

- Timers, PWM, IO peripherals generate event triggers which are directly routed to event managers such as ADC, for example, to start measurement/conversion without processor intervention.
- ADC is connected to nine trigger inputs defined as two groups:
  - One group of eight elements for Timer Counter (TC0 to TC4), ADTRIG and PMW0 event0, PWM0 event1
  - One group of one element for low-rate trigger, RTC
- UART, USART, SPI, TWI, PWM, CLASSD, AES, SHA, ADC, PIO, TIMER (Capture mode) generate event triggers directly connected to DMA controllers (XDMAC) for data transfer without processor intervention.
- PWM safety events (faults) are in combinational form and directly routed from event generators (ADC, ACC, PMC, TIMER) to the PWM module.
- PWM receives external triggers to provide PFC, DC/DC functions.
- PWM output comparators generate events directly connected to TIMER.
- PMC safety event (clock failure detection) can be programmed to switch the MCK on a reliable main RC internal clock without processor intervention.

8.2 Real-time Event Mapping

Table 8-1. Real-time Event Mapping List

Function	Application	Description	Event Source	Event Destination
Safety	General-purpose	Automatic switch to reliable main RC oscillator in case of main crystal clock failure[1]	Power Management Controller (PMC)	PMC
	General-purpose, motor control, power factor correction (PFC)	Puts the PWM outputs in Safe mode (main crystal clock failure detection)[1][2]		PWM
	Motor control, PFC	Puts the PWM outputs in Safe mode (overspeed, overcurrent detection, etc.)[2][3]	ADC	
	Motor control	Puts the PWM outputs in Safe mode (overspeed detection through TIMER quadrature decoder)[2][4]	Timer Counter Block (TC 0, 1, 2)	
			Timer Counter Block (TC 3, 4, 5)	
	General-purpose	Puts the PWM outputs in Safe mode (general-purpose fault inputs)[2]	2 IOs (PWM_Flx)	
Measurement trigger	General-purpose	Programmable delay in PWM[7]	PWM Event Line 0	ADC
			PWM Event Line 1	
	General-purpose	Trigger source selection in ADC[5]	IO (ADC_ADTRG)	
			TC Output 0	
			TC Output 1	
			TC Output 2	
			TC Output 3	
			TC Output 4	RTCOUT0
	General-purpose	Low-speed measurement[6]	RTC	RTCOUT1
GTSUCOMP synchronous clock generation trigger	Audio	Trigger source selection in TC	GMAC GTSUCOMP Line	TC5
Delay measurement	Motor control	Delay measurement between PWM outputs and TC inputs externally connected to power transistor bridge driver.[8][9]	PWM Compare Line 0	TC Input (A/B) 0
			PWM Compare Line 1	TC Input (A/B) 1
			PWM Compare Line 2	TC Input (A/B) 2

Notes:
1. Refer to Section 30.17 “Main Clock Failure Detector”.
2. Refer to Section 53.5.4 “Fault Inputs” and Section 53.6.2.7 “Fault Protection”.
3. Refer to Section 61.5.5 “Fault Output”.
4. Refer to Section 51.6.18 “Fault Mode”.
5. Refer to Section 61.7.25 “ADC Trigger Register”.
6. Refer to Section 25.5.8 “Waveform Generation”.
7. Refer to Section 53.6.3 “PWM Comparison Units” and Section 53.6.4 “PWM Event Lines”.
8. Refer to Section 51.6.14 “Synchronization with PWM”.
9. Refer to Section 53.6.2.2 “Comparator”.

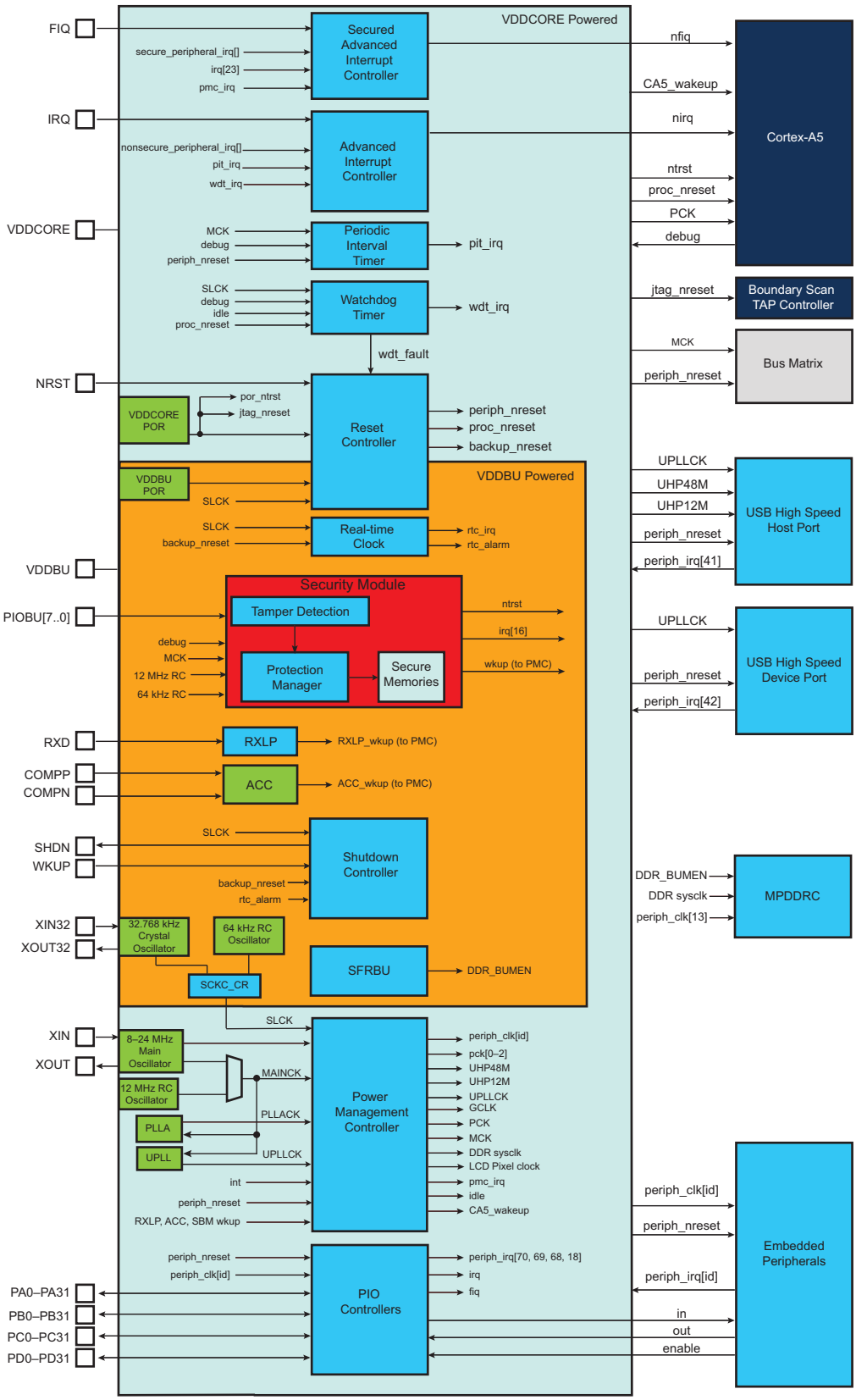
9. System Controller

The system controller is a set of peripherals handling key elements of the system, such as power, resets, clocks, time, interrupts, watchdog, etc.

The system controller's peripherals are all mapped between addresses 0xF8049000 and 0xF8048000.

Figure 9-1 shows the system controller block diagram.

Figure 9-1. System Controller Block Diagram



9.1 Power-On Reset

The SAMA5D2 embeds several Power-On Resets (PORs) to ensure the power supply is established when the reset is released. These PORs are dedicated to monitoring VDDBU, VDDIOP and VDDCORE respectively.

10. Peripherals

10.1 Peripheral Mapping

As shown in Figure 7-1. Memory Mapping, the peripherals are mapped in the upper 256 Mbytes of the address space, between addresses 0xF000 0000 and 0xFFFC 0000.

10.2 Peripheral Identifiers

Table 10-1. Peripheral identifiers

Instance ID	Instance Name	Internal Interrupt	PMC Clock Control	Instance Description	Clock Type	Security[2]	In Matrix
0	SAIC	FIQ	–	FIQ Interrupt ID	SYS_CLK_LS	AS	–
1	–	–	–	–	–	–	–
2	ARM	PMU	X	Performance Monitor Unit (PMU)	PROC_CLK	PS	H64
3	PIT	X	–	Periodic Interval Timer Interrupt	SYS_CLK_LS	PS	H32
4	WDT	X	–	Watchdog timer Interrupt	SYS_CLK_LS	PS	H32
5	GMAC	X	X	Ethernet MAC	HCLOCK_LS PCLOCK_LS	PS	H32
6	XDMAC0	X	X	DMA Controller 0	HCLOCK_HS	PS	H64
7	XDMAC1	X	X	DMA Controller 1	HCLOCK_HS	PS	H64
8	ICM	X	X	Integrity Check Monitor	HCLOCK_LS	PS	H32
9	AES	X	X	Advanced Encryption Standard	PCLK_HS	PS	H64
10	AESB	X	X	AES bridge	HCLOCK_HS	PS	H64
11	TDES	X	X	Triple Data Encryption Standard	PCLOCK_LS	PS	H32
12	SHA	X	X	SHA Signature	PCLK_HS	PS	H64
13	MPDDRC	X	X	MPDDR Controller	HCLOCK_HS	PS	H64
14	MATRIX1	X	X	H32MX, 32-bit AHB Matrix	SYS_CLK_LS	AS	H32
15	MATRIX0	X	X	H64MX, 64-bit AHB Matrix	SYS_CLOCK	AS	H64
16	SECUMOD	X	X	Secure Module	SLOW_CLOCK	AS	H32
17	HSMC	X	X	Multibit ECC Interrupt	HCLOCK_LS	PS	H32
18	PIOA	X	X	Parallel I/O Controller	PCLOCK_LS	AS	H32
19	FLEXCOM0	X	X	FLEXCOM 0	PCLOCK_LS	PS	H32
20	FLEXCOM1	X	X	FLEXCOM 1	PCLOCK_LS	PS	H32
21	FLEXCOM2	X	X	FLEXCOM 2	PCLOCK_LS	PS	H32
22	FLEXCOM3	X	X	FLEXCOM 3	PCLOCK_LS	PS	H32
23	FLEXCOM4	X	X	FLEXCOM 4	PCLOCK_LS	PS	H32
24	UART0	X	X	Universal Asynchronous Receiver Transmitter 0	PCLOCK_LS	PS	H32
25	UART1	X	X	Universal Asynchronous Receiver Transmitter 1	PCLOCK_LS	PS	H32

Table 10-1. Peripheral identifiers (Continued)

Instance ID	Instance Name	Internal Interrupt	PMC Clock Control	Instance Description	Clock Type	Security[2]	In Matrix
26	UART2	X	X	Universal Asynchronous Receiver Transmitter 2	PCLOCK_LS	PS	H32
27	UART3	X	X	Universal Asynchronous Receiver Transmitter 3	PCLOCK_LS	PS	H32
28	UART4	X	X	Universal Asynchronous Receiver Transmitter 4	PCLOCK_LS	PS	H32
29	TWIHS0	X	X	Two-Wire Interface 0	PCLOCK_LS	PS	H32
30	TWIHS1	X	X	Two-Wire Interface 1	PCLOCK_LS	PS	H32
31	SDMMC0	X	X	Secure Data Memory Card Controller 0	HCLOCK_HS	PS	H64
32	SDMMC1	X	X	Secure Data Memory Card Controller 1	HCLOCK_HS	PS	H64
33	SPI0	X	X	Serial Peripheral Interface 0	PCLOCK_LS	PS	H32
34	SPI1	X	X	Serial Peripheral Interface 1	PCLOCK_LS	PS	H32
35	TC0	X	X	Timer Counter 0 (ch. 0, 1, 2)	PCLOCK_LS	PS	H32
36	TC1	X	X	Timer Counter 1 (ch. 3, 4, 5)	PCLOCK_LS	PS	H32
37	–	–	–	–	–	–	–
38	PWM	X	X	Pulse Width Modulation Controller 0 (ch. 0, 1, 2, 3)	PCLOCK_LS	PS	H32
39	–	–	–	–	–	–	–
40	ADC	X	X	Touchscreen ADC Controller	PCLOCK_LS	PS	H32
41	UHPHS	X	X	USB Host High-Speed	HCLOCK_LS	PS	H32
42	UDPHS	X	X	USB Device High-Speed	HCLOCK_LS	PS	H32
43	SSC0	X	X	Synchronous Serial Controller 0	PCLOCK_LS	PS	H32
44	SSC1	X	X	Synchronous Serial Controller 1	PCLOCK_LS	PS	H32
45	LCDC	X	X	LCD Controller	HCLOCK_HS	PS	H64
46	ISC	X	X	Image Sensor Controller	HCLOCK_HS	PS	H64
47	TRNG	X	X	True Random Number Generator	PCLOCK_LS	PS	H32
48	PDMIC	X	X	Pulse Density Modulation Interface Controller	PCLOCK_LS	PS	H32
49	AIC	IRQ	–	IRQ Interrupt ID	SYS_CLK_LS	NS	H32
50	SFC	X	X	Secure Fuse Controller	PCLOCK_LS	PS	H32
51	SECURAM	X	X	Secured RAM	PCLOCK_LS	AS	H32
52	QSPI0	X	X	Quad SPI Interface 0	HCLOCK_HS	PS	H64
53	QSPI1	X	X	Quad SPI Interface 1	HCLOCK_HS	PS	H64
54	I2SC0	X	X	Inter-IC Sound Controller 0	PCLOCK_LS	PS	H32
55	I2SC1	X	X	Inter-IC Sound Controller 1	PCLOCK_LS	PS	H32
56	MCAN0	INT0	X	MCAN 0 Interrupt0	HCLOCK_LS	PS	H32
57	MCAN1	INT0	X	MCAN 1 Interrupt0	HCLOCK_LS	PS	H32

Table 10-1. Peripheral identifiers (Continued)

Instance ID	Instance Name	Internal Interrupt	PMC Clock Control	Instance Description	Clock Type	Security[2]	In Matrix
58	–	–	–	–	–	–	–
59	CLASSD	X	X	Audio Class D Amplifier	PCLOCK_LS	PS	H32
60	SFR	–	–	Special Function Register[2]	SYS_CLK_LS	PS	H32
61	SAIC	–	–	Secured Advanced Interrupt Controller[2]	SYS_CLK_LS	AS	H32
62	AIC	–	–	Advanced Interrupt Controller[2]	SYS_CLK_LS	NS	H32
63	L2CC	X	–	L2 Cache Controller	–	PS	H64
64	MCAN0	INT1	–	MCAN 0 Interrupt1	–	PS	H32
65	MCAN1	INT1	–	MCAN 1 Interrupt1	–	PS	H32
66	GMAC	Q1	–	GMAC Queue 1 Interrupt	–	PS	H32
67	GMAC	Q2	–	GMAC Queue 2 Interrupt	–	PS	H32
68	PIOB	X	–	–	–	AS	H32
69	PIOC	X	–	–	–	AS	H32
70	PIOD	X	–	–	–	AS	H32
71	SDMMC0	TIMER	–	–	–	PS	H32
72	SDMMC1	TIMER	–	–	–	PS	H32
73	–	–	–	–	–	–	–
74	PMC, RTC, RSTC	X	–	System Controller Interrupt	SYS_CLK_LS	PS	H32
75	ACC	X	–	Analog Comparator	SYS_CLK_LS	PS	H32
76	RXLP	X	–	UART Low-Power	SYS_CLK_LS	PS	H32
77	SFRBU	–	–	Special Function Register BackUp[2]	–	PS	H32
78	CHIPID	–	–	Chip ID	–	PS	H32

Notes: 1. AS = Always Secure; PS = Programmable Secure; NS = Never Secure
2. For security purposes, there is no matching clock but a peripheral ID only.

10.3 Peripheral Signal Multiplexing on I/O Lines

The SAMA5D2 features several PIO Controllers that multiplex the I/O lines of the peripheral set.

Table 5-2 Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A) defines how the I/O lines are multiplexed on the different PIO Controllers. Several I/O sets are available for each peripheral. However, selecting I/Os from different I/O sets for one peripheral is prohibited.

The column “Reset State” shows whether the PIO line resets in I/O mode or in Peripheral mode. If I/O is shown, the PIO line resets in input with the pull-up enabled, so that the device is maintained in a static state as soon as the reset is released. As a result, the bit corresponding to the PIO line in register PIO_CFGR (PIO Configuration Register) resets low.

If a signal name is shown in the “Reset State” column, the PIO line is assigned to this function and the corresponding bit in PIO_CFGR resets high. That is the case for pins controlling memories, in particular address lines, which require the pin to be driven as soon as the reset is released.

The PIO state can be retained when the system enters in Backup mode.

10.4 Peripheral Clock Types

The SAMA5D2 series embeds peripherals with the following clock types:

- HCLOCK_HS, HCLOCK_LS: AHB Clock, managed with the PMC_PCER, PMC_PCDR, PMC_PCSR and PMC_PCR registers of Peripheral Clock
- PCLOCK_HS, PCLOCK_LS: APB Clock, managed with the PMC_PCER, PMC_PCDR, PMC_PCSR and PMC_PCR registers of Peripheral Clock
- HCLOCK + PCLOCK: Both clock types coexist. The clock is managed with the PMC_PCER, PMC_PCDR, PMC_PCSR and PMC_PCR registers of Peripheral Clock
- SYS_CLK_LS: This clock cannot be disabled.
- SYS_CLOCK: This clock cannot be disabled.
- PROC_CLK: The clock related to Processor Clock (PCK) and managed with the PMC_SCDR and PMC_SCSR registers of PMC System Clock
- SLOW_CLOCK: The clock related to backup area and the RTC and managed with the SCKC_CR. This clock can be generated either by an external 32.768 kHz crystal oscillator or by the on-chip 64 kHz RC oscillator.

Refer to Table 10-1 Peripheral identifiers for details. In the table, clock type suffixes HS and LS refer to MATRIX0 and MATRIX1, respectively.

11. Chip Identifier (CHIPID)

11.1 Description

Chip Identifier (CHIPID) registers are used to recognize the device and its revision. These registers provide the sizes and types of the on-chip memories, as well as the set of embedded peripherals.

Two CHIPID registers are embedded: Chip ID Register (CHIPID_CIDR) and Chip ID Extension Register (CHIPID_EXID). Both registers contain a hard-wired value that is read-only.

The CHIPID_CIDR register contains the following fields:

- VERSION: Identifies the revision of the silicon
- EPROC: Indicates the embedded ARM processor
- NVPTYP and NVPSIZ: Identify the type of embedded non-volatile memory and the size
- SRAMSIZ: Indicates the size of the embedded SRAM
- ARCH: Identifies the set of embedded peripherals
- EXT: Shows the use of the extension identifier register

The CHIPID_EXID register is device-dependent and reads 0 if CHIPID_CIDR.EXT = 0.

11.2 Embedded Characteristics

- Chip ID Registers
  - Identification of the Device Revision, Sizes of the Embedded Memories, Set of Peripherals, Embedded Processor

Table 11-1. SAMA5D2 Chip ID Registers

Chip Name	CHIPID_CIDR	CHIPID_EXID
ATSAMA5D22A-CU	0x8A5C08C0	0x00000059
ATSAMA5D24A-CU	0x8A5C08C0	0x00000014
ATSAMA5D27A-CU	0x8A5C08C0	0x00000011
ATSAMA5D28A-CU	0x8A5C08C0	0x00000010
ATSAMA5D21B-CU	0x8A5C08C1	0x0000005A
ATSAMA5D22B-CN	0x8A5C08C1	0x00000069
ATSAMA5D22B-CU	0x8A5C08C1	0x00000059
ATSAMA5D23B-CN	0x8A5C08C1	0x00000068
ATSAMA5D23B-CU	0x8A5C08C1	0x00000058
ATSAMA5D24B-CU	0x8A5C08C1	0x00000014
ATSAMA5D26B-CN	0x8A5C08C1	0x00000022
ATSAMA5D26B-CU	0x8A5C08C1	0x00000012
ATSAMA5D27B-CN	0x8A5C08C1	0x00000021
ATSAMA5D27B-CU	0x8A5C08C1	0x00000011
ATSAMA5D28B-CN	0x8A5C08C1	0x00000020
ATSAMA5D28B-CU	0x8A5C08C1	0x00000010

11.3 Chip Identifier (CHIPID) User Interface

Table 11-2. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Chip ID Register	CHIPID_CIDR	Read-only	–
0x4	Chip ID Extension Register	CHIPID_EXID	Read-only	–

11.3.1 Chip ID Register

Name: CHIPID_CIDR

Address: 0xFC069000

Access: Read-only

31	30	29	28	27	26	25	24
EXT	NVPTYP			ARCH			

23	22	21	20	19	18	17	16
ARCH				SRAMSIZ			

15	14	13	12	11	10	9	8
NVPSIZ2				NVPSIZ			

7	6	5	4	3	2	1	0
EPROC			VERSION				

- **VERSION: Version of the Device**

Current version of the device.

- **EPROC: Embedded Processor**

Value	Name	Description
0	SAM x7	Cortex-M7
1	ARM946ES	ARM946ES
2	ARM7TDMI	ARM7TDMI
3	CM3	Cortex-M3
4	ARM920T	ARM920T
5	ARM926EJS	ARM926EJS
6	CA5	Cortex-A5
7	CM4	Cortex-M4

- **NVPSIZ: Nonvolatile Program Memory Size**

Value	Name	Description
0	NONE	None
1	8K	8 Kbytes
2	16K	16 Kbytes
3	32K	32 Kbytes
4	–	Reserved
5	64K	64 Kbytes
6	–	Reserved
7	128K	128 Kbytes
8	160K	160 Kbytes
9	256K	256 Kbytes
10	512K	512 Kbytes

Value	Name	Description
11	–	Reserved
12	1024K	1024 Kbytes
13	–	Reserved
14	2048K	2048 Kbytes
15	–	Reserved

- **NVPSIZ2: Second Nonvolatile Program Memory Size**

Value	Name	Description
0	NONE	None
1	8K	8 Kbytes
2	16K	16 Kbytes
3	32K	32 Kbytes
4	–	Reserved
5	64K	64 Kbytes
6	–	Reserved
7	128K	128 Kbytes
8	–	Reserved
9	256K	256 Kbytes
10	512K	512 Kbytes
11	–	Reserved
12	1024K	1024 Kbytes
13	–	Reserved
14	2048K	2048 Kbytes
15	–	Reserved

- **SRAMSIZ: Internal SRAM Size**

Value	Name	Description
0	48K	48 Kbytes
1	192K	192 Kbytes
2	384K	384 Kbytes
3	6K	6 Kbytes
4	24K	24 Kbytes
5	4K	4 Kbytes
6	80K	80 Kbytes
7	160K	160 Kbytes
8	8K	8 Kbytes
9	16K	16 Kbytes
10	32K	32 Kbytes
11	64K	64 Kbytes

Value	Name	Description
12	128K	128 Kbytes
13	256K	256 Kbytes
14	96K	96 Kbytes
15	512K	512 Kbytes

- **ARCH: Architecture Identifier**

Value	Name	Description
0xA5	SAMA5	SAMA5

- **NVPTYP: Nonvolatile Program Memory Type**

Value	Name	Description
0	ROM	ROM
1	ROMLESS	ROMless or on-chip Flash
2	FLASH	Embedded Flash Memory
3	ROM_FLASH	ROM and Embedded Flash Memory ● NVPSIZ is ROM size ● NVPSIZ2 is Flash size
4	SRAM	SRAM emulating ROM

- **EXT: Extension Flag**

0: Chip ID has a single register definition without extension.

1: An extended Chip ID exists.

11.3.2 Chip ID Extension Register

Name: CHIPID_EXID

Address: 0xFC069004

Access: Read-only

31	30	29	28	27	26	25	24
EXID							

23	22	21	20	19	18	17	16
EXID							

15	14	13	12	11	10	9	8
EXID							

7	6	5	4	3	2	1	0
EXID							

- **EXID: Chip ID Extension**

  This field is cleared if CHIPID_CIDR.EXT = 0.

12. ARM Cortex-A5

12.1 Description

The ARM Cortex-A5 processor is a high-performance, low-power, ARM macrocell with an L1 cache subsystem that provides full virtual memory capabilities. The Cortex-A5 processor implements the ARMv7 architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb instructions, and 8-bit Java™ byte codes in Jazelle® state.

The Cortex-A5 NEON Media Processing Engine (MPE) extends the Cortex-A5 functionality to provide support for the ARM v7 Advanced SIMD v2 and *Vector Floating-Point v4* (VFPv4) instruction sets. The Cortex-A5 NEON MPE provides flexible and powerful acceleration for signal processing algorithms including multimedia such as image processing, video decode/encode, 2D/3D graphics, and audio. See the *Cortex-A5 NEON Media Processing Engine Technical Reference Manual*.

The Cortex-A5 processor includes TrustZone® technology to enhance security by partitioning the SoC's hardware and software resources in a Secure world for the security subsystem and a Normal world for the rest, enabling a strong security perimeter to be built between the two. See *Security Extensions overview* in the *Cortex-A5 Technical Reference Manual*. See the *ARM Architecture Reference Manual* for details on how TrustZone works in the architecture.

Note: All ARM publications referenced in this datasheet can be found at www.arm.com.

12.1.1 Power Management

The Cortex-A5 design supports the following main levels of power management:

- Run Mode
- Standby Mode

12.1.1.1 Run Mode

Run mode is the normal mode of operation where all of the processor functionality is available. Everything, including core logic and embedded RAM arrays, is clocked and powered up.

12.1.1.2 Standby Mode

Standby mode disables most of the clocks of the processor, while keeping it powered up. This reduces the power drawn to the static leakage current, plus a small clock power overhead required to enable the processor to wake up from Standby mode. The transition from Standby mode to Run mode is caused by one of the following:

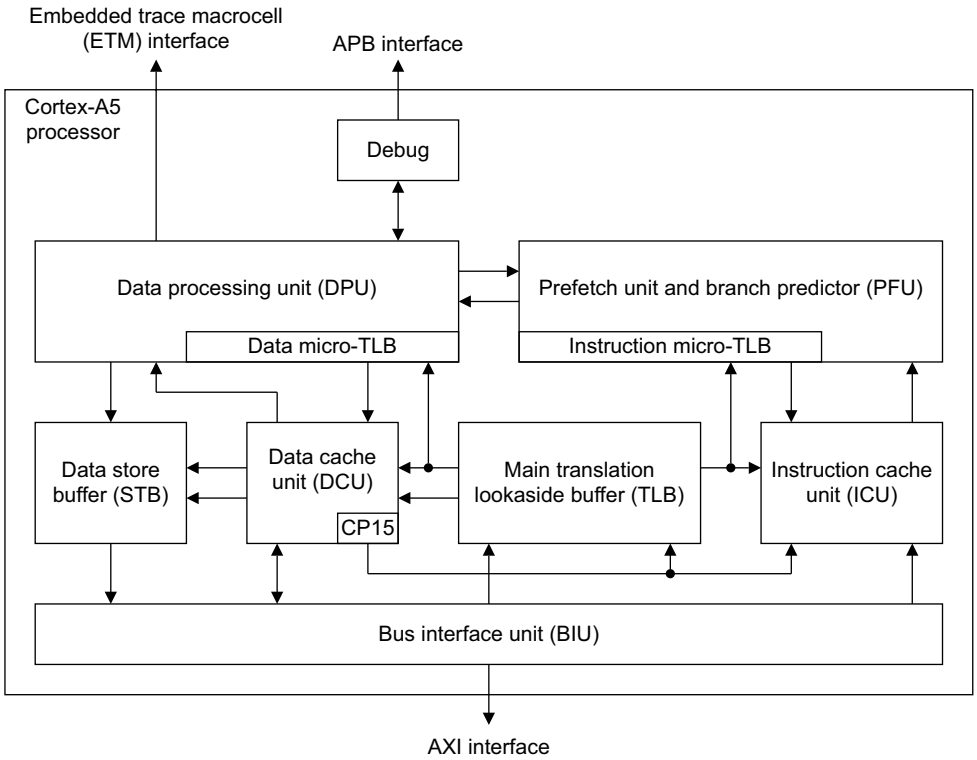
- the arrival of an interrupt, either masked or unmasked
- the arrival of an event, if standby mode was initiated by a Wait for Event (WFE) instruction
- a debug request, when either debug is enabled or disabled
- a reset.

12.2 Embedded Characteristics

- In-order pipeline with dynamic branch prediction
- ARM, Thumb, and ThumbEE instruction set support
- TrustZone security extensions
- Harvard level 1 memory system with a Memory Management Unit (MMU)
- 32-Kbyte Data Cache
- 32-Kbyte Instruction Cache
- 64-bit AXI master interface
- ARM v7 debug architecture
- Trace support through an Embedded Trace Macrocell (ETM) interface
- Media Processing Engine (MPE) with NEON technology
- Jazelle hardware acceleration

12.3 Block Diagram

Figure 12-1. Cortex-A5 Processor Top-level Diagram



12.4 Programmer Model

12.4.1 Processor Operating Modes

The following operating modes are present in all states:

- User mode (USR) is the usual ARM program execution state. It is used for executing most application programs.
- Fast Interrupt (FIQ) mode is used for handling fast interrupts. It is suitable for high-speed data transfer or channel process.
- Interrupt (IRQ) mode is used for general-purpose interrupt handling.
- Supervisor mode (SVC) is a protected mode for the operating system.
- Abort mode (ABT) is entered after a data or instruction prefetch abort.
- System mode (SYS) is a privileged user mode for the operating system.
- Undefined mode (UND) is entered when an undefined instruction exception occurs.
- Monitor mode (MON) is secure mode that enables change between Secure and Non-secure states, and can also be used to handle any of FIQs, IRQs and external aborts. Entered on execution of a Secure Monitor Call (SMC) instruction.

Mode changes may be made under software control, or may be brought about by external interrupts or exception processing. Most application programs execute in User Mode. The non-user modes, known as privileged modes, are entered in order to service interrupts or exceptions or to access protected resources.

12.4.2 Processor Operating States

The processor has the following instruction set states controlled by the T bit and J bit in the CPSR.

- ARM state:

  The processor executes 32-bit, word-aligned ARM instructions.
- Thumb state:

  The processor executes 16-bit and 32-bit, halfword-aligned Thumb instructions.
- ThumbEE state:

  The processor executes a variant of the Thumb instruction set designed as a target for dynamically generated code. This is code compiled on the device either shortly before or during execution from a portable bytecode or other intermediate or native representation.
- Jazelle state:

  The processor executes variable length, byte-aligned Java bytecodes.

The J bit and the T bit determine the instruction set used by the processor. Table 12-1 shows the encoding of these bits.

Table 12-1. CPSR J and T Bit Encoding

J	T	Instruction Set State
0	0	ARM
0	1	Thumb
1	0	Jazelle
1	1	ThumbEE

Changing between ARM and Thumb states does not affect the processor mode or the register contents. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for information on entering and exiting ThumbEE state.

12.4.2.1 Switching State

It is possible to change the instruction set state of the processor between:

- ARM state and Thumb state using the BX and BLX instructions.
- Thumb state and ThumbEE state using the ENTERX and LEAVEX instructions.
- ARM and Jazelle state using the BXJ instruction.
- Thumb and Jazelle state using the BXJ instruction.

See the *ARM Architecture Reference Manual* for more information about changing instruction set state.

12.4.3 Cortex-A5 Registers

This view provides 16 ARM core registers, R0 to R15, that include the Stack Pointer (SP), Link Register (LR), and Program Counter (PC). These registers are selected from a total set of either 31 or 33 registers, depending on whether or not the Security Extensions are implemented. The current execution mode determines the selected set of registers, as shown in Table 12-2. This shows that the arrangement of the registers provides duplicate copies of some registers, with the current register selected by the execution mode. This arrangement is described as banking of the registers, and the duplicated copies of registers are referred to as banked registers.

Table 12-2. Cortex-A5 Modes and Registers Layout

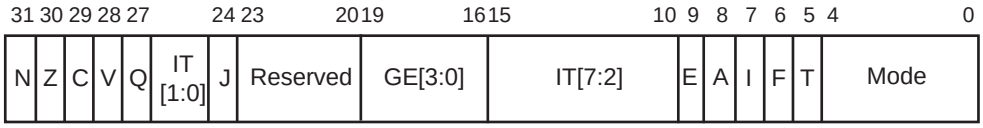
User and System	Monitor	Supervisor	Abort	Undefined	Interrupt	Fast Interrupt
R0	R0	R0	R0	R0	R0	R0
R1	R1	R1	R1	R1	R1	R1
R2	R2	R2	R2	R2	R2	R2
R3	R3	R3	R3	R3	R3	R3
R4	R4	R4	R4	R4	R4	R4
R5	R5	R5	R5	R5	R5	R5
R6	R6	R6	R6	R6	R6	R6
R7	R7	R7	R7	R7	R7	R7
R8	R8	R8	R8	R8	R8	R8_FIQ
R9	R9	R9	R9	R9	R9	R9_FIQ
R10	R10	R10	R10	R10	R10	R10_FIQ
R11	R11	R11	R11	R11	R11	R11_FIQ
R12	R12	R12	R12	R12	R12	R12_FIQ
R13	R13_MON	R13_SVC	R13_ABT	R13_UND	R13_IRQ	R13_FIQ
R14	R14_MON	R14_SVC	R14_ABT	R14_UND	R14_IRQ	R14_FIQ
PC	PC	PC	PC	PC	PC	PC
CPSR	CPSR	CPSR	CPSR	CPSR	CPSR	CPSR
	SPSR_MON	SPSR_SVC	SPSR_ABT	SPSR_UND	SPSR_IRQ	SPSR_FIQ

	Mode-specific banked registers

The core contains one CPSR, and six SPSRs for exception handlers to use. The program status registers:

- hold information about the most recently performed ALU operation
- control the enabling and disabling of interrupts
- set the processor operating mode

Figure 12-2. Status Register Format



- N: Negative, Z: Zero, C: Carry, and V: Overflow are the four ALU flags
- Q: cumulative saturation flag
- IT: If-Then execution state bits for the Thumb IT (If-Then) instruction
- J: Jazelle bit, see the description of the T bit
- GE: Greater than or Equal flags, for SIMD instructions
- E: Endianness execution state bit. Controls the load and store endianness for data accesses. This bit is ignored by instruction fetches.
  - E = 0: Little endian operation
  - E = 1: Big endian operation
- A: Asynchronous abort disable bit. Used to mask asynchronous aborts.
- I: Interrupt disable bit. Used to mask IRQ interrupts.
- F: Fast interrupt disable bit. Used to mask FIQ interrupts.
- T: Thumb execution state bit. This bit and the J execution state bit, bit [24], determine the instruction set state of the processor, ARM, Thumb, Jazelle, or ThumbEE.
- Mode: five bits to encode the current processor mode. The effect of setting M[4:0] to a reserved value is UNPREDICTABLE.

Table 12-3. Processor Mode vs. Mode Field

Mode	M[4:0]
USR	10000
FIQ	10001
IRQ	10010
SVC	10011
MON	10110
ABT	10111
UND	11011
SYS	11111
Reserved	Other

12.4.3.1 CP15 Coprocessor

Coprocessor 15, or System Control Coprocessor CP15, is used to configure and control all the items in the list below:

- Cortex A5
- Caches (ICache, DCache and write buffer)
- MMU
- Security
- Other system options

To control these features, CP15 provides 16 additional registers. See Table 12-4.

Table 12-4. CP15 Registers

Register	Name	Read/Write
0	ID Code(1)	Read/Unpredictable
0	Cache type(1)	Read/Unpredictable
1	Control(1)	Read/Write
1	Security(1)	Read/Write
2	Translation Table Base	Read/Write
3	Domain Access Control	Read/Write
4	Reserved	None
5	Data fault Status(1)	Read/Write
5	Instruction fault status	Read/Write
6	Fault Address	Read/Write
7	Cache and MMU Operations(1)	Read/Write
8	TLB operations	Unpredictable/Write
9	Cache lockdown(1)	Read/Write
10	TLB lockdown	Read/Write
11	Reserved	None
12	Interrupts management	Read/Write
12	Monitor vectors	Read-only
13	FCSE PID(1)	Read/Write
13	Context ID(1)	Read/Write
14	Reserved	None
15	Test configuration	Read/Write

Note: 1. This register provides access to more than one register. The register accessed depends on the value of the CRm field or Opcode_2 field.

12.4.4 CP 15 Register Access

CP15 registers can only be accessed in privileged mode by:

- MCR (Move to Coprocessor from ARM Register) instruction is used to write an ARM register to CP15.
- MRC (Move to ARM Register from Coprocessor) instruction is used to read the value of CP15 to an ARM register.

Other instructions such as CDP, LDC, STC can cause an undefined instruction exception.

The assembler code for these instructions is:

```
MCR/MRC{cond} p15, opcode_1, Rd, CRn, CRm, opcode_2.
```

The MCR/MRC instructions bit pattern is shown below:

31	30	29	28	27	26	25	24
cond				1	1	1	0

23	22	21	20	19	18	17	16
opcode_1			L	CRn			

15	14	13	12	11	10	9	8
Rd				1	1	1	1

7	6	5	4	3	2	1	0
opcode_2			1	CRm			

- **CRm[3:0]: Specified Coprocessor Action**

Determines specific coprocessor action. Its value is dependent on the CP15 register used. For details, refer to CP15 specific register behavior.

- **opcode_2[7:5]**

Determines specific coprocessor operation code. By default, set to 0.

- **Rd[15:12]: ARM Register**

Defines the ARM register whose value is transferred to the coprocessor. If R15 is chosen, the result is unpredictable.

- **CRn[19:16]: Coprocessor Register**

Determines the destination coprocessor register.

- **L: Instruction Bit**

0: MCR instruction

1: MRC instruction

- **opcode_1[23:20]: Coprocessor Code**

Defines the coprocessor specific code. Value is c15 for CP15.

- **cond [31:28]: Condition**

12.4.5 Addresses in the Cortex-A5 processor

The Cortex-A5 processor operates using *virtual addresses* (VAs). The *Memory Management Unit* (MMU) translates these VAs into the *physical addresses* (PAs) used to access the memory system. Translation tables hold the mappings between VAs and PAs.

See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for more information.

When the Cortex-A5 processor is executing in Non-secure state, the processor performs translation table look-ups using the Non-secure versions of the Translation Table Base Registers. In this situation, any VA can only translate into a Non-secure PA. When it is in Secure state, the Cortex-A5 processor performs translation table look-ups using the Secure versions of the Translation Table Base Registers. In this situation, the security state of any VA is determined by the NS bit of the translation table descriptors for that address.

Following is an example of the address manipulation that occurs when the Cortex-A5 processor requests an instruction:

1. The Cortex-A5 processor issues the VA of the instruction as Secure or Non-secure VA accesses according to the state the processor is in.
2. The instruction cache is indexed by the bits of the VA. The MMU performs the translation table look-up in parallel with the cache access. If the processor is in the Secure state it uses the Secure translation tables, otherwise it uses the Non-secure translation tables.
3. If the protection check carried out by the MMU on the VA does not abort and the PA tag is in the instruction cache, the instruction data is returned to the processor.
4. If there is a cache miss, the MMU passes the PA to the AXI bus interface to perform an external access. The external access is always Non-secure when the core is in the Non-secure state. In the Secure state, the external access is Secure or Non-secure according to the NS attribute value in the selected translation table entry. In Secure state, both L1 and L2 translation table walk accesses are marked as Secure, even if the first level descriptor is marked as NS.

12.4.6 Security Extension Overview

The purpose of the Security Extensions is to enable the construction of a secure software environment. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for details of the Security Extensions.

12.4.6.1 System Boot Sequence

CAUTION: The Security Extensions enable the construction of an isolated software environment for more secure execution, depending on a suitable system design around the processor. The technology does not protect the processor from hardware attacks, and care must be taken to be sure that the hardware containing the reset handling code is appropriately secure.

The processor always boots in the privileged Supervisor mode in the Secure state, with the NS bit set to 0. This means that code that does not attempt to use the Security Extensions always runs in the Secure state. If the software uses both Secure and Non-secure states, the less trusted software, such as a complex operating system and application code running under that operating system, executes in Non-secure state, and the most trusted software executes in the Secure state.

The following sequence is expected to be typical use of the Security Extensions:

1. Exit from reset in Secure state.
2. Configure the security state of memory and peripherals. Some memory and peripherals are accessible only to the software running in Secure state.
3. Initialize the secure operating system. The required operations depend on the operating system, and include initialization of caches, MMU, exception vectors, and stacks.

4. Initialize Secure Monitor software to handle exceptions that switch execution between the Secure and Non-secure operating systems.
5. Optionally lock aspects of the secure state environment against further configuration.
6. Pass control through the Secure Monitor software to the non-secure OS with an SMC instruction.
7. Enable the Non-secure operating system to initialize. The required operations depend on the operating system, and typically include initialization of caches, MMU, exception vectors, and stacks.

The overall security of the secure software depends on the system design, and on the secure software itself.

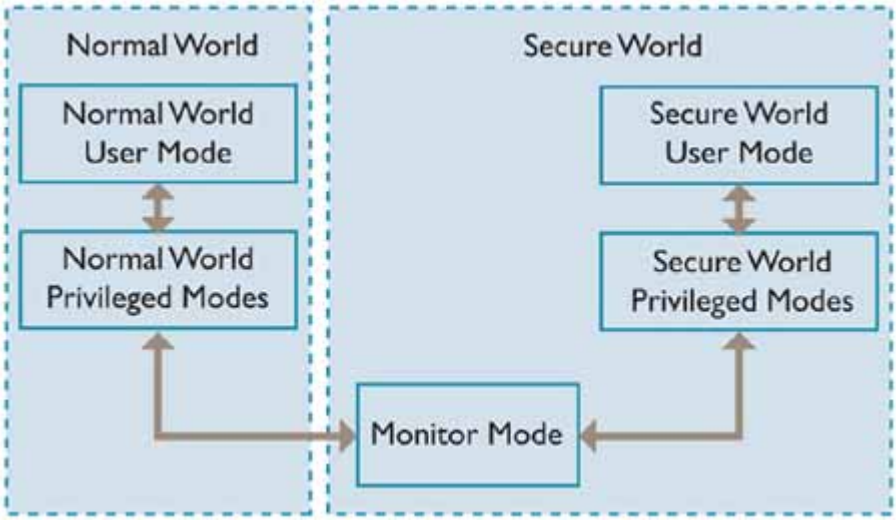
12.4.7 TrustZone

12.4.7.1 Hardware

TrustZone enables a single physical processor core to execute code safely and efficiently from both the Normal world and the Secure world. This removes the need for a dedicated security processor core, saving silicon area and power, and allowing high performance security software to run alongside the Normal world operating environment.

The two virtual processors context switch via a new processor mode called monitor mode when changing the currently running virtual processor.

Figure 12-3. TrustZone Hardware Implementation

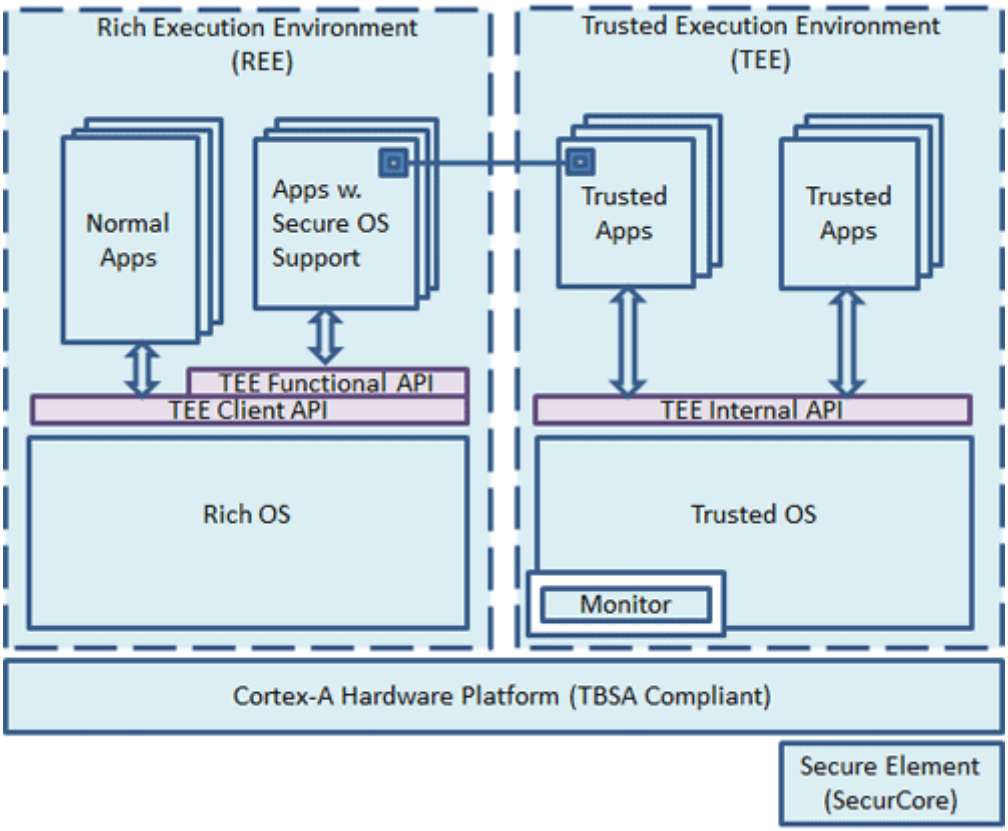


12.4.7.2 Software

The mechanisms by which the physical processor can enter monitor mode from the Normal world are tightly controlled, and are all viewed as exceptions to the monitor mode software. Software executing a dedicated instruction can trigger entry to monitor, the Secure Monitor Call (SMC) instruction, or by a subset of the hardware exception mechanisms. Configuration of the IRQ, FIQ, external Data Abort, and external Prefetch Abort exceptions can cause the processor to switch into monitor mode.

The software that executes within monitor mode is implementation defined, but it generally saves the state of the current world and restores the state of the world at the location to which it switches. It then performs a return-from-exception to restart processing in the restored world.

Figure 12-4. TrustZone Software implementation in a Trusted Execution Environment (TEE)



12.4.7.3 Debug

TrustZone hardware architecture is a security-aware debug infrastructure that can enable control over access to secure world debug, without impairing debug visibility of the Normal world. This is controlled with bits in the Secure Fuse Controller.

Note: Secure debug modes are described in the document “Secure Box Module (SBM)”. This document is available under Non-Disclosure Agreement (NDA). Contact an Atmel Sales Representative for further details.

12.5 Memory Management Unit

12.5.1 About the MMU

The MMU works with the L1 and L2 memory system to translate virtual addresses to physical addresses. It also controls accesses to and from external memory.

The ARM v7 Virtual Memory System Architecture (VMSA) features include the following:

- Page table entries that support:
  - 16 Mbyte supersections. The processor supports supersections that consist of 16 Mbyte blocks of memory.
  - 1 Mbyte sections
  - 64 Kbyte large pages
  - 4 Kbyte small pages
- 16 access domains
- Global and application-specific identifiers to remove the requirement for context switch TLB flushes.
- Extended permissions checking capability.

TLB maintenance and configuration operations are controlled through a dedicated coprocessor, CP15, integrated with the core. This coprocessor provides a standard mechanism for configuring the L1 memory system.

See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for a full architectural description of the ARMv7 VMSA.

12.5.2 Memory Management System

The Cortex-A5 processor supports the ARM v7 VMSA including the TrustZone security extension. The translation of a Virtual Address (VA) used by the instruction set architecture to a Physical Address (PA) used in the memory system and the management of the associated attributes and permissions is carried out using a two-level MMU.

The first level MMU uses a Harvard design with separate micro TLB structures in the PFU for instruction fetches (IuTLB) and in the DPU for data read and write requests (DuTLB).

A miss in the micro TLB results in a request to the main unified TLB shared between the data and instruction sides of the memory system. The TLB consists of a 128-entry two-way set-associative RAM based structure. The TLB page-walk mechanism supports page descriptors held in the L1 data cache. The caching of page descriptors is configured globally for each translation table base register, TTBRx, in the system coprocessor, CP15.

The TLB contains a hitmap cache of the page types which have already been stored in the TLB.

12.5.2.1 Memory types

Although various different memory types can be specified in the page tables, the Cortex-A5 processor does not implement all possible combinations:

- Write-through caches are not supported. Any memory marked as write-through is treated as Non-cacheable.
- The outer shareable attribute is not supported. Anything marked as outer shareable is treated in the same way as inner shareable.
- Write-back no write-allocate is not supported. It is treated as write-back write-allocate.

Table 12-5 shows the treatment of each different memory type in the Cortex-A5 processor in addition to the architectural requirements.

Table 12-5. Treatment of Memory Attributes

Memory Type Attribute	Shareability	Other Attributes	Notes
Strongly Ordered	—	—	—
Device	Non-shareable	—	—
	Shareable	—	—
Normal	Non-shareable	Non-cacheable	Does not access L1 caches
		Write-through cacheable	Treated as non-cacheable
		Write-back cacheable, write allocate	Can dynamically switch to no write allocate, if more than three full cache lines are written in succession
		Write-back cacheable, no write allocate	Treated as non-shareable write-back cacheable, write allocate
	Inner shareable	Non-cacheable	—
		Write-through cacheable	Treated as inner shareable non-cacheable
		Write-back cacheable, write allocate	Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate.
		Write-back cacheable, no write allocate	
	Outer shareable	Non-cacheable	Treated as inner shareable non-cacheable
		Write-through cacheable	
		Write-back cacheable, write allocate	Treated as inner shareable non-cacheable unless the SMP bit in the Auxiliary Control Register is set (ACTLR[6] = b1). If this bit is set the area is treated as write-back cacheable write allocate.
		Write-back cacheable, no write allocate	

12.5.3 TLB Organization

TLB Organization is described in the sections that follow:

- Micro TLB
- Main TLB

12.5.3.1 Micro TLB

The first level of caching for the page table information is a micro TLB of 10 entries that is implemented on each of the instruction and data sides. These blocks provide a lookup of the virtual addresses in a single cycle.

The micro TLB returns the physical address to the cache for the address comparison, and also checks the access permissions to signal either a Prefetch Abort or a Data Abort.

All main TLB related maintenance operations affect both the instruction and data micro TLBs, causing them to be flushed. In the same way, any change of the following registers causes the micro TLBs to be flushed:

- Context ID Register (CONTEXTIDR)
- Domain Access Control Register (DACR)
- Primary Region Remap Register (PRRR)
- Normal Memory Remap Register (NMRR)
- Translation Table Base Registers (TTBR0 and TTBR1)

12.5.3.2 Main TLB

Misses from the instruction and data micro TLBs are handled by a unified main TLB. Accesses to the main TLB take a variable number of cycles, according to competing requests from each of the micro TLBs and other implementation-dependent factors.

The main TLB is 128-entry two-way set-associative.

TLB match process

Each TLB entry contains a virtual address, a page size, a physical address, and a set of memory properties. Each is marked as being associated with a particular application space (ASID), or as global for all application spaces. The CONTEXTIDR determines the currently selected application space.

A TLB entry matches when these conditions are true:

- Its virtual address matches that of the requested address.
- Its Non-secure TLB ID (NSTID) matches the Secure or Non-secure state of the MMU request.
- Its ASID matches the current ASID in the CONTEXTIDR or is global.

The operating system must ensure that, at most, one TLB entry matches at any time. The TLB can store entries based on the following block sizes:

Supersections	Describe 16 Mbyte blocks of memory
Sections	Describe 1 Mbyte blocks of memory
Large pages	Describe 64 Kbyte blocks of memory
Small pages	Describe 4 Kbyte blocks of memory

Supersections, sections and large pages are supported to permit mapping of a large region of memory while using only a single entry in the TLB. If no mapping for an address is found within the TLB, then the translation table is automatically read by hardware and a mapping is placed in the TLB.

12.5.4 Memory Access Sequence

When the processor generates a memory access, the MMU:

1. Performs a lookup for the requested virtual address and current ASID and security state in the relevant instruction or data micro TLB.
2. If there is a miss in the micro TLB, performs a lookup for the requested virtual address and current ASID and security state in the main TLB.
3. If there is a miss in main TLB, performs a hardware translation table walk.

The MMU can be configured to perform hardware translation table walks in cacheable regions by setting the IRGN bits in Translation Table Base Register 0 and Translation Table Base Register 1. If the encoding of the IRGN bits is write-back, an L1 data cache lookup is performed and data is read from the data cache. If the encoding of the IRGN bits is write-through or non-cacheable, an access to external memory is performed. For more information refer to: *Cortex-A5 Technical Reference Manual*.

The MMU might not find a global mapping, or a mapping for the currently selected ASID, with a matching Non-secure TLB ID (NSTID) for the virtual address in the TLB. In this case, the hardware does a translation table walk if the translation table walk is enabled by the PD0 or PD1 bit in the Translation Table Base Control Register. If translation table walks are disabled, the processor returns a Section Translation fault. For more information refer to: *Cortex-A5 Technical Reference Manual*.

If the TLB finds a matching entry, it uses the information in the entry as follows:

1. The access permission bits and the domain determine if the access is enabled. If the matching entry does not pass the permission checks, the MMU signals a memory abort. See the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition* for a description of access permission bits, abort types and priorities, and for a description of the Instruction Fault Status Register (IFSR) and Data Fault Status Register (DFSR).
2. The memory region attributes specified in both the TLB entry and the CP15 c10 remap registers determine if the access is
   - Secure or Non-secure
   - Shared or not
   - Normal memory, Device, or Strongly-ordered

For more information refer to: *Cortex-A5 Technical Reference Manual*, Memory region remap.

3. The TLB translates the virtual address to a physical address for the memory access.

12.5.5 Interaction with Memory System

The MMU can be enabled or disabled as described in the *ARM Architecture Reference Manual, ARMv7-A and ARMv7-R edition*.

12.5.6 External Aborts

External memory errors are defined as those that occur in the memory system rather than those that are detected by the MMU. External memory errors are expected to be extremely rare. External aborts are caused by errors flagged by the AXI interfaces when the request goes external to the Cortex-A5 processor. External aborts can be configured to trap to Monitor mode by setting the EA bit in the Secure Configuration Register. For more information refer to: *Cortex-A5 Technical Reference Manual*.

12.5.6.1 External Aborts on Data Write

Externally generated errors during a data write can be asynchronous. This means that the r14_abt on entry into the abort handler on such an abort might not hold the address of the instruction that caused the exception.

The DFAR is Unpredictable when an asynchronous abort occurs.

Externally generated errors during data read are always synchronous. The address captured in the DFAR matches the address which generated the external abort.

12.5.6.2 Synchronous and Asynchronous Aborts

Chapter 4, System Control in the *Cortex-A5 Technical Reference Manual* describes synchronous and asynchronous aborts, their priorities, and the IFSR and DFSR. To determine a fault type, read the DFSR for a data abort or the IFSR for an instruction abort.

The processor supports an Auxiliary Fault Status Register for software compatibility reasons only. The processor does not modify this register because of any generated abort.

12.5.7 MMU Software Accessible Registers

The system control coprocessor registers, CP15, in conjunction with page table descriptors stored in memory, control the MMU.

Access all the registers with instructions of the form:

MRC p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>

MCR p15, 0, <Rd>, <CRn>, <CRm>, <Opcode_2>

CRn is the system control coprocessor register. Unless specified otherwise, CRm and Opcode_2 Should Be Zero.

13. L2 Cache Controller (L2CC)

13.1 Description

The L2 Cache Controller (L2CC) is based on the L2CC-PL310 ARM multiway cache macrocell, version r3p2. The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a method of improving the system performance when significant memory traffic is generated by the processor.

13.2 Embedded Characteristics

- 8-way set associative cache architecture
- Data banking is not supported
- No parity bit embedded
- Lockdown by master is not supported
- Lockdown by line is not supported
- TrustZone architecture for enhanced OS security

13.3 Product Dependencies

13.3.1 Power Management

The L2 Cache Controller is continuously clocked by the Processor Clock. The Power Management Controller has no effect on the behavior of the L2 Cache Controller.

13.4 Functional Description

The addition of an on-chip secondary cache, also referred to as a Level 2 or L2 cache, is a recognized method of improving the performance of ARM-based systems when significant memory traffic is generated by the processor. By definition a secondary cache assumes the presence of a Level 1 or primary cache, closely coupled or internal to the processor. Memory access is fastest to L1 cache, followed closely by L2 cache. Memory access is typically significantly slower with L3 main memory.

The cache controller is a unified, physically addressed, physically tagged cache with up to 8 ways. The user can lock the replacement algorithm on a way basis, enabling the associativity to be reduced from 8-way down to 1-way (directly mapped).

The cache controller does not have snooping hardware to maintain coherency between caches, so the user has to maintain coherency by software.

13.4.1 Double Linefill Issuing

The L2CC cache line length is 32-byte. Therefore, by default, on each L2 cache miss,

L2CC issues 32-byte linefills, 4 x 64-bit read bursts, to the L3 memory system. L2CC can issue 64-byte linefills, 8 x 64-bit read bursts, on an L2 cache miss. When the L2CC is waiting for the data from L3, it performs a lookup on the second cache line targeted by the 64-byte linefill. If it misses, data corresponding to the second cache line are allocated to the L2 cache. If it hits, data corresponding to the second cache line are discarded.

The user can control this feature using the DLEN, DLFWRDIS and DLEN bits of the L2CC Prefetch Control Register. The IDLEN and DLFWRDIS bits are only used if you set the DLEN bit HIGH. Table 13-1 shows the behavior of the L2CC master ports, depending on the configuration chosen by the user.

Table 13-1. L2CC Master Port Behavior

Bit 30 DLEN	Bit 27 DLFWRDIS	Bit 23 IDLEN	Original Read Address from L1	Read Address to L3	AXI Burst Type	AXI Burst Length	Targeted Cache Lines
0	0 or 1	0 or 1	0x00	0x00	WRAP	0x3, 4x64-bit	0x00
0	0 or 1	0 or 1	0x20	0x20	WRAP	0x3, 4x64-bit	0x20
1	0 or 1	0	0x00	0x00	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	1	0	0x08 or 0x10 or 0x18	0x08	WRAP	0x3, 4x64-bit	0x00
1	0	0	0x08 or 0x10 or 0x18	0x00	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	0 or 1	0	0x20	0x20	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	1	0	0x28 or 0x30 or 0x38	0x28	WRAP	0x3, 4x64-bit	0x20
1	0	0	0x28 or 0x30 or 0x38	0x20	WRAP	0x7, 8x64-bit	0x00 and 0x20
1	0 or 1	1	0x00	0x00	INCR or WRAP	0x7, 8x64-bit	0x00 and 0x20
1	1	1	0x08 or 0x10 or 0x18	0x08	WRAP	0x3, 4x64-bit	0x00
1	0	1	0x08 or 0x10 or 0x18	0x00	INCR or WRAP	0x7, 8x64-bit	0x00 and 0x20
1	0 or 1	1	0x20	0x20	INCR	0x7, 8x64-bit	0x20 and 0x40
1	1	1	0x28 or 0x30 or 0x38	0x28	WRAP	0x3, 4x64-bit	0x20
1	0	1	0x28 or 0x30 or 0x38	0x20	INCR	0x7, 8x64-bit	0x20 and 0x40

Notes:
1. Double linefills are not issued for prefetch reads if you enable exclusive cache configuration.
2. Double linefills are not launched when crossing a 4-Kbyte boundary.
3. Double linefills only occur if a WRAP4 or an INCR4 64-bit transaction is received on the slave ports. This transaction is most commonly seen as a result of a cache linefill in a master, but can be produced by a master when accessing memory marked as inner non-cacheable.

13.5 L2 Cache Controller (L2CC) User Interface

Table 13-2. Register Mapping

Offset	Register	Name	Access	Reset
0x000	Cache ID Register	L2CC_IDR	Read-only	0x4100_00C9
0x004	Cache Type Register	L2CC_TYPR	Read-only	0x0010_0100
0x100	Control Register	L2CC_CR	Read/Write, Read-only(1)	0x0000_0000
0x104	Auxiliary Control Register	L2CC_ACR	Read/Write, Read-only(1)	0x0202_0000
0x108	Tag RAM Control Register	L2CC_TRCR	Read/Write, Read-only(1)	0x0000_0111
0x10C	Data RAM Control Register	L2CC_DRCR	Read/Write, Read-only(1)	0x0000_0111
0x110–0x1FC	Reserved	–	–	–
0x200	Event Counter Control Register	L2CC_ECR	Read/Write	0x0000_0000
0x204	Event Counter 1 Configuration Register	L2CC_ECFGR1	Read/Write	0x0202_0000
0x208	Event Counter 0 Configuration Register	L2CC_ECFGR0	Read/Write	0x0000_0000
0x20C	Event Counter 1 Value Register	L2CC_EVR1	Read/Write	0x0000_0000
0x210	Event Counter 0 Value Register	L2CC_EVR0	Read/Write	0x0000_0000
0x214	Interrupt Mask Register	L2CC_IMR	Programmable(2)	0x0000_0000
0x218	Masked Interrupt Status Register	L2CC_MISR	Read-only	0x0000_0000
0x21C	Raw Interrupt Status Register	L2CC_RISR	Read-only	0x0000_0000
0x220	Interrupt Clear Register	L2CC_ICR	Programmable(2)	0x0000_0000
0x224–0x72C	Reserved	–	–	–
0x730	Cache Synchronization Register	L2CC_CSR	Read/Write	0x0000_0000
0x734–0x76C	Reserved	–	–	–
0x770	Invalidate Physical Address Line Register	L2CC_IPALR	Read/Write	0x0000_0000
0x774–0x778	Reserved	–	–	–
0x77C	Invalidate Way Register	L2CC_IWR	Read/Write	0x0000_0000
0x780–0x7AF	Reserved	–	–	–
0x7B0	Clean Physical Address Line Register	L2CC_CPALR	Read/Write	0x0000_0000
0x7B4	Reserved	–	–	–
0x7B8	Clean Index Register	L2CC_CIR	Read/Write	0x0000_0000
0x7BC	Clean Way Register	L2CC_CWR	Read/Write	0x0000_0000
0x7C0–0x7EC	Reserved	–	–	–
0x7F0	Clean Invalidate Physical Address Line Register	L2CC_CIPALR	Read/Write	0x0000_0000
0x7F4	Reserved	–	–	–
0x7F8	Clean Invalidate Index Register	L2CC_CIIR	Read/Write	0x0000_0000
0x7FC	Clean Invalidate Way Register	L2CC_CIWR	Read/Write	0x0000_0000
0x800–0x8FC	Reserved	–	–	–
0x900	Data Lockdown Register	L2CC_DLKR	Programmable(2)	0x0000_0000
0x904	Instruction Lockdown Register	L2CC_ILKR	Programmable(2)	0x0000_0000

Table 13-2. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x908–0xF3C	Reserved	–	–	–
0xF40	Debug Control Register	L2CC_DCR	Read/Write, Read-only(1)	0x0000_0000
0xF44–0xF5C	Reserved	–	–	–
0xF60	Prefetch Control Register	L2CC_PCR	Read/Write, Read-only(1)	0x0000_0000
0xF64–0xF7C	Reserved	–	–	–
0xF80	Power Control Register	L2CC_POWCR	Read/Write, Read-only(1)	0x0000_0000

Notes: 1. Read/Write in Secure mode, Read-only in Non-secure mode.
2. Programmable in Auxiliary Control Register.

13.5.1 L2CC Cache ID Register

Name: L2CC_IDR

Address: 0x00A00000

Access: Read-only

31	30	29	28	27	26	25	24
ID							

23	22	21	20	19	18	17	16
ID							

15	14	13	12	11	10	9	8
ID							

7	6	5	4	3	2	1	0
ID							

- **ID: Cache Controller ID**

The cache ID is 0x410000C9.

13.5.2 L2CC Type Register

Name: L2CC_TYPR

Address: 0x00A00004

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	DL2WSIZE			–	DL2ASS	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	IL2WSIZE		

7	6	5	4	3	2	1	0
–	IL2ASS	–	–	–	–	–	–

- **IL2ASS: Instruction L2 Cache Associativity**

The value is read from the field ASS in Auxiliary Control Register, should be 0.

- **IL2WSIZE: Instruction L2 Cache Way Size**

The value is read from the field WAYSIZE in Auxiliary Control Register, should be 0x1.

- **DL2ASS: Data L2 Cache Associativity**

The value is read from the field ASS in Auxiliary Control Register, should be 0.

- **DL2WSIZE: Data L2 Cache Way Size**

The value is read from the field WAYSIZE in Auxiliary Control Register, should be 0x1.

13.5.3 L2CC Control Register

Name: L2CC_CR

Address: 0x00A00100

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	L2CEN

- **L2CEN: L2 Cache Enable**

0: L2 Cache is disabled. This is the default value.

1: L2 Cache is enabled.

13.5.4 L2CC Auxiliary Control Register

Name: L2CC_ACR

Address: 0x00A00104

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	IPEN	DPEN	NSIAC	NSLEN	CRPOL	FWA
23	22	21	20	19	18	17	16
FWA	SAOEN	PEN	EMBEN	WAYSIZE			ASS
15	14	13	12	11	10	9	8
–	–	SAIE	EXCC	SBDLE	HPSO	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

Note: The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

- **HPSO: High Priority for SO and Dev Reads Enable**

0: Strongly Ordered and Device reads have lower priority than cacheable accesses when arbitrated in the L2CC master ports. This is the default value.

1: Strongly Ordered and Device reads get the highest priority when arbitrated in the L2CC master ports.

- **SBDLE: Store Buffer Device Limitation Enable**

0: Store buffer device limitation is disabled. Device writes can take all slots in the store buffer. This is the default value.

1: Store buffer device limitation is enabled.

- **EXCC: Exclusive Cache Configuration**

0: Disabled. This is the default value.

1: Enabled.

- **SAIE: Shared Attribute Invalidate Enable**

0: Shared invalidate behavior is disabled. This is the default value.

1: Shared invalidate behavior is enabled if the Shared Attribute Override Enable bit is not set.

Shared invalidate behavior is enabled if both:

- Shareable Attribute Invalidate Enable bit is set in the Auxiliary Control Register, bit[13]
- Shared Attribute Override Enable bit is not set in the Auxiliary Control Register, bit[22]

- **ASS: Associativity**

0: 8-way.This is the default value.

1: Reserved.

- **WAYSIZE: Way Size**

Value	Name	Description
0x0	RESERVED	Reserved
0x1	16KB_WAY	16-Kbyte way set associative
0x2	RESERVED	Reserved
0x3	RESERVED	Reserved
0x4	RESERVED	Reserved
0x5	RESERVED	Reserved
0x6	RESERVED	Reserved
0x7	RESERVED	Reserved

- **EMBEN: Event Monitor Bus Enable**

0: Disabled. This is the default value.

1: Enabled.

- **PEN: Parity Enable**

0: Disabled. This is the default value.

1: Enabled.

- **SAOEN: Shared Attribute Override Enable**

0: Treats shared accesses. This is the default value.

1: Shared attribute is internally ignored.

- **FWA: Force Write Allocate**

0: The L2 Cache controller uses AWCACHE attributes for WA. This is the default value.

1: User forces no allocate, WA bit must be set to 0.

2: User overrides AWCACHE attributes, WA bit must be set to 1. All cacheable write misses become write allocated.

3: The write allocation is internally mapped to 00.

- **CRPOL: Cache Replacement Policy**

0: Pseudo-random replacement using the LFSR algorithm.

1: Round-robin replacement. This is always the default value.

- **NSLEN: Non-Secure Lockdown Enable**

0: Lockdown registers cannot be modified using non-secure accesses. This is the default value.

1: Non-secure accesses can write to the lockdown registers.

- **NSIAC: Non-Secure Interrupt Access Control**

0: Interrupt Clear Register and Interrupt Mask Register can only be modified or read with secure accesses. This is the default value.

1: Interrupt Clear Register and Interrupt Mask Register can be modified or read with secure or non-secure accesses.

- **DPEN: Data Prefetch Enable**

0: Data prefetching is disabled. This is the default value.

1: Data prefetching is enabled.

- **IPEN: Instruction Prefetch Enable**

0: Instruction prefetching is disabled. This is the default value.

1: Instruction prefetching is enabled.

13.5.5 L2CC Tag RAM Latency Control Register

Name: L2CC_TRCR

Address: 0x00A00108

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	TWRLAT		
7	6	5	4	3	2	1	0
–	TRDLAT			–	TSETLAT		

Note: The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

- **TSETLAT: Setup Latency**
- **TRDLAT: Read Access Latency**
- **TWRLAT: Write Access Latency**

Latency to Tag RAM is the programmed value + 1.

Default value is 0.

13.5.6 L2CC Data RAM Latency Control Register

Name: L2CC_DRCR

Address: 0x00A0010C

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	DWRLAT		
7	6	5	4	3	2	1	0
–	DRDLAT			–	DSETLAT		

Note: The L2 Cache Controller (L2CC) must be disabled in the L2CC Control Register prior to any write access to this register.

- **DSETLAT: Setup Latency**
- **DRDLAT: Read Access Latency**
- **DWRLAT: Write Access Latency**

Latency to Data RAM is the programmed value + 1.

Default value is 0.

13.5.7 L2CC Event Counter Control Register

Name: L2CC_ECR

Address: 0x00A00200

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	EVC1RST	EVC0RST	EVCEN

- **EVCEN: Event Counter Enable**

0: Disables Event Counter. This is the default value.

1: Enables Event Counter.

- **EVC0RST: Event Counter 0 Reset**

0: No effect, always read as zero.

1: Resets Counter 0.

- **EVC1RST: Event Counter 1 Reset**

0: No effect, always read as zero.

1: Resets Counter 1.

13.5.8 L2CC Event Counter 1 Configuration Register

Name: L2CC_ECFGR1

Address: 0x00A00204

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	ESRC				EIGEN	

- **EIGEN: Event Counter Interrupt Generation**

Value	Name	Description
0x0	INT_DIS	Disables (default)
0x1	INT_EN_INCR	Enables with Increment condition
0x2	INT_EN_OVER	Enables with Overflow condition
0x3	INT_GEN_DIS	Disables Interrupt generation

- **ESRC: Event Counter Source**

Value	Name	Description
0x0	CNT_DIS	Counter Disabled
0x1	SRC_CO	Source is CO
0x2	SRC_DRHIT	Source is DRHIT
0x3	SRC_DRREQ	Source is DRREQ
0x4	SRC_DWHIT	Source is DWHIT
0x5	SRC_DWREQ	Source is DWREQ
0x6	SRC_DWTREQ	Source is DWTREQ
0x7	SRC_IRHIT	Source is IRHIT
0x8	SRC_IRREQ	Source is IRREQ
0x9	SRC_WA	Source is WA
0xa	SRC_IPFALLOC	Source is IPFALLOC
0xb	SRC_EPFHIT	Source is EPFHIT
0xc	SRC_EPFALLOC	Source is EPFALLOC
0xd	SRC_SRRCVD	Source is SRRCVD
0xe	SRC_SRCONF	Source is SRCONF
0xf	SRC_EPFRCVD	Source is EPFRCVD

13.5.9 L2CC Event Counter 0 Configuration Register

Name: L2CC_ECFGR0

Address: 0x00A00208

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	ESRC				EIGEN	

- **EIGEN: Event Counter Interrupt Generation**

Value	Name	Description
0x0	INT_DIS	Disables (default)
0x1	INT_EN_INCR	Enables with Increment condition
0x2	INT_EN_OVER	Enables with Overflow condition
0x3	INT_GEN_DIS	Disables Interrupt generation

- **ESRC: Event Counter Source**

Value	Name	Description
0x0	CNT_DIS	Counter Disabled
0x1	SRC_CO	Source is CO
0x2	SRC_DRHIT	Source is DRHIT
0x3	SRC_DRREQ	Source is DRREQ
0x4	SRC_DWHIT	Source is DWHIT
0x5	SRC_DWREQ	Source is DWREQ
0x6	SRC_DWTREQ	Source is DWTREQ
0x7	SRC_IRHIT	Source is IRHIT
0x8	SRC_IRREQ	Source is IRREQ
0x9	SRC_WA	Source is WA
0xa	SRC_IPFALLOC	Source is IPFALLOC
0xb	SRC_EPFHIT	Source is EPFHIT
0xc	SRC_EPFALLOC	Source is EPFALLOC
0xd	SRC_SRRCVD	Source is SRRCVD
0xe	SRC_SRCONF	Source is SRCONF
0xf	SRC_EPFRCVD	Source is EPFRCVD

13.5.10 L2CC Event Counter 1 Value Register

Name: L2CC_EVR1

Address: 0x00A0020C

Access: Read/Write

31	30	29	28	27	26	25	24
VALUE							

23	22	21	20	19	18	17	16
VALUE							

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
VALUE							

Note: Counter 1 must be disabled in the L2CC Event Counter 1 Configuration Register prior to any write access to this register.

- **VALUE: Event Counter Value**

Value returns the number of instance of the selected event.

If a counter reaches its maximum value, it remains saturated at that value until it is reset.

13.5.11 L2CC Event Counter 0 Value Register

Name: L2CC_EVR0

Address: 0x00A00210

Access: Read/Write

31	30	29	28	27	26	25	24
VALUE							

23	22	21	20	19	18	17	16
VALUE							

15	14	13	12	11	10	9	8
VALUE							

7	6	5	4	3	2	1	0
VALUE							

Note: Counter 0 must be disabled in the L2CC Event Counter 0 Configuration Register prior to any write access to this register.

- **VALUE: Event Counter Value**

Value returns the number of instance of the selected event.

If a counter reaches its maximum value, it remains saturated at that value until it is reset.

13.5.12 L2CC Interrupt Mask Register

Name: L2CC_IMR

Address: 0x00A00214

Access: Programmable in Auxiliary Control Register

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 Memory**
- **DECERR: DECERR from L3 Memory**

0: Masked. This is the default value.

1: Enabled.

13.5.13 L2CC Masked Interrupt Status Register

Name: L2CC_MISR

Address: 0x00A00218

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 memory**
- **DECERR: DECERR from L3 memory**

0: No interrupt has been generated or the interrupt is masked.

1: The input lines have triggered an interrupt.

13.5.14 L2CC Raw Interrupt Status Register

Name: L2CC_RISR

Address: 0x00A0021C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 memory**
- **DECERR: DECERR from L3 memory**

0: No interrupt has been generated.

1: The input lines have triggered an interrupt.

13.5.15 L2CC Interrupt Clear Register

Name: L2CC_ICR

Address: 0x00A00220

Access: Programmable in Auxiliary Control Register

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DECERR

7	6	5	4	3	2	1	0
SLVERR	ERRRD	ERRRT	ERRWD	ERRWT	PARRD	PARRT	ECNTR

- **ECNTR: Event Counter 1/0 Overflow Increment**
- **PARRT: Parity Error on L2 Tag RAM, Read**
- **PARRD: Parity Error on L2 Data RAM, Read**
- **ERRWT: Error on L2 Tag RAM, Write**
- **ERRWD: Error on L2 Data RAM, Write**
- **ERRRT: Error on L2 Tag RAM, Read**
- **ERRRD: Error on L2 Data RAM, Read**
- **SLVERR: SLVERR from L3 memory**
- **DECERR: DECERR from L3 memory**

0: No effect. Read returns zero.

1: Clears the corresponding bit in the Raw Interrupt Status Register.

13.5.16 L2CC Cache Synchronization Register

Name: L2CC_CSR

Address: 0x00A00730

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

13.5.17 L2CC Invalidate Physical Address Line Register

Name: L2CC_IPALR

Address: 0x00A00770

Access: Read/Write

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG		IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index Number**
- **TAG: Tag Number**

13.5.18 L2CC Invalidate Way Register

Name: L2CC_IWR

Address: 0x00A0077C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
WAY7	WAY6	WAY5	WAY4	WAY3	WAY2	WAY1	WAY0

- **WAYx: Invalidate Way Number x**

0: The corresponding way is totally invalidated.

1: Invalidates the way. This bit is read as ‘1’ as long as invalidation of the way is in progress.

13.5.19 L2CC Clean Physical Address Line Register

Name: L2CC_CPALR

Address: 0x00A007B0

Access: Read/Write

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG		IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index number**
- **TAG: Tag number**

13.5.20 L2CC Clean Index Register

Name: L2CC_CIR

Address: 0x00A007B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	WAY			–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index number**
- **WAY: Way number**

13.5.21 L2CC Clean Way Register

Name: L2CC_CWR

Address: 0x00A007BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
WAY7	WAY6	WAY5	WAY4	WAY3	WAY2	WAY1	WAY0

- **WAYx: Clean Way Number x**

0: The corresponding way is totally cleaned.

1: Cleans the way. This bit is read as ‘1’ as long as cleaning of the way is in progress.

13.5.22 L2CC Clean Invalidate Physical Address Line Register

Name: L2CC_CIPALR

Address: 0x00A007F0

Access: Read/Write

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG		IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index Number**
- **TAG: Tag Number**

13.5.23 L2CC Clean Invalidate Index Register

Name: L2CC_CIIR

Address: 0x00A007F8

Access: Read/Write

31	30	29	28	27	26	25	24
–	WAY			–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	IDX					

7	6	5	4	3	2	1	0
IDX			–	–	–	–	C

- **C: Cache Synchronization Status**

0: No background operation is in progress. When written, must be zero.

1: A background operation is in progress.

- **IDX: Index Number**
- **WAY: Way Number**

13.5.24 L2CC Clean Invalidate Way Register

Name: L2CC_CIWR

Address: 0x00A007FC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
WAY7	WAY6	WAY5	WAY4	WAY3	WAY2	WAY1	WAY0

- **WAYx: Clean Invalidate Way Number x**

0: The corresponding way is totally invalidated and cleaned.

1: Invalidates and cleans the way. This bit is read as ‘1’ as long as invalidation and cleaning of the way is in progress.

13.5.25 L2CC Data Lockdown Register

Name: L2CC_DLKR

Address: 0x00A00900

Access: Programmable in Auxiliary Control Register

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
DLK7	DLK6	DLK5	DLK4	DLK3	DLK2	DLK1	DLK0

- **DLKx: Data Lockdown in Way Number x**

0: Allocation can occur in the corresponding way.

1: There is no allocation in the corresponding way.

13.5.26 L2CC Instruction Lockdown Register

Name: L2CC_ILKR

Address: 0x00A00904

Access: Programmable in Auxiliary Control Register

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ILK7	ILK6	ILK5	ILK4	ILK3	ILK2	ILK1	ILK0

- **ILKx: Instruction Lockdown in Way Number x**

0: Allocation can occur in the corresponding way.

1: There is no allocation in the corresponding way.

13.5.27 L2CC Debug Control Register

Name: L2CC_DCR

Address: 0x00A00F40

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SPNIDEN	DWB	DCL

- **DCL: Disable Cache Linefill**

0: Enables cache linefills. This is the default value.

1: Disables cache linefills.

- **DWB: Disable Write-back, Force Write-through**

0: Enables write-back behavior. This is the default value.

1: Forces write-through behavior.

- **SPNIDEN: SPNIDEN Value**

Reads value of the SPNIDEN input.

13.5.28 L2CC Prefetch Control Register

Name: L2CC_PCR

Address: 0x00A00F60

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	DLEN	INSPEN	DATPEN	DLFWRDIS	–	–	PDEN
23	22	21	20	19	18	17	16
IDLEN	–	NSIDEN	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	OFFSET				

- **OFFSET: Prefetch Offset**

You must only use the Prefetch offset values of 0-7, 15, 23, and 31 for these bits. The L2CC does not support the other values.

- **NSIDEN: Not Same ID on Exclusive Sequence Enable**

0: Read and write portions of a non-cacheable exclusive sequence have the same AXI ID when issued to L3. This is the default value.

1: Read and write portions of a non-cacheable exclusive sequence do not have the same AXI ID when issued to L3.

- **IDLEN: INCR Double Linefill Enable**

0: The L2CC does not issue INCR 8x64-bit read bursts to L3 on reads that miss in the L2 cache. This is the default value.

1: The L2CC can issue INCR 8x64-bit read bursts to L3 on reads that miss in the L2 cache.

Note: This bit can only be used if the DLEN bit is set HIGH. See Section 13.4.1 “Double Linefill Issuing” for details on double linefill functionality.

- **PDEN: Prefetch Drop Enable**

0: The L2CC does not discard prefetch reads issued to L3. This is the default value.

1: The L2CC discards prefetch reads issued to L3 when there is a resource conflict with explicit reads.

- **DLFWRDIS: Double Linefill on WRAP Read Disable**

0: Double linefill on WRAP read is enabled. This is the default value.

1: Double linefill on WRAP read is disabled.

Note: This bit can only be used if the DLEN bit is set HIGH. See Section 13.4.1 “Double Linefill Issuing” for details on double linefill functionality.

- **DATPEN: Data Prefetch Enable**

0: Data prefetching is disabled. This is the default value.

1: Data prefetching is enabled.

- **INSPEN: Instruction Prefetch Enable**

0: Instruction prefetching is disabled. This is the default value.

1: Instruction prefetching is enabled.

- **DLEN: Double Linefill Enable**

0: The L2CC always issues 4x64-bit read bursts to L3 on reads that miss in the L2 cache. This is the default value.

1: The L2CC issues 8x64-bit read bursts to L3 on reads that miss in the L2 cache.

See Section 13.4.1 “Double Linefill Issuing” for details on double linefill functionality.

13.5.29 L2CC Power Control Register

Name: L2CC_POWCR

Address: 0x00A00F80

Access: Read/Write in Secure mode

Read-only in Non-secure mode

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	DCKGATEN	STBYEN

- **STBYEN: Standby Mode Enable**

0: Disabled. This is the default value.

1: Enabled.

- **DCKGATEN: Dynamic Clock Gating Enable**

0: Disabled. This is the default value.

1: Enabled.

14. Debug and Test Features

14.1 Description

The device features a number of complementary debug and test capabilities.

A common JTAG/ICE (In-Circuit Emulator) port is used for standard debugging functions, such as downloading code and single-stepping through programs.

A 2-pin debug port Serial Wire Debug (SWD) replaces the 5-pin JTAG port and provides an easy and risk-free alternative to JTAG as the two signals, SWDIO and SWCLK, are overlaid on the TMS and TCK pins, allowing for bi-modal devices that provide the other JTAG signals. These extra JTAG pins can be switched to other uses when in SWD mode.

A set of dedicated debug and test input/output pins gives direct access to these capabilities from a PC-based test environment.

14.2 Embedded Characteristics

- Cortex-A5 In-circuit Emulator
  - Two Real-time Watchpoint Units
  - Two Independent Registers: Debug Control Register and Debug Status Register
  - Test Access Port Accessible through JTAG Protocol
  - Debug Communications Channel
  - Serial Wire Debug
  - Trace
- Chip ID Register
- IEEE1149.1 JTAG Boundary-scan on All Digital Pins
- ETM, ETB: 8-Kbyte Embedded Trace Buffer

14.3 Debug and Test Block Diagrams

Figure 14-1. Debug and Test General Block Diagram

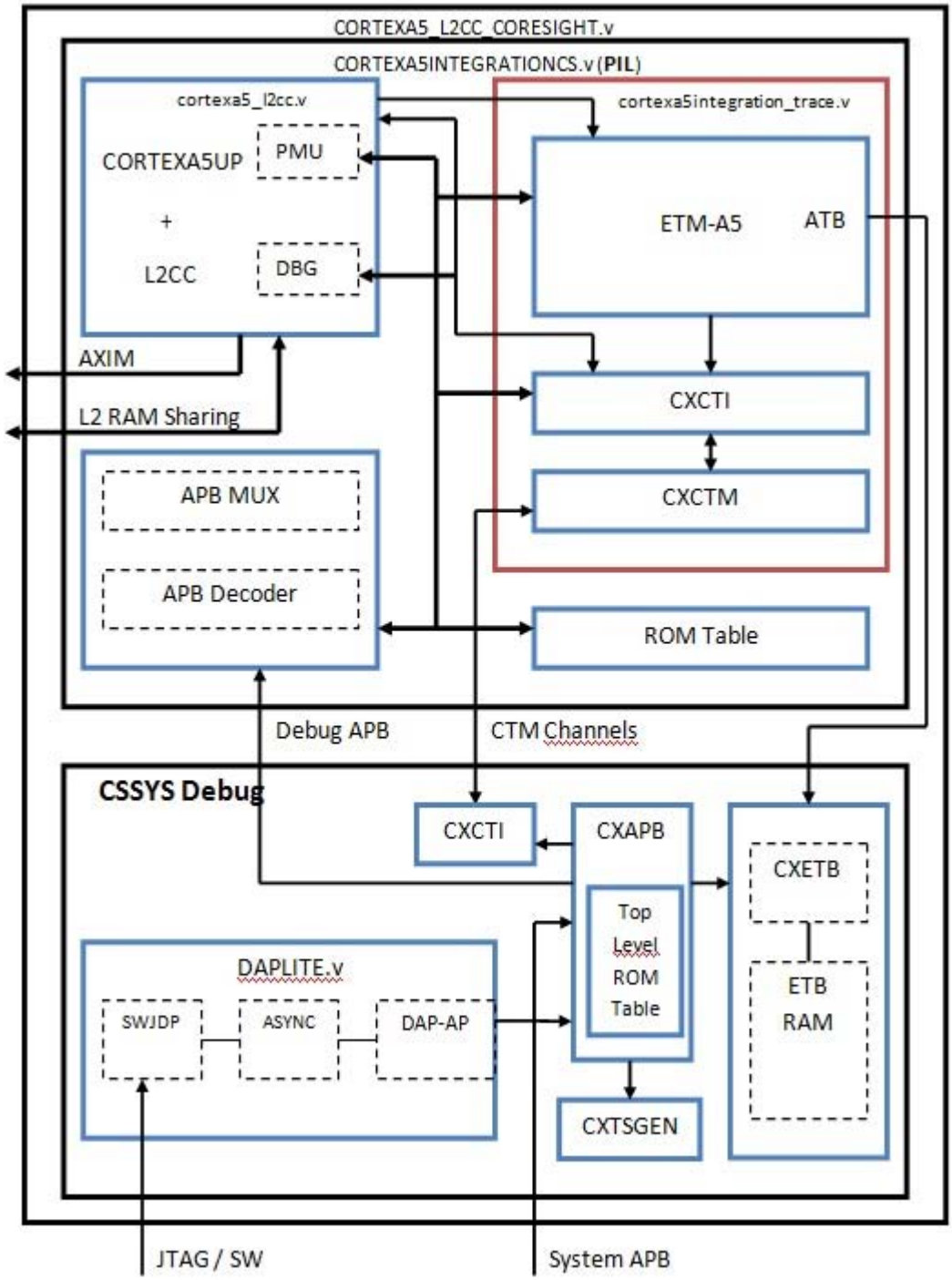
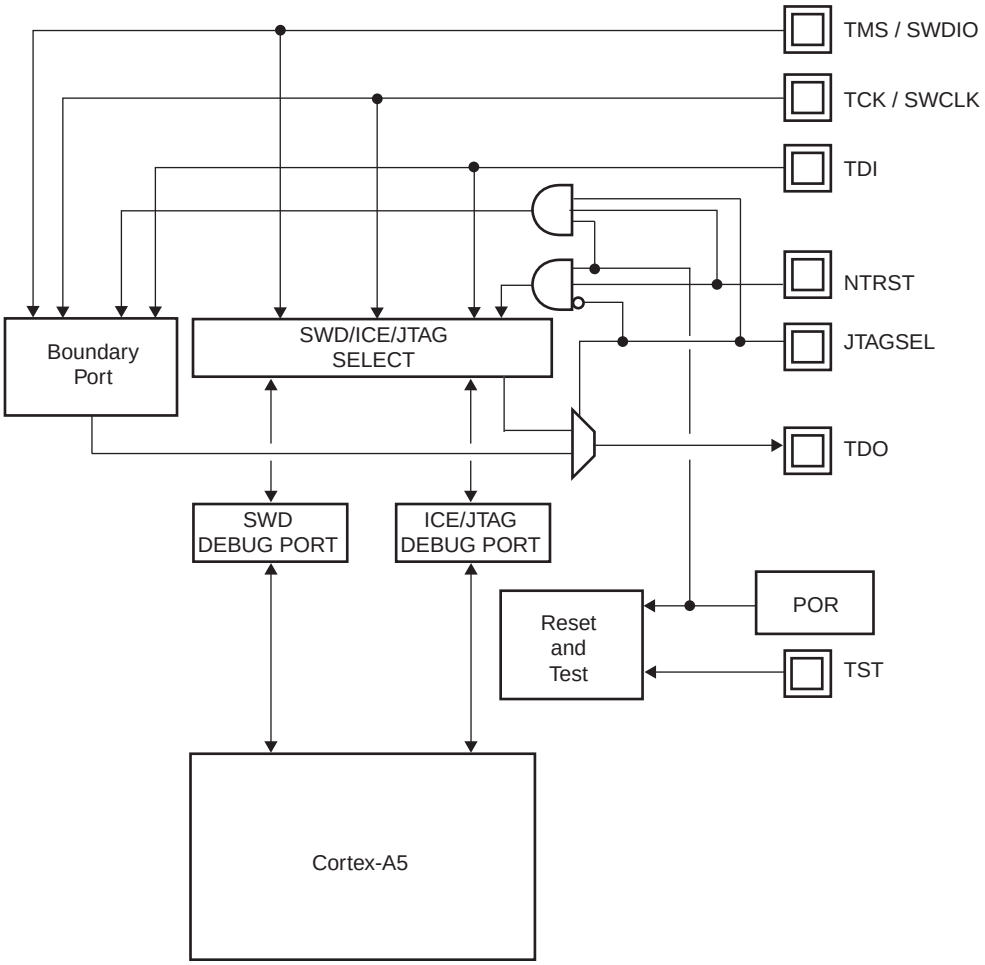


Figure 14-2. Debug and Test Interface Block Diagram

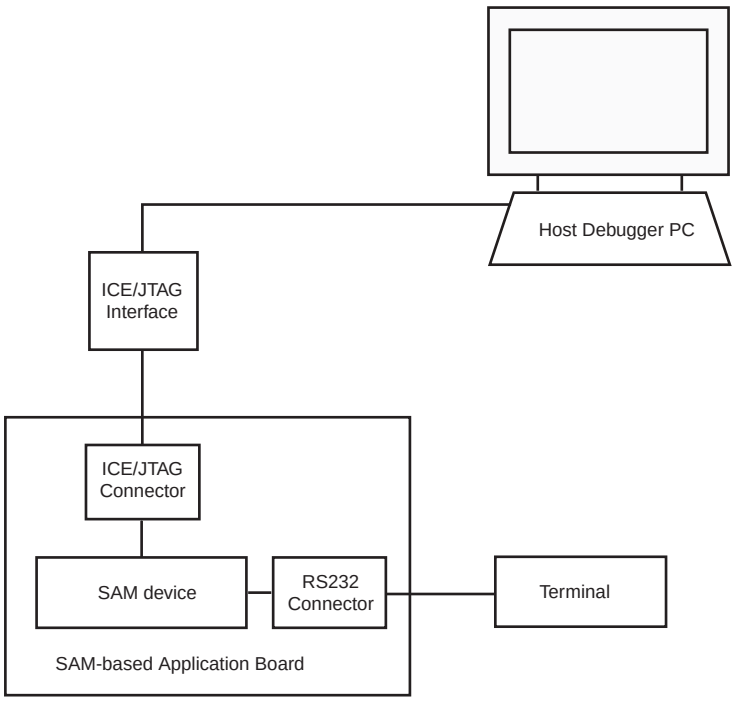


14.4 Application Examples

14.4.1 Debug Environment

Figure 14-3 shows a complete debug environment example. The ICE/JTAG interface is used for standard debugging functions, such as downloading code and single-stepping through the program. A software debugger running on a personal computer provides the user interface for configuring a Trace Port interface utilizing the ICE/JTAG interface.

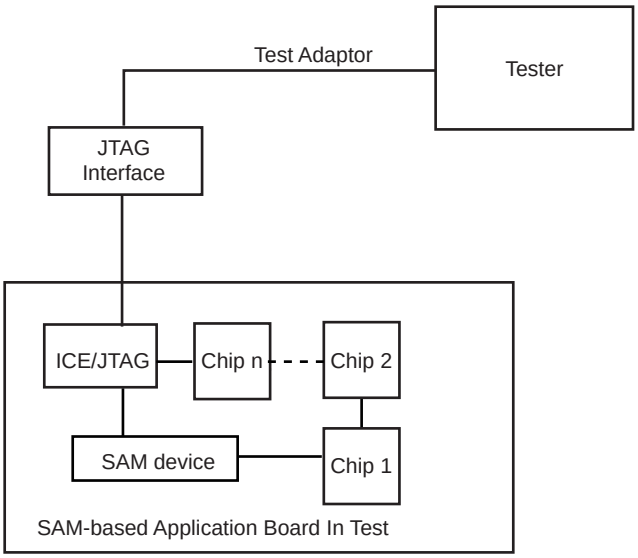
Figure 14-3. Application Debug and Trace Environment Example



14.4.2 Test Environment

Figure 14-4 shows a test environment example. Test vectors are sent and interpreted by the tester. In this example, the “board in test” is designed using a number of JTAG-compliant devices. These devices can be connected to form a single scan chain.

Figure 14-4. Application Test Environment Example



14.5 Debug and Test Pin Description

Table 14-1. Debug and Test Pin List

Pin Name	Function	Type	Active Level
Reset/Test			
NRST	Microprocessor Reset	Input	Low
TST	Test Mode Select	Input	–
NTRST	Test Reset Signal	Input	–
ICE and JTAG			
TCK/SWCLK	Test Clock/Serial Wire Clock	Input	–
TDI	Test Data In	Input	–
TDO	Test Data Out	Output	–
TMS/SWDIO	Test Mode Select/Serial Wire Input/Output	I/O	–
JTAGSEL	JTAG Selection	Input	–

14.6 Functional Description

14.6.1 Test Pin

One dedicated pin, TST, is used to define the device operating mode. The user must make sure that this pin is tied at low level to ensure normal operating conditions. Other values associated with this pin are reserved for manufacturing test.

14.6.2 EmbeddedICE

The Cortex-A5 EmbeddedICE-RT is supported via the ICE/JTAG port. It is connected to a host computer via an ICE interface.The internal state of the Cortex-A5 is examined through an ICE/JTAG port which allows instructions to be serially inserted into the pipeline of the core without using the external data bus. Therefore, when in debug state, a store-multiple (STM) can be inserted into the instruction pipeline. This exports the contents of the Cortex-A5 registers. This data can be serially shifted out without affecting the rest of the system.

There are two scan chains inside the Cortex-A5 processor which support testing, debugging, and programming of the EmbeddedICE-RT. The scan chains are controlled by the ICE/JTAG port.

EmbeddedICE mode is selected when JTAGSEL is low. It is not possible to switch directly between ICE and JTAG operations. A chip reset must be performed after JTAGSEL is changed.

For further details on the EmbeddedICE-RT, see the ARM document:

ARM IHI 0031A_ARM_debug_interface_v5.pdf

14.6.3 JTAG Signal Description

TMS is the Test Mode Select input which controls the transitions of the test interface state machine.

TDI is the Test Data Input line which supplies the data to the JTAG registers (Boundary Scan Register, Instruction Register, or other data registers).

TDO is the Test Data Output line which is used to serially output the data from the JTAG registers to the equipment controlling the test. It carries the sampled values from the boundary scan chain (or other JTAG registers) and propagates them to the next chip in the serial test circuit.

NTRST (optional in IEEE Standard 1149.1) is a Test-ReSeT input which is mandatory in ARM cores and used to reset the debug logic. On Atmel Cortex-A5-based cores, NTRST is a Power On Reset output. It is asserted on power on. If necessary, the user can also reset the debug logic with the NTRST pin assertion during 2.5 MCK periods.

TCK is the Test ClocK input which enables the test interface. TCK is pulsed by the equipment controlling the test and not by the tested device. It can be pulsed at any frequency.

14.6.4 Chip Access Using JTAG Connection

The JTAG connection is not enabled by default on this chip at delivery due to the secure ROM code implementation.

By default, the SAMA5D2 devices boot in Standard mode and not in Secure mode. When the secure ROM code starts, it disables the JTAG access for the entire boot sequence.

If the secure ROM code does not find any program in the external memory, it enables the USB connection and the serial port and waits for a dedicated command to switch the chip into Secure mode.

If any other character is received, the secure ROM code starts the standard SAM-BA® monitor, locks access to the ROM memory, and enables the JTAG.

The chip can then be accessed using the JTAG connection.

If the secure ROM code finds a bootable program, it automatically disables ROM access and enables the JTAG connection just before launching the program.

The procedure to enable JTAG access is as follows:

- Connect your computer to the board with JTAG and USB (J20 USB-A)
- Power on the chip
- Open a terminal console (TeraTerm or HyperTerminal, etc.) on your computer and connect to the USB CDC Serial COM port related to the J14 connector on the board
- Send the '#' character. You will see then the prompt '>' character sent by the device (indicating that the Standard SAM-BA Monitor is running)
- Use the Standard SAM-BA Monitor to connect to the chip with JTAG

Note that you don't need to follow this sequence in order to connect the Standard SAM-BA Monitor with USB.

14.6.5 IEEE 1149.1 JTAG Boundary Scan

IEEE 1149.1 JTAG Boundary Scan allows pin-level access independent of the device packaging technology.

IEEE 1149.1 JTAG Boundary Scan is enabled when JTAGSEL is high. The SAMPLE, EXTEST and BYPASS functions are implemented. In ICE debug mode, the ARM processor responds with a non-JTAG chip ID that identifies the processor to the ICE system. This is not IEEE 1149.1 JTAG-compliant.

It is not possible to switch directly between JTAG and ICE operations. A chip reset must be performed after JTAGSEL is changed.

A Boundary-scan Descriptor Language (BSDL) file is provided to set up test.

14.7 Boundary JTAG ID Register

Access: Read-only

31	30	29	28	27	26	25	24
VERSION				PART NUMBER			

23	22	21	20	19	18	17	16
PART NUMBER							

15	14	13	12	11	10	9	8
PART NUMBER				MANUFACTURER IDENTITY			

7	6	5	4	3	2	1	0
MANUFACTURER IDENTITY							1

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part number is 0x5B39.

- **MANUFACTURER IDENTITY[11:1]**

Set to 0x01F.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

JTAG ID Code value is 0x05B3903F.

14.8 Cortex-A5 DP Identification Code Register IDCODE

The Identification Code Register is always present on all DP implementations. It provides identification information about the ARM Debug Interface.

14.8.1 JTAG Debug Port (JTAG-DP)

It is accessed using its own scan chain, the JTAG-DP Device ID Code Register.

Access: Read-only

31	30	29	28	27	26	25	24
VERSION				PART NUMBER			

23	22	21	20	19	18	17	16
PART NUMBER							

15	14	13	12	11	10	9	8
PART NUMBER				DESIGNER			

7	6	5	4	3	2	1	0
DESIGNER							1

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part Number is 0xBA00

- **DESIGNER[11:1]**

Set to 0x23B.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

Debug Port JTAG IDCODE value is 0x4BA00477.

14.8.2 Serial Wire Debug Port (SW-DP)

It is at address 0x0 on read operations when the APnDP bit = 0. Access to the Identification Code Register is not affected by the value of the CTRLSEL bit in the Select Register

Access: Read-only

31	30	29	28	27	26	25	24
VERSION				PART NUMBER			

23	22	21	20	19	18	17	16
PART NUMBER							

15	14	13	12	11	10	9	8
PART NUMBER				DESIGNER			

7	6	5	4	3	2	1	0
DESIGNER							1

- **VERSION[31:28]: Product Version Number**

Set to 0x0.

- **PART NUMBER[27:12]: Product Part Number**

Product part Number is 0xBA01

- **DESIGNER[11:1]**

Set to 0x23B.

Bit[0] required by IEEE Std. 1149.1.

Set to 0x1.

Debug Port Serial Wire IDCODE is 0x5ba02477.

15. Standard Boot Strategies

15.1 Description

The system always boots from the ROM memory at address 0x0.

The ROM code is a boot program contained in the embedded ROM. It is also called “First level bootloader”.

This microcontroller can be configured to run a Standard Boot mode or a Secure Boot mode. More information on how the Secure Boot mode can be enabled, and how the chip operates in this mode, is provided in the application note “SAMA5D2x Secure Boot Strategy”, Atmel literature No. 44040. To obtain this application note and additional information about the secure boot and related tools, contact an Atmel Sales Representative.

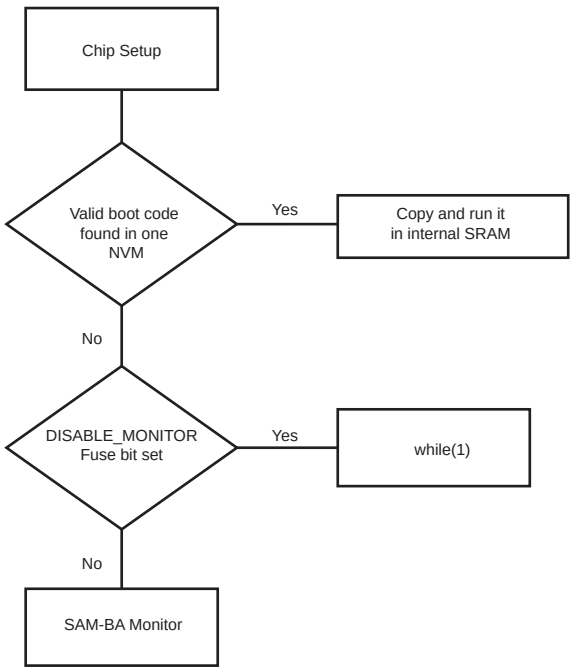
By default, the chip starts in Standard Boot mode.

Note: JTAG access is disabled during the execution of the ROM code sequence. It is re-enabled when jumping into SRAM when a valid code has been found on an external NVM, at the same time the ROM memory is hidden. If no valid boot has been found on an external NVM, the ROM code enables the USB connection and one UART serial port, the ROM code starts the standard SAM-BA monitor, locks access to the ROM memory and re-enables the JTAG connection.

15.2 Flow Diagram

The ROM code global flow is shown in Figure 15-1.

Figure 15-1. ROM Code Flow Diagram



15.3 Chip Setup

When the chip is powered on, the processor clock (PCK) and the master clock (MCK) source is the 12 MHz fast RC oscillator.

The ROM code performs a low-level initialization that follows the steps described below:

1. Stack Setup for ARM supervisor mode.
2. PLLA Initialization: PLLA is configured to get a PCK at 396 MHz and an MCK at 132 MHz.
3. Master Clock Selection: when the PLLA is stabilized, the Master Clock source is switched from internal 12-MHz RC to PLLA. The PMC Status Register is polled to wait for MCK Ready. PCK and MCK are now the Main Clock.
4. C Variable Initialization: non zero-initialized data is initialized in the RAM (copy from ROM to RAM). Zero-initialized data is set to 0 in the RAM.

Note: No external crystal or clock is needed during the external boot memories sequence. An external clock source is checked before the launch of the SAM-BA monitor to get a more accurate clock signal for USB.

15.4 Boot configuration

The boot sequence is controlled using a Boot Configuration Word in the Fuse area.

15.4.1 Boot Configuration Word

The Boot Configuration Word allows several customizations of the Boot Sequence:

- To configure the IO Set where the external memories are connected (refer to Section 15.4.7 “Hardware and Software Constraints” for a description of the IO Sets)
- To disable the boot on selected memories
- To configure the UART port used as a terminal console
- To configure the JTAG pins used for debug

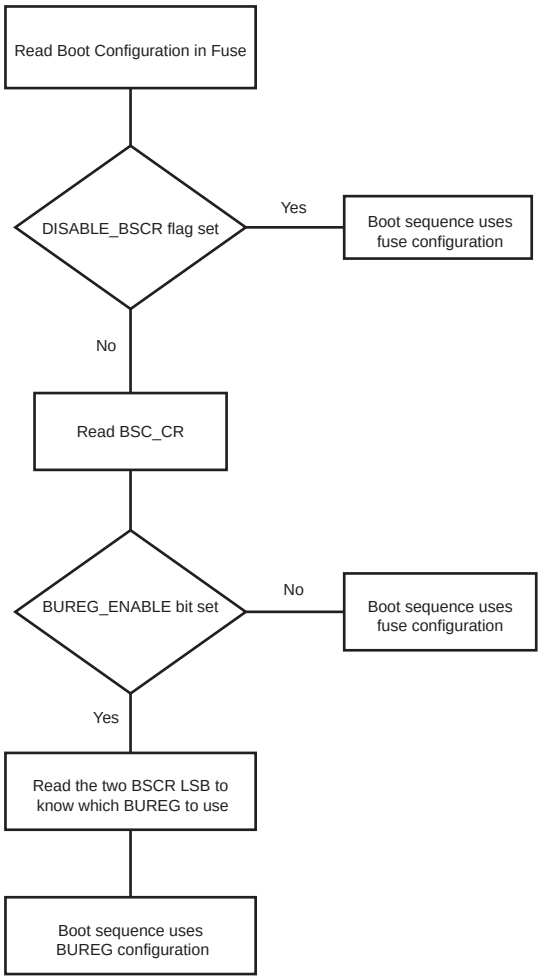
Refer to “Section 15.4.3 “Boot Configuration Word” for a detailed description of all the bitfields in this word.

By default, the value of this word is 0x0. In this configuration, the ROM code does not try to detect a valid bootable software in any external memory, and runs directly the SAM-BA monitor.

The value of this fuse word can be overridden by the content of a General Purpose Backup Register. The conditions to enable this feature are as follows:

- The fuse bit DISABLE_BSCR must not be set (default value).
- The Boot Sequence Controller Configuration Register (BSC_CR) must have the BUREG_VALID bit set and indicate in BUREG_INDEX which register has to be used.

Figure 15-2. Boot Configuration Loading



15.4.2 Boot Sequence Controller Configuration Register

Name: BSC_CR

Address: 0xF8048054

Access: Read-write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	_	BUREG_VALID	BUREG_INDEX	

- **WPKEY: Write Protect Key (Write-only)**

Value	Name	Description
0x6683	PASSWD	Writing any other value in this field aborts the write operation of the BOOT field. Always reads as 0.

- **BUREG_VALID: Validate the data in BUREG_INDEX field**

0: No BUREG contains valid Boot Configuration data.

1: The BUREG indicated in BUREG_INDEX contains valid Boot Configuration data.

- **BUREG_INDEX: Select the BUREG where the Boot Configuration data shall be read**

Value	Name	Description
0	BUREG_0	Use BUREG 0 value
1	BUREG_1	Use BUREG 1 value
2	BUREG_2	Use BUREG 2 value
3	BUREG_3	Use BUREG 3 value

15.4.3 Boot Configuration Word

Reset: 0x00000000

31	30	29	28	27	26	25	24
–	–	SECURE_ MODE	DNU	DNU	DNU	DNU	DISABLE_ MONITOR

23	22	21	20	19	18	17	16
DNU	DISABLE_ BSCR	QSPI_XIP_ MODE	DNU	DNU	EXT_MEM_ BOOT_ ENABLE	JTAG_IO_SET	

15	14	13	12	11	10	9	8
UART_CONSOLE				SDMMC_1	SDMMC_0	NFC	

7	6	5	4	3	2	1	0
SPI_1		SPI_0		QSPI_1		QSPI_0	

The Boot Configuration Word comprises the 32 boot configuration bits (see Table 15-7 “Customer Fuse Matrix”).

Note: To avoid any malfunctioning, the user must not write the “DO NOT USE (DNU)” fuse bits.

- **SECURE_MODE: Enable Secure Boot Mode**

0: Standard Boot Sequence

1: Secure Boot Sequence

- **DISABLE_MONITOR: Disable SAM-BA Monitor**

0: If no boot file found, launch SAM-BA Monitor.

1: SAM-BA Monitor never launched.

- **DISABLE_BSCR: Disable Read of BSC_CR**

0: If the BUREG index in the BSC_CR is valid, its data replace Fuse configuration bits.

1: Does not read BSC_CR content, so the Boot settings are those from the Fuse.

- **QSPI_XIP_MODE: Enable XIP Mode on QSPI Flash**

0: QSPI is accessed in QSPI mode and data copied into internal SRAM.

1: QSPI is accessed in XIP mode, and the bootstrap directly executed from it.

- **EXT_MEM_BOOT_ENABLE: Enable Boot on External Memories**

0: No external memory boot performed.

1: External memory boot enabled.

- **JTAG_IO_SET: Select the pins used for JTAG access**

Value	Name	Description
0	JTAG_IOSET_1	Use JTAG IO Set 1
1	JTAG_IOSET_2	Use JTAG IO Set 2
2	JTAG_IOSET_3	Use JTAG IO Set 3
3	JTAG_IOSET_4	Use JTAG IO Set 4

- **UART_CONSOLE: Select the Pins and UART Interface Used as a Console Terminal**

Value	Name	Description
0	UART_1_IOSET_1	Use UART1 IO Set 1
1	UART_0_IOSET_1	Use UART0 IO Set 1
2	UART_1_IOSET_2	Use UART1 IO Set 2
3	UART_2_IOSET_1	Use UART2 IO Set 1
4	UART_2_IOSET_2	Use UART2 IO Set 2
5	UART_2_IOSET_3	Use UART2 IO Set 3
6	UART_3_IOSET_1	Use UART3 IO Set 1
7	UART_3_IOSET_2	Use UART3 IO Set 2
8	UART_3_IOSET_3	Use UART3 IO Set 3
9	UART_4_IOSET_1	Use UART4 IO Set 1
10	DISABLED	No console terminal
11	DISABLED	No console terminal
12	DISABLED	No console terminal
13	DISABLED	No console terminal
14	DISABLED	No console terminal
15	DISABLED	No console terminal

- **SDMMC_1: Disable SDCard/e.MMC Boot on SDMMC_1**

0: Boot on SDMMC_1 using SDMMC_1 PIO Set 1.

1: Disable boot on SDMMC_1.

- **SDMMC_0: Disable SDCard/e.MMC Boot on SDMMC_0**

0: Boot on SDMMC_0 using SDMMC_0 PIO Set 1.

1: Disable boot on SDMMC_0.

- **NFC: Select the PIO Set Used for NFC Boot**

Value	Name	Description
0	NFC_IOSET_1	Use NFC PIO Set 1
1	NFC_IOSET_2	Use NFC PIO Set 2
2	DISABLED	NFC boot is disabled
3	DISABLED	NFC boot is disabled

- **SPI_1: Select the PIO Set Used for SPI_1 Boot**

Value	Name	Description
0	SPI_1_IOSET_1	Use SPI_1 PIO Set 1
1	SPI_1_IOSET_2	Use SPI_1 PIO Set 2
2	SPI_1_IOSET_3	Use SPI_1 PIO Set 3
3	DISABLED	SPI_1 boot is disabled

- **SPI_0: Select the PIO Set Used for SPI_0 Boot**

Value	Name	Description
0	SPI_0_IOSET_1	Use SPI_0 PIO Set 1
1	SPI_0_IOSET_2	Use SPI_0 PIO Set 2
2	DISABLED	SPI_0 boot is disabled
3	DISABLED	SPI_0 boot is disabled

- **QSPI_1: Select the PIO Set Used for QSPI_1 Boot**

Value	Name	Description
0	QSPI_1_IOSET_1	Use QSPI_1 PIO Set 1
1	QSPI_1_IOSET_2	Use QSPI_1 PIO Set 2
2	QSPI_1_IOSET_3	Use QSPI_1 PIO Set 3
3	DISABLED	QSPI_1 boot is disabled

- **QSPI_0: Select the PIO Set Used for QSPI_0 Boot**

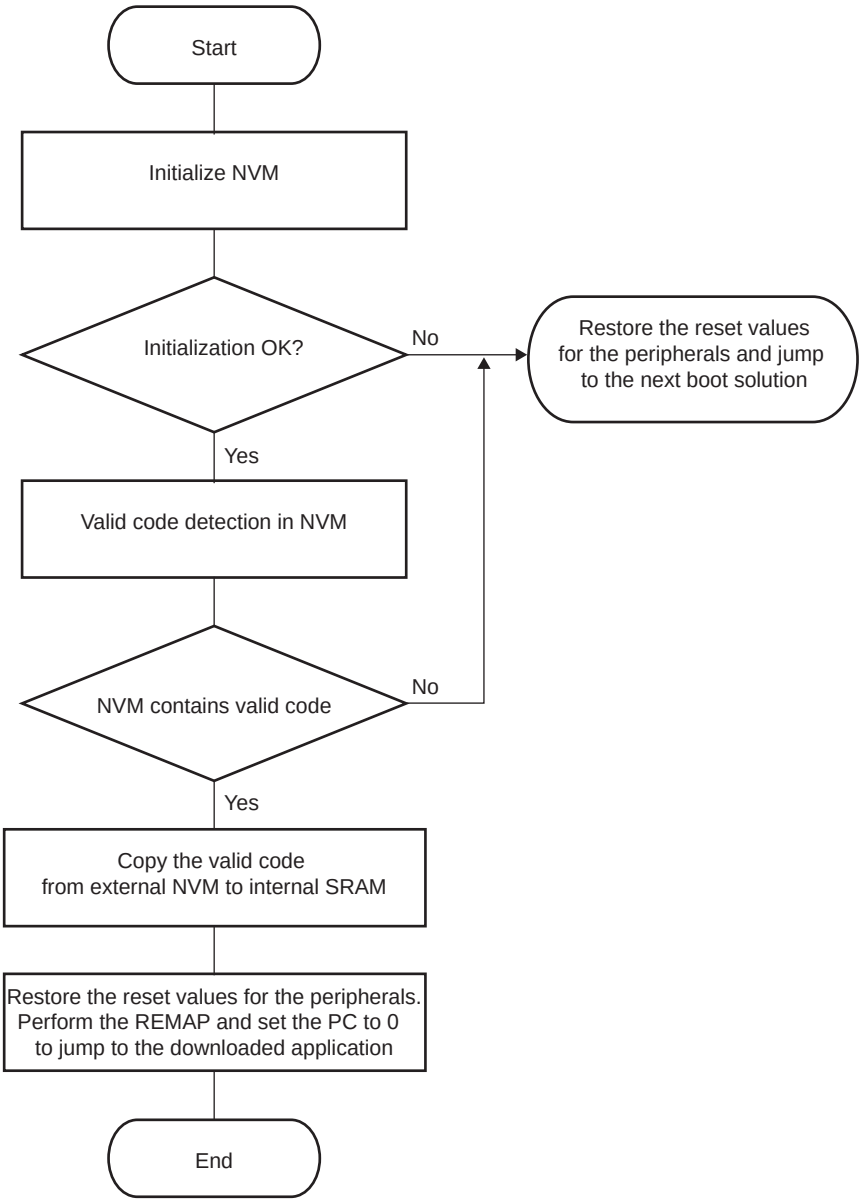
Value	Name	Description
0	QSPI_0_IOSET_1	Use QSPI_0 PIO Set 1
1	QSPI_0_IOSET_2	Use QSPI_0 PIO Set 2
2	QSPI_0_IOSET_3	Use QSPI_0 PIO Set 3
3	DISABLED	QSPI_0 boot is disabled

15.4.4 NVM Boot Sequence

The ROM code performs the initialization and valid code detection for the external memories as described below only if those memories are not disabled in the Boot Configuration word.

Figure 15-3. NVM Bootloader Program Description

Figure 15-4. NVM Boot Diagram



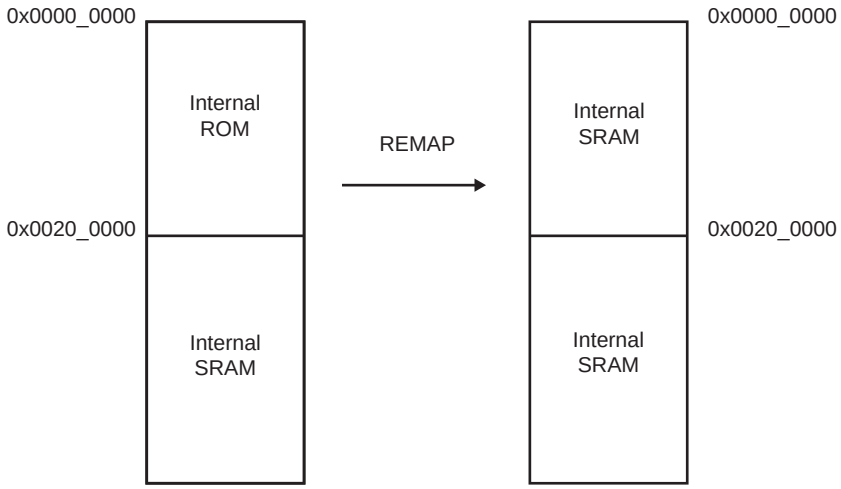
The NVM bootloader program first initializes the PIOs related to the NVM device. Then it configures the right peripheral depending on the NVM and tries to access this memory. If the initialization fails, it restores the reset values for the PIO and the peripheral, and then tries to fulfill the same operations on the next NVM of the sequence.

If the initialization is successful, the NVM bootloader program reads the beginning of the NVM and determines if the NVM contains a valid code.

If the NVM does not contain a valid code, the NVM bootloader program restores the reset value for the peripherals and then tries to fulfill the same operations on the next NVM of the sequence.

If a valid code is found, this code is loaded from the NVM into the internal SRAM and executed by branching at address 0x0000_0000 after remap. This code may be the application code or a second-level bootloader. All the calls to functions are PC-relative and do not use absolute addresses.

Figure 15-5. Remap Action after Download Completion



15.4.5 Valid Code Detection

There are two kinds of valid code detection.

15.4.5.1 ARM Exception Vectors Check

The NVM bootloader program reads and analyzes the first 28 bytes corresponding to the first seven ARM exception vectors. Except for the sixth vector, these bytes must implement the ARM instructions for either branch or load PC with PC-relative addressing.

Figure 15-6. LDR Opcode

31 28	27 24	23 20	19 16	15 12	11 0
1 1 1 0	0 1 I P	U 1 W 0	Rn	Rd	Offset

Figure 15-7. B Opcode

31 28	27 24	23 0
1 1 1 0	1 0 1 0	Offset (24 bits)

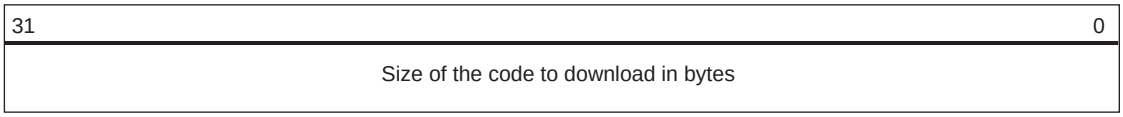
Unconditional instruction: 0xE for bits 31 to 28.

Load PC with the PC-relative addressing instruction:

- Rn = Rd = PC = 0xF
- I==0 (12-bit immediate value)
- P==1 (pre-indexed)
- U offset added (U==1) or subtracted (U==0)
- W==1

The sixth vector, at the offset 0x14, contains the size of the image to download. The user must replace this vector with the user's own vector. This procedure is described below.

Figure 15-8. Arm Vector 6 Structure



The value has to be smaller than 64 Kbytes.

Example

An example of valid vectors:

```
00          ea000006        B 0x20
04          eafffffe        B 0x04
08          ea00002f        B _main
0c          eafffffe        B 0x0c
10          eafffffe        B 0x10
14          00001234        B 0x14 ← Code size = 4660 bytes
18          eafffffe        B 0x18
```

15.4.5.2 boot.bin File Check

This method is the one used on FAT-formatted SD Card and e.MMC. The boot program must be a file named "`boot.bin`" written in the root directory of the file system. Its size must not exceed the maximum size allowed: 64 Kbytes (0x10000).

15.4.6 Detailed Memory Boot Procedures

15.4.6.1 NAND Flash Boot: NAND Flash Detection

After the NAND Flash interface configuration, a reset command is sent to the memory.

Hardware ECC detection and correction are provided by the PMECC peripheral. Refer to Section 34.18 "PMECC Controller Functional Description" for more details.

The Boot Program is able to retrieve NAND Flash parameters and ECC requirements using two methods as follows:

- The detection of a specific header written at the beginning of the first page of the NAND Flash,

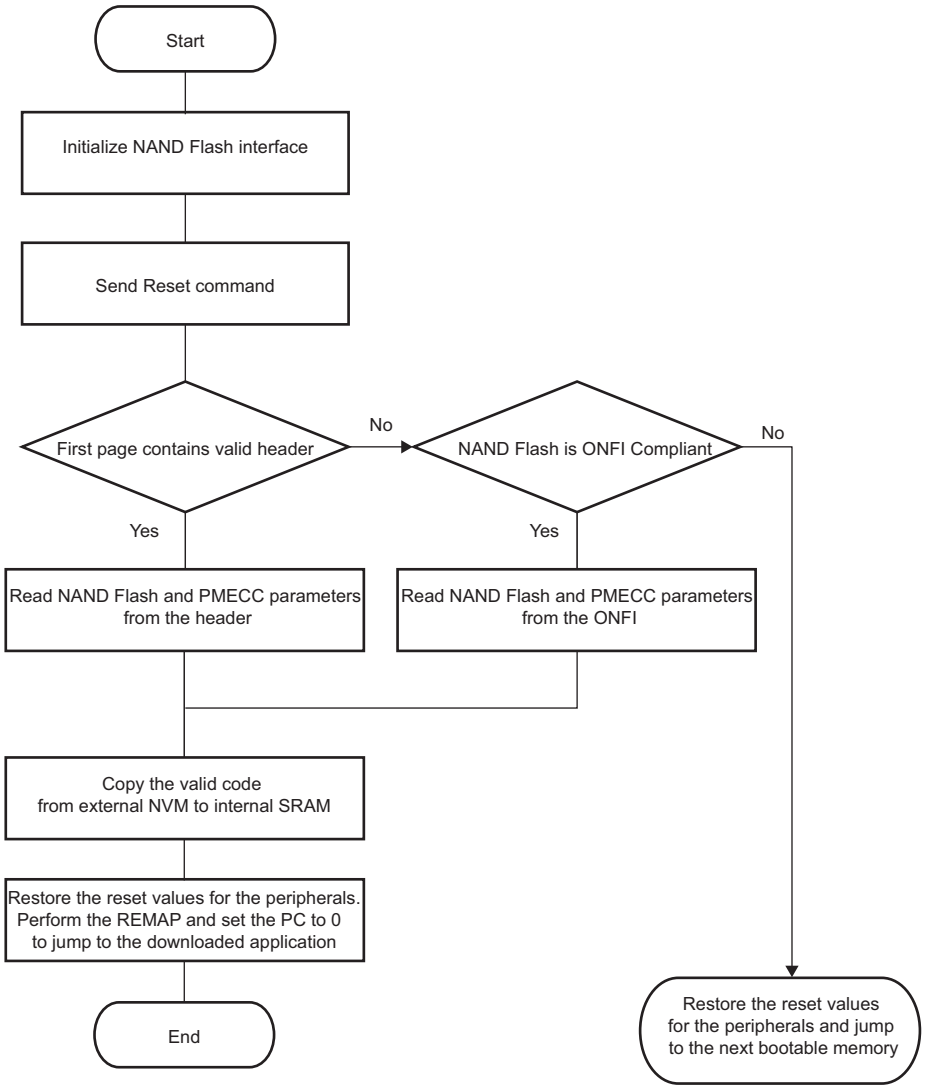
or

- Through the ONFI parameters for the ONFI-compliant memories

However, it is highly recommended to use the NAND Flash Header method (first bullet above) since it indicates exactly how the PMECC has been configured to write the bootable program in the NAND Flash, and not to rely only on the NAND Flash capabilities.

Note: Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

Figure 15-9. Boot NAND Flash Download



NAND Flash Specific Header Detection (Recommended Solution)

This is the first method used to determine NAND Flash parameters. After Initialization and Reset command, the Boot Program reads the first page without an ECC check, to determine whether the NAND parameter header is present. The header is made of 52 times the same 32-bit word (for redundancy reasons) which must contain NAND and PMECC parameters used to correctly perform the read of the rest of the data in the NAND. This 32-bit word is described below:

31	30	29	28	27	26	25	24
key				-	eccOffset		

23	22	21	20	19	18	17	16
eccOffset						sectorSize	

15	14	13	12	11	10	9	8
eccBitReq			spareSize				

7	6	5	4	3	2	1	0
spareSize				nbSectorPerPage			usePmecc

Note: Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

- **usePmecc: Use PMECC**

0: Do not use PMECC to detect and correct the data.

1: Use PMECC to detect and correct the data.

- **nbSectorPerPage: Number of Sectors per Page**
- **spareSize: Size of the Spare Zone in Bytes**
- **eccBitReq: Number of ECC Bits Required**

0: 2-bit ECC

1: 4-bit ECC

2: 8-bit ECC

3: 12-bit ECC

4: 24-bit ECC

5: 32-bit ECC

- **sectorSize: Size of the ECC Sector**

0: For 512 bytes

1: For 1024 bytes per sector

Other value for future use.

- **eccOffset: Offset of the First ECC Byte in the Spare Zone**

A value below 2 is not allowed and is considered as 2.

- **key: Value 0xC Must be Written here to Validate the Content of the Whole Word.**

If the header is valid, the Boot Program continues with the detection of a valid code.

ONFI 2.2 Parameters (Not Recommended)

In case no valid header is found, the Boot Program checks if the NAND Flash is ONFI-compliant, sending a Read Id command (0x90) with 0x20 as parameter for the address. If the NAND Flash is ONFI-compliant, the Boot Program retrieves the following parameters with the help of the Get Parameter Page command:

- Number of bytes per page (byte 80)
- Number of bytes in spare zone (byte 84)
- Number of ECC bit corrections required (byte 112)
- ECC sector size: by default, set to 512 bytes; or to 1024 bytes if the ECC bit capability above is 0xFF

By default, the ONFI NAND Flash detection turns ON the usePmecc parameter, and the ECC correction algorithm is automatically activated.

Once the Boot Program retrieves the parameter, using one of the two methods described above, it reads the first page again, with or without ECC, depending on the usePmecc parameter. Then it looks for a valid code programmed just after the header offset 0xD0. If the code is valid, the program is copied at the beginning of the internal SRAM.

Note: Booting on 16-bit NAND Flash is not possible; only 8-bit NAND Flash memories are supported.

15.4.6.2 NAND Flash Boot: PMECC Error Detection and Correction

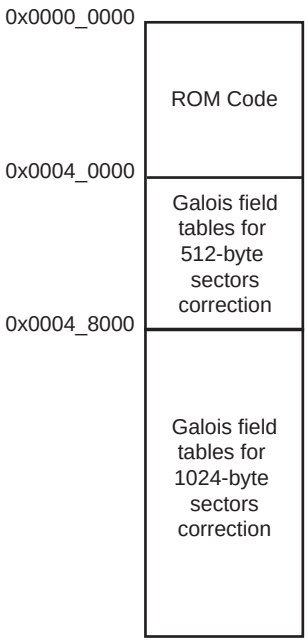
NAND Flash boot procedure uses PMECC to detect and correct errors during NAND Flash read operations in two cases:

- When the usePmecc flag is set in a specific NAND header.
  If the flag is not set, no ECC correction is performed during the NAND Flash page read.
- When the NAND Flash has been detected using ONFI parameters.

The ROM memory embeds the Galois field tables. The user does not need to embed them in his own software.

The Galois field tables are mapped in the ROM just after the ROM code, as described in Figure 15-10.

Figure 15-10. Galois Field Table Mapping



For a full description and an example of how to use the PMECC detection and correction feature, refer to the software package dedicated to this device on Atmel's website.

15.4.6.3 SD Card / e.MMC Boot

Supported SD Card Devices

SD Card Boot supports all SD Card memories compliant with the SD Memory Card Specification V3.0. This includes SDMMC cards.

e.MMC with Boot Partition

The ROM code first checks if the e.MMC Boot Partition is enabled. If enabled, the ROM code reads the first 64 Kbytes of the boot partition, and copy them into the internal SRAM.

FAT Filesystem boot

If no boot partition is enabled on an e.MMC, the boot process continues with a Standard SDCard/e.MMC detection, and the ROM code looks for a "`boot.bin`" file in the root directory of a FAT12/16/32 file system.

The SDCard/e.MMC boot requires the Card Detect pin to be connected.

15.4.6.4 SPI Flash Boot

Two types of SPI Flash are supported: SPI Serial Flash and SPI DataFlash.

The SPI Flash bootloader tries to boot on SPI0, first looking for SPI Serial Flash, and then for SPI DataFlash.

It uses only one valid code detection: analysis of ARM exception vectors.

The SPI Flash read is done by means of a Continuous Read command from the address 0x0. This command is 0xE8 for DataFlash and 0x0B for Serial Flash devices.

Supported DataFlash Devices

The SPI Flash Boot program supports the DataFlash devices listed in Table 15-1.

Table 15-1. DataFlash Devices

Device	Density	Page Size (bytes)	Number of Pages
AT45DB011	1 Mbit	264	512
AT45DB021	2 Mbits	264	1024
AT45DB041	4 Mbits	264	2048
AT45DB081	8 Mbits	264	4096
AT45DB161	16 Mbits	528	4096
AT45DB321	32 Mbits	528	8192
AT45DB642	64 Mbits	1056	8192
AT45DB641	64 Mbits	264	37768

Supported Serial Flash Devices

The SPI Flash Boot program supports all SPI Serial Flash devices responding correctly to both Get Status and Continuous Read commands.

15.4.6.5 QSPI Flash Boot

Definitions

SPI x-y-z protocol:

- Command opcode is sent on x I/O data line(s) with x in {1, 2, 4}
- Address is sent on y I/O data line(s) with y in {1, 2, 4}
- Data are sent or received on z I/O data lin(s) with z in {1, 2, 4}

Relevant combinations are:

SPI 1-1-1: legacy SPI protocol using MOSI/IO0 and MISO/IO1 lines

SPI 1-1-2: SPI Dual Output using IO0 and IO1 lines

SPI 1-2-2: SPI Dual I/O using IO0 and IO1 lines

SPI 2-2-2: SPI Dual Command using IO0 and IO1 lines

SPI 1-1-4: SPI Quad Output using IO0, IO1, IO2 and IO3 lines

SPI 1-4-4: SPI Quad I/O using IO0, IO1, IO2 and IO3 lines

SPI 4-4-4: SPI Quad Command using IO0, IO1, IO2 and IO3 lines

Supported QSPI Memory Manufacturers

The ROM code only supports the three following manufacturers (manufacturer ID):

- Spansion (01h)
- Micron (20h)
- Macronix (C2h)

Other manufacturer IDs are **ignored**: The ROM code jumps to the next Non-Volatile Memory in the Boot Sequence.

SPI Clock Frequency, Phase and Polarity

The peripheral clock of each QSPI controller is gated from the Master Clock (MCK). The ROM code configures MCK at 132 MHz and the QSPI Serial Clock (QSCK) to **44 MHz**.

The QSPI controller is configured to use Clock Mode 0: Both CPHA and CPOL are cleared in QSPI_SCR.

CPOL = 0: The inactive state value of QSCK is logic level zero.

CPHA = 0: Data is captured on the leading edge of QSCK and changed on the following edge of QSCK.

QSPI Memory Detection

The ROM code probes the QSPI memory using JEDEC Read ID commands. However the opcode and the SPI protocol to be used to read the JEDEC ID of the QSPI memory depend on its Manufacturer and its current internal state.

- Spansion

  Spansion memories do not support the SPI 4-4-4 protocol. The command opcode is always sent on the single MOSI/IO1 data line. Hence when writing the 9Fh opcode on MOSI during the first 8 cycles, Spansion memories should always reply on MISO with their JEDEC ID during the following cycles.

- Micron

  Micron memories provide three modes of operation:

  – Extended SPI: standard SPI protocol upgraded with dual (SPI 1-1-2, SPI 1-2-2) and quad (SPI 1-1-4, SPI 1-4-4) operations
  – Dual I/O SPI: **all** commands use the SPI 2-2-2 protocol
  – Quad I/O SPI: **all** commands use the SPI 4-4-4 protocol

  The ROM code supports the Extended and Quad I/O SPI modes but not Dual I/O SPI.

  In Extended SPI mode, Micron memories replies to the regular Read JEDEC ID opcode using the protocol SPI 1-1-1: the 9Fh opcode is sent on MOSI using eight clock cycles then the JEDEC ID is read from MISO only.

  In Quad I/O SPI mode, Micron memories no longer reply to the regular Read JEDEC ID (9Fh) but answer the new Read JEDEC ID Multiple I/O command instead: The AFh op code is sent on the 4 I/O lines using 2

clock cycles, then **only the 3 first bytes** (1 byte for the Manufacturer ID followed by 2 bytes for the Device ID) of the JEDEC ID are returned by the memory on the 4 I/O lines.

The AFh opcode is not supported in Extended SPI mode.

- Macronix

  Macronix memories provide two modes of operation:

  – SPI: standard SPI protocol upgraded with dual (SPI 1-1-2, SPI 1-2-2) and quad (SPI 1-1-4, SPI 1-4-4) operations.

  – QPI: **all** commands use the SPI 4-4-4 protocol

The ROM code supports only the Macronix SPI mode.

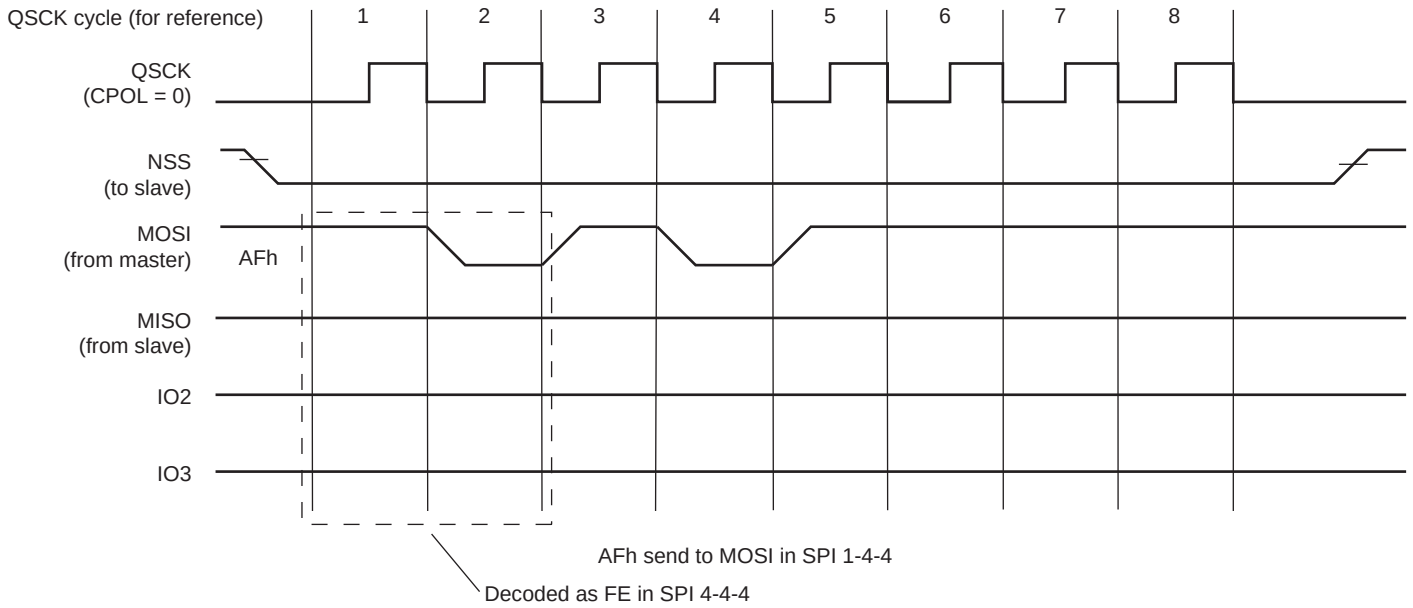
In SPI mode, Macronix memories reply to the regular Read JEDEC ID opcode using the protocol SPI 1-1-1: The 9Fh opcode is sent on MOSI using 8 clock cycles then the JEDEC ID is read from MISO only.

Hence the ROM code uses the following sequence to read the JEDEC ID:

Step	SPI Protocol	Opcode	Support by Manufacturer Modes
1. 1	1-1-1	9Fh	Spansion, Micron Extended SPI, Macronix SPI
2.	1-4-4	AFh	(1)
3.	4-4-4	AFh	Micron Quad I/O SPI

Note: 1. Step 2 is a wrong combination but should not change the internal state of any QSPI memory. Indeed, **assuming pull-up resistors are used on the four I/O lines**, sending the AFh op code with SPI 1-x-y protocols (the opcode is sent only to MOSI during eight clock cycles) to a memory in Quad I/O SPI or QPI mode should be harmless (FEh opcode decoded by the memory when in Quad I/O SPI or QPI mode: unknown opcode). See Figure 15-11.

Figure 15-11. QSPI Transfer Format (CPHA = 0, 8 bits per opcode)



Allowing Quad I/O Commands

On most QSPI memories, some pins are shared between legacy functions such as Write Protect (#WP), Hold (#HOLD) or Reset (#RST) and I/O data lines 2 and 3.

Hence before sending any Quad I/O commands, the ROM code updates the relevant register to reassign those pins to function IO2 and IO3:

- Spansion

  The ROM code sets the Quad Enable bit (bit1) in the Configuration Register (CR) / Status Register 2 (SR2). The bit is volatile or non-volatile depending on memory versions. This operation is performed using the Write Status command (01h), setting SR1 to 00h and SR2 to 02h.

- Micron

  The ROM code updates the Enhanced Volatile Configuration Register (EVCR) to clear the Quad I/O protocol bit (bit7) hence enabling the Quad I/O protocol. From this point, all command must use the SPI 4-4-4 protocol.

- Macronix

  The ROM code updates the Status Register (SR1) to set its Quad Enable non-volatile bit (bit6) using the Write Status command (01h).

Configuration of Fast Read Quad I/O (EBh) Operations

The ROM code performs **all** read operations using the Fast Read Quad I/O (EBh) opcode followed by a 3-byte address.

Since we cannot afford to add an exhaustive table of Read JEDEC IDs and to provide support of future products of memory manufacturers, the ROM code only relies on the very first byte of the JEDEC ID, i.e., the Manufacturer ID, to configure read operations. The ROM code matches the Manufacturer ID as shown in the following table.

Table 15-2. Fast Read Quad I/O (EBh) configuration by Manufacturer ID

Manufacturer ID	Manufacturer	SPI Protocol	# of Mode Cycles	# of Dummy Cycles	Mode Cycle Value	
					(no XIP)	(XIP)
01h	Spansion	SPI 1-4-4	2 (1)	4 (1)	00h	A0h
20h	Micron	SPI 4-4-4	1 (2)	9 (2)	1h	0h
C2h	Macronix	SPI 1-4-4	2 (3)	4 (3)	00h	F0h

Notes:
1. The ROM code **expects** the Latency Control non-volatile bits of the Spansion Status Register 3 (SR3) / Control Register 1 (CR1) to be zero (LC = 0).
   The ROM code **does not update** this value.
2. The ROM code sets the number of mode/dummy cycles for Micron memories updating bits [7:4] of their Volatile Configuration Register (VCR) with the 81h opcode. During this update of the VCR:
   – ROM code v1.1 always clears bit3 to enable XIP.
   – ROM code v1.2 clears bit3 to enable XIP if and only if XIP bit is set in the Boot Config word, otherwise it sets bit3 to disable XIP.
3. The ROM code configures the number of mode/dummy cycles for Macronix memories by clearing the volatile DC0 and DC1 bits (bits [7:6]) in the Configuration Register (CR) / Status Register 2 (SR2). It also clears the 4-byte volatile bit (bit5), resulting in the memory going back to its 3-byte address mode.
   This register updated (read, modify, write) using a Write Status command (01h).

Miscellaneous Information

Pull-up Resistors

The ROM code **removes** the internal pull-up resistors when it configures PIO controller to mux the QSPI controller I/O lines. Therefore the probing step may fail if the Quad I/O mode of the memory has not been enabled yet and if this memory does not embed an internal pull-up resistor on #HOLD or #RESET pin.

This is why we recommend to add external pull-up resistors if needed on the four I/O data lines MOSI/IO0, MISO/IO1, #WP/IO2 and #HOLD/IO3.

Another solution is to update the Quad Enable non-volatile bit in the relevant register to reassign #WP and #HOLD/#RESET pins to functions IO2 and IO3.

4-byte Address Mode (> 16 MB memories)

Except for Macronix, the ROM code never sends any command to the memory to leave its 4-byte address mode or to select its first memory bank.

The ROM code expects to read from the very beginning of the QSPI memory using the Fast Read Quad I/O (EBh) command with a 3-byte address.

Therefore we recommend that the customer application does not change the internal state of the QSPI memory but uses 4-byte opcodes when needed instead. Hence the ROM code can still read from the QSPI memory after a reset of the SoCs.

15.4.7 Hardware and Software Constraints

The NVM drivers use several PIOs in Peripheral mode to communicate with external memory devices. Care must be taken when these PIOs are used by the application. The connected devices could be unintentionally driven at boot time, and thus electrical conflicts between the output pins used by the NVM drivers and the connected devices could occur.

To ensure the correct functionality, it is recommended to plug in critical devices to other pins not used by the NVM.

Table 15-3 on page 141 contains a list of pins that are driven during the boot program execution. These pins are driven during the boot sequence for a period of less than 1 second if no correct boot program is found.

Before performing the jump to the application in the internal SRAM, all the PIOs and peripherals used in the boot program are set to their reset state.

Table 15-3. PIO Driven during Boot Program Execution

NVM Bootloader	Peripheral	IO Set	Pin	PIO Line
SD Card / e.MMC	SDMMC_0	1	SDMMC0_CK	PIOA0
			SDMMC0_CMD	PIOA1
			SDMMC0_DAT0	PIOA2
			SDMMC0_DAT1	PIOA3
			SDMMC0_DAT2	PIOA4
			SDMMC0_DAT3	PIOA5
			SDMMC0_DAT4	PIOA6
			SDMMC0_DAT5	PIOA7
			SDMMC0_DAT6	PIOA8
			SDMMC0_DAT7	PIOA9
			SDMMC0_RSTN	PIOA10
			SDMMC0_VDDSEL	PIOA11
			SDMMC0_WP	PIOA12
			SDMMC0_CD	PIOA13
	SDMMC_1	1	SDMMC1_DAT0	PIOA18
			SDMMC1_DAT1	PIOA19
			SDMMC1_DAT2	PIOA20
			SDMMC1_DAT3	PIOA21
			SDMMC1_CK	PIOA22
			SDMMC1_RSTN	PIOA27
			SDMMC1_CMD	PIOA28
			SDMMC1_WP	PIOA29
			SDMMC1_CD	PIOA30
NAND Flash	HSMC	1	D0–D7	PIOA22-PIOA29
			NANDWE	PIOA30
			NANDCS3	PIOA31
			NAND ALE	PIOB0
			NAND CLE	PIOB1
			NANDOE	PIOB2
		2	D0–D7	PIOA0–PIOA7
			NANDWE	PIOA8
			NANDCS3	PIOA9
			NAND ALE	PIOA10
			NAND CLE	PIOA11
			NANDOE	PIOA12

Table 15-3. PIO Driven during Boot Program Execution (Continued)

NVM Bootloader	Peripheral	IO Set	Pin	PIO Line
SPI Flash	SPI_0	1	SPCK	PIOA14
			MOSI	PIOA15
			MISO	PIOA16
			NPCS0	PIOA17
		2	NPCS0	PIOA30
			MISO	PIOA31
			MOSI	PIOB0
			SPCK	PIOB1
	SPI_1	1	SPCK	PIOC1
			MOSI	PIOC2
			MISO	PIOC3
			NPCS0	PIOC4
		2	SPCK	PIOA22
			MOSI	PIOA23
			MISO	PIOA24
			NPCS0	PIOA25
		3	SPCK	PIOD25
			MOSI	PIOD26
			MISO	PIOD27
			NPCS0	PIOD28

Table 15-3. PIO Driven during Boot Program Execution (Continued)

NVM Bootloader	Peripheral	IO Set	Pin	PIO Line
QSPI Flash	QSPI_0	1	SCK	PIOA0
			CS	PIOA1
			IO0	PIOA2
			IO1	PIOA3
			IO2	PIOA4
			IO3	PIOA5
	QSPI_0	2	SCK	PIOA14
			CS	PIOA15
			IO0	PIOA16
			IO1	PIOA17
			IO2	PIOA18
			IO3	PIOA19
	QSPI_0	3	SCK	PIOA22
			CS	PIOA23
			IO0	PIOA24
			IO1	PIOA25
			IO2	PIOA26
			IO3	PIOA27
	QSPI_1	1	SCK	PIOA6
			CS	PIOA7
			IO0	PIOA8
			IO1	PIOA9
			IO2	PIOA10
			IO3	PIOA11
	QSPI_1	2	SCK	PIOB5
			CS	PIOB6
			IO0	PIOB7
			IO1	PIOB8
			IO2	PIOB9
			IO3	PIOB10
	QSPI_1	3	SCK	PIOB14
			CS	PIOB15
			IO0	PIOB16
			IO1	PIOB17
			IO2	PIOB18
			IO3	PIOB19

Table 15-3. PIO Driven during Boot Program Execution (Continued)

NVM Bootloader	Peripheral	IO Set	Pin	PIO Line
Console Terminal and SAM-BA Monitor	UART_0	1	DRXD	PIOB26
			DTXD	PIOB27
	UART_1	1	DRXD	PIOD2
			DTXD	PIOD3
		2	DRXD	PIOC7
			DTXD	PIOC8
	UART_2	1	DRXD	PIOD4
			DTXD	PIOD5
		2	DRXD	PIOD23
			DTXD	PIOD24
		3	DRXD	PIOD19
			DTXD	PIOD20
	UART_3	1	DRXD	PIOC12
			DTXD	PIOC13
		2	DRXD	PIOC31
			DTXD	PIOD0
		3	DRXD	PIOB11
			DTXD	PIOB12
	UART_4	1	DRXD	PIOB3
			DTXD	PIOB4

Table 15-3. PIO Driven during Boot Program Execution (Continued)

NVM Bootloader	Peripheral	IO Set	Pin	PIO Line
Debug Port	JTAG	1	TCK	PIOD14
			TDI	PIOD15
			TDO	PIOD16
			TMS	PIOD17
			NTRST	PIOD18
		2	TCK	PIOD6
			TDI	PIOD7
			TDO	PIOD8
			TMS	PIOD9
			NTRST	PIOD10
		3	TCK	PIOD27
			TDI	PIOD28
			TDO	PIOD29
			TMS	PIOD30
			NTRST	PIOD31
		4	TCK	PIOA22
			TDI	PIOA23
			TDO	PIOA24
			TMS	PIOA25
			NTRST	PIOA26

15.5 SAM-BA Monitor

This part of the ROM code is executed when no valid code is found in any NVM during the NVM boot sequence, and if the DISABLE_MONITOR Fuse bit is not set.

The Main Clock is switched to the 32 kHz RC oscillator to allow external clock frequency to be measured.

The Main Oscillator is enabled and set in the bypass mode. If the MOSCSELS bit rises, an external clock is connected**.** If not, the Bypass mode is cleared to attempt external quartz detection. This detection is successful when the MOSCXTS and MOSCSELS bits rise, else the internal 12 MHz fast RC oscillator is used as the Main Clock.

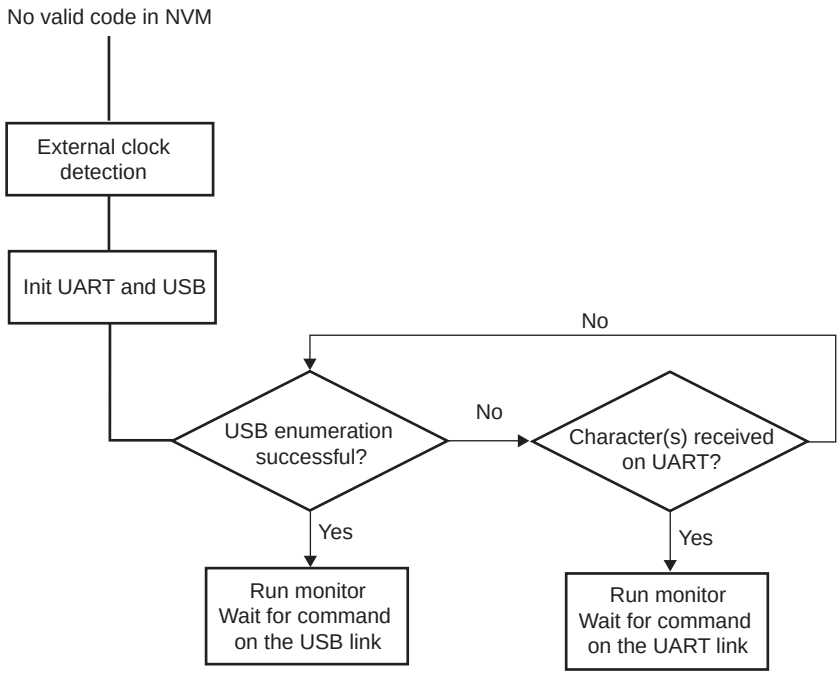
If an external clock or crystal frequency is found, then the PLLA is configured to allow communication on the USB link for the SAM-BA Monitor, else the Main Clock is switched to the internal 12 MHz fast RC oscillator, but USB is not activated.

The SAM-BA Monitor steps are:

- Initialize UART and USB
- Check if USB Device enumeration occurred
- Check if characters are received on the UART

Once the communication interface is identified, the application runs in an infinite loop waiting for different commands as listed in Table 15-4.

Figure 15-12. SAM-BA Monitor Diagram



15.5.1 Command List

Table 15-4. Commands Available through the SAM-BA Monitor

Command	Action	Argument(s)	Example
N	Set Normal Mode	No argument	**N**#
T	Set Terminal Mode	No argument	**T**#
O	Write a byte	Address, Value#	**O**200001,CA#
o	Read a byte	Address,#	**o**200001,#
H	Write a half word	Address, Value#	**H**200002,CAFE#
h	Read a half word	Address,#	**h**200002,#
W	Write a word	Address, Value#	**W**200000,CAFEDECA#
w	Read a word	Address,#	**w**200000,#
S	Send a file	Address,#	**S**200000,#
R	Receive a file	Address, NbOfBytes#	**R**200000,1234#
G	Go	Address#	**G**200200#
V	Display version	No argument	**V**#

- Mode commands:
  - Normal mode configures SAM-BA Monitor to send / receive data in binary format,
  - Terminal mode configures SAM-BA Monitor to send / receive data in ASCII format.
- Write commands: Writes a byte (**O**), a halfword (**H**) or a word (**W**) to the target
  - *Address*: Address in hexadecimal
  - *Value*: Byte, halfword or word to write in hexadecimal
  - *Output*: '>'
- Read commands: Reads a byte (**o**), a halfword (**h**) or a word (**w**) from the target
  - *Address*: Address in hexadecimal
  - *Output*: The byte, halfword or word read in hexadecimal followed by '>'
- Send a file (**S**): Sends a file to a specified address
  - *Address*: Address in hexadecimal
  - *Output*: '>'

Note: There is a timeout on this command which is reached when the prompt '>' appears before the end of the command execution.

- Receive a file (**R**): Receives data into a file from a specified address
  - *Address*: Address in hexadecimal
  - *NbOfBytes*: Number of bytes in hexadecimal to receive
  - *Output*: '>'
- Go (**G**): Jumps to a specified address and executes the code
  - *Address*: Address to jump to in hexadecimal
  - *Output*: '>' once returned from the program execution. If the executed program does not handle the link register at its entry and does not return, the prompt is not displayed.
- Get Version (**V**): Returns the Boot Program version
  - *Output*: version, date and time of ROM code followed by '>'

15.5.2 UART Port

Communication is performed through the UART port initialized to 115,200 bauds, 8 bits of data, no parity, 1 stop bit.

15.5.2.1 Xmodem Protocol

The Send and Receive File commands use the Xmodem protocol to communicate. Any terminal using this protocol can be used to send the application file to the target. The size of the binary file to send depends on the SRAM size embedded in the product. In all cases, the size of the binary file must be lower than the SRAM size because the Xmodem protocol requires some SRAM memory in order to work.

The Xmodem protocol supported is the 128-byte length block. This protocol uses a two-character CRC16 to guarantee detection of maximum bit errors.

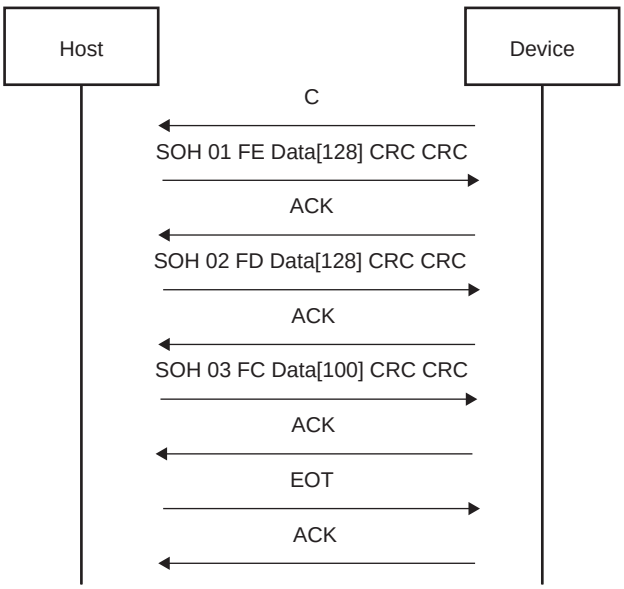
Xmodem protocol with CRC is supported by successful transmission reports provided both by a sender and by a receiver. Each transfer block is as follows:

<SOH><blk #><255-blk #><--128 data bytes--><checksum> in which:

- <SOH> = 01 hex
- <blk #> = binary number, starts at 01, increments by 1, and wraps 0FFH to 00H (not to 01)
- <255-blk #> = 1's complement of the blk#.
- <checksum> = 2 bytes CRC16

Figure 15-13 shows a transmission using this protocol.

Figure 15-13. Xmodem Transfer Example



15.5.3 USB Device Port

15.5.3.1 Supported External Crystal / External Clocks

The SAM-BA Monitor only supports an external crystal or external clock frequency at 12 MHz to allow USB communication.

15.5.3.2 USB Class

The device uses the USB Communication Device Class (CDC) drivers to take advantage of the installed PC Serial Communication software to talk over the USB. The CDC is implemented in all releases of Windows®, starting from Windows 98SE®. The CDC document, available at www.usb.org, describes how to implement devices such as ISDN modems and virtual COM ports.

The Vendor ID is the Atmel's vendor ID 0x03EB. The product ID is 0x6124. These references are used by the host operating system to mount the correct driver. On Windows systems, INF files contain the correspondence between vendor ID and product ID.

15.5.3.3 Enumeration Process

The USB protocol is a master/slave protocol. The host starts the enumeration, sending requests to the device through the control endpoint. The device handles standard requests as defined in the USB Specification.

Table 15-5. Handled Standard Requests

Request	Definition
GET_DESCRIPTOR	Returns the current device configuration value
SET_ADDRESS	Sets the device address for all future device access
SET_CONFIGURATION	Sets the device configuration
GET_CONFIGURATION	Returns the current device configuration value
GET_STATUS	Returns status for the specified recipient
SET_FEATURE	Used to set or enable a specific feature
CLEAR_FEATURE	Used to clear or disable a specific feature

The device also handles some class requests defined in the CDC class.

Table 15-6. Handled Class Requests

Request	Definition
SET_LINE_CODING	Configures DTE rate, stop bits, parity and number of character bits
GET_LINE_CODING	Requests current DTE rate, stop bits, parity and number of character bits
SET_CONTROL_LINE_STATE	RS-232 signal used to indicate to the DCE device that the DTE device is now present

Unhandled requests are stalled.

15.5.3.4 Communication Endpoints

Endpoint 0 is used for the enumeration process.

Endpoint 1 (64-byte Bulk OUT) and endpoint 2 (64-byte Bulk IN) are used as communication endpoints.

SAM-BA Boot commands are sent by the host through Endpoint 1. If required, the message is split into several data payloads by the host driver.

If the command requires a response, the host sends IN transactions to pick up the response.

15.6 Fuse Box Controller

Read/write access to the fuse bits requires that the internal 12 MHz RC oscillator is enabled.

15.6.1 Fuse Bit Mapping

One 32-bit word is reserved for boot configuration. 512 fuse bits are available for customer needs.

The write-once FUSE bit in register SFR_SECURE enables and disables access to the Secure Fuse Controller (SFC).

To avoid any malfunctioning, the user must not write the “DO NOT USE (DNU)” fuse bits.

Table 15-7. Customer Fuse Matrix

SFC_DR	Bits	Use		
16	[543:512]	JTAG_DIS[543]	SEC_DEBUG_DIS[542]	Boot Configuration bits[541:512](1)
15	[511:480]	USER_DATA[511:0]		
14	[479:448]			
13	[447:416]			
12	[415:384]			
11	[383:352]			
10	[351:320]			
9	[319:288]			
8	[287:256]			
7	[255:224]			
6	[223:192]			
5	[191:160]			
4	[159:128]			
3	[127:96]			
2	[95:64]			
1	[63:32]			
0	[31:0]			

Note: 1. See Section 15.4.3 “Boot Configuration Word” for details on the contents of these bits.

Table 15-8. Special Function Bits

JTAG Disable (Fuse bit 543)	Secure Debug Disable (Fuse bit 542)	Description
0	0	Full JTAG debug allowed in Secure and Normal modes
0	1	JTAG debug allowed in Normal mode only (not in Secure mode)
1	X	JTAG debug disabled

16. AXI Matrix (AXIMX)

16.1 Description

The AXI Matrix comprises the embedded Advanced Extensible Interface (AXI) bus protocol which supports separate address/control and data phases, unaligned data transfers using byte strobes, burst-based transactions with only start address issued, separate read and write data channels to enable low-cost DMA, ability to issue multiple outstanding addresses, out-of-order transaction completion, and easy addition of register stages to provide timing closure.

16.2 Embedded Characteristics

- High performance AXI network interconnect
- 1 Master:
  - Cortex-A5 Core
- 2 Slaves:
  - ROM
  - AXI/AHB bridge to AHB Matrix
- Single-cycle arbitration
- Full pipelining to prevent master stalls
- 1 remap state

16.3 Operation

16.3.1 Remap

Remap states are managed in the AXI Matrix Remap Register (AXIMX_REMAP): AXIMX_REMAP.REMAP0 (register bit 0) is used to remap RAM @ addr 0x00000000.

Refer to Section 16.4 “AXI Matrix (AXIMX) User Interface”.

The number of remap states can be defined using eight bits of the AXIMX_REMAP register, and a bit in AXIMX_REMAP controls each remap state.

Each remap state can be used to control the address decoding for one or more slave interfaces. If a slave interface is affected by two remap states that are both asserted, the remap state with the lowest remap bit number takes precedence.

Each slave interface can be configured independently so that a remap state can perform different functions for different masters.

A remap state can:

- Alias a memory region into two different address ranges
- Move an address region
- Remove an address region

Because of the nature of the distributed register subsystem, the masters receive the updated remap bit states in sequence, and not simultaneously.

A slave interface does not update to the latest remap bit setting until:

- The address completion handshake accepts any transaction that is pending
- Any current lock sequence completes

At powerup, ROM is seen at address 0. After powerup, the internal SRAM can be moved down to address 0 by means of the remap bits.

16.4 AXI Matrix (AXIMX) User Interface

Table 16-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	AXI Matrix Remap Register	AXIMX_REMAP	Write-only	–
0x04–0x43108	Reserved	–	–	–

16.4.1 AXI Matrix Remap Register

Name: AXIMX_REMAP

Address: 0x00600000

Access: Read/Write

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
–							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	REMAP0

- **REMAP0: Remap State 0**

SRAM is seen at address 0x00000000 (through AHB slave interface) instead of ROM.

17. Matrix (H64MX/H32MX)

17.1 Description

In order to reduce power consumption without loss in performance, the system embeds three matrixes: one based on the AXI protocol (AXIMX) and two based on the AHB protocol (H64MX and H32MX). A description of the 64-bit AHB Matrix (H64MX) and the 32-bit AHB Matrix (H32MX) implementation follows.

Refer to the product AXIMX description for a complete information on the AXI Matrix.

Each AHB Matrix implements a multilayer AHB, based on the AHB-Lite protocol, which enables parallel access paths between multiple AHB masters and slaves in a system, thus increasing the overall bandwidth. The normal latency to connect a master to a slave is one cycle, except for the default master of the accessed slave which is connected directly (zero cycle latency). The Bus Matrix user interface is compliant with the ARM Advanced Peripheral Bus (APB).

Note: When a master and a slave are on different bus matrixes (AXIMX, H64MX, or H32MX), both matrixes (H64MX and H32MX) and the bridge between the bus matrixes must be configured accordingly.

17.2 Embedded Characteristics

- AMBA Advanced High-performance Bus (AHB-Lite) compliant interface
- 32-bit or 64-bit data bus
- MATRIX0—a 64-bit AHB matrix (H64MX) providing 12 masters for 15 slaves
- MATRIX1—a 32-bit AHB matrix (H32MX) providing 8 masters for 6 slaves
- APB-compliant user interface
- One decoder for each master
- Support for long bursts of 32, 64, 128 and up to the 256-bit word burst AHB limit
- Enhanced programmable mixed arbitration for each slave:
  - Round-robin
  - Fixed priority
  - Latency quality of service
- Programmable default master for each slave:
  - No default master
  - Last accessed default master
  - Fixed default master
- Deterministic maximum access latency for masters
- Zero or one cycle arbitration latency for the first access of a burst
- Bus lock forwarding to slaves
- One special function register for each slave (not dedicated)
- Register write protection
- ARM TrustZone technology extension to AHB and APB

17.3 MATRIX0 (H64MX)

17.3.1 Matrix Masters

The H64MX manages 12 masters, which means that each master can perform an access, concurrently with others, to an available slave.

This matrix operates at MCK.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

Table 17-1. List of H64MX Masters

Master No.	Name	Security Type
0	Bridge from AXI matrix (Core)	Not applicable
1, 2	DMA Controller 0	Peripheral Securable
3, 4	DMA Controller 1	Peripheral Securable
5, 6	LCDC DMA	Peripheral Securable
7	SDMMC0	Peripheral Securable
8	SDMMC1	Peripheral Securable
9	ISC DMA	Peripheral Securable
10	AESB	Not applicable(1)
11	Bridge from H32MX to H64MX	Not applicable

Note: 1. Master signals secure/not secure are propagated through the AES bridge.

17.3.2 Matrix Slaves

The H64MX manages 15 slaves. Each slave has its own arbiter providing a dedicated arbitration per slave.

Table 17-2. List of H64MX Slaves

Slave No.	Description	TZ Access Management
0	Bridge from H64MX to H32MX	Not applicable
1	H64MX APB - User interfaces	HSEL0: not applicable
	SDMMC0	HSEL1: Internal Securable to Peripheral: 1 region(1)
	SDMMC1	HSEL2: Internal Securable to Peripheral: 1 region(1)
2	DDR2 Port0 - AESB	Scalable Securable: 4 regions(2)
3	DDR2 Port1	Scalable Securable: 4 regions(2)
4	DDR2 Port2	Scalable Securable: 4 regions(2)
5	DDR2 Port3	Scalable Securable: 4 regions(2)
6	DDR2 Port4	Scalable Securable: 4 regions(2)
7	DDR2 Port5	Scalable Securable: 4 regions(2)
8	DDR2 Port6	Scalable Securable: 4 regions(2)
9	DDR2 Port7	Scalable Securable: 4 regions(2)
10	Internal SRAM 128K	Internal Securable: 1 region
11	Internal SRAM 128K (Cache L2)	Internal Securable: 1 region

Table 17-2. List of H64MX Slaves (Continued)

Slave No.	Description	TZ Access Management
12	QSPI0	Internal Securable: 1 region
13	QSPI1	Internal Securable: 1 region
14	AESB	Not applicable

Notes: 1. Particular case: see Section 17.12.4 “Security Types of AHB Slave Peripherals” for Internal Securable to Peripheral type configuration. For each SDMMCx, a coherent configuration must be done on
- the AHB Slave,
- MATRIX_SPSELSR for general interrupt and AHB Master,
- MATRIX_SPSELSR for TIMER interrupt

2. For coherency, each DDR2 port shall have the same TZ access management configuration.

17.3.3 Master to Slave Access

Table 17-3 shows how masters and slaves interconnect. Writing in a register or field not dedicated to a master or a slave has no effect.

Table 17-3. Master to Slave Access on H64MX

Matrixes interconnection between Masters and		MASTER											
		0	1	2	3	4	5	6	7	8	9	10	11
SLAVE		Bridge from AXIMX (Core)	XDMAC0		XDMAC1		LCDC DMA		SDMMC0 DMA	SDMMC1 DMA	ISC DMA	AESB	Bridge from H32MX
0	Bridge from H64MX to H32MX	X	X	X	X	X							
1	H64MX APB - User interfaces	X	X	X	X	X							X
	SDMMC0-SDMMC1	X	X	X	X	X							X
2	DDR2 port0											X	
3	DDR2 port1	X											
4	DDR2 port2						X						
5	DDR2 port3							X					
6	DDR2 port4								X	X	X		
7	DDR2 port5		X		X								
8	DDR2 port6			X		X							
9	DDR2 port7												X
10	Internal SRAM	X	X	X	X	X	X	X	X	X	X		X
11	L2C SRAM	X	X	X	X	X	X	X	X	X	X		X
12	QSPI0	X	X	X	X	X						X	X
13	QSPI1	X	X	X	X	X						X	X
14	AESB	X	X	X	X	X							X

17.4 MATRIX1 (H32MX)

17.4.1 Matrix Masters

The H32MX manages eight masters, which means that each master can perform an access, concurrently with others, to an available slave.

This matrix can operate at MCK if MCK is lower than 83 MHz, or at MCK/2 if MCK is higher than 83 MHz. Refer to the PMC section for more details.

Each master has its own decoder, which is defined specifically for each master. In order to simplify the addressing, all the masters have the same decodings.

Table 17-4. List of H32MX Masters

Master No.	Name	Security Type
0	Bridge from H64MX to H32MX	Not applicable
1	Integrity Check Monitor (ICM)	Peripheral Securable
2	UHPHS EHCI DMA	Peripheral Securable
3	UHPHS OHCI DMA	Peripheral Securable
4	UDPHS DMA	Peripheral Securable
5	GMAC DMA	Peripheral Securable
6	CAN0 DMA	Peripheral Securable
7	CAN1 DMA	Peripheral Securable

17.4.2 Matrix Slaves

The H32MX manages six slaves. Each slave has its own arbiter providing a dedicated arbitration per slave.

Table 17-5. List of H32MX Slaves

Slave No.	Description	TZ Access Management
0	Bridge from H32MX to H64MX	Not applicable
1	H32MX Peripheral Bridge 0	Not applicable
2	H32MX Peripheral Bridge 1	Not applicable
3	External Bus Interface	External Securable: 7 regions: HSEL0: 0x10000000 128MB CS0 HSEL1: 0x18000000 128MB CS0 HSEL2: 0x60000000 128MB CS1 HSEL3: 0x68000000 128MB CS1 HSEL4: 0x70000000 128MB CS2 HSEL5: 0x78000000 128MB CS2 HSEL6: 0x80000000 128MB CS3
	NFC command register	Internal Securable to Peripheral: 1 region HSEL7: 0xC0000000 256MB NFCCMD
4	NFC SRAM	Internal Securable: 1 region

Table 17-5. List of H32MX Slaves (Continued)

5	USB Device High Speed (UDPHS) Dual Port RAM (DPR)	HSEL0: Internal Securable: 1 region
	USB Host (UHPHS) OHCI registers	HSEL1: Internal Securable to Peripheral: 1 region(1)
	USB Host (UHPHS) EHCI registers	HSEL2: Internal Securable to Peripheral: 1 region(1)

Note: 1. UHPHS: Coherent configuration must be done on:
- AHB Slave UHPHS OHCI Internal Securable Peripheral
- AHB Slave UHPHS EHCI Internal Securable Peripheral
- MATRIX_SPSELSR for Interrupt and AHB Master

17.4.3 Master to Slave Access

Table 17-6 shows how masters and slaves interconnect. Writing in a register or field not dedicated to a master or a slave has no effect.

Table 17-6. Master to Slave Access on H32MX

		MASTER										
		0 (Through Bridge from H64MX)					1	2	3	4	5	6
SLAVE		Core	XDMAC0 IF0	XDMAC0 IF1	XDMAC1 IF0	XDMAC1 IF1	ICM	UHPHS EHCI DMA	UHPHS OHCI DMA	UDPHS DMA	GMAC DMA	CAN0 DMA
0	Bridge from H32MX to H64MX						X	X	X	X	X	X
1	H32MX APB0 - user interfaces	X		X		X	X					
2	H32MX APB1 - user interfaces	X		X		X	X					
3	EBI CS0..CS3	X	X		X		X					
	NFC Command Register	X	X		X							
4	NFC SRAM	X	X		X							
5	UDPHS RAM	X				X						
	UHP OHCI Reg	X				X						
	UHP EHCI Reg	X				X						

17.5 Memory Mapping

The Bus Matrix provides one decoder for every AHB master interface. The decoder offers each AHB master several memory mappings. Each memory area may be assigned to several slaves. Booting at the same address while using different AHB slaves (i.e., external RAM, internal ROM or internal Flash, etc.) becomes possible.

17.6 Special Bus Granting Mechanism

The Bus Matrix provides some speculative bus granting techniques in order to anticipate access requests from masters. This mechanism reduces latency at first access of a burst, or for a single transfer, as long as the slave is free from any other master access. It does not provide any benefit if the slave is continuously accessed by more than one master, since arbitration is pipelined and has no negative effect on the slave bandwidth or access latency.

This bus granting mechanism sets a different default master for every slave.

At the end of the current access, if no other request is pending, the slave remains connected to its associated default master. A slave can be associated with three kinds of default masters:

- No default master
- Last access master
- Fixed default master

To change from one type of default master to another, the Bus Matrix user interface provides Slave Configuration Registers, one for every slave, which set a default master for each slave. The Slave Configuration Register contains two fields to manage master selection: DEFMSTR_TYPE and FIXED_DEFMSTR. The 2-bit DEFMSTR_TYPE field selects the default master type (no default, last access master, fixed default master), whereas the 4-bit FIXED_DEFMSTR field selects a fixed default master provided that DEFMSTR_TYPE is set to fixed default master. Refer to Section 17.13.2 “Bus Matrix Slave Configuration Registers”.

17.7 No Default Master

After the end of the current access, if no other request is pending, the slave is disconnected from all masters.

This configuration incurs one latency clock cycle for the first access of a burst after bus Idle. Arbitration without default master may be used for masters that perform significant bursts or several transfers with no Idle in between, or if the slave bus bandwidth is widely used by one or more masters.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput regardless of the number of requesting masters.

17.8 Last Access Master

After the end of the current access, if no other request is pending, the slave remains connected to the last master that performed an access request.

This allows the Bus Matrix to remove the one latency cycle for the last master that accessed the slave. Other nonprivileged masters still get one latency clock cycle if they need to access the same slave. This technique is used for masters that mainly perform single accesses or short bursts with some Idle cycles in between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput whatever is the number of requesting masters.

17.9 Fixed Default Master

After the end of the current access, if no other request is pending, the slave connects to its fixed default master. Unlike the last access master, the fixed default master does not change unless the user modifies it by software (FIXED_DEFMSTR field of the related MATRIX_SCFG).

This allows the Bus Matrix arbiters to remove the one latency clock cycle for the fixed default master of the slave. All requests attempted by the fixed default master do not cause any arbitration latency, whereas other non-privileged masters will get one latency cycle. This technique is used for a master that mainly performs single accesses or short bursts with Idle cycles in between.

This configuration provides no benefit on access latency or bandwidth when reaching maximum slave bus throughput, regardless of the number of requesting masters.

17.10 Arbitration

The Bus Matrix provides an arbitration mechanism that reduces latency when conflicts occur, i.e., when two or more masters try to access the same slave at the same time. One arbiter per AHB slave is provided, thus arbitrating each slave specifically.

The Bus Matrix provides the user with the possibility of choosing between two arbitration types or mixing them for each slave:

- Round-robin Arbitration (default)
- Fixed Priority Arbitration

The resulting algorithm may be complemented by selecting a default master configuration for each slave.

When re-arbitration must be done, specific conditions apply. See Section 17.10.1 “Arbitration Scheduling”.

17.10.1 Arbitration Scheduling

Each arbiter has the ability to arbitrate between two or more master requests. In order to avoid burst breaking and also to provide the maximum throughput for slave interfaces, arbitration may only take place during the following cycles:

- Idle Cycles: when a slave is not connected to any master or is connected to a master which is not currently accessing it.
- Single Cycles: when a slave is currently performing a single access.
- End of Burst Cycles: when the current cycle is the last cycle of a burst transfer. For defined burst length, predicted end of burst matches the size of the transfer but is managed differently for undefined burst length. See Section 17.10.1.1 “Undefined Length Burst Arbitration”.
- Slot Cycle Limit: when the slot cycle counter has reached the limit value indicating that the current master access is too long and must be broken. See Section 17.10.1.2 “Slot Cycle Limit Arbitration”.

17.10.1.1 Undefined Length Burst Arbitration

In order to prevent long AHB burst lengths that can lock the access to the slave for an excessive period of time, the user can trigger the re-arbitration before the end of the incremental bursts. The re-arbitration period can be selected from the following Undefined Length Burst Type (ULBT) possibilities:

- Unlimited: no predetermined end of burst is generated. This value enables 1 Kbyte burst lengths.
- 1-beat bursts: predetermined end of burst is generated at each single transfer during the INCR transfer.
- 4-beat bursts: predetermined end of burst is generated at the end of each 4-beat boundary during INCR transfer.
- 8-beat bursts: predetermined end of burst is generated at the end of each 8-beat boundary during INCR transfer.
- 16-beat bursts: predetermined end of burst is generated at the end of each 16-beat boundary during INCR transfer.
- 32-beat bursts: predetermined end of burst is generated at the end of each 32-beat boundary during INCR transfer.
- 64-beat bursts: predetermined end of burst is generated at the end of each 64-beat boundary during INCR transfer.
- 128-beat bursts: predetermined end of burst is generated at the end of each 128-beat boundary during INCR transfer.

The use of undefined length 8-beat bursts, or less, is discouraged since this may decrease the overall bus bandwidth due to arbitration and slave latencies at each first access of a burst.

However, if the usual length of undefined length bursts is known for a master it is recommended to configure the ULBT accordingly.

This selection can be done through the ULBT field of the Master Configuration Registers (MATRIX_MCFG).

17.10.1.2 Slot Cycle Limit Arbitration

The Bus Matrix contains specific logic to break long accesses, such as very long bursts on a very slow slave (e.g., an external low speed memory). At each arbitration time, a counter is loaded with the value previously written in the SLOT_CYCLE field of the related Slave Configuration Register (MATRIX_SCFG) and decreased at each clock cycle. When the counter elapses, the arbiter has the ability to re-arbitrate at the end of the current AHB access cycle.

Unless a master has a very tight access latency constraint, which could lead to data overflow or underflow due to a badly undersized internal FIFO with respect to its throughput, the Slot Cycle Limit should be disabled (SLOT_CYCLE = 0) or set to its default maximum value in order not to inefficiently break long bursts performed by some Atmel masters.

In most cases, this feature is not needed and should be disabled for power saving.

Warning: This feature cannot prevent any slave from locking its access indefinitely.

17.10.2 Arbitration Priority Scheme

The bus Matrix arbitration scheme is organized in priority pools, each corresponding to an access criticality class as shown in the "Latency Quality of Service" column in Table 17-7.

Table 17-7. Arbitration Priority Pools

Priority pool	Latency Quality of Service
3	Latency Critical
2	Latency Sensitive
1	Bandwidth Sensitive
0	Background Transfers

Round-robin priority is used in the highest and lowest priority pools 3 and 0, whereas fixed level priority is used between priority pools and in the intermediate priority pools 2 and 1. See Section 17.10.2.2 "Round-robin Arbitration".

For each slave, each master is assigned to one of the slave priority pools through the priority registers for slaves (MxPR fields of MATRIX_PRAS and MATRIX_PRBS). When evaluating master requests, this priority pool level always takes precedence.

After reset, most of the masters belong to the lowest priority pool (MxPR = 0, Background Transfer) and are therefore granted bus access in a true round-robin order.

The highest priority pool must be specifically reserved for masters requiring very low access latency. If more than one master belongs to this pool, they will be granted bus access in a biased round-robin manner which allows tight and deterministic maximum access latency from AHB requests. In the worst case, any currently occurring high-priority master request will be granted after the current bus master access has ended and other high priority pool master requests, if any, have been granted once each.

The lowest priority pool shares the remaining bus bandwidth between AHB masters.

Intermediate priority pools allow fine priority tuning. Typically, a latency-sensitive master or a bandwidth-sensitive master will use such a priority level. The higher the priority level (MxPR value), the higher the master priority.

For good CPU performance, it is recommended configure CPU priority with the default reset value 2 (Latency Sensitive).

All combinations of MxPR values are allowed for all masters and slaves. For example, some masters might be assigned the highest priority pool (round-robin), and remaining masters the lowest priority pool (round-robin), with no master for intermediate fixed priority levels.

17.10.2.1 Fixed Priority Arbitration

Fixed priority arbitration algorithm is the first and only arbitration algorithm applied between masters from distinct priority pools. It is also used in priority pools other than the highest and lowest priority pools (intermediate priority pools).

Fixed priority arbitration allows the Bus Matrix arbiters to dispatch the requests from different masters to the same slave by using the fixed priority defined by the user in the MxPR field for each master in the Priority Registers, MATRIX_PRAS and MATRIX_PRBS. If two or more master requests are active at the same time, the master with the highest priority MxPR number is serviced first.

In intermediate priority pools, if two or more master requests with the same priority are active at the same time, the master with the highest number is serviced first.

17.10.2.2 Round-robin Arbitration

This algorithm is only used in the highest and lowest priority pools. It allows the Bus Matrix arbiters to properly dispatch requests from different masters to the same slave. If two or more master requests are active at the same time in the priority pool, they are serviced in a round-robin increasing master number order.

17.11 Register Write Protection

To prevent any single software error from corrupting Bus Matrix behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the Write Protection Mode Register (MATRIX_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the Write Protection Status Register (MATRIX_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is reset by writing the Bus Matrix Write Protect Mode Register (MATRIX_WPMR) with the appropriate access key WPKEY.

The following registers can be write-protected:

- Bus Matrix Master Configuration Registers
- Bus Matrix Slave Configuration Registers
- Bus Matrix Priority Registers A For Slaves
- Bus Matrix Priority Registers B For Slaves
- Master Error Interrupt Enable Register
- Master Error Interrupt Disable Register
- Security Slave Registers
- Security Areas Split Slave Registers
- Security Region Top Slave Registers
- Security Peripheral Select x Registers

17.12 TrustZone Extension to AHB and APB

TrustZone secure software is supported through the filtering of each slave access with master security bit AMBA hprot[6] extension signals.

The TrustZone extension adds the ability to manage the access rights for Secure and Non-Secure accesses. The access rights are defined through the hardware and software configuration of the device. The operating mode is as follows:

- With the TrustZone extension, the Bus Masters transmit requests with the Secure or Non-Secure Security option.
- The AHB Matrix, according to its configuration and the request, grants or denies the access.

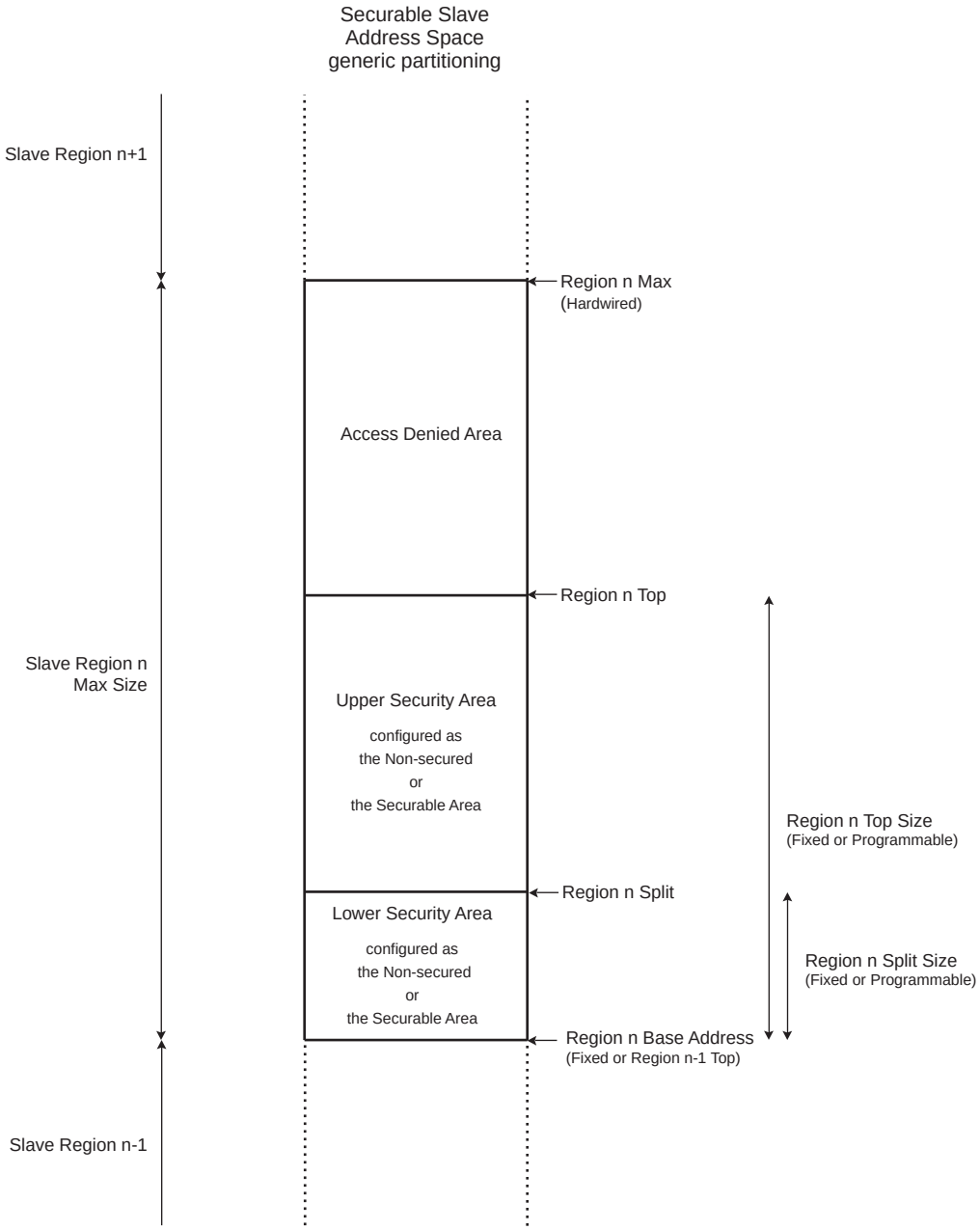
The slave address space is divided into one or more slave regions. The slave regions are generally contiguous parts of the slave address space. The slave region is potentially split into an access denied area (upper part) and a security region which can be split (lower part), unless the slave security region occupies the whole slave region. The security region itself may or may not be split into one Secured area and one Non-Secured area. The Secured area may be independently secured for read access and for write access.

For one slave region, the following characteristics are configured by hardware or software:

- Base Address of the slave region
- Max Size of the slave region: the maximum size for the region's physical content
- Top Size of the slave security region: the actually programmed or fixed size for the region's physical content
- Split Size of the slave security region: the size of the lower security area of the region.

Figure 17-1 shows how the terms defined here are implemented in an AHB slave address space.

Figure 17-1. Generic Partitioning of the AHB Slave Address Space



A set of Bus Matrix security registers allows to specify, for each AHB slave, slave security region or slave security area, the security mode required for accessing this slave, slave security region or slave security area.

Additional Bus Matrix security registers allow to specify, for each APB slave, the security mode required for accessing this slave (see Section 17.13.15 “Security Peripheral Select x Registers”).

See Section 17.13.12 “Security Slave Registers”.

The Bus Matrix registers can only be accessed in Secure Mode.

The Bus Matrix propagates the AHB security bit down to the AHB slaves to let them perform additional security checks, and the Bus Matrix itself allows, or not, the access to the slaves by means of its TrustZone embedded controller.

Access violations may be reported either by an AHB slave through the bus error response (example from the AHB/APB Bridge), or by the Bus Matrix embedded TrustZone controller. In both cases, a bus error response is sent to the offending master and the error is flagged in the Master Error Status Register. An interrupt can be sent to the Secure world, if it has been enabled for that master by writing into the Master Error Interrupt Enable Register. Thus, the offending master is identified. The offending address is registered in the Master Error Address Registers, so that the slave and the targeted security region are also known.

Depending on the hardware parameters and software configuration, the address space of each AHB slave security region may or may not be split into two parts, one belonging to the Secure world and the other one to the Normal world.

Five different security types of AHB slaves are supported. The number of security regions is set by design for each slave, independently, from 1 to 8, totalling from 1 up to 16 security areas for security configurable slaves.

17.12.1 Security Types of AHB Slaves

17.12.1.1 Principles

The Bus Matrix supports five different security types of AHB slaves: two fixed types and three configurable types. The security type of an AHB slave is set at hardware design among the following:

- Always Non-Secured
- Always Secured
- Internal Securable
- External Securable
- Scalable Securable

The security type is set at hardware design on a per-master and a per-slave basis. **Always Non-Secured** and **Always Secured** security types are not software configurable.

The different security types have the following characteristics:

- **Always Non-Secured** slaves have no security mode access restriction. Their address space is precisely set by design. Any out-of-address range access is denied and reported.
- **Always Secured** slaves can only be accessed by a secure master request. Their address space is precisely set by design. Any non-secure or out-of-address range access is denied and reported.
- **Internal Securable** is intended for internal memories such as RAM, ROM or embedded Flash. The Internal Securable slave has one slave region which has a hardware fixed base address and Security Region Top. This slave region may be split through software configuration into one Non-Secured area plus one Secured area. Inside the slave security region, the split boundary is programmable in powers of 2 from 4 Kbytes up to the full slave security region address space. The security area located below the split boundary may be configured as the Non-Secured or the Secured one. The Securable area may be independently configured as Read Secured and/or Write Secured. Any access with security or address range violation is denied and reported.
- **External Securable** is intended for external memories on the EBI, such as DDR, SDRAM, external ROM or NAND Flash. The External Securable slave has identical features as the Internal Securable slave, plus the ability to configure each of its slave security region address space sizes according to the external memory parts used. This avoids mirroring Secured areas into Non-secured areas, and further restricts the overall accessible address range. Any access with security or configured address range violation is denied and reported.
- **Scalable Securable** is intended for external memories with a dedicated slave, such as DDR. The Scalable Securable slave is divided into a fixed number of scalable, equally sized, and contiguous security regions. Each of them can be split in the same way as for Internal or External Securable slaves. The security region size must be configured by software, so that the equally-sized regions fill the actual available memory. This

avoids mirroring Secured areas into Non-secured areas, and further restricts the overall accessible address range. Any access with security or configured address range violation is denied and reported.

As the security type is set at hardware design on a per-master and per-slave basis, it is possible to set some slave access security as configurable from one or some particular masters, and to set the access as Always Secured from all the other masters.

As the security type is set by design at the slave region level, different security region types can be mixed inside a single slave.

Likewise, the mapping base address and the accessible address range of each AHB slave or slave region may have been hardware-restricted on a per-master basis from no access to full slave address space.

17.12.1.2 Examples

Table 17-8 shows an example of Security Type settings.

Table 17-8. Security Type Setting Example

Slave	Master0	Master1	Master2
Slave0 Internal Memory	Always Non-Secured	Internal Securable 1 region	Internal Securable 1 region
Slave1 EBI	External Securable 2 regions	Always Secured	External Securable 2 regions

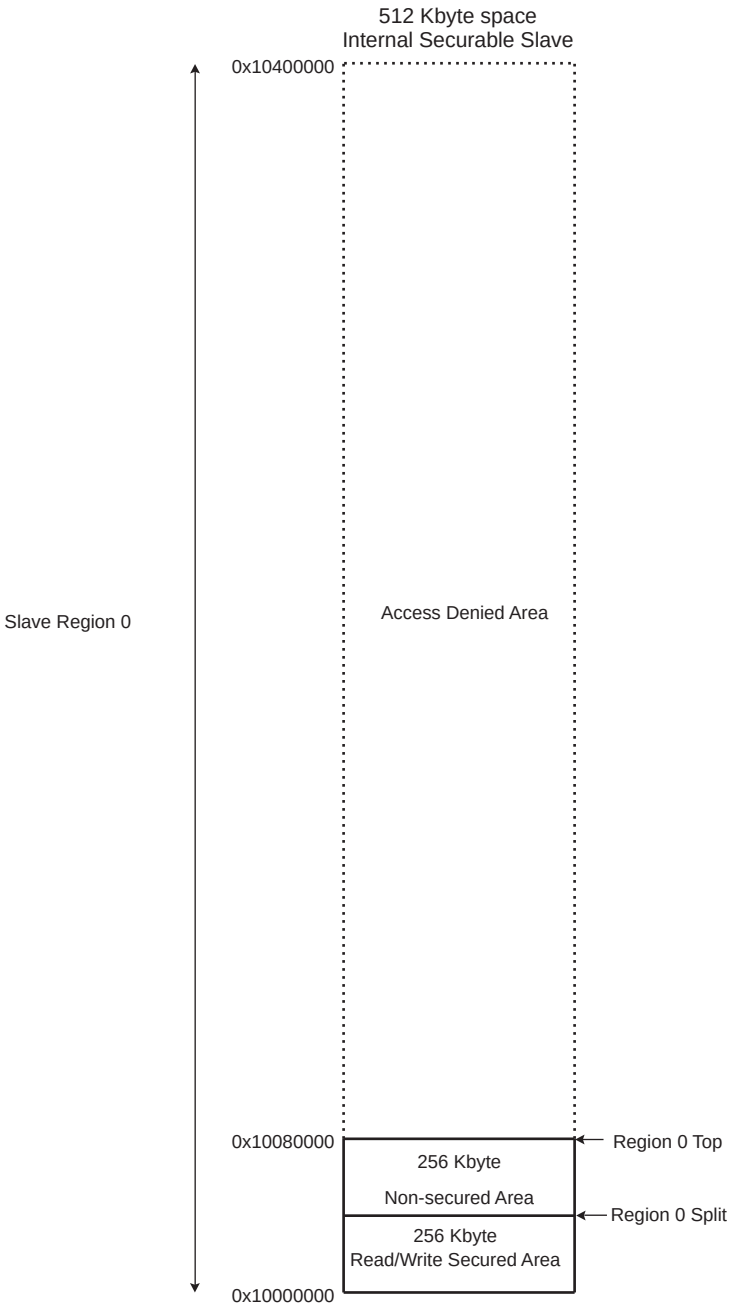
This example is constructed with the following characteristics:

- Slave0 is an Internal Memory containing one region:
  - The Access from Master0 to Slave0 is Always Non-Secured
  - The Access from Master1 and Master2 to Slave0 is Internal Securable with one region and with the same Software Configuration (Choice of SPLIT0 and the Security Configuration bits LANSECH, RDNSECH, WRNSECH).
- Slave1 is an EBI containing two regions:
  - The Access from Master1 to Slave1 is Always Secured
  - The Access from Master0 and Master2 to Slave1 is External Securable with two regions and with the same Software Configuration (Choice of TOP0, TOP1, SPLIT0, SPLIT1 and the Security Configuration bits LANSECH, RDNSECH, WRNSECH).

Figure 17-2 shows an Internal Securable slave example. This example is constructed with the following hypothesis:

- The slave is an Internal Memory containing one region. The Slave region Max Size is 4 Mbytes.
- The slave region 0 base address equals 0x10000000. Its Top Size is 512 Kbytes (hardware configuration).
- The slave software configuration is:
  - SPLIT0 is set to 256 Kbytes
  - LANSECH0 is set to 0, the low area of region 0 is the securable one
  - RDNSECH0 is set to 0, region 0 Securable area is secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes

Figure 17-2. Partitioning Example of an Internal Securable Slave Featuring 1 Security Region of 512 KB Split into 1 or 2 Security Areas of 4 KB to 512 KB

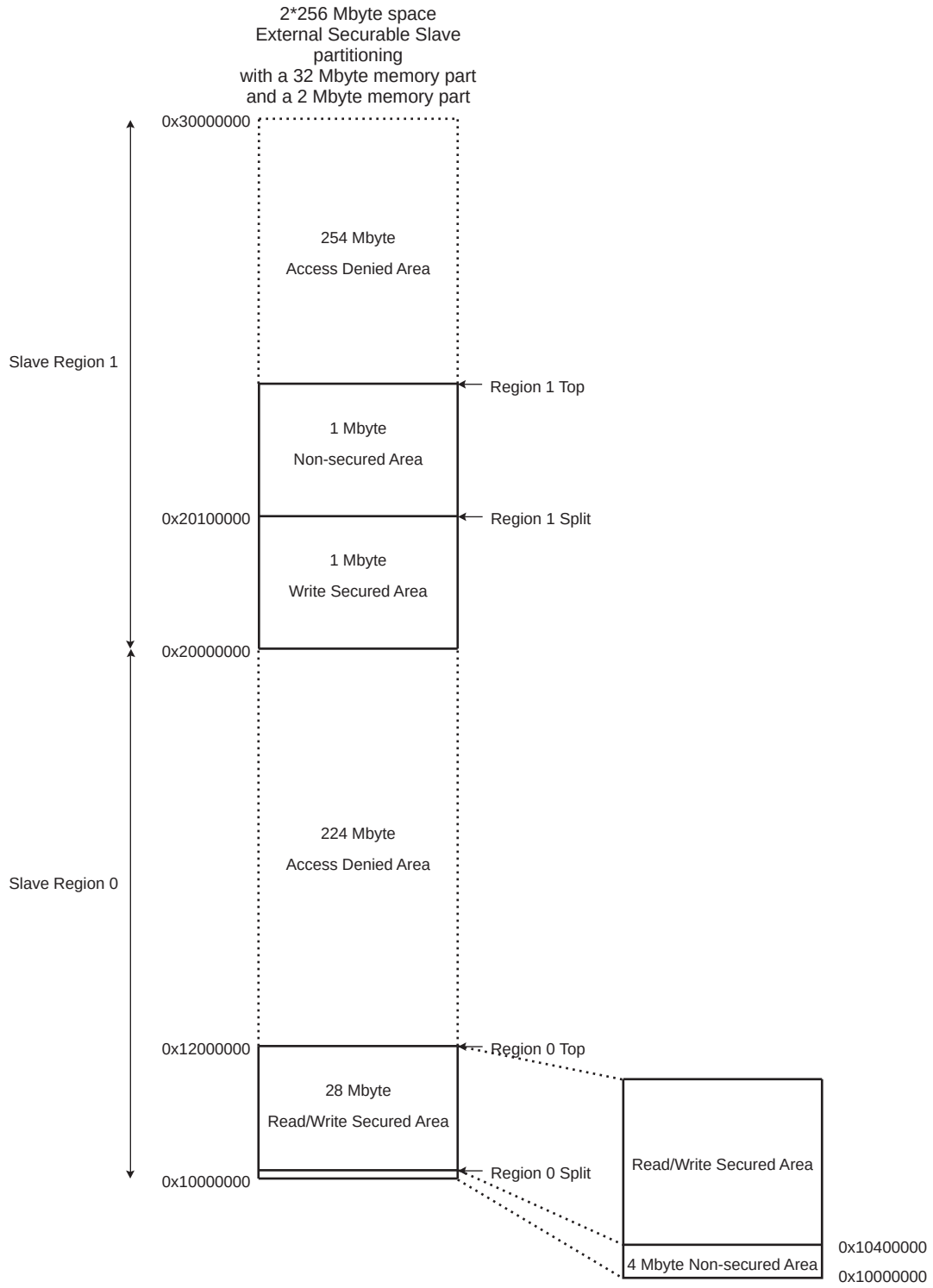


Note: The slave security areas split inside the security region are configured by writing into the Security Areas Split Slave Registers.

Figure 17-3 shows an External Securable slave example. This example is constructed with the following hypothesis:

- The slave is an interface with the external bus (EBI) containing two regions. The slave size is 2 × 256 Mbytes. Each slave region Max Size is 256 Mbytes.
- The slave region 0 base address equals 0x10000000. It is connected to a 32 Mbyte memory, for example an external DDR. The slave region 0 Top Size must be set to 32 Mbytes.
- The slave region 1 base address equals 0x20000000. It is connected to a 2 Mbyte memory, for example an external NAND Flash. The slave region 1 Top Size must be set to 2 Mbytes.
- The slave software configuration is:
  - TOP0 is set to 32 Mbytes
  - TOP1 is set to 2 Mbytes
  - SPLIT0 is set to 4 Mbytes
  - SPLIT1 is set to 1 Mbyte
  - LANSECH0 is set to 1, the low area of region 0 is the non-securable one
  - RDNSECH0 is set to 0, region 0 Securable area is secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes
  - LANSECH1 is set to 0, the low area of region 1 is the Securable one
  - RDNSECH1 is set to 1, region 1 Securable area is non-secured for reads
  - WRNSECH1 is set to 0, region 1 Securable area is secured for writes

Figure 17-3. Partitioning Example of an External Securable Slave Featuring 2 Security Regions of 4 KB to 128 MB each and up to 4 Security Areas of 4 KB to 128 MB

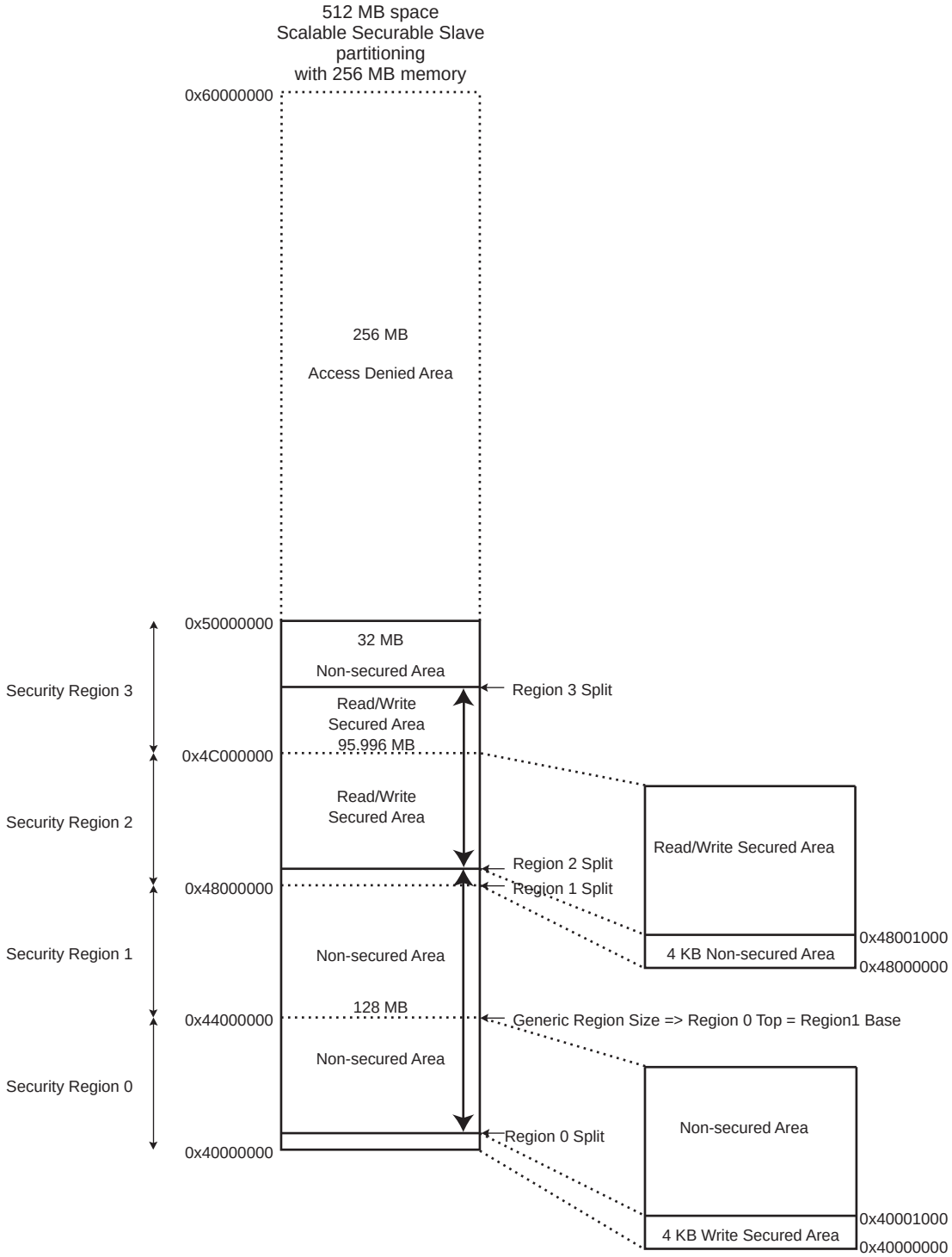


Note: The slave region sizes are configured by writing into the Security Region Top Slave Registers.
The slave security area split inside each region is configured by writing into the Security Areas Split Slave Registers.

Figure 17-4 shows a Scalable Securable slave example. This example is constructed with the following hypothesis:

- The slave is an external memory with dedicated slave containing four regions, for example an external DDR.
- The slave size is 512 Mbytes.
- The slave base address equals 0x40000000. It is connected to a 256 Mbyte external memory.
- As the connected memory size is 256 Mbytes and there are four regions, the size of each region is 64 Mbytes. This gives the value of the slave region Max Size and Top Size. The slave region 0 Top Size must be configured to 64 Mbytes.
- The slave software configuration is:
  - TOP0 is set to 64 Mbytes
  - SPLIT0 is set to 4 Kbytes
  - SPLIT1 is set to 64 Mbytes, so its low area occupies the whole region 1
  - SPLIT2 is set to 4 Kbytes
  - SPLIT3 is set to 32 Mbytes
  - LANSECH0 is set to 0, the low area of region 0 is the Securable one
  - RDNSECH0 is set to 1, region 0 Securable area is non-secured for reads
  - WRNSECH0 is set to 0, region 0 Securable area is secured for writes
  - LANSECH1 is set to 1, the low area of region 1 is the non-securable one
  - RDNSECH1 is ‘don’t care’ since the low area occupies the whole region 1
  - WRNSECH1 is ‘don’t care’ since the low area occupies the whole region 1
  - LANSECH2 is set to 1, the low area of region 2 is the non-securable one
  - RDNSECH2 is set to 0, region 2 Securable area is secured for reads
  - WRNSECH2 is set to 0, region 2 Securable area is secured for writes
  - LANSECH3 is set to 0, the low area of region 3 is the Securable one
  - RDNSECH3 is set to 0, region 3 Securable area is secured for reads
  - WRNSECH3 is set to 0, region 3 Securable area is secured for writes

Figure 17-4. Partitioning Example of a Scalable Securable Slave Featuring 4 Equally-sized Security Regions of 1 MB to 128 MB each and up to 8 Security Areas of 4 KB to 128 MB



Note: The slaves' generic security regions sizes are configured by writing into field SRTOP0 of the Security Region Top Slave Registers and the custom slave security areas splits inside each region is configured by writing into the Security Areas Split Slave Registers.

17.12.2 Security of APB Slaves

The security type of an APB slave is set at hardware design among the following:

- Peripheral Always Secured (PAS)
- Peripheral Always Non-Secured (PNS)
- Peripheral Securable (PS)

To be able to configure the security mode required for accessing a particular APB slave connected to the AHB/APB Bridge, the Bus Matrix features three 32-bit Security Peripheral Select x Registers. Some of these bits may have been set to a Secured or a Non-Secured value by design, whereas others are programmed by software (see Section 17.13.15 “Security Peripheral Select x Registers”).

Peripheral security state, “Secure” or “Non-Secure” is an AND operation between H32MX MATRIX_SPSELRx and H64MX MATRIX_SPSELRx for the bit corresponding to the peripheral.

As a general rule:

- The peripheral security state is applied to the corresponding peripheral interrupt line. Exceptions may occur on some peripherals (PIO Controller, etc.). In such case, refer to the peripheral description.
- The peripheral security state is applied to the peripheral master part, if any. Exceptions may occur on some peripherals. In such case, refer to the peripheral description. See Section 17.12.3 “Security Types of AHB Master Peripherals”.

MATRIX_SPSELRx bits in the H32MX or H64MX user interface are respectively read/write or read-only to ‘1’ depending on whether the peripheral is connected or not, on the Matrix.

All bit values in Table 17-9 except those marked ‘UD’ (User Defined) are read-only and cannot be changed. Values marked ‘UD’ can be changed. Refer to the following examples.

- Example for GMAC, Peripheral ID 5, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR1[5] = 1 (read-only); no influence on the security configuration
  - H32MX MATRIX_SPSELR1[5] can be written by user to program the security.
- Example for LCDC, Peripheral ID 45, which is connected to the H64MX Matrix
  - H64MX MATRIX_SPSELR2[13] can be written by user to program the security.
  - H32MX MATRIX_SPSELR2[13] = 1 (read-only); no influence on the security configuration
- Example for AIC, Peripheral ID 49, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR2[17] = 1 (read-only); sets the peripheral as Non-Secure by hardware, also called “Peripheral Always Non-Secured”
  - H32MX MATRIX_SPSELR2[17] = 1 (read-only); no influence on the security configuration
- Example for SAIC, Peripheral ID 0, which is connected to the H32MX Matrix
  - H64MX MATRIX_SPSELR1[0] = 1 (read-only); no influence on the security configuration
  - H32MX MATRIX_SPSELR1[0] = 0 (read-only); sets the peripheral as Secure by hardware, also called “Peripheral Always Secured”

Table 17-9. Peripheral Identifiers

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
0	SAIC	Peripheral Always Secured (PAS)		MATRIX_SPSELR1[0]	0	1
1	–	–	–	–	–	–
2	ARM	Peripheral Securable (PS)	H64	MATRIX_SPSELR1[2]	1	UD
3	PIT	PS	H32	MATRIX_SPSELR1[3]	UD	1
4	WDT	PS	H32	MATRIX_SPSELR1[4]	UD	1

Table 17-9. Peripheral Identifiers (Continued)

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
5	GMAC	PS	H32	MATRIX_SPSELR1[5]	UD	1
6	XDMAC0	PS	H64	MATRIX_SPSELR1[6]	1	UD
7	XDMAC1	PS	H64	MATRIX_SPSELR1[7]	1	UD
8	ICM	PS	H32	MATRIX_SPSELR1[8]	UD	1
9	AES	PS	H64	MATRIX_SPSELR1[9]	1	UD
10	AESB	PS	H64	MATRIX_SPSELR1[10]	1	UD
11	TDES	PS	H32	MATRIX_SPSELR1[11]	UD	1
12	SHA	PS	H64	MATRIX_SPSELR1[12]	1	UD
13	MPDDRC	PS	H64	MATRIX_SPSELR1[13]	1	UD
14	MATRIX1	PAS	H32	MATRIX_SPSELR1[14]	0	1
15	MATRIX0	PAS	H64	MATRIX_SPSELR1[15]	1	0
16	SECUMOD	PAS	H32	MATRIX_SPSELR1[16]	0	1
17	HSMC	PS	H32	MATRIX_SPSELR1[17]	UD	1
18	PIOA	PAS	H32	MATRIX_SPSELR1[18]	0	1
19	FLEXCOM0	PS	H32	MATRIX_SPSELR1[19]	UD	1
20	FLEXCOM1	PS	H32	MATRIX_SPSELR1[20]	UD	1
21	FLEXCOM2	PS	H32	MATRIX_SPSELR1[21]	UD	1
22	FLEXCOM3	PS	H32	MATRIX_SPSELR1[22]	UD	1
23	FLEXCOM4	PS	H32	MATRIX_SPSELR1[23]	UD	1
24	UART0	PS	H32	MATRIX_SPSELR1[24]	UD	1
25	UART1	PS	H32	MATRIX_SPSELR1[25]	UD	1
26	UART2	PS	H32	MATRIX_SPSELR1[26]	UD	1
27	UART3	PS	H32	MATRIX_SPSELR1[27]	UD	1
28	UART4	PS	H32	MATRIX_SPSELR1[28]	UD	1
29	TWIHS0	PS	H32	MATRIX_SPSELR1[29]	UD	1
30	TWIHS1	PS	H32	MATRIX_SPSELR1[30]	UD	1
31	SDMMC0	PS	H64	MATRIX_SPSELR1[31]	1	UD
32	SDMMC1	PS	H64	MATRIX_SPSELR2[0]	1	UD
33	SPI0	PS	H32	MATRIX_SPSELR2[1]	UD	1
34	SPI1	PS	H32	MATRIX_SPSELR2[2]	UD	1
35	TC0	PS	H32	MATRIX_SPSELR2[3]	UD	1
36	TC1	PS	H32	MATRIX_SPSELR2[4]	UD	1
37	–	–	–	–	–	–
38	PWM	PS	H32	MATRIX_SPSELR2[6]	UD	1
39	–	–	–	–	–	–
40	ADC	PS	H32	MATRIX_SPSELR2[8]	UD	1

Table 17-9. Peripheral Identifiers (Continued)

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
41	UHPHS	PS	H32	MATRIX_SPSELR2[9]	UD	1
42	UDPHS	PS	H32	MATRIX_SPSELR2[10]	UD	1
43	SSC0	PS	H32	MATRIX_SPSELR2[11]	UD	1
44	SSC1	PS	H32	MATRIX_SPSELR2[12]	UD	1
45	LCDC	PS	H64	MATRIX_SPSELR2[13]	1	UD
46	ISC	PS	H64	MATRIX_SPSELR2[14]	1	UD
47	TRNG	PS	H32	MATRIX_SPSELR2[15]	UD	1
48	PDMIC	PS	H32	MATRIX_SPSELR2[16]	UD	1
49	AIC	Peripheral Always Non-Secured (PNS)	H32	MATRIX_SPSELR2[17]	1	1
50	SFC	PS	H32	MATRIX_SPSELR2[18]	UD	1
51	SECURAM	PAS	H32	MATRIX_SPSELR2[19]	0	1
52	QSPI0	PS	H64	MATRIX_SPSELR2[20]	1	UD
53	QSPI1	PS	H64	MATRIX_SPSELR2[21]	1	UD
54	I2SC0	PS	H32	MATRIX_SPSELR2[22]	UD	1
55	I2SC1	PS	H32	MATRIX_SPSELR2[23]	UD	1
56	CAN0	PS	H32	MATRIX_SPSELR2[24]	UD	1
57	CAN1	PS	H32	MATRIX_SPSELR2[25]	UD	1
58	–	–	–	–	–	–
59	CLASSD	PS	H32	MATRIX_SPSELR2[27]	UD	1
60	SFR	PS	H32	MATRIX_SPSELR2[28]	UD	1
61	SAIC	PAS	H32	MATRIX_SPSELR2[29]	0	1
62	AIC	PNS	H32	MATRIX_SPSELR2[30]	1	1
63	L2CC	PS	H64	MATRIX_SPSELR2[31]	1	UD
64	CAN0	PS	H32	MATRIX_SPSELR3[0]	UD	1
65	CAN1	PS	H32	MATRIX_SPSELR3[1]	UD	1
66	GMAC	PS	H32	MATRIX_SPSELR3[2]	UD	1
67	GMAC	PS	H32	MATRIX_SPSELR3[3]	UD	1
68	PIOB	PAS	H32	MATRIX_SPSELR3[4]	0	1
69	PIOC	PAS	H32	MATRIX_SPSELR3[5]	0	1
70	PIOD	PAS	H32	MATRIX_SPSELR3[6]	0	1
71	SDMMC0	PS	H32	MATRIX_SPSELR3[7]	UD	1
72	SDMMC1	PS	H32	MATRIX_SPSELR3[8]	UD	1
73	–	–	–	–	–	–
74	RTC, RSTC, PMC	PS	H32	MATRIX_SPSELR3[9]	UD	1
75	ACC	PS	H32	MATRIX_SPSELR3[10]	UD	1

Table 17-9. Peripheral Identifiers (Continued)

ID	Peripheral	Security Type	Matrix	MATRIX_SPSELRx Bit	Bit Value in H32MX	Bit Value in H64MX
76	RXLP	PS	H32	MATRIX_SPSELR3[11]	UD	1
77	SFRBU	PS	H32	MATRIX_SPSELR3[12]	UD	1
78	CHIPID	PS	H32	MATRIX_SPSELR3[13]	UD	1

The AHB/APB Bridge compares the incoming master request security bit with the required security mode for the selected peripheral, and accepts or denies access. In the last case, its bus error response is internally flagged in the Bus Matrix Master Error Status Register; the offending address is registered in the Master Error Address Registers so that the slave and the targeted protected region are also known.

17.12.3 Security Types of AHB Master Peripherals

Master AHB peripherals send requests on the AHB matrix with a security attribute that depends on:

- The master security type: the master security type is identical to the security type of the IP peripheral slave user interface.
- Possibly for some masters with one or more channels: it may be possible to choose the TrustZone security attribute for each channel. If this is the case, refer to the peripheral's user interface description.

When the peripheral security type is Peripheral Securable, the slave security configuration applies to the peripheral master AHB part.

17.12.4 Security Types of AHB Slave Peripherals

In this particular case, the AHB interface is connected to one or more peripheral user interfaces.

The type is Internal Securable to Peripheral (ISP).

Important: in this case, each region in the "Internal Securable to Peripheral" AHB slave type must be programmed with the following characteristics:

- The region must be programmed to be entirely secure or entirely nonsecure. This is done with:
  - The split offset must be equal to the maximum size of 128 Mbytes so that the whole peripheral user interface is in the low area below the split. Code sample:
    MATRIX_SASSRx.SASPLITy = 0xF
  - The bits WRNSECH and RDNSECH must be set respectively to 0 = "write secure" and 0 = "read secure". Code sample:
    MATRIX_SSRx.WRNSECHy = 0; MATRIX_SSRx.RDNSECHy = 0;
  - To set the peripheral to "secure": the bit LANSECHy must be set to 0 (low area according to RDNSECH0 and WRNSECH0, hence secure).
  - To set the peripheral to "nonsecure": the bit LANSECHy must be set to 1 (low area is non-secure).

Note: The MATRIX_SRTSRx register is not applicable for the "Internal Securable to Peripheral" type.

- The Security Peripheral Select Registers must be set to the same security attributes for the corresponding Peripheral identifiers: SPSELRx.NSECPy.

17.13 AHB Matrix (MATRIX) User Interface

The user interface below is constructed with the maximum numbers of masters, slaves and regions by slave that are possible on the two product matrixes. The exact number of these elements must be used to deduce the exact register description of the matrix user interface.

The exact numbers of these elements can be found in:

- Section 17.3 “MATRIX0 (H64MX)” for MATRIX0 (H64MX)
- Section 17.4 “MATRIX1 (H32MX)” for MATRIX1 (H32MX)

Table 17-10. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Master Configuration Register 0	MATRIX_MCFG0	Read/Write	0x00000004
0x0004	Master Configuration Register 1	MATRIX_MCFG1	Read/Write	0x00000004
0x0008	Master Configuration Register 2	MATRIX_MCFG2	Read/Write	0x00000004
0x000C	Master Configuration Register 3	MATRIX_MCFG3	Read/Write	0x00000004
0x0010	Master Configuration Register 4	MATRIX_MCFG4	Read/Write	0x00000004
0x0014	Master Configuration Register 5	MATRIX_MCFG5	Read/Write	0x00000004
0x0018	Master Configuration Register 6	MATRIX_MCFG6	Read/Write	0x00000004
0x001C	Master Configuration Register 7	MATRIX_MCFG7	Read/Write	0x00000004
0x0020	Master Configuration Register 8	MATRIX_MCFG8	Read/Write	0x00000004
0x0024	Master Configuration Register 9	MATRIX_MCFG9	Read/Write	0x00000004
0x0028	Master Configuration Register 10	MATRIX_MCFG10	Read/Write	0x00000004
0x002C	Master Configuration Register 11	MATRIX_MCFG11	Read/Write	0x00000004
0x0030–0x003C	Reserved	–	–	–
0x0040	Slave Configuration Register 0	MATRIX_SCFG0	Read/Write	0x000001FF
0x0044	Slave Configuration Register 1	MATRIX_SCFG1	Read/Write	0x000001FF
0x0048	Slave Configuration Register 2	MATRIX_SCFG2	Read/Write	0x000001FF
0x004C	Slave Configuration Register 3	MATRIX_SCFG3	Read/Write	0x000001FF
0x0050	Slave Configuration Register 4	MATRIX_SCFG4	Read/Write	0x000001FF
0x0054	Slave Configuration Register 5	MATRIX_SCFG5	Read/Write	0x000001FF
0x0058	Slave Configuration Register 6	MATRIX_SCFG6	Read/Write	0x000001FF
0x005C	Slave Configuration Register 7	MATRIX_SCFG7	Read/Write	0x000001FF
0x0060	Slave Configuration Register 8	MATRIX_SCFG8	Read/Write	0x000001FF
0x0064	Slave Configuration Register 9	MATRIX_SCFG9	Read/Write	0x000001FF
0x0068	Slave Configuration Register 10	MATRIX_SCFG10	Read/Write	0x000001FF
0x006C	Slave Configuration Register 11	MATRIX_SCFG11	Read/Write	0x000001FF
0x0070	Slave Configuration Register 12	MATRIX_SCFG12	Read/Write	0x000001FF
0x0074	Slave Configuration Register 13	MATRIX_SCFG13	Read/Write	0x000001FF
0x0078	Slave Configuration Register 14	MATRIX_SCFG14	Read/Write	0x000001FF
0x007C	Reserved	–	–	–
0x0080	Priority Register A for Slave 0	MATRIX_PRAS0	Read/Write	0x00000000

Table 17-10. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0084	Priority Register B for Slave 0	MATRIX_PRBS0	Read/Write	0x00000000
0x0088	Priority Register A for Slave 1	MATRIX_PRAS1	Read/Write	0x00000000
0x008C	Priority Register B for Slave 1	MATRIX_PRBS1	Read/Write	0x00000000
0x0090	Priority Register A for Slave 2	MATRIX_PRAS2	Read/Write	0x00000000
0x0094	Priority Register B for Slave 2	MATRIX_PRBS2	Read/Write	0x00000000
0x0098	Priority Register A for Slave 3	MATRIX_PRAS3	Read/Write	0x00000000
0x009C	Priority Register B for Slave 3	MATRIX_PRBS3	Read/Write	0x00000000
0x00A0	Priority Register A for Slave 4	MATRIX_PRAS4	Read/Write	0x00000000
0x00A4	Priority Register B for Slave 4	MATRIX_PRBS4	Read/Write	0x00000000
0x00A8	Priority Register A for Slave 5	MATRIX_PRAS5	Read/Write	0x00000000
0x00AC	Priority Register B for Slave 5	MATRIX_PRBS5	Read/Write	0x00000000
0x00B0	Priority Register A for Slave 6	MATRIX_PRAS6	Read/Write	0x00000000
0x00B4	Priority Register B for Slave 6	MATRIX_PRBS6	Read/Write	0x00000000
0x00B8	Priority Register A for Slave 7	MATRIX_PRAS7	Read/Write	0x00000000
0x00BC	Priority Register B for Slave 7	MATRIX_PRBS7	Read/Write	0x00000000
0x00C0	Priority Register A for Slave 8	MATRIX_PRAS8	Read/Write	0x00000000
0x00C4	Priority Register B for Slave 8	MATRIX_PRBS8	Read/Write	0x00000000
0x00C8	Priority Register A for Slave 9	MATRIX_PRAS9	Read/Write	0x00000000
0x00CC	Priority Register B for Slave 9	MATRIX_PRBS9	Read/Write	0x00000000
0x00D0	Priority Register A for Slave 10	MATRIX_PRAS10	Read/Write	0x00000000
0x00D4	Priority Register B for Slave 10	MATRIX_PRBS10	Read/Write	0x00000000
0x00D8	Priority Register A for Slave 11	MATRIX_PRAS11	Read/Write	0x00000000
0x00DC	Priority Register B for Slave 11	MATRIX_PRBS11	Read/Write	0x00000000
0x00E0	Priority Register A for Slave 12	MATRIX_PRAS12	Read/Write	0x00000000
0x00E4	Priority Register B for Slave 12	MATRIX_PRBS12	Read/Write	0x00000000
0x00E8	Priority Register A for Slave 13	MATRIX_PRAS13	Read/Write	0x00000000
0x00EC	Priority Register B for Slave 13	MATRIX_PRBS13	Read/Write	0x00000000
0x00F0	Priority Register A for Slave 14	MATRIX_PRAS14	Read/Write	0x00000000
0x00F4	Priority Register B for Slave 14	MATRIX_PRBS14	Read/Write	0x00000000
0x00FC–0x014C	Reserved	–	–	–
0x0150	Master Error Interrupt Enable Register	MATRIX_MEIER	Write-only	–
0x0154	Master Error Interrupt Disable Register	MATRIX_MEIDR	Write-only	–
0x0158	Master Error Interrupt Mask Register	MATRIX_MEIMR	Read-only	0x00000000
0x015C	Master Error Status Register	MATRIX_MESR	Read-only	0x00000000
0x0160	Master 0 Error Address Register	MATRIX_MEAR0	Read-only	0x00000000
0x0164	Master 1 Error Address Register	MATRIX_MEAR1	Read-only	0x00000000
0x0168	Master 2 Error Address Register	MATRIX_MEAR2	Read-only	0x00000000

Table 17-10. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x016C	Master 3 Error Address Register	MATRIX_MEAR3	Read-only	0x00000000
0x0170	Master 4 Error Address Register	MATRIX_MEAR4	Read-only	0x00000000
0x0174	Master 5 Error Address Register	MATRIX_MEAR5	Read-only	0x00000000
0x0178	Master 6 Error Address Register	MATRIX_MEAR6	Read-only	0x00000000
0x017C	Master 7 Error Address Register	MATRIX_MEAR7	Read-only	0x00000000
0x0180	Master 8 Error Address Register	MATRIX_MEAR8	Read-only	0x00000000
0x0184	Master 9 Error Address Register	MATRIX_MEAR9	Read-only	0x00000000
0x0188	Master 10 Error Address Register	MATRIX_MEAR10	Read-only	0x00000000
0x018C	Master 11 Error Address Register	MATRIX_MEAR11	Read-only	0x00000000
0x0190–0x01E0	Reserved	–	–	–
0x01E4	Write Protection Mode Register	MATRIX_WPMR	Read/Write	0x00000000
0x01E8	Write Protection Status Register	MATRIX_WPSR	Read-only	0x00000000
0x01EC–0x01FC	Reserved	–	–	–
0x0200	Security Slave 0 Register	MATRIX_SSR0	Read/Write	0x00000000
0x0204	Security Slave 1 Register	MATRIX_SSR1	Read/Write	0x00000000
0x0208	Security Slave 2 Register	MATRIX_SSR2	Read/Write	0x00000000
0x020C	Security Slave 3 Register	MATRIX_SSR3	Read/Write	0x00000000
0x0210	Security Slave 4 Register	MATRIX_SSR4	Read/Write	0x00000000
0x0214	Security Slave 5 Register	MATRIX_SSR5	Read/Write	0x00000000
0x0218	Security Slave 6 Register	MATRIX_SSR6	Read/Write	0x00000000
0x021C	Security Slave 7 Register	MATRIX_SSR7	Read/Write	0x00000000
0x0220	Security Slave 8 Register	MATRIX_SSR8	Read/Write	0x00000000
0x0224	Security Slave 9 Register	MATRIX_SSR9	Read/Write	0x00000000
0x0228	Security Slave 10 Register	MATRIX_SSR10	Read/Write	0x00000000
0x022C	Security Slave 11 Register	MATRIX_SSR11	Read/Write	0x00000000
0x0230	Security Slave 12 Register	MATRIX_SSR12	Read/Write	0x00000000
0x0234	Security Slave 13 Register	MATRIX_SSR13	Read/Write	0x00000000
0x0238	Security Slave 14 Register	MATRIX_SSR14	Read/Write	0x00000000
0x023C	Reserved	–	–	–
0x0240	Security Areas Split Slave 0 Register	MATRIX_SASSR0	Read/Write	(1)
0x0244	Security Areas Split Slave 1 Register	MATRIX_SASSR1	Read/Write	(1)
0x0248	Security Areas Split Slave 2 Register	MATRIX_SASSR2	Read/Write	(1)
0x024C	Security Areas Split Slave 3 Register	MATRIX_SASSR3	Read/Write	(1)
0x0250	Security Areas Split Slave 4 Register	MATRIX_SASSR4	Read/Write	(1)
0x0254	Security Areas Split Slave 5 Register	MATRIX_SASSR5	Read/Write	(1)
0x0258	Security Areas Split Slave 6 Register	MATRIX_SASSR6	Read/Write	(1)
0x025C	Security Areas Split Slave 7 Register	MATRIX_SASSR7	Read/Write	(1)

Table 17-10. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0260	Security Areas Split Slave 8 Register	MATRIX_SASSR8	Read/Write	(1)
0x0264	Security Areas Split Slave 9 Register	MATRIX_SASSR9	Read/Write	(1)
0x0268	Security Areas Split Slave 10 Register	MATRIX_SASSR10	Read/Write	(1)
0x026C	Security Areas Split Slave 11 Register	MATRIX_SASSR11	Read/Write	(1)
0x0270	Security Areas Split Slave 12 Register	MATRIX_SASSR12	Read/Write	(1)
0x0274	Security Areas Split Slave 13 Register	MATRIX_SASSR13	Read/Write	(1)
0x0278	Security Areas Split Slave 14 Register	MATRIX_SASSR14	Read/Write	(1)
0x027C–0x0280	Reserved	–	–	–
0x0284	Security Region Top Slave 1 Register	MATRIX_SRTSR1	Read/Write	0x00000000
0x0288	Security Region Top Slave 2 Register	MATRIX_SRTSR2	Read/Write	0x00000000
0x028C	Security Region Top Slave 3 Register	MATRIX_SRTSR3	Read/Write	0x00000000
0x0290	Security Region Top Slave 4 Register	MATRIX_SRTSR4	Read/Write	0x00000000
0x0294	Security Region Top Slave 5 Register	MATRIX_SRTSR5	Read/Write	0x00000000
0x0298	Security Region Top Slave 6 Register	MATRIX_SRTSR6	Read/Write	0x00000000
0x029C	Security Region Top Slave 7 Register	MATRIX_SRTSR7	Read/Write	0x00000000
0x02A0	Security Region Top Slave 8 Register	MATRIX_SRTSR8	Read/Write	0x00000000
0x02A4	Security Region Top Slave 9 Register	MATRIX_SRTSR9	Read/Write	0x00000000
0x02A8	Security Region Top Slave 10 Register	MATRIX_SRTSR10	Read/Write	0x00000000
0x02AC	Security Region Top Slave 11 Register	MATRIX_SRTSR11	Read/Write	0x00000000
0x02B0	Security Region Top Slave 12 Register	MATRIX_SRTSR12	Read/Write	0x00000000
0x02B4	Security Region Top Slave 13 Register	MATRIX_SRTSR13	Read/Write	0x00000000
0x02B8	Security Region Top Slave 14 Register	MATRIX_SRTSR14	Read/Write	0x00000000
0x02BC	Reserved	–	–	–
0x02C0	Security Peripheral Select 1 Register	MATRIX_SPSELR1	Read/Write	0x00000000(2)
0x02C4	Security Peripheral Select 2 Register	MATRIX_SPSELR2	Read/Write	0x00000000(3)
0x02C8	Security Peripheral Select 3 Register	MATRIX_SPSELR3	Read/Write	0x00000000(4)

Notes:
1. When applicable to an AHB slave region, the initial value of SAXXRx.SASPLITy is 0xF. When not applicable to an AHB slave region, the initial value of SAXXRx.SASPLITy is 0x0.
2. This value is 0x000D2504 for H32MX and 0xFFF2DAFB for H64MX.
3. This value is 0x011C0000 for H32MX and 0xFFE7FFFF for H64MX.
4. This value is 0xFFFFFFFA for H32MX and 0xFFFFFFE7 for H64MX.

17.13.1 Bus Matrix Master Configuration Registers

Name: MATRIX_MCFGx [x=0..11]

Address: 0xF0018000 (0), 0xFC03C000 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	ULBT		

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **ULBT: Undefined Length Burst Type**

Value	Name	Description
0	UNLIMITED	Unlimited Length Burst—No predicted end of burst is generated, therefore INCR bursts coming from this master can only be broken if the Slave Slot Cycle Limit is reached. If the Slot Cycle Limit is not reached, the burst is normally completed by the master, at the latest, on the next AHB 1 Kbyte address boundary, allowing up to 256-beat word bursts or 128-beat double-word bursts. This value should not be used in the very particular case of a master capable of performing back-to-back undefined length bursts on a single slave, since this could indefinitely freeze the slave arbitration and thus prevent another master from accessing this slave.
1	SINGLE	Single Access—The undefined length burst is treated as a succession of single accesses, allowing re-arbitration at each beat of the INCR burst or bursts sequence.
2	4_BEAT	4-beat Burst—The undefined length burst or bursts sequence is split into 4-beat bursts or less, allowing re-arbitration every 4 beats.
3	8_BEAT	8-beat Burst—The undefined length burst or bursts sequence is split into 8-beat bursts or less, allowing re-arbitration every 8 beats.
4	16_BEAT	16-beat Burst—The undefined length burst or bursts sequence is split into 16-beat bursts or less, allowing re-arbitration every 16 beats.
5	32_BEAT	32-beat Burst—The undefined length burst or bursts sequence is split into 32-beat bursts or less, allowing re-arbitration every 32 beats.
6	64_BEAT	64-beat Burst—The undefined length burst or bursts sequence is split into 64-beat bursts or less, allowing re-arbitration every 64 beats.
7	128_BEAT	128-beat Burst—The undefined length burst or bursts sequence is split into 128-beat bursts or less, allowing re-arbitration every 128 beats. Unless duly needed, the ULBT should be left at its default 0 value for power saving.

17.13.2 Bus Matrix Slave Configuration Registers

Name: MATRIX_SCFGx [x=0..14]

Address: 0xF0018040 (0), 0xFC03C040 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	FIXED_DEFMSTR				DEFMSTR_TYPE	

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SLOT_CYCLE

7	6	5	4	3	2	1	0
SLOT_CYCLE							

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **SLOT_CYCLE: Maximum Bus Grant Duration for Masters**

When SLOT_CYCLE AHB clock cycles have elapsed since the last arbitration, a new arbitration takes place to let another master access this slave. If another master is requesting the slave bus, then the current master burst is broken.

If SLOT_CYCLE = 0, the Slot Cycle Limit feature is disabled and bursts always complete unless broken according to the ULBT.

This limit has been placed in order to enforce arbitration so as to meet potential latency constraints of masters waiting for slave access.

This limit must not be too small. Unreasonably small values break every burst and the Bus Matrix arbitrates without performing any data transfer. The default maximum value is usually an optimal conservative choice.

In most cases, this feature is not needed and should be disabled for power saving.

See Section 17.10.1.2 “Slot Cycle Limit Arbitration” for details.

- **DEFMSTR_TYPE: Default Master Type**

Value	Name	Description
0	NONE	No Default Master—At the end of the current slave access, if no other master request is pending, the slave is disconnected from all masters. This results in a one clock cycle latency for the first access of a burst transfer or for a single access.
1	LAST	Last Default Master—At the end of the current slave access, if no other master request is pending, the slave stays connected to the last master having accessed it. This results in not having one clock cycle latency when the last master tries to access the slave again.
2	FIXED	Fixed Default Master—At the end of the current slave access, if no other master request is pending, the slave connects to the fixed master the number that has been written in the FIXED_DEFMSTR field. This results in not having one clock cycle latency when the fixed master tries to access the slave again.

- **FIXED_DEFMSTR: Fixed Default Master**

This is the number of the Default Master for this slave. Only used if DEFMSTR_TYPE value = 2. Specifying the number of a master which is not connected to the selected slave is equivalent to clearing DEFMSTR_TYPE.

17.13.3 Bus Matrix Priority Registers A For Slaves

Name: MATRIX_PRASx [x=0..14]

Address: 0xF0018080 (0)[0], 0xF0018088 (0)[1], 0xF0018090 (0)[2], 0xF0018098 (0)[3], 0xF00180A0 (0)[4], 0xF00180A8 (0)[5], 0xF00180B0 (0)[6], 0xF00180B8 (0)[7], 0xF00180C0 (0)[8], 0xF00180C8 (0)[9], 0xF00180D0 (0)[10], 0xF00180D8 (0)[11], 0xF00180E0 (0)[12], 0xF00180E8 (0)[13], 0xF00180F0 (0)[14], 0xFC03C080 (1)[0], 0xFC03C088 (1)[1], 0xFC03C090 (1)[2], 0xFC03C098 (1)[3], 0xFC03C0A0 (1)[4], 0xFC03C0A8 (1)[5], 0xFC03C0B0 (1)[6], 0xFC03C0B8 (1)[7], 0xFC03C0C0 (1)[8], 0xFC03C0C8 (1)[9], 0xFC03C0D0 (1)[10], 0xFC03C0D8 (1)[11], 0xFC03C0E0 (1)[12], 0xFC03C0E8 (1)[13], 0xFC03C0F0 (1)[14]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	M7PR		–	–	M6PR	

23	22	21	20	19	18	17	16
–	–	M5PR		–	–	M4PR	

15	14	13	12	11	10	9	8
–	–	M3PR		–	–	M2PR	

7	6	5	4	3	2	1	0
–	–	M1PR		–	–	M0PR	

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **MxPR: Master x Priority**

Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.

All the masters programmed with the same MxPR value for the slave make up a priority pool.

Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.

Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).

See Section 17.10.2 “Arbitration Priority Scheme” for details.

17.13.4 Bus Matrix Priority Registers B For Slaves

Name: MATRIX_PRBSx [x=0..14]

Address: 0xF0018084 (0)[0], 0xF001808C (0)[1], 0xF0018094 (0)[2], 0xF001809C (0)[3], 0xF00180A4 (0)[4], 0xF00180AC (0)[5], 0xF00180B4 (0)[6], 0xF00180BC (0)[7], 0xF00180C4 (0)[8], 0xF00180CC (0)[9], 0xF00180D4 (0)[10], 0xF00180DC (0)[11], 0xF00180E4 (0)[12], 0xF00180EC (0)[13], 0xF00180F4 (0)[14], 0xFC03C084 (1)[0], 0xFC03C08C (1)[1], 0xFC03C094 (1)[2], 0xFC03C09C (1)[3], 0xFC03C0A4 (1)[4], 0xFC03C0AC (1)[5], 0xFC03C0B4 (1)[6], 0xFC03C0BC (1)[7], 0xFC03C0C4 (1)[8], 0xFC03C0CC (1)[9], 0xFC03C0D4 (1)[10], 0xFC03C0DC (1)[11], 0xFC03C0E4 (1)[12], 0xFC03C0EC (1)[13], 0xFC03C0F4 (1)[14]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	M11PR		–	–	M10PR	

7	6	5	4	3	2	1	0
–	–	M9PR		–	–	M8PR	

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **MxPR: Master x Priority**

Fixed priority of Master x for accessing the selected slave. The higher the number, the higher the priority.

All the masters programmed with the same MxPR value for the slave make up a priority pool.

Round-robin arbitration is used in the lowest (MxPR = 0) and highest (MxPR = 3) priority pools.

Fixed priority is used in intermediate priority pools (MxPR = 1) and (MxPR = 2).

See Section 17.10.2 “Arbitration Priority Scheme” for details.

17.13.5 Master Error Interrupt Enable Register

Name: MATRIX_MEIER

Address: 0xF0018150 (0), 0xFC03C150 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	MERR11	MERR10	MERR9	MERR8
7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **MERRx: Master x Access Error**

0: No effect

1: Enables Master x Access Error interrupt source

17.13.6 Master Error Interrupt Disable Register

Name: MATRIX_MEIDR

Address: 0xF0018154 (0), 0xFC03C154 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	MERR11	MERR10	MERR9	MERR8
7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **MERRx: Master x Access Error**

0: No effect

1: Disables Master x Access Error interrupt source

17.13.7 Master Error Interrupt Mask Register

Name: MATRIX_MEIMR

Address: 0xF0018158 (0), 0xFC03C158 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	MERR11	MERR10	MERR9	MERR8

7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

- **MERRx: Master x Access Error**

0: Master x Access Error does not trigger any interrupt.

1: Master x Access Error triggers the Bus Matrix interrupt line.

17.13.8 Master Error Status Register

Name: MATRIX_MESR

Address: 0xF001815C (0), 0xFC03C15C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	MERR11	MERR10	MERR9	MERR8
7	6	5	4	3	2	1	0
MERR7	MERR6	MERR5	MERR4	MERR3	MERR2	MERR1	MERR0

- **MERRx: Master x Access Error**

0: No Master Access Error has occurred since the last read of the MATRIX_MESR.

1: At least one Master Access Error has occurred since the last read of the MATRIX_MESR.

17.13.9 Master Error Address Registers

Name: MATRIX_MEARx [x=0..11]

Address: 0xF0018160 (0), 0xFC03C160 (1)

Access: Read-only

31	30	29	28	27	26	25	24
ERRADD							

23	22	21	20	19	18	17	16
ERRADD							

15	14	13	12	11	10	9	8
ERRADD							

7	6	5	4	3	2	1	0
ERRADD							

- **ERRADD: Master Error Address**

Master Last Access Error Address

17.13.10 Write Protection Mode Register

Name: MATRIX_WPMR

Address: 0xF00181E4 (0), 0xFC03C1E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the Write Protection if WPKEY corresponds to 0x4D4154 (“MAT” in ASCII).

1: Enables the Write Protection if WPKEY corresponds to 0x4D4154 (“MAT” in ASCII).

See Section 17.11 “Register Write Protection” for list of registers that can be write-protected.

- **WPKEY: Write Protection Key** (Write-only)

Value	Name	Description
0x4D4154	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

17.13.11 Write Protection Status Register

Name: MATRIX_WPSR

Address: 0xF00181E8 (0), 0xFC03C1E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of MATRIX_WPSR.

1: A write protection violation has occurred since the last write of MATRIX_WPMR.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

17.13.12 Security Slave Registers

Name: MATRIX_SSRx [x=0..14]

Address: 0xF0018200 (0), 0xFC03C200 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WRNSECH7	WRNSECH6	WRNSECH5	WRNSECH4	WRNSECH3	WRNSECH2	WRNSECH1	WRNSECH0

15	14	13	12	11	10	9	8
RDNSECH7	RDNSECH6	RDNSECH5	RDNSECH4	RDNSECH3	RDNSECH2	RDNSECH1	RDNSECH0

7	6	5	4	3	2	1	0
LANSECH7	LANSECH6	LANSECH5	LANSECH4	LANSECH3	LANSECH2	LANSECH1	LANSECH0

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **LANSECHx: Low Area Non-Secured in HSELx Security Region**

0: The security of the HSELx AHB slave area laying below the corresponding MATRIX_SASSR / SASPLITx boundary is configured according to RDNSECHx and WRNSECHx. The whole remaining HSELx upper address space is configured as Non-Secured access.

1: The HSELx AHB slave address area laying below the corresponding MATRIX_SASSR / SASPLITx boundary is configured as Non-Secured access, and the whole remaining upper address space according to RDNSECHx and WRNSECHx.

- **RDNSECHx: Read Non-Secured for HSELx Security Region**

0: The HSELx AHB slave security region is split into one Read Secured and one Read Non-Secured area, according to LANSECHx and MATRIX_SASSR / SASPLITx. That is, the so defined Securable high or low area is Secured for Read access.

1: The HSELx AHB slave security region is Non-Secured for Read access.

- **WRNSECHx: Write Non-Secured for HSELx Security Region**

0: The HSELx AHB slave security region is split into one Write Secured and one Write Non-Secured area, according to LANSECHx and MATRIX_SASSR / SASPLITx. That is, the so defined Securable high or low area is Secured for Write access.

1: The HSELx AHB slave security region is Non-Secured for Write access.

Securable Area access rights:

WRNSECHx / RDNSECHx	Non-Secure Access	Secure Access
00	Denied	Write - Read
01	Read	Write - Read
10	Write	Write - Read
11	Write - Read	Write - Read

17.13.13 Security Areas Split Slave Registers

Name: MATRIX_SASSRx [x=0..14]

Address: 0xF0018240 (0), 0xFC03C240 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
SASPLIT7				SASPLIT6			

23	22	21	20	19	18	17	16
SASPLIT5				SASPLIT4			

15	14	13	12	11	10	9	8
SASPLIT3				SASPLIT2			

7	6	5	4	3	2	1	0
SASPLIT1				SASPLIT0			

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **SASPLITx: Security Areas Split for HSELx Security Region**

This field defines the boundary address offset where the HSELx AHB slave security region splits into two Security Areas whose access is controlled according to the corresponding MATRIX_SSR. So it also defines the Security Low Area size inside the HSELx region.

If this Low Area size is set at or above the HSELx Region Size, then there is no more Security High Area and the MATRIX_SSR settings for the Low area apply to the whole HSELx Security Region.

SASPLITx	Split Offset	Security Low Area Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes
1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.13.14 Security Region Top Slave Registers

Name: MATRIX_SRTSRx [x=0..14]

Address: 0xF0018284 (0), 0xFC03C284 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
SRTOP7				SRTOP6			
23	22	21	20	19	18	17	16
SRTOP5				SRTOP4			
15	14	13	12	11	10	9	8
SRTOP3				SRTOP2			
7	6	5	4	3	2	1	0
SRTOP1				SRTOP0			

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **SRTOPx: HSELx Security Region Top**

This field defines the size of the HSELx security region address space. Invalid sizes for the slave region must never be programmed. Valid sizes and number of security regions are product, slave and slave configuration dependent.

Note: The slaves featuring multiple scalable contiguous security regions have a single SRTOP0 field for all the security regions.

If this HSELx security region size is set at or below the HSELx low area size, then there is no more security high area and the MATRIX_SSR settings for the low area apply to the whole HSELx security region.

.

SRTOPx	Top Offset	Security Region Size
0000	0x00001000	4 Kbytes
0001	0x00002000	8 Kbytes
0010	0x00004000	16 Kbytes
0011	0x00008000	32 Kbytes
0100	0x00010000	64 Kbytes
0101	0x00020000	128 Kbytes
0110	0x00040000	256 Kbytes
0111	0x00080000	512 Kbytes
1000	0x00100000	1 Mbyte
1001	0x00200000	2 Mbytes
1010	0x00400000	4 Mbytes
1011	0x00800000	8 Mbytes
1100	0x01000000	16 Mbytes
1101	0x02000000	32 Mbytes
1110	0x04000000	64 Mbytes
1111	0x08000000	128 Mbytes

17.13.15 Security Peripheral Select x Registers

Name: MATRIX_SPSELRx [x=1..3]

Address: 0xF00182C0 (0), 0xFC03C2C0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
NSECP31	NSECP30	NSECP29	NSECP28	NSECP27	NSECP26	NSECP25	NSECP24
23	22	21	20	19	18	17	16
NSECP23	NSECP22	NSECP21	NSECP20	NSECP19	NSECP18	NSECP17	NSECP16
15	14	13	12	11	10	9	8
NSECP15	NSECP14	NSECP13	NSECP12	NSECP11	NSECP10	NSECP9	NSECP8
7	6	5	4	3	2	1	0
NSECP7	NSECP6	NSECP5	NSECP4	NSECP3	NSECP2	NSECP1	NSECP0

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

Each MATRIX_SPSELR can configure the access security type for up to 32 peripherals:

- MATRIX_SPSELR1 configures the access security type for peripheral identifiers 0–31 (bits NSECP0–NSECP31).
- MATRIX_SPSELR2 configures the access security type for peripheral identifiers 32–63 (bits NSECP0–NSECP31).
- MATRIX_SPSELR3 configures the access security type for peripheral identifiers 64–95 (bits NSECP0–NSECP31).

Note: The actual number of peripherals implemented is device-specific; refer to the “Peripheral Identifiers” section for details.

- **NSECPy: Non-Secured Peripheral**

0: The selected peripheral address space is configured as “Secured” access (value of ‘0’ has no effect if the peripheral security type is “Peripheral Always Non-Secured”).

1: The selected peripheral address space is configured as “Non-Secured” access (value of ‘1’ has no effect if the peripheral security type is “Peripheral Always Secured”).

18. Special Function Registers (SFR)

18.1 Description

Special Function Registers (SFR) manage specific aspects of the integrated memory, bridge implementations, processor and other functionality not controlled elsewhere.

18.2 Embedded Characteristics

- 32-bit Special Function Registers control specific behavior of the product

18.3 Special Function Registers (SFR) User Interface

Table 18-1. Register Mapping

Offset[1]	Register	Name	Access	Reset
0x00	Reserved	–	–	–
0x04	DDR Configuration Register	SFR_DDRCFG	Read/Write	0x01
0x08–0x0C	Reserved	–	–	–
0x10	OHCI Interrupt Configuration Register	SFR_OHCIICR	Read/Write	0x0
0x14	OHCI Interrupt Status Register	SFR_OHCIISR	Read-only	–
0x18	Reserved	–	–	–
0x1C	Reserved	–	–	–
0x20–0x24	Reserved	–	–	–
0x28	Security Configuration Register	SFR_SECURE	Read/Write	0x0
0x2C	Reserved	–	–	–
0x30	UTMI Clock Trimming Register	SFR_UTMICKTRIM	Read/Write	0x00010000
0x34	UTMI High Speed Trimming Register	SFR_UTMIHSTRIM	Read/Write	0x00044433
0x38	UTMI Full Speed Trimming Register	SFR_UTMIFSTRIM	Read/Write	0x00430211
0x3C	UTMI DP/DM Pin Swapping Register	SFR_UTMISWAP	Read/Write	0x0
0x40	Reserved	–	–	–
0x44	Reserved	–	–	–
0x48	CAN Memories Address-based Register	SFR_CAN	Read/Write	0x00200020
0x4C	Serial Number 0 Register	SFR_SN0	Read-only	–
0x50	Serial Number 1 Register	SFR_SN1	Read-only	–
0x54	AIC Interrupt Redirection Register	SFR_AICREDIR	Read/Write	0x0
0x58	L2CC_HRAMC1	SFR_L2CC_HRAMC	Read/Write	0x0
0x5C–0x8C	Reserved	–	–	–
0x90	I2SC Register	SFR_I2SCLKSEL	Read/Write	0x0
0x94	QSPI Clock Pad Supply Select Register	QSPICLK_REG	Read/Write	0x1
0x98–0x3FFC	Reserved	–	–	–

Note: 1. If an offset is not listed in the table it must be considered as reserved.

18.3.1 DDR Configuration Register

Name: SFR_DDRCFG

Address: 0xF8030004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	FDQSIEN	FDQIEN
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **FDQIEN: Force DDR_DQ Input Buffer Always On**

0: DDR_DQ input buffer controlled by DDR controller.

1: DDR_DQ input buffer always on.

- **FDQSIEN: Force DDR_DQS Input Buffer Always On**

0: DDR_DQS input buffer controlled by DDR controller.

1: DDR_DQS input buffer always on.

Note: FDQIEN and FDQSIEN = 1 are used to force the selection of the analog comparator inside the IO. If those bits are cleared the DDR controller automatically manages the selection of the analog comparator. Forcing the bits to 0 reduces power consumption.

18.3.2 OHCI Interrupt Configuration Register

Name: SFR_OHCIICR

Address: 0xF8030010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	HSIC_SEL	–	–	–
23	22	21	20	19	18	17	16
UDPPUDIS	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	SUSPEND_C	SUSPEND_B	SUSPEND_A
7	6	5	4	3	2	1	0
–	–	APPSTART	ARIE	–	RES2	RES1	RES0

- **RESx: USB PORTx RESET**

0: Resets USB PORT.

1: Usable USB PORT.

- **ARIE: OHCI Asynchronous Resume Interrupt Enable**

0: Interrupt disabled.

1: Interrupt enabled.

- **APPSTART: Reserved**

0: Must write 0.

- **SUSPEND_A: USB PORT A**

0: Suspends controlled by EHCI-OCHO

1: Forces the suspend for PORTA

- **SUSPEND_B: USB PORT B**

0: Suspend controlled by EHCI-OCHO

1: Forces the suspend for PORTB

- **SUSPEND_C: USB PORT C**

0: Suspends controlled by EHCI-OCHO

1: Forces the suspend for PORTC

- **UDPPUDIS: USB DEVICE PULL-UP DISABLE**

0: USB device pull-up connection is enabled.

1: USB device pull-up connection is disabled.

- **HSIC_SEL: Reserved**

0: Must write 0.

18.3.3 OHCI Interrupt Status Register

Name: SFR_OHCIISR

Address: 0xF8030014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	RIS2	RIS1	RIS0

- **RISx: OHCI Resume Interrupt Status Port x**

0: OHCI port resume not detected.

1: OHCI port resume detected.

18.3.4 Security Configuration Register

Name: SFR_SECURE

Address: 0xF8030028

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	FUSE
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ROM

- **ROM: Disable Access to ROM Code**

This bit is writable once only. When the ROM is secured, only a reset signal can clear this bit.

0: ROM is enabled.

1: ROM is disabled.

- **FUSE: Disable Access to Fuse Controller**

This bit is writable once only. When the Fuse Controller is secured, only a reset signal can clear this bit.

0: Fuse Controller is enabled.

1: Fuse Controller is disabled.

18.3.5 UTMI Clock Trimming Register

Name: SFR_UTMICKTRIM

Address: 0xF8030030

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	VBG			
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	FREQ	

- **FREQ: UTMI Reference Clock Frequency**

Value	Name	Description
0	12	12 MHz reference clock
1	16	16 MHz reference clock
2	24	24 MHz reference clock
3	12	12 MHz reference clock

- **VBG: UTMI Band Gap Voltage Trimming**

18.3.6 UTMI High Speed Trimming Register

Name: SFR_UTMIHSTRIM

Address: 0xF8030034

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	SLOPE2		
15	14	13	12	11	10	9	8
–	SLOPE1				SLOPE0		
7	6	5	4	3	2	1	0
–	DISC			–	SQUELCH		

- **SQUELCH: UTMI HS SQUELCH Voltage Trimming**

Calibration bits to adjust squelch threshold.

- **DISC: UTMI Disconnect Voltage Trimming**

Calibration bits to adjust disconnect threshold.

- **SLOPEx: UTMI HS PORTx Transceiver Slope Trimming**

Calibration bits to adjust HS Transceiver output slope for PORTx.

18.3.7 UTMI Full Speed Trimming Register

Name: SFR_UTMIFSTRIM

Address: 0xF8030038

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	ZP			–	ZN		
15	14	13	12	11	10	9	8
–	–	–	–	–	–	XCVR	
7	6	5	4	3	2	1	0
–	FALL			–	RISE		

- **RISE: FS Transceiver Output Rising Slope Trimming**

Calibration bits to adjust the FS transceiver output rising slope.

- **FALL: FS Transceiver Output Falling Slope Trimming**

Calibration bits to adjust the FS transceiver output falling slope.

- **XCVR: FS Transceiver Crossover Voltage Trimming**

Calibration bits to adjust the FS transceiver crossover voltage.

- **ZN: FS Transceiver NMOS Impedance Trimming**

Calibration bits to adjust the FS transceiver NMOS output impedance.

- **ZP: FS Transceiver PMOS Impedance Trimming**

Calibration bits to adjust the FS transceiver PMOS output impedance.

18.3.8 UMTI DP/DM Pin Swapping Register

Name: SFR_UTMISWAP

Address: 0xF803003C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	PORT2	PORT1	PORT0

- **PORTx: PORT x DP/DM Pin Swapping**

0 (NORMAL): DP/DM normal pinout.

1 (SWAPPED): DP/DM swapped pinout.

18.3.9 CAN Memories Address-based Register

Name: SFR_CAN

Address: 0xF8030048

Access: Read/Write

31	30	29	28	27	26	25	24
EXT_MEM_CAN1_ADDR							
23	22	21	20	19	18	17	16
EXT_MEM_CAN1_ADDR							
15	14	13	12	11	10	9	8
EXT_MEM_CAN0_ADDR							
7	6	5	4	3	2	1	0
EXT_MEM_CAN0_ADDR							

- **EXT_MEM_CAN0_ADDR: MSB CAN0 DMA Base Address**

Gives the 16-bit MSB of the CAN0 DMA base address. The 16-bit LSB must be programmed in the CAN0 user interface.

- **EXT_MEM_CAN1_ADDR: MSB CAN1 DMA Base Address**

Gives the 16-bit MSB of the CAN1 DMA base address. The 16-bit LSB must be programmed in the CAN1 user interface.

18.3.10 Serial Number 0 Register

Name: SFR_SN0

Address: 0xF803004C

Access: Read-only

31	30	29	28	27	26	25	24
SN0							
23	22	21	20	19	18	17	16
SN0							
15	14	13	12	11	10	9	8
SN0							
7	6	5	4	3	2	1	0
SN0							

This register is used to read the first 32 bits of the 64-bit Serial Number (unique ID).

- **SN0: Serial Number 0**

18.3.11 Serial Number 1 Register

Name: SFR_SN1

Address: 0xF8030050

Access: Read-only

31	30	29	28	27	26	25	24
SN1							

23	22	21	20	19	18	17	16
SN1							

15	14	13	12	11	10	9	8
SN1							

7	6	5	4	3	2	1	0
SN1							

This register is used to read the last 32 bits of the 64-bit Serial Number (unique ID).

- **SN1: Serial Number 1**

18.3.12 AIC Interrupt Redirection Register

Name: SFR_AICREDIR

Address: 0xF8030054

Access: Read/Write

31	30	29	28	27	26	25	24
AICREDIRKEY							
23	22	21	20	19	18	17	16
AICREDIRKEY							
15	14	13	12	11	10	9	8
AICREDIRKEY							
7	6	5	4	3	2	1	0
AICREDIRKEY							NSAIC

- **NSAIC: Interrupt Redirection to Non-Secure AIC**

0: Interrupts are managed by the AIC corresponding to the Secure State of the peripheral (secure AIC or non-secure AIC).

1: All interrupts are managed by the non-secure AIC.

- **AICREDIRKEY: Unlock Key**

Value is a XOR between 0xB6D81C4D and SN1[31:0] but only field [31:1] of the result must be written in this field. In case of set in Secure mode by fuse configuration, this register is read_only 0 (it is not possible to redirect secure interrupts on non-secure AIC for products set in secure mode for security reasons).

Note: After three tries, entering a wrong key results in locking the NSAIC bit. A reset is needed.

18.3.13 HRAMC L2CC Register

Name: SFR_L2CC_HRAMC

Address: 0xF8030058

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SRAM_SEL

This register is used to configure the L2 cache to be used as an internal SRAM.

- **SRAM_SEL: SRAM Selector**

0: Selects SRAM.

1: Selects L2CC.

18.3.14 I2S Register

Name: SFR_I2SCLKSEL

Address: 0xF8030090

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	CLKSEL1	CLKSEL0

- **CLKSEL0: Clock Selection 0**

0: Selects PCLK (peripheral clock).

1: Selects GCLK.

- **CLKSEL1: Clock Selection 1**

0: Selects PCLK (peripheral clock).

1: Selects GCLK.

18.3.15 QSPI Clock Pad Supply Select Register

Name: QSPICLK_REG

Address: 0xF8030094

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SUP_SEL

- **SUP_SEL: Supply Selection**

0: 1.8V supply selected

1: 3.3V supply selected

19. Special Function Registers Backup (SFRBU)

19.1 Description

Special Function Registers Backup (SFRBU) manages specific aspects of the integrated memory, bridge implementations, processor and other functionality not controlled elsewhere.

19.2 Embedded Characteristics

- 32-bit Special Function Registers Backup controls specific behavior of the product.

19.3 Special Function Registers Backup (SFRBU) User Interface

Table 19-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Power Switch BU Control Register	SFRBU_PSWBUCTRL	Read/Write	0x09
0x04	TS Range Configuration Register	SFRBU_TSRANGECFG	Read/Write	0x00
0x08	Reserved	–	–	–
0x0C	Reserved	–	–	–
0x10	DDR BU Mode Control Register	SFRBU_DDRBUMCR	Read/Write	0x00
0x14	RXLP Pull-Up Control Register	SFRBU_RXLPPUCR	Read/Write	0x01
0x18-0xFC	Reserved	–	–	–
0x100	Reserved	–	–	–
0x104–0x3FFC	Reserved	–	–	–

19.3.1 Power Switch BU Control Register

Name: SFRBU_PSWBUCTRL

Address: 0xFC05C000

Access: Read/Write

31	30	29	28	27	26	25	24
KEY PSW MODE							
23	22	21	20	19	18	17	16
KEY PSW MODE							
15	14	13	12	11	10	9	8
KEY PSW MODE							
7	6	5	4	3	2	1	0
–	–	–	–	STATE	SMCTRL	SSWCTRL	SCTRL

- **SCTRL: Power Switch BU Software Control**

This bit is used to control the Power Switch BU state by software in addition to the SSWCTRL bit.

0: Power Switch BU is controlled by hardware (SSWCTRL bit has no action).

1 (Reset value): Power Switch BU is controlled by software (SSWCTRL bit has an action).

- **SSWCTRL: Power Switch BU Source Selection**

This bit has an action only if SCTRL bit value is “1”.

0 (Reset value): LDO Supply source is VDDBU.

1: LDO Supply source is VDDANA.

- **SMCTRL: Allow Power Switch BU Control by Security Module Autobackup (Hardware)**

Enables to select automatically the VDDBU source when the security module enters Backup mode.

This automatic supply source switching is independent from the SCTRL and SSWCTRL bits.

0 (Reset value): No automatic supply source switching from security module.

1: Automatic supply source switching from security module activated.

- **STATE: Power Switch BU state (Read-only)**

This bit reflects the power switch BU supply source selection in real time. After a switching request, the user must wait for the analog cell switching time to have an updated status (see Electrical Characteristics section).

0: LDO BU Supply source is VDDBU.

1: LDO BU Supply source is VDDANA.

- **KEY_PSW_MODE: Specific value mandatory to allow writing of other register bits (Write-only)**

This field is a security key preventing power switch changes due to software error or malicious code.

0x4BD20C: SFRBU_PSWBUCTRL register write possible.

Other values: SFRBU_PSWBUCTRL register write impossible.

19.3.2 Temperature Sensor Range Configuration Register

Name: SFRBU_TSRANGECFG

Address: 0xFC05C004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	TSHRSEL

- **TSHRSEL: Temperature Sensor Range Selection**

0 (Reset value): Temperature Sensor High Triggering level is +105°C (internal transistor junction temperature).

1: Temperature Sensor High Triggering level is +115°C (internal transistor junction temperature).

19.3.3 DDR BU Mode Control Register

Name: SFRBU_DDRBUMCR

Address: 0xFC05C010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	BUMEN

- **BUMEN: DDR BU Mode Enable**

This bit is used to isolate the DDR Pads from the CPU domain (VCCCORE).

It has to be set after enabling the Self-refresh mode on the DDR memory and before doing powerdown on VCCCORE.(1)

0 (Reset value): DDR Backup mode disabled. The DDR pads are not isolated from CPU domain.

1: DDR Backup mode enabled. The DDR pads are isolated from CPU domain (IOs are in memory state).

Note: 1. To enable Self-refresh mode, refer to MPDDRC Low-power Register (in Multiport DDR-SDRAM Controller section) and to Self-refresh Backup mode (in Electrical Characteristics section).

19.3.4 RXLP Pull-Up Control Register

Name: SFRBU_RXLPPUCR

Address: 0xFC05C014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	RXDPUCTRL

- **RXDPUCTRL: RXLP RXD Pull-Up Control**

0 (Reset value): Pull-up enabled on RXD IO.

1: Pull-up disabled on RXD IO.

Note: If the RXLP is not used, it is recommended to enable the pull-up to avoid power consumption on VDDBU rail.

20. Advanced Interrupt Controller (AIC)

20.1 Description

The Advanced Interrupt Controller (AIC) is an 8-level priority, individually maskable, vectored interrupt controller providing handling of up to one hundred and twenty-eight interrupt sources. It is designed to substantially reduce the software and real-time overhead in handling internal and external interrupts.

The AIC drives the nFIQ (fast interrupt request) and the nIRQ (standard interrupt request) inputs of an ARM processor. Inputs of the AIC are either internal peripheral interrupts or external interrupts coming from the product's pins.

The 8-level Priority Controller allows the user to define the priority for each interrupt source, thus permitting higher priority interrupts to be serviced even if a lower priority interrupt is being processed.

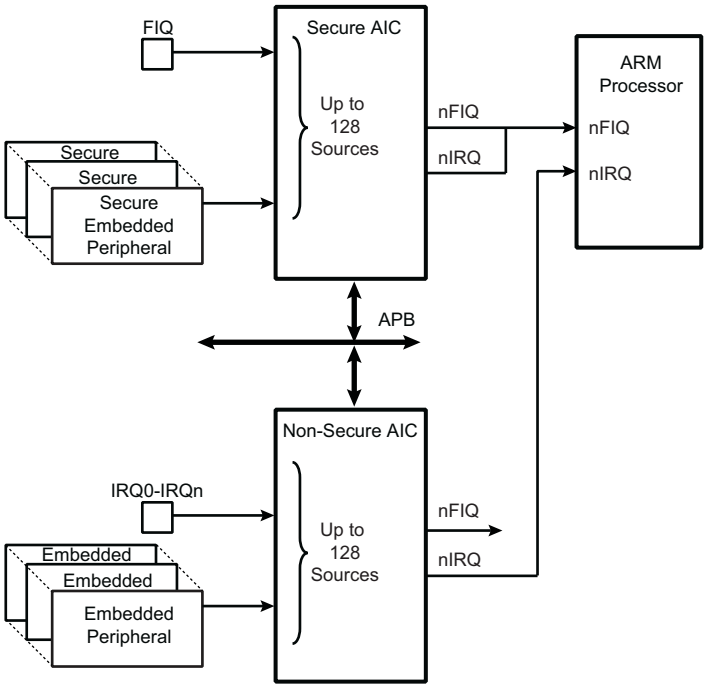
Internal interrupt sources can be programmed to be level-sensitive or edge-triggered. External interrupt sources can be programmed to be positive-edge or negative-edge triggered or high-level or low-level sensitive.

20.2 Embedded Characteristics

- Controls the Interrupt Lines (nIRQ and nFIQ) of an ARM Processor
- 128 Individually Maskable and Vectored Interrupt Sources
  - Source 0 is Reserved for the Fast Interrupt Input (FIQ)
  - Source 74 is Reserved for System Peripheral Interrupts
  - Sources 2 to 73 and Sources 75 to 127 Control up to 125 Embedded Peripheral Interrupts or External Interrupts
  - Programmable Edge-triggered or Level-sensitive Internal Sources
  - Programmable Positive/Negative Edge-triggered or High/Low Level-sensitive External Sources
- 8-level Priority Controller
  - Drives the Normal Interrupt of the Processor
  - Handles Priority of the Interrupt Sources 1 to 127
  - Higher Priority Interrupts Can Be Served During Service of Lower Priority Interrupt
- Vectoring
  - Optimizes Interrupt Service Routine Branch and Execution
  - One 32-bit Vector Register for all Interrupt Sources
  - Interrupt Vector Register Reads the Corresponding Current Interrupt Vector
- Protect Mode
  - Easy Debugging by Preventing Automatic Operations when Protect Models are Enabled
- General Interrupt Mask
  - Provides Processor Synchronization on Events Without Triggering an Interrupt
- Register Write Protection
- AIC0 is Non-Secure AIC, AIC1 is Secure AIC
- AIC0 manages nIRQ line, AIC1 manages nFIQ line

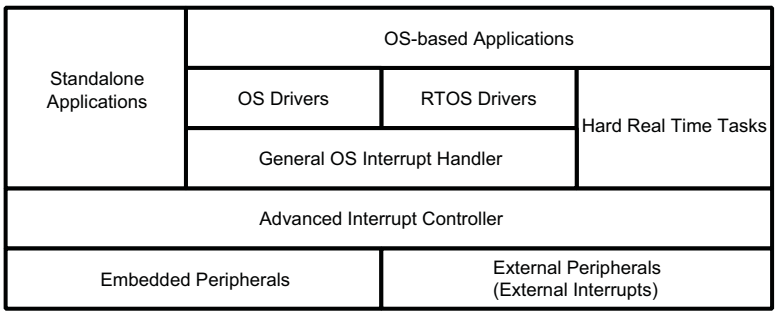
20.3 Block Diagram

Figure 20-1. Block Diagram



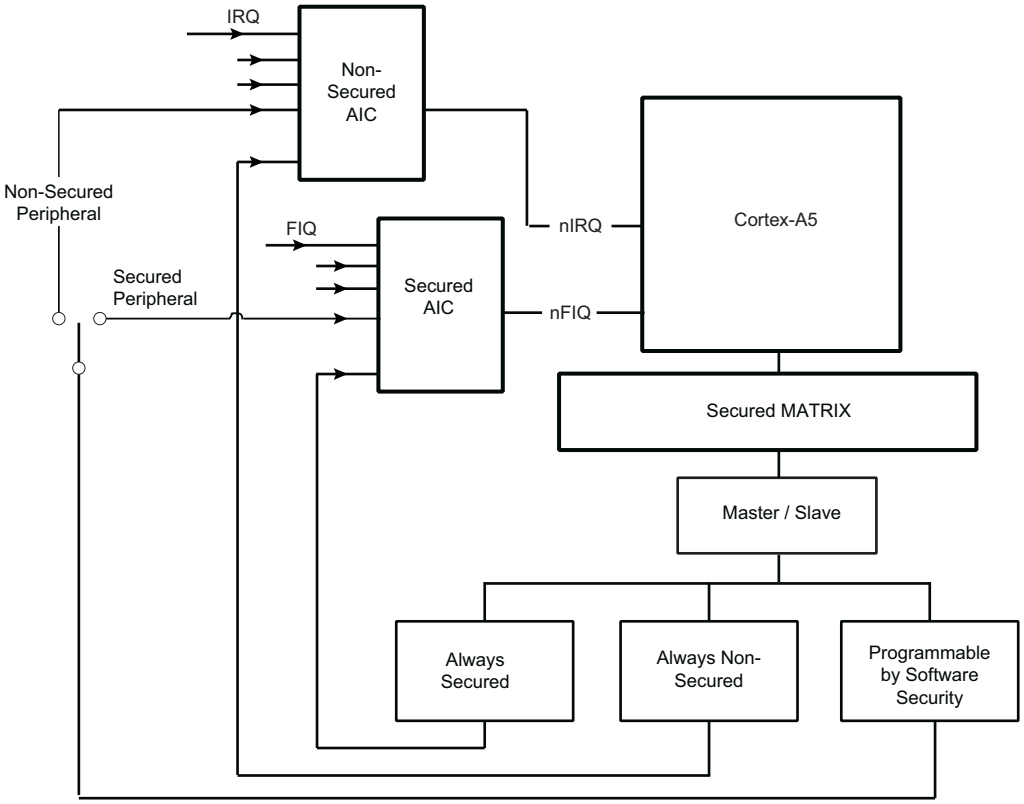
20.4 Application Block Diagram

Figure 20-2. Description of the Application Block



20.5 AIC Detailed Block Diagram

Figure 20-3. AIC Detailed Block Diagram



20.6 I/O Line Description

Table 20-1. I/O Line Description

Pin Name	Pin Description	Type
FIQ	Fast Interrupt	Input
IRQ0–IRQn	Interrupt 0–Interrupt n	Input

20.7 Product Dependencies

20.7.1 I/O Lines

The interrupt signals FIQ and IRQ0 to IRQn are normally multiplexed through the PIO controllers. Depending on the features of the PIO controller used in the product, the pins must be programmed in accordance with their assigned interrupt functions. This is not applicable when the PIO controller used in the product is transparent on the input path.

Table 20-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
AIC	FIQ	PB4	C
AIC	FIQ	PC8	C
AIC	FIQ	PC9	A
AIC	FIQ	PD3	B
AIC	IRQ	PA12	B
AIC	IRQ	PA21	A
AIC	IRQ	PB3	C
AIC	IRQ	PD31	C

20.7.2 Power Management

The Advanced Interrupt Controller is continuously clocked. The Power Management Controller has no effect on the Advanced Interrupt Controller behavior.

The assertion of the Advanced Interrupt Controller outputs, either nIRQ or nFIQ, wakes up the ARM processor while it is in Idle mode. The General Interrupt Mask feature enables the AIC to wake up the processor without asserting the interrupt line of the processor, thus providing synchronization of the processor on an event.

20.7.3 Interrupt Sources

Interrupt Source 0 is always located at FIQ. If the product does not feature any FIQ pin, Interrupt Source 0 cannot be used.

Interrupt Source 1 is always located at System Interrupt. This is the result of the OR-wiring of the system peripheral interrupt lines. When a system interrupt occurs, the service routine must first distinguish the cause of the interrupt. This is performed by reading successively the status registers of the above-mentioned system peripherals.

Interrupt sources 2 to 73 and 75 to 127 can either be connected to the interrupt outputs of an embedded user peripheral, or to external interrupt lines. The external interrupt lines can be connected either directly or through the PIO Controller.

PIO controllers are considered as user peripherals in the scope of interrupt handling. Accordingly, the PIO controller interrupt lines are connected to interrupt sources 2 to 73 and 75 to 127.

The peripheral identification defined at the product level corresponds to the interrupt source number (as well as the bit number controlling the clock of the peripheral). Consequently, to simplify the description of the functional operations and the user interface, the interrupt sources are named FIQ, SYS, and PID2 to PID73 and PID75 to PID127.

AIC0 manages all Non-Secure Interrupts including IRQn; AIC1 manages all Secure Interrupts including FIQ.

Each AIC has its own User Interface. The user should pay attention to use the relevant user interface for each source.

20.8 Functional Description

20.8.1 Interrupt Source Control

20.8.1.1 Interrupt Source Mode

The Advanced Interrupt Controller independently programs each interrupt source. The SRCTYPE field of the Source Mode Register (AIC_SMR) selects the interrupt condition of the interrupt source selected by the INTSEL field of the Source Select Register (AIC_SSR).

Note: Configuration registers such as AIC_SMR and AIC_SSR return the values corresponding to the interrupt source selected by INTSEL.

The internal interrupt sources wired on the interrupt outputs of the embedded peripherals can be programmed either in Level-Sensitive mode or in Edge-Triggered mode. The active level of the internal interrupts is not important for the user.

The external interrupt sources can be programmed either in High Level-Sensitive or Low Level-Sensitive modes, or in Positive Edge-Triggered or Negative Edge-Triggered modes.

20.8.1.2 Interrupt Source Enabling

Each interrupt source, including the FIQ in source 0, can be enabled or disabled by using the command registers Interrupt Enable Command Register (AIC_IECR) and Interrupt Disable Command Register (AIC_IDCR). The interrupt mask of the selected interrupt source can be read in the Interrupt Mask Register (AIC_IMR). A disabled interrupt does not affect servicing of other interrupts.

20.8.1.3 Interrupt Clearing and Setting

All interrupt sources programmed to be edge-triggered (including the FIQ in source 0) can be individually set or cleared by writing respectively the Interrupt Set Command Register (AIC_ISCR) and Interrupt Clear Command Register (AIC_ICCR). Clearing or setting interrupt sources programmed in Level-Sensitive mode has no effect.

The clear operation is perfunctory, as the software must perform an action to reset the “memorization” circuitry activated when the source is programmed in Edge-Triggered mode. However, the set operation is available for auto-test or software debug purposes. It can also be used to execute an AIC-implementation of a software interrupt.

The AIC features an automatic clear of the current interrupt when AIC_IVR (Interrupt Vector Register) is read. Only the interrupt source being detected by the AIC as the current interrupt is affected by this operation. (See Section 20.8.3.1 “Priority Controller”.) The automatic clear reduces the operations required by the interrupt service routine entry code to read AIC_IVR.

The automatic clear of interrupt source 0 is performed when AIC_FVR is read.

20.8.1.4 Interrupt Status

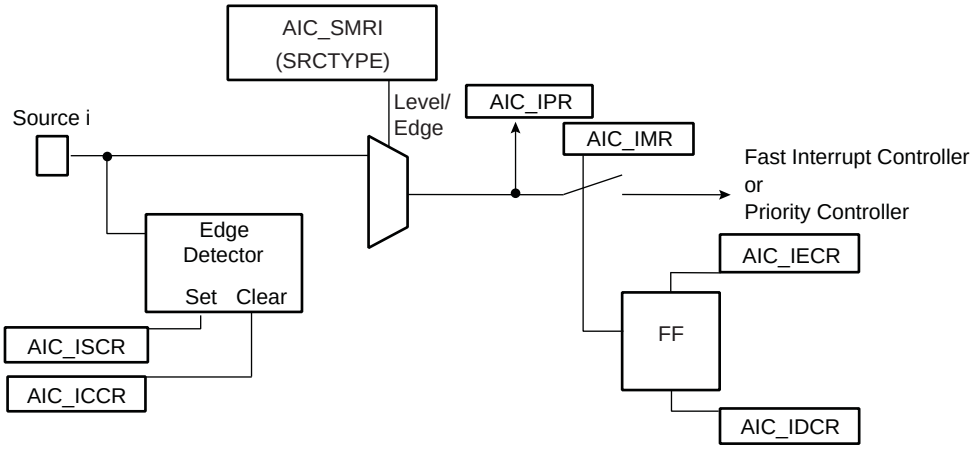
Interrupt Pending Registers (AIC_IPR) represent the state of the interrupt lines, whether they are masked or not. AIC_IMR can be used to define the mask of the interrupt lines.

The Interrupt Status Register (AIC_ISR) reads the number of the current interrupt (see Section 20.8.3.1 “Priority Controller”) and the Core Interrupt Status Register (AIC_CISR) gives an image of the nIRQ and nFIQ signals driven on the processor.

Each status referred to above can be used to optimize the interrupt handling of the systems.

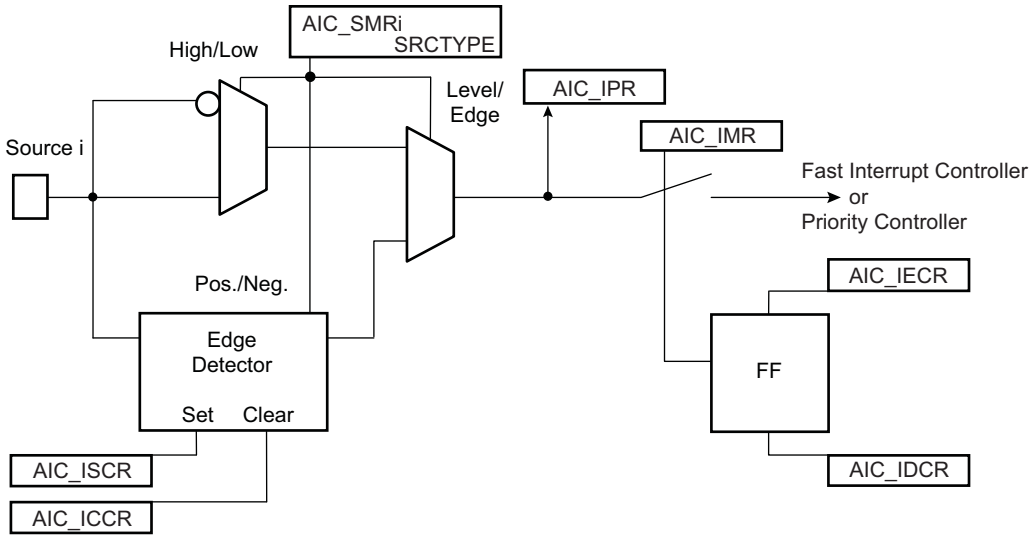
20.8.1.5 Internal Interrupt Source Input Stage

Figure 20-4. Internal Interrupt Source Input Stage



20.8.1.6 External Interrupt Source Input Stage

Figure 20-5. External Interrupt Source Input Stage



20.8.2 Interrupt Latencies

Global interrupt latencies depend on several parameters, including:

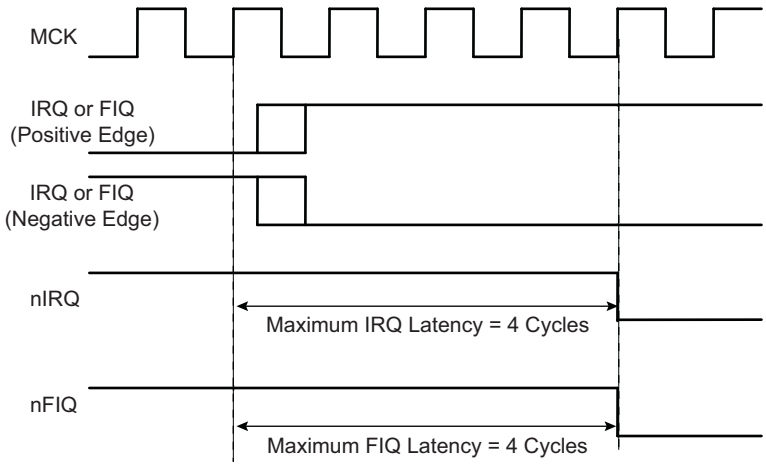
- The time the software masks the interrupts
- Occurrence, either at the processor level or at the AIC level
- The execution time of the instruction in progress when the interrupt occurs
- The treatment of higher priority interrupts and the resynchronization of the hardware signals

This section addresses hardware resynchronizations only. It gives details about the latency times between the events on an external interrupt leading to a valid interrupt (edge or level) or the assertion of an internal interrupt source and the assertion of the nIRQ or nFIQ line on the processor. The resynchronization time depends on the programming of the interrupt source and on its type (internal or external). For the standard interrupt, resynchronization times are given assuming there is no higher priority in progress.

The PIO Controller multiplexing has no effect on the interrupt latencies of the external interrupt sources.

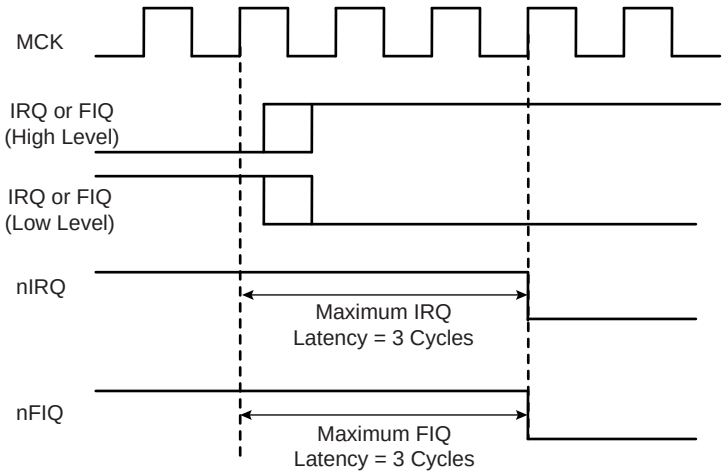
20.8.2.1 External Interrupt Edge Triggered Source

Figure 20-6. External Interrupt Edge Triggered Source



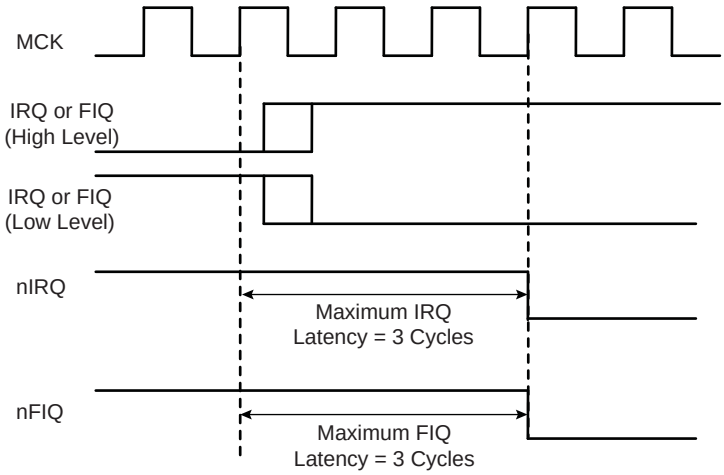
20.8.2.2 External Interrupt Level Sensitive Source

Figure 20-7. External Interrupt Level Sensitive Source



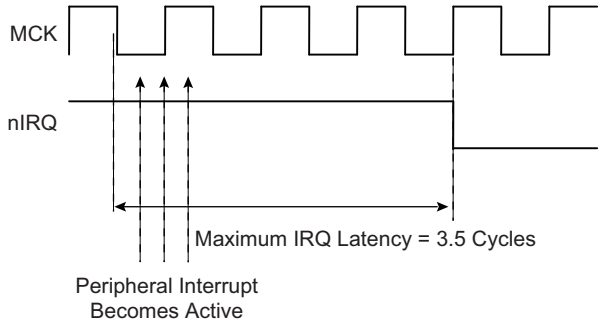
20.8.2.3 Internal Interrupt Edge Triggered Source

Figure 20-8. Internal Interrupt Edge Triggered Source



20.8.2.4 Internal Interrupt Level Sensitive Source

Figure 20-9. Internal Interrupt Level Sensitive Source



20.8.3 Normal Interrupt

20.8.3.1 Priority Controller

An 8-level priority controller drives the nIRQ line of the processor, depending on the interrupt conditions occurring on the interrupt sources 1 to 127.

Each interrupt source has a programmable priority level of 7 to 0, which is user-definable by writing the PRIOR field of AIC_SMR (Source Mode Register). Level 7 is the highest priority and level 0 the lowest.

As soon as an interrupt condition occurs, as defined by the SRCTYPE field in AIC_SMR (Source Mode Register), the nIRQ line is asserted. As a new interrupt condition might have happened on other interrupt sources since the nIRQ has been asserted, the priority controller determines the current interrupt at the time AIC_IVR (Interrupt Vector Register) is read. The read of AIC_IVR is the entry point of the interrupt handling which allows the AIC to consider that the interrupt has been taken into account by the software.

The current priority level is defined as the priority level of the current interrupt.

If several interrupt sources of equal priority are pending and enabled when AIC_IVR is read, the interrupt with the lowest interrupt source number is serviced first.

The nIRQ line can be asserted only if an interrupt condition occurs on an interrupt source with a higher priority. If an interrupt condition happens (or is pending) during the interrupt treatment in progress, it is delayed until the software indicates to the AIC the end of the current service by writing AIC_EOICR (End of Interrupt Command Register). The write of AIC_EOICR is the exit point of the interrupt handling.

20.8.3.2 Interrupt Nesting

The priority controller utilizes interrupt nesting in order for the high priority interrupt to be handled during the service of lower priority interrupts. This requires the interrupt service routines of the lower interrupts to re-enable the interrupt at the processor level.

When an interrupt of a higher priority happens during an already occurring interrupt service routine, the nIRQ line is re-asserted. If the interrupt is enabled at the core level, the current execution is interrupted and the new interrupt service routine should read AIC_IVR. At this time, the current interrupt number and its priority level are pushed into an embedded hardware stack, so that they are saved and restored when the higher priority interrupt servicing is finished and AIC_EOICR is written.

The AIC is equipped with an 8-level wide hardware stack in order to support up to eight interrupt nestings to match the eight priority levels.

20.8.3.3 Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and the associated status bits.

It is assumed that:

1. The Advanced Interrupt Controller has been programmed, AIC_SVR registers are loaded with corresponding interrupt service routine addresses and interrupts are enabled.
2. The instruction at the ARM interrupt exception vector address is required to work with the vectoring. Load the PC with the absolute address of the interrupt handler.

When nIRQ is asserted, if the bit “I” of CPSR is 0, the sequence is as follows:

1. The CPSR is stored in SPSR_irq, the current value of the Program Counter is loaded in the Interrupt link register (R14_irq) and the Program Counter (R15) is loaded with 0x18. In the following cycle during fetch at address 0x1C, the ARM core adjusts R14_irq, decrementing it by four.
2. The ARM core enters Interrupt mode, if it has not already done so.
3. When the instruction loaded at address 0x18 is executed, the program counter is loaded with the value read in AIC_IVR. Reading AIC_IVR has the following effects:
   - Sets the current interrupt to be the pending and enabled interrupt with the highest priority. The current level is the priority level of the current interrupt.
   - deasserts the nIRQ line on the processor. Even if vectoring is not used, AIC_IVR must be read in order to deassert nIRQ.
   - Automatically clears the interrupt, if it has been programmed to be edge-triggered.
   - Pushes the current level and the current interrupt number on to the stack.
   - Returns the value written in AIC_SVR corresponding to the current interrupt.
4. The previous step has the effect of branching to the corresponding interrupt service routine. This should start by saving the link register (R14_irq) and SPSR_IRQ. The link register must be decremented by four when it is saved if it is to be restored directly into the program counter at the end of the interrupt. For example, the instruction `SUB PC, LR, #4` may be used.
5. Further interrupts can then be unmasked by clearing the “I” bit in CPSR, allowing re-assertion of the nIRQ to be taken into account by the core. This can happen if an interrupt with a higher priority than the current interrupt occurs.
6. The interrupt handler can then proceed as required, saving the registers that will be used and restoring them at the end. During this phase, an interrupt of higher priority than the current level will restart the sequence from step 1.

Note: If the interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase.

7. The “I” bit in CPSR must be set in order to mask interrupts before exiting to ensure that the interrupt is completed in an orderly manner.
8. The End of Interrupt Command Register (AIC_EOICR) must be written in order to indicate to the AIC that the current interrupt is finished. This causes the current level to be popped from the stack, restoring the previous current level if one exists on the stack. If another interrupt is pending, with lower or equal priority than the old current level but with higher priority than the new current level, the nIRQ line is re-asserted, but the interrupt sequence does not immediately start because the “I” bit is set in the core. SPSR_irq is restored. Finally, the saved value of the link register is restored directly into the PC. This has the effect of returning from the interrupt to whatever was being executed before, and of loading the CPSR with the stored SPSR, masking or unmasking the interrupts depending on the state saved in SPSR_irq.

Note: The “I” bit in SPSR is significant. If it is set, it indicates that the ARM core was on the verge of masking an interrupt when the mask instruction was interrupted. Hence, when SPSR is restored, the mask instruction is completed (interrupt is masked).

20.8.4 Fast Interrupt

20.8.4.1 Fast Interrupt Source

Interrupt source 0 is the only source which can raise a fast interrupt request to the processor. Interrupt source 0 is generally connected to a FIQ pin of the product, either directly or through a PIO Controller.

20.8.4.2 Fast Interrupt Control

The fast interrupt logic of the AIC has no priority controller. The mode of interrupt source 0 is programmed with AIC_SMR and INTSEL = 0; the PRIOR field of this register is not used even if it reads what has been written. The SRCTYPE field of AIC_SMR enables programming the fast interrupt source to be positive-edge triggered or negative-edge triggered or high-level sensitive or low-level sensitive.

Writing 0x1 in AIC_IECR (Interrupt Enable Command Register) and AIC_IDCR (Interrupt Disable Command Register) respectively enables and disables the fast interrupt when INTSEL = 0. Bit 0 of AIC_IMR indicates whether the fast interrupt is enabled or disabled.

20.8.4.3 Fast Interrupt Handlers

This section gives an overview of the fast interrupt handling sequence when using the AIC. It is assumed that the programmer understands the architecture of the ARM processor, and especially the Processor Interrupt modes and associated status bits.

Assuming that:

1. The Advanced Interrupt Controller has been programmed, AIC_SVR is loaded with the fast interrupt service routine address, and interrupt source 0 is enabled.
2. The Instruction at address 0x1C (FIQ exception vector address) is required to vector the fast interrupt. Load the PC with the absolute address of the interrupt handler.
3. The user does not need nested fast interrupts.

When nFIQ is asserted, if bit “F” of CPSR is 0, the sequence is:

1. The CPSR is stored in SPSR_fiq, the current value of the program counter is loaded in the FIQ link register (R14_FIQ) and the program counter (R15) is loaded with 0x1C. In the following cycle, during fetch at address 0x20, the ARM core adjusts R14_fiq, decrementing it by four.
2. The ARM core enters FIQ mode.
3. The routine must read AIC1_CISR to know if the interrupt is the FIQ or a Secure Internal interrupt.

```
ldr         r1, =REG_SAIC_CISR
ldr         r1, [r1]
cmp         r1, #AIC_CISR_NFIQ
beq         get_fiqvec_addr
```

If FIQ is active, it is processed in priority, even if another interrupt is active.

```
get_irqvec_addr
        ldr         r14, =REG_SAIC_IVR
        b           read_vec
get_fiqvec_addr
        ldr         r14, =REG_SAIC_FVR
 read_vec
        ldr         r0, [r14]
```

Now r0 contains the correct vector address, IVR for a Secure Internal interrupt or FVR for FIQ.

The system can branch to the routine pointed to by r0.

```
FIQ_Handler_Branch
        mov     r14, pc
        bx      r0
```

4. The previous step enables branching to the corresponding interrupt service routine. It is not necessary to save the link register R14_fiq and SPSR_fiq if nested fast interrupts are not needed.
5. The Interrupt Handler can then proceed as required. It is not necessary to save registers R8 to R13 because the FIQ mode has its own dedicated registers and registers R8 to R13 are banked. The other registers, R0 to R7, must be saved before being used, and restored at the end (before the next step).

Note: If the fast interrupt is programmed to be level-sensitive, the source of the interrupt must be cleared during this phase in order to deassert interrupt source 0.

6. Finally, Link Register R14_fiq is restored into the PC after decrementing it by four (with instruction `SUB PC, LR, #4` for example). This has the effect of returning from the interrupt to whatever was being executed before, loading the CPSR with the SPSR and masking or unmasking the fast interrupt depending on the state saved in the SPSR.

Note: The “F” bit in SPSR is significant. If it is set, it indicates that the ARM core was just about to mask FIQ interrupts when the mask instruction was interrupted. Hence, when the SPSR is restored, the interrupted instruction is completed (FIQ is masked).

Another way to handle the fast interrupt is to map the interrupt service routine at the address of the ARM vector 0x1C. This method does not use vectoring, so that reading AIC_FVR must be performed at the very beginning of the handler operation. However, this method saves the execution of a branch instruction.

20.8.5 Protect Mode

The Protect mode is used to read the Interrupt Vector Register without performing the associated automatic operations. This is necessary when working with a debug system. When a debugger, working either with a Debug Monitor or the ARM processor's ICE, stops the applications and updates the opened windows, it might read the AIC User Interface and thus the IVR. This has adverse consequences:

- If an enabled interrupt with a higher priority than the current one is pending, it is stacked.
- If there is no enabled pending interrupt, the spurious vector is returned.

In either case, an End of Interrupt command is necessary to acknowledge and restore the context of the AIC. This operation is generally not performed by the debug system, as the debug system would become strongly intrusive and cause the application to enter an undesired state.

This is avoided by using the Protect mode. Writing PROT in AIC_DCR (Debug Control Register) at 0x1 enables the Protect mode.

When the Protect mode is enabled, the AIC performs interrupt stacking only when a write access is performed on AIC_IVR. Therefore, the Interrupt Service Routines must write (arbitrary data) to AIC_IVR just after reading it. The new context of the AIC, including the value of AIC_ISR, is updated with the current interrupt only when AIC_IVR is written.

An AIC_IVR read on its own (e.g., by a debugger) modifies neither the AIC context nor AIC_ISR. Extra AIC_IVR reads perform the same operations. However, it is recommended to not stop the processor between the read and the write of AIC_IVR of the interrupt service routine to make sure the debugger does not modify the AIC context.

To summarize, in normal operating mode, the read of AIC_IVR performs the following operations within the AIC:

1. Calculates active interrupt (higher than current or spurious).
2. Determines and returns the vector of the active interrupt.
3. Memorizes the interrupt.
4. Pushes the current priority level onto the internal stack.
5. Acknowledges the interrupt.

However, while the Protect mode is activated, only operations 1 to 3 are performed when AIC_IVR is read. Operations 4 and 5 are only performed by the AIC when AIC_IVR is written.

Software that has been written and debugged using the Protect mode runs correctly in normal mode without modification. However, in normal mode, the AIC_IVR write has no effect and can be removed to optimize the code.

20.8.6 Spurious Interrupt

The Advanced Interrupt Controller features a protection against spurious interrupts. A spurious interrupt is defined as being the assertion of an interrupt source long enough for the AIC to assert the nIRQ, but no longer present when AIC_IVR is read. This is most prone to occur when:

- An external interrupt source is programmed in Level-Sensitive mode and an active level occurs for only a short time.
- An internal interrupt source is programmed in level-sensitive and the output signal of the corresponding embedded peripheral is activated for a short time (as is the case for the watchdog).
- An interrupt occurs just a few cycles before the software begins to mask it, thus resulting in a pulse on the interrupt source.

The AIC detects a spurious interrupt at the time AIC_IVR is read while no enabled interrupt source is pending. When this happens, the AIC returns the value stored by the programmer in the Spurious Vector Register (AIC_SPU). The programmer must store the address of a spurious interrupt handler in AIC_SPU as part of the application, to enable an as fast as possible return to the normal execution flow. This handler writes in AIC_EOICR and performs a return from interrupt.

20.8.7 General Interrupt Mask

The AIC features a General Interrupt Mask bit (GMSK in AIC_DCR) to prevent interrupts from reaching the processor. Both the nIRQ and the nFIQ lines are driven to their inactive state if the GMSK is set. However, this mask does not prevent waking up the processor if it has entered Idle mode. This function facilitates synchronizing the processor on a next event and, as soon as the event occurs, performs subsequent operations without having to handle an interrupt. It is strongly recommended to use this mask with caution.

20.8.8 Register Write Protection

To prevent any single software error from corrupting AIC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the AIC Write Protection Mode Register (AIC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the AIC Write Protection Status Register (AIC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading AIC_WPSR.

The following registers can be write-protected:

- AIC Source Mode Register
- AIC Source Vector Register
- AIC Spurious Interrupt Vector Register
- AIC Debug Control Register

20.9 Advanced Interrupt Controller (AIC) User Interface

Table 20-3. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Source Select Register	AIC_SSR	Read/Write	0x0
0x04	Source Mode Register	AIC_SMR	Read/Write	0x0
0x08	Source Vector Register	AIC_SVR	Read/Write	0x0
0x0C	Reserved	–	–	–
0x10	Interrupt Vector Register	AIC_IVR	Read-only	0x0
0x14	FIQ Vector Register	AIC_FVR	Read-only	0x0
0x18	Interrupt Status Register	AIC_ISR	Read-only	0x0
0x1C	Reserved	–	–	–
0x20	Interrupt Pending Register 0(2)	AIC_IPR0	Read-only	0x0(1)
0x24	Interrupt Pending Register 1(2)	AIC_IPR1	Read-only	0x0(1)
0x28	Interrupt Pending Register 2(2)	AIC_IPR2	Read-only	0x0(1)
0x2C	Interrupt Pending Register 3(2)	AIC_IPR3	Read-only	0x0(1)
0x30	Interrupt Mask Register	AIC_IMR	Read-only	0x0
0x34	Core Interrupt Status Register	AIC_CISR	Read-only	0x0
0x38	End of Interrupt Command Register	AIC_EOICR	Write-only	–
0x3C	Spurious Interrupt Vector Register	AIC_SPU	Read/Write	0x0
0x40	Interrupt Enable Command Register	AIC_IECR	Write-only	–
0x44	Interrupt Disable Command Register	AIC_IDCR	Write-only	–
0x48	Interrupt Clear Command Register	AIC_ICCR	Write-only	–
0x4C	Interrupt Set Command Register	AIC_ISCR	Write-only	–
0x50–0x5C	Reserved	–	–	–
0x60–0x68	Reserved	–	–	–
0x6C	Debug Control Register	AIC_DCR	Read/Write	0x0
0x70–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	AIC_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	AIC_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–

Notes: 1. The reset value of this register depends on the level of the external interrupt source. All other sources are cleared at reset, thus not pending.

2. PID2...PID127 bit fields refer to the identifiers as defined in the Peripheral Identifiers section.

20.9.1 AIC Source Select Register

Name: AIC_SSR

Address: 0xFC020000 (AIC), 0xF803C000 (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	INTSEL						

- **INTSEL: Interrupt Line Selection**

0–127 = Selects the interrupt line to handle.

See Section 20.8.1.1 “Interrupt Source Mode”.

20.9.2 AIC Source Mode Register

Name: AIC_SMR

Address: 0xFC020004 (AIC), 0xF803C004 (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	SRCTYPE		–	–	PRIOR		

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

- **PRIOR: Priority Level**

Programs the priority level of the source selected by INTSEL except FIQ source (source 0).

The priority level can be between 0 (lowest) and 7 (highest).

The priority level is not used for the FIQ.

- **SRCTYPE: Interrupt Source Type**

The active level or edge is not programmable for the internal interrupt source selected by INTSEL.

Value	Name	Description
0	INT_LEVEL_SENSITIVE	High level Sensitive for internal source Low level Sensitive for external source
1	INT_EDGE_TRIGGERED	Positive edge triggered for internal source Negative edge triggered for external source
2	EXT_HIGH_LEVEL	High level Sensitive for internal source High level Sensitive for external source
3	EXT_POSITIVE_EDGE	Positive edge triggered for internal source Positive edge triggered for external source

20.9.3 AIC Source Vector Register

Name: AIC_SVR

Address: 0xFC020008 (AIC), 0xF803C008 (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
VECTOR							

23	22	21	20	19	18	17	16
VECTOR							

15	14	13	12	11	10	9	8
VECTOR							

7	6	5	4	3	2	1	0
VECTOR							

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

- **VECTOR: Source Vector**

The user may store in this register the address of the corresponding handler for the interrupt source selected by INTSEL.

20.9.4 AIC Interrupt Vector Register

Name: AIC_IVR

Address: 0xFC020010 (AIC), 0xF803C010 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
IRQV							

23	22	21	20	19	18	17	16
IRQV							

15	14	13	12	11	10	9	8
IRQV							

7	6	5	4	3	2	1	0
IRQV							

- **IRQV: Interrupt Vector Register**

The Interrupt Vector Register contains the vector programmed by the user in the Source Vector Register corresponding to the current interrupt.

The Source Vector Register is indexed using the current interrupt number when the Interrupt Vector Register is read.

When there is no current interrupt, the Interrupt Vector Register reads the value stored in AIC_SPU.

20.9.5 AIC FIQ Vector Register

Name: AIC_FVR

Address: 0xFC020014 (AIC), 0xF803C014 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
FIQV							

23	22	21	20	19	18	17	16
FIQV							

15	14	13	12	11	10	9	8
FIQV							

7	6	5	4	3	2	1	0
FIQV							

- **FIQV: FIQ Vector Register**

The FIQ Vector Register contains the vector programmed by the user in the Source Vector Register when INTSEL = 0. When there is no fast interrupt, the FIQ Vector Register reads the value stored in AIC_SPU.

20.9.6 AIC Interrupt Status Register

Name: AIC_ISR

Address: 0xFC020018 (AIC), 0xF803C018 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	IRQID						

- **IRQID: Current Interrupt Identifier**

The Interrupt Status Register returns the current interrupt source number.

20.9.7 AIC Interrupt Pending Register 0

Name: AIC_IPR0

Address: 0xFC020020 (AIC), 0xF803C020 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	PID1	FIQ

- **FIQ: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

20.9.8 AIC Interrupt Pending Register 1

Name: AIC_IPR1

Address: 0xFC020024 (AIC), 0xF803C024 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

20.9.9 AIC Interrupt Pending Register 2

Name: AIC_IPR2

Address: 0xFC020028 (AIC), 0xF803C028 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID95	PID94	PID93	PID92	PID91	PID90	PID89	PID88
23	22	21	20	19	18	17	16
PID87	PID86	PID85	PID84	PID83	PID82	PID81	PID80
15	14	13	12	11	10	9	8
PID79	PID78	PID77	PID76	PID75	SYS	PID73	PID72
7	6	5	4	3	2	1	0
PID71	PID70	PID69	PID68	PID67	PID66	PID65	PID64

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

- **SYS: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

20.9.10 AIC Interrupt Pending Register 3

Name: AIC_IPR3

Address: 0xFC02002C (AIC), 0xF803C02C (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
PID127	PID126	PID125	PID124	PID123	PID122	PID121	PID120
23	22	21	20	19	18	17	16
PID119	PID118	PID117	PID116	PID115	PID114	PID113	PID112
15	14	13	12	11	10	9	8
PID111	PID110	PID109	PID108	PID107	PID106	PID105	PID104
7	6	5	4	3	2	1	0
PID103	PID102	PID101	PID100	PID99	PID98	PID97	PID96

- **PIDx: Interrupt Pending**

0: The corresponding interrupt is not pending.

1: The corresponding interrupt is pending.

20.9.11 AIC Interrupt Mask Register

Name: AIC_IMR

Address: 0xFC020030 (AIC), 0xF803C030 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	INTM

- **INTM: Interrupt Mask**

0: The interrupt source selected by INTSEL is disabled.

1: The interrupt source selected by INTSEL is enabled.

20.9.12 AIC Core Interrupt Status Register

Name: AIC_CISR

Address: 0xFC020034 (AIC), 0xF803C034 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	NIRQ	NFIQ

- **NFIQ: NFIQ Status**

0: nFIQ line is deactivated.

1: nFIQ line is active.

- **NIRQ: NIRQ Status**

0: nIRQ line is deactivated.

1: nIRQ line is active.

20.9.13 AIC End of Interrupt Command Register

Name: AIC_EOICR

Address: 0xFC020038 (AIC), 0xF803C038 (SAIC)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENDIT

- **ENDIT: Interrupt Processing Complete Command**

The End of Interrupt Command Register is used by the interrupt routine to indicate that the interrupt treatment is complete. Any value can be written because it is only necessary to make a write to this register location to signal the end of interrupt treatment.

20.9.14 AIC Spurious Interrupt Vector Register

Name: AIC_SPU

Address: 0xFC02003C (AIC), 0xF803C03C (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
SIVR							

23	22	21	20	19	18	17	16
SIVR							

15	14	13	12	11	10	9	8
SIVR							

7	6	5	4	3	2	1	0
SIVR							

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

- **SIVR: Spurious Interrupt Vector Register**

The user may store the address of a spurious interrupt handler in this register. The written value is returned in AIC_IVR in case of a spurious interrupt, or in AIC_FVR in case of a spurious fast interrupt.

20.9.15 AIC Interrupt Enable Command Register

Name: AIC_IECR

Address: 0xFC020040 (AIC), 0xF803C040 (SAIC)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	INTEN

- **INTEN: Interrupt Enable**

0: No effect.

1: Enables the interrupt source selected by INTSEL.

20.9.16 AIC Interrupt Disable Command Register

Name: AIC_IDCR

Address: 0xFC020044 (AIC), 0xF803C044 (SAIC)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	INTD

- **INTD: Interrupt Disable**

0: No effect.

1: Disables the interrupt source selected by INTSEL.

20.9.17 AIC Interrupt Clear Command Register

Name: AIC_ICCR

Address: 0xFC020048 (AIC), 0xF803C048 (SAIC)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	INTCLR

- **INTCLR: Interrupt Clear**

Clears one the following depending on the setting of the INTSEL bit FIQ, SYS, PID2-PID73 and PID75-PID127

0: No effect.

1: Clears the interrupt source selected by INTSEL.

20.9.18 AIC Interrupt Set Command Register

Name: AIC_ISCR

Address: 0xFC02004C (AIC), 0xF803C04C (SAIC)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	INTSET

- **INTSET: Interrupt Set**

0: No effect.

1: Sets the interrupt source selected by INTSEL.

20.9.19 AIC Debug Control Register

Name: AIC_DCR

Address: 0xFC02006C (AIC), 0xF803C06C (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	GMSK	PROT

This register can only be written if the WPEN bit is cleared in the AIC Write Protection Mode Register.

- **PROT: Protection Mode**

0: The Protection mode is disabled.

1: The Protection mode is enabled.

- **GMSK: General Interrupt Mask**

0: The nIRQ and nFIQ lines are normally controlled by the AIC.

1: The nIRQ and nFIQ lines are tied to their inactive state.

20.9.20 AIC Write Protection Mode Register

Name: AIC_WPMR

Address: 0xFC0200E4 (AIC), 0xF803C0E4 (SAIC)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x414943 (“AIC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x414943 (“AIC” in ASCII).

See Section 20.8.8 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x414943	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

20.9.21 AIC Write Protection Status Register

Name: AIC_WPSR

Address: 0xFC0200E8 (AIC), 0xF803C0E8 (SAIC)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of AIC_WPSR.

1: A write protection violation has occurred since the last read of AIC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

21. Watchdog Timer (WDT)

21.1 Description

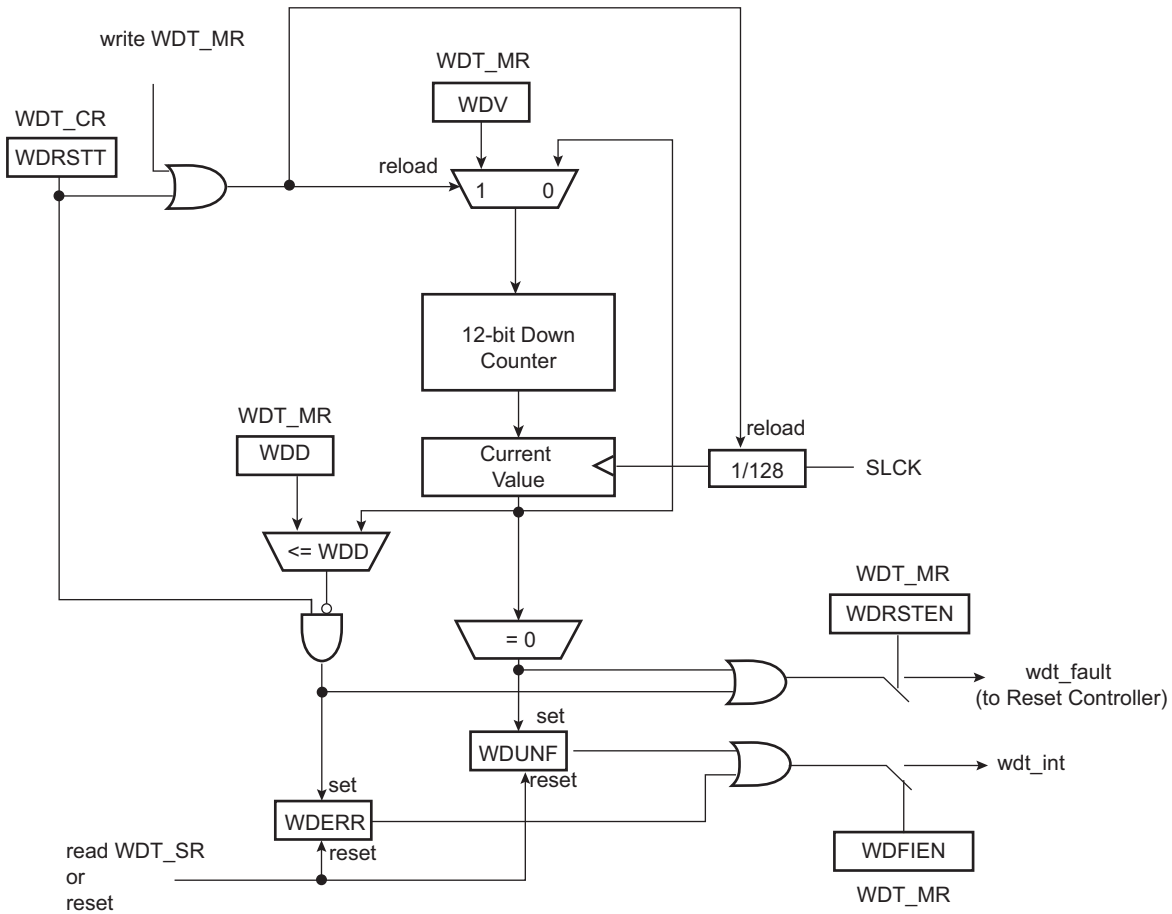
The Watchdog Timer (WDT) is used to prevent system lock-up if the software becomes trapped in a deadlock. It features a 12-bit down counter that allows a watchdog period of up to 16 seconds (slow clock around 32 kHz). It can generate a general reset or a processor reset only. In addition, it can be stopped while the processor is in Debug mode or Sleep mode (Idle mode).

21.2 Embedded Characteristics

- 12-bit Key-protected Programmable Counter
- Watchdog Clock is Independent from Processor Clock
- Provides Reset or Interrupt Signals to the System
- Counter May Be Stopped while the Processor is in Debug State or in Idle Mode

21.3 Block Diagram

Figure 21-1. Watchdog Timer Block Diagram



21.4 Functional Description

The Watchdog Timer is used to prevent system lock-up if the software becomes trapped in a deadlock. It is supplied with VDDCORE. It restarts with initial values on processor reset.

The watchdog is built around a 12-bit down counter, which is loaded with the value defined in the field WDV of the Mode Register (WDT_MR). The Watchdog Timer uses the slow clock divided by 128 to establish the maximum watchdog period to be 16 seconds (with a typical slow clock of 32.768 kHz).

After a processor reset, the value of WDV is 0xFFF, corresponding to the maximum value of the counter with the external reset generation enabled (field WDRSTEN at 1 after a backup reset). This means that a default watchdog is running at reset, i.e., at powerup. The user can either disable the WDT by setting bit WDT_MR.WDDIS or reprogram the WDT to meet the maximum watchdog period the application requires.

When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

If the watchdog is restarted by writing into the Control Register (WDT_CR), WDT_MR must not be programmed during a period of time of three slow clock periods following the WDT_CR write access. In any case, programming a new value in WDT_MR automatically initiates a restart instruction.

WDT_MR can be written until a LOCKMR command is issued in WDT_CR. Only a processor reset resets it. Writing WDT_MR reloads the timer with the newly programmed mode parameters.

In normal operation, the user reloads the watchdog at regular intervals before the timer underflow occurs, by setting bit WDT_CR.WDRSTT. The watchdog counter is then immediately reloaded from WDT_MR and restarted, and the slow clock 128 divider is reset and restarted. WDT_CR is write-protected. As a result, writing WDT_CR without the correct hard-coded key has no effect. If an underflow does occur, the “wdt_fault” signal to the Reset Controller is asserted if bit WDT_MR.WDRSTEN is set. Moreover, the bit WDUNF is set in the Status Register (WDT_SR).

The reload of the watchdog must occur while the watchdog counter is within a window between 0 and WDD. WDD is defined in WDT_MR.

Any attempt to restart the watchdog while the watchdog counter is between WDV and WDD results in a watchdog error, even if the watchdog is disabled. The bit WDT_SR.WDERR is updated and the “wdt_fault” signal to the Reset Controller is asserted.

Note that this feature can be disabled by programming a WDD value greater than or equal to the WDV value. In such a configuration, restarting the Watchdog Timer is permitted in the whole range [0; WDV] and does not generate an error. This is the default configuration on reset (the WDD and WDV values are equal).

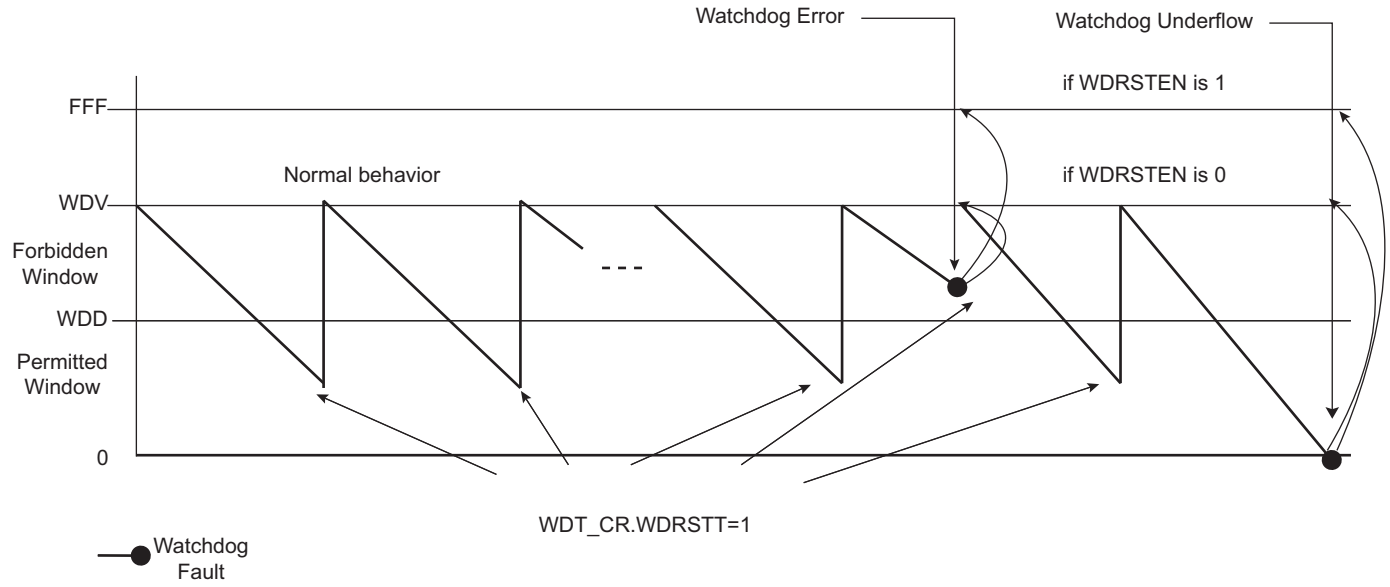
The status bits WDUNF (Watchdog Underflow) and WDERR (Watchdog Error) trigger an interrupt, provided the bit WDT_MR.WDFIEN is set. The signal “wdt_fault” to the Reset Controller causes a watchdog reset if the WDRSTEN bit is set as already explained in the Reset Controller documentation. In this case, the processor and the Watchdog Timer are reset, and the WDERR and WDUNF flags are reset.

If a reset is generated or if WDT_SR is read, the status bits are reset, the interrupt is cleared, and the “wdt_fault” signal to the reset controller is deasserted.

Writing WDT_MR reloads and restarts the down counter.

While the processor is in debug state or in Sleep mode, the counter may be stopped depending on the value programmed for the bits WDIDLEHLT and WDDBGHLT in WDT_MR.

Figure 21-2. Watchdog Behavior



21.5 Watchdog Timer (WDT) User Interface

Table 21-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	WDT_CR	Write-only	–
0x04	Mode Register	WDT_MR	Read/Write	0x3FFF_2FFF
0x08	Status Register	WDT_SR	Read-only	0x0000_0000

21.5.1 Watchdog Timer Control Register

Name: WDT_CR

Address: 0xF8048040

Access: Write-only

31	30	29	28	27	26	25	24
KEY							
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	LOCKMR	–	–	–	WDRSTT

Note: The WDT_CR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

- **WDRSTT: Watchdog Restart**

0: No effect.

1: Restarts the watchdog if KEY is written to 0xA5.

- **LOCKMR: Lock Mode Register Write Access**

0: No effect.

1: Locks the Mode Register (WDT_MR) if KEY is written to 0xA5, write access to WDT_MR has no effect.

- **KEY: Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

21.5.2 Watchdog Timer Mode Register

Name: WDT_MR

Address: 0xF8048044

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	WDIDLEHLT	WDDBGHLT	WDD			
23	22	21	20	19	18	17	16
WDD							
15	14	13	12	11	10	9	8
WDDIS	–	WDRSTEN	WDFIEN	WDV			
7	6	5	4	3	2	1	0
WDV							

Notes:
1. Write access to this register has no effect if the LOCKMR command is issued in WDT_CR (unlocked on hardware reset).
2. The WDT_MR register values must not be modified within three slow clock periods following a restart of the watchdog performed by a write access in WDT_CR. Any modification will cause the watchdog to trigger an end of period earlier than expected.

- **WDV: Watchdog Counter Value**

Defines the value loaded in the 12-bit watchdog counter.

- **WDFIEN: Watchdog Fault Interrupt Enable**

0: A watchdog fault (underflow or error) has no effect on interrupt.

1: A watchdog fault (underflow or error) asserts interrupt.

- **WDRSTEN: Watchdog Reset Enable**

0: A watchdog fault (underflow or error) has no effect on the resets.

1: A watchdog fault (underflow or error) triggers a watchdog reset.

- **WDDIS: Watchdog Disable**

0: Enables the Watchdog Timer.

1: Disables the Watchdog Timer.

Note: When setting the WDDIS bit, and while it is set, the fields WDV and WDD must not be modified.

- **WDD: Watchdog Delta Value**

Defines the permitted range for reloading the Watchdog Timer.

If the Watchdog Timer value is less than or equal to WDD, setting bit WDT_CR.WDRSTT restarts the timer.

If the Watchdog Timer value is greater than WDD, setting bit WDT_CR.WDRSTT causes a watchdog error.

- **WDDBGHLT: Watchdog Debug Halt**

0: The watchdog runs when the processor is in debug state.

1: The watchdog stops when the processor is in debug state.

- **WDIDLEHLT: Watchdog Idle Halt**

0: The watchdog runs when the system is in idle state.

1: The watchdog stops when the system is in idle state.

21.5.3 Watchdog Timer Status Register

Name: WDT_SR

Address: 0xF8048048

Access Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	WDERR	WDUNF

- **WDUNF: Watchdog Underflow (cleared on read)**

0: No watchdog underflow occurred since the last read of WDT_SR.

1: At least one watchdog underflow occurred since the last read of WDT_SR.

- **WDERR: Watchdog Error (cleared on read)**

0: No watchdog error occurred since the last read of WDT_SR.

1: At least one watchdog error occurred since the last read of WDT_SR.

22. Reset Controller (RSTC)

22.1 Description

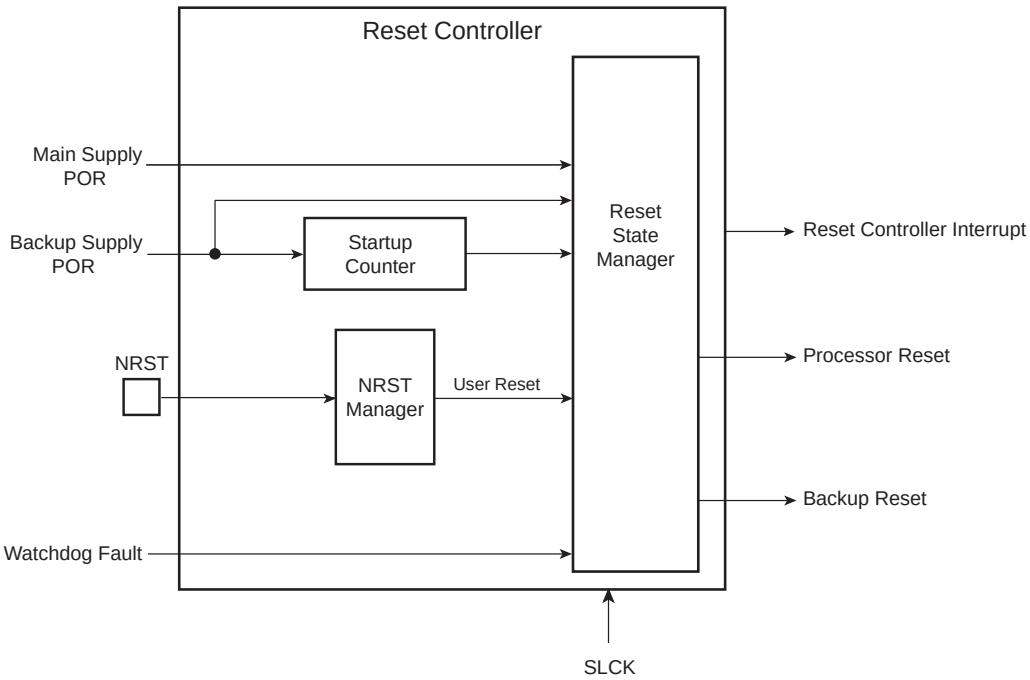
The Reset Controller (RSTC), based on power-on reset cells, handles all the resets of the system without any external components. It reports which reset occurred last.

22.2 Embedded Characteristics

- Manages All Resets of the System, Including
  - Processor Reset
  - Backed-up Peripheral Reset
- Based on 2 Embedded Power-on Reset Cells
- Reset Source Status
  - Status of the Last Reset
  - Either General Reset, Wakeup Reset, Software Reset, User Reset, Watchdog Reset

22.3 Block Diagram

Figure 22-1. Reset Controller Block Diagram



22.4 Functional Description

22.4.1 Reset Controller Overview

The Reset Controller is made up of an NRST Manager, a Startup Counter and a Reset State Manager. It runs at Slow Clock and generates the following reset signals:

- Processor Reset: resets the processor and the whole set of embedded peripherals.
- Backup Reset: resets all the peripherals powered by VDDBU.

These reset signals are asserted by the Reset Controller, either on external events or on software action. The Reset State Manager controls the generation of reset signals.

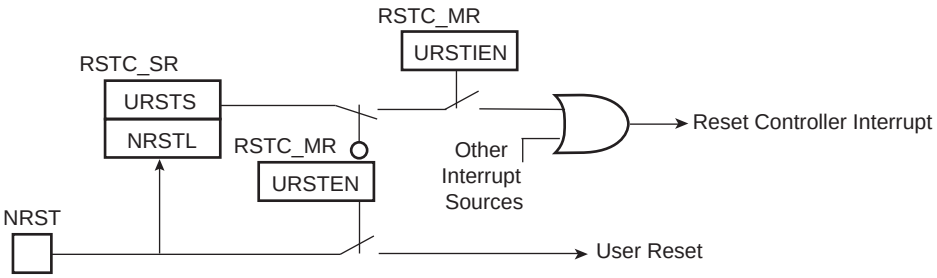
The startup counter waits for the complete crystal oscillator startup. The wait delay is given by the crystal oscillator startup time maximum value that can be found under “Crystal Oscillator Characteristics” in the “Electrical Characteristics” section.

The Reset Controller Mode Register (RSTC_MR), used to configure the reset controller, is powered with VDDBU, so that its configuration is saved as long as VDDBU is on.

22.4.2 NRST Manager

The NRST Manager samples the NRST input pin and drives this pin low when required by the Reset State Manager. Figure 22-2 shows the block diagram of the NRST Manager.

Figure 22-2. NRST Manager



22.4.2.1 NRST Signal or Interrupt

The NRST Manager samples the NRST pin at Slow Clock speed. When the line is detected low, a User Reset is reported to the Reset State Manager.

However, the NRST Manager can be programmed to not trigger a reset when an assertion of NRST occurs. Writing a zero to the URSTEN bit in the RSTC_MR disables the User Reset trigger.

The level of the pin NRST can be read at any time in the bit NRSTL (NRST level) in the Reset Controller Status Register (RSTC_SR). As soon as the pin NRST is asserted, the bit URSTS in the RSTC_SR is set. This bit clears only when RSTC_SR is read.

The reset controller can also be programmed to generate an interrupt instead of generating a reset. To do so, the bit URSTIEN in the RSTC_MR must be set.

22.4.3 Reset States

The Reset State Manager handles the different reset sources and generates the internal reset signals. It reports the reset status in the field RSTTYP of the RSTC_SR. The update of the field RSTTYP is performed when the processor reset is released.

22.4.3.1 General Reset

A general reset occurs when VDDBU and VDDCORE are powered on. The backup supply POR cell output rises and is filtered with a Startup Counter, which operates at Slow Clock. The purpose of this counter is to make sure the Slow Clock oscillator is stable before starting up the device. The length of startup time is hardcoded to comply with the Slow Clock Oscillator startup time.

After this time, the processor clock is released at Slow Clock and all the other signals remain valid for 2 cycles for proper processor and logic reset. Then, all the reset signals are released and the field RSTTYP in the RSTC_SR reports a General Reset.

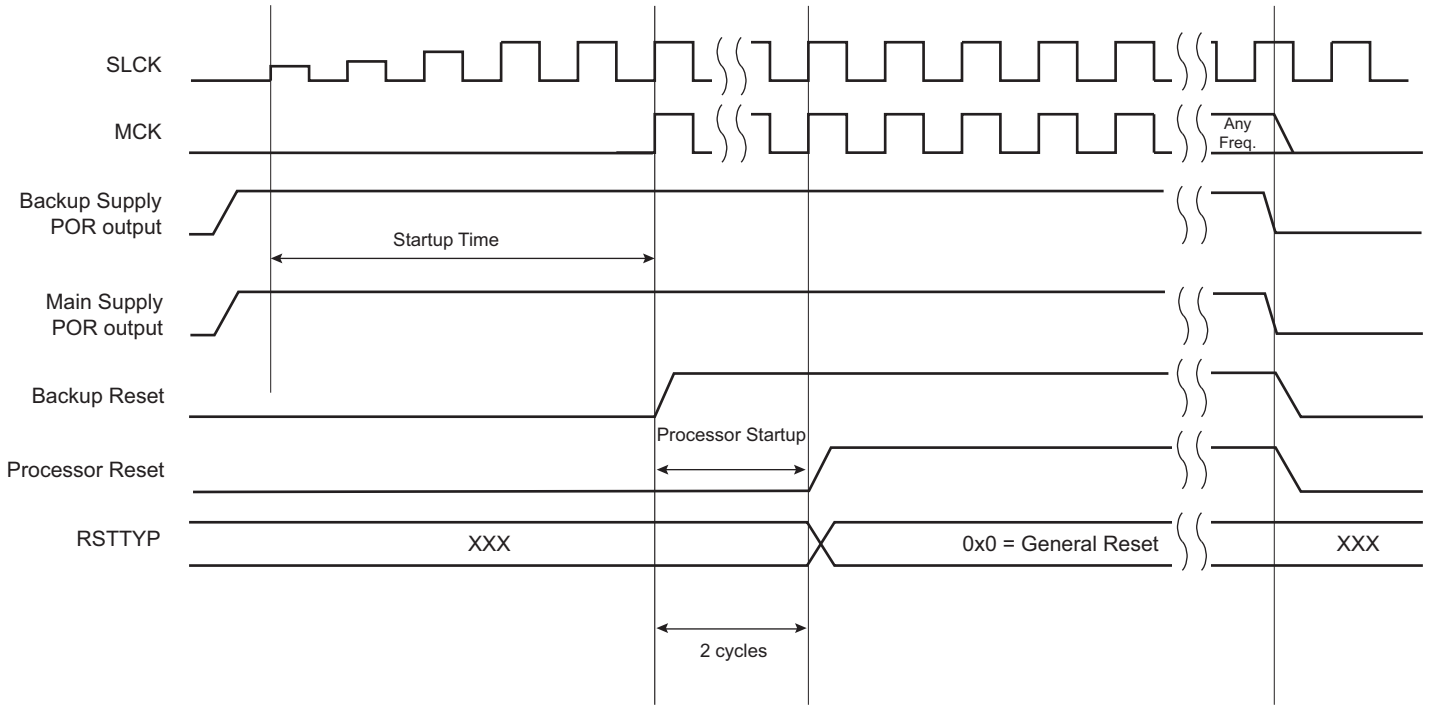
When VDDBU is detected low by the backup supply POR cell, all resets signals are immediately asserted, even if the main supply POR cell does not report a main supply shutdown.

VDDBU only activates the Backup Reset signal.

Backup Reset must be released so that any other reset can be generated by VDDCORE (main supply POR output).

Figure 22-3 shows how the General Reset affects the reset signals.

Figure 22-3. General Reset State



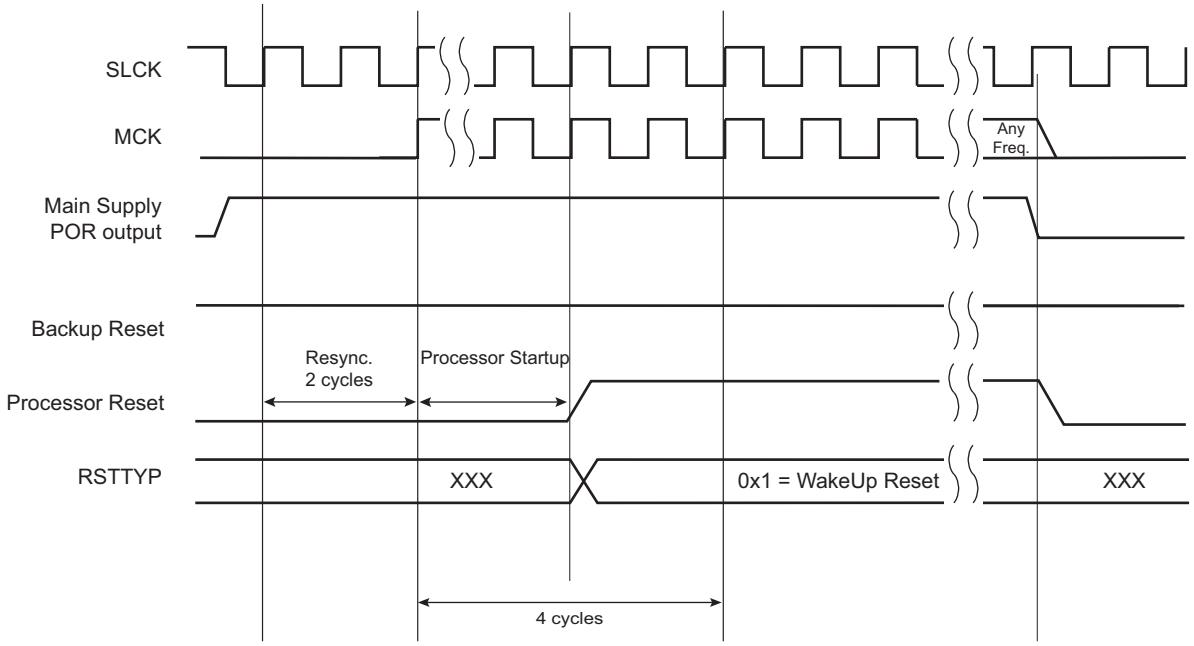
22.4.3.2 Wakeup Reset

The wakeup reset occurs when the main supply is down. When the main supply POR output is active, all the reset signals are asserted except Backup Reset. When the main supply powers up, the POR output is resynchronized on Slow Clock. The processor clock is then re-enabled during 2 Slow Clock cycles, depending on the requirements of the ARM processor.

At the end of this delay, the processor and other reset signals rise. The field RSTTYP in the RSTC_SR is updated to report a wakeup reset.

When the main supply is detected falling, the reset signals are immediately asserted. This transition is synchronous with the output of the main supply POR.

Figure 22-4. Wakeup Reset



22.4.3.3 User Reset

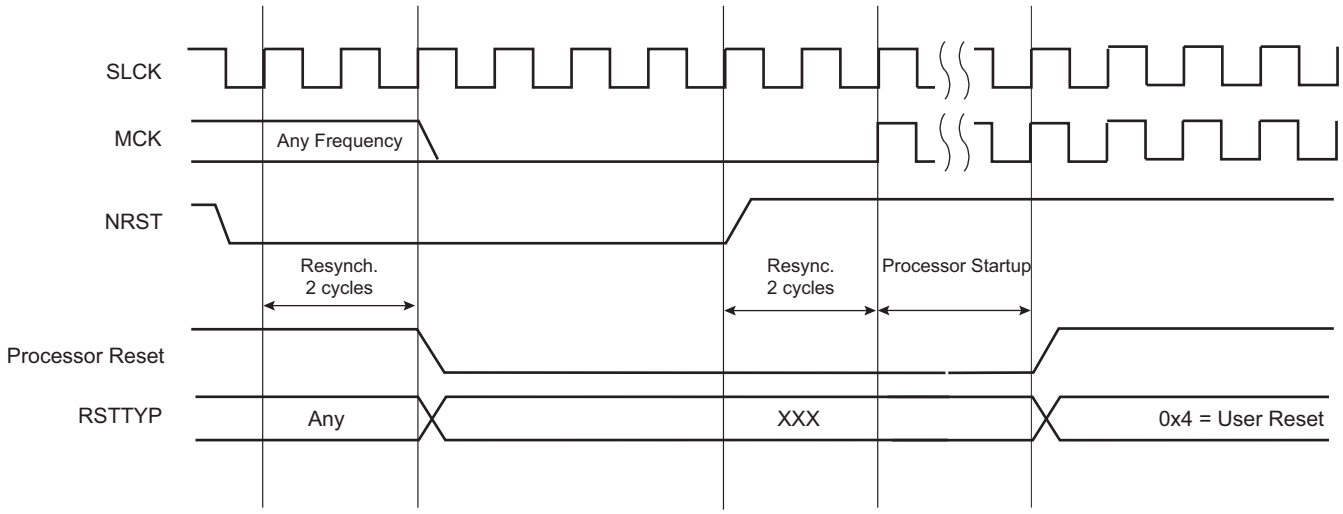
The User Reset is entered when a low level is detected on the NRST pin and the bit URSTEN in RSTC_MR is at 1. The NRST input signal is resynchronized with SLCK to ensure proper behavior of the system.

The Processor Reset and the Peripheral Reset are asserted.

The User Reset is left when NRST rises, after a two-cycle resynchronization time and a 2-cycle processor startup. The processor clock is re-enabled as soon as NRST is confirmed high.

When the processor reset signal is released, the RSTTYP field of the RSTC_SR is loaded with the value 0x4, indicating a User Reset.

Figure 22-5. User Reset State



22.4.3.4 Software Reset

The Reset Controller offers several commands used to assert the different reset signals. These commands are performed by writing the Control Register (RSTC_CR) with the following bits at 1:

- PROCRST: Writing PROCRST at 1 resets the processor, the watchdog timer and all the embedded peripherals, including the memory system, and, in particular, the Remap Command.

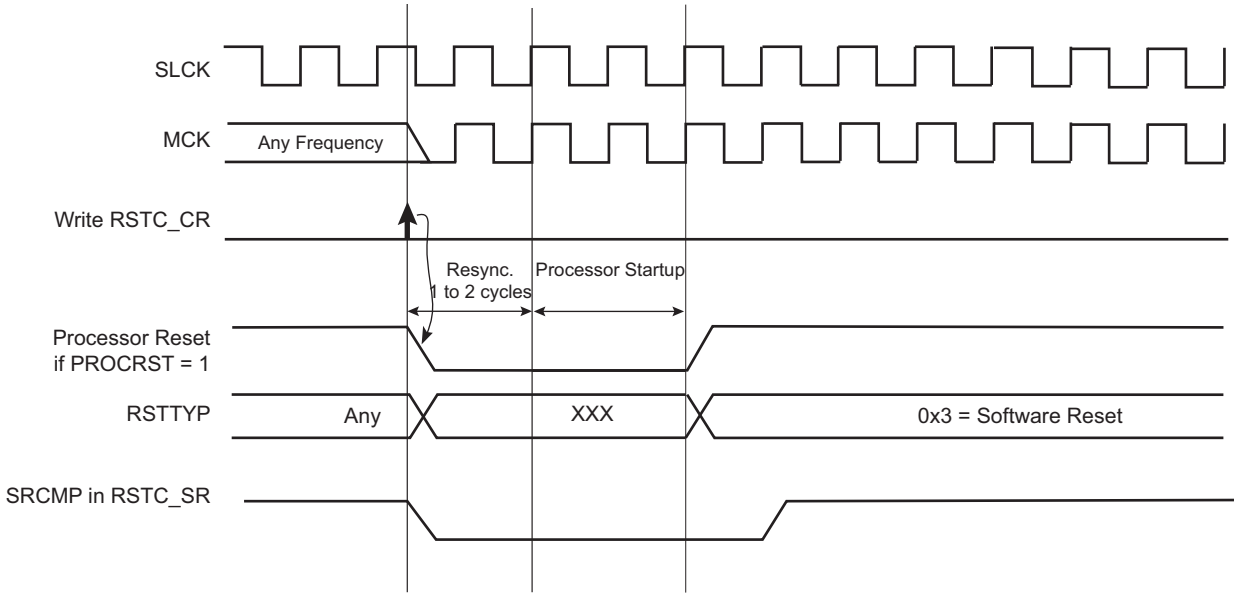
The software reset is entered if at least one of these bits is set by the software. All these commands can be performed independently or simultaneously. The software reset lasts two Slow Clock cycles.

The internal reset signals are asserted as soon as the register write is performed. This is detected on the Master Clock (MCK). They are released when the software reset is left, i.e., synchronously to SLCK.

If and only if the PROCRST bit is set, the reset controller reports the software status in the field RSTTYP of the RSTC_SR. Other software resets are not reported in RSTTYP.

As soon as a software operation is detected, the bit SRCMP (Software Reset Command in Progress) is set in the RSTC_SR. It is cleared as soon as the software reset is left. No other software reset can be performed while the SRCMP bit is set, and writing any value in RSTC_CR has no effect.

Figure 22-6. Software Reset



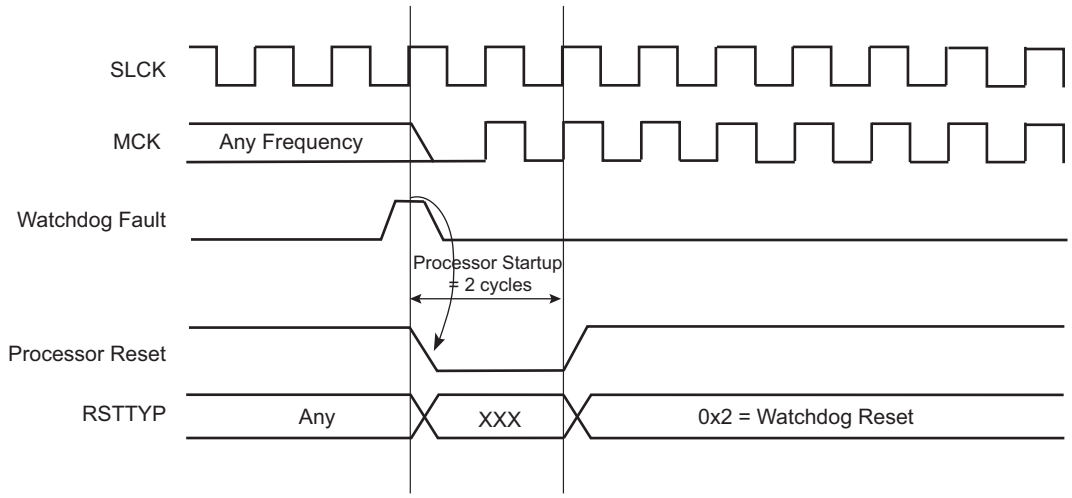
22.4.3.5 Watchdog Reset

The Watchdog Reset is entered when a watchdog fault occurs. This state lasts two Slow Clock cycles.

The Watchdog Timer is reset by the Processor Reset signal. As the watchdog fault always causes a processor reset if WDRSTEN is set, the Watchdog Timer is always reset after a Watchdog Reset and the Watchdog is enabled by default and with a period set to a maximum.

When the WDRSTEN in WDT_MR bit is reset, the watchdog fault has no impact on the reset controller.

Figure 22-7. Watchdog Reset



22.4.4 Reset State Priorities

The Reset State Manager manages the following priorities between the different reset sources, given in descending order:

- Backup Reset
- Wakeup Reset
- Watchdog Reset
- Software Reset
- User Reset

Particular cases are listed below:

- When in User Reset:
  - A watchdog event is impossible because the Watchdog Timer is being reset by the Processor Reset signal.
  - A Software Reset is impossible, since the processor reset is being activated.
- When in Software Reset:
  - A watchdog event has priority over the current state.
  - The NRST has no effect.
- When in Watchdog Reset:
  - The processor reset is active and so a Software Reset cannot be programmed.
  - A User Reset cannot be entered.

22.5 Reset Controller (RSTC) User Interface

Table 22-1. Register Mapping

Offset	Register	Name	Access	Reset	Backup Reset
0x00	Control Register	RSTC_CR	Write-only	–	–
0x04	Status Register	RSTC_SR	Read-only	0x0000_0100 (1)	0x0000_0000 (2)
0x08	Mode Register	RSTC_MR	Read/Write	–	0x0000_0000

Notes: 1. Only power supply VDDCORE rising
2. Both power supplies VDDCORE and VDDBU rising

22.5.1 Reset Controller Control Register

Name: RSTC_CR

Address: 0xF8048000

Access: Write-only

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–		–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	PROCRST

- **PROCRST: Processor Reset**

0: No effect

1: If KEY value = 0xA5, resets the processor and the peripherals

- **KEY: Write Access Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

22.5.2 Reset Controller Status Register

Name: RSTC_SR

Address: 0xF8048004

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	SRCMP	NRSTL

15	14	13	12	11	10	9	8
–	–	–	–	–	RSTTYP		

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	URSTS

- **URSTS: User Reset Status**

0: No high-to-low edge on NRST happened since the last read of RSTC_SR.

1: At least one high-to-low transition of NRST has been detected since the last read of RSTC_SR. Reading the RSTC_SR resets the URSTS bit and clears the interrupt.

- **RSTTYP: Reset Type**

This field reports the cause of the last processor reset. Reading this RSTC_SR does not reset this field.

Value	Name	Description
0	GENERAL_RST	Both VDDCORE and VDDBU rising
1	WKUP_RST	VDDCORE rising
2	WDT_RST	Watchdog fault occurred
3	SOFT_RST	Processor reset required by the software
4	USER_RST	NRST pin detected low

- **NRSTL: NRST Pin Level**

This bit records the level of the NRST pin sampled on each Master Clock (MCK) rising edge.

- **SRCMP: Software Reset Command in Progress**

0: No software command is being performed by the reset controller. The reset controller is ready for a software command.

1: A software reset command is being performed by the reset controller. The reset controller is busy.

22.5.3 Reset Controller Mode Register

Name: RSTC_MR

Address: 0xF8048008

Access: Read/Write

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	URSTIEN	–	–	–	URSTEN

- **URSTEN: User Reset Enable**

0: The detection of a low level on the pin NRST does not generate a User Reset.

1: The detection of a low level on the pin NRST triggers a User Reset.

- **URSTIEN: User Reset Interrupt Enable**

0: USRTS bit in RSTC_SR at 1 has no effect on the Reset Controller Interrupt.

1: USRTS bit in RSTC_SR at 1 asserts the Reset Controller Interrupt if URSTEN = 0.

- **KEY: Write Access Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

23. Shutdown Controller (SHDWC)

23.1 Description

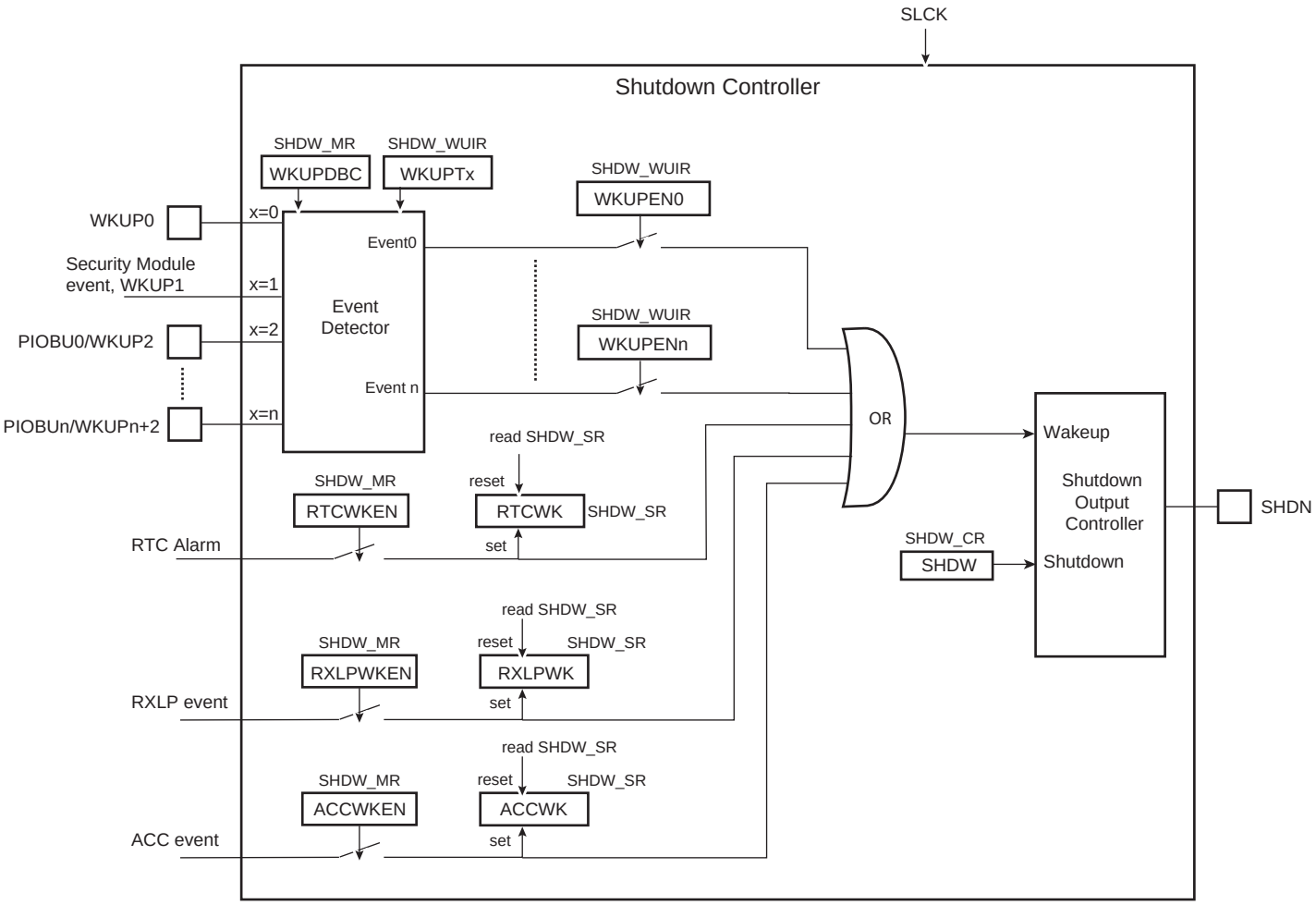
The Shutdown Controller (SHDWC) controls the power supplies VDDIO and VDDCORE and the wakeup detection on debounced input lines.

23.2 Embedded Characteristics

- Shutdown Logic
  - Software Assertion of the Shutdown Output Pin (SHDN)
  - Programmable deassertion from the PIOBU, WKUP Input Pins
- Wakeup Logic
  - Programmable Assertion from the PIOBU, WKUP Input Pins, and Internal Wakeup Event from RTC, RXLP, ACC, Security Module

23.3 Block Diagram

Figure 23-1. Shutdown Controller Block Diagram



23.4 I/O Lines Description

Table 23-1. I/O Lines Description

Name	Description	Type
WKUP0	Wakeup inputs	Input
PIOBU 0-7	Wakeup inputs, WKUP(2-9)	Input
SHDN	Shutdown output	Output

23.5 Product Dependencies

23.5.1 Power Management

The Shutdown Controller is continuously clocked by the Slow Clock (SLCK). The Power Management Controller has no effect on the behavior of the Shutdown Controller.

23.6 Functional Description

The Shutdown Controller manages the main power supply. To do so, it is supplied with VDDBU and manages wakeup input pins and one output pin, SHDN.

A typical application connects the pin SHDN to the shutdown input of the DC/DC Converter providing the main power supplies of the system, and especially VDDCORE and/or VDDIO. The wakeup inputs (WKUPn) connect to any push-buttons or signal that wake up the system.

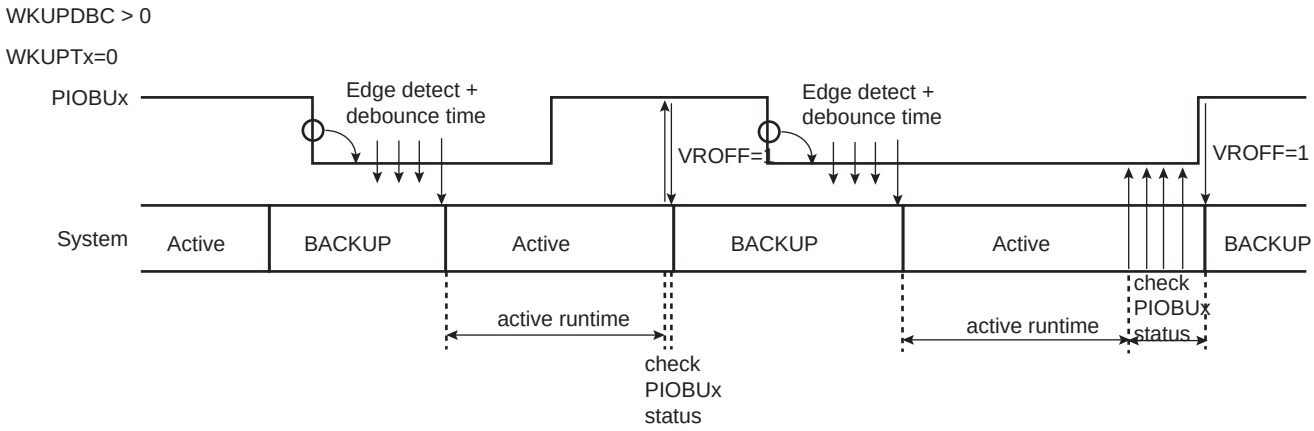
The software is able to control the pin SHDN by writing the Shutdown Control Register (SHDW_CR) with the bit SHDW at 1. The shutdown is taken into account only two slow clock cycles after the write of SHDW_CR. This register is password-protected and so the value written should contain the correct key for the command to be taken into account. As a result, the system should be powered down.

23.6.1 Wakeup Inputs

Any level change on a PIOBUx, WKUP pin, or Security Module event, can trigger a wakeup. Wakeup is configured in the Shutdown Mode Register (SHDW_MR) and Shutdown Wakeup Inputs Register (SHDW_WUIR). The transition detector can be programmed to detect either a positive or negative transition on any PIOBUx, WKUP pin. The detection can also be disabled. Programming is performed by enabling the Wakeup Input (WKUPENx bit) and defining the Wakeup Input Type (WKUPTx bit) in the SHDW_WUIR.

Moreover, a debouncing circuit can be programmed for PIOBUx, WKUP. The debouncing circuit filters pulses on PIOBUx, WKUP shorter than the programmed value in the WKUPDBC field in SHDW_MR. If the programmed level change is detected on a pin, a counter starts. When the counter reaches the value programmed in the corresponding field WKUPDBC, the SHDN pin is released. If a new input change is detected before the counter reaches the corresponding value, the counter is stopped and cleared. One counter is shared among all PIOBUx, WKUP inputs and all programmed level detection is merged into this counter. The WKUPISx bit of the Status Register (SHDW_SR) reports the detection of the programmed events on PIOBUx, WKUP with a reset after the read of SHDW_SR.

Figure 23-2. Entering and Exiting Backup Mode with a PIOBUx, WKUP Pin



The Shutdown Controller can be programmed so as to activate the wakeup using the RTC alarm, RXLP event, ACC comparison event, security module event (detection of the rising edge event is synchronized with SLCK). This is done by writing the SHDW_MR using the RTCWKEN bit, RXLPWKEN bit and ACCWKEN bit. When enabled, the detection of RTC alarm, RXLP event, ACC comparison event, security module event is reported in the RTCWK bit, RXLPWK and ACCWK bits of SHDW_SR. They are cleared after reading SHDW_SR. When using the RTC alarm to wake up the system, the user must ensure that RTC alarm, RXLPWK and ACCWK status flags are cleared before shutting down the system. Otherwise, no rising edge of the status flags may be detected and the wakeup will fail.

23.7 Shutdown Controller (SHDWC) User Interface

Table 23-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Shutdown Control Register	SHDW_CR	Write-only	–
0x04	Shutdown Mode Register	SHDW_MR	Read/Write	0x0000_0000
0x08	Shutdown Status Register	SHDW_SR	Read-only	0x0000_0000
0x0C	Shutdown Wakeup Inputs Register	SHDW_WUIR	Read/Write	0x0000_0000

23.7.1 Shutdown Control Register

Name: SHDW_CR

Address: 0xF8048010

Access: Write-only

31	30	29	28	27	26	25	24
KEY							

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SHDW

- **SHDW: Shutdown Command**

0: No effect.

1: If KEY value is correct, asserts the SHDN pin.

- **KEY: Password**

Value	Name	Description
0xA5	PASSWD	Writing any other value in this field aborts the write operation.

23.7.2 Shutdown Mode Register

Name: SHDW_MR

Address: 0xF8048014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	WKUPDBC		

23	22	21	20	19	18	17	16
–	–	–	–	RXLPWKEN	ACCWKEN	RTCWKEN	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **RTCWKEN: Real-time Clock Wakeup Enable**

0: The RTC Alarm signal has no effect on the Shutdown Controller.

1: The RTC Alarm signal forces the deassertion of the SHDN pin.

- **ACCWKEN: Analog Comparator Controller Wakeup Enable**

0: The Analog comparator alarm signal has no effect on the Shutdown Controller.

1: The Analog comparator alarm signal forces the deassertion of the SHDN pin.

- **RXLPWKEN: Debug Unit Wakeup Enable**

0: The Backup RX UART Comparison event has no effect on the Shutdown Controller.

1: The Backup RX UART Comparison event forces the deassertion of the SHDN pin.

- **WKUPDBC: Wakeup Inputs Debouncer Period**

Value	Name	Description
0	IMMEDIATE	Immediate, no debouncing, detected active at least on one Slow Clock edge
1	3_SLCK	PIOBUx shall be in its active state for at least 3 SLCK periods
2	32_SLCK	PIOBUx shall be in its active state for at least 32 SLCK periods
3	512_SLCK	PIOBUx shall be in its active state for at least 512 SLCK periods
4	4096_SLCK	PIOBUx shall be in its active state for at least 4,096 SLCK periods
5	32768_SLCK	PIOBUx shall be in its active state for at least 32,768 SLCK periods

23.7.3 Shutdown Status Register

Name: SHDW_SR

Address: 0xF8048018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	WKUPIS9	WKUPIS8
23	22	21	20	19	18	17	16
WKUPIS7	WKUPIS6	WKUPIS5	WKUPIS4	WKUPIS3	WKUPIS2	WKUPIS1	WKUPIS0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXLPWK	ACCWK	RTCWK	–	–	–	–	WKUPS

- **WKUPS: PIOBU, WKUP Wakeup Status**

0 (NO): No wakeup due to the assertion of the PIOBU, WKUP pins has occurred since the last read of SHDW_SR.

1 (PRESENT): At least one wakeup due to the assertion of the PIOBU, WKUP pins has occurred since the last read of SHDW_SR.

Note: WKUPIS1 reports the status of the Security Module event.

- **ACCWK: Analog Comparator Controller Wakeup**

0: No wakeup alarm from the ACC occurred since the last read of SHDW_SR.

1: At least one wakeup alarm from the ACC occurred since the last read of SHDW_SR.

- **RXLPWK: Debug Unit Wakeup**

0: No wakeup alarm from the Backup RX UART Comparison unit (RXLP) occurred since the last read of SHDW_SR.

1: At least one wakeup alarm from the Backup RX UART Comparison unit (RXLP) occurred since the last read of SHDW_SR.

- **WKUPIS0–WKUPIS9: Wakeup 0 to 9 Input Status**

0 (DISABLE): The corresponding wakeup input is disabled, or was inactive at the time the debouncer triggered a wakeup event.

1 (ENABLE): The corresponding wakeup input was active at the time the debouncer triggered a wakeup event.

23.7.4 Shutdown Wakeup Inputs Register

Name: SHDW_WUIR

Address: 0xF804801C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	WKUPT9	WKUPT8
23	22	21	20	19	18	17	16
WKUPT7	WKUPT6	WKUPT5	WKUPT4	WKUPT3	WKUPT2	WKUPT1	WKUPT0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	WKUPEN9	WKUPEN8
7	6	5	4	3	2	1	0
WKUPEN7	WKUPEN6	WKUPEN5	WKUPEN4	WKUPEN3	WKUPEN2	WKUPEN1	WKUPEN0

- **WKUPEN0–WKUPEN9: Wakeup 0 to 9 Input Enable**

0 (DISABLE): The corresponding wakeup input has no wakeup effect.

1 (ENABLE): The corresponding wakeup input forces the wakeup of the core power supply.

- **WKUPT0–WKUPT9: Wakeup 0 to 9 Input Type**

0 (LOW): A falling edge followed by a low level, for a period defined by WKUPDBC, on the corresponding wakeup input forces the wakeup of the core power supply.

1 (HIGH): A rising edge followed by a high level, for a period defined by WKUPDBC, on the corresponding wakeup input forces the wakeup of the core power supply.

24. Periodic Interval Timer (PIT)

24.1 Description

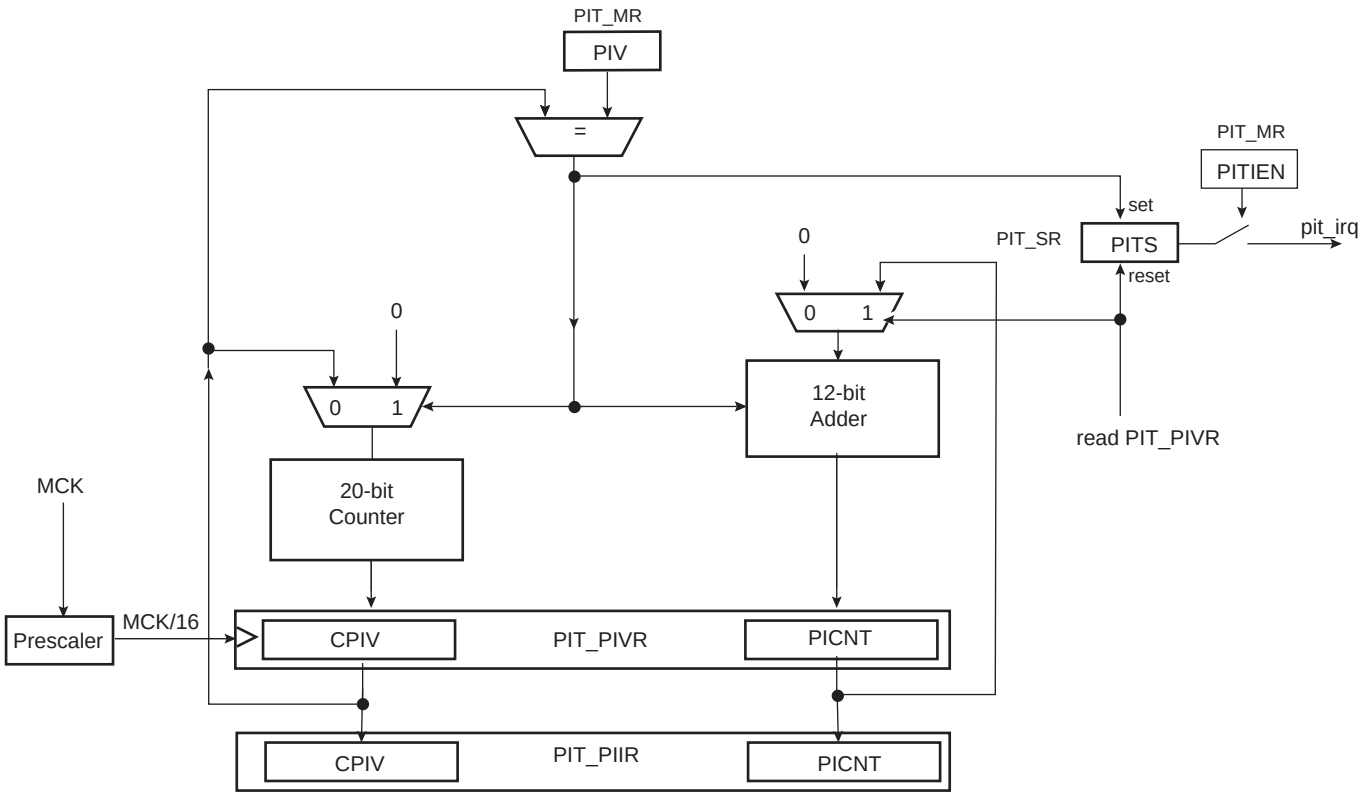
The Periodic Interval Timer (PIT) provides the operating system's scheduler interrupt. It is designed to offer maximum accuracy and efficient management, even for systems with long response time.

24.2 Embedded Characteristics

- 20-bit Programmable Counter plus 12-bit Interval Counter
- Reset-on-read Feature
- Both Counters Work on Master Clock/16

24.3 Block Diagram

Figure 24-1. Periodic Interval Timer



24.4 Functional Description

The Periodic Interval Timer aims at providing periodic interrupts for use by operating systems.

The PIT provides a programmable overflow counter and a reset-on-read feature. It is built around two counters: a 20-bit CPIV counter and a 12-bit PICNT counter. Both counters work at Master Clock /16.

The first 20-bit CPIV counter increments from 0 up to a programmable overflow value set in the field PIV of the Mode Register (PIT_MR). When the counter CPIV reaches this value, it resets to 0 and increments the Periodic Interval Counter, PICNT. The status bit PITS in the Status Register (PIT_SR) rises and triggers an interrupt, provided the interrupt is enabled (PITIEN in PIT_MR).

Writing a new PIV value in PIT_MR does not reset/restart the counters.

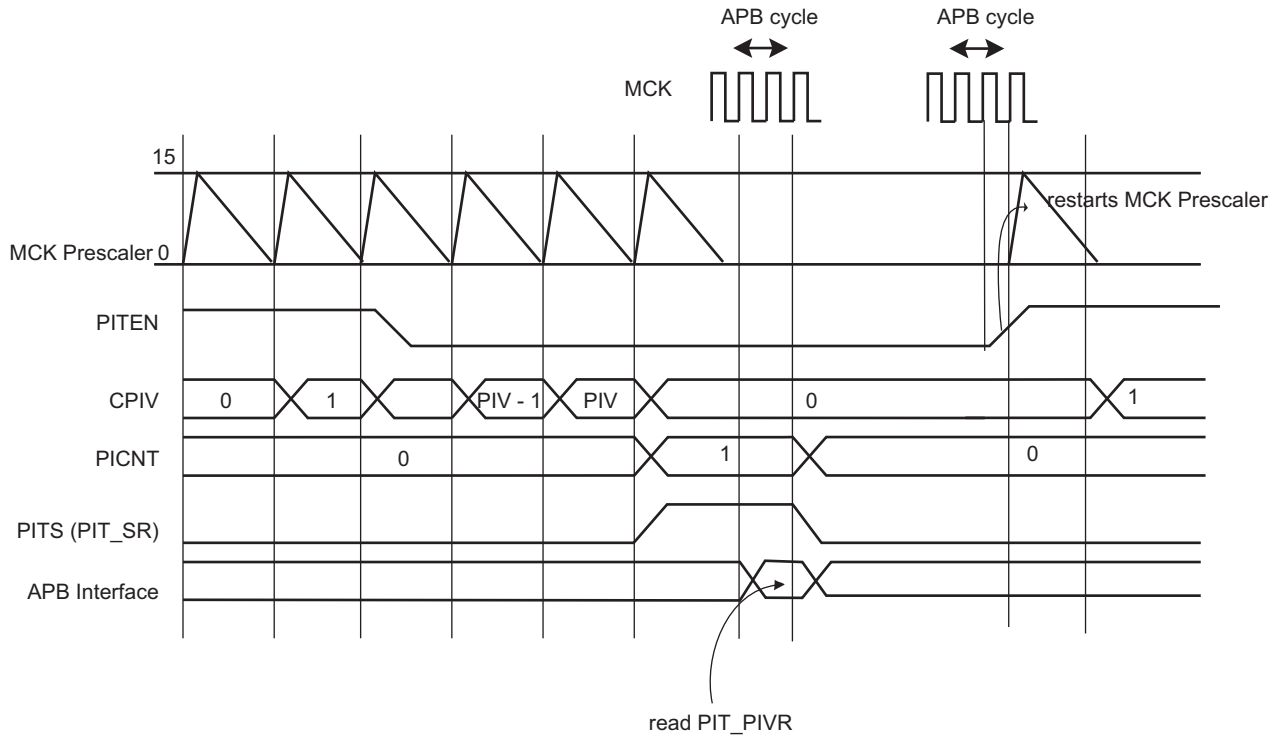
When CPIV and PICNT values are obtained by reading the Periodic Interval Value Register (PIT_PIVR), the overflow counter (PICNT) is reset and the PITS bit is cleared, thus acknowledging the interrupt. The value of PICNT gives the number of periodic intervals elapsed since the last read of PIT_PIVR.

When CPIV and PICNT values are obtained by reading the Periodic Interval Image Register (PIT_PIIR), there is no effect on the counters CPIV and PICNT, nor on the bit PITS. For example, a profiler can read PIT_PIIR without clearing any pending interrupt, whereas a timer interrupt clears the interrupt by reading PIT_PIVR.

The PIT may be enabled/disabled using the PITEN bit in the PIT_MR register (disabled on reset). The PITEN bit only becomes effective when the CPIV value is 0. Figure 24-2 illustrates the PIT counting. After the PIT Enable bit is reset (PITEN = 0), the CPIV goes on counting until the PIV value is reached, and is then reset. PIT restarts counting, only if the PITEN is set again.

The PIT is stopped when the core enters debug state.

Figure 24-2. Enabling/Disabling PIT with PITEN



24.5 Periodic Interval Timer (PIT) User Interface

Table 24-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Mode Register	PIT_MR	Read/Write	0x000F_FFFF
0x04	Status Register	PIT_SR	Read-only	0x0000_0000
0x08	Periodic Interval Value Register	PIT_PIVR	Read-only	0x0000_0000
0x0C	Periodic Interval Image Register	PIT_PIIR	Read-only	0x0000_0000

24.5.1 Periodic Interval Timer Mode Register

Name: PIT_MR

Address: 0xF8048030

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	PITIEN	PITEN

23	22	21	20	19	18	17	16
–	–	–	–	PIV			

15	14	13	12	11	10	9	8
PIV							

7	6	5	4	3	2	1	0
PIV							

- **PIV: Periodic Interval Value**

Defines the value compared with the primary 20-bit counter of the Periodic Interval Timer (CPIV). The period is equal to (PIV + 1).

- **PITEN: Period Interval Timer Enabled**

0: The Periodic Interval Timer is disabled when the PIV value is reached.

1: The Periodic Interval Timer is enabled.

- **PITIEN: Periodic Interval Timer Interrupt Enable**

0: The bit PITS in PIT_SR has no effect on interrupt.

1: The bit PITS in PIT_SR asserts interrupt.

24.5.2 Periodic Interval Timer Status Register

Name: PIT_SR

Address: 0xF8048034

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	PITS

- **PITS: Periodic Interval Timer Status**

0: The Periodic Interval timer has not reached PIV since the last read of PIT_PIVR.

1: The Periodic Interval timer has reached PIV since the last read of PIT_PIVR.

24.5.3 Periodic Interval Timer Value Register

Name: PIT_PIVR

Address: 0xF8048038

Access: Read-only

31	30	29	28	27	26	25	24
PICNT							

23	22	21	20	19	18	17	16
PICNT				CPIV			

15	14	13	12	11	10	9	8
CPIV							

7	6	5	4	3	2	1	0
CPIV							

Reading this register clears PITS in PIT_SR.

- **CPIV: Current Periodic Interval Value**

Returns the current value of the periodic interval timer.

- **PICNT: Periodic Interval Counter**

Returns the number of occurrences of periodic intervals since the last read of PIT_PIVR.

24.5.4 Periodic Interval Timer Image Register

Name: PIT_PIIR

Address: 0xF804803C

Access: Read-only

31	30	29	28	27	26	25	24
PICNT							

23	22	21	20	19	18	17	16
PICNT				CPIV			

15	14	13	12	11	10	9	8
CPIV							

7	6	5	4	3	2	1	0
CPIV							

- **CPIV: Current Periodic Interval Value**

Returns the current value of the periodic interval timer.

- **PICNT: Periodic Interval Counter**

Returns the number of occurrences of periodic intervals since the last read of PIT_PIVR.

25. Real-time Clock (RTC)

25.1 Description

The Real-time Clock (RTC) peripheral is designed for very low power consumption. For optimal functionality, the RTC requires an accurate external 32.768 kHz clock, which can be provided by a crystal oscillator.

It combines a complete time-of-day clock with alarm and a Gregorian or Persian calendar, complemented by a programmable periodic interrupt. The alarm and calendar registers are accessed by a 32-bit data bus.

The RTC can also be configured for the UTC time format.

The time and calendar values are coded in binary-coded decimal (BCD) format. The time format can be 24-hour mode or 12-hour mode with an AM/PM indicator.

Updating time and calendar fields and configuring the alarm fields are performed by a parallel capture on the 32-bit data bus. An entry control is performed to avoid loading registers with incompatible BCD format data or with an incompatible date according to the current month/year/century.

A clock divider calibration circuitry can be used to compensate for crystal oscillator frequency variations.

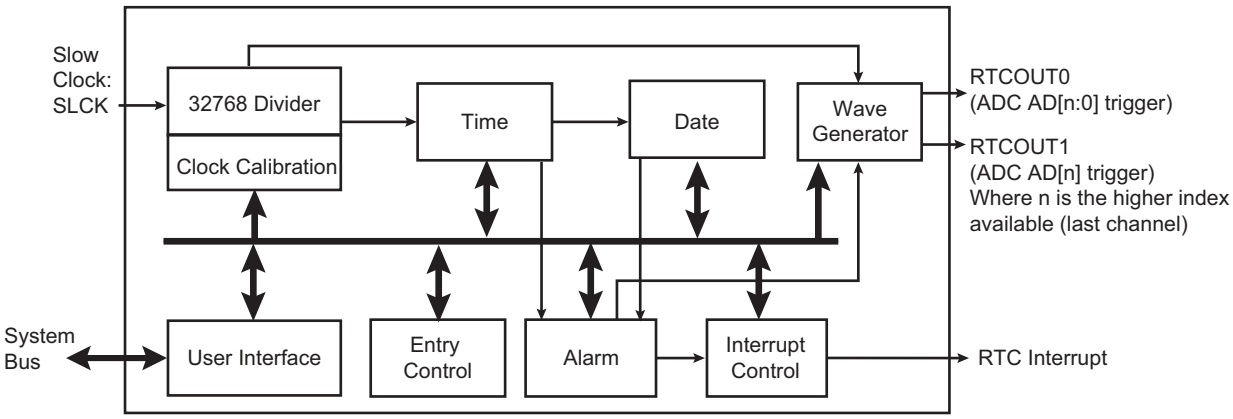
Timestamping capability reports the first and last occurrences of tamper events.

25.2 Embedded Characteristics

- Full Asynchronous Design for Ultra Low Power Consumption
- Gregorian, UTC and Persian Modes Supported
- Programmable Periodic Interrupt
- Safety/security Features:
  - Valid Time and Date Programming Check
  - On-The-Fly Time and Date Validity Check
- Counters Calibration Circuitry to Compensate for Crystal Oscillator Variations
- Waveform Generation for Trigger Event
- Tamper Timestamping Registers
- Register Write Protection

25.3 Block Diagram

Figure 25-1. Real-time Clock Block Diagram



25.4 Product Dependencies

25.4.1 Power Management

The Real-time Clock is continuously clocked at 32.768 kHz. The Power Management Controller has no effect on RTC behavior.

25.4.2 Interrupt

Within the System Controller, the RTC interrupt is OR-wired with all the other module interrupts.

Only one System Controller interrupt line is connected on one of the internal sources of the interrupt controller.

RTC interrupt requires the interrupt controller to be programmed first.

When a System Controller interrupt occurs, the service routine must first determine the cause of the interrupt. This is done by reading each status register of the System Controller peripherals successively.

25.5 Functional Description

The RTC provides a full binary-coded decimal (BCD) clock that includes century (19/20), year (with leap years), month, date, day, hours, minutes and seconds reported in RTC Time Register (RTC_TIMR) and RTC Time Register (UTC_MODE) (RTC_CALR).

The RTC can operate in UTC mode, giving the number of seconds elapsed since a reference time defined by the user (the UTC standard—ISO 8601—reference time is the 30th of June 1972). In this mode, the timefield is 32-bit wide and coded in hexadecimal format.

The valid year range is up to 2099 in Gregorian mode (or 1300 to 1499 in Persian mode).

The RTC can operate in 24-hour mode or in 12-hour mode with an AM/PM indicator.

Corrections for leap years are included (all years divisible by 4 being leap years except 1900). This is correct up to the year 2099.

The RTC can generate events to trigger ADC measurements.

25.5.1 Reference Clock

The reference clock is the Slow Clock (SLCK). It can be driven internally or by an external 32.768 kHz crystal.

During low power modes of the processor, the oscillator runs and power consumption is critical. The crystal selection has to take into account the current consumption for power saving and the frequency drift due to temperature effect on the circuit for time accuracy.

25.5.2 Timing

In Gregorian and Persian modes, the RTC is updated in real time at one-second intervals in Normal mode for the counters of seconds, at one-minute intervals for the counter of minutes and so on.

In UTC mode, the RTC is updated in real time at one-second intervals (32-bit UTC counter default configuration).

Due to the asynchronous operation of the RTC with respect to the rest of the chip, to be certain that the value read in the RTC registers (century, year, month, date, day, hours, minutes, seconds) are valid and stable, it is necessary to read these registers twice. If the data is the same both times, then it is valid. Therefore, a minimum of two and a maximum of three accesses are required.

25.5.3 Alarm

In Gregorian and Persian modes, the RTC has five programmable fields: month, date, hours, minutes and seconds.

Each of these fields can be enabled or disabled to match the alarm condition:

- If all the fields are enabled, an alarm flag is generated (the corresponding flag is asserted and an interrupt generated if enabled) at a given month, date, hour/minute/second.
- If only the “seconds” field is enabled, then an alarm is generated every minute.

Depending on the combination of fields enabled, a large number of possibilities are available to the user ranging from minutes to 365/366 days.

Hour, minute and second matching alarm (SECEN, MINEN, HOUREN) can be enabled independently of SEC, MIN, HOUR fields.

Note: To change one of the SEC, MIN, HOUR, DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR or RTC_CALALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN, DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR, DATE, MONTH). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREn, DATEEN, MTHEN fields.

In UTC mode, RTC_TIMALR must be configured to set the UTC alarm value and bit 0 in RTC_CALALR must be used to enable or disable the UTC alarm. If the UTC alarm is enabled, the alarm is generated once the UTC time matches the programmed UTC_TIME alarm field.

To change the UTC_TIME alarm field, proceed as follows:

1. Disable the UTC alarm by clearing the UTCEN bit in RTC_CALALR if it is not already cleared.
2. Change the UTC_TIME alarm value in RTC_TIMALR.
3. Re-enable the UTC alarm by setting the UTCEN bit in RTC_CALALR.

25.5.4 Error Checking when Programming

Verification on user interface data is performed when accessing the century, year, month, date, day, hours, minutes, seconds and alarms. A check is performed on illegal BCD entries such as illegal date of the month with regard to the year and century configured.

If one of the time fields is not correct, the data is not loaded into the register/counter and a flag is set in the validity register. The user can not reset this flag. It is reset as soon as an acceptable value is programmed. This avoids any further side effects in the hardware. The same procedure is followed for the alarm.

The following checks are performed:

1. Century (check if it is in range 19–20 or 13–14 in Persian mode)
2. Year (BCD entry check)
3. Date (check range 01–31)
4. Month (check if it is in BCD range 01–12, check validity regarding “date”)
5. Day (check range 1–7)
6. Hour (BCD checks: in 24-hour mode, check range 00–23 and check that AM/PM flag is not set if RTC is set in 24-hour mode; in 12-hour mode check range 01–12)
7. Minute (check BCD and range 00–59)
8. Second (check BCD and range 00–59)

Note: If the 12-hour mode is selected by means of the RTC Mode Register (RTC_MR), a 12-hour value can be programmed and the returned value on RTC_TIMR will be the corresponding 24-hour value. The entry control checks the value of the AM/PM indicator (bit 22 of RTC_TIMR) to determine the range to be checked.

Note: In UTC mode, no check is performed on the entries. The RTC does not report any failure.

25.5.5 RTC Internal Free Running Counter Error Checking

To improve the reliability and security of the RTC, a permanent check is performed on the internal free running counters to report non-BCD or invalid date/time values.

An error is reported by TDERR bit in the status register (RTC_SR) if an incorrect value has been detected. The flag can be cleared by setting the TDERRCLR bit in the Status Clear Command Register (RTC_SCCR).

Anyway the TDERR error flag will be set again if the source of the error has not been cleared before clearing the TDERR flag. The clearing of the source of such error can be done by reprogramming a correct value on RTC_CALR and/or RTC_TIMR.

The RTC internal free running counters may automatically clear the source of TDERR due to their roll-over (i.e., every 10 seconds for SECONDS[3:0] field in RTC_TIMR). In this case the TDERR is held high until a clear command is asserted by TDERRCLR bit in RTC_SCCR.

25.5.6 Updating Time/Calendar

25.5.6.1 Gregorian and Persian Modes

The update of the time/calendar must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER register.

Once the second event occurs, the user must stop the RTC by setting the corresponding field in the Control Register (RTC_CR). Bit UPDTIM must be set to update time fields (hour, minute, second) and bit UPDCAL must be set to update calendar fields (century, year, month, date, day).

The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register, the Calendar Register, or both.

Once the update is finished, the user must write UPDTIM and/or UPDCAL to 0 in the RTC_CR.

The timing sequence of the time/calendar update is described in Figure 25-2 ”Time/Calendar Update Timing Diagram”.

When entering the Programming mode of the calendar fields, the time fields remain enabled. When entering the Programming mode of the time fields, both the time and the calendar fields are stopped. This is due to the location of the calendar logical circuity (downstream for low-power considerations). It is highly recommended to prepare all the fields to be updated before entering Programming mode. In successive update operations, the user must wait for at least one second after resetting the UPDTIM/UPDCAL bit in the RTC_CR before setting these bits again. This is done by waiting for the SEC flag in the RTC_SR before setting the UPDTIM/UPDCAL bit. After resetting UPDTIM/UPDCAL, the SEC flag must also be cleared.

Figure 25-2. Time/Calendar Update Timing Diagram

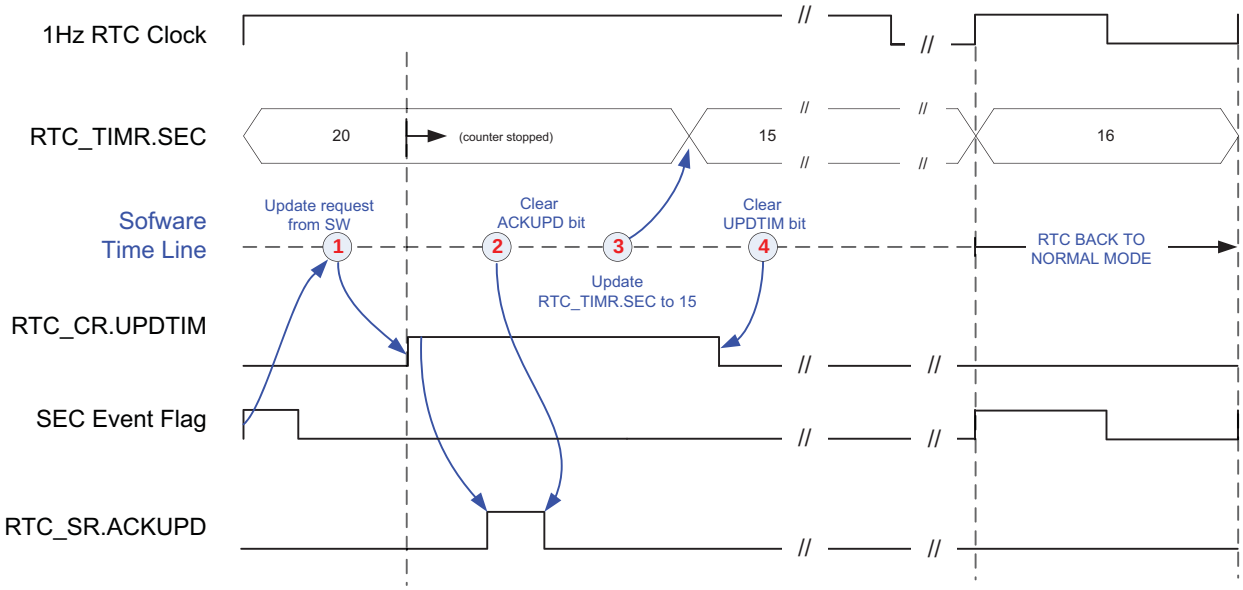
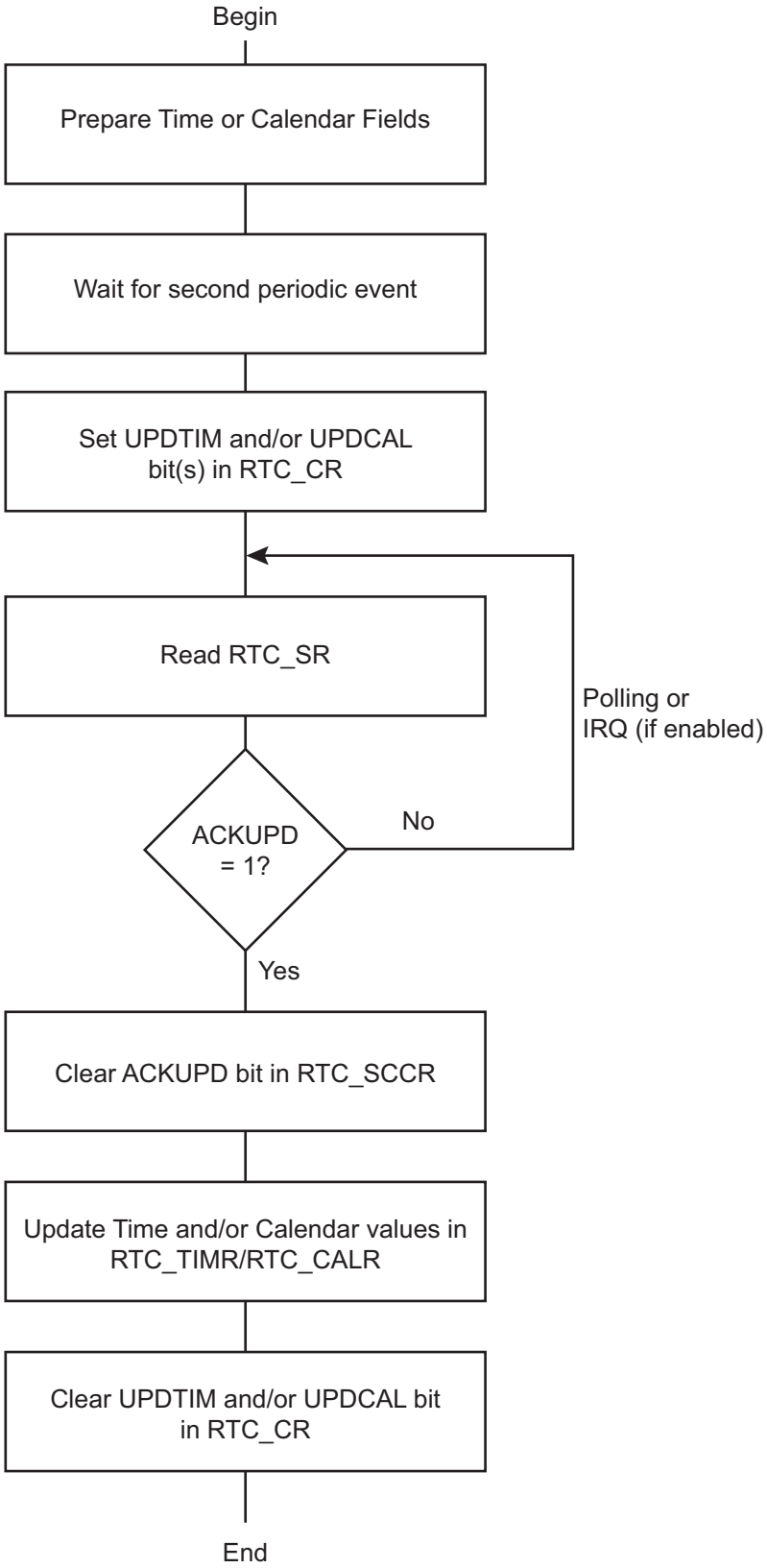


Figure 25-3. Gregorian and Persian Modes Update Sequence



25.5.6.2 UTC Mode

The update of the UTC time field must be synchronized on a second periodic event by either polling the RTC_SR.SEC status bit or by enabling the SECEN interrupt in the RTC_IER.

Once the second event occurs, the user must stop the RTC by setting the UPDTIM field in the Control Register (RTC_CR).

The ACKUPD bit must then be read to 1 by either polling the RTC_SR or by enabling the ACKUPD interrupt in the RTC_IER. Once ACKUPD is read to 1, it is mandatory to clear this flag by writing the corresponding bit in the RTC_SCCR, after which the user can write to the Time Register.

Once the update is finished, the user must write UPDTIM to 0 in the RTC_CR.

The timing sequence of the UTC time update is described in Figure 25-4 ”UTC Time Update Timing Diagram”.

In successive update operations, the user must wait for at least one second after resetting the UPDTIM bit in the RTC_CR before setting this bit again. This is done by waiting for the SEC flag in the RTC_SR before setting UPDTIM bit. After resetting UPDTIM, the SEC flag must also be cleared.

Figure 25-4. UTC Time Update Timing Diagram

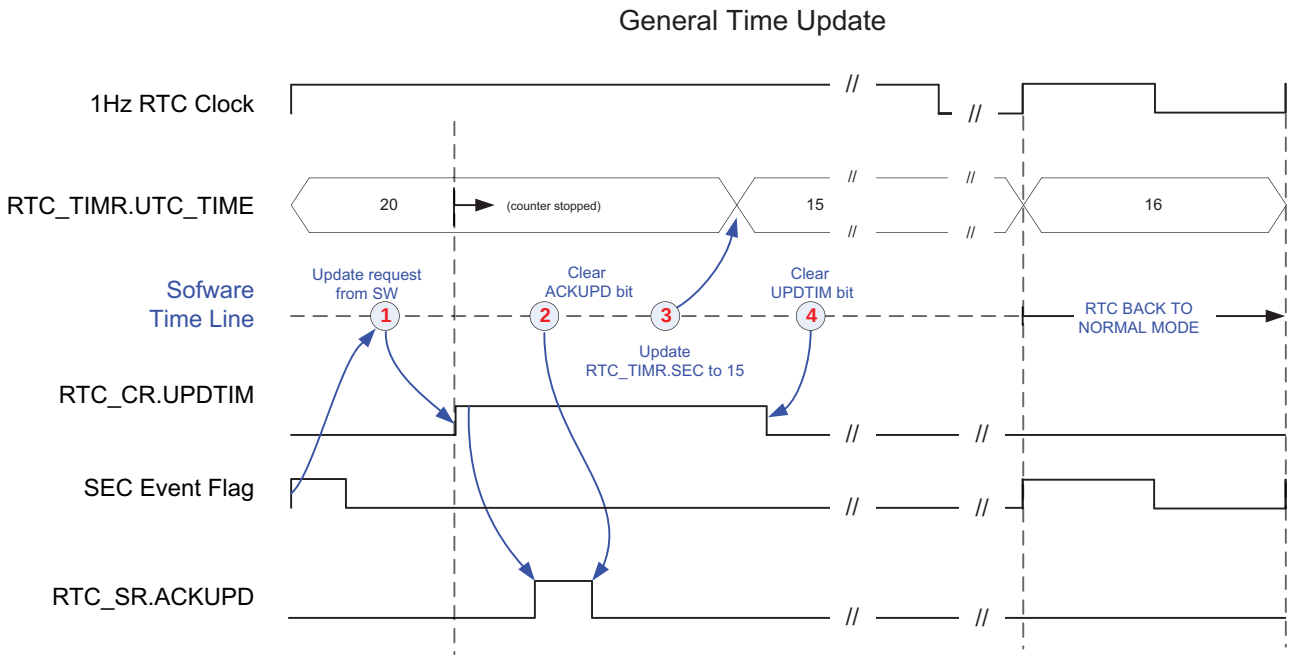
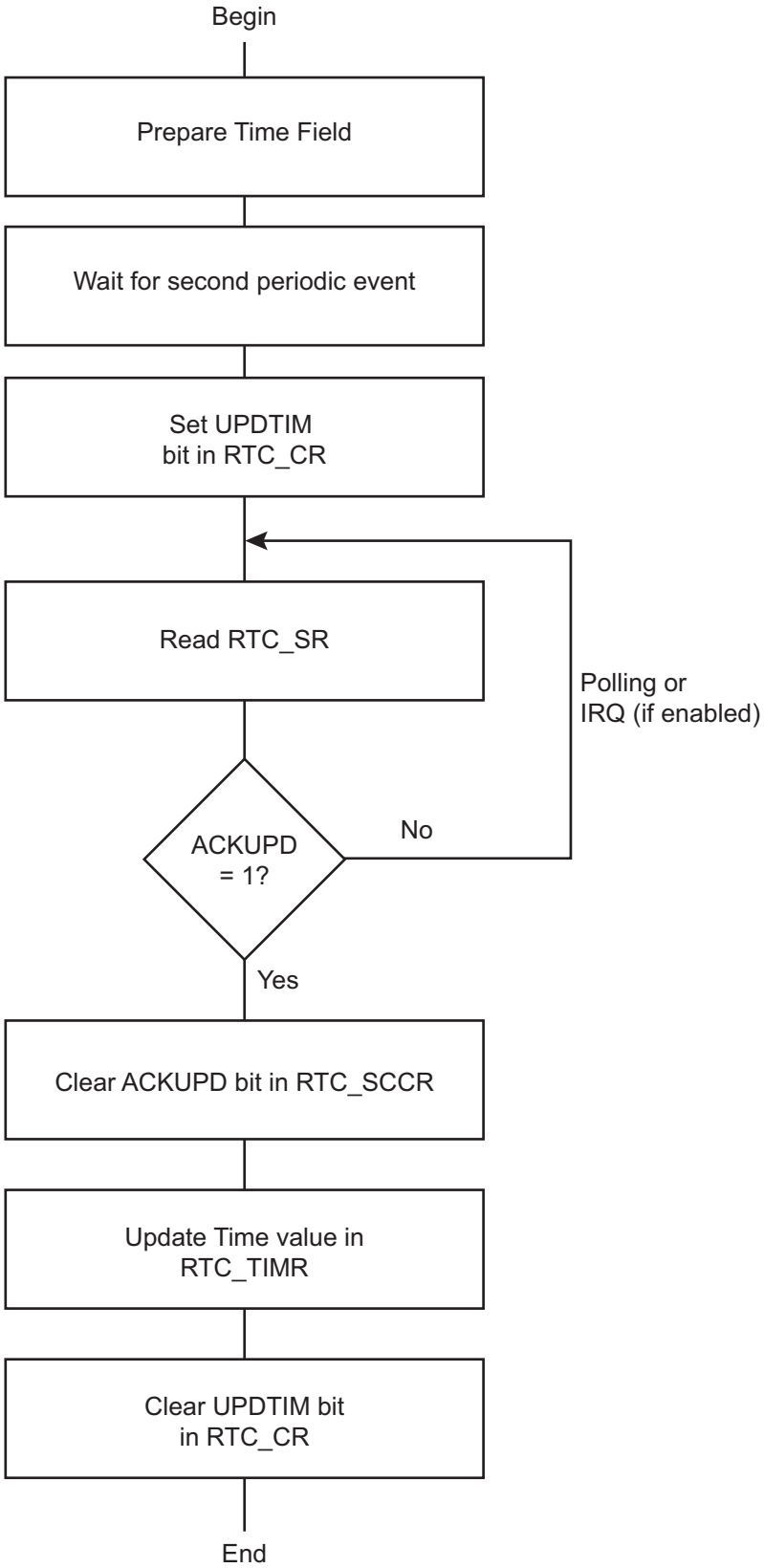


Figure 25-5. UTC Mode Update Sequence



25.5.7 RTC Accurate Clock Calibration

The crystal oscillator that drives the RTC may not be as accurate as expected mainly due to temperature variation. The RTC is equipped with circuitry able to correct slow clock crystal drift.

To compensate for possible temperature variations over time, this accurate clock calibration circuitry can be programmed on-the-fly and also programmed during application manufacturing, in order to correct the crystal frequency accuracy at room temperature (20–25°C). The typical clock drift range at room temperature is ±20 ppm.

In the device operating temperature range, the 32.768 kHz crystal oscillator clock inaccuracy can be up to -200 ppm.

The RTC clock calibration circuitry allows positive or negative correction in a range of 1.5 ppm to 1950 ppm.

The calibration circuitry is fully digital. Thus, the configured correction is independent of temperature, voltage, process, etc., and no additional measurement is required to check that the correction is effective.

If the correction value configured in the calibration circuitry results from an accurate crystal frequency measure, the remaining accuracy is bounded by the values listed below:

- Below 1 ppm, for an initial crystal drift between 1.5 ppm up to 20 ppm, and from 30 ppm to 90 ppm
- Below 2 ppm, for an initial crystal drift between 20 ppm up to 30 ppm, and from 90 ppm to 130 ppm
- Below 5 ppm, for an initial crystal drift between 130 ppm up to 200 ppm

The calibration circuitry does not modify the 32.768 kHz crystal oscillator clock frequency but it acts by slightly modifying the 1 Hz clock period from time to time. The correction event occurs every 1 + [(20 - (19 x HIGHPPM)) x CORRECTION] seconds. When the period is modified, depending on the sign of the correction, the 1 Hz clock period increases or reduces by around 4 ms. Depending on the CORRECTION, NEGPPM and HIGHPPM values configured in RTC_MR, the period interval between two correction events differs.

Figure 25-6. Calibration Circuitry

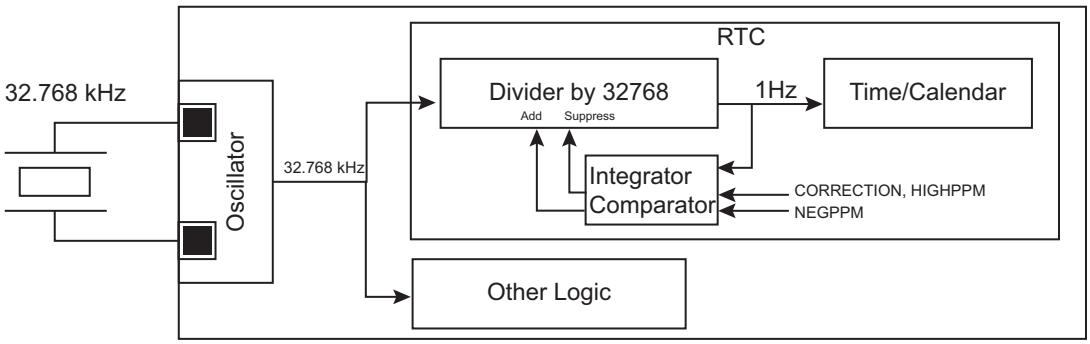
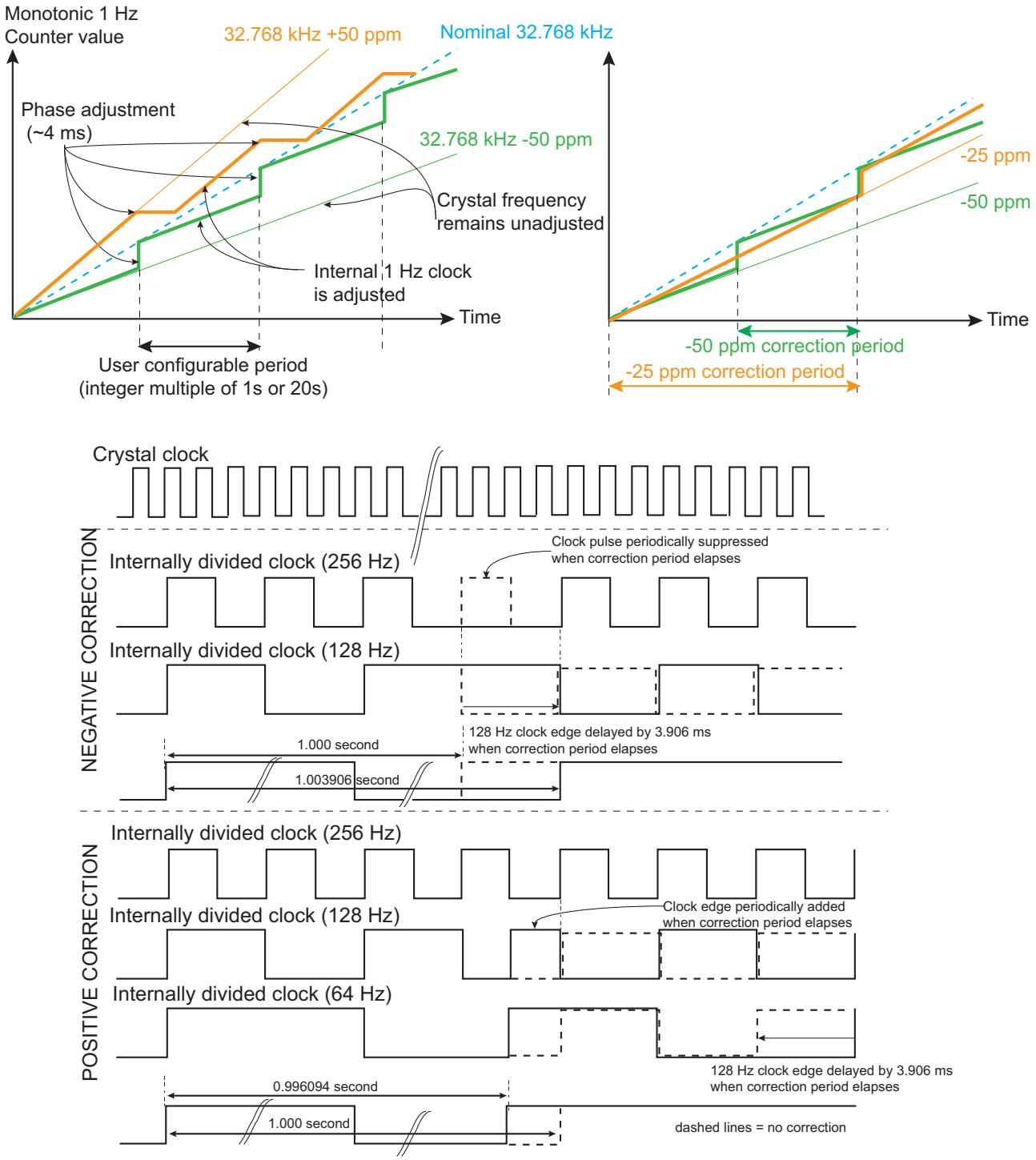


Figure 25-7. Calibration Circuitry Waveforms



The inaccuracy of a crystal oscillator at typical room temperature (±20 ppm at 20–25 °C) can be compensated if a reference clock/signal is used to measure such inaccuracy. This kind of calibration operation can be set up during the final product manufacturing by means of measurement equipment embedding such a reference clock. The correction of value must be programmed into the (RTC_MR), and this value is kept as long as the circuitry is powered (backup area). Removing the backup power supply cancels this calibration. This room temperature calibration can be further processed by means of the networking capability of the target application.

In any event, this adjustment does not take into account the temperature variation.

The frequency drift (up to -200 ppm) due to temperature variation can be compensated using a reference time if the application can access such a reference. If a reference time cannot be used, a temperature sensor can be placed close to the crystal oscillator in order to get the operating temperature of the crystal oscillator. Once obtained, the temperature may be converted using a lookup table (describing the accuracy/temperature curve of the crystal oscillator used) and RTC_MR configured accordingly. The calibration can be performed on-the-fly. This adjustment method is not based on a measurement of the crystal frequency/drift and therefore can be improved by means of the networking capability of the target application.

If no crystal frequency adjustment has been done during manufacturing, it is still possible to do it. In the case where a reference time of the day can be obtained through LAN/WAN network, it is possible to calculate the drift of the application crystal oscillator by comparing the values read on RTC Time Register (RTC_TIMR) and programming the HIGHPPM and CORRECTION fields on RTC_MR according to the difference measured between the reference time and those of RTC_TIMR.

25.5.8 Waveform Generation

Waveforms can be generated by the RTC in order to take advantage of the RTC inherent prescalers while the RTC is the only powered circuitry (Low-power mode of operation, Backup mode) or in any active mode. Going into Backup or Low-power operating modes does not affect the waveform generation outputs.

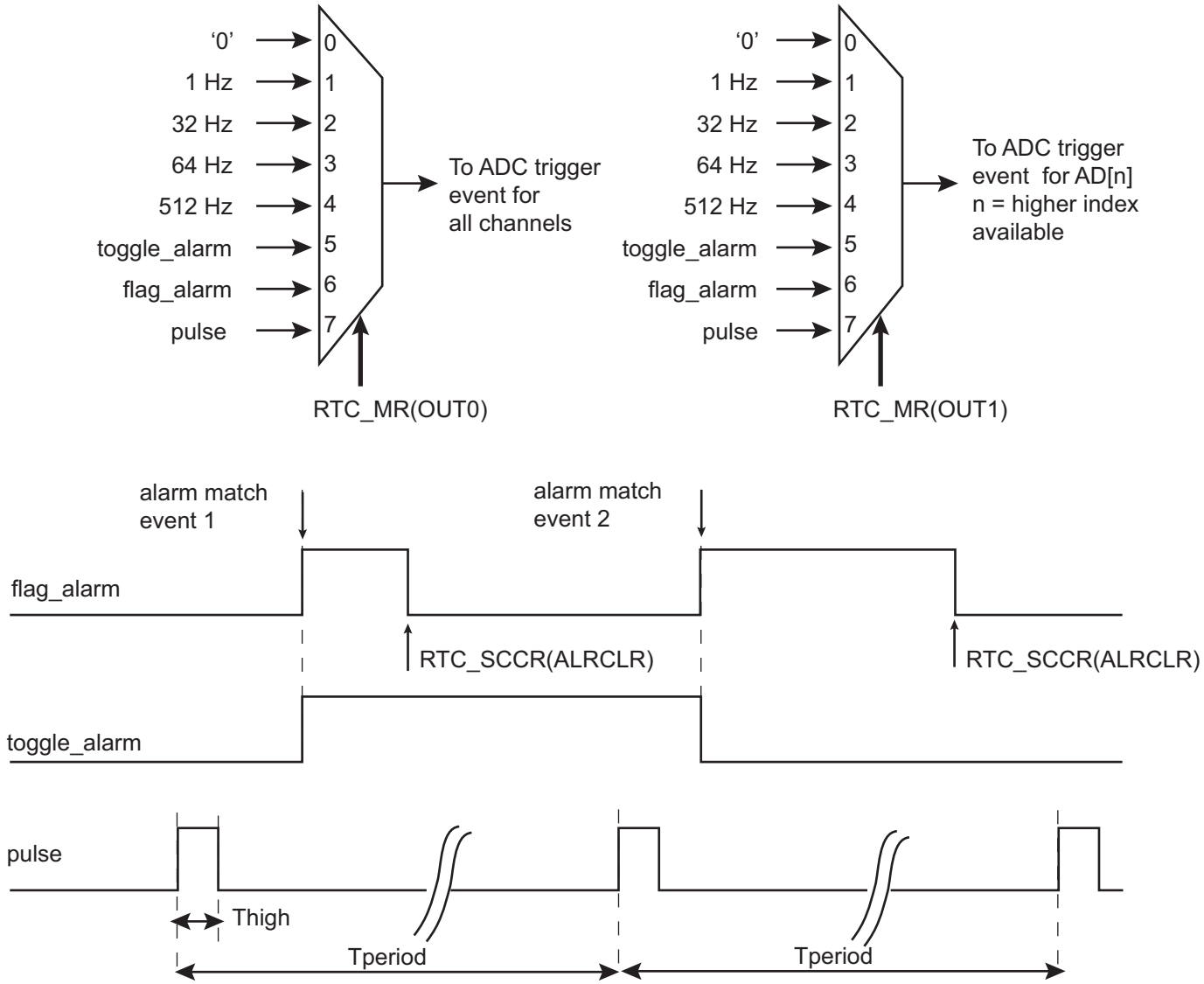
The RTC waveforms are internally routed to ADC trigger events and those events have a source driver selected among five possibilities. Two different triggers can be generated at a time, the first one is configurable through field OUT0 in RTC_MR while the second trigger is configurable through field OUT1 in RTC_MR. OUT0 field manages the trigger for channel AD[n:0] (where n is the higher index available (last channel)), while OUT1 manages the channel AD[n] only for specific modes. See the ADC section for selection of the measurement triggers and associated mode of operations.

The first selection choice sticks the associated output at 0 (This is the reset value and it can be used at any time to disable the waveform generation).

Selection choices 1 to 4 respectively select 1 Hz, 32 Hz, 64 Hz and 512 Hz.

Selection choice 6 provides a copy of the alarm flag, so the associated output is set high (logical 1) when an alarm occurs and immediately cleared when software clears the alarm interrupt source.

Figure 25-8. Waveform Generation for ADC Trigger Event



25.5.9 Tamper Timestamping

As soon as a tamper is detected, the tamper counter is incremented and the RTC stores the time of the day, the date and the source of the tamper event in registers located in the backup area. Up to two tamper events can be stored.

In UTC mode, only the UTC time is stored. The date information is not relevant.

The tamper counter saturates at 15. Once this limit is reached, the exact number of tamper occurrences since the last read of stamping registers cannot be known.

The first set of timestamping registers (RTC_TSTR0, RTC_TSDR0, RTC_TSSR0) cannot be overwritten, so once they have been written all data are stored until the registers are reset. Therefore these registers are storing the first tamper occurrence after a read.

The second set of timestamping registers (RTC_TSTR1, RTC_TSDR1, RTC_TSSR1) are overwritten each time a tamper event is detected. Thus the date and the time data of the first and the second stamping registers may be equal. This occurs when the tamper counter value carried on field TEVCNT in RTC_TSTR0 equals 1. Thus this second set of registers stores the last occurrence of tamper before a read.

Reading a set of timestamping registers requires three accesses, one for the time of the day, one for the date and one for the tamper source.

Reading the third part (RTC_TSSR0/1) of a timestamping register set clears the whole content of the registers (time, date and tamper source) and makes the timestamping registers available to store a new event.

25.6 Real-time Clock (RTC) User Interface

Table 25-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	RTC_CR	Read/Write	0x00000000
0x04	Mode Register	RTC_MR	Read/Write	0x00000000
0x08	Time Register	RTC_TIMR	Read/Write	0x00000000
0x0C	Calendar Register	RTC_CALR	Read/Write	0x01E11220
0x10	Time Alarm Register	RTC_TIMALR	Read/Write	0x00000000
0x14	Calendar Alarm Register	RTC_CALALR	Read/Write	0x01010000
0x18	Status Register	RTC_SR	Read-only	0x00000000
0x1C	Status Clear Command Register	RTC_SCCR	Write-only	–
0x20	Interrupt Enable Register	RTC_IER	Write-only	–
0x24	Interrupt Disable Register	RTC_IDR	Write-only	–
0x28	Interrupt Mask Register	RTC_IMR	Read-only	0x00000000
0x2C	Valid Entry Register	RTC_VER	Read-only	0x00000000
0xB0	TimeStamp Time Register 0	RTC_TSTR0	Read-only	0x00000000
0xB4	TimeStamp Date Register 0	RTC_TSDR0	Read-only	0x00000000
0xB8	TimeStamp Source Register 0	RTC_TSSR0	Read-only	0x00000000
0xBC	TimeStamp Time Register 1	RTC_TSTR1	Read-only	0x00000000
0xC0	TimeStamp Date Register 1	RTC_TSDR1	Read-only	0x00000000
0xC4	TimeStamp Source Register 1	RTC_TSSR1	Read-only	0x00000000
0xC8	Reserved	–	–	–
0xCC	Reserved	–	–	–
0xD0	Reserved	–	–	–
0xD4–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	RTC_WPMR	Read/Write	0x00000000
0xE8–0xF8	Reserved	–	–	–
0xFC	Reserved	–	–	–

Note: If an offset is not listed in the table it must be considered as reserved.

25.6.1 RTC Control Register

Name: RTC_CR

Address: 0xF80480B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	CALEVSEL	
15	14	13	12	11	10	9	8
–	–	–	–	–	–	TIMEVSEL	
7	6	5	4	3	2	1	0
–	–	–	–	–	–	UPDCAL	UPDTIM

This register can only be written if the WPEN bit is cleared in the RTC Write Protection Mode Register.

- **UPDTIM: Update Request Time Register**

0: No effect or, if UPDTIM has been previously written to 1, stops the update procedure.

1: Stops the RTC time counting.

Time counting consists of second, minute and hour counters. Time counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

- **UPDCAL: Update Request Calendar Register**

0: No effect or, if UPDCAL has been previously written to 1, stops the update procedure.

1: Stops the RTC calendar counting.

Calendar counting consists of day, date, month, year and century counters. Calendar counters can be programmed once this bit is set and acknowledged by the bit ACKUPD of the RTC_SR.

Note: In UTC mode, this bit has no effect on the RTC behavior.

- **TIMEVSEL: Time Event Selection**

The event that generates the flag TIMEV in RTC_SR depends on the value of TIMEVSEL.

Value	Name	Description
0	MINUTE	Minute change
1	HOUR	Hour change
2	MIDNIGHT	Every day at midnight
3	NOON	Every day at noon

Note: In UTC mode, this field has no effect on the RTC_SR.

- **CALEVSEL: Calendar Event Selection**

The event that generates the flag CALEV in RTC_SR depends on the value of CALEVSEL

Value	Name	Description
0	WEEK	Week change (every Monday at time 00:00:00)
1	MONTH	Month change (every 01 of each month at time 00:00:00)
2	YEAR	Year change (every January 1 at time 00:00:00)
3	–	Reserved

Note: In UTC mode, this field has no effect on the RTC_SR.

25.6.2 RTC Mode Register

Name: RTC_MR

Address: 0xF80480B4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	TPERIOD		–	THIGH		
23	22	21	20	19	18	17	16
–	OUT1			–	OUT0		
15	14	13	12	11	10	9	8
HIGHPPM	CORRECTION						
7	6	5	4	3	2	1	0
–	–	–	NEGPPM	–	UTC	PERSIAN	HRMOD

This register can only be written if the WPEN bit is cleared in the RTC Write Protection Mode Register.

- **HRMOD: 12-/24-hour Mode**

0: 24-hour mode is selected.

1: 12-hour mode is selected.

- **PERSIAN: PERSIAN Calendar**

0: Gregorian calendar.

1: Persian calendar.

- **UTC: UTC Time Format**

0: Gregorian or Persian calendar.

1: UTC format.

It is forbidden to write a one to the UTC and PERSIAN bits at the same time.

- **NEGPPM: NEGative PPM Correction**

0: Positive correction (the divider will be slightly higher than 32768).

1: Negative correction (the divider will be slightly lower than 32768).

Refer to CORRECTION and HIGHPPM field descriptions.

Note: NEGPPM must be cleared to correct a crystal slower than 32.768 kHz.

- **CORRECTION: Slow Clock Correction**

0: No correction

1–127: The slow clock will be corrected according to the formula given in HIGHPPM description.

- **HIGHPPM: HIGH PPM Correction**

0: Lower range ppm correction with accurate correction.

1: Higher range ppm correction with accurate correction.

If the absolute value of the correction to be applied is lower than 30 ppm, it is recommended to clear HIGHPPM. HIGHPPM set to 1 is recommended for 30 ppm correction and above.

Formula:

If HIGHPPM = 0, then the clock frequency correction range is from 1.5 ppm up to 98 ppm. The RTC accuracy is less than 1 ppm for a range correction from 1.5 ppm up to 30 ppm.

The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$CORRECTION = \frac{3906}{20 \times ppm} - 1$$

The value obtained must be rounded to the nearest integer prior to being programmed into CORRECTION field.

If HIGHPPM = 1, then the clock frequency correction range is from 30.5 ppm up to 1950 ppm. The RTC accuracy is less than 1 ppm for a range correction from 30.5 ppm up to 90 ppm.

The correction field must be programmed according to the required correction in ppm; the formula is as follows:

$$CORRECTION = \frac{3906}{ppm} - 1$$

The value obtained must be rounded to the nearest integer prior to be programmed into CORRECTION field.

If NEGPPM is set to 1, the ppm correction is negative (used to correct crystals that are faster than the nominal 32.768 kHz).

- **OUT0: All ADC Channel Trigger Event Source Selection**

Value	Name	Description
0	NO_WAVE	No waveform, stuck at ‘0’
1	FREQ1HZ	1 Hz square wave
2	FREQ32HZ	32 Hz square wave
3	FREQ64HZ	64 Hz square wave
4	FREQ512HZ	512 Hz square wave
5	ALARM_TOGGLE	Output toggles when alarm flag rises
6	ALARM_FLAG	Output is a copy of the alarm flag
7	PROG_PULSE	Duty cycle programmable pulse

- **OUT1: ADC Last Channel Trigger Event Source Selection**

Value	Name	Description
0	NO_WAVE	No waveform, stuck at ‘0’
1	FREQ1HZ	1 Hz square wave
2	FREQ32HZ	32 Hz square wave
3	FREQ64HZ	64 Hz square wave
4	FREQ512HZ	512 Hz square wave
5	ALARM_TOGGLE	Output toggles when alarm flag rises
6	ALARM_FLAG	Output is a copy of the alarm flag
7	PROG_PULSE	Duty cycle programmable pulse

- **THIGH: High Duration of the Output Pulse**

Value	Name	Description
0	H_31MS	31.2 ms
1	H_16MS	15.6 ms
2	H_4MS	3.91 ms
3	H_976US	976 µs
4	H_488US	488 µs
5	H_122US	122 µs
6	H_30US	30.5 µs
7	H_15US	15.2 µs

- **TPERIOD: Period of the Output Pulse**

Value	Name	Description
0	P_1S	1 second
1	P_500MS	500 ms
2	P_250MS	250 ms
3	P_125MS	125 ms

25.6.3 RTC Time Register

Name: RTC_TIMR

Address: 0xF80480B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	AMPM	HOUR					

15	14	13	12	11	10	9	8
–	MIN						

7	6	5	4	3	2	1	0
–	SEC						

- **SEC: Current Second**

The range that can be set is 0–59 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

- **MIN: Current Minute**

The range that can be set is 0–59 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

- **HOUR: Current Hour**

The range that can be set is 1–12 (BCD) in 12-hour mode or 0–23 (BCD) in 24-hour mode.

- **AMPM: Ante Meridiem Post Meridiem Indicator**

This bit is the AM/PM indicator in 12-hour mode.

0: AM.

1: PM.

25.6.4 RTC Time Register (UTC_MODE)

Name: RTC_TIMR (UTC_MODE)

Address: 0xF80480B8

Access: Read/Write

31	30	29	28	27	26	25	24
UTC_TIME							
23	22	21	20	19	18	17	16
UTC_TIME							
15	14	13	12	11	10	9	8
UTC_TIME							
7	6	5	4	3	2	1	0
UTC_TIME							

This configuration is relevant only if UTC = 1 in RTC_MR

- **UTC_TIME: Current UTC Time**

Any value can be set.

25.6.5 RTC Calendar Register

Name: RTC_CALR

Address: 0xF80480BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	DATE					

23	22	21	20	19	18	17	16
DAY			MONTH				

15	14	13	12	11	10	9	8
YEAR							

7	6	5	4	3	2	1	0
–	CENT						

Note: In UTC mode, values read in this register are not relevant

- **CENT: Current Century**

The range that can be set is 19–20 (Gregorian) or 13–14 (Persian) (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

- **YEAR: Current Year**

The range that can be set is 00–99 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

- **MONTH: Current Month**

The range that can be set is 01–12 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

- **DAY: Current Day in Current Week**

The range that can be set is 1–7 (BCD).

The coding of the number (which number represents which day) is user-defined as it has no effect on the date counter.

- **DATE: Current Day in Current Month**

The range that can be set is 01–31 (BCD).

The lowest four bits encode the units. The higher bits encode the tens.

25.6.6 RTC Time Alarm Register

Name: RTC_TIMALR

Address: 0xF80480C0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
HOUREN	AMPM	HOUR					
15	14	13	12	11	10	9	8
MINEN	MIN						
7	6	5	4	3	2	1	0
SECEN	SEC						

This register can only be written if the WPEN bit is cleared in the RTC Write Protection Mode Register.

Note: To change one of the SEC, MIN, HOUR fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_TIMALR. The first access clears the enable corresponding to the field to change (SECEN, MINEN, HOUREN). If the field is already cleared, this access is not required. The second access performs the change of the value (SEC, MIN, HOUR). The third access is required to re-enable the field by writing 1 in SECEN, MINEN, HOUREN fields.

- **SEC: Second Alarm**

This field is the alarm field corresponding to the BCD-coded second counter.

- **SECEN: Second Alarm Enable**

0: The second-matching alarm is disabled.

1: The second-matching alarm is enabled.

- **MIN: Minute Alarm**

This field is the alarm field corresponding to the BCD-coded minute counter.

- **MINEN: Minute Alarm Enable**

0: The minute-matching alarm is disabled.

1: The minute-matching alarm is enabled.

- **HOUR: Hour Alarm**

This field is the alarm field corresponding to the BCD-coded hour counter.

- **AMPM: AM/PM Indicator**

This field is the alarm field corresponding to the BCD-coded hour counter.

- **HOUREN: Hour Alarm Enable**

0: The hour-matching alarm is disabled.

1: The hour-matching alarm is enabled.

25.6.7 RTC Time Alarm Register (UTC_MODE)

Name: RTC_TIMALR (UTC_MODE)

Address: 0xF80480C0

Access: Read/Write

31	30	29	28	27	26	25	24
UTC_TIME							
23	22	21	20	19	18	17	16
UTC_TIME							
15	14	13	12	11	10	9	8
UTC_TIME							
7	6	5	4	3	2	1	0
UTC_TIME							

This configuration is relevant only if UTC = 1 in RTC_MR.

- **UTC_TIME: UTC_TIME Alarm**

This field is the alarm field corresponding to the UTC time counter. To change it, proceed as follows:

1. Disable the UTC alarm by clearing the UTCEN bit in RTC_CALALR if it is not already cleared.
2. Change the UTC_TIME alarm value.
3. Enable the UTC alarm by setting the UTCEN bit in RTC_CALALR.

25.6.8 RTC Calendar Alarm Register

Name: RTC_CALALR

Address: 0xF80480C4

Access: Read/Write

31	30	29	28	27	26	25	24
DATEEN	–	DATE					
23	22	21	20	19	18	17	16
MTHEN	–	–	MONTH				
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	

This register can only be written if the WPEN bit is cleared in the RTC Write Protection Mode Register.

Note: To change one of the DATE, MONTH fields, it is recommended to disable the field before changing the value and then re-enable it after the change has been made. This requires up to three accesses to the RTC_CALALR. The first access clears the enable corresponding to the field to change (DATEEN, MTHEN). If the field is already cleared, this access is not required. The second access performs the change of the value (DATE, MONTH). The third access is required to re-enable the field by writing 1 in DATEEN, MTHEN fields.

- **MONTH: Month Alarm**

This field is the alarm field corresponding to the BCD-coded month counter.

- **MTHEN: Month Alarm Enable**

0: The month-matching alarm is disabled.

1: The month-matching alarm is enabled.

- **DATE: Date Alarm**

This field is the alarm field corresponding to the BCD-coded date counter.

- **DATEEN: Date Alarm Enable**

0: The date-matching alarm is disabled.

1: The date-matching alarm is enabled.

25.6.9 RTC Calendar Alarm Register (UTC_MODE)

Name: RTC_CALALR (UTC_MODE)

Address: 0xF80480C4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	UTCEN

This register can only be written if the WPEN bit is cleared in the System Controller Write Protection Mode Register (SYSC_WPMR).

- **UTCEN: UTC Alarm Enable**

0: The UTC-matching alarm is disabled.

1: The UTC-matching alarm is enabled.

25.6.10 RTC Status Register

Name: RTC_SR

Address: 0xF80480C8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TDERR	CALEV	TIMEV	SEC	ALARM	ACKUPD

- **ACKUPD: Acknowledge for Update**

Value	Name	Description
0	FREERUN	Time and calendar registers cannot be updated.
1	UPDATE	Time and calendar registers can be updated.

- **ALARM: Alarm Flag**

Value	Name	Description
0	NO_ALARMEVENT	No alarm matching condition occurred.
1	ALARMEVENT	An alarm matching condition has occurred.

- **SEC: Second Event**

Value	Name	Description
0	NO_SECEVENT	No second event has occurred since the last clear.
1	SECEVENT	At least one second event has occurred since the last clear.

- **TIMEV: Time Event**

Value	Name	Description
0	NO_TIMEVENT	No time event has occurred since the last clear.
1	TIMEVENT	At least one time event has occurred since the last clear.

Note: The time event is selected in the TIMEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: minute change, hour change, noon, midnight (day change). If the RTC is configured in UTC mode, the value returned by this field is not relevant.

- **CALEV: Calendar Event**

Value	Name	Description
0	NO_CALEVENT	No calendar event has occurred since the last clear.
1	CALEVENT	At least one calendar event has occurred since the last clear.

Note: The calendar event is selected in the CALEVSEL field in the Control Register (RTC_CR) and can be any one of the following events: week change, month change and year change. If the RTC is configured in UTC mode, the value returned by this field is not relevant.

- **TDERR: Time and/or Date Free Running Error**

Value	Name	Description
0	CORRECT	The internal free running counters are carrying valid values since the last read of the Status Register (RTC_SR).
1	ERR_TIMEDATE	The internal free running counters have been corrupted (invalid date or time, non-BCD values) since the last read and/or they are still invalid.

Note: If the RTC is configured in UTC mode, the value returned by this field is not relevant.

25.6.11 RTC Status Clear Command Register

Name: RTC_SCCR

Address: 0xF80480CC

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TDERRCLR	CALCLR	TIMCLR	SECCLR	ALRCLR	ACKCLR

- **ACKCLR: Acknowledge Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **ALRCLR: Alarm Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **SECCLR: Second Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

- **TIMCLR: Time Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

If the RTC is configured in UTC mode, this bit has no effect.

- **CALCLR: Calendar Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

If the RTC is configured in UTC mode, this bit has no effect.

- **TDERRCLR: Time and/or Date Free Running Error Clear**

0: No effect.

1: Clears corresponding status flag in the Status Register (RTC_SR).

If the RTC is configured in UTC mode, this bit has no effect.

25.6.12 RTC Interrupt Enable Register

Name: RTC_IER

Address: 0xF80480D0

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TDERREN	CALEN	TIMEN	SECEN	ALREN	ACKEN

- **ACKEN: Acknowledge Update Interrupt Enable**

0: No effect.

1: The acknowledge for update interrupt is enabled.

- **ALREN: Alarm Interrupt Enable**

0: No effect.

1: The alarm interrupt is enabled.

- **SECEN: Second Event Interrupt Enable**

0: No effect.

1: The second periodic interrupt is enabled.

- **TIMEN: Time Event Interrupt Enable**

0: No effect.

1: The selected time event interrupt is enabled.

If the RTC is configured in UTC mode, this bit has no effect.

- **CALEN: Calendar Event Interrupt Enable**

0: No effect.

1: The selected calendar event interrupt is enabled.

If the RTC is configured in UTC mode, this bit has no effect.

- **TDERREN: Time and/or Date Error Interrupt Enable**

0: No effect.

1: The time and date error interrupt is enabled.

If the RTC is configured in UTC mode, this bit has no effect.

25.6.13 RTC Interrupt Disable Register

Name: RTC_IDR

Address: 0xF80480D4

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TDERRDIS	CALDIS	TIMDIS	SECDIS	ALRDIS	ACKDIS

- **ACKDIS: Acknowledge Update Interrupt Disable**

0: No effect.

1: The acknowledge for update interrupt is disabled.

- **ALRDIS: Alarm Interrupt Disable**

0: No effect.

1: The alarm interrupt is disabled.

- **SECDIS: Second Event Interrupt Disable**

0: No effect.

1: The second periodic interrupt is disabled.

- **TIMDIS: Time Event Interrupt Disable**

0: No effect.

1: The selected time event interrupt is disabled.

If the RTC is configured in UTC mode, this bit has no effect.

- **CALDIS: Calendar Event Interrupt Disable**

0: No effect.

1: The selected calendar event interrupt is disabled.

If the RTC is configured in UTC mode, this bit has no effect.

- **TDERRDIS: Time and/or Date Error Interrupt Disable**

0: No effect.

1: The time and date error interrupt is disabled.

If the RTC is configured in UTC mode, this bit has no effect.

25.6.14 RTC Interrupt Mask Register

Name: RTC_IMR

Address: 0xF80480D8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TDERR	CAL	TIM	SEC	ALR	ACK

- **ACK: Acknowledge Update Interrupt Mask**

0: The acknowledge for update interrupt is disabled.

1: The acknowledge for update interrupt is enabled.

- **ALR: Alarm Interrupt Mask**

0: The alarm interrupt is disabled.

1: The alarm interrupt is enabled.

- **SEC: Second Event Interrupt Mask**

0: The second periodic interrupt is disabled.

1: The second periodic interrupt is enabled.

- **TIM: Time Event Interrupt Mask**

0: The selected time event interrupt is disabled.

1: The selected time event interrupt is enabled.

If the RTC is configured in UTC mode, this bit is not relevant.

- **CAL: Calendar Event Interrupt Mask**

0: The selected calendar event interrupt is disabled.

1: The selected calendar event interrupt is enabled.

If the RTC is configured in UTC mode, this bit is not relevant.

- **TDERR: Time and/or Date Error Mask**

0: The time and/or date error event is disabled.

1: The time and/or date error event is enabled.

If the RTC is configured in UTC mode, this bit has no effect.

25.6.15 RTC Valid Entry Register

Name: RTC_VER

Address: 0xF80480DC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	NVCALALR	NVTIMALR	NVCAL	NVTIM

If the RTC is configured in UTC mode, the values returned by this register are not relevant.

- **NVTIM: Non-valid Time**

0: No invalid data has been detected in RTC_TIMR (Time Register).

1: RTC_TIMR has contained invalid data since it was last programmed.

- **NVCAL: Non-valid Calendar**

0: No invalid data has been detected in RTC_CALR (Calendar Register).

1: RTC_CALR has contained invalid data since it was last programmed.

- **NVTIMALR: Non-valid Time Alarm**

0: No invalid data has been detected in RTC_TIMALR (Time Alarm Register).

1: RTC_TIMALR has contained invalid data since it was last programmed.

- **NVCALALR: Non-valid Calendar Alarm**

0: No invalid data has been detected in RTC_CALALR (Calendar Alarm Register).

1: RTC_CALALR has contained invalid data since it was last programmed.

25.6.16 RTC TimeStamp Time Register 0

Name: RTC_TSTR0

Address: 0xF8048160

Access: Read-only

31	30	29	28	27	26	25	24
BACKUP	–	–	–	TEVCNT			
23	22	21	20	19	18	17	16
–	AMPM	HOUR					
15	14	13	12	11	10	9	8
–	MIN						
7	6	5	4	3	2	1	0
–	SEC						

These fields are valid for non-UTC mode only.

RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0.

This register is cleared by reading RTC_TSSR0.

- **SEC: Seconds of the Tamper**
- **MIN: Minutes of the Tamper**
- **HOUR: Hours of the Tamper**
- **AMPM: AM/PM Indicator of the Tamper**
- **TEVCNT: Tamper Events Counter**

Each time a tamper event occurs, this counter is incremented. This counter saturates at 15. Once this value is reached, it is no more possible to know the exact number of tamper events.

If this field is not null, this implies that at least one tamper event occurs since last register reset and that the values stored in timestamping registers are valid.

- **BACKUP: System Mode of the Tamper**

0: The state of the system is different from backup mode when the tamper event occurs.

1: The system is in backup mode when the tamper event occurs.

25.6.17 RTC TimeStamp Time Register 0 (UTC_MODE)

Name: RTC_TSTR0 (UTC_MODE)

Address: 0xF8048160

Access: Read-only

31	30	29	28	27	26	25	24
BACKUP	–	–	–	TEVCNT			
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

RTC_TSTR0 reports the timestamp of the first tamper event.

- **TEVCNT: Tamper Events Counter (cleared by reading RTC_TSSR0)**

Each time a tamper event occurs, this counter is incremented. This counter saturates at 15. Once this value is reached, it is no more possible to know the exact number of tamper events.

If this field is not null, this implies that at least one tamper event occurs since last register reset and that the values stored in timestamping registers are valid.

- **BACKUP: System Mode of the Tamper (cleared by reading RTC_TSSR0)**

0: The state of the system is different from Backup mode when the tamper event occurs.

1: The system is in Backup mode when the tamper event occurs.

25.6.18 RTC TimeStamp Time Register 1

Name: RTC_TSTR1

Address: 0xF804816C

Access: Read-only

31	30	29	28	27	26	25	24
BACKUP	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	AMPM	HOUR					

15	14	13	12	11	10	9	8
–	MIN						

7	6	5	4	3	2	1	0
–	SEC						

These fields are valid for non-UTC mode only.

RTC_TSTR1 reports the timestamp of the last tamper event.

This register is cleared by reading RTC_TSSR1.

- **SEC: Seconds of the Tamper**
- **MIN: Minutes of the Tamper**
- **HOUR: Hours of the Tamper**
- **AMPM: AM/PM Indicator of the Tamper**
- **BACKUP: System Mode of the Tamper**

0: The state of the system is different from Backup mode when the tamper event occurs.

1: The system is in Backup mode when the tamper event occurs.

25.6.19 RTC TimeStamp Time Register 1 (UTC_MODE)

Name: RTC_TSTR1 (UTC_MODE)

Address: 0xF804816C

Access: Read-only

31	30	29	28	27	26	25	24
BACKUP	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

RTC_TSTR1 reports the timestamp of the last tamper event.

- **BACKUP: System Mode of the Tamper**

0: The state of the system is different from Backup mode when the tamper event occurs.

1: The system is in Backup mode when the tamper event occurs.

25.6.20 RTC TimeStamp Date Register

Name: RTC_TSDRx

Address: 0xF8048164 [0], 0xF8048170 [1]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	DATE					

23	22	21	20	19	18	17	16
DAY			MONTH				

15	14	13	12	11	10	9	8
YEAR							

7	6	5	4	3	2	1	0
–	CENT						

These fields are relevant for non-UTC mode only.

RTC_TSTR0 reports the timestamp of the first tamper event after reading RTC_TSSR0, and RTC_TSTR1 reports the timestamp of the last tamper event.

This register is cleared by reading RTC_TSSR.

- **CENT: Century of the Tamper**
- **YEAR: Year of the Tamper**
- **MONTH: Month of the Tamper**
- **DAY: Day of the Tamper**
- **DATE: Date of the Tamper**

The fields contain the date and the source of a tamper occurrence if the TEVCNT is not null.

25.6.21 RTC TimeStamp Date Register (UTC_MODE)

Name: RTC_TSDRx (UTC_MODE)

Address: 0xF8048164 [0], 0xF8048170 [1]

Access: Read-only

31	30	29	28	27	26	25	24
UTC_TIME							

23	22	21	20	19	18	17	16
UTC_TIME							

15	14	13	12	11	10	9	8
UTC_TIME							

7	6	5	4	3	2	1	0
UTC_TIME							

- **UTC_TIME: Time of the Tamper (UTC format)**

This configuration is relevant only if UTC = 1 in RTC_MR.

25.6.22 RTC TimeStamp Source Register

Name: RTC_TSSRx

Address: 0xF8048168 [0], 0xF8048174 [1]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
DET7	DET6	DET5	DET4	DET3	DET2	DET1	DET0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	JTAG	TST	–	–

The following configuration values are valid for all listed bit names of this register:

0: No alarm generated since the last clear.

1: An alarm has been generated by the corresponding monitor since the last clear.

- **TST: Test Pin Monitor**
- **JTAG: JTAG Pins Monitor**
- **DETx: PIOBU Intrusion Detector**

25.6.23 RTC Write Protection Mode Register

Name: RTC_WPMR

Address: 0xF8048194

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x525443 (“RTC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x525443 (“RTC” in ASCII).

The following registers can be write-protected:

- RTC Mode Register
- RTC Time Alarm Register
- RTC Calendar Alarm Register

- **WPKEY: Write Protection Key**

Value	Name	Description
0x525443	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

26. Slow Clock Controller (SCKC)

26.1 Description

The System Controller embeds a Slow Clock Controller (SCKC). The SCKC selects the slow clock from one of two sources:

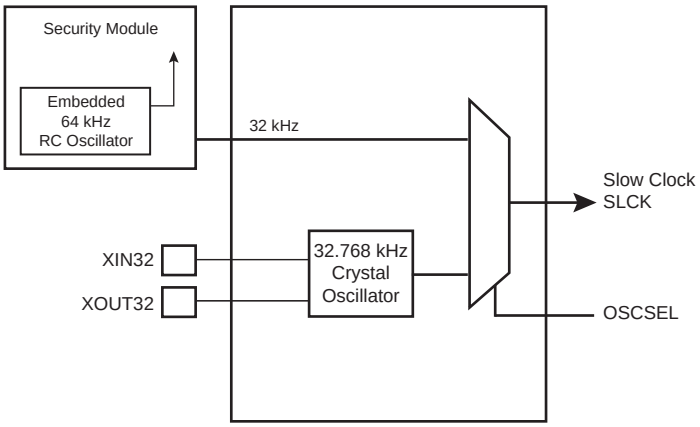
- External 32.768 kHz crystal oscillator
- Embedded 64 kHz (typical) RC oscillator

26.2 Embedded Characteristics

- 64 kHz (typical) RC Oscillator or 32.768 kHz Crystal Oscillator Selector
- VDDBU Powered

26.3 Block Diagram

Figure 26-1. Block Diagram



26.4 Functional Description

The OSCSEL bit is located in the Slow Clock Controller Configuration Register (SCKC_CR) located at the address 0xFC068650 in the backed-up part of the System Controller and, thus, it is preserved while VDDBU is present.

The embedded 64 kHz (typical) RC oscillator and the 32.768 kHz crystal oscillator are always enabled as soon as VDDBU is established. The Slow Clock Selector command (OSCSEL bit) selects the slow clock source.

After the VDDBU power-on reset, the default configuration is OSCSEL = 0, allowing the system to start on the embedded 64 kHz (typical) RC oscillator.

The programmer controls the slow clock switching by software and so must take precautions during the switching phase.

26.4.1 Switching from Embedded 64 kHz RC Oscillator to 32.768 kHz Crystal Oscillator

The sequence to switch from the embedded 64 kHz (typical) RC oscillator to the 32.768 kHz crystal oscillator is the following:

1. Switch the master clock to a source different from slow clock (PLL or Main Oscillator) through the Power Management Controller.
2. Switch from the embedded 64 kHz (typical) RC oscillator to the 32.768 kHz crystal oscillator by writing a 1 to the OSCSEL bit.
3. Wait 5 slow clock cycles for internal resynchronization.

26.4.2 Switching from 32.768 kHz Crystal Oscillator to Embedded 64 kHz RC Oscillator

The sequence to switch from the 32.768 kHz crystal oscillator to the embedded 64 kHz (typical) RC oscillator is the following:

1. Switch the master clock to a source different from slow clock (PLL or Main Oscillator).
2. Switch from the 32.768 kHz crystal oscillator to the embedded RC oscillator by writing a 0 to the OSCSEL bit.
3. Wait 5 slow clock cycles for internal resynchronization.

26.5 Slow Clock Controller (SCKC) User Interface

Table 26-1. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Slow Clock Controller Configuration Register	SCKC_CR	Read/Write	0x0000_0001

26.5.1 Slow Clock Controller Configuration Register

Name: SCKC_CR

Address: 0xF8048050

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSCSEL	–	–	–

- **OSCSEL: Slow Clock Selector**

0 (RC): Slow clock is the embedded 64 kHz (typical) RC oscillator.

1 (XTAL): Slow clock is the 32.768 kHz crystal oscillator.

27. Low Power Asynchronous Receiver (RXLP)

27.1 Description

The Low Power Asynchronous Receiver (RXLP) is a low-power UART with a slow clock. It works only in Receive mode. It features a Receive Data (RXD) pin that can be used to wake up the system. The wakeup occurs only on data matching—expected data can be a single value, two values, or a range of values.

The RXLP operates on a slow clock domain to reduce power consumption.

27.2 Embedded Characteristics

- Exit from Backup Mode on Comparison Match
- Programmable Baud Rate Generator
- Even, Odd, Mark or Space Parity Check
- Parity and Framing Error Detection
- Digital Filter on Receive Line
- Comparison Function on Received Character
- Register Write Protection

27.3 Block Diagram

Figure 27-1. RXLP Functional Block Diagram

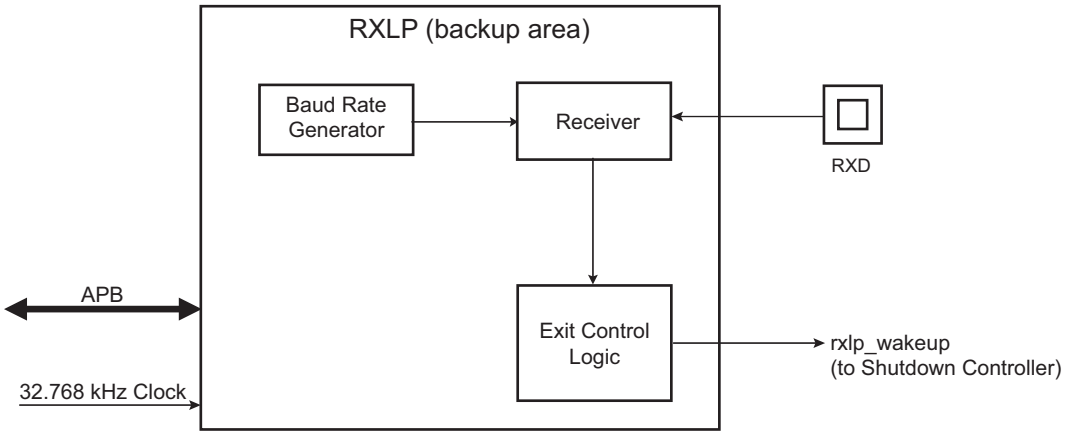


Table 27-1. RXLP Pin Description

Pin Name	Description	Type
RXD	RXLP Receive Data	Input

27.4 Product Dependencies

27.4.1 Power Management

The peripheral clock is not managed by the PMC but rather automatically activated when the RXLP is enabled and receive line is active. The peripheral clock is automatically deactivated after transmission of the wakeup signal.

27.5 Functional Description

The RXLP features an RS232 receive-only circuitry able to decode and compare data and parity while the system is in Backup mode. If a matching comparison occurs, the RXLP instructs the system to wake up (if enabled).

The RXLP operates in Asynchronous mode only and supports only 8-bit character handling (with or without parity).

The RXLP is made up of a receiver and a baud rate generator. Receiver timeout is not implemented and there is no interrupt line.

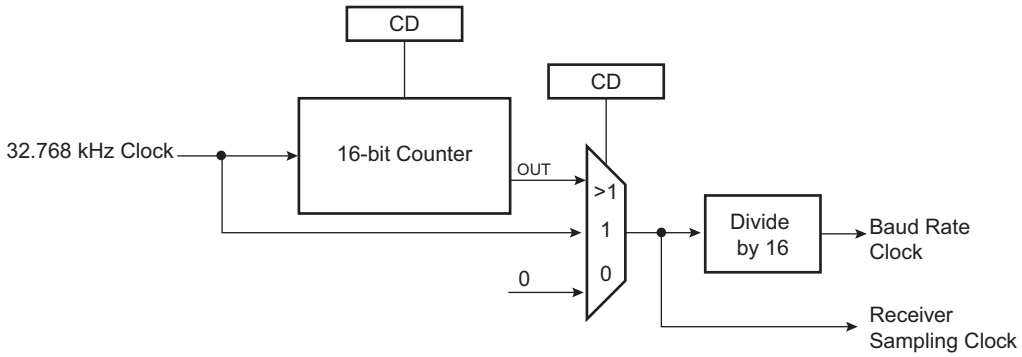
27.5.1 Baud Rate Generator

The baud rate generator provides the bit period clock named baud rate clock to the receiver.

The baud rate clock is the 32.768 kHz clock from the crystal oscillator, divided by 16 times the value (CD) written in the Baud Rate Generator Register (RXLP_BRGR). If RXLP_BRGR is set to 0, the baud rate clock is disabled and the RXLP remains inactive. The maximum allowable baud rate is 32.768 kHz clock divided by 16. The minimum allowable baud rate is 32.768 kHz clock divided by (16 × 3).

$$\text{Baud Rate} = \frac{f_{32.768\ \text{kHz clock}}}{16 \times \text{CD}}$$

Figure 27-2. Baud Rate Generator



27.5.2 Receiver

27.5.2.1 Receiver Reset, Enable and Disable

After device reset, the RXLP is disabled and must be enabled before being used. The receiver can be enabled by setting bit RXEN in the Control Register (RXLP_CR). At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by setting bit RXLP_CR.RXDIS. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The receiver can be put in reset state by setting bit RXLP_CR.RSTRX. In this case, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost. After initiating a reset it is mandatory to clear bit RXLP_CR.RSTRX.

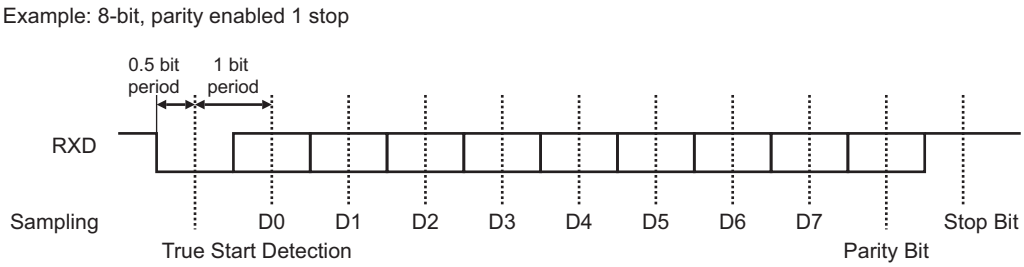
27.5.2.2 Start Detection and Data Sampling

The RXLP only supports asynchronous operations, and this affects only its receiver. The RXLP detects the start of a received character by sampling the RXD signal until it detects a valid start bit. A low level (space) on RXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the RXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after detecting the falling edge of the start bit.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

Figure 27-3. Character Reception



27.5.2.3 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Mode Register (RXLP_MR). It then compares the result with the received parity bit. If different, the received character is ignored and the receiver continues to wait for a new valid start bit.

27.5.2.4 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the received character is ignored and the receiver continues to wait for a new valid start bit.

27.5.2.5 Receiver Digital Filter

The RXLP embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in the FILTER bit of RXLP_MR. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

27.5.3 Comparison Function on Received Character

Each time a valid character is received (without parity error and without frame error) it is compared to the wakeup trigger values. If the received character matches to the condition of wakeup, it is stored in the Receiver Holding Register (RXLP_RHR), a system wakeup is generated and the RXLP is automatically disabled. If the character received does not match, it is ignored and the receiver continues to wait for a new valid start bit.

RXLP_CMPR (see Section 27.6.5 “RXLP Comparison Register”) can be programmed to provide three different comparison methods:

- VAL1 equals VAL2—the comparison is performed on a single value and the wakeup request is generated if the received character equals VAL1.
- VAL1 is strictly lower than VAL2—any value between VAL1 and VAL2 generates a wakeup request.
- VAL1 is strictly higher than VAL2—the wakeup request is generated if either received character equals VAL1 or VAL2.

27.5.4 Register Write Protection

To prevent any single software error from corrupting RXLP behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the RXLP Write Protection Mode Register (RXLP_WPMR).

The following registers can be write-protected:

- RXLP Mode Register
- RXLP Baud Rate Generator Register
- RXLP Comparison Register

27.6 Low Power Asynchronous Receiver (RXLP) User Interface

Table 27-2. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Control Register	RXLP_CR	Write-only	–
0x0004	Mode Register	RXLP_MR	Read/Write	0x0
0x0008–0x0014	Reserved	–	–	–
0x0018	Receive Holding Register	RXLP_RHR	Read-only	0x0
0x001C	Reserved	–	–	–
0x0020	Baud Rate Generator Register	RXLP_BRGR	Read/Write	0x0
0x0024	Comparison Register	RXLP_CMPR	Read/Write	0x0
0x0028–0x00E0	Reserved	–	–	–
0x00E4	Write Protection Mode Register	RXLP_WPMR	Read/Write	0x0
0x00E8–0x00FC	Reserved	–	–	–

27.6.1 RXLP Control Register

Name: RXLP_CR

Address: 0xF8049000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RXDIS	RXEN	–	RSTRX	–	–

- **RSTRX: Reset Receiver**

0: Deactivate the reset of the receiver logic.

1: The receiver logic is reset and disabled. If a character is being received, the reception is aborted. The receiver logic remains in reset state until RSTRX is written to 0.

- **RXEN: Receiver Enable**

0: No effect.

1: The receiver is enabled if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped.

27.6.2 RXLP Mode Register

Name: RXLP_MR

Address: 0xF8049004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PAR			–
7	6	5	4	3	2	1	0
–	–	–	FILTER	–	–	–	–

- **FILTER: Receiver Digital Filter**

Value	Name	Description
0	DISABLED	RXLP does not filter the receive line.
1	ENABLED	RXLP filters the receive line using a three-sample filter (16x-bit clock) (2 over 3 majority).

- **PAR: Parity Type**

Value	Name	Description
0	EVEN	Even Parity
1	ODD	Odd Parity
2	SPACE	Parity forced to 0
3	MARK	Parity forced to 1
4	NO	No parity

27.6.3 RXLP Receiver Holding Register

Name: RXLP_RHR

Address: 0xF8049018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXCHR							

- **RXCHR: Received Character**

Last received character

27.6.4 RXLP Baud Rate Generator Register

Name: RXLP_BRGR

Address: 0xF8049020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	CD	

- **CD: Clock Divisor**

0: Baud rate clock is disabled

1 to 3: $f_{32.768\ kHz\ clock}$ / (CD × 16)

27.6.5 RXLP Comparison Register

Name: RXLP_CMPR

Address: 0xF8049024

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
VAL2							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
VAL1							

- **VAL1: First Comparison Value for Received Character**

0 to 255.

The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to request a system wakeup.

- **VAL2: Second Comparison Value for Received Character**

0 to 255.

The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to request a system wakeup.

27.6.6 RXLP Write Protection Mode Register

Name: RXLP_WPMR

Address: 0xF80490E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x52584C (RXL in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x52584C (RXL in ASCII).

See Section 27.5.4 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x52584C	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

28. Analog Comparator Controller (ACC)

28.1 Description

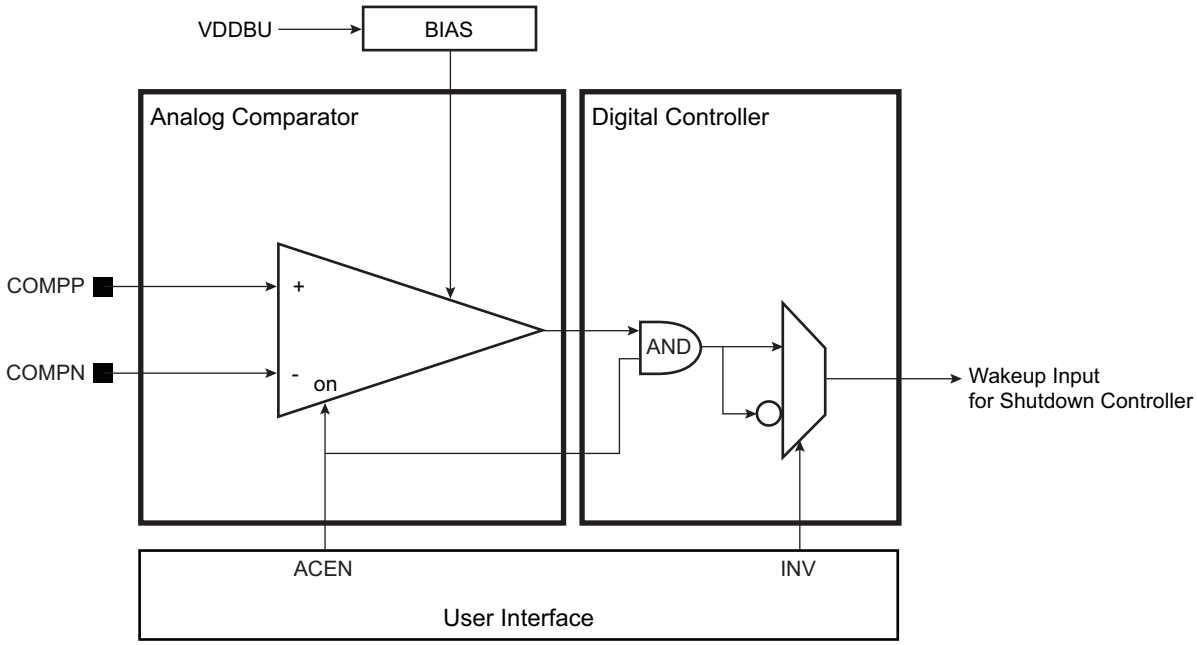
The Analog Comparator Controller (ACC) controls the analog comparator in order to provide an additional source of wakeup when the system wakes up from Wait mode.

28.2 Embedded Characteristics

- Source of Wakeup When System Wakes Up from Wait Mode and ULP1 Mode

28.3 Block Diagram

Figure 28-1. Analog Comparator Controller Block Diagram



28.4 Signal Description

Table 28-1. ACC Signal Description

Pin Name	Description	Type
COMPP, COMPN	External analog data inputs	Input

28.5 Product Dependencies

28.5.1 I/O Lines

The analog input pins (COMPP and COMPN) are not multiplexed with digital functions (PIO) on the IO line.

28.5.2 Power Management

By clearing the ACEN bit in the ACC Mode Register (ACC_MR), the analog comparator power consumption is reduced to current leakage only.

28.6 Functional Description

28.6.1 Description

The analog comparator is enabled by writing a one to the ACEN bit in the ACC Mode Register (ACC_MR) and the polarity of the comparator output can be configured with bit ACC_MR.INV.

The ACC registers are listed in Table 28-2.

28.6.2 Register Write Protection

To prevent any single software error from corrupting ACC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the ACC Write Protection Mode Register (ACC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the ACC Write Protection Status Register (ACC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the ACC_WPSR register.

The following registers can be write-protected:

- ACC Mode Register

28.7 Analog Comparator Controller (ACC) User Interface

Table 28-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	ACC_CR	Write-only	–
0x04	Mode Register	ACC_MR	Read/Write	0
0x08–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	ACC_WPMR	Read/Write	0
0xE8	Write Protection Status Register	ACC_WPSR	Read-only	0
0xEC–0xFC	Reserved	–	–	–

28.7.1 ACC Control Register

Name: ACC_CR

Address: 0xF804A000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SWRST

- **SWRST: Software Reset**

0: No effect.

1: Resets the module.

28.7.2 ACC Mode Register

Name: ACC_MR

Address: 0xF804A004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	INV	–	–	–	ACEN
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

This register can only be written if the WPEN bit is cleared in the ACC Write Protection Mode Register.

- **ACEN: Analog Comparator Enable**

0 (DIS): Analog comparator disabled.

1 (EN): Analog comparator enabled.

- **INV: Invert Comparator Output**

0 (DIS): Analog comparator output is directly processed.

1 (EN): Analog comparator output is inverted prior to being processed.

28.7.3 ACC Write Protection Mode Register

Name: ACC_WPMR

Address: 0xF804A0E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x414343 (“ACC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x414343 (“ACC” in ASCII).

See Section 28.6.2 ”Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x414343	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

28.7.4 ACC Write Protection Status Register

Name: ACC_WPSR

Address: 0xF804A0E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of ACC_WPSR.

1: A write protection violation (WPEN = 1) has occurred since the last read of ACC_WPSR.

29. Clock Generator

29.1 Description

The Clock Generator User Interface is embedded within the Power Management Controller and is described in Section 30.22 “Power Management Controller (PMC) User Interface”. However, the Clock Generator registers are named CKGR_.

29.2 Embedded Characteristics

The Clock Generator is made up of:

- A low-power 32.768 kHz crystal oscillator
- A low-power embedded 64 kHz (typical) RC oscillator generating the 32 kHz source clock
- A 8 to 24 MHz crystal oscillator or a 12 to 48 MHz XRCGB crystal resonator with Bypass mode
- A 12 MHz RC oscillator
- A 480 MHz UTMI PLL providing a clock for the USB High Speed Device Controller
- A 600 to 1200 MHz programmable PLL (input from 8 to 50 MHz), capable of providing the clock MCK to the processor and to the peripherals (HCLOCK_LS/HS and PCLOCK_LS/HS)
- A 700 MHz fractional-N programmable audio PLL, with 22-bit frequency resolution and two independent programmable post dividers to drive the CLK_AUDIO output pin and the internal peripherals (AUDIOPLLCLK)

The Clock Generator provides the following clocks:

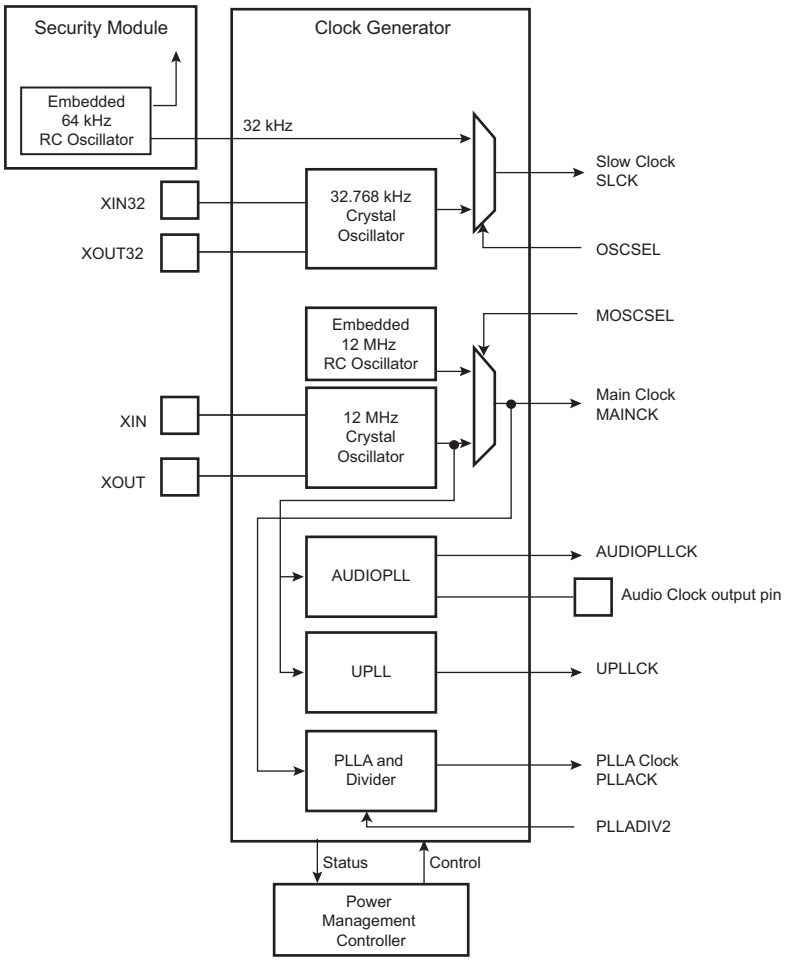
- SLCK, the Slow Clock, which is the only permanent clock within the system
- MAINCK is the output of the main clock oscillator selection: either 8 to 24 MHz crystal oscillator or 12 MHz RC oscillator
- PLLACK is the output of the divider and the 600 to 1200 MHz programmable PLL (PLLA)
- AUDIOPLLCLK is the output of the first Audio PLL post-divider, with a frequency range from 24 to 125 MHz
- AUDIOPINCLK is the output of the second Audio PLL post-divider, with a frequency range from 8 to 30 MHz
- UPLLCK is the output of the 480 MHz UTMI PLL (UPLL)

The Power Management Controller also provides the following operations on clocks:

- 8 to 24 MHz crystal oscillator clock failure detector
- 32.768 kHz crystal oscillator frequency monitor
- Frequency counter on main clock and an on-the-fly adjustable 12 MHz RC oscillator frequency

29.3 Block Diagram

Figure 29-1. Clock Generator Block Diagram



29.4 Slow Clock

The Slow Clock Controller embeds a slow clock generator that is supplied with the VDDBU power supply. As soon as VDDBU is supplied, both the 32.768 kHz crystal oscillator and the embedded 64 kHz (typical) RC oscillator are powered, but only the RC oscillator is enabled.

The slow clock is generated either by the 32.768 kHz crystal oscillator or by the embedded 64 kHz (typical) RC oscillator divided by two.

The selection of the slow clock source is made via the OSCSEL bit in the Slow Clock Controller Configuration Register (SCKC_CR).

SCKC_CR.OSCSEL and PMC_SR.OSCSELS report which oscillator is selected as the slow clock source. PMC_SR.OSCSELS informs when the switch sequence initiated by a new value written in SCKC_CR.OSCSEL is done.

29.4.1 Embedded 64 kHz (typical) RC Oscillator

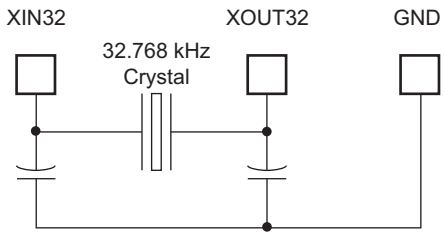
By default, the embedded 64 kHz (typical) RC oscillator is enabled and selected. The user has to take into account the possible drifts of this oscillator. Refer to the section “DC Characteristics”.

29.4.2 32.768 kHz Crystal Oscillator

The Clock Generator integrates a low-power 32.768 kHz crystal oscillator. To use this oscillator, the XIN32 and XOUT32 pins must be connected to a 32.768 kHz crystal. Two external capacitors must be wired as shown in Figure 29-2. More details are given in the section “DC Characteristics”.

Note that the user is not obliged to use the 32.768 kHz crystal oscillator and can use the 64 kHz (typical) RC oscillator instead.

Figure 29-2. Typical 32.768 kHz Crystal Oscillator Connection



The 32.768 kHz crystal oscillator provides a more accurate frequency than the 64 kHz (typical) RC oscillator.

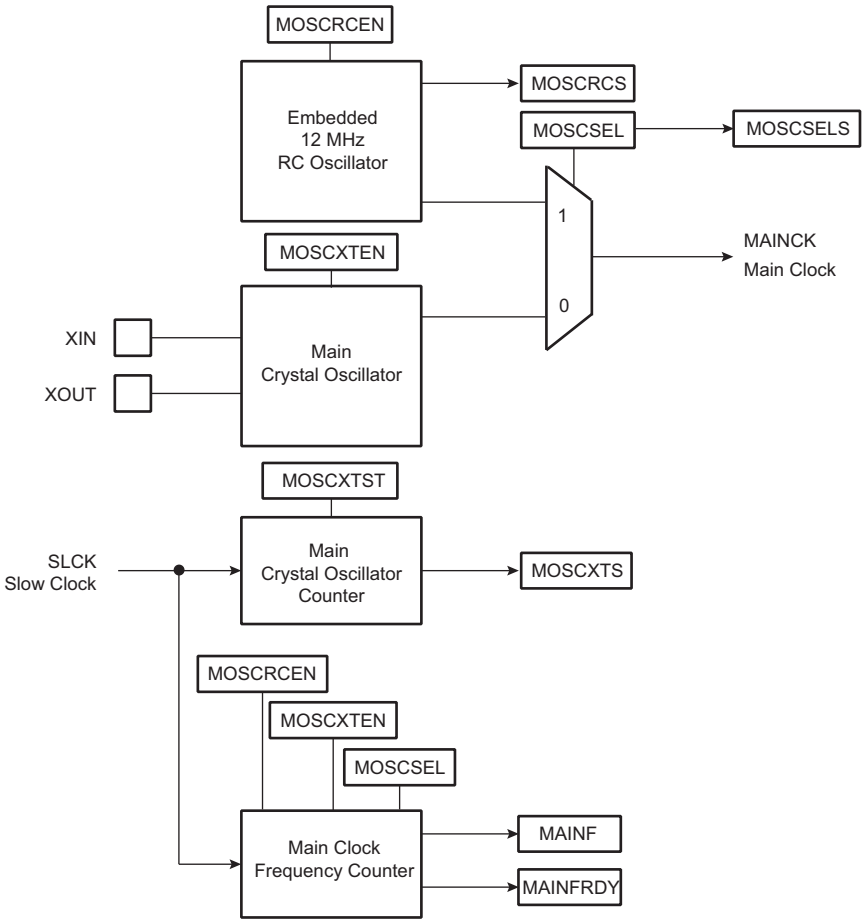
To select the 32.768 kHz crystal oscillator as the source of the slow clock, the bit SCKC_CR.OSCSEL must be set. This results in a sequence which enables the 32.768 kHz crystal oscillator. The switch of the slow clock source is glitch-free.

29.5 Main Clock

The main clock has two sources:

- a 12 MHz RC oscillator with a fast startup time and used at startup
- a 8 to 24 MHz crystal oscillator which can be bypassed

Figure 29-3. Main Clock Block Diagram



29.5.1 12 MHz RC Oscillator

After reset, the 12 MHz RC oscillator is enabled and it is selected as the source of MCK. MCK is the default clock selected to start up the system.

Refer to the table “DC Characteristics”.

The software can disable or enable the 12 MHz RC oscillator with the MOSCRCEN bit in CKGR_MOR.

When disabling the main clock by clearing the MOSCRCEN bit in CKGR_MOR, the MOSCRCS bit in PMC_SR is automatically cleared, indicating the main clock is off.

Setting the MOSCRCS bit in the Power Management Controller Interrupt Enable Register (PMC_IER) can trigger an interrupt to the processor.

29.5.2 12 MHz RC Oscillator Clock Frequency Adjustment

It is possible for the user to adjust the 12 MHz RC oscillator frequency through the PMC Oscillator Calibration Register (PMC_OCR). By default, the SEL bit in PMC_OCR is low, so the RC oscillator is driven with fuse calibration bits which are programmed during the chip production.

The user can adjust the trimming of the 12 MHz RC oscillator through PMC_OCR in order to obtain more accurate frequency (to compensate derating factors such as temperature and voltage).

In order to calibrate the 12 MHz oscillator frequency, SEL must be set to 1 and a correct frequency value must be configured in the CAL field.

It is possible to restart, at anytime, a measurement of the frequency of the selected clock by means of the RCMEAS bit in the Clock Generator Main Clock Frequency Register (CKGR_MCFR). Thus, when MAINFRDY flag reads 1, another read access on CKGR_MCFR provides an image of the frequency of the main clock on MAINF field. The software can calculate the error with an expected frequency and correct the CAL field in PMC_OCR accordingly. This may be used to compensate frequency drift due to derating factors such as temperature and/or voltage.

29.5.3 8 to 24 MHz Crystal Oscillator

After reset, the 8 to 24 MHz crystal oscillator is disabled and is not selected as the source of MAINCK.

As the source of MAINCK, the 8 to 24 MHz crystal oscillator provides an accurate frequency. The software enables or disables this oscillator in order to reduce power consumption via CKGR_MOR.MOSCXTEN.

When disabling this oscillator by clearing the CKGR_MOR.MOSCXTEN bit, the PMC_SR.MOSCXTS bit is automatically cleared, indicating the 8 to 24 MHz crystal oscillator is off.

When enabling this oscillator, the user must initiate the startup time counter. This startup time depends on the characteristics of the external device connected to this oscillator. Refer to the section “Electrical Characteristics” for the startup time.

When CKGR_MOR.MOSCXTEN and CKGR_MOR.MOSCXTST are written to enable this oscillator, the PMC_SR.MOSCXTS bit is cleared and the counter starts counting down on the slow clock divided by 8 from the MOSCXTST value. When the counter reaches 0, the PMC_SR.MOSCXTS is set, indicating that the 8 to 24 MHz crystal oscillator is stabilized. Setting PMC_IMR.MOSCXTS triggers an interrupt to the processor.

29.5.4 Main Clock Source Selection

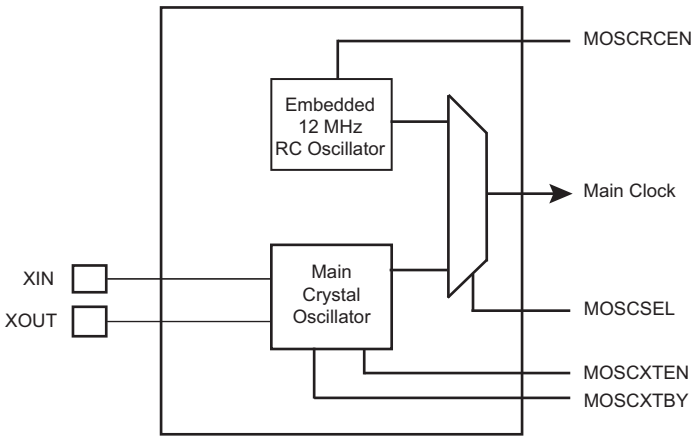
The main clock is generated by the 8 to 24 MHz crystal oscillator, an XRCGB crystal resonator or by the embedded 12 MHz RC oscillator.

The selection is made by writing CKGR_MOR.MOSCSEL. The switch of the main clock source is glitch-free, so there is no need to run out of SLCK or PLLACK in order to change the selection. PMC_SR.MOSCSELS indicates when the switch sequence is done.

Setting PMC_IMR.MOSCSELS triggers an interrupt to the processor.

The 8 to 24 MHz crystal oscillator can be bypassed by setting the MOSCXTBY bit in CKGR_MOR to accept an external main clock on XIN (refer to Section 29.5.5 “Bypassing the 8 to 24 MHz Crystal Oscillator”).

Figure 29-4. Main Clock Source Selection



MOSCRCEN, MOSCXTEN, MOSCSEL and MOSCXTBY bits are located in the PMC Clock Generator Main Oscillator Register (CKGR_MOR).

After a VDDBU power-on reset, the default configuration is MOSCRCEN = 1, MOSCXTEN = 0 and MOSCSEL = 0, allowing the 12 MHz RC oscillator to start as Main clock.

29.5.5 Bypassing the 8 to 24 MHz Crystal Oscillator

Prior to bypassing the 8 to 24 MHz crystal oscillator, the external clock frequency provided on the XIN pin must be stable and within the values specified in the XIN Clock characteristics in the section “Electrical Characteristics”.

The sequence to bypass the crystal oscillator is the following:

1. Ensure that an external clock is connected on XIN.
2. Enable the bypass by setting CKGR_MOR.MOSCXTBY.
3. Disable the 8 to 24 MHz crystal oscillator by clearing CKGR_MOR.MOSCXTEN.

29.5.6 Main Clock Frequency Counter

The frequency counter is managed by CKGR_MCFR.

During the measurement period, the frequency counter increments at the speed of the clock defined by the bit CKGR_MCFR.CCSS.

A measurement is started in the following cases:

- When the RCMEAS bit of CKGR_MCFR is written to 1.
- When the 12 MHz RC oscillator is selected as the source of the main clock and when this oscillator becomes stable (i.e., when the MOSCRCS bit is set)
- When the 8 to 24 MHz crystal oscillator is selected as the source of the main clock and when this oscillator becomes stable (i.e., when the MOSCXTS bit is set)
- When the main clock source selection is modified

The measurement period ends at the 16th falling edge of the slow clock, the MAINFRDY bit in CKGR_MCFR is set and the counter stops counting. Its value can be read in the MAINF field of CKGR_MCFR and gives the number of main clock cycles during 16 periods of slow clock, so that the frequency of the 12 MHz RC oscillator or the crystal oscillator can be determined.

29.5.7 Switching Main Clock Between the RC Oscillator and the Crystal Oscillator

When switching the source of the main clock between the RC oscillator and the crystal oscillator, both oscillators must be enabled. After completion of the switch, the unused oscillator can be disabled.

If switching to the crystal oscillator, a check must be carried out to ensure that the oscillator is present and that its frequency is valid. Follow the sequence below:

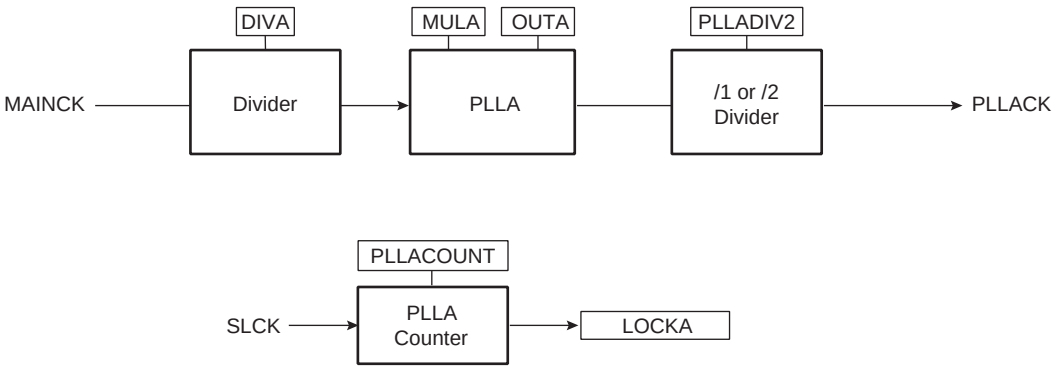
1. Enable the crystal oscillator by setting CKGR_MOR.MOSCXTEN. Configure the CKGR_MOR. MOSCXTST field with the crystal oscillator startup time as defined in the section “Electrical Characteristics”.
2. Wait for PMC_SR.MOSCXTS flag to rise, indicating the end of a startup period of the crystal oscillator.
3. Select the crystal oscillator as the source clock of the frequency meter by setting CKGR_MCFR.CCSS
4. Initiate a frequency measurement by setting CKGR_MCFR.RCMEAS.
5. Read CKGR_MCFR.MAINFRDY until its value equals 1.
6. Read CKGR_MCFR.MAINF and compute the value of the crystal frequency.
   – If the MAINF value is valid, the main clock can be switched to the crystal oscillator.

29.6 Divider and PLLA Block

The PLLA embeds an input divider to increase the accuracy of the resulting clock signals. However, the user must respect the PLLA minimum input frequency when programming the divider.

Figure 29-5 shows the block diagram of the divider and PLLA block.

Figure 29-5. Divider and PLLA Block Diagram



29.6.1 Divider and Phase Lock Loop Programming

The divider can be set between 1 and 255 in steps of 1. When a divider field (DIV) is cleared, the output of the corresponding divider and the PLL output is a continuous signal at level 0. On reset, each DIV field is cleared, thus the corresponding PLL input clock is stuck at 0.

The PLLA allows multiplication of the divider’s outputs. The PLLA clock signal has a frequency that depends on the respective source signal frequency and on the parameters DIVA and MULA. The factor applied to the source signal frequency is (MULA + 1)/DIVA. When MULA is written to 0, the PLLA is disabled and its power consumption is saved. Re-enabling the PLLA can be performed by writing a value higher than 0 in the MUL field.

Whenever the PLLA is re-enabled or one of its parameters is changed, the LOCKA bit in PMC_SR is automatically cleared. The values written in the PLLACOUNT field in CKGR_PLLAR are loaded in the PLLA counter. The PLLA counter then decrements at the speed of the slow clock until it reaches 0. At this time, the LOCK bit is set in PMC_SR and can trigger an interrupt to the processor. The user has to load the number of slow clock cycles required to cover the PLLA transient time into the PLLACOUNT field.

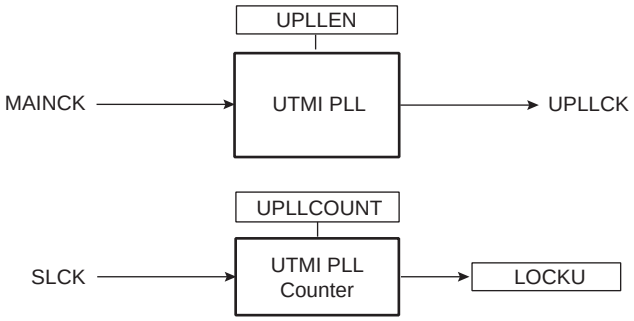
The PLLA clock must be divided by 2 by writing the PLLADIV2 bit in PMC_MCKR, if the ratio between Processor Clock (PCK) and MCK is 3 (MDIV = 3).

29.7 UTMI Phase Lock Loop Programming

The source clock of the UTMI PLL is the main clock (MAINCK). The MAINCK must select the fast crystal oscillator to meet the frequency accuracy required by USB.

The crystal frequency selection among 12, 16 or 24 MHz must be configured to the correct value in the field SFR_UTMICKTRIM.FREQ, in order to apply the correct multiplier, x40, x30 or x20, respectively.

Figure 29-6. UTMI PLL Block Diagram



Whenever the UTMI PLL is enabled by writing UPLLEN in CKGR_UCKR, the LOCKU bit in PMC_SR is automatically cleared. The values written in the PLLCOUNT field in CKGR_UCKR are loaded in the UTMI PLL counter. The UTMI PLL counter then decrements at the speed of the slow clock divided by 8 until it reaches 0. At this time, the LOCKU bit is set in PMC_SR and can trigger an interrupt to the processor. The user has to load the number of slow clock cycles required to cover the UTMI PLL transient time into the PLLCOUNT field.

29.8 Audio PLL

The Audio PLL is a high-resolution fractional-N digital PLL specifically designed for low jitter operation.

In audio applications, the CLK_AUDIO output pin typically serves as the master clock frequency generator for external components such as Audio DAC, Audio ADCs, or Audio Codecs, thus saving one crystal on the board.

The reference clock of the Audio PLL is the fast crystal oscillator. The PLL core operating frequency is defined as:

$$f_{AUDIOCORECLK} = f_{ref}\left(ND + 1 + \frac{FRACR}{2^{22}}\right)$$

where f_{ref} is the frequency of the main crystal oscillator. Refer to the section “PLL Characteristics” for the limits of $f_{AUDIOCORECLK}$.

The PLL core features two post-dividers enabling the generation of two output clock signals, AUDIOPLLCLK and AUDIOPINCLK. AUDIOPLLCLK is dedicated to the PMC and can be sent to the GCLK input of peripherals or to the Programmable Clock Outputs PCKx. AUDIOPINCLK is dedicated to driving the external audio pin CLK_AUDIO.

The AUDIOPLLCLK frequency is defined by the following formula:

$$f_{AUDIOPLLCLK} = \frac{f_{AUDIOCORECLK}}{(QDPMC + 1)}$$

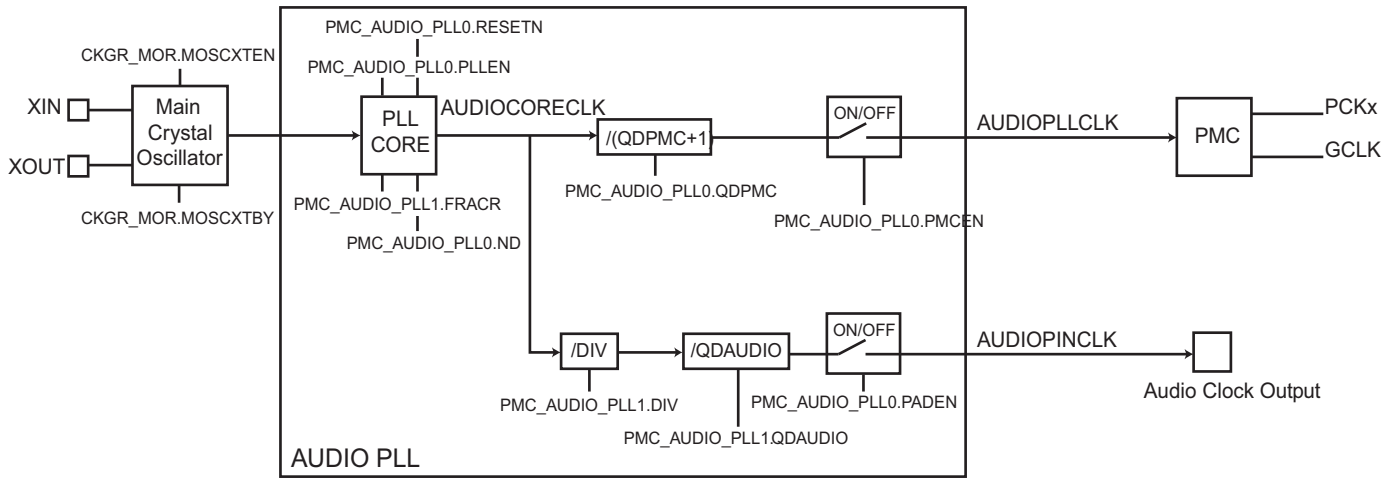
The AUDIOPINCLK frequency is defined by the following formula:

$$f_{AUDIOPINCLK} = \frac{f_{AUDIOCORECLK}}{(DIV \times QDAUDIO)}$$

The typical programming sequence of the audio PLL is the following:

1. Disable the PLL by writing 0 in PMC_AUDIO_PLL0.PLLEN and 0 in PMC_AUDIO_PLL0.RESETN.
2. Release the reset of the PLL by writing 1 in PMC_AUDIO_PLL0.RESETN.
3. Configure the PLL frequency by writing PMC_AUDIO_PLL0.QDPMC, PMC_AUDIO_PLL1.QDAUDIO, PMC_AUDIO_PLL1.DIV, PMC_AUDIO_PLL1.FRACR and PMC_AUDIO_PLL0.ND fields. ND and FRACR must be configured so as to set AUDIOCORECLK frequency in its authorized range. Refer to the “Electrical Characteristics” section.
4. Enable the PLL by writing 1 in PMC_AUDIO_PLL0.PLLEN, PMC_AUDIO_PLL0.PMCEN and PMC_AUDIO_PLL0.PADEN.
5. Wait for the startup time of this PLL. Refer to the “Electrical Characteristics” section.
6. If needed, ND or FRACR can be adjusted at any time. The typical frequency settling time of this PLL is indicated in the “Electrical Characteristics” section.

Figure 29-7. Audio PLL



30. Power Management Controller (PMC)

30.1 Description

The Power Management Controller (PMC) optimizes power consumption by controlling all system and user peripheral clocks. The PMC enables/disables the clock inputs to many of the peripherals and the Core.

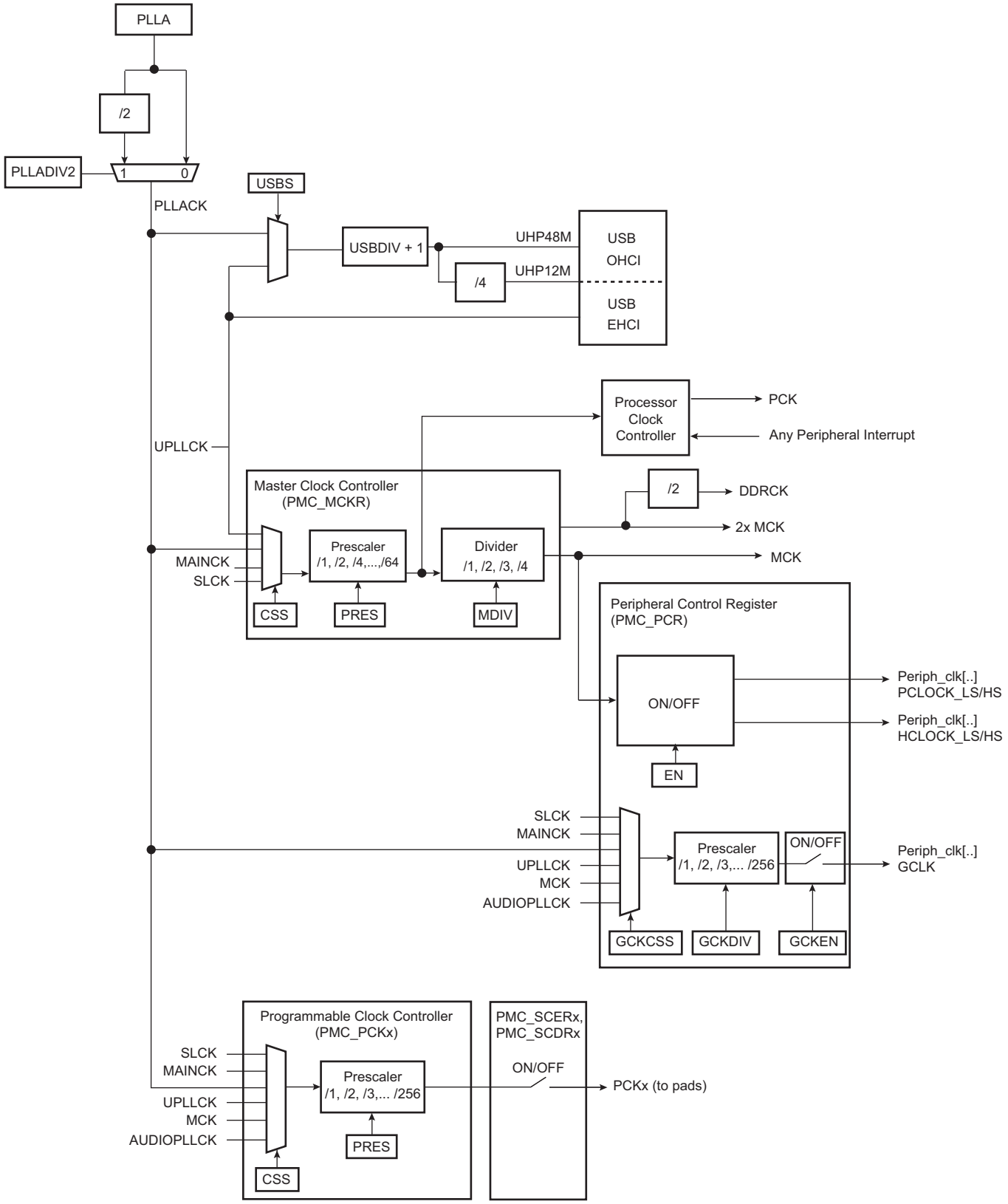
30.2 Embedded Characteristics

The Power Management Controller provides the following clocks:

- Master Clock (MCK)—programmable from a few hundred Hz to the maximum operating frequency of the device. It is available to the modules running permanently.
- Processor Clock (PCK)—must be switched off when processor is entering Idle mode
- USB Device HS Clock (UDPCK)
- H64MX and H32MX Matrixes Clocks
- Peripheral Clocks (typically MCK)—provided to the embedded peripherals (USART, SSC, SPI, TWI, TC, HSMCI, etc.) and independently controllable. In order to reduce the number of clock names in a product, the Peripheral Clocks are named MCK.
- Programmable Clock Outputs can be selected from the clocks provided by the clock generator and driven on the PCKx pins.
- Generic Clock for peripherals that can accept a second clock source
- Asynchronous partial wakeup (SleepWalking) for FLEXCOMx, SPIx, TWIx, UARTx and ADC

30.3 Block Diagram

Figure 30-1. General Clock Block Diagram



30.4 Master Clock Controller

The Master Clock Controller provides selection and division of the Master Clock (MCK). MCK is the source clock of the peripheral clocks.

The Master Clock is selected from one of the clocks provided by the Clock Generator. Selecting the slow clock provides a slow clock signal to the whole device. Selecting the main clock saves power consumption of the PLLs.

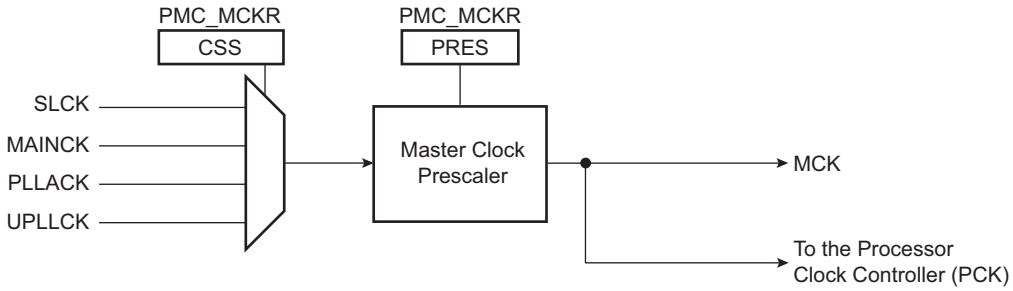
The Master Clock Controller is made up of a clock selector and a prescaler. It also contains a Master Clock divider which allows the processor clock to be faster than the Master Clock.

The Master Clock selection is made by writing the CSS (Clock Source Selection) field in PMC_MCKR (Master Clock Register). The prescaler supports the division by a power of 2 of the selected clock between 1 and 64, and the division by 6. The PRES field in PMC_MCKR programs the prescaler.

Note: It is forbidden to modify MDIV and CSS at the same access. Each field must be modified separately with a wait for MCKRDY flag between the first field modification and the second field modification.

Each time PMC_MCKR is written to define a new Master Clock, the MCKRDY bit is cleared in PMC_SR. It reads 0 until the Master Clock is established. Then, the MCKRDY bit is set and can trigger an interrupt to the processor. This feature is useful when switching from a high-speed clock to a lower one to inform the software when the change is actually done.

Figure 30-2. Master Clock Controller



30.5 Processor Clock Controller

The PMC features a Processor Clock (PCK) Controller that implements the processor Idle mode.

The processor clock can be disabled by executing the WFI (WaitForInterrupt) processor instruction or the WFE (WaitForEvent) processor instruction while the LPM bit is at 0 in the PMC Fast Startup Mode Register (PMC_FSMR).

The processor clock can be disabled by writing the PMC System Clock Disable Register (PMC_SCDR). The status of this clock (at least for debug purposes) can be read in the PMC System Clock Status Register (PMC_SCSR).

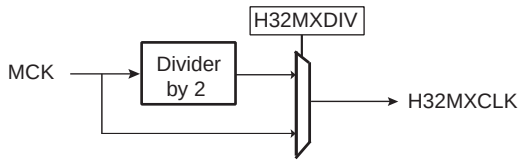
The processor clock is enabled after a reset and is automatically re-enabled by any enabled interrupt. The processor Idle mode is achieved by disabling the processor clock, which is automatically re-enabled by any enabled fast or normal interrupt, or by the reset of the product.

When processor Idle mode is entered, the current instruction is finished before the clock is stopped, but this does not prevent data transfers from other masters of the system bus.

30.6 Matrix Clock Controller

The AXI Matrix and H64MX 64-bit Matrix clocks are MCK. The H32MX 32-bit matrix clock is to be configured as MCK if MCK does not exceed 83 MHz (see “Master Clock Characteristics” in Electrical Characteristics section), else it is to be configured as MCK/2. Selection is done with the H32MXDIV bit in “PMC Master Clock Register” .

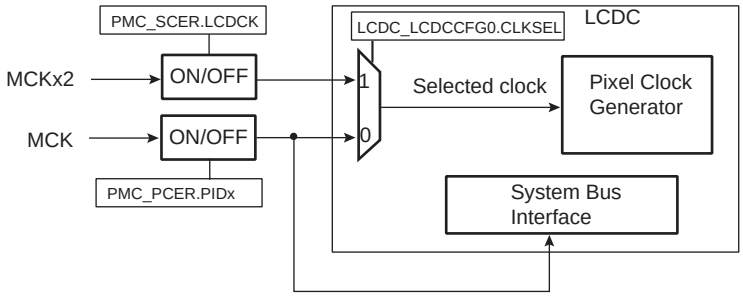
Figure 30-3. H32MX 32-bit Matrix Clock Configuration



30.7 LCDC Clock Controller

In order to have more flexibility on the pixel clock, the LCDC can use MCK or MCKx2, if LCDCK is set in the PMC System Clock Enable Register (PMC_SCER).

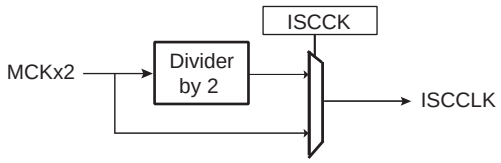
Figure 30-4. LCDCLK Clock Configuration



30.8 ISC Clock Controller

In order to have more flexibility on the pixel clock, the ISC can use MCK or MCKx2, if ISCCK is set in the PMC System Clock Enable Register (PMC_SCER).

Figure 30-5. ISCCLK Clock Configuration



30.9 USB Device and Host Clocks

The USB Device and Host High Speed ports (UDPHS and UHPHS) clocks are enabled by the corresponding PIDx bits in PMC_PCERx. To save power on this peripheral when they are not used, the user can set these bits in PMC_PCDR. Corresponding PIDx bits in PMC_PCSR give the status of these clocks.

The PMC also provides the clocks UHP48M and UHP12M to the USB Host OHCI. The USB Host OHCI clocks are controlled by the UHP bit in PMC_SCER. To save power on this peripheral when they are not used, the user can set the UHP bit in PMC_SCDR. The UHP bit in PMC_SCSR gives the status of this clock. The USB host OHCI requires both the 12/48 MHz signal and the Master Clock. The USBDIV field in PMC_USB register is to be programmed to 9 (division by 10) for normal operations.

To further reduce power consumption the user can stop UTMI PLL, in this case USB high-speed operations are not possible. Nevertheless, as the USB OHCI Input clock can be selected with USBS bit (PLLA or UTMI PLL) in PMC_USB register, OHCI full-speed operation remains possible.

The user must program the USB OHCI Input Clock and the USBDIV divider in the PMC_USB register to generate a 48 MHz and a 12 MHz signal with an accuracy of ±0.25%.

The USB clock input is to be defined according to main oscillator via the FREQ field. This field is defined in the UTMI Clock Trimming Register (SFR_UTMICKTRIM). Refer to the section “Special Function Registers (SFR)”. This input clock can be 12, 16, or 24 MHz.

30.10 DDR2/LPDDR/LPDDR2 Clock

The PMC controls the clocks of the DDR memory.

The DDR clock can be enabled and disabled with the DDRCK bit respectively in PMC_SCER and PMC_SDER. At reset, the DDR clock is disabled to reduce power consumption.

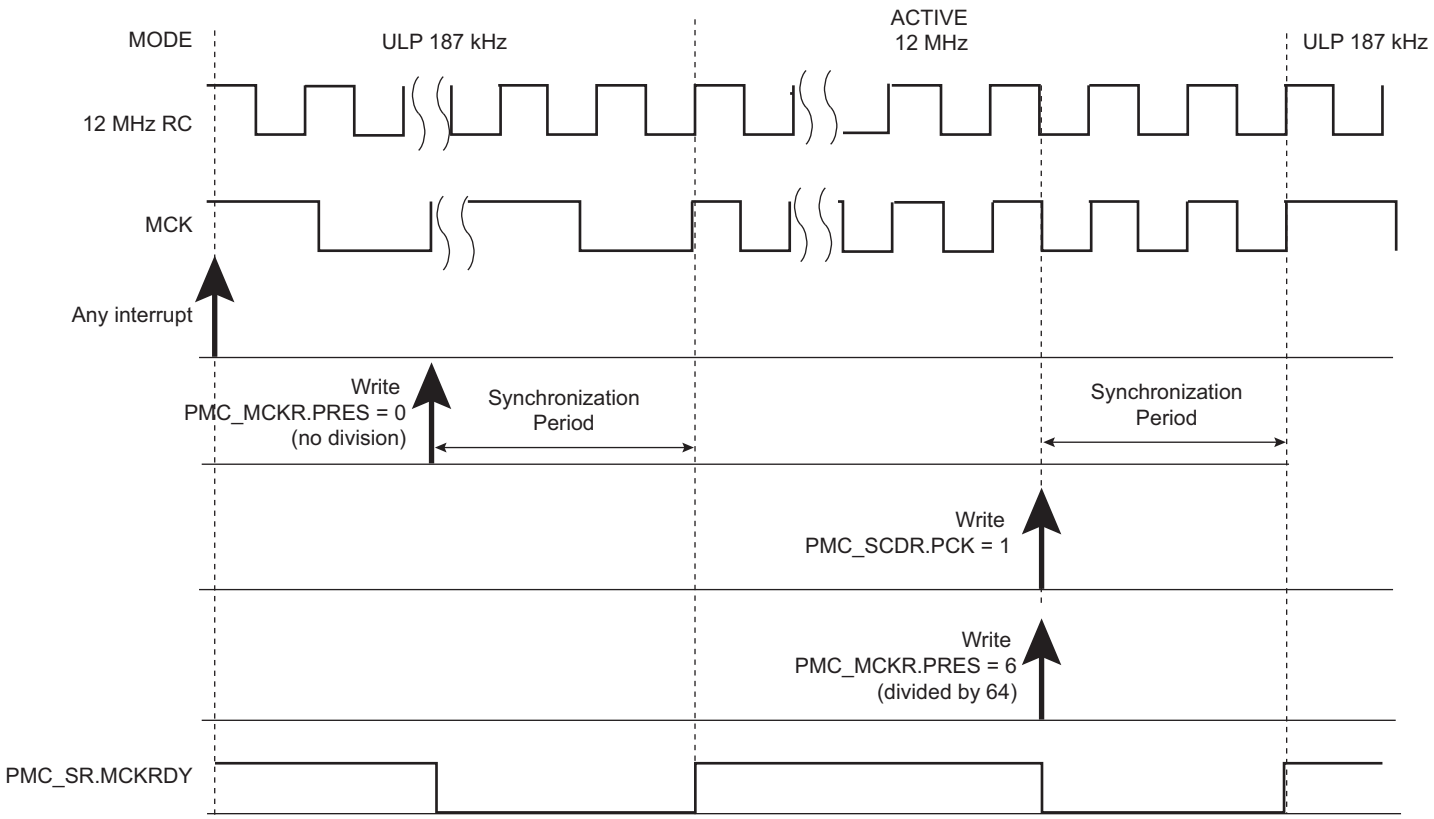
In case MDIV = 0 (PCK = MCK), the DDRCK clock is not available.

To reduce PLLA power consumption, the user can choose UPLLCK as an input clock for the system. In this case the DDR Controller can drive LPDDR or LPDDR2 at up to 120 MHz.

30.11 Fast Startup from Ultra Low-power (ULP) Mode 0

In Ultra Low-power (ULP) mode 0, the main clock (MAINCK) must be running, thus either the 12 MHz crystal oscillator or the Fast RC oscillator must be enabled. The lowest power consumption that can be achieved in ULP Mode 0, can be obtained when dividing the selected oscillator frequency by 64 by writing the field PMC_MCKR.PRES to 6. Any interrupt exits the system from ULP Mode. The software must write PMC_MCKR.PRES to 1 to provide MCK with the fastest clock. If the PLL is used, the startup procedure must be done prior to write PMC_MCKR.PRES to 1. Figure 30-6 illustrates an example of startup phase from ULP Mode 0 without use of PLL.

Figure 30-6. Fast Startup from Ultra Low-Power Mode 0



Warning: The duration of the WKUPx pins active level must be greater than four MAINCK cycles.

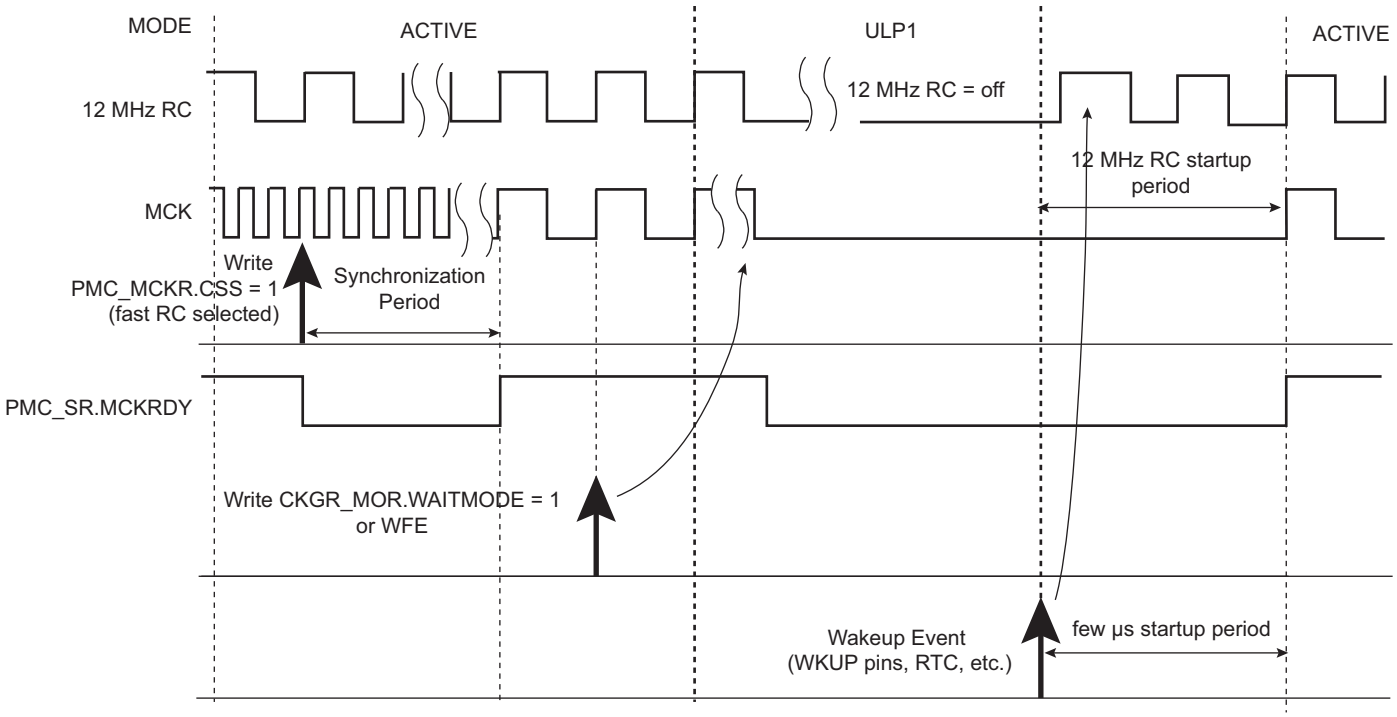
30.12 Fast Startup from Ultra Low-power (ULP) Mode 1

The device allows the processor to restart in less than 10 µs while the device exits Ultra Low-power (ULP) mode 1 only if the C-code function managing the ULP mode 1 entry and exit is linked to and executed from on-chip SRAM.

Prior to instructing the device to enter ULP mode 1, the RC oscillator must be selected as the master clock source (the field PMC_MCKR.CSS must be written to 1, wait for the PMC_SR.MCKRDY bit to be set) and the internal sources of wakeup must be cleared. It must be verified that none of the enabled external wakeup inputs (WKUP) hold an active polarity.

The system enters ULP mode 1 either by setting the WAITMODE bit in CKGR_MOR, or by executing the WaitForEvent (WFE) instruction of the processor while the PMC_FSMR.LPM bit is at 1. Immediately after setting the WAITMODE bit or using the WFE instruction, wait for the PMC_SR.MCKRDY bit to be set. See details in Figure 30-7.

Figure 30-7. Fast Startup from ULP Mode 1



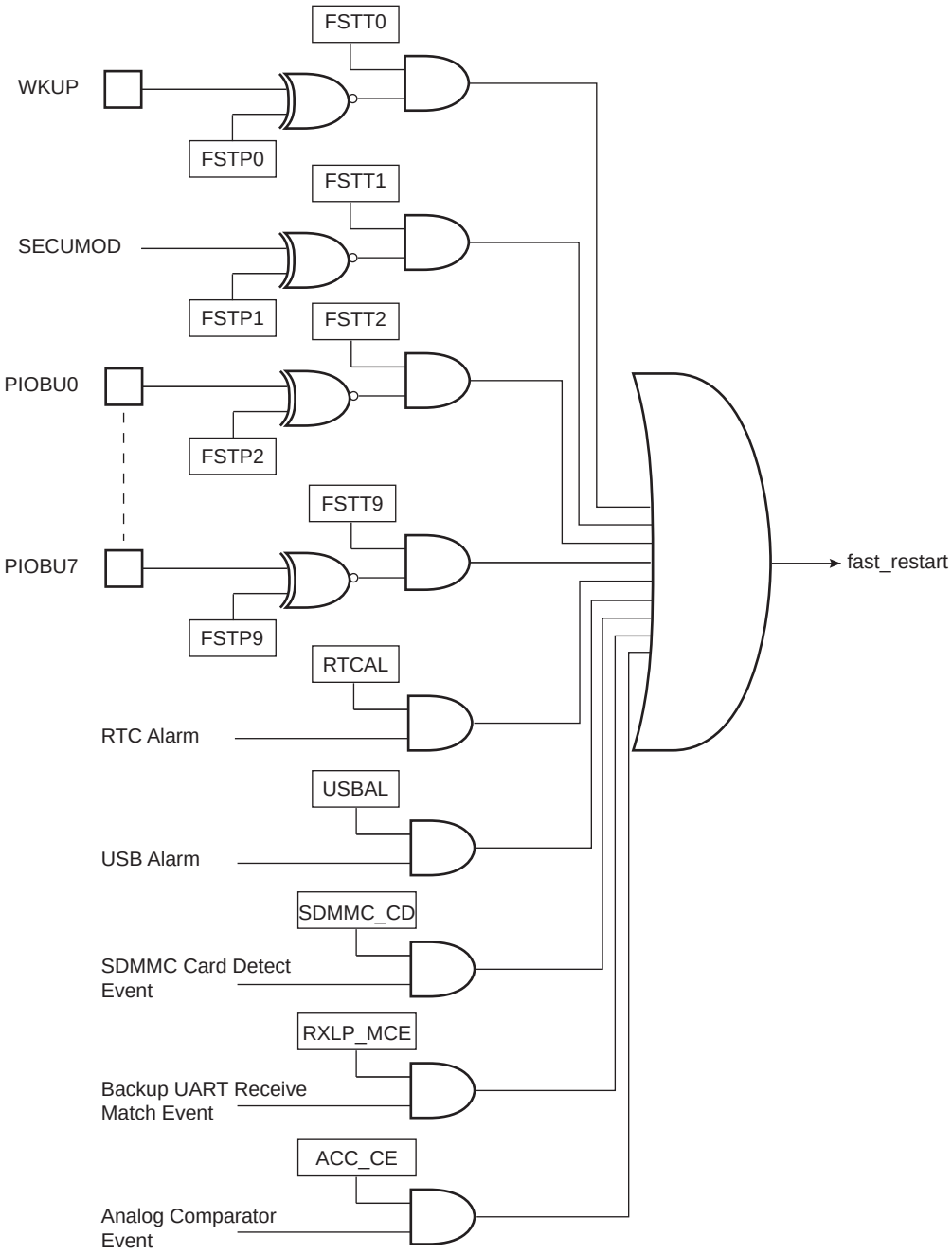
A fast startup is enabled upon any of the following events:

- detection of a programmed level on one of the nine wakeup inputs (WKUP, PIOBUx)
- an active alarm from the RTC
- a resume from the USB Controller
- SDMMC card detect
- backup UART (RXLP) received character comparison match
- an analog comparison (ACC)
- any SleepWalking event coming from TWI, FLEXCOMx, SPI, ADC

The polarity of the nine wakeup inputs is programmable by writing the PMC Fast Startup Polarity Register (PMC_FSPR). All the fast restart event sources except SleepWalking can be individually enabled/disabled by writing in PMC_FSMR. SleepWalking events can be individually enabled/disabled by writing in PMC_SLPWK_ERx/PMC_SLPWK_DRx (see Section 30.14 “Asynchronous Partial Wakeup (SleepWalking)”).

The fast startup circuitry, as shown in Figure 30-8, is fully asynchronous and provides a fast startup signal to the PMC. As soon as the fast startup signal is asserted, the embedded 12 MHz RC oscillator restarts automatically.

Figure 30-8. Fast Startup Circuitry



The PMC user interface does not provide the source of the fast startup, but the user can recover this information by reading the PIO Controller and the status registers of the RTC, ACC, RXLP, and USB Controller.

30.13 Peripheral Clock Controller

The PMC controls the clocks of each embedded peripheral by means of the Peripheral Clock Controller. The user can individually enable and disable the clock on the peripherals and select a division factor from MCK. The user can also select the source, the division ratio, enable and disable the generic clock (GCLK) of the peripherals. This is done in the Peripheral Control Register (PMC_PCR).

When a peripheral clock is disabled, the clock is immediately stopped. The peripheral clocks are automatically disabled after a reset.

In order to stop a peripheral, it is recommended that the system software wait until the peripheral has executed its last programmed operation before disabling the clock. This is to avoid data corruption or erroneous behavior of the system.

The value written in the PID field in PMC_PCR is the Peripheral Identifier defined at the product level (refer to section “Peripheral Identifiers”). Generally, the field value corresponds to the interrupt source number assigned to the peripheral.

30.14 Asynchronous Partial Wakeup (SleepWalking)

30.14.1 Description

The asynchronous partial wakeup (SleepWalking) wakes up a peripheral in a fully asynchronous way when activity is detected on the communication line. Moreover, under some user configurable conditions, the asynchronous partial wakeup can trigger an exit of the system from ULP mode 1 (full system wakeup).

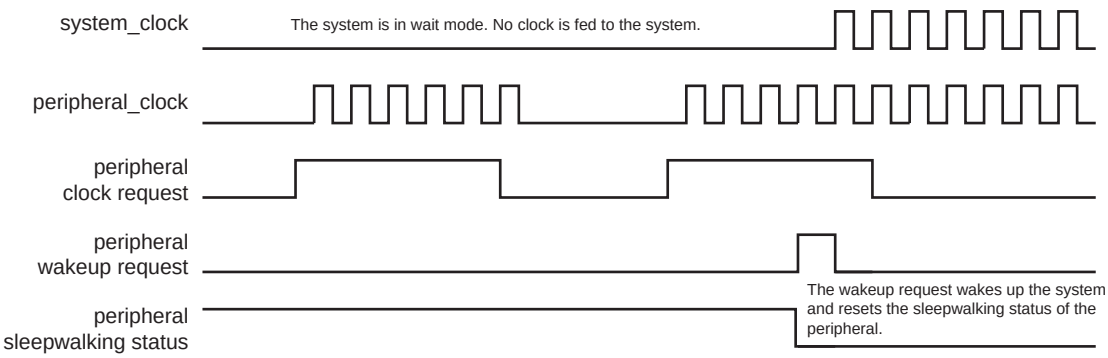
The asynchronous partial wakeup function automatically manages the peripheral clock. It improves the overall power consumption of the system by clocking peripherals only when needed.

Only the following peripherals can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The peripheral selected for asynchronous partial wakeup must be first configured so that its clock is enabled by setting the appropriate PIDx bit in PMC_PCERx.

When the system is in ULP mode 1, all clocks of the system (except SLCK) are stopped. When an asynchronous clock request from a peripheral occurs, the PMC partially wakes up the system to feed the clock only to this peripheral. The rest of the system is not fed with the clock, thus optimizing power consumption. Finally, depending on user-configurable conditions, the peripheral either wakes up the whole system if these conditions are met or stops the peripheral clock until the next clock request. If a wakeup request occurs, the Asynchronous Partial Wakeup mode is automatically disabled until the user instructs the PMC to enable asynchronous partial wakeup. This is done by setting PIDx in the PMC SleepWalking Enable Register (PMC_SLPWK_ER).

Figure 30-9. SleepWalking During Ultra Low-Power Mode 1

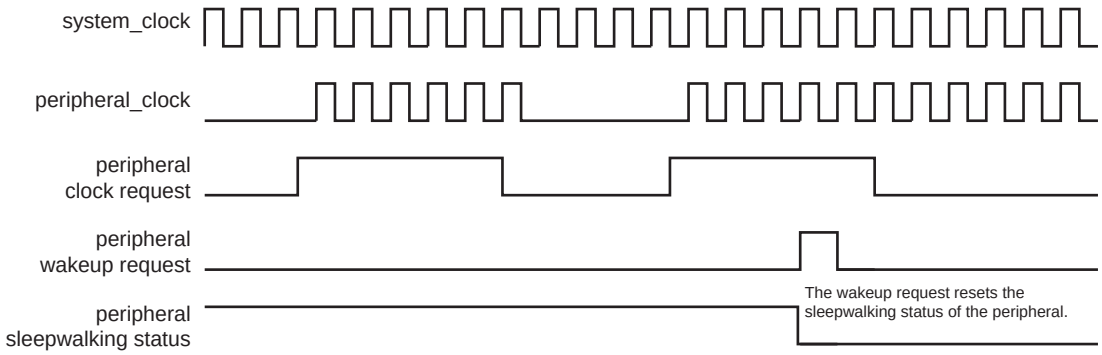


When the system is in Active mode, peripherals enabled for asynchronous partial wakeup have their respective clocks stopped until the peripherals request a clock. When a peripheral requests the clock, the PMC provides the clock without CPU intervention.

The triggering of the peripheral clock request depends on conditions which can be configured for each peripheral. If these conditions are met, the peripheral asserts a request to the PMC. The PMC disables the Asynchronous Partial Wakeup mode of the peripheral and provides the clock to the peripheral until the user instructs the PMC to re-enable partial wakeup on the peripheral. This is done by setting PIDx in PMC_SLPWK_ER.

If the conditions are not met, the peripheral clears the clock request and PMC stops the peripheral clock until the clock request is re-asserted by the peripheral.

Figure 30-10. SleepWalking During Active Mode



30.14.2 Configuration Procedure

Before configuring the asynchronous partial wakeup (SleepWalking) function of a peripheral, check that the peripheral clock is enabled (the PIDx bit in the PMC Peripheral Clock Status Register (PMC_PCSR) must be set).

The procedure to enable the asynchronous partial wakeup (SleepWalking) function of a peripheral is the following:

1. Check that the corresponding PIDx bit in the PMC SleepWalking Activity Status Register (PMC_SLPWK_ASR) is cleared. This ensures that the peripheral has no activity in progress.
2. Enable the asynchronous partial wakeup function of the peripheral by writing a one to the corresponding PIDx bit in PMC_SLPWK_ER.
3. Check that the corresponding PIDx bit in PMC_SLPWK_ASR is cleared. This ensures that no activity has started during the enable phase.
4. In PMC_SLPWK_ASR, if the corresponding PIDx bit is set, the asynchronous partial wakeup function must be immediately disabled by writing a one to the PIDx bit in the PMC SleepWalking Disable Register (PMC_SLPWK_DR). Wait for the end of peripheral activity before reinitializing the procedure.
   If the corresponding PIDx bit is cleared, then the peripheral clock is disabled and the system can now be placed in ULP mode 1.

Before entering ULP mode 1, check that the AIP bit in the PMC SleepWalking Activity In Progress Register (PMC_SLPWK_AIPR) is cleared. This ensures that none of the peripherals has any activity in progress.

Note: When asynchronous partial wakeup (SleepWalking) of a peripheral is enabled and the core is running (system not in ULP mode 1), the peripheral must not be accessed before a wakeup of the peripheral is performed.

30.15 Programmable Clock Controller

The PMC controls three signals to be outputs on external pins PCKx. Each signal can be independently programmed via the PMC Programmable Clock Register (PMC_PCKx).

PCKx can be independently selected between the Slow Clock (SLCK), the Master Clock (MAINCK), the PLLACK, the UTMI PLL output, the Main Clock and the AUDIO PLL (AUDIOPLLCLK) output by writing the CSS field in PMC_PCKx. Each output signal can also be divided by a factor between 1 and 256 by writing the PRES (Prescaler) field in PMC_PCKx.

Each output signal can be enabled and disabled by writing a 1 in the corresponding bit, PCKx of PMC_SCER and PMC_SCDR, respectively. The status of the active programmable output clocks are given in the PCKx bits of PMC_SCSR.

The status bit PCKRDYx in PMC_SR indicates that the Programmable Clock programmed in PMC_PCKx is ready.

As the Programmable Clock Controller does not implement glitch prevention when switching clocks, it is strongly recommended to disable the Programmable Clock before any configuration change and to re-enable it after the change is actually performed.

30.16 Generic Clock Controller

Some peripherals may need a second clock source that may be different from the system clock. This second clock is the generic clock (GCLK) and is managed by the PMC via PMC_PCR.

The source of each GCLK can be selected between the Slow Clock (SLCK), the Master Clock (MAINCK), the PLLACK, the UTMI PLL output, the Main Clock and the Audio PLL (AUDIOPLLCLK) output by writing the GCKCSS field in PMC_PCR. Each output signal can also be divided by a factor between 1 and 256 by writing the GCKDIV (Prescaler) field in PMC_PCR.

The status bit GCKRDY = 1 in PMC_SR indicates that all the generic clocks are actually what has been programmed in PMC_PCRx.

If the update of any generic clock is not ended, GCKRDY stays low.

As the Generic Clock Controller does not implement glitch prevention when switching clocks, it is strongly recommended to disable the GCLK before any configuration change and to re-enable it after the change is actually performed.

30.17 Main Clock Failure Detector

The clock failure detector monitors the 8 to 24 MHz crystal oscillator or ceramic resonator-based oscillator to identify a possible failure of this oscillator.

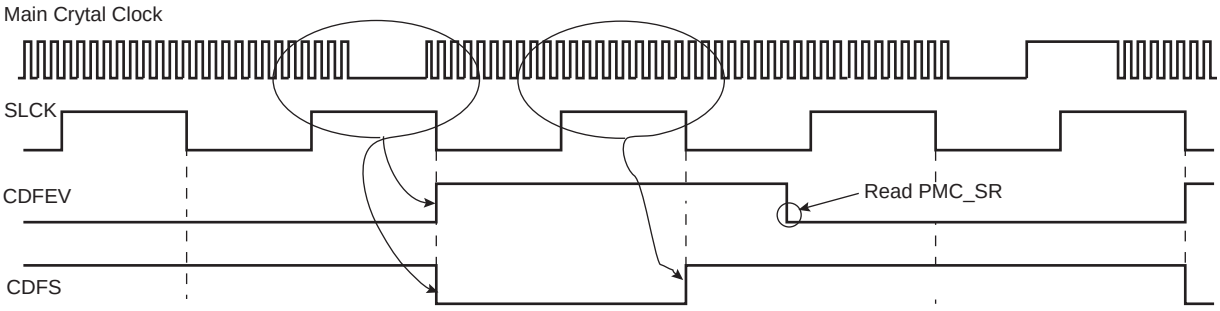
The clock failure detector can be enabled or disabled by bit CFDEN in CKGR_MOR. After a VDDCORE reset, the detector is disabled. However, if the oscillator is disabled (MOSCXTEN = 0), the detector is also disabled.

A failure is detected by means of a counter incrementing on the main oscillator clock edge and detection logic is triggered by the 32 kHz generated by the 64 kHz (typical) RC oscillator. This oscillator is automatically enabled when CFDEN = 1.

The counter is cleared when the 32 kHz generated by the 64 kHz (typical) RC oscillator clock signal is low, and enabled when the signal is high. Thus, the failure detection time is one RC oscillator period. If, during the high level period of the 32 kHz generated by the 64 kHz (typical) RC oscillator clock signal, less than eight 8 to 24 MHz crystal oscillator clock periods have been counted, then a failure is reported.

If a failure of the main clock is detected, bit PMC_SR.CFDEV indicates a failure event and generates an interrupt if the corresponding interrupt source is enabled. The interrupt remains active until a read occurs in PMC_SR. The user can know the status of the clock failure detection at any time by reading bit PMC_SR.CFDS.

Figure 30-11. Clock Failure Detection (Example)



Note: Ratio of clock periods is for illustration purposes only.

If the 8 to 24 MHz crystal oscillator or ceramic resonator-based oscillator is selected as the source clock of MAINCK (CKGR_MOR.MOSCSEL = 1), and if MCK source is PLLACK or UPLLCK (PMC_MCKR.CSS = 2 or 3), a clock failure detection automatically forces MAINCK to be the source clock for the master clock (MCK). Then, regardless of the PMC configuration, a clock failure detection automatically forces the 12 MHz RC oscillator to be the source clock for MAINCK. If this oscillator is disabled when a clock failure detection occurs, it is automatically re-enabled by the clock failure detection mechanism.

It takes two 32 kHz (typical) clock cycles to detect and switch from the 8 to 24 MHz crystal oscillator to the 12 MHz RC oscillator if the source master clock (MCK) is main clock (MAINCK), or three 32 kHz (typical) cycles if the source of MCK is PLLACK or UPLLCK.

A clock failure detection activates a fault output that is connected to the Pulse Width Modulator (PWM) Controller. With this connection, the PWM controller is able to force its outputs and to protect the driven device, if a clock failure is detected.

The user can know the status of the clock failure detector at any time by reading bit PMC_SR.FOS.

This fault output remains active until the defect is detected and until it is cleared by the bit FOCLR in the PMC Fault Output Clear Register (PMC_FOCR).

30.18 32.768 kHz Crystal Oscillator Frequency Monitor

The frequency of the 32.768 kHz crystal oscillator can be monitored by means of logic driven by the 12 MHz RC oscillator known as a reliable clock source. This function is enabled by configuring the XT32KFME bit of CKGR_MOR.

The error flag XT32KERR in PMC_SR is asserted when the 32.768 kHz crystal oscillator frequency is out of the ±10% nominal frequency value (i.e., 32.768 kHz). The error flag can be cleared only if the slow clock frequency monitoring is disabled.

The monitored clock frequency is declared invalid if at least four consecutive clock period measurement results are over the nominal period ±10%.

Due to the possible frequency variation of the embedded 12 MHz RC oscillator acting as reference clock for the monitor logic, any slow clock crystal frequency deviation over ±10% of the nominal frequency is systematically reported as an error by means of XT32KERR in PMC_SR. Between -1% and -10% and +1% and +10%, the error is not systematically reported.

Thus, only a crystal running at a 32.768 kHz frequency ensures that the error flag is not asserted. The permitted drift of the crystal is 10000 ppm (1%), which allows any standard crystal to be used.

The error flag can be defined as an interrupt source of the PMC by setting the XT32KERR bit of PMC_IER.

30.19 Programming Sequence

1. If the 8 to 24 MHz crystal oscillator is not required, PLL can be directly configured (begin with Step 6. or Step 7.) else this oscillator must be started (begin with Step 2.).
2. Enable the 8 to 24 MHz crystal oscillator by setting the MOSCXTEN bit in CKGR_MOR. The user can define a startup time. This can be achieved by writing a value in the MOSCXTST field in CKGR_MOR. Once this register has been correctly configured, the user must wait for the MOSCXTS field in PMC_SR to be set. This can be done either by polling MOSCXTS in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (MOSCXTS) has been enabled in PMC_IER.
3. Switch the MAINCK to the 8 to 24 MHz crystal oscillator by setting MOSCSEL in CKGR_MOR.
4. Wait for the MOSCSELS to be set in PMC_SR to ensure the switchover is complete.
5. Check the main clock frequency:

   The main clock frequency can be measured via the Main Clock Frequency Register (CKGR_MCFR).

   Read CKGR_MCFR until the MAINFRDY field is set, after which the user can read the field CKGR_MCFR.MAINF by performing an additional read. This provides the number of main clock cycles that have been counted during a period of 16 slow clock cycles.

   If MAINF = 0, switch the MAINCK to the 12 MHz RC oscillator by clearing CKGR_MOR.MOSCSEL. If MAINF ≠ 0, proceed to Step 6.
6. Setting PLLA and divider (if not required, proceed to Step 7.)

   All parameters needed to configure PLLA and the divider are located in CKGR_PLLAR.

   The DIVA field is used to control the divider itself. A value between 0 and 255 can be programmed. Divider output is divider input divided by DIVA parameter. By default, the DIVA field is cleared, which means that the divider and PLLA are turned off.

   The MULA field is the PLLA multiplier factor. This parameter can be programmed between 0 and 127. If MULA is cleared, PLLA is turned off, otherwise the PLLA output frequency is PLLA input frequency multiplied by (MULA + 1).

   The PLLACOUNT field specifies the number of slow clock cycles before LOCKA bit is set in PMC_SR after CKGR_PLLAR has been written.

   Once CKGR_PLLAR has been written, the user must wait for the LOCKA bit to be set in PMC_SR. This can be done either by polling LOCKA in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKA) has been enabled in PMC_IER. All parameters in CKGR_PLLAR can be programmed in a single write operation. If at some stage parameter MULA or DIVA is modified, LOCKA bit goes low to indicate that PLLA is not yet ready. When PLLA is locked, LOCKA is set again.

   The user must wait for the LOCKA bit to be set before using the PLLA output clock.
7. Setting Bias and High-speed PLL (UPLL) for UTMI

   The UTMI PLL is enabled by setting the UPLLEN field in CKGR_UCKR. The UTMI Bias must be enabled by setting the BIASEN field in CKGR_UCKR at the same time. In some cases, it may be preferable to define a startup time. This can be achieved by writing a value in the PLLCOUNT field in CKGR_UCKR.

   Once this register has been correctly configured, the user must wait for the LOCKU field in PMC_SR to be set. This can be done either by polling LOCKU in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (LOCKU) has been enabled in PMC_IER.
8. Selecting Master Clock and Processor Clock

   The Master Clock and the Processor Clock are configurable via PMC_MCKR.

   The CSS field is used to select the clock source of the Master Clock and Processor Clock dividers. By default, the selected clock source is the main clock.

The PRES field is used to define the Processor Clock and Master Clock prescaler. The user can choose between different values from 1 to 256). Prescaler output is the selected clock source frequency divided by the PRES value.

The MDIV field is used to define the Master Clock divider. It is possible to choose between different values (0, 1, 2, 3). The Master Clock output is Processor Clock frequency divided by 1, 2, 3 or 4, depending on the value programmed in MDIV.

The PMC PLLA Clock input must be divided by 2 by writing the PLLADIV2 bit if MDIV is set to 3.

By default, MDIV and PLLLADIV2 are cleared, which indicates that Processor Clock is equal to the Master Clock.

Once PMC_MCKR has been written, the user must wait for the MCKRDY bit to be set in PMC_SR. This can be done either by polling MCKRDY in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (MCKRDY) has been enabled in PMC_IER.

PMC_MCKR must not be programmed in a single write operation. The programming sequence for PMC_MCKR is the following:

If a new value for CSS field corresponds to PLL Clock,

1. Program PMC_MCKR.PRES field
2. Wait for PMC_SR.MCKRDY bit to be set
3. Program PMC_MCKR.MDIV field
4. Wait for PMC_SR.MCKRDY bit to be set
5. Program PMC_MCKR.CSS field
6. Wait for PMC_SR.MCKRDY bit to be set

If a new value for CSS field corresponds to main clock or slow clock,

1. Program PMC_MCKR.CSS field
2. Wait for PMC_SR.MCKRDY bit to be set
3. Program PMC_MCKR.PRES field
4. Wait for PMC_SR.MCKRDY bit to be set

If at some stage parameter CSS, MDIV or PRES is modified, the MCKRDY bit goes low to indicate that the Master Clock and the Processor Clock are not yet ready. The user must wait for the MCKRDY bit to be set again before using the Master and Processor Clocks.

Note: If PLLA clock was selected as the Master Clock and the user decides to modify it by writing in CKGR_PLLR, the MCKRDY flag goes low while PLL is unlocked. Once PLL is locked again, LOCKA goes high and MCKRDY is set. While PLL is unlocked, the Master Clock selection is automatically changed to slow clock. For further information, see Section 30.20.2 “Clock Switching Waveforms”.

Code Example:

```
write_register(PMC_MCKR,0x00000001)
wait (MCKRDY=1)
write_register(PMC_MCKR,0x00000011)
wait (MCKRDY=1)
```

The Master Clock is main clock divided by 2.

The Processor Clock is the Master Clock.

9. Selecting Programmable Clocks

Programmable clocks can be enabled and/or disabled via PMC_SCER and PMC_SCDR. 3 programmable clocks can be used. PMC_SCSR indicates which programmable clock is enabled. By default all programmable clocks are disabled.

PMC_PCKx registers are used to configure programmable clocks.

The PMC_PCKx.CSSfield selects the programmable clock divider source. Five clock options are available: main clock, slow clock, master clock, PLLACK, UPLLCK. The slow clock is the default clock source.

The PRES field is used to control the programmable clock prescaler. It is possible to choose among different values (from 1 to 256). Programmable clock output is prescaler input divided by PRES parameter. By default, the PRES value is cleared which means that PCKx is equal to slow clock.

Once the PMC_PCKx register has been configured, The corresponding programmable clock must be enabled and the user is constrained to wait for the PCKRDYx bit to be set in PMC_SR. This can be done either by polling PCKRDYx in PMC_SR or by waiting for the interrupt line to be raised if the associated interrupt source (PCKRDYx) has been enabled in PMC_IER. All parameters in PMC_PCKx can be programmed in a single write operation.

If the CSS and PRES parameters are to be modified, the corresponding programmable clock must be disabled first. The parameters can then be modified. Once this has been done, the user must re-enable the programmable clock and wait for the PCKRDYx bit to be set.

10. Enabling Peripheral Clocks

    Once all of the previous steps have been completed, the peripheral clocks can be enabled and/or disabled via PMC_PCERx and PMC_PCDRx.

30.20 Clock Switching Details

30.20.1 Master Clock Switching Timings

Table 30-1 and Table 30-2 give the worst case timings required for the Master Clock to switch from one selected clock to another one. This is in the event that the prescaler is deactivated. When the prescaler is activated, an additional time of 64 clock cycles of the new selected clock has to be added.

Table 30-1. Clock Switching Timings (Worst Case)

To	From		
	Main Clock	**SLCK**	**PLL Clock**
Main Clock	–	4 × SLCK + 2.5 × Main Clock	3 × PLL Clock + 4 × SLCK + 1 × Main Clock
SLCK	0.5 × Main Clock + 4.5 × SLCK	–	3 × PLL Clock + 5 × SLCK
PLL Clock	0.5 × Main Clock + 4 × SLCK + PLLCOUNT × SLCK + 2.5 × PLL Clock	2.5 × PLL Clock + 5 × SLCK + PLLCOUNT × SLCK	2.5 × PLL Clock + 4 × SLCK + PLLCOUNT × SLCK

Notes: 1. PLL designates either the PLLA or the UPLL Clock.
2. PLLCOUNT designates either PLLACOUNT or UPLLCOUNT.

Table 30-2. Clock Switching Timings Between Two PLLs (Worst Case)

To	From	
	PLLA Clock	UPLL Clock
PLLA Clock	2.5 × PLLA Clock + 4 × SLCK + PLLACOUNT × SLCK	3 × PLLA Clock + 4 × SLCK + 1.5 × PLLA Clock
UPLL Clock	3 × UPLL Clock + 4 × SLCK + 1.5 × UPLL Clock	2.5 × UPLL Clock + 4 × SLCK + UPLLCOUNT × SLCK

30.20.2 Clock Switching Waveforms

Figure 30-12. Switch Master Clock from Slow Clock to PLL Clock

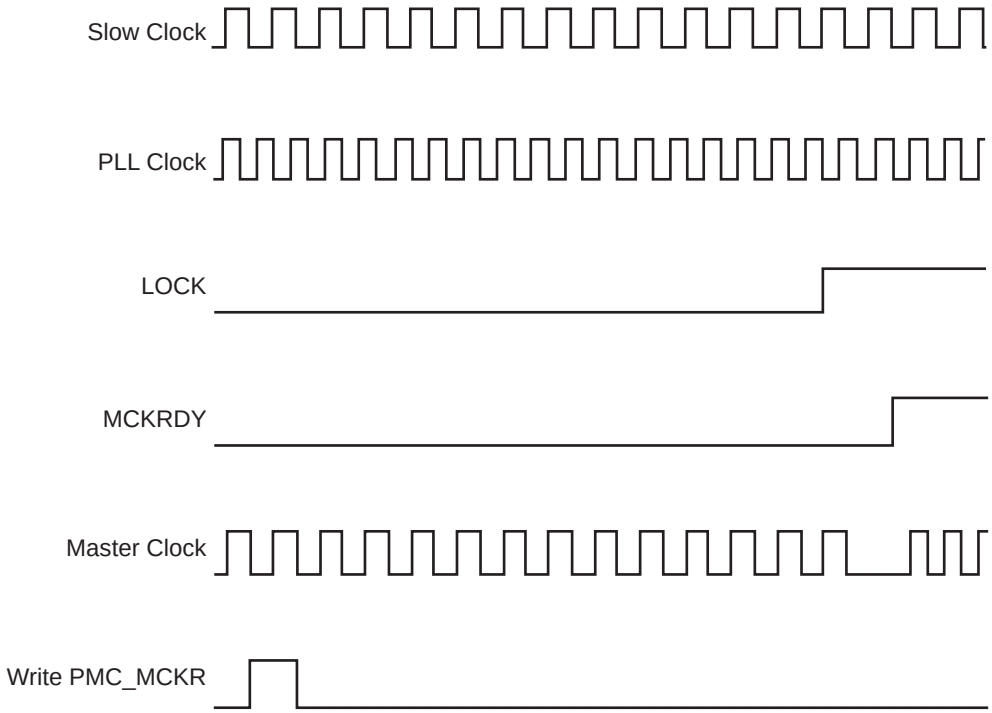


Figure 30-13. Switch Master Clock from Main Clock to Slow Clock

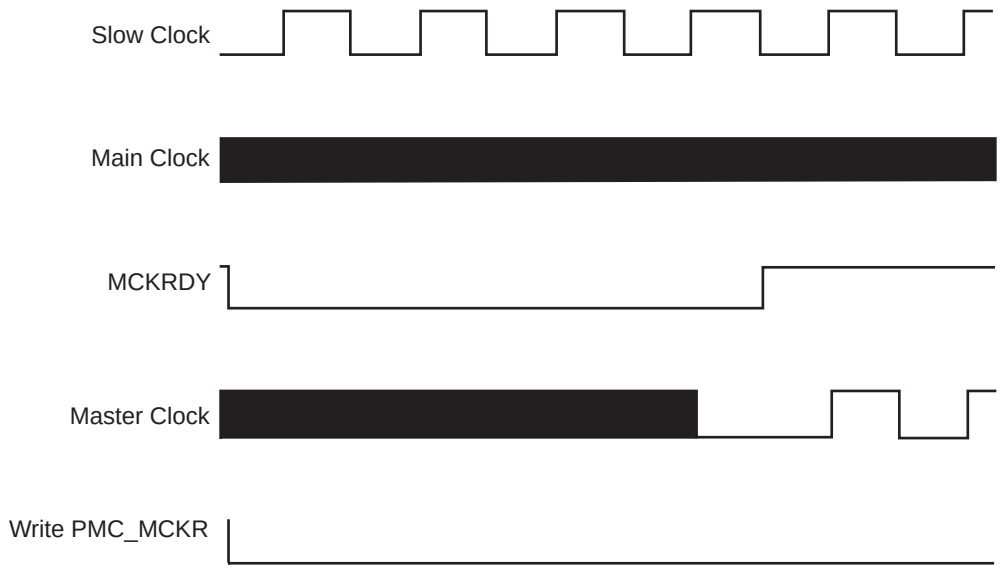


Figure 30-14. Change PLLA Programming

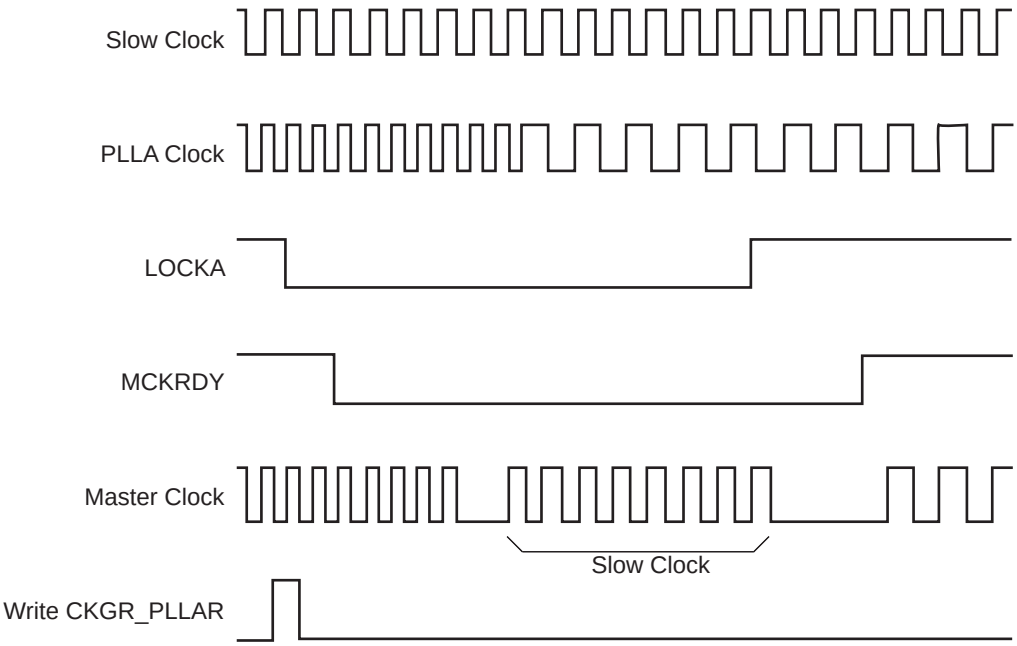
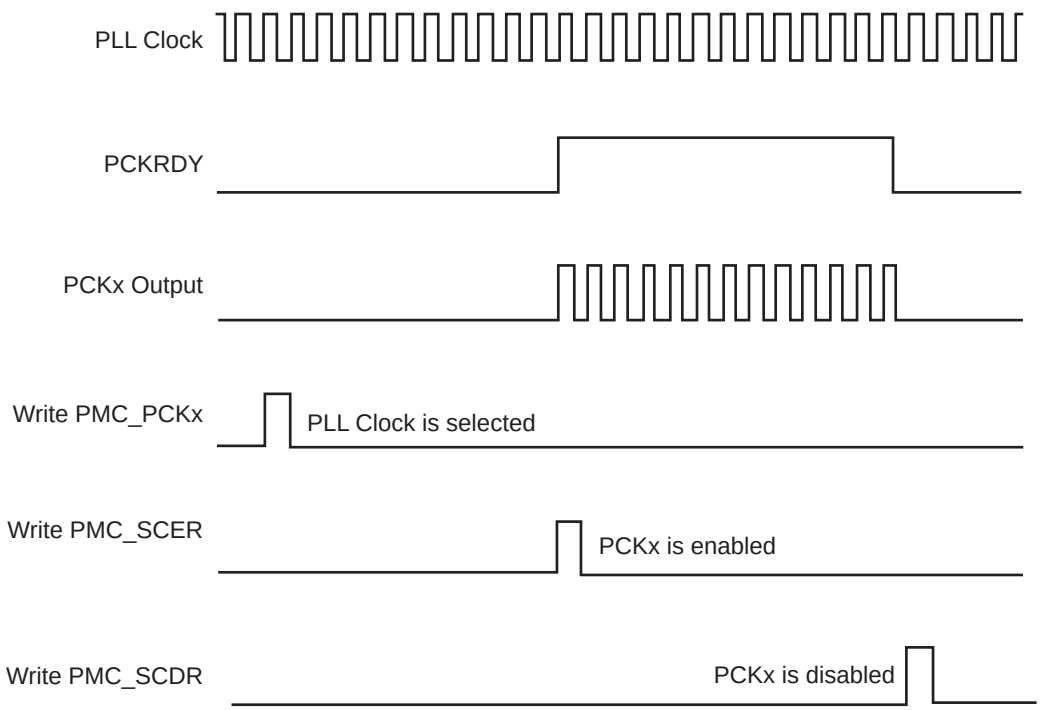


Figure 30-15. Programmable Clock Output Programming



30.21 Register Write Protection

To prevent any single software error from corrupting PMC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the PMC Write Protection Mode Register (PMC_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the PMC Write Protection Status Register (PMC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading PMC_WPSR.

The following registers can be write-protected:

- PMC System Clock Enable Register
- PMC System Clock Disable Register
- PMC Peripheral Clock Enable Register 0
- PMC Peripheral Clock Disable Register 0
- PMC Clock Generator Main Oscillator Register
- PMC Clock Generator Main Clock Frequency Register
- PMC Clock Generator PLLA Register
- PMC Master Clock Register
- PMC USB Clock Register
- PMC Programmable Clock Register
- PMC Fast Startup Polarity Register
- PMC Fast Startup Mode Register
- PLL Charge Pump Current Register
- PMC Oscillator Calibration Register
- PMC SleepWalking Enable Register 0
- PMC SleepWalking Disable Register 1
- PMC SleepWalking Enable Register 1
- PMC SleepWalking Disable Register 1
- PMC SleepWalking Control Register

30.22 Power Management Controller (PMC) User Interface

Table 30-3. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	System Clock Enable Register	PMC_SCER	Write-only	–
0x0004	System Clock Disable Register	PMC_SCDR	Write-only	–
0x0008	System Clock Status Register	PMC_SCSR	Read-only	0x0000_0005
0x000C	Reserved	–	–	–
0x0010	Peripheral Clock Enable Register 0	PMC_PCER0	Write-only	–
0x0014	Peripheral Clock Disable Register 0	PMC_PCDR0	Write-only	–
0x0018	Peripheral Clock Status Register 0	PMC_PCSR0	Read-only	0x0000_0000
0x001C	UTMI Clock Register	CKGR_UCKR	Read/Write	0x1020_0000
0x0020	Main Oscillator Register	CKGR_MOR	Read/Write	0x0100_0021
0x0024	Main Clock Frequency Register	CKGR_MCFR	Read/Write	0x0000_0000
0x0028	PLLA Register	CKGR_PLLAR	Read/Write	0x0000_3F00
0x002C	Reserved	–	–	–
0x0030	Master Clock Register	PMC_MCKR	Read/Write	0x0000_0001
0x0034	Reserved	–	–	–
0x0038	USB Clock Register	PMC_USB	Read/Write	0x0000_0000
0x003C	Reserved	–	–	–
0x0040	Programmable Clock 0 Register	PMC_PCK0	Read/Write	0x0000_0000
0x0044	Programmable Clock 1 Register	PMC_PCK1	Read/Write	0x0000_0000
0x0048	Programmable Clock 2 Register	PMC_PCK2	Read/Write	0x0000_0000
0x004C–0x005C	Reserved	–	–	–
0x0060	Interrupt Enable Register	PMC_IER	Write-only	–
0x0064	Interrupt Disable Register	PMC_IDR	Write-only	–
0x0068	Status Register	PMC_SR	Read-only	0x0001_0008
0x006C	Interrupt Mask Register	PMC_IMR	Read-only	0x0000_0000
0x0070	Fast Startup Mode Register	PMC_FSMR	Read/Write	0x0000_0000
0x0074	Fast Startup Polarity Register	PMC_FSPR	Read/Write	0x0000_0000
0x0078	Fault Output Clear Register	PMC_FOCR	Write-only	–
0x007C	Reserved	–	–	–
0x0080	PLL Charge Pump Current Register	PMC_PLLICPR	Read/Write	0x0000_0000
0x0084–0x00E0	Reserved	–	–	–
0x00E4	Write ProtectIon Mode Register	PMC_WPMR	Read/Write	0x0000_0000
0x00E8	Write Protection Status Register	PMC_WPSR	Read-only	0x0000_0000
0x00EC–0x00FC	Reserved	–	–	–
0x0100	Peripheral Clock Enable Register 1	PMC_PCER1	Write-only	–
0x0104	Peripheral Clock Disable Register 1	PMC_PCDR1	Write-only	–

Table 30-3. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0108	Peripheral Clock Status Register 1	PMC_PCSR1	Read-only	0x0000_0000
0x010C	Peripheral Control Register	PMC_PCR	Read/Write	0x0000_0000
0x0110	Oscillator Calibration Register	PMC_OCR	Read/Write	0x0040_4040
0x0114	SleepWalking Enable Register 0	PMC_SLPWK_ER0	Write-only	–
0x0118	SleepWalking Disable Register 0	PMC_SLPWK_DR0	Write-only	–
0x011C	SleepWalking Status Register 0	PMC_SLPWK_SR0	Read-only	0x0000_0000
0x0120	SleepWalking Activity Status Register 0	PMC_SLPWK_ASR0	Read-Only	–
0x0124–0x0130	Reserved	–	–	–
0x0134	SleepWalking Enable Register 1	PMC_SLPWK_ER1	Write-only	–
0x0138	SleepWalking Disable Register 1	PMC_SLPWK_DR1	Write-only	–
0x013C	SleepWalking Status Register 1	PMC_SLPWK_SR1	Read-only	0x0000_0000
0x0140	SleepWalking Activity Status Register 1	PMC_SLPWK_ASR1	Read-Only	–
0x0144	SleepWalking Activity In Progress Register	PMC_SLPWK_AIPR	Read-Only	–
0x0148	SleepWalking Control Register	PMC_SLPWKCR	Read/Write	0x0000_0000
0x014C	Audio PLL Register 0	PMC_AUDIO_PLL0	Read/Write	0x0000_00D0
0x0150	Audio PLL Register 1	PMC_AUDIO_PLL1	Read/Write	0x0000_0000

30.22.1 PMC System Clock Enable Register

Name: PMC_SCER

Address: 0xF0014000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	ISCCK	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	–	LCDCK	DDRCK	–	–

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **DDRCK: DDR Clock Enable**

0: No effect.

1: Enables the DDR clock.

- **LCDCK: MCK2x Clock Enable**

0: No effect.

1: Enables the MCK2x clock.

Note: MCK2x is selected as LCD Pixel source clock if LCDC_LCDCFG0.CLKSEL = 1.

- **UHP: USB Host OHCI Clocks Enable**

0: No effect.

1: Enables the UHP48M and UHP12M OHCI clocks.

- **UDP: USB Device Clock Enable**

0: No effect.

1: Enables the USB Device clock.

- **PCKx: Programmable Clock x Output Enable**

0: No effect.

1: Enables the corresponding Programmable Clock output.

- **ISCCK: ISC Clock Enable**

0: No effect.

1: Enables the ISC clock.

30.22.2 PMC System Clock Disable Register

Name: PMC_SCDR

Address: 0xF0014004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	ISCCK	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	PCK2	PCK1	PCK0
7	6	5	4	3	2	1	0
UDP	UHP	–	–	LCDCK	DDRCK	–	PCK

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **PCK: Processor Clock Disable**

0: No effect.

1: Disables the Processor clock. This is used to enter the processor in Idle mode.

- **DDRCK: DDR Clock Disable**

0: No effect.

1: Disables the DDR clock.

- **LCDCK: MCK2x Clock Disable**

0: No effect.

1: Disables the MCK2x clock.

- **UHP: USB Host OHCI Clock Disable**

0: No effect.

1: Disables the UHP48M and UHP12M OHCI clocks.

- **UDP: USB Device Clock Enable**

0: No effect.

1: Disables the USB Device clock.

- **PCKx: Programmable Clock x Output Disable**

0: No effect.

1: Disables the corresponding Programmable Clock output.

- **ISCCK: ISC Clock Disable**

0: No effect.

1: Disables the ISC clock.

30.22.3 PMC System Clock Status Register

Name: PMC_SCSR

Address: 0xF0014008

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	ISCCK	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	PCK2	PCK1	PCK0

7	6	5	4	3	2	1	0
UDP	UHP	–	–	LCDCK	DDRCK	–	PCK

- **PCK: Processor Clock Status**

0: The Processor clock is disabled.

1: The Processor clock is enabled.

- **DDRCK: DDR Clock Status**

0: The DDR clock is disabled.

1: The DDR clock is enabled.

- **LCDCK: MCK2x Clock Status**

0: The MCK2x clock is disabled.

1: The MCK2x clock is enabled.

Note: MCK2x is selected as LCD Pixel source clock if LCDC_LCDCFG0.CLKSEL = 1.

- **UHP: USB Host Port Clock Status**

0: The UHP48M and UHP12M OHCI clocks are disabled.

1: The UHP48M and UHP12M OHCI clocks are enabled.

- **UDP: USB Device Port Clock Status**

0: The USB Device clock is disabled.

1: The USB Device clock is enabled.

- **PCKx: Programmable Clock x Output Status**

0: The corresponding Programmable Clock output is disabled.

1: The corresponding Programmable Clock output is enabled.

- **ISCCK: ISC Clock Status**

0: The ISC clock is disabled.

1: The ISC clock is enabled.

30.22.4 PMC Peripheral Clock Enable Register 0

Name: PMC_PCER0

Address: 0xF0014010

Access: Write-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **PIDx: Peripheral Clock x Enable**

0: No effect.

1: Enables the corresponding peripheral clock.

Notes:
1. PID2 to PID31 refer to identifiers as defined in the section “Peripheral Identifiers”. Other peripherals can be enabled in PMC_PCER1.
2. Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

30.22.5 PMC Peripheral Clock Disable Register 0

Name: PMC_PCDR0

Address: 0xF0014014

Access: Write-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **PIDx: Peripheral Clock x Disable**

0: No effect.

1: Disables the corresponding peripheral clock.

Note: PID2 to PID31 refer to identifiers as defined in the section “Peripheral Identifiers”. Other peripherals can be disabled in PMC_PCDR1.

30.22.6 PMC Peripheral Clock Status Register 0

Name: PMC_PCSR0

Address: 0xF0014018

Access: Read-only

31	30	29	28	27	26	25	24
PID31	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	PID18	PID17	PID16
15	14	13	12	11	10	9	8
PID15	PID14	PID13	PID12	PID11	PID10	PID9	PID8
7	6	5	4	3	2	1	0
PID7	PID6	PID5	PID4	PID3	PID2	–	–

- **PIDx: Peripheral Clock x Status**

0: The corresponding peripheral clock is disabled.

1: The corresponding peripheral clock is enabled.

Note: PID2 to PID31 refer to identifiers as defined in the section “Peripheral Identifiers”. Other peripherals status can be read in PMC_PCSR1.

30.22.7 PMC UTMI Clock Configuration Register

Name: CKGR_UCKR

Address: 0xF001401C

Access: Read/Write

31	30	29	28	27	26	25	24
BIASCOUNT				–	–	–	BIASEN

23	22	21	20	19	18	17	16
UPLLCOUNT				–	–	–	UPLLEN

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **UPLLEN: UTMI PLL Enable**

0: The UTMI PLL is disabled.

1: The UTMI PLL is enabled.

When UPLLEN is set, the LOCKU flag is set once the UTMI PLL startup time is achieved.

- **UPLLCOUNT: UTMI PLL Startup Time**

Specifies the number of slow clock cycles multiplied by 8 for the UTMI PLL startup time.

- **BIASEN: UTMI BIAS Enable**

0: The UTMI BIAS is disabled.

1: The UTMI BIAS is enabled.

- **BIASCOUNT: UTMI BIAS Startup Time**

Specifies the number of slow clock cycles for the UTMI BIAS startup time.

30.22.8 PMC Clock Generator Main Oscillator Register

Name: CKGR_MOR

Address: 0xF0014020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	CFDEN	MOSCSEL

23	22	21	20	19	18	17	16
KEY							

15	14	13	12	11	10	9	8
MOSCXTST							

7	6	5	4	3	2	1	0
–	0	ONE	0	MOSCRCEN	WAITMODE	MOSCXTBY	MOSCXTEN

This register can only be written if the WCKGR_MOR_ONEPEN bit is cleared in the PMC Write Protection Mode Register.

Warning: bits 6:4 must always be configured to 010 when programming CKGR_MOR.

- **MOSCXTEN: 8 to 24 MHz Crystal Oscillator Enable**

A crystal must be connected between XIN and XOUT.

0: The 8 to 24 MHz crystal oscillator is disabled.

1: The 8 to 24 MHz crystal oscillator is enabled. MOSCXTBY must be cleared.

When MOSCXTEN is set, the MOSCXTS flag is set once the crystal oscillator startup time is achieved.

- **MOSCXTBY: 8 to 24 MHz Crystal Oscillator Bypass**

0: No effect.

1: The 8 to 24 MHz crystal oscillator is bypassed. MOSCXTEN must be cleared. An external clock must be connected on XIN.

When MOSCXTBY is set, the MOSCXTS flag in PMC_SR is automatically set.

Clearing MOSCXTEN and MOSCXTBY bits allows resetting the MOSCXTS flag.

Note: When Main Oscillator Bypass is disabled (MOSCXTBY = 0), the MOSCXTS flag must be read as 0 in PMC_SR prior to enabling the main crystal oscillator (MOSCXTEN = 1).

- **WAITMODE: Wait Mode Command (Write-only)**

0: No effect.

1: Puts the device in Wait mode.

- **MOSCRCEN: 12 MHz RC Oscillator Enable**

0: The 12 MHz RC oscillator is disabled.

1: The 12 MHz RC oscillator is enabled.

When MOSCRCEN is set, the MOSCRCS flag is set once the RC oscillator startup time is achieved.

- **ONE: Must Be Set to 1**

When programming CKGR_MOR, bit 5 must always be set to 1; bits 6 and 4 must always be set to 0.

- **MOSCXTST: 8 to 24 MHz Crystal Oscillator Startup Time**

Specifies the number of slow clock cycles multiplied by 8 for the crystal oscillator startup time.

- **KEY: Password**

Value	Name	Description
0x37	PASSWD	Writing any other value in this field aborts the write operation.

- **MOSCSEL: Main Clock Oscillator Selection**

0: The 12 MHz oscillator is selected.

1: The 8 to 24 MHz crystal oscillator is selected.

- **CFDEN: Clock Failure Detector Enable**

0: The clock failure detector is disabled.

1: The clock failure detector is enabled.

30.22.9 PMC Clock Generator Main Clock Frequency Register

Name: CKGR_MCFR

Address: 0xF0014024

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	CCSS

23	22	21	20	19	18	17	16
–	–	–	RCMEAS	–	–	–	MAINFRDY

15	14	13	12	11	10	9	8
MAINF							

7	6	5	4	3	2	1	0
MAINF							

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **MAINF: Main Clock Frequency**

Gives the number of cycles of the clock selected by the bit CCSS within 16 slow clock periods. To calculate the frequency of the measured clock:

$f_{SELCK} = (MAINF \times f_{SLCK}) / 16$

where frequency is in MHz.

- **MAINFRDY: Main Clock Frequency Measure Ready**

0: MAINF value is not valid or the measured oscillator is disabled or a measure has just been started by means of RCMEAS.

1: The measured oscillator has been enabled previously and MAINF value is available.

Note: To ensure that a correct value is read on the MAINF field, the MAINFRDY flag must be read at 1 then another read access must be performed on the register to get a stable value on the MAINF field.

- **RCMEAS: RC Oscillator Frequency Measure (write-only)**

0: No effect.

1: Restarts measuring of the frequency of the main clock source. MAINF will carry the new frequency as soon as a low to high transition occurs on the MAINFRDY flag.

The measure is performed on the main frequency (i.e., not limited to RC oscillator only), but if the main clock frequency source is the 8 to 24 MHz crystal oscillator, the restart of measuring is not needed because of the well known stability of crystal oscillators.

- **CCSS: Counter Clock Source Selection**

0: The clock of the MAINF counter is the RC oscillator.

1: The clock of the MAINF counter is the crystal oscillator.

30.22.10 PMC Clock Generator PLLA Register

Name: CKGR_PLLAR

Address: 0xF0014028

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ONE	–	–	–	–	MULA

23	22	21	20	19	18	17	16
MULA						OUTA	

15	14	13	12	11	10	9	8
OUTA		PLLACOUNT					

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DIVA

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

Possible limitations on PLL input frequencies and multiplier factors should be checked before using the PMC.

- **DIVA: Divider A**

0: Divider output is 0

1: Divider is bypassed and the PLL input entry is main clock.

- **PLLACOUNT: PLLA Counter**

Specifies the number of slow clock cycles before the LOCKA bit is set in PMC_SR after CKGR_PLLAR is written.

- **OUTA: PLLA Clock Frequency Range**

To be programmed to 0.

- **MULA: PLLA Multiplier**

0: The PLLA is deactivated.

1–127: The PLLA Clock frequency is the PLLA input frequency multiplied by MULA + 1.

- **ONE: Must Be Set to 1**

Bit 29 must always be set to 1 when programming CKGR_PLLAR.

30.22.11 PMC Master Clock Register

Name: PMC_MCKR

Address: 0xF0014030

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	H32MXDIV

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	PLLADIV2	–	–	MDIV	

7	6	5	4	3	2	1	0
–	PRES			–	–	CSS	

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **CSS: Master/Processor Clock Source Selection**

Value	Name	Description
0	SLOW_CLK	Slow clock is selected
1	MAIN_CLK	Main clock is selected
2	PLLA_CLK	PLLACK is selected
3	UPLL_CLK	UPLL Clock is selected

- **PRES: Master/Processor Clock Prescaler**

Value	Name	Description
0	CLOCK	Selected clock
1	CLOCK_DIV2	Selected clock divided by 2
2	CLOCK_DIV4	Selected clock divided by 4
3	CLOCK_DIV8	Selected clock divided by 8
4	CLOCK_DIV16	Selected clock divided by 16
5	CLOCK_DIV32	Selected clock divided by 32
6	CLOCK_DIV64	Selected clock divided by 64
7	–	Reserved

- **MDIV: Master Clock Division**

Value	Name	Description
0	EQ_PCK	Master Clock is Prescaler Output Clock divided by 1. **Warning**: DDRCK is not available.
1	PCK_DIV2	Master Clock is Prescaler Output Clock divided by 2. DDRCK is equal to MCK.
2	PCK_DIV4	Master Clock is Prescaler Output Clock divided by 4. DDRCK is equal to MCK.
3	PCK_DIV3	Master Clock is Prescaler Output Clock divided by 3. DDRCK is equal to MCK.

- **PLLADIV2: PLLA Divisor by 2**

Bit PLLADIV2 must always be set to 1 when MDIV is set to 3.

- **H32MXDIV: AHB 32-bit Matrix Divisor**

Value	Name	Description
0	H32MXDIV1	The AHB 32-bit Matrix frequency is equal to the AHB 64-bit Matrix frequency. It is possible only if the AHB 64-bit Matrix frequency does not exceed 83 MHz.
1	H32MXDIV2	The AHB 32-bit Matrix frequency is equal to the AHB 64-bit Matrix frequency divided by 2.

30.22.12 PMC USB Clock Register

Name: PMC_USB

Address: 0xF0014038

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	USBDIV			
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	USBS

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **USBS: USB OHCI Input Clock Selection**

0: USB Clock Input is PLLA.

1: USB Clock Input is UPLL.

- **USBDIV: Divider for USB OHCI Clock**

USB Clock is Input clock divided by USBDIV + 1.

30.22.13 PMC Programmable Clock Register

Name: PMC_PCKx[x = 0..2]

Address: 0xF0014040, 0xF0014044, 0xF0014048

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	PRES			

7	6	5	4	3	2	1	0
PRES				–	CSS		

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **CSS: Master Clock Source Selection**

Value	Name	Description
0	SLOW_CLK	Slow clock is selected
1	MAIN_CLK	Main clock is selected
2	PLLA_CLK	PLLACK is selected
3	UPLL_CLK	UPLL clock is selected
4	MCK_CLK	Master clock is selected
5	AUDIO_CLK	Audio PLL clock is selected

- **PRES: Programmable Clock Prescaler**

Programmable Clock Frequency = Selected Clock Frequency / (PRES + 1)

30.22.14 PMC Interrupt Enable Register

Name: PMC_IER

Address: 0xF0014060

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	CFDEV	MOSCRCS	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	LOCKU	–	–	MCKRDY	–	LOCKA	MOSCXTS

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt

- **MOSCXTS: 8 to 24 MHz Crystal Oscillator Status Interrupt Enable**
- **LOCKA: PLLA Lock Interrupt Enable**
- **MCKRDY: Master Clock Ready Interrupt Enable**
- **LOCKU: UTMI PLL Lock Interrupt Enable**
- **PCKRDYx: Programmable Clock Ready x Interrupt Enable**
- **MOSCSELS: Main Clock Source Oscillator Selection Status Interrupt Enable**
- **MOSCRCS: 12 MHz RC Oscillator Status Interrupt Enable**
- **CFDEV: Clock Failure Detector Event Interrupt Enable**

30.22.15 PMC Interrupt Disable Register

Name: PMC_IDR

Address: 0xF0014064

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	CFDEV	MOSCRCS	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	LOCKU	–	–	MCKRDY	–	LOCKA	MOSCXTS

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt

- **MOSCXTS: 8 to 24 MHz Crystal Oscillator Status Interrupt Disable**
- **LOCKA: PLLA Lock Interrupt Disable**
- **MCKRDY: Master Clock Ready Interrupt Disable**
- **LOCKU: UTMI PLL Lock Interrupt Enable**
- **PCKRDYx: Programmable Clock Ready x Interrupt Disable**
- **MOSCSELS: Main Oscillator Clock Source Selection Status Interrupt Disable**
- **MOSCRCS: 12 MHz RC Oscillator Status Interrupt Disable**
- **CFDEV: Clock Failure Detector Event Interrupt Disable**

30.22.16 PMC Status Register

Name: PMC_SR

Address: 0xF0014068

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	GCKRDY
23	22	21	20	19	18	17	16
–	–	–	FOS	CFDS	CFDEV	MOSCRCS	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
OSCSELS	LOCKU	–	–	MCKRDY	–	LOCKA	MOSCXTS

- **MOSCXTS: 8 to 24 MHz Crystal Oscillator Status**

0: 8 to 24 MHz crystal oscillator is not stabilized.

1: 8 to 24 MHz crystal oscillator is stabilized.

- **LOCKA: PLLA Lock Status**

0: PLLA is not locked.

1: PLLA is locked.

- **MCKRDY: Master Clock Status**

0: Master Clock is not ready.

1: Master Clock is ready.

- **LOCKU: UPLL Clock Status**

0: UPLL Clock is not ready.

1: UPLL Clock is ready.

- **OSCSELS: Slow Clock Oscillator Selection**

0: Embedded 64 kHz RC oscillator is selected.

1: 32.768 kHz crystal oscillator is selected.

- **PCKRDYx: Programmable Clock Ready Status**

0: Programmable Clock x is not ready.

1: Programmable Clock x is ready.

- **MOSCSELS: Main Oscillator Selection Status**

0: Selection is in progress.

1: Selection is done.

- **MOSCRCS: 12 MHz RC Oscillator Status**

0: 12 MHz RC oscillator is not stabilized.

1: 12 MHz RC oscillator is stabilized.

- **CFDEV: Clock Failure Detector Event**

0: No clock failure detection of the 8 to 24 MHz crystal oscillator has occurred since the last read of PMC_SR.

1: At least one clock failure detection of the 8 to 24 MHz crystal oscillator has occurred since the last read of PMC_SR.

- **CFDS: Clock Failure Detector Status**

0: A clock failure of the 8 to 24 MHz crystal oscillator is not detected.

1: A clock failure of the 8 to 24 MHz crystal oscillator is detected.

- **FOS: Clock Failure Detector Fault Output Status**

0: The fault output of the clock failure detector is inactive.

1: The fault output of the clock failure detector is active.

- **GCKRDY: Generic Clock Status**

0: One of the generic clocks is not ready yet.

1: All generic clocks are ready.

30.22.17 PMC Interrupt Mask Register

Name: PMC_IMR

Address: 0xF001406C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	CFDEV	MOSCRCS	MOSCSELS
15	14	13	12	11	10	9	8
–	–	–	–	–	PCKRDY2	PCKRDY1	PCKRDY0
7	6	5	4	3	2	1	0
–	–	–	–	MCKRDY	–	LOCKA	MOSCXTS

The following configuration values are valid for all listed bit names of this register:

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

- **MOSCXTS: 8 to 24 MHz Crystal Oscillator Status Interrupt Mask**
- **LOCKA: PLLA Lock Interrupt Mask**
- **MCKRDY: Master Clock Ready Interrupt Mask**
- **PCKRDYx: Programmable Clock Ready x Interrupt Mask**
- **MOSCSELS: Main Oscillator Clock Source Selection Status Interrupt Mask**
- **MOSCRCS: 12 MHz RC Oscillator Status Interrupt Mask**
- **CFDEV: Clock Failure Detector Event Interrupt Mask**

30.22.18 PMC Fast Startup Polarity Register

Name: PMC_FSPR

Address: 0xF0014074

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	FSTP10	FSTP9	FSTP8
7	6	5	4	3	2	1	0
FSTP7	FSTP6	FSTP5	FSTP4	FSTP3	FSTP2	FSTP1	FSTP0

This register can only be written if the WPEN bit is cleared in PMC Write Protection Mode Register.

- **FSTP0: WKUP Pin Polarity for Fast Startup**

Defines the active polarity of the wakeup input. If the wakeup input is enabled and at the FSTP level, it enables a fast restart signal.

- **FSTP1: Security Module Polarity for Fast Startup**

If PMC_FSMR.FSTT1 = 1, FSTP1 must be written to 1.

- **FSTP2–FSTP9: PIOBU0–7 Pin Polarity for Fast Startup**

Defines the active polarity of the corresponding PIOBUx input. If the corresponding wakeup input is enabled and at the FSTP level, it enables a fast restart signal.

- **FSTP10: GMAC Wakeup On LAN Polarity for Fast Startup**

If PMC_FSMR.FSTT10 = 1, FSTP10 must be written to 1.

30.22.19 PMC Fast Startup Mode Register

Name: PMC_FSMR

Address: 0xF0014070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	ACC_CE	RXLP_MCE
23	22	21	20	19	18	17	16
–	–	–	LPM	SDMMC_CD	USBAL	RTCAL	–
15	14	13	12	11	10	9	8
–	–	–	–	–	FSTT10	FSTT9	FSTT8
7	6	5	4	3	2	1	0
FSTT7	FSTT6	FSTT5	FSTT4	FSTT3	FSTT2	FSTT1	FSTT0

This register can only be written if the WPEN bit is cleared in PMC Write Protection Mode Register.

- **FSTT0: Fast Startup from WKUP Pin Enable**

0: The wakeup input (WKUP) has no effect on the PMC.

1: The wakeup input (WKUP) can trigger a fast restart signal to the PMC.

- **FSTT1: Fast Startup from Security Module Enable**

0: The SECUMOD has no effect on the PMC.

1: The SECUMOD can trigger a fast restart signal to the PMC.

- **FSTT2–FSTT9: Fast Startup from PIOBU0–7 Input Enable**

0: The corresponding PIOBUx input has no effect on the PMC.

1: The corresponding PIOBUx input can trigger a fast restart signal to the PMC.

- **FSTT10: Fast Startup from GMAC Wakeup On LAN Enable**

0: The GMAC_WOL input has no effect on the PMC.

1: The GMAC_WOL input can trigger a fast restart signal to the PMC.

- **RTCAL: Fast Startup from RTC Alarm Enable**

0: The RTC alarm has no effect on the PMC.

1: The RTC alarm can trigger a fast restart signal to the PMC.

- **USBAL: Fast Startup from USB Resume Enable**

0: The USB resume has no effect on the PMC.

1: The USB resume can trigger a fast restart signal to the PMC.

- **SDMMC_CD: Fast Startup from SDMMC Card Detect Enable**

0: The SDMMC card detect has no effect on the PMC.

1: The SDMMC card detect can trigger a fast restart signal to the PMC.

- **LPM: Low-power Mode**

0: The WaitForInterrupt (WFI) or the WaitForEvent (WFE) instruction of the processor instructs the processor to enter Idle mode.

1: The WaitForEvent (WFE) instruction of the processor instructs the system to enter ULP mode 1.

- **RXLP_MCE: Fast Startup from Backup UART Receive Match Condition Enable**

0: The matching condition on the RXLP has no effect on the PMC.

1: The matching condition on the RXLP can trigger a fast restart signal to the PMC.

- **ACC_CE: Fast Startup from Analog Comparator Controller Comparison Enable**

0: The ACC (Analog Comparator Controller) comparison has no effect on the PMC.

1: The ACC (Analog Comparator Controller) comparison can trigger a fast restart signal to the PMC.

30.22.20 PMC Fault Output Clear Register

Name: PMC_FOCR

Address: 0xF0014078

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	FOCLR

- **FOCLR: Fault Output Clear**

Clears the clock failure detector fault output.

30.22.21 PLL Charge Pump Current Register

Name: PMC_PLLICPR

Address: 0xF0014080

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	IVCO_PLLU	

23	22	21	20	19	18	17	16
–	–	–	–	–	–	ICP_PLLU	

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	ICP_PLLA	

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **ICP_PLLA: Charge Pump Current**

To optimize clock performance, this field must be programmed as specified in “PLL A Characteristics” in the Electrical Characteristics section.

- **ICP_PLLU: Charge Pump Current PLL UTMI**

Should be written to 0.

- **IVCO_PLLU: Voltage Control Output Current PLL UTMI**

Should be written to 0.

30.22.22 PMC Write Protection Mode Register

Name: PMC_WPMR

Address: 0xF00140E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x504D43 (“PMC” in ASCII).

See Section 30.21 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x504D43	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

30.22.23 PMC Write Protection Status Register

Name: PMC_WPSR

Address: 0xF00140E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of PMC_WPSR.

1: A write protection violation has occurred since the last read of PMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

30.22.24 PMC Peripheral Clock Enable Register 1

Name: PMC_PCER1

Address: 0xF0014100

Access: Write-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **PIDx: Peripheral Clock x Enable**

0: No effect.

1: Enables the corresponding peripheral clock.

Notes: 1. PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”.
2. Programming the control bits of the Peripheral ID that are not implemented has no effect on the behavior of the PMC.

30.22.25 PMC Peripheral Clock Disable Register 1

Name: PMC_PCDR1

Address: 0xF0014104

Access: Write-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **PIDx: Peripheral Clock x Disable**

0: No effect.

1: Disables the corresponding peripheral clock.

Note: PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”.

30.22.26 PMC Peripheral Clock Status Register 1

Name: PMC_PCSR1

Address: 0xF0014108

Access: Read-only

31	30	29	28	27	26	25	24
PID63	PID62	PID61	PID60	PID59	PID58	PID57	PID56
23	22	21	20	19	18	17	16
PID55	PID54	PID53	PID52	PID51	PID50	PID49	PID48
15	14	13	12	11	10	9	8
PID47	PID46	PID45	PID44	PID43	PID42	PID41	PID40
7	6	5	4	3	2	1	0
PID39	PID38	PID37	PID36	PID35	PID34	PID33	PID32

- **PIDx: Peripheral Clock x Status**

0: The corresponding peripheral clock is disabled.

1: The corresponding peripheral clock is enabled.

Note: PID32 to PID63 refer to identifiers as defined in the section “Peripheral Identifiers”.

30.22.27 PMC Peripheral Control Register

Name: PMC_PCR

Address: 0xF001410C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	GCKEN	EN	GCKDIV			

23	22	21	20	19	18	17	16
GCKDIV				–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	CMD	–	GCKCSS		

7	6	5	4	3	2	1	0
–	PID						

- **PID: Peripheral ID**

Peripheral ID selection from PID2 to the maximum PID number. This refers to identifiers as defined in the section “Peripheral Identifiers”.

- **GCKCSS: Generic Clock Source Selection**

Value	Name	Description
0	SLOW_CLK	Slow clock is selected
1	MAIN_CLK	Main clock is selected
2	PLLA_CLK	PLLACK is selected
3	UPLL_CLK	UPLL Clock is selected
4	MCK_CLK	Master Clock is selected
5	AUDIO_CLK	Audio PLL clock is selected

- **CMD: Command**

0: Read mode

1: Write mode

- **GCKDIV: Generic Clock Division Ratio**

Generic clock is: selected clock period divided by GCKDIV + 1. GCKDIV must not be changed while the peripheral selects GCLK (e.g., bit rate, etc.).

- **EN: Enable**

0: The selected peripheral clock is disabled.

1: The selected peripheral clock is enabled.

- **GCKEN: Generic Clock Enable**

0: The selected generic clock is disabled.

1: The selected generic clock is enabled.

30.22.28 PMC Oscillator Calibration Register

Name: PMC_OCR

Address: 0xF0014110

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
SEL	CAL						

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **CAL: 12 MHz RC Oscillator Calibration Bits**

Calibration bits applied to the RC oscillator when SEL is set.

- **SEL: Selection of RC Oscillator Calibration Bits**

0: Factory determined value.

1: Value written by user in CAL field of this register.

30.22.29 PMC SleepWalking Enable Register 0

Name: PMC_SLPWK_ER0

Address: 0xF0014114

Access: Write-only

31	30	29	28	27	26	25	24
–	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

This register can only be written if the WPEN bit is cleared in the "PMC Write Protection Mode Register" .

- **PIDx: Peripheral x SleepWalking Enable**

0: No effect.

1: The asynchronous partial wakeup (SleepWalking) function of the corresponding peripheral is enabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The clock of the peripheral must be enabled before using its asynchronous partial wakeup (SleepWalking) function (its associated PIDx field in "ISCCK: ISC Clock Status" or "PMC Peripheral Clock Status Register 1" is set to '1').

Note: The values for PIDx are defined in section "Peripheral Identifiers".

30.22.30 PMC SleepWalking Disable Register 0

Name: PMC_SLPWK_DR0

Address: 0xF0014118

Access: Write-only

31	30	29	28	27	26	25	24
–	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

This register can only be written if the WPEN bit is cleared in the “PMC Write Protection Mode Register” .

- **PIDx: Peripheral x SleepWalking Disable**

0: No effect.

1: The asynchronous partial wakeup (SleepWalking) function of the corresponding peripheral is disabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.31 PMC SleepWalking Status Register 0

Name: PMC_SLPWK_SR0

Address: 0xF001411C

Access: Read-only

31	30	29	28	27	26	25	24
–	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **PIDx: Peripheral x SleepWalking Status**

0: The asynchronous partial wakeup (SleepWalking) function of the peripheral is currently disabled or the peripheral enabled for asynchronous partial wakeup (SleepWalking) cleared the PIDx bit upon detection of a wakeup condition.

1: The asynchronous partial wakeup (SleepWalking) function of the peripheral is currently enabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.32 PMC SleepWalking Activity Status Register 0

Name: PMC_SLPWK_ASR0

Address: 0xF0014120

Access: Read-only

31	30	29	28	27	26	25	24
–	PID30	PID29	PID28	PID27	PID26	PID25	PID24
23	22	21	20	19	18	17	16
PID23	PID22	PID21	PID20	PID19	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **PIDx: Peripheral x Activity Status**

0: The peripheral x is not presently active. The asynchronous partial wakeup (SleepWalking) function can be activated.

1: The peripheral x is presently active. The asynchronous partial wakeup (SleepWalking) function must not be activated.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC. All other PIDs are always read at 0.

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.33 PMC SleepWalking Enable Register 1

Name: PMC_SLPWK_ER1

Address: 0xF0014134

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	PID40
7	6	5	4	3	2	1	0
–	–	–	–	–	PID34	PID33	–

This register can only be written if the WPEN bit is cleared in the “PMC Write Protection Mode Register” .

- **PIDx: Peripheral x SleepWalking Enable**

0: No effect.

1: The asynchronous partial wakeup (SleepWalking) function of the corresponding peripheral is enabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

The clock of the peripheral must be enabled before using its asynchronous partial wakeup (SleepWalking) function (the associated PIDx field in “PMC Peripheral Clock Status Register 1” or “ISCCK: ISC Clock Status” is set to ‘1’).

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.34 PMC SleepWalking Disable Register 1

Name: PMC_SLPWK_DR1

Address: 0xF0014138

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	PID40
7	6	5	4	3	2	1	0
–	–	–	–	–	PID34	PID33	–

This register can only be written if the WPEN bit is cleared in the “PMC Write Protection Mode Register” .

- **PIDx: Peripheral x SleepWalking Disable**

0: No effect.

1: The asynchronous partial wakeup (SleepWalking) function of the corresponding peripheral is disabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.35 PMC SleepWalking Status Register 1

Name: PMC_SLPWK_SR1

Address: 0xF001413C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	PID40
7	6	5	4	3	2	1	0
–	–	–	–	–	PID34	PID33	–

- **PIDx: Peripheral x SleepWalking Status**

0: The asynchronous partial wakeup (SleepWalking) function of the peripheral is currently disabled or the peripheral enabled for asynchronous partial wakeup (SleepWalking) cleared the PIDx bit upon detection of a wakeup condition.

1: The asynchronous partial wakeup (SleepWalking) function of the peripheral is currently enabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.36 PMC SleepWalking Activity Status Register 1

Name: PMC_SLPWK_ASR1

Address: 0xF0014140

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	PID40
7	6	5	4	3	2	1	0
–	–	–	–	–	PID34	PID33	–

- **PIDx: Peripheral x Activity Status**

0: The peripheral x is not currently active; the asynchronous partial wakeup (SleepWalking) function can be activated.

1: The peripheral x is currently active; the asynchronous partial wakeup (SleepWalking) function must not be activated.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC. All other PIDs are always read at 0.

Note: The values for PIDx are defined in the section “Peripheral Identifiers”.

30.22.37 PMC SleepWalking Activity In Progress Register

Name: PMC_SLPWK_AIPR

Address: 0xF0014144

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	AIP

- **AIP: Activity In Progress**

0: There is no activity on peripherals. The asynchronous partial wakeup (SleepWalking) function can be activated on one or more peripherals. The device can enter ULP mode 1.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

1: One or more peripherals are currently active. The device must not enter ULP mode 1 if the asynchronous partial wakeup is enabled for one of the following PIDs: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

30.22.38 PMC SleepWalking Control Register

Name: PMC_SLPWKCR

Address: 0xF0014148

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	SLPWKSR	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	ASR
15	14	13	12	11	10	9	8
–	–	–	CMD	–	–	–	–
7	6	5	4	3	2	1	0
–	PID						

This register can only be written if the WPEN bit is cleared in the PMC Write Protection Mode Register.

- **PID: Peripheral ID**

Peripheral ID selection from PID2 to the maximum PID number. This refers to identifiers as defined in the section “Peripheral Identifiers”.

- **CMD: Command**

0: Read mode

1: Write mode

- **ASR: Activity Status Register**

0: The peripheral x is not currently active; the asynchronous partial wakeup (SleepWalking) function can be activated.

1: The peripheral x is currently active; the asynchronous partial wakeup (SleepWalking) function must not be activated.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

- **SLPWKSR: SleepWalking Status Register**

0: The asynchronous partial wakeup (SleepWalking) function of the peripheral is disabled.

1: The asynchronous partial wakeup (SleepWalking) function of the peripheral is enabled.

Not all PIDs can be configured with asynchronous partial wakeup.

Only the following PIDs can be configured with asynchronous partial wakeup: FLEXCOMx, SPIx, TWIx, UARTx and ADC.

30.22.39 PMC Audio PLL Control Register 0

Name: PMC_AUDIO_PLL0

Address: 0xF001414C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	DCO_GAIN		DCO_FILTER			

23	22	21	20	19	18	17	16
–	QDPMC						

15	14	13	12	11	10	9	8
–	ND						

7	6	5	4	3	2	1	0
PLLFLT				RESETN	PMCEN	PADEN	PLLEN

- **PLLEN: PLL Enable**

0: The Audio PLL is disabled.

1: The Audio PLL is enabled

- **PADEN: Pad Clock Enable**

0: The external audio pin CLK_AUDIO is driven low.

1: The external audio pin CLK_AUDIO is driven by AUDIOPINCLK.

- **PMCEN: PMC Clock Enable**

0: The output clock of the audio PLL is not sent to the PMC.

1: The output clock of the audio PLL is sent to the PMC.

- **RESETN: Audio PLL Reset**

0: The audio PLL is in reset state.

1: The audio PLL is in active state.

- **PLLFLT: PLL Loop Filter Selection**

Default value should be 13 (0xD)

- **ND: Loop Divider Ratio**

- **QDPMC: Output Divider Ratio for PMC Clock**

$f_{pmc} = f_{ref} \times ((ND + 1) + FRACR \div 2^{22}) / (QDPMC + 1)$

- **DCO_FILTER: Digitally Controlled Oscillator Filter Selection**

For optimization, the value of this field must be configured to 0.

- **DCO_GAIN: Digitally Controlled Oscillator Gain Selection**

For optimization, the value of this field must be configured to 0.

30.22.40 PMC Audio PLL Control Register 1

Name: PMC_AUDIO_PLL1

Address: 0xF0014150

Access: Read/Write

31	30	29	28	27	26	25	24
–	QDAUDIO					DIV	

23	22	21	20	19	18	17	16
–	–	FRACR					

15	14	13	12	11	10	9	8
FRACR							

7	6	5	4	3	2	1	0
FRACR							

- **FRACR: Fractional Loop Divider Setting**

- **DIV: Divider Value**

Value	Name	Description
0	FORBIDDEN	Reserved
1	FORBIDDEN	Reserved
2	DIV2	Divide by 2
3	DIV3	Divide by 3

- **QDAUDIO: Output Divider Ratio for Pad Clock**

$f_{audio} = f_{ref} \times ((ND + 1) + FRACR \div 2^{22}) / (DIV \times QDAUDIO)$

31. Parallel Input/Output Controller (PIO)

31.1 Description

The Parallel Input/Output Controller (PIO) manages up to 128 fully programmable input/output lines. Each I/O line may be dedicated as a general purpose I/O or be assigned to a function of an embedded peripheral. This ensures effective optimization of the pins of the product.

Each I/O line of the PIO Controller features:

- An input change interrupt enabling level change detection on any I/O line
- Rising edge, falling edge, both edge, low-level or high-level detection on any I/O line
- A glitch filter providing rejection of glitches lower than one-half of PIO clock cycle
- A debouncing filter providing rejection of unwanted pulses from key or push button operations
- Multi-drive capability similar to an open drain I/O line
- Control of the pull-up and pull-down of the I/O line
- Input visibility and output control
- Secure or Non-Secure management of the I/O line

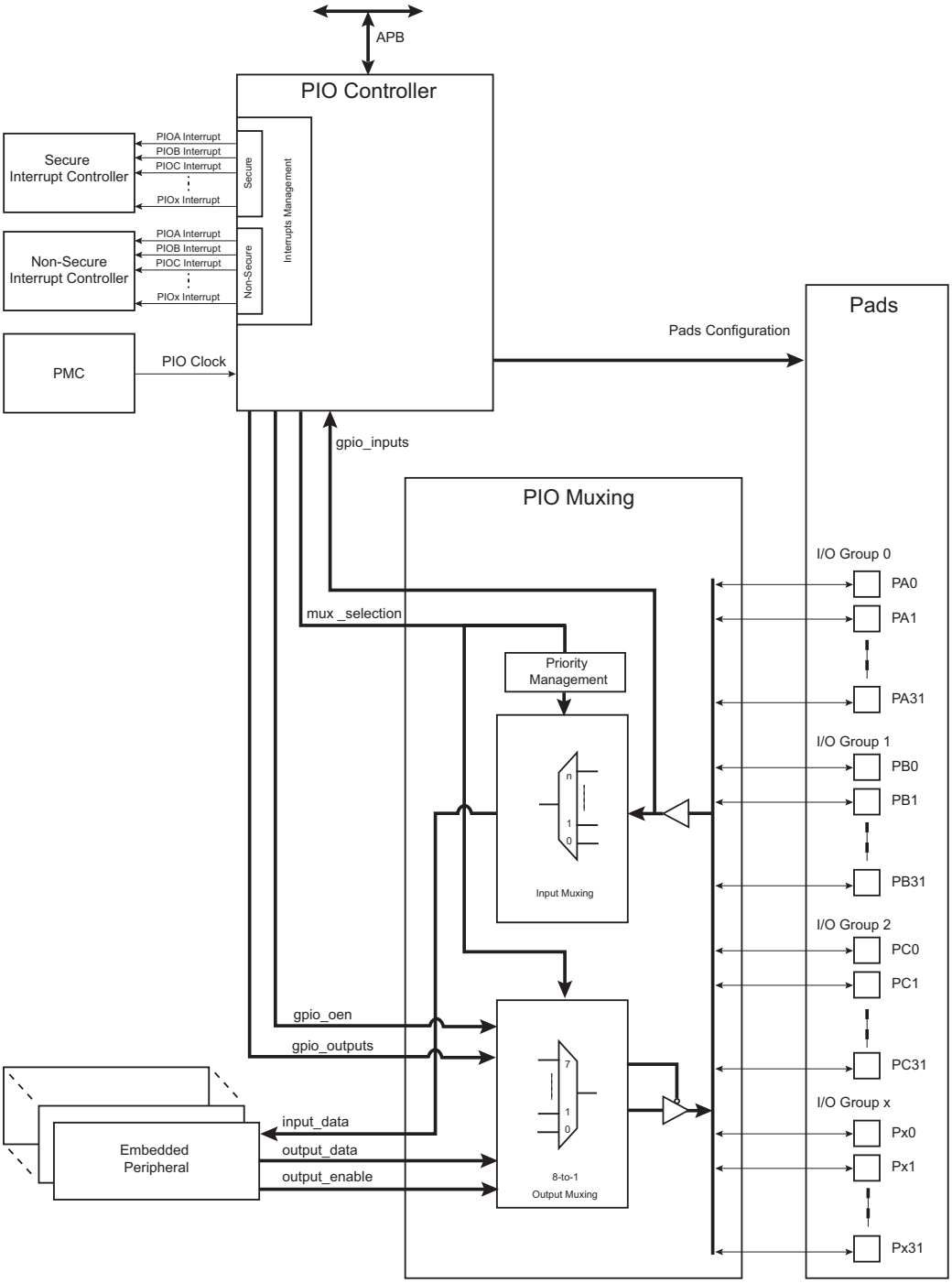
The PIO Controller also features a synchronous output providing up to 32 bits of data output in a single write operation.

31.2 Embedded Characteristics

- Up to 128 Programmable I/O Lines
- Multiplexing of up to Seven Peripheral Functions per I/O Line
- For each I/O Line (whether assigned to a peripheral or used as general purpose I/O)
  - Input Change Interrupt
  - Programmable Glitch Filter
  - Programmable Debouncing Filter
  - Multi-drive Option Enables Driving in Open Drain
  - Programmable Pull-Up/Pull-Down on Each I/O Line
  - Pin Data Status Register, Supplies Visibility of the Level on the Pin at Any Time
  - Programmable Event: Rising Edge, Falling Edge, Both edge, Low-Level or High-Level
  - Configuration Lock by the Connected Peripheral
  - Security management of each I/O line
  - Programmable Configuration Lock (Active Until Next V_{DDCORE} Reset) to protect Against Further Software Modifications (intentional or unintentional)
- Register Write Protection against unintentional software modifications:
  - One Configuration Bit to Enable or Disable Protection of I/O Line Settings
  - One Configuration Bit to Enable or Disable Protection of Interrupt Settings
- Synchronous Output, possibility to set or clear simultaneously up to 32 I/O Lines in a Single Write
- Programmable Schmitt Trigger Inputs
- Programmable I/O Drive

31.3 Block Diagram

Figure 31-1. Block Diagram



Notes: 1. x = 3 (the number of I/O group is 4)
2. n depends on the number of I/O lines affected to the IP input

31.4 Product Dependencies

31.4.1 Pin Multiplexing

Each pin is configurable, depending on the product, as either a general purpose I/O line only, or as an I/O line multiplexed with up to 6 peripheral I/Os. As the multiplexing is hardware defined and thus product-dependent, the hardware designer and programmer must carefully determine the configuration of the PIO Controllers required by their application. When an I/O line is general purpose only, i.e., not multiplexed with any peripheral I/O, programming of the PIO Controller regarding the assignment to a peripheral has no effect and only the PIO Controller can control how the pin is driven by the product.

31.4.2 External Interrupt Lines

The interrupt signals FIQ and IRQ0 to IRQn are multiplexed through the PIO Controllers.

31.4.3 Power Management

The Power Management Controller (PMC) controls the PIO Controller clock in order to save power. Writing any of the registers of the user interface does not require the PIO Controller clock to be enabled. This means that the configuration of the I/O lines does not require the PIO Controller clock to be enabled.

However, when the clock is disabled, not all of the features of the PIO Controller are available, including glitch filtering. Note that the input change interrupt, the interrupt modes on a programmable event and the read of the pin level require the clock to be validated.

After a hardware reset, the PIO clock is disabled by default.

The user must configure the PMC before any access to the input line information.

31.4.4 Interrupt Generation

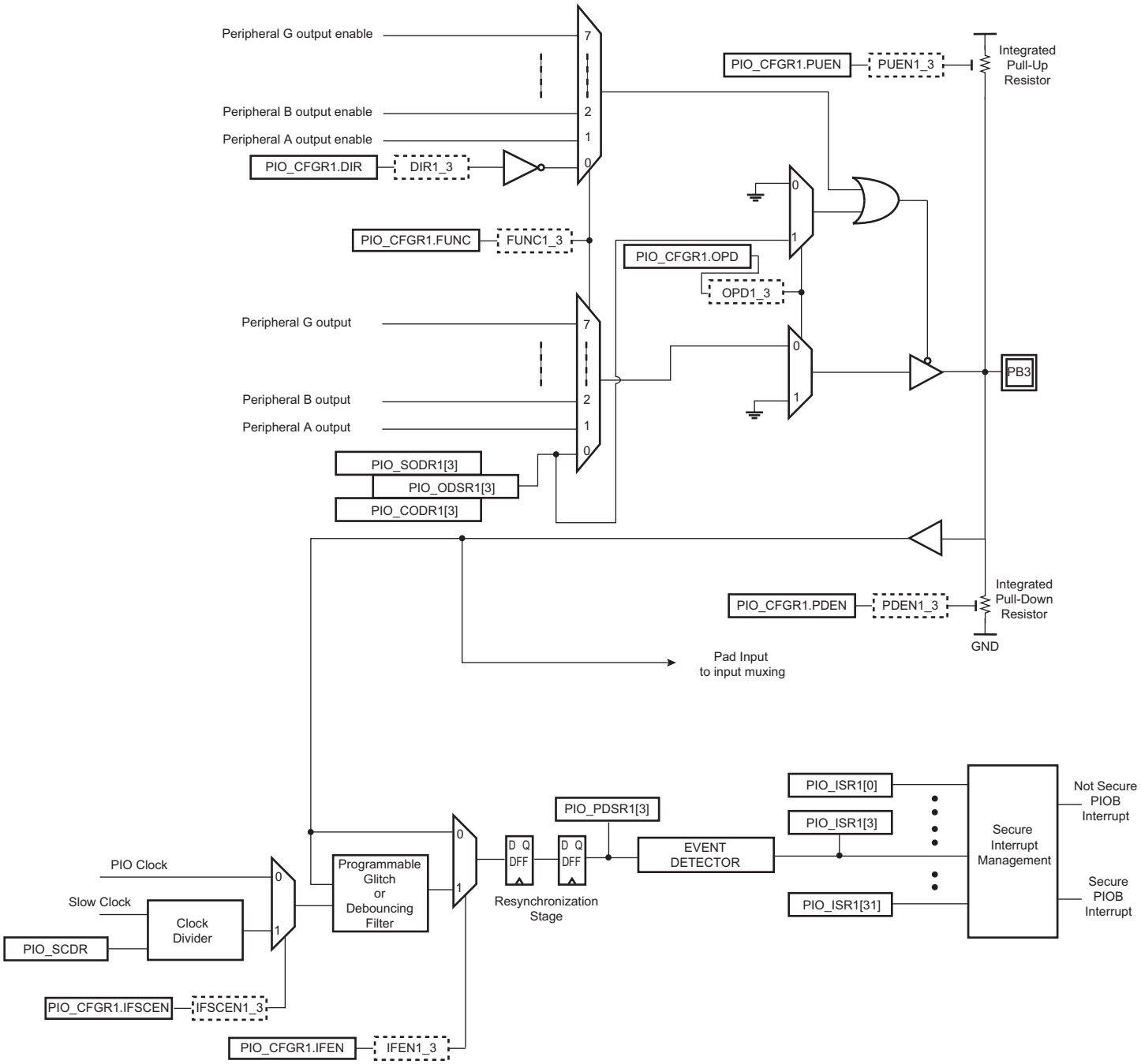
For interrupt handling, the PIO Controllers are considered as user peripherals. This means that the PIO Controller interrupt lines are connected among the interrupt sources. The PIO Controller supply one interrupt signal per I/O group. Refer to the PIO Controller peripheral identifier in the product description to identify the interrupt sources dedicated to the PIO Controller. The PIO Controller can target either the Secure or Non-Secure Interrupt Controller according to security level of the I/O line which triggers the interruption. Using the PIO Controller requires the Interrupt Controller to be programmed first.

The PIO Controller interrupt can be generated only if the PIO Controller clock is enabled.

31.5 Functional Description

The PIO Controller features up to 512 fully-programmable I/O lines. Most of the control logic associated to each I/O is represented in Figure 31-2 where the I/O line 3 of the PIOB (PB3) is described as an example. In this description each signal shown represents one of up to 512 possible indexes.

Figure 31-2. I/O Line Control Logic



31.5.1 I/O Line Configuration Method

The user interface of the PIO Controller provides several sets of registers. Each set of registers interfaces with one I/O group.

Table 31-1. I/O Group List

I/O Group Number	PIO
0	PIOA
1	PIOB
...	...
3	PIOD

31.5.1.1 Security Management

First of all, the user must defined the security level of the I/O line. Each I/O line of each I/O group can be defined as either secure or non-secure lines. Each I/O line of the I/O group x can be set as non-secure I/O line by writing a 1 to the corresponding bit P0–P31 of the Secure PIO Set I/O Non-Secure Register (S_PIO_SIONRx) of the I/O group x.

To define an I/O line of I/O group x as a secure I/O line, write a 1 to the corresponding bit P0–P31 of the "Secure PIO Set I/O Secure Register" (S_PIO_SIOSRx) of the I/O group x.

Examples:

Setting the I/O line PC4 as non-secure line:

Write the value 16 (bit No. 4 at 1) at address 0x10B0 (S_PIO_SIONR2)

Setting the I/O line PB3 as secure line:

Write the value 8 (bit No. 3 at 1) at address 0x1074 (S_PIO_SIOSR1)

The security level of each I/O line is reported by the Secure PIO I/O Security Status Register (S_PIO_IOSSRx) of the corresponding I/O group. Reading 0 at the corresponding bit P0–P31 means that the corresponding I/O line of the I/O group is defined as secure. Reading 1 means that this I/O line of the I/O group is non-secure.

The PIO Controller user interface is divided into two register mapping areas:

- The Non-Secure area, located from address 0x0 to 0x1000, can be accessed by any master (Secure or Non-Secure master). This area interfaces with all the I/O lines defined as non-secure. Trying to access to an I/O line defined as secure through this area will have no effect on I/O line and read values will be 0.
- The Secure area, located above address 0x1000, can only be accessed by a Secure master (if the PIO Controller is defined as secure at the HMATRIX level). This area interfaces with all the I/O lines defined as secure. Trying to access to an I/O line defined as non-secure through this area will have no effect on I/O line and read values will be 0.

31.5.1.2 Programming I/O Line Configuration

The user must first define which I/O line in the group will be targeted by writing a 1 to the corresponding bit in the PIO Mask Register (PIO_MSKRx). Several I/O lines in an I/O group can be configured at the same time by setting the corresponding bits in PIO_MSKRx. Then, writing the PIO Configuration Register (PIO_CFGRx) apply the configuration to the I/O line(s) defined in PIO_MSKRx. All the I/O lines defined as secure in the S_PIO_SIOSRx must be configured by writing the S_PIO_CFGRx and S_PIO_MSKRx registers.

For more details concerning the I/O line configuration using PIO_MSKRx and PIO_CFGRx, see Section 31.6 "I/O Lines Programming Example".

31.5.1.3 Reading I/O line configuration

As for programming operation, reading configuration requires the user to first define which I/O line in the group x will be targeted by writing a 1 to the corresponding bit in the PIO Mask Register (PIO_MSKRx). The value of the targeted I/O line is read in PIO_CFGRx.

If several bits are set in PIO_MSKRx, then the read configuration in PIO_CFGRx is the configuration of the I/O line with the lowest index.

Note that S_PIO_MSKRx and S_PIO_CFGRx must be used to read the configuration of a secure I/O line.

31.5.2 Pull-up and Pull-down Resistor Control

Each I/O line is designed with an embedded pull-up resistor and an embedded pull-down resistor.

The pull-up resistor on the I/O line(s) defined in PIO_MSKRx can be enabled by setting the PUEN bit in PIO_CFGRx. Clearing the PUEN bit in PIO_CFGRx disables the pull-up resistor of I/O lines defined in PIO_MSKRx.

The pull-down resistor on the I/O line(s) defined in PIO_MSKRx can be enabled by setting the PDEN bit in PIO_CFGRx. Clearing the PDEN bit in PIO_CFGRx disables the pull-down resistor of I/O lines defined in PIO_MSKRx.

If both PUEN and PDEN bit are set in PIO_CFGRx, only the pull-up resistor is enabled for I/O line(s) defined in PIO_MSKRx and the PDEN bit is discarded.

Control of the pull-up resistor is possible regardless of the configuration of the I/O line (Input, Output, Open-drain).

Note that S_PIO_MSKRx and S_PIO_CFGRx must be used to program the pull-up or pull-down configuration of a secure I/O line.

For more details concerning Pull-up and Pull-down configuration, see Section 31.7.2 “PIO Configuration Register” or Section 31.7.16 “Secure PIO Configuration Register” for secure I/O line configuration.

The reset value of PUEN and PDEN bits of each I/O line is defined at the product level and depends on the multiplexing of the device.

31.5.3 General Purpose or Peripheral Function Selection

The PIO Controller provides multiplexing of up to 6 peripheral functions on a single pin. The selection is performed by writing the FUNC field in PIO_CFGRx. The selected function is applied to the I/O line(s) defined in PIO_MSKRx.

When FUNC is 0, no peripheral is selected and the General Purpose PIO (GPIO) mode is selected (in this mode, the I/O line is controlled by the PIO Controller).

When FUNC is not 0, the peripheral selected to control the I/O line depends on the FUNC value.

Note that S_PIO_MSKRx and S_PIO_CFGRx must be used to program the FUNC field of a secure I/O line.

For more details, see Section 31.7.2 “PIO Configuration Register” or Section 31.7.16 “Secure PIO Configuration Register” for secure I/O line configuration.

Note that multiplexing of peripheral lines affects both input and output peripheral lines. When a peripheral is not selected on any I/O line, its inputs are assigned with constant default values defined at the product level. The user must ensure that only one I/O line is affected to a peripheral input at a time. For more details concerning input muxing, see Section 31.5.4 “Output Control”.

The reset value of the FUNC field of each I/O line is defined at the product level and depends on the multiplexing of the device.

31.5.4 Output Control

When the I/O line is assigned to a peripheral function, i.e., the corresponding FUNC field of the line configuration is not 0, the drive of the I/O line is controlled by the peripheral. According to the FUNC value, the selected peripheral determines whether the pin is driven or not.

When the FUNC field of a I/O line is 0, then the I/O line is set in General Purpose mode and the I/O line can be configured to be driven by the PIO Controller instead of the peripheral.

If the DIR bit of the I/O line configuration (PIO_CFGRx) is set (OUTPUT) then the I/O line can be driven by the PIO Controller. The level driven on an I/O line can be determined by writing in the PIO Set Output Data Register (PIO_SODRx) and the PIO Clear Output Data Register (PIO_CODRx). These write operations, respectively, set and clear the PIO Output Data Status Register (PIO_ODSRx), which represents the data driven on the I/O lines. Writing PIO_ODSRx directly is possible and only affects the I/O line set to 1 in PIO_MSKRx (see Section 31.5.5 “Synchronous Data Output”).

When the DIR bit of the I/O line configuration is at zero, the corresponding I/O line is used as an input only.

The DIR bit has no effect if the corresponding line is assigned to a peripheral function, but writing the DIR bit is managed whether the pin is configured to be controlled by the PIO Controller or assigned to a peripheral function. This enables configuration of the I/O line prior to setting it to be managed by the PIO Controller.

Similarly, writing in PIO_SODRx and PIO_CODRx affects PIO_ODSRx. This is important as it defines the first level driven on the I/O line.

31.5.5 Synchronous Data Output

Clearing one or more PIO line(s) and setting another one or more PIO line(s) synchronously cannot be done by using PIO_SODRx and PIO_CODRx. It requires two successive write operations into two different registers. To overcome this, the PIO Controller offers a direct control of PIO outputs by single write access to PIO_ODSRx. Only I/O lines set to 1 in PIO_MSKRx are written.

31.5.6 Open-Drain Mode

Each I/O can be independently programmed in Open-Drain mode. This feature permits several drivers to be connected on the I/O line which is driven low only by each device. An external pull-up resistor (or enabling of the internal one) is generally required to guarantee a high level on the line.

The Open-Drain mode is controlled by the OPD bit in the I/O line configuration (PIO_CFGRx). An I/O line is switched in Open-Drain mode by setting the PIO_CFGRx.OPD bit. The Open-Drain mode can be selected if the I/O line is not controlled by a peripheral (the FUNC field must be cleared in PIO_CFGRx).

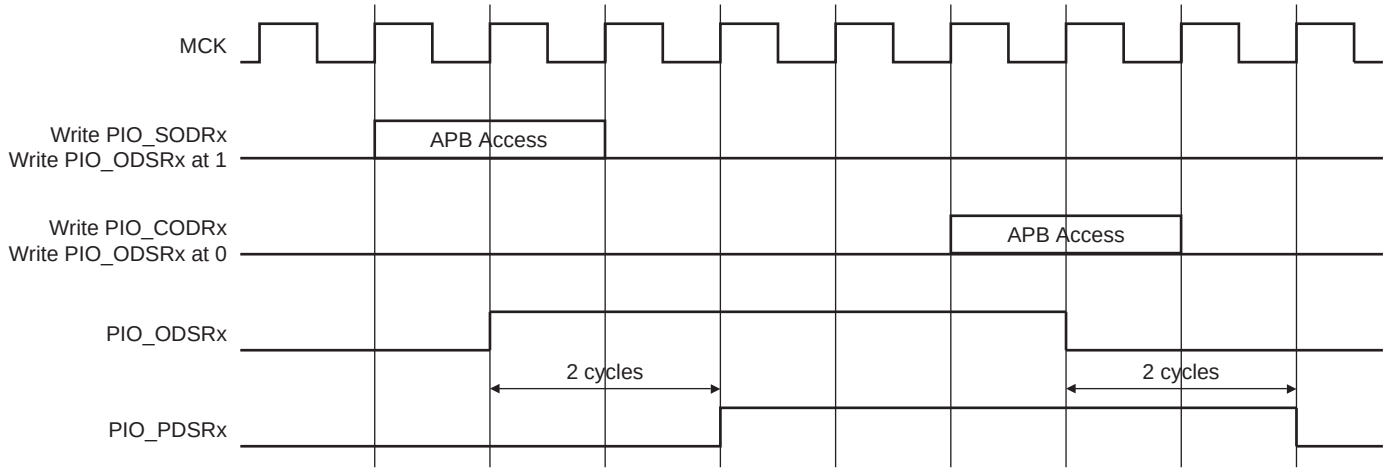
For more details concerning the Open-Drain mode, see Section 31.7.2 “PIO Configuration Register” or Section 31.7.16 “Secure PIO Configuration Register” for secure I/O line configuration.

After reset, the OPD bit of each I/O line is defined at the product level and depends on the multiplexing of the device.

31.5.7 Output Line Timings

Figure 31-3 shows how the outputs are driven either by writing PIO_SODRx or PIO_CODRx, or by directly writing PIO_ODSRx. This last case is valid only if the corresponding bit in PIO_MSKRx is set. Figure 31-3 also shows when the feedback in the Pin Data Status Register (PIO_PDSRx) is available.

Figure 31-3. Output Line Timings



31.5.8 Inputs

The level on each I/O line of the I/O group x can be read through PIO_PDSRx. This register indicates the level of the I/O lines regardless of their configuration, whether uniquely as an input, or driven by the PIO Controller, or driven by a peripheral.

Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSRx reads the levels present on the I/O line at the time the clock was disabled.

31.5.9 Input Glitch and Debouncing Filters

Optional input glitch and debouncing filters are independently programmable on each I/O line.

The glitch filter can filter a glitch with a duration of less than 1/2 master clock (MCK) and the debouncing filter can filter a pulse of less than 1/2 period of a programmable divided slow clock.

The selection between glitch filtering or debounce filtering is done by writing the bit IFSCEN in the PIO Configuration Register (PIO_CFGRx). The selected filtering mode is applied to the I/O line(s) defined in PIO_MSKRx.

- If IFSCEN = 0: The glitch filter can filter a glitch with a duration of less than 1/2 master clock period.
- If IFSCEN = 1: The debouncing filter can filter a pulse with a duration of less than 1/2 programmable divided slow clock period.

For the debouncing filter, the period of the divided slow clock is performed by writing in the DIV field of the Secure PIO Slow Clock Divider Debouncing Register (S_PIO_SCDR): $t_{div_slck} = ((DIV + 1) \times 2) \times t_{slck}$.

When the glitch or debouncing filter is enabled, a glitch or pulse with a duration of less than 1/2 selected clock cycle (selected clock represents MCK or divided slow clock depending on IFSCEN bit programming) is automatically rejected, while a pulse with a duration of one selected clock (MCK or divided slow clock) cycle or more is accepted. For pulse durations between 1/2 selected clock cycle and one selected clock cycle, the pulse may or may not be taken into account, depending on the precise timing of its occurrence. Thus for a pulse to be visible, it must exceed one selected clock cycle, whereas for a glitch to be reliably filtered out, its duration must not exceed 1/2 selected clock cycle.

The filters also introduce some latencies, illustrated in Figure 31-4 and Figure 31-5.

The glitch filter of each I/O lines is controlled by the IFEN bit of the PIO Configuration Register (PIO_CFGRx). Setting the PIO_CFGRx.IFEN bit enables the glitch filter of the I/O line(s) defined in PIO_MSKRx.

When the glitch and/or debouncing filter is enabled, it does not modify the behavior of the inputs on the peripherals. It acts only on the value read in PIO_PDSRx and on the input change interrupt detection. The glitch and debouncing filters require that the PIO Controller clock is enabled.

Figure 31-4. Input Glitch Filter Timing

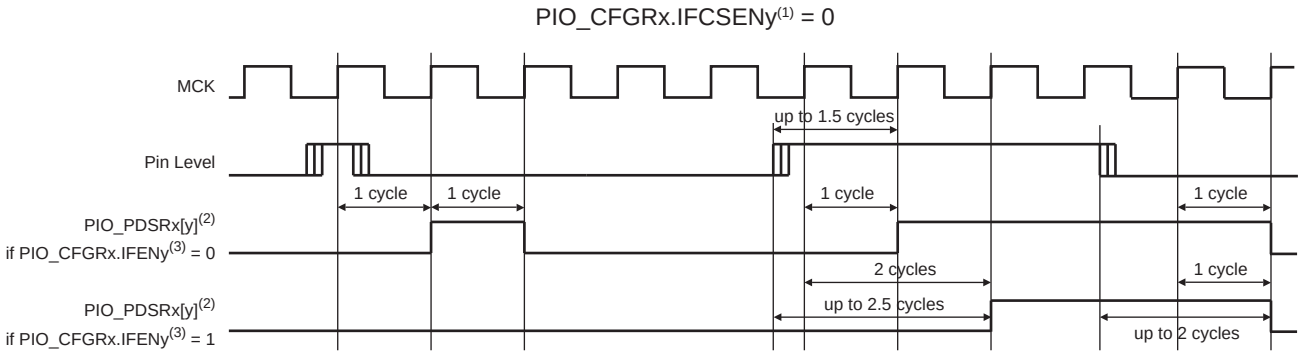
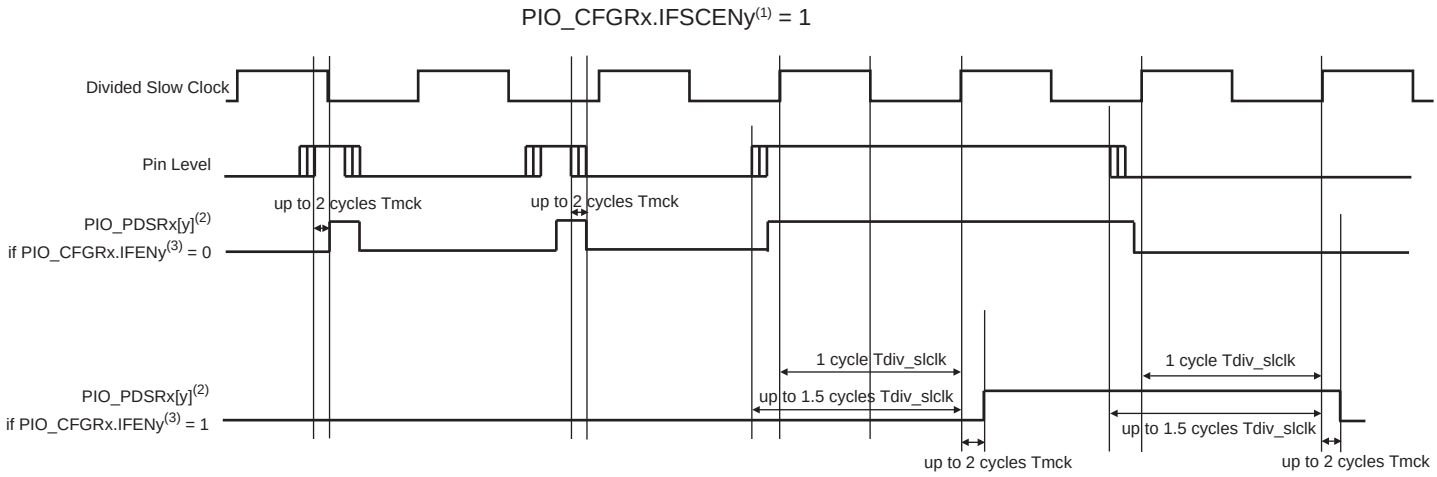


Figure 31-5. Input Debouncing Filter Timing



Notes:
1. Means IFCSEN bit of the I/O line y of the I/O group x
2. Means PIO Data Status value of the I/O line y of the I/O group x
3. Means IFEN bit of the I/O line y of the I/O group x

31.5.10 Input Edge/Level Interrupt

Each I/O group can be programmed to generate an interrupt when it detects an edge or a level on an I/O line. The Input Edge/Level interrupts are controlled by writing the PIO Interrupt Enable Register (PIO_IERx) and the PIO Interrupt Disable Register (PIO_IDRx), which enable and disable the input change interrupt respectively by setting and clearing the corresponding bit in the PIO Interrupt Mask Register (PIO_IMRx). For the Secure I/O lines, the Input Edge/Level interrupts are controlled by writing S_PIO_IERx and S_PIO_IDRx, which enable and disable input change interrupts respectively by setting and clearing the corresponding bit in the S_PIO_IMRx. As input change detection is possible only by comparing two successive samplings of the input of the I/O line, the PIO Controller clock must be enabled. The Input Change interrupt is available regardless of the configuration of the I/O line, i.e., configured as an input only, controlled by the PIO Controller or assigned to a peripheral function.

Each I/O group can generate a Non-Secure interrupt and a Secure interrupt according to the security level of the I/O line which triggers the interrupt.

According to the EVTSEL field value in the PIO Configuration Register (PIO_CFGRx) or the Secure PIO Configuration Register (S_PIO_CFGRx) in case of a Secure I/O line, the interrupt signal of the I/O group x can be generated on the following occurrence:

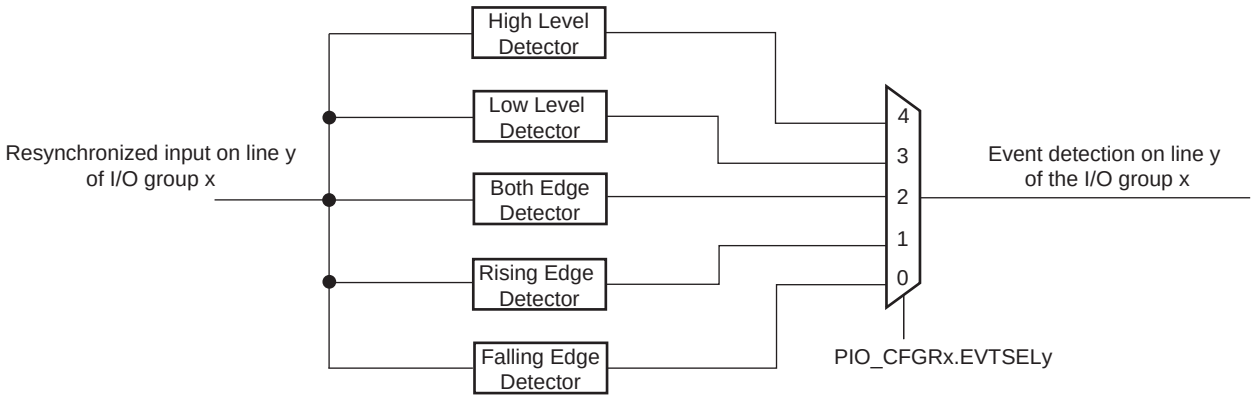
- (S_)PIO_CFGRx.EVTSELy = 0: The interrupt signal of the I/O group x is generated on the I/O line y falling edge detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 1: The interrupt signal of the I/O group x is generated on the I/O line y rising edge detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 2: The interrupt signal of the I/O group x is generated on the I/O line y both rising and falling edge detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 3: The interrupt signal of the I/O group x is generated on the I/O line y low level detection (assuming that (S_)PIO_IMRx[y] = 1).
- (S_)PIO_CFGRx.EVTSELy = 4: The interrupt signal of the I/O group x is generated on the I/O line y high level detection (assuming that (S_)PIO_IMRx[y] = 1).

By default, the interrupt can be generated at any time a falling edge is detected on the input.

When an input edge or level is detected on an I/O line, the corresponding bit in the PIO Interrupt Status Register (PIO_ISRx), or in the Secure PIO Interrupt Status Register (S_PIO_ISRx) if the I/O line is Secure, is set. For a Non-Secure I/O line, if the corresponding bit in PIO_IMRx is set, the PIO Controller Non-Secure interrupt line of the I/O group x is asserted. For a Secure I/O line, if the corresponding bit in S_PIO_IMRx is set, the PIO Controller Secure interrupt line of the I/O group x is asserted.

When the software reads PIO_ISRx, all the Non-Secure interrupts of the I/O group x are automatically cleared. When the software reads S_PIO_ISRx, all the Secure interrupts of the I/O group x are automatically cleared. This signifies that all the interrupts that are pending when PIO_ISRx or S_PIO_ISRx is read must be handled. When an Interrupt is enabled on a “level”, the interrupt is generated as long as the interrupt source is not cleared, even if some read accesses in PIO_ISRx or S_PIO_ISRx are performed.

Figure 31-6. Event Detector on Input Lines



Example of interrupt generation on following lines:

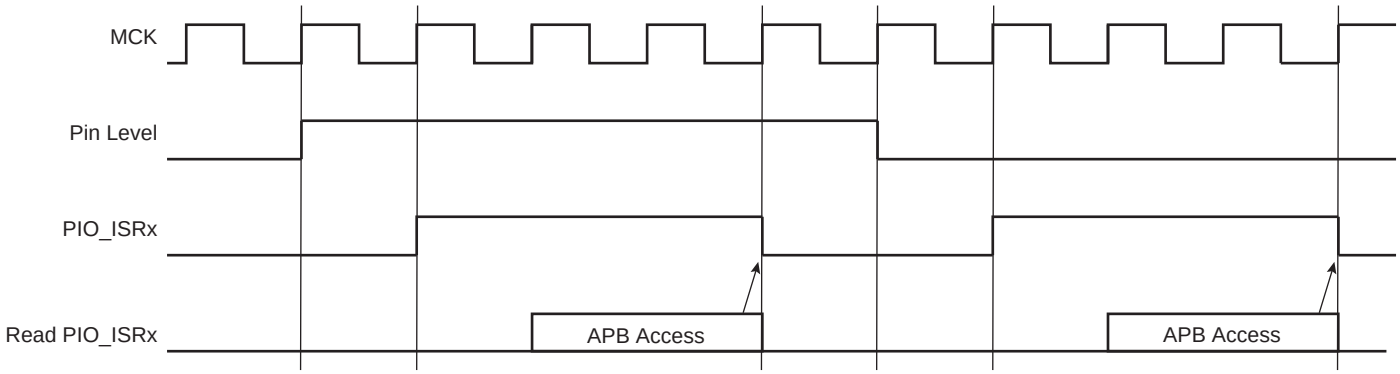
- Rising edge on the Secure PIO line 0 of the I/O group 0 (PIOA)
- Low-level edge on the Secure PIO line 1of the I/O group 0 (PIOA)
- Rising edge on the Secure PIO line 2 of the I/O group 0 (PIOA)
- High-level on the Secure PIO line 3 of the I/O group 0 (PIOA)
- Low-level on the Non-Secure PIO line 4 of the I/O group 0 (PIOA)
- High-level on the Secure PIO line 0 of the I/O group 1 (PIOB)
- Falling edge on the Secure PIO line 1 of the I/O group 1 (PIOB)
- Rising edge on the Secure PIO line 2 of the I/O group 1 (PIOB)
- Any edge on the other Non-Secure lines of the I/O group 1 (PIOB)

Table 31-2 details the required configuration for this example.

Table 31-2. Configuration for Example Interrupt Generation

Configuration	Name
PIOA: I/O line security level	Define the I/O lines 0 to 3 of the PIOA as Secure by writing 32'h0000_000F in the S_PIO_SIOSR0 (offset 0x1034) Define the I/O lines 4 of the PIOA as Non-Secure by writing 32'h0000_0010 in the S_PIO_SIONR0 (offset 0x1030)
PIOA: Interrupt Mode	Enable interrupt sources for lines 0 to 3 of PIOA by writing 32'h0000_000F in S_PIO_IER0 (offset 0x1020) Enable interrupt source for the line 4 of PIOA by writing 32'h0000_0010 in PIO_IER0 (offset 0x20)
PIOA: Event Selection	Configure Rising Edge detection for Secure lines 0 and 2: Write 32'h0000_0005 in S_PIO_MSKR0 (offset 0x1000) Write 32'h0100_0000 in S_PIO_CFGR0 (offset 0x1004)
	Configure Low Level detection for Secure line 1: Write 32'h0000_0002 in S_PIO_MSKR0 (offset 0x1000) Write 32'h0300_0000 in S_PIO_CFGR0 (offset 0x1004)
	Configure High Level detection for Secure line 3: Write 32'h0000_0008 in S_PIO_MSKR0 (offset 0x1000) Write 32'h0400_0000 in S_PIO_CFGR0 (offset 0x1004)
	Configure Low Level detection for Non-Secure line 4: Write 32'h0000_0010 in PIO_MSKR0 (offset 0x0) Write 32'h0300_0000 in PIO_CFGR0 (offset 0x4)
PIOB: I/O line security level	Define the I/O lines 0 to 2 of the PIOB as Secure by writing 32'h0000_0007 in the S_PIO_SIOSR1 (offset 0x1074) Define the other I/O lines of the PIOB as Non-Secure by writing 32'hFFFF_FFF8 in the S_PIO_SIONR1 (offset 0x1070)
PIOB: Interrupt Mode	Enable interrupt sources for lines 0 to 2 of PIOB by writing 32'h0000_0007 in S_PIO_IER1 (offset 0x1060) Enable interrupt sources for all other lines of PIOB by writing 32'hFFFF_FFF8 in PIO_IER1 (offset 0x60)
PIOB: Event Selection	Configure High Level detection for Secure line 0: Write 32'h0000_0001 in S_PIO_MSKR1 (offset 0x1040) Write 32'h0400_0000 in S_PIO_CFGR1 (offset 0x1044)
	Configure Falling Edge detection for Secure line 1: Write 32'h0000_0002 in S_PIO_MSKR1 (offset 0x1040) Write 32'h0000_0000 in S_PIO_CFGR1 (offset 0x1044)
	Configure Rising Edge detection for Secure line 2: Write 32'h0000_0004 in S_PIO_MSKR1 (offset 0x1040) Write 32'h0100_000 in S_PIO_CFGR1 (offset 0x1044)
	Configure Low Level detection for Non-Secure lines: Write 32'hFFFF_FFF8 in PIO_MSKR1 (offset 0x40) Write 32'h0200_000 in PIO_CFGR1 (offset 0x44)

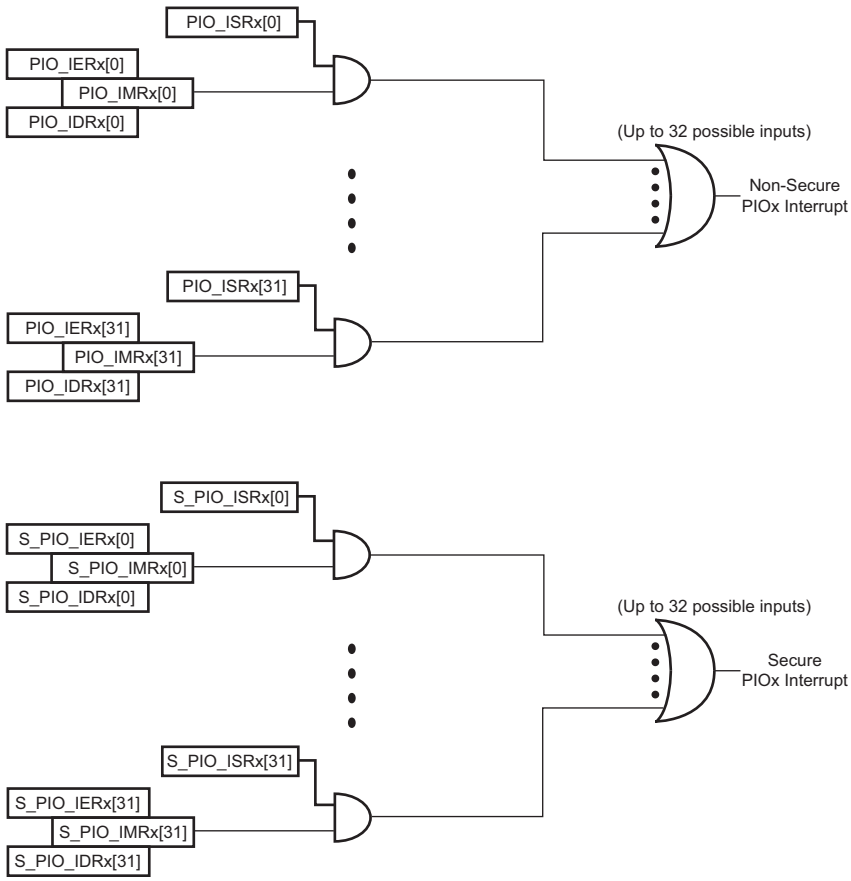
Figure 31-7. Input Change Interrupt Timings When No Additional Interrupt Modes



31.5.11 Interrupt Management

The PIO Controller can drive one secure interrupt signal and one non-secure interrupt signal per I/O group (see Figure 31-1). Secure interrupt signals are connected to the secure interrupt controller of the system. Non-secure interrupt signals are connected to the non-secure interrupt controller of the system.

Figure 31-8. PIO Interrupt Management



31.5.12 I/O Lines Lock

When an I/O line is controlled by a peripheral (particularly the Pulse Width Modulation Controller PWM), it can become locked by the action of this peripheral via an input of the PIO Controller. When an I/O line is locked, the following fields in PIO_CFGRx are locked and cannot be modified:

- FUNC: Peripheral selection cannot be changed when the corresponding I/O line is locked.
- PUEN: Pull-Up configuration cannot be changed when the corresponding I/O line is locked.
- PDEN: Pull-Down configuration cannot be changed when the corresponding I/O line is locked.
- OPD: Open Drain configuration cannot be changed when the corresponding I/O line is locked.

Writing to one of these fields while the corresponding I/O line is locked will have no effect.

The user can know at anytime which I/O line is locked by reading the PIO Lock Status Register (PIO_LOCKSR) or Secure PIO Lock Status Register (S_PIO_LOCKSR) for locked Secure I/O lines. Once an I/O line is locked, the only way to unlock it is to apply a hardware reset to the PIO Controller.

31.5.13 Programmable I/O Drive

It is possible to configure the I/O drive for pads PA0 to PD31. The I/O drive of the pad can be programmed by writing the DRVSTR field in the PIO Configuration Register (PIO_CFGRx) if the corresponding line is Non-Secure or the Secure PIO Configuration Register (S_PIO_CFGRx) if the I/O line is Secure. For any details, refer to the product electrical characteristics.

31.5.14 Programmable Schmitt Trigger

It is possible to configure each input for the Schmitt trigger. The Schmitt trigger can be enabled by setting the SCHMITT bit of the PIO Configuration Register (PIO_CFGRx) if the corresponding line is Non-Secure or the Secure PIO Configuration Register (S_PIO_CFGRx) if the I/O line is Secure. By default the Schmitt trigger is active. Disabling the Schmitt trigger is requested when using the QTouch® Library.

31.5.15 I/O Line Configuration Freeze

31.5.15.1 Software Freeze

Once the I/O line configuration is done, it can be frozen by using the PIO I/O Freeze Configuration Register (PIO_IOFRx) of the corresponding group or the Secure PIO I/O Freeze Configuration Register (S_PIO_IOFRx) if the I/O line is Secure.

Physical Freeze

Setting the FPHY bit of PIO_IOFRx freezes the following fields of the Non-Secure I/O lines defined in PIO_MSKRx:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

For Secure I/O lines, use the FPHY bit of the S_PIO_IOFRx and the S_PIO_MSKRx to freeze the fields above.

When the physical freeze is currently active on an I/O line, the PCFS flag is set when reading the PIO_CFGRx of the I/O line (or the S_PIO_CFGRx if the I/O line is Secure).

Only a hardware reset can release fields listed above.

Interrupt Freeze

Setting the FINT bit of the PIO_IOFRx will freeze the following fields of the Non-Secure I/O lines defined in the PIO_MSKRx:

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

For Secure I/O lines, use the FINT bit of the S_PIO_IOFRx and the S_PIO_MSKRx to freeze the fields above.

When the “Interrupt Freeze” is currently active on an I/O line, the ICFS flag is set when reading the PIO_CFGRx of the I/O line (or the S_PIO_CFGRx if the I/O line is Secure).

Only a hardware reset can release fields listed above.

31.5.16 Register Write Protection

To prevent any single software error from corrupting PIO behavior, certain registers in the address space can be write-protected by setting bit WPEN in the PIO Write Protection Mode Register (PIO_WPMR) or the Secure PIO Write Protection Mode Register (S_PIO_WPMR).

If a write access to a Non-Secure write-protected register is detected, the WPVS flag in the PIO Write Protection Status Register (PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

If a write access to a Secure write-protected register is detected, the WPVS flag in the Secure PIO Write Protection Status Register (S_PIO_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The respective WPVS bit is automatically cleared after reading the PIO_WPSR or S_PIO_WPSR.

The following registers are write-protected when WPEN is set in PIO_WPMR:

- PIO Mask Register
- PIO Configuration Register

The following registers are write-protected when WPEN is set in S_PIO_WPMR:

- Secure PIO Mask Register
- Secure PIO Configuration Register
- Secure PIO Slow Clock Divider Debouncing Register

31.6 I/O Lines Programming Example

The programming example shown in Table 31-3 is used to obtain the following configurations:

- PIOA Configuration:
  - 4-bit output port on Secure I/O lines 0 to 3, open-drain, with pull-up resistor
  - Four output signals on Non-Secure I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor
  - Secure I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor
  - Non-Secure I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor
- PIOB Configuration:
  - Four input signals on Secure I/O lines 0 to 3 (to read push-button states for example), with pull-up resistors, glitch filters and input change interrupts
  - Four input signals on Non-Secure I/O lines 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter
  - Secure I/O lines 16 to 23 assigned to peripheral B functions with pull-down resistor
  - Non-Secure I/O lines 24 to 27 assigned to peripheral D with Input Change Interrupt, no pull-up resistor and no pull-down resistor

Table 31-3. Programming Example

Action	Register	Value to be Written
PIOA: Set I/O lines 0 to 3 and 16 to 19 as Secure	S_PIO_SIOSR0 (offset 0x1034)	0x000F_000F
PIOA: Set I/O lines 4 to 7 and 20 to 23 as Non-Secure	S_PIO_SIONR0 (offset 0x1030)	0x00F0_00F0
PIOA: 4-bit output port on Secure I/O lines 0 to 3, open-drain, with pull-up resistor	S_PIO_MSKR0 (offset 0x1000)	0x0000_000F
	S_PIO_CFGR0 (offset 0x1004)	0x0000_4300
PIOA: Four output signals on Non-Secure I/O lines 4 to 7 (to drive LEDs for example), driven high and low, no pull-up resistor, no pull-down resistor	PIO_MSKR0 (offset 0x0)	0x0000_00F0
	PIO_CFGR0 (offset 0x4)	0x0000_0100
PIOA: Secure I/O lines 16 to 19 assigned to peripheral A functions with pull-up resistor	S_PIO_MSKR0 (offset 0x1000)	0x000F_0000
	S_PIO_CFGR0 (offset 0x1004)	0x0000_0201
PIOA: Non-Secure I/O lines 20 to 23 assigned to peripheral B functions with pull-down resistor	PIO_MSKR0 (offset 0x0)	0x00F0_0000
	PIO_CFGR0 (offset 0x4)	0x0000_0402
PIOB: Set I/O lines 0 to 3 and 16 to 23as Secure	S_PIO_SIOSR1 (offset 0x1074)	0x00FF_000F
PIOB: Set I/O lines 12 to 15 and 24 to 27 as Non-Secure	S_PIO_SIONR1 (offset 0x1070)	0x0F00_F000

Table 31-3. Programming Example (Continued)

Action	Register	Value to be Written
PIOB: Four input signals on Secure I/O lines 0 to 3 (to read push-button states for example), with pull-up resistors, glitch filters and interrupts on rising edge	S_PIO_MSKR1 (offset 0x1040)	0x0000_000F
	S_PIO_CFGR1 (offset 0x1044)	0x0100_1200
PIOB: Four input signals on Non-Secure I/O line 12 to 15 to read an external device status (polled, thus no input change interrupt), no pull-up resistor, no glitch filter	PIO_MSKR1 (offset 0x40)	0x0000_F000
	PIO_CFGR1 (offset 0x44)	0x0100_1200
PIOB: Secure I/O lines 16 to 23 assigned to peripheral B functions with pull-down resistor	S_PIO_MSKR1 (offset 0x1040)	0x00FF_0000
	S_PIO_CFGR1 (offset 0x1044)	0x0000_0402
PIOB: Non-Secure I/O line 24 to 27 assigned to peripheral D with Input Interrupt on both edges, no pull-up resistor and no pull-down resistor	PIO_MSKR1 (offset 0x40)	0x0F00_0000
	PIO_CFGR1 (offset 0x44)	0x0200_0004
PIOB: Enable interrupt	S_PIO_IER1 (offset 0x1060)	0x0000_000F
	PIO_IER1 (offset 0x60)	0x0F00_0000

31.7 Parallel Input/Output Controller (PIO) User Interface

Each I/O line controlled by the PIO Controller is associated with a bit in each of the PIO Controller User Interface registers. Each register is 32 bits wide. If a parallel I/O line is not defined, writing to the corresponding bits has no effect. Undefined bits read zero.

Table 31-4. Register Mapping

Offset[3][4]	Register	Name	Access	Reset
0x000 + (io_group * 0x40) + 0x00	PIO Mask Register	PIO_MSKR	Read/Write	0x00000000
0x000 + (io_group * 0x40) + 0x04	PIO Configuration Register	PIO_CFGR	Read/Write	0x00000200
0x000 + (io_group * 0x40) + 0x08	PIO Pin Data Status Register	PIO_PDSR	Read-only	[1]
0x000 + (io_group * 0x40) + 0x0C	PIO Lock Status Register	PIO_LOCKSR	Read-only	0x00000000
0x000 + (io_group * 0x40) + 0x10	PIO Set Output Data Register	PIO_SODR	Write-only	–
0x000 + (io_group * 0x40) + 0x14	PIO Clear Output Data Register	PIO_CODR	Write-only	–
0x000 + (io_group * 0x40) + 0x18	PIO Output Data Status Register	PIO_ODSR	Read/Write	0x00000000
0x000 + (io_group * 0x40) + 0x1C	Reserved	–	–	–
0x000 + (io_group * 0x40) + 0x20	PIO Interrupt Enable Register	PIO_IER	Write-only	–
0x000 + (io_group * 0x40) + 0x24	PIO Interrupt Disable Register	PIO_IDR	Write-only	–
0x000 + (io_group * 0x40) + 0x28	PIO Interrupt Mask Register	PIO_IMR	Read-only	0x00000000
0x000 + (io_group * 0x40) + 0x2C	PIO Interrupt Status Register[2]	PIO_ISR	Read-only	0x00000000
0x000 + (io_group * 0x40) + 0x30	Reserved	–	–	–
0x000 + (io_group * 0x40) + 0x34	Reserved	–	–	–
0x000 + (io_group * 0x40) + 0x38	Reserved	–	–	–
0x000 + (io_group * 0x40) + 0x3C	PIO I/O Freeze Configuration Register	PIO_IOFR	Write-only	–
0x400–0x4FC	Reserved	–	–	–
0x500	Reserved	–	–	–
0x5D0	Reserved	–	–	–
0x5D4	Reserved	–	–	–
0x5E0	PIO Write Protection Mode Register	PIO_WPMR	Read/Write	0x00000000
0x5E4	PIO Write Protection Status Register	PIO_WPSR	Read-only	0x00000000
0x5E8–0x5FC	Reserved	–	–	–
0x1000 + (io_group * 0x40) + 0x00	Secure PIO Mask Register	S_PIO_MSKR	Read/Write	0x00000000

Table 31-4. Register Mapping (Continued)

Offset[3][4]	Register	Name	Access	Reset
0x1000 + (io_group * 0x40) + 0x04	Secure PIO Configuration Register	S_PIO_CFGR	Read/Write	0x00000200
0x1000 + (io_group * 0x40) + 0x08	Secure PIO Pin Data Status Register	S_PIO_PDSR	Read-only	[1]
0x1000 + (io_group * 0x40) + 0x0C	Secure PIO Lock Status Register	S_PIO_LOCKSR	Read-only	0x00000000
0x1000 + (io_group * 0x40) + 0x10	Secure PIO Set Output Data Register	S_PIO_SODR	Write-only	–
0x1000 + (io_group * 0x40) + 0x14	Secure PIO Clear Output Data Register	S_PIO_CODR	Write-only	–
0x1000 + (io_group * 0x40) +0x18	Secure PIO Output Data Status Register	S_PIO_ODSR	Read/Write	0x00000000
0x1000 + (io_group * 0x40) + 0x1C	Reserved	–	–	–
0x1000 + (io_group * 0x40) + 0x20	Secure PIO Interrupt Enable Register	S_PIO_IER	Write-only	–
0x1000 + (io_group * 0x40) + 0x24	Secure PIO Interrupt Disable Register	S_PIO_IDR	Write-only	–
0x1000 + (io_group * 0x40) + 0x28	Secure PIO Interrupt Mask Register	S_PIO_IMR	Read-only	0x00000000
0x1000 + (io_group * 0x40) + 0x2C	Secure PIO Interrupt Status Register[2]	S_PIO_ISR	Read-only	0x00000000
0x1000 + (io_group * 0x40) + 0x30	Secure PIO Set I/O Non-Secure Register	S_PIO_SIONR	Write-only	–
0x1000 + (io_group * 0x40) + 0x34	Secure PIO Set I/O Secure Register	S_PIO_SIOSR	Write-only	–
0x1000 + (io_group * 0x40) + 0x38	Secure PIO I/O Security Status Register	S_PIO_IOSSR	Read-only	0x00000000
0x1000 + (io_group * 0x40) + 0x3C	Secure PIO I/O Freeze Configuration Register	S_PIO_IOFR	Write-only	–
0x1400–0x14FC	Reserved	–	–	–
0x1500	Secure PIO Slow Clock Divider Debouncing Register	S_PIO_SCDR	Read/Write	0x00000000
0x15D0	Reserved	–	–	–
0x15D4	Reserved	–	–	–
0x15E0	Secure PIO Write Protection Mode Register	S_PIO_WPMR	Read/Write	0x00000000
0x15E4	Secure PIO Write Protection Status Register	S_PIO_WPSR	Read-only	0x00000000

Notes:
1. Reset value of PIO_PDSR and S_PIO_PDSR depend on the level of the I/O lines. Reading the I/O line levels requires the clock of the PIO Controller to be enabled, otherwise PIO_PDSR reads the levels present on the I/O line at the time the clock was disabled.
2. PIO_ISR and S_PIO_ISR are reset at 0x0. However, the first read of the register may read a different value as input changes may have occurred.
3. If an offset is not listed in the table it must be considered as reserved.
4. Some registers are indexed with “io_group” index ranging from 0 to 3.

31.7.1 PIO Mask Register

Name: PIO_MSKRx [x=0..3]

Address: 0xFC038000 [0], 0xFC038040 [1], 0xFC038080 [2], 0xFC0380C0 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
23	22	21	20	19	18	17	16
MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
15	14	13	12	11	10	9	8
MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
7	6	5	4	3	2	1	0
MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

- **MSKy: PIO Line y Mask**

These bits define the I/O lines to be configured when writing the PIO Configuration Register.

0 (DISABLED): Writing the PIO_CFGRx, PIO_ODSRx or PIO_IOFRx does not affect the corresponding I/O line configuration.

1 (ENABLED): Writing the PIO_CFGRx, PIO_ODSRx or PIO_IOFRx updates the corresponding I/O line configuration.

31.7.2 PIO Configuration Register

Name: PIO_CFGRx [x=0..3]

Address: 0xFC038004 [0], 0xFC038044 [1], 0xFC038084 [2], 0xFC0380C4 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	ICFS	PCFS	–	EVTSEL			

23	22	21	20	19	18	17	16
–	–	–	–	–	–	DRVSTR	

15	14	13	12	11	10	9	8
SCHMITT	OPD	IFSCEN	IFEN	–	PDEN	PUEN	DIR

7	6	5	4	3	2	1	0
–	–	–	–	–	FUNC		

This register can only be written if the WPEN bit is cleared in the PIO Write Protection Mode Register.

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

- **FUNC: I/O Line Function**

This field defines the function for I/O lines of the I/O group x according to the PIO Mask Register.

Value	Name	Description
0	GPIO	Select the PIO mode for the selected I/O lines.
1	PERIPH_A	Select the peripheral A for the selected I/O lines.
2	PERIPH_B	Select the peripheral B for the selected I/O lines.
3	PERIPH_C	Select the peripheral C for the selected I/O lines.
4	PERIPH_D	Select the peripheral D for the selected I/O lines.
5	PERIPH_E	Select the peripheral E for the selected I/O lines.
6	PERIPH_F	Select the peripheral F for the selected I/O lines.
7	PERIPH_G	Select the peripheral G for the selected I/O lines.

- **DIR: Direction**

This bit defines the direction of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (INPUT): The selected I/O lines are pure inputs.

1 (OUTPUT): The selected I/O lines are enabled in output.

- **PUEN: Pull-Up Enable**

This bit defines the pull-up configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): Pull-Up is disabled for the selected I/O lines.

1 (ENABLED): Pull-Up is enabled for the selected I/O lines.

- **PDEN: Pull-Down Enable**

This bit defines the pull-down configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): Pull-Down is disabled for the selected I/O lines.

1 (ENABLED): Pull-Down is enabled for the selected I/O lines only if PUEN is 0[(1)].

Note: 1. PDEN can be written to 1 only if PUEN is written to 0.

- **IFEN: Input Filter Enable**

This bit defines if the glitch filtering is used for the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): The input filter is disabled for the selected I/O lines.

1 (ENABLED): The input filter is enabled for the selected I/O lines.

- **IFSCEN: Input Filter Slow Clock Enable**

This bit defines the clock source of the glitch filtering for the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): The glitch filter is able to filter glitches with a duration < $t_{mck}/2$ for the selected I/O lines.

1 (ENABLED): The debouncing filter is able to filter pulses with a duration < $t_{div_slck}/2$ for the selected I/O lines.

- **OPD: Open-Drain**

This bit defines the open drain configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (DISABLED): The open-drain is disabled for the selected I/O lines. I/O lines are driven at high- and low-level.

1 (ENABLED): The open-drain is enabled for the selected I/O lines. I/O lines are driven at low-level only.

- **SCHMITT: Schmitt Trigger**

This bit defines the Schmitt trigger configuration of the I/O lines of the I/O group x according to the PIO Mask Register.

0 (ENABLED): Schmitt trigger is enabled for the selected I/O lines.

1 (DISABLED): Schmitt trigger is disabled for the selected I/O lines.

- **DRVSTR: Drive Strength**

This field defines the drive strength of the I/O lines of the I/O group x according to the PIO Mask Register.

Value	Name	Description
0	LO	Low drive
1	LO	Low drive
2	ME	Medium drive
3	HI	High drive

- **EVTSEL: Event Selection**

This field defines the type of event to detect on the I/O lines of the I/O group x according to the PIO Mask Register.

Value	Name	Description
0	FALLING	Event detection on input falling edge
1	RISING	Event detection on input rising edge
2	BOTH	Event detection on input both edge
3	LOW	Event detection on low level input
4	HIGH	Event detection on high level input
5	–	Reserved
6	–	Reserved
7	–	Reserved

- **PCFS: Physical Configuration Freeze Status (read-only)**

This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

- **ICFS: Interrupt Configuration Freeze Status (read-only)**

This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

31.7.3 PIO Pin Data Status Register

Name: PIO_PDSRx [x=0..3]

Address: 0xFC038008 [0], 0xFC038048 [1], 0xFC038088 [2], 0xFC0380C8 [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Data Status**

0: The I/O line of the I/O group x is at level 0.

1: The I/O line of the I/O group x is at level 1.

31.7.4 PIO Lock Status Register

Name: PIO_LOCKSRx [x=0..3]

Address: 0xFC03800C [0], 0xFC03804C [1], 0xFC03808C [2], 0xFC0380CC [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Lock Status**

0: The I/O line of the I/O group x is not locked.

1: The I/O line of the I/O group x is locked.

31.7.5 PIO Set Output Data Register

Name: PIO_SODRx [x=0..3]

Address: 0xFC038010 [0], 0xFC038050 [1], 0xFC038090 [2], 0xFC0380D0 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Set Output Data**

0: No effect.

1: Sets the data to be driven on the I/O line of I/O group x.

31.7.6 PIO Clear Output Data Register

Name: PIO_CODRx [x=0..3]

Address: 0xFC038014 [0], 0xFC038054 [1], 0xFC038094 [2], 0xFC0380D4 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Clear Output Data**

0: No effect.

1: Clears the data to be driven on the I/O line of the I/O group x.

31.7.7 PIO Output Data Status Register

Name: PIO_ODSRx [x=0..3]

Address: 0xFC038018 [0], 0xFC038058 [1], 0xFC038098 [2], 0xFC0380D8 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

- **P0–P31: Output Data Status**

0: The data to be driven on the I/O line of the I/O group x is 0.

1: The data to be driven on the I/O line of the I/O group x is 1.

31.7.8 PIO Interrupt Enable Register

Name: PIO_IERx [x=0..3]

Address: 0xFC038020 [0], 0xFC038060 [1], 0xFC0380A0 [2], 0xFC0380E0 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Enable**

0: No effect.

1: Enables the Input Change interrupt on the I/O line of the I/O group x.

31.7.9 PIO Interrupt Disable Register

Name: PIO_IDRx [x=0..3]

Address: 0xFC038024 [0], 0xFC038064 [1], 0xFC0380A4 [2], 0xFC0380E4 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Disable**

0: No effect.

1: Disables the Input Change interrupt on the I/O line of the I/O group x.

31.7.10 PIO Interrupt Mask Register

Name: PIO_IMRx [x=0..3]

Address: 0xFC038028 [0], 0xFC038068 [1], 0xFC0380A8 [2], 0xFC0380E8 [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Mask**

0: Input Change interrupt is disabled on the I/O line of the I/O group x.

1: Input Change interrupt is enabled on the I/O line of the I/O group x.

31.7.11 PIO Interrupt Status Register

Name: PIO_ISRx [x=0..3]

Address: 0xFC03802C [0], 0xFC03806C [1], 0xFC0380AC [2], 0xFC0380EC [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Status**

0: No Input Change has been detected on the I/O line of the I/O group x since PIO_ISRx was last read or since reset.

1: At least one Input Change has been detected on the I/O line of the I/O group since PIO_ISRx was last read or since reset.

31.7.12 PIO I/O Freeze Configuration Register

Name: PIO_IOFRx [x=0..3]

Address: 0xFC03803C [0], 0xFC03807C [1], 0xFC0380BC [2], 0xFC0380FC [3]

Access: Write-only

31	30	29	28	27	26	25	24
FRZKEY							
23	22	21	20	19	18	17	16
FRZKEY							
15	14	13	12	11	10	9	8
FRZKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	FINT	FPHY

Writing this register will only affect I/O lines enabled in the PIO_MSKRx.

- **FPHY: Freeze Physical Configuration**

0: No effect.

1: Freezes the following configuration fields of Non-Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

Only a hardware reset can reset the FPHY bit.

- **FINT: Freeze Interrupt Configuration**

0: No effect.

1: Freezes the following configuration fields of Non-Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

Only a hardware reset can reset the FINT bit.

- **FRZKEY: Freeze Key**

Value	Name	Description
0x494F46	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit.

31.7.13 PIO Write Protection Mode Register

Name: PIO_WPMR

Address: 0xFC0385E0

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII).

See Section 31.5.16 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x50494F	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

31.7.14 PIO Write Protection Status Register

Name: PIO_WPSR

Address: 0xFC0385E4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
WPVSRC							
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the PIO_WPSR.

1: A write protection violation has occurred since the last read of the PIO_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

31.7.15 Secure PIO Mask Register

Name: S_PIO_MSKRx [x=0..3]

Address: 0xFC039000 [0], 0xFC039040 [1], 0xFC039080 [2], 0xFC0390C0 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
MSK31	MSK30	MSK29	MSK28	MSK27	MSK26	MSK25	MSK24
23	22	21	20	19	18	17	16
MSK23	MSK22	MSK21	MSK20	MSK19	MSK18	MSK17	MSK16
15	14	13	12	11	10	9	8
MSK15	MSK14	MSK13	MSK12	MSK11	MSK10	MSK9	MSK8
7	6	5	4	3	2	1	0
MSK7	MSK6	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0

This register can only be written if the WPEN bit is cleared in the Secure PIO Write Protection Mode Register.

- **MSKy: PIO Line y Mask**

These bits define the I/O lines to be configured when writing the Secure PIO Configuration Register.

0 (DISABLED): Writing the S_PIO_CFGRx, S_PIO_ODSRx or S_PIO_IOFRx does not affect the corresponding I/O line configuration.

1 (ENABLED): Writing the S_PIO_CFGRx, S_PIO_ODSRx or S_PIO_IOFRX updates the corresponding I/O line configuration.

31.7.16 Secure PIO Configuration Register

Name: S_PIO_CFGRx [x=0..3]

Address: 0xFC039004 [0], 0xFC039044 [1], 0xFC039084 [2], 0xFC0390C4 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	ICFS	PCFS	–	EVTSEL			
23	22	21	20	19	18	17	16
–	–	–	–	–	–	DRVSTR	
15	14	13	12	11	10	9	8
SCHMITT	OPD	IFSCEN	IFEN	–	PDEN	PUEN	DIR
7	6	5	4	3	2	1	0
–	–	–	–	–	FUNC		

This register can only be written if the WPEN bit is cleared in the Secure PIO Write Protection Mode Register.

Writing this register will only affect I/O lines enabled in the S_PIO_MSKRx.

- **FUNC: I/O Line Function**

This field defines the function for I/O lines of the I/O group x according to the Secure PIO Mask Register.

Value	Name	Description
0	GPIO	Select the PIO mode for the selected I/O lines.
1	PERIPH_A	Select the peripheral A for the selected I/O lines.
2	PERIPH_B	Select the peripheral B for the selected I/O lines.
3	PERIPH_C	Select the peripheral C for the selected I/O lines.
4	PERIPH_D	Select the peripheral D for the selected I/O lines.
5	PERIPH_E	Select the peripheral E for the selected I/O lines.
6	PERIPH_F	Select the peripheral F for the selected I/O lines.
7	PERIPH_G	Select the peripheral G for the selected I/O lines.

- **DIR: Direction**

This bit defines the direction of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (INPUT): The selected I/O lines are pure inputs.

1 (OUTPUT): The selected I/O lines are enabled in output.

- **PUEN: Pull-Up Enable**

This bit defines the pull-up configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): Pull-Up is disabled for the selected I/O lines.

1 (ENABLED): Pull-Up is enabled for the selected I/O lines.

- **PDEN: Pull-Down Enable**

This bit defines the pull-down configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): Pull-Down is disabled for the selected I/O lines.

1 (ENABLED): Pull-Down is enabled for the selected I/O lines only if PUEN is 0[(1)].

Note: 1. PDEN can be written to 1 only if PUEN is written to 0.

- **IFEN: Input Filter Enable**

This bit defines if the glitch filtering is used for the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): The input filter is disabled for the selected I/O lines.

1 (ENABLED): The input filter is enabled for the selected I/O lines.

- **IFSCEN: Input Filter Slow Clock Enable**

This bit defines the clock source of the glitch filtering for the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0: The glitch filter is able to filter glitches with a duration < $t_{mck}/2$ for the selected I/O lines.

1: The debouncing filter is able to filter pulses with a duration < $t_{div_slck}/2$ for the selected I/O lines.

- **OPD: Open-Drain**

This bit defines the open drain configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (DISABLED): The open-drain is disabled for the selected I/O lines. I/O lines are driven at high- and low-level.

1 (ENABLED): The open-drain is enabled for the selected I/O lines. I/O lines are driven at low-level only.

- **SCHMITT: Schmitt Trigger**

This bit defines the Schmitt trigger configuration of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

0 (ENABLED): Schmitt trigger is enabled for the selected I/O lines.

1 (DISABLED): Schmitt trigger is disabled for the selected I/O lines.

- **DRVSTR: Drive Strength**

This field defines the drive strength of the I/O lines of the I/O group x according to the Secure PIO Mask Register.

Value	Name	Description
0	LO	Low drive
1	LO	Low drive
2	ME	Medium drive
3	HI	High drive

- **EVTSEL: Event Selection**

This field defines the type of event to detect on the I/O lines of the I/O group x according to the Secure PIO Mask Register.

Value	Name	Description
0	FALLING	Event detection on input falling edge
1	RISING	Event detection on input rising edge
2	BOTH	Event detection on input both edge
3	LOW	Event detection on low level input
4	HIGH	Event detection on high level input
5	–	Reserved
6	–	Reserved
7	–	Reserved

- **PCFS: Physical Configuration Freeze Status (read-only)**

This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

- **ICFS: Interrupt Configuration Freeze Status (read-only)**

This bit gives information about the freeze state of the following fields of the read I/O line configuration:

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

0 (NOT_FROZEN): The fields are not frozen and can be written for this I/O line.

1 (FROZEN): The fields are frozen and cannot be written for this I/O line. Only a hardware reset can release these fields.

31.7.17 Secure PIO Pin Data Status Register

Name: S_PIO_PDSRx [x=0..3]

Address: 0xFC039008 [0], 0xFC039048 [1], 0xFC039088 [2], 0xFC0390C8 [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Data Status**

0: The I/O line of the I/O group x is at level 0.

1: The I/O line of the I/O group x is at level 1.

31.7.18 Secure PIO Lock Status Register

Name: S_PIO_LOCKSRx [x=0..3]

Address: 0xFC03900C [0], 0xFC03904C [1], 0xFC03908C [2], 0xFC0390CC [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Lock Status**

0: The I/O line of the I/O group x is not locked.

1: The I/O line of the I/O group x is locked.

31.7.19 Secure PIO Set Output Data Register

Name: S_PIO_SODRx [x=0..3]

Address: 0xFC039010 [0], 0xFC039050 [1], 0xFC039090 [2], 0xFC0390D0 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Set Output Data**

0: No effect.

1: Sets the data to be driven on the I/O line of I/O group x.

31.7.20 Secure PIO Clear Output Data Register

Name: S_PIO_CODRx [x=0..3]

Address: 0xFC039014 [0], 0xFC039054 [1], 0xFC039094 [2], 0xFC0390D4 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Clear Output Data**

0: No effect.

1: Clears the data to be driven on the I/O line of the I/O group x.

31.7.21 Secure PIO Output Data Status Register

Name: S_PIO_ODSRx [x=0..3]

Address: 0xFC039018 [0], 0xFC039058 [1], 0xFC039098 [2], 0xFC0390D8 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

Writing this register will only affect I/O lines enabled in the S_PIO_MSKRx.

- **P0–P31: Output Data Status**

0: The data to be driven on the I/O line of the I/O group x is 0.

1: The data to be driven on the I/O line of the I/O group x is 1.

31.7.22 Secure PIO Interrupt Enable Register

Name: S_PIO_IERx [x=0..3]

Address: 0xFC039020 [0], 0xFC039060 [1], 0xFC0390A0 [2], 0xFC0390E0 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Enable**

0: No effect.

1: Enables the Input Change interrupt on the I/O line of the I/O group x.

31.7.23 Secure PIO Interrupt Disable Register

Name: S_PIO_IDRx [x=0..3]

Address: 0xFC039024 [0], 0xFC039064 [1], 0xFC0390A4 [2], 0xFC0390E4 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Disable**

0: No effect.

1: Disables the Input Change interrupt on the I/O line of the I/O group x.

31.7.24 Secure PIO Interrupt Mask Register

Name: S_PIO_IMRx [x=0..3]

Address: 0xFC039028 [0], 0xFC039068 [1], 0xFC0390A8 [2], 0xFC0390E8 [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Mask**

0: Input Change interrupt is disabled on the I/O line of the I/O group x.

1: Input Change interrupt is enabled on the I/O line of the I/O group x.

31.7.25 Secure PIO Interrupt Status Register

Name: S_PIO_ISRx [x=0..3]

Address: 0xFC03902C [0], 0xFC03906C [1], 0xFC0390AC [2], 0xFC0390EC [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Input Change Interrupt Status**

0: No Input Change has been detected on the I/O line of the I/O group x since S_PIO_ISRx was last read or since reset.

1: At least one Input Change has been detected on the I/O line of the I/O group since S_PIO_ISRx was last read or since reset.

31.7.26 Secure PIO Set I/O Non-Secure Register

Name: S_PIO_SIONRx [x=0..3]

Address: 0xFC039030 [0], 0xFC039070 [1], 0xFC0390B0 [2], 0xFC0390F0 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Set I/O Non-Secure**

0: No effect.

1: Set the I/O line of the I/O group x in Non-Secure mode.

31.7.27 Secure PIO Set I/O Secure Register

Name: S_PIO_SIOSRx [x=0..3]

Address: 0xFC039034 [0], 0xFC039074 [1], 0xFC0390B4 [2], 0xFC0390F4 [3]

Access: Write-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: Set I/O Secure**

0: No effect.

1: Set the I/O line of the I/O group x in Secure mode.

31.7.28 Secure PIO I/O Security Status Register

Name: S_PIO_IOSSRx [x=0..3]

Address: 0xFC039038 [0], 0xFC039078 [1], 0xFC0390B8 [2], 0xFC0390F8 [3]

Access: Read-only

31	30	29	28	27	26	25	24
P31	P30	P29	P28	P27	P26	P25	P24
23	22	21	20	19	18	17	16
P23	P22	P21	P20	P19	P18	P17	P16
15	14	13	12	11	10	9	8
P15	P14	P13	P12	P11	P10	P9	P8
7	6	5	4	3	2	1	0
P7	P6	P5	P4	P3	P2	P1	P0

- **P0–P31: I/O Security Status**

0 (SECURE): The I/O line of the I/O group x is in Secure mode.

1 (NON_SECURE): The I/O line of the I/O group x is in Non-Secure mode.

31.7.29 Secure PIO I/O Freeze Configuration Register

Name: S_PIO_IOFRx [x=0..3]

Address: 0xFC03903C [0], 0xFC03907C [1], 0xFC0390BC [2], 0xFC0390FC [3]

Access: Write-only

31	30	29	28	27	26	25	24
FRZKEY							
23	22	21	20	19	18	17	16
FRZKEY							
15	14	13	12	11	10	9	8
FRZKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	FINT	FPHY

Writing this register will only affect I/O lines enabled in the S_PIO_MSKRx.

- **FPHY: Freeze Physical Configuration**

0: No effect.

1: Freezes the following configuration fields of Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):

- FUNC: I/O Line Function
- DIR: Direction
- PUEN: Pull-Up Enable
- PDEN: Pull-Down Enable
- OPD: Open-Drain
- SCHMITT: Schmitt Trigger
- DRVSTR: Drive Strength

Only a hardware reset can reset the FPHY bit.

- **FINT: Freeze Interrupt Configuration**

0: No effect.

1: Freezes the following configuration fields of Secure I/O lines if FRZKEY corresponds to 0x494F46 (“IOF” in ASCII):

- IFEN: Input Filter Enable
- IFSCEN: Input Filter Slow Clock Enable
- EVTSEL: Event Selection

Only a hardware reset can reset the FINT bit.

- **FRZKEY: Freeze Key**

Value	Name	Description
0x494F46	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit.

31.7.30 Secure PIO Slow Clock Divider Debouncing Register

Name: S_PIO_SCDR

Address: 0xFC039500

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	DIV					
7	6	5	4	3	2	1	0
DIV							

This register can only be written if the WPEN bit is cleared in the Secure PIO Write Protection Mode Register.

- **DIV: Slow Clock Divider Selection for Debouncing**

$t_{div_slck} = ((DIV + 1) \times 2) \times t_{slck}$

31.7.31 Secure PIO Write Protection Mode Register

Name: S_PIO_WPMR

Address: 0xFC0395E0

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x50494F (“PIO” in ASCII).

See Section 31.5.16 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x50494F	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

31.7.32 Secure PIO Write Protection Status Register

Name: S_PIO_WPSR

Address: 0xFC0395E4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
WPVSRC							
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the S_PIO_WPSR.

1: A write protection violation has occurred since the last read of the S_PIO_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

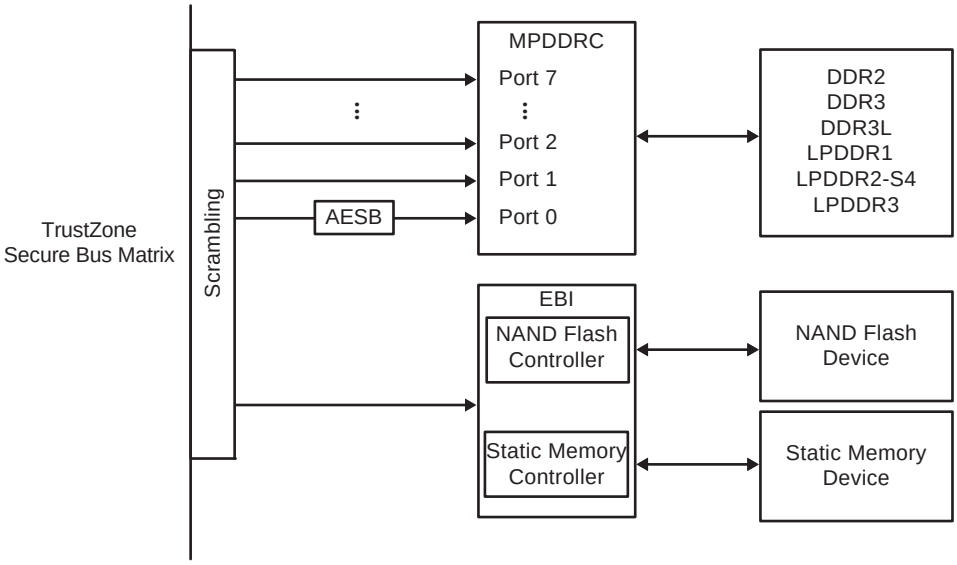
When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

32. External Memories

The product features:

- Multiport DDR-SDRAM Controller (MPDDRC)
- External Bus Interface (EBI) that embeds a NAND Flash controller and a Static Memory Controller (HSMC)

Figure 32-1. External Memory Controllers



- The MPDDRC is a multiport DDRSDR controller supporting DDR2, DDR3, DDR3L, LPDDR1, LPDDR2-S4 and LPDDR3 devices. The MPDDRC user interface is located at 0xF000C000. All the paths can be scrambled and Port 0 can be connected to an AES encryption/decryption engine.
- The HSMC supports Static Memories and MLC/SLC NAND Flash. It embeds MultiBit ECC correction (PMECC). Its user interface is located at 0xF8014000. The HSMC buses can be scrambled.

32.1 Multiport DDR-SDRAM Controller (MPDDRC)

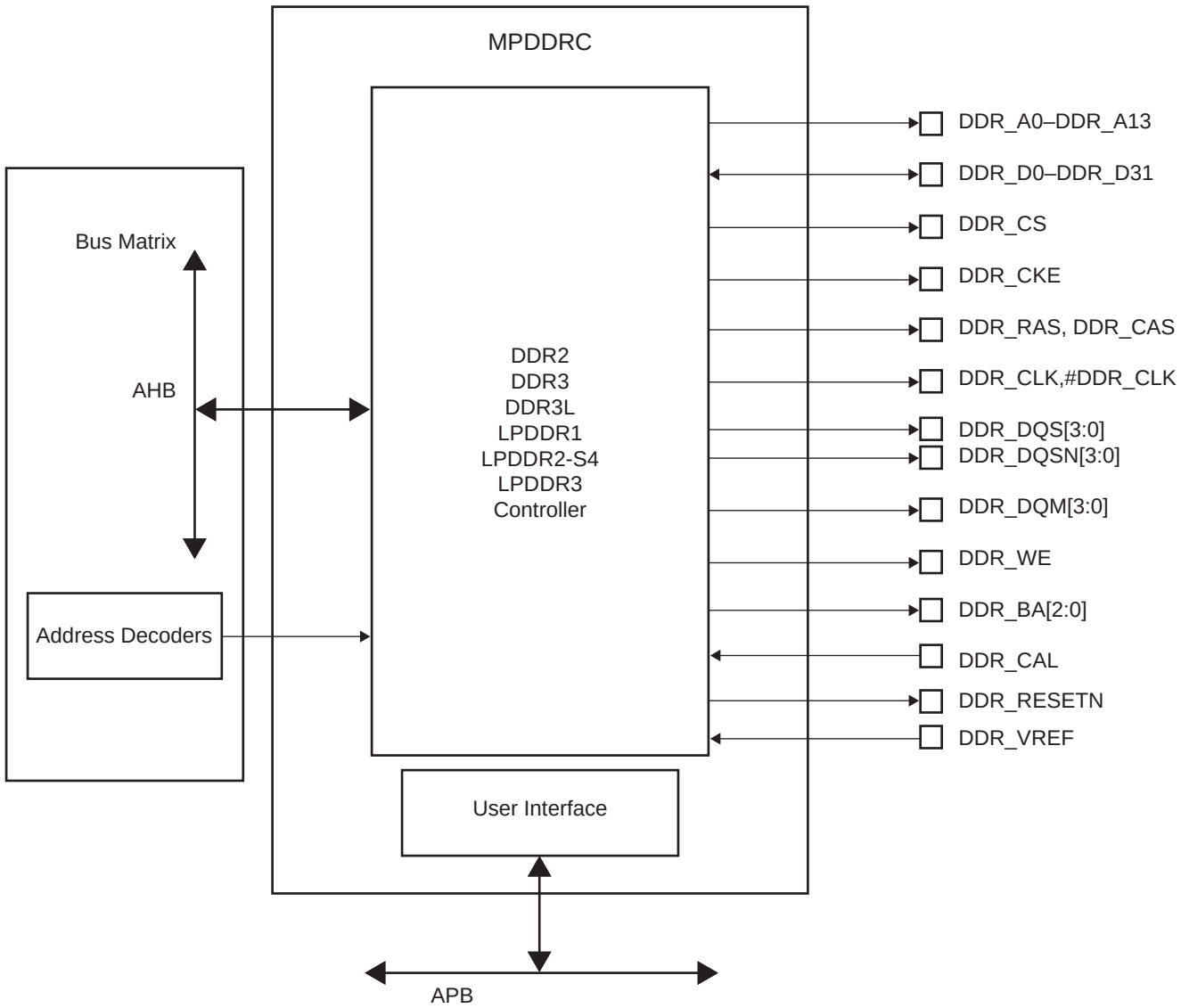
32.1.1 Description

The MPDDRC is an 8-port memory controller supporting DDR-SDRAM and low-power DDR devices. Data transfers are performed through a 16/32-bit data bus on one chip select. The controller operates with a 1.8V power supply for DDR2, DDR3, LPDDR1, 1.35V for DDR3L and 1.2V for LPDDR2, LPDDR3.

For full details, see Section 33. “Multiport DDR-SDRAM Controller (MPDDRC)”.

32.1.2 MPDDR Controller Block Diagram

Figure 32-2. MPDDRC Block Diagram



32.1.3 I/O Lines Description

Table 32-1. DDR/LPDDR I/O Lines Description

Name	Function	Type	Active Level
DDR/LPDDR Controller			
VDDIODDR	Power Supply of memory interface	Power	–
DDR_VREF	Reference Voltage	Input	–
DDR_CAL	Calibration reference	Input	–
DDR_D0–DDR_D31	Data Bus	I/O	–
DDR_A0–DDR_A13	Address Bus	Output	–
DDR_DQM0–DDR_DQM3	Data Mask	Output	–
DDR_DQS0–DDR_DQS3	Data Strobe	Output	–
DDR_DQSN0–DDR_DQSN3	Negative Data Strobe	Output	–
DDR_CS	Chip Select	Output	Low
DDR_RESETN	DDR3 Active Low Asynchronous Reset	Output	Low
DDR_CLK–DDR_CLK#	Differential Clock	Output	–
DDR_CKE	Clock enable	Output	High
DDR_RAS	Row signal	Output	Low
DDR_CAS	Column signal	Output	Low
DDR_WE	Write enable	Output	Low
DDR_BA0–DDR_BA2	Bank Select	Output	–

32.1.4 Product Dependencies

The pins used for interfacing the DDR/LPDDR memories are not multiplexed with the PIO lines.

The table below gives the connections to the various memory types.

Table 32-2. I/O Lines Usage vs Operating Modes

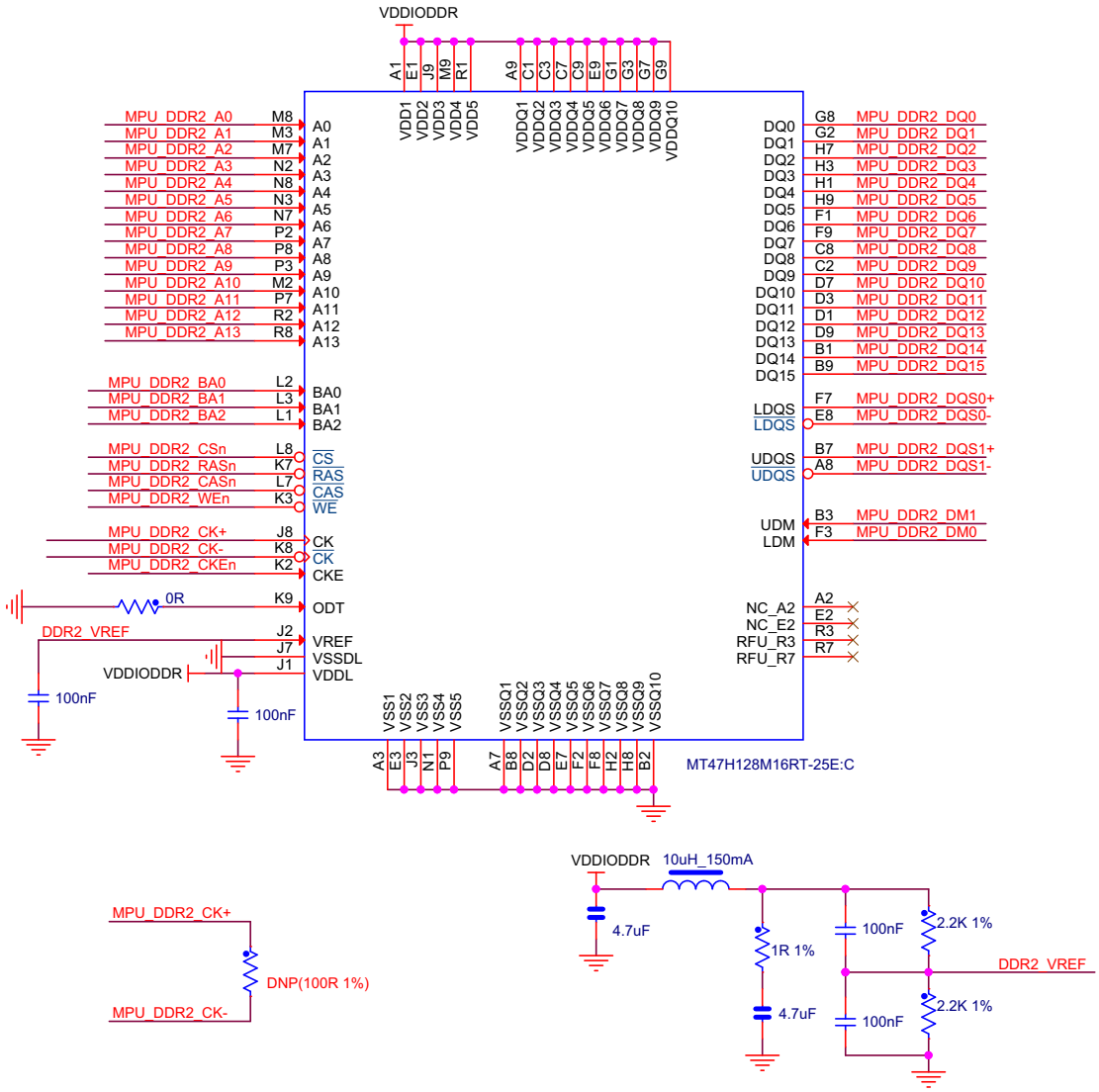
Signal Name	DDR2	DDR3	DDR3L	LPDDR1	LPDDR2/LPDDR3
DDR_VREF	VDDIODDR/2	VDDIODDR/2	VDDIODDR/2	VDDIODDR/2	VDDIODDR/2
DDR_CAL	GND via 21KΩ	GND via 22KΩ	GND via 23KΩ	GND via 21KΩ	GND via 24KΩ
DDR_CK, DDR_CKN	CLK, CLKN	CLK, CLKN	CLK, CLKN	CLK, CLKN	CLK, CLKN
DDR_CKE	CLKE	CLKE	CLKE	CLKE	CLKE
DDR_CS	CS	CS	CS	CS	CS
DDR_RESETN	Not connected	DDR_RESETN	DDR_RESETN	Not connected	Not connected
DDR_BA[2..0]	BA[2..0]	BA[2..0]	BA[2..0]	BA[1..0]	Not connected
DDR_WE	WE	WE	WE	WE	CA2
DDR_RAS, DDR_CAS	RAS, CAS	RAS, CAS	RAS, CAS	RAS, CAS	CA0, CA1
DDR_A[13..0]	A[13:0]	A[13:0]	A[13:0]	A[13:0]	CAx, with x > 2
DDR_D[31..0]	D[31:0]	D[31:0]	D[31:0]	D[31:0]	D[31:0]
DDR_DQS[3..0], DDR_DQSN[3..0]	LDQS,UDQS DDR_VREF	DQS[3:0] DQSN[3:0]	DQS[3:0] DQSN[3:0]	DQS[3..0], DDR_VREF	DQS[3:0] DQSN[3:0]
DDR_DQM[3..0]	UDM, LDM	DQM[3..0]	DQM[3..0]	DQM[3..0]	DQM[3..0]

32.1.5 Implementation Example

The following hardware configurations are given for illustration only. The user should refer to the memory manufacturer website to check current device availability.

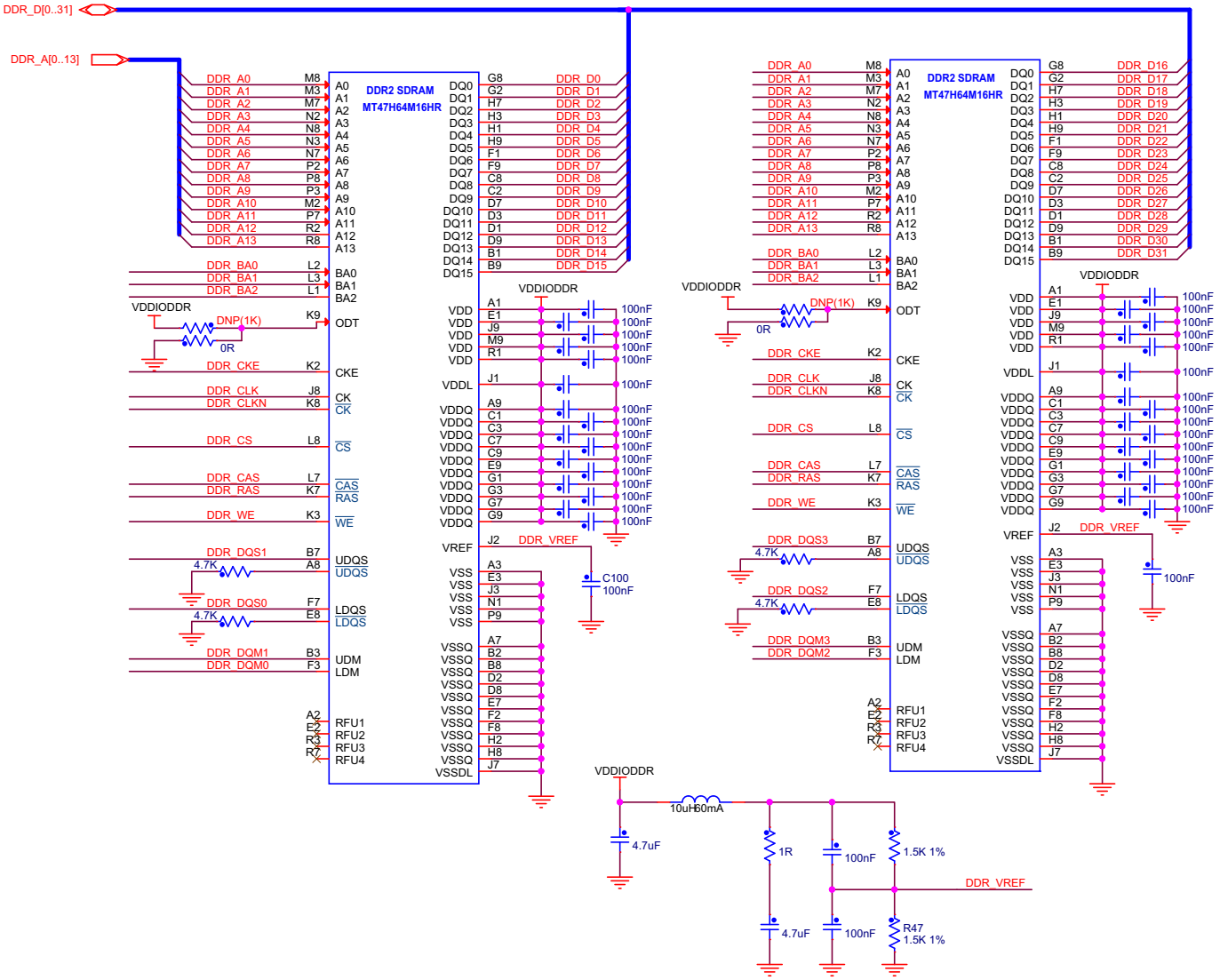
32.1.5.1 16-bit DDR2

Figure 32-3. 16-bit DDR2 Hardware Configuration



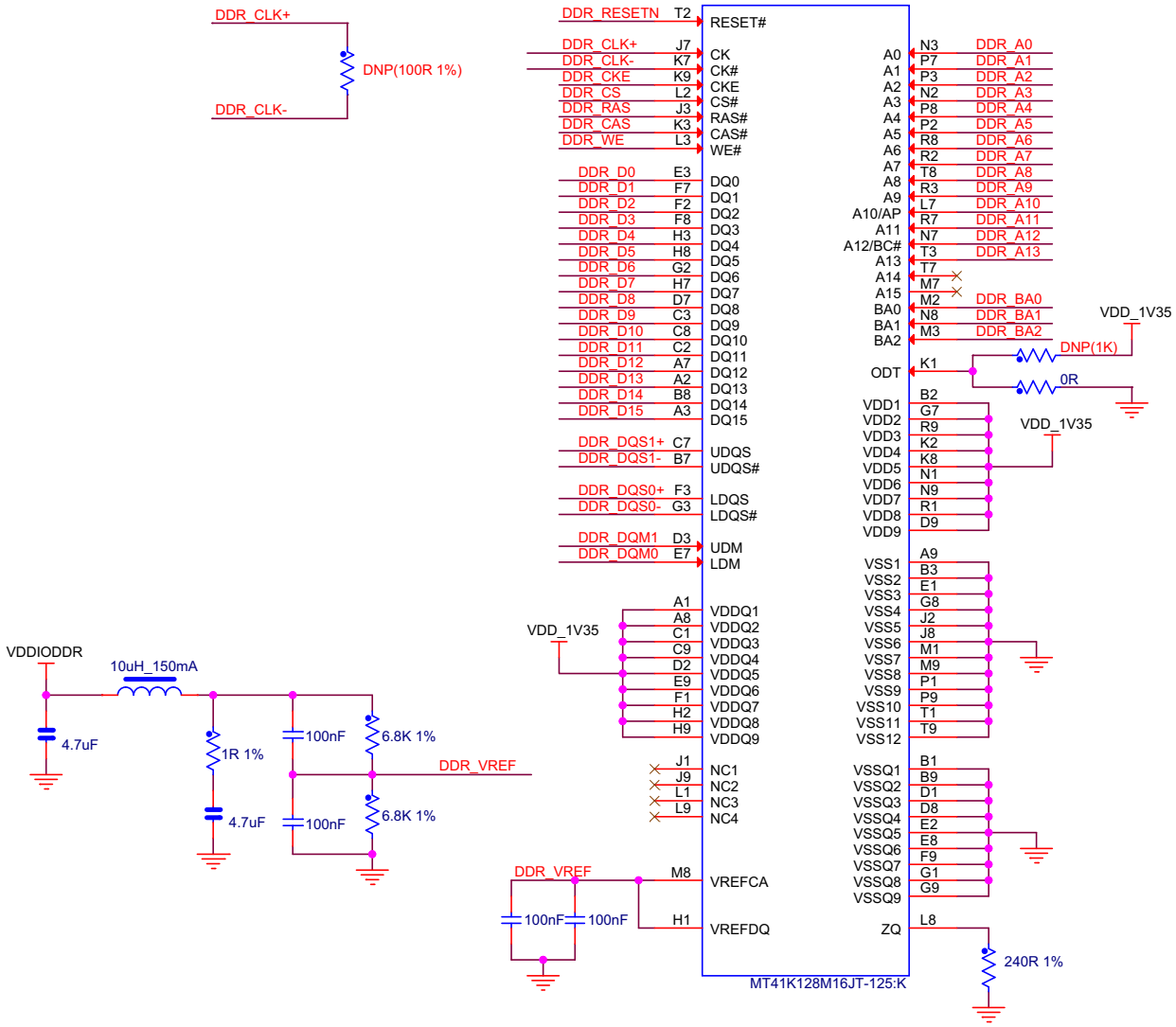
32.1.5.2 2x16-bit DDR2

Figure 32-4. 2x16-bit DDR2 Hardware Configuration



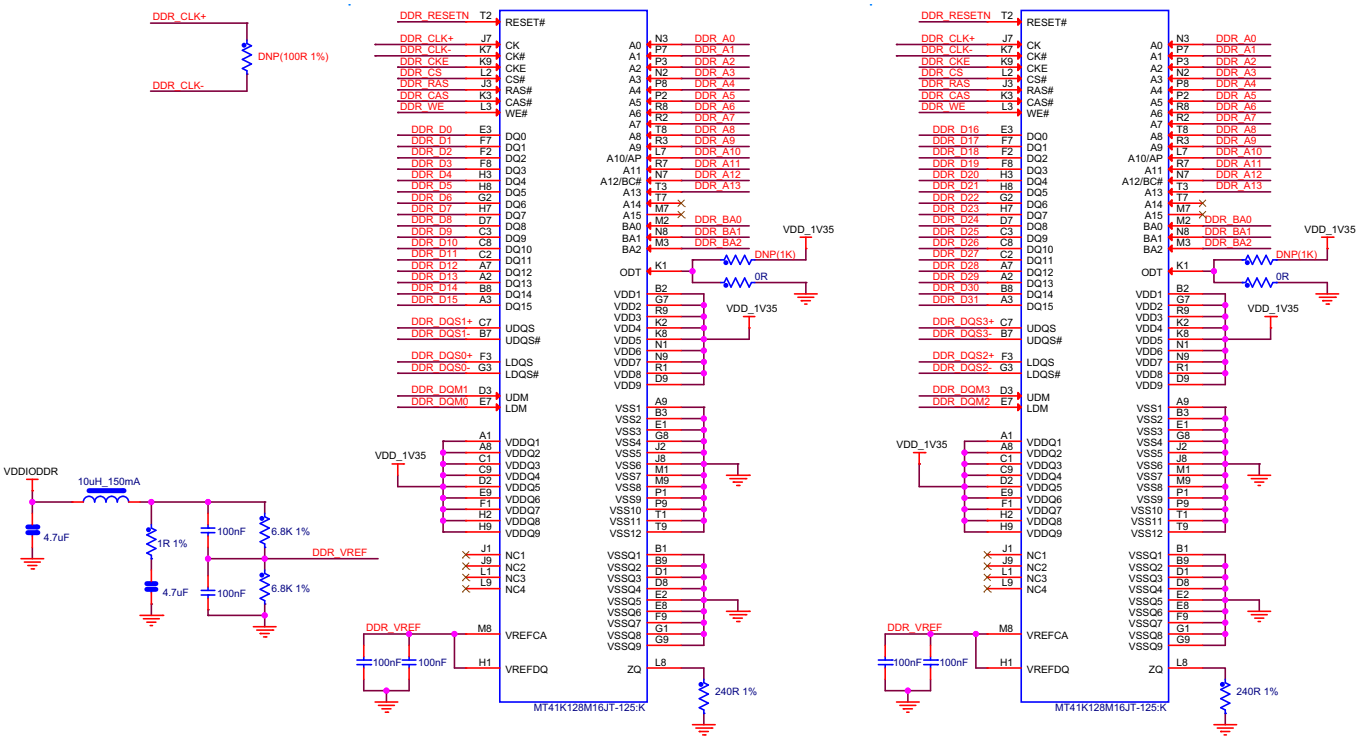
32.1.5.3 16-bit DDR3/DDR3L

Figure 32-5. 16-bit DDR3/DDR3L Hardware Configuration



32.1.5.4 2x16-bit DDR3/DDR3L

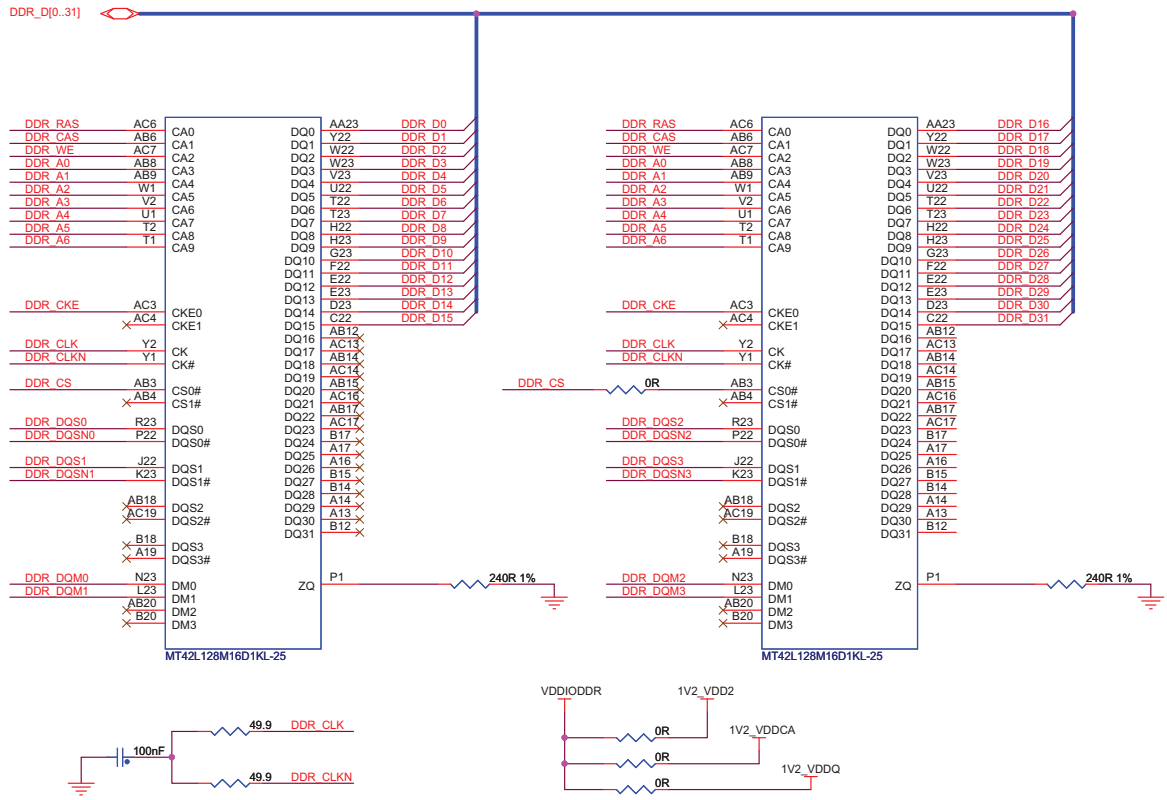
Figure 32-6. 2x16-bit DDR3/DDR3L Hardware Configuration



32.1.5.5 2x16-bit LPDDR2/LPDDR3

The schematic below is given for LPDDR2 but it is also valid for LPDDR3.

Figure 32-7. 2x16-bit LPDDR2 Hardware Configuration



CAx LPDDR2/LPDDR3 signals are to be connected as indicated in Table 32-3.

Table 32-3. CAx LPDDR2 Signal Connection

DDR Controller Signal	LPDDR2 Signal
RAS	CA0
CAS	CA1
WE	CA2
DDR_A0	CA3
DDR_A1	CA4
DDR_A2	CA5
DDR_A3	CA6
DDR_A4	CA7
DDR_A5	CA8
DDR_A6	CA9
Higher addresses	Higher CAs

32.2 External Bus Interface (EBI)

32.2.1 Description

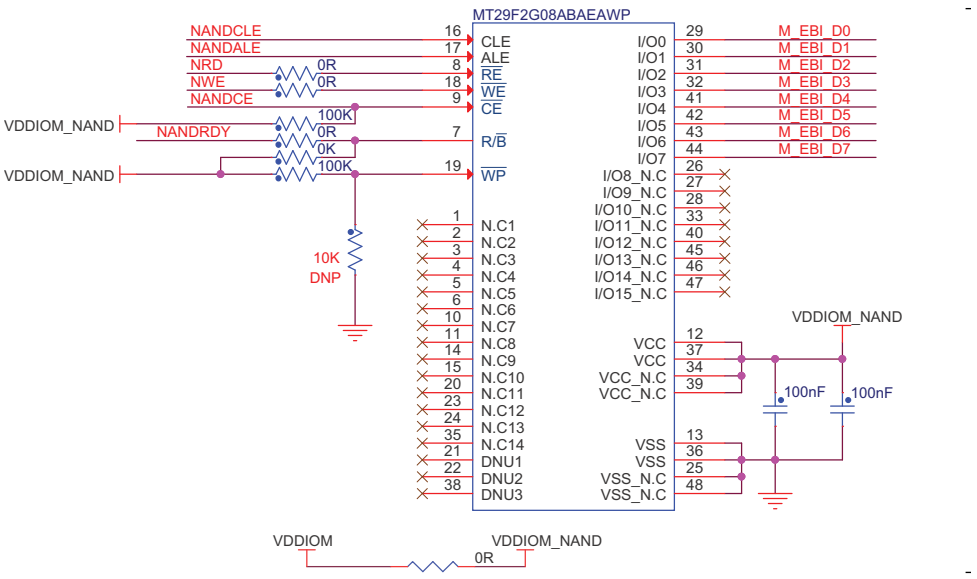
The External Bus Interface is designed to ensure the successful data transfer between several external devices and the ARM processor-based device. The External Bus Interface of the device consists of a Static Memory Controller (HSMC).

32.2.2 Implementation Examples

The following hardware configurations are given for illustration only. The user should refer to the memory manufacturer website to check current device availability.

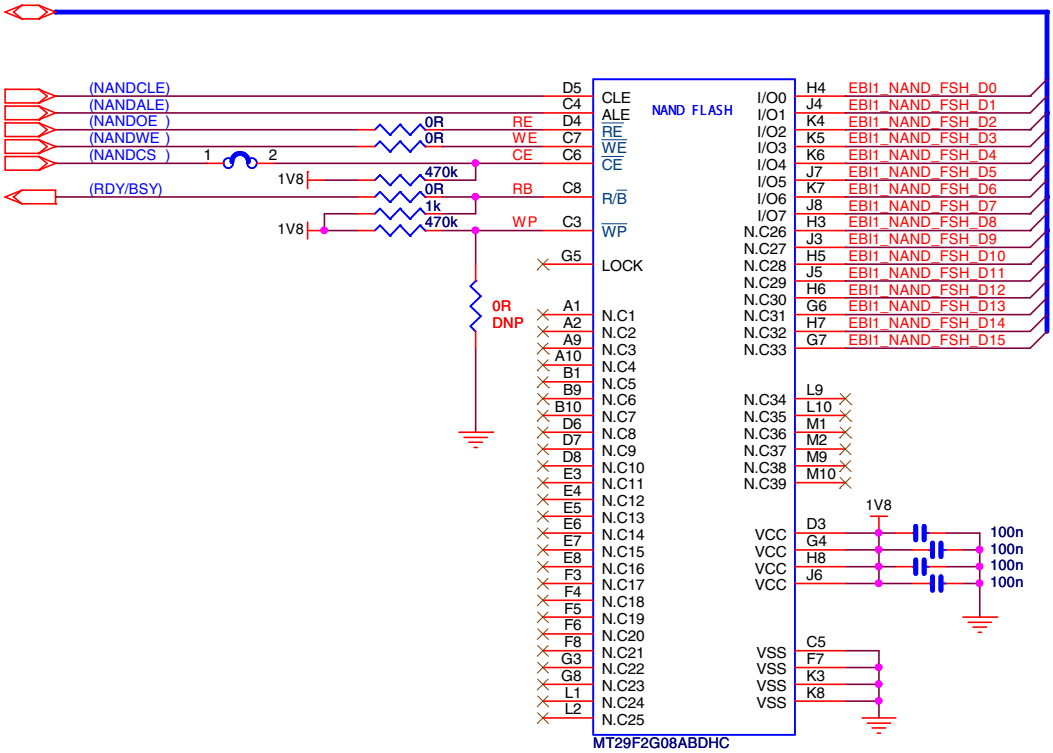
32.2.2.1 8-bit NAND Flash

Figure 32-8. 8-bit NAND Flash Hardware Configuration



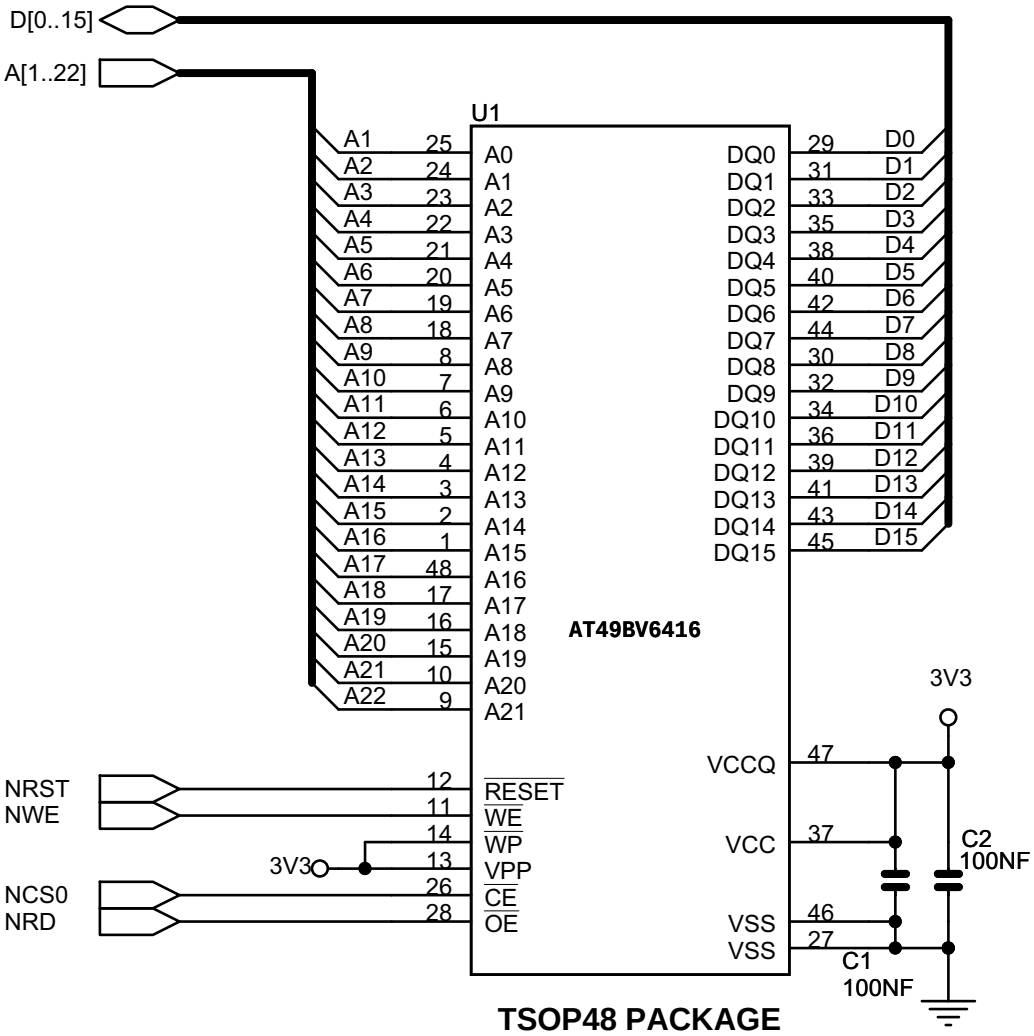
32.2.2.2 16-bit NAND Flash

Figure 32-9. 16-bit NAND Flash Hardware Configuration



32.2.2.3 NOR Flash on NCS0

Figure 32-10. NOR Flash on NCS0 Hardware Configuration



33. Multiport DDR-SDRAM Controller (MPDDRC)

33.1 Description

The Multiport DDR-SDRAM Controller (MPDDRC) is a multiport memory controller. It comprises eight slave AHB interfaces. All simultaneous accesses (eight independent AHB ports) are interleaved to maximize memory bandwidth and minimize transaction latency due to DDR-SDRAM protocol.

The MPDDRC extends the memory capabilities of a chip by providing the interface to the external 16-bit or 32-bit DDR-SDRAM device. The page size supports ranges from 2048 to 16384 rows and from 256 to 4096 columns. It supports dword (64-bit), word (32-bit), half-word (16-bit), and byte (8-bit) accesses.

The MPDDRC supports a read or write burst length of eight locations. This enables the command and address bus to anticipate the next command, thus reducing latency imposed by the DDR-SDRAM protocol and improving the DDR-SDRAM bandwidth. Moreover, MPDDRC keeps track of the active row in each bank, thus maximizing DDR-SDRAM performance, e.g., the application may be placed in one bank and data in other banks. To optimize performance, avoid accessing different rows in the same bank. The MPDDRC supports a CAS latency of 2, 3 or 6 and optimizes the read access depending on the frequency.

Self-refresh, Powerdown and Deep Powerdown modes minimize the consumption of the DDR-SDRAM device.

OCD (Off-chip Driver) and ODT (On-die Termination) modes are not supported.

The MPDDRC supports DDR3-SDRAM and DDR3L-SDRAM devices with DLL disabled, in DLL Off mode. In this mode, according to JEDEC standard, the maximum clock frequency is 125 MHz. However, check with memory suppliers for higher speed support. DDR3-SDRAM supports high capacity, 1 Gbit and more, and allows to reduce power consumption with a 1.5V supply (DDR3-SDRAM) or a 1.35V supply (DDR3L-SDRAM). The DLL Off mode sets the CAS Read Latency (CRL) and the CAS Write Latency (CWL) to 6. The latency is automatically set by the controller.

33.2 Embedded Characteristics

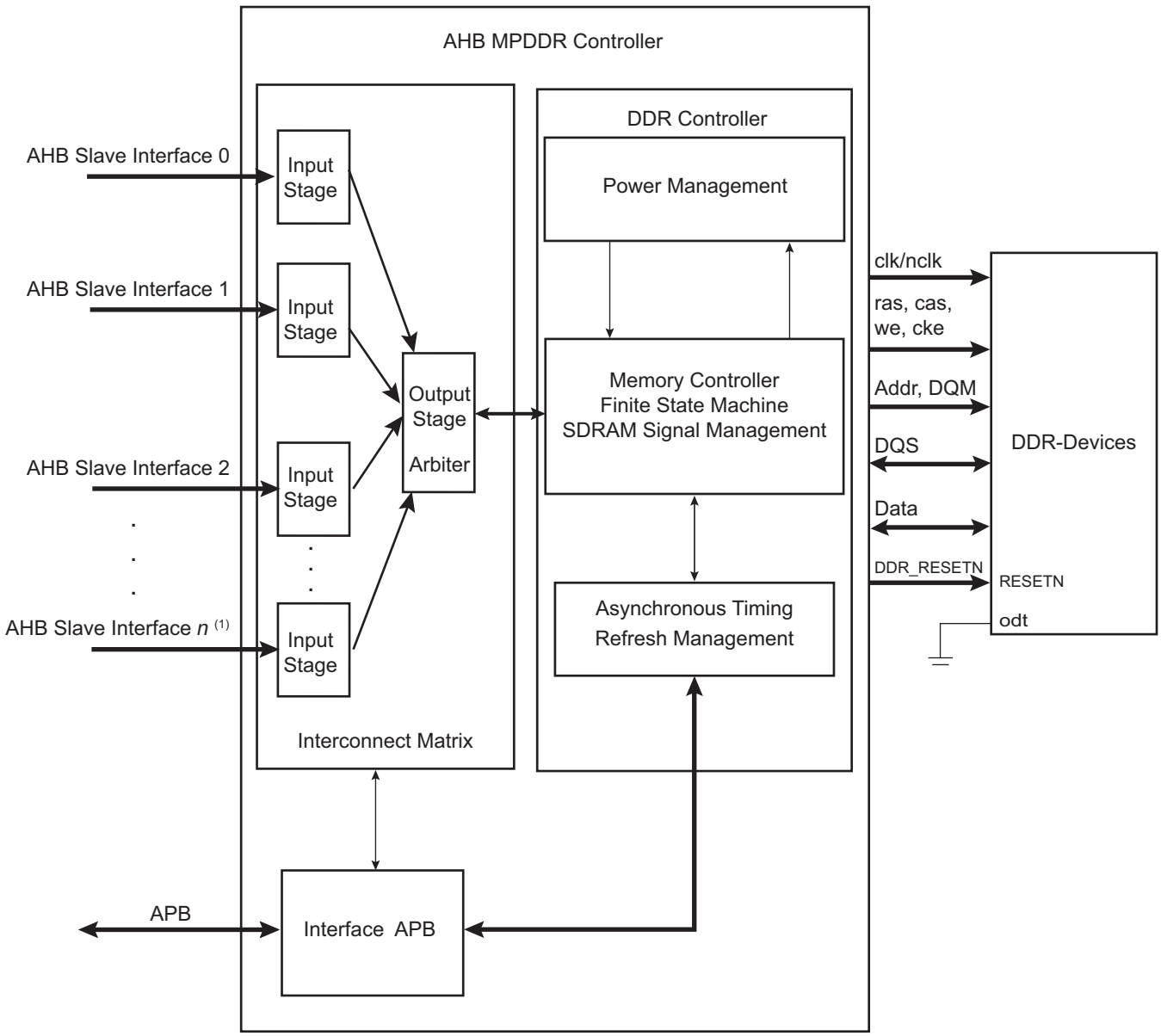
- Eight advanced high performance bus (AHB) interfaces, management of all accesses maximizes memory bandwidth and minimizes transaction latency
- Bus transfer: dword, word, half word, byte access
- Supports low-power DDR3-SDRAM (LPDDR3), DDR3-SDRAM (DLL Off mode), DDR3L-SDRAM (DLL Off mode), low-power DDR2-SDRAM-S4 (LPDDR2), DDR2-SDRAM, low-power DDR1-SDRAM (LPDDR1)
- Numerous configurations supported
  - 2K, 4K, 8K, 16K row address memory parts
  - DDR-SDRAM with two or four internal banks (low-power DDR1-SDRAM)
  - DDR-SDRAM with four or eight internal banks (DDR2-SDRAM/Low-power DDR2-SDRAM-S4/DDR3-SDRAM/DDR3-SDRAM-L/Low-power DDR3-SDRAM)
  - DDR-SDRAM with 16-bit or 32-bit data
  - One chip select for SDRAM device (512-Mbyte address space, 256-Mbyte address space with 16-bit data path)
- Programming Facilities
  - Multibank ping-pong access (up to four or eight banks opened at the same time = reduced average latency of transactions)
  - Timing parameters specified by software
  - Automatic refresh operation, refresh rate is programmable
  - Automatic update of DS, TCR and PASR parameters (low-power DDR-SDRAM devices)
- Energy-saving capabilities
  - Self-refresh, Powerdown, Active Powerdown and Deep Powerdown modes supported
- DDR-SDRAM powerup initialization by software
- CAS latency of 2, 3, 5, 6 supported
- Reset function supported (DDR2-SDRAM)
- Clock frequency change in Self-refresh mode supported (low-power DDR-SDRAM/DDR3-SDRAM/DDR3L-SDRAM)
- Auto-refresh per bank supported (low-power DDR2-SDRAM-S4/low-power DDR3-SDRAM)
- Automatic adjust refresh rate (low-power DDR2-SDRAM-S4/low-power DDR3-SDRAM)
- Auto-precharge command not used
- OCD (Off-chip Driver) mode, ODT (On-die Termination) are not supported
- Dynamic Scrambling with user key (no impact on bandwidth)

33.3 MPDDRC Module Diagram

MPDDRC is partitioned in two blocks (see Figure 33-1):

- Interconnect Matrix block that manages concurrent accesses on the AHB bus between four AHB masters and integrates an arbiter
- DDR controller that translates AHB requests (read/write) in the DDR-SDRAM protocol

Figure 33-1. MPDDRC Module Diagram



Note: 1. "*n*" can equal 3 or 7 (value is device-specific).

33.4 Product Dependencies, Initialization Sequence

33.4.1 Low-power DDR1-SDRAM Initialization

The initialization sequence is generated by software.

The low-power DDR1-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program the features of the low-power DDR1-SDRAM device in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output drive strength) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 1 Register (asynchronous timing (TRC, TRAS, etc.)).
3. Program Temperature Compensated Self-refresh (TCR), Partial Array Self-refresh (PASR) and Drive Strength (DS) parameters in the MPDDRC Low-power Register.
4. A NOP command is issued to the low-power DDR1-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command. The clocks which drive the low-power DDR1-SDRAM device are now enabled.
5. A pause of at least 200 µs must be observed before a signal toggle.
6. A NOP command is issued to the low-power DDR1-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command. A calibration request is now made to the I/O pad.
7. An All Banks Precharge command is issued to the low-power DDR1-SDRAM. Program All Banks Precharge command in the MPDDRC_MR. The application must write a 2 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM address to acknowledge this command.
8. Two auto-refresh (CBR) cycles are provided. Program the Auto Refresh command (CBR) in the MPDDRC_MR. The application must write a 4 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR1-SDRAM location twice to acknowledge these commands.
9. An Extended Mode Register Set (EMRS) cycle is issued to program the low-power DDR1-SDRAM parameters (TCSR, PASR, DS). The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and BA[0] is set to 0. For example: with a 16-bit, 128-Mbit, low-power DDR1-SDRAM (12 rows, 9 columns, 4 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00800000; with a 32-bit, 1-Gbit, low-power DDR1-SDRAM (14 rows, 10 columns, 4 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000. In the case of low-cost and low-density low-power DDR1-SDRAM (2 internal banks), the write address must be chosen so that signal BA[0] is set to 1. BA[1] is not used.

Note: This address is given as an example only. The real address depends on implementation in the product.

10. A Mode Register Set (MRS) cycle is issued to program parameters of the low-power DDR1-SDRAM devices, in particular CAS latency. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the low-power DDR1-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.

11. The application must enter Normal mode, write a zero to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access at any location in the low-power DDR1-SDRAM to acknowledge this command.
12. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The low-power DDR1-SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 100 MHz frequency, MPDDRC_RTR must be set with (15.625 × 100 MHz) = 1562 i.e., 0x061A or (7.81 × 100 MHz) = 781 i.e., 0x030D.

After initialization, the low-power DDR1-SDRAM device is fully functional.

33.4.2 DDR2-SDRAM Initialization

The initialization sequence is generated by software. The DDR2-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program features of the DDR2-SDRAM device in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output driver impedance control) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 1 Register (asynchronous timing (TRC, TRAS, etc.).
3. A NOP command is issued to the DDR2-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command. The clocks which drive the DDR2-SDRAM device are now enabled.
4. A pause of at least 200 µs must be observed before a signal toggle.
5. A NOP command is issued to the DDR2-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command. CKE is now driven high.
6. An All Banks Precharge command is issued to the DDR2-SDRAM. Program All Banks Precharge command in the MPDDRC_MR. The application must write a 2 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
7. An Extended Mode Register Set (EMRS2) cycle is issued to choose between commercial or high temperature operations. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and signal BA[0] is set to 0. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00800000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000.

Note: This address is given as an example only. The real address depends on implementation in the product.

8. An Extended Mode Register Set (EMRS3) cycle is issued to set the Extended Mode Register to 0. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 1 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00C00000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x0C000000.
9. An Extended Mode Register Set (EMRS1) cycle is issued to enable DLL and to program D.I.C. (Output Driver Impedance Control). The application must write a 5 to the MODE field in the MPDDRC_MR. Read the

MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.

10. An additional 200 cycles of clock are required for locking DLL
11. Write a one to the DLL bit (enable DLL reset) in the MPDDRC Configuration Register (MPDDRC_CR).
12. A Mode Register Set (MRS) cycle is issued to reset DLL. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
13. An All Banks Precharge command is issued to the DDR2-SDRAM. Program the All Banks Precharge command in the MPDDRC_MR. The application must write a 2 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
14. Two auto-refresh (CBR) cycles are provided. Program the Auto Refresh command (CBR) in the MPDDRC_MR. The application must write a 4 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM location twice to acknowledge these commands.
15. Write a zero to the DLL bit (disable DLL reset) in the MPDDRC_CR.
16. A Mode Register Set (MRS) cycle is issued to program parameters of the DDR2-SDRAM device, in particular CAS latency and to disable DLL reset. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[1:0] are set to 0. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks) bank address, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
17. Write a seven to the OCD field (default OCD calibration) in the MPDDRC_CR.
18. An Extended Mode Register Set (EMRS1) cycle is issued to the default OCD value. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks), the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.
19. Write a zero to the OCD field (exit OCD calibration mode) in the MPDDRC_CR.
20. An Extended Mode Register Set (EMRS1) cycle is issued to enable OCD exit. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR2-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 128-Mbit, DDR2-SDRAM (12 rows, 9 columns, 4 banks) bank address, the DDR2-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x00400000; with a 32-bit, 1-Gbit, DDR2-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.

21. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR2-SDRAM address to acknowledge this command.
22. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The DDR2-SDRAM device requires a refresh every 15.625 µs or 7.81 µs. With a 133 MHz frequency, the COUNT field in the MPDDRC_RTR must be set with (15.625 × 133 MHz) = 2079 i.e., 0x081F or (7.81 × 133 MHz) = 1039 i.e., 0x040F.

After initialization, the DDR2-SDRAM devices are fully functional.

33.4.3 Low-power DDR2-SDRAM Initialization

The initialization sequence is generated by software. The low-power DDR2-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program features of the low-power DDR2-SDRAM device into and in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output drive strength) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 0 Register (asynchronous timing, TRC, TRAS, etc.).
3. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The clocks which drive the Low-power DDR2-SDRAM devices are now enabled.
4. A pause of at least 100 ns must be observed before a signal toggle.
5. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. CKE is now driven high.
6. A pause of at least 200 µs must be observed before issuing a Reset command.
7. A Reset command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 63 to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Reset command is now issued.
8. A pause of at least t_{INIT5} must be observed before issuing any commands.
9. A Calibration command is issued to the low-power DDR2-SDRAM. Program the type of calibration in the MPDDRC Configuration Register (MPDDRC_CR): set the ZQ field to the RESET value. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 10 to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The ZQ Calibration command is now issued. Program the type of calibration in the MPDDRC_CR: set the ZQ field to the SHORT value.
10. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a one to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
11. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a two to the MRS field. The Mode Register Write command cycle is issued to

program parameters of the low-power DDR2-SDRAM device, in particular CAS latency. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.

12. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a three to the MRS field. The Mode Register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular Drive Strength and Slew Rate. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
13. A Mode Register Write command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 16 to the MRS field. Mode Register Write command cycle is issued to program parameters of the low-power DDR2-SDRAM device, in particular Partial Array Self Refresh (PASR). Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
14. In the DDR Configuration Register (SFR_DDRCFG), the application must write a 1 to fields 17 and 16 to open the input buffers (See section “Special Function Registers (SFR)”).
15. A NOP command is issued to the low-power DDR2-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command.
16. A Mode Register Read command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a five to the MRS field. The Mode Register Read command cycle is used to read the LPDDR2 Manufacturer ID from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. The LPDDR2 Manufacturer ID is set in register MPDDRC_MD. See Section 33.7.8 “MPDDRC Memory Device Register”.
17. A Mode Register Read command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a six to the MRS field. The Mode Register Read command cycle is used to read Revision ID1 from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. Revision ID1 is set in register MPDDRC_MD. See Section 33.7.8 “MPDDRC Memory Device Register”.
18. A Mode Register Read command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and an eight to the MRS field. The Mode Register Read command cycle is used to read the memory organization (I/O width, Density, Type) from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. Memory organization is set in register MPDDRC_MD. See Section 33.7.8 “MPDDRC Memory Device Register”.
19. A Mode Register Read command is issued to the low-power DDR2-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The

application must write a 7 to the MODE field and a zero to the MRS field. The Mode Register Read command cycle is used to read device information (RZQI, DAI) from the low-power DDR2-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. Device information RZQI is set in register Timing Calibration (see Section 33.7.11 “MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register”) and DAI is set in Mode Register (see Section 33.7.1 “MPDDRC Mode Register”).

20. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR2-SDRAM address to acknowledge this command.
21. In the DDR configuration Register (SFR_DDRCCFG), the application must write a 0 to fields 17 and 16 to close the input buffers. The buffers are then driven by the HMPDDRC controller.
22. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The low-power DDR2-SDRAM device requires a refresh every 7.81 µs. With a 133 MHz frequency, the COUNT field in the MPDDRC_RTR must be set with (7.81 × 133 MHz) = 1039 i.e., 0x040F.

After initialization, the low-power DDR2-SDRAM devices are fully functional.

33.4.4 DDR3-SDRAM/DDR3L-SDRAM Initialization

The initialization sequence is generated by software. The DDR3-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program features of the DDR3-SDRAM device in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output driver impedance control) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 1 Register (asynchronous timing - TRC, TRAS, etc.).
3. A NOP command is issued to the DDR3-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR3-SDRAM address to acknowledge this command. The clocks which drive the DDR3-SDRAM device are now enabled.
4. A pause of at least 500 µs must be observed before a signal toggle.
5. A NOP command is issued to the DDR3-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR3-SDRAM address to acknowledge this command. CKE is now driven high.
6. An Extended Mode Register Set (EMRS2) cycle is issued to choose between commercial or high temperature operations. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[2] is set to 0, BA[1] is set to 1 and signal BA[0] is set to 0. For example: with a 16-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the DDR3-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x040000000; with a 32-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x08000000.

Note: This address is given as an example only. The real address depends on the implementation in the product.

7. An Extended Mode Register Set (EMRS3) cycle is issued to set the Extended Mode Register to 0. The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[2] is set to 0, BA[1] is set to 1 and signal BA[0] is set to 1. For example: with a 16-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8

banks), the DDR3-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x06000000; with a 32-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x0C000000.

8. An Extended Mode Register Set (EMRS1) cycle is issued to disable and to program O.D.S. (Output Drive Strength). The application must write a 5 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signal BA[2:1] is set to 0 and signal BA[0] is set to 1. For example: with a 16-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10columns, 8 banks), the DDR3-SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x02000000; with a 32-bit, 1-Gbit, DDR3-SDRAM (14 rows, 10 columns, 8 banks), the SDRAM write access should be done at the address: BASE_ADDRESS_DDR + 0x04000000.
9. Write a one to the DLL bit (enable DLL reset) in the MPDDRC Configuration Register (MPDDRC_CR).
10. A Mode Register Set (MRS) cycle is issued to reset DLL. The application must write a 3 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[2:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
11. A Calibration command (MRS) is issued to calibrate RTT and RON values for the Process Voltage Temperature (PVT). The application must write a 6 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to the DDR3-SDRAM to acknowledge this command. The write address must be chosen so that signals BA[2:0] are set to 0. For example, the SDRAM write access should be done at the address: BASE_ADDRESS_DDR.
12. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any DDR3-SDRAM address to acknowledge this command.
13. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The DDR3-SDRAM device requires a refresh every 7.81 µs. With a 125-MHz frequency, the COUNT field in the MPDDRC_RTR must be set as follows: $(7.81 \times 125\ \text{MHz}) = 977$ i.e., 0x03D1.

After initialization, the DDR3-SDRAM devices are fully functional.

33.4.5 Low-power DDR3-SDRAM Initialization

The initialization sequence is generated by software. The low-power DDR3-SDRAM devices are initialized by the following sequence:

1. Program the memory device type in the MPDDRC Memory Device Register (MPDDRC_MD).
2. Program features of the low-power DDR3-SDRAM device into and in the MPDDRC Configuration Register (number of columns, rows, banks, CAS latency and output drive strength) and in the MPDDRC Timing Parameter 0 Register/MPDDRC Timing Parameter 0 Register (asynchronous timing, TRC, TRAS, etc.).
3. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The clocks which drive the low-power DDR3-SDRAM devices are now enabled.
4. A pause of at least 100 ns must be observed before a signal toggle.
5. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the MPDDRC_MR. The application must write a 1 to the MODE field in the MPDDRC_MR. Read the

MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. CKE is now driven high.

6. A pause of at least 200 µs must be observed before issuing a Reset command.
7. A Reset command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 63 to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Reset command is now issued.
8. A pause of at least t_{INIT5} must be observed before issuing any commands.
9. A Calibration command is issued to the low-power DDR3-SDRAM. Program the type of calibration in the MPDDRC Configuration Register (MPDDRC_CR): set the ZQ field to the RESET value. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 10 to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The ZQ Calibration command is now issued. Program the type of calibration in the MPDDRC_CR: set the ZQ field to the SHORT value.
10. A Mode Register Write command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a one to the MRS field. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
11. A Mode Register Write command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a two to the MRS field. The Mode Register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular CAS Latency. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
12. A Mode Register Write command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a three to the MRS field. The Mode Register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular Drive Strength and Slew Rate. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
13. A Mode Register Write command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a 16 to the MRS field. The Mode Register Write command cycle is issued to program parameters of the low-power DDR3-SDRAM device, in particular Partial Array Self Refresh (PASR). Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Write command is now issued.
14. In the DDR Configuration Register (SFR_DDRCFG), the application must write a 1 to fields 17 and 16 to open the input buffers.
15. A NOP command is issued to the low-power DDR3-SDRAM. Program the NOP command in the MPDDRC Mode Register (MPDDRC_MR). The application must write a 1 to the MODE field in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command.

16. A Mode Register Read command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a five to the MRS field. The Mode Register Read command cycle is used to read the LPDDR3 Manufacturer ID from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. The LPDDR3 Manufacturer ID is set in register MPDDRC_MD. See Section 33.7.8 “MPDDRC Memory Device Register”.
17. A Mode Register Read command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a six to the MRS field. The Mode Register Read command cycle is used to read the Revision ID1 from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. Revision ID1 is set in register MPDDRC_MD. See Section 33.7.8 “MPDDRC Memory Device Register”.
18. A Mode Register Read command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and an eight to the MRS field. The Mode Register Read command cycle is used to read memory organization (I/O width, Density, Type) from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. Memory organization is set in register MPDDRC_MD. See Section 33.7.8 “MPDDRC Memory Device Register”.
19. A Mode Register Read command is issued to the low-power DDR3-SDRAM. In the MPDDRC_MR, configure the MODE field to the LPDDR2_LPDDR3_CMD value and configure the MRS field. The application must write a 7 to the MODE field and a zero to the MRS field. The Mode Register Read command cycle is used to read the device information (RZQI, DAI) from the low-power DDR3-SDRAM mode registers. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command. The Mode Register Read command is now issued. Device information RZQI is set in register Timing Calibration (see Section 33.7.11 “MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register”) and DAI is set in Mode Register (see Section 33.7.1 “MPDDRC Mode Register”).
20. A Normal Mode command is provided. Program the Normal mode in the MPDDRC_MR. Read the MPDDRC_MR and add a memory barrier assembler instruction just after the read. Perform a write access to any low-power DDR3-SDRAM address to acknowledge this command.
21. In the DDR configuration Register (SFR_DDRCCFG), the application must write a 0 to fields 17 and 16 to close the input buffers. The buffers are then driven by the HMPDDRC controller.
22. Write the refresh rate into the COUNT field in the MPDDRC Refresh Timer Register (MPDDRC_RTR): refresh rate = delay between refresh cycles. The low-power DDR3-SDRAM device requires a refresh every 3.9 µs. With a 133-MHz frequency, the COUNT field in the MPDDRC_RTR must be set as follows: (3.9 × 133 MHz) = 518 i.e., 0x0206.

After initialization, the low-power DDR3-SDRAM devices are fully functional.

33.5 Functional Description

33.5.1 DDR-SDRAM Controller Write Cycle

The MPDDRC provides burst access or single access in Normal mode (MPDDRC_MR.MODE = 0). Whatever the access type, the MPDDRC keeps track of the active row in each bank, thus maximizing performance.

The DDR-SDRAM device is programmed with a burst length (bl) equal to 8. This determines the length of a sequential data input by the write command that is set to 8. The latency from write command to data input depends on the memory type, as shown in Table 33-1.

Table 33-1. CAS Write Latency

Memory Devices	CAS Write Latency (CWL)
Low-power DDR1-SDRAM	1
Low-power DDR2-SDRAM	1
DDR2-SDRAM	2
DDR3-SDRAM (DLL OFF)	6

Note: In the case of low-power DDR3-SDRAM, the CAS Write Latency (CWL) of 1 is optional. The MPDDRD supports this feature. Refer to the low-power DDR3-SDRAM datasheet for details.

To initiate a single access, the MPDDRC checks if the page access is already open. If row/bank addresses match with the previous row/bank addresses, the controller generates a write command. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a write command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/write (t_{RCD}) command. As the burst length is set to 8, in case of single access, it has to stop the burst, otherwise seven invalid values may be written. In case of the DDR-SDRAM device, the burst stop command is not supported for the burst write operation. Thus, in order to interrupt the write operation, the DM (data mask) input signal must be set to 1 to mask invalid data (see Figure 33-2 and Figure 33-4), and DQS must continue to toggle.

To initiate a burst access, the MPDDRC uses the transfer type signal provided by the master requesting the access. If the next access is a sequential write access, writing to the DDR-SDRAM device is carried out. If the next access is a write non-sequential access, then an automatic access break is inserted, the MPDDRC generates a precharge command, activates the new row and initiates a write command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/write (t_{RCD}) commands.

For the definition of timing parameters, refer to Section 33.7.4 “MPDDRC Timing Parameter 0 Register”.

Write accesses to the DDR-SDRAM device are burst oriented and the burst length is programmed to 8. It determines the maximum number of column locations that can be accessed for a given write command. When the write command is issued, eight columns are selected. All accesses for that burst take place within these eight columns, thus the burst wraps within these eight columns if a boundary is reached. These eight columns are selected by addr[13:3]. addr[2:0] is used to select the starting location within the block.

In case of incrementing burst (INCR/INCR4/INCR8/INCR16), the addresses can cross the 16-byte boundary of the DDR-SDRAM device. For example, when a transfer (INCR4) starts at address 0x0C, the next access is 0x10, but since the burst length is programmed to 8, the next access is at 0x00. Since the boundary is reached, the burst is wrapped. The MPDDRC takes this feature of the DDR-SDRAM device into account. In case of a transfer starting at address 0x04/0x08/0x0C or starting at address 0x10/0x14/0x18/0x1C, two write commands are issued to avoid wrapping when the boundary is reached. The last write command is subject to DM input logic level. If DM is registered high, the corresponding data input is ignored and the write access is not done. This avoids additional writing.

Figure 33-2. Single Write Access, Row Closed, DDR-SDRAM Devices

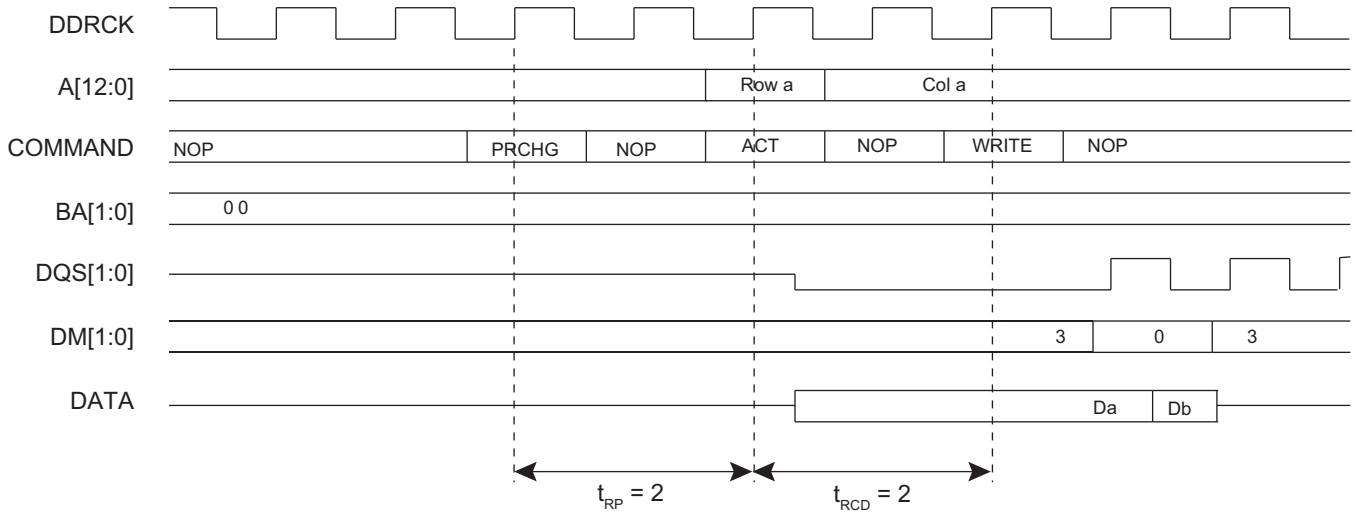


Figure 33-3. Single Write Access, Row Closed, DDR2-SDRAM Devices

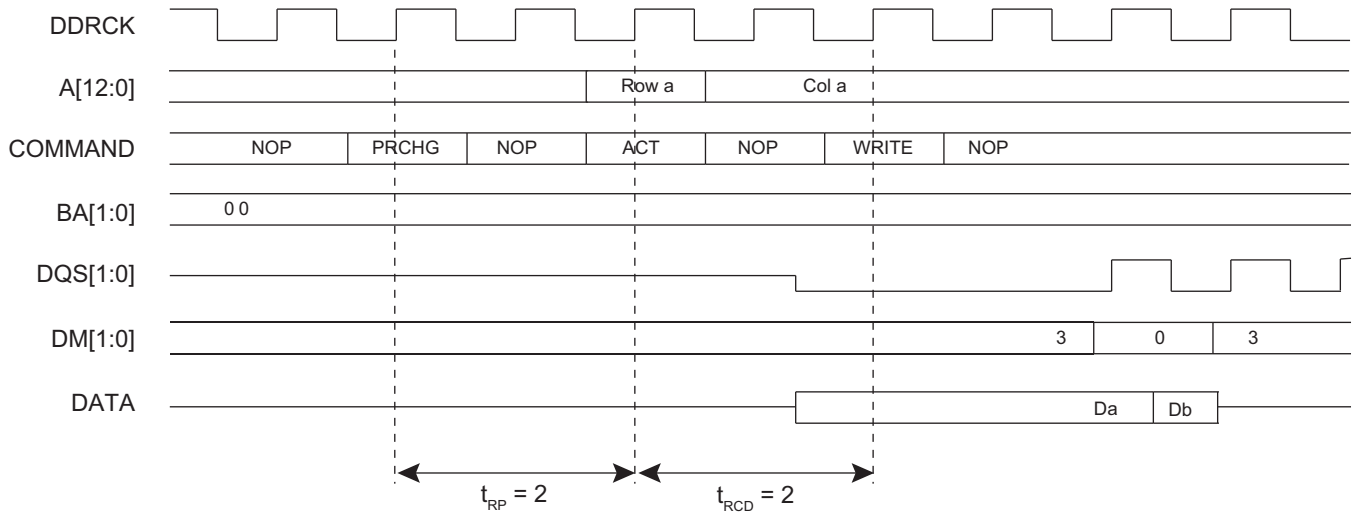


Figure 33-4. Burst Write Access, Row Closed, DDR-SDRAM Devices

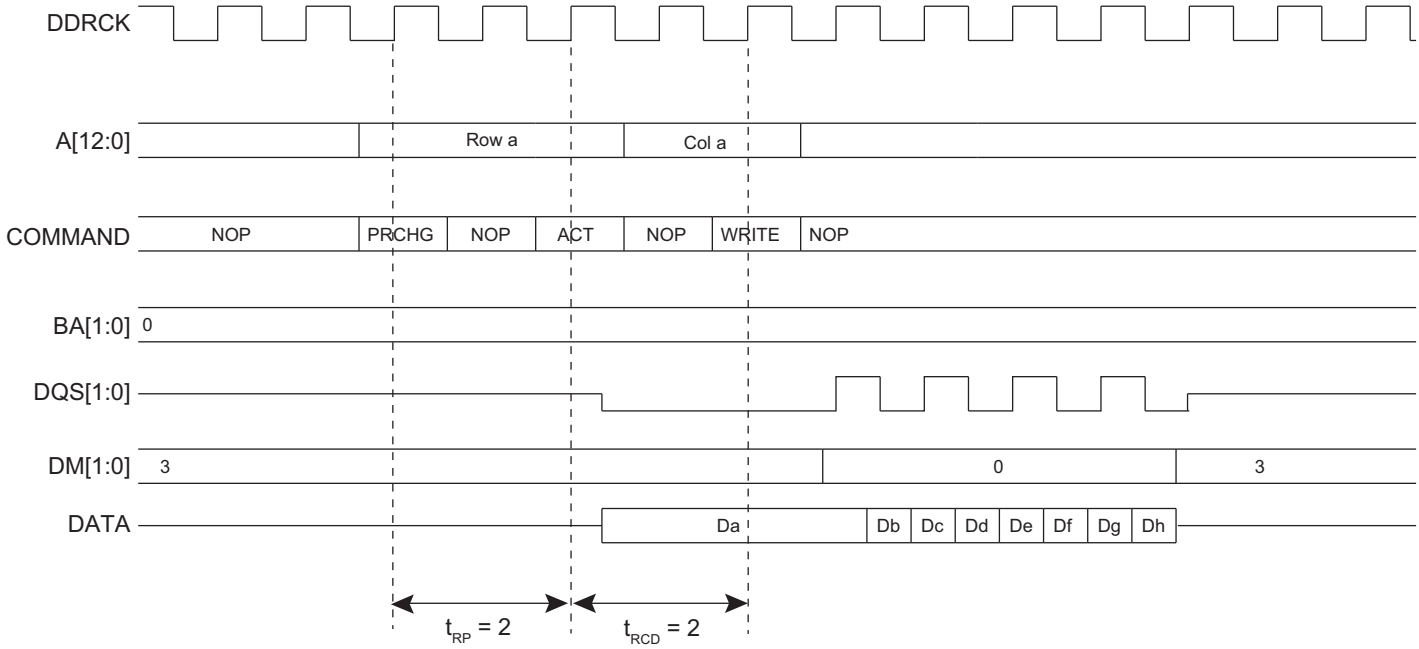
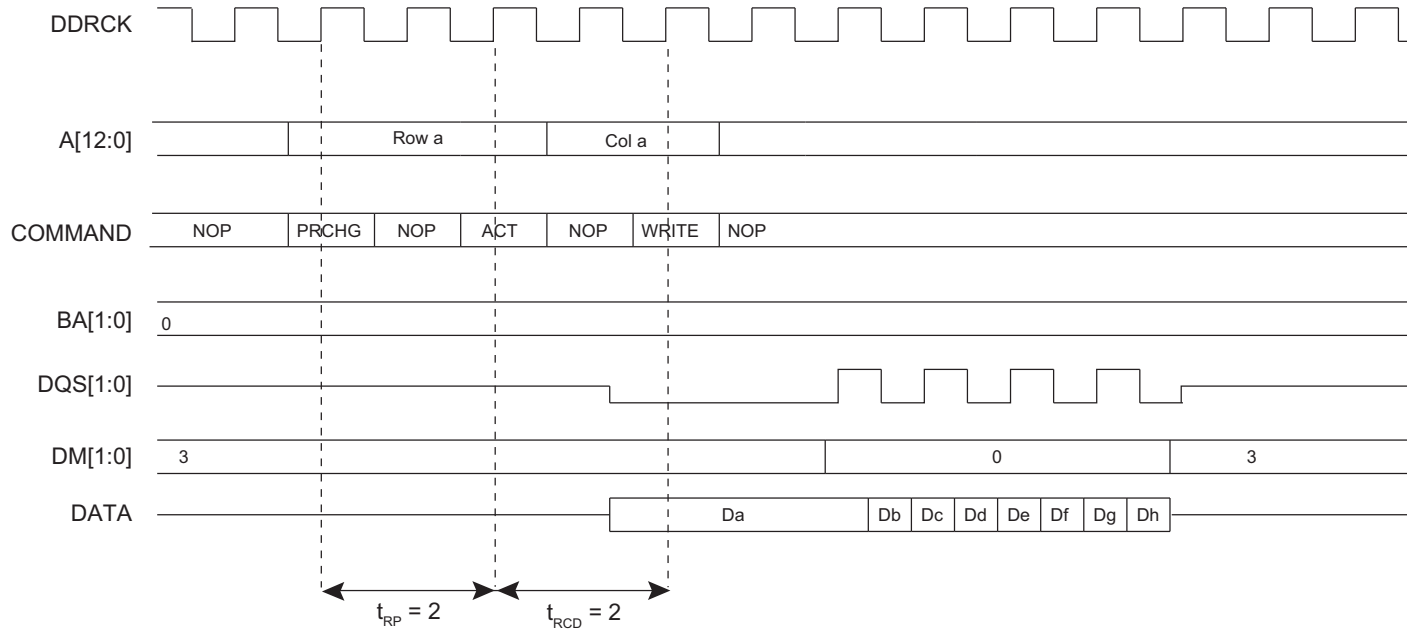
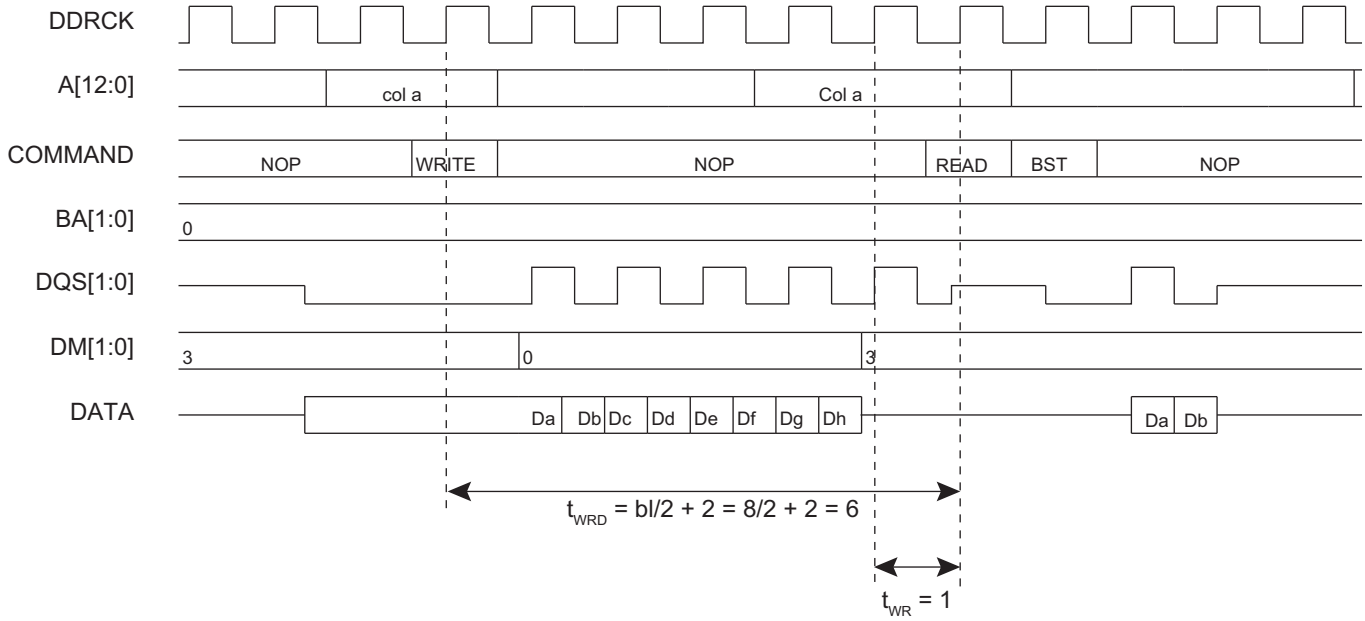


Figure 33-5. Burst Write Access, Row Closed, DDR2-SDRAM Devices



A write command can be followed by a read command. To avoid breaking the current write burst, t_{WTR}/t_{WRD} (bl/2 + 2 = 6 cycles) should be met. See Figure 33-6.

Figure 33-6. Write Command Followed by a Read Command without Burst Write Interrupt, DDR-SDRAM Devices



In case of a single write access, write operation should be interrupted by a read access but DM must be input 1 cycle prior to the read command to avoid writing invalid data. See Figure 33-7.

Figure 33-7. SINGLE Write Access Followed by a Read Access, DDR-SDRAM Devices

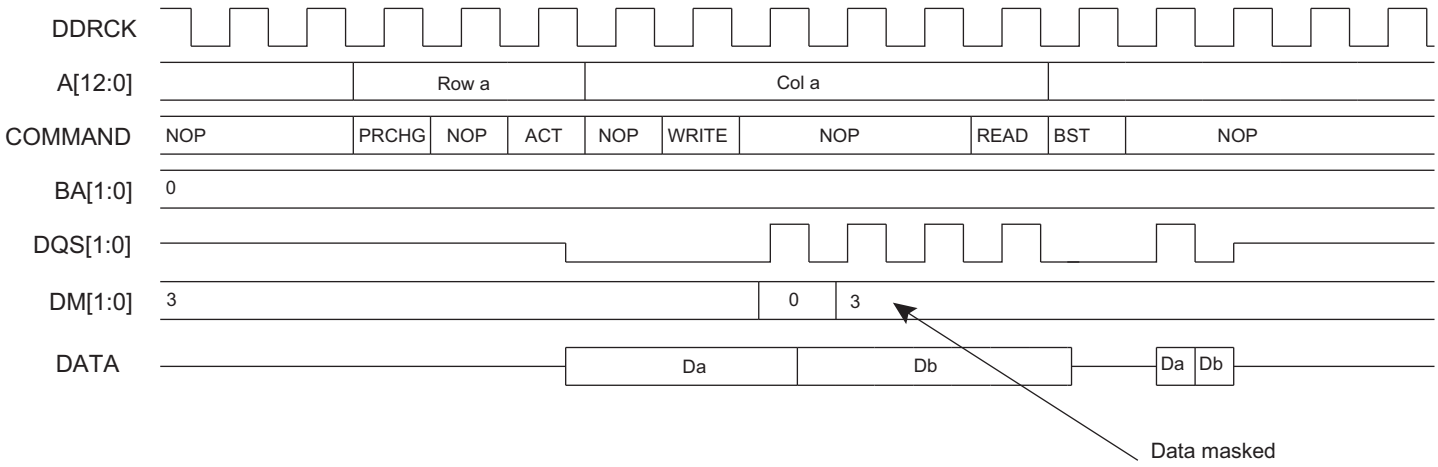
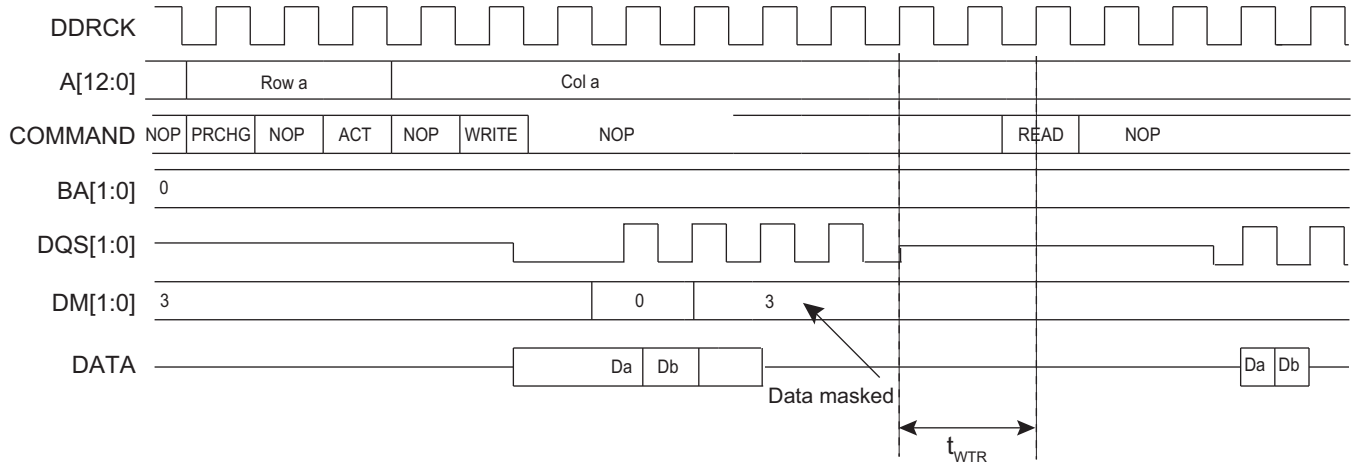


Figure 33-8. SINGLE Write Access Followed by a Read Access, DDR2-SDRAM Devices



33.5.2 DDR-SDRAM Controller Read Cycle

The MPDDRC provides burst access or single access in Normal mode (MPDDRC_MR.MODE = 0). Whatever the access type, the MPDDRC keeps track of the active row in each bank, thus maximizing performance of the MPDDRC.

The DDR-SDRAM devices are programmed with a burst length equal to 8 which determines the length of a sequential data output by the read command that is set to 8. The latency from read command to data output depends on the memory type, as shown in Table 33-2. This value is programmed during the initialization phase (see Section 33.4 “Product Dependencies, Initialization Sequence”).

Table 33-2. CAS Read Latency

Memory Devices	CAS Read Latency
Low-power DDR1-SDRAM	2/3
Low-power DDR2-SDRAM	3
DDR2-SDRAM	3
DDR3-SDRAM (DLL OFF)	5/6

Note: In the case of low-power DDR3-SDRAM, the CAS Read Latency (CRL) of 3 is optional. The MPDDRD supports this feature. Refer to the low-power DDR3-SDRAM datasheet for details.

To initiate a single access, the MPDDRC checks if the page access is already open. If row/bank addresses match with the previous row/bank addresses, the controller generates a read command. If the bank addresses are not identical or if bank addresses are identical but the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a read command. To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/read (t_{RCD}) command. After a read command, additional wait states are generated to comply with CAS latency. The MPDDRC supports a CAS latency of two to three (2 to 3 clock cycle delay). As the burst length is set to 8, in case of a single access or a burst access inferior to 8 data requests, it has to stop the burst, otherwise an additional seven or X values could be read. The Burst Stop command (BST) is used to stop output during a burst read. If the DDR2-SDRAM Burst Stop command is not supported by the JEDEC standard, in a single read access, an additional seven unwanted data will be read.

To initiate a burst access, the MPDDRC checks the transfer type signal. If the next accesses are sequential read accesses, reading to the SDRAM device is carried out. If the next access is a read non-sequential access, then an automatic page break can be inserted. If the bank addresses are not identical or if bank addresses are identical but

the row addresses are not identical, the controller generates a precharge command, activates the new row and initiates a read command. If page access is already open, a read command is generated.

To comply with DDR-SDRAM timing parameters, additional clock cycles are inserted between precharge/active (t_{RP}) commands and active/read (t_{RCD}) commands. The MPDDRC supports a CAS latency of two to three (2 to 3 clocks delay). During this delay, the controller uses internal signals to anticipate the next access and improve the performance of the controller. Depending on the latency, the MPDDRC anticipates two to three read accesses. In case of burst of specified length, accesses are not anticipated, but if the burst is broken (border, Busy mode, etc.), the next access is treated as an incrementing burst of unspecified length, and depending on the latency, the MPDDRC anticipates two to three read accesses.

For the definition of timing parameters, refer to Section 33.7.3 “MPDDRC Configuration Register”.

Read accesses to the DDR-SDRAM are burst oriented and the burst length is programmed to 8. The burst length determines the maximum number of column locations that can be accessed for a given read command. When the read command is issued, eight columns are selected. All accesses for that burst take place within these eight columns, meaning that the burst wraps within these eight columns if the boundary is reached. These eight columns are selected by addr[13:3]; addr[2:0] is used to select the starting location within the block.

In case of incrementing burst (INCR/INCR4/INCR8/INCR16), the addresses can cross the 16-byte boundary of the DDR-SDRAM device. For example, when a transfer (INCR4) starts at address 0x0C, the next access is 0x10, but since the burst length is programmed to 8, the next access is 0x00. Since the boundary is reached, the burst wraps. The MPDDRC takes into account this feature of the SDRAM device. In case of the DDR-SDRAM device, transfers start at address 0x04/0x08/0x0C. Two read commands are issued to avoid wrapping when the boundary is reached. The last read command may generate additional reading (1 read cmd = 4 DDR words).

To avoid additional reading, it is possible to use the burst stop command to truncate the read burst and to decrease power consumption. The DDR2-SDRAM devices do not support the burst stop command.

Figure 33-9. Single Read Access, Row Closed, Latency = 2, DDR-SDRAM Devices

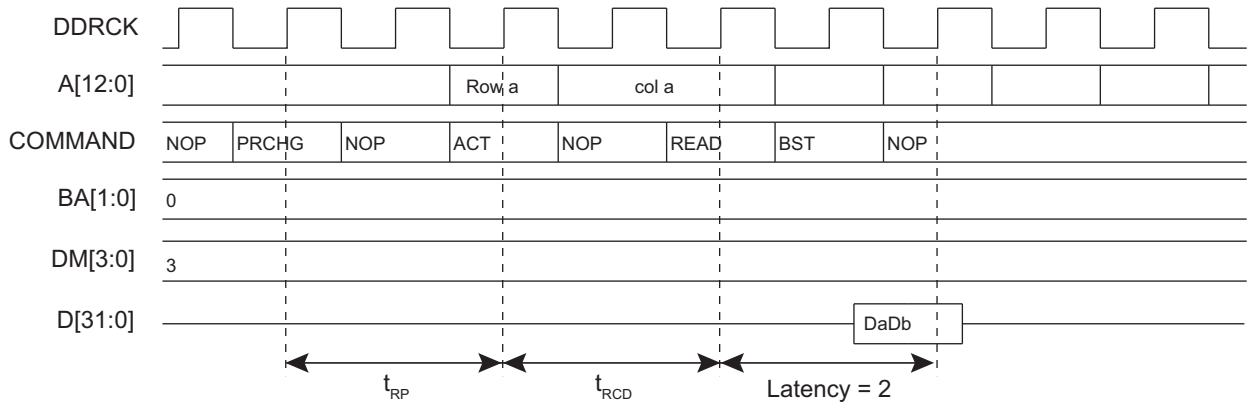


Figure 33-10. Single Read Access, Row Closed, Latency = 3, DDR2-SDRAM Devices

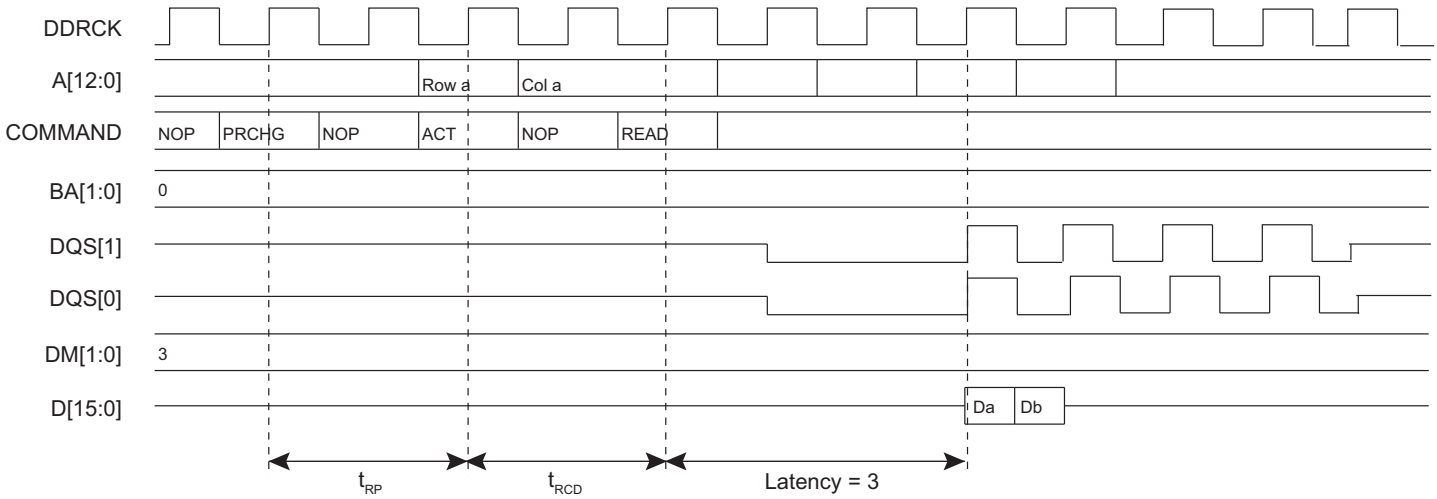


Figure 33-11. Burst Read Access, Latency = 2, DDR-SDRAM Devices

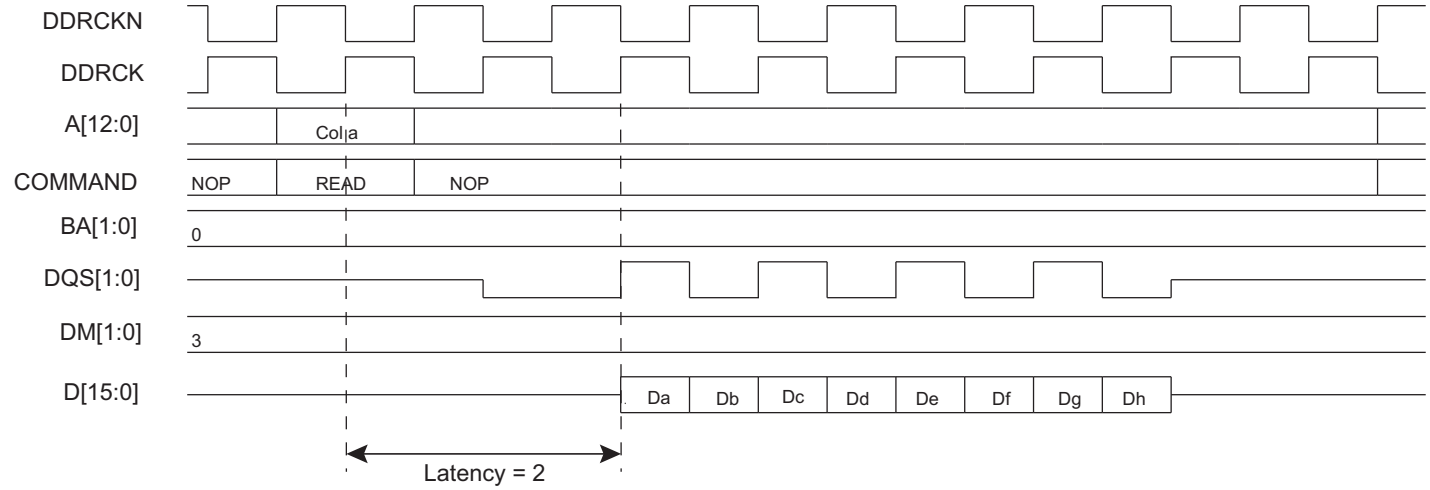
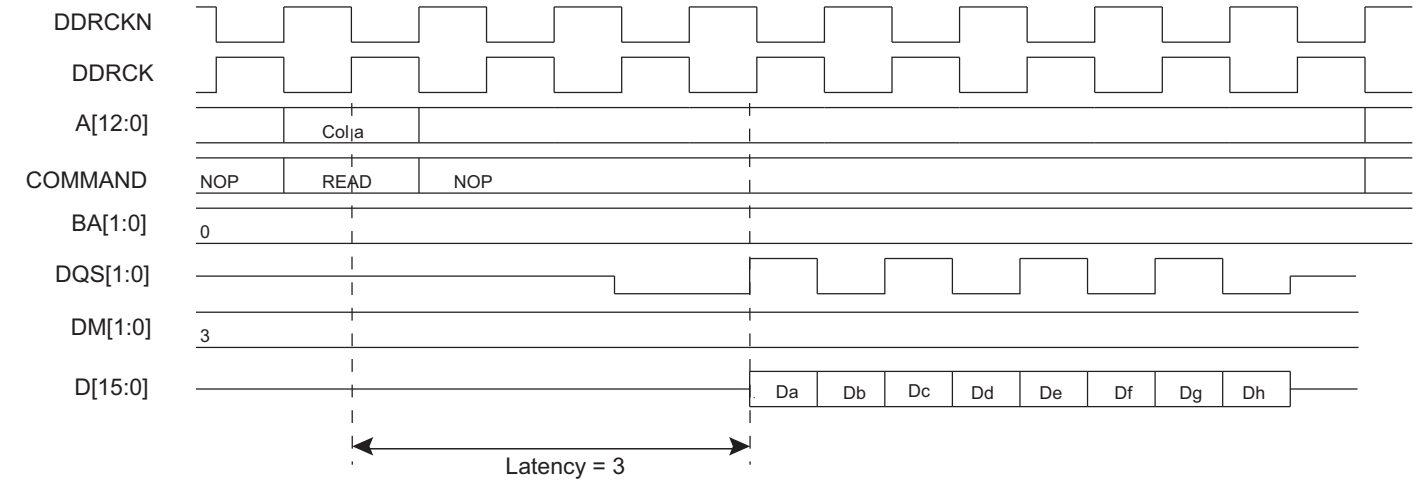


Figure 33-12. Burst Read Access, Latency = 3, DDR2-SDRAM Devices



33.5.2.1 All Banks Auto Refresh

The All Banks Auto Refresh command performs a refresh operation on all banks. An auto refresh command is used to refresh the external device. Refresh addresses are generated internally by the DDR-SDRAM device and incremented after each auto-refresh automatically. The MPDDRC generates these auto-refresh commands periodically. A timer is loaded in the MPDDRC_RTR with the value that indicates the number of clock cycles between refresh cycles (see Section 33.7.2 “MPDDRC Refresh Timer Register”). When the MPDDRC initiates a refresh of the DDR-SDRAM device, internal memory accesses are not delayed. However, if the CPU tries to access the DDR-SDRAM device, the slave indicates that the device is busy. A refresh request does not interrupt a burst transfer in progress. This feature is activated by setting Per-bank Refresh bit (REF_PB) to 0 in the MPDDRC_RTR (see Section 33.7.2 “MPDDRC Refresh Timer Register”).

33.5.2.2 Per-bank Auto Refresh

The low-power DDR2-SDRAM and low-power DDR3-SDRAM embeds a new Per-bank Refresh command which performs a refresh operation on the bank scheduled by the bank counter in the memory device. The Per-bank Refresh command is executed in a fixed sequence order of round-robin type: “0-1-2-3-4-5-6-7-0-1-...”. The bank counter is automatically cleared upon issuing a RESET command or when exiting from Self-refresh mode, in order to ensure the synchronism between SDRAM memory device and the MPDDRC. The bank addressing for the Per-bank Refresh count is the same as established in the Single-bank Precharge command. This feature is activated by setting the Per-bank Refresh bit (REF_PB) to 1 in the MPDDRC_RTR (see Section 33.7.2 “MPDDRC Refresh Timer Register”). This feature masks the latency due to the refresh procedure. The target bank is inaccessible during the Per-bank Refresh cycle period (t_{RFCpb}), however other banks within the device are accessible and may be addressed during the “Per-bank Refresh” cycle. During the REFpb operation, any bank other than the one being refreshed can be maintained in active state or accessed by a read or a write command. When the “Per-bank Refresh” cycle is completed, the affected bank will be in idle state.

33.5.2.3 Adjust Auto Refresh Rate

The low-power DDR2-SDRAM and low-power DDR3-SDRAM embeds an internal register, Mode Register 19 (Refresh mode). The content of this register allows to adjust the interval of auto-refresh operations according to temperature variation. This feature is activated by setting the Adjust Refresh bit [ADJ_REF] to 1 in the MPDDRC_RTR (see Section 33.7.2 “MPDDRC Refresh Timer Register”). When this feature is enabled, a Mode Register Read (MRR) command is performed every $16 \times t_{REFI}$ (average time between REFRESH commands). Depending on the read value, the auto refresh interval will be modified. In case of high temperature, the interval is reduced and in case of low temperature, the interval is increased.

33.5.3 Power Management

33.5.3.1 Self-refresh Mode

This mode is activated by writing a 1 to the Low-power Command bit (LPCB) in the MPDDRC Low-power Register (MPDDRC_LPR).

Self-refresh mode is used in Powerdown mode, i.e., when no access to the DDR-SDRAM device is possible. In this case, power consumption is very low. In Self-refresh mode, the DDR-SDRAM device retains data without external clocking and provides its own internal clocking, thus performing its own auto refresh cycles. During the self-refresh period, CKE is driven low. As soon as the DDR-SDRAM device is selected, the MPDDRC provides a sequence of commands and exits Self-refresh mode.

The MPDDRC re-enables Self-refresh mode as soon as the DDR-SDRAM device is not selected. It is possible to define when Self-refresh mode is to be enabled by configuring the TIMEOUT field in the MPDDRC_LPR:

0: Self-refresh mode is enabled as soon as the DDR-SDRAM device is not selected.

1: Self-refresh mode is enabled 64 clock cycles after completion of the last access.

2: Self-refresh mode is enabled 128 clock cycles after completion of the last access.

This controller also interfaces the low-power DDR-SDRAM. To optimize power consumption, the Low Power DDR SDRAM provides programmable self-refresh options comprised of Partial Array Self Refresh (full, half, quarter and 1/8 and 1/16 array).

Disabled banks are not refreshed in Self-refresh mode. This feature permits to reduce the self-refresh current. In case of low-power DDR1-SDRAM, the Extended Mode register controls this feature. It includes Temperature Compensated Self-refresh (TSCR) and Partial Array Self-refresh (PASR) parameters and the drive strength (DS) (see Section 33.7.7 “MPDDRC Low-power Register”). In case of low-power DDR2-SDRAM and low-power DDR3-SDRAM, the Mode Registers 16 and 17 control this feature, including PASR Bank Mask (BK_MASK) and PASR Segment Mask (SEG_MASK) parameters and drives strength (DS) (see Section 33.7.9 “MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register”). These parameters are set during the initialization phase. After initialization, as soon as the PASR/DS/TCSR fields or BK_MASK/SEG_MASK/DS are modified, the memory device Extended Mode Register or Mode Registers 3/16/17 are automatically accessed. Thus if MPDDRC does not share an external bus with another controller, PASR/DS/TCSR and BK_MASK/SEG_MASK/DS bits are updated before entering Self-refresh mode or during a refresh command. If MPDDRC does share an external bus with another controller, PASR/DS/TCSR and BK_MASK/SEG_MASK/DS bits are also updated during a pending read or write access. This type of update depends on the UPD_MR bit (see Section 33.7.7 “MPDDRC Low-power Register”).

The low-power DDR1-SDRAM must remain in Self-refresh mode during the minimum of TRFC periods (see Section 33.7.5 “MPDDRC Timing Parameter 1 Register”), and may remain in Self-refresh mode for an indefinite period.

The DDR2-SDRAM must remain in Self-refresh mode during the minimum of t_{CKE} periods (see the memory device datasheet), and may remain in Self-refresh mode for an indefinite period.

The low-power DDR2-SDRAM and low-power DDR3-SDRAM must remain in Self-refresh mode for the minimum of t_{CKESR} periods (see the memory device datasheet) and may remain in Self-refresh mode for an indefinite period.

The DDR3-SDRAM must remain in Self-refresh mode for the minimum of t_{CKESR} periods (see the memory device datasheet) and may remain in Self-refresh mode for an indefinite period.

Figure 33-13. Self-refresh Mode Entry, TIMEOUT = 0

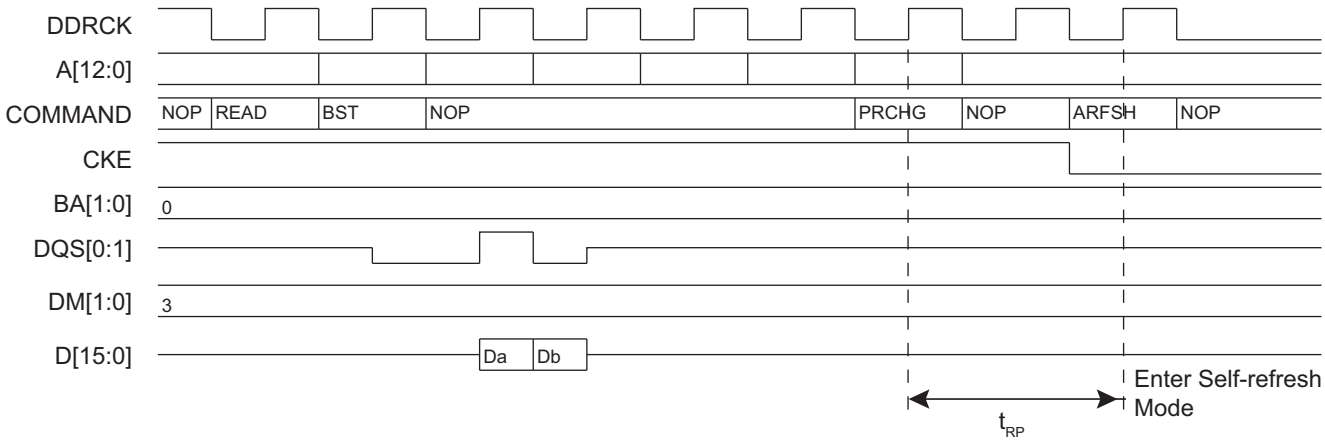


Figure 33-14. Self-refresh Mode Entry, TIMEOUT = 1 or 2

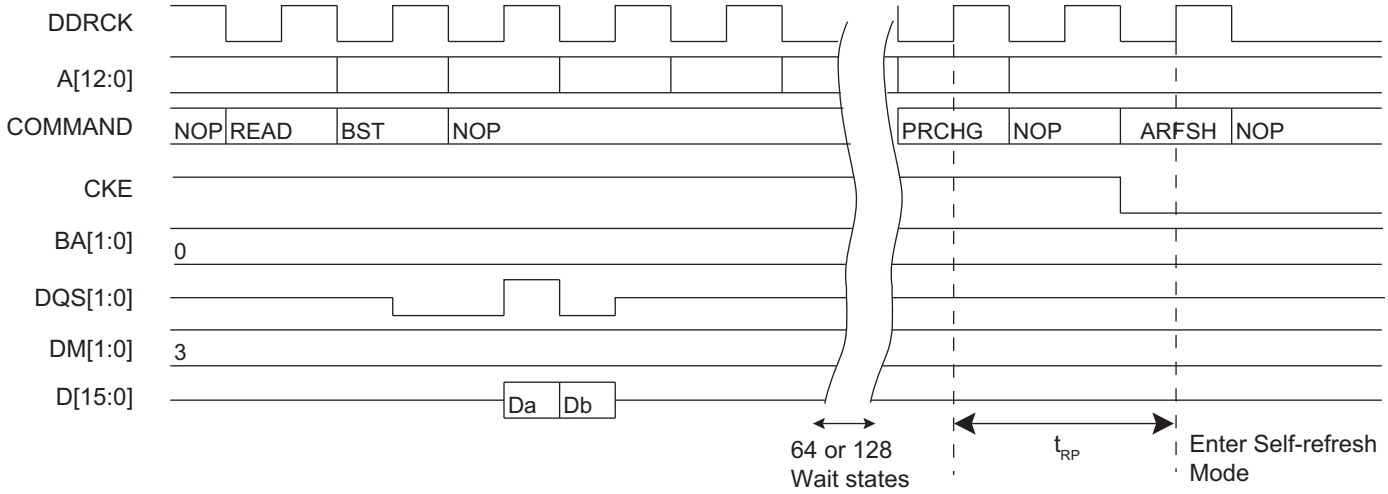
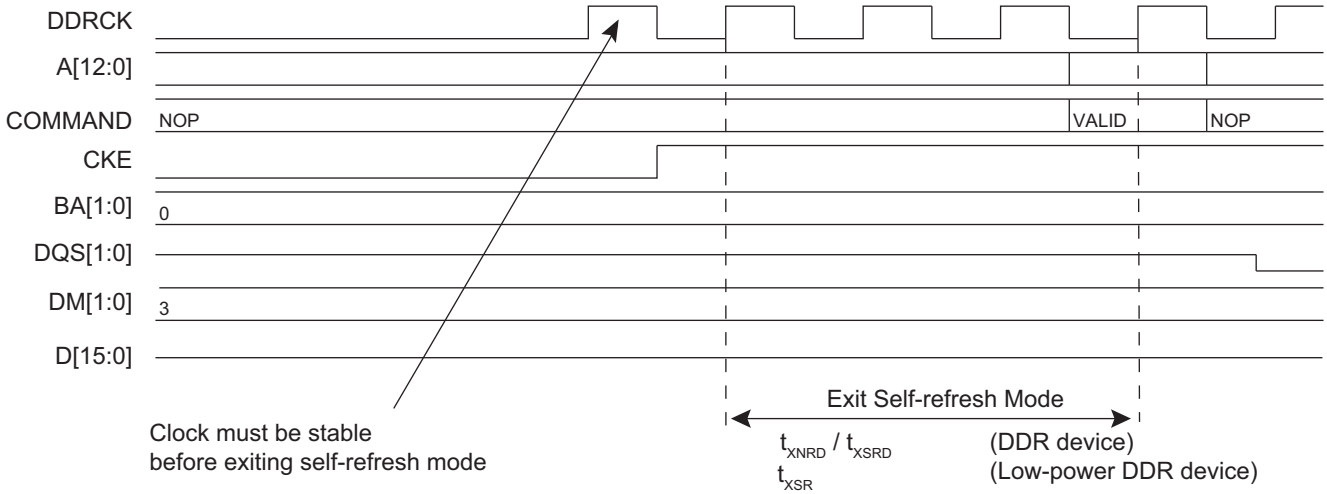


Figure 33-15. Self-refresh Mode Exit



33.5.3.2 Powerdown Mode

This mode is activated by writing a 10 to the Low-power Command bit (LPCB).

Powerdown mode is used when no access to the DDR-SDRAM device is possible. In this mode, power consumption is greater than in Self-refresh mode. This state is similar to Normal mode (no Low-power mode/no Self-refresh mode), but the CKE pin is low and the input and output buffers are deactivated as soon the DDR-SDRAM device is no longer accessible. In contrast to Self-refresh mode, the DDR-SDRAM device cannot remain in Low-power mode longer than one refresh period (64 ms/32 ms). As no auto-refresh operations are performed in this mode, the MPDDRC carries out the refresh operation. For the low-power DDR-SDRAM devices, a NOP command must be generated for a minimum period defined in the TXP field of the MPDDRC Timing Parameter 1 Register (MPDDRC_TPR1). For DDR-SDRAM devices, a NOP command must be generated for a minimum period defined in the TXP field of MPDDRC_TPR1 (see Section 33.7.5 “MPDDRC Timing Parameter 1 Register”) and in the TXARD and TXARDS fields of MPDDRC_TPR2 (see Section 33.7.6 “MPDDRC Timing Parameter 2 Register”) for DDR2_SDRAM devices. In addition, low-power DDR-SDRAM and DDR-SDRAM must remain in Powerdown mode for a minimum period corresponding to t_{CKE}, t_{PD}, etc. (see the memory device datasheet).

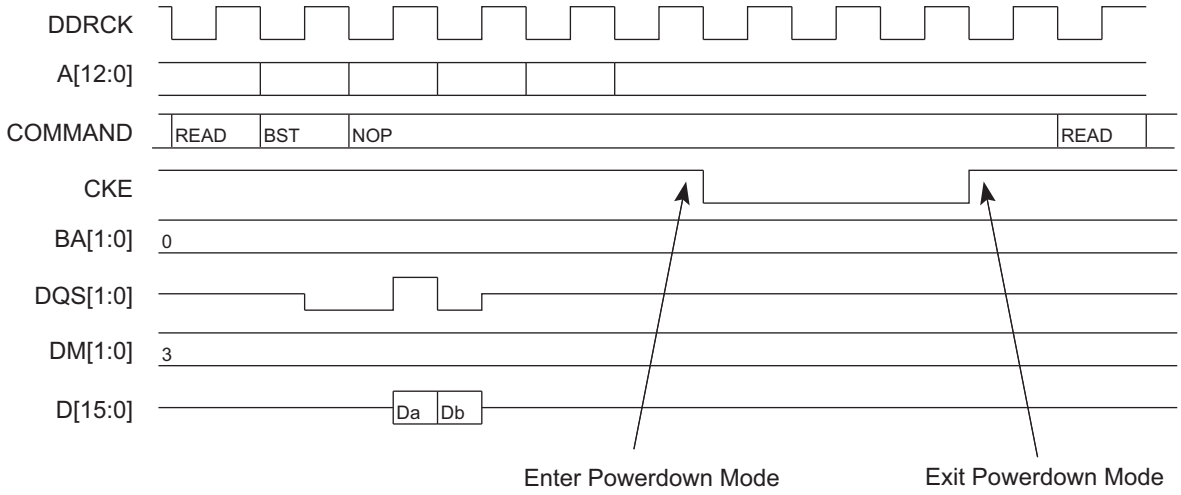
The exit procedure is faster than in Self-refresh mode. See Figure 33-16. The MPDDRC returns to Powerdown mode as soon as the DDR-SDRAM device is not selected. It is possible to define when Powerdown mode is enabled by configuring the TIMEOUT field in the MPDDRC_LPR:

0: Powerdown mode is enabled as soon as the DDR-SDRAM device is not selected.

1: Powerdown mode is enabled 64 clock cycles after completion of the last access.

2: Powerdown mode is enabled 128 clock cycles after completion of the last access.

Figure 33-16. Powerdown Entry/Exit, TIMEOUT = 0



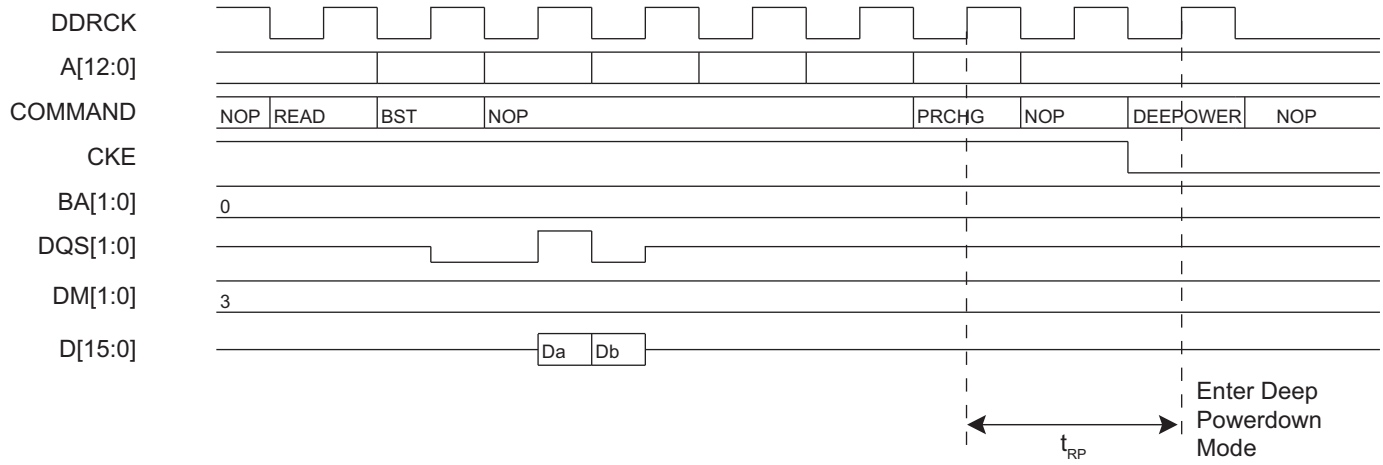
33.5.3.3 Deep Powerdown Mode

The Deep Powerdown mode is a feature of low-power DDR-SDRAM. When this mode is activated, all internal voltage generators inside the device are stopped and all data is lost.

Deep Powerdown mode is activated by writing a 3 to the Low-power Command bit (LPCB). When this mode is enabled, the MPDDRC leaves Normal mode (MPDDRC_MR.MODE = 0) and the controller is frozen. The clock can be stopped during Deep Powerdown mode by setting the CLK_FR field to 1.

Before enabling this mode, the user must make sure there is no access in progress. To exit Deep Powerdown mode, the Low-power Command bit (LPCB) and clock frozen bit (CLK_FR) must be written to zero and the initialization sequence must be generated by software. See Section 33.4.1 “Low-power DDR1-SDRAM Initialization” or Section 33.4.3 “Low-power DDR2-SDRAM Initialization” or Section 33.4.5 “Low-power DDR3-SDRAM Initialization”.

Figure 33-17. Deep Powerdown Mode Entry



33.5.3.4 Change Frequency During Self-Refresh Mode with Low-power DDR-SDRAM Devices and DDR3-SDRAM

To change frequency, Self-refresh mode must be activated by writing a 1 to the Low-power Command bit (LPCB) and a one to the Change Frequency Command bit (CHG_FR) in the MPDDRC Low-power Register.

Once the low-power DDR-SDRAM is in Self-refresh mode, the user must make sure there is no access in progress. Then, the user can change the clock frequency. The device input clock frequency changes only within minimum and maximum operating frequencies as specified by the low-power DDR-SDRAM and DDR3-SDRAM providers. Once the input clock frequency is changed, new stable clocks must be provided to the device before exiting from Self-refresh mode.

To exit from Self-refresh mode, the DDR-SDRAM device must be selected. The MPDDRC provides a sequence of commands and exits Self-refresh mode.

During a change frequency procedure, the Change Frequency Command bit (CHG_FR) is set to 0 automatically. The Enable Read Measure feature is not supported during a change frequency procedure (see “ENRDM: Enable Read Measure”).

It is not possible to change the frequency with DDR2-SDRAM devices.

33.5.3.5 Reset Mode

The Reset mode is a feature of DDR2-SDRAM. This mode is activated by writing a 3 to the Low-power Command bit (LPCB) and a one to the Clock Frozen Command bit (CLK_FR) in the MPDDRC Low-power Register.

When this mode is enabled, the MPDDRC leaves Normal mode (MPDDRC_MR.MODE = 0) and the controller is frozen. Before enabling this mode, the user must make sure there is no access in progress.

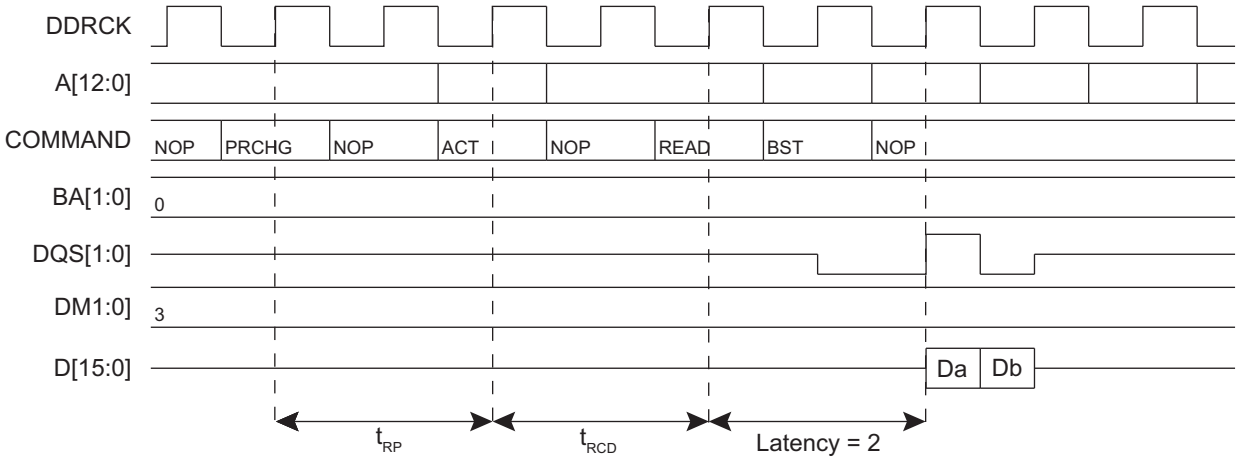
To exit Reset mode, the Low-power Command bit (LPCB) must be written to zero, the Clock Frozen Command bit (CLK_FR) must be written to zero and the initialization sequence must be generated by software (see Section 33.4.2 “DDR2-SDRAM Initialization”).

33.5.4 Multiport Functionality

The DDR-SDRAM protocol imposes a check of timings prior to performing a read or a write access, thus decreasing system performance. An access to DDR-SDRAM is performed if banks and rows are open (or active). To activate a row in a particular bank, the last open row must be deactivated and a new row must be open. Two DDR-SDRAM commands must be performed to open a bank: Precharge command and Activate command with respect to t_{RP} timing. Before performing a read or write command, t_{RCD} timing must be checked.

This operation generates a significant bandwidth loss (see Figure 33-18).

Figure 33-18. t_{RP} and t_{RCD} Timings

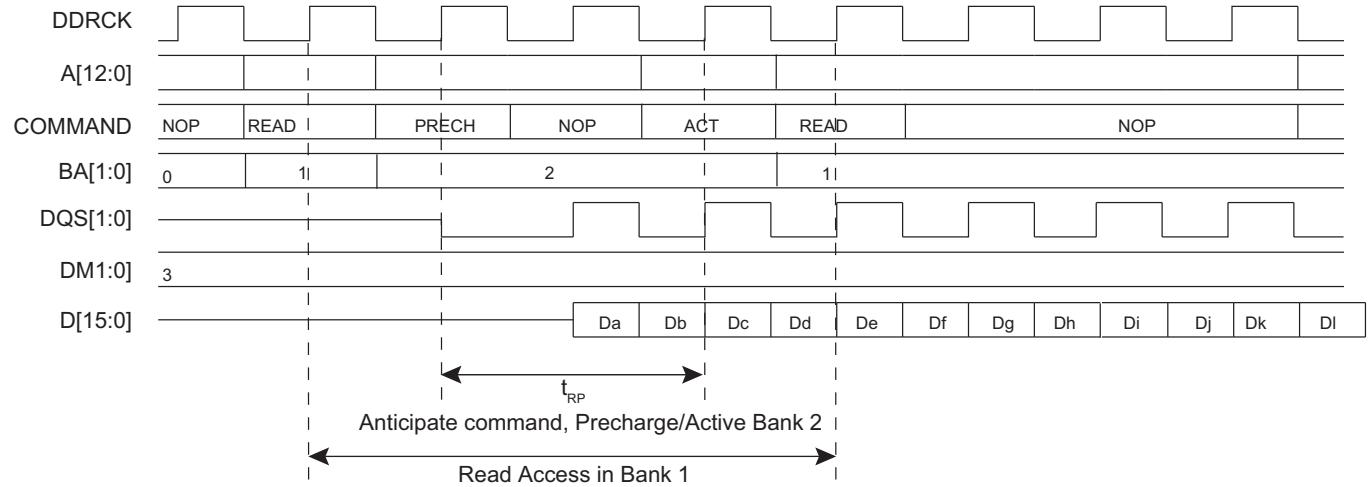


4 cycles before performing a read command

The multiport controller is designed to mask these timings and thus improve the bandwidth of the system.

The MPDDRC is a multiport controller whereby eight masters can simultaneously reach the controller. This feature improves the bandwidth of the system because it can detect eight requests on the AHB slave inputs and thus anticipate the commands that follow, Precharge and Activate command in bank X during the current access in bank Y. This masks t_{RP} and t_{RCD} timings (see Figure 33-19). In the best case, all accesses are done as if the banks and rows were already open. The best condition is met when the eight masters work in different banks. In case of eight simultaneous read accesses, when the four or eight banks and associated rows are open, the controller reads with a continuous flow and masks the CAS latency for each access. To allow a continuous flow, the read command must be set at 2 or 3 cycles (CAS latency) before the end of the current access. This requires that the scheme of arbitration changes since arbitration cannot be respected. If the controller anticipates a read access, and thus a master with a high priority arises before the end of the current access, then this master will not be serviced.

Figure 33-19. Anticipate Precharge/Activate Command in Bank 2 during Read Access in Bank 1



MPDDRC is a multiport controller that embeds three arbitration mechanisms based on round -robin arbitration which allows to share the external device between different masters when two or more masters try to access the DDR-SDRAM device at the same time.

The three arbitration types are round-robin arbitration and two weighted round-robin arbitrations. For weighted round-robin arbitrations, the priority can be given either depending on the number of requests or words per port, or depending on the required bandwidth per port. The type of arbitration can be chosen by setting the ARB field in the MPDDRC Configuration Arbiter Register (MPDDRC_CONF_ARBITER) (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

33.5.4.1 Round-robin Arbitration

Round-robin arbitration is used when the ARB field is set to 0 (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”). This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner.

To avoid burst breaking and to provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Idle cycles: when no master is connected to the DDR-SDRAM device.
2. Single cycles: when a slave is currently doing a single access.
3. End of Burst cycles: when the current cycle is the last cycle of a burst transfer:
   - For bursts of defined length, predicted end of burst matches the size of the transfer.
   - For bursts of undefined length, predicted end of burst is generated at the end of each four-beat boundary inside the INCR transfer.
4. Anticipated Access: when an anticipated read access is done while the current access is not complete, the arbitration scheme can be changed if the anticipated access is not the next access serviced by the arbitration scheme.

33.5.4.2 Request-word Weighted Round-robin Arbitration

In request-word weighted round-robin arbitration, the weight is the number of requests or the number of words per port.

This arbitration scheme is enabled by writing a 1 to the ARB field (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”). This algorithm grants a port for X[(1)] consecutive first transfer (htrans = NON SEQUENTIAL) of a burst or X single transfer, or for X word transfers. It is possible to chose between an arbitration scheme by request or by word per port by setting the RQ_WD_Px field (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

Note: 1. X is an integer value provided by some master modules to the arbiter.

It is also possible for the user to provide the number of requests or words (by overwriting the information provided by a master) on master basis by configuring the MA_PR_Px field. Depending on the application, it is possible to reduce or increase the number of these requests or words by configuring the NRD_NWD_BDW_Px fields (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

The TIMEOUT_Px field defines the delay between two accesses on the same port in number of cycles before to arbitrate and to give the hand to another port. This field allows to avoid a timeout on the system because some masters have the particularity to add idle cycles between two consecutive accesses (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner when the number of requests or words is reached or when the timeout value is reached.

To avoid burst breaking and to provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Timeout is reached: the delay between two accesses is equal to TIMEOUT_Px.
2. Number of requests or words is reached: when the current cycle is the last cycle of a transfer.

33.5.4.3 Bandwidth Weighted Round-robin Arbitration

In bandwidth weighted round-robin arbitration, a minimum bandwidth is guaranteed per port.

This arbitration scheme is enabled when the ARB field is set to 2 (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

This algorithm grants to each port a percentage of the bandwidth. The NRD_NWD_BDW_Px field defines the percentage allocated to each port.

The percentage of the bandwidth is programmed with the NRD_NWD_BDW_Px fields (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

The TIMEOUT_Px field defines the delay between two accesses on the same port in number of cycles before to arbitrate and to give the hand to another port. This field allows to avoid a timeout on the system because some masters have the particularity to add idle cycles between two consecutive accesses (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

This algorithm dispatches the requests from different masters to the DDR-SDRAM device in a round-robin manner. If two or more master requests arise at the same time, the master with the lowest number is serviced first, then the others are serviced in a round-robin manner when the allocated bandwidth is reached or when the timeout value is reached.

The BDW_BURST field allows to arbitrate either when the current master reaches exactly the programmed bandwidth, or when the current master reaches exactly the programmed bandwidth and the current access is ended (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

To provide the maximum throughput for the DDR-SDRAM device, arbitration must only take place during the following cycles:

1. Timeout is reached: the delay between two accesses is equal to TIMEOUT_Px.
2. Allocated Bandwidth is reach although the current cycle is not ended.
3. Allocated Bandwidth is reach and the current cycle is the last cycle of a transfer

33.5.5 Scrambling/Unscrambling Function

The external data bus can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling method depends on two user-configurable key registers, MPDDRC_KEY1 in the “MPDDRC OCMS KEY1 Register” and MPDDRC_KEY2 in the “MPDDRC OCMS KEY2 Register” . These key registers are only accessible in Write mode.

The key must be securely stored in a reliable non-volatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

The scrambling/unscrambling function can be enabled or disabled by programming the “MPDDRC OCMS Register” .

33.5.6 Register Write Protection

To prevent any single software error from corrupting MPDDRC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the MPDDRC Write Protection Mode Register (MPDDRC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the MPDDRC Write Protection Status Register (MPDDRC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the MPDDRC_WPSR.

The following registers can be write-protected:

- MPDDRC Mode Register
- MPDDRC Refresh Timer Register
- MPDDRC Configuration Register
- MPDDRC Timing Parameter 0 Register
- MPDDRC Timing Parameter 1 Register
- MPDDRC Memory Device Register
- MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register
- MPDDRC OCMS Register
- MPDDRC OCMS KEY1 Register
- MPDDRC OCMS KEY2 Register

33.6 Software Interface/SDRAM Organization, Address Mapping

The DDR-SDRAM address space is organized into banks, rows and columns. The MPDDRC maps different memory types depending on values set in the MPDDRC Configuration Register (see Section 33.7.3 “MPDDRC Configuration Register”). The tables that follow illustrate the relation between CPU addresses and columns, rows and banks addresses for 16-bit memory data bus widths and 32-bit memory data bus widths.

The MPDDRC supports address mapping in Linear mode.

Sequential mode is a method for address mapping where banks alternate at each last DDR-SDRAM page of the current bank.

Interleaved mode is a method for address mapping where banks alternate at each DDR-SDRAM end of page of the current bank.

The MPDDRC makes the DDR-SDRAM device access protocol transparent to the user. The tables that follow illustrate the DDR-SDRAM device memory mapping seen by the user in correlation with the device structure. Various configurations are illustrated.

33.6.1 DDR-SDRAM Address Mapping for 16-bit Memory Data Bus Width

Table 33-3. Sequential Mapping for DDR-SDRAM Configuration, 2K Rows, 256/512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Bk[1:0]		Row[10:0]											Column[7:0]								M0
					Bk[1:0]		Row[10:0]												Column[8:0]								M0
				Bk[1:0]		Row[10:0]													Column[9:0]								M0
			Bk[1:0]		Row[10:0]														Column[10:0]								M0
		Bk[1:0]		Row[10:0]															Column[11:0]								M0

Table 33-4. Interleaved Mapping for DDR-SDRAM Configuration, 2K Rows, 256/512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
							Row[10:0]										Bk[1:0]		Column[7:0]								M0
						Row[10:0]										Bk[1:0]			Column[8:0]								M0
					Row[10:0]										Bk[1:0]				Column[9:0]								M0
				Row[10:0]										Bk[1:0]					Column[10:0]								M0
			Row[10:0]										Bk[1:0]						Column[11:0]								M0

Table 33-5. Sequential Mapping for DDR-SDRAM Configuration: 4K Rows, 256/512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Bk[1:0]		Row[11:0]												Column[7:0]								M0
				Bk[1:0]		Row[11:0]													Column[8:0]								M0
			Bk[1:0]		Row[11:0]														Column[9:0]								M0
		Bk[1:0]		Row[11:0]															Column[10:0]								M0
	Bk[1:0]		Row[11:0]																Column[11:0]								M0

Table 33-6. Interleaved Mapping for DDR-SDRAM Configuration: 4K Rows, 256/512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Row[11:0]											Bk[1:0]		Column[7:0]								M0
					Row[11:0]											Bk[1:0]			Column[8:0]								M0
				Row[11:0]											Bk[1:0]				Column[9:0]								M0
			Row[11:0]											Bk[1:0]					Column[10:0]								M0
		Row[11:0]											Bk[1:0]						Column[11:0]								M0

Table 33-7. Sequential Mapping for DDR-SDRAM Configuration: 8K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
			Bk[1:0]	Row[12:0]														Column[8:0]									M0
		Bk[1:0]	Row[12:0]															Column[9:0]									M0
	Bk[1:0]	Row[12:0]																Column[10:0]									M0
Bk[1:0]	Row[12:0]																	Column[11:0]									M0

Table 33-8. Interleaved Mapping for DDR-SDRAM Configuration: 8K Rows, 512/1024/2048/4096 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
			Row[12:0]													Bk[1:0]		Column[8:0]									M0
		Row[12:0]													Bk[1:0]			Column[9:0]									M0
	Row[12:0]													Bk[1:0]				Column[10:0]									M0
Row[12:0]													Bk[1:0]					Column[11:0]									M0

Table 33-9. Sequential Mapping for DDR-SDRAM Configuration: 16K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
		Bk[1:0]	Row[13:0]															Column[8:0]									M0
	Bk[1:0]	Row[13:0]																Column[9:0]									M0
Bk[1:0]	Row[13:0]																	Column[10:0]									M0

Table 33-10. Interleaved Mapping for DDR-SDRAM Configuration: 16K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
		Row[13:0]														Bk[1:0]		Column[8:0]									M0
	Row[13:0]														Bk[1:0]			Column[9:0]									M0
Row[13:0]														Bk[1:0]				Column[10:0]									M0

Table 33-11. Sequential Mapping for DDR-SDRAM Configuration: 8K Rows,1024/ Columns, 8 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Bk[2:0]			Row[12:0]														Column[9:0]									M0

Table 33-12. Interleaved Mapping for DDR-SDRAM Configuration: 8K Rows,1024/ Columns, 8 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Row[12:0]													Bk[2:0]				Column[9:0]									M0

Table 33-13. Sequential Mapping for DDR-SDRAM Configuration: 16K Rows,1024/ Columns, 8 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Bk[2:0]			Row[13:0]														Column[9:0]										M0

Table 33-14. Interleaved Mapping for DDR-SDRAM Configuration: 16K Rows,1024/ Columns, 8 banks

CPU Address Line																										
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Row[13:0]														Bk[2:0]			Column[9:0]										M0

33.6.2 DDR-SDRAM Address Mapping for 32-bit Memory Data Bus Width

Table 33-15. Sequential Mapping DDR-SDRAM Configuration Mapping: 2K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Bk[1:0]		Row[10:0]											Column[8:0]										M[1:0]	
				Bk[1:0]			Row[10:0]											Column[9:0]										M[1:0]	
			Bk[1:0]				Row[10:0]											Column[10:0]										M[1:0]	

Table 33-16. Interleaved Mapping DDR-SDRAM Configuration Mapping: 2K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Row[10:0]											Bk[1:0]		Column[8:0]										M[1:0]	
				Row[10:0]											Bk[1:0]			Column[9:0]										M[1:0]	
			Row[10:0]											Bk[1:0]				Column[10:0]										M[1:0]	

Table 33-17. Sequential Mapping DDR-SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Bk[1:0]		Row[11:0]											Column[7:0]										M[1:0]	
				Bk[1:0]			Row[11:0]											Column[8:0]										M[1:0]	
			Bk[1:0]				Row[11:0]											Column[9:0]										M[1:0]	
		Bk[1:0]					Row[11:0]											Column[10:0]										M[1:0]	

Table 33-18. Interleaved Mapping DDR-SDRAM Configuration Mapping: 4K Rows, 256/512/1024/2048 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
					Row[11:0]												Bk[1:0]		Column[7:0]									M[1:0]	
				Row[11:0]												Bk[1:0]			Column[8:0]									M[1:0]	
			Row[11:0]												Bk[1:0]				Column[9:0]									M[1:0]	
		Row[11:0]												Bk[1:0]					Column[10:0]									M[1:0]	

Table 33-19. Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows, 512/1024/2048 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
			Bk[1:0]				Row[12:0]											Column[8:0]										M[1:0]	
		Bk[1:0]					Row[12:0]											Column[9:0]										M[1:0]	
	Bk[1:0]						Row[12:0]											Column[10:0]										M[1:0]	

Table 33-20. Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows /512/1024/2048 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
			Row[13:0]													Bk[1:0]	Column[8:0]										M[1:0]	
		Row[13:0]													Bk[1:0]	Column[9:0]											M[1:0]	
	Row[13:0]													Bk[1:0]	Column[10:0]												M[1:0]	

Table 33-21. Sequential Mapping DDR-SDRAM Configuration Mapping: 8K Rows /1024 Columns, 8 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Bk[2:0]			Row[12:0]													Column[9:0]										M[1:0]	

Table 33-22. Interleaved Mapping DDR-SDRAM Configuration Mapping: 8K Rows /1024 Columns, 8 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Row[12:0]													Bk[2:0]			Column[9:0]										M[1:0]	

Table 33-23. Sequential Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Bk[1:0]		Row[13:0]														Column[9:0]										M[1:0]	

Table 33-24. Interleaved Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 4 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
	Row[13:0]														Bk[1:0]		Column[9:0]										M[1:0]	

Table 33-25. Sequential Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 8 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Bk[2:0]			Row[13:0]														Column[9:0]										M[1:0]	

Table 33-26. Interleaved Mapping DDR-SDRAM Configuration Mapping: 16K Rows /1024 Columns, 8 banks

CPU Address Line																											
28	**27**	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Row[13:0]														Bk[2:0]			Column[9:0]										M[1:0]	

33.6.3 DDR-SDRAM Address Mapping for Low-cost Memories

Table 33-27. Sequential Mapping for DDR-SDRAM Configuration, 2K Rows, 512 Columns, 2 banks, 16 bits

CPU Address Line																									
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Bk	Row[10:0]											Column[8:0]									M0

Table 33-28. Interleaved Mapping for DDR-SDRAM Configuration, 2K Rows, 512 Columns, 2 banks, 16 bits

CPU Address Line																									
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Row[10:0]											Bk	Column[8:0]									M0

Table 33-29. Sequential Mapping for DDR-SDRAM Configuration: 4K Rows, 256 Columns, 2 banks, 32 bits

CPU Address Line																									
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Bk	Row[11:0]											Column[7:0]								M[1:0]	

Table 33-30. Interleaved Mapping for DDR-SDRAM Configuration: 4K Rows, 256 Columns, 2 banks, 32 bits

CPU Address Line																									
27	**26**	**25**	**24**	**23**	**22**	**21**	**20**	**19**	**18**	**17**	**16**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
						Row[11:0]											Bk	Column[7:0]								M[1:0]	

Notes: 1. M[1:0] is the byte address inside a 32-bit word.
2. Bk[2] = BA2, Bk[1] = BA1, Bk[0] = BA0

33.7 AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface

The User Interface is connected to the APB bus. The MPDDRC is programmed using the registers listed in Table 33-31.

Table 33-31. Register Mapping

Offset	Register	Name	Access	Reset
0x00	MPDDRC Mode Register	MPDDRC_MR	Read/Write	0x00000000
0x04	MPDDRC Refresh Timer Register	MPDDRC_RTR	Read/Write	0x03000000
0x08	MPDDRC Configuration Register	MPDDRC_CR	Read/Write	0x00207024
0x0C	MPDDRC Timing Parameter 0 Register	MPDDRC_TPR0	Read/Write	0x20227225
0x10	MPDDRC Timing Parameter 1 Register	MPDDRC_TPR1	Read/Write	0x3C80808
0x14	MPDDRC Timing Parameter 2 Register	MPDDRC_TPR2	Read/Write	0x00042062
0x18	Reserved	–	–	–
0x1C	MPDDRC Low-power Register	MPDDRC_LPR	Read/Write	0x00010000
0x20	MPDDRC Memory Device Register	MPDDRC_MD	Read/Write	0x13
0x24	Reserved	–	–	–
0x28	MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register	MPDDRC_LPDDR23_LPR	Read/Write	0x00000000
0x2C	MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register	MPDDRC_LPDDR2_LPDDR3_DDR3_CAL_MR4	Read/Write	0x00000000
0x30	MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register	MPDDRC_LPDDR2_LPDDR3_DDR3_TIM_CAL	Read/Write	0x06
0x34	MPDDRC I/O Calibration Register	MPDDRC_IO_CALIBR	Read/Write	0x00870000
0x38	MPDDRC OCMS Register	MPDDRC_OCMS	Read/Write	0x00000000
0x3C	MPDDRC OCMS KEY1 Register	MPDDRC_OCMS_KEY1	Write-only	–
0x40	MPDDRC OCMS KEY2 Register	MPDDRC_OCMS_KEY2	Write-only	–
0x44	MPDDRC Configuration Arbiter Register	MPDDRC_CONF_ARBITER	Read/Write	0x00000000
0x48	MPDDRC Timeout Register	MPDDRC_TIMEOUT	Read/Write	0x00000000
0x4C	MPDDRC Request Port 0-1-2-3 Register	MPDDRC_REQ_PORT_0123	Read/Write	0x00000000
0x50	MPDDRC Request Port 4-5-6-7 Register	MPDDRC_REQ_PORT_4567	Read/Write	0x00000000
0x54	MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register	MPDDRC_BDW_PORT_0123	Read-only	0x00000000
0x58	MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register	MPDDRC_BDW_PORT_4567	Read-only	0x00000000
0x5C	MPDDRC Read Data Path Register	MPDDRC_RD_DATA_PATH	Read/Write	0x00000000
0x60	MPDDRC Monitor Configuration Register	MPDDRC_MCFGR	Read/Write	0x00000000
0x64	MPDDRC Monitor Address High/Low Port 0 Register	MPDDRC_MADDR0	Read/Write	0x00000000
0x68	MPDDRC Monitor Address High/Low Port 1 Register	MPDDRC_MADDR1	Read/Write	0x00000000
0x6C	MPDDRC Monitor Address High/Low Port 2 Register	MPDDRC_MADDR2	Read/Write	0x00000000
0x70	MPDDRC Monitor Address High/Low Port 3 Register	MPDDRC_MADDR3	Read/Write	0x00000000

Table 33-31. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x74	MPDDRC Monitor Address High/Low Port 4 Register	MPDDRC_MADDR4	Read/Write	0x00000000
0x78	MPDDRC Monitor Address High/Low Port 5 Register	MPDDRC_MADDR5	Read/Write	0x00000000
0x7C	MPDDRC Monitor Address High/Low Port 6 Register	MPDDRC_MADDR6	Read/Write	0x00000000
0x80	MPDDRC Monitor Address High/Low Port 7 Register	MPDDRC_MADDR7	Read/Write	0x00000000
0x84	MPDDRC Monitor Information Port 0 Register	MPDDRC_MINFO0	Read-only	0x00000000
0x88	MPDDRC Monitor Information Port 1 Register	MPDDRC_MINFO1	Read-only	0x00000000
0x8C	MPDDRC Monitor Information Port 2 Register	MPDDRC_MINFO2	Read-only	0x00000000
0x90	MPDDRC Monitor Information Port 3 Register	MPDDRC_MINFO3	Read-only	0x00000000
0x94	MPDDRC Monitor Information Port 4 Register	MPDDRC_MINFO4	Read-only	0x00000000
0x98	MPDDRC Monitor Information Port 5 Register	MPDDRC_MINFO5	Read-only	0x00000000
0x9C	MPDDRC Monitor Information Port 6 Register	MPDDRC_MINFO6	Read-only	0x00000000
0xA0	MPDDRC Monitor Information Port 7 Register	MPDDRC_MINFO7	Read-only	0x00000000
0xA4–0xE0	Reserved	–	–	–
0xE4	MPDDRC Write Protection Mode Register	MPDDRC_WPMR	Read/Write	0x00000000
0xE8	MPDDRC Write Protection Status Register	MPDDRC_WPSR	Read-only	0x00000000
0xEC–0x1FC	Reserved	–	–	–

33.7.1 MPDDRC Mode Register

Name: MPDDRC_MR

Address: 0xF000C000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
MRS							
7	6	5	4	3	2	1	0
–	–	–	DAI	–	MODE		

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **MODE: MPDDRC Command Mode**

This field defines the command issued by the MPDDRC when the SDRAM device is accessed. This register is used to initialize the SDRAM device and to activate Deep Powerdown mode.

Value	Name	Description
0	NORMAL_CMD	Normal Mode. Any access to the MPDDRC is decoded normally. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
1	NOP_CMD	The MPDDRC issues a NOP command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
2	PRCGALL_CMD	The MPDDRC issues the All Banks Precharge command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the SDRAM.
3	LMR_CMD	The MPDDRC issues a Load Mode Register command when the DDR-SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
4	RFSH_CMD	The MPDDRC issues an Auto-Refresh command when the DDR-SDRAM device is accessed regardless of the cycle. Previously, an All Banks Precharge command must be issued. To activate this mode, the command must be followed by a write to the DDR-SDRAM.
5	EXT_LMR_CMD	The MPDDRC issues an Extended Load Mode Register command when the SDRAM device is accessed regardless of the cycle. To activate this mode, the command must be followed by a write to the DDR-SDRAM. The write in the DDR-SDRAM must be done in the appropriate bank.
6	DEEP_CALIB_MD	Deep Power mode: Access to Deep Powerdown mode Calibration command: to calibrate RTT and RON values for the Process Voltage Temperature (PVT) (DDR3-SDRAM device)
7	LPDDR2_LPDDR3_CMD	The MPDDRC issues an LPDDR2/LPDDR3 Mode Register command when the device is accessed regardless of the cycle. To activate this mode, the Mode Register command must be followed by a write to the low-power DDR2-SDRAM or to the low-power DDR3-SDRAM.

- **DAI: Device Auto-Initialization Status (read-only)**

Reset value is 1.

This field reports when the device auto-initialization is complete.

Value	Name	Description
0	DAI_COMPLETE	DAI complete
1	DAI_IN_PROGESSS	DAI still in progress

- **MRS: Mode Register Select LPDDR2/LPDDR3**

Configure this 8-bit field to program all mode registers included in the low-power DDR2-SDRAM device. This field is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

33.7.2 MPDDRC Refresh Timer Register

Name: MPDDRC_RTR

Address: 0xF000C004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	MR4_VALUE			–	–	REF_PB	ADJ_REF

15	14	13	12	11	10	9	8
–	–	–	–	COUNT			

7	6	5	4	3	2	1	0
COUNT							

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **COUNT: MPDDRC Refresh Timer Count**

This 12-bit field is loaded into a timer which generates the refresh pulse. Each time the refresh pulse is generated, a refresh sequence is initiated.

The SDRAM devices require a refresh of all rows every 64 ms. The value to be loaded depends on the MPDDRC clock frequency (MCK: Master Clock) and the number of rows in the device.

For example, for an SDRAM with 8192 rows and a 100 MHz Master clock, the value of the COUNT field is configured: $((64 \times 10^{-3}) / 8192) \times 100 \times 10^{6} = 781$ or 0x030D.

Low-power DDR2-SDRAM and low-power DDR3-SDRAM devices support Per Bank Refresh operation. In this configuration, average time between refresh command is 0.975 µs. The value of the COUNT field is configured depending on this value. For example, the value of a 100 MHz Master clock refresh timer is 98 or 0x0062.

- **ADJ_REF: Adjust Refresh Rate**

Reset value is 0.

0: Adjust refresh rate is not enabled.

1: Adjust refresh rate is enabled.

This mode is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

- **REF_PB: Refresh Per Bank**

Reset value is 0.

0: Refresh all banks during auto-refresh operation.

1: Refresh the scheduled bank by the bank counter in the memory interface.

This mode is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

- **MR4_VALUE: Content of MR4 Register (read-only)**

Reset value is 3.

This field gives the content of MR4 register. This field is updated when MRR command is generated and Adjust Refresh Rate bit is enabled. Update is done when read value is different from MR4_VALUE.

LPDDR2 and LPDDR3 JEDEC memory standards impose derating LPDDR2/LPDDR3 AC timings (t_{RCD}, t_{RC}, t_{RAS}, t_{RP} and t_{RRD}) when the value of MR4 is equal to 6. If the application needs to work in extreme conditions, the derating value must be added to AC timings before the power up sequence.

This mode is unique to the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

33.7.3 MPDDRC Configuration Register

Name: MPDDRC_CR

Address: 0xF000C008

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
UNAL	DECOD	NDQS	NB	LC_LPDDR1	–	ENRDM	DQMS

15	14	13	12	11	10	9	8
–	OCD			ZQ		DIS_DLL	DIC_DS

7	6	5	4	3	2	1	0
DLL	CAS			NR		NC	

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **NC: Number of Column Bits**

Reset value is 0 (9/8 column bits).

Value	Name	Description
0	DDR9_MDDR8_COL_BITS	9 bits for DDR, 8-bit for low-power DDR1-SDRAM
1	DDR10_MDDR9_COL_BITS	10 bits for DDR, 9-bit for low-power DDR1-SDRAM
2	DDR11_MDDR10_COL_BITS	11 bits for DDR, 10-bit for low-power DDR1-SDRAM
3	DDR12_MDDR11_COL_BITS	12 bits for DDR, 11-bit for low-power DDR1-SDRAM

- **NR: Number of Row Bits**

Reset value is 1 (12 row bits).

Value	Name	Description
0	11_ROW_BITS	11 bits to define the row number, up to 2048 rows
1	12_ROW_BITS	12 bits to define the row number, up to 4096 rows
2	13_ROW_BITS	13 bits to define the row number, up to 8192 rows
3	14_ROW_BITS	14 bits to define the row number, up to 16384 rows

- **CAS: CAS Latency**

Reset value is 2 (2 cycles).

Value	Name	Description
2	DDR_CAS2	LPDDR1 CAS Latency 2
3	DDR_CAS3	LPDDR3/DDR2/LPDDR2/LPDDR1 CAS Latency 3
5	DDR_CAS5	DDR3 CAS Latency 5
6	DDR_CAS6	DDR3LPDDR3 CAS Latency 6

In the case of DDR3-SDRAM devices, the CAS field must be set to 5 and the SHIFT_SAMPLING field must be set to 2. See “SHIFT_SAMPLING: Shift Sampling Point of Data” . This field is not used to set the DDR3-SDRAM. In the case of

DDR3-SDRAM devices, the DLL Off mode sets the CAS Read Latency (CRL) and the CAS Write Latency (CWL) to 6. The latency is automatically set by the controller.

- **DLL: Reset DLL**

Reset value is 0.

This bit defines the value of Reset DLL.

Value	Name	Description
0	RESET_DISABLED	Disable DLL reset
1	RESET_ENABLED	Enable DLL reset

This value is used during the powerup sequence.

This bit is found only in the DDR2-SDRAM devices and DDR3-SDRAM devices.

- **DIC_DS: Output Driver Impedance Control (Drive Strength)**

Reset value is 0.

This bit name is described as “DS” in some memory datasheets. It defines the output drive strength. This value is used during the powerup sequence. For DDR3-SDRAM devices, this field is equivalent to ODS, Output Drive Strength.

Value	Name	Description
0	DDR2_NORMALSTRENGTH_DDR3_RZQ/6	Normal drive strength (DDR2) - RZQ/6 (40 [NOM], DDR3)
1	DDR2_WEAKSTRENGTH_DDR3_RZQ/7	Weak drive strength (DDR2) - RZQ/7 (34 [NOM], DDR3)

This bit is found only in the DDR2-SDRAM devices and DDR3-SDRAM devices.

- **DIS_DLL: DISABLE DLL**

Reset value is 0.

0: Enable DLL.

1: Disable DLL.

This value is used during the powerup sequence. It is only found in the DDR2-SDRAM devices and DDR3-SDRAM devices and low-power DDR3-SDRAM devices.

- **ZQ: ZQ Calibration**

Reset value is 0.

Value	Name	Description
0	INIT	Calibration command after initialization
1	LONG	Long calibration
2	SHORT	Short calibration
3	RESET	ZQ Reset

This parameter is used to calibrate DRAM On resistance (Ron) values over PVT.

This field is found only in the low-power DDR2-SDRAM devices and low-power DDR3-SDRAM devices.

- **OCD: Off-chip Driver**

Reset value is 7.

Note: SDRAM Controller supports only two values for OCD (default calibration and exit from calibration). These values MUST always be programmed during the initialization sequence. The default calibration must be programmed first, after which the exit calibration and maintain settings must be programmed.

This field is found only in the DDR2-SDRAM devices.

Value	Name	Description
0	DDR2_EXITCALIB	Exit from OCD Calibration mode and maintain settings
7	DDR2_DEFAULT_CALIB	OCD calibration default

- **DQMS: Mask Data is Shared**

Reset value is 0.

Value	Name	Description
0	NOT_SHARED	DQM is not shared with another controller
1	SHARED	DQM is shared with another controller

- **ENRDM: Enable Read Measure**

Reset value is 0.

Value	Name	Description
0	OFF	DQS/DDR_DATA phase error correction is disabled
1	ON	DQS/DDR_DATA phase error correction is enabled

This feature is not supported during a change frequency. See “CHG_FRQ: Change Clock Frequency During Self-refresh Mode” .

- **LC_LPDDR1: Low-cost Low-power DDR1**

Reset value is 0.

Value	Name	Description
0	NOT_2_BANKS	Any type of memory devices except of low cost, low density Low Power DDR1.
1	2_BANKS_LPDDR1	Low-cost and low-density low-power DDR1. These devices have a density of 32 Mbits and are organized as two internal banks. To use this feature, the user has to define the type of memory and the data bus width (see Section 33.7.8 “MPDDRC Memory Device Register”). The 16-bit memory device is organized as 2 banks, 9 columns and 11 rows. The 32-bit memory device is organized as 2 banks, 8 columns and 11 rows. It is impossible to use two 16-bit memory devices (2 x 32 Mbits) for creating one 32-bit memory device (64 Mbits). In this case, it is recommended to use one 32-bit memory device which embeds four internal banks.

- **NB: Number of Banks**

Reset value is 0 (4 banks). If LC_LPDDR1 is set to 1, NB is not relevant.

Value	Name	Description
0	4_BANKS	4-bank memory devices
1	8_BANKS	8 banks. Only possible when using the DDR2-SDRAM and low-power DDR2-SDRAM and DDR3-SDRAM and low-power DDR3-SDRAM devices.

- **NDQS: Not DQS**

Reset value is 1 (Not DQS is disabled).

Value	Name	Description
0	ENABLED	Not DQS is enabled
1	DISABLED	Not DQS is disabled

This bit is found only in the DDR2-SDRAM devices.

- **DECOD: Type of Decoding**

Reset value is 0.

Value	Name	Description
0	SEQUENTIAL	Method for address mapping where banks alternate at each last DDR-SDRAM page of the current bank.
1	INTERLEAVED	Method for address mapping where banks alternate at each DDR-SDRAM end of page of the current bank.

- **UNAL: Support Unaligned Access**

Reset value is 0 (unaligned access is not supported).

Value	Name	Description
0	UNSUPPORTED	Unaligned access is not supported.
1	SUPPORTED	Unaligned access is supported.

This mode is enabled with masters which have an AXI interface.

33.7.4 MPDDRC Timing Parameter 0 Register

Name: MPDDRC_TPR0

Address: 0xF000C00C

Access: Read/Write

31	30	29	28	27	26	25	24
TMRD				TWTR			
23	22	21	20	19	18	17	16
TRRD				TRP			
15	14	13	12	11	10	9	8
TRC				TWR			
7	6	5	4	3	2	1	0
TRCD				TRAS			

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **TRAS: Active to Precharge Delay**

Reset value is 5 DDRCK(1) clock cycles.

This field defines the delay between an Activate command and a Precharge command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TRCD: Row to Column Delay**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between an Activate command and a Read/Write command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TWR: Write Recovery Delay**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the Write Recovery Time in number of DDRCK(1) clock cycles. The number of cycles is between 1 and 15.

- **TRC: Row Cycle Delay**

Reset value is 7 DDRCK(1) clock cycles.

This field defines the delay between an Activate command and a Refresh command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TRP: Row Precharge Delay**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between a Precharge command and another command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

- **TRRD: Active BankA to Active BankB**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between an Activate command in BankA and an Activate command in BankB in number of DDRCK(1) clock cycles. The number of cycles is between 1 and 15.

- **TWTR: Internal Write to Read Delay**

Reset value is 0.

This field defines the internal Write to Read command time in number of DDRCK[1] clock cycles. The number of cycles is between 1 and 7.

- **TMRD: Load Mode Register Command to Activate or Refresh Command**

Reset value is 2 DDRCK[1] clock cycles.

This field defines the delay between a Load mode register command and an Activate or Refresh command in number of DDRCK[1] clock cycles. The number of cycles is between 0 and 15. For low-power DDR2-SDRAM and low-power DDR3-SDRAM, this field is equivalent to TMRW timing.

Note: 1. DDRCK is the clock that drives the SDRAM device.

33.7.5 MPDDRC Timing Parameter 1 Register

Name: MPDDRC_TPR1

Address: 0xF000C010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	TXP			

23	22	21	20	19	18	17	16
TXSRD							

15	14	13	12	11	10	9	8
TXSNR							

7	6	5	4	3	2	1	0
–	TRFC						

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **TRFC: Row Cycle Delay**

Reset value is 8 DDRCK(1) clock cycles.

This field defines the delay between a Refresh command or a Refresh and Activate command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 127.

In case of low-power DDR2-SDRAM and low-power DDR3-SDRAM, this field is equivalent to t_{RFCab} timing. If the user enables the function “Refresh Per Bank” (see “REF_PB: Refresh Per Bank”), this field is equivalent to t_{RFCpb}.

- **TXSNR: Exit Self-refresh Delay to Non-Read Command**

Reset value is 8 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Non Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 255. This field is used by the DDR-SDRAM devices. In case of low-power DDR-SDRAM, this field is equivalent to t_{XSR} timing. In case of DDR3-SDRAM, this field is equivalent to t_{XS} timing.

- **TXSRD: Exit Self-refresh Delay to Read Command**

Reset value is 200 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 255.

This field is found only in DDR2-SDRAM and DDR3-SDRAM devices.

In case of DDR3-SDRAM, this field is equivalent to t_{XSDLL} timing. In DLL Off mode, this timing is not used. The field must be set to 0.

- **TXP: Exit Powerdown Delay to First Command**

Reset value is 3 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a valid command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

Note: 1. DDRCK is the clock that drives the SDRAM device.

33.7.6 MPDDRC Timing Parameter 2 Register

Name: MPDDRC_TPR2

Address: 0xF000C014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	TFAW			

15	14	13	12	11	10	9	8
–	TRTP			TRPA			

7	6	5	4	3	2	1	0
TXARDS				TXARD			

- **TXARD: Exit Active Powerdown Delay to Read Command in Mode “Fast Exit”**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

- **TXARDS: Exit Active Powerdown Delay to Read Command in Mode “Slow Exit”**

Reset value is 6 DDRCK(1) clock cycles.

This field defines the delay between CKE set high and a Read command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

- **TRPA: Row Precharge All Delay**

Reset value is 0 DDRCK(1) clock cycles.

This field defines the delay between a Precharge All Banks command and another command in number of DDRCK(1) clock cycles. The number of cycles is between 0 and 15.

This field is found only in the DDR2-SDRAM devices.

- **TRTP: Read to Precharge**

Reset value is 2 DDRCK(1) clock cycles.

This field defines the delay between a Read command and a Precharge command in number of DDRCK(1) clock cycles.

The number of cycles is between 0 and 7.

- **TFAW: Four Active Windows**

Reset value is 4 DDRCK(1) clock cycles.

DDR2 and DDR3 devices with eight banks (1 Gbit or larger) have an additional requirement concerning t_{FAW} timing. This requires that no more than four Activate commands may be issued in any given t_{FAW} (MIN) period.

The number of cycles is between 0 and 15.

This field is found only in DDR2-SDRAM and LPDDR2-SDRAM and DDR3-SDRAM and LPDDR3-SDRAM devices.

Note: 1. DDRCK is the clock that drives the SDRAM device.

33.7.7 MPDDRC Low-power Register

Name: MPDDRC_LPR

Address: 0xF000C01C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	SELF_DONE	CHG_FRQ

23	22	21	20	19	18	17	16
–	–	UPD_MR		–	–	–	APDE

15	14	13	12	11	10	9	8
–	–	TIMEOUT		–	DS		

7	6	5	4	3	2	1	0
–	PASR			LPDDR2_LPDDR3_PWOFF	CLK_FR	LPCB	

- **LPCB: Low-power Command Bit**

Reset value is 0.

Value	Name	Description
0	NOLOWPOWER	Low-power feature is inhibited. No Powerdown, Self-refresh and Deep-power modes are issued to the DDR-SDRAM device.
1	SELFREFRESH	The MPDDRC issues a self-refresh command to the DDR-SDRAM device, the clock(s) is/are deactivated and the CKE signal is set low. The DDR-SDRAM device leaves the Self-refresh mode when accessed and reenters it after the access.
2	POWERDOWN	The MPDDRC issues a Powerdown command to the DDR-SDRAM device after each access, the CKE signal is set low. The DDR-SDRAM device leaves the Powerdown mode when accessed and reenters it after the access.
3	DEEPPOWERDOWN	The MPDDRC issues a Deep Powerdown command to the low-power DDR-SDRAM device.

- **CLK_FR: Clock Frozen Command Bit**

Reset value is 0.

This field sets the clock low during Powerdown mode. Some DDR-SDRAM devices do not support freezing the clock during Powerdown mode. Refer to the relevant DDR-SDRAM device datasheet for details.

Value	Name	Description
0	DISABLED	Clock(s) is/are not frozen.
1	ENABLED	Clock(s) is/are frozen.

- **LPDDR2_LPDDR3_PWOFF: LPDDR2 - LPDDR3 Power Off Bit**

Reset value is 0.

LPDDR2/LPDDR3 power off sequence must be controlled to preserve the LPDDR2/LPDDR3 device. The power failure is handled at system level (IRQ or FIQ) and the LPDDR2/LPDDR3 power off sequence is applied using the LPDDR2_LPDDR3_PWOFF bit.

LPDDR2_LPDDR3_PWOFF bit is used to impose CKE low before a power off sequence. Uncontrolled power off sequence can be applied only up to 400 times in the life of a LPDDR2/LPDDR3 device.

Value	Name	Description
0	DISABLED	No power-off sequence applied to LPDDR2/LPDDR3.
1	ENABLED	A power-off sequence is applied to the LPDDR2/LPDDR3 device. CKE is forced low.

- **PASR: Partial Array Self-refresh**

Reset value is 0.

This field is unique to low-power DDR1-SDRAM. It is used to specify whether only one-quarter, one-half or all banks of the DDR-SDRAM array are enabled. Disabled banks are not refreshed in Self-refresh mode.

The values of this field are dependent on the low-power DDR-SDRAM devices.

After the initialization sequence, as soon as the PASR field is modified, the Extended Mode Register in the external device memory is accessed automatically and PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

- **DS: Drive Strength**

Reset value is 0.

This field is unique to low-power DDR1-SDRAM. It selects the output drive strength.

Value	Name	Description
0	DS_FULL	Full drive strength
1	DS_HALF	Half drive strength
2	DS_QUARTER	Quarter drive strength
3	DS_OCTANT	Octant drive strength
4–7	–	Reserved

After the initialization sequence, as soon as the DS field is modified, the Extended Mode Register is accessed automatically and DS bits are updated. Depending on the UPD_MR bit, update is done before entering self-refresh mode or during a refresh command and a pending read or write access.

- **TIMEOUT: Time Between Last Transfer and Low-Power Mode**

Reset value is 0.

This field defines when Low-power mode is activated.

Value	Name	Description
0	NONE	SDRAM Low-power mode is activated immediately after the end of the last transfer.
1	DELAY_64_CLK	SDRAM Low-power mode is activated 64 clock cycles after the end of the last transfer.
2	DELAY_128_CLK	SDRAM Low-power mode is activated 128 clock cycles after the end of the last transfer.
3	–	Reserved

- **APDE: Active Powerdown Exit Time**

Reset value is 1.

This mode is unique to the DDR2-SDRAM and DDR3-SDRAM devices.

This mode manages the active Powerdown mode which determines performance versus power saving.

Value	Name	Description
0	DDR2_FAST_EXIT	Fast Exit from Powerdown. DDR2-SDRAM and DDR3-SDRAM devices only.
1	DDR2_SLOW_EXIT	Slow Exit from Powerdown. DDR2-SDRAM and DDR3-SDRAM devices only.

After the initialization sequence, as soon as the APDE field is modified, the Extended Mode Register (located in the memory of the external device) is accessed automatically and APDE bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access

- **UPD_MR: Update Load Mode Register and Extended Mode Register**

Reset value is 0.

This bit is used to enable or disable automatic update of the Load Mode Register and Extended Mode Register. This update depends on the MPDDRC integration in a system. MPDDRC can either share or not an external bus with another controller.

Value	Name	Description
0	NO_UPDATE	Update of Load Mode and Extended Mode registers is disabled.
1	UPDATE_SHAREDBUS	MPDDRC shares an external bus. Automatic update is done during a refresh command and a pending read or write access in the SDRAM device.
2	UPDATE_NOSHAREDBUS	MPDDRC does not share an external bus. Automatic update is done before entering Self-refresh mode.
3	–	Reserved

- **CHG_FRQ: Change Clock Frequency During Self-refresh Mode**

Reset value is 0.

This mode allows to change the Low-power DDR-DRAM input clock frequency. This mode is unique to the Low-power DDR-DRAM devices.

- **SELF_DONE: Self-refresh is done (read-only)**

Reset value is 0.

This bit indicates that external device is in Self-refresh mode.

33.7.8 MPDDRC Memory Device Register

Name: MPDDRC_MD

Address: 0xF000C020

Access: Read/Write

31	30	29	28	27	26	25	24
IO_WIDTH		DENSITY				TYPE	

23	22	21	20	19	18	17	16
REV_ID							

15	14	13	12	11	10	9	8
MANU_ID							

7	6	5	4	3	2	1	0
RL3	WL	–	DBW	–	MD		

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **MD: Memory Device**

Indicates the type of memory used.

Reset value is that for the DDR-SDRAM device.

Value	Name	Description
3	LPDDR_SDRAM	Low-power DDR1-SDRAM
4	DDR3_SDRAM	DDR3-SDRAM
5	LPDDR3_SDRAM	Low-power DDR3-SDRAM
6	DDR2_SDRAM	DDR2-SDRAM
7	LPDDR2_SDRAM	Low-power DDR2-SDRAM

- **DBW: Data Bus Width**

Value	Name	Description
0	DBW_32_BITS	Data bus width is 32 bits
1	DBW_16_BITS	Data bus width is 16 bits.

- **WL: Write Latency (read-only)**

Reset value is 0.

This field gives the write latency supported by the memory device. This field is unique to low-power DDR3-SDRAM.

Value	Name	Description
0	WL_SETA	Write Latency Set A
1	WL_SETB	Write Latency Set B

- **RL3: Read Latency 3 Option Support (read-only)**

Reset value is 0.

This field gives information concerning the read latency supported. Read latency 3 has been defined per Jedec for frequency ≤ 166 MHz. This feature is optional. If the LPDDR3 device does not support this feature, a CAS latency of 6 is used. This field is unique to low-power DDR3-SDRAM.

Value	Name	Description
0	RL3_SUPPORT	Read latency of 3 is supported
1	RL3_NOT_SUPPORTED	Read latency 0f 3 is not supported

- **MANU_ID: Manufacturer Identification (read-only)**

Reset value is 0.

This field gives information concerning the Manufacturer ID. For more information concerning the Manufacturer ID, see document JC-42.6 “Manufacturer Identification (ID) Code for Low Power Memories”. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM.

- **REV_ID: Revision Identification (read-only)**

Reset value is 0.

This field gives the revision ID. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM.

- **TYPE: DRAM Architecture (read-only)**

Reset value is 0.

This field gives the DRAM architecture. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM.

Value	Name	Description
0	S4_SDRAM	4n prefetch architecture
1	S2_SDRAM	2n prefetch architecture
2	NVM	Non-volatile device
3	S8_SDRAM	8n prefetch architecture

- **DENSITY: Density of Memory (read-only)**

Reset value is 0.

This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM.

This field gives the density of the memory.

Value	Name	Description
0	DENSITY_64MBITS	The device density is 64 Mbits.
1	DENSITY_128MBITS	The device density is 128 Mbits.
2	DENSITY_256MBITS	The device density is 256 Mbits.
3	DENSITY_512MBITS	The device density is 512 Mbits.
4	DENSITY_1GBITS	The device density is 1 Gbit.
5	DENSITY_2GBITS	The device density is 2 Gbits.
6	DENSITY_4GBITS	The device density is 4 Gbits.
7	DENSITY_8GBITS	The device density is 8 Gbits.
8	DENSITY_16GBITS	The device density is 16 Gbits.
9	DENSITY_32GBITS	The device density is 32 Gbits.

- **IO_WIDTH: Width of Memory (read-only)**

Reset value is 0.

This field gives the width of the memory. This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM.

Value	Name	Description
0	WIDTH_32	The data bus width is 32 bits.
1	WIDTH_16	The data bus width is 16 bits.
2	WIDTH_8	The data bus width is 8 bits.
3	NOT_USED	–

33.7.9 MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register

Name: MPDDRC_LPDDR23_LPR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	DS			

23	22	21	20	19	18	17	16
SEG_MASK							

15	14	13	12	11	10	9	8
SEG_MASK							

7	6	5	4	3	2	1	0
BK_MASK_PASR							

- **BK_MASK_PASR: Bank Mask Bit/PASR**

Partial Array Self-Refresh (low-power DDR2-SDRAM-S4 devices and low-power DDR3-SDRAM only)

Reset value is 0.

After the initialization sequence, as soon as the BK_MASK_PASR field is modified, Mode Register 16 is accessed automatically and BK_MASK_PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

0: Refresh is enabled (= unmasked).

1: Refresh is disabled (= masked).

This mode is unique to the low-power DDR2-SDRAM-S4 and low-power DDR3-SDRAM devices. In Self-refresh mode, each bank of LPDDR2/LPDDR3 can be independently configured whether a self-refresh operation is taking place or not.

After the initialization sequence, as soon as the BK_MASK_PASR field is modified, the Extended Mode Register is accessed automatically and BK_MASK_PASR bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

- **SEG_MASK: Segment Mask Bit**

Reset value is 0.

After the initialization sequence, as soon as the SEG_MASK field is modified, Mode Register 17 is accessed automatically and SEG_MASK bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

0: Segment is refreshed (= unmasked).

1: Segment is not refreshed (= masked).

This mode is unique to the low-power DDR2-SDRAM-S4 and low-power DDR3-SDRAM devices. The number of Segment Mask bits differs with the density. For 1 Gbit density, 8 segments are used. In Self-refresh mode, when the Segment Mask bit is configured, the refresh operation is masked in the segment.

- **DS: Drive Strength**

Reset value is 2.

After the initialization sequence, as soon as the DS field is modified, Mode Register 3 is accessed automatically and DS bits are updated. Depending on the UPD_MR bit, update is done before entering Self-refresh mode or during a refresh command and a pending read or write access.

This field is unique to low-power DDR2-SDRAM and low-power DDR3-SDRAM. It selects the I/O drive strength:

Value	Name	Description
0	–	Reserved
1	DS_34_3	34.3 ohm typical
2	DS_40	40 ohm typical (default)
3	DS_48	48 ohm typical
4	DS_60	60 ohm typical
5	–	Reserved
6	DS_80	80 ohm typical
7	DS_120	120 ohm typical
8–15	–	Reserved

In case of low-power DDR2-SDRAM or low-power DDR3-SDRAM, the RDIV field in the MPDDRC_IO_CALIBR register must be set to same value of DS field.

33.7.10 MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register

Name: MPDDRC_LPDDR2_LPDDR3_DDR3_CAL_MR4

Access: Read/Write

31	30	29	28	27	26	25	24
MR4_READ							

23	22	21	20	19	18	17	16
MR4_READ							

15	14	13	12	11	10	9	8
COUNT_CAL							

7	6	5	4	3	2	1	0
COUNT_CAL							

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **COUNT_CAL: LPDDR2 LPDDR3 and DDR3 Calibration Timer Count**

This 16-bit field is loaded into a timer which generates the calibration pulse. Each time the calibration pulse is generated, a ZQCS calibration sequence is initiated.

The ZQCS Calibration command is used to calibrate DRAM Ron values over PVT.

One method for calculating the interval between ZQCS commands gives the temperature ($T_{driftrate}$) and voltage ($V_{driftrate}$) drift rates that the SDRAM is subject to in the application. The interval could be defined by the following formula: ZQCorrection/((T_{Sens} x $T_{driftrate}$) + (V_{Sens} x $V_{driftrate}$))

Where T_{Sens} = max(dRONdTM) and V_{Sens} = max(dRONdVM) define the SDRAM temperature and voltage sensitivities.

For example, if T_{Sens} = 0.75%/C, V_{Sens} = 0.2%/mV, $T_{driftrate}$ = 1C/sec and $V_{driftrate}$ = 15 mV/s, then the interval between ZQCS commands is calculated as:

1.5/((0.75 x 1) + (0.2 x 15)) = 0.4s

In this example, the devices require a calibration every 0.4s. The value to be loaded depends on average time between REFRESH commands, t_{REF}.

For example, for a device with the time between refresh of 7.8 µs, the value of the Calibration Timer Count field is programmed: (0.4/7.8 x 10^{-6}) = 0xC852.

- **MR4_READ: Mode Register 4 Read Interval**

MR4_READ defines the time period between MR4 reads (for LPDDR2-SDRAM). The formula is (MR4_READ+1) × t_{REF}. The value to be loaded depends on the average time between REFRESH commands, t_{REF}. For example, for an LPDDR2-SDRAM with the time between refresh of 7.8 µs, if the MR4_READ value is 2, the time period between MR4 reads is 23.4 µs.

The LPDDR2-SDRAM and LPDDR3-SDRAM devices feature a temperature sensor whose status can be read from MR4 register. This sensor can be used to determine an appropriate refresh rate. Temperature sensor data may be read from MR4 register using the Mode Register Read protocol. The Adjust Refresh Rate bit (ADJ_REF) in the Refresh Timer Register (MPDDRC_RTR) must be written to a one to activate these reads.

33.7.11 MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Timing Calibration Register

Name: MPDDRC_LPDDR2_LPDDR3_DDR3_TIM_CAL

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	RZQI	

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
ZQCS							

- **ZQCS: ZQ Calibration Short**

Reset value is 6 DDRCK(1) clock cycles.

This field defines the delay between the ZQ Calibration command and any valid command in number of DDRCK(1) clock cycles.

The number of cycles is between 0 and 255. This field applies to LPDDR2, LPDDR3 and DDR3 devices.

- **RZQI: Built-in Self-Test for RZQ Information (read-only)**

Reset value is 0.

This field indicates whether the device has detected a resistor connection to the ZQ pin.

This mode is unique to low-power DDR3-SDRAM devices.

Value	Name	Description
0	RZQ_NOT_SUPPORTED	RZQ self test not supported
1	ZQ_VDDCA_FLOAT	The ZQ pin can be connected to VDDCA or left floating.
2	ZQ_SHORTED_GROUND	The ZQ pin can be shorted to ground.
3	ZQ_SELF_TEST_OK	ZQ pin self test complete; no error condition detected

Note: 1. DDRCK is the clock that drives the SDRAM device.

33.7.12 MPDDRC I/O Calibration Register

Name: MPDDRC_IO_CALIBR

Address: 0xF000C034

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CALCODEN				CALCODEP			

15	14	13	12	11	10	9	8
–	TZQIO						

7	6	5	4	3	2	1	0
–	–	–	EN_CALIB	–	RDIV		

- **RDIV: Resistor Divider, Output Driver Impedance**

Reset value is 0.

With the LPDDR2-SDRAM device, the RDIV field must be equal to the DS (Drive Strength) field of the MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register.

RDIV is used with the external precision resistor RZQ to define the output driver impedance. The value of RZQ is either 24K ohms (LPDDR2/LPDDR3 device) or 23K ohms (DDR3L device) or 22K ohms (DDR3 device) or 21K ohms (DDR2/LPDDR1 device).

Value	Name	Description
1	RZQ_34	LPDDR2 serial impedance line = 34.3 ohms, DDR2/LPDDR1 serial impedance line: Not applicable
2	RZQ_40_RZQ_38_RZQ_37_RZQ_35	LPDDR2 serial impedance line = 40 ohms, LPDDR3 serial impedance line = 38 ohms, DDR3 serial impedance line = 37 ohms, DDR2/LPDDR1 serial impedance line = 35 ohms
3	RZQ_48_RZQ_46_RZQ_44_RZQ_43	LPDDR2 serial impedance line = 48 ohms, LPDDR3 serial impedance line = 46 ohms, DDR3 serial impedance line = 44 ohms, DDR2/LPDDR1 serial impedance line = 43 ohms
4	RZQ_60_RZQ_57_RZQ_55_RZQ_52	LPDDR2 serial impedance line = 60 ohms, LPDDR3 serial impedance line = 57 ohms, DDR3 serial impedance line = 55 ohms, DDR2/LPDDR1 serial impedance line = 52 ohms
6	RZQ_80_RZQ_77_RZQ_73_RZQ_70	LPDDR2 serial impedance line = 80 ohms, LPDDR3 serial impedance line = 77 ohms, DDR3 serial impedance line = 73 ohms, DDR2/LPDDR1 serial impedance line = 70 ohms
7	RZQ_120_RZQ_115_RZQ_110_RZQ_105	LPDDR2 serial impedance line = 120 ohms, LPDDR3 serial impedance line = 115 ohms, DDR3 serial impedance line = 110 ohms, DDR2/LPDDR1 serial impedance line = 105 ohms

- **TZQIO: IO Calibration**

This field defines the delay between the start up of the amplifier and the beginning of the calibration in number of DDRCK[1] clock cycles. The value of this field must be set to 600 ns.

The number of cycles is between 0 and 127.

The TZQIO configuration code must be set correctly depending on the clock frequency using the following formula:

TZQIO = (DDRCK × 600e-9) + 1

where DDRCK frequency is in Hz.

For example, for a frequency of 176 MHz, the value of the TZQIO field is configured (176 ×10e6) x (20 × 600e-9) + 1.

- **EN_CALIB: Enable Calibration**

Reset value is 0.

This field enables calibration for the LPDDR1 and DDR2 devices. When the calibration is enabled, it is recommended to define the COUNT_CAL field (see “COUNT_CAL: LPDDR2 LPDDR3 and DDR3 Calibration Timer Count”).

This 16-bit field is loaded into a timer which generates the calibration pulse. Each time the calibration pulse is generated, a calibration sequence is initiated.

Value	Name	Description
0	DISABLE_CALIBRATION	Calibration is disabled.
1	ENABLE_CALIBRATION	Calibration is enabled.

- **CALCODEP: Number of Transistor P (read-only)**

Reset value is 7.

This value gives the number of transistor P to perform the calibration.

- **CALCODEN: Number of Transistor N (read-only)**

Reset value is 8.

This value gives the number of transistor N to perform the calibration.

Note: 1. DDRCK is the clock that drives the SDRAM device.

33.7.13 MPDDRC OCMS Register

Name: MPDDRC_OCMS

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SCR_EN

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **SCR_EN: Scrambling Enable**

0: Disables “Off-chip” scrambling for SDRAM access.

1: Enables “Off-chip” scrambling for SDRAM access.

33.7.14 MPDDRC OCMS KEY1 Register

Name: MPDDRC_OCMS_KEY1

Access: Write once

31	30	29	28	27	26	25	24
KEY1							

23	22	21	20	19	18	17	16
KEY1							

15	14	13	12	11	10	9	8
KEY1							

7	6	5	4	3	2	1	0
KEY1							

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **KEY1: Off-chip Memory Scrambling (OCMS) Key Part 1**

When Off-chip Memory Scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

33.7.15 MPDDRC OCMS KEY2 Register

Name: MPDDRC_OCMS_KEY2

Access: Write once

31	30	29	28	27	26	25	24
KEY2							
23	22	21	20	19	18	17	16
KEY2							
15	14	13	12	11	10	9	8
KEY2							
7	6	5	4	3	2	1	0
KEY2							

This register can only be written if the WPEN bit is cleared in the MPDDRC Write Protection Mode Register.

- **KEY2: Off-chip Memory Scrambling (OCMS) Key Part 2**

When Off-chip Memory Scrambling is enabled, the data scrambling depends on KEY1 and KEY2 values.

33.7.16 MPDDRC Configuration Arbiter Register

Name: MPDDRC_CONF_ARBITER

Access: Read/Write

31	30	29	28	27	26	25	24
BDW_BURST_P7	BDW_BURST_P6	BDW_BURST_P5	BDW_BURST_P4	BDW_BURST_P3	BDW_BURST_P2	BDW_BURST_P1	BDW_BURST_P0
23	22	21	20	19	18	17	16
MA_PR_P7	MA_PR_P6	MA_PR_P5	MA_PR_P4	MA_PR_P3	MA_PR_P2	MA_PR_P1	MA_PR_P0
15	14	13	12	11	10	9	8
RQ_WD_P7	RQ_WD_P6	RQ_WD_P5	RQ_WD_P4	RQ_WD_P3	RQ_WD_P2	RQ_WD_P1	RQ_WD_P0
7	6	5	4	3	2	1	0
–	–	–	–	BDW_MAX_CUR	–	ARB	

- **ARB: Type of Arbitration**

Reset value is 0.

This field allows to choose the type of arbitration: round-robin, number of requests per port or bandwidth per port.

Value	Name	Description
0	ROUND	Round Robin
1	NB_REQUEST	Request Policy
2	BANDWIDTH	Bandwidth Policy
3	–	Reserved

- **RQ_WD_Px: Request or Word from Port X**

Reset value is 0.

0: Number of requests is selected.

1: Number of words is selected.

- **BDW_BURST_Px: Bandwidth is Reached or Bandwidth and Current Burst Access is Ended on port X**

Reset value is 0.

0: The arbitration is done when bandwidth is reached and burst access is ended.

1: The arbitration is done when bandwidth is reached.

- **BDW_MAX_CUR: Bandwidth Max or Current**

This field displays the maximum of the bandwidth or the current bandwidth for each port.

The maximum of the bandwidth is computed when at least two ports of MPDDRC are used.

That information is given in Section 33.7.20 “MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register” and Section 33.7.21 “MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register”.

Reset value is 0.

0: Current bandwidth is displayed.

1: Maximum of the bandwidth is displayed.

- **MA_PR_Px: Master or Software Provide Information**

Reset value is 0.

0: Number of requests or words is provided by the master, if the master supports this feature.

1: Number of requests or words is provided by software, see “NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth Allocation from Port 0-1-2-3” .

33.7.17 MPDDRC Timeout Register

Name: MPDDRC_TIMEOUT

Access: Read/Write

31	30	29	28	27	26	25	24
TIMEOUT_P7				TIMEOUT_P6			
23	22	21	20	19	18	17	16
TIMEOUT_P5				TIMEOUT_P4			
15	14	13	12	11	10	9	8
TIMEOUT_P3				TIMEOUT_P2			
7	6	5	4	3	2	1	0
TIMEOUT_P1				TIMEOUT_P0			

- **TIMEOUT_Px: Timeout for Ports 0, 1, 2, 3, 4, 5, 6 and 7**

Reset value is 0.

Some masters have the particularity to insert idle state between two accesses. This field defines the delay between two accesses on the same port in number of DDRCK(1) clock cycles before arbitration and handling the access over to another port.

This field is not used with round-robin and bandwidth arbitrations.

The number of cycles is between 1 and 15.

Note: 1. DDRCKDDRCK is the clock that drives the SDRAM device.

33.7.18 MPDDRC Request Port 0-1-2-3 Register

Name: MPDDRC_REQ_PORT_0123

Access: Read/Write

31	30	29	28	27	26	25	24
NRQ_NWD_BDW_P3							
23	22	21	20	19	18	17	16
NRQ_NWD_BDW_P2							
15	14	13	12	11	10	9	8
NRQ_NWD_BDW_P1							
7	6	5	4	3	2	1	0
NRQ_NWD_BDW_P0							

- **NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth Allocation from Port 0-1-2-3**

Reset value is 0.

The number of requests corresponds to the number of start transfers. For example, setting this field to 2 performs two burst accesses regardless of the burst type (INCR4, INCR8,...). The number of words corresponds exactly to the number of accesses; setting this field to 2 performs two accesses. In this example, burst accesses will be broken.

These values depend on scheme arbitration (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

In case of round-robin arbitration, this field is not used. In case of “bandwidth arbitration”, this field corresponds to percentage allocated for each port. In case of “request” arbitration, this field corresponds to number of start transfers or to number of accesses allocated for each port.

33.7.19 MPDDRC Request Port 4-5-6-7 Register

Name: MPDDRC_REQ_PORT_4567

Access: Read/Write

31	30	29	28	27	26	25	24
NRQ_NWD_BDW_P7							
23	22	21	20	19	18	17	16
NRQ_NWD_BDW_P6							
15	14	13	12	11	10	9	8
NRQ_NWD_BDW_P5							
7	6	5	4	3	2	1	0
NRQ_NWD_BDW_P4							

- **NRQ_NWD_BDW_Px: Number of Requests, Number of Words or Bandwidth Allocation from Port 4-5-6-7**

Reset value is 0.

The number of requests corresponds to the number of start transfers. For example, setting this field to 2 performs two burst accesses regardless of the burst type (INCR4, INCR8,...). The number of words corresponds exactly to the number of accesses; setting this field to 2 performs two accesses. In this example, burst accesses will be broken.

These values depend on scheme arbitration (see Section 33.7.16 “MPDDRC Configuration Arbiter Register”).

In case of round-robin arbitration, this field is not used. In case of “bandwidth arbitration”, this field corresponds to percentage allocated for each port. In case of “request” arbitration, this field corresponds to number of start transfers or to number of accesses allocated for each port.

33.7.20 MPDDRC Current/Maximum Bandwidth Port 0-1-2-3 Register

Name: MPDDRC_BDW_PORT_0123

Access: Read-only

31	30	29	28	27	26	25	24
–	BDW_P3						

23	22	21	20	19	18	17	16
–	BDW_P2						

15	14	13	12	11	10	9	8
–	BDW_P1						

7	6	5	4	3	2	1	0
–	BDW_P0						

- **BDW_Px: Current/Maximum Bandwidth from Port 0-1-2-3**

Reset value is 0.

This field displays the current bandwidth or the maximum bandwidth for each port. This information is given in the "BDW_MAX_CUR: Bandwidth Max or Current" field description.

33.7.21 MPDDRC Current/Maximum Bandwidth Port 4-5-6-7 Register

Name: MPDDRC_BDW_PORT_4567

Access: Read-only

31	30	29	28	27	26	25	24
–	BDW_P7						

23	22	21	20	19	18	17	16
–	BDW_P6						

15	14	13	12	11	10	9	8
–	BDW_P5						

7	6	5	4	3	2	1	0
–	BDW_P4						

- **BDW_Px: Current/Maximum Bandwidth from Port 4-5-6-7**

Reset value is 0.

This field displays the current bandwidth or the maximum bandwidth for each port. This information is given in the "BDW_MAX_CUR: Bandwidth Max or Current" field description.

33.7.22 MPDDRC Read Data Path Register

Name: MPDDRC_RD_DATA_PATH

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	SHIFT_SAMPLING	

- **SHIFT_SAMPLING: Shift Sampling Point of Data**

Reset value is 0.

This field shifts the sampling point of data that comes from the memory device. This sampling point depends on the external bus frequency. The higher the frequency, the more the sampling point will be shifted.

Value	Name	Description
0	NO_SHIFT	Initial sampling point.
1	SHIFT_ONE_CYCLE	Sampling point is shifted by one cycle.
2	SHIFT_TWO_CYCLES	Sampling point is shifted by two cycles.
3	SHIFT_THREE_CYCLES	Sampling point is shifted by three cycles, unique for LPDDR2 and DDR3 and LPDDR3. Not applicable for DDR2 and LPDDR1 devices.

In the case of DDR3-SDRAM devices, the field SHIFT_SAMPLING must be set to 2, and the field CAS must be set to 5. See “CAS: CAS Latency” .

33.7.23 MPDDRC Monitor Configuration Register

Name: MPDDRC_MCFGR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	INFO		REFR_CALIB	READ_WRITE	
7	6	5	4	3	2	1	0
–	–	–	RUN	–	–	SOFT_RESET	EN_MONI

- **EN_MONI: Enable Monitor**

0: Monitor is disabled.

1: Monitor is enabled.

- **SOFT_RESET: Soft Reset**

0: Soft reset is not performed.

1: Soft reset is performed.

- **RUN: Control Monitor**

0: Monitoring is halted. All counters are stopped.

1: Monitoring is launched.

- **READ_WRITE: Read/Write Access**

This field is used to monitor different types of access.

Value	Name	Description
0	TRIG_RD_WR	Read and Write accesses are triggered.
1	TRIG_WR	Only Write accesses are triggered.
2	TRIG_RD	Only Read accesses are triggered.
3	–	Reserved

- **REFR_CALIB: Refresh Calibration**

0: Monitoring depends on the refresh and calibration impact.

1: Monitoring depends on the refresh and calibration impact.

- **INFO: Information Type**

This field reports information such as latency and the number of transfers monitored on port x [x = 0..7].

Value	Name	Description
0	MAX_WAIT	Information concerning the transfer with the longest waiting time
1	NB_TRANSFERS	Number of transfers on the port
2	TOTAL_LATENCY	Total latency on the port
3	–	Reserved

33.7.24 MPDDRC Monitor Address High/Low Port x Register

Name: MPDDRC_MADDRx [x = 0..7]

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR_HIGH_PORTx							

23	22	21	20	19	18	17	16
ADDR_HIGH_PORTx							

15	14	13	12	11	10	9	8
ADDR_LOW_PORTx							

7	6	5	4	3	2	1	0
ADDR_LOW_PORTx							

- **ADDR_LOW_PORTx: Address Low on Port x [x = 0..7]**

Address low which defines the interval to be monitored on port x [x = 0..7]. This address must be programmed according to the memory mapping of the product.

- **ADDR_HIGH_PORTx: Address High on Port x [x = 0..7]**

Address high which defines the interval to be monitored on port x [x = 0..7]. This address must be programmed according to the memory mapping of the product.

33.7.25 MPDDRC Monitor Information Port x Register (MAX_WAIT)

Name: MPDDRC_MINFOx [x = 0..7] (MAX_WAIT)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	READ_WRITE
23	22	21	20	19	18	17	16
–	SIZE			–	BURST		
15	14	13	12	11	10	9	8
MAX_PORTx_WAITING							
7	6	5	4	3	2	1	0
MAX_PORTx_WAITING							

The following fields can be read if the INFO field in the MPDDRC Monitor Configuration register is set to 0.

- **MAX_PORTx_WAITING: Address High on Port x [x = 0..7]**

This field reports the maximum waiting time and the associated type of transfer (burst, size, read or write).

- **BURST: Type of Burst on Port x [x = 0..7]**

This field reports the type of burst for the maximum waiting time.

Value	Name	Description
0	SINGLE	Single transfer
1	INCR	Incrementing burst of unspecified length
2	WRAP4	4-beat wrapping burst
3	INCR4	4-beat incrementing burst
4	WRAP8	8-beat wrapping burst
5	INCR8	8-beat incrementing burst
6	WRAP16	16-beat wrapping burst
7	INCR16	16-beat incrementing burst

- **SIZE: Transfer Size on Port x [x = 0..7]**

This field reports the size of the transfer for the maximum waiting time.

Value	Name	Description
0	8BITS	Byte transfer
1	16BITS	Halfword transfer
2	32BITS	Word transfer
3	64BITS	Dword transfer
4–7	–	Reserved

- **READ_WRITE: Read or Write Access on Port x [x = 0..7]**

This field reports the transfer direction for the maximum waiting time.

0: Read transfer.

1: Write transfer.

33.7.26 MPDDRC Monitor Information Port x Register (NB_TRANSFERS)

Name: MPDDRC_MINFOx [x = 0..7] (NB_TRANSFERS)

Access: Read-only

31	30	29	28	27	26	25	24
Px_NB_TRANSFERS							

23	22	21	20	19	18	17	16
Px_NB_TRANSFERS							

15	14	13	12	11	10	9	8
Px_NB_TRANSFERS							

7	6	5	4	3	2	1	0
Px_NB_TRANSFERS							

- **Px_NB_TRANSFERS: Number of Transfers on Port x [x = 0..7]**

This field can be read if the INFO field is set to 1. This field reports the number of transfers performed within an interval (ADDR_HIGH_PORT and ADDR_LOW_PORT) when the port is used.

33.7.27 MPDDRC Monitor Information Port x Register (TOTAL_LATENCY)

Name: MPDDRC_MINFOx [x = 0..7] (TOTAL_LATENCY)

Address: 0xF000C084 [0] .. 0xF000C0A0 [7]

Access: Read-only

31	30	29	28	27	26	25	24
Px_TOTAL_LATENCY							

23	22	21	20	19	18	17	16
Px_TOTAL_LATENCY							

15	14	13	12	11	10	9	8
Px_TOTAL_LATENCY							

7	6	5	4	3	2	1	0
Px_TOTAL_LATENCY							

- **Px_TOTAL_LATENCY: Total Latency on Port x [x = 0..7]**

This field can be read if the INFO field is set to 2. This field reports the total latency within an interval (ADDR_HIGH_PORT and ADDR_LOW_PORT) when the port is used.

33.7.28 MPDDRC Write Protection Mode Register

Name: MPDDRC_WPMR

Address: 0xF000C0E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x444452 (“DDR” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x444452 (“DDR” in ASCII).

See Section 33.7 “AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x444452	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

33.7.29 MPDDRC Write Protection Status Register

Name: MPDDRC_WPSR

Address: 0xF000C0E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Enable**

0: No write protection violation occurred since the last read of this register (MPDDRC_WPSR).

1: A write protection violation occurred since the last read of this register (MPDDRC_WPSR). If this violation is an unauthorized attempt to write a control register, the associated violation is reported into the WPVSRC field.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

34. Static Memory Controller (SMC)

34.1 Description

This Static Memory Controller (SMC) is capable of handling several types of external memory and peripheral devices, such as SRAM, PSRAM, PROM, EPROM, EEPROM, LCD Module, NOR Flash and NAND Flash.

The SMC generates the signals that control the access to external memory devices or peripheral devices. It has 4 Chip Selects and a 26-bit address bus. The 16-bit data bus can be configured to interface with 8- or 16-bit external devices. Separate read and write control signals allow for direct memory and peripheral interfacing. Read and write signal waveforms are fully parametrizable.

The SMC can manage wait requests from external devices to extend the current access. The SMC is provided with an automatic Slow Clock mode. In Slow Clock mode, it switches from user-programmed waveforms to slow-rate specific waveforms on read and write signals.

The SMC embeds a NAND Flash Controller (NFC). The NFC can handle automatic transfers, sending the commands and address cycles to the NAND Flash and transferring the contents of the page (for read and write) to the NFC SRAM. It minimizes the CPU overhead.

The SMC includes programmable hardware error correcting code with one-bit error correction capability and supports two-bit error detection. In order to improve the overall system performance, the DATA phase of the transfer can be DMA-assisted.

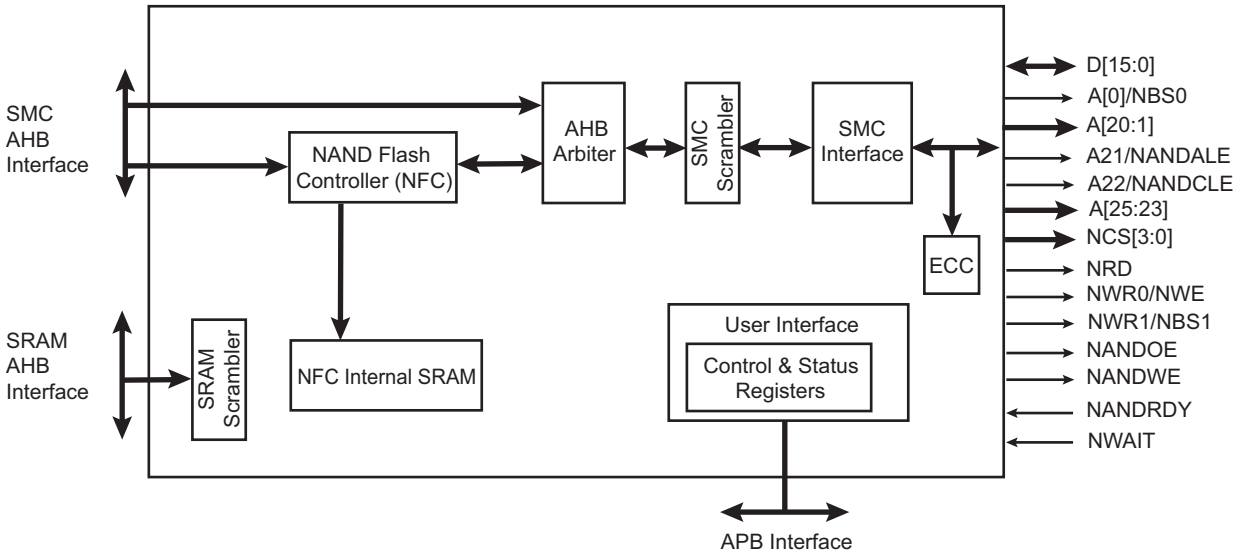
The External Data Bus can be scrambled/unscrambled by means of user keys.

34.2 Embedded Characteristics

- 64-Mbyte Address Space per Chip Select
- 8- or 16-bit Data Bus
- Word, Halfword, Byte Transfers
- Byte Write or Byte Select Lines
- Programmable Setup, Pulse and Hold Time for Read Signals per Chip Select
- Programmable Setup, Pulse and Hold Time for Write Signals per Chip Select
- Programmable Data Float Time per Chip Select
- External Data Bus Scrambling/Unscrambling Function
- External Wait Request
- Automatic Switch to Slow Clock Mode
- Hardware Configurable Number of Chip Selects from 1 to 4
- Programmable Timing on a per Chip Select Basis
- NAND Flash Controller Supporting NAND Flash with Multiplexed Data/Address Buses
- Supports SLC and MLC NAND Flash Technology
- Supports NAND Flash Devices with 8 or 16-bit Data Paths
- Multibit Error Correcting Code (ECC) supporting NAND Flash devices with 8-bit only Data Path
- ECC Algorithm Based on Binary Shortened Bose, Chaudhuri and Hocquenghem (BCH) Codes
- Programmable Error Correcting Capability: 2, 4, 8, 12, 24 and 32 bits of Errors per Block
- 9 Kbytes NFC SRAM
- Programmable Block Size: 512 bytes or 1024 bytes
- Programmable Number of Block per Page: 1, 2, 4 or 8 Blocks of Data per Page
- Programmable Spare Area Size up to 512 bytes
- Supports Spare Area ECC Protection
- Supports 8 Kbytes Page Size Using 1024 bytes/block and 4 Kbytes Page Size Using 512 bytes/block
- Multibit Error Detection Is Interrupt Driven
- Provides Hardware Acceleration for Determining Roots of Polynomials Defined over a Finite Field
- Programmable Finite Field GF(2^13) or GF(2^14)
- Finds Roots of Error-locator Polynomial
- Programmable Number of Roots
- Register Write Protection

34.3 Block Diagram

Figure 34-1. Block Diagram



34.4 I/O Lines Description

Table 34-1. I/O Line Description

Name	Description	Type	Active Level
NCS[3:0]	Static Memory Controller Chip Select Lines	Output	Low
NRD	Read Signal	Output	Low
NWR0/NWE	Write 0/Write Enable Signal	Output	Low
A0/NBS0	Address Bit 0/Byte 0 Select Signal	Output	Low
NWR1/NBS1	Write 1/Byte 1 Select Signal	Output	Low
A[25:1]	Address Bus	Output	–
D[15:0]	Data Bus	I/O	–
NWAIT	External Wait Signal	Input	Low
NANDRDY	NAND Flash Ready/Busy	Input	–
NANDWE	NAND Flash Write Enable	Output	Low
NANDOE	NAND Flash Output Enable	Output	Low
NANDALE	NAND Flash Address Latch Enable	Output	–
NANDCLE	NAND Flash Command Latch Enable	Output	–

34.5 Multiplexed Signals

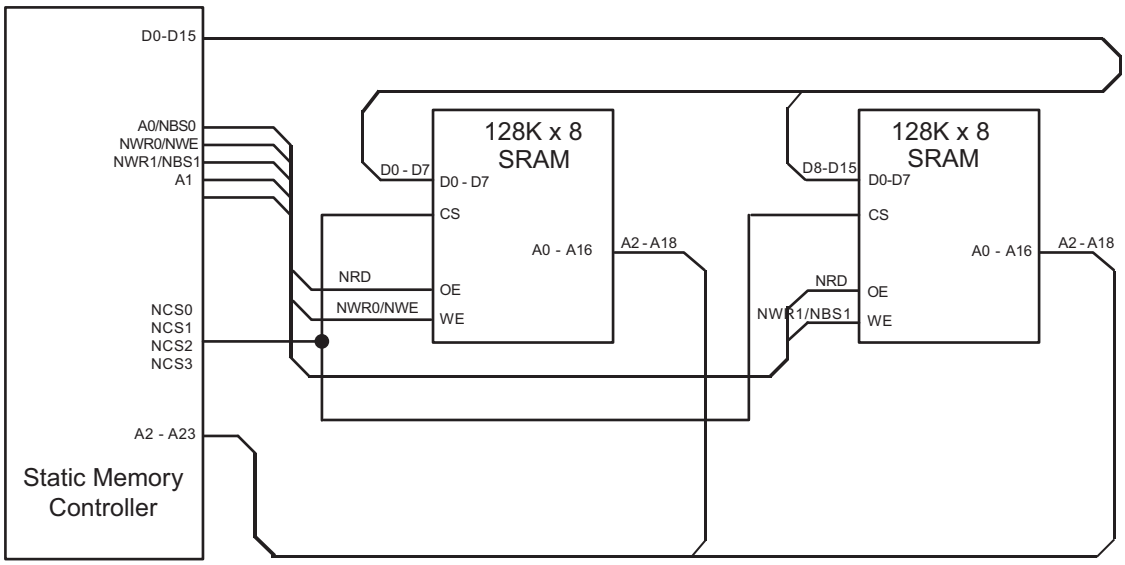
Table 34-2. Static Memory Controller (SMC) Multiplexed Signals

Multiplexed Signals		Related Function
NWR0	NWE	Byte-write or Byte-select access, see Section 34.9.2.1 “Byte Write Access” and Section 34.9.2.2 “Byte Select Access”
A0	NBS0	8-bit or 16-bit data bus, see Section 34.9.1 “Data Bus Width”
A22	NANDCLE	NAND Flash Command Latch Enable
A21	NANDALE	NAND Flash Address Latch Enable
NWR1	NBS1	Byte-write or Byte-select access, see Section 34.9.2.1 “Byte Write Access” and Section 34.9.2.2 “Byte Select Access”
A1	–	8-/16-bit data bus, see Section 34.9.1 “Data Bus Width” Byte-write or Byte-select access, see Section 34.9.2.1 “Byte Write Access” and Section 34.9.2.2 “Byte Select Access”

34.6 Application Example

34.6.1 Hardware Interface

Figure 34-2. SMC Connections to Static Memory Devices



34.7 Product Dependencies

34.7.1 I/O Lines

The pins used for interfacing the Static Memory Controller are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the Static Memory Controller pins to their peripheral function. If I/O lines of the SMC are not used by the application, they can be used for other purposes by the PIO controller.

34.7.2 Power Management

The SMC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SMC clock.

34.7.3 Interrupt Sources

The SMC has an interrupt line connected to the interrupt controller. Handling the SMC interrupt requires programming the interrupt controller before configuring the SMC.

Table 34-3. Peripheral IDs

Instance	ID
HSMC	17

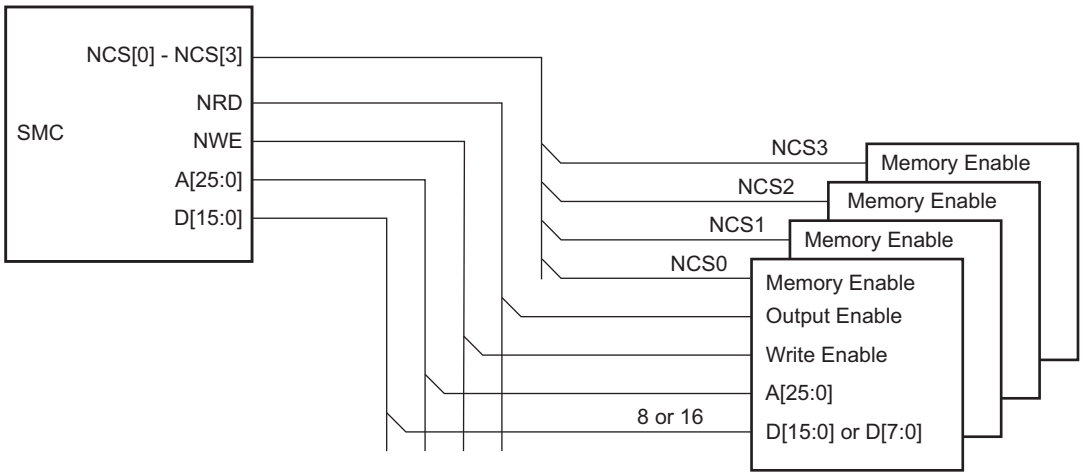
34.8 External Memory Mapping

The SMC provides up to 26 address lines, A[25:0]. This allows each chip select line to address up to 64 Mbytes of memory.

If the physical memory device connected on one chip select is smaller than 64 Mbytes, it wraps around and appears to be repeated within this space. The SMC correctly handles any valid access to the memory device within the page (see Figure 34-3).

A[25:0] is only significant for 8-bit memory; A[25:1] is used for 16-bit memory.

Figure 34-3. Memory Connections for External Devices



34.9 Connection to External Devices

34.9.1 Data Bus Width

A data bus width of 8 or 16 bits can be selected for each chip select. This option is controlled by the bit DBW in the SMC Mode Register (HSMC_MODE) for the corresponding chip select.

Figure 34-4 shows how to connect a 512 KB x 8-bit memory on NCS2. Figure 34-5 shows how to connect a 512 KB x 16-bit memory on NCS2.

Figure 34-4. Memory Connection for an 8-bit Data Bus

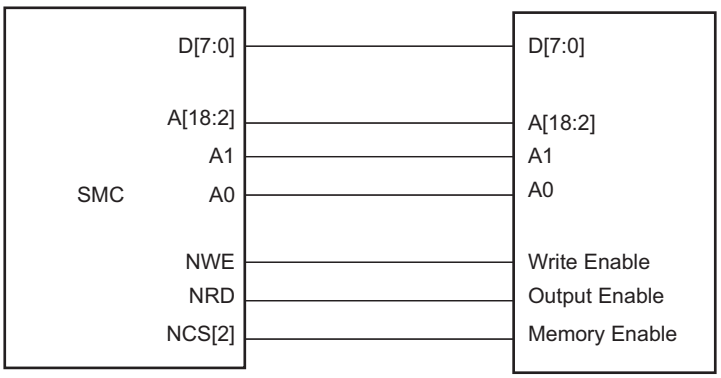
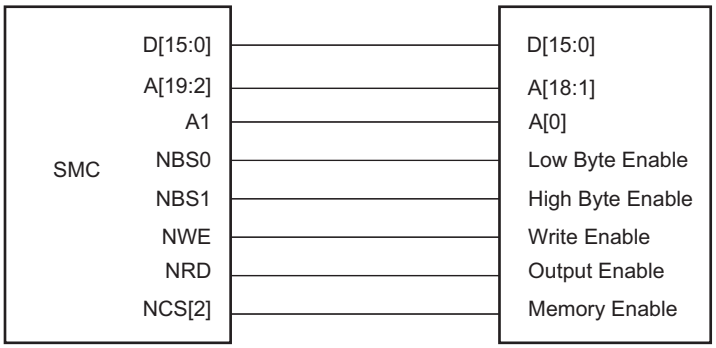


Figure 34-5. Memory Connection for a 16-bit Data Bus



34.9.2 Byte Write or Byte Select Access

Each chip select with a 16-bit data bus can operate with one of two different types of write access: Byte Write or Byte Select. This is controlled by the BAT bit of the HSMC_MODE register for the corresponding chip select.

34.9.2.1 Byte Write Access

Byte Write Access is used to connect 2 x 8-bit devices as a 16-bit memory, and supports one write signal per byte of the data bus and a single read signal.

Note that the SMC does not allow boot in Byte Write Access mode.

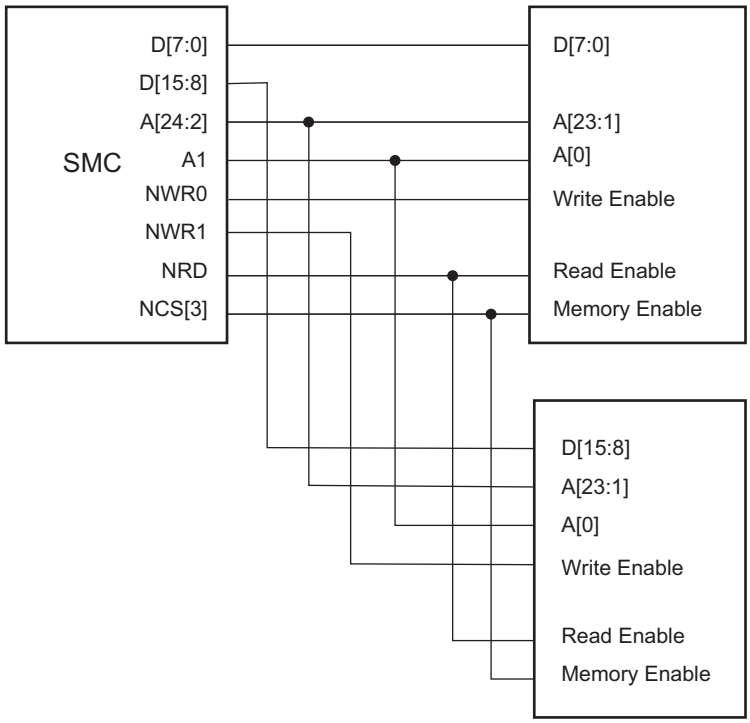
For 16-bit devices, the SMC provides NWR0 and NWR1 write signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus. One single read signal (NRD) is provided.

34.9.2.2 Byte Select Access

Byte Select Access is used to connect one 16-bit device. In this mode, read/write operations can be enabled/disabled at Byte level. One Byte-select line per byte of the data bus is provided. One NRD and one NWE signal control read and write.

For 16-bit devices, the SMC provides NBS0 and NBS1 selection signals for respectively Byte0 (lower byte) and Byte1 (upper byte) of a 16-bit bus.

Figure 34-6. Connection of 2 x 8-bit Devices on a 16-bit Bus: Byte Write Option



34.9.2.3 Signal Multiplexing

Depending on the Byte Access Type (BAT), only the write signals or the byte select signals are used. To save IOs at the external bus interface, control signals at the SMC interface are multiplexed. Table 34-4 shows signal multiplexing depending on the data bus width and the Byte Access Type.

For 16-bit devices, bit A0 of address is unused. When Byte Select Option is selected, NWR1 is unused. When Byte Write option is selected, NBS0 is unused.

Table 34-4. SMC Multiplexed Signal Translation

Device Type	Signal Name		
	16-bit Bus		8-bit Bus
	1 x 16-bit	2 x 8-bit	1 x 8-bit
Byte Access Type (BAT)	Byte Select	Byte Write	–
NBS0_A0	NBS0	–	A0
NWE_NWR0	NWE	NWR0	NWE
NBS1_NWR1	NBS1	NWR1	–
A1	A1	A1	A1

34.10 Standard Read and Write Protocols

In the following sections, the Byte Access Type is not considered. Byte select lines (NBS0 to NBS1) always have the same timing as the A address bus. NWE represents either the NWE signal in byte select access type or one of the byte write lines (NWR0 to NWR1) in byte write access type. NWR0 to NWR1 have the same timings and protocol as NWE. In the same way, NCS represents one of the NCS[0..3] chip select lines.

34.10.1 Read Waveforms

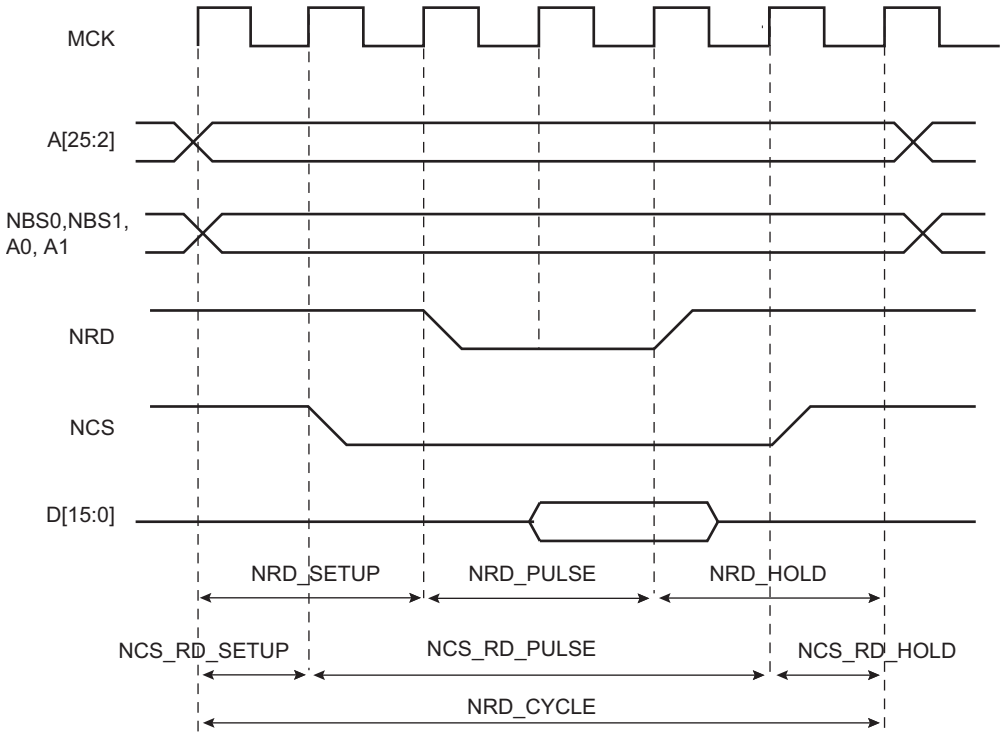
The read cycle is shown on Figure 34-7.

The read cycle starts with the address setting on the memory address bus, i.e.,:

{A[25:2], A1, A0} for 8-bit devices

{A[25:2], A1} for 16-bit devices

Figure 34-7. Standard Read Cycle



34.10.1.1 NRD Waveform

The NRD signal is characterized by a setup timing, a pulse width and a hold timing:

1. NRD_SETUP: The NRD setup time is defined as the setup of address before the NRD falling edge.
2. NRD_PULSE: The NRD pulse length is the time between NRD falling edge and NRD rising edge.
3. NRD_HOLD: The NRD hold time is defined as the hold time of address after the NRD rising edge.

34.10.1.2 NCS Waveform

Similar to the NRD signal, the NCS signal can be divided into a setup time, pulse length and hold time:

- NCS_RD_SETUP: The NCS setup time is defined as the setup time of address before the NCS falling edge.
- NCS_RD_PULSE: The NCS pulse length is the time between NCS falling edge and NCS rising edge.
- NCS_RD_HOLD: The NCS hold time is defined as the hold time of address after the NCS rising edge.

34.10.1.3 Read Cycle

The NRD_CYCLE time is defined as the total duration of the read cycle, that is, from the time where address is set on the address bus to the point where address may change. The total read cycle time is defined as:

NRD_CYCLE = NRD_SETUP + NRD_PULSE + NRD_HOLD,

as well as

NRD_CYCLE = NCS_RD_SETUP + NCS_RD_PULSE + NCS_RD_HOLD

All NRD and NCS timings are defined separately for each chip select as an integer number of Master Clock cycles. The NRD_CYCLE field is common to both the NRD and NCS signals, thus the timing period is of the same duration.

NRD_CYCLE, NRD_SETUP, and NRD_PULSE implicitly define the NRD_HOLD value as:

NRD_HOLD = NRD_CYCLE - NRD SETUP - NRD PULSE

NRD_CYCLE, NCS_RD_SETUP, and NCS_RD_PULSE implicitly define the NCS_RD_HOLD value as:

NCS_RD_HOLD = NRD_CYCLE - NCS_RD_SETUP - NCS_RD_PULSE

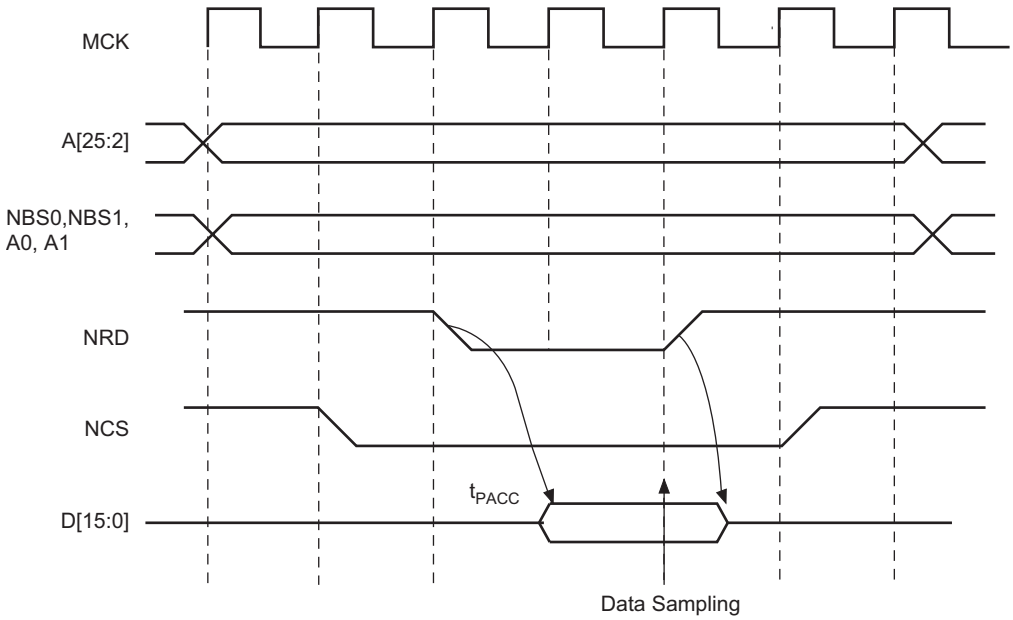
34.10.2 Read Mode

As NCS and NRD waveforms are defined independently of one another, the SMC needs to know when the read data is available on the data bus. The SMC does not compare NCS and NRD timings to know which signal rises first. The READ_MODE parameter in the HSMC_MODE register of the corresponding chip select indicates which signal of NRD and NCS controls the read operation.

34.10.2.1 Read is Controlled by NRD (READ_MODE = 1)

Figure 34-8 shows the waveforms of a read operation of a typical asynchronous RAM. The read data is available t_{PACC} after the falling edge of NRD, and turns to 'Z' after the rising edge of NRD. In this case, the READ_MODE must be set to 1 (read is controlled by NRD), to indicate that data is available with the rising edge of NRD. The SMC samples the read data internally on the rising edge of the Master Clock that generates the rising edge of NRD, whatever the programmed waveform of NCS.

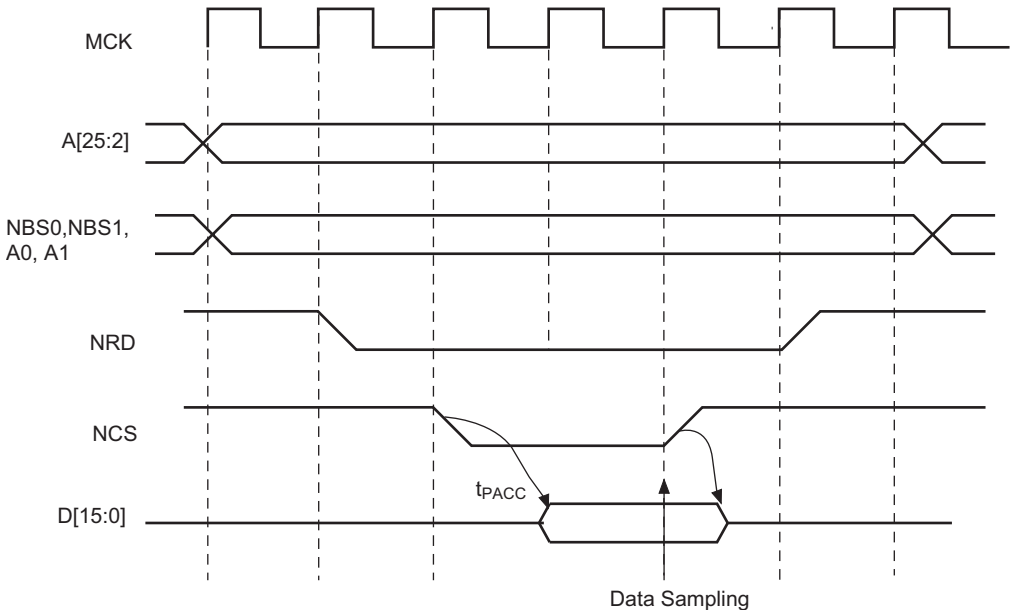
Figure 34-8. READ_MODE = 1: Data is Sampled by SMC before the Rising Edge of NRD



34.10.2.2 Read is Controlled by NCS (READ_MODE = 0)

Figure 34-9 shows the typical read cycle. The read data is valid t_{PACC} after the falling edge of the NCS signal and remains valid until the rising edge of NCS. Data must be sampled when NCS is raised. In that case, the READ_MODE must be configured to 0 (read is controlled by NCS): the SMC internally samples the data on the rising edge of the Master Clock that generates the rising edge of NCS, whatever the programmed waveform of NRD.

Figure 34-9. READ_MODE = 0: Data is Sampled by SMC before the Rising Edge of NCS



34.10.3 Write Waveforms

The write protocol is similar to the read protocol. It is depicted in Figure 34-10. The write cycle starts with the address setting on the memory address bus.

34.10.3.1 NWE Waveforms

The NWE signal is characterized by a setup timing, a pulse width and a hold timing:

- NWE_SETUP: The NWE setup time is defined as the setup of address and data before the NWE falling edge.
- NWE_PULSE: The NWE pulse length is the time between NWE falling edge and NWE rising edge.
- NWE_HOLD: The NWE hold time is defined as the hold time of address and data after the NWE rising edge.

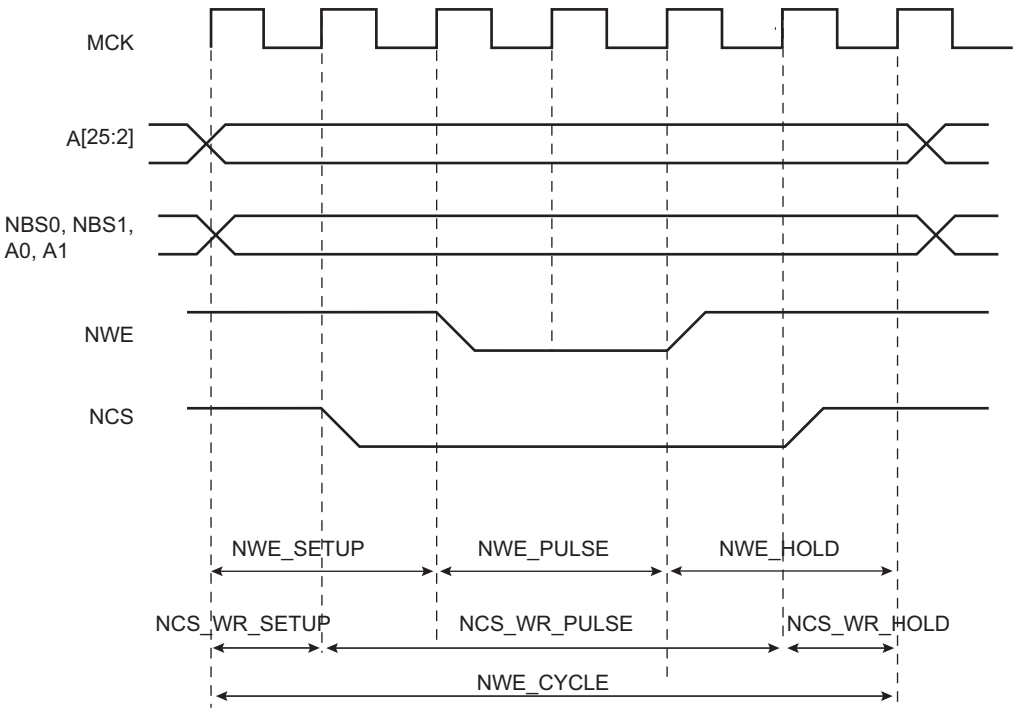
The NWE waveforms apply to all byte-write lines in Byte Write Access mode: NWR0 to NWR3.

34.10.3.2 NCS Waveforms

The NCS signal waveforms in write operations are not the same as those applied in read operations, but are separately defined:

- NCS_WR_SETUP: The NCS setup time is defined as the setup time of address before the NCS falling edge.
- NCS_WR_PULSE: The NCS pulse length is the time between NCS falling edge and NCS rising edge.
- NCS_WR_HOLD: The NCS hold time is defined as the hold time of address after the NCS rising edge.

Figure 34-10. Write Cycle



34.10.3.3 Write Cycle

The write cycle time is defined as the total duration of the write cycle, that is, from the time where address is set on the address bus to the point where address may change. The total write cycle time is equal to:

NWE_CYCLE = NWE_SETUP + NWE_PULSE + NWE_HOLD,

as well as

NWE_CYCLE = NCS_WR_SETUP + NCS_WR_PULSE + NCS_WR_HOLD

All NWE and NCS (write) timings are defined separately for each chip select as an integer number of Master Clock cycles. The NWE_CYCLE field is common to both the NWE and NCS signals, thus the timing period is of the same duration.

NWE_CYCLE, NWE_SETUP, and NWE_PULSE implicitly define the NWE_HOLD value as:

NWE_HOLD = NWE_CYCLE - NWE_SETUP - NWE_PULSE

NWE_CYCLE, NCS_WR_SETUP, and NCS_WR_PULSE implicitly define the NCS_WR_HOLD value as:

NCS_WR_HOLD = NWE_CYCLE - NCS_WR_SETUP - NCS_WR_PULSE

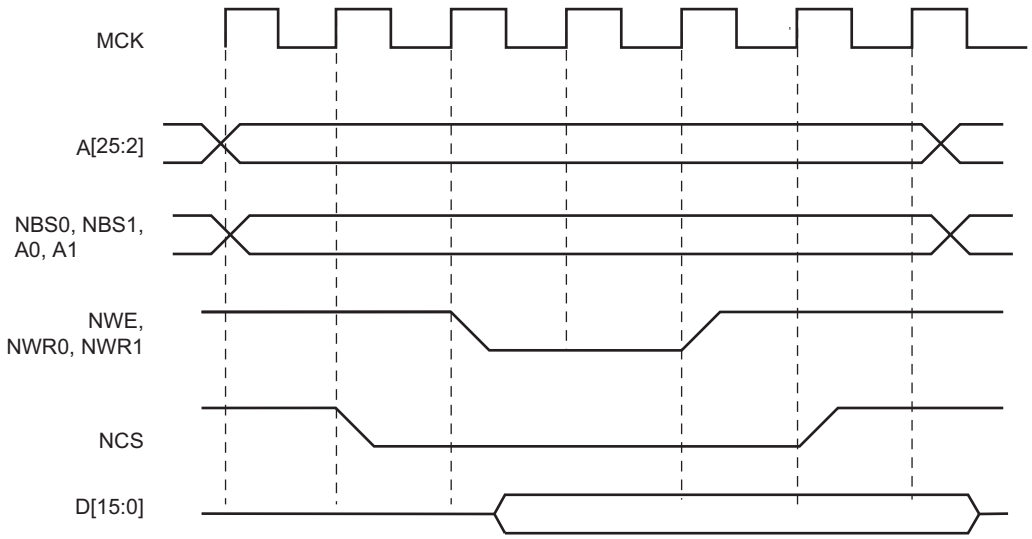
34.10.4 Write Mode

The WRITE_MODE parameter in the HSMC_MODE register of the corresponding chip select indicates which signal controls the write operation.

34.10.4.1 Write is Controlled by NWE (WRITE_MODE = 1)

Figure 34-11 shows the waveforms of a write operation with WRITE_MODE set to 1. The data is put on the bus during the pulse and hold steps of the NWE signal. The internal data buffers are switched to Output mode after the NWE_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NCS.

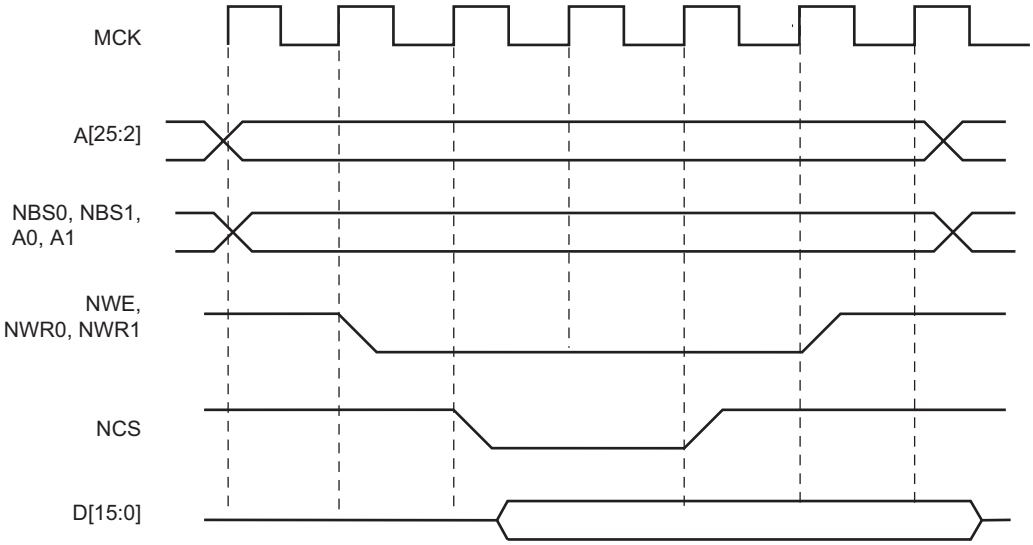
Figure 34-11. WRITE_MODE = 1. The write operation is controlled by NWE



34.10.4.2 Write is Controlled by NCS (WRITE_MODE = 0)

Figure 34-12 shows the waveforms of a write operation with WRITE_MODE configured to 0. The data is put on the bus during the pulse and hold steps of the NCS signal. The internal data buffers are switched to Output mode after the NCS_WR_SETUP time, and until the end of the write cycle, regardless of the programmed waveform on NWE.

Figure 34-12. WRITE_MODE = 0. The write operation is controlled by NCS



34.10.5 Coding Timing Parameters

All timing parameters are defined for one chip select and are grouped together in one register according to their type:

- The HSMC_SETUP register groups the definition of all setup parameters:
  NRD_SETUP, NCS_RD_SETUP, NWE_SETUP, NCS_WR_SETUP
- The HSMC_PULSE register groups the definition of all pulse parameters:
  NRD_PULSE, NCS_RD_PULSE, NWE_PULSE, NCS_WR_PULSE
- The HSMC_CYCLE register groups the definition of all cycle parameters:
  NRD_CYCLE, NWE_CYCLE

Table 34-5 shows how the timing parameters are coded and their permitted range.

Table 34-5. Coding and Range of Timing Parameters

Coded Value	Number of Bits	Effective Value	Permitted Range	
			Coded Value	Effective Value
setup [5:0]	6	128 x setup[5] + setup[4:0]	0 ≤ setup ≤ 31	0..31
			32 ≤ setup ≤ 63	128..(128 + 31)
pulse [6:0]	7	256 x pulse[6] + pulse[5:0]	0 ≤ pulse ≤ 63	0..63
			64 ≤ pulse ≤ 127	256..(256 + 63)
cycle[8:0]	9	256 x cycle[8:7] + cycle[6:0]	0 ≤ cycle ≤ 127	0..127
			128 ≤ cycle ≤ 255	256..(256 + 127)
			256 ≤ cycle ≤ 383	512..(512 + 127)
			384 ≤ cycle ≤ 511	768..(768 + 127)

34.10.6 Reset Values of Timing Parameters

Table 34-6 gives the default value of timing parameters at reset.

Table 34-6. Reset Values of Timing Parameters

Register	Reset Value	Description
HSMC_SETUP	0x0101_0101	All setup timings are set to 1
HSMC_PULSE	0x0101_0101	All pulse timings are set to 1
HSMC_CYCLE	0x0003_0003	The read and write operations last three Master Clock cycles and provide one hold cycle.
WRITE_MODE	1	Write is controlled with NWE
READ_MODE	1	Read is controlled with NRD

34.10.7 Usage Restriction

The SMC does not check the validity of the user-programmed parameters. If the sum of SETUP and PULSE parameters is larger than the corresponding CYCLE parameter, this leads to an unpredictable behavior of the SMC.

34.10.7.1 For Read Operations

Null but positive setup and hold of address and NRD and/or NCS cannot be guaranteed at the memory interface because of the propagation delay of these signals through external logic and pads. When positive setup and hold values must be verified, then it is strictly recommended to program non-null values so as to cover possible skews between address, NCS and NRD signals.

34.10.7.2 For Write Operations

If a null hold value is programmed on NWE, the SMC can guarantee a positive hold of address, byte select lines, and NCS signal after the rising edge of NWE. This is true for WRITE_MODE = 1 only. See Section 34.12.2 “Early Read Wait State”.

34.10.7.3 For Read and Write Operations

A null value for pulse parameters is forbidden and may lead to an unpredictable behavior.

In read and write cycles, the setup and hold time parameters are defined in reference to the address bus. For external devices that require setup and hold time between NCS and NRD signals (read), or between NCS and NWE signals (write), these setup and hold times must be converted into setup and hold times in reference to the address bus.

34.11 Scrambling/Unscrambling Function

The external data bus D[15:0] can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the memory device.

The scrambling and unscrambling are performed on-the-fly without additional wait states.

The scrambling method depends on two user-configurable key registers, HSMC_KEY1 and HSMC_KEY2. These key registers are only accessible in Write mode.

The key must be securely stored in a reliable nonvolatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

The scrambling/unscrambling function is enabled or disabled by configuring specific bits in the HSMC_OCMS and the HSMC_TIMINGSx registers. The bit configuration values to enable memory scrambling are summarized in Table 34-7.

Table 34-7. Scrambling Function Bit Encoding

Memories	Bit Values		
	HSMC_OCMS.SMSE	HSMC_OCMS.SRSE	HSMC_TIMINGSx.OCMS
Off-chip Memories	1	0	1
NAND Flash with NFC	0	1	0

When the NAND Flash memory content is scrambled, the on-chip NFC SRAM page buffer associated for the transfer is also scrambled.

34.12 Automatic Wait States

Under certain circumstances, the SMC automatically inserts idle cycles between accesses to avoid bus contention or operation conflict.

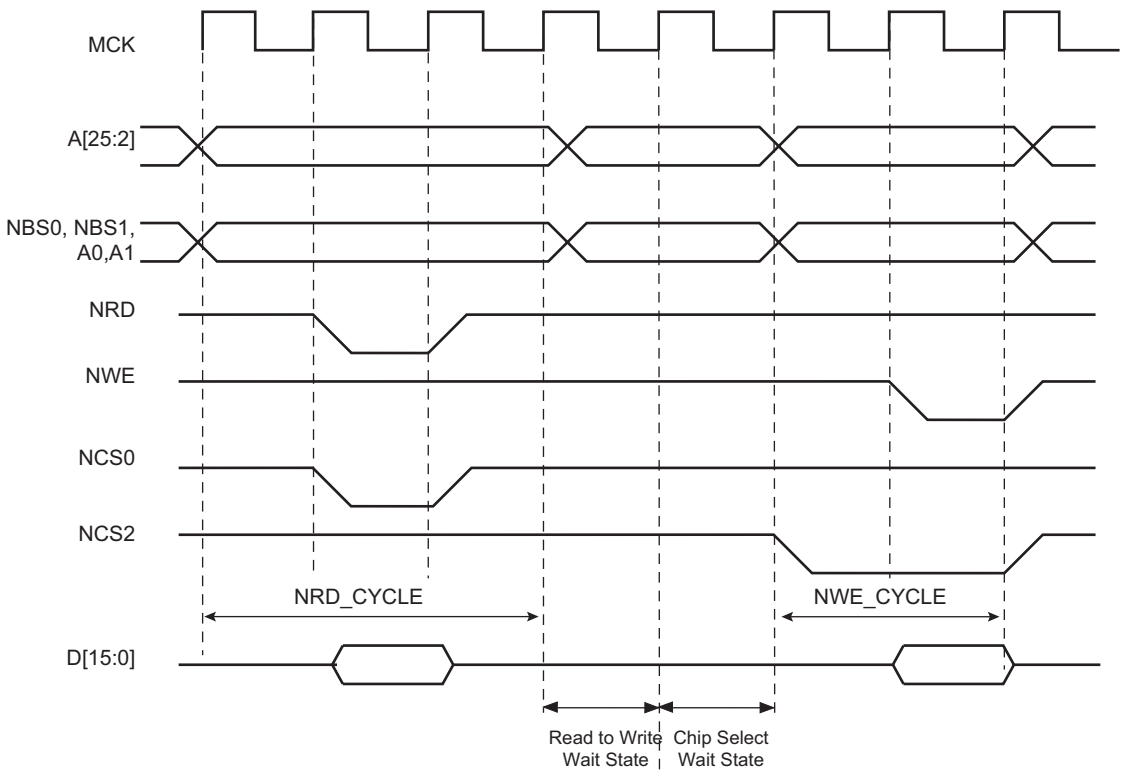
34.12.1 Chip Select Wait States

The SMC always inserts an idle cycle between two transfers on separate chip selects. This idle cycle ensures that there is no bus contention between the deactivation of one device and the activation of the next one.

During chip select wait state, all control lines are turned inactive: NBS0 to NBS1, NWR0 to NWR1, NCS[0..3], and NRD lines. They are all set to 1.

Figure 34-13 illustrates a chip select wait state between access on Chip Select 0 and Chip Select 2.

Figure 34-13. Chip Select Wait State between a Read Access on NCS0 and a Write Access on NCS2



34.12.2 Early Read Wait State

In some cases, the SMC inserts a wait state cycle between a write access and a read access to allow time for the write cycle to end before the subsequent read cycle begins. This wait state is not generated in addition to a chip select wait state. The early read cycle thus only occurs between a write and read access to the same memory device (same chip select).

An early read wait state is automatically inserted if at least one of the following conditions is valid:

- if the write controlling signal has no hold time and the read controlling signal has no setup time (Figure 34-14).
- in NCS Write Controlled mode (WRITE_MODE = 0), if there is no hold timing on the NCS signal and the NCS_RD_SETUP parameter is configured to 0, regardless of the Read mode (Figure 34-15). The write operation must end with an NCS rising edge. Without an Early Read Wait State, the write operation could not complete properly.
- in NWE Controlled mode (WRITE_MODE = 1) and if there is no hold timing (NWE_HOLD = 0), the feedback of the write control signal is used to control address, data, chip select and byte select lines. If the external write control signal is not inactivated as expected due to load capacitances, an Early Read Wait State is inserted and address, data and control signals are maintained one more cycle. See Figure 34-16.

Figure 34-14. Early Read Wait State: Write with No Hold Followed by Read with No Setup

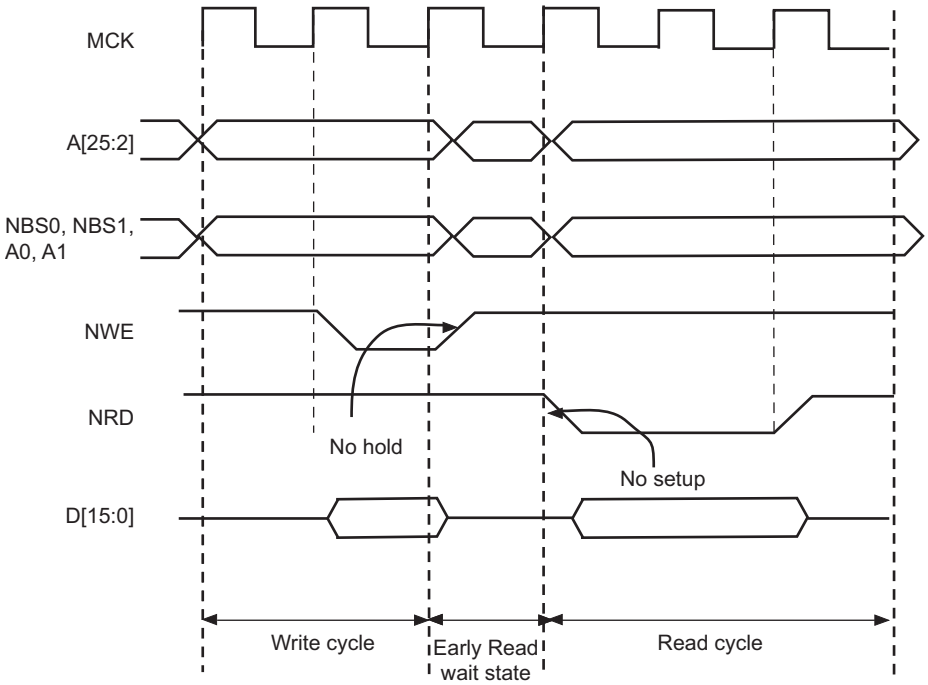


Figure 34-15. Early Read Wait State: NCS Controlled Write with No Hold Followed by a Read with No NCS Setup

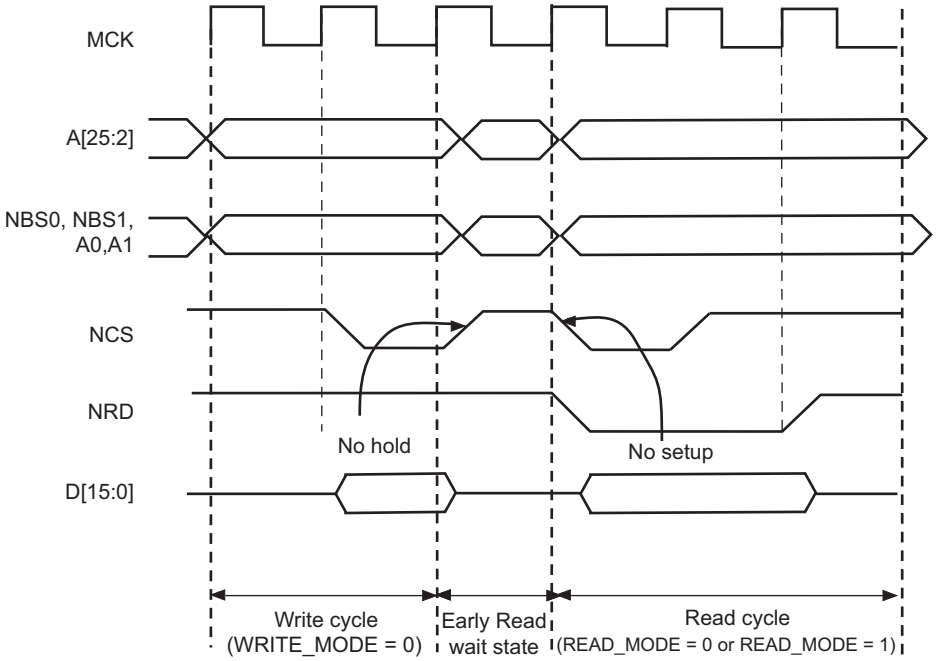
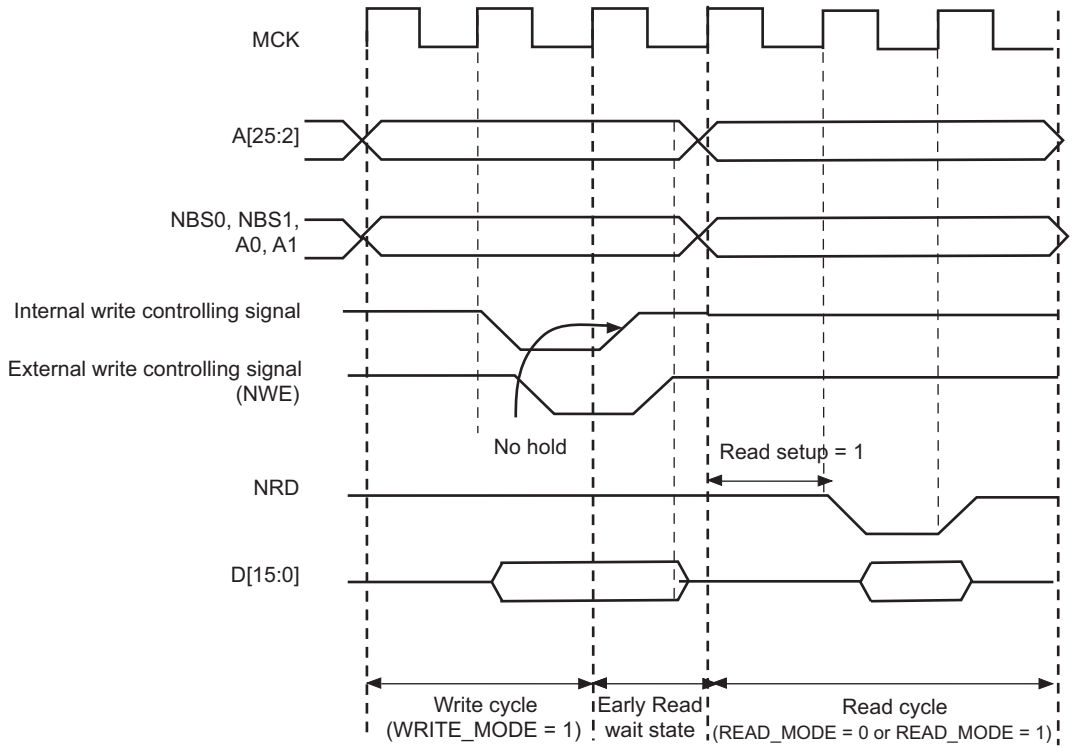


Figure 34-16. Early Read Wait State: NWE-controlled Write with No Hold Followed by a Read with one Set-up Cycle



34.12.3 Reload User Configuration Wait State

The user may change any of the configuration parameters by writing the SMC user interface.

When detecting that a new user configuration has been written in the user interface, the SMC inserts a wait state before starting the next access. The so called “Reload User Configuration Wait State” is used by the SMC to load the new set of parameters to apply to next accesses.

The Reload Configuration Wait State is not applied in addition to the Chip Select Wait State. If accesses before and after reprogramming the user interface are made to different devices (Chip Selects), then one single Chip Select Wait State is applied.

On the other hand, if accesses before and after writing the user interface are made to the same device, a Reload Configuration Wait State is inserted, even if the change does not concern the current Chip Select.

34.12.3.1 User Procedure

To insert a Reload Configuration Wait State, the SMC detects a write access to any HSMC_MODE register of the user interface. If only the timing registers are modified (HSMC_SETUP, HSMC_PULSE, HSMC_CYCLE registers) in the user interface, the user must validate the modification by writing the HSMC_MODE register, even if no change was made on the mode parameters.

34.12.3.2 Slow Clock Mode Transition

A Reload Configuration Wait State is also inserted when the Slow Clock Mode is entered or exited, after the end of the current transfer (see Section 34.15 “Slow Clock Mode”).

34.12.4 Read to Write Wait State

Due to an internal mechanism, a wait cycle is always inserted between consecutive read and write SMC accesses.

This wait cycle is referred to as a read to write wait state in this document.

This wait cycle is applied in addition to chip select and reload user configuration wait states when they are to be inserted. See Figure 34-13.

34.13 Data Float Wait States

Some memory devices are slow to release the external bus. For such devices, it is necessary to add wait states (data float wait states) after a read access:

- before starting a read access to a different external memory
- before starting a write access to the same device or to a different external one.

The Data Float Output Time (t_{DF}) for each external memory device is programmed in the TDF_CYCLES field of the HSMC_MODE register for the corresponding chip select. The value of TDF_CYCLES indicates the number of data float wait cycles (between 0 and 15) before the external device releases the bus, and represents the time allowed for the data output to go to high impedance after the memory is disabled.

Data float wait states do not delay internal memory accesses. Hence, a single access to an external memory with long t_{DF} will not slow down the execution of a program from internal memory.

The data float wait states management depends on the READ_MODE and the TDF_MODE bits of the HSMC_MODE register for the corresponding chip select.

34.13.1 READ_MODE

Setting READ_MODE to 1 indicates to the SMC that the NRD signal is responsible for turning off the tri-state buffers of the external memory device. The Data Float Period then begins after the rising edge of the NRD signal and lasts TDF_CYCLES MCK cycles.

When the read operation is controlled by the NCS signal (READ_MODE = 0), the TDF_CYCLES field in HSMC_MODEx gives the number of MCK cycles during which the data bus remains busy after the rising edge of NCS.

Figure 34-17 illustrates the Data Float Period in NRD-controlled mode (READ_MODE = 1), assuming a data float period of two cycles (TDF_CYCLES = 2). Figure 34-18 shows the read operation when controlled by NCS (READ_MODE = 0) and the TDF_CYCLES parameter equals 3.

Figure 34-17. TDF Period in NRD Controlled Read Access (TDF = 2)

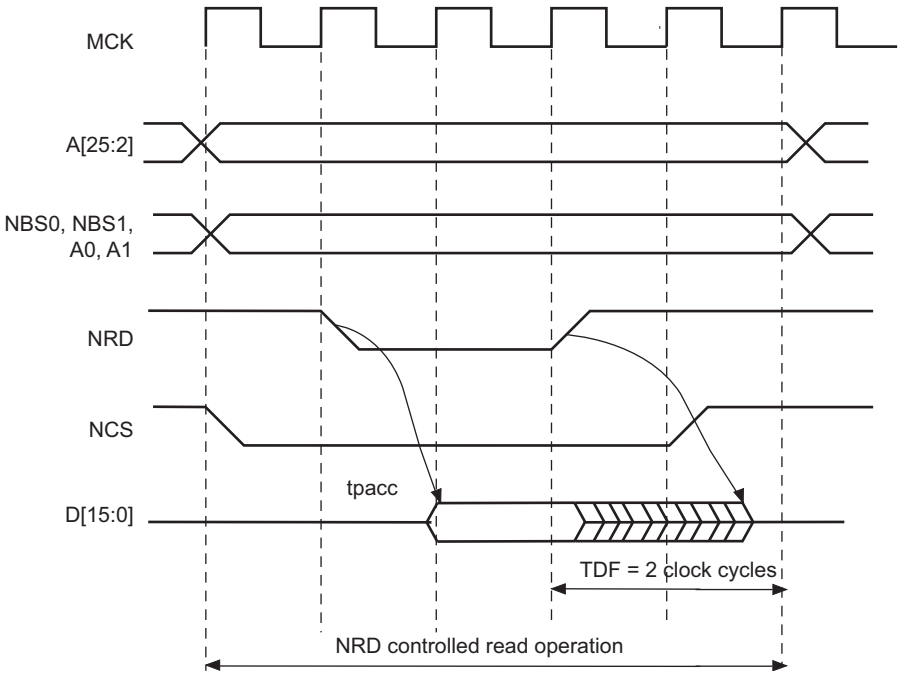
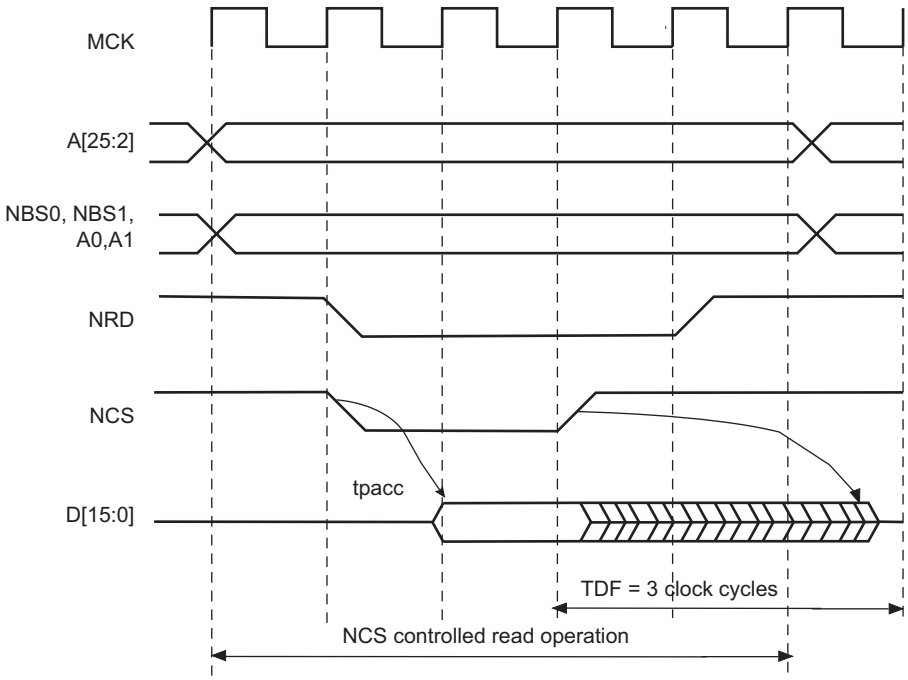


Figure 34-18. TDF Period in NCS Controlled Read Operation (TDF = 3)



34.13.2 TDF Optimization Enabled (TDF_MODE = 1)

When the TDF_MODE of the HSMC_MODE register is set to 1 (TDF optimization is enabled), the SMC takes advantage of the setup period of the next access to optimize the number of wait states cycle to insert.

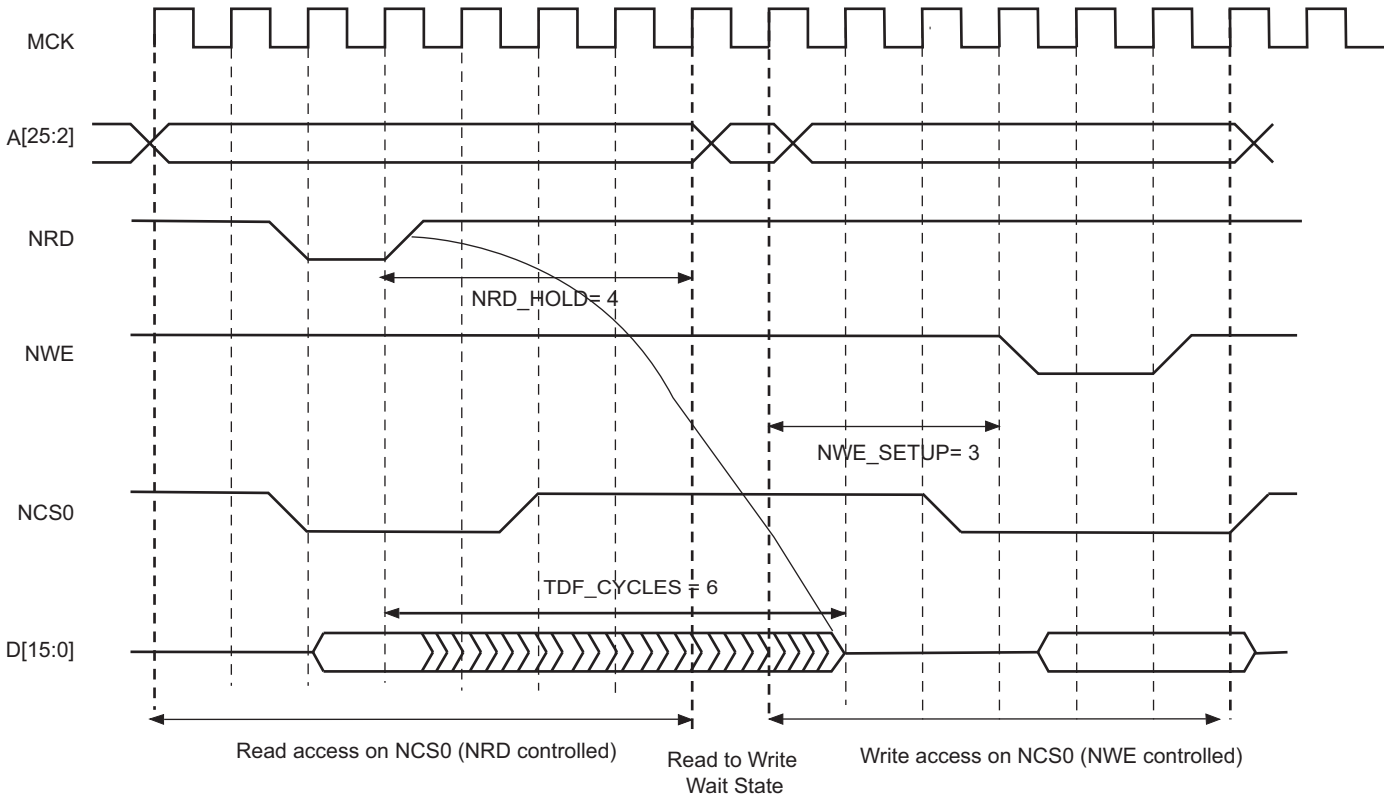
Figure 34-19 shows a read access controlled by NRD, followed by a write access controlled by NWE, on Chip Select 0. Chip Select 0 has been programmed with:

NRD_HOLD = 4; READ_MODE = 1 (NRD controlled)

NWE_SETUP = 3; WRITE_MODE = 1 (NWE controlled)

TDF_CYCLES = 6; TDF_MODE = 1 (optimization enabled).

Figure 34-19. TDF Optimization: No TDF wait states are inserted if the TDF period is over when the next access begins



34.13.3 TDF Optimization Disabled (TDF_MODE = 0)

When optimization is disabled, TDF wait states are inserted at the end of the read transfer, so that the data float period ends when the second access begins. If the hold period of the read1 controlling signal overlaps the data float period, no additional TDF wait states will be inserted.

Figure 34-20, Figure 34-21 and Figure 34-22 illustrate the cases:

- read access followed by a read access on another chip select,
- read access followed by a write access on another chip select,
- read access followed by a write access on the same chip select,

with no TDF optimization.

Figure 34-20. TDF Optimization Disabled (TDF Mode = 0). TDF wait states between 2 read accesses on different chip selects

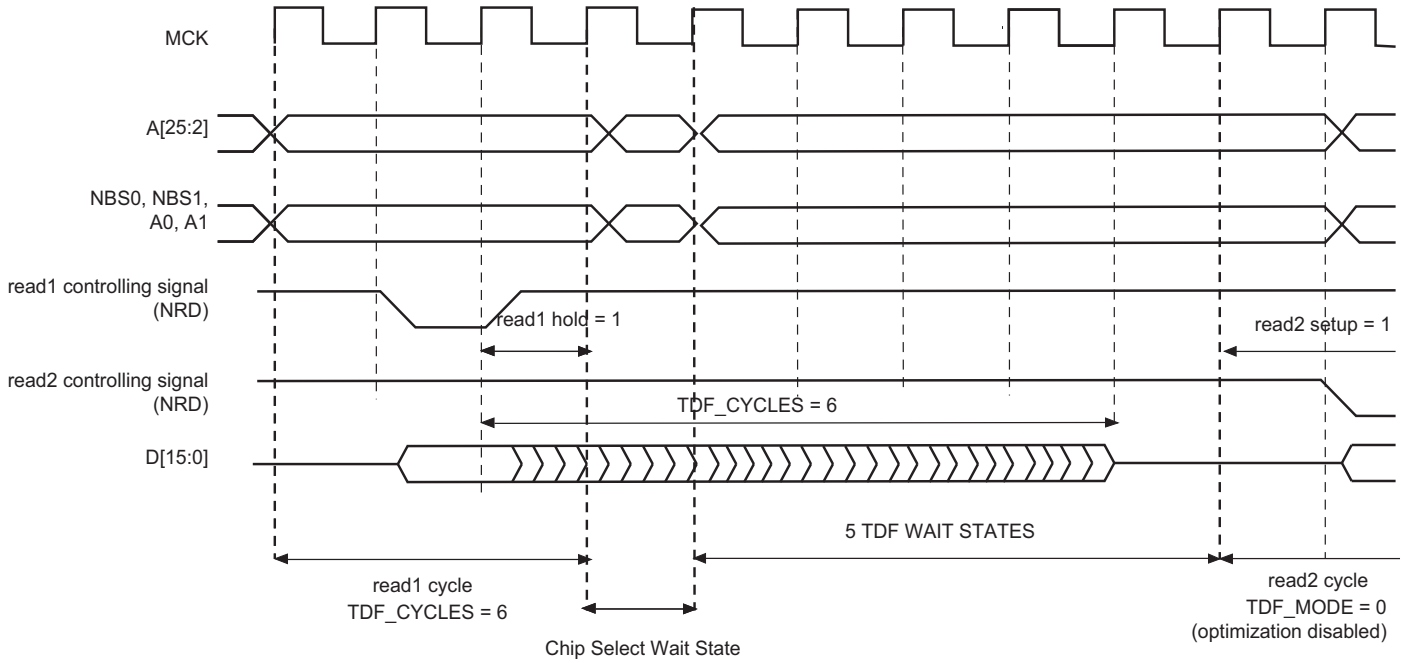


Figure 34-21. TDF Mode = 0: TDF wait states between a read and a write access on different chip selects

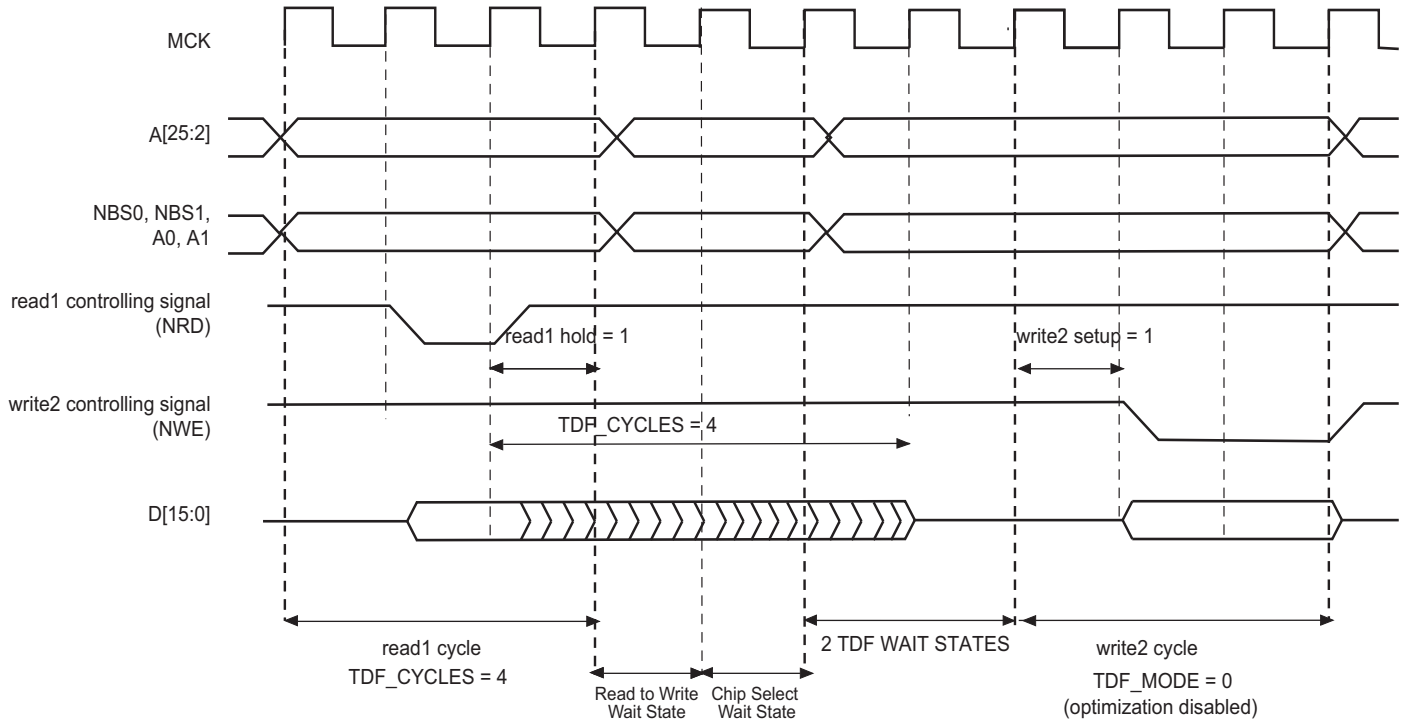
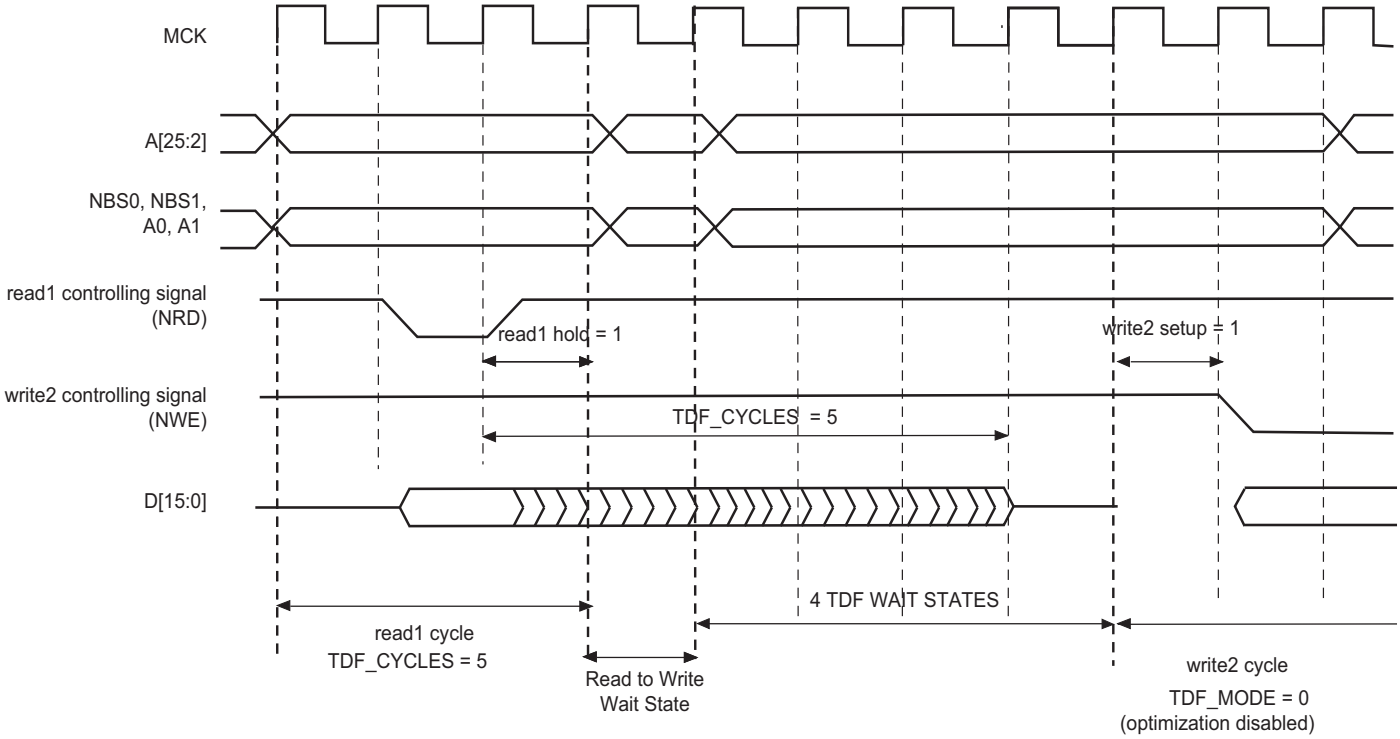


Figure 34-22. TDF Mode = 0: TDF wait states between read and write accesses on the same chip select



34.14 External Wait

Any access can be extended by an external device using the NWAIT input signal of the SMC. The EXNW_MODE field of the HSMC_MODE register on the corresponding chip select must be set to either ‘10’ (Frozen mode) or ‘11’ (Ready mode). When the EXNW_MODE is set to ‘00’ (disabled), the NWAIT signal is simply ignored on the corresponding chip select. The NWAIT signal delays the read or write operation in regards to the read or write controlling signal, depending on the Read and Write modes of the corresponding chip select.

34.14.1 Restriction

When one of the EXNW_MODE is enabled, it is mandatory to program at least one hold cycle for the read/write controlling signal. For that reason, the NWAIT signal cannot be used in Slow Clock Mode (Section 34.15 “Slow Clock Mode”).

The NWAIT signal is assumed to be a response of the external device to the read/write request of the SMC. NWAIT is then examined by the SMC in the pulse state of the read or write controlling signal. The assertion of the NWAIT signal outside the expected period has no impact on the SMC behavior.

34.14.2 Frozen Mode

When the external device asserts the NWAIT signal (active low), and after an internal synchronization of this signal, the SMC state is frozen, i.e., SMC internal counters are frozen, and all control signals remain unchanged. When the resynchronized NWAIT signal is deasserted, the SMC completes the access, resuming the access from the point where it was stopped. See Figure 34-23. This mode must be selected when the external device uses the NWAIT signal to delay the access and to freeze the SMC.

The assertion of the NWAIT signal outside the expected period is ignored as illustrated in Figure 34-24.

Figure 34-23. Write Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)

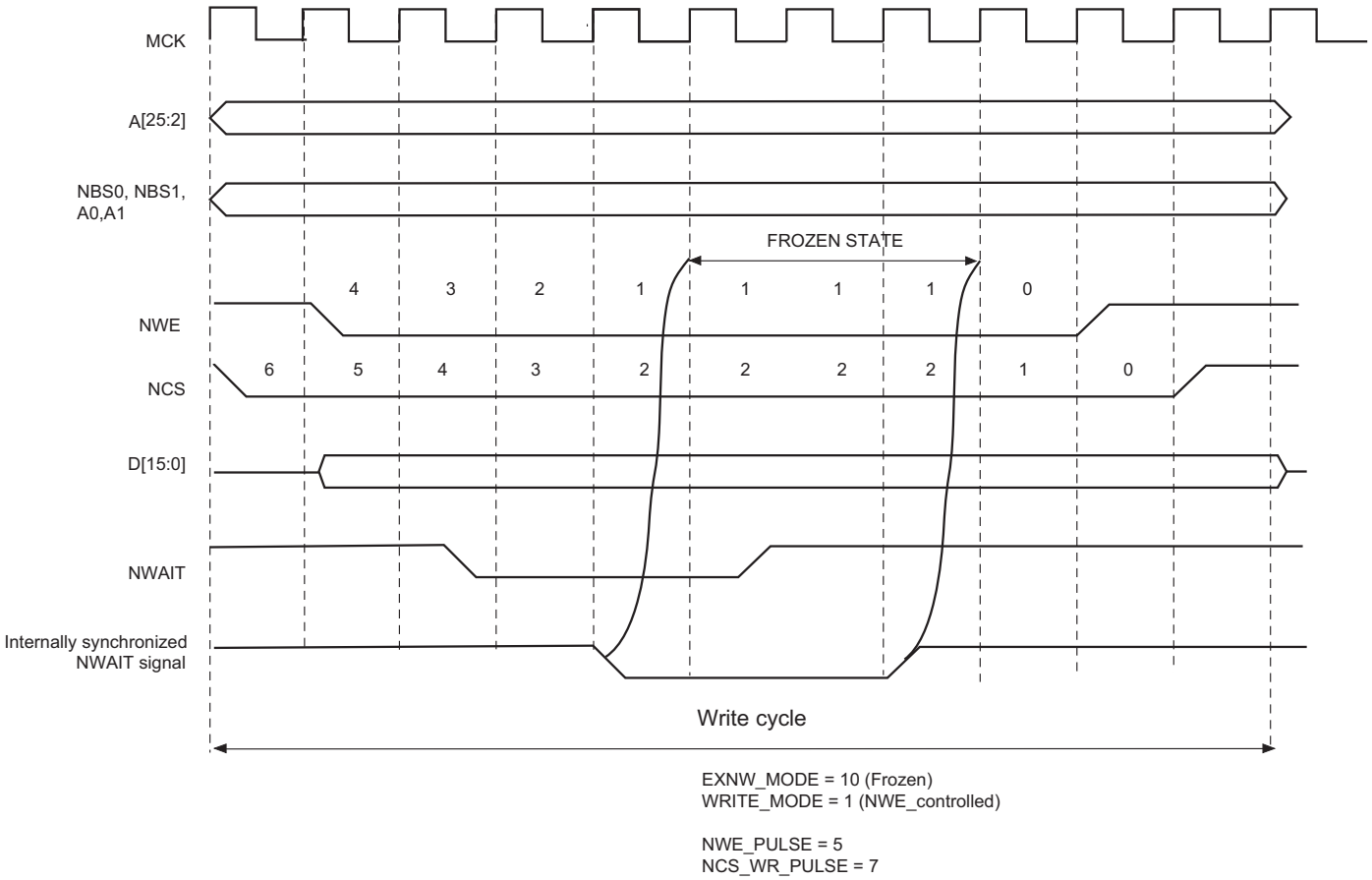
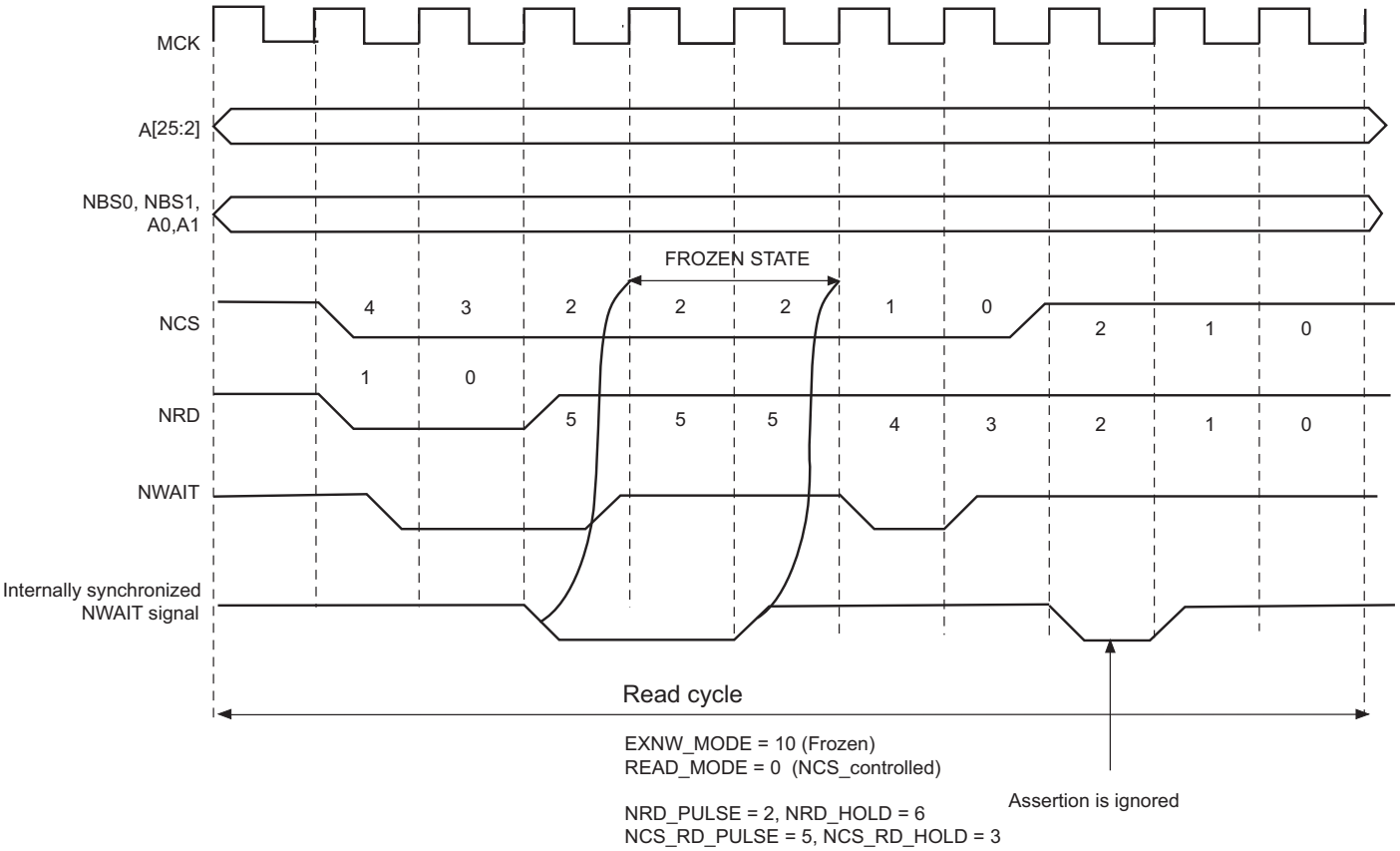


Figure 34-24. Read Access with NWAIT Assertion in Frozen Mode (EXNW_MODE = 10)



34.14.3 Ready Mode

In Ready mode (EXNW_MODE = 11), the SMC behaves differently. Normally, the SMC begins the access by down counting the setup and pulse counters of the read/write controlling signal. In the last cycle of the pulse phase, the resynchronized NWAIT signal is examined.

If asserted, the SMC suspends the access as shown in Figure 34-25 and Figure 34-26. After deassertion, the access is completed: the hold step of the access is performed.

This mode must be selected when the external device uses deassertion of the NWAIT signal to indicate its ability to complete the read or write operation.

If the NWAIT signal is deasserted before the end of the pulse, or asserted after the end of the pulse of the controlling read/write signal, it has no impact on the access length as shown in Figure 34-26.

Figure 34-25. NWAIT Assertion in Write Access: Ready Mode (EXNW_MODE = 11)

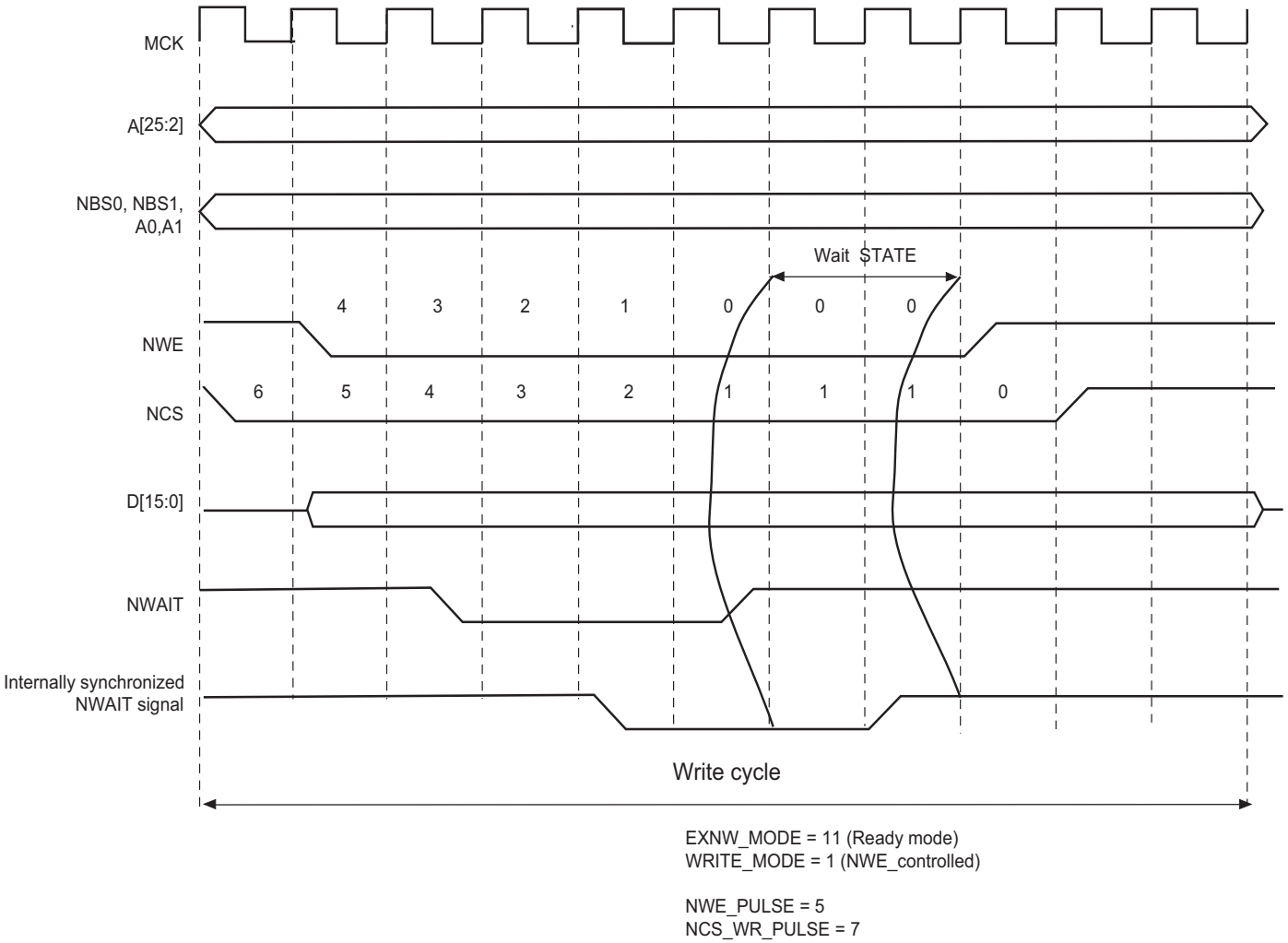
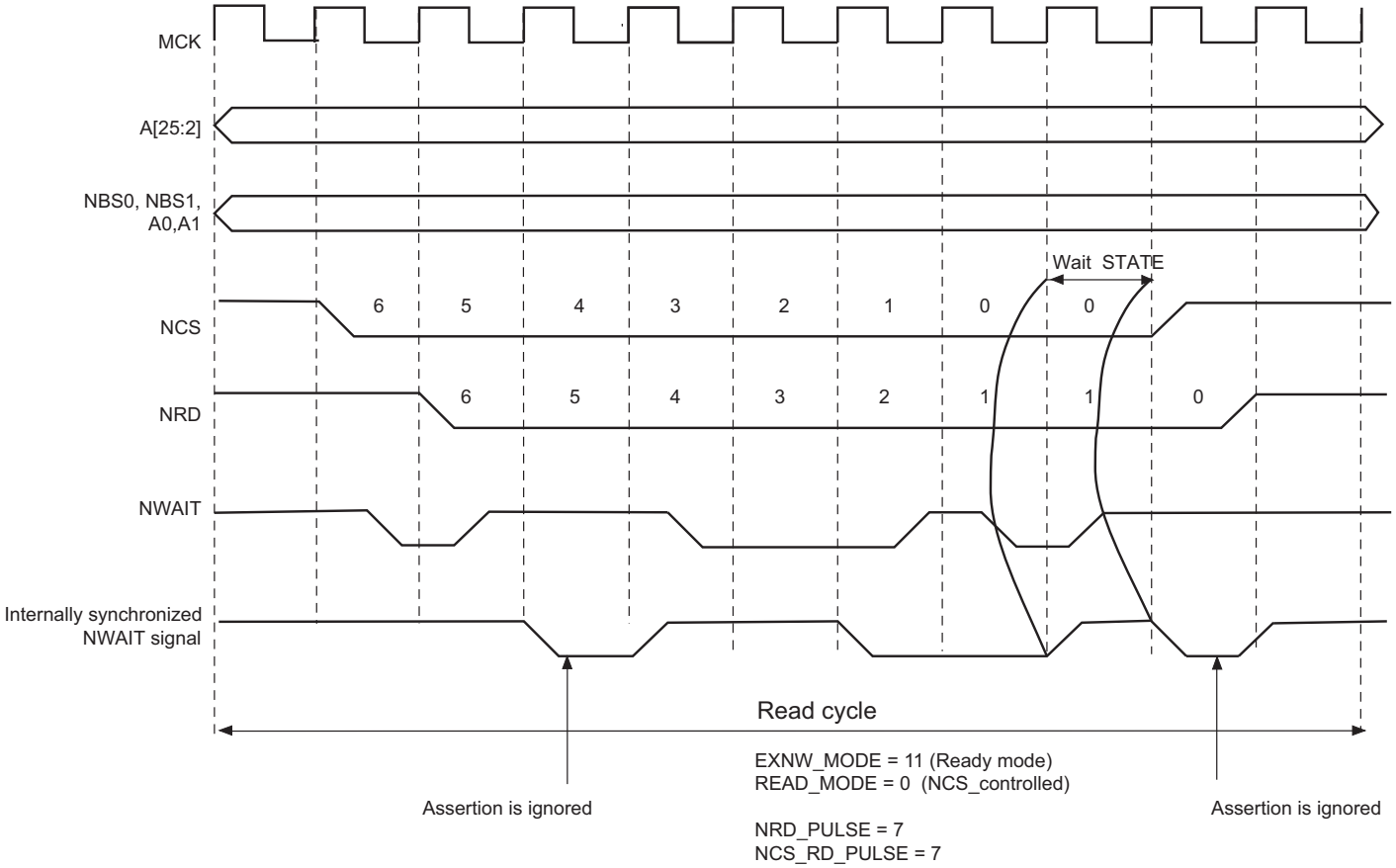


Figure 34-26. NWAIT Assertion in Read Access: Ready Mode (EXNW_MODE = 11)



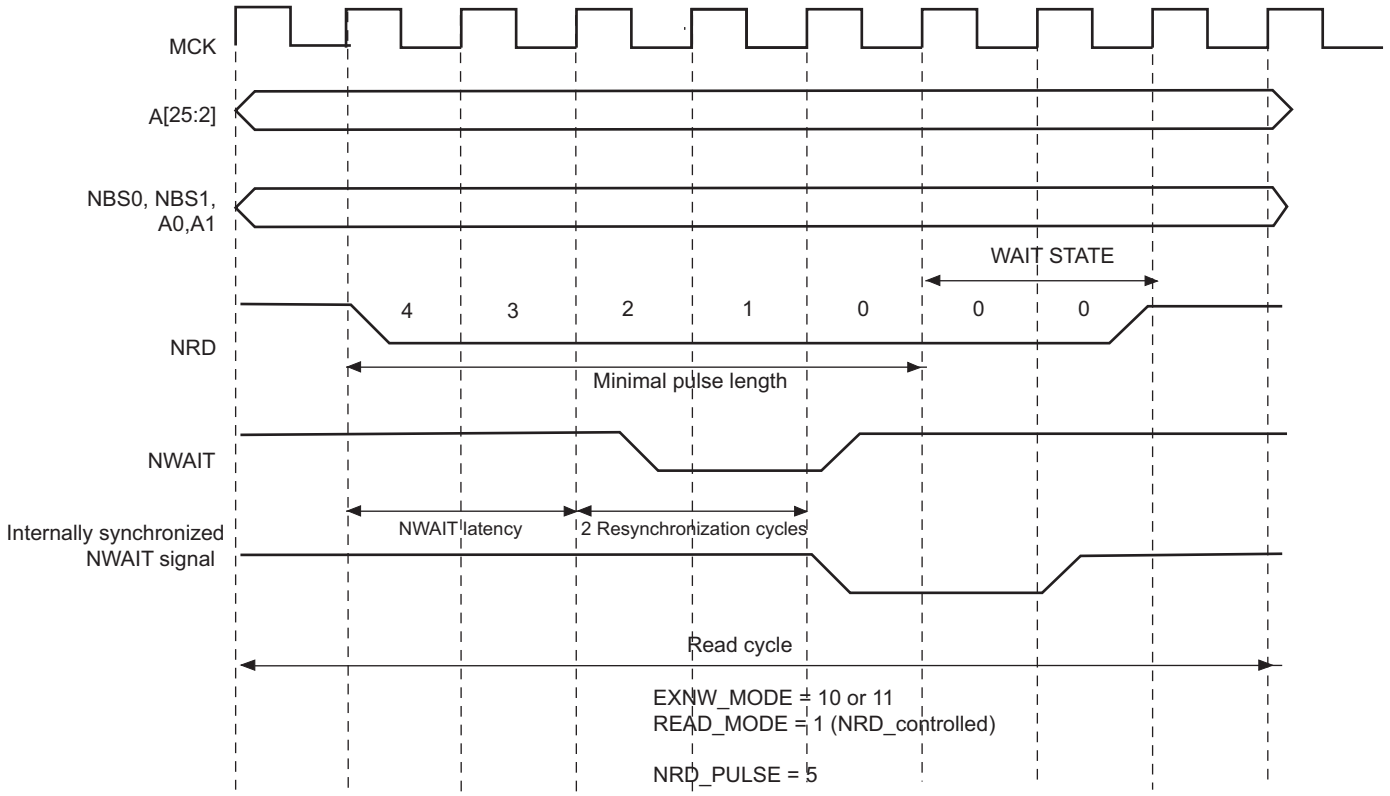
34.14.4 NWAIT Latency and Read/Write Timings

There may be a latency between the assertion of the read/write controlling signal and the assertion of the NWAIT signal by the device. The programmed pulse length of the read/write controlling signal must be at least equal to this latency plus the 2 cycles of resynchronization + 1 cycle. Otherwise, the SMC may enter the hold state of the access without detecting the NWAIT signal assertion. This is true in Frozen mode as well as in Ready mode. This is illustrated on Figure 34-27.

When EXNW_MODE is enabled (ready or frozen), the user must program a pulse length of the read and write controlling signal of at least:

minimal pulse length = NWAIT latency + 2 resynchronization cycles + 1 cycle

Figure 34-27. NWAIT Latency



34.15 Slow Clock Mode

The SMC is able to automatically apply a set of “Slow Clock mode” read/write waveforms when an internal signal driven by the Power Management Controller is asserted because MCK has been turned to a very slow clock rate (typically 32 kHz clock rate). In this mode, the user-programmed waveforms are ignored and the Slow Clock mode waveforms are applied. This mode is provided so as to avoid reprogramming the User Interface with appropriate waveforms at very slow clock rate. When activated, the Slow mode is active on all chip selects.

34.15.1 Slow Clock Mode Waveforms

Figure 34-28 illustrates the read and write operations in Slow Clock mode. They are valid on all chip selects. Table 34-8 indicates the value of read and write parameters in Slow Clock mode.

Figure 34-28. Write/Read Cycles in Slow Clock Mode

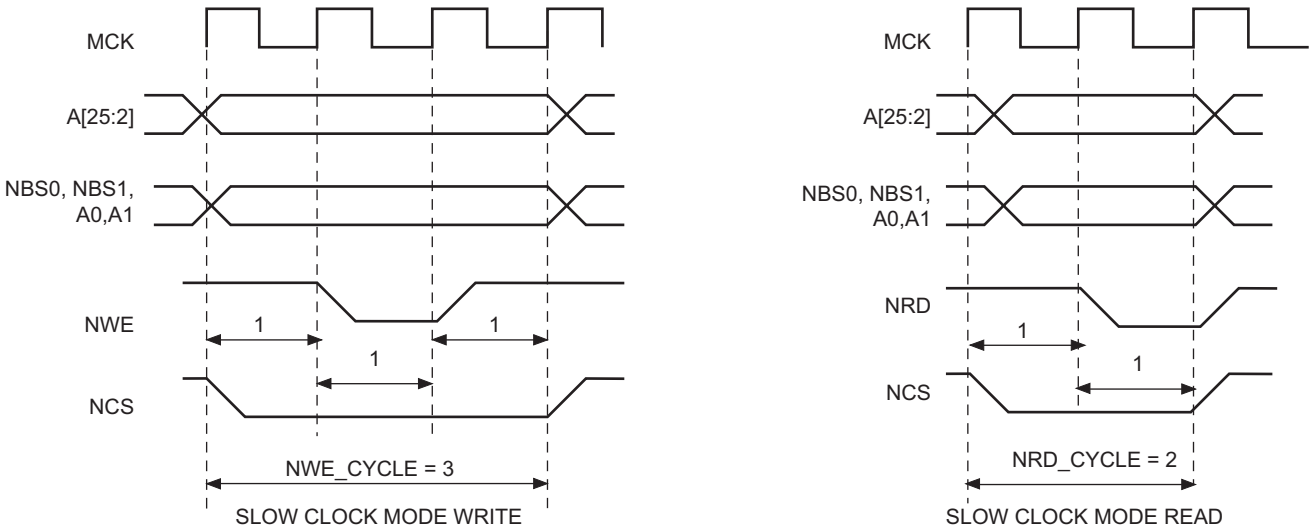


Table 34-8. Read and Write Timing Parameters in Slow Clock Mode

Read Parameters	Duration (cycles)	Write Parameters	Duration (cycles)
NRD_SETUP	1	NWE_SETUP	1
NRD_PULSE	1	NWE_PULSE	1
NCS_RD_SETUP	0	NCS_WR_SETUP	0
NCS_RD_PULSE	2	NCS_WR_PULSE	3
NRD_CYCLE	2	NWE_CYCLE	3

34.15.2 Switching from (to) Slow Clock Mode to (from) Normal Mode

When switching from Slow Clock mode to Normal mode, the current Slow Clock mode transfer is completed at high clock rate, with the set of Slow Clock mode parameters. See Figure 34-29. The external device may not be fast enough to support such timings.

Figure 34-30 illustrates the recommended procedure to properly switch from one mode to the other.

Figure 34-29. Clock Rate Transition occurs while the SMC is performing a Write Operation

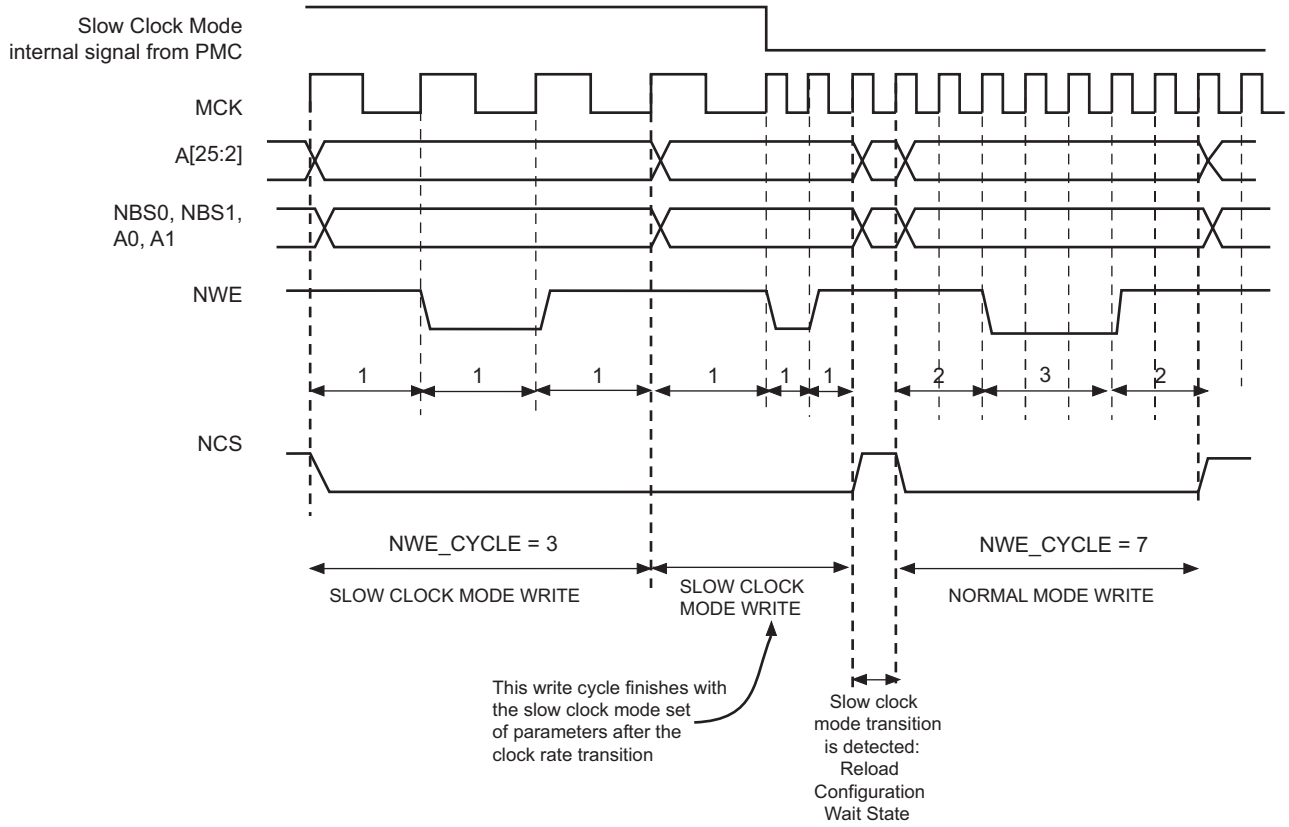
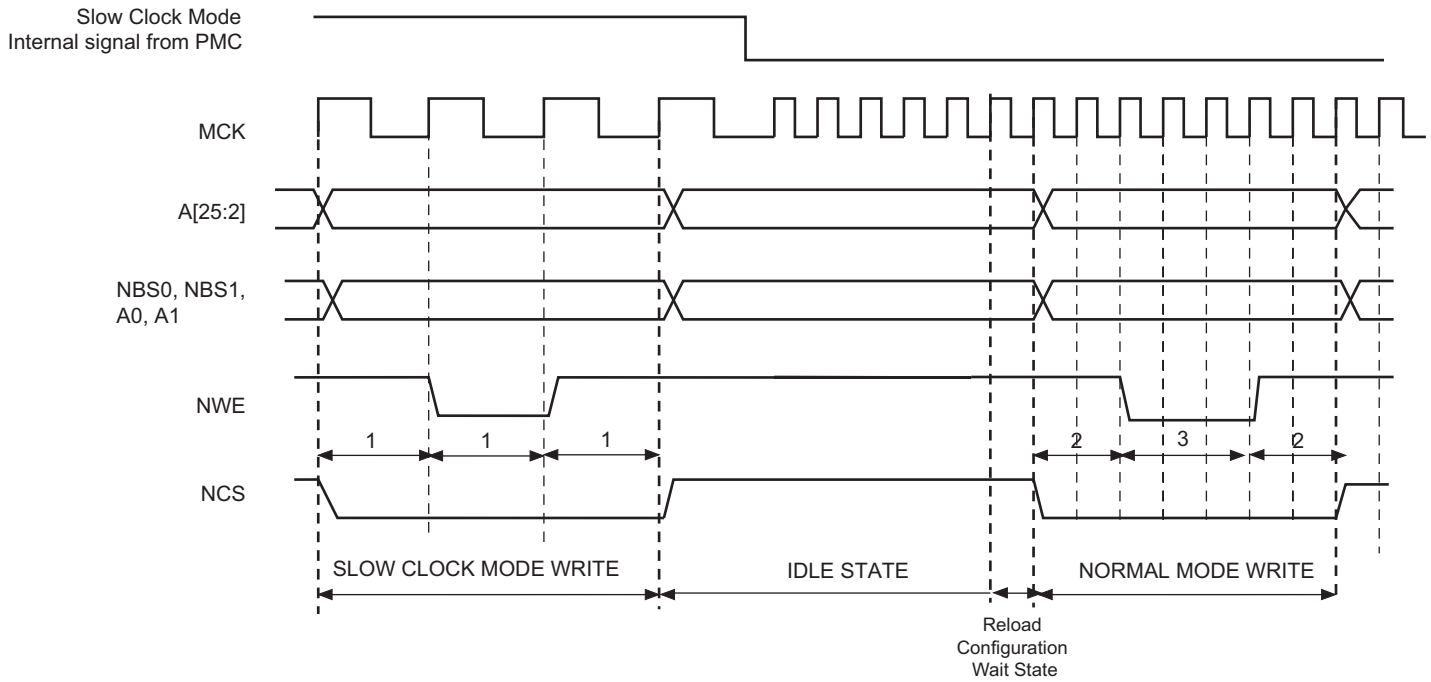


Figure 34-30. Recommended Procedure to Switch from Slow Clock Mode to Normal Mode or from Normal Mode to Slow Clock Mode



34.16 Register Write Protection

To prevent any single software error that may corrupt SMC behavior, selected registers can be write-protected by setting the WPEN bit in the Write Protection Mode Register (HSMC_WPMR).

If a write access in a write-protected register is detected, then the WPVS flag in the Write Protection Status Register (HSMC_WPSR) is set and the field WPVSRC indicates in which register the write access has been attempted.

The WPVS flag is automatically reset after reading the HSMC_WPSR.

The following registers can be write-protected:

- Setup Register
- Pulse Register
- Cycle Register
- Timings Register
- Mode Register

34.17 NFC Operations

34.17.1 NFC Overview

The NFC handles all the command, address and data sequences of the NAND low level protocol. An SRAM is used as an internal read/write buffer when data is transferred from or to the NAND.

34.17.2 NFC Control Registers

NAND Flash Read and NAND Flash Program operations can be performed through the NFC Command Registers. In order to minimize CPU intervention and latency, commands are posted in a command buffer. This buffer provides zero wait state latency. The detailed description of the command encoding scheme is explained below.

The NFC handles an automatic transfer between the external NAND Flash and the chip via the NFC SRAM. It is done via NFC Command Registers.

The NFC Command Registers are very efficient to use. When writing to these registers:

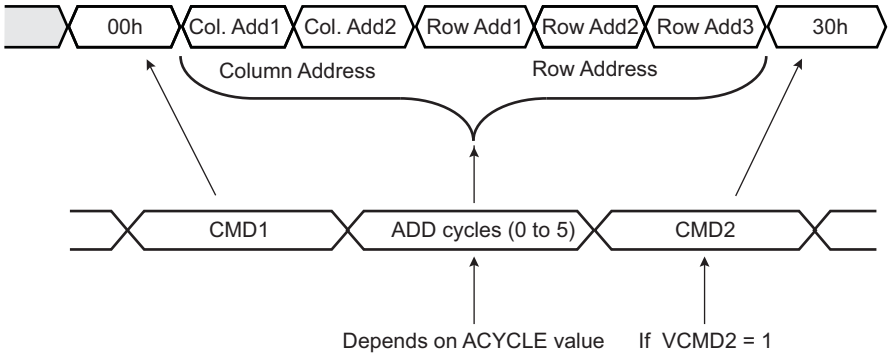
- the address of the register (NFCADDR_CMD) is the command used
- the data of the register (NFCDATA_ADDT) is the address to be sent to the NAND Flash

So, in one single access the command is sent and immediately executed by the NFC. Two commands can even be programmed within a single access (CMD1, CMD2) depending on the VCMD2 value.

The NFC can send up to five address cycles.

Figure 34-31 shows a typical NAND Flash Page Read Command of a NAND Flash Memory and correspondence with NFC Address Command Register.

Figure 34-31. NFC/NAND Flash Access Example



For more details refer to Section 34.17.2.2 “NFC Address Command”.

Reading the NFC Command Register (to any address) will give the status of the NFC. This is especially useful to know if the NFC is busy, for example.

34.17.2.1 Building NFC Address Command Example

The base address is made of HOST_ADDR address.

Page read operation example:

```
// Build the Address Command (NFCADDR_CMD)
AddressCommand = (HOST_ADDR                          |
                  NFCWR=0                            |// NFC Read Data from NAND
Flash
                  DATAEN=1                           |    // NFC Data phase
Enable.
                  CSID=1                             |        // Chip Select ID =
1
                 ACYCLE= 5                           |     // Number of address
cycle.
                 VCMD2=1                             | // CMD2 is sent after
Address Cycles
                 CMD2=0x30                           | // CMD2 = 30h
                 CMD1=0x0)    // CMD1 = Read Command = 00h

  // Set the Address for Cycle 0
  HSMC_ADDR = Col. Add1

  // Write command with the Address Command built above
*AddressCommand = (Col. Add2   |// ADDR_CYCLE1
           Row Add1  | // ADDR_CYCLE2
           Row Add2  |// ADDR_CYCLE3
           Row Add3  )// ADDR_CYCLE4
```

34.17.2.2 NFC Address Command

Name: NFCADDR_CMD

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	NFCWR	DATAEN	CSID

23	22	21	20	19	18	17	16
CSID		ACYCLE			VCMD2	CMD2	

15	14	13	12	11	10	9	8
CMD2						CMD1	

7	6	5	4	3	2	1	0
CMD1						–	–

- **CMD1: Command Register Value for Cycle 1**

When a write access occurs, the NFC sends this command.

- **CMD2: Command Register Value for Cycle 2**

When a write access occurs with the VCMD2 field set, the NFC sends this command after CMD1.

- **VCMD2: Valid Cycle 2 Command**

When set to true, the CMD2 field is issued after the address cycle.

- **ACYCLE: Number of Address Required for the Current Command**

When ACYCLE field is different from zero, ACYCLE Address cycles are performed after Command Cycle 1. The maximum number of cycles is 5.

- **CSID: Chip Select Identifier**

Chip select used

- **DATAEN: NFC Data Phase Enable**

When set to true, the NFC will automatically read or write data after the command.

- **NFCWR: NFC Write Enable**

0: NFC reads data from the NAND Flash.

1: NFC writes data into the NAND Flash.

34.17.2.3 NFC Data Address

Name: NFCDATA_ADDT

Access: Write-only

31	30	29	28	27	26	25	24
ADDR_CYCLE4							

23	22	21	20	19	18	17	16
ADDR_CYCLE3							

15	14	13	12	11	10	9	8
ADDR_CYCLE2							

7	6	5	4	3	2	1	0
ADDR_CYCLE1							

- **ADDR_CYCLE1: NAND Flash Array Address Cycle 1**

When less than five address cycles are used, ADDR_CYCLE1 is the first byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE1 is the second byte written to NAND Flash.

- **ADDR_CYCLE2: NAND Flash Array Address Cycle 2**

When less than five address cycles are used, ADDR_CYCLE2 is the second byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE2 is the third byte written to the NAND Flash.

- **ADDR_CYCLE3: NAND Flash Array Address Cycle 3**

When less than five address cycles are used, ADDR_CYCLE3 is the third byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE3 is the fourth byte written to the NAND Flash.

- **ADDR_CYCLE4: NAND Flash Array Address Cycle 4**

When less than five address cycles are used, ADDR_CYCLE4 is the fourth byte written to the NAND Flash.

When five address cycles are used, ADDR_CYCLE4 is the fifth byte written to the NAND Flash.

Note: If five address cycles are used, the first address cycle is ADDR_CYCLE0. Refer to HSMC_ADDR register.

34.17.2.4 NFC DATA Status

Name: NFCDATA_STATUS

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	NFCBUSY	NFCWR	DATAEN	CSID

23	22	21	20	19	18	17	16
CSID		ACYCLE			VCMD2	CMD2	

15	14	13	12	11	10	9	8
CMD2						CMD1	

7	6	5	4	3	2	1	0
CMD1						–	–

- **CMD1: Command Register Value for Cycle 1**

When a Read or Write Access occurs, the Physical Memory Interface drives the IO bus with CMD1 field during the Command Latch cycle 1.

- **CMD2: Command Register Value for Cycle 2**

When VCMD2 bit is set to true, the Physical Memory Interface drives the IO bus with CMD2 field during the Command Latch cycle 2.

- **VCMD2: Valid Cycle 2 Command**

When set to true, the CMD2 field is issued after addressing cycle.

- **ACYCLE: Number of Address Required for the Current Command**

When ACYCLE field is different from zero, ACYCLE Address cycles are performed after Command Cycle 1.

- **CSID: Chip Select Identifier**

Chip select used

- **DATAEN: NFC Data Phase Enable**

When set to true, the NFC data phase is enabled.

- **NFCWR: NFC Write Enable**

0: NFC is in Read mode.

1: NFC is in Write mode.

- **NFCBUSY: NFC Busy Status Flag**

If set to true, it indicates that the NFC is busy.

34.17.3 NFC Initialization

Prior to any Command and Data Transfer, the SMC User Interface must be configured to meet the device timing requirements.

- Write enable Configuration

Use NWE_SETUP, NWE_PULSE and NWE_CYCLE to define the write enable waveform according to the external device datasheet.

Use TADL field in the HSMC_TIMINGS register to configure the timing between the last address latch cycle and the first rising edge of WEN for data input.

Figure 34-32. Write Enable Timing Configuration

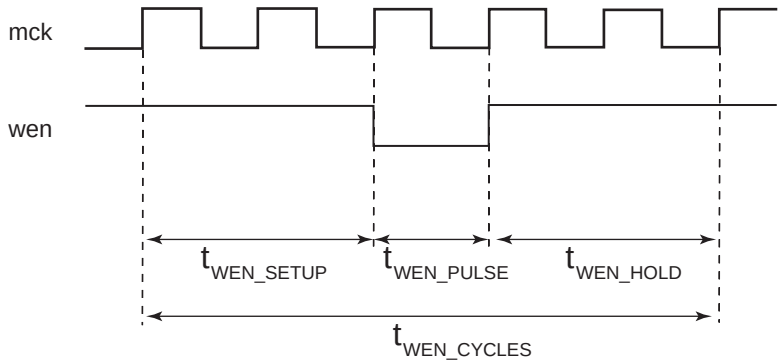
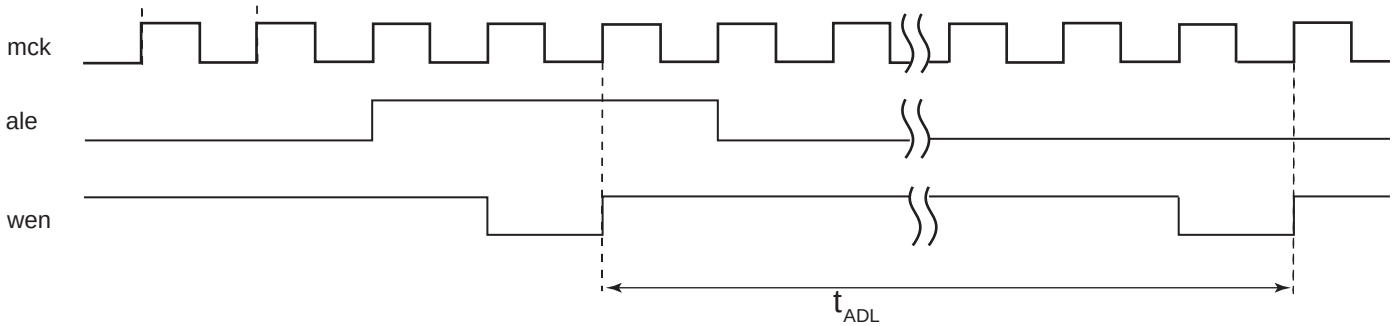


Figure 34-33. Write Enable Timing for NAND Flash Device Data Input Mode



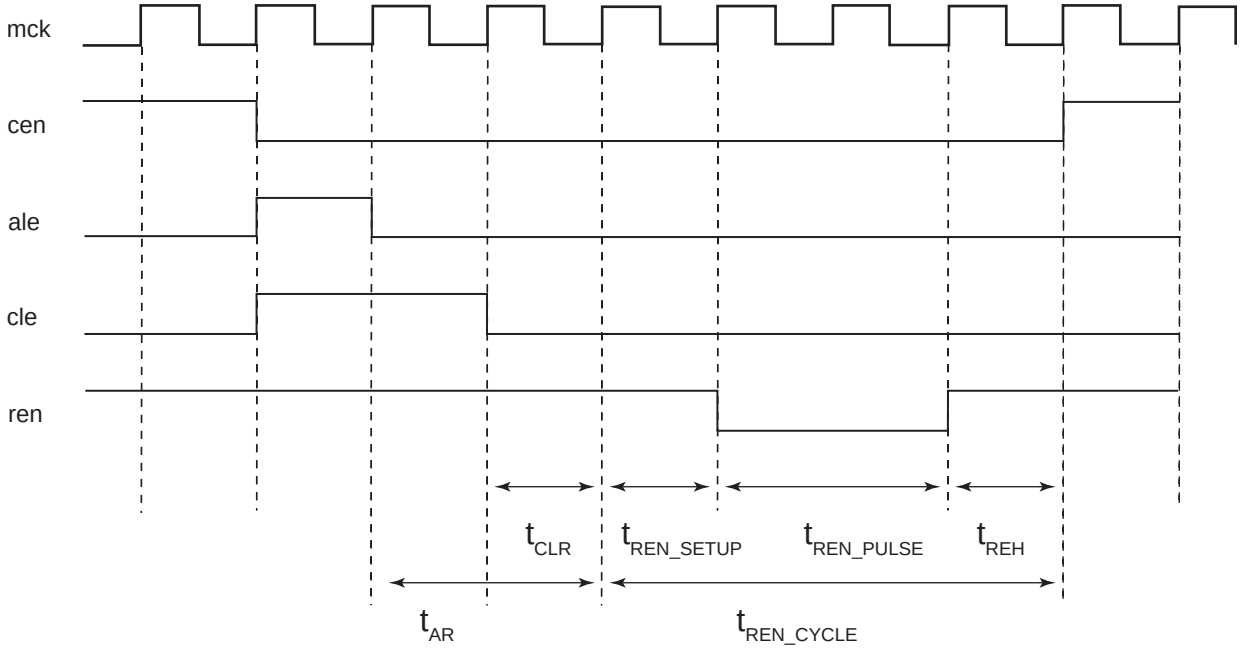
- Read Enable Configuration

Use NRD_SETUP, NRD_PULSE and NRD_CYCLE to define the read enable waveform according to the external device datasheet.

Use TAR field in the HSMC_TIMINGS register to configure the timings between the address latch enable falling edge to read the enable falling edge.

Use TCLR field in the HSMC_TIMINGS register to configure the timings between the command latch enable falling edge to read the enable falling edge.

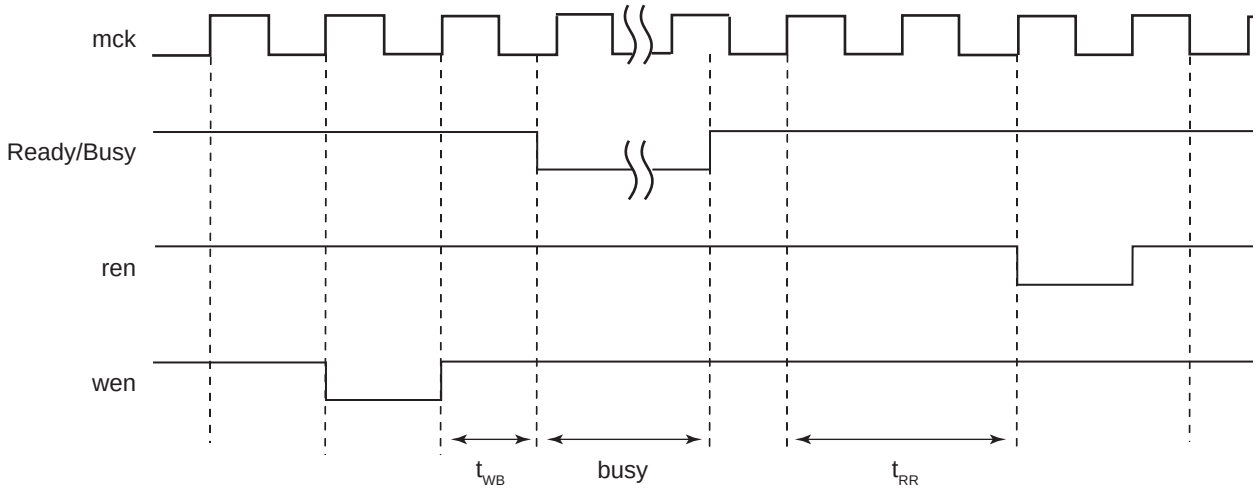
Figure 34-34. Read Enable Timing Configuration Working with NAND Flash Device



- Ready/Busy Signal Timing configuration working with a NAND Flash device

Use TWB field in HSMC_TIMINGS register to configure the maximum elapsed time between the rising edge of the wen signal and the falling edge of the Ready/Busy signal. Use TRR field in the HSMC_TIMINGS register to program the number of clock cycles between the rising edge of the Ready/Busy signal and the falling edge of the ren signal.

Figure 34-35. Ready/Busy Timing Configuration

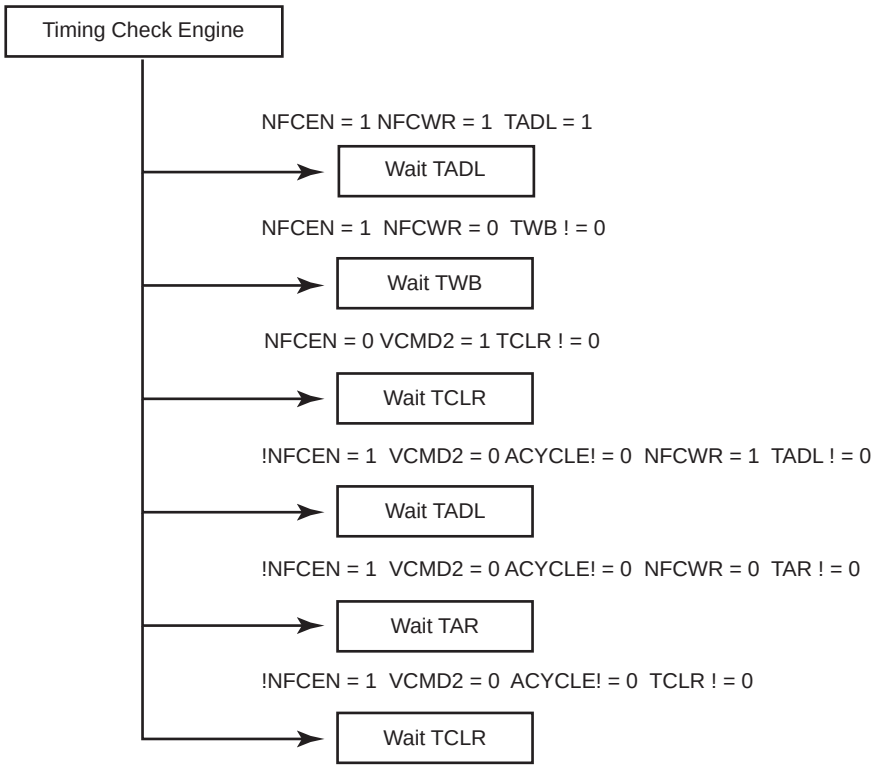


34.17.3.1 NFC Timing Engine

When the NFC Command register is written, the NFC issues a NAND Flash Command and optionally performs a data transfer between the NFC SRAM and the NAND Flash device. The NFC Timing Engine guarantees valid NAND Flash timings, depending on the set of parameters decoded from the address bus. These timings are defined in the HSMC_TIMINGS register.

For information on the timing used depending on the command, see Figure 34-36.

Figure 34-36. NFC Timing Engine



See the NFC Address Command register description and the Timings Register.

34.17.4 NFC SRAM

34.17.4.1 NFC SRAM Mapping

If the NFC is used to read and write data from and to the NAND Flash, the configuration depends on the page size (PAGESIZE field in HSMC_CFG register). See Table 34-9 to Table 34-13 for detailed mapping.

The NFC can handle the NAND Flash with a page size of 8 Kbytes or lower (such as 2 Kbytes, for example). In case of a 4 Kbyte or lower page size, the NFC SRAM can be split into two banks. The BANK bit in the HSMC_BANK register is used to select where NAND flash data are written or read. For an 8 Kbyte page size this field is not relevant.

Note that a “Ping-Pong” mode (write or read to a bank while the NFC writes or reads to another bank) is accessible with the NFC (using two different banks).

If the NFC is not used, the NFC SRAM can be used for a general purpose by the application.

Table 34-9. NFC SRAM Bank Mapping for 512 bytes

Offset	Use	Access
0x00000000–0x000001FF	Main Area Bank 0	Read/Write
0x00000200–0x000003FF	Spare Area Bank 0	Read/Write
0x00001200–0x000013FF	Main Area Bank 1	Read/Write
0x00001400–0x000015FF	Spare Area Bank 1	Read/Write

Table 34-10. NFC SRAM Bank Mapping for 1 Kbyte

Offset	Use	Access
0x00000000–0x000003FF	Main Area Bank 0	Read/Write
0x00000400–0x000005FF	Spare Area Bank 0	Read/Write
0x00001200–0x000015FF	Main Area Bank 1	Read/Write
0x00001600–0x000017FF	Spare Area Bank 1	Read/Write

Table 34-11. NFC SRAM Bank Mapping for 2 Kbytes

Offset	Use	Access
0x00000000–0x000007FF	Main Area Bank 0	Read/Write
0x00000800–0x000009FF	Spare Area Bank 0	Read/Write
0x00001200–0x000019FF	Main Area Bank 1	Read/Write
0x00001A00–0x00001BFF	Spare Area Bank 1	Read/Write

Table 34-12. NFC SRAM Bank Mapping for 4 Kbytes

Offset	Use	Access
0x00000000–0x00000FFF	Main Area Bank 0	Read/Write
0x00001000–0x000011FF	Spare Area Bank 0	Read/Write
0x00001200–0x000021FF	Main Area Bank 1	Read/Write
0x00002200–0x000023FF	Spare Area Bank 1	Read/Write

Table 34-13. NFC SRAM Bank Mapping for 8 Kbytes, only one bank is available

Offset	Use	Access
0x00000000–0x00001FFF	Main Area Bank 0	Read/Write
0x00002000–0x000023FF	Spare Area Bank 0	Read/Write

34.17.4.2 NFC SRAM Access Prioritization Algorithm

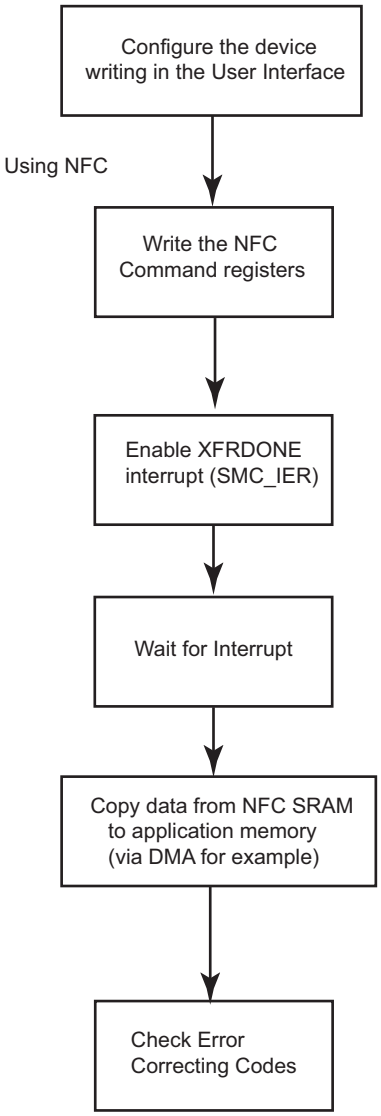
When the NFC is reading from or writing to an NFC SRAM bank, the other bank is available. If an NFC SRAM access occurs when the NFC performs a read or write operation in the same bank, then the access is discarded. The write operation is not performed. The read operation returns undefined data. If this situation is encountered, the AWB status flag located in the NFC Status Register is raised and indicates that a shared resource access violation has occurred.

34.17.5 NAND Flash Operations

This section describes the software operations needed to issue commands to the NAND Flash device and to perform data transfers using the NFC.

34.17.5.1 Page Read

Figure 34-37. Page Read Flow Chart

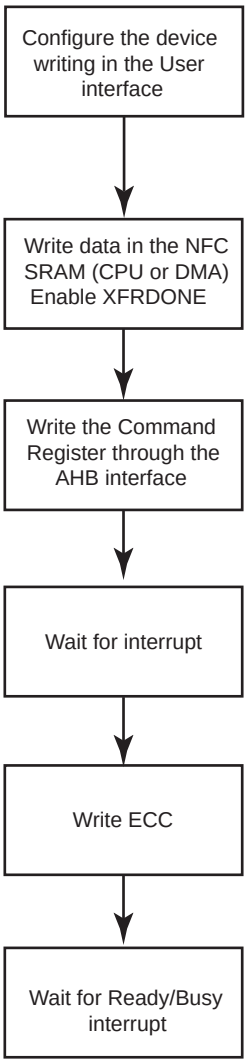


Note that, instead of using the interrupt, one can poll the NFCBUSY flag.

For more information on the NFC Control Register, see Section 34.17.2.2 “NFC Address Command”.

34.17.5.2 Program Page

Figure 34-38. Program Page Flow Chart



Writing the ECC cannot be done using the NFC; it needs to be done “manually”.

Note that, instead of using the interrupt, one can poll the NFCBUSY flag.

For more information on the NFC Control Register, see Section 34.17.2.2 “NFC Address Command”.

34.18 PMECC Controller Functional Description

The Programmable Multibit Error Correcting Code (PMECC) controller is a programmable binary BCH (Bose, Chaudhuri and Hocquenghem) encoder/decoder. This controller can be used to generate redundancy information for both SLC and MLC NAND devices. It supports redundancy for correction of 2, 4, 8, 12, 24, or 32 errors per sector of data. The sector size is programmable and can be set to 512 bytes or 1024 bytes. The PMECC module generates redundancy at encoding time, when a NAND write page operation is performed. The redundancy is appended to the page and written in the spare area. This operation is performed by the processor. It moves the content of the PMECCX registers into the NAND flash memory. The number of registers depends on the selected error correction capability (see Table 34-14 "Relevant Redundancy Registers"). This operation shall be executed for each sector. At decoding time, the PMECC module generates the remainders of the received codeword by the minimal polynomials. When all remainders for a given sector are set to zero, no error occurred. When the remainders are different from zero, the codeword is corrupted and further processing is required.

The PMECC module generates an interrupt indicating that an error occurred. The processor must read the PMECC Interrupt Status Register (HSMC_PMECCISR). This register indicates which sector is corrupted.

The processor must execute the following decoding steps to find the error location within a sector:

1. Syndrome computation.
2. Finding the error location polynomial.
3. Finding the roots of the error location polynomial.

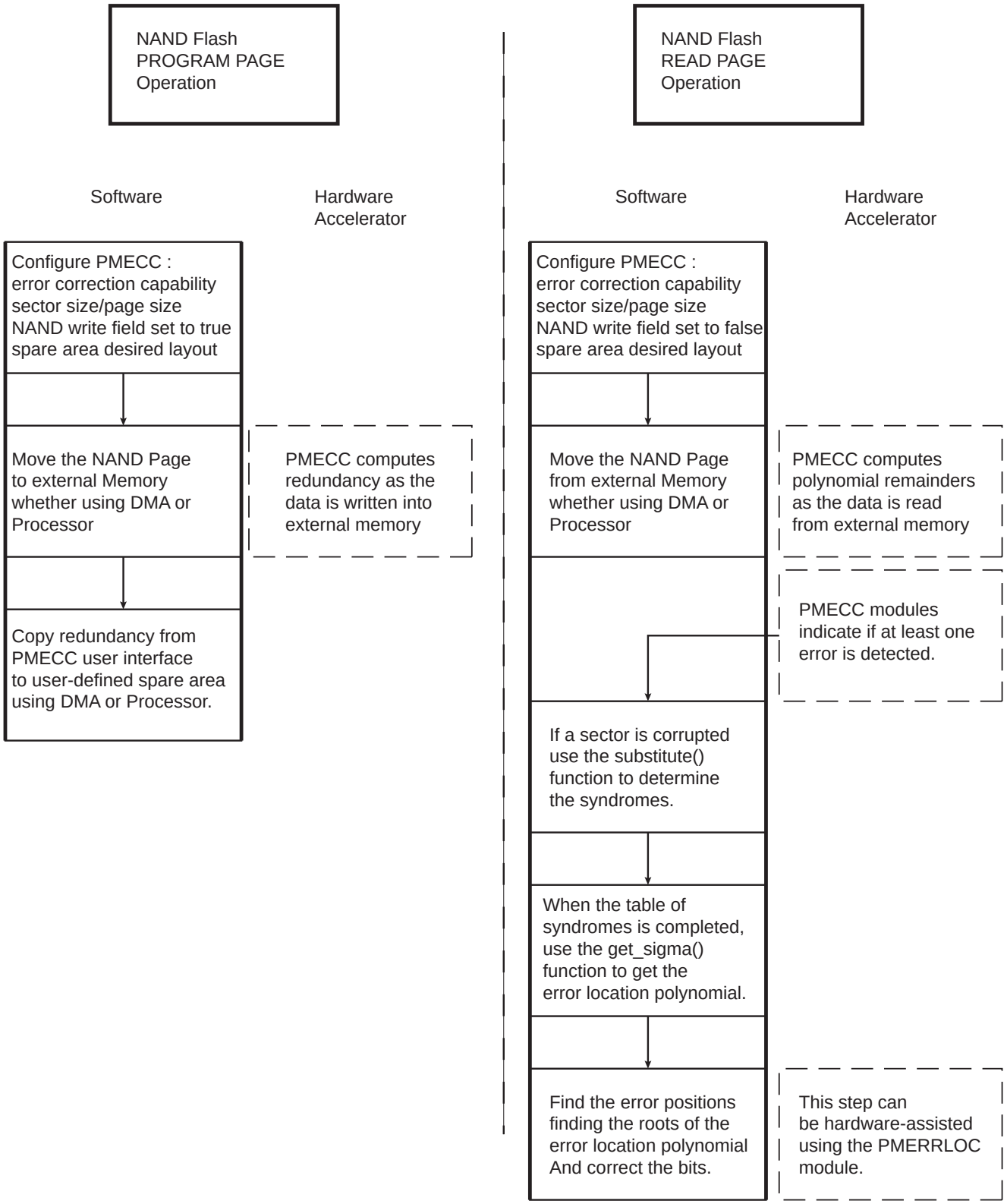
All decoding steps involve finite field computation. It means that a library of finite field arithmetic must be available to perform addition, multiplication and inversion. These arithmetic operations can be performed through the use of a memory mapped lookup table, or direct software implementation. The software implementation presented is based on lookup tables. Two tables named gf_log and gf_antilog are used. If alpha is the primitive element of the field, then a power of alpha is in the field. Assuming that beta = alpha ^ index, then beta belongs to the field, and gf_log(beta) = gf_log(alpha ^ index) = index. The gf_antilog table provides exponent inverse of the element; if beta = alpha ^ index, then gf_antilog(index) = beta.

The first step consists in the syndrome computation. The PMECC module computes the remainders and the software must substitute the power of the primitive element. The procedure implementation is given in Section 34.19.1 "Remainder Substitution Procedure".

The second step is the most software intensive. It is the Berlekamp's iterative algorithm for finding the error-location polynomial. The procedure implementation is given in Section 34.19.2 "Finding the Error Location Polynomial Sigma(x)".

The Last step is finding the root of the error location polynomial. This step can be very software intensive. Indeed there is no straightforward method of finding the roots, except evaluating each element of the field in the error location polynomial. However, a hardware accelerator can be used to find the roots of the polynomial. The PMERRLOC module provides this kind of hardware acceleration.

Figure 34-39. Software Hardware Multibit Error Correction Dataflow



34.18.1 MLC/SLC Write Page Operation Using PMECC

When an MLC write page operation is performed, the PMECC controller is configured with the NANDWR bit of the PMECCFG register set to one. When the NAND spare area contains file system information and redundancy (PMECCx), the spare area is error protected, then the SPAREEN bit of the PMECCFG register is set. When the NAND spare area contains only redundancy information, the SPAREEN bit is cleared.

When the write page operation is terminated, the user writes the redundancy in the NAND spare area. This operation can be done with DMA assistance.

Table 34-14. Relevant Redundancy Registers

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
0	PMECC0	PMECC0
1	PMECC0, PMECC1	PMECC0, PMECC1
2	PMECC0, PMECC1, PMECC2, PMECC3	PMECC0, PMECC1, PMECC2, PMECC3
3	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6
4	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10
5	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10, PMECC11, PMECC12	PMECC0, PMECC1, PMECC2, PMECC3, PMECC4, PMECC5, PMECC6, PMECC7, PMECC8, PMECC9, PMECC10, PMECC11, PMECC12, PMECC13

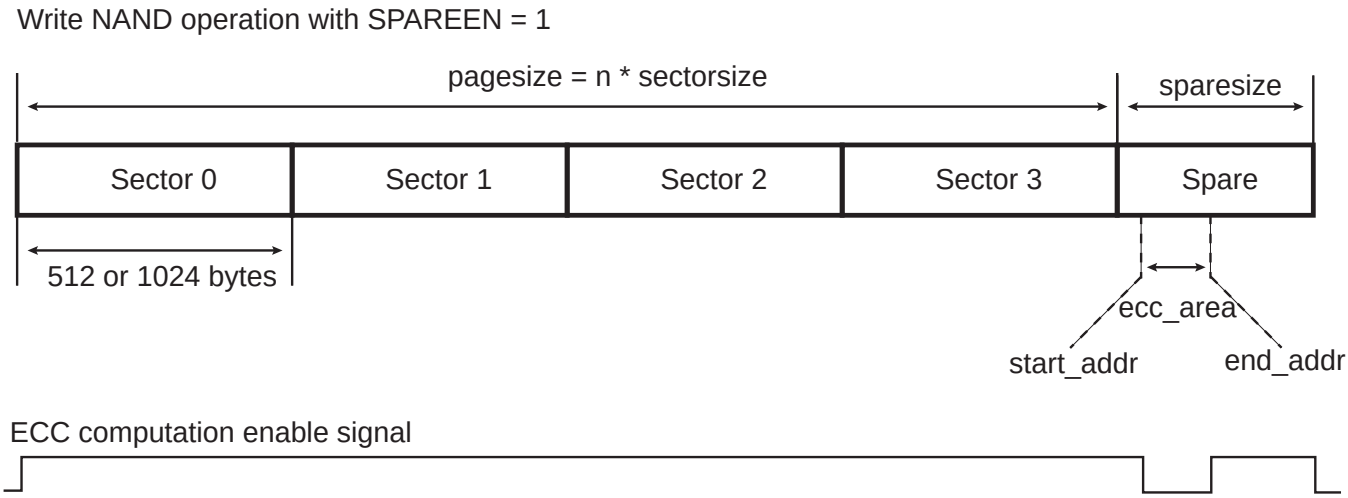
Table 34-15. Number of Relevant ECC Bytes per Sector, Copied from LSByte to MSByte

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
0	4 bytes	4 bytes
1	7 bytes	7 bytes
2	13 bytes	14 bytes
3	20 bytes	21 bytes
4	39 bytes	42 bytes
5	52 bytes	56 bytes

34.18.1.1 SLC/MLC Write Operation with Spare Enable Bit Set

When the SPAREEN bit of the PMECCFG register is set, the spare area of the page is encoded with the stream of data of the last sector of the page. This mode is entered by setting the DATA bit of the PMECCTRL register. When the encoding process is over, the redundancy shall be written to the spare area in User mode. The USER bit of the PMECCTRL register must be set.

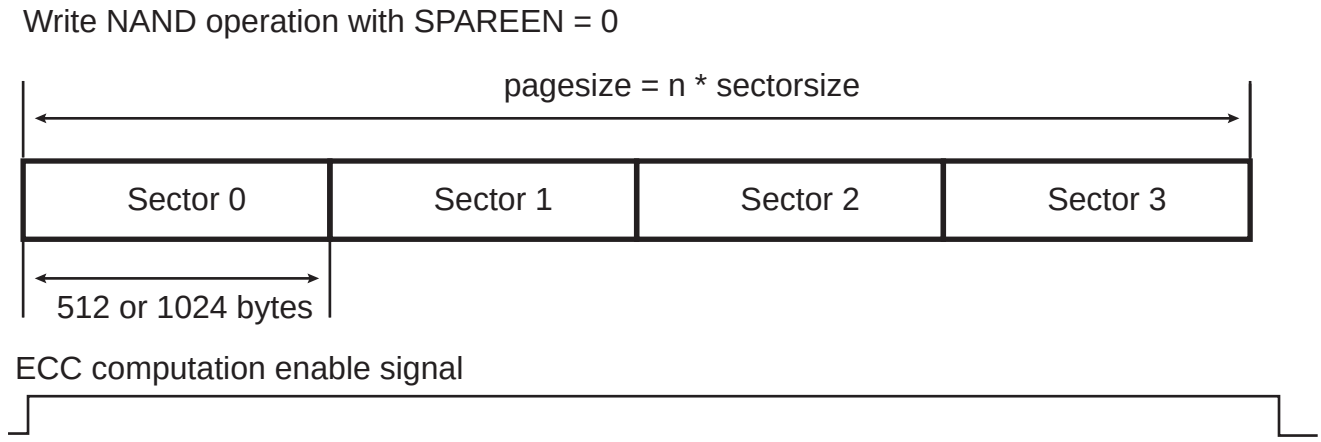
Figure 34-40. NAND Write Operation with Spare Encoding



34.18.1.2 SLC/MLC Write Operation with Spare Disable

When the SPAREEN bit of PMECCFG is cleared, the spare area is not encoded with the stream of data. This mode is entered by setting the DATA bit of the PMECCTRL register.

Figure 34-41. NAND Write Operation



34.18.2 MLC/SLC Read Page Operation Using PMECC

Table 34-16. Relevant Remainder Registers

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
0	PMECCREM0	PMECCREM0
1	PMECCREM0, PMECCREM1	PMECCREM0, PMECCREM1
2	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3,	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3
3	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7

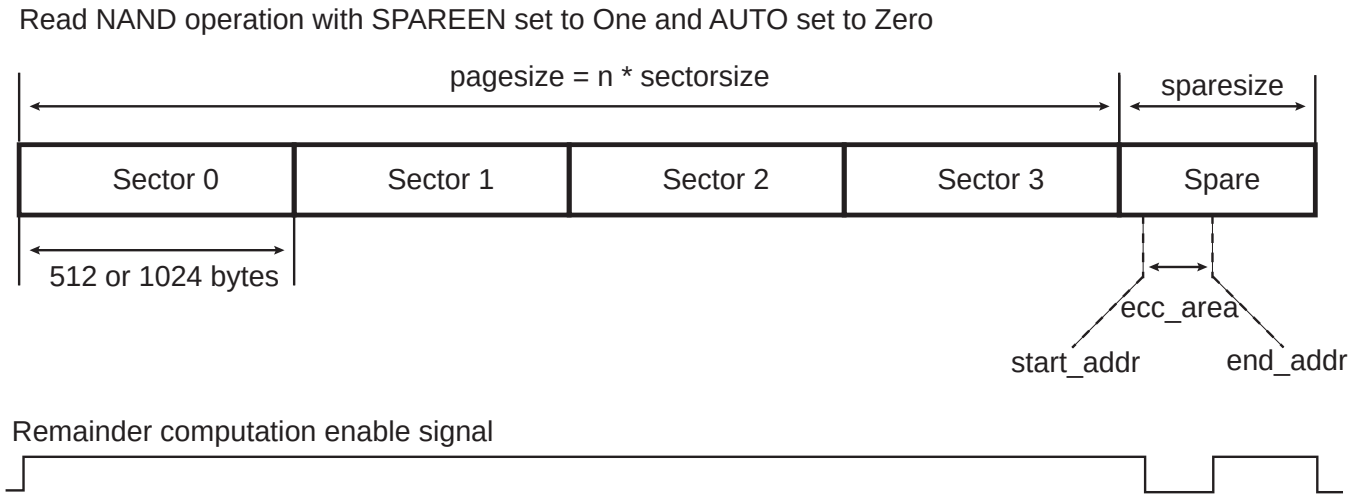
Table 34-16. Relevant Remainder Registers (Continued)

BCH_ERR Field	Sector Size Set to 512 Bytes	Sector Size Set to 1024 Bytes
4	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11
5	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11, PMECCREM12, PMECCREM13, PMECCREM14, PMECCREM15	PMECCREM0, PMECCREM1, PMECCREM2, PMECCREM3, PMECCREM4, PMECCREM5, PMECCREM6, PMECCREM7, PMECCREM8, PMECCREM9, PMECCREM10, PMECCREM11, PMECCREM12, PMECCREM13, PMECCREM14, PMECCREM15

34.18.2.1 MLC/SLC Read Operation with Spare Decoding

When the spare area is protected, it contains valid data. As the redundancy may be included in the middle of the information stream, the user shall program the start address and the end address of the ECC area. The controller will automatically skip the ECC area. This mode is entered writing a 1 in the DATA bit of the PMECCTRL register. When the page has been fully retrieved from the NAND, the ECC area shall be read using the User mode, writing a 1 to the USER bit of the PMECCTRL register.

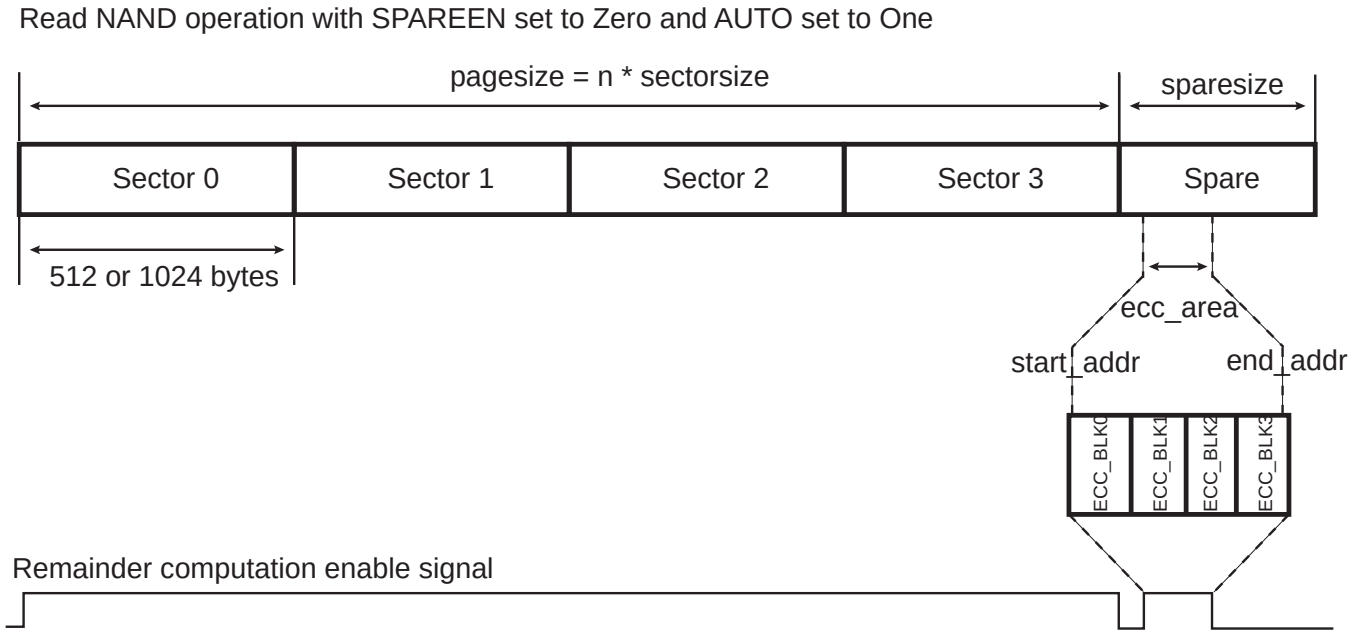
Figure 34-42. Read Operation with Spare Decoding



34.18.2.2 MLC/SLC Read Operation

If the spare area is not protected with the error correcting code, the redundancy area is retrieved directly. This mode is entered writing a 1 in the DATA bit of the PMECCTRL register. When AUTO field is set to one, the ECC is retrieved automatically; otherwise, the ECC must be read using the User mode.

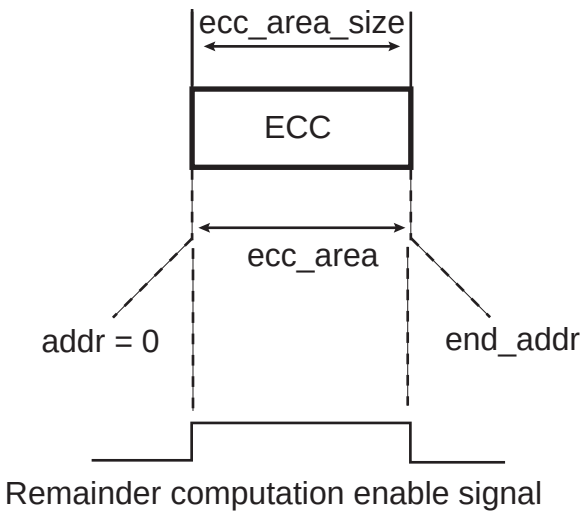
Figure 34-43. Read Operation



34.18.2.3 MLC/SLC User Read ECC Area

This mode allows a manual retrieve of the ECC. It is entered writing a 1 in the USER field of the PMECCTRL register.

Figure 34-44. Read User Mode



34.18.2.4 MLC Controller Working with NFC

Table 34-17. MLC Controller Configuration when the Host Controller is Used

Transfer Type	NFC		PMECC		
	RSPARE	WSPARE	SPAREEN	AUTO	User Mode
Program Page main area is protected, spare is not protected, spare is written manually	0	0	0	0	Not used
Program Page main area is protected, spare is protected, spare is written by NFC	0	1	1	0	Not used
Read Page main area is protected, spare is not protected, spare is not retrieved by NFC	0	0	0	0	Used
Read Page main area is protected, spare is not protected, spare is retrieved by NFC	1	0	0	1	Not used
Read Page main area is protected, spare is protected, spare is retrieved by NFC	1	0	1	0	Used

34.19 Software Implementation

34.19.1 Remainder Substitution Procedure

The substitute function evaluates the remainder polynomial, with different values of the field primitive element. The addition arithmetic operation is performed with the exclusive OR. The multiplication arithmetic operation is performed through the gf_log and gf_antilog lookup tables.

The REM2NP1 and REMN2NP3 fields of the PMECCREMN registers contain only odd remainders. Each bit indicates whether the coefficient of the remainder polynomial is set to zero or not.

NB_ERROR_MAX defines the maximum value of the error correcting capability.

NB_ERROR defines the error correcting capability selected at encoding/decoding time.

NB_FIELD_ELEMENTS defines the number of elements in the field.

si[] is a table that holds the current syndrome value. An element of that table belongs to the field. This is also a shared variable for the next step of the decoding operation.

oo[] is a table that contains the degree of the remainders.

```
int substitute()
{
int i;
int j;
for (i = 1; i < 2 * NB_ERROR_MAX; i++)
{
      si[i] = 0;
}
for (i = 1; i < 2*NB_ERROR; i++)
{
      for (j = 0; j < oo[i]; j++)
      {
            if (REM2NPX[i][j])
            {
```

```
                    si[i] = gf_antilog[(i * j)%NB_FIELD_ELEMENTS] ^ si[i];
                }
            }
        }
        return 0;
        }
```

34.19.2 Finding the Error Location Polynomial Sigma(x)

The sample code below gives a Berlekamp iterative procedure for finding the value of the error location polynomial.

The input of the procedure is the si[] table defined in the remainder substitution procedure.

The output of the procedure is the error location polynomial named smu (sigma mu). The polynomial coefficients belong to the field. The smu[NB_ERROR+1][] is a table that contains all these coefficients.

NB_ERROR_MAX defines the maximum value of the error correcting capability.

NB_ERROR defines the error correcting capability selected at encoding/decoding time.

NB_FIELD_ELEMENTS defines the number of elements in the field.

```
        int get_sigma()
        {
        int i;
        int j;
        int k;
        /* mu          */
        int mu[NB_ERROR_MAX+2];
        /* sigma ro   */
        int sro[2*NB_ERROR_MAX+1];
        /* discrepancy */
        int dmu[NB_ERROR_MAX+2];
        /* delta order   */
        int delta[NB_ERROR_MAX+2];
        /* index of largest delta */
        int ro;
        int largest;
        int diff;
        /*                    */
        /*     First Row      */
        /*                    */
        /* Mu */
        mu[0]  = -1; /* Actually -1/2 */
        /* Sigma(x) set to 1 */
        for (i = 0; i < (2*NB_ERROR_MAX+1); i++)
              smu[0][i] = 0;
        smu[0][0] = 1;
        /* discrepancy set to 1 */
        dmu[0] = 1;
        /* polynom order set to 0 */
        lmu[0] = 0;
        /* delta set to -1 */
        delta[0]  = (mu[0] * 2 - lmu[0]) >> 1;
        /*                     */
        /*     Second Row      */
        /*                     */
        /* Mu */
```

```
mu[1]  = 0;
/* Sigma(x) set to 1 */
for (i = 0; i < (2*NB_ERROR_MAX+1); i++)
      smu[1][i] = 0;
smu[1][0] = 1;
/* discrepancy set to Syndrome 1 */
dmu[1] = si[1];
/* polynom order set to 0 */
lmu[1] = 0;
/* delta set to 0 */
delta[1]  = (mu[1] * 2 - lmu[1]) >> 1;
for (i=1; i <= NB_ERROR; i++)
{
      mu[i+1] = i << 1;
      /*********************************************/
      /*                                           */
      /*                                           */
      /*          Compute Sigma (Mu+1)             */
      /*          And L(mu)                        */
      /* check if discrepancy is set to 0 */
      if (dmu[i] == 0)
      {
      /* copy polynom */
      for (j=0; j<2*NB_ERROR_MAX+1; j++)
      {
            smu[i+1][j] = smu[i][j];
      }
      /* copy previous polynom order to the next */
      lmu[i+1] = lmu[i];
      }
      else
      {
      ro      = 0;
      largest = -1;
      /* find largest delta with dmu != 0 */
      for (j=0; j<i; j++)
            {
            if (dmu[j])
                  {
                  if (delta[j] > largest)
                        {
                        largest = delta[j];
                        ro   = j;
                        }
                  }
            }
      /* initialize signal ro */
      for (k = 0; k < 2*NB_ERROR_MAX+1; k ++)
      {
            sro[k] = 0;
      }
      /* compute difference */
      diff = (mu[i] - mu[ro]);
      /* compute X ^ (2(mu-ro)) */
      for (k = 0; k < (2*NB_ERROR_MAX+1); k ++)
```

```
	{
		sro[k+diff] = smu[ro][k];
	}
	/* multiply by dmu * dmu[ro]^-1 */
	for (k = 0; k < 2*NB_ERROR_MAX+1; k ++)
	{
		/* dmu[ro] is not equal to zero by definition */
		/* check that operand are different from 0    */
		if (sro[k] && dmu[i])
		{
			/* galois inverse */
			sro[k] = gf_antilog[(gf_log[dmu[i]] + (NB_FIELD_ELEMENTS-
gf_log[dmu[ro]]) + gf_log[sro[k]]) % NB_FIELD_ELEMENTS];
		}
	}
	/* multiply by dmu * dmu[ro]^-1 */
	for (k = 0; k < 2*NB_ERROR_MAX+1; k++)
	{
		smu[i+1][k] = smu[i][k] ^ sro[k];
		if (smu[i+1][k])
		{
			/* find the order of the polynom */
			lmu[i+1] = k << 1;
		}
	}
}
/*                                                */
/*                                                */
/*      End Compute Sigma (Mu+1)                  */
/*      And L(mu)                                 */
/************************************************/
/* In either case compute delta */
delta[i+1]  = (mu[i+1] * 2 - lmu[i+1]) >> 1;
/* In either case compute the discrepancy */
for (k = 0 ; k <= (lmu[i+1]>>1); k++)
{
	if (k == 0)
	dmu[i+1] = si[2*(i-1)+3];
	/* check if one operand of the multiplier is null, its index is -1 */
	else if (smu[i+1][k] && si[2*(i-1)+3-k])
	dmu[i+1] = gf_antilog[(gf_log[smu[i+1][k]] + gf_log[si[2*(i-1)+3-
k]])%nn] ^ dmu[i+1];
}
}
return 0;
}
```

34.19.3 Finding the Error Position

The output of the get_sigma() procedure is a polynomial stored in the smu[NB_ERROR+1][] table. The error positions are the roots of that polynomial. The degree of that polynomial is a very important information, as it gives the number of errors. PMERRLOC module provides hardware accelerator for that step.

34.19.3.1 Error Location

The PMECC Error Location controller provides hardware acceleration for determining roots of polynomials over two finite fields: GF(2^13) and GF(2^14). It integrates 32 fully programmable coefficients. These coefficients belong to GF(2^13) or GF(2^14). The coefficient programmed in the PMERRLOC{*i*} is the coefficient of X ^ *i* in the polynomial.

The search operation is started as soon as a write access is detected in the ELEN register and can be disabled writing to the ELDIS register. The ENINIT field of the ELEN register shall be initialized with the number of galois field elements to test. The set of the roots can be limited to a valid range.

Table 34-18. ENINIT Field Value for a Sector Size of 512 Bytes

Error Correcting Capability	ENINIT Value
2	4122
4	4148
8	4200
12	4252
24	4408
32	4512

Table 34-19. ENINIT Field Value for a Sector Size of 1024 Bytes

Error Correcting Capability	ENINIT Value
2	8220
4	8248
8	8304
12	8360
24	8528
32	8640

When the PMECC engine is searching for roots, the BUSY field of the ELSR register remains asserted. An interrupt is asserted at the end of the computation, and the DONE bit of the PMECC Error Location Interrupt Status Register (HSMC_ELSIR) is set. The ERR_CNT field of the HSMC_ELISR indicates the number of errors. The error position can be read in the PMERRLOCX registers.

34.20 Static Memory Controller (SMC) User Interface

The SMC is programmed using the registers listed in Table 34-20. For each chip select, a set of four registers is used to program the parameters of the external device. In Table 34-20, “CS_number” denotes the chip select number. Sixteen bytes per chip select are required.

Table 34-20. Register Mapping

Offset	Register	Name	Access	Reset
0x000	NFC Configuration Register	HSMC_CFG	Read/Write	0x0
0x004	NFC Control Register	HSMC_CTRL	Write-only	–
0x008	NFC Status Register	HSMC_SR	Read-only	0x0
0x00C	NFC Interrupt Enable Register	HSMC_IER	Write-only	–
0x010	NFC Interrupt Disable Register	HSMC_IDR	Write-only	–
0x014	NFC Interrupt Mask Register	HSMC_IMR	Read-only	0x0
0x018	NFC Address Cycle Zero Register	HSMC_ADDR	Read/Write	0x0
0x01C	Bank Address Register	HSMC_BANK	Read/Write	0x0
0x020–0x06C	Reserved	–	–	–
0x070	PMECC Configuration Register	HSMC_PMECCFG	Read/Write	0x0
0x074	PMECC Spare Area Size Register	HSMC_PMECCSAREA	Read/Write	0x0
0x078	PMECC Start Address Register	HSMC_PMECCSADDR	Read/Write	0x0
0x07C	PMECC End Address Register	HSMC_PMECCEADDR	Read/Write	0x0
0x080	Reserved	–	–	–
0x084	PMECC Control Register	HSMC_PMECCTRL	Write-only	–
0x088	PMECC Status Register	HSMC_PMECCSR	Read-only	0x0
0x08C	PMECC Interrupt Enable register	HSMC_PMECCIER	Write-only	–
0x090	PMECC Interrupt Disable Register	HSMC_PMECCIDR	Write-only	–
0x094	PMECC Interrupt Mask Register	HSMC_PMECCIMR	Read-only	0x0
0x098	PMECC Interrupt Status Register	HSMC_PMECCISR	Read-only	0x0
0x09C–0x0AC	Reserved	–	–	–
0x0B0+sec_num*(0x40)+0x00	PMECC Redundancy 0 Register	HSMC_PMECC0	Read-only	0x0
0x0B0+sec_num*(0x40)+0x04	PMECC Redundancy 1 Register	HSMC_PMECC1	Read-only	0x0
0x0B0+sec_num*(0x40)+0x08	PMECC Redundancy 2 Register	HSMC_PMECC2	Read-only	0x0
0x0B0+sec_num*(0x40)+0x0C	PMECC Redundancy 3 Register	HSMC_PMECC3	Read-only	0x0
0x0B0+sec_num*(0x40)+0x10	PMECC Redundancy 4 Register	HSMC_PMECC4	Read-only	0x0
0x0B0+sec_num*(0x40)+0x14	PMECC Redundancy 5 Register	HSMC_PMECC5	Read-only	0x0
0x0B0+sec_num*(0x40)+0x18	PMECC Redundancy 6 Register	HSMC_PMECC6	Read-only	0x0
0x0B0+sec_num*(0x40)+0x1C	PMECC Redundancy 7 Register	HSMC_PMECC7	Read-only	0x0
0x0B0+sec_num*(0x40)+0x20	PMECC Redundancy 8 Register	HSMC_PMECC8	Read-only	0x0
0x0B0+sec_num*(0x40)+0x24	PMECC Redundancy 9 Register	HSMC_PMECC9	Read-only	0x0
0x0B0+sec_num*(0x40)+0x28	PMECC Redundancy 10 Register	HSMC_PMECC10	Read-only	0x0
0x0B0+sec_num*(0x40)+0x2C	PMECC Redundancy 11 Register	HSMC_PMECC11	Read-only	0x0

Table 34-20. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0B0+sec_num*(0x40)+0x30	PMECC Redundancy 12 Register	HSMC_PMECC12	Read-only	0x0
0x0B0+sec_num*(0x40)+0x34	PMECC Redundancy 13 Register	HSMC_PMECC13	Read-only	0x0
0x2B0+sec_num*(0x40)+0x00	PMECC Remainder 0 Register	HSMC_REM0	Read-only	0x0
0x2B0+sec_num*(0x40)+0x04	PMECC Remainder 1 Register	HSMC_REM1	Read-only	0x0
0x2B0+sec_num*(0x40)+0x08	PMECC Remainder 2 Register	HSMC_REM2	Read-only	0x0
0x2B0+sec_num*(0x40)+0x0C	PMECC Remainder 3 Register	HSMC_REM3	Read-only	0x0
0x2B0+sec_num*(0x40)+0x10	PMECC Remainder 4 Register	HSMC_REM4	Read-only	0x0
0x2B0+sec_num*(0x40)+0x14	PMECC Remainder 5 Register	HSMC_REM5	Read-only	0x0
0x2B0+sec_num*(0x40)+0x18	PMECC Remainder 6 Register	HSMC_REM6	Read-only	0x0
0x2B0+sec_num*(0x40)+0x1C	PMECC Remainder 7 Register	HSMC_REM7	Read-only	0x0
0x2B0+sec_num*(0x40)+0x20	PMECC Remainder 8 Register	HSMC_REM8	Read-only	0x0
0x2B0+sec_num*(0x40)+0x24	PMECC Remainder 9 Register	HSMC_REM9	Read-only	0x0
0x2B0+sec_num*(0x40)+0x28	PMECC Remainder 10 Register	HSMC_REM10	Read-only	0x0
0x2B0+sec_num*(0x40)+0x2C	PMECC Remainder 11 Register	HSMC_REM11	Read-only	0x0
0x2B0+sec_num*(0x40)+0x30	PMECC Remainder 12 Register	HSMC_REM12	Read-only	0x0
0x2B0+sec_num*(0x40)+0x34	PMECC Remainder 13 Register	HSMC_REM13	Read-only	0x0
0x2B0+sec_num*(0x40)+0x38	PMECC Remainder 14 Register	HSMC_REM14	Read-only	0x0
0x2B0+sec_num*(0x40)+0x3C	PMECC Remainder 15 Register	HSMC_REM15	Read-only	0x0
0x4A0–0x4FC	Reserved	–	–	–
0x500	PMECC Error Location Configuration Register	HSMC_ELCFG	Read/Write	0x0
0x504	PMECC Error Location Primitive Register	HSMC_ELPRIM	Read-only	0x401A
0x508	PMECC Error Location Enable Register	HSMC_ELEN	Write-only	–
0x50C	PMECC Error Location Disable Register	HSMC_ELDIS	Write-only	–
0x510	PMECC Error Location Status Register	HSMC_ELSR	Read-only	0x0
0x514	PMECC Error Location Interrupt Enable register	HSMC_ELIER	Write-only	–
0x518	PMECC Error Location Interrupt Disable Register	HSMC_ELIDR	Write-only	–
0x51C	PMECC Error Location Interrupt Mask Register	HSMC_ELIMR	Read-only	0x0
0x520	PMECC Error Location Interrupt Status Register	HSMC_ELISR	Read-only	0x0
0x524	Reserved	–	–	–
0x528	PMECC Error Location SIGMA 0 Register	HSMC_SIGMA0	Read-only	0x1
0x52C	PMECC Error Location SIGMA 1 Register	HSMC_SIGMA1	Read/Write	0x0

Table 34-20. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x530	PMECC Error Location SIGMA 2 Register	HSMC_SIGMA2	Read/Write	0x0
0x534	PMECC Error Location SIGMA 3 Register	HSMC_SIGMA3	Read/Write	0x0
0x538	PMECC Error Location SIGMA 4 Register	HSMC_SIGMA4	Read/Write	0x0
0x53C	PMECC Error Location SIGMA 5 Register	HSMC_SIGMA5	Read/Write	0x0
0x540	PMECC Error Location SIGMA 6 Register	HSMC_SIGMA6	Read/Write	0x0
0x544	PMECC Error Location SIGMA 7 Register	HSMC_SIGMA7	Read/Write	0x0
0x548	PMECC Error Location SIGMA 8 Register	HSMC_SIGMA8	Read/Write	0x0
0x54C	PMECC Error Location SIGMA 9 Register	HSMC_SIGMA9	Read/Write	0x0
0x550	PMECC Error Location SIGMA 10 Register	HSMC_SIGMA10	Read/Write	0x0
0x554	PMECC Error Location SIGMA 11 Register	HSMC_SIGMA11	Read/Write	0x0
0x558	PMECC Error Location SIGMA 12 Register	HSMC_SIGMA12	Read/Write	0x0
0x55C	PMECC Error Location SIGMA 13 Register	HSMC_SIGMA13	Read/Write	0x0
0x560	PMECC Error Location SIGMA 14 Register	HSMC_SIGMA14	Read/Write	0x0
0x564	PMECC Error Location SIGMA 15 Register	HSMC_SIGMA15	Read/Write	0x0
0x568	PMECC Error Location SIGMA 16 Register	HSMC_SIGMA16	Read/Write	0x0
0x56C	PMECC Error Location SIGMA 17 Register	HSMC_SIGMA17	Read/Write	0x0
0x570	PMECC Error Location SIGMA 18 Register	HSMC_SIGMA18	Read/Write	0x0
0x574	PMECC Error Location SIGMA 19 Register	HSMC_SIGMA19	Read/Write	0x0
0x578	PMECC Error Location SIGMA 20 Register	HSMC_SIGMA20	Read/Write	0x0
0x57C	PMECC Error Location SIGMA 21 Register	HSMC_SIGMA21	Read/Write	0x0
0x580	PMECC Error Location SIGMA 22 Register	HSMC_SIGMA22	Read/Write	0x0
0x584	PMECC Error Location SIGMA 23 Register	HSMC_SIGMA23	Read/Write	0x0

Table 34-20. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x588	PMECC Error Location SIGMA 24 Register	HSMC_SIGMA24	Read/Write	0x0
0x58C	PMECC Error Location SIGMA 25 Register	HSMC_SIGMA25	Read/Write	0x0
0x590	PMECC Error Location SIGMA 26 Register	HSMC_SIGMA26	Read/Write	0x0
0x594	PMECC Error Location SIGMA 27 Register	HSMC_SIGMA27	Read/Write	0x0
0x598	PMECC Error Location SIGMA 28 Register	HSMC_SIGMA28	Read/Write	0x0
0x59C	PMECC Error Location SIGMA 29 Register	HSMC_SIGMA29	Read/Write	0x0
0x5A0	PMECC Error Location SIGMA 30 Register	HSMC_SIGMA30	Read/Write	0x0
0x5A4	PMECC Error Location SIGMA 31 Register	HSMC_SIGMA31	Read/Write	0x0
0x5A8	PMECC Error Location SIGMA 32 Register	HSMC_SIGMA32	Read/Write	0x0
0x5AC	PMECC Error Location 0 Register	HSMC_ERRLOC0	Read-only	0x0
...	...	...	...	...
0x628	PMECC Error Location 31 Register	HSMC_ERRLOC31	Read-only	0x0
0x62C–0x6FC	Reserved	–	–	–
0x14*CS_number+0x700	Setup Register	HSMC_SETUP	Read/Write	0x0101_0101
0x14*CS_number+0x704	Pulse Register	HSMC_PULSE	Read/Write	0x0101_0101
0x14*CS_number+0x708	Cycle Register	HSMC_CYCLE	Read/Write	0x0003_0003
0x14*CS_number+0x70C	Timings Register	HSMC_TIMINGS	Read/Write	0x0000_0000
0x14*CS_number+0x710	Mode Register	HSMC_MODE	Read/Write	0x0000_1003
0x7A0	Off Chip Memory Scrambling Register	HSMC_OCMS	Read/Write	0x0
0x7A4	Off Chip Memory Scrambling KEY1 Register	HSMC_KEY1	Write-once	0x0
0x7A8	Off Chip Memory Scrambling KEY2 Register	HSMC_KEY2	Write-once	0x0
0x7AC–0x7E0	Reserved	–	–	–
0x7E4	Write Protection Mode Register	HSMC_WPMR	Read/Write	0x0
0x7E8	Write Protection Status Register	HSMC_WPSR	Read-only	0x0
0x7EC–0x7FC	Reserved	–	–	–

34.20.1 NFC Configuration Register

Name: HSMC_CFG

Address: 0xF8014000

Access: Read/Write

31	30	29	28	27	26	25	24
–	NFCSPARESIZE						

23	22	21	20	19	18	17	16
–	DTOMUL			DTOCYC			

15	14	13	12	11	10	9	8
–	–	RBEDGE	EDGECTRL	–	–	RSPARE	WSPARE

7	6	5	4	3	2	1	0
–	–	–	–	–	PAGESIZE		

- **PAGESIZE: Page Size of the NAND Flash Device**

Value	Name	Description
0	PS512	Main area 512 bytes
1	PS1024	Main area 1024 bytes
2	PS2048	Main area 2048 bytes
3	PS4096	Main area 4096 bytes
4	PS8192	Main area 8192 bytes

- **WSPARE: Write Spare Area**

0: The NFC skips the spare area in Write mode.

1: The NFC writes both main area and spare area in Write mode.

- **RSPARE: Read Spare Area**

0: The NFC skips the spare area in Read mode.

1: The NFC reads both main area and spare area in Read mode.

- **EDGECTRL: Rising/Falling Edge Detection Control**

0: Rising edge is detected

1: Falling edge is detected

- **RBEDGE: Ready/Busy Signal Edge Detection**

0: When configured to zero, RB_EDGE fields indicate the level of the Ready/Busy lines.

1: When set to one, RB_EDGE fields indicate only transition on Ready/Busy lines.

- **DTOCYC: Data Timeout Cycle Number**

- **DTOMUL: Data Timeout Multiplier**

These fields determine the maximum number of Master Clock cycles that the SMC waits until the detection of a rising edge on Ready/Busy signal.

Data Timeout Multiplier is defined by DTOMUL as shown in the following table:

Value	Name	Description
0	X1	DTOCYC
1	X16	DTOCYC x 16
2	X128	DTOCYC x 128
3	X256	DTOCYC x 256
4	X1024	DTOCYC x 1024
5	X4096	DTOCYC x 4096
6	X65536	DTOCYC x 65536
7	X1048576	DTOCYC x 1048576

If the data timeout set by DTOCYC and DTOMUL has been exceeded, the Data Timeout Error flag (DTOE) in the NFC Status Register (NFC_SR) raises.

- **NFCSPARESIZE: NAND Flash Spare Area Size Retrieved by the Host Controller**

The spare size is set to (NFCSPARESIZE + 1) * 4 bytes. The spare area is only retrieved when RSPARE or WSPARE is activated.

34.20.2 NFC Control Register

Name: HSMC_CTRL

Address: 0xF8014004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	NFCDIS	NFCEN

- **NFCEN: NAND Flash Controller Enable**

0: No effect

1: Enable the NAND Flash controller.

- **NFCDIS: NAND Flash Controller Disable**

0: No effect

1: Disable the NAND Flash controller.

34.20.3 NFC Status Register

Name: HSMC_SR

Address: 0xF8014008

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	NFCSID			NFCWR	–	–	NFCBUSY
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	SMCSTS

- **SMCSTS: NAND Flash Controller Status (this field cannot be reset)**

0: NAND Flash Controller disabled

1: NAND Flash Controller enabled

- **RB_RISE: Selected Ready Busy Rising Edge Detected**

When set to one, this flag indicates that a rising edge on the Ready/Busy Line has been detected. This flag is reset after a status read operation. The Ready/Busy line is selected through the decoding of field HSMC_SR.NFCSID.

- **RB_FALL: Selected Ready Busy Falling Edge Detected**

When set to one, this flag indicates that a falling edge on the Ready/Busy Line has been detected. This flag is reset after a status read operation. The Ready/Busy line is selected through the decoding of field HSMC_SR.NFCSID.

- **NFCBUSY: NFC Busy (this field cannot be reset)**

When set to one, this flag indicates that the Controller is activated and accesses the memory device.

- **NFCWR: NFC Write/Read Operation (this field cannot be reset)**

When a command is issued, this field indicates the current Read or Write Operation.

- **NFCSID: NFC Chip Select ID (this field cannot be reset)**

When a command is issued, this field indicates the value of the targeted chip select.

- **XFRDONE: NFC Data Transfer Terminated**

When set to one, this flag indicates that the NFC has terminated the Data Transfer. This flag is reset after a status read operation.

- **CMDDONE: Command Done**

When set to one, this flag indicates that the NFC has terminated the Command. This flag is reset after a status read operation.

- **DTOE: Data Timeout Error**

When set to one, this flag indicates that the Data timeout set be by DTOMUL and DTOCYC has been exceeded. This flag is reset after a status read operation.

- **UNDEF: Undefined Area Error**

When set to one, this flag indicates that the processor performed an access in an undefined memory area. This flag is reset after a status read operation.

- **AWB: Accessing While Busy**

If set to one, this flag indicates that an AHB master has performed an access during the busy phase. This flag is reset after a status read operation.

- **NFCASE: NFC Access Size Error**

If set to one, this flag indicates that an illegal access has been detected in the NFC Memory Area. Only Word Access is allowed within the NFC memory area. This flag is reset after a status read operation.

- **RB_EDGEx: Ready/Busy Line x Edge Detected**

If set to one, this flag indicates that an edge has been detected on the Ready/Busy Line x. Depending on the EDGE CTRL field located in the HSMC_CFG register, only rising or falling edge is detected. This flag is reset after a status read operation.

34.20.4 NFC Interrupt Enable Register

Name: HSMC_IER

Address: 0xF801400C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	–

- **RB_RISE: Ready Busy Rising Edge Detection Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **RB_FALL: Ready Busy Falling Edge Detection Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **XFRDONE: Transfer Done Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **CMDDONE: Command Done Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **DTOE: Data Timeout Error Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **UNDEF: Undefined Area Access Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **AWB: Accessing While Busy Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **NFCASE: NFC Access Size Error Interrupt Enable**

0: No effect

1: Interrupt source enabled

- **RB_EDGEx: Ready/Busy Line x Interrupt Enable**

0: No effect

1: Interrupt source enabled

34.20.5 NFC Interrupt Disable Register

Name: HSMC_IDR

Address: 0xF8014010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	–

- **RB_RISE: Ready Busy Rising Edge Detection Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **RB_FALL: Ready Busy Falling Edge Detection Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **XFRDONE: Transfer Done Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **CMDDONE: Command Done Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **DTOE: Data Timeout Error Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **UNDEF: Undefined Area Access Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **AWB: Accessing While Busy Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **NFCASE: NFC Access Size Error Interrupt Disable**

0: No effect

1: Interrupt source disabled

- **RB_EDGEx: Ready/Busy Line x Interrupt Disable**

0: No effect

1: Interrupt source disabled

34.20.6 NFC Interrupt Mask Register

Name: HSMC_IMR

Address: 0xF8014014

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	RB_EDGE0
23	22	21	20	19	18	17	16
NFCASE	AWB	UNDEF	DTOE	–	–	CMDDONE	XFRDONE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RB_FALL	RB_RISE	–	–	–	–

- **RB_RISE: Ready Busy Rising Edge Detection Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **RB_FALL: Ready Busy Falling Edge Detection Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **XFRDONE: Transfer Done Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **CMDDONE: Command Done Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **DTOE: Data Timeout Error Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **UNDEF: Undefined Area Access Interrupt Mask5**

0: Interrupt source disabled

1: Interrupt source enabled

- **AWB: Accessing While Busy Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **NFCASE: NFC Access Size Error Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

- **RB_EDGEx: Ready/Busy Line x Interrupt Mask**

0: Interrupt source disabled

1: Interrupt source enabled

34.20.7 NFC Address Cycle Zero Register

Name: HSMC_ADDR

Address: 0xF8014018

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ADDR_CYCLE0							

- **ADDR_CYCLE0: NAND Flash Array Address Cycle 0**

When five address cycles are used, ADDR_CYCLE0 is the first byte written to the NAND Flash (used by the NFC).

34.20.8 NFC Bank Register

Name: HSMC_BANK

Address: 0xF801401C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	BANK

- **BANK: Bank Identifier**

0: Bank 0 is used.

1: Bank 1 is used.

34.20.9 PMECC Configuration Register

Name: HSMC_PMECCFG

Address: 0xF8014070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	AUTO	–	–	–	SPAREEN

15	14	13	12	11	10	9	8
–	–	–	NANDWR	–	–	PAGESIZE	

7	6	5	4	3	2	1	0
–	–	–	SECTORSZ	–	BCH_ERR		

- **BCH_ERR: Error Correcting Capability**

Value	Name	Description
0	BCH_ERR2	2 errors
1	BCH_ERR4	4 errors
2	BCH_ERR8	8 errors
3	BCH_ERR12	12 errors
4	BCH_ERR24	24 errors
5	BCH_ERR32	32 errors

- **SECTORSZ: Sector Size**

0: The ECC computation is based on a sector of 512 bytes.

1: The ECC computation is based on a sector of 1024 bytes.

- **PAGESIZE: Number of Sectors in the Page**

Value	Name	Description
0	PAGESIZE_1SEC	1 sector for main area (512 or 1024 bytes)
1	PAGESIZE_2SEC	2 sectors for main area (1024 or 2048 bytes)
2	PAGESIZE_4SEC	4 sectors for main area (2048 or 4096 bytes)
3	PAGESIZE_8SEC	8 sectors for main area (4096 or 8192 bytes)

- **NANDWR: NAND Write Access**

0: NAND read access

1: NAND write access

- **SPAREEN: Spare Enable**
  - for NAND write access:

0: The spare area is skipped

1: The spare area is protected with the last sector of data.

  - for NAND read access:

0: The spare area is skipped.

1: The spare area contains protected data or only redundancy information.

- **AUTO: Automatic Mode Enable**

This bit is only relevant in NAND Read Mode, when spare enable is activated.

0: Indicates that the spare area is not protected. In that case, the ECC computation takes into account the ECC area located in the spare area. (within the start address and the end address).

1: Indicates that the spare area is error protected. In this case, the ECC computation takes into account the whole spare area minus the ECC area in the ECC computation operation.

34.20.10 PMECC Spare Area Size Register

Name: HSMC_PMECCSAREA

Address: 0xF8014074

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SPARESIZE
7	6	5	4	3	2	1	0
SPARESIZE							

- **SPARESIZE: Spare Area Size**

Number of bytes in the spare area. The spare area size is equal to (SPARESIZE + 1) bytes.

34.20.11 PMECC Start Address Register

Name: HSMC_PMECCSADDR

Address: 0xF8014078

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	STARTADDR
7	6	5	4	3	2	1	0
STARTADDR							

- **STARTADDR: ECC Area Start Address**

This register is programmed with the start ECC start address. When STARTADDR is equal to 0, then the first ECC byte is located at the first byte of the spare area.

34.20.12 PMECC End Address Register

Name: HSMC_PMECCEADDR

Address: 0xF801407C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	ENDADDR

7	6	5	4	3	2	1	0
ENDADDR							

- **ENDADDR: ECC Area End Address**

This register is programmed with the start ECC end address. When ENDADDR is equal to *N*, then the first ECC byte is located at byte *N* of the spare area.

34.20.13 PMECC Control Register

Name: HSMC_PMECCTRL

Address: 0xF8014084

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	DISABLE	ENABLE	–	USER	DATA	RST

- **RST: Reset the PMECC Module**

0: No effect

1: Reset the PMECC controller.

- **DATA: Start a Data Phase**

0: No effect

1: The PMECC controller enters a Data phase.

- **USER: Start a User Mode Phase**

0: No effect

1: The PMECC controller enters a User mode phase.

- **ENABLE: PMECC Enable**

0: No effect

1: Enable the PMECC controller.

- **DISABLE: PMECC Enable**

0: No effect

1: Disable the PMECC controller.

34.20.14 PMECC Status Register

Name: HSMC_PMECCSR

Address: 0xF8014088

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	ENABLE	–	–	–	BUSY

- **BUSY: The kernel of the PMECC is busy**

0: PMECC controller finite state machine reached idle state

1: PMECC controller finite state machine is processing the incoming byte stream

- **ENABLE: PMECC Enable bit**

0: PMECC controller disabled

1: PMECC controller enabled

34.20.15 PMECC Interrupt Enable Register

Name: HSMC_PMECCIER

Address: 0xF801408C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ERRIE

- **ERRIE: Error Interrupt Enable**

0: No effect

1: The Multibit Error interrupt is enabled. An interrupt will be raised if at least one error is detected in at least one sector.

34.20.16 PMECC Interrupt Disable Register

Name: HSMC_PMECCIDR

Address: 0xF8014090

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ERRID

- **ERRID: Error Interrupt Disable**

0: No effect

1: Multibit Error interrupt disabled

34.20.17 PMECC Interrupt Mask Register

Name: HSMC_PMECCIMR

Address: 0xF8014094

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ERRIM

- **ERRIM: Error Interrupt Mask**

0: Multibit Error disabled

1: Multibit Error enabled

34.20.18 PMECC Interrupt Status Register

Name: HSMC_PMECCISR

Address: 0xF8014098

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
ERRIS							

- **ERRIS: Error Interrupt Status Register**

When set to one, bit *i* of the HSMC_PMECCISR indicates that sector *i* is corrupted.

34.20.19 PMECC Redundancy x Register

Name: HSMC_PMECCx [x=0..13] [sec_num=0..7]

Address: 0xF80140B0 [0][0] .. 0xF80140E4 [13][0]
0xF80140F0 [0][1] .. 0xF8014124 [13][1]
0xF8014130 [0][2] .. 0xF8014164 [13][2]
0xF8014170 [0][3] .. 0xF80141A4 [13][3]
0xF80141B0 [0][4] .. 0xF80141E4 [13][4]
0xF80141F0 [0][5] .. 0xF8014224 [13][5]
0xF8014230 [0][6] .. 0xF8014264 [13][6]
0xF8014270 [0][7] .. 0xF80142A4 [13][7]

Access: Read-only

31	30	29	28	27	26	25	24
ECC							

23	22	21	20	19	18	17	16
ECC							

15	14	13	12	11	10	9	8
ECC							

7	6	5	4	3	2	1	0
ECC							

- **ECC: BCH Redundancy**

This register contains the remainder of the division of the codeword by the generator polynomial.

34.20.20 PMECC Remainder x Register

Name: HSMC_REMx [x=0..15] [sec_num=0..7]

Address: 0xF80142B0 [0][0] .. 0xF80142EC [15][0]
0xF80142F0 [0][1] .. 0xF801432C [15][1]
0xF8014330 [0][2] .. 0xF801436C [15][2]
0xF8014370 [0][3] .. 0xF80143AC [15][3]
0xF80143B0 [0][4] .. 0xF80143EC [15][4]
0xF80143F0 [0][5] .. 0xF801442C [15][5]
0xF8014430 [0][6] .. 0xF801446C [15][6]
0xF8014470 [0][7] .. 0xF80144AC [15][7]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	REM2NP3					

23	22	21	20	19	18	17	16
REM2NP3							

15	14	13	12	11	10	9	8
–	–	REM2NP1					

7	6	5	4	3	2	1	0
REM2NP1							

- **REM2NP1: BCH Remainder 2 * N + 1**

When sector size is set to 512 bytes, bit REM2NP1[13] is not used and read as zero.

If bit *i* of the REM2NP1 field is set to one, then the coefficient of the X ^ *i* is set to one; otherwise, the coefficient is zero.

- **REM2NP3: BCH Remainder 2 * N + 3**

When sector size is set to 512 bytes, bit REM2NP3[29] is not used and read as zero.

If bit *i* of the REM2NP3 field is set to one, then the coefficient of the X ^ *i* is set to one; otherwise, the coefficient is zero.

34.20.21 PMECC Error Location Configuration Register

Name: HSMC_ELCFG

Address: 0xF8014500

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	ERRNUM				

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SECTORSZ

- **ERRNUM: Number of Errors**
- **SECTORSZ: Sector Size**

0: The ECC computation is based on a 512 bytes sector.

1: The ECC computation is based on a 1024 bytes sector.

34.20.22 PMECC Error Location Primitive Register

Name: HSMC_ELPRIM

Address: 0xF8014504

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PRIMITIV							

7	6	5	4	3	2	1	0
PRIMITIV							

- **PRIMITIV: Primitive Polynomial**

This field indicates the Primitive Polynomial used in the ECC computation.

34.20.23 PMECC Error Location Enable Register

Name: HSMC_ELEN

Address: 0xF8014508

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	ENINIT					

7	6	5	4	3	2	1	0
ENINIT							

- **ENINIT: Error Location Enable**

Initial bit number in the codeword.

34.20.24 PMECC Error Location Disable Register

Name: HSMC_ELDIS

Address: 0xF801450C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DIS

- **DIS: Disable Error Location Engine**

0: No effect

1: Disable the Error location engine.

34.20.25 PMECC Error Location Status Register

Name: HSMC_ELSR

Address: 0xF8014510

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	BUSY

- **BUSY: Error Location Engine Busy**

0: Error location engine is disabled.

1: Error location engine is enabled and is finding roots of the polynomial.

34.20.26 PMECC Error Location Interrupt Enable Register

Name: HSMC_ELIER

Address: 0xF8014514

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Enable**

0: No effect

1: Interrupt Enable.

34.20.27 PMECC Error Location Interrupt Disable Register

Name: HSMC_ELIDR

Address: 0xF8014518

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Disable**

0: No effect

1: Interrupt disable.

34.20.28 PMECC Error Location Interrupt Mask Register

Name: HSMC_ELIMR

Address: 0xF801451C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Mask**

0: Computation Terminated interrupt disabled

1: Computation Terminated interrupt enabled

34.20.29 PMECC Error Location Interrupt Status Register

Name: HSMC_ELISR

Address: 0xF8014520

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	ERR_CNT					
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DONE

- **DONE: Computation Terminated Interrupt Status**

When set to one, this indicates that the error location engine has completed the root finding algorithm.

- **ERR_CNT: Error Counter value**

This field indicates the number of roots of the polynomial.

34.20.30 PMECC Error Location SIGMA0 Register

Name: HSMC_SIGMA0

Address: 0xF8014528

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	SIGMA0					

7	6	5	4	3	2	1	0
SIGMA0							

- **SIGMA0: Coefficient of degree 0 in the SIGMA polynomial**

SIGMA0 belongs to the finite field GF(2^13) when the sector size is set to 512 bytes.

SIGMA0 belongs to the finite field GF(2^14) when the sector size is set to 1024 bytes.

34.20.31 PMECC Error Location SIGMAx Register

Name: HSMC_SIGMAx [x=1..32]

Address: 0xF801452C [1] .. 0xF80145A8 [32]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	SIGMAx					

7	6	5	4	3	2	1	0
SIGMAx							

- **SIGMAx: Coefficient of degree x in the SIGMA polynomial**

SIGMAx belongs to the finite field GF(2^13) when the sector size is set to 512 bytes.

SIGMAx belongs to the finite field GF(2^14) when the sector size is set to 1024 bytes.

34.20.32 PMECC Error Location x Register

Name: HSMC_ERRLOCx [x=0..31]

Address: 0xF80145AC [0] .. 0xF8014628 [31]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	ERRLOCN					

7	6	5	4	3	2	1	0
ERRLOCN							

- **ERRLOCN: Error Position within the Set {sector area, spare area}**

ERRLOCN points to 1 when the first bit of the main area is corrupted.

If the sector size is set to 512 bytes, the ERRLOCN points to 4096 when the last bit of the sector area is corrupted.

If the sector size is set to 1024 bytes, the ERRLOCN points to 8192 when the last bit of the sector area is corrupted.

If the sector size is set to 512 bytes, the ERRLOCN points to 4097 when the first bit of the spare area is corrupted.

If the sector size is set to 1024 bytes, the ERRLOCN points to 8193 when the first bit of the spare area is corrupted.

34.20.33 Setup Register

Name: HSMC_SETUPx [x=0..3]

Address: 0xF8014700 [0], 0xF8014714 [1], 0xF8014728 [2], 0xF801473C [3]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	NCS_RD_SETUP					

23	22	21	20	19	18	17	16
–	–	NRD_SETUP					

15	14	13	12	11	10	9	8
–	–	NCS_WR_SETUP					

7	6	5	4	3	2	1	0
–	–	NWE_SETUP					

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **NWE_SETUP: NWE Setup Length**

The NWE signal setup length is defined as:

NWE setup length = (128 * NWE_SETUP[5] + NWE_SETUP[4:0]) clock cycles.

- **NCS_WR_SETUP: NCS Setup Length in Write Access**

In write access, the NCS signal setup length is defined as:

NCS setup length = (128 * NCS_WR_SETUP[5] + NCS_WR_SETUP[4:0]) clock cycles.

- **NRD_SETUP: NRD Setup Length**

The NRD signal setup length is defined as:

NRD setup length = (128 * NRD_SETUP[5] + NRD_SETUP[4:0]) clock cycles.

- **NCS_RD_SETUP: NCS Setup Length in Read Access**

In Read access, the NCS signal setup length is defined as:

NCS setup length = (128 * NCS_RD_SETUP[5] + NCS_RD_SETUP[4:0]) clock cycles.

34.20.34 Pulse Register

Name: HSMC_PULSEx [x=0..3]

Address: 0xF8014704 [0], 0xF8014718 [1], 0xF801472C [2], 0xF8014740 [3]

Access: Write-only

31	30	29	28	27	26	25	24
–	NCS_RD_PULSE						

23	22	21	20	19	18	17	16
–	NRD_PULSE						

15	14	13	12	11	10	9	8
–	NCS_WR_PULSE						

7	6	5	4	3	2	1	0
–	NWE_PULSE						

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **NWE_PULSE: NWE Pulse Length**

The NWE signal pulse length is defined as:

NWE pulse length = (256 * NWE_PULSE[6]+NWE_PULSE[5:0]) clock cycles.

The NWE pulse must be at least one clock cycle.

- **NCS_WR_PULSE: NCS Pulse Length in WRITE Access**

In Write access, The NCS signal pulse length is defined as:

NCS pulse length = (256 * NCS_WR_PULSE[6] + NCS_WR_PULSE[5:0]) clock cycles.

The NCS pulse must be at least one clock cycle.

- **NRD_PULSE: NRD Pulse Length**

The NRD signal pulse length is defined as:

NRD pulse length = (256 * NRD_PULSE[6] + NRD_PULSE[5:0]) clock cycles.

The NRD pulse width must be as least 1 clock cycle.

- **NCS_RD_PULSE: NCS Pulse Length in READ Access**

In READ mode, The NCS signal pulse length is defined as:

NCS pulse length = (256 * NCS_RD_PULSE[6] + NCS_RD_PULSE[5:0]) clock cycles.

34.20.35 Cycle Register

Name: HSMC_CYCLEx [x=0..3]

Address: 0xF8014708 [0], 0xF801471C [1], 0xF8014730 [2], 0xF8014744 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	NRD_CYCLE

23	22	21	20	19	18	17	16
NRD_CYCLE							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	NWE_CYCLE

7	6	5	4	3	2	1	0
NWE_CYCLE							

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **NWE_CYCLE: Total Write Cycle Length**

The total write cycle length is the total duration in clock cycles of the write cycle. It is equal to the sum of the setup, pulse and hold steps of the NWE and NCS signals. It is defined as:

Write cycle length = (NWE_CYCLE[8:7] * 256) + NWE_CYCLE[6:0] clock cycles.

- **NRD_CYCLE: Total Read Cycle Length**

The total read cycle length is the total duration in clock cycles of the read cycle. It is equal to the sum of the setup, pulse and hold steps of the NRD and NCS signals. It is defined as:

Read cycle length = (NRD_CYCLE[8:7] * 256) + NRD_CYCLE[6:0] clock cycles.

34.20.36 Timings Register

Name: HSMC_TIMINGSx [x=0..3]

Address: 0xF801470C [0], 0xF8014720 [1], 0xF8014734 [2], 0xF8014748 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
NFSEL	–	–	–	TWB			

23	22	21	20	19	18	17	16
–	–	–	–	TRR			

15	14	13	12	11	10	9	8
–	–	–	OCMS	TAR			

7	6	5	4	3	2	1	0
TADL				TCLR			

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **TCLR: CLE to REN Low Delay**

Command Latch Enable falling edge to Read Enable falling edge timing.

Latch Enable Falling to Read Enable Falling = (TCLR[3] * 64) + TCLR[2:0] clock cycles.

- **TADL: ALE to Data Start**

Last address latch cycle to the first rising edge of WEN for data input.

Last address latch to first rising edge of WEN = (TADL[3] * 64) + TADL[2:0] clock cycles.

- **TAR: ALE to REN Low Delay**

Address Latch Enable falling edge to Read Enable falling edge timing.

Address Latch Enable to Read Enable = (TAR[3] * 64) + TAR[2:0] clock cycles.

- **OCMS: Off Chip Memory Scrambling Enable**

When set to one, the memory scrambling is activated. (Value must be zero if external memory is NAND Flash and NFC is used).

- **TRR: Ready to REN Low Delay**

Ready/Busy signal to Read Enable falling edge timing.

Read to REN = (TRR[3] * 64) + TRR[2:0] clock cycles.

- **TWB: WEN High to REN to Busy**

Write Enable rising edge to Ready/Busy falling edge timing.

Write Enable to Read/Busy = (TWB[3] * 64) + TWB[2:0] clock cycles.

- **NFSEL: NAND Flash Selection**

If this bit is set to one, the chip select is assigned to NAND Flash write enable and read enable lines drive the Error Correcting Code module.

34.20.37 Mode Register

Name: HSMC_MODEx [x=0..3]

Address: 0xF8014710 [0], 0xF8014724 [1], 0xF8014738 [2], 0xF801474C [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	TDF_MODE	TDF_CYCLES			
15	14	13	12	11	10	9	8
–	–	–	DBW	–	–	–	BAT
7	6	5	4	3	2	1	0
–	–	EXNW_MODE		–	–	WRITE_MODE	READ_MODE

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **READ_MODE: Selection of the Control Signal for Read Operation**

Value	Name	Description
0	NCS_CTRL	The Read operation is controlled by the NCS signal.
1	NRD_CTRL	The Read operation is controlled by the NRD signal.

- **WRITE_MODE: Selection of the Control Signal for Write Operation**

Value	Name	Description
0	NCS_CTRL	The Write operation is controller by the NCS signal.
1	NWE_CTRL	The Write operation is controlled by the NWE signal

- **EXNW_MODE: NWAIT Mode**

The NWAIT signal is used to extend the current read or write signal. It is only taken into account during the pulse phase Read and Write controlling signal. When the use of NWAIT is enabled, at least one cycle hold duration must be programmed for the read and write controlling signal.

Value	Name	Description
0	DISABLED	Disabled—The NWAIT input signal is ignored on the corresponding Chip Select.
1	–	Reserved
2	FROZEN	Frozen Mode—If asserted, the NWAIT signal freezes the current read or write cycle. After deassertion, the read/write cycle is resumed from the point where it was stopped.
3	READY	Ready Mode—The NWAIT signal indicates the availability of the external device at the end of the pulse of the controlling read or write signal, to complete the access. If high, the access normally completes. If low, the access is extended until NWAIT returns high.

- **BAT: Byte Access Type**

This field is used only if DBW defines a 16-bit data bus.

Value	Name	Description
0	BYTE_SELECT	Byte select access type: - Write operation is controlled using NCS, NWE, NBS0, NBS1. - Read operation is controlled using NCS, NRD, NBS0, NBS1.
1	BYTE_WRITE	Byte write access type: - Write operation is controlled using NCS, NWR0, NWR1. - Read operation is controlled using NCS and NRD.

- **DBW: Data Bus Width**

Value	Name	Description
0	BIT_8	8-bit bus
1	BIT_16	16-bit bus

- **TDF_CYCLES: Data Float Time**

This field gives the integer number of clock cycles required by the external device to release the data after the rising edge of the read controlling signal. The SMC always provide one full cycle of bus turnaround after the TDF_CYCLES period. The external bus cannot be used by another chip select during TDF_CYCLES + 1 cycles. From 0 up to 15 TDF_CYCLES can be set.

- **TDF_MODE: TDF Optimization**

1: TDF optimization enabled

– The number of TDF wait states is optimized using the setup period of the next read/write access.

0: TDF optimization disabled

– The number of TDF wait states is inserted before the next access begins.

34.20.38 Off Chip Memory Scrambling Register

Name: HSMC_OCMS

Address: 0xF80147A0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	SRSE	SMSE

- **SMSE: Static Memory Controller Scrambling Enable**

0: Disable “Off Chip” Scrambling for SMC access.

1: Enable “Off Chip” Scrambling for SMC access. (If OCMS bit is set in the corresponding HSMC_TIMINGSx register.)

- **SRSE: NFC Internal SRAM Scrambling Enable**

0: Disable Scrambling for NFC internal SRAM access.

1: Enable Scrambling for NFC internal SRAM access. (OCMS bit must be cleared in the corresponding HSMC_TIMINGSx register.)

34.20.39 Off Chip Memory Scrambling Key1 Register

Name: HSMC_KEY1

Address: 0xF80147A4

Access: Write-once

31	30	29	28	27	26	25	24
KEY1							

23	22	21	20	19	18	17	16
KEY1							

15	14	13	12	11	10	9	8
KEY1							

7	6	5	4	3	2	1	0
KEY1							

- **KEY1: Off Chip Memory Scrambling (OCMS) Key Part 1**

When Off Chip Memory Scrambling is enabled by setting the HSMC_OCMS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY1 and KEY2 values.

34.20.40 Off Chip Memory Scrambling Key2 Register

Name: HSMC_KEY2

Address: 0xF80147A8

Access: Write-once

31	30	29	28	27	26	25	24
KEY2							

23	22	21	20	19	18	17	16
KEY2							

15	14	13	12	11	10	9	8
KEY2							

7	6	5	4	3	2	1	0
KEY2							

- **KEY2: Off Chip Memory Scrambling (OCMS) Key Part 2**

When Off Chip Memory Scrambling is enabled by setting the HSMC_OCMS and HSMC_TIMINGS registers in accordance, the data scrambling depends on KEY2 and KEY1 values.

34.20.41 Write Protection Mode Register

Name: HSMC_WPMR

Address: 0xF80147E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables write protection if WPKEY value corresponds to 0x534D43 (“SMC” in ASCII)

1: Enables write protection if WPKEY value corresponds to 0x534D43 (“SMC” in ASCII)

See Section 34.16 “Register Write Protection” for list of write-protected registers.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x534D43	PASSWD	Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0.

34.20.42 Write Protection Status Register

Name: HSMC_WPSR

Address: 0xF80147E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protect violation has occurred since the last read of the HSMC_WPSR.

1: A write protect violation has occurred since the last read of the HSMC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

35. DMA Controller (XDMAC)

35.1 Description

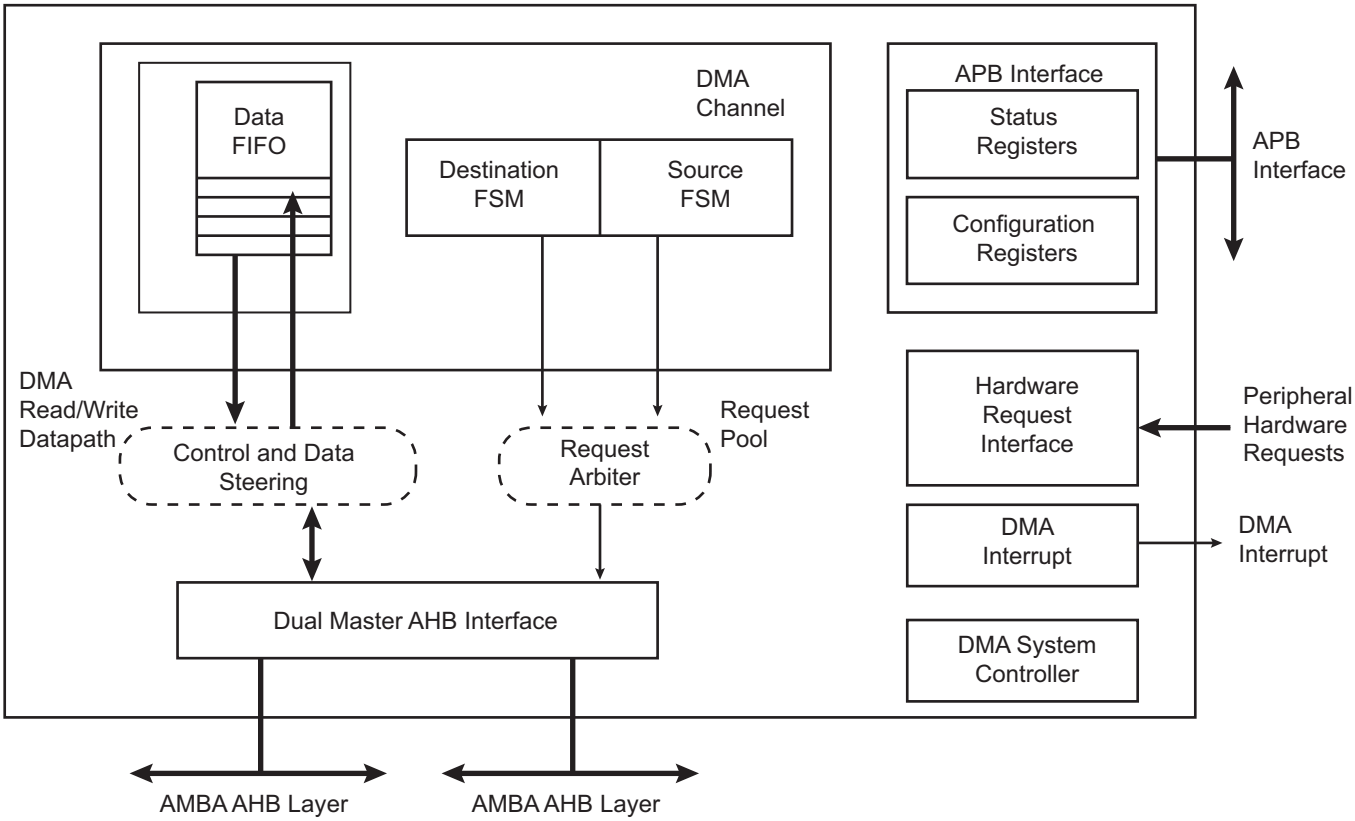
The DMA Controller (XDMAC) is a AHB-protocol central direct memory access controller. It performs peripheral data transfer and memory move operations over one or two bus ports through the unidirectional communication channel. Each channel is fully programmable and provides both peripheral or memory-to-memory transfers. The channel features are configurable at implementation.

35.2 Embedded Characteristics

- 2 AHB Master Interfaces
- 16 DMA Channels
- 51 Hardware Requests
- 4 Kbytes Embedded FIFO
- Supports Peripheral-to-Memory, Memory-to-Peripheral, or Memory-to-Memory Transfer Operations
- Peripheral DMA Operation Runs on Bytes (8-bit), Half-Word (16-bit) and Word (32-bit)
- Memory DMA Operation Runs on Bytes (8 bit), Half-Word (16-bit), Word (32-bit) and Double-Word (64-bit)
- Supports Hardware and Software Initiated Transfers
- Supports Linked List Operations
- Supports Incrementing or Fixed Addressing Mode
- Supports Programmable Independent Data Striding for Source and Destination
- Supports Programmable Independent Microblock Striding for Source and Destination
- Configurable Priority Group and Arbitration Policy
- Programmable AHB Burst Length
- Configuration Interface Accessible through APB Interface
- XDMAC Architecture Includes Multiport FIFO
- Supports Multiple View Channel Descriptor
- Automatic Flush of Channel Trailing Bytes
- Automatic Coarse-Grain and Fine-Grain Clock Gating
- Hardware Acceleration of Memset Pattern

35.3 Block Diagram

Figure 35-1. DMA Controller (XDMAC) Block Diagram



35.4 DMA Controller Peripheral Connections

The SAMA5D2 features two DMACs: XDMAC0 and XDMAC1. Both have the same features:

- Programmable secure access
- Two 64-bit masters
- 16 channels and 55 hardware requests embedded
- Sixteen 64-bit-word FIFOs on all channels
- Linked list support with status write back operation at end of transfer
- Word, half-word, byte transfer support
- Memory-to-memory transfer
- Peripheral-to-memory transfer
- Memory-to-peripheral transfer

The DMA controller can handle the transfer between peripherals and memory and so receives the triggers from the peripherals below.

Table 35-1 gives an overview of the different access when secure/non-secure DMA needs to access a secure/non-secure peripheral, and when a secure/non-secure peripheral needs to access secure/non-secure DMA.

Table 35-1. DMA Configuration vs. Peripheral

Peripheral	DMA	
	Secure	Non-secure
Secure	x	–
Non-secure	x	x

DMA Controller 0 manages transfers between peripherals and memory, and receives the triggers from the peripherals listed in Table 35-2.

Table 35-2. DMA Channels Definition (XDMAC0)

Instance Name	Channel T/R	Interface Number	XDMAC_CCx.CSIZE Required Value
TWIHS0	Transmit	0	0
TWIHS0	Receive	1	
TWIHS1	Transmit	2	0
TWIHS1	Receive	3	
QSPI0	Transmit	4	0
QSPI0	Receive	5	
SPI0	Transmit	6	0
SPI0	Receive	7	
SPI1	Transmit	8	0
SPI1	Receive	9	
PWM	Transmit	10	0
FLEXCOM0	Transmit	11	0
FLEXCOM0	Receive	12	
FLEXCOM1	Transmit	13	0
FLEXCOM1	Receive	14	

Table 35-2. DMA Channels Definition (XDMAC0) (Continued)

Instance Name	Channel T/R	Interface Number	XDMAC_CCx.CSIZE Required Value
FLEXCOM2	Transmit	15	0
FLEXCOM2	Receive	16	
FLEXCOM3	Transmit	17	0
FLEXCOM3	Receive	18	
FLEXCOM4	Transmit	19	0
FLEXCOM4	Receive	20	
SSC0	Transmit	21	0
SSC0	Receive	22	
SSC1	Transmit	23	0
SSC1	Receive	24	
ADC	Receive	25	0
AES	Transmit	26	0 or 2 (see AES Section 57.4.4.3 “DMA Mode”)
AES	Receive	27	
TDES	Transmit	28	0
TDES	Receive	29	
SHA	Transmit	30	4
I2SC0	Transmit	31	0
I2SC0	Receive	32	
I2SC1	Transmit	33	0
I2SC1	Receive	34	
UART0	Transmit	35	0
UART0	Receive	36	
UART1	Transmit	37	0
UART1	Receive	38	
UART2	Transmit	39	0
UART2	Receive	40	
UART3	Transmit	41	0
UART3	Receive	42	
UART4	Transmit	43	0
UART4	Receive	44	
TC0	Receive	45	0
TC1	Receive	46	
CLASSD	Transmit	47	0
QSPI1	Transmit	48	0
QSPI1	Receive	49	
PDMIC	Receive	50	0

DMA Controller 1 manages transfers between peripherals and memory, and receives the triggers from the peripherals listed in Table 35-3.

Table 35-3. DMA Channels Definition (XDMAC1)

Instance Name	Channel T/R	Interface Number	XDMAC_CCx.CSIZE Required Value
TWIHS0	Transmit	0	0
TWIHS0	Receive	1	
TWIHS1	Transmit	2	0
TWIHS1	Receive	3	
QSPI0	Transmit	4	0
QSPI0	Receive	5	
SPI0	Transmit	6	0
SPI0	Receive	7	
SPI1	Transmit	8	0
SPI1	Receive	9	
PWM	Transmit	10	0
FLEXCOM0	Transmit	11	0
FLEXCOM0	Receive	12	
FLEXCOM1	Transmit	13	0
FLEXCOM1	Receive	14	
FLEXCOM2	Transmit	15	0
FLEXCOM2	Receive	16	
FLEXCOM3	Transmit	17	0
FLEXCOM3	Receive	18	
FLEXCOM4	Transmit	19	0
FLEXCOM4	Receive	20	
SSC0	Transmit	21	0
SSC0	Receive	22	
SSC1	Transmit	23	0
SSC1	Receive	24	
ADC	Receive	25	0
AES	Transmit	26	0 or 2 (see AES Section 57.4.4.3 “DMA Mode”)
AES	Receive	27	
TDES	Transmit	28	0
TDES	Receive	29	
SHA	Transmit	30	4
I2SC0	Transmit	31	0
I2SC0	Receive	32	
I2SC1	Transmit	33	0
I2SC1	Receive	34	

Table 35-3. DMA Channels Definition (XDMAC1) (Continued)

Instance Name	Channel T/R	Interface Number	XDMAC_CCx.CSIZE Required Value
UART0	Transmit	35	0
UART0	Receive	36	
UART1	Transmit	37	0
UART1	Receive	38	
UART2	Transmit	39	0
UART2	Receive	40	
UART3	Transmit	41	0
UART3	Receive	42	
UART4	Transmit	43	0
UART4	Receive	44	
TC0	Receive	45	0
TC1	Receive	46	
CLASSD	Transmit	47	0
QSPI1	Transmit	48	0
QSPI1	Receive	49	
PDMIC	Receive	50	0

35.5 Functional Description

35.5.1 Basic Definitions

Source Peripheral: Slave device, memory mapped on the interconnection network, from where the XDMAC reads data. The source peripheral teams up with a destination peripheral to form a channel. A data read operation is scheduled when the peripheral transfer request is asserted.

Destination Peripheral: Slave device, memory mapped on the interconnection network, to which the XDMAC writes. A write data operation is scheduled when the peripheral transfer request is asserted.

Channel: The data movement between source and destination creates a logical channel.

Transfer Type: The transfer is hardware synchronized when it is paced by the peripheral hardware request, otherwise the transfer is self-triggered (memory to memory transfer).

35.5.2 Transfer Hierarchy Diagram

XDMAC Master Transfer: The Master Transfer is composed of a linked list of blocks. The channel address, control and configuration registers can be modified at the inter block boundary. The descriptor structure modifies the channel registers conditionally. Interrupts can be generated on a per block basis or when the end of linked list event occurs.

XDMAC Block: An XDMAC block is composed of a programmable number of microblocks. The channel configuration registers remain unchanged at the inter microblock boundary. The source and destination addresses are conditionally updated with a programmable signed number.

XDMAC Microblock: The microblock is composed of a programmable number of data. The channel configuration registers remain unchanged at the data boundary. The data address may be fixed (a FIFO location, a peripheral transmit or receive register), incrementing (a memory mapped area) by a programmable signed number.

XDMAC Burst and Incomplete Burst: In order to improve the overall performance when accessing dynamic external memory, burst access is mandatory. Each data of the microblock is considered as a part of a memory burst. The programmable burst value indicates the largest memory burst allowed on a per channel basis. When the microblock length is not an integral multiple of the burst size, an incomplete burst is performed to read or write the last trailing bytes.

XDMAC Chunk and Incomplete Chunk: When a peripheral synchronized transfer is activated, the microblock splits into a number of data chunks. The chunk size is programmable. The larger the chunk is, the better the performance is. When the transfer size is not a multiple of the chunk size, the last chunk may be incomplete.

Figure 35-2. XDMAC Memory Transfer Hierarchy

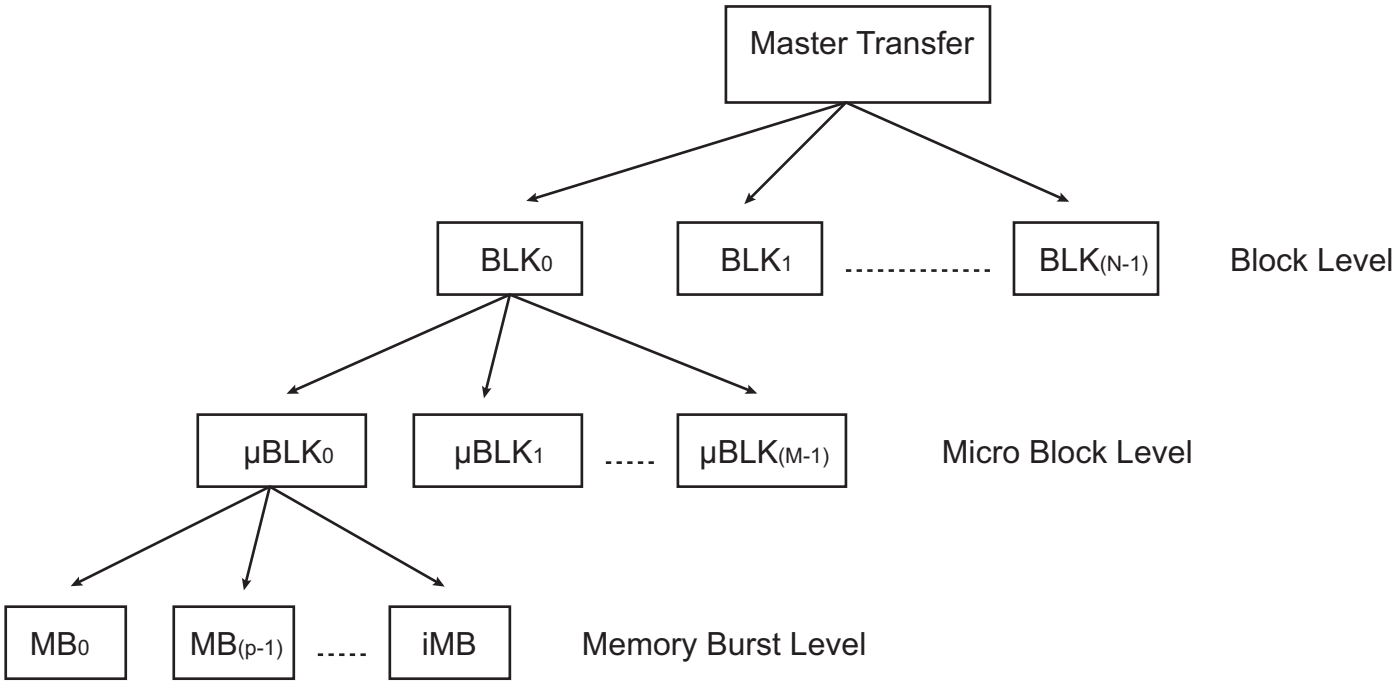
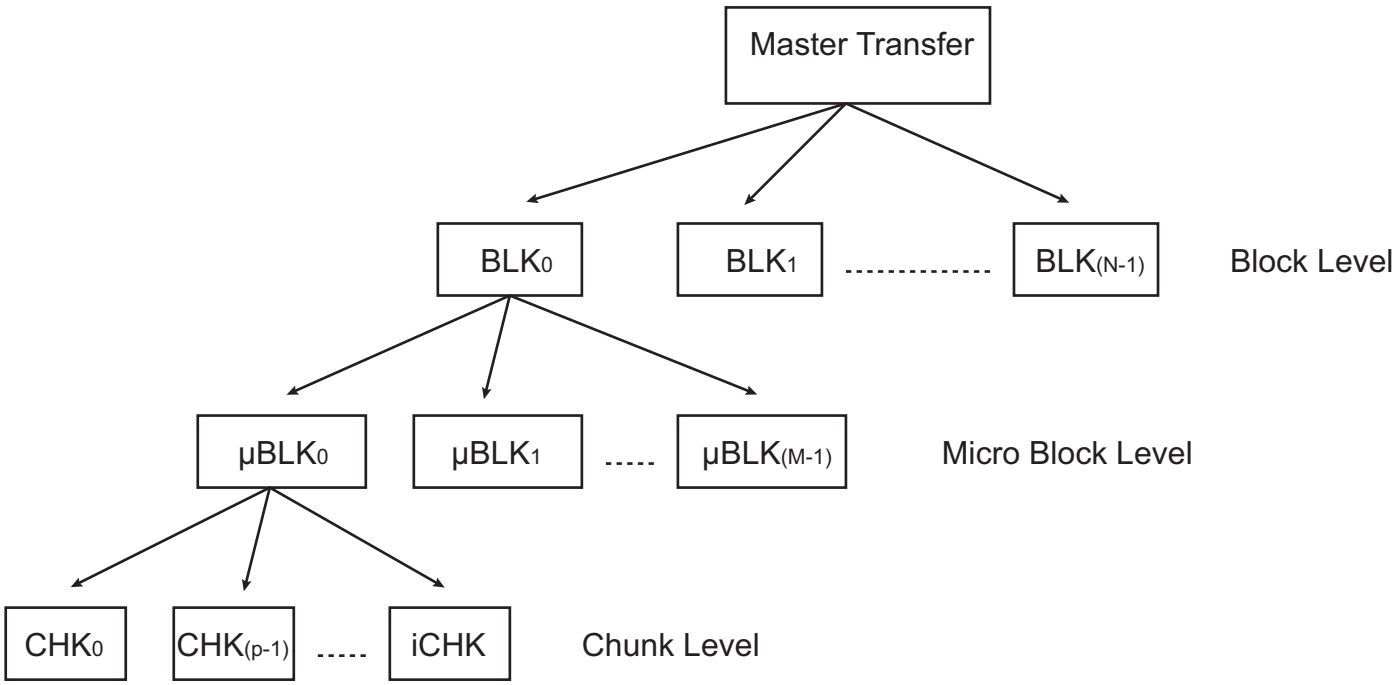


Figure 35-3. XDAMC Peripheral Transfer Hierarchy



35.5.3 Peripheral Synchronized Transfer

A peripheral hardware request interface is used to control the pace of the chunk transfer. When a peripheral is ready to transmit or receive a chunk of data, it asserts its request line and the DMA Controller transfers a data to or from the memory to the peripheral.

35.5.3.1 Software Triggered Synchronized Transfer

The Peripheral hardware request can be software controlled using the SWREQ field of the XDMAC Global Channel Software Request Register (XDMAC_GSWR). The peripheral synchronized transfer is paced using a processor write access in the XDMAC_GSWR. Each bit of that register triggers a transfer request. The XDMAC Global Channel Software Request Status Register (XDMAC_GSWS) indicates the status of the request; when set, the request is still pending.

35.5.4 XDMAC Transfer Software Operation

Note: When a memory-to-memory transfer is performed, configure the field XDMAC_CCx.PERID (where ‘x’ is the index of the channel used for transfer) to an unused peripheral ID (refer to table “Peripheral Identifiers”).

35.5.4.1 Single Block With Single Microblock Transfer

1. Read the XDMAC Global Channel Status Register (XDMAC_GS) to select a free channel.
2. Clear the pending Interrupt Status bit(s) by reading the selected XDMAC Channel x Interrupt Status Register (XDMAC_CISx).
3. Write the XDMAC Channel x Source Address Register (XDMAC_CSAx) for channel x.
4. Write the XDMAC Channel x Destination Address Register (XDMAC_CDAx) for channel x.
5. Program field UBLEN in the XDMAC Channel x Microblock Control Register (XDMAC_CUBCx) with the number of data.
6. Program the XDMAC Channel x Configuration Register (XDMAC_CCx):
    1. Clear XDMAC_CCx.TYPE for a memory-to-memory transfer, otherwise set this bit.
    2. Configure XDMAC_CCx.MBSIZE to the memory burst size used.
    3. Configure XDMAC_CCx.SAM and DAM to the memory addressing mode.
    4. Configure XDMAC_CCx.DSYNC to select the peripheral transfer direction.
    5. Set XDMAC_CCx.PROT to activate a secure channel.
    6. Configure XDMAC_CCx.CSIZE to configure the channel chunk size (only relevant for peripheral synchronized transfer).
    7. Configure XDMAC_CCx.DWIDTH to configure the transfer data width.
    8. Configure XDMAC_CCx.SIF, XDMAC_CCx.DIF to configure the master interface used to read data and write data, respectively.
    9. Configure XDMAC_CCx.PERID to select the active hardware request line (only relevant for a peripheral synchronized transfer).
    10. Set XDMAC_CCx.SWREQ to use a software request (only relevant for a peripheral synchronized transfer).
7. Clear the following five registers:
    - XDMAC Channel x Next Descriptor Control Register (XDMAC_CNDCx)
    - XDMAC Channel x Block Control Register (XDMAC_CBCx)
    - XDMAC Channel x Data Stride Memory Set Pattern Register (XDMAC_CDS_MSPx)
    - XDMAC Channel x Source Microblock Stride Register (XDMAC_CSUSx)
    - XDMAC Channel x Destination Microblock Stride Register (XDMAC_CDUSx)

    This indicates that the linked list is disabled, there is only one block and striding is disabled.
8. Enable the Microblock interrupt by writing a ‘1’ to bit BIE in the XDMAC Channel x Interrupt Enable Register (XDMAC_CIEx). Enable the Channel x Interrupt Enable bit by writing a ‘1’ to bit IEx in the XDMAC Global Interrupt Enable Register (XDMAC_GIE).

9. Enable channel x by writing a ‘1’ to bit ENx in the XDMAC Global Channel Enable Register (XDMAC_GE). XDMAC_GS.STx (XDMAC Channel x Status bit) is set by hardware.
10. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

35.5.4.2 Single Block Transfer With Multiple Microblock

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Write the XDMAC_CSAx register for channel x.
4. Write the XDMAC_CDAx register for channel x.
5. Program XDMAC_CUBCx.UBLEN with the number of data.
6. Program XDMAC_CCx register (see single block transfer configuration).
7. Program XDMAC_CBCx.BLEN with the number of microblocks of data.
8. Clear the following four registers:
   - XDMAC_CNDCx
   - XDMAC_CDS_MSPx
   - XDMAC_CSUSx XDMAC_CDUSx

   This indicates that the linked list is disabled and striding is disabled.
9. Enable the Block interrupt by writing a ‘1’ to XDMAC_CIEx.BIE, enable the Channel x Interrupt Enable bit by writing a ‘1’ to XDMAC_GIEx.IEx.
10. Enable channel x by writing a ‘1’ to the XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
11. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

35.5.4.3 Master Transfer

1. Read the XDMAC_GS register to choose a free channel.
2. Clear the pending Interrupt Status bit by reading the chosen XDMAC_CISx register.
3. Build a linked list of transfer descriptors in memory. The descriptor view is programmable on a per descriptor basis. The linked list items structure must be word aligned. MBR_UBC.NDE must be configured to 0 in the last descriptor to terminate the list.
4. Configure field NDA in the XDMAC Channel x Next Descriptor Address Register (XDMAC_CNDAx) with the first descriptor address and bit XDMAC_CNDAx.NDAIF with the master interface identifier.
5. Configure the XDMAC_CNDCx register:
   1. Set XDMAC_CNDCx.NDE to enable the descriptor fetch.
   2. Set XDMAC_CNDCx.NDSUP to update the source address at the descriptor fetch time, otherwise clear this bit.
   3. Set XDMAC_CNDCx.NDDUP to update the destination address at the descriptor fetch time, otherwise clear this bit.
   4. Configure XDMAC_CNDCx.NDVIEW to define the length of the first descriptor.
6. Enable the End of Linked List interrupt by writing a ‘1’ to XDMAC_CIEx.LIE.
7. Enable channel x by writing a ‘1’ to XDMAC_GE.ENx. XDMAC_GS.STx is set by hardware.
8. Once completed, the DMA channel sets XDMAC_CISx.BIS (End of Block Interrupt Status bit) and generates an interrupt. XDMAC_GS.STx is cleared by hardware. The software can either wait for an interrupt or poll the channel status bit.

35.5.4.4 Disabling A Channel Before Transfer Completion

Under normal operation, the software enables a channel by writing a ‘1’ to XDMAC_GE.ENx, then the hardware disables a channel on transfer completion by clearing bit XDMAC_GS.STx. To disable a channel, write a ‘1’ to bit XDMAC_GD.DIx and poll the XDMAC_GS register.

35.6 Linked List Descriptor Operation

35.6.1 Linked List Descriptor View

35.6.1.1 Channel Next Descriptor View 0–3 Structures

Table 35-4. Channel Next Descriptor View 0–3 Structures

Channel Next Descriptor	Offset	Structure member	Name
View 0 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Transfer Address Member	MBR_TA
View 1 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Source Address Member	MBR_SA
	DSCR_ADDR+0x0C	Destination Address Member	MBR_DA
View 2 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Source Address Member	MBR_SA
	DSCR_ADDR+0x0C	Destination Address Member	MBR_DA
	DSCR_ADDR+0x10	Configuration Register	MBR_CFG
View 3 Structure	DSCR_ADDR+0x00	Next Descriptor Address Member	MBR_NDA
	DSCR_ADDR+0x04	Microblock Control Member	MBR_UBC
	DSCR_ADDR+0x08	Source Address Member	MBR_SA
	DSCR_ADDR+0x0C	Destination Address Member	MBR_DA
	DSCR_ADDR+0x10	Configuration Member	MBR_CFG
	DSCR_ADDR+0x14	Block Control Member	MBR_BC
	DSCR_ADDR+0x18	Data Stride Member	MBR_DS
	DSCR_ADDR+0x1C	Source Microblock Stride Member	MBR_SUS
	DSCR_ADDR+0x20	Destination Microblock Stride Member	MBR_DUS

35.6.2 Descriptor Structure Members Description

35.6.2.1 Descriptor Structure Microblock Control Member

Name: MBR_UBC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	NVIEW		NDEN	NSEN	NDE
23	22	21	20	19	18	17	16
UBLEN							
15	14	13	12	11	10	9	8
UBLEN							
7	6	5	4	3	2	1	0
UBLEN							

- **UBLEN: Microblock Length**

This field indicates the number of data in the microblock. The microblock contains UBLEN data.

- **NDE: Next Descriptor Enable**

0: Descriptor fetch is disabled.

1: Descriptor fetch is enabled.

- **NSEN: Next Descriptor Source Update**

0: Source parameters remain unchanged.

1: Source parameters are updated when the descriptor is retrieved.

- **NDEN: Next Descriptor Destination Update**

0: Destination parameters remain unchanged.

1: Destination parameters are updated when the descriptor is retrieved.

- **NVIEW: Next Descriptor View**

Value	Name	Description
0	NDV0	Next Descriptor View 0
1	NDV1	Next Descriptor View 1
2	NDV2	Next Descriptor View 2
3	NDV3	Next Descriptor View 3

35.7 XDMAC Maintenance Software Operations

35.7.1 Disabling a Channel

A disable channel request occurs when a write operation is performed in the XDMAC_GD register. If the channel is source peripheral synchronized (bit XDMAC_CCx.TYPE is set and bit XDMAC_CCx.DSYNC is cleared), then pending bytes (bytes located in the FIFO) are written to memory and bit XDMAC_CISx.DIS is set. If the channel is not source peripheral synchronized, the current channel transaction (read or write) is terminated and XDMAC_CISx.DIS is set. XDMAC_GS.STx is cleared by hardware when the current transfer is completed. The channel is no longer active and can be reused.

35.7.2 Suspending a Channel

A read request suspend command is issued by writing to the XDMAC_GRS register. A write request suspend command is issued by writing to the XDMAC_GWS register. A read write suspend channel is issued by writing to the XDMAC_GRWS register. These commands have an immediate effect on the scheduling of both read and write transactions. If a transaction is already in progress, it is terminated normally. The channel is not disabled. The FIFO content is preserved. The scheduling mechanism can resume normally, clearing the bit in the same registers. Pending bytes located in the FIFO are not written out to memory. The write suspend command does not affect read request operations, i.e., read operations can still occur until the FIFO is full.

35.7.3 Flushing a Channel

A FIFO flush command is issued writing to the XDMAC_SWF register. The content of the FIFO is written to memory. XDMAC_CISx.FIS (End of Flush Interrupt Status bit) is set when the last byte is successfully transferred to memory. The channel is not disabled. The flush operation is not blocking, meaning that read operation can be scheduled during the flush write operation. The flush operation is only relevant for peripheral to memory transfer where pending peripheral bytes are buffered into the channel FIFO.

35.7.4 Maintenance Operation Priority

35.7.4.1 Disable Operation Priority

- When a disable request occurs on a suspended channel, the XDMAC_GWS.WSx (Channel x Write Suspend bit) is cleared. If the transfer is source peripheral synchronized, the pending bytes are drained to memory. The bit XDMAC_CISx.DIS is set.
- When a disable request follows a flush request, if the flush last transaction is not yet scheduled, the flush request is discarded and the disable procedure is applied. The bit XDMAC_CISx.FIS is not set. Bit XDMAC_CISx.DIS will be set when the disable request is completed. If the flush request transaction is already scheduled, the XDMAC_CISx.FIS will be set. XDMAC_CISx.DIS will also be set when the disable request is completed.

35.7.4.2 Flush Operation Priority

- When a flush request occurs on a suspended channel, if there are pending bytes in the FIFO, they are written out to memory, XDMAC_CISx.FIS is set. If the FIFO is empty, XDMAC_CISx.FIS is also set.
- If the flush operation is performed after a disable request, the flush command is ignored. XDMAC_CISx.FIS is not set.

35.7.4.3 Suspend Operation Priority

If the suspend operation is performed after a disable request, the write suspend operation is ignored.

35.8 XDMAC Software Requirements

- Write operations to channel registers are not be performed in an active channel after the channel is enabled. If any channel parameters must be reprogrammed, this can only be done after disabling the XDMAC channel.
- XDMAC_CSAx and XDMAC_CDAx channel registers are to be programmed with a byte, half-word, word or double-word aligned address depending on the Channel x Data Width field (DWIDTH) of the XDMAC Channel x Configuration Register.
- When a memory-to-memory transfer is performed, configure the field XDMAC_CCx.PERID (where ‘x’ is the index of the channel used for transfer) to an unused peripheral ID (refer to table “Peripheral Identifiers”).

35.9 Extensible DMA Controller (XDMAC) User Interface

Table 35-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Global Type Register	XDMAC_GTYPE	Read-only	0x00000000
0x04	Global Configuration Register	XDMAC_GCFG	Read/Write	0x00000000
0x08	Global Weighted Arbiter Configuration Register	XDMAC_GWAC	Read/Write	0x00000000
0x0C	Global Interrupt Enable Register	XDMAC_GIE	Write-only	–
0x10	Global Interrupt Disable Register	XDMAC_GID	Write-only	–
0x14	Global Interrupt Mask Register	XDMAC_GIM	Read-only	0x00000000
0x18	Global Interrupt Status Register	XDMAC_GIS	Read-only	0x00000000
0x1C	Global Channel Enable Register	XDMAC_GE	Write-only	–
0x20	Global Channel Disable Register	XDMAC_GD	Write-only	–
0x24	Global Channel Status Register	XDMAC_GS	Read-only	0x00000000
0x28	Global Channel Read Suspend Register	XDMAC_GRS	Read/Write	0x00000000
0x2C	Global Channel Write Suspend Register	XDMAC_GWS	Read/Write	0x00000000
0x30	Global Channel Read Write Suspend Register	XDMAC_GRWS	Write-only	–
0x34	Global Channel Read Write Resume Register	XDMAC_GRWR	Write-only	–
0x38	Global Channel Software Request Register	XDMAC_GSWR	Write-only	–
0x3C	Global Channel Software Request Status Register	XDMAC_GSWS	Read-only	0x00000000
0x40	Global Channel Software Flush Request Register	XDMAC_GSWF	Write-only	–
0x44–0x4C	Reserved	–	–	–
0x50+chid*0x40	Channel Interrupt Enable Register	XDMAC_CIE	Write-only	–
0x54+chid*0x40	Channel Interrupt Disable Register	XDMAC_CID	Write-only	–
0x58+chid*0x40	Channel Interrupt Mask Register	XDMAC_CIM	Read-only	–
0x5C+chid*0x40	Channel Interrupt Status Register	XDMAC_CIS	Read-only	0x00000000
0x60+chid*0x40	Channel Source Address Register	XDMAC_CSA	Read/Write	0x00000000
0x64+chid*0x40	Channel Destination Address Register	XDMAC_CDA	Read/Write	0x00000000
0x68+chid*0x40	Channel Next Descriptor Address Register	XDMAC_CNDA	Read/Write	0x00000000
0x6C+chid*0x40	Channel Next Descriptor Control Register	XDMAC_CNDC	Read/Write	0x00000000
0x70+chid*0x40	Channel Microblock Control Register	XDMAC_CUBC	Read/Write	0x00000000
0x74+chid*0x40	Channel Block Control Register	XDMAC_CBC	Read/Write	0x00000000
0x78+chid*0x40	Channel Configuration Register	XDMAC_CC	Read/Write	0x00000000
0x7C+chid*0x40	Channel Data Stride Memory Set Pattern	XDMAC_CDS_MSP	Read/Write	0x00000000
0x80+chid*0x40	Channel Source Microblock Stride	XDMAC_CSUS	Read/Write	0x00000000
0x84+chid*0x40	Channel Destination Microblock Stride	XDMAC_CDUS	Read/Write	0x00000000
0x88+chid*0x40	Reserved	–	–	–
0x8C+chid*0x40	Reserved	–	–	–
0xFEC–0xFFC	Reserved	–	–	–

35.9.1 XDMAC Global Type Register

Name: XDMAC_GTYPE

Address: 0xF0010000 (0), 0xF0004000 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	NB_REQ						

15	14	13	12	11	10	9	8
FIFO_SZ							

7	6	5	4	3	2	1	0
FIFO_SZ			NB_CH				

- **NB_CH: Number of Channels Minus One**
- **FIFO_SZ: Number of Bytes**
- **NB_REQ: Number of Peripheral Requests Minus One**

35.9.2 XDMAC Global Configuration Register

Name: XDMAC_GCFG

Address: 0xF0010004 (0), 0xF0004004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	BXKBEN
7	6	5	4	3	2	1	0
–	–	–	–	CGDISIF	CGDISFIFO	CGDISPIPE	CGDISREG

- **CGDISREG: Configuration Registers Clock Gating Disable**

0: The automatic clock gating is enabled for the configuration registers.

1: The automatic clock gating is disabled for the configuration registers.

- **CGDISPIPE: Pipeline Clock Gating Disable**

0: The automatic clock gating is enabled for the main pipeline.

1: The automatic clock gating is disabled for the main pipeline.

- **CGDISFIFO: FIFO Clock Gating Disable**

0: The automatic clock gating is enabled for the main FIFO.

1: The automatic clock gating is disabled for the main FIFO.

- **CGDISIF: Bus Interface Clock Gating Disable**

0: The automatic clock gating is enabled for the system bus interface.

1: The automatic clock gating is disabled for the system bus interface.

- **BXKBEN: Boundary X Kilobyte Enable**

0: The 1 Kbyte boundary is used.

1: The controller does not meet the AHB specification.

35.9.3 XDMAC Global Weighted Arbiter Configuration Register

Name: XDMAC_GWAC

Address: 0xF0010008 (0), 0xF0004008 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PW3				PW2			

7	6	5	4	3	2	1	0
PW1				PW0			

- **PW0: Pool Weight 0**

This field indicates the weight of the pool 0 in the arbitration scheme of the XDMA scheduler.

- **PW1: Pool Weight 1**

This field indicates the weight of the pool 1 in the arbitration scheme of the XDMA scheduler.

- **PW2: Pool Weight 2**

This field indicates the weight of the pool 2 in the arbitration scheme of the XDMA scheduler.

- **PW3: Pool Weight 3**

This field indicates the weight of the pool 3 in the arbitration scheme of the XDMA scheduler.

35.9.4 XDMAC Global Interrupt Enable Register

Name: XDMAC_GIE

Address: 0xF001000C (0), 0xF000400C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
IE15	IE14	IE13	IE12	IE11	IE10	IE9	IE8
7	6	5	4	3	2	1	0
IE7	IE6	IE5	IE4	IE3	IE2	IE1	IE0

- **IEx: XDMAC Channel x Interrupt Enable Bit**

0: This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified.

1: The corresponding mask bit is set. The XDMAC Channel x Interrupt Status register (XDMAC_GIS) can generate an interrupt.

35.9.5 XDMAC Global Interrupt Disable Register

Name: XDMAC_GID

Address: 0xF0010010 (0), 0xF0004010 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
7	6	5	4	3	2	1	0
ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0

- **IDx: XDMAC Channel x Interrupt Disable Bit**

0: This bit has no effect. The Channel x Interrupt Mask bit (XDMAC_GIM.IMx) is not modified.

1: The corresponding mask bit is reset. The Channel x Interrupt Status register interrupt (XDMAC_GIS) is masked.

35.9.6 XDMAC Global Interrupt Mask Register

Name: XDMAC_GIM

Address: 0xF0010014 (0), 0xF0004014 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
IM15	IM14	IM13	IM12	IM11	IM10	IM9	IM8
7	6	5	4	3	2	1	0
IM7	IM6	IM5	IM4	IM3	IM2	IM1	IM0

- **IMx: XDMAC Channel x Interrupt Mask Bit**

0: This bit indicates that the channel x interrupt source is masked. The interrupt line is not raised.

1: This bit indicates that the channel x interrupt source is unmasked.

35.9.7 XDMAC Global Interrupt Status Register

Name: XDMAC_GIS

Address: 0xF0010018 (0), 0xF0004018 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
IS15	IS14	IS13	IS12	IS11	IS10	IS9	IS8
7	6	5	4	3	2	1	0
IS7	IS6	IS5	IS4	IS3	IS2	IS1	IS0

- **ISx: XDMAC Channel x Interrupt Status Bit**

0: This bit indicates that either the interrupt source is masked at the channel level or no interrupt is pending for channel x.

1: This bit indicates that an interrupt is pending for the channel x.

35.9.8 XDMAC Global Channel Enable Register

Name: XDMAC_GE

Address: 0xF001001C (0), 0xF000401C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
EN15	EN14	EN13	EN12	EN11	EN10	EN9	EN8
7	6	5	4	3	2	1	0
EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0

- **ENx: XDMAC Channel x Enable Bit**

0: This bit has no effect.

1: Enables channel x. This operation is permitted if the Channel x Status bit (XDMAC_GS.STx) was read as 0.

35.9.9 XDMAC Global Channel Disable Register

Name: XDMAC_GD

Address: 0xF0010020 (0), 0xF0004020 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
DI15	DI14	DI13	DI12	DI11	DI10	DI9	DI8
7	6	5	4	3	2	1	0
DI7	DI6	DI5	DI4	DI3	DI2	DI1	DI0

- **DIx: XDMAC Channel x Disable Bit**

0: This bit has no effect.

1: Disables channel x.

35.9.10 XDMAC Global Channel Status Register

Name: XDMAC_GS

Address: 0xF0010024 (0), 0xF0004024 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
ST15	ST14	ST13	ST12	ST11	ST10	ST9	ST8
7	6	5	4	3	2	1	0
ST7	ST6	ST5	ST4	ST3	ST2	ST1	ST0

- **STx: XDMAC Channel x Status Bit**

0: This bit indicates that the channel x is disabled.

1: This bit indicates that the channel x is enabled. If a channel disable request is issued, this bit remains asserted until pending transaction is completed.

35.9.11 XDMAC Global Channel Read Suspend Register

Name: XDMAC_GRS

Address: 0xF0010028 (0), 0xF0004028 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RS15	RS14	RS13	RS12	RS11	RS10	RS9	RS8

7	6	5	4	3	2	1	0
RS7	RS6	RS5	RS4	RS3	RS2	RS1	RS0

- **RSx: XDMAC Channel x Read Suspend Bit**

0: The read channel is not suspended.

1: The source requests for channel x are no longer serviced by the system scheduler.

35.9.12 XDMAC Global Channel Write Suspend Register

Name: XDMAC_GWS

Address: 0xF001002C (0), 0xF000402C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
WS15	WS14	WS13	WS12	WS11	WS10	WS9	WS8
7	6	5	4	3	2	1	0
WS7	WS6	WS5	WS4	WS3	WS2	WS1	WS0

- **WSx: XDMAC Channel x Write Suspend Bit**

0: The write channel is not suspended.

1: Destination requests are no longer routed to the scheduler.

35.9.13 XDMAC Global Channel Read Write Suspend Register

Name: XDMAC_GRWS

Address: 0xF0010030 (0), 0xF0004030 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RWS15	RWS14	RWS13	RWS12	RWS11	RWS10	RWS9	RWS8
7	6	5	4	3	2	1	0
RWS7	RWS6	RWS5	RWS4	RWS3	RWS2	RWS1	RWS0

- **RWSx: XDMAC Channel x Read Write Suspend Bit**

0: No effect.

1: Read and write requests are suspended.

35.9.14 XDMAC Global Channel Read Write Resume Register

Name: XDMAC_GRWR

Address: 0xF0010034 (0), 0xF0004034 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RWR15	RWR14	RWR13	RWR12	RWR11	RWR10	RWR9	RWR8
7	6	5	4	3	2	1	0
RWR7	RWR6	RWR5	RWR4	RWR3	RWR2	RWR1	RWR0

- **RWRx: XDMAC Channel x Read Write Resume Bit**

0: No effect.

1: Read and write requests are serviced.

35.9.15 XDMAC Global Channel Software Request Register

Name: XDMAC_GSWR

Address: 0xF0010038 (0), 0xF0004038 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
SWREQ15	SWREQ14	SWREQ13	SWREQ12	SWREQ11	SWREQ10	SWREQ9	SWREQ8

7	6	5	4	3	2	1	0
SWREQ7	SWREQ6	SWREQ5	SWREQ4	SWREQ3	SWREQ2	SWREQ1	SWREQ0

- **SWREQx: XDMAC Channel x Software Request Bit**

0: No effect.

1: Requests a DMA transfer for channel x.

35.9.16 XDMAC Global Channel Software Request Status Register

Name: XDMAC_GSWS

Address: 0xF001003C (0), 0xF000403C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	-	–	–	–	–	–	–

15	14	13	12	11	10	9	8
SWRS15	SWRS14	SWRS13	SWRS12	SWRS11	SWRS10	SWRS9	SWRS8

7	6	5	4	3	2	1	0
SWRS7	SWRS6	SWRS5	SWRS4	SWRS3	SWRS2	SWRS1	SWRS0

- **SWRSx: XDMAC Channel x Software Request Status Bit**

0: Channel x source request is serviced.

1: Channel x source request is pending.

35.9.17 XDMAC Global Channel Software Flush Request Register

Name: XDMAC_GSWF

Address: 0xF0010040 (0), 0xF0004040 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
SWF15	SWF14	SWF13	SWF12	SWF11	SWF10	SWF9	SWF8
7	6	5	4	3	2	1	0
SWF7	SWF6	SWF5	SWF4	SWF3	SWF2	SWF1	SWF0

- **SWFx: XDMAC Channel x Software Flush Request Bit**

0: No effect.

1: Requests a DMA transfer flush for channel x. This bit is only relevant when the transfer is source peripheral synchronized.

35.9.18 XDMAC Channel x [x = 0..15] Interrupt Enable Register

Name: XDMAC_CIEx [x = 0..15]

Address: 0xF0004050 (1)[0], 0xF0004090 (1)[1], 0xF00040D0 (1)[2], 0xF0004110 (1)[3], 0xF0004150 (1)[4], 0xF0004190 (1)[5], 0xF00041D0 (1)[6], 0xF0004210 (1)[7], 0xF0004250 (1)[8], 0xF0004290 (1)[9], 0xF00042D0 (1)[10], 0xF0004310 (1)[11], 0xF0004350 (1)[12], 0xF0004390 (1)[13], 0xF00043D0 (1)[14], 0xF0004410 (1)[15], 0xF0010050 (0)[0], 0xF0010090 (0)[1], 0xF00100D0 (0)[2], 0xF0010110 (0)[3], 0xF0010150 (0)[4], 0xF0010190 (0)[5], 0xF00101D0 (0)[6], 0xF0010210 (0)[7], 0xF0010250 (0)[8], 0xF0010290 (0)[9], 0xF00102D0 (0)[10], 0xF0010310 (0)[11], 0xF0010350 (0)[12], 0xF0010390 (0)[13], 0xF00103D0 (0)[14], 0xF0010410 (0)[15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	ROIE	WBIE	RBIE	FIE	DIE	LIE	BIE

- **BIE: End of Block Interrupt Enable Bit**

0: No effect.

1: Enables end of block interrupt.

- **LIE: End of Linked List Interrupt Enable Bit**

0: No effect.

1: Enables end of linked list interrupt.

- **DIE: End of Disable Interrupt Enable Bit**

0: No effect.

1: Enables end of disable interrupt.

- **FIE: End of Flush Interrupt Enable Bit**

0: No effect.

1: Enables end of flush interrupt.

- **RBIE: Read Bus Error Interrupt Enable Bit**

0: No effect.

1: Enables read bus error interrupt.

- **WBIE: Write Bus Error Interrupt Enable Bit**

0: No effect.

1: Enables write bus error interrupt.

- **ROIE: Request Overflow Error Interrupt Enable Bit**

0: No effect.

1: Enables request overflow error interrupt.

35.9.19 XDMAC Channel x [x = 0..15] Interrupt Disable Register

Name: XDMAC_CIDx [x = 0..15]

Address: 0xF0004054 (1)[0], 0xF0004094 (1)[1], 0xF00040D4 (1)[2], 0xF0004114 (1)[3], 0xF0004154 (1)[4], 0xF0004194 (1)[5], 0xF00041D4 (1)[6], 0xF0004214 (1)[7], 0xF0004254 (1)[8], 0xF0004294 (1)[9], 0xF00042D4 (1)[10], 0xF0004314 (1)[11], 0xF0004354 (1)[12], 0xF0004394 (1)[13], 0xF00043D4 (1)[14], 0xF0004414 (1)[15], 0xF0010054 (0)[0], 0xF0010094 (0)[1], 0xF00100D4 (0)[2], 0xF0010114 (0)[3], 0xF0010154 (0)[4], 0xF0010194 (0)[5], 0xF00101D4 (0)[6], 0xF0010214 (0)[7], 0xF0010254 (0)[8], 0xF0010294 (0)[9], 0xF00102D4 (0)[10], 0xF0010314 (0)[11], 0xF0010354 (0)[12], 0xF0010394 (0)[13], 0xF00103D4 (0)[14], 0xF0010414 (0)[15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	ROID	WBEID	RBEID	FID	DID	LID	BID

- **BID: End of Block Interrupt Disable Bit**

0: No effect.

1: Disables end of block interrupt.

- **LID: End of Linked List Interrupt Disable Bit**

0: No effect.

1: Disables end of linked list interrupt.

- **DID: End of Disable Interrupt Disable Bit**

0: No effect.

1: Disables end of disable interrupt.

- **FID: End of Flush Interrupt Disable Bit**

0: No effect.

1: Disables end of flush interrupt.

- **RBEID: Read Bus Error Interrupt Disable Bit**

0: No effect.

1: Disables bus error interrupt.

- **WBEID: Write Bus Error Interrupt Disable Bit**

0: No effect.

1: Disables bus error interrupt.

- **ROID: Request Overflow Error Interrupt Disable Bit**

0: No effect.

1: Disables request overflow error interrupt.

35.9.20 XDMAC Channel x [x = 0..15] Interrupt Mask Register

Name: XDMAC_CIMx [x = 0..15]

Address: 0xF0004058 (1)[0], 0xF0004098 (1)[1], 0xF00040D8 (1)[2], 0xF0004118 (1)[3], 0xF0004158 (1)[4], 0xF0004198 (1)[5], 0xF00041D8 (1)[6], 0xF0004218 (1)[7], 0xF0004258 (1)[8], 0xF0004298 (1)[9], 0xF00042D8 (1)[10], 0xF0004318 (1)[11], 0xF0004358 (1)[12], 0xF0004398 (1)[13], 0xF00043D8 (1)[14], 0xF0004418 (1)[15], 0xF0010058 (0)[0], 0xF0010098 (0)[1], 0xF00100D8 (0)[2], 0xF0010118 (0)[3], 0xF0010158 (0)[4], 0xF0010198 (0)[5], 0xF00101D8 (0)[6], 0xF0010218 (0)[7], 0xF0010258 (0)[8], 0xF0010298 (0)[9], 0xF00102D8 (0)[10], 0xF0010318 (0)[11], 0xF0010358 (0)[12], 0xF0010398 (0)[13], 0xF00103D8 (0)[14], 0xF0010418 (0)[15]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	ROIM	WBEIM	RBEIM	FIM	DIM	LIM	BIM

- **BIM: End of Block Interrupt Mask Bit**

0: Block interrupt is masked.

1: Block interrupt is activated.

- **LIM: End of Linked List Interrupt Mask Bit**

0: End of linked list interrupt is masked.

1: End of linked list interrupt is activated.

- **DIM: End of Disable Interrupt Mask Bit**

0: End of disable interrupt is masked.

1: End of disable interrupt is activated.

- **FIM: End of Flush Interrupt Mask Bit**

0: End of flush interrupt is masked.

1: End of flush interrupt is activated.

- **RBEIM: Read Bus Error Interrupt Mask Bit**

0: Bus error interrupt is masked.

1: Bus error interrupt is activated.

- **WBEIM: Write Bus Error Interrupt Mask Bit**

0: Bus error interrupt is masked.

1: Bus error interrupt is activated.

- **ROIM: Request Overflow Error Interrupt Mask Bit**

0: Request overflow interrupt is masked.

1: Request overflow interrupt is activated.

35.9.21 XDMAC Channel x [x = 0..15] Interrupt Status Register

Name: XDMAC_CISx [x = 0..15]

Address: 0xF000405C (1)[0], 0xF000409C (1)[1], 0xF00040DC (1)[2], 0xF000411C (1)[3], 0xF000415C (1)[4], 0xF000419C (1)[5], 0xF00041DC (1)[6], 0xF000421C (1)[7], 0xF000425C (1)[8], 0xF000429C (1)[9], 0xF00042DC (1)[10], 0xF000431C (1)[11], 0xF000435C (1)[12], 0xF000439C (1)[13], 0xF00043DC (1)[14], 0xF000441C (1)[15], 0xF001005C (0)[0], 0xF001009C (0)[1], 0xF00100DC (0)[2], 0xF001011C (0)[3], 0xF001015C (0)[4], 0xF001019C (0)[5], 0xF00101DC (0)[6], 0xF001021C (0)[7], 0xF001025C (0)[8], 0xF001029C (0)[9], 0xF00102DC (0)[10], 0xF001031C (0)[11], 0xF001035C (0)[12], 0xF001039C (0)[13], 0xF00103DC (0)[14], 0xF001041C (0)[15]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	ROIS	WBEIS	RBEIS	FIS	DIS	LIS	BIS

- **BIS: End of Block Interrupt Status Bit**

0: End of block interrupt has not occurred.

1: End of block interrupt has occurred since the last read of the Status register.

- **LIS: End of Linked List Interrupt Status Bit**

0: End of linked list condition has not occurred.

1: End of linked list condition has occurred since the last read of the Status register.

- **DIS: End of Disable Interrupt Status Bit**

0: End of disable condition has not occurred.

1: End of disable condition has occurred since the last read of the Status register.

- **FIS: End of Flush Interrupt Status Bit**

0: End of flush condition has not occurred.

1: End of flush condition has occurred since the last read of the Status register.

- **RBEIS: Read Bus Error Interrupt Status Bit**

0: Read bus error condition has not occurred.

1: At least one bus error has been detected in a read access since the last read of the Status register.

- **WBEIS: Write Bus Error Interrupt Status Bit**

0: Write bus error condition has not occurred.

1: At least one bus error has been detected in a write access since the last read of the Status register.

- **ROIS: Request Overflow Error Interrupt Status Bit**

0: Overflow condition has not occurred.

1: Overflow condition has occurred at least once. (This information is only relevant for peripheral synchronized transfers.)

35.9.22 XDMAC Channel x [x = 0..15] Source Address Register

Name: XDMAC_CSAx [x = 0..15]

Address: 0xF0004060 (1)[0], 0xF00040A0 (1)[1], 0xF00040E0 (1)[2], 0xF0004120 (1)[3], 0xF0004160 (1)[4], 0xF00041A0 (1)[5], 0xF00041E0 (1)[6], 0xF0004220 (1)[7], 0xF0004260 (1)[8], 0xF00042A0 (1)[9], 0xF00042E0 (1)[10], 0xF0004320 (1)[11], 0xF0004360 (1)[12], 0xF00043A0 (1)[13], 0xF00043E0 (1)[14], 0xF0004420 (1)[15], 0xF0010060 (0)[0], 0xF00100A0 (0)[1], 0xF00100E0 (0)[2], 0xF0010120 (0)[3], 0xF0010160 (0)[4], 0xF00101A0 (0)[5], 0xF00101E0 (0)[6], 0xF0010220 (0)[7], 0xF0010260 (0)[8], 0xF00102A0 (0)[9], 0xF00102E0 (0)[10], 0xF0010320 (0)[11], 0xF0010360 (0)[12], 0xF00103A0 (0)[13], 0xF00103E0 (0)[14], 0xF0010420 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
SA							

23	22	21	20	19	18	17	16
SA							

15	14	13	12	11	10	9	8
SA							

7	6	5	4	3	2	1	0
SA							

- **SA: Channel x Source Address**

Program this register with the source address of the DMA transfer.

A configuration error is generated when this address is not aligned with the transfer data size.

35.9.23 XDMAC Channel x [x = 0..15] Destination Address Register

Name: XDMAC_CDAx [x = 0..15]

Address: 0xF0004064 (1)[0], 0xF00040A4 (1)[1], 0xF00040E4 (1)[2], 0xF0004124 (1)[3], 0xF0004164 (1)[4], 0xF00041A4 (1)[5], 0xF00041E4 (1)[6], 0xF0004224 (1)[7], 0xF0004264 (1)[8], 0xF00042A4 (1)[9], 0xF00042E4 (1)[10], 0xF0004324 (1)[11], 0xF0004364 (1)[12], 0xF00043A4 (1)[13], 0xF00043E4 (1)[14], 0xF0004424 (1)[15], 0xF0010064 (0)[0], 0xF00100A4 (0)[1], 0xF00100E4 (0)[2], 0xF0010124 (0)[3], 0xF0010164 (0)[4], 0xF00101A4 (0)[5], 0xF00101E4 (0)[6], 0xF0010224 (0)[7], 0xF0010264 (0)[8], 0xF00102A4 (0)[9], 0xF00102E4 (0)[10], 0xF0010324 (0)[11], 0xF0010364 (0)[12], 0xF00103A4 (0)[13], 0xF00103E4 (0)[14], 0xF0010424 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
DA							

23	22	21	20	19	18	17	16
DA							

15	14	13	12	11	10	9	8
DA							

7	6	5	4	3	2	1	0
DA							

- **DA: Channel x Destination Address**

Program this register with the destination address of the DMA transfer.

A configuration error is generated when this address is not aligned with the transfer data size.

35.9.24 XDMAC Channel x [x = 0..15] Next Descriptor Address Register

Name: XDMAC_CNDAx [x = 0..15]

Address: 0xF0004068 (1)[0], 0xF00040A8 (1)[1], 0xF00040E8 (1)[2], 0xF0004128 (1)[3], 0xF0004168 (1)[4], 0xF00041A8 (1)[5], 0xF00041E8 (1)[6], 0xF0004228 (1)[7], 0xF0004268 (1)[8], 0xF00042A8 (1)[9], 0xF00042E8 (1)[10], 0xF0004328 (1)[11], 0xF0004368 (1)[12], 0xF00043A8 (1)[13], 0xF00043E8 (1)[14], 0xF0004428 (1)[15], 0xF0010068 (0)[0], 0xF00100A8 (0)[1], 0xF00100E8 (0)[2], 0xF0010128 (0)[3], 0xF0010168 (0)[4], 0xF00101A8 (0)[5], 0xF00101E8 (0)[6], 0xF0010228 (0)[7], 0xF0010268 (0)[8], 0xF00102A8 (0)[9], 0xF00102E8 (0)[10], 0xF0010328 (0)[11], 0xF0010368 (0)[12], 0xF00103A8 (0)[13], 0xF00103E8 (0)[14], 0xF0010428 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
NDA							

23	22	21	20	19	18	17	16
NDA							

15	14	13	12	11	10	9	8
NDA							

7	6	5	4	3	2	1	0
NDA						–	NDAIF

- **NDAIF: Channel x Next Descriptor Interface**

0: The channel descriptor is retrieved through the system interface 0.

1: The channel descriptor is retrieved through the system interface 1.

- **NDA: Channel x Next Descriptor Address**

The 30-bit width of the NDA field represents the next descriptor address range 31:2. The descriptor is word-aligned and the two least significant register bits 1:0 are ignored.

35.9.25 XDMAC Channel x [x = 0..15] Next Descriptor Control Register

Name: XDMAC_CNDCx [x = 0..15]

Address: 0xF000406C (1)[0], 0xF00040AC (1)[1], 0xF00040EC (1)[2], 0xF000412C (1)[3], 0xF000416C (1)[4], 0xF00041AC (1)[5], 0xF00041EC (1)[6], 0xF000422C (1)[7], 0xF000426C (1)[8], 0xF00042AC (1)[9], 0xF00042EC (1)[10], 0xF000432C (1)[11], 0xF000436C (1)[12], 0xF00043AC (1)[13], 0xF00043EC (1)[14], 0xF000442C (1)[15], 0xF001006C (0)[0], 0xF00100AC (0)[1], 0xF00100EC (0)[2], 0xF001012C (0)[3], 0xF001016C (0)[4], 0xF00101AC (0)[5], 0xF00101EC (0)[6], 0xF001022C (0)[7], 0xF001026C (0)[8], 0xF00102AC (0)[9], 0xF00102EC (0)[10], 0xF001032C (0)[11], 0xF001036C (0)[12], 0xF00103AC (0)[13], 0xF00103EC (0)[14], 0xF001042C (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	NDVIEW		NDDUP	NDSUP	NDE

- **NDE: Channel x Next Descriptor Enable**

0 (DSCR_FETCH_DIS): Descriptor fetch is disabled.

1 (DSCR_FETCH_EN): Descriptor fetch is enabled.

- **NDSUP: Channel x Next Descriptor Source Update**

0 (SRC_PARAMS_UNCHANGED): Source parameters remain unchanged.

1 (SRC_PARAMS_UPDATED): Source parameters are updated when the descriptor is retrieved.

- **NDDUP: Channel x Next Descriptor Destination Update**

0 (DST_PARAMS_UNCHANGED): Destination parameters remain unchanged.

1 (DST_PARAMS_UPDATED): Destination parameters are updated when the descriptor is retrieved.

- **NDVIEW: Channel x Next Descriptor View**

Value	Name	Description
0	NDV0	Next Descriptor View 0
1	NDV1	Next Descriptor View 1
2	NDV2	Next Descriptor View 2
3	NDV3	Next Descriptor View 3

35.9.26 XDMAC Channel x [x = 0..15] Microblock Control Register

Name: XDMAC_CUBCx [x = 0..15]

Address: 0xF0004070 (1)[0], 0xF00040B0 (1)[1], 0xF00040F0 (1)[2], 0xF0004130 (1)[3], 0xF0004170 (1)[4], 0xF00041B0 (1)[5], 0xF00041F0 (1)[6], 0xF0004230 (1)[7], 0xF0004270 (1)[8], 0xF00042B0 (1)[9], 0xF00042F0 (1)[10], 0xF0004330 (1)[11], 0xF0004370 (1)[12], 0xF00043B0 (1)[13], 0xF00043F0 (1)[14], 0xF0004430 (1)[15], 0xF0010070 (0)[0], 0xF00100B0 (0)[1], 0xF00100F0 (0)[2], 0xF0010130 (0)[3], 0xF0010170 (0)[4], 0xF00101B0 (0)[5], 0xF00101F0 (0)[6], 0xF0010230 (0)[7], 0xF0010270 (0)[8], 0xF00102B0 (0)[9], 0xF00102F0 (0)[10], 0xF0010330 (0)[11], 0xF0010370 (0)[12], 0xF00103B0 (0)[13], 0xF00103F0 (0)[14], 0xF0010430 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
UBLEN							

15	14	13	12	11	10	9	8
UBLEN							

7	6	5	4	3	2	1	0
UBLEN							

- **UBLEN: Channel x Microblock Length**

This field indicates the number of data in the microblock. The microblock contains UBLEN data.

35.9.27 XDMAC Channel x [x = 0..15] Block Control Register

Name: XDMAC_CBCx [x = 0..15]

Address: 0xF0004074 (1)[0], 0xF00040B4 (1)[1], 0xF00040F4 (1)[2], 0xF0004134 (1)[3], 0xF0004174 (1)[4], 0xF00041B4 (1)[5], 0xF00041F4 (1)[6], 0xF0004234 (1)[7], 0xF0004274 (1)[8], 0xF00042B4 (1)[9], 0xF00042F4 (1)[10], 0xF0004334 (1)[11], 0xF0004374 (1)[12], 0xF00043B4 (1)[13], 0xF00043F4 (1)[14], 0xF0004434 (1)[15], 0xF0010074 (0)[0], 0xF00100B4 (0)[1], 0xF00100F4 (0)[2], 0xF0010134 (0)[3], 0xF0010174 (0)[4], 0xF00101B4 (0)[5], 0xF00101F4 (0)[6], 0xF0010234 (0)[7], 0xF0010274 (0)[8], 0xF00102B4 (0)[9], 0xF00102F4 (0)[10], 0xF0010334 (0)[11], 0xF0010374 (0)[12], 0xF00103B4 (0)[13], 0xF00103F4 (0)[14], 0xF0010434 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	BLEN			

7	6	5	4	3	2	1	0
BLEN							

- **BLEN: Channel x Block Length**

The length of the block is (BLEN+1) microblocks.

35.9.28 XDMAC Channel x [x = 0..15] Configuration Register

Name: XDMAC_CCx[x = 0..15]

Address: 0xF0004078 (1)[0], 0xF00040B8 (1)[1], 0xF00040F8 (1)[2], 0xF0004138 (1)[3], 0xF0004178 (1)[4], 0xF00041B8 (1)[5], 0xF00041F8 (1)[6], 0xF0004238 (1)[7], 0xF0004278 (1)[8], 0xF00042B8 (1)[9], 0xF00042F8 (1)[10], 0xF0004338 (1)[11], 0xF0004378 (1)[12], 0xF00043B8 (1)[13], 0xF00043F8 (1)[14], 0xF0004438 (1)[15], 0xF0010078 (0)[0], 0xF00100B8 (0)[1], 0xF00100F8 (0)[2], 0xF0010138 (0)[3], 0xF0010178 (0)[4], 0xF00101B8 (0)[5], 0xF00101F8 (0)[6], 0xF0010238 (0)[7], 0xF0010278 (0)[8], 0xF00102B8 (0)[9], 0xF00102F8 (0)[10], 0xF0010338 (0)[11], 0xF0010378 (0)[12], 0xF00103B8 (0)[13], 0xF00103F8 (0)[14], 0xF0010438 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	PERID						

23	22	21	20	19	18	17	16
WRIP	RDIP	INITD	–	DAM		SAM	

15	14	13	12	11	10	9	8
–	DIF	SIF	DWIDTH		CSIZE		

7	6	5	4	3	2	1	0
MEMSET	SWREQ	PROT	DSYNC	–	MBSIZE		TYPE

- **TYPE: Channel x Transfer Type**

0 (MEM_TRAN): Self-triggered mode (memory-to-memory transfer).

1 (PER_TRAN): Synchronized mode (peripheral-to-memory or memory-to-peripheral transfer).

- **MBSIZE: Channel x Memory Burst Size**

Value	Name	Description
0	SINGLE	The memory burst size is set to one.
1	FOUR	The memory burst size is set to four.
2	EIGHT	The memory burst size is set to eight.
3	SIXTEEN	The memory burst size is set to sixteen.

- **DSYNC: Channel x Synchronization**

0 (PER2MEM): Peripheral-to-memory transfer.

1 (MEM2PER): Memory-to-peripheral transfer.

- **PROT: Channel x Protection**

0 (SEC): Channel is secured.

1 (UNSEC): Channel is unsecured.

- **SWREQ: Channel x Software Request Trigger**

0 (HWR_CONNECTED): Hardware request line is connected to the peripheral request line.

1 (SWR_CONNECTED): Software request is connected to the peripheral request line.

- **MEMSET: Channel x Fill Block of Memory**

0 (NORMAL_MODE): Memset is not activated.

1 (HW_MODE): Sets the block of memory pointed by DA field to the specified value. This operation is performed on 8-, 16- or 32-bit basis.

- **CSIZE: Channel x Chunk Size**

Value	Name	Description
0	CHK_1	1 data transferred
1	CHK_2	2 data transferred
2	CHK_4	4 data transferred
3	CHK_8	8 data transferred
4	CHK_16	16 data transferred

- **DWIDTH: Channel x Data Width**

Value	Name	Description
0	BYTE	The data size is set to 8 bits
1	HALFWORD	The data size is set to 16 bits
2	WORD	The data size is set to 32 bits
3	DWORD	The data size is set to 64 bits

- **SIF: Channel x Source Interface Identifier**

0 (AHB_IF0): The data is read through the system bus interface 0.

1 (AHB_IF1): The data is read through the system bus interface 1.

- **DIF: Channel x Destination Interface Identifier**

0 (AHB_IF0): The data is written through the system bus interface 0.

1 (AHB_IF1): The data is written though the system bus interface 1.

- **SAM: Channel x Source Addressing Mode**

Value	Name	Description
0	FIXED_AM	The address remains unchanged.
1	INCREMENTED_AM	The addressing mode is incremented (the increment size is set to the data size).
2	UBS_AM	The microblock stride is added at the microblock boundary.
3	UBS_DS_AM	The microblock stride is added at the microblock boundary, the data stride is added at the data boundary.

- **DAM: Channel x Destination Addressing Mode**

Value	Name	Description
0	FIXED_AM	The address remains unchanged.
1	INCREMENTED_AM	The addressing mode is incremented (the increment size is set to the data size).
2	UBS_AM	The microblock stride is added at the microblock boundary.
3	UBS_DS_AM	The microblock stride is added at the microblock boundary; the data stride is added at the data boundary.

- **INITD: Channel Initialization Done (this bit is read-only)**

0 (IN_PROGRESS): Channel initialization is in progress.

1 (TERMINATED): Channel initialization is completed.

- **RDIP: Read in Progress (this bit is read-only)**

0 (DONE): No active read transaction on the bus.

1 (IN_PROGRESS): A read transaction is in progress.

- **WRIP: Write in Progress (this bit is read-only)**

0 (DONE): No active write transaction on the bus.

1 (IN_PROGRESS): A write transaction is in progress.

- **PERID: Channel x Peripheral Hardware Request Line Identifier**

This field contains the peripheral hardware request line identifier. PERID refers to identifiers defined in Section 35.4 “DMA Controller Peripheral Connections”.

Note: When a memory-to-memory transfer is performed, configure PERID to an unused peripheral ID (refer to table “Peripheral Identifiers”).

35.9.29 XDMAC Channel x [x = 0..15] Data Stride Memory Set Pattern Register

Name: XDMAC_CDS_MSPx [x = 0..15]

Address: 0xF000407C (1)[0], 0xF00040BC (1)[1], 0xF00040FC (1)[2], 0xF000413C (1)[3], 0xF000417C (1)[4], 0xF00041BC (1)[5], 0xF00041FC (1)[6], 0xF000423C (1)[7], 0xF000427C (1)[8], 0xF00042BC (1)[9], 0xF00042FC (1)[10], 0xF000433C (1)[11], 0xF000437C (1)[12], 0xF00043BC (1)[13], 0xF00043FC (1)[14], 0xF000443C (1)[15], 0xF001007C (0)[0], 0xF00100BC (0)[1], 0xF00100FC (0)[2], 0xF001013C (0)[3], 0xF001017C (0)[4], 0xF00101BC (0)[5], 0xF00101FC (0)[6], 0xF001023C (0)[7], 0xF001027C (0)[8], 0xF00102BC (0)[9], 0xF00102FC (0)[10], 0xF001033C (0)[11], 0xF001037C (0)[12], 0xF00103BC (0)[13], 0xF00103FC (0)[14], 0xF001043C (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
DDS_MSP							
23	22	21	20	19	18	17	16
DDS_MSP							
15	14	13	12	11	10	9	8
SDS_MSP							
7	6	5	4	3	2	1	0
SDS_MSP							

- **SDS_MSP: Channel x Source Data stride or Memory Set Pattern**

When XDMAC_CCx.MEMSET = 0, this field indicates the source data stride.

When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

- **DDS_MSP: Channel x Destination Data Stride or Memory Set Pattern**

When XDMAC_CCx.MEMSET = 0, this field indicates the destination data stride.

When XDMAC_CCx.MEMSET = 1, this field indicates the memory set pattern.

35.9.30 XDMAC Channel x [x = 0..15] Source Microblock Stride Register

Name: XDMAC_CSUSx [x = 0..15]

Address: 0xF0004080 (1)[0], 0xF00040C0 (1)[1], 0xF0004100 (1)[2], 0xF0004140 (1)[3], 0xF0004180 (1)[4], 0xF00041C0 (1)[5], 0xF0004200 (1)[6], 0xF0004240 (1)[7], 0xF0004280 (1)[8], 0xF00042C0 (1)[9], 0xF0004300 (1)[10], 0xF0004340 (1)[11], 0xF0004380 (1)[12], 0xF00043C0 (1)[13], 0xF0004400 (1)[14], 0xF0004440 (1)[15], 0xF0010080 (0)[0], 0xF00100C0 (0)[1], 0xF0010100 (0)[2], 0xF0010140 (0)[3], 0xF0010180 (0)[4], 0xF00101C0 (0)[5], 0xF0010200 (0)[6], 0xF0010240 (0)[7], 0xF0010280 (0)[8], 0xF00102C0 (0)[9], 0xF0010300 (0)[10], 0xF0010340 (0)[11], 0xF0010380 (0)[12], 0xF00103C0 (0)[13], 0xF0010400 (0)[14], 0xF0010440 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
SUBS							

15	14	13	12	11	10	9	8
SUBS							

7	6	5	4	3	2	1	0
SUBS							

- **SUBS: Channel x Source Microblock Stride**

Two’s complement microblock stride for channel x.

35.9.31 XDMAC Channel x [x = 0..15] Destination Microblock Stride Register

Name: XDMAC_CDUSx [x = 0..15]

Address: 0xF0004084 (1)[0], 0xF00040C4 (1)[1], 0xF0004104 (1)[2], 0xF0004144 (1)[3], 0xF0004184 (1)[4], 0xF00041C4 (1)[5], 0xF0004204 (1)[6], 0xF0004244 (1)[7], 0xF0004284 (1)[8], 0xF00042C4 (1)[9], 0xF0004304 (1)[10], 0xF0004344 (1)[11], 0xF0004384 (1)[12], 0xF00043C4 (1)[13], 0xF0004404 (1)[14], 0xF0004444 (1)[15], 0xF0010084 (0)[0], 0xF00100C4 (0)[1], 0xF0010104 (0)[2], 0xF0010144 (0)[3], 0xF0010184 (0)[4], 0xF00101C4 (0)[5], 0xF0010204 (0)[6], 0xF0010244 (0)[7], 0xF0010284 (0)[8], 0xF00102C4 (0)[9], 0xF0010304 (0)[10], 0xF0010344 (0)[11], 0xF0010384 (0)[12], 0xF00103C4 (0)[13], 0xF0010404 (0)[14], 0xF0010444 (0)[15]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
DUBS							

15	14	13	12	11	10	9	8
DUBS							

7	6	5	4	3	2	1	0
DUBS							

- **DUBS: Channel x Destination Microblock Stride**

Two’s complement microblock stride for channel x.

36. LCD Controller (LCDC)

36.1 Description

The LCD Controller (LCDC) consists of logic for transferring LCD image data from an external display buffer to an LCD module. The LCD has one display input buffer per overlay that fetches pixels through the dual AHB master interface and a lookup table to allow palletized display configurations. The LCD controller is programmable on a per overlay basis, and supports different LCD resolutions, window sizes, image formats and pixel depths.

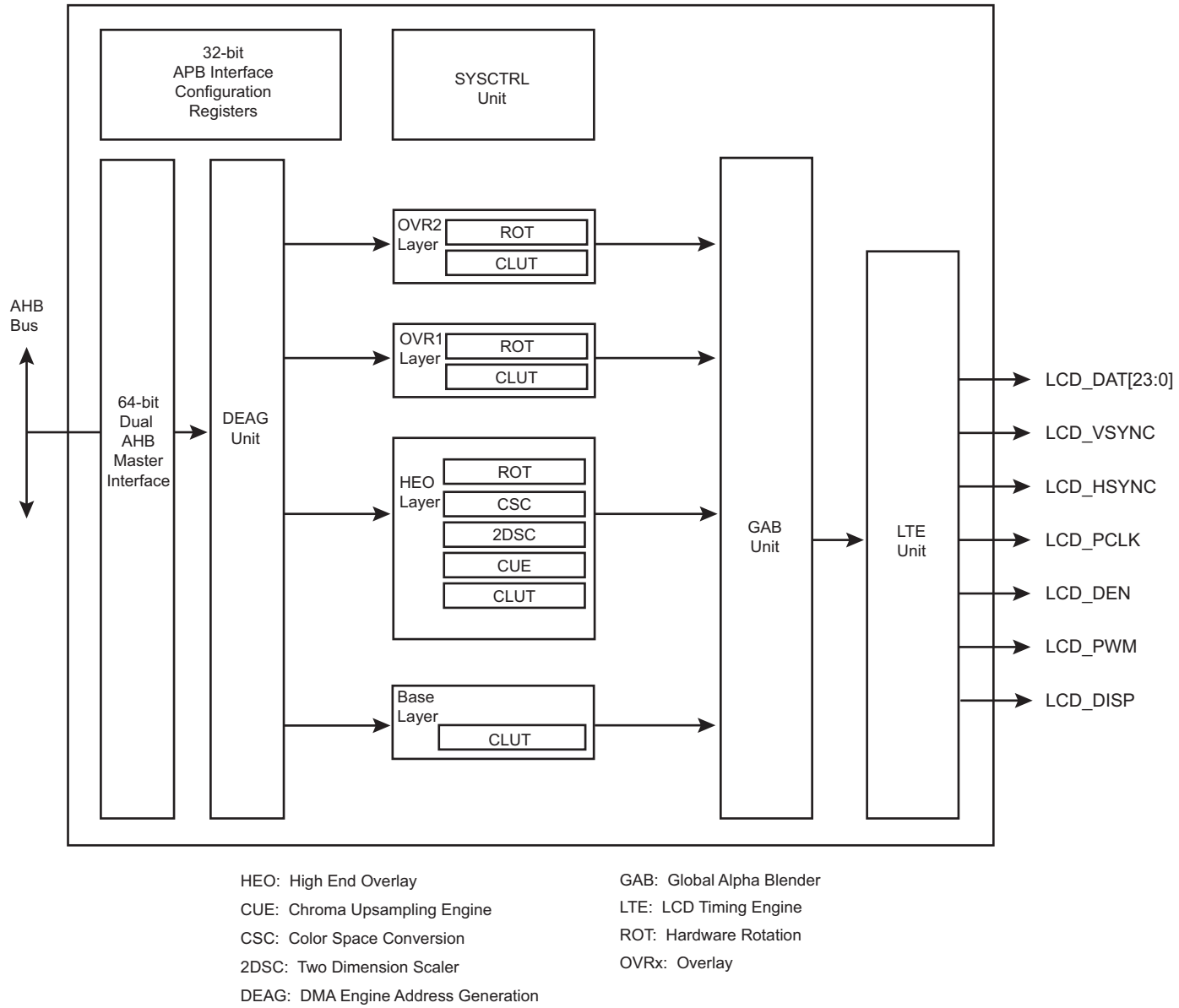
The LCD is connected to the ARM Advanced High Performance Bus (AHB) as a master for reading pixel data. It also integrates an APB interface to configure its registers.

36.2 Embedded Characteristics

- Dual AHB Master Interface
- Supports Single Scan Active TFT Display
- Supports 12-, 16-, 18- and 24-bit Output Mode through the Spatial Dithering Unit
- Asynchronous Output Mode Supported (at synthesis time)
- 1, 2, 4, 8 bits per Pixel (Palletized)
- 12, 16, 18, 19, 24, 25 and 32 bits per Pixel (Non-palletized)
- Supports One Base Layer (Background)
- Supports One Overlay 1 Layer Window
- Supports One Overlay 2 Layer Window
- Supports One High End Overlay (HEO) Window
- Little Endian Memory Organization
- Programmable Timing Engine, with Integer Clock Divider
- Programmable Polarity for Data, Line Synchro and Frame Synchro
- Up to 1024x768 (XGA) with Overlay (Application-Dependent). Still Image up to WXGA.
- Color Lookup Table with up to 256 Entries and Predefined 8-bit Alpha
- Programmable Negative and Positive Row Striding for all Layers
- Programmable Negative and Positive Pixel Striding for Layers
- High End Overlay supports 4:2:0 Planar Mode and Semiplanar Mode
- High End Overlay supports 4:2:2 Planar Mode, Semiplanar Mode and Packed
- High End Overlay includes Chroma Upsampling Unit
- Horizontal and Vertical Rescaling Unit with Edge Interpolation and Independent Non-Integer Ratio, up to 1024x768
- Hidden Layer Removal supported
- Integrates Fully Programmable Color Space Conversion
- Blender Function Supports Arbitrary 8-bit Alpha Value and Chroma Keying
- DMA User Interface uses Linked List Structure and Add-to-queue Structure

36.3 Block Diagram

Figure 36-1. Block Diagram



HEO: High End Overlay
CUE: Chroma Upsampling Engine
CSC: Color Space Conversion
2DSC: Two Dimension Scaler
DEAG: DMA Engine Address Generation

GAB: Global Alpha Blender
LTE: LCD Timing Engine
ROT: Hardware Rotation
OVRx: Overlay

36.4 I/O Lines Description

Table 36-1. I/O Lines Description

Name	Description	Type
LCD_PWM	Contrast control signal, using Pulse Width Modulation	**Output**
LCD_HSYNC	Horizontal Synchronization Pulse	Output
LCD_VSYNC	Vertical Synchronization Pulse	Output
LCD_DAT[23:0]	LCD 24-bit data bus	Output
LCD_DEN	Data Enable	Output

Table 36-1. I/O Lines Description (Continued)

Name	Description	Type
LCD_DISP	Display Enable signal	Output
LCD_PCLK	Pixel Clock	Output

36.5 Product Dependencies

36.5.1 I/O Lines

The pins used for interfacing the LCD Controller may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the LCD Controller are not used by the application, they can be used for other purposes by the PIO Controller.

Table 36-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
LCDC	LCDDAT0	PB11	A
LCDC	LCDDAT1	PB12	A
LCDC	LCDDAT2	PB13	A
LCDC	LCDDAT2	PC10	A
LCDC	LCDDAT3	PB14	A
LCDC	LCDDAT3	PC11	A
LCDC	LCDDAT4	PB15	A
LCDC	LCDDAT4	PC12	A
LCDC	LCDDAT5	PB16	A
LCDC	LCDDAT5	PC13	A
LCDC	LCDDAT6	PB17	A
LCDC	LCDDAT6	PC14	A
LCDC	LCDDAT7	PB18	A
LCDC	LCDDAT7	PC15	A
LCDC	LCDDAT8	PB19	A
LCDC	LCDDAT9	PB20	A
LCDC	LCDDAT10	PB21	A
LCDC	LCDDAT10	PC16	A
LCDC	LCDDAT11	PB22	A
LCDC	LCDDAT11	PC17	A
LCDC	LCDDAT12	PB23	A
LCDC	LCDDAT12	PC18	A
LCDC	LCDDAT13	PB24	A
LCDC	LCDDAT13	PC19	A
LCDC	LCDDAT14	PB25	A
LCDC	LCDDAT14	PC20	A
LCDC	LCDDAT15	PB26	A

Table 36-2. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
LCDC	LCDDAT15	PC21	A
LCDC	LCDDAT16	PB27	A
LCDC	LCDDAT17	PB28	A
LCDC	LCDDAT18	PB29	A
LCDC	LCDDAT18	PC22	A
LCDC	LCDDAT19	PB30	A
LCDC	LCDDAT19	PC23	A
LCDC	LCDDAT20	PB31	A
LCDC	LCDDAT20	PC24	A
LCDC	LCDDAT21	PC0	A
LCDC	LCDDAT21	PC25	A
LCDC	LCDDAT22	PC1	A
LCDC	LCDDAT22	PC26	A
LCDC	LCDDAT23	PC2	A
LCDC	LCDDAT23	PC27	A
LCDC	LCDDEN	PC8	A
LCDC	LCDDEN	PD1	A
LCDC	LCDDISP	PC4	A
LCDC	LCDDISP	PC29	A
LCDC	LCDHSYNC	PC6	A
LCDC	LCDHSYNC	PC31	A
LCDC	LCDPCK	PC7	A
LCDC	LCDPCK	PD0	A
LCDC	LCDPWM	PC3	A
LCDC	LCDPWM	PC28	A
LCDC	LCDVSYNC	PC5	A
LCDC	LCDVSYNC	PC30	A

36.5.2 Power Management

The LCD Controller is not continuously clocked. The user must first enable the LCD Controller clock in the Power Management Controller (PMC_PCER) before using it.

36.5.3 Interrupt Sources

The LCD Controller interrupt line is connected to one of the internal sources of the interrupt controller. Using the LCD Controller interrupt requires prior programming of the interrupt controller.

Table 36-3. Peripheral IDs

Instance	ID
LCDC	45

36.6 Functional Description

The LCD module integrates the following digital blocks:

- DMA Engine Address Generation (DEAG)—This block performs data prefetch and requests access to the AHB interface.
- Input Overlay FIFO—stores the stream of pixels
- Color Lookup Table (CLUT)—These 256 RAM-based lookup table entries are selected when the color depth is set to 1, 2, 4 or 8 bpp.
- Chroma Upsampling Engine (CUE)—This block is selected when the input image sampling format is YUV (Y'CbCr) 4:2:0 and converts it to higher quality 4:4:4 image.
- Color Space Conversion (CSC)—changes the color spare from YUV to RGB
- Two Dimension Scaler (2DSC)—resizes the image
- Global Alpha Blender (GAB)—performs programmable 256-level alpha blending
- Output FIFO—stores the blended pixel prior to display
- LCD Timing Engine—provides a fully programmable HSYNC-VSYNC interface

The DMA controller reads the image through the AHB master interface. The LCD controller engine formats the display data, then the GAB performs alpha blending if required, and writes the final pixel into the output FIFO. The programmable timing engine drives a valid pixel onto the LCD_DAT[23:0] display bus.

36.6.1 Timing Engine Configuration

36.6.1.1 Pixel Clock Period Configuration

The pixel clock (LCD_PCLK) generated by the timing engine is the source clock divided by the field CLKDIV in the LCDC_LCDCFG0 register. The source clock can be selected between the system clock and the 2x system clock with the field CLKSEL located in the LCDC_LCDCFG0 register.

Pixel clock period formula:

$$\text{LCD_PCLK} = \frac{\text{source clock}}{\text{CLKDIV} + 2}$$

The pixel clock polarity is also programmable.

36.6.1.2 Horizontal and Vertical Synchronization Configuration

The following fields are used to configure the timing engine:

- LCDC_LCDCFG1.HSPW
- LCDC_LCDCFG1.VSPW
- LCDC_LCDCFG2.VFPW
- LCDC_LCDCFG2.VBPW
- LCDC_LCDCFG3.HFPW
- LCDC_LCDCFG3.HBPW
- LCDC_LCDCFG4.PPL
- LCDC_LCDCFG4.RPF

The polarity of output signals is also programmable.

36.6.1.3 Timing Engine Power Up Software Operation

The following sequence is used to enable the display:

1. Configure LCD timing parameters, signal polarity and clock period.
2. Enable the pixel clock by writing a one to bit LCDC_LCDEN.CLKEN.
3. Poll bit LCDC_LCDSR.CLKSTS to check that the clock is running.
4. Enable Horizontal and Vertical Synchronization by writing a one to bit LCDC_LCDEN.SYNCEN.

5. Poll bit LCDC_LCDSR.LCDSTS to check that the synchronization is up.
6. Enable the display power signal by writing a one to bit LCDC_LCDEN.DISPEN.
7. Poll bit LCDC_LCDSR.DISPSTS to check that the power signal is activated.

The field LCDC_LCDCFG5.GUARDTIME is used to configure the number of frames before the assertion of the DISP signal.

36.6.1.4 Timing Engine Power Down Software Operation

The following sequence is used to disable the display:

1. Disable the DISP signal by writing bit LCDC_LCDDIS.DISPDIS.
2. Poll bit LCDC_LCDSR.DISPSTS to verify that the DISP is no longer activated.
3. Disable the HSYNC and VSYNC signals by writing a one to bit LCDC_LCDDIS.SYNCDIS.
4. Poll bit LCDC_LCDSR.LCDSTS to check that the synchronization is off.
5. Disable the pixel clock by writing a one to bit LCDC_LCDDIS.CLKDIS.

36.6.2 DMA Software Operations

36.6.2.1 DMA Channel Descriptor (DSCR) Alignment and Structure

The DMA Channel Descriptor (DSCR) must be aligned on a 64-bit boundary.

The DMA Channel Descriptor structure contains three fields:

- DSCR.CHXADDR: Frame Buffer base address register
- DSCR.CHXCTRL: Transfer Control register
- DSCR.CHXNEXT: Next Descriptor Address register

Table 36-4. DMA Channel Descriptor Structure

System Memory	Structure Field for Channel CHX
DSCR + 0x0	ADDR
DSCR + 0x4	CTRL
DSCR + 0x8	NEXT

36.6.2.2 Enabling a DMA Channel

Follow the steps below to enable a DMA channel:

1. Check the status of the channel by reading the CHXCHSR register.
2. Write the channel descriptor (DSCR) structure in the system memory by writing DSCR.CHXADDR Frame base address, DSCR.CHXCTRL channel control and DSCR.CHXNEXT next descriptor location.
3. If more than one descriptor is expected, the field DFETCH of DSCR.CHXCTRL is set to ‘1’ to enable the descriptor fetch operation.
4. Write the DSCR.CHXNEXT register with the address location of the descriptor structure and set DFETCH field of the DSCR.CHXCTRL register to ‘1’.
5. Enable the relevant channel by writing one to the CHEN field of the CHXCHER register.
6. An interrupt may be raised if unmasked when the descriptor has been loaded.

36.6.2.3 Disabling a DMA Channel

Follow the steps below to disable a DMA channel:

1. Clearing the DFETCH bit in the DSCR.CHXCTRL field of the DSCR structure disables the channel at the end of the frame.
2. Setting the DSCR.CHXNEXT field of the DSCR structure disables the channel at the end of the frame.
3. Writing one to the CHDIS field of the CHXCHDR register disables the channel at the end of the frame.

4. Writing one to the CHRST field of the CHXCHDR register disables the channel immediately. This may occur in the middle of the image.
5. Polling CHSR field in the CHXCHSR register until the channel is successfully disabled.

36.6.2.4 DMA Dynamic Linking of a New Transfer Descriptor

1. Write the new descriptor structure in the system memory.
2. Write the address of the new structure in the CHXHEAD register.
3. Add the new structure to the queue of descriptors by writing one to the A2QEN field of the CHXCHER register.
4. The new descriptor will be added to the queue on the next frame.
5. An interrupt will be raised if unmasked, when the head descriptor structure has been loaded by the DMA channel.

36.6.2.5 DMA Interrupt Generation

The DMA Controller operation sets the following interrupt flags in the Interrupt Status register CHXISR:

- DMA field indicates that the DMA transfer is completed.
- DSCR field indicates that the descriptor structure is loaded in the DMA controller.
- ADD field indicates that a descriptor has been added to the descriptor queue.
- DONE field indicates that the channel transfer has terminated and the channel is automatically disabled.

36.6.2.6 DMA Address Alignment Requirements

When programming the DSCR.CHXADDR field of the DSCR structure, the following requirement must be met.

Table 36-5. DMA Address Alignment when CLUT Mode is Selected

CLUT Mode	DMA Address Alignment
1 bpp	8 bits
2 bpp	8 bits
4 bpp	8 bits
8 bpp	8 bits

Table 36-6. DMA Address Alignment when RGB Mode is Selected

RGB Mode	DMA Address Alignment
12 bpp RGB 444	16 bits
16 bpp ARGB 4444	16 bits
16 bpp RGBA 4444	16 bits
16 bpp RGB 565	16 bits
16 bpp TRGB 1555	16 bits
18 bpp RGB 666	32 bits
18 bpp RGB 666 PACKED	8 bits
19 bpp TRGB 1666	32 bits
19 bpp TRGB 1666	8 bits
24 bpp RGB 888	32 bits
24 bpp RGB 888 PACKED	8 bits
25 bpp TRGB 1888	32 bits
32 bpp ARGB 8888	32 bits
32 bpp RGBA 8888	32 bits

Table 36-7. DMA Address Alignment when YUV Mode is Selected

YUV Mode	DMA Address Alignment
32 bpp AYCrCb	32 bits
16 bpp YCrCb 4:2:2	32 bits
16 bpp semiplanar YCrCb 4:2:2	Y 8 bits
	CrCb 16 bits
16 bpp planar YCrCb 4:2:2	Y 8 bits
	Cr 8 bits
	Cb 8 bits
12 bpp YCrCb 4:2:0	Y 8 bits
	CrCb 16 bits
12 bpp YCrCb 4:2:0	Y 8 bits
	Cr 8 bits
	Cb 8 bits

36.6.3 Overlay Software Configuration

36.6.3.1 System Bus Access Attributes

These attributes are defined to improve bandwidth of the overlay.

- LOCKDIS bit—When set to ‘1’, the AHB lock signal is not asserted when the PSTRIDE value is different from zero (rotation in progress).
- ROTDIS bit—When set to ‘1’, the Pixel Striding optimization is disabled.
- DLBO bit—When set to ‘1’, only defined burst lengths are performed when the DMA channel retrieves the data from the memory.
- BLEN field—Defines the maximum burst length of the DMA channel.
- SIF bit—Defines the targeted DMA interface.

36.6.3.2 Color Attributes

- CLUTMODE field—Selects the Color Lookup Table mode.
- RGBMODE field—Selects the RGB mode.
- YUVMODE field—Selects the Luminance Chrominance mode.

36.6.3.3 Window Position, Size, Scaling and Striding Attributes

- XPOS and YPOS fields—Define the position of the overlay window.
- XSIZE and YSIZE fields—Define the size of the displayed window.
- XMEMSIZE and YMEMSIZE fields—Define the size of the image frame buffer.
- XSTRIDE and PSTRIDE fields—Define the line and pixel striding.
- XFACTOR and YFACTOR fields—Define the scaling ratio.

The position and size attributes are to be programmed to keep the window within the display area.

When the Color Lookup Table mode is enabled, the restrictions detailed in the following table apply on the horizontal and vertical window sizes.

Table 36-8. Color Lookup Table Mode and Window Size

CLUT Mode	X-Y Size Requirement
1 bpp	Multiple of 8 pixels
2 bpp	Multiple of 4 pixels
4 bpp	Multiple of 2 pixels
8 bpp	Free size

Pixel striding is disabled when CLUT mode is enabled.

When YUV mode is enabled, the restrictions detailed in the following table apply on the window size.

Table 36-9. YUV Mode and Window Size

YUV Mode	X-Y Requirement, Scaling Turned Off	X-Y Requirement, Scaling Turned On
AYUV	Free size	X-Y size is greater than 5
YUV 4:2:2 packed	XSIZE is greater than 2 pixels	X-Y size is greater than 5
YUV 4:2:2 semiplanar	XSIZE is greater than 2 pixels	X-Y size is greater than 5
YUV 4:2:2 planar	XSIZE is greater than 2 pixels	X-Y size is greater than 5
YUV 4:2:0 semiplanar	XSIZE is greater that 2 pixels	X-Y size is greater than 5
YUV 4:2:0 planar	XSIZE is greater than 2 pixels	X-Y size is greater than 5

In RGB mode, there is no restriction on the line length.

36.6.3.4 Overlay Blender Attributes

When two or more video layers are used, alpha blending is performed to define the final image displayed. Each window has its own blending attributes.

- CRKEY bit—Enables the chroma keying and match logic.
- INV bit—Performs bit inversion at pixel level.
- ITER2BL bit—When written to ‘1’, the iterated data path is selected.
- ITER bit—When written to ‘1’, the iterated value is used in the iterated data path, otherwise the iterated value is set to 0.
- REVALPHA bit—Uses the reverse alpha value.
- GAEN bit—Enables the global alpha value in the data path.
- LAEN bit—Enables the local alpha value from the pixel.
- OVR bit—When written to ‘1’, the overlay is selected as an input of the blender.
- DMA bit—The DMA data path is activated.
- REP bit—Enables the bit replication to fill the 24-bit internal data path.
- DSTKEY bit—When written to ‘1’, Destination keying is enabled.
- GA field—Defines the global alpha value.

36.6.3.5 Overlay Attributes Software Operation

1. When required, write the overlay attributes configuration registers.
2. Set UPDATEEN field of the CHXCHER register.
3. Poll UPDATESR field in the CHXCHSR, the update applies when that field is reset.

36.6.4 RGB Frame Buffer Memory Bitmap

36.6.4.1 1 bpp Through Color Lookup Table

Table 36-10. 1 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 1 bpp	p31	p30	p29	p28	p27	p26	p25	p24	p23	p22	p21	p20	p19	p18	p17	p16	p15	p14	p13	p12	p11	p10	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0

36.6.4.2 2 bpp Through Color Lookup Table

Table 36-11. 2 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 2 bpp	p15		p14		p13		p12		p11		p10		p9		p8		p7		p6		p5		p4		p3		p2		p1		p0	

36.6.4.3 4 bpp Through Color Lookup Table

Table 36-12. 4 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 4 bpp	p7				p6				p5				p4				p3				p2				p1				p0			

36.6.4.4 8 bpp Through Color Lookup Table

Table 36-13. 8 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 8 bpp	p3								p2								p1								p0							

36.6.4.5 12 bpp Memory Mapping, RGB 4:4:4

Table 36-14. 12 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	–				R1[3:0]				G1[3:0]				B1[3:0]				–				R0[3:0]				G0[3:0]				B0[3:0]			

36.6.4.6 16 bpp Memory Mapping with Alpha Channel, ARGB 4:4:4:4

Table 36-15. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	A1[3:0]				R1[3:0]				G1[3:0]				B1[3:0]				A0[3:0]				R0[3:0]				G0[3:0]				B0[3:0]			

36.6.4.7 16 bpp Memory Mapping with Alpha Channel, RGBA 4:4:4:4

Table 36-16. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	R1[3:0]				G13:0]				B1[3:0]				A1[3:0]				R0[3:0]				G0[3:0]				B0[3:0]				A0[3:0]			

36.6.4.8 16 bpp Memory Mapping with Alpha Channel, RGB 5:6:5

Table 36-17. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16bpp	R1[4:0]					G1[5:0]				B1[4:0]							R0[4:0]					G0[5:0]						B0[4:0]			

36.6.4.9 16 bpp Memory Mapping with Transparency Bit, ARGB 1:5:5:5

Table 36-18. 16 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 4 bpp	A1	R1[4:0]				G1[4:0]				B1[4:0]							A0	R0[4:0]				G0[4:0]				B0[4:0]							

36.6.4.10 18 bpp Unpacked Memory Mapping with Transparency Bit, RGB 6:6:6

Table 36-19. 18 bpp Unpacked Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	–								–						R0[5:0]						G0[5:0]						B0[5:0]					

36.6.4.11 18 bpp Packed Memory Mapping with Transparency Bit, RGB 6:6:6

Table 36-20. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	G1[1:0]		B1[5:0]						–						R0[5:0]						G0[5:0]						B0[5:0]					

Table 36-21. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	R2[3:0]				G2[5:0]				B2[5:0]								–						R1[5:2]						G1[5:2]			

Table 36-22. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x8, 0x9, 0xA, 0xB

Mem addr	0xB								0xA								0x9								0x8							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 18 bpp	G4[1:0]		B4[5:0]						–				R3[5:0]				G3[5:0]								B3[3:0]						R2[5:4]	

36.6.4.12 19 bpp Unpacked Memory Mapping with Transparency Bit, RGB 1:6:6:6

Table 36-23. 19 bpp Unpacked Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	–								–					A0	R0[5:0]						G0[5:0]				B0[5:0]							

36.6.4.13 19 bpp Packed Memory Mapping with Transparency Bit, ARGB 1:6:6:6

Table 36-24. 19 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	G1[1:0]		B1[5:0]						–					A0	R0[5:0]						G0[5:0]				B0[5:0]							

Table 36-25. 19 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	R2[3:0]				G2[5:0]				B2[5:0]								–					A1	R1[5:2]						G1[5:2]			

Table 36-26. 18 bpp Packed Memory Mapping, Little Endian Organization at Address 0x8, 0x9, 0xA, 0xB

Mem addr	0xB								0xA								0x9								0x8							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 19 bpp	G4[1:0]		B4[5:0]						–			A3	R3[5:0]				G3[5:0]								B3[3:0]						R2[5:4]	

36.6.4.14 24 bpp Unpacked Memory Mapping, RGB 8:8:8

Table 36-27. 24 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 24 bpp	–								R0[7:0]								G0[7:0]								B0[7:0]							

36.6.4.15 24 bpp Packed Memory Mapping, RGB 8:8:8

Table 36-28. 24 bpp Packed Memory Mapping, Little Endian Organization at Address 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 24 bpp	B1[7:0]								R0[7:0]								G0[7:0]								B0[7:0]							

Table 36-29. 24 bpp Packed Memory Mapping, Little Endian Organization at Address 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 24 bpp	G2[7:0]								B2[7:0]								R1[7:0]								G1[7:0]							

36.6.4.16 25 bpp Memory Mapping, ARGB 1:8:8:8

Table 36-30. 25 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 25 bpp	–							A0	R0[7:0]								G0[7:0]								B0[7:0]							

36.6.4.17 32 bpp Memory Mapping, ARGB 8:8:8:8

Table 36-31. 32 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 32 bpp	A0[7:0]								R0[7:0]								G0[7:0]								B0[7:0]							

36.6.4.18 32 bpp Memory Mapping, RGBA 8:8:8:8

Table 36-32. 32 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 32 bpp	R0[7:0]								G0[7:0]								B0[7:0]								A0[7:0]							

36.6.5 YUV Frame Buffer Memory Mapping

36.6.5.1 AYCbCr 4:4:4 Interleaved Frame Buffer Memory Mapping

Table 36-33. 32 bpp Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	A0[7:0]								Y0[7:0]								Cb0[7:0]								Cr0[7:0]							

36.6.5.2 4:2:2 Interleaved Mode Frame Buffer Memory Mapping

Table 36-34. 16 bpp 4:2:2 interleaved Mode 0

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Cr0[7:0]								Y1[7:0]								Cb0[7:0]								Y0[7:0]							

Table 36-35. 16 bpp 4:2:2 interleaved Mode 1

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y1[7:0]								Cr0[7:0]								Y0[7:0]								Cb0[7:0]							

Table 36-36. 16 bpp 4:2:2 interleaved Mode 2

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Cb0[7:0]								Y1[7:0]								Cr0[7:0]								Y0[7:0]							

Table 36-37. 16 bpp 4:2:2 interleaved Mode 3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y1[7:0]								Cb0[7:0]								Y0[7:0]								Cr0[7:0]							

36.6.5.3 4:2:2 Semiplanar Mode Frame Buffer Memory Mapping

Table 36-38. 4:2:2 Semiplanar Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 36-39. 4:2:2 Semiplanar Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Cb2[7:0]								Cr2[7:0]								Cb0[7:0]								Cr0[7:0]							

36.6.5.4 4:2:2 Planar Mode Frame Buffer Memory Mapping

Table 36-40. 4:2:2 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 36-41. 4:2:2 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 16 bpp	C3[7:0]								C2[7:0]								C1[7:0]								C0[7:0]							

36.6.5.5 4:2:0 Planar Mode Frame Buffer Memory Mapping

In Planar Mode, the three video components Y, Cr and Cb are split into three memory areas and stored in a raster-scan order. These three memory planes are contiguous and always aligned on a 32-bit boundary.

Table 36-42. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 36-43. 4:2:0 Planar Mode Luminance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y7[7:0]								Y6[7:0]								Y5[7:0]								Y4[7:0]							

Table 36-44. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x0, 0x1, 0x2, 0x3

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	C3[7:0]								C2[7:0]								C1[7:0]								C0[7:0]							

Table 36-45. 4:2:0 Planar Mode Chrominance Memory Mapping, Little Endian Organization for Byte 0x4, 0x5, 0x6, 0x7

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	C7[7:0]								C6:[7:0]								C5[7:0]								C4[7:0]							

36.6.5.6 4:2:0 Semiplanar Frame Buffer Memory Mapping

Table 36-46. 4:2:0 Semiplanar Mode Luminance Memory Mapping, Little Endian Organization

Mem addr	0x7								0x6								0x5								0x4							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Y3[7:0]								Y2[7:0]								Y1[7:0]								Y0[7:0]							

Table 36-47. 4:2:0 Semiplanar Mode Chrominance Memory Mapping, Little Endian Organization

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Pixel 12 bpp	Cb1[7:0]								Cr1[7:0]								Cb0[7:0]								Cr0[7:0]							

36.6.6 Chrominance Upsampling Unit

Both 4:2:2 and 4:2:0 input formats are supported by the LCD module. In 4:2:2, the two chrominance components are sampled at half the sample rate of the luminance. The horizontal chrominance resolution is halved. When this input format is selected, the chrominance upsampling unit uses two chrominances to interpolate the missing component.

In 4:2:0, Cr and Cb components are subsampled at a factor of two vertically and horizontally. When this input mode is selected, the chrominance upsampling unit uses two and four chroma components to generate the missing horizontal and vertical components.

Figure 36-2. 4:2:2 Upsampling Algorithm

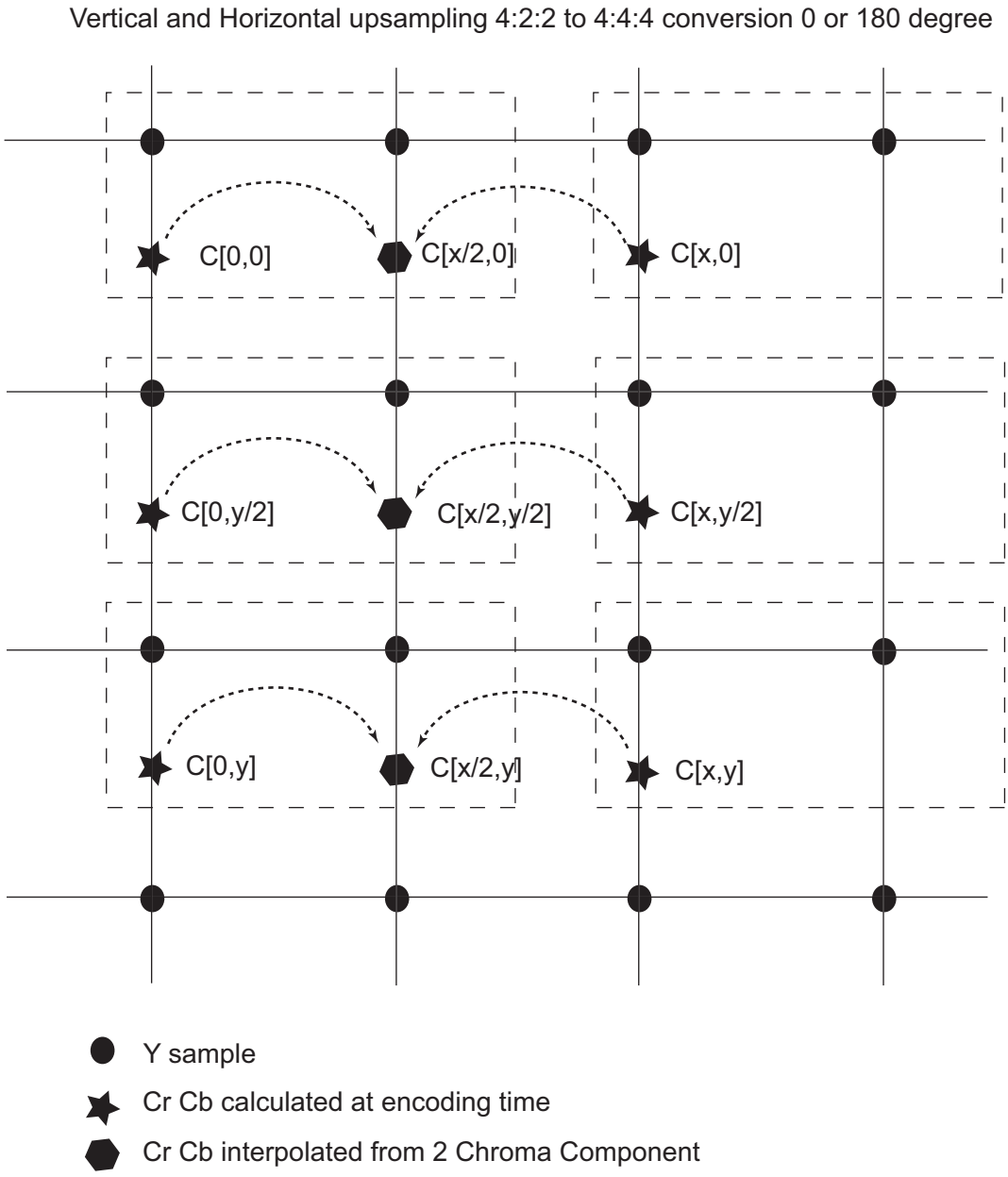


Figure 36-3. 4:2:2 Packed Upsampling Algorithm

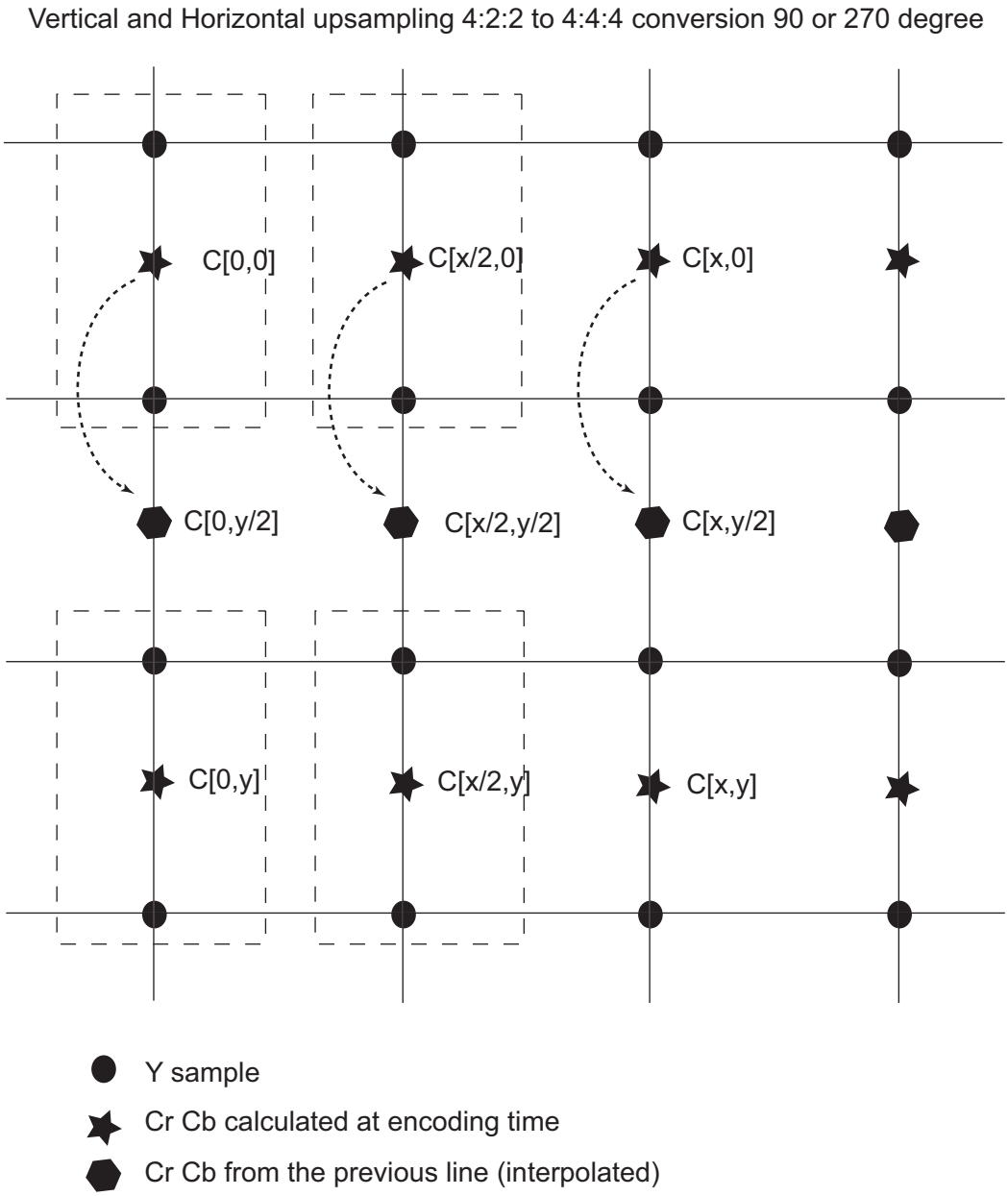


Figure 36-4. 4:2:2 Semiplanar and Planar Upsampling Algorithm - 90 or 270 Degree R Rotation Activated

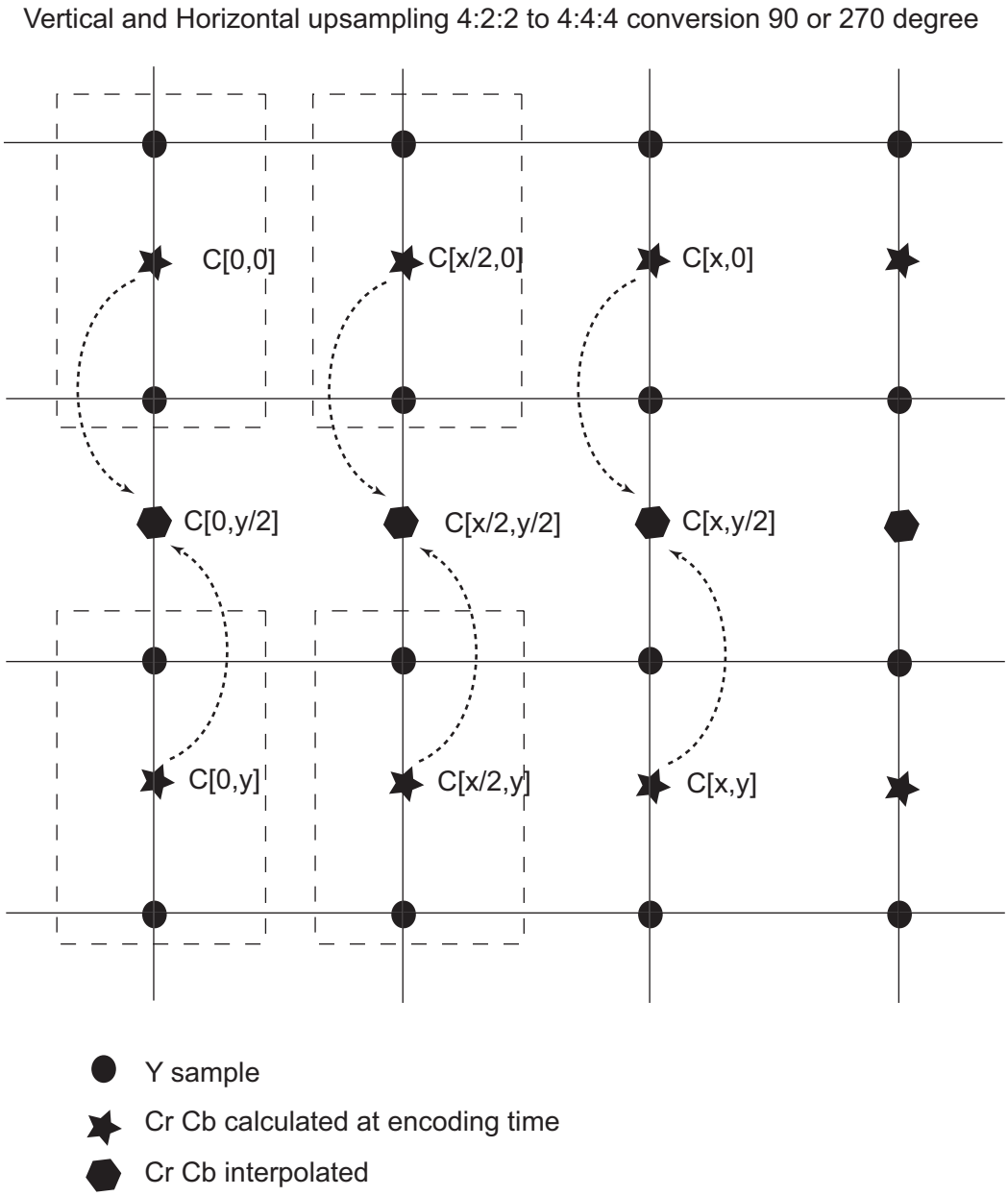
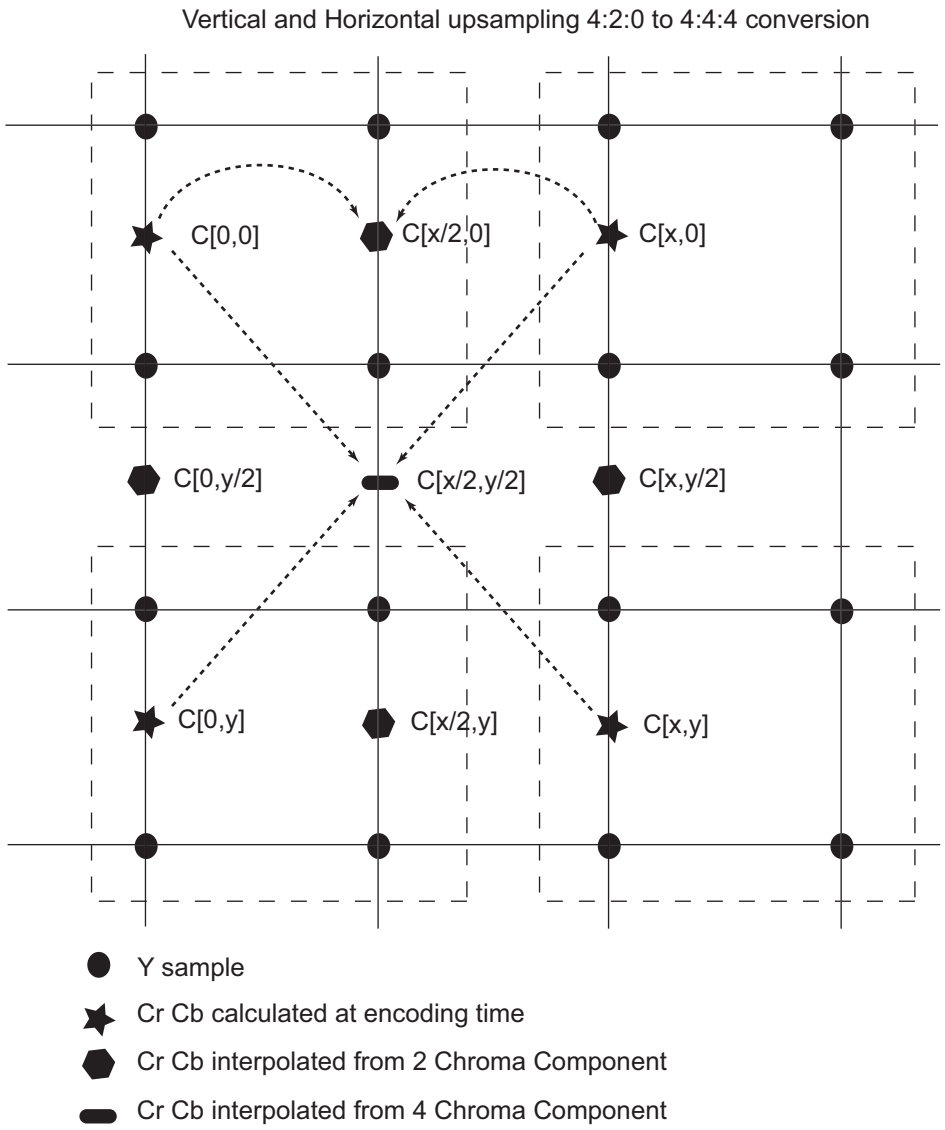


Figure 36-5. 4:2:0 Upsampling Algorithm



$$Chroma\left[\frac{x}{2}, 0\right] = \frac{Cr[0,0] + Cr[0,x]}{2}$$

$$Chroma\left[0, \frac{y}{2}\right] = \frac{Cr[0,0] + C[0,y]}{2}$$

$$Chroma\left[\frac{x}{2}, \frac{y}{2}\right] = \frac{Cr[0,0] + Cr[x,0] + Cr[y,0] + Cr[x,y]}{4}$$

$$Chroma\left[x, \frac{y}{2}\right] = \frac{Cr[x,0] + Cr[x,y]}{2}$$

$$Chroma\left[\frac{x}{2}, y\right] = \frac{Cr[0,y] + Cr[x,y]}{2}$$

36.6.6.1 Chrominance Upsampling Algorithm

1. Read line n from chrominance cache and interpolate [x/2,0] chrominance component filling the 1 x 2 kernel with line n. If the chrominance cache is empty, then fetch the first line from external memory and interpolate from the external memory. Duplicate the last chrominance at the end of line.
2. Fetch line n+1 from external memory, write line n + 1 to chrominance cache, read line n from the chrominance cache. interpolate [0,y/2], [x/2,y/2] and [x, y/2] filling the 2x2 kernel with line n and n+1. Duplicate the last chrominance line to generate the last interpolated line.
3. Repeat step 1 and step 2.

36.6.7 Line and Pixel Striding

The LCD module includes a technique to increment the memory address by a programmable amount when the end of line has been reached. This offset is referred to as XSTRIDE and is defined on a per overlay basis. Additionally, the PSTRIDE field allows a programmable jump at the pixel level. Pixel stride is the value from one pixel to the next.

36.6.7.1 Line Striding

When the end of line has been reached, the DMA address counter points to the next pixel address. The channel DMA address register is added to the XSTRIDE field, and then updated. If XSTRIDE is set to '0', the DMA address register remains unchanged. The XSTRIDE field of the channel configuration register is aligned to the pixel size boundary. The XSTRIDE field is a two's complement number. The following formula applies at the line boundary and indicates how the DMA controller computes the next pixel address. The function Sizeof() returns the number of bytes required to store a pixel.

$$NextPixelAddress = CurrentPixelAddress + Sizeof(pixel) + XSTRIDE$$

36.6.7.2 Pixel Striding

The DMA channel engine may optionally fetch non-contiguous pixels. The channel DMA address register is added to the PSTRIDE field and then updated. If PSTRIDE is set to zero, the DMA address register remains unchanged and pixels are contiguous. The PSTRIDE field of the channel configuration register is aligned to the pixel size boundary. The PSTRIDE is a two's complement number. The following formula applies at the pixel boundary and indicates how the DMA controller computes the next pixel address. The function Sizeof() returns the number of bytes required to store a pixel.

$$NextPixelAddress = CurrentPixelAddress + Sizeof(pixel) + PSTRIDE$$

36.6.8 Color Space Conversion Unit

The color space conversion unit converts Luminance Chrominance color space into the Red Green Blue color space. The conversion matrix is defined below and is fully programmable through the LCD user interface.

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} CSCRY & CSCRU & CSCRV \\ CSCGY & CSCGU & CSCGV \\ CSCBY & CSCBU & CSCBV \end{bmatrix} \bullet \begin{bmatrix} Y - Yoff \\ Cb - Cboff \\ Cr - Croff \end{bmatrix}$$

Color space conversion coefficients are defined with the following equation:

$$CSC_{ij} = \frac{1}{2^7} \cdot \left[-2^9 \cdot c_9 + \sum_{n=0}^{8} c_n \cdot 2^n \right]$$

Color space conversion coefficients are defined with one sign bit, 2 integer bits and 7 fractional bits. The range of the CSCij coefficients is defined below with a step of 1/128.

$$-4 \leq CSC_{ij} \leq 3.9921875$$

Additionally, a set scaling factor {Yoff, Cboff, Croff} can be applied.

36.6.9 Two Dimension Scaler

The High End Overlay (HEO) data path includes a hardware scaler that allows an image resize in both horizontal and vertical directions.

36.6.9.1 Video Scaler Description

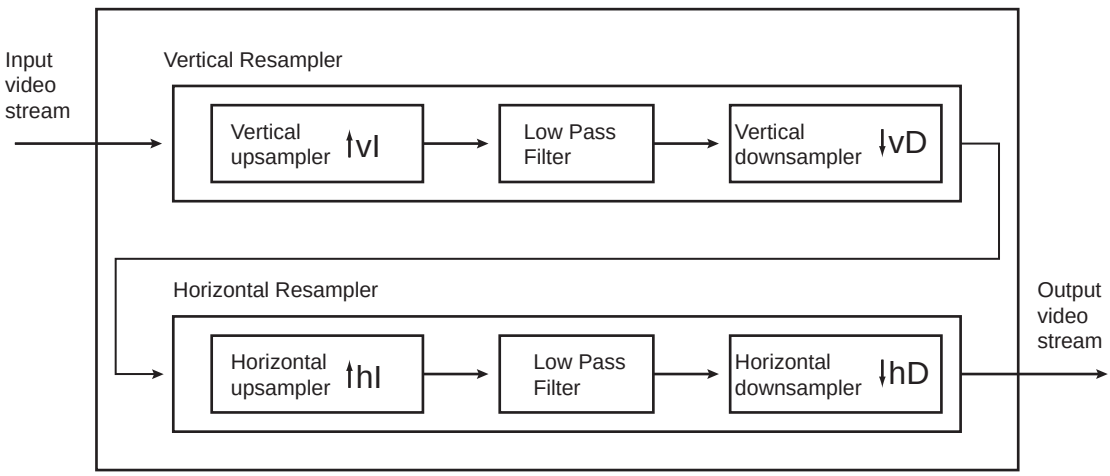
The scaling operation is based on a vertical and horizontal resampling algorithm. The sampling rate of the original image is increased when the video is upscaled, and decreased when the video is downscaled. A Vertical resampler is used to perform a vertical interpolation by a factor of *vI,* and a decimation by a factor of *vD*. A Horizontal resampler is used to perform a vertical interpolation by a factor of *hI,* and a decimation by a factor of *hD.* Both horizontal and vertical low pass filters are designed to minimize the aliasing effect. The frequency response of the low pass filter has the following characteristics:

$$H(\omega) = \begin{cases} I & \text{when } 0 \leq |\omega| \leq min(\frac{\pi}{I}, \frac{\pi}{D}) \\ 0 & \text{otherwise} \end{cases}$$

Taking into account the linear phase condition and anticipating the filter length M, the desired frequency response is modified.

$$H(\omega) = \begin{cases} Ie^{-j\omega\frac{M}{2}} & \text{when } 0 \leq |\omega| \leq min(\frac{\pi}{I}, \frac{\pi}{D}) \\ 0 & \text{otherwise} \end{cases}$$

Figure 36-6. Video Resampler Architecture



The impulse response of the low pass filter defined is:

$$h(n) = \begin{cases} I \times \frac{\omega_c}{\pi} \text{ when } n = 0 \\ I \times \frac{\omega_c}{\pi} \times \frac{\sin(\omega_c n)}{\omega_c n} \text{otherwise} \end{cases}$$

Or, for the filter of length M:

$$h(n) = \begin{cases} I \times \frac{\omega_c}{\pi} \text{ when } n = \frac{M}{2} \\ I \times \frac{\omega_c}{\pi} \times \frac{\sin\left(\omega_c\left(n - \frac{M}{2}\right)\right)}{\omega_c\left(n - \frac{M}{2}\right)} \text{otherwise} \end{cases}$$

This ideal filter is non-causal and cannot be realized. The unit sample response *h(n)* is infinite in duration and must be truncated depending on the expected length *M* of the filter. This truncation is equivalent to the multiplication of the impulse response by a window function *w(n)*.

Table 36-48. Window Function for a Filter Length M

Name of Window Function	Time Domain Sequence *w(n)*
Barlett	$1 - \frac{2 \times \left\lvert n - \frac{M-1}{2} \right\rvert}{M-1}$
Blackman	$0.42 - 0.5 \times \cos\frac{2\pi n}{M-1} + 0.08 \times \cos\frac{4\pi n}{M-1}$
Hamming	$0.54 - 0.46 \times \cos\frac{2\pi n}{M-1}$
Hanning	$0.5 - 0.5 \times \cos\frac{2\pi n}{M-1}$

The horizontal resampler includes an 8-phase 5-tap filter equivalent to a 40-tap FIR described in Figure 36-7.

The vertical resampler includes an 8-phase 3-tap filter equivalent to a 24-tap FIR described in Figure 36-8.

Figure 36-7. Horizontal Resampler Filter Architecture

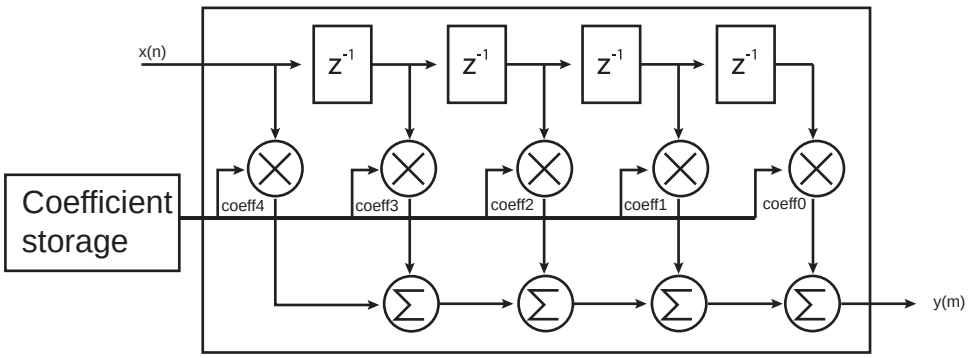
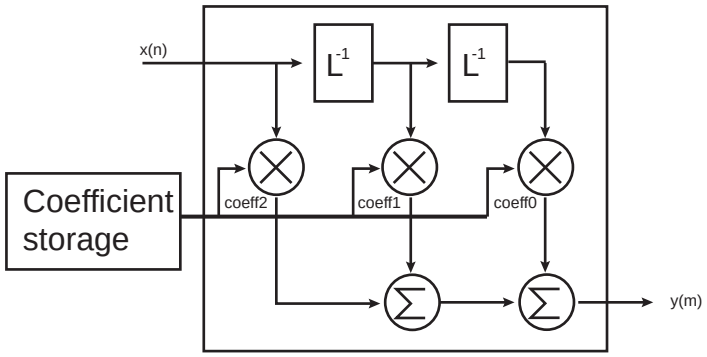


Figure 36-8. Vertical Resampler Filter Architecture



36.6.9.2 Horizontal Scaler

The XMEMSIZE field of the LCDC_HEOCFG4 register indicates the horizontal size minus one of the image in the system memory. The XSIZE field of the LCDC_HEOCFG3 register contains the horizontal size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to '1'. The scaling factor is programmed in the XFACTOR field of the LCDC_HEOCFG13 register. Use the following algorithm to find the XFACTOR value:

$$XFACTOR_{1st} = floor\left(\frac{8 \times 256 \times XMEMSIZE - 256 \times XPHIDEF}{XSIZE}\right)$$

$$XFACTOR_{1st} = XFACTOR_{1st} + 1$$

$$XMEMSIZE_{max} = floor\left(\frac{XFACTOR_{1st} \times XSIZE + 256 \times XPHIDEF}{2048}\right)$$

$$\begin{cases} XFACTOR = XFACTOR_{1st} - 1 & when(XMEMSIZE_{max} > XMEMSIZE) \\ XFACTOR = XFACTOR_{1st} & otherwise \end{cases}$$

36.6.9.3 Vertical Scaler

The YMEMSIZE field of the LCDC_HEOCFG4 register indicates the vertical size minus one of the image in the system memory. The YSIZE field of the LCDC_HEOCFG3 register contains the vertical size minus one of the window. The SCALEN bit of the LCDC_HEOCFG13 register is set to one. The scaling factor is programmed in the YFACTOR field of the LCDC_HEOCFG13 register.

$$YFACTOR_{1st} = floor\left(\frac{8 \times 256 \times YMEMSIZE - 256 \times YPHIDEF}{YSIZE}\right)$$

$$YFACTOR_{1st} = YFACTOR_{1st} + 1$$

$$YMEMSIZE_{max} = floor\left(\frac{YFACTOR_{1st} \times YSIZE + 256 \times YPHIDEF}{2048}\right)$$

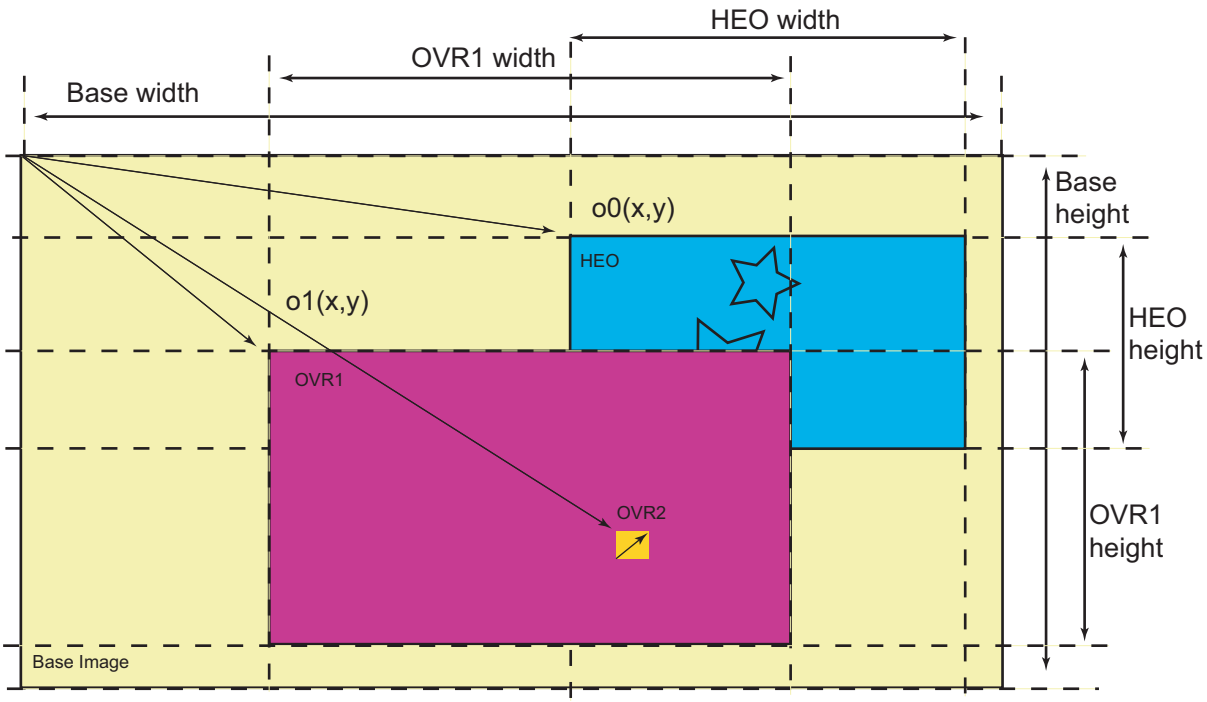
$$\begin{cases} YFACTOR = YFACTOR_{1st} - 1 & when(YMEMSIZE_{max} > YMEMSIZE) \\ YFACTOR = YFACTOR_{1st} & otherwise \end{cases}$$

36.6.10 Color Combine Unit

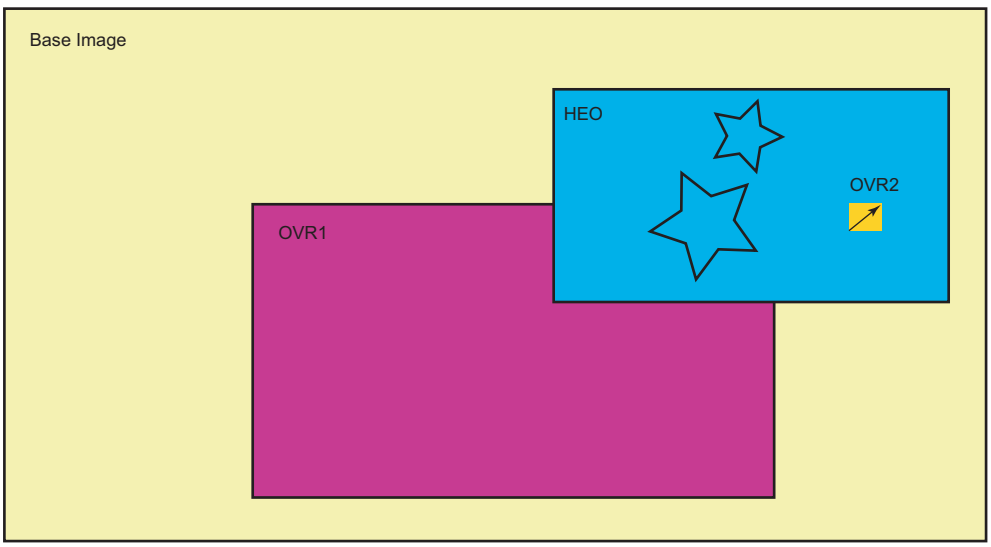
36.6.10.1 Window Overlay

The LCD module provides hardware support for multiple “overlay plane” that can be used to display windows on top of the image without destroying the image located below. The overlay image can use any color depth. Using the overlay alleviates the need to re-render the occluded portion of the image. When pixels are combined together through the alpha blending unit, a new color is created. This new pixel is called an iterated pixel and is passed to the next blending stage. Then, this pixel may be combined again with another pixel. The VIDPRI bit located in the LCDC_HEOCFG12 register configures the video priority algorithm used to display the layers. When the VIDPRI bit is written to ‘0’, the OVR1 layer is located above the HEO layer. When the VIDPRI bit is written to ‘1’, OVR1 is located below the HEO layer.

Figure 36-9. Overlay Example with Two Different Video Prioritization Algorithms



Video Prioritization Algorithm 1 : OVR2 > OVR1 > HEO > BASE



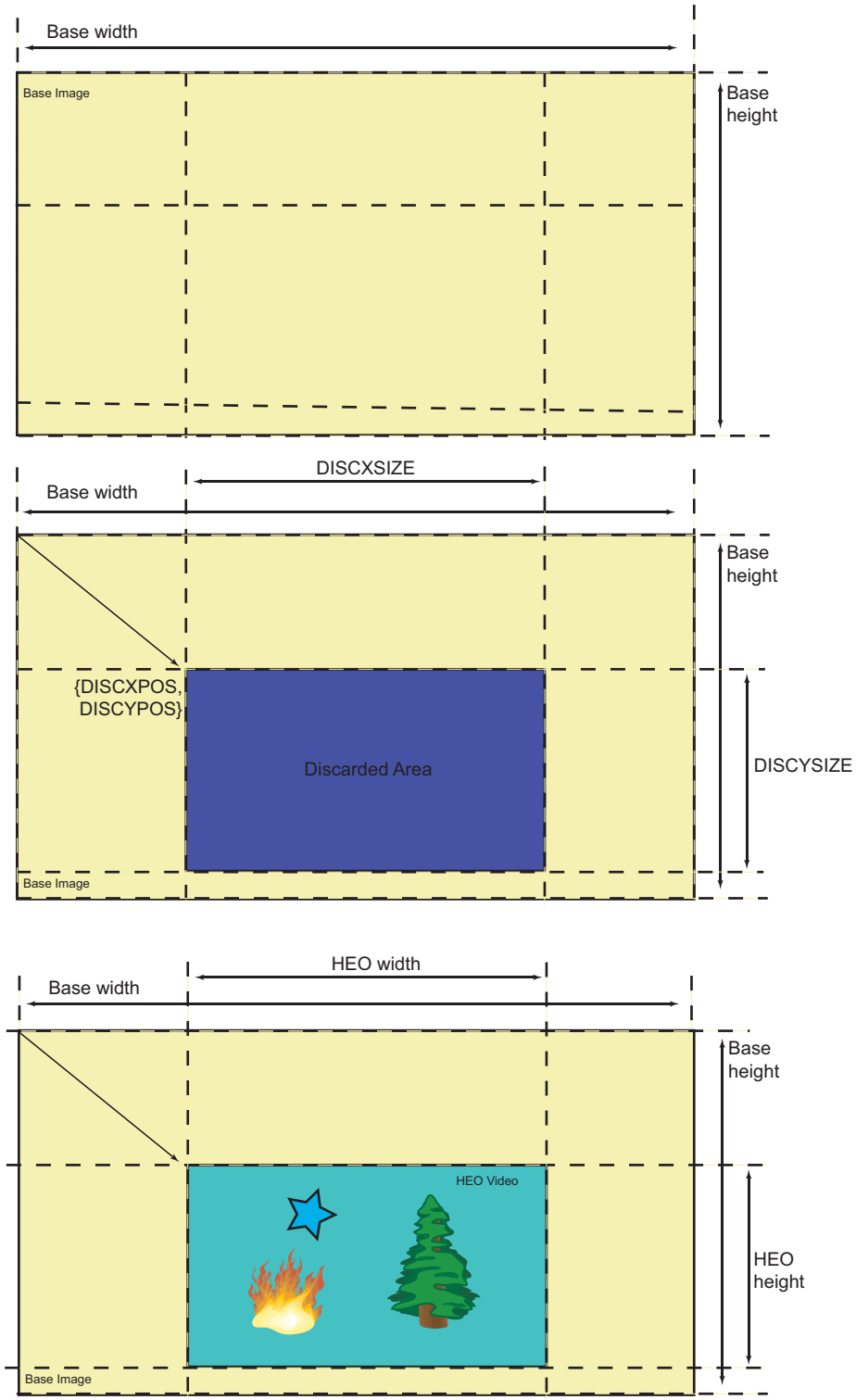
Video Prioritization Algorithm 2 : OVR2 > HEO > OVR1 > BASE

36.6.10.2 Base Layer with Window Overlay Optimization

When the base layer is combined with at least one active overlay, the whole base layer frame is retrieved from the memory though it is not visible. A set of registers is used to disable the Base DMA when this condition is met. These registers are the following:

- LCDC_BASECFG5:
  - field DISCXPOS (Discard Area Horizontal Position)
  - field DISCYPOS (Discard Area Vertical Position)
- LCDC_BASECFG6:
  - field DISCXSIZE (Discard Area Horizontal Size)
  - field DISCYSIZE (Discard Area Vertical Size)
- LCDC_BASECFG4: bit DISCEN (Discard Area Enable)

Figure 36-10. Base Layer Discard Area



36.6.10.3 Overlay Blending

The blending function requires two pixels (one iterated from the previous blending stage and one from the current overlay color) and a set of blending configuration parameters. These parameters define the color operation.

Figure 36-11. Alpha Blender Function

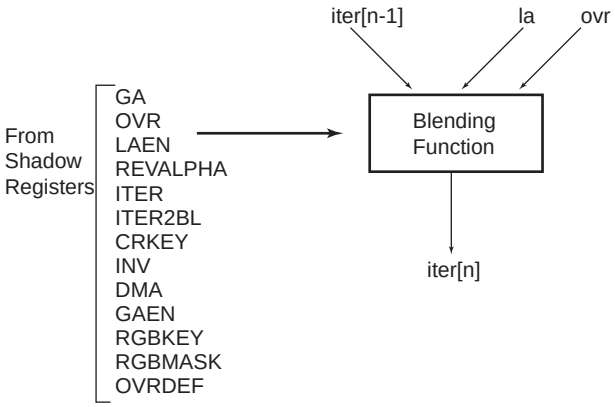
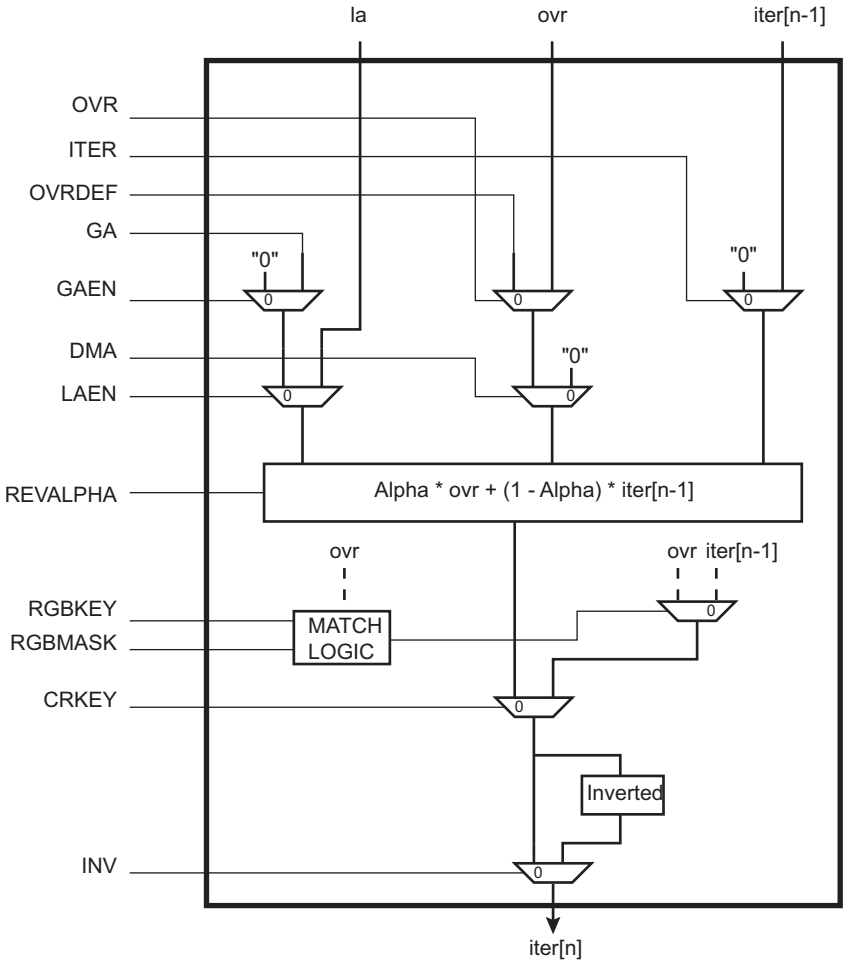


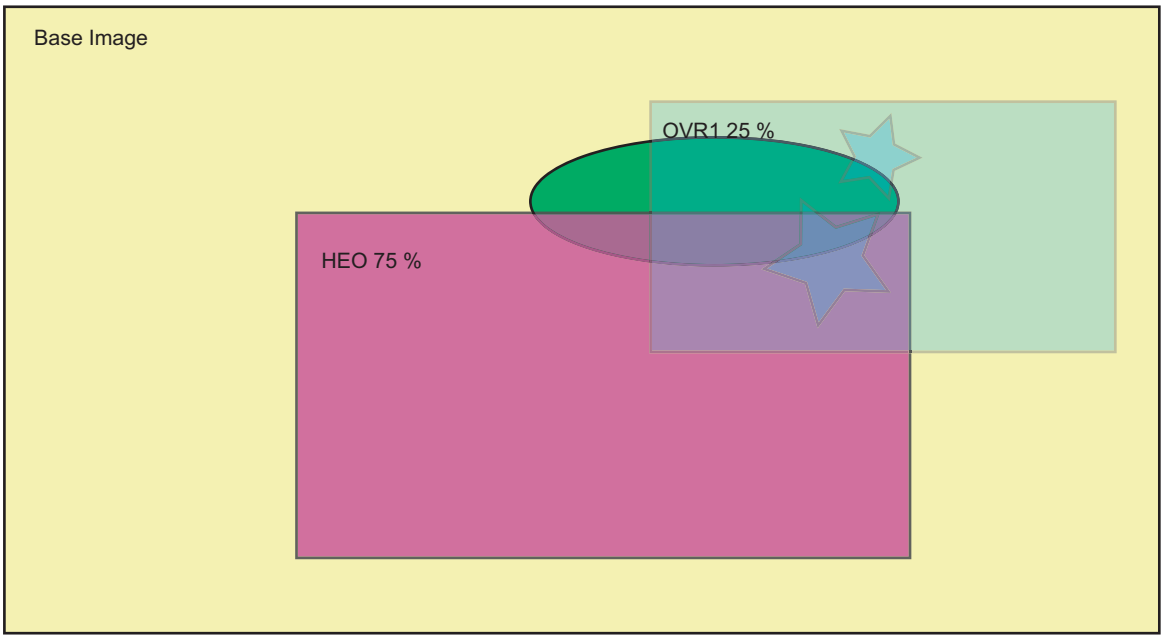
Figure 36-12. Alpha Blender Database



36.6.10.4 Global Alpha Blender

36.6.10.5 Window Blending

Figure 36-13. 256-level Alpha Blending



Video Prioritization Algorithm 1: OVR1 > HEO > BASE

36.6.10.6 Color Keying

Color keying involves a method of bit-block image transfer (Blit). This entails blitting one image onto another where not all the pixels are copied. Blitting usually involves two bitmaps: a source bitmap and a destination bitmap. A raster operation (ROP) is performed to define whether the iterated color or the overlay color is to be visible or not.

Source Color Keying

If the masked overlay color matches the color key, the iterated color is selected and Source Color Keying is activated using the following configuration sequence:

1. Select the overlay to blit.
2. Write a ‘0’ to DSTKEY.
3. Activate Color Keying by writing a ‘1’ to CRKEY.
4. Configure the Color Key by writing RKEY, GKEY and BKEY fields.
5. Configure the Color Mask by writing RKEY, GKEY and BKEY fields..

When the field RMASK, GMASK, or BMASK is configured to ‘0’, the comparison is disabled and the raster operation is activated.

Destination Color Keying

If the iterated masked color matches the color key then the overlay color is selected, Destination Color Keying is activated using the following configuration sequence:

1. Select the overlay to blit.
2. Write a ‘1’ to DSTKEY.
3. Activate Color Keying by writing a ‘1’ to CRKEY bit
4. Configure the Color Key by writing RKEY, GKEY and BKEY fields.
5. Configure the Color Mask by writing RKEY, GKEY and BKEY fields.

When the field RMASK, GMASK, or BMASK is configured to ‘0’, the comparison is disabled and the raster operation is activated.

36.6.11 LCDC PWM Controller

This block generates the LCD contrast control signal (LCD_PWM) to make possible the control of the display's contrast by software. This is an 8-bit PWM (Pulse Width Modulation) signal that can be converted to an analog voltage with a simple passive filter.

The PWM module has a free-running counter whose value is compared against a compare register (PWMCVAL field of the LCDC_LCDCFG6 register). If the value in the counter is less than that in the register, the output brings the value of the signal polarity (PWMPOL) bit in the PWM control register: LCDC_LCDCFG6. Otherwise, the opposite value is output. Thus, a periodic waveform with a pulse width proportional to the value in the compare register is generated.

Due to the comparison mechanism, the output pulse has a width between zero and 255 PWM counter cycles. Thus by adding a simple passive filter outside the chip, an analog voltage between 0 and (255/256) × V_{DD} can be obtained (for the positive polarity case, or between (1/256) × V_{DD} and V_{DD} for the negative polarity case). Other voltage values can be obtained by adding active external circuitry.

For PWM mode, the counter frequency can be adjusted to four different values using the PWMPS field of the LCDC_LCDCFG6 register.

The PWM module can be fed with the slow clock or the system clock, depending on the CLKPWMSEL bit of the LCDC_CFG0 register.

36.6.12 Post Processing Controller

The output stream of pixels can be either displayed on the screen or written to the memory using the Post Processing Controller (PPC). When the PPC is used, the screen display is disabled, but synchronization signals remain active (if enabled). The stream of pixel can be written in RGB mode or encoded in YCbCr 422 mode. A programmable color space conversion stage is available.

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} CSCYR & CSCYG & CSCYB \\ CSCUR & CSCUG & CSCUB \\ CSCVR & CSCVG & CSCUB \end{bmatrix} \bullet \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} Yoff \\ Uoff \\ Voff \end{bmatrix}$$

36.6.13 LCD Overall Performance

36.6.13.1 Color Lookup Table (CLUT)

Table 36-49. CLUT Pixel Performance

CLUT Mode	Pixels/Cycle	Rotation	Scaling
1 bpp	64	Not supported	Supported
2 bpp	32	Not supported	Supported
3 bpp	16	Not supported	Supported
4 bpp	8	Not supported	Supported

36.6.13.2 RGB Mode Fetch Performance

Table 36-50. RGB Mode Performance

RGB Mode	Pixels/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Available
		Rotation Optimization (1)	Normal Mode	
12 bpp	4	1	0.2	Supported
16 bpp	4	1	0.2	Supported
18 bpp	2	1	0.2	Supported
18 bpp RGB PACKED	2.666	Not supported	0.2	Supported
19 bpp	2	1	0.2	Supported
19 bpp PACKED	2.666	Not Supported	0.2	Supported
24 bpp	2	1	0.2	Supported
24 bpp PACKED	2.666	Not Supported	0.2	Supported
25 bpp	2	1	0.2	Supported
32 bpp	2	1	0.2	Supported

Note: 1. Rotation optimization = AHB lock asserted on consecutive single access.

36.6.13.3 YUV Mode Fetch Performance

Table 36-51. Single Stream for 0 Wait State Memory

YUV Mode	Pixels/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Is Available
		Rotation Optimization (1)	Normal Mode	
32 bpp AYUV	2	1	0.2	Supported
16 bpp 422	4	Not Supported	Not Supported	Supported

Note: 1. Rotation optimization = AHB lock asserted on consecutive single access

Table 36-52. Multiple Stream for 0 Wait State Memory

YUV Mode	Comp/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Is Available
		Rotation Optimization	Normal Mode	
16 bpp 422 semiplanar	8 Y, 4 UV	1 Y, 1 UV (2 streams)	0.2 Y 0.2 UV (2 streams)	Supported
16 bpp 422 planar	8 Y, 8 U, 8 V	1 Y, 1 U, 1 V (3 streams)	0.2 Y, 0.2 U, 0.2 V (3 streams)	Supported
12 bpp 4:2:0 semiplanar	8 Y, 4 UV	1 Y, 1 UV (2 streams)	0.2 Y 0.2 UV (2 streams)	Supported
12 bpp 4:2:0 planar	8 Y, 8 U, 8 V	1 Y, 1 U, 1 V (3 streams)	0.2 Y, 0.2 U, 0.2 V (3 streams)	Supported

Note: In order to provide more bandwidth, when multiple streams are used to transfer Y, UV, U or V components, two AHB interfaces are recommended or multiple AXI ID are required.

Table 36-53. YUV Planar Overall Performance 1 AHB Interface for 0 Wait State Memory

YUV Mode	Pix/Cycle Memory Burst Mode	Rotation Peak Random Memory Access (pixels/cycle)		Scaling Burst Mode or Rotation Optimization Is Available
		Rotation Optimization	Normal Mode	
16 bpp 422 semiplanar	4	0.66	0.132	Supported
16 bpp 422 planar	4	0.5	0.1	Supported
12 bpp 4:2:0 semiplanar	5.32	0.8	0.16	Supported
12 bpp 4:2:0 planar	5.32	0.66	0.132	Supported

Note: In order to provide more bandwidth, when multiple streams are used to transfer Y, UV, U or V components, two AHB interfaces are recommended or multiple AXI ID are required.

36.6.14 Input FIFO

The LCD module includes one input FIFO per overlay. These input FIFOs are used to buffer the AHB burst and serialize the stream of pixels.

36.6.15 Output FIFO

The LCD module includes one output FIFO that stores the blended pixel.

36.6.16 Output Timing Generation

36.6.16.1 Active Display Timing Mode

Figure 36-14. Active Display Timing

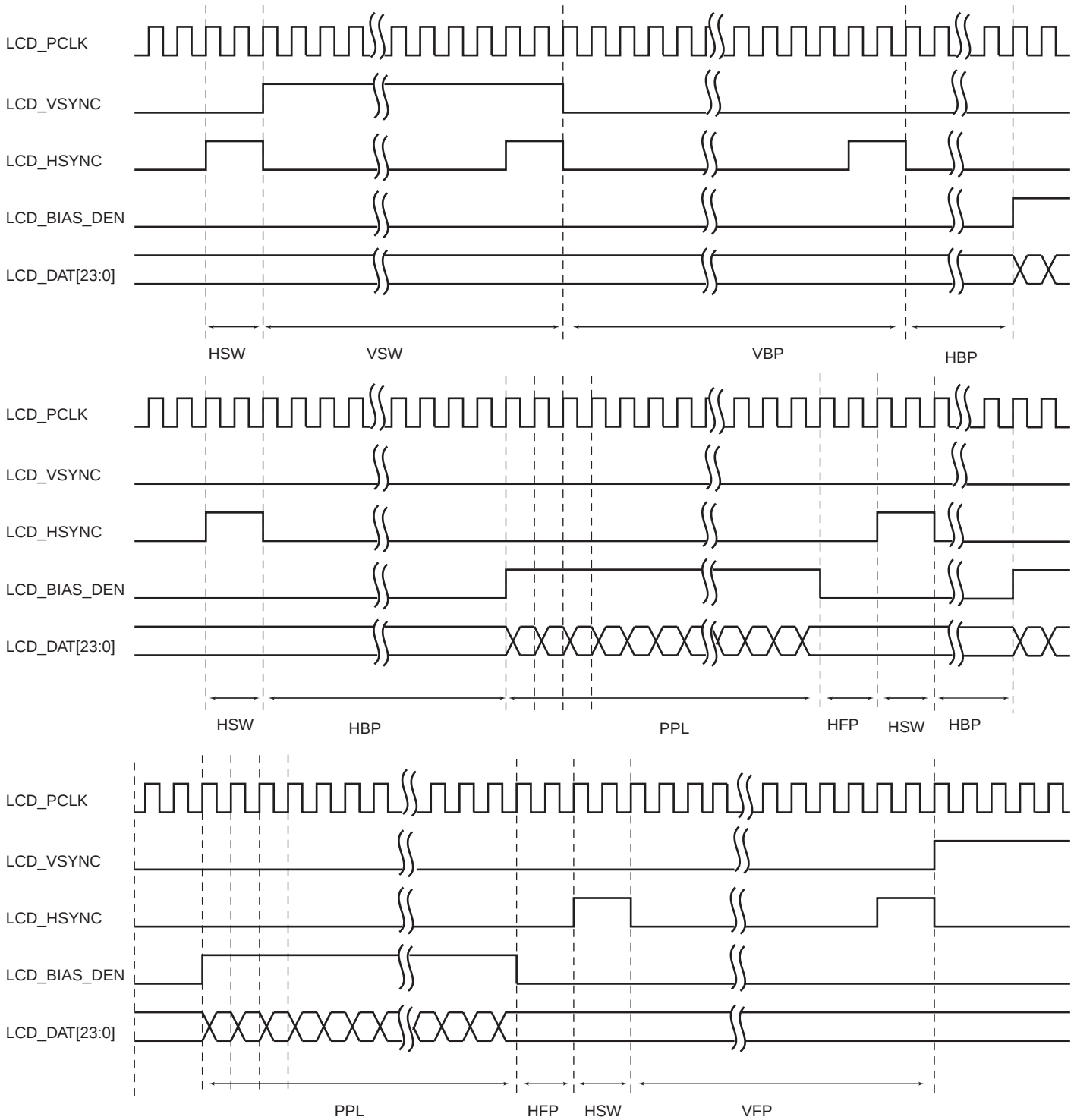


Figure 36-15. Vertical Synchronization Timing (part 1)

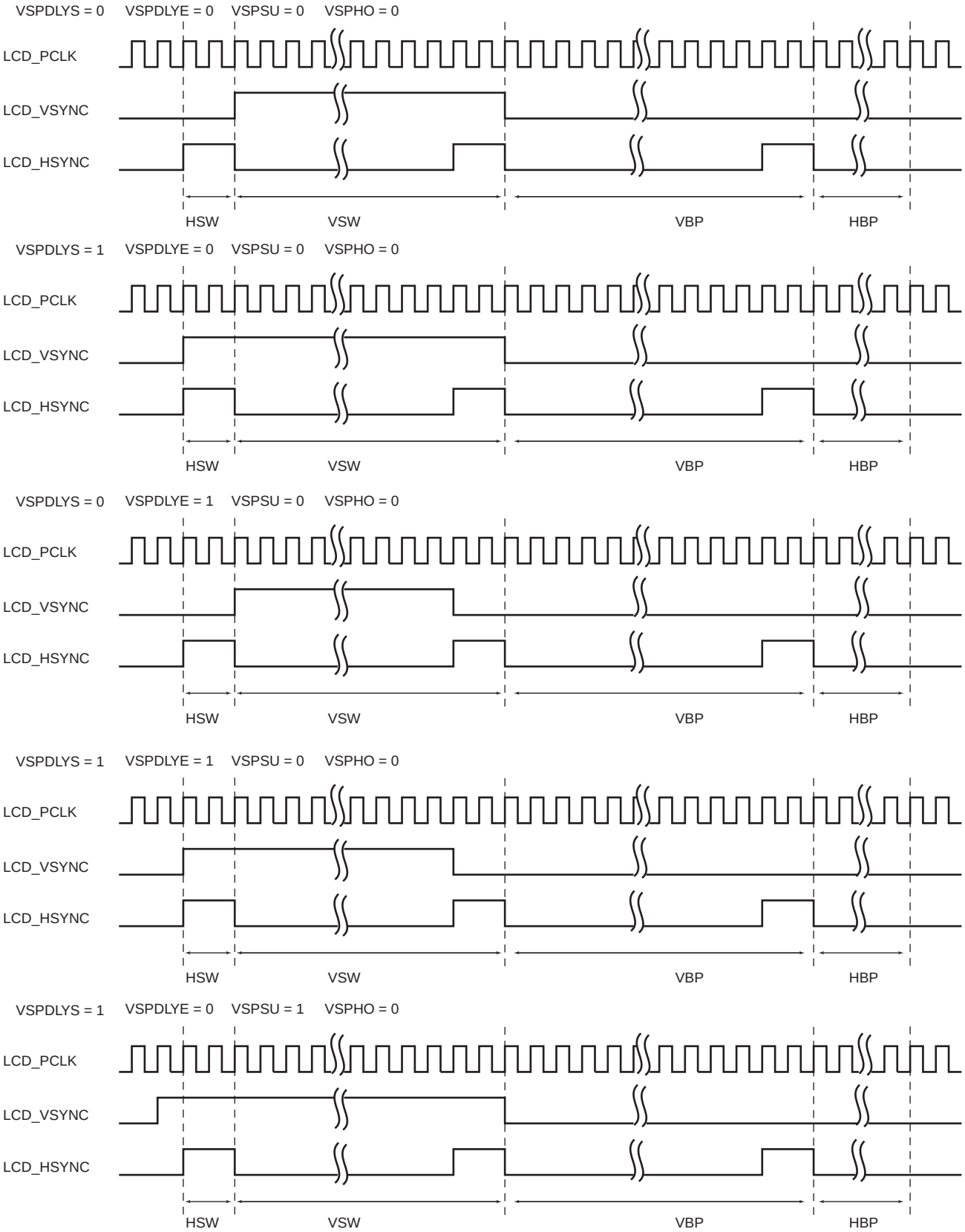


Figure 36-16. Vertical Synchronization Timing (part 2)

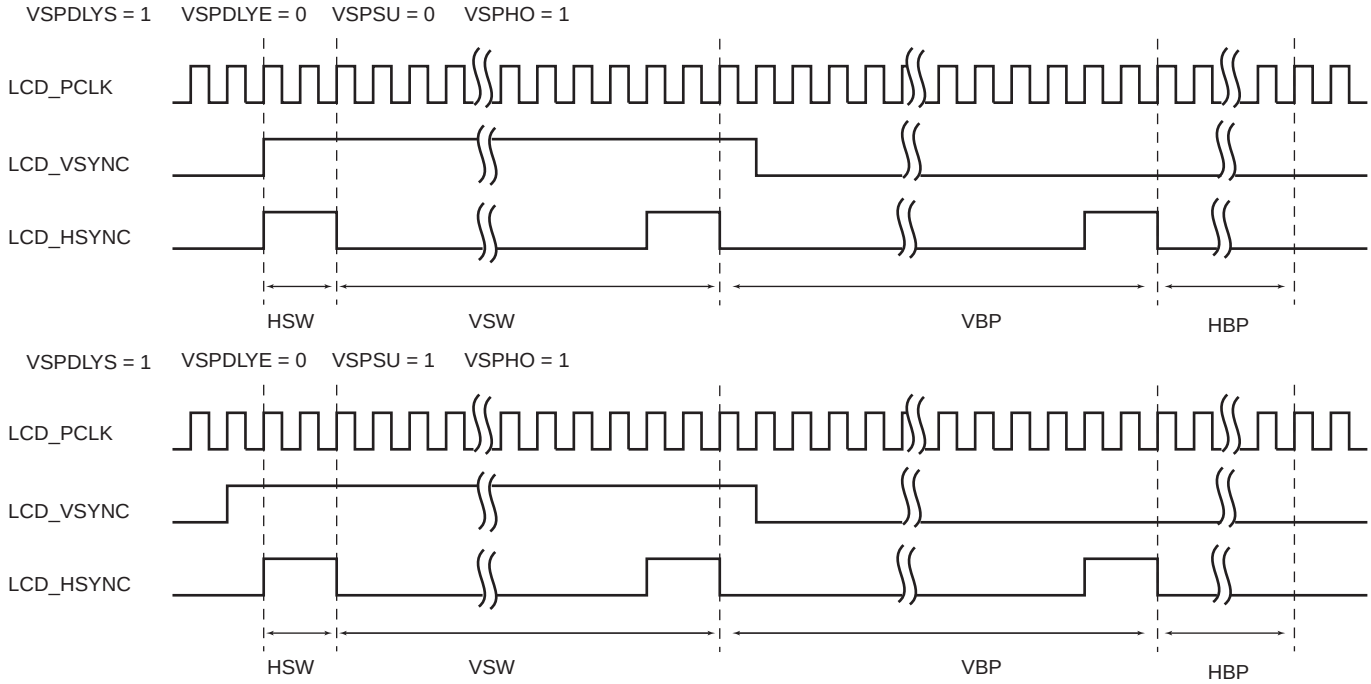
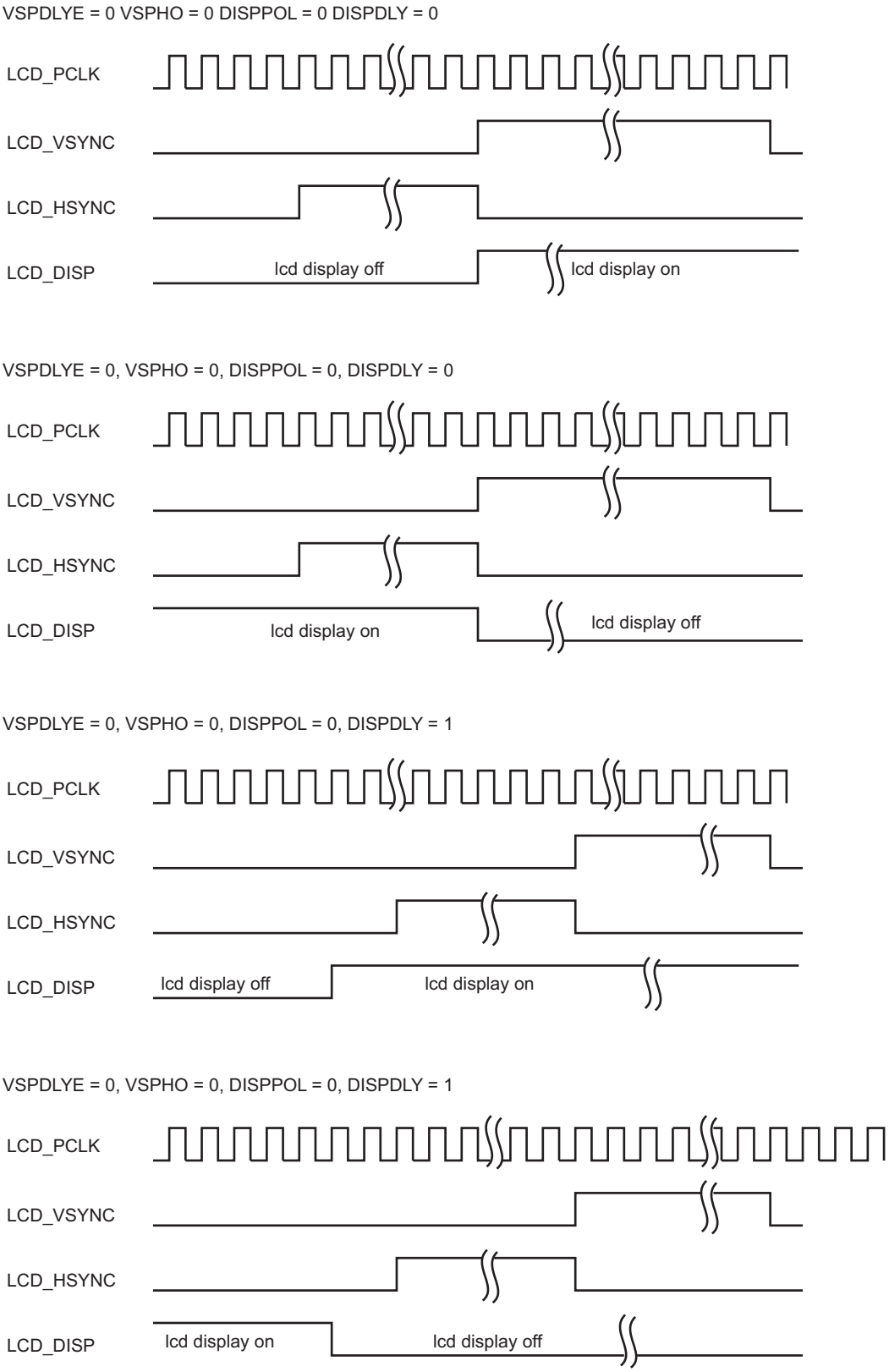


Figure 36-17. DISP Signal Timing Diagram



36.6.17 Output Format

36.6.17.1 Active Mode Output Pin Assignment

Table 36-54. Active Mode Output with 24-bit Bus Interface Configuration

Pin ID	TFT 24 bits	TFT 18 bits	TFT 16 bits	TFT 12 bits
LCD_DAT[23]	R[7]	R[5]	R[4]	R[3]
LCD_DAT[22]	R[6]	R[4]	R[3]	R[2]
LCD_DAT[21]	R[5]	R[3]	R[2]	R[1]
LCD_DAT[20]	R[4]	R[2]	R[1]	R[0]
LCD_DAT[19]	R[3]	R[1]	R[0]	–
LCD_DAT[18]	R[2]	R[0]	–	–
LCD_DAT[17]	R[1]	–	–	–
LCD_DAT[16]	R[0]	–	–	–
LCD_DAT[15]	G[7]	G[5]	G[5]	G[3]
LCD_DAT[14]	G[6]	G[4]	G[4]	G[2]
LCD_DAT[13]	G[5]	G[3]	G[3]	G[1]
LCD_DAT[12]	G[4]	G[2]	G[2]	G[0]
LCD_DAT[11]	G[3]	G[1]	G[1]	–
LCD_DAT[10]	G[2]	G[0]	G[0]	–
LCD_DAT[9]	G[1]	–	–	–
LCD_DAT[8]	G[0]	–	–	–
LCD_DAT[7]	B[7]	B[5]	B[4]	B[3]
LCD_DAT[6]	B[6]	B[4]	B[3]	B[2]
LCD_DAT[5]	B[5]	B[3]	B[2]	B[1]
LCD_DAT[4]	B[4]	B[2]	B[1]	B[0]
LCD_DAT[3]	B[3]	B[1]	B[0]	–
LCD_DAT[2]	B[2]	B[0]	–	–
LCD_DAT[1]	B[1]	–	–	–
LCD_DAT[0]	B[0]	–	–	–

36.7 LCD Controller (LCDC) User Interface

Table 36-55. Register Mapping

Offset	Register	Name	Access	Reset
0x00000000	LCD Controller Configuration Register 0	LCDC_LCDCFG0	Read/Write	0x00000000
0x00000004	LCD Controller Configuration Register 1	LCDC_LCDCFG1	Read/Write	0x00000000
0x00000008	LCD Controller Configuration Register 2	LCDC_LCDCFG2	Read/Write	0x00000000
0x0000000C	LCD Controller Configuration Register 3	LCDC_LCDCFG3	Read/Write	0x00000000
0x00000010	LCD Controller Configuration Register 4	LCDC_LCDCFG4	Read/Write	0x00000000
0x00000014	LCD Controller Configuration Register 5	LCDC_LCDCFG5	Read/Write	0x00000000
0x00000018	LCD Controller Configuration Register 6	LCDC_LCDCFG6	Read/Write	0x00000000
0x0000001C	Reserved	–	–	–
0x00000020	LCD Controller Enable Register	LCDC_LCDEN	Write-only	–
0x00000024	LCD Controller Disable Register	LCDC_LCDDIS	Write-only	–
0x00000028	LCD Controller Status Register	LCDC_LCDSR	Read-only	0x00000000
0x0000002C	LCD Controller Interrupt Enable Register	LCDC_LCDIER	Write-only	–
0x00000030	LCD Controller Interrupt Disable Register	LCDC_LCDIDR	Write-only	–
0x00000034	LCD Controller Interrupt Mask Register	LCDC_LCDIMR	Read-only	0x00000000
0x00000038	LCD Controller Interrupt Status Register	LCDC_LCDISR	Read-only	0x00000000
0x0000003C	LCD Controller Attribute Register	LCDC_ATTR	Write-only	–
0x00000040	Base Layer Channel Enable Register	LCDC_BASECHER	Write-only	–
0x00000044	Base Layer Channel Disable Register	LCDC_BASECHDR	Write-only	–
0x00000048	Base Layer Channel Status Register	LCDC_BASECHSR	Read-only	0x00000000
0x0000004C	Base Layer Interrupt Enable Register	LCDC_BASEIER	Write-only	–
0x00000050	Base Layer Interrupt Disabled Register	LCDC_BASEIDR	Write-only	–
0x00000054	Base Layer Interrupt Mask Register	LCDC_BASEIMR	Read-only	0x00000000
0x00000058	Base Layer Interrupt Status Register	LCDC_BASEISR	Read-only	0x00000000
0x0000005C	Base DMA Head Register	LCDC_BASEHEAD	Read/Write	0x00000000
0x00000060	Base DMA Address Register	LCDC_BASEADDR	Read/Write	0x00000000
0x00000064	Base DMA Control Register	LCDC_BASECTRL	Read/Write	0x00000000
0x00000068	Base DMA Next Register	LCDC_BASENEXT	Read/Write	0x00000000
0x0000006C	Base Layer Configuration Register 0	LCDC_BASECFG0	Read/Write	0x00000000
0x00000070	Base Layer Configuration Register 1	LCDC_BASECFG1	Read/Write	0x00000000
0x00000074	Base Layer Configuration Register 2	LCDC_BASECFG2	Read/Write	0x00000000
0x00000078	Base Layer Configuration Register 3	LCDC_BASECFG3	Read/Write	0x00000000
0x0000007C	Base Layer Configuration Register 4	LCDC_BASECFG4	Read/Write	0x00000000
0x00000080	Base Layer Configuration Register 5	LCDC_BASECFG5	Read/Write	0x00000000
0x00000084	Base Layer Configuration Register 6	LCDC_BASECFG6	Read/Write	0x00000000
0x00000088–0x0000013C	Reserved	–	–	–

Table 36-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x00000140	Overlay 1 Channel Enable Register	LCDC_OVR1CHER	Write-only	–
0x00000144	Overlay 1 Channel Disable Register	LCDC_OVR1CHDR	Write-only	–
0x00000148	Overlay 1 Channel Status Register	LCDC_OVR1CHSR	Read-only	0x00000000
0x0000014C	Overlay 1 Interrupt Enable Register	LCDC_OVR1IER	Write-only	–
0x00000150	Overlay 1 Interrupt Disable Register	LCDC_OVR1IDR	Write-only	–
0x00000154	Overlay 1 Interrupt Mask Register	LCDC_OVR1IMR	Read-only	0x00000000
0x00000158	Overlay 1 Interrupt Status Register	LCDC_OVR1ISR	Read-only	0x00000000
0x0000015C	Overlay 1 DMA Head Register	LCDC_OVR1HEAD	Read/Write	0x00000000
0x00000160	Overlay 1 DMA Address Register	LCDC_OVR1ADDR	Read/Write	0x00000000
0x00000164	Overlay 1 DMA Control Register	LCDC_OVR1CTRL	Read/Write	0x00000000
0x00000168	Overlay 1 DMA Next Register	LCDC_OVR1NEXT	Read/Write	0x00000000
0x0000016C	Overlay 1 Configuration Register 0	LCDC_OVR1CFG0	Read/Write	0x00000000
0x00000170	Overlay 1 Configuration Register 1	LCDC_OVR1CFG1	Read/Write	0x00000000
0x00000174	Overlay 1 Configuration Register 2	LCDC_OVR1CFG2	Read/Write	0x00000000
0x00000178	Overlay 1 Configuration Register 3	LCDC_OVR1CFG3	Read/Write	0x00000000
0x0000017C	Overlay 1 Configuration Register 4	LCDC_OVR1CFG4	Read/Write	0x00000000
0x00000180	Overlay 1 Configuration Register 5	LCDC_OVR1CFG5	Read/Write	0x00000000
0x00000184	Overlay 1 Configuration Register 6	LCDC_OVR1CFG6	Read/Write	0x00000000
0x00000188	Overlay 1 Configuration Register 7	LCDC_OVR1CFG7	Read/Write	0x00000000
0x0000018C	Overlay 1 Configuration Register 8	LCDC_OVR1CFG8	Read/Write	0x00000000
0x00000190	Overlay 1 Configuration Register 9	LCDC_OVR1CFG9	Read/Write	0x00000000
0x00000194–0x0000023C	Reserved	–	–	–
0x00000240	Overlay 2 Channel Enable Register	LCDC_OVR2CHER	Write-only	–
0x00000244	Overlay 2 Channel Disable Register	LCDC_OVR2CHDR	Write-only	–
0x00000248	Overlay 2 Channel Status Register	LCDC_OVR2CHSR	Read-only	0x00000000
0x0000024C	Overlay 2 Interrupt Enable Register	LCDC_OVR2IER	Write-only	–
0x00000250	Overlay 2 Interrupt Disable Register	LCDC_OVR2IDR	Write-only	–
0x00000254	Overlay 2 Interrupt Mask Register	LCDC_OVR2IMR	Read-only	0x00000000
0x00000258	Overlay 2 Interrupt Status Register	LCDC_OVR2ISR	Read-only	0x00000000
0x0000025C	Overlay 2 DMA Head Register	LCDC_OVR2HEAD	Read/Write	0x00000000
0x00000260	Overlay 2 DMA Address Register	LCDC_OVR2ADDR	Read/Write	0x00000000
0x00000264	Overlay 2 DMA Control Register	LCDC_OVR2CTRL	Read/Write	0x00000000
0x00000268	Overlay 2 DMA Next Register	LCDC_OVR2NEXT	Read/Write	0x00000000
0x0000026C	Overlay 2 Configuration Register 0	LCDC_OVR2CFG0	Read/Write	0x00000000
0x00000270	Overlay 2 Configuration Register 1	LCDC_OVR2CFG1	Read/Write	0x00000000
0x00000274	Overlay 2 Configuration Register 2	LCDC_OVR2CFG2	Read/Write	0x00000000
0x00000278	Overlay 2 Configuration Register 3	LCDC_OVR2CFG3	Read/Write	0x00000000

Table 36-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x0000027C	Overlay 2 Configuration Register 4	LCDC_OVR2CFG4	Read/Write	0x00000000
0x00000280	Overlay 2 Configuration Register 5	LCDC_OVR2CFG5	Read/Write	0x00000000
0x00000284	Overlay 2 Configuration Register 6	LCDC_OVR2CFG6	Read/Write	0x00000000
0x00000288	Overlay 2 Configuration Register 7	LCDC_OVR2CFG7	Read/Write	0x00000000
0x0000028C	Overlay 2 Configuration Register 8	LCDC_OVR2CFG8	Read/Write	0x00000000
0x00000290	Overlay 2 Configuration Register 8	LCDC_OVR2CFG9	Read/Write	0x00000000
0x00000294–0x0000033C	Reserved	–	–	–
0x00000340	High End Overlay Channel Enable Register	LCDC_HEOCHER	Write-only	–
0x00000344	High End Overlay Channel Disable Register	LCDC_HEOCHDR	Write-only	–
0x00000348	High End Overlay Channel Status Register	LCDC_HEOCHSR	Read-only	0x00000000
0x0000034C	High End Overlay Interrupt Enable Register	LCDC_HEOIER	Write-only	–
0x00000350	High End Overlay Interrupt Disable Register	LCDC_HEOIDR	Write-only	–
0x00000354	High End Overlay Interrupt Mask Register	LCDC_HEOIMR	Read-only	0x00000000
0x00000358	High End Overlay Interrupt Status Register	LCDC_HEOISR	Read-only	0x00000000
0x0000035C	High End Overlay DMA Head Register	LCDC_HEOHEAD	Read/Write	0x00000000
0x00000360	High End Overlay DMA Address Register	LCDC_HEOADDR	Read/Write	0x00000000
0x00000364	High End Overlay DMA Control Register	LCDC_HEOCTRL	Read/Write	0x00000000
0x00000368	High End Overlay DMA Next Register	LCDC_HEONEXT	Read/Write	0x00000000
0x0000036C	High End Overlay U-UV DMA Head Register	LCDC_HEOUHEAD	Read/Write	0x00000000
0x00000370	High End Overlay U-UV DMA Address Register	LCDC_HEOUADDR	Read/Write	0x00000000
0x00000374	High End Overlay U-UV DMA Control Register	LCDC_HEOUCTRL	Read/Write	0x00000000
0x00000378	High End Overlay U-UV DMA Next Register	LCDC_HEOUNEXT	Read/Write	0x00000000
0x0000037C	High End Overlay V DMA Head Register	LCDC_HEOVHEAD	Read/Write	0x00000000
0x00000380	High End Overlay V DMA Address Register	LCDC_HEOVADDR	Read/Write	0x00000000
0x00000384	High End Overlay V DMA Control Register	LCDC_HEOVCTRL	Read/Write	0x00000000
0x00000388	High End Overlay V DMA Next Register	LCDC_HEOVNEXT	Read/Write	0x00000000
0x0000038C	High End Overlay Configuration Register 0	LCDC_HEOCFG0	Read/Write	0x00000000
0x00000390	High End Overlay Configuration Register 1	LCDC_HEOCFG1	Read/Write	0x00000000
0x00000394	High End Overlay Configuration Register 2	LCDC_HEOCFG2	Read/Write	0x00000000
0x00000398	High End Overlay Configuration Register 3	LCDC_HEOCFG3	Read/Write	0x00000000
0x0000039C	High End Overlay Configuration Register 4	LCDC_HEOCFG4	Read/Write	0x00000000
0x000003A0	High End Overlay Configuration Register 5	LCDC_HEOCFG5	Read/Write	0x00000000
0x000003A4	High End Overlay Configuration Register 6	LCDC_HEOCFG6	Read/Write	0x00000000
0x000003A8	High End Overlay Configuration Register 7	LCDC_HEOCFG7	Read/Write	0x00000000
0x000003AC	High End Overlay Configuration Register 8	LCDC_HEOCFG8	Read/Write	0x00000000
0x000003B0	High End Overlay Configuration Register 9	LCDC_HEOCFG9	Read/Write	0x00000000

Table 36-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x000003B4	High End Overlay Configuration Register 10	LCDC_HEOCFG10	Read/Write	0x00000000
0x000003B8	High End Overlay Configuration Register 11	LCDC_HEOCFG11	Read/Write	0x00000000
0x000003BC	High End Overlay Configuration Register 12	LCDC_HEOCFG12	Read/Write	0x00000000
0x000003C0	High End Overlay Configuration Register 13	LCDC_HEOCFG13	Read/Write	0x00000000
0x000003C4	High End Overlay Configuration Register 14	LCDC_HEOCFG14	Read/Write	0x00000000
0x000003C8	High End Overlay Configuration Register 15	LCDC_HEOCFG15	Read/Write	0x00000000
0x000003CC	High End Overlay Configuration Register 16	LCDC_HEOCFG16	Read/Write	0x00000000
0x000003D0	High End Overlay Configuration Register 17	LCDC_HEOCFG17	Read/Write	0x00000000
0x000003D4	High End Overlay Configuration Register 18	LCDC_HEOCFG18	Read/Write	0x00000000
0x000003D8	High End Overlay Configuration Register 19	LCDC_HEOCFG19	Read/Write	0x00000000
0x000003DC	High End Overlay Configuration Register 20	LCDC_HEOCFG20	Read/Write	0x00000000
0x000003E0	High End Overlay Configuration Register 21	LCDC_HEOCFG21	Read/Write	0x00000000
0x000003E4	High End Overlay Configuration Register 22	LCDC_HEOCFG22	Read/Write	0x00000000
0x000003E8	High End Overlay Configuration Register 23	LCDC_HEOCFG23	Read/Write	0x00000000
0x000003EC	High End Overlay Configuration Register 24	LCDC_HEOCFG24	Read/Write	0x00000000
0x000003F0	High End Overlay Configuration Register 25	LCDC_HEOCFG25	Read/Write	0x00000000
0x000003F4	High End Overlay Configuration Register 26	LCDC_HEOCFG26	Read/Write	0x00000000
0x000003F8	High End Overlay Configuration Register 27	LCDC_HEOCFG27	Read/Write	0x00000000
0x000003FC	High End Overlay Configuration Register 28	LCDC_HEOCFG28	Read/Write	0x00000000
0x00000400	High End Overlay Configuration Register 29	LCDC_HEOCFG29	Read/Write	0x00000000
0x00000404	High End Overlay Configuration Register 30	LCDC_HEOCFG30	Read/Write	0x00000000
0x00000408	High End Overlay Configuration Register 31	LCDC_HEOCFG31	Read/Write	0x00000000
0x0000040C	High End Overlay Configuration Register 32	LCDC_HEOCFG32	Read/Write	0x00000000
0x00000410	High End Overlay Configuration Register 33	LCDC_HEOCFG33	Read/Write	0x00000000
0x00000414	High End Overlay Configuration Register 34	LCDC_HEOCFG34	Read/Write	0x00000000
0x00000418	High End Overlay Configuration Register 35	LCDC_HEOCFG35	Read/Write	0x00000000
0x0000041C	High End Overlay Configuration Register 36	LCDC_HEOCFG36	Read/Write	0x00000000
0x00000420	High End Overlay Configuration Register 37	LCDC_HEOCFG37	Read/Write	0x00000000
0x00000424	High End Overlay Configuration Register 38	LCDC_HEOCFG38	Read/Write	0x00000000
0x00000428	High End Overlay Configuration Register 39	LCDC_HEOCFG39	Read/Write	0x00000000
0x0000042C	High End Overlay Configuration Register 40	LCDC_HEOCFG40	Read/Write	0x00000000
0x00000430	High End Overlay Configuration Register 41	LCDC_HEOCFG41	Read/Write	0x00000000
0x00000434–0x0000053C	Reserved	–	–	–
0x00000540	Post Processing Channel Enable Register	LCDC_PPCHER	Write-only	–
0x00000544	Post Processing Channel Disable Register	LCDC_PPCHDR	Write-only	–
0x00000548	Post Processing Channel Status Register	LCDC_PPCHSR	Read-only	0x00000000
0x0000054C	Post Processing Interrupt Enable Register	LCDC_PPIER	Write-only	–

Table 36-55. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x00000550	Post Processing Interrupt Disable Register	LCDC_PPIDR	Write-only	–
0x00000554	Post Processing Interrupt Mask Register	LCDC_PPIMR	Read-only	0x00000000
0x00000558	Post Processing Interrupt Status Register	LCDC_PPISR	Read-only	0x00000000
0x0000055C	Post Processing Head Register	LCDC_PPHEAD	Read/Write	0x00000000
0x00000560	Post Processing Address Register	LCDC_PPADDR	Read/Write	0x00000000
0x00000564	Post Processing Control Register	LCDC_PPCTRL	Read/Write	0x00000000
0x00000568	Post Processing Next Register	LCDC_PPNEXT	Read/Write	0x00000000
0x0000056C	Post Processing Configuration Register 0	LCDC_PPCFG0	Read/Write	0x00000000
0x00000570	Post Processing Configuration Register 1	LCDC_PPCFG1	Read/Write	0x00000000
0x00000574	Post Processing Configuration Register 2	LCDC_PPCFG2	Read/Write	0x00000000
0x00000578	Post Processing Configuration Register 3	LCDC_PPCFG3	Read/Write	0x00000000
0x0000057C	Post Processing Configuration Register 4	LCDC_PPCFG4	Read/Write	0x00000000
0x00000580	Post Processing Configuration Register 5	LCDC_PPCFG5	Read/Write	0x00000000
0x00000584–0x000005FC	Reserved	–	–	–
0x00000600	Base CLUT Register 0	LCDC_BASECLUT0	Read/Write	0x00000000
...	...	...	...	...
0x000008FC	Base CLUT Register 255	LCDC_BASECLUT255	Read/Write	0x00000000
0x00000A00	Overlay 1 CLUT Register 0	LCDC_OVR1CLUT0	Read/Write	0x00000000
...	...	...	...	...
0x00000DFC	Overlay 1 CLUT Register 255	LCDC_OVR1CLUT255	Read/Write	0x00000000
0x00000E00	Overlay 2 CLUT Register 0	LCDC_OVR2CLUT0	Read/Write	0x00000000
...	...	...	...	...
0x000011FC	Overlay 2 CLUT Register 255	LCDC_OVR2CLUT255	Read/Write	0x00000000
0x00001200	High End Overlay CLUT Register 0	LCDC_HEOCLUT0	Read/Write	0x00000000
...	...	...	...	...
0x000015FC	High End Overlay CLUT Register 255	LCDC_HEOCLUT255	Read/Write	0x00000000
0x00001600–0x00001FFC	Reserved	–	–	–

Note: 1. The CLUT registers are located in embedded RAM.

36.7.1 LCD Controller Configuration Register 0

Name: LCDC_LCDCFG0

Address: 0xF0000000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CLKDIV							

15	14	13	12	11	10	9	8
–	–	CGDISPP	–	CGDISHEO	CGDISOVR2	CGDISOVR1	CGDISBASE

7	6	5	4	3	2	1	0
–	–	–	–	CLKPWMSEL	CLKSEL	–	CLKPOL

- **CLKPOL: LCD Controller Clock Polarity**

0: Data/Control signals are launched on the rising edge of the pixel clock.

1: Data/Control signals are launched on the falling edge of the pixel clock.

- **CLKSEL: LCD Controller Clock Source Selection**

0: The asynchronous output stage of the LCD controller is fed by the System Clock.

1: The asynchronous output state of the LCD controller is fed by the 2x System Clock.

- **CLKPWMSEL: LCD Controller PWM Clock Source Selection**

0: The slow clock is selected and feeds the PWM module.

1: The system clock is selected and feeds the PWM module.

- **CGDISBASE: Clock Gating Disable Control for the Base Layer**

0: Automatic Clock Gating is enabled for the Base Layer.

1: Clock is running continuously.

- **CGDISOVR1: Clock Gating Disable Control for the Overlay 1 Layer**

0: Automatic Clock Gating is enabled for the Overlay 1 Layer.

1: Clock is running continuously.

- **CGDISOVR2: Clock Gating Disable Control for the Overlay 2 Layer**

0: Automatic Clock Gating is enabled for the Overlay 2 Layer.

1: Clock is running continuously.

- **CGDISHEO: Clock Gating Disable Control for the High End Overlay**

0: Automatic Clock Gating is enabled for the High End Overlay Layer.

1: Clock is running continuously.

- **CGDISPP: Clock Gating Disable Control for the Post Processing Layer**

0: Automatic Clock Gating is enabled for the Post Processing Layer.

1: Clock is running continuously.

- **CLKDIV: LCD Controller Clock Divider**

8-bit width clock divider for pixel clock (LCD_PCLK). The pixel clock period formula is:

LCD_PCLK = source clock / (CLKDIV+2)

where source clock is the system clock when CLKSEL is written to ‘0’ and to the 2x system_clock when CLKSEL is written to ‘1’.

36.7.2 LCD Controller Configuration Register 1

Name: LCDC_LCDCFG1

Address: 0xF0000004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	VSPW	

23	22	21	20	19	18	17	16
VSPW							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	HSPW	

7	6	5	4	3	2	1	0
HSPW							

- **HSPW: Horizontal Synchronization Pulse Width**

Width of the LCD_HSYNC pulse, given in pixel clock cycles. Width is (HSPW+1) LCD_PCLK cycles.

- **VSPW: Vertical Synchronization Pulse Width**

Width of the LCD_VSYNC pulse, given in number of lines. Width is (VSPW+1) lines.

36.7.3 LCD Controller Configuration Register 2

Name: LCDC_LCDCFG2

Address: 0xF0000008

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	VBPW	

23	22	21	20	19	18	17	16
VBPW							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	VFPW	

7	6	5	4	3	2	1	0
VFPW							

- **VFPW: Vertical Front Porch Width**

This field indicates the number of lines at the end of the Frame. The blanking interval is equal to (VFPW+1) lines.

- **VBPW: Vertical Back Porch Width**

This field indicates the number of lines at the beginning of the Frame. The blanking interval is equal to VBPW lines.

36.7.4 LCD Controller Configuration Register 3

Name: LCDC_LCDCFG3

Address: 0xF000000C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	HBPW	

23	22	21	20	19	18	17	16
HBPW							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	HFPW	

7	6	5	4	3	2	1	0
HFPW							

- **HFPW: Horizontal Front Porch Width**

Number of pixel clock cycles inserted at the end of the active line. The interval is equal to (HFPW+1) LCD_PCLK cycles.

- **HBPW: Horizontal Back Porch Width**

Number of pixel clock cycles inserted at the beginning of the line. The interval is equal to (HBPW+1) LCD_PCLK cycles.

36.7.5 LCD Controller Configuration Register 4

Name: LCDC_LCDCFG4

Address: 0xF0000010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	RPF		

23	22	21	20	19	18	17	16
RPF							

15	14	13	12	11	10	9	8
–	–	–	–	–	PPL		

7	6	5	4	3	2	1	0
PPL							

- **RPF: Number of Active Row Per Frame**

Number of active lines in the frame. The frame height is equal to (RPF+1) lines.

- **PPL: Number of Pixels Per Line**

Number of pixel in the frame. The number of active pixels in the frame is equal to (PPL+1) pixels.

36.7.6 LCD Controller Configuration Register 5

Name: LCDC_LCDCFG5

Address: 0xF0000014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GUARDTIME							

15	14	13	12	11	10	9	8
–	–	VSPHO	VSPSU	–	PP	MODE	

7	6	5	4	3	2	1	0
DISPDLY	DITHER	–	DISPPOL	VSPDLYE	VSPDLYS	VSPOL	HSPOL

- **HSPOL: Horizontal Synchronization Pulse Polarity**

0: Active High

1: Active Low

- **VSPOL: Vertical Synchronization Pulse Polarity**

0: Active High

1: Active Low

- **VSPDLYS: Vertical Synchronization Pulse Start**

0: The first active edge of the Vertical synchronization pulse is synchronous with the second edge of the horizontal pulse.

1: The first active edge of the Vertical synchronization pulse is synchronous with the first edge of the horizontal pulse.

- **VSPDLYE: Vertical Synchronization Pulse End**

0: The second active edge of the Vertical synchronization pulse is synchronous with the second edge of the horizontal pulse.

1: The second active edge of the Vertical synchronization pulse is synchronous with the first edge of the horizontal pulse.

- **DISPPOL: Display Signal Polarity**

0: Active High

1: Active Low

- **DITHER: LCD Controller Dithering**

0: Dithering logical unit is disabled

1: Dithering logical unit is activated

- **DISPDLY: LCD Controller Display Power Signal Synchronization**

0: The LCD_DISP signal is asserted synchronously with the second active edge of the horizontal pulse.

1: The LCD_DISP signal is asserted asynchronously with both edges of the horizontal pulse.

- **MODE: LCD Controller Output Mode**

Value	Name	Description
0	OUTPUT_12BPP	LCD Output mode is set to 12 bits per pixel
1	OUTPUT_16BPP	LCD Output mode is set to 16 bits per pixel
2	OUTPUT_18BPP	LCD Output mode is set to 18 bits per pixel
3	OUTPUT_24BPP	LCD Output mode is set to 24 bits per pixel

- **PP: Post Processing Enable**

0: The blended pixel is pushed into the output FIFO.

1: The blended pixel is written back to memory, the post-processing stage is enabled.

- **VSPSU: LCD Controller Vertical synchronization Pulse Setup Configuration**

0: The vertical synchronization pulse is asserted synchronously with horizontal pulse edge.

1: The vertical synchronization pulse is asserted one pixel clock cycle before the horizontal pulse.

- **VSPHO: LCD Controller Vertical synchronization Pulse Hold Configuration**

0: The vertical synchronization pulse is asserted synchronously with horizontal pulse edge.

1: The vertical synchronization pulse is held active one pixel clock cycle after the horizontal pulse.

- **GUARDTIME: LCD DISPLAY Guard Time**

Number of frames inserted during start up before LCD_DISP assertion.

Number of frames inserted after LCD_DISP reset.

36.7.7 LCD Controller Configuration Register 6

Name: LCDC_LCDCFG6

Address: 0xF0000018

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
PWMCVAL							
7	6	5	4	3	2	1	0
–	–	–	PWMPOL	–	PWMPS		

- **PWMPS: PWM Clock Prescaler**

Selects the configuration of the counter prescaler module.

Value	Name	Description
000	DIV_1	The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK}$
001	DIV_2	The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/2}$
010	DIV_4	The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/4}$
011	DIV_8	The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/8}$
100	DIV_16	The counter advances at a rate of $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/16}$
101	DIV_32	The counter advances at a of rate $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/32}$
110	DIV_64	The counter advances at a of rate $f_{COUNTER} = f_{PWM_SELECTED_CLOCK/64}$

- **PWMPOL: LCD Controller PWM Signal Polarity**

This bit defines the polarity of the PWM output signal.

0: The output pulses are low level.

1: The output pulses are high level (the output will be high whenever the value in the counter is less than the value CVAL).

- **PWMCVAL: LCD Controller PWM Compare Value**

PWM compare value. Used to adjust the analog value obtained after an external filter to control the contrast of the display.

36.7.8 LCD Controller Enable Register

Name: LCDC_LCDEN

Address: 0xF0000020

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	PWMEN	DISPEN	SYNCEN	CLKEN

- **CLKEN: LCD Controller Pixel Clock Enable**

0: No effect

1: Pixel clock logical unit is activated.

- **SYNCEN: LCD Controller Horizontal and Vertical Synchronization Enable**

0: No effect

1: Both horizontal and vertical synchronization (LCD_VSYNC and LCD_HSYNC) signals are generated.

- **DISPEN: LCD Controller DISP Signal Enable**

0: No effect

1: LCD_DISP signal is generated.

- **PWMEN: LCD Controller Pulse Width Modulation Enable**

0: No effect

1: PWM is enabled.

36.7.9 LCD Controller Disable Register

Name: LCDC_LCDDIS

Address: 0xF0000024

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PWMRST	DISPRST	SYNCRST	CLKRST
7	6	5	4	3	2	1	0
–	–	–	–	PWMDIS	DISPDIS	SYNCDIS	CLKDIS

- **CLKDIS: LCD Controller Pixel Clock Disable**

0: No effect.

1: Disables the pixel clock.

- **SYNCDIS: LCD Controller Horizontal and Vertical Synchronization Disable**

0: No effect.

1: Disables the synchronization signals after the end of the frame.

- **DISPDIS: LCD Controller DISP Signal Disable**

0: No effect.

1: Disables the DISP signal.

- **PWMDIS: LCD Controller Pulse Width Modulation Disable**

0: No effect.

1: Disables the pulse width modulation signal.

- **CLKRST: LCD Controller Clock Reset**

0: No effect.

1: Resets the pixel clock generator module. The pixel clock duty cycle may be violated.

- **SYNCRST: LCD Controller Horizontal and Vertical Synchronization Reset**

0: No effect.

1: Resets the timing engine. Both Horizontal and vertical pulse width are violated.

- **DISPRST: LCD Controller DISP Signal Reset**

0: No effect.

1: Resets the DISP signal.

- **PWMRST: LCD Controller PWM Reset**

0: No effect.

1: Resets the PWM module. The duty cycle may be violated.

36.7.10 LCD Controller Status Register

Name: LCDC_LCDSR

Address: 0xF0000028

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	SIPSTS	PWMSTS	DISPSTS	LCDSTS	CLKSTS

- **CLKSTS: Clock Status**

0: Pixel clock is disabled.

1: Pixel clock is running.

- **LCDSTS: LCD Controller Synchronization status**

0: Timing engine is disabled.

1: Timing engine is running.

- **DISPSTS: LCD Controller DISP Signal Status**

0: DISP is disabled.

1: DISP signal is activated.

- **PWMSTS: LCD Controller PWM Signal Status**

0: PWM is disabled.

1: PWM signal is activated.

- **SIPSTS: Synchronization In Progress**

0: Clock domain synchronization is terminated.

1: Synchronization is in progress. Access to the registers LCDC_LCDCCFG[0..6], LCDC_LCDEN and LCDC_LCDDIS has no effect.

36.7.11 LCD Controller Interrupt Enable Register

Name: LCDC_LCDIER

Address: 0xF000002C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	PPIE	–	HEOIE	OVR2IE	OVR1IE	BASEIE

7	6	5	4	3	2	1	0
–	–	–	FIFOERRIE	–	DISPIE	DISIE	SOFIE

- **SOFIE: Start of Frame Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **DISIE: LCD Disable Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **DISPIE: Power UP/Down Sequence Terminated Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **FIFOERRIE: Output FIFO Error Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **BASEIE: Base Layer Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **OVR1IE: Overlay 1 Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **OVR2IE: Overlay 2 Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **HEOIE: High End Overlay Interrupt Enable**

0: No effect.

1: Enables the interrupt.

- **PPIE: Post Processing Interrupt Enable**

0: No effect.

1: Enables the interrupt.

36.7.12 LCD Controller Interrupt Disable Register

Name: LCDC_LCDIDR

Address: 0xF0000030

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	PPID	–	HEOID	OVR2ID	OVR1ID	BASEID
7	6	5	4	3	2	1	0
–	–	–	FIFOERRID	–	DISPID	DISID	SOFID

- **SOFID: Start of Frame Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **DISID: LCD Disable Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **DISPID: Power UP/Down Sequence Terminated Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **FIFOERRID: Output FIFO Error Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **BASEID: Base Layer Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **OVR1ID: Overlay 1 Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **OVR2ID: Overlay 2 Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **HEOID: High End Overlay Interrupt Disable**

0: No effect.

1: Disables the interrupt.

- **PPID: Post Processing Interrupt Disable**

0: No effect

1: Disables the interrupt

36.7.13 LCD Controller Interrupt Mask Register

Name: LCDC_LCDIMR

Address: 0xF0000034

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	PPIM	–	HEOIM	OVR2IM	OVR1IM	BASEIM

7	6	5	4	3	2	1	0
–	–	–	FIFOERRIM	–	DISPIM	DISIM	SOFIM

- **SOFIM: Start of Frame Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **DISIM: LCD Disable Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **DISPIM: Power UP/Down Sequence Terminated Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **FIFOERRIM: Output FIFO Error Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **BASEIM: Base Layer Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **OVR1IM: Overlay 1 Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **OVR2IM: Overlay 2 Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **HEOIM: High End Overlay Interrupt Mask**

0: Interrupt source is disabled.

1: Interrupt source is enabled.

- **PPIM: Post Processing Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

36.7.14 LCD Controller Interrupt Status Register

Name: LCDC_LCDISR

Address: 0xF0000038

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	PP	–	HEO	OVR2	OVR1	BASE
7	6	5	4	3	2	1	0
–	–	–	FIFOERR	–	DISP	DIS	SOF

- **SOF: Start of Frame Interrupt Status**

0: No detection since last read of LCDC_LCDISR.

1: Indicates that a start of frame event has been detected. This flag is reset after a read operation.

- **DIS: LCD Disable Interrupt Status**

0: Horizontal and vertical timing generator has not yet been disabled.

1: Indicates that the horizontal and vertical timing generator has been disabled. This flag is reset after a read operation.

- **DISP: Power-up/Power-down Sequence Terminated Interrupt Status**

0: Power-up sequence or power-down sequence has not yet terminated.

1: Indicates the power-up sequence or power-down sequence has terminated. This flag is reset after a read operation.

- **FIFOERR: Output FIFO Error**

0: No underflow has occurred in the output FIFO since last read of LCDC_LCDISR.

1: Indicates that an underflow has occurred in the output FIFO. This flag is reset after a read operation.

- **BASE: Base Layer Raw Interrupt Status**

0: No base layer interrupt detected since last read of LCDC_BASEISR.

1: Indicates that a base layer interrupt is pending. This flag is reset as soon as the LCDC_BASEISR is read.

- **OVR1: Overlay 1 Raw Interrupt Status**

0: No Overlay 1 layer interrupt detected since last read of LCDC_OVR1ISR.

1: Indicates that an Overlay 1 layer interrupt is pending. This flag is reset as soon as the LCDC_OVR1ISR is read.

- **OVR2: Overlay 2 Raw Interrupt Status**

0: No Overlay 2 layer interrupt detected since last read of LCDC_OVR2ISR.

1: Indicates that an Overlay 2 layer interrupt is pending. This flag is reset as soon as the LCDC_OVR2ISR is read.

- **HEO: High End Overlay Raw Interrupt Status**

0: No High End layer interrupt detected since last read of LCDC_HEOISR.

1: Indicates that a High End layer interrupt is pending. This flag is reset as soon as the LCDC_HEOISR is read.

- **PP: Post Processing Raw Interrupt Status**

0: No Post Processing interrupt detected since last read of LCDC_PPISR

1: Indicates that Post Processing interrupt is pending. This flag is reset as soon as the LCDC_PPISR is read.

36.7.15 LCD Controller Attribute Register

Name: LCDC_ATTR

Address: 0xF000003C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	PPA2Q	–	HEOA2Q	OVR2A2Q	OVR1A2Q	BASEA2Q
7	6	5	4	3	2	1	0
–	–	PP	–	HEO	OVR2	OVR1	BASE

- **BASE: Base Layer Update Attribute**

0: No effect.

1: Update the BASE window attributes.

- **OVR1: Overlay 1 Update Attribute**

0: No effect.

1: Update the OVR1 window attribute.

- **OVR2: Overlay 2 Update Attribute**

0: No effect.

1: Update the OVR2 window attribute.

- **HEO: High End Overlay Update Attribute**

0: No effect.

1: Update the HEO window attribute.

- **PP: Post-Processing Update Attribute**

0: No effect.

1: Update the PP window attribute.

- **BASEA2Q: Base Layer Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_BASEHEAD register to the descriptor list.

- **OVR1A2Q: Overlay 1 Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_OVR1HEAD register to the descriptor list.

- **OVR2A2Q: Overlay 2 Update Add to Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_OVR2HEAD register to the descriptor list.

- **HEOA2Q: High End Overlay Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_HEOHEAD register to the descriptor list.

- **PPA2Q: Post-Processing Update Add To Queue**

0: No effect.

1: Add the descriptor pointed to by the LCDC_PPHEAD register to the descriptor list.

36.7.16 Base Layer Channel Enable Register

Name: LCDC_BASECHER

Address: 0xF0000040

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates windows attributes on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

36.7.17 Base Layer Channel Disable Register

Name: LCDC_BASECHDR

Address: 0xF0000044

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

36.7.18 Base Layer Channel Status Register

Name: LCDC_BASECHSR

Address: 0xF0000048

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

36.7.19 Base Layer Interrupt Enable Register

Name: LCDC_BASEIER

Address: 0xF000004C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

36.7.20 Base Layer Interrupt Disable Register

Name: LCDC_BASEIDR

Address: 0xF0000050

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

36.7.21 Base Layer Interrupt Mask Register

Name: LCDC_BASEIMR

Address: 0xF0000054

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

36.7.22 Base Layer Interrupt Status Register

Name: LCDC_BASEISR

Address: 0xF0000058

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No end of DMA transfer has been detected since last read of LCDC_BASEISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_BASEISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_BASEISR

1: The descriptor pointed to by the LCDC_BASEHEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition occurred since last read of LCDC_BASEISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_BASEISR

1: An overflow occurred. This flag is reset after a read operation.

36.7.23 Base DMA Head Register

Name: LCDC_BASEHEAD

Address: 0xF000005C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							

23	22	21	20	19	18	17	16
HEAD							

15	14	13	12	11	10	9	8
HEAD							

7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.24 Base DMA Address Register

Name: LCDC_BASEADDR

Address: 0xF0000060

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Start Address**

Frame buffer base address

36.7.25 Base DMA Control Register

Name: LCDC_BASECTRL

Address: 0xF0000064

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled

1: Transfer Descriptor fetch is enabled

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled

1: Lookup Table DMA fetch is enabled

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled

1: DMA transfer completed interrupt is disabled

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled

1: Transfer descriptor loaded interrupt is disabled

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled

1: Transfer descriptor added to queue interrupt is enabled

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled

1: End of list interrupt is enabled

36.7.26 Base DMA Next Register

Name: LCDC_BASENEXT

Address: 0xF0000068

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.27 Base Layer Configuration Register 0

Name: LCDC_BASECFG0

Address: 0xF000006C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DLBO
7	6	5	4	3	2	1	0
–	–	BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only For Channel Bus Transaction**

0: Undefined length INCR burst is used for a burst of 2 and 3 beats.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

36.7.28 Base Layer Configuration Register 1

Name: LCDC_BASECFG1

Address: 0xF0000070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	-	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–				–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	–	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

36.7.29 Base Layer Configuration Register 2

Name: LCDC_BASECFG2

Address: 0xF0000074

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

36.7.30 Base Layer Configuration Register 3

Name: LCDC_BASECFG3

Address: 0xF0000078

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RDEF							

15	14	13	12	11	10	9	8
GDEF							

7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the Base DMA channel is disabled

- **GDEF: Green Default**

Default Green color when the Base DMA channel is disabled

- **BDEF: Blue Default**

Default Blue color when the Base DMA channel is disabled

36.7.31 Base Layer Configuration Register 4

Name: LCDC_BASECFG4

Address: 0xF000007C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	DISCEN	–	REP	DMA
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **DMA: Use DMA Data Path**

0: The default color is used on the Base Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DISCEN: Discard Area Enable**

0: The whole frame is retrieved from memory.

1: The DMA channel discards the area located at screen coordinate {DISCXPOS, DISCYPOS}.

36.7.32 Base Layer Configuration Register 5

Name: LCDC_BASECFG5

Address: 0xF0000080

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	DISCYPOS		

23	22	21	20	19	18	17	16
DISCYPOS							

15	14	13	12	11	10	9	8
–	–	–	–		DISCXPOS		

7	6	5	4	3	2	1	0
DISCXPOS							

- **DISCXPOS: Discard Area Horizontal Coordinate**

Horizontal Position of the Discard Area

- **DISCYPOS: Discard Area Vertical Coordinate**

Vertical Position of the Discard Area

36.7.33 Base Layer Configuration Register 6

Name: LCDC_BASECFG6

Address: 0xF0000084

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	DISCYSIZE		

23	22	21	20	19	18	17	16
DISCYSIZE							

15	14	13	12	11	10	9	8
–	–	–	–		DISCXSIZE		

7	6	5	4	3	2	1	0
DISCXSIZE							

- **DISCXSIZE: Discard Area Horizontal Size**

Discard Horizontal size in pixels. The Discard size is set to (DISCXSIZE + 1) pixels horizontally.

- **DISCYSIZE: Discard Area Vertical Size**

Discard Vertical size in pixels. The Discard size is set to (DISCYSIZE + 1) pixels vertically.

36.7.34 Overlay 1 Channel Enable Register

Name: LCDC_OVR1CHER

Address: 0xF0000140

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates window attributes (size, alpha blending, etc.) on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

36.7.35 Overlay 1 Channel Disable Register

Name: LCDC_OVR1CHDR

Address: 0xF0000144

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

36.7.36 Overlay 1 Channel Status Register

Name: LCDC_OVR1CHSR

Address: 0xF0000148

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

36.7.37 Overlay 1 Interrupt Enable Register

Name: LCDC_OVR1IER

Address: 0xF000014C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

36.7.38 Overlay 1 Interrupt Disable Register

Name: LCDC_OVR1IDR

Address: 0xF0000150

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

36.7.39 Overlay 1 Interrupt Mask Register

Name: LCDC_OVR1IMR

Address: 0xF0000154

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

36.7.40 Overlay 1 Interrupt Status Register

Name: LCDC_OVR1ISR

Address: 0xF0000158

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No End of Transfer has been detected since last read of LCDC_OVR1ISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_OVR1ISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_OVR1ISR

1: The descriptor pointed to by the LCDC_OVR1HEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition has occurred since last read of LCDC_OVR1ISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_OVR1ISR

1: An overflow occurred. This flag is reset after a read operation.

36.7.41 Overlay 1 Head Register

Name: LCDC_OVR1HEAD

Address: 0xF000015C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							

23	22	21	20	19	18	17	16
HEAD							

15	14	13	12	11	10	9	8
HEAD							

7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.42 Overlay 1 Address Register

Name: LCDC_OVR1ADDR

Address: 0xF0000160

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Overlay 1 Address**

Overlay 1 frame buffer base address

36.7.43 Overlay 1 Control Register

Name: LCDC_OVR1CTRL

Address: 0xF0000164

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled

1: Transfer Descriptor fetch is enabled

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled

1: Lookup Table DMA fetch is enabled

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled

1: DMA transfer completed interrupt is disabled

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled

1: Transfer descriptor loaded interrupt is disabled

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled

1: Transfer descriptor added to queue interrupt is enabled

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled

1: End of list interrupt is enabled

36.7.44 Overlay 1 Next Register

Name: LCDC_OVR1NEXT

Address: 0xF0000168

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.45 Overlay 1 Configuration Register 0

Name: LCDC_OVR1CFG0

Address: 0xF000016C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	LOCKDIS	ROTDIS	–	–	–	DLBO
7	6	5	4	3	2	1	0
–	–	BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_BLEN_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_BLEN_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_BLEN_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_BLEN_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only for Channel Bus Transaction**

0: Undefined length INCR burst is used for a burst of 2 and 3 beats.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

- **ROTDIS: Hardware Rotation Optimization Disable**

0: Rotation optimization is enabled.

1: Rotation optimization is disabled.

- **LOCKDIS: Hardware Rotation Lock Disable**

0: AHB lock signal is asserted when a rotation is performed.

1: AHB lock signal is cleared when a rotation is performed.

36.7.46 Overlay 1 Configuration Register 1

Name: LCDC_OVR1CFG1

Address: 0xF0000170

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	–	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

36.7.47 Overlay 1 Configuration Register 2

Name: LCDC_OVR1CFG2

Address: 0xF0000174

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YPOS		

23	22	21	20	19	18	17	16
YPOS							

15	14	13	12	11	10	9	8
–	–	–	–	–	XPOS		

7	6	5	4	3	2	1	0
XPOS							

- **XPOS: Horizontal Window Position**

Overlay 1 Horizontal window position.

- **YPOS: Vertical Window Position**

Overlay 1 Vertical window position.

36.7.48 Overlay 1 Configuration Register 3

Name: LCDC_OVR1CFG3

Address: 0xF0000178

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YSIZE		

23	22	21	20	19	18	17	16
YSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XSIZE		

7	6	5	4	3	2	1	0
XSIZE							

- **XSIZE: Horizontal Window Size**

Overlay 1 window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

- **YSIZE: Vertical Window Size**

Overlay 1 window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

36.7.49 Overlay 1 Configuration Register 4

Name: LCDC_OVR1CFG4

Address: 0xF000017C

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

36.7.50 Overlay 1 Configuration Register 5

Name: LCDC_OVR1CFG5

Address: 0xF0000180

Access: Read/Write

31	30	29	28	27	26	25	24
PSTRIDE							

23	22	21	20	19	18	17	16
PSTRIDE							

15	14	13	12	11	10	9	8
PSTRIDE							

7	6	5	4	3	2	1	0
PSTRIDE							

- **PSTRIDE: Pixel Stride**

PSTRIDE represents the memory offset, in bytes, between two pixels of the image.

36.7.51 Overlay 1 Configuration Register 6

Name: LCDC_OVR1CFG6

Address: 0xF0000184

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
RDEF							
15	14	13	12	11	10	9	8
GDEF							
7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the Overlay 1 DMA channel is disabled.

- **GDEF: Green Default**

Default Green color when the Overlay 1 DMA channel is disabled.

- **BDEF: Blue Default**

Default Blue color when the Overlay 1 DMA channel is disabled.

36.7.52 Overlay 1 Configuration Register 7

Name: LCDC_OVR1CFG7

Address: 0xF0000188

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RKEY							

15	14	13	12	11	10	9	8
GKEY							

7	6	5	4	3	2	1	0
BKEY							

- **RKEY: Red Color Component Chroma Key**

Reference Red chroma key used to match the Red color of the current overlay.

- **GKEY: Green Color Component Chroma Key**

Reference Green chroma key used to match the Green color of the current overlay.

- **BKEY: Blue Color Component Chroma Key**

Reference Blue chroma key used to match the Blue color of the current overlay.

36.7.53 Overlay 1 Configuration Register 8

Name: LCDC_OVR1CFG8

Address: 0xF000018C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RMASK							

15	14	13	12	11	10	9	8
GMASK							

7	6	5	4	3	2	1	0
BMASK							

- **RMASK: Red Color Component Chroma Key Mask**

Red Mask used when the compare function is used. If a bit is set then this bit is compared.

- **GMASK: Green Color Component Chroma Key Mask**

Green Mask used when the compare function is used. If a bit is set then this bit is compared.

- **BMASK: Blue Color Component Chroma Key Mask**

Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

36.7.54 Overlay 1 Configuration Register 9

Name: LCDC_OVR1CFG9

Address: 0xF0000190

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GA							

15	14	13	12	11	10	9	8
–	–	–	–	–	DSTKEY	REP	DMA

7	6	5	4	3	2	1	0
OVR	LAEN	GAEN	REVALPHA	ITER	ITER2BL	INV	CRKEY

- **CRKEY: Blender Chroma Key Enable**

0: Chroma key matching is disabled.

1: Chroma key matching is enabled.

- **INV: Blender Inverted Blender Output Enable**

0: Iterated pixel is the blended pixel.

1: Iterated pixel is the inverted pixel.

- **ITER2BL: Blender Iterated Color Enable**

0: Final adder stage operand is set to 0.

1: Final adder stage operand is set to the iterated pixel value.

- **ITER: Blender Use Iterated Color**

0: Pixel difference is set to 0.

1: Pixel difference is set to the iterated pixel value.

- **REVALPHA: Blender Reverse Alpha**

0: Pixel difference is multiplied by alpha.

1: Pixel difference is multiplied by 1 - alpha.

- **GAEN: Blender Global Alpha Enable**

0: Global alpha blending coefficient is disabled.

1: Global alpha blending coefficient is enabled.

- **LAEN: Blender Local Alpha Enable**

0: Local alpha blending coefficient is disabled.

1: Local alpha blending coefficient is enabled.

- **OVR: Blender Overlay Layer Enable**

0: Overlay pixel color is set to the default overlay pixel color.

1: Overlay pixel color is set to the DMA channel pixel color.

- **DMA: Blender DMA Layer Enable**

0: The default color is used on the Overlay 1 Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DSTKEY: Destination Chroma Keying**

0: Source Chroma keying is enabled.

1: Destination Chroma keying is used.

- **GA: Blender Global Alpha**

Global alpha blender for the current layer.

36.7.55 Overlay 2 Channel Enable Register

Name: LCDC_OVR2CHER

Address: 0xF0000240

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates windows attributes on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

36.7.56 Overlay 2 Channel Disable Register

Name: LCDC_OVR2CHDR

Address: 0xF0000244

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

36.7.57 Overlay 2 Channel Status Register

Name: LCDC_OVR2CHSR

Address: 0xF0000248

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

36.7.58 Overlay 2 Interrupt Enable Register

Name: LCDC_OVR2IER

Address: 0xF000024C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

36.7.59 Overlay 2 Interrupt Disable Register

Name: LCDC_OVR2IDR

Address: 0xF0000250

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

36.7.60 Overlay 2 Interrupt Mask Register

Name: LCDC_OVR2IMR

Address: 0xF0000254

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

36.7.61 Overlay 2 Interrupt Status Register

Name: LCDC_OVR2ISR

Address: 0xF0000258

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No End of Transfer has been detected since last read of LCDC_OVR2ISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_OVR2ISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_OVR2ISR

1: The descriptor pointed to by the LCDC_OVR2HEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition occurred since last read of LCDC_OVR2ISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_OVR2ISR

1: An overflow occurred. This flag is reset after a read operation.

36.7.62 Overlay 2 Head Register

Name: LCDC_OVR2HEAD

Address: 0xF000025C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							
23	22	21	20	19	18	17	16
HEAD							
15	14	13	12	11	10	9	8
HEAD							
7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.63 Overlay 2 Address Register

Name: LCDC_OVR2ADDR

Address: 0xF0000260

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Overlay 2 Address**

Overlay 2 frame buffer base address.

36.7.64 Overlay 2 Control Register

Name: LCDC_OVR2CTRL

Address: 0xF0000264

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled.

1: Lookup Table DMA fetch is enabled.

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

36.7.65 Overlay 2 Next Register

Name: LCDC_OVR2NEXT

Address: 0xF0000268

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.66 Overlay 2 Configuration Register 0

Name: LCDC_OVR2CFG0

Address: 0xF000026C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	LOCKDIS	ROTDIS	–	–	–	DLBO

7	6	5	4	3	2	1	0
–	–	BLEN		–	–	–	–

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only For Channel Bus Transaction**

0: Undefined length INCR burst is used for 2 and 3 beats burst.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

- **ROTDIS: Hardware Rotation Optimization Disable**

0: Rotation optimization is enabled.

1: Rotation optimization is disabled.

- **LOCKDIS: Hardware Rotation Lock Disable**

0: AHB lock signal is asserted when a rotation is performed.

1: AHB lock signal is cleared when a rotation is performed.

36.7.67 Overlay 2 Configuration Register 1

Name: LCDC_OVR2CFG1

Address: 0xF0000270

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	–	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

36.7.68 Overlay 2 Configuration Register 2

Name: LCDC_OVR2CFG2

Address: 0xF0000274

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YPOS		

23	22	21	20	19	18	17	16
YPOS							

15	14	13	12	11	10	9	8
–	–	–	–	–	XPOS		

7	6	5	4	3	2	1	0
XPOS							

- **XPOS: Horizontal Window Position**

Overlay 2 Horizontal window position.

- **YPOS: Vertical Window Position**

Overlay 2 Vertical window position.

36.7.69 Overlay 2 Configuration Register 3

Name: LCDC_OVR2CFG3

Address: 0xF0000278

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YSIZE		

23	22	21	20	19	18	17	16
YSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XSIZE		

7	6	5	4	3	2	1	0
XSIZE							

- **XSIZE: Horizontal Window Size**

Overlay 2 window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

- **YSIZE: Vertical Window Size**

Overlay 2 window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

36.7.70 Overlay 2 Configuration Register 4

Name: LCDC_OVR2CFG4

Address: 0xF000027C

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

36.7.71 Overlay 2 Configuration Register 5

Name: LCDC_OVR2CFG5

Address: 0xF0000280

Access: Read/Write

31	30	29	28	27	26	25	24
PSTRIDE							

23	22	21	20	19	18	17	16
PSTRIDE							

15	14	13	12	11	10	9	8
PSTRIDE							

7	6	5	4	3	2	1	0
PSTRIDE							

- **PSTRIDE: Pixel Stride**

PSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

36.7.72 Overlay 2 Configuration Register 6

Name: LCDC_OVR2CFG6

Address: 0xF0000284

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
RDEF							
15	14	13	12	11	10	9	8
GDEF							
7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the Overlay 1 DMA channel is disabled.

- **GDEF: Green Default**

Default Green color when the Overlay 1 DMA channel is disabled.

- **BDEF: Blue Default**

Default Blue color when the Overlay 1 DMA channel is disabled.

36.7.73 Overlay 2 Configuration Register 7

Name: LCDC_OVR2CFG7

Address: 0xF0000288

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RKEY							

15	14	13	12	11	10	9	8
GKEY							

7	6	5	4	3	2	1	0
BKEY							

- **RKEY: Red Color Component Chroma Key**

Reference Red chroma key used to match the Red color of the current overlay.

- **GKEY: Green Color Component Chroma Key**

Reference Green chroma key used to match the Green color of the current overlay.

- **BKEY: Blue Color Component Chroma Key**

Reference Blue chroma key used to match the Blue color of the current overlay.

36.7.74 Overlay 2 Configuration Register 8

Name: LCDC_OVR2CFG8

Address: 0xF000028C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RMASK							

15	14	13	12	11	10	9	8
GMASK							

7	6	5	4	3	2	1	0
BMASK							

- **RMASK: Red Color Component Chroma Key Mask**

Red Mask used when the compare function is used. If a bit is set then this bit is compared.

- **GMASK: Green Color Component Chroma Key Mask**

Green Mask used when the compare function is used. If a bit is set then this bit is compared.

- **BMASK: Blue Color Component Chroma Key Mask**

Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

36.7.75 Overlay 2 Configuration Register 9

Name: LCDC_OVR2CFG9

Address: 0xF0000290

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GA							

15	14	13	12	11	10	9	8
–	–	–	–	–	DSTKEY	REP	DMA

7	6	5	4	3	2	1	0
OVR	LAEN	GAEN	REVALPHA	ITER	ITER2BL	INV	CRKEY

- **CRKEY: Blender Chroma Key Enable**

0: Chroma key matching is disabled.

1: Chroma key matching is enabled.

- **INV: Blender Inverted Blender Output Enable**

0: Iterated pixel is the blended pixel.

1: Iterated pixel is the inverted pixel.

- **ITER2BL: Blender Iterated Color Enable**

0: Final adder stage operand is set to 0.

1: Final adder stage operand is set to the iterated pixel value.

- **ITER: Blender Use Iterated Color**

0: Pixel difference is set to 0.

1: Pixel difference is set to the iterated pixel value.

- **REVALPHA: Blender Reverse Alpha**

0: Pixel difference is multiplied by alpha.

1: Pixel difference is multiplied by 1 - alpha.

- **GAEN: Blender Global Alpha Enable**

0: Global alpha blending coefficient is disabled.

1: Global alpha blending coefficient is enabled.

- **LAEN: Blender Local Alpha Enable**

0: Local alpha blending coefficient is disabled.

1: Local alpha blending coefficient is enabled.

- **OVR: Blender Overlay Layer Enable**

0: Overlay pixel color is set to the default overlay pixel color.

1: Overlay pixel color is set to the DMA channel pixel color.

- **DMA: Blender DMA Layer Enable**

0: The default color is used on the Overlay 1 Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DSTKEY: Destination Chroma Keying**

0: Source Chroma keying is enabled.

1: Destination Chroma keying is used.

- **GA: Blender Global Alpha**

Global alpha blender for the current layer.

36.7.76 High End Overlay Channel Enable Register

Name: LCDC_HEOCHER

Address: 0xF0000340

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates windows attributes on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

36.7.77 High End Overlay Channel Disable Register

Name: LCDC_HEOCHDR

Address: 0xF0000344

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

36.7.78 High End Overlay Channel Status Register

Name: LCDC_HEOCHSR

Address: 0xF0000348

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

36.7.79 High End Overlay Interrupt Enable Register

Name: LCDC_HEOIER

Address: 0xF000034C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UDMA: End of DMA Transfer for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UDSCR: Descriptor Loaded for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UADD: Head Descriptor Loaded for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UDONE: End of List for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **UOVR: Overflow for U or UV Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VDMA: End of DMA for V Chrominance Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VDSCR: Descriptor Loaded for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VADD: Head Descriptor Loaded for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VDONE: End of List for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **VOVR: Overflow for V Chrominance Interrupt Enable**

0: No effect

1: Interrupt source is enabled

36.7.80 High End Overlay Interrupt Disable Register

Name: LCDC_HEOIDR

Address: 0xF0000350

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **OVR: Overflow Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UDMA: End of DMA Transfer for U or UV Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UDSCR: Descriptor Loaded for U or UV Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UADD: Head Descriptor Loaded for U or UV Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **UDONE: End of List Interrupt for U or UV Chrominance Component Disable**

0: No effect

1: Interrupt source is disabled

- **UOVR: Overflow Interrupt for U or UV Chrominance Component Disable**

0: No effect

1: Interrupt source is disabled

- **VDMA: End of DMA Transfer for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VDSCR: Descriptor Loaded for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VADD: Head Descriptor Loaded for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VDONE: End of List for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **VOVR: Overflow for V Chrominance Component Interrupt Disable**

0: No effect

1: Interrupt source is disabled

36.7.81 High End Overlay Interrupt Mask Register

Name: LCDC_HEOIMR

Address: 0xF0000354

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **OVR: Overflow Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UDMA: End of DMA Transfer for U or UV Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UDSCR: Descriptor Loaded for U or UV Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UADD: Head Descriptor Loaded for U or UV Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UDONE: End of List for U or UV Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **UOVR: Overflow for U Chrominance Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VDMA: End of DMA Transfer for V Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VDSCR: Descriptor Loaded for V Chrominance Component Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VADD: Head Descriptor Loaded for V Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VDONE: End of List for V Chrominance Component Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **VOVR: Overflow for V Chrominance Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

36.7.82 High End Overlay Interrupt Status Register

Name: LCDC_HEOISR

Address: 0xF0000358

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	VOVR	VDONE	VADD	VDSCR	VDMA	–	–

15	14	13	12	11	10	9	8
–	UOVR	UDONE	UADD	UDSCR	UDMA	–	–

7	6	5	4	3	2	1	0
–	OVR	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No end of transfer has been detected since last read of LCDC_HEOISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: The descriptor pointed to by the LCDC_HEOHEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition occurred since last read of LCDC_HEOISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **OVR: Overflow Detected**

0: No overflow occurred since last read of LCDC_HEOISR

1: An overflow occurred. This flag is reset after a read operation.

- **UDMA: End of DMA Transfer for U Component**

0: No End of Transfer has been detected since last read of LCDC_HEOISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **UDSCR: DMA Descriptor Loaded for U Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **UADD: Head Descriptor Loaded for U Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: The descriptor pointed to by the LCDC_HEOUHEAD register has been loaded successfully. This flag is reset after a read operation.

- **UDONE: End of List Detected for U Component**

0: No End of List condition occurred since last read of LCDC_HEOISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **UOVR: Overflow Detected for U Component**

0: No overflow occurred since last read of LCDC_HEOISR

1: An overflow occurred. This flag is reset after a read operation.

- **VDMA: End of DMA Transfer for V Component**

0: No End of Transfer has been detected since last read of LCDC_HEOISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **VDSCR: DMA Descriptor Loaded for V Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **VADD: Head Descriptor Loaded for V Component**

0: No descriptor has been loaded since last read of LCDC_HEOISR

1: The descriptor pointed to by the LCDC_HEOVHEAD register has been loaded successfully. This flag is reset after a read operation.

- **VDONE: End of List Detected for V Component**

0: No End of List condition occurred since last read of LCDC_HEOISR

1: End of List condition has occurred. This flag is reset after a read operation.

- **VOVR: Overflow Detected for V Component**

0: No overflow occurred since last read of LCDC_HEOISR

1: An overflow occurred. This flag is reset after a read operation.

36.7.83 High End Overlay DMA Head Register

Name: LCDC_HEOHEAD

Address: 0xF000035C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							
23	22	21	20	19	18	17	16
HEAD							
15	14	13	12	11	10	9	8
HEAD							
7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.84 High End Overlay DMA Address Register

Name: LCDC_HEOADDR

Address: 0xF0000360

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Start Address**

Frame Buffer Base Address.

36.7.85 High End Overlay DMA Control Register

Name: LCDC_HEOCTRL

Address: 0xF0000364

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	LFETCH	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **LFETCH: Lookup Table Fetch Enable**

0: Lookup Table DMA fetch is disabled.

1: Lookup Table DMA fetch is enabled.

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

36.7.86 High End Overlay DMA Next Register

Name: LCDC_HEONEXT

Address: 0xF0000368

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.87 High End Overlay U-UV DMA Head Register

Name: LCDC_HEOUHEAD

Address: 0xF000036C

Access: Read/Write

31	30	29	28	27	26	25	24
UHEAD							

23	22	21	20	19	18	17	16
UHEAD							

15	14	13	12	11	10	9	8
UHEAD							

7	6	5	4	3	2	1	0
UHEAD							

- **UHEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.88 High End Overlay U-UV DMA Address Register

Name: LCDC_HEOUADDR

Address: 0xF0000370

Access: Read/Write

31	30	29	28	27	26	25	24
UADDR							

23	22	21	20	19	18	17	16
UADDR							

15	14	13	12	11	10	9	8
UADDR							

7	6	5	4	3	2	1	0
UADDR							

- **UADDR: DMA Transfer Start Address for U or UV Chrominance**

U or UV frame buffer address.

36.7.89 High End Overlay U-UV DMA Control Register

Name: LCDC_HEOUCTRL

Address: 0xF0000374

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	UDONEIEN	UADDIEN	UDSCRIEN	UDMAIEN	–	UDFETCH

- **UDFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **UDMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **UDSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **UADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **UDONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

36.7.90 High End Overlay U-UV DMA Next Register

Name: LCDC_HEOUNEXT

Address: 0xF0000378

Access: Read/Write

31	30	29	28	27	26	25	24
UNEXT							

23	22	21	20	19	18	17	16
UNEXT							

15	14	13	12	11	10	9	8
UNEXT							

7	6	5	4	3	2	1	0
UNEXT							

- **UNEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.91 High End Overlay V DMA Head Register

Name: LCDC_HEOVHEAD

Address: 0xF000037C

Access: Read/Write

31	30	29	28	27	26	25	24
VHEAD							

23	22	21	20	19	18	17	16
VHEAD							

15	14	13	12	11	10	9	8
VHEAD							

7	6	5	4	3	2	1	0
VHEAD							

- **VHEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.92 High End Overlay V DMA Address Register

Name: LCDC_HEOVADDR

Address: 0xF0000380

Access: Read/Write

31	30	29	28	27	26	25	24
VADDR							

23	22	21	20	19	18	17	16
VADDR							

15	14	13	12	11	10	9	8
VADDR							

7	6	5	4	3	2	1	0
VADDR							

- **VADDR: DMA Transfer Start Address for V Chrominance**

Frame Buffer Base Address.

36.7.93 High End Overlay V DMA Control Register

Name: LCDC_HEOVCTRL

Address: 0xF0000384

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	VDONEIEN	VADDIEN	VDSCRIEN	VDMAIEN	–	VDFETCH

- **VDFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **VDMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **VDSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **VADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **VDONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

36.7.94 High End Overlay V DMA Next Register

Name: LCDC_HEOVNEXT

Address: 0xF0000388

Access: Read/Write

31	30	29	28	27	26	25	24
VNEXT							

23	22	21	20	19	18	17	16
VNEXT							

15	14	13	12	11	10	9	8
VNEXT							

7	6	5	4	3	2	1	0
VNEXT							

- **VNEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.95 High End Overlay Configuration Register 0

Name: LCDC_HEOCFG0

Address: 0xF000038C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	LOCKDIS	ROTDIS	–	–	–	DLBO

7	6	5	4	3	2	1	0
BLENUV		BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_BLEN_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_BLEN_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_BLEN_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_BLEN_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **BLENUV: AHB Burst Length for U-V Channel**

Value	Name	Description
0	AHB_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only For Channel Bus Transaction**

0: Undefined length INCR burst is used for a burst of 2 and 3 beats.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

- **ROTDIS: Hardware Rotation Optimization Disable**

0: Rotation optimization is enabled.

1: Rotation optimization is disabled.

- **LOCKDIS: Hardware Rotation Lock Disable**

0: AHB lock signal is asserted when a rotation is performed.

1: AHB lock signal is cleared when a rotation is performed.

36.7.96 High End Overlay Configuration Register 1

Name: LCDC_HEOCFG1

Address: 0xF0000390

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	DSCALEOPT	–	–	YUV422SWP	YUV422ROT

15	14	13	12	11	10	9	8
YUVMODE				–	–	CLUTMODE	

7	6	5	4	3	2	1	0
RGBMODE				–	–	YUVEN	CLUTEN

- **CLUTEN: Color Lookup Table Mode Enable**

0: RGB mode is selected.

1: Color Lookup Table mode is selected.

- **YUVEN: YUV Color Space Enable**

0: Color space is RGB

1: Color Space is YUV

- **RGBMODE: RGB Mode Input Selection**

Value	Name	Description
0	12BPP_RGB_444	12 bpp RGB 444
1	16BPP_ARGB_4444	16 bpp ARGB 4444
2	16BPP_RGBA_4444	16 bpp RGBA 4444
3	16BPP_RGB_565	16 bpp RGB 565
4	16BPP_TRGB_1555	16 bpp TRGB 1555
5	18BPP_RGB_666	18 bpp RGB 666
6	18BPP_RGB_666PACKED	18 bpp RGB 666 PACKED
7	19BPP_TRGB_1666	19 bpp TRGB 1666
8	19BPP_TRGB_PACKED	19 bpp TRGB 1666 PACKED
9	24BPP_RGB_888	24 bpp RGB 888
10	24BPP_RGB_888_PACKED	24 bpp RGB 888 PACKED
11	25BPP_TRGB_1888	25 bpp TRGB 1888
12	32BPP_ARGB_8888	32 bpp ARGB 8888
13	32BPP_RGBA_8888	32 bpp RGBA 8888

- **CLUTMODE: Color Lookup Table Mode Input Selection**

Value	Name	Description
0	CLUT_1BPP	Color Lookup Table mode set to 1 bit per pixel
1	CLUT_2BPP	Color Lookup Table mode set to 2 bits per pixel
2	CLUT_4BPP	Color Lookup Table mode set to 4 bits per pixel
3	CLUT_8BPP	Color Lookup Table mode set to 8 bits per pixel

- **YUVMODE: YUV Mode Input Selection**

Value	Name	Description
0	32BPP_AYCBCR	32 bpp AYCbCr 444
1	16BPP_YCBCR_MODE0	16 bpp Cr(n)Y(n+1)Cb(n)Y(n) 422
2	16BPP_YCBCR_MODE1	16 bpp Y(n+1)Cr(n)Y(n)Cb(n) 422
3	16BPP_YCBCR_MODE2	16 bpp Cb(n)Y(+1)Cr(n)Y(n) 422
4	16BPP_YCBCR_MODE3	16 bpp Y(n+1)Cb(n)Y(n)Cr(n) 422
5	16BPP_YCBCR_SEMIPLANAR	16 bpp Semiplanar 422 YCbCr
6	16BPP_YCBCR_PLANAR	16 bpp Planar 422 YCbCr
7	12BPP_YCBCR_SEMIPLANAR	12 bpp Semiplanar 420 YCbCr
8	12BPP_YCBCR_PLANAR	12 bpp Planar 420 YCbCr

- **YUV422ROT: YUV 4:2:2 Rotation**

0: Chroma Upsampling kernel is configured to use 0 and 180 degrees algorithm

1: Indicates that the Chroma Upsampling kernel is configured to use the 4:2:2 Rotation Algorithm. This bit is relevant only when a rotation angle of 90 degrees or 270 degrees is used.

- **YUV422SWP: YUV 4:2:2 Swap**

0: The two Y components of the YUV 4:2:2 packed data stream are not swapped.

1: The two Y components of the YUV 4:2:2 packed data stream are swapped.

- **DSCALEOPT: Down Scaling Bandwidth Optimization**

0: Scaler Optimization is disabled.

1: Scaler Optimization is enabled, only relevant pixels are retrieved from memory to fill the scaler filter.

36.7.97 High End Overlay Configuration Register 2

Name: LCDC_HEOCFG2

Address: 0xF0000394

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YPOS		

23	22	21	20	19	18	17	16
YPOS							

15	14	13	12	11	10	9	8
–	–	–	–	–	XPOS		

7	6	5	4	3	2	1	0
XPOS							

- **XPOS: Horizontal Window Position**

High End Overlay Horizontal window position.

- **YPOS: Vertical Window Position**

High End Overlay Vertical window position.

36.7.98 High End Overlay Configuration Register 3

Name: LCDC_HEOCFG3

Address: 0xF0000398

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YSIZE		

23	22	21	20	19	18	17	16
YSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XSIZE		

7	6	5	4	3	2	1	0
XSIZE							

- **XSIZE: Horizontal Window Size**

High End Overlay window width in pixels. The window width is set to (XSIZE + 1).

The following constraint must be met: XPOS + XSIZE ≤ PPL

- **YSIZE: Vertical Window Size**

High End Overlay window height in pixels. The window height is set to (YSIZE + 1).

The following constraint must be met: YPOS + YSIZE ≤ RPF

36.7.99 High End Overlay Configuration Register 4

Name: LCDC_HEOCFG4

Address: 0xF000039C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YMEMSIZE		

23	22	21	20	19	18	17	16
YMEMSIZE							

15	14	13	12	11	10	9	8
–	–	–	–	–	XMEMSIZE		

7	6	5	4	3	2	1	0
XMEMSIZE							

- **XMEMSIZE: Horizontal image Size in Memory**

High End Overlay image width in pixels. The image width is set to (XMEMSIZE + 1).

- **YMEMSIZE: Vertical image Size in Memory**

High End Overlay image height in pixels. The image height is set to (YMEMSIZE + 1).

36.7.100 High End Overlay Configuration Register 5

Name: LCDC_HEOCFG5

Address: 0xF00003A0

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

36.7.101 High End Overlay Configuration Register 6

Name: LCDC_HEOCFG6

Address: 0xF00003A4

Access: Read/Write

31	30	29	28	27	26	25	24
PSTRIDE							

23	22	21	20	19	18	17	16
PSTRIDE							

15	14	13	12	11	10	9	8
PSTRIDE							

7	6	5	4	3	2	1	0
PSTRIDE							

- **PSTRIDE: Pixel Stride**

PSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

36.7.102 High End Overlay Configuration Register 7

Name: LCDC_HEOCFG7

Address: 0xF00003A8

Access: Read/Write

31	30	29	28	27	26	25	24
UVXSTRIDE							

23	22	21	20	19	18	17	16
UVXSTRIDE							

15	14	13	12	11	10	9	8
UVXSTRIDE							

7	6	5	4	3	2	1	0
UVXSTRIDE							

- **UVXSTRIDE: UV Horizontal Stride**

UVXSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

36.7.103 High End Overlay Configuration Register 8

Name: LCDC_HEOCFG8

Address: 0xF00003AC

Access: Read/Write

31	30	29	28	27	26	25	24
UVPSTRIDE							

23	22	21	20	19	18	17	16
UVPSTRIDE							

15	14	13	12	11	10	9	8
UVPSTRIDE							

7	6	5	4	3	2	1	0
UVPSTRIDE							

- **UVPSTRIDE: UV Pixel Stride**

UVPSTRIDE represents the memory offset, in bytes, between two pixels of the image memory.

36.7.104 High End Overlay Configuration Register 9

Name: LCDC_HEOCFG9

Address: 0xF00003B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RDEF							

15	14	13	12	11	10	9	8
GDEF							

7	6	5	4	3	2	1	0
BDEF							

- **RDEF: Red Default**

Default Red color when the High End Overlay DMA channel is disabled.

- **GDEF: Green Default**

Default Green color when the High End Overlay DMA channel is disabled.

- **BDEF: Blue Default**

Default Blue color when the High End Overlay DMA channel is disabled.

36.7.105 High End Overlay Configuration Register 10

Name: LCDC_HEOCFG10

Address: 0xF00003B4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RKEY							

15	14	13	12	11	10	9	8
GKEY							

7	6	5	4	3	2	1	0
BKEY							

- **RKEY: Red Color Component Chroma Key**

Reference Red chroma key used to match the Red color of the current overlay.

- **GKEY: Green Color Component Chroma Key**

Reference Green chroma key used to match the Green color of the current overlay.

- **BKEY: Blue Color Component Chroma Key**

Reference Blue chroma key used to match the Blue color of the current overlay.

36.7.106 High End Overlay Configuration Register 11

Name: LCDC_HEOCFG11

Address: 0xF00003B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RMASK							

15	14	13	12	11	10	9	8
GMASK							

7	6	5	4	3	2	1	0
BMASK							

- **RMASK: Red Color Component Chroma Key Mask**

Red Mask used when the compare function is used. If a bit is set then this bit is compared.

- **GMASK: Green Color Component Chroma Key Mask**

Green Mask used when the compare function is used. If a bit is set then this bit is compared.

- **BMASK: Blue Color Component Chroma Key Mask**

Blue Mask used when the compare function is used. If a bit is set then this bit is compared.

36.7.107 High End Overlay Configuration Register 12

Name: LCDC_HEOCFG12

Address: 0xF00003BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
GA							

15	14	13	12	11	10	9	8
–	–	–	VIDPRI	–	DSTKEY	REP	DMA

7	6	5	4	3	2	1	0
OVR	LAEN	GAEN	REVALPHA	ITER	ITER2BL	INV	CRKEY

- **CRKEY: Blender Chroma Key Enable**

0: Chroma key matching is disabled.

1: Chroma key matching is enabled.

- **INV: Blender Inverted Blender Output Enable**

0: Iterated pixel is the blended pixel.

1: Iterated pixel is the inverted pixel.

- **ITER2BL: Blender Iterated Color Enable**

0: Final adder stage operand is set to 0.

1: Final adder stage operand is set to the iterated pixel value.

- **ITER: Blender Use Iterated Color**

0: Pixel difference is set to 0.

1: Pixel difference is set to the iterated pixel value.

- **REVALPHA: Blender Reverse Alpha**

0: Pixel difference is multiplied by alpha.

1: Pixel difference is multiplied by 1 - alpha.

- **GAEN: Blender Global Alpha Enable**

0: Global alpha blending coefficient is disabled.

1: Global alpha blending coefficient is enabled.

- **LAEN: Blender Local Alpha Enable**

0: Local alpha blending coefficient is disabled.

1: Local alpha blending coefficient is enabled.

- **OVR: Blender Overlay Layer Enable**

0: Overlay pixel color is set to the default overlay pixel color.

1: Overlay pixel color is set to the DMA channel pixel color.

- **DMA: Blender DMA Layer Enable**

0: The default color is used on the Overlay 1 Layer.

1: The DMA channel retrieves the pixels stream from the memory.

- **REP: Use Replication logic to expand RGB color to 24 bits**

0: When the selected pixel depth is less than 24 bpp the pixel is shifted and least significant bits are set to 0.

1: When the selected pixel depth is less than 24 bpp the pixel is shifted and the least significant bit replicates the msb.

- **DSTKEY: Destination Chroma Keying**

0: Source Chroma keying is enabled.

1: Destination Chroma keying is used.

- **VIDPRI: Video Priority**

0: OVR1 layer is above HEO layer.

1: OVR1 layer is below HEO layer.

- **GA: Blender Global Alpha**

Global alpha blender for the current layer.

36.7.108 High End Overlay Configuration Register 13

Name: LCDC_HEOCFG13

Address: 0xF00003C0

Access: Read/Write

31	30	29	28	27	26	25	24
SCALEN	–	YFACTOR					

23	22	21	20	19	18	17	16
YFACTOR							

15	14	13	12	11	10	9	8
–	–	XFACTOR					

7	6	5	4	3	2	1	0
XFACTOR							

- **SCALEN: Hardware Scaler Enable**

0: Scaler is disabled

1: Scaler is enabled.

- **YFACTOR: Vertical Scaling Factor**

Scaler Vertical Factor.

- **XFACTOR: Horizontal Scaling Factor**

Scaler Horizontal Factor.

36.7.109 High End Overlay Configuration Register 14

Name: LCDC_HEOCFG14

Address: 0xF00003C4

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCYOFF	CSCRV					

23	22	21	20	19	18	17	16
CSCRV				CSCRU			

15	14	13	12	11	10	9	8
CSCRU						CSCRY	

7	6	5	4	3	2	1	0
CSCRY							

- **CSCRY: Color Space Conversion Y coefficient for Red Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCRU: Color Space Conversion U coefficient for Red Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCRV: Color Space Conversion V coefficient for Red Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCYOFF: Color Space Conversion Offset**

0: Offset is set to 0

1: Offset is set to 16

36.7.110 High End Overlay Configuration Register 15

Name: LCDC_HEOCFG15

Address: 0xF00003C8

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCUOFF	CSCGV					

23	22	21	20	19	18	17	16
CSCGV				CSCGU			

15	14	13	12	11	10	9	8
CSCGU						CSCGY	

7	6	5	4	3	2	1	0
CSCGY							

- **CSCGY: Color Space Conversion Y coefficient for Green Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCGU: Color Space Conversion U coefficient for Green Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCGV: Color Space Conversion V coefficient for Green Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCUOFF: Color Space Conversion Offset**

0: Offset is set to 0

1: Offset is set to 128

36.7.111 High End Overlay Configuration Register 16

Name: LCDC_HEOCFG16

Address: 0xF00003CC

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCVOFF	CSCBV					

23	22	21	20	19	18	17	16
CSCBV				CSCBU			

15	14	13	12	11	10	9	8
CSCBU						CSCBY	

7	6	5	4	3	2	1	0
CSCBY							

- **CSCBY: Color Space Conversion Y coefficient for Blue Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCBU: Color Space Conversion U coefficient for Blue Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCBV: Color Space Conversion V coefficient for Blue Component 1:2:7 format**

Color Space Conversion coefficient format is 1 sign bit, 2 magnitude bits and 7 fractional bits.

- **CSCVOFF: Color Space Conversion Offset**

0: Offset is set to 0

1: Offset is set to 128

36.7.112 High End Overlay Configuration Register 17

Name: LCDC_HEOCFG17

Address: 0xF00003D0

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI0COEFF3							
23	22	21	20	19	18	17	16
XPHI0COEFF2							
15	14	13	12	11	10	9	8
XPHI0COEFF1							
7	6	5	4	3	2	1	0
XPHI0COEFF0							

- **XPHI0COEFF0: Horizontal Coefficient for phase 0 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI0COEFF1: Horizontal Coefficient for phase 0 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI0COEFF2: Horizontal Coefficient for phase 0 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI0COEFF3: Horizontal Coefficient for phase 0 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.113 High End Overlay Configuration Register 18

Name: LCDC_HEOCFG18

Address: 0xF00003D4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI0COEFF4							

- **XPHI0COEFF4: Horizontal Coefficient for phase 0 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.114 High End Overlay Configuration Register 19

Name: LCDC_HEOCFG19

Address: 0xF00003D8

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI1COEFF3							
23	22	21	20	19	18	17	16
XPHI1COEFF2							
15	14	13	12	11	10	9	8
XPHI1COEFF1							
7	6	5	4	3	2	1	0
XPHI1COEFF0							

- **XPHI1COEFF0: Horizontal Coefficient for phase 1 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI1COEFF1: Horizontal Coefficient for phase 1 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI1COEFF2: Horizontal Coefficient for phase 1 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI1COEFF3: Horizontal Coefficient for phase 1 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.115 High End Overlay Configuration Register 20

Name: LCDC_HEOCFG20

Address: 0xF00003DC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI1COEFF4							

- **XPHI1COEFF4: Horizontal Coefficient for phase 1 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.116 High End Overlay Configuration Register 21

Name: LCDC_HEOCFG21

Address: 0xF00003E0

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI2COEFF3							
23	22	21	20	19	18	17	16
XPHI2COEFF2							
15	14	13	12	11	10	9	8
XPHI2COEFF1							
7	6	5	4	3	2	1	0
XPHI2COEFF0							

- **XPHI2COEFF0: Horizontal Coefficient for phase 2 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI2COEFF1: Horizontal Coefficient for phase 2 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI2COEFF2: Horizontal Coefficient for phase 2 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI2COEFF3: Horizontal Coefficient for phase 2 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.117 High End Overlay Configuration Register 22

Name: LCDC_HEOCFG22

Address: 0xF00003E4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI2COEFF4							

- **XPHI2COEFF4: Horizontal Coefficient for phase 2 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.118 High End Overlay Configuration Register 23

Name: LCDC_HEOCFG23

Address: 0xF00003E8

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI3COEFF3							

23	22	21	20	19	18	17	16
XPHI3COEFF2							

15	14	13	12	11	10	9	8
XPHI3COEFF1							

7	6	5	4	3	2	1	0
XPHI3COEFF0							

- **XPHI3COEFF0: Horizontal Coefficient for phase 3 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI3COEFF1: Horizontal Coefficient for phase 3 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI3COEFF2: Horizontal Coefficient for phase 3 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI3COEFF3: Horizontal Coefficient for phase 3 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.119 High End Overlay Configuration Register 24

Name: LCDC_HEOCFG24

Address: 0xF00003EC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI3COEFF4							

- **XPHI3COEFF4: Horizontal Coefficient for phase 3 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.120 High End Overlay Configuration Register 25

Name: LCDC_HEOCFG25

Address: 0xF00003F0

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI4COEFF3							

23	22	21	20	19	18	17	16
XPHI4COEFF2							

15	14	13	12	11	10	9	8
XPHI4COEFF1							

7	6	5	4	3	2	1	0
XPHI4COEFF0							

- **XPHI4COEFF0: Horizontal Coefficient for phase 4 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI4COEFF1: Horizontal Coefficient for phase 4 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI4COEFF2: Horizontal Coefficient for phase 4 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI4COEFF3: Horizontal Coefficient for phase 4 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.121 High End Overlay Configuration Register 26

Name: LCDC_HEOCFG26

Address: 0xF00003F4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI4COEFF4							

- **XPHI4COEFF4: Horizontal Coefficient for phase 4 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.122 High End Overlay Configuration Register 27

Name: LCDC_HEOCFG27

Address: 0xF00003F8

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI5COEFF3							

23	22	21	20	19	18	17	16
XPHI5COEFF2							

15	14	13	12	11	10	9	8
XPHI5COEFF1							

7	6	5	4	3	2	1	0
XPHI5COEFF0							

- **XPHI5COEFF0: Horizontal Coefficient for phase 5 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI5COEFF1: Horizontal Coefficient for phase 5 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI5COEFF2: Horizontal Coefficient for phase 5 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI5COEFF3: Horizontal Coefficient for phase 5 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.123 High End Overlay Configuration Register 28

Name: LCDC_HEOCFG28

Address: 0xF00003FC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI5COEFF4							

- **XPHI5COEFF4: Horizontal Coefficient for phase 5 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.124 High End Overlay Configuration Register 29

Name: LCDC_HEOCFG29

Address: 0xF0000400

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI6COEFF3							
23	22	21	20	19	18	17	16
XPHI6COEFF2							
15	14	13	12	11	10	9	8
XPHI6COEFF1							
7	6	5	4	3	2	1	0
XPHI6COEFF0							

- **XPHI6COEFF0: Horizontal Coefficient for phase 6 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI6COEFF1: Horizontal Coefficient for phase 6 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI6COEFF2: Horizontal Coefficient for phase 6 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI6COEFF3: Horizontal Coefficient for phase 6 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.125 High End Overlay Configuration Register 30

Name: LCDC_HEOCFG30

Address: 0xF0000404

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI6COEFF4							

- **XPHI6COEFF4: Horizontal Coefficient for phase 6 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.126 High End Overlay Configuration Register 31

Name: LCDC_HEOCFG31

Address: 0xF0000408

Access: Read/Write

31	30	29	28	27	26	25	24
XPHI7COEFF3							

23	22	21	20	19	18	17	16
XPHI7COEFF2							

15	14	13	12	11	10	9	8
XPHI7COEFF1							

7	6	5	4	3	2	1	0
XPHI7COEFF0							

- **XPHI7COEFF0: Horizontal Coefficient for phase 7 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI7COEFF1: Horizontal Coefficient for phase 7 tap 1**

Coefficient format is 1 sign bit and 7 fractional bits.

- **XPHI7COEFF2: Horizontal Coefficient for phase 7 tap 2**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **XPHI7COEFF3: Horizontal Coefficient for phase 7 tap 3**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.127 High End Overlay Configuration Register 32

Name: LCDC_HEOCFG32

Address: 0xF000040C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
XPHI7COEFF4							

- **XPHI7COEFF4: Horizontal Coefficient for phase 7 tap 4**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.128 High End Overlay Configuration Register 33

Name: LCDC_HEOCFG33

Address: 0xF0000410

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI0COEFF2							

15	14	13	12	11	10	9	8
YPHI0COEFF1							

7	6	5	4	3	2	1	0
YPHI0COEFF0							

- **YPHI0COEFF0: Vertical Coefficient for phase 0 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI0COEFF1: Vertical Coefficient for phase 0 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI0COEFF2: Vertical Coefficient for phase 0 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.129 High End Overlay Configuration Register 34

Name: LCDC_HEOCFG34

Address: 0xF0000414

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI1COEFF2							

15	14	13	12	11	10	9	8
YPHI1COEFF1							

7	6	5	4	3	2	1	0
YPHI1COEFF0							

- **YPHI1COEFF0: Vertical Coefficient for phase 1 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI1COEFF1: Vertical Coefficient for phase 1 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI1COEFF2: Vertical Coefficient for phase 1 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.130 High End Overlay Configuration Register 35

Name: LCDC_HEOCFG35

Address: 0xF0000418

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI2COEFF2							

15	14	13	12	11	10	9	8
YPHI2COEFF1							

7	6	5	4	3	2	1	0
YPHI2COEFF0							

- **YPHI2COEFF0: Vertical Coefficient for phase 2 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI2COEFF1: Vertical Coefficient for phase 2 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI2COEFF2: Vertical Coefficient for phase 2 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.131 High End Overlay Configuration Register 36

Name: LCDC_HEOCFG36

Address: 0xF000041C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI3COEFF2							

15	14	13	12	11	10	9	8
YPHI3COEFF1							

7	6	5	4	3	2	1	0
YPHI3COEFF0							

- **YPHI3COEFF0: Vertical Coefficient for phase 3 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI3COEFF1: Vertical Coefficient for phase 3 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI3COEFF2: Vertical Coefficient for phase 3 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.132 High End Overlay Configuration Register 37

Name: LCDC_HEOCFG37

Address: 0xF0000420

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI4COEFF2							

15	14	13	12	11	10	9	8
YPHI4COEFF1							

7	6	5	4	3	2	1	0
YPHI4COEFF0							

- **YPHI4COEFF0: Vertical Coefficient for phase 4 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI4COEFF1: Vertical Coefficient for phase 4 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI4COEFF2: Vertical Coefficient for phase 4 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.133 High End Overlay Configuration Register 38

Name: LCDC_HEOCFG38

Address: 0xF0000424

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI5COEFF2							

15	14	13	12	11	10	9	8
YPHI5COEFF1							

7	6	5	4	3	2	1	0
YPHI5COEFF0							

- **YPHI5COEFF0: Vertical Coefficient for phase 5 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI5COEFF1: Vertical Coefficient for phase 5 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI5COEFF2: Vertical Coefficient for phase 5 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.134 High End Overlay Configuration Register 39

Name: LCDC_HEOCFG39

Address: 0xF0000428

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
YPHI6COEFF2							
15	14	13	12	11	10	9	8
YPHI6COEFF1							
7	6	5	4	3	2	1	0
YPHI6COEFF0							

- **YPHI6COEFF0: Vertical Coefficient for phase 6 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI6COEFF1: Vertical Coefficient for phase 6 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI6COEFF2: Vertical Coefficient for phase 6 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.135 High End Overlay Configuration Register 40

Name: LCDC_HEOCFG40

Address: 0xF000042C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
YPHI7COEFF2							

15	14	13	12	11	10	9	8
YPHI7COEFF1							

7	6	5	4	3	2	1	0
YPHI7COEFF0							

- **YPHI7COEFF0: Vertical Coefficient for phase 7 tap 0**

Coefficient format is 1 sign bit and 7 fractional bits.

- **YPHI7COEFF1: Vertical Coefficient for phase 7 tap 1**

Coefficient format is 1 magnitude bit and 7 fractional bits.

- **YPHI7COEFF2: Vertical Coefficient for phase 7 tap 2**

Coefficient format is 1 sign bit and 7 fractional bits.

36.7.136 High End Overlay Configuration Register 41

Name: LCDC_HEOCFG41

Address: 0xF0000430

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	YPHIDEF		

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	XPHIDEF		

- **XPHIDEF: Horizontal Filter Phase Offset**

XPHIDEF defines the index of the first coefficient set used when the horizontal resampling operation is started.

- **YPHIDEF: Vertical Filter Phase Offset**

XPHIDEF defines the index of the first coefficient set used when the vertical resampling operation is started.

36.7.137 Post Processing Channel Enable Register

Name: LCDC_PPCHER

Address: 0xF0000540

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	A2QEN	UPDATEEN	CHEN

- **CHEN: Channel Enable**

0: No effect

1: Enables the DMA channel

- **UPDATEEN: Update Overlay Attributes Enable**

0: No effect

1: Updates windows attributes on the next start of frame.

- **A2QEN: Add To Queue Enable**

0: No effect

1: Indicates that a valid descriptor has been written to memory, its memory location should be written to the DMA head pointer. The A2QSR status bit is set to one, and it is reset by hardware as soon as the descriptor pointed to by the DMA head pointer is added to the list.

36.7.138 Post Processing Channel Disable Register

Name: LCDC_PPCHDR

Address: 0xF0000544

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CHRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CHDIS

- **CHDIS: Channel Disable**

0: No effect

1: Disables the layer at the end of the current frame. The frame is completed.

- **CHRST: Channel Reset**

0: No effect

1: Resets the layer immediately. The frame is aborted.

36.7.139 Post Processing Channel Status Register

Name: LCDC_PPCHSR

Address: 0xF0000548

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	A2QSR	UPDATESR	CHSR

- **CHSR: Channel Status**

0: Layer disabled

1: Layer enabled

- **UPDATESR: Update Overlay Attributes In Progress Status**

0: No update pending

1: Overlay attributes will be updated on the next frame

- **A2QSR: Add To Queue Status**

0: Add to queue not pending

1: Add to queue pending

36.7.140 Post Processing Interrupt Enable Register

Name: LCDC_PPIER

Address: 0xF000054C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Enable**

0: No effect

1: Interrupt source is enabled

- **DONE: End of List Interrupt Enable**

0: No effect

1: Interrupt source is enabled

36.7.141 Post Processing Interrupt Disable Register

Name: LCDC_PPIDR

Address: 0xF0000550

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DSCR: Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **ADD: Head Descriptor Loaded Interrupt Disable**

0: No effect

1: Interrupt source is disabled

- **DONE: End of List Interrupt Disable**

0: No effect

1: Interrupt source is disabled

36.7.142 Post Processing Interrupt Mask Register

Name: LCDC_PPIMR

Address: 0xF0000554

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DSCR: Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **ADD: Head Descriptor Loaded Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

- **DONE: End of List Interrupt Mask**

0: Interrupt source is disabled

1: Interrupt source is enabled

36.7.143 Post Processing Interrupt Status Register

Name: LCDC_PPISR

Address: 0xF0000558

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	DONE	ADD	DSCR	DMA	–	–

- **DMA: End of DMA Transfer**

0: No End of Transfer has been detected since last read of LCDC_PPISR

1: End of Transfer has been detected. This flag is reset after a read operation.

- **DSCR: DMA Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_PPISR

1: A descriptor has been loaded successfully. This flag is reset after a read operation.

- **ADD: Head Descriptor Loaded**

0: No descriptor has been loaded since last read of LCDC_PPISR

1: The descriptor pointed to by the LCDC_PPHEAD register has been loaded successfully. This flag is reset after a read operation.

- **DONE: End of List Detected**

0: No End of List condition has occurred since last read of LCDC_PPISR

1: End of List condition has occurred. This flag is reset after a read operation.

36.7.144 Post Processing Head Register

Name: LCDC_PPHEAD

Address: 0xF000055C

Access: Read/Write

31	30	29	28	27	26	25	24
HEAD							

23	22	21	20	19	18	17	16
HEAD							

15	14	13	12	11	10	9	8
HEAD							

7	6	5	4	3	2	1	0
HEAD						–	–

- **HEAD: DMA Head Pointer**

The Head Pointer points to a new descriptor.

36.7.145 Post Processing Address Register

Name: LCDC_PPADDR

Address: 0xF0000560

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: DMA Transfer Start Address**

Post Processing Destination frame buffer address.

36.7.146 Post Processing Control Register

Name: LCDC_PPCTRL

Address: 0xF0000564

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	DONEIEN	ADDIEN	DSCRIEN	DMAIEN	–	DFETCH

- **DFETCH: Transfer Descriptor Fetch Enable**

0: Transfer Descriptor fetch is disabled.

1: Transfer Descriptor fetch is enabled.

- **DMAIEN: End of DMA Transfer Interrupt Enable**

0: DMA transfer completed interrupt is enabled.

1: DMA transfer completed interrupt is disabled.

- **DSCRIEN: Descriptor Loaded Interrupt Enable**

0: Transfer descriptor loaded interrupt is enabled.

1: Transfer descriptor loaded interrupt is disabled.

- **ADDIEN: Add Head Descriptor to Queue Interrupt Enable**

0: Transfer descriptor added to queue interrupt is enabled.

1: Transfer descriptor added to queue interrupt is enabled.

- **DONEIEN: End of List Interrupt Enable**

0: End of list interrupt is disabled.

1: End of list interrupt is enabled.

36.7.147 Post Processing Next Register

Name: LCDC_PPNEXT

Address: 0xF0000568

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							

23	22	21	20	19	18	17	16
NEXT							

15	14	13	12	11	10	9	8
NEXT							

7	6	5	4	3	2	1	0
NEXT							

- **NEXT: DMA Descriptor Next Address**

The transfer descriptor address must be aligned on a 64-bit boundary.

36.7.148 Post Processing Configuration Register 0

Name: LCDC_PPCFG0

Address: 0xF000056C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DLBO
7	6	5	4	3	2	1	0
–	–	BLEN		–	–	–	SIF

- **SIF: Source Interface**

0: Base Layer data is retrieved through AHB interface 0.

1: Base Layer data is retrieved through AHB interface 1.

- **BLEN: AHB Burst Length**

Value	Name	Description
0	AHB_BLEN_SINGLE	AHB Access is started as soon as there is enough space in the FIFO to store one data. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.
1	AHB_BLEN_INCR4	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 4 data. An AHB INCR4 Burst is used. SINGLE, INCR and INCR4 bursts are used. INCR is used for a burst of 2 and 3 beats.
2	AHB_BLEN_INCR8	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 8 data. An AHB INCR8 Burst is used. SINGLE, INCR, INCR4 and INCR8 bursts are used. INCR is used for a burst of 2 and 3 beats.
3	AHB_BLEN_INCR16	AHB Access is started as soon as there is enough space in the FIFO to store a total amount of 16 data. An AHB INCR16 Burst is used. SINGLE, INCR, INCR4, INCR8 and INCR16 bursts are used. INCR is used for a burst of 2 and 3 beats.

- **DLBO: Defined Length Burst Only For Channel Bus Transaction**

0: Undefined length INCR burst is used for 2 and 3 beats burst.

1: Only Defined Length burst is used (SINGLE, INCR4, INCR8 and INCR16).

36.7.149 Post Processing Configuration Register 1

Name: LCDC_PPCFG1

Address: 0xF0000570

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	ITUBT601	–	PPMODE		

- **PPMODE: Post Processing Output Format Selection**

Value	Name	Description
0	PPMODE_RGB_16BPP	RGB 16 bpp
1	PPMODE_RGB_24BPP_PACKED	RGB 24 bpp PACKED
2	PPMODE_RGB_24BPP_UNPACKED	RGB 24 bpp UNPACKED
3	PPMODE_YCBCR_422_MODE0	YCbCr 422 16 bpp (Mode 0)
4	PPMODE_YCBCR_422_MODE1	YCbCr 422 16 bpp (Mode 1)
5	PPMODE_YCBCR_422_MODE2	YCbCr 422 16 bpp (Mode 2)
6	PPMODE_YCBCR_422_MODE3	YCbCr 422 16 bpp (Mode 3)

- **ITUBT601: Color Space Conversion Luminance**

0: Luminance and chrominance range is [0;255]

1: Luminance values are clamped to [16;235] range. Chrominance values are clamped to [16;240] range.

36.7.150 Post Processing Configuration Register 2

Name: LCDC_PPCFG2

Address: 0xF0000574

Access: Read/Write

31	30	29	28	27	26	25	24
XSTRIDE							

23	22	21	20	19	18	17	16
XSTRIDE							

15	14	13	12	11	10	9	8
XSTRIDE							

7	6	5	4	3	2	1	0
XSTRIDE							

- **XSTRIDE: Horizontal Stride**

XSTRIDE represents the memory offset, in bytes, between two rows of the image memory.

36.7.151 Post Processing Configuration Register 3

Name: LCDC_PPCFG3

Address: 0xF0000578

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCYOFF	CSCYB					

23	22	21	20	19	18	17	16
CSCYB				CSCYG			

15	14	13	12	11	10	9	8
CSCYG						CSCYR	

7	6	5	4	3	2	1	0
CSCYR							

- **CSCYR: Color Space Conversion R coefficient for Luminance component, signed format, step set to 1/1024**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCYG: Color Space Conversion G coefficient for Luminance component, signed format, step set to 1/512**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCYB: Color Space Conversion B coefficient for Luminance component, signed format, step set to 1/1024**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCYOFF: Color Space Conversion Luminance Offset**

0: The *Yoff* parameter value is set to 0.

1: The *Yoff* parameter value is set to 16.

36.7.152 Post Processing Configuration Register 4

Name: LCDC_PPCFG4

Address: 0xF000057C

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCUOFF	CSCUB					

23	22	21	20	19	18	17	16
CSCUB				CSCUG			

15	14	13	12	11	10	9	8
CSCUG						CSCUR	

7	6	5	4	3	2	1	0
CSCUR							

- **CSCUR: Color Space Conversion R coefficient for Chrominance B component, signed format. (step 1/1024)**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCUG: Color Space Conversion G coefficient for Chrominance B component, signed format. (step 1/512)**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCUB: Color Space Conversion B coefficient for Chrominance B component, signed format. (step 1/512)**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCUOFF: Color Space Conversion Chrominance B Offset**

0: The *Cboff* parameter value is set to 0.

1: The *Cboff* parameter value is set to 128.

36.7.153 Post Processing Configuration Register 5

Name: LCDC_PPCFG5

Address: 0xF0000580

Access: Read/Write

31	30	29	28	27	26	25	24
–	CSCVOFF	CSCVB					

23	22	21	20	19	18	17	16
CSCVB				CSCVG			

15	14	13	12	11	10	9	8
CSCVG						CSCVR	

7	6	5	4	3	2	1	0
CSCVR							

- **CSCVR: Color Space Conversion R coefficient for Chrominance R component, signed format. (step 1/1024)**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCVG: Color Space Conversion G coefficient for Chrominance R component, signed format. (step 1/512)**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCVB: Color Space Conversion B coefficient for Chrominance R component, signed format. (step 1/1024)**

Color Space Conversion coefficient format is 1 sign bit, 9 fractional bits.

- **CSCVOFF: Color Space Conversion Chrominance R Offset**

0: The *Croff* parameter value is set to 0.

1: The *Croff* parameter value is set to 128.

36.7.154 Base CLUT Register x

Name: LCDC_BASECLUTx [x=0..255]

Address: 0xF0000600

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

36.7.155 Overlay 1 CLUT Register x

Name: LCDC_OVR1CLUTx [x=0..255]

Address: 0xF0000A00

Access: Read/Write

31	30	29	28	27	26	25	24
ACLUT							

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

- **ACLUT: Alpha Color Entry**

This field indicates the 8-bit width Alpha channel of the color lookup table.

36.7.156 Overlay 2 CLUT Register x

Name: LCDC_OVR2CLUTx [x=0..255]

Address: 0xF0000E00

Access: Read/Write

31	30	29	28	27	26	25	24
ACLUT							

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

- **ACLUT: Alpha Color Entry**

This field indicates the 8-bit width Alpha channel of the color lookup table.

36.7.157 High End Overlay CLUT Register x

Name: LCDC_HEOCLUTx [x=0..255]

Address: 0xF0001200

Access: Read/Write

31	30	29	28	27	26	25	24
ACLUT							

23	22	21	20	19	18	17	16
RCLUT							

15	14	13	12	11	10	9	8
GCLUT							

7	6	5	4	3	2	1	0
BCLUT							

- **BCLUT: Blue Color Entry**

This field indicates the 8-bit width Blue color of the color lookup table.

- **GCLUT: Green Color Entry**

This field indicates the 8-bit width Green color of the color lookup table.

- **RCLUT: Red Color Entry**

This field indicates the 8-bit width Red color of the color lookup table.

- **ACLUT: Alpha Color Entry**

This field indicates the 8-bit width Alpha channel of the color lookup table.

37. Ethernet MAC (GMAC)

37.1 Description

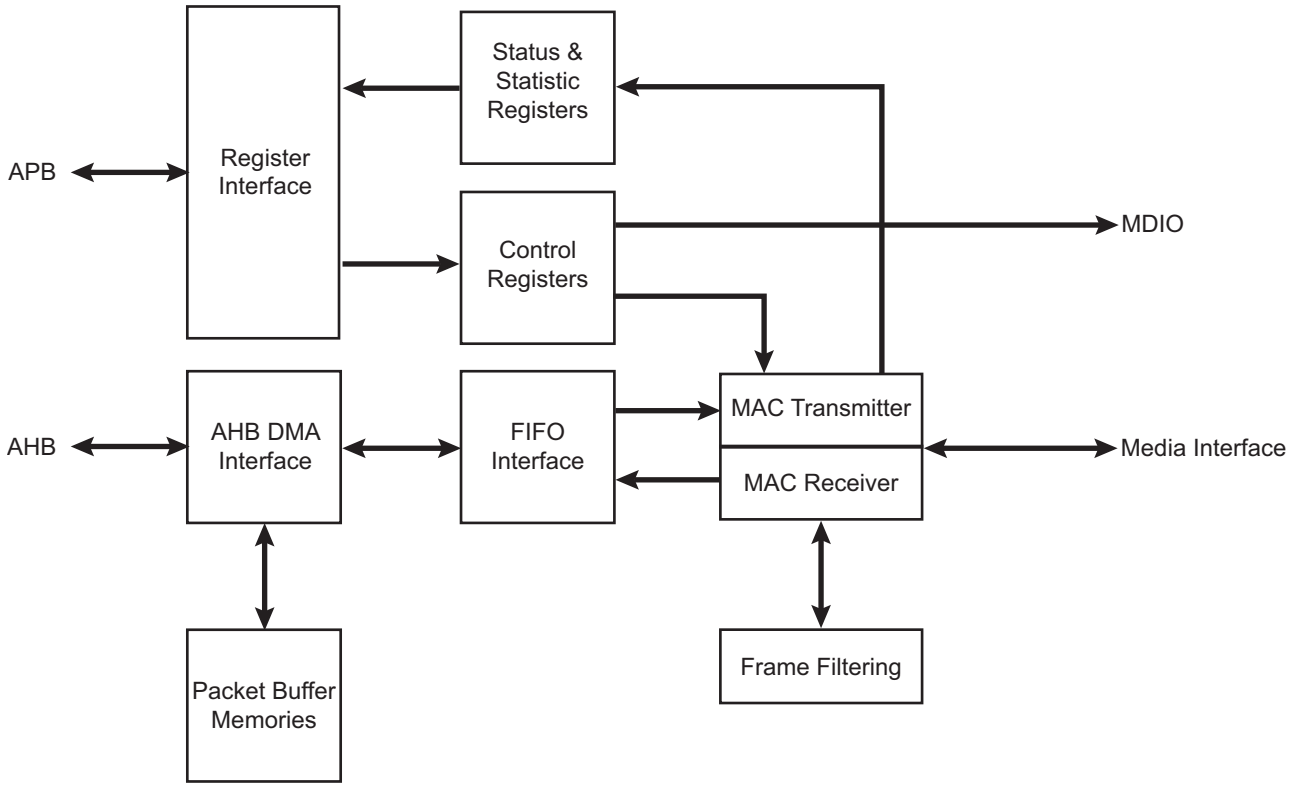
The Ethernet MAC (GMAC) module implements a 10/100 Mbps Ethernet MAC compatible with the IEEE 802.3 standard. The GMAC can operate in either half or full duplex mode at all supported speeds. The GMAC Network Configuration Register is used to select the speed, duplex mode and interface type (MII, RMII).

37.2 Embedded Characteristics

- Compatible with IEEE Standard 802.3
- 10, 100 Mbps Operation
- Full and Half Duplex Operation at all Supported Speeds of Operation
- Statistics Counter Registers for RMON/MIB
- MII/RMII Interface to the Physical Layer
- Integrated Physical Coding
- Direct Memory Access (DMA) Interface to External Memory
- Support for 3 Priority Queues in DMA
- 8 Kbytes Transmit RAM (2 KB for Queue 0, 2 KB for Queue 1, 4 KB for Queue 2) and 4 Kbytes Receive RAM
- Programmable Burst Length and Endianism for DMA
- Interrupt Generation to Signal Receive and Transmit Completion, Errors or Other Events
- Automatic Pad and Cyclic Redundancy Check (CRC) Generation on Transmitted Frames
- Automatic Discard of Frames Received with Errors
- Receive and Transmit IP, TCP and UDP Checksum Offload. Both IPv4 and IPv6 Packet Types Supported
- Address Checking Logic for Four Specific 48-bit Addresses, Four Type IDs, Promiscuous Mode, Hash Matching of Unicast and Multicast Destination Addresses and Wake-on-LAN
- Management Data Input/Output (MDIO) Interface for Physical Layer Management
- Support for Jumbo Frames up to 10240 Bytes
- Full Duplex Flow Control with Recognition of Incoming Pause Frames and Hardware Generation of Transmitted Pause Frames
- Half Duplex Flow Control by Forcing Collisions on Incoming Frames
- Support for 802.1Q VLAN Tagging with Recognition of Incoming VLAN and Priority Tagged Frames
- Support for 802.1Qbb Priority-based Flow Control
- Programmable Inter Packet Gap (IPG) Stretch
- Recognition of IEEE 1588 PTP Frames
- IEEE 1588 Time Stamp Unit (TSU)
- Support for 802.1AS Timing and Synchronization
- Supports 802.1Qav Traffic Shaping on Two Highest Priority Queues

37.3 Block Diagram

Figure 37-1. Block Diagram



37.4 Signal Interfaces

The GMAC includes the following signal interfaces:

- MII, RMII to an external PHY
- MDIO interface for external PHY management
- Slave APB interface for accessing GMAC registers
- Master AHB interface for memory access
- GTSUCOMP signal for TSU timer count value comparison

Table 37-1. GMAC Connections in Different Modes

Signal Name	Function	MII	RMII
GTXCK[(1)]	Transmit Clock or Reference Clock	TXCK	REFCK
GTXEN	Transmit Enable	TXEN	TXEN
GTX[3..0]	Transmit Data	TXD[3:0]	TXD[1:0]
GTXER	Transmit Coding Error	TXER	Not Used
GRXCK	Receive Clock	RXCK	Not Used
GRXDV	Receive Data Valid	RXDV	CRSDV
GRX[3..0]	Receive Data	RXD[3:0]	RXD[1:0]
GRXER	Receive Error	RXER	RXER
GCRS	Carrier Sense and Data Valid	CRS	Not Used

Table 37-1. GMAC Connections in Different Modes (Continued)

Signal Name	Function	MII	RMII
GCOL	Collision Detect	COL	Not Used
GMDC	Management Data Clock	MDC	MDC
GMDIO	Management Data Input/Output	MDIO	MDIO

Note: 1. Input only. GTXCK must be provided with a 25 MHz / 50 MHz external crystal oscillator for MII / RMII interfaces, respectively.

37.5 Product Dependencies

37.5.1 I/O Lines

The pins used for interfacing the GMAC may be multiplexed with PIO lines. The programmer must first program the PIO Controller to assign the pins to their peripheral function. If I/O lines of the GMAC are not used by the application, they can be used for other purposes by the PIO Controller.

Table 37-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
GMAC	GCOL	PB9	F
GMAC	GCOL	PC23	B
GMAC	GCOL	PD4	D
GMAC	GCRS	PB8	F
GMAC	GCRS	PC22	B
GMAC	GCRS	PD3	D
GMAC	GMDC	PB22	F
GMAC	GMDC	PC18	B
GMAC	GMDC	PD17	D
GMAC	GMDIO	PB23	F
GMAC	GMDIO	PC19	B
GMAC	GMDIO	PD18	D
GMAC	GRXCK	PB7	F
GMAC	GRXCK	PC20	B
GMAC	GRXCK	PD1	D
GMAC	GRXDV	PB16	F
GMAC	GRXDV	PC12	B
GMAC	GRXDV	PD11	D
GMAC	GRXER	PB17	F
GMAC	GRXER	PC13	B
GMAC	GRXER	PD12	D
GMAC	GRX0	PB18	F
GMAC	GRX0	PC14	B
GMAC	GRX0	PD13	D
GMAC	GRX1	PB19	F

Table 37-2. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
GMAC	GRX1	PC15	B
GMAC	GRX1	PD14	D
GMAC	GRX2	PB10	F
GMAC	GRX2	PC24	B
GMAC	GRX2	PD5	D
GMAC	GRX3	PB11	F
GMAC	GRX3	PC25	B
GMAC	GRX3	PD6	D
GMAC	GTSUCOMP	PB5	F
GMAC	GTSUCOMP	PC9	B
GMAC	GTSUCOMP	PD0	D
GMAC	GTXCK	PB14	F
GMAC	GTXCK	PC10	B
GMAC	GTXCK	PD9	D
GMAC	GTXEN	PB15	F
GMAC	GTXEN	PC11	B
GMAC	GTXEN	PD10	D
GMAC	GTXER	PB6	F
GMAC	GTXER	PC21	B
GMAC	GTXER	PD2	D
GMAC	GTX0	PB20	F
GMAC	GTX0	PC16	B
GMAC	GTX0	PD15	D
GMAC	GTX1	PB21	F
GMAC	GTX1	PC17	B
GMAC	GTX1	PD16	D
GMAC	GTX2	PB12	F
GMAC	GTX2	PC26	B
GMAC	GTX2	PD7	D
GMAC	GTX3	PB13	F
GMAC	GTX3	PC27	B
GMAC	GTX3	PD8	D

37.5.2 Power Management

The GMAC is not continuously clocked. The user must first enable the GMAC clock in the Power Management Controller before using it.

37.5.3 Interrupt Sources

The GMAC interrupt line is connected to one of the internal sources of the interrupt controller. Using the GMAC interrupt requires prior programming of the interrupt controller.

The GMAC features 3 interrupt sources. Refer to the table "Peripheral Identifiers" in the section "Peripherals" for the interrupt numbers for GMAC priority queues.

Table 37-3. Peripheral IDs

Instance	ID
GMAC	5

37.6 Functional Description

37.6.1 Media Access Controller

The Media Access Controller (MAC) transmit block takes data from FIFO, adds preamble and, if necessary, pad and frame check sequence (FCS). Both half duplex and full duplex Ethernet modes of operation are supported. When operating in half duplex mode, the MAC transmit block generates data according to the carrier sense multiple access with collision detect (CSMA/CD) protocol. The start of transmission is deferred if carrier sense (CRS) is active. If collision (COL) becomes active during transmission, a jam sequence is asserted and the transmission is retried after a random back off. The CRS and COL signals have no effect in full duplex mode.

The MAC receive block checks for valid preamble, FCS, alignment and length, and presents received frames to the MAC address checking block and FIFO. Software can configure the GMAC to receive jumbo frames up to 10240 bytes. It can optionally strip CRC from the received frame prior to transfer to FIFO.

The address checker recognizes four specific 48-bit addresses, can recognize four different type ID values, and contains a 64-bit Hash register for matching multicast and unicast addresses as required. It can recognize the broadcast address of all ones and copy all frames. The MAC can also reject all frames that are not VLAN tagged and recognize Wake on LAN events.

The MAC receive block supports offloading of IP, TCP and UDP checksum calculations (both IPv4 and IPv6 packet types supported), and can automatically discard bad checksum frames.

37.6.2 1588 Time Stamp Unit

The 1588 time stamp unit (TSU) is implemented as a 94-bit timer.

The 48 upper bits [93:46] of the timer count seconds and are accessible in the "GMAC 1588 Timer Seconds High Register" (GMAC_TSH) and "GMAC 1588 Timer Seconds Low Register" (GMAC_TSL). The 30 lower bits [45:16] of the timer count nanoseconds and are accessible in the "GMAC 1588 Timer Nanoseconds Register" (GMAC_TN). The lowest 16 bits [15:0] of the timer count sub-nanoseconds.

The 46 lower bits roll over when they have counted to one second. The timer increments by a programmable period (to approximately 15.2 femtoseconds resolution) with each MCK period and can also be adjusted in 1ns resolution (incremented or decremented) through APB register accesses.

37.6.3 AHB Direct Memory Access Interface

The GMAC DMA controller is connected to the MAC FIFO interface and provides a scatter-gather type capability for packet data storage.

The DMA implements packet buffering where dual-port memories are used to buffer multiple frames.

37.6.3.1 Packet Buffer DMA

- Easier to guarantee maximum line rate due to the ability to store multiple frames in the packet buffer, where the number of frames is limited by the amount of packet buffer memory and Ethernet frame size
- Full store and forward, or partial store and forward programmable options (partial store will cater for shorter latency requirements)
- Support for Transmit TCP/IP checksum offload
- Support for priority queueing
- When a collision on the line occurs during transmission, the packet will be automatically replayed directly from the packet buffer memory rather than having to re-fetch through the AHB (full store and forward ONLY)
- Received errored packets are automatically dropped before any of the packet is presented to the AHB (full store and forward ONLY), thus reducing AHB activity
- Supports manual RX packet flush capabilities
- Optional RX packet flush when there is lack of AHB resource

37.6.3.2 Partial Store and Forward Using Packet Buffer DMA

The DMA uses SRAM-based packet buffers, and can be programmed into a low latency mode, known as Partial Store and Forward. This allows for a reduced latency as the full packet is not buffered before forwarding. Note that this option is only available when the device is configured for full duplex operation.

This feature is enabled via the programmable TX and RX Partial Store and Forward registers. When the transmit Partial Store and Forward mode is activated, the transmitter will only begin to forward the packet to the MAC when there is enough packet data stored in the packet buffer. Likewise, when the receive Partial Store and Forward mode is activated, the receiver will only begin to forward the packet to the AHB when enough packet data is stored in the packet buffer. The amount of packet data required to activate the forwarding process is programmable via watermark registers which are located at the same address as the partial store and forward enable bits.

Note that the minimum operational value for the TX partial store and forward watermark is 20. There is no operational limit for the RX partial store and forward watermark. Enabling partial store and forward is a useful means to reduce latency, but there are performance implications. The GMAC DMA uses separate transmit and receive lists of buffer descriptors, with each descriptor describing a buffer area in memory. This allows Ethernet packets to be broken up and scattered around the AHB memory space.

37.6.3.3 Receive AHB Buffers

Received frames, optionally including FCS, are written to receive AHB buffers stored in memory. The receive buffer depth is programmable in the range of 64 bytes to 16 Kbytes through the DMA Configuration register, with the default being 128 bytes.

The start location for each receive AHB buffer is stored in memory in a list of receive buffer descriptors at an address location pointed to by the receive buffer queue pointer. The base address for the receive buffer queue pointer is configured in software using the Receive Buffer Queue Base Address register.

Each list entry consists of two words. The first is the address of the receive AHB buffer and the second the receive status. If the length of a receive frame exceeds the AHB buffer length, the status word for the used buffer is written with zeroes except for the “start of frame” bit, which is always set for the first buffer in a frame. Bit zero of the address field is written to 1 to show the buffer has been used. The receive buffer manager then reads the location of the next receive AHB buffer and fills that with the next part of the received frame data. AHB buffers are filled until the frame is complete and the final buffer descriptor status word contains the complete frame status. Refer to Table 37-4 for details of the receive buffer descriptor list.

Each receive AHB buffer start location is a word address. The start of the first AHB buffer in a frame can be offset by up to three bytes, depending on the value written to bits 14 and 15 of the Network Configuration register. If the start location of the AHB buffer is offset, the available length of the first AHB buffer is reduced by the corresponding number of bytes.

Table 37-4. Receive Buffer Descriptor Entry

Bit	Function
	Word 0
31:2	Address of beginning of buffer
1	Wrap—marks last descriptor in receive buffer descriptor list.
0	Ownership—needs to be zero for the GMAC to write data to the receive buffer. The GMAC sets this to one once it has successfully written a frame to memory. Software has to clear this bit before the buffer can be used again.
	Word 1
31	Global all ones broadcast address detected
30	Multicast hash match
29	Unicast hash match
28	–
27	Specific Address Register match found, bit 25 and bit 26 indicate which Specific Address Register causes the match.
26:25	Specific Address Register match. Encoded as follows: 00: Specific Address Register 1 match 01: Specific Address Register 2 match 10: Specific Address Register 3 match 11: Specific Address Register 4 match If more than one specific address is matched only one is indicated with priority 4 down to 1.
24	This bit has a different meaning depending on whether RX checksum offloading is enabled. **With RX checksum offloading disabled:** (bit 24 clear in Network Configuration Register) Type ID register match found, bit 22 and bit 23 indicate which type ID register causes the match. **With RX checksum offloading enabled:** (bit 24 set in Network Configuration Register) 0: The frame was not SNAP encoded and/or had a VLAN tag with the Canonical Format Indicator (CFI) bit set. 1: The frame was SNAP encoded and had either no VLAN tag or a VLAN tag with the CFI bit not set.
23:22	This bit has a different meaning depending on whether RX checksum offloading is enabled. **With RX checksum offloading disabled:** (bit 24 clear in Network Configuration) Type ID register match. Encoded as follows: 00: Type ID register 1 match 01: Type ID register 2 match 10: Type ID register 3 match 11: Type ID register 4 match If more than one Type ID is matched only one is indicated with priority 4 down to 1. **With RX checksum offloading enabled:** (bit 24 set in Network Configuration Register) 00: Neither the IP header checksum nor the TCP/UDP checksum was checked. 01: The IP header checksum was checked and was correct. Neither the TCP nor UDP checksum was checked. 10: Both the IP header and TCP checksum were checked and were correct. 11: Both the IP header and UDP checksum were checked and were correct.
21	VLAN tag detected—type ID of 0x8100. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100

Table 37-4. Receive Buffer Descriptor Entry (Continued)

Bit	Function
20	Priority tag detected—type ID of 0x8100 and null VLAN identifier. For packets incorporating the stacked VLAN processing feature, this bit will be set if the second VLAN tag has a type ID of 0x8100 and a null VLAN identifier.
19:17	VLAN priority—only valid if bit 21 is set.
16	Canonical format indicator (CFI) bit (only valid if bit 21 is set).
15	End of frame—when set the buffer contains the end of a frame. If end of frame is not set, then the only valid status bit is start of frame (bit 14).
14	Start of frame—when set the buffer contains the start of a frame. If both bits 15 and 14 are set, the buffer contains a whole frame.
13	This bit has a different meaning depending on whether jumbo frames and ignore FCS modes are enabled. If neither mode is enabled this bit will be zero. **With jumbo frame mode enabled:** (bit 3 set in Network Configuration Register) Additional bit for length of frame (bit[13]), that is concatenated with bits[12:0] **With ignore FCS mode enabled and jumbo frames disabled:** (bit 26 set in Network Configuration Register and bit 3 clear in Network Configuration Register) This indicates per frame FCS status as follows: 0: Frame had good FCS 1: Frame had bad FCS, but was copied to memory as ignore FCS enabled.
12:0	These bits represent the length of the received frame which may or may not include FCS depending on whether FCS discard mode is enabled. **With FCS discard mode disabled:** (bit 17 clear in Network Configuration Register) Least significant 12 bits for length of frame including FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above. **With FCS discard mode enabled:** (bit 17 set in Network Configuration Register) Least significant 12 bits for length of frame excluding FCS. If jumbo frames are enabled, these 12 bits are concatenated with bit[13] of the descriptor above.

To receive frames, the AHB buffer descriptors must be initialized by writing an appropriate address to bits 31:2 in the first word of each list entry. Bit 0 must be written with zero. Bit 1 is the wrap bit and indicates the last entry in the buffer descriptor list.

The start location of the receive buffer descriptor list must be written with the receive buffer queue base address before reception is enabled (receive enable in the Network Control register). Once reception is enabled, any writes to the Receive Buffer Queue Base Address register are ignored. When read, it will return the current pointer position in the descriptor list, though this is only valid and stable when receive is disabled.

If the filter block indicates that a frame should be copied to memory, the receive data DMA operation starts writing data into the receive buffer. If an error occurs, the buffer is recovered.

An internal counter within the GMAC represents the receive buffer queue pointer and it is not visible through the CPU interface. The receive buffer queue pointer increments by two words after each buffer has been used. It reinitializes to the receive buffer queue base address if any descriptor has its wrap bit set.

As receive AHB buffers are used, the receive AHB buffer manager sets bit zero of the first word of the descriptor to logic one indicating the AHB buffer has been used.

Software should search through the “used” bits in the AHB buffer descriptors to find out how many frames have been received, checking the start of frame and end of frame bits.

When the DMA is configured in the packet buffer Partial Store And Forward mode, received frames are written out to the AHB buffers as soon as enough frame data exists in the packet buffer. For both cases, this may mean

several full AHB buffers are used before some error conditions can be detected. If a receive error is detected the receive buffer currently being written will be recovered. Previous buffers will not be recovered. As an example, when receiving frames with cyclic redundancy check (CRC) errors or excessive length, it is possible that a frame fragment might be stored in a sequence of AHB receive buffers. Software can detect this by looking for start of frame bit set in a buffer following a buffer with no end of frame bit set.

To function properly, a 10/100 Ethernet system should have no excessive length frames or frames greater than 128 bytes with CRC errors. Collision fragments will be less than 128 bytes long, therefore it will be a rare occurrence to find a frame fragment in a receive AHB buffer, when using the default value of 128 bytes for the receive buffers size.

When in packet buffer full store and forward mode, only good received frames are written out of the DMA, so no fragments will exist in the AHB buffers due to MAC receiver errors. There is still the possibility of fragments due to DMA errors, for example used bit read on the second buffer of a multibuffer frame.

If bit zero of the receive buffer descriptor is already set when the receive buffer manager reads the location of the receive AHB buffer, then the buffer has been already used and cannot be used again until software has processed the frame and cleared bit zero. In this case, the “buffer not available” bit in the Receive Status register is set and an interrupt triggered. The Receive Resource Error statistics register is also incremented.

When the DMA is configured in the packet buffer full store and forward mode, the user can optionally select whether received frames should be automatically discarded when no AHB buffer resource is available. This feature is selected via bit 24 of the DMA Configuration register (by default, the received frames are not automatically discarded). If this feature is off, then received packets will remain to be stored in the SRAM-based packet buffer until AHB buffer resource next becomes available. This may lead to an eventual packet buffer overflow if packets continue to be received when bit zero (used bit) of the receive buffer descriptor remains set. Note that after a used bit has been read, the receive buffer manager will re-read the location of the receive buffer descriptor every time a new packet is received. When the DMA is not configured in the packet buffer full store and forward mode and a used bit is read, the frame currently being received will be automatically discarded.

When the DMA is configured in the packet buffer full store and forward mode, a receive overrun condition occurs when the receive SRAM-based packet buffer is full, or because HRESP was not OK. In all other modes, a receive overrun condition occurs when either the AHB bus was not granted quickly enough, or because HRESP was not OK, or because a new frame has been detected by the receive block, but the status update or write back for the previous frame has not yet finished. For a receive overrun condition, the receive overrun interrupt is asserted and the buffer currently being written is recovered. The next frame that is received whose address is recognized reuses the buffer.

In any packet buffer mode, a write to bit 18 of GMAC_NCR will force a packet from the external SRAM-based receive packet buffer to be flushed. This feature is only acted upon when the RX DMA is not currently writing packet data out to AHB, i.e., it is in an IDLE state. If the RX DMA is active, a write to this bit is ignored.

37.6.3.4 Transmit AHB Buffers

Frames to transmit are stored in one or more transmit AHB buffers. Transmit frames can be between 1 and 16384 bytes long, so it is possible to transmit frames longer than the maximum length specified in the IEEE 802.3 standard. It should be noted that zero length AHB buffers are allowed and that the maximum number of buffers permitted for each transmit frame is 128.

The start location for each transmit AHB buffer is stored in memory in a list of transmit buffer descriptors at a location pointed to by the transmit buffer queue pointer. The base address for this queue pointer is set in software using the Transmit Buffer Queue Base Address register. Each list entry consists of two words. The first is the byte address of the transmit buffer and the second containing the transmit control and status. For the packet buffer DMA, the start location for each AHB buffer is a byte address, the bottom bits of the address being used to offset the start of the data from the data-word boundary (i.e., bits 2,1 and 0 are used to offset the address for 64-bit datapaths).

Frames can be transmitted with or without automatic CRC generation. If CRC is automatically generated, pad will also be automatically generated to take frames to a minimum length of 64 bytes. When CRC is not automatically generated (as defined in word 1 of the transmit buffer descriptor), the frame is assumed to be at least 64 bytes long and pad is not generated.

An entry in the transmit buffer descriptor list is described in Table 37-5.

To transmit frames, the buffer descriptors must be initialized by writing an appropriate byte address to bits [31:0] in the first word of each descriptor list entry.

The second word of the transmit buffer descriptor is initialized with control information that indicates the length of the frame, whether or not the MAC is to append CRC and whether the buffer is the last buffer in the frame.

After transmission the status bits are written back to the second word of the first buffer along with the used bit. Bit 31 is the used bit which must be zero when the control word is read if transmission is to take place. It is written to one once the frame has been transmitted. Bits[29:20] indicate various transmit error conditions. Bit 30 is the wrap bit which can be set for any buffer within a frame. If no wrap bit is encountered the queue pointer continues to increment.

The Transmit Buffer Queue Base Address register can only be updated while transmission is disabled or halted; otherwise any attempted write will be ignored. When transmission is halted the transmit buffer queue pointer will maintain its value. Therefore when transmission is restarted the next descriptor read from the queue will be from immediately after the last successfully transmitted frame. while transmit is disabled (bit 3 of the Network Control register set low), the transmit buffer queue pointer resets to point to the address indicated by the Transmit Buffer Queue Base Address register. Note that disabling receive does not have the same effect on the receive buffer queue pointer.

Once the transmit queue is initialized, transmit is activated by writing to the transmit start bit (bit 9) of the Network Control register. Transmit is halted when a buffer descriptor with its used bit set is read, a transmit error occurs, or by writing to the transmit halt bit of the Network Control register. Transmission is suspended if a pause frame is received while the pause enable bit is set in the Network Configuration register. Rewriting the start bit while transmission is active is allowed. This is implemented with TXGO variable which is readable in the Transmit Status register at bit location 3. The TXGO variable is reset when:

- Transmit is disabled.
- A buffer descriptor with its ownership bit set is read.
- Bit 10, THALT, of the Network Control register is written.
- There is a transmit error such as too many retries or a transmit underrun.

To set TXGO, write TSTART to the bit 9 of the Network Control register. Transmit halt does not take effect until any ongoing transmit finishes.

If the DMA is configured for packet buffer Partial Store and Forward mode and a collision occurs during transmission of a multibuffer frame, transmission will automatically restart from the first buffer of the frame. For packet buffer mode, the entire contents of the frame are read into the transmit packet buffer memory, so the retry attempt will be replayed directly from the packet buffer memory rather than having to re-fetch through the AHB.

If a used bit is read midway through transmission of a multibuffer frame, this is treated as a transmit error. Transmission stops, GTXER is asserted and the FCS will be bad.

If transmission stops due to a transmit error or a used bit being read, transmission restarts from the first buffer descriptor of the frame being transmitted when the transmit start bit is rewritten.

Table 37-5. Transmit Buffer Descriptor Entry

Bit	Function
Word 0	
31:0	Byte address of buffer
Word 1	
31	Used—must be zero for the GMAC to read data to the transmit buffer. The GMAC sets this to one for the first buffer of a frame once it has been successfully transmitted. Software must clear this bit before the buffer can be used again.
30	Wrap—marks last descriptor in transmit buffer descriptor list. This can be set for any buffer within the frame.
29	Retry limit exceeded, transmit error detected
28	Reserved.
27	Transmit frame corruption due to AHB error—set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted). Also set if single frame is too large for configured packet buffer memory size.
26	Late collision, transmit error detected.
25:23	Reserved
22:20	Transmit IP/TCP/UDP checksum generation offload errors: 000: No Error. 001: The Packet was identified as a VLAN type, but the header was not fully complete, or had an error in it. 010: The Packet was identified as a SNAP type, but the header was not fully complete, or had an error in it. 011: The Packet was not of an IP type, or the IP packet was invalidly short, or the IP was not of type IPv4/IPv6. 100: The Packet was not identified as VLAN, SNAP or IP. 101: Nonsupported packet fragmentation occurred. For IPv4 packets, the IP checksum was generated and inserted. 110: Packet type detected was not TCP or UDP. TCP/UDP checksum was therefore not generated. For IPv4 packets, the IP checksum was generated and inserted. 111: A premature end of packet was detected and the TCP/UDP checksum could not be generated.
19:17	Reserved
16	No CRC to be appended by MAC. When set, this implies that the data in the buffers already contains a valid CRC, hence no CRC or padding is to be appended to the current frame by the MAC. This control bit must be set for the first buffer in a frame and will be ignored for the subsequent buffers of a frame. Note that this bit must be clear when using the transmit IP/TCP/UDP checksum generation offload, otherwise checksum generation and substitution will not occur.
15	Last buffer, when set this bit will indicate the last buffer in the current frame has been reached.
14	Reserved
13:0	Length of buffer

37.6.3.5 DMA Bursting on the AHB

The DMA will always use SINGLE, or INCR type AHB accesses for buffer management operations. When performing data transfers, the AHB burst length used can be programmed using bits 4:0 of the DMA Configuration register so that either SINGLE, INCR or fixed length incrementing bursts (INCR4, INCR8 or INCR16) are used where possible.

When there is enough space and enough data to be transferred, the programmed fixed length bursts will be used. If there is not enough data or space available, for example when at the beginning or the end of a buffer, SINGLE type accesses are used. Also SINGLE type accesses are used at 1024 byte boundaries, so that the 1 Kbyte boundaries are not burst over as per AHB requirements.

The DMA will not terminate a fixed length burst early, unless an error condition occurs on the AHB or if receive or transmit are disabled in the Network Control register.

37.6.3.6 DMA Packet Buffer

The DMA uses packet buffers for both transmit and receive paths. This mode allows multiple packets to be buffered in both transmit and receive directions. This allows the DMA to withstand far greater access latencies on the AHB and make more efficient use of the AHB bandwidth. There are two modes of operation—Full Store and Forward and Partial Store and Forward.

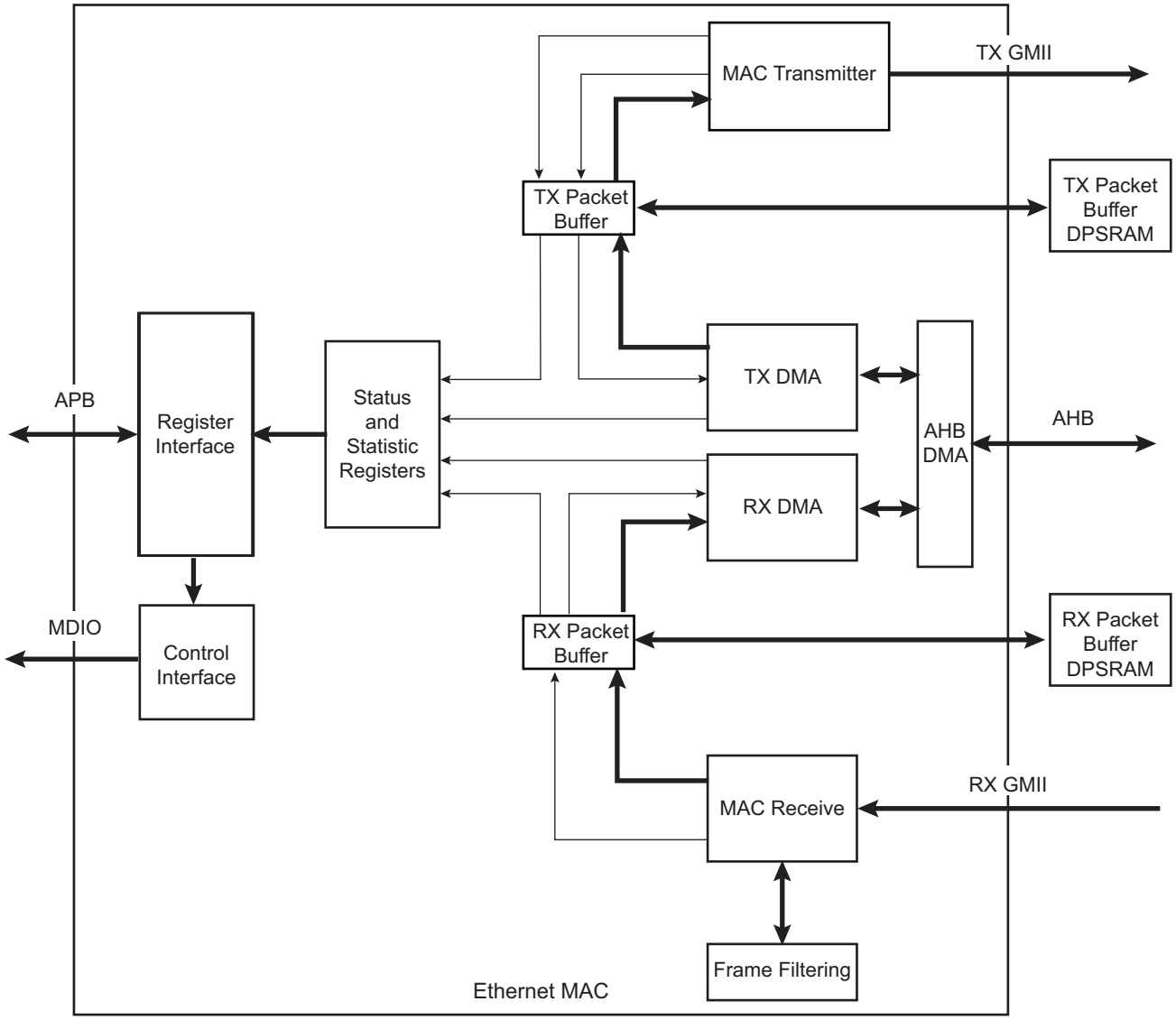
As described above (Section 37.6.3.2 ”Partial Store and Forward Using Packet Buffer DMA”), the DMA can be programmed into a low latency mode, known as Partial Store and Forward. For further details of this mode, see Section 37.6.3.2.

When the DMA is in full store and forward mode, full packets are buffered which provides the possibility to:

- Discard packets with error on the receive path before they are partially written out of the DMA, thus saving AHB bus bandwidth and driver processing overhead,
- Retry collided transmit frames from the buffer, thus saving AHB bus bandwidth,
- Implement transmit IP/TCP/UDP checksum generation offload.

With the packet buffers included, the structure of the GMAC data paths is shown in Figure 37-2.

Figure 37-2. Data Paths with Packet Buffers Included



37.6.3.7 Transmit Packet Buffer

The transmitter packet buffer will continue attempting to fetch frame data from the AHB system memory until the packet buffer itself is full, at which point it will attempt to maintain its full level.

To accommodate the status and statistics associated with each frame, three words per packet (or two if the GMAC is configured in 64-bit datapath mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet. Storing the status in the DPRAM is required in order to decouple the DMA interface of the buffer from the MAC interface, to update the MAC status/statistics and to generate interrupts in the order in which the packets that they represent were fetched from the AHB memory.

If any errors occur on the AHB while reading the transmit frame, the fetching of packet data from AHB memory is halted. The MAC transmitter will continue to fetch packet data, thereby emptying the packet buffer and allowing any good non-errored frames to be transmitted successfully. Once these have been fully transmitted, the status/statistics for the errored frame will be updated and software will be informed via an interrupt that an AHB error occurred. This way, the error is reported in the correct packet order.

The transmit packet buffer will only attempt to read more frame data from the AHB when space is available in the packet buffer memory. If space is not available it must wait until the a packet fetched by the MAC completes transmission and is subsequently removed from the packet buffer memory. Note that if full store and forward mode is active and if a single frame is fetched that is too large for the packet buffer memory, the frame is flushed and the DMA halted with an error status. This is because a complete frame must be written into the packet buffer before transmission can begin, and therefore the minimum packet buffer memory size should be chosen to satisfy the maximum frame to be transmitted in the application.

In full store and forward mode, once the complete transmit frame is written into the packet buffer memory, a trigger is sent across to the MAC transmitter, which will then begin reading the frame from the packet buffer memory. Since the whole frame is present and stable in the packet buffer memory an underflow of the transmitter is not possible. The frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In Partial Store and Forward mode, a trigger is sent across to the MAC transmitter as soon as sufficient packet data is available, which will then begin fetching the frame from the packet buffer memory. If, after this point, the MAC transmitter is able to fetch data from the packet buffer faster than the AHB DMA can fill it, an underflow of the transmitter is possible. In this case, the transmission is terminated early, and the packet buffer is completely flushed. Transmission can only be restarted by writing to the transmit START bit.

In half duplex mode, the frame is kept in the packet buffer until notification is received from the MAC that the frame data has either been successfully transmitted or can no longer be retransmitted (too many retries in half duplex mode). When this notification is received the frame is flushed from memory to make room for a new frame to be fetched from AHB system memory.

In full duplex mode, the frame is removed from the packet buffer on the fly.

Other than underflow, the only MAC related errors that can occur are due to collisions during half duplex transmissions. When a collision occurs the frame still exists in the packet buffer memory so can be retried directly from there. Only once the MAC transmitter has failed to transmit after sixteen attempts is the frame finally flushed from the packet buffer.

37.6.3.8 Receive Packet Buffer

The receive packet buffer stores frames from the MAC receiver along with their status and statistics. Frames with errors are flushed from the packet buffer memory, while good frames are pushed onto the DMA AHB interface.

The receiver packet buffer monitors the FIFO write interface from the MAC receiver and translates the FIFO pushes into packet buffer writes. At the end of the received frame the status and statistics are buffered so that the information can be used when the frame is read out. When programmed in full store and forward mode, if the frame has an error the frame data is immediately flushed from the packet buffer memory allowing subsequent frames to utilise the freed up space. The status and statistics for bad frames are still used to update the GMAC registers.

To accommodate the status and statistics associated with each frame, three words per packet (or two if configured in 64-bit datapath mode) are reserved at the end of the packet data. If the packet is bad and requires to be dropped, the status and statistics are the only information held on that packet.

The receiver packet buffer will also detect a full condition so that an overflow condition can be detected. If this occurs, subsequent packets are dropped and an RX overflow interrupt is raised.

For full store and forward, the DMA only begins packet fetches once the status and statistics for a frame are available. If the frame has a bad status due to a frame error, the status and statistics are passed on to the GMAC registers. If the frame has a good status, the information is used to read the frame from the packet buffer memory and burst onto the AHB using the DMA buffer management protocol. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GMAC registers.

If Partial Store and Forward mode is active, the DMA will begin fetching the packet data before the status is available. As soon as the status becomes available, the DMA will fetch this information as soon as possible before continuing to fetch the remainder of the frame. Once the last frame data has been transferred to the packet buffer, the status and statistics are updated to the GMAC registers.

37.6.3.9 Priority Queueing in the DMA

The DMA by default uses a single transmit and receive queue. This means the list of transmit/receive buffer descriptors point to data buffers associated with a single transmit/receive data stream. The GMAC can select up to 3 priority queues. Each queue has an independent list of buffer descriptors pointing to separate data streams.

In the transmit direction, higher priority queues are always serviced before lower priority queues, with Q0 as lowest priority and Q2 as highest priority. This strict priority scheme requires the user to ensure that high priority traffic is constrained so that lower priority traffic will have required bandwidth. The GMAC DMA will determine the next queue to service by initiating a sequence of buffer descriptor reads interrogating the ownership bits of each. The buffer descriptor corresponding to the highest priority queue is read first. As an example, if the ownership bit of this descriptor is set, then the DMA will progress to reading the 2nd highest priority queue's descriptor. If that ownership bit read of this lower priority queue is set, then the DMA will read the 3rd highest priority queue's descriptor. If all the descriptors return an ownership bit set, then a resource error has occurred, an interrupt is generated and transmission is automatically halted. Transmission can only be restarted by setting the START bit in the Network Control register. The GMAC DMA will need to identify the highest available queue to transmit from when the START bit in the Network Control register is written to and the TX is in a halted state, or when the last word of any packet has been fetched from external AHB memory.

The GMAC transmit DMA maximizes the effectiveness of priority queuing by ensuring that high priority traffic be transmitted as early as possible after being fetched from AHB. High priority traffic fetched from AHB will be pushed to the MAC layer, depending on traffic shaping being enabled and the associated credit value for that queue, before any lower priority traffic that may pre-exist in the transmit SRAM-based packet buffer. This is achieved by separating the transmit SRAM-based packet buffer into regions, one region per queue. The size of each region determines the amount of SRAM space allocated per queue.

For each queue, there is an associated Transmit Buffer Queue Base Address register. For the lowest priority queue (or the only queue when only one queue is selected), the Transmit Buffer Queue Base Address is located at address 0x1C. For all other queues, the Transmit Buffer Queue Base Address registers are located at sequential addresses starting at address 0x440.

In the receive direction each packet is written to AHB data buffers in the order that it is received. For each queue, there is an independent set of receive AHB buffers for each queue. There is therefore a separate Receive Buffer Queue Base Address register for each queue. For the lowest priority queue (or the only queue when only one queue is selected), the Receive Buffer Queue Base Address is located at address 0x18. For all other queues, the Receive Buffer Queue Base Address registers are located at sequential addresses starting at address 0x480. Every received packet will pass through a programmable screening algorithm which will allocate a particular queue to that frame. The user interface to the screeners is through two types of programmable registers:

- Screening Type 1 registers—The module features 4 Screening Type 1 registers. Screening Type 1 registers hold values to match against specific IP and UDP fields of the received frames. The fields matched against are DS (Differentiated Services field of IPv4 frames), TC (Traffic class field of IPv6 frames) and/or the UDP destination port.
- Screening Type 2 registers—The module features 8 Screening Type 2 registers GMAC_ST2RPQ. Screening Type 2 registers operate independently of Screening Type 1 registers and offer additional match capabilities. Screening Type 2 allows a screen to be configured that is the combination of all or any of the following comparisons:

1. An enable bit VLAN priority, VLANE. A VLAN priority match will be performed if the VLAN priority enable is set. The extracted priority field in the VLAN header is compared against VLANP in the GMAC_ST2RPQ register itself.

2. An enable bit EtherType, ETHE. The EtherType field I2ETH inside GMAC_ST2RPQ maps to one of 4 EtherType match registers, GMAC_ST2ER. The extracted EtherType is compared against GMAC_ST2ER designated by this EtherType field.
3. An enable bit Compare A, COMPAE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
4. An enable bit Compare B, COMPBE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.
5. An enable bit Compare C, COMPCE. This bit is associated with a Screening Type 2 Compare Word 0/1 register x, GMAC_ST2CW0/1.

Each screener type has an enable bit, a match pattern and a queue number. If a received frame matches on an enabled Screening register, then the frame will be tagged with the queue value in the associated Screening register, and forwarded onto the DMA and subsequently into the external memory associated with that queue. If two screeners are matched, then the one which resides at the lowest register address will take priority so care must be taken on the selection of the screener location.

When the priority queuing feature is enabled, the number of interrupt outputs from the GMAC core is increased to match the number of supported queues. The number of Interrupt Status registers is increased by the same number. Only DMA related events are reported using the individual interrupt outputs, as the GMAC can relate these events to specific queues. All other events generated within the GMAC are reported in the interrupt associated with the lowest priority queue. For the lowest priority queue (or the only queue when only 1 queue is selected), the Interrupt Status register is located at address 0x24. For all other queues, the Interrupt Status register is located at sequential addresses starting at address 0x400.

Note: The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See Section 37.6.7 "MAC Filtering Block" for more details.

The additional screening done by the functions Compare A, B, and C each have an enable bit and compare register field. COMPA, COMPB and COMPC in GMAC_ST2RPQ are pointers to a configured offset (OFFSVAL), value (COMPVAL), and mask (MASKVAL). If enabled, the compare is true if the data at the offset into the frame, ANDed with MASKVAL, is equal to the value of COMPVAL ANDed with MASKVAL. A 16-bit word comparison is done. The byte at the offset number of bytes from the index start is compared to bits 7:0 of the configured COMPVAL and MASKVAL. The byte at the offset number of bytes + 1 from the index start is compared to bits 15:8 of the configured COMPVAL and MASKVAL.

The offset value in bytes, OFFSVAL, ranges from 0 to 127 bytes from either the start of the frame, the byte after the EtherType field, the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header. Note the logic to decode the IP header or the TCP/UDP header is reused from the TCP/UDP/IP checksum offload logic and therefore has the same restrictions on use (the main limitation is that IP fragmentation is not supported). Refer to the Checksum Offload for IP, TCP and UDP section of this documentation for further details.

Compare A, B, and C use a common set of 24 GMAC_ST2CW0/1 registers, thus all COMPA, COMPB and COMPC fields in the registers GMAC_ST2RPQ point to a single pool of 24 GMAC_ST2CW0/1 registers.

Note that Compare A, B and C together allow matching against an arbitrary 48 bits of data and so can be used to match against a MAC address.

All enabled comparisons are ANDed together to form the overall type 2 screening match.

37.6.4 MAC Transmit Block

The MAC transmitter can operate in either half duplex or full duplex mode and transmits frames in accordance with the Ethernet IEEE 802.3 standard. In half duplex mode, the CSMA/CD protocol of the IEEE 802.3 specification is followed.

A small input buffer receives data through the FIFO interface which will extract data in 32-bit form. All subsequent processing prior to the final output is performed in bytes.

Transmit data can be output using the MII interface.

Frame assembly starts by adding preamble and the start frame delimiter. Data is taken from the transmit FIFO interface a word at a time.

If necessary, padding is added to take the frame length to 60 bytes. CRC is calculated using an order 32-bit polynomial. This is inverted and appended to the end of the frame taking the frame length to a minimum of 64 bytes. If the no CRC bit is set in the second word of the last buffer descriptor of a transmit frame, neither pad nor CRC are appended. The no CRC bit can also be set through the FIFO interface.

In full duplex mode (at all data rates), frames are transmitted immediately. Back to back frames are transmitted at least 96 bit times apart to guarantee the interframe gap.

In half duplex mode, the transmitter checks carrier sense. If asserted, the transmitter waits for the signal to become inactive, and then starts transmission after the interframe gap of 96 bit times. If the collision signal is asserted during transmission, the transmitter will transmit a jam sequence of 32 bits taken from the data register and then retry transmission after the back off time has elapsed. If the collision occurs during either the preamble or Start Frame Delimiter (SFD), then these fields will be completed prior to generation of the jam sequence.

The back off time is based on an XOR of the 10 least significant bits of the data coming from the transmit FIFO interface and a 10-bit pseudo random number generator. The number of bits used depends on the number of collisions seen. After the first collision 1 bit is used, then the second 2 bits and so on up to the maximum of 10 bits. All 10 bits are used above ten collisions. An error will be indicated and no further attempts will be made if 16 consecutive attempts cause collision. This operation is compliant with the description in Clause 4.2.3.2.5 of the IEEE 802.3 standard which refers to the truncated binary exponential back off algorithm.

In 10/100 mode, both collisions and late collisions are treated identically, and back off and retry will be performed up to 16 times. This condition is reported in the transmit buffer descriptor word 1 (late collision, bit 26) and also in the Transmit Status register (late collision, bit 7). An interrupt can also be generated (if enabled) when this exception occurs, and bit 5 in the Interrupt Status register will be set.

In all modes of operation, if the transmit DMA underruns, a bad CRC is automatically appended using the same mechanism as jam insertion and the GTXER signal is asserted. For a properly configured system this should never happen and also it is impossible if configured to use the DMA with packet buffers, as the complete frame is buffered in local packet buffer memory.

By setting when bit 28 is set in the Network Configuration register, the Inter Packet Gap (IPG) may be stretched beyond 96 bits depending on the length of the previously transmitted frame and the value written to the IPG Stretch register (GMAC_IPGS). The least significant 8 bits of the IPG Stretch register multiply the previous frame length (including preamble). The next significant 8 bits (+1 so as not to get a divide by zero) divide the frame length to generate the IPG. IPG stretch only works in full duplex mode and when bit 28 is set in the Network Configuration register. The IPG Stretch register cannot be used to shrink the IPG below 96 bits.

If the back pressure bit is set in the Network Control register, or if the HDFC configuration bit is set in the GMAC_UR register (10M or 100M half duplex mode), the transmit block transmits 64 bits of data, which can consist of 16 nibbles of 1011 or in bit rate mode 64 1s, whenever it sees an incoming frame to force a collision. This provides a way of implementing flow control in half duplex mode.

37.6.5 MAC Receive Block

All processing within the MAC receive block is implemented using a 16-bit data path. The MAC receive block checks for valid preamble, FCS, alignment and length, presents received frames to the FIFO interface and stores the frame destination address for use by the address checking block.

If, during the frame reception, the frame is found to be too long, a bad frame indication is sent to the FIFO interface. The receiver logic ceases to send data to memory as soon as this condition occurs.

At end of frame reception the receive block indicates to the DMA block whether the frame is good or bad. The DMA block will recover the current receive buffer if the frame was bad.

Ethernet frames are normally stored in DMA memory complete with the FCS. Setting the FCS remove bit in the network configuration (bit 17) causes frames to be stored without their corresponding FCS. The reported frame length field is reduced by four bytes to reflect this operation.

The receive block signals to the register block to increment the alignment, CRC (FCS), short frame, long frame, jabber or receive symbol errors when any of these exception conditions occur.

If bit 26 is set in the network configuration, CRC errors will be ignored and CRC errored frames will not be discarded, though the Frame Check Sequence Errors statistic register will still be incremented. Additionally, if not enabled for jumbo frames mode, then bit[13] of the receiver descriptor word 1 will be updated to indicate the FCS validity for the particular frame. This is useful for applications such as EtherCAT whereby individual frames with FCS errors must be identified.

Received frames can be checked for length field error by setting the length field error frame discard bit of the Network Configuration register (bit-16). When this bit is set, the receiver compares a frame's measured length with the length field (bytes 13 and 14) extracted from the frame. The frame is discarded if the measured length is shorter. This checking procedure is for received frames between 64 bytes and 1518 bytes in length.

Each discarded frame is counted in the 10-bit Length Field Frame Error statistics register. Frames where the length field is greater than or equal to 0x0600 hex will not be checked.

37.6.6 Checksum Offload for IP, TCP and UDP

The GMAC can be programmed to perform IP, TCP and UDP checksum offloading in both receive and transmit directions, which is enabled by setting bit 24 in the Network Configuration register for receive and bit 11 in the DMA Configuration register for transmit.

IPv4 packets contain a 16-bit checksum field, which is the 16-bit 1’s complement of the 1’s complement sum of all 16-bit words in the header. TCP and UDP packets contain a 16-bit checksum field, which is the 16-bit 1’s complement of the 1’s complement sum of all 16-bit words in the header, the data and a conceptual IP pseudo header.

To calculate these checksums in software requires each byte of the packet to be processed. For TCP and UDP this can use a large amount of processing power. Offloading the checksum calculation to hardware can result in significant performance improvements.

For IP, TCP or UDP checksum offload to be useful, the operating system containing the protocol stack must be aware that this offload is available so that it can make use of the fact that the hardware can either generate or verify the checksum.

37.6.6.1 Receiver Checksum Offload

When receive checksum offloading is enabled in the GMAC, the IPv4 header checksum is checked as per RFC 791, where the packet meets the following criteria:

- If present, the VLAN header must be four octets long and the CFI bit must not be set.
- Encapsulation must be RFC 894 Ethernet Type Encoding or RFC 1042 SNAP Encoding.
- IPv4 packet
- IP header is of a valid length

The GMAC also checks the TCP checksum as per RFC 793, or the UDP checksum as per RFC 768, if the following criteria are met:

- IPv4 or IPv6 packet
- Good IP header checksum (if IPv4)
- No IP fragmentation
- TCP or UDP packet

When an IP, TCP or UDP frame is received, the receive buffer descriptor gives an indication if the GMAC was able to verify the checksums. There is also an indication if the frame had SNAP encapsulation. These indication bits will

replace the type ID match indication bits when the receive checksum offload is enabled. For details of these indication bits refer to Table 37-4 “Receive Buffer Descriptor Entry”.

If any of the checksums are verified as incorrect by the GMAC, the packet is discarded and the appropriate statistics counter incremented.

37.6.6.2 Transmitter Checksum Offload

The transmitter checksum offload is only available if the full store and forward mode is enabled. This is because the complete frame to be transmitted must be read into the packet buffer memory before the checksum can be calculated and written back into the headers at the beginning of the frame.

Transmitter checksum offload is enabled by setting bit [11] in the DMA Configuration register. When enabled, it will monitor the frame as it is written into the transmitter packet buffer memory to automatically detect the protocol of the frame. Protocol support is identical to the receiver checksum offload.

For transmit checksum generation and substitution to occur, the protocol of the frame must be recognized and the frame must be provided without the FCS field, by making sure that bit [16] of the transmit descriptor word 1 is clear. If the frame data already had the FCS field, this would be corrupted by the substitution of the new checksum fields.

If these conditions are met, the transmit checksum offload engine will calculate the IP, TCP and UDP checksums as appropriate. Once the full packet is completely written into packet buffer memory, the checksums will be valid and the relevant DPRAM locations will be updated for the new checksum fields as per standard IP/TCP and UDP packet structures.

If the transmitter checksum engine is prevented from generating the relevant checksums, bits [22:20] of the transmitter DMA writeback status will be updated to identify the reason for the error. Note that the frame will still be transmitted but without the checksum substitution, as typically the reason that the substitution did not occur was that the protocol was not recognized.

37.6.7 MAC Filtering Block

The filter block determines which frames should be written to the FIFO interface and on to the DMA.

Whether a frame is passed depends on what is enabled in the Network Configuration register, the state of the external matching pins, the contents of the specific address, type and Hash registers and the frame's destination address and type field.

If bit 25 of the Network Configuration register is not set, a frame will not be copied to memory if the GMAC is transmitting in half duplex mode at the time a destination address is received.

Ethernet frames are transmitted a byte at a time, least significant bit first. The first six bytes (48 bits) of an Ethernet frame make up the destination address. The first bit of the destination address, which is the LSB of the first byte of the frame, is the group or individual bit. This is one for multicast addresses and zero for unicast. The all ones address is the broadcast address and a special case of multicast.

The GMAC supports recognition of four specific addresses. Each specific address requires two registers, Specific Address Bottom register and Specific Address Top register. Specific Address Bottom register stores the first four bytes of the destination address and Specific Address Top register contains the last two bytes. The addresses stored can be specific, group, local or universal.

The destination address of received frames is compared against the data stored in the Specific Address registers once they have been activated. The addresses are deactivated at reset or when their corresponding Specific Address Bottom register is written. They are activated when Specific Address Top register is written. If a receive frame address matches an active address, the frame is written to the FIFO interface and on to DMA memory.

Frames may be filtered using the type ID field for matching. Four type ID registers exist in the register address space and each can be enabled for matching by writing a one to the MSB (bit 31) of the respective register. When a frame is received, the matching is implemented as an OR function of the various types of match.

The contents of each type ID register (when enabled) are compared against the length/type ID of the frame being received (e.g., bytes 13 and 14 in non-VLAN and non-SNAP encapsulated frames) and copied to memory if a match is found. The encoded type ID match bits (Word 0, Bit 22 and Bit 23) in the receive buffer descriptor status are set indicating which type ID register generated the match, if the receive checksum offload is disabled.

The reset state of the type ID registers is zero, hence each is initially disabled.

The following example illustrates the use of the address and type ID match registers for a MAC address of 21:43:65:87:A9:CB:

Preamble	55
SFD	D5
DA (Octet 0 - LSB)	21
DA (Octet 1)	43
DA (Octet 2)	65
DA (Octet 3)	87
DA (Octet 4)	A9
DA (Octet 5 - MSB)	CB
SA (LSB)	00(1)
SA	00(1)
SA	00(1)
SA	00(1)
SA	00(1)
SA (MSB)	00(1)
Type ID (MSB)	43
Type ID (LSB)	21

Note: 1. Contains the address of the transmitting device

The sequence above shows the beginning of an Ethernet frame. Byte order of transmission is from top to bottom as shown. For a successful match to specific address 1, the following address matching registers must be set up:

Specific Address 1 Bottom register (GMAC_SAB1) (Address 0x088) 0x87654321

Specific Address 1 Top register (GMAC_SAT1) (Address 0x08C) 0x0000CBA9

For a successful match to the type ID, the following Type ID Match 1 register must be set up:

Type ID Match 1 register (GMAC_TIDM1) (Address 0x0A8) 0x80004321

37.6.8 Broadcast Address

Frames with the broadcast address of 0xFFFFFFFFFFFF are stored to memory only if the 'no broadcast' bit in the Network Configuration register is set to zero.

37.6.9 Hash Addressing

The hash address register is 64 bits long and takes up two locations in the memory map. The least significant bits are stored in Hash Register Bottom and the most significant bits in Hash Register Top.

The unicast hash enable and the multicast hash enable bits in the Network Configuration register enable the reception of hash matched frames. The destination address is reduced to a 6-bit index into the 64-bit Hash register using the following hash function: The hash function is an XOR of every sixth bit of the destination address.

```
hash_index[05] = da[05] ^ da[11] ^ da[17] ^ da[23] ^ da[29] ^ da[35] ^ da[41] ^
da[47]
hash_index[04] = da[04] ^ da[10] ^ da[16] ^ da[22] ^ da[28] ^ da[34] ^ da[40] ^
da[46]
hash_index[03] = da[03] ^ da[09] ^ da[15] ^ da[21] ^ da[27] ^ da[33] ^ da[39] ^
da[45]
hash_index[02] = da[02] ^ da[08] ^ da[14] ^ da[20] ^ da[26] ^ da[32] ^ da[38] ^
da[44]
hash_index[01] = da[01] ^ da[07] ^ da[13] ^ da[19] ^ da[25] ^ da[31] ^ da[37] ^
da[43]
hash_index[00] = da[00] ^ da[06] ^ da[12] ^ da[18] ^ da[24] ^ da[30] ^ da[36] ^
da[42]
```

`da[0]`

represents the least significant bit of the first byte received, that is, the multicast/unicast indicator, and `da[47]` represents the most significant bit of the last byte received.

If the hash index points to a bit that is set in the Hash register then the frame will be matched according to whether the frame is multicast or unicast.

A multicast match will be signalled if the multicast hash enable bit is set, `da[0]` is logic 1 and the hash index points to a bit set in the Hash register.

A unicast match will be signalled if the unicast hash enable bit is set, `da[0]` is logic 0 and the hash index points to a bit set in the Hash register.

To receive all multicast frames, the Hash register should be set with all ones and the multicast hash enable bit should be set in the Network Configuration register.

37.6.10 Copy all Frames (Promiscuous Mode)

If the Copy All Frames bit is set in the Network Configuration register then all frames (except those that are too long, too short, have FCS errors or have GRXER asserted during reception) will be copied to memory. Frames with FCS errors will be copied if bit 26 is set in the Network Configuration register.

37.6.11 Disable Copy of Pause Frames

Pause frames can be prevented from being written to memory by setting the disable copying of pause frames control bit 23 in the Network Configuration register. When set, pause frames are not copied to memory regardless of the Copy All Frames bit, whether a hash match is found, a type ID match is identified or if a destination address match is found.

37.6.12 VLAN Support

The following table describes an Ethernet encoded 802.1Q VLAN tag.

Table 37-6. 802.1Q VLAN Tag

TPID (Tag Protocol Identifier) 16 bits	TCI (Tag Control Information) 16 bits
0x8100	First 3 bits priority, then CFI bit, last 12 bits VID

The VLAN tag is inserted at the 13th byte of the frame adding an extra four bytes to the frame. To support these extra four bytes, the GMAC can accept frame lengths up to 1536 bytes by setting bit 8 in the Network Configuration register.

If the VID (VLAN identifier) is null (0x000) this indicates a priority-tagged frame.

The following bits in the receive buffer descriptor status word give information about VLAN tagged frames:-

- Bit 21 set if receive frame is VLAN tagged (i.e., type ID of 0x8100).

- Bit 20 set if receive frame is priority tagged (i.e., type ID of 0x8100 and null VID). (If bit 20 is set, bit 21 will be set also.)
- Bit 19, 18 and 17 set to priority if bit 21 is set.
- Bit 16 set to CFI if bit 21 is set.

The GMAC can be configured to reject all frames except VLAN tagged frames by setting the discard non-VLAN frames bit in the Network Configuration register.

37.6.13 Wake on LAN Support

The receive block supports Wake on LAN by detecting the following events on incoming receive frames:

- Magic packet
- Address Resolution Protocol (ARP) request to the device IP address
- Specific address 1 filter match
- Multicast hash filter match

These events can be individually enabled through bits [19:16] of the Wake on LAN register. Also, for Wake on LAN detection to occur, receive enable must be set in the Network Control register, however a receive buffer does not have to be available.

In case of an ARP request, specific address 1 or multicast filter events will occur even if the frame is errored. For magic packet events, the frame must be correctly formed and error free.

A magic packet event is detected if all of the following are true:

- Magic packet events are enabled through bit 16 of the Wake on LAN register
- The frame's destination address matches specific address 1
- The frame is correctly formed with no errors
- The frame contains at least 6 bytes of 0xFF for synchronization
- There are 16 repetitions of the contents of Specific Address 1 register immediately following the synchronization

An ARP request event is detected if all of the following are true:

- ARP request events are enabled through bit 17 of the Wake on LAN register
- Broadcasts are allowed by bit 5 in the Network Configuration register
- The frame has a broadcast destination address (bytes 1 to 6)
- The frame has a type ID field of 0x0806 (bytes 13 and 14)
- The frame has an ARP operation field of 0x0001 (bytes 21 and 22)
- The least significant 16 bits of the frame's ARP target protocol address (bytes 41 and 42) match the value programmed in bits[15:0] of the Wake on LAN register

The decoding of the ARP fields adjusts automatically if a VLAN tag is detected within the frame. The reserved value of 0x0000 for the Wake on LAN target address value will not cause an ARP request event, even if matched by the frame.

A specific address 1 filter match event will occur if all of the following are true:

- Specific address 1 events are enabled through bit 18 of the Wake on LAN register
- The frame's destination address matches the value programmed in the Specific Address 1 registers

A multicast filter match event will occur if all of the following are true:

- Multicast hash events are enabled through bit 19 of the Wake on LAN register
- Multicast hash filtering is enabled through bit 6 of the Network Configuration register
- The frame destination address matches against the multicast hash filter
- The frame destination address is not a broadcast

37.6.14 IEEE 1588 Support

IEEE 1588 is a standard for precision time synchronization in local area networks. It works with the exchange of special Precision Time Protocol (PTP) frames. The PTP messages can be transported over IEEE 802.3/Ethernet, over Internet Protocol Version 4 or over Internet Protocol Version 6 as described in the annex of IEEE P1588.D2.1.

The GMAC indicates the message time-stamp point (asserted on the start packet delimiter and deasserted at end of frame) for all frames and the passage of PTP event frames (asserted when a PTP event frame is detected and deasserted at end of frame).

IEEE 802.1AS is a subset of IEEE 1588. One difference is that IEEE 802.1AS uses the Ethernet multicast address 0180C200000E for sync frame recognition whereas IEEE 1588 does not. GMAC is designed to recognize sync frames with both IEEE 802.1AS and IEEE 1588 addresses and so can support both 1588 and 802.1AS frame recognition simultaneously.

Synchronization between master and slave clocks is a two stage process.

First, the offset between the master and slave clocks is corrected by the master sending a sync frame to the slave with a follow up frame containing the exact time the sync frame was sent. Hardware assist modules at the master and slave side detect exactly when the sync frame was sent by the master and received by the slave. The slave then corrects its clock to match the master clock.

Second, the transmission delay between the master and slave is corrected. The slave sends a delay request frame to the master which sends a delay response frame in reply. Hardware assist modules at the master and slave side detect exactly when the delay request frame was sent by the slave and received by the master. The slave will now have enough information to adjust its clock to account for delay. For example, if the slave was assuming zero delay, the actual delay will be half the difference between the transmit and receive time of the delay request frame (assuming equal transmit and receive times) because the slave clock will be lagging the master clock by the delay time already.

The time-stamp is taken when the message time-stamp point passes the clock time-stamp point. This can generate an interrupt if enabled (GMAC_IER). However, MAC Filtering configuration is needed to actually ‘copy’ the message to memory. For Ethernet, the message time-stamp point is the SFD and the clock time-stamp point is the MII interface. (The IEEE 1588 specification refers to sync and delay_req messages as event messages as these require time-stamping. These events are captured in the registers GMAC_EFTx and GMAC_EFRx, respectively. Follow up, delay response and management messages do not require time-stamping and are referred to as general messages.)

1588 version 2 defines two additional PTP event messages. These are the peer delay request (Pdelay_Req) and peer delay response (Pdelay_Resp) messages. These events are captured in the registers GMAC_PEFTx and GMAC_PEFRx, respectively. These messages are used to calculate the delay on a link. Nodes at both ends of a link send both types of frames (regardless of whether they contain a master or slave clock). The Pdelay_Resp message contains the time at which a Pdelay_Req was received and is itself an event message. The time at which a Pdelay_Resp message is received is returned in a Pdelay_Resp_Follow_Up message.

1588 version 2 introduces transparent clocks of which there are two kinds, peer-to-peer (P2P) and end-to-end (E2E). Transparent clocks measure the transit time of event messages through a bridge and amend a correction field within the message to allow for the transit time. P2P transparent clocks additionally correct for the delay in the receive path of the link using the information gathered from the peer delay frames. With P2P transparent clocks delay_req messages are not used to measure link delay. This simplifies the protocol and makes larger systems more stable.

The GMAC recognizes four different encapsulations for PTP event messages:

1. 1588 version 1 (UDP/IPv4 multicast)
2. 1588 version 2 (UDP/IPv4 multicast)
3. 1588 version 2 (UDP/IPv6 multicast)
4. 1588 version 2 (Ethernet multicast)

Table 37-7. Example of Sync Frame in 1588 Version 1 Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	—
UDP (Octet 23)	11
IP stuff (Octets 24–29)	—
IP DA (Octets 30–32)	E00001
IP DA (Octet 33)	81 or 82 or 83 or 84
Source IP port (Octets 34–35)	—
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–42)	—
Version PTP (Octet 43)	01
Other stuff (Octets 44–73)	—
Control (Octet 74)	00
Other stuff (Octets 75–168)	—

Table 37-8. Example of Delay Request Frame in 1588 Version 1 Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	—
UDP (Octet 23)	11
IP stuff (Octets 24–29)	—
IP DA (Octets 30–32)	E00001
IP DA (Octet 33)	81 or 82 or 83 or 84
Source IP port (Octets 34–35)	—
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–42)	—
Version PTP (Octet 43)	01
Other stuff (Octets 44–73)	—
Control (Octet 74)	01
Other stuff (Octets 75–168)	—

For 1588 version 1 messages, sync and delay request frames are indicated by the GMAC if the frame type field indicates TCP/IP, UDP protocol is indicated, the destination IP address is 224.0.1.129/130/131 or 132, the destination UDP port is 319 and the control field is correct.

The control field is 0x00 for sync frames and 0x01 for delay request frames.

For 1588 version 2 messages, the type of frame is determined by looking at the message type field in the first byte of the PTP frame. Whether a frame is version 1 or version 2 can be determined by looking at the version PTP field in the second byte of both version 1 and version 2 PTP frames.

In version 2 messages sync frames have a message type value of 0x0, delay_req have 0x1, Pdelay_Req have 0x2 and Pdelay_Resp have 0x3.

Table 37-9. Example of Sync Frame in 1588 Version 2 (UDP/IPv4) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	—
UDP (Octet 23)	11
IP stuff (Octets 24–29)	—
IP DA (Octets 30–33)	E0000181
Source IP port (Octets 34–35)	—
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–41)	—
Message type (Octet 42)	00
Version PTP (Octet 43)	02

Table 37-10. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	0800
IP stuff (Octets 14–22)	—
UDP (Octet 23)	11
IP stuff (Octets 24–29)	—
IP DA (Octets 30–33)	E000006B
Source IP port (Octets 34–35)	—
Dest IP port (Octets 36–37)	013F
Other stuff (Octets 38–41)	—
Message type (Octet 42)	02

Table 37-10. Example of Pdelay_Req Frame in 1588 Version 2 (UDP/IPv4) Format (Continued)

Frame Segment	Value
Version PTP (Octet 43)	02

Table 37-11. Example of Sync Frame in 1588 Version 2 (UDP/IPv6) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	86dd
IP stuff (Octets 14–19)	—
UDP (Octet 20)	11
IP stuff (Octets 21–37)	—
IP DA (Octets 38–53)	FF0X00000000018
Source IP port (Octets 54–55)	—
Dest IP port (Octets 56–57)	013F
Other stuff (Octets 58–61)	—
Message type (Octet 62)	00
Other stuff (Octets 63–93)	—
Version PTP (Octet 94)	02

Table 37-12. Example of Pdelay_Resp Frame in 1588 Version 2 (UDP/IPv6) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	—
SA (Octets 6–11)	—
Type (Octets 12–13)	86dd
IP stuff (Octets 14–19)	—
UDP (Octet 20)	11
IP stuff (Octets 21–37)	—
IP DA (Octets 38–53)	FF0200000000006B
Source IP port (Octets 54–55)	—
Dest IP port (Octets 56–57)	013F
Other stuff (Octets 58–61)	—
Message type (Octet 62)	03
Other stuff (Octets 63–93)	—
Version PTP (Octet 94)	02

For the multicast address 011B19000000 sync and delay request frames are recognized depending on the message type field, 00 for sync and 01 for delay request.

Table 37-13. Example of Sync Frame in 1588 Version 2 (Ethernet Multicast) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	011B19000000
SA (Octets 6–11)	—
Type (Octets 12–13)	88F7
Message type (Octet 14)	00
Version PTP (Octet 15)	02

Pdelay request frames need a special multicast address so they can pass through ports blocked by the spanning tree protocol. For the multicast address 0180C200000E sync, Pdelay_Req and Pdelay_Resp frames are recognized depending on the message type field, 00 for sync, 02 for pdelay request and 03 for pdelay response.

Table 37-14. Example of Pdelay_Req Frame in 1588 Version 2 (Ethernet Multicast) Format

Frame Segment	Value
Preamble/SFD	55555555555555D5
DA (Octets 0–5)	0180C200000E
SA (Octets 6–11)	—
Type (Octets 12–13)	88F7
Message type (Octet 14)	00
Version PTP (Octet 15)	02

37.6.15 Time Stamp Unit

The TSU consists of a timer and registers to capture the time at which PTP event frames cross the message timestamp point. An interrupt is issued when a capture register is updated.

The timer is implemented as a 94-bit register with the upper 48 bits counting seconds, the next 30 bits counting nanoseconds and the lowest 16 bits counting sub-nanoseconds. The lower 46 bits rolls over when they have counted to one second. An interrupt is generated when the seconds increment. The timer value can be read, written and adjusted through the APB interface. The timer is clocked by MCK.

The amount by which the timer increments each clock cycle is controlled by the timer increment registers (GMAC_TI). Bits 7:0 are the default increment value in nanoseconds and an additional 16 bits of sub-nanosecond resolution are available using the Timer Increment Sub-nanoseconds register (GMAC_TISUBN). If the rest of the register is written with zero, the timer increments by the value in [7:0], plus the value of GMAC_TISUBN, each clock cycle.

The GMAC_TISUBN register allows a resolution of approximately 15 femtoseconds.

Bits 15:8 of the increment register are the alternative increment value in nanoseconds and bits 23:16 are the number of increments after which the alternative increment value is used. If 23:16 are zero then the alternative increment value will never be used.

Taking the example of 10.2 MHz, there are 102 cycles every ten microseconds or 51 every five microseconds. So a timer with a 10.2 MHz clock source is constructed by incrementing by 98 ns for fifty cycles and then incrementing

by 100 ns (98 × 50 + 100 = 5000). This is programmed by setting the 1588 Timer Increment register to 0x00326462.

For a 49.8 MHz clock source it would be 20 ns for 248 cycles followed by an increment of 40 ns (20 × 248 + 40 = 5000) programmed as 0x00F82814.

Having eight bits for the “number of increments” field allows frequencies up to 50 MHz to be supported with 200 kHz resolution.

Without the alternative increment field the period of the clock would be limited to an integer number of nanoseconds, resulting in supported clock frequencies of 8, 10, 20, 25, 40, 50, 100, 125, 200 and 250 MHz.

There are eight additional 80-bit registers that capture the time at which PTP event frames are transmitted and received. An interrupt is issued when these registers are updated. The TSU timer count value can be compared to a programmable comparison value. For the comparison, the 48 bits of the seconds value and the upper 22 bits of the nanoseconds value are used. A signal (GTSUCOMP) is provided to indicate when the TSU timer count value is equal to the comparison value stored in the TSU timer comparison value registers (0x0DC, 0x0E0, and 0x0E4). The GTSUCOMP signal can be routed to the Timer peripheral to automatically toggle pin TIOB11/PD22. This can be used as the reference clock for an external PLL to regenerate the audio clock in Ethernet AVB. An interrupt can also be generated (if enabled) when the TSU timer count value and comparison value are equal, mapped to bit 29 of the Interrupt Status register.

37.6.16 MAC 802.3 Pause Frame Support

Note: See Clause 31, and Annex 31A and 31B of the IEEE standard 802.3 for a full description of MAC 802.3 pause operation.

The following table shows the start of a MAC 802.3 pause frame.

Table 37-15. Start of an 802.3 Pause Frame

Address		Type (MAC Control Frame)	Pause	
Destination	Source		Opcode	Time
0x0180C2000001	6 bytes	0x8808	0x0001	2 bytes

The GMAC supports both hardware controlled pause of the transmitter, upon reception of a pause frame, and hardware generated pause frame transmission.

37.6.16.1 802.3 Pause Frame Reception

Bit 13 of the Network Configuration register is the pause enable control for reception. If this bit is set, transmission pauses if a non-zero pause quantum frame is received.

If a valid pause frame is received, then the Pause Time register is updated with the new frame's pause time, regardless of whether a previous pause frame is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non-zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register.

Once the Pause Time register is loaded and the frame currently being transmitted has been sent, no new frames are transmitted until the pause time reaches zero. The loading of a new pause time, and hence the pausing of transmission, only occurs when the GMAC is configured for full duplex operation. If the GMAC is configured for half duplex there will be no transmission pause, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0001.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. 802.3 Pause frames that are received after Priority-based Flow Control (PFC) has been negotiated will also be discarded. Valid pause frames received will increment the Pause Frames Received statistic register.

The Pause Time register decrements every 512 bit times once transmission has stopped. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GTXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

37.6.16.2 802.3 Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit pause frame bits of the Network Control register. If either bit 11 or bit 12 of the Network Control register is written with logic 1, an 802.3 pause frame will be transmitted, providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address 1 register
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 00-01
- A Pause Quantum register
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The pause quantum used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 11 is written with a one, the pause quantum will be taken from the Transmit Pause Quantum register. The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.
- If bit 12 is written with a one, the pause quantum will be zero.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only the statistics register Pause Frames Transmitted is incremented.

Pause frames can also be transmitted by the MAC using normal frame transmission methods.

37.6.17 MAC PFC Priority-based Pause Frame Support

Note: Refer to the 802.1Qbb standard for a full description of priority-based pause operation.

The following table shows the start of a Priority-based Flow Control (PFC) pause frame.

Table 37-16. Start of a PFC Pause Frame

Address		Type (Mac Control Frame)	Pause Opcode	Priority Enable Vector	Pause Time
Destination	Source				
0x0180C2000001	6 bytes	0x8808	0x1001	2 bytes	8 × 2 bytes

The GMAC supports PFC priority-based pause transmission and reception. Before PFC pause frames can be received, bit 16 of the Network Control register must be set.

37.6.17.1 PFC Pause Frame Reception

The ability to receive and decode priority-based pause frames is enabled by setting bit 16 of the Network Control register. When this bit is set, the GMAC will match either classic 802.3 pause frames or PFC priority-based pause frames. Once a priority-based pause frame has been received and matched, then from that moment on the GMAC will only match on priority-based pause frames (this is an 802.1Qbb requirement, known as PFC negotiation). Once priority-based pause has been negotiated, any received 802.3x format pause frames will not be acted upon.

If a valid priority-based pause frame is received then the GMAC will decode the frame and determine which, if any, of the eight priorities require to be paused. Up to eight Pause Time registers are then updated with the eight pause times extracted from the frame regardless of whether a previous pause operation is active or not. An interrupt (either bit 12 or bit 13 of the Interrupt Status register) is triggered when a pause frame is received, but only if the interrupt has been enabled (bit 12 and bit 13 of the Interrupt Mask register). Pause frames received with non-zero quantum are indicated through the interrupt bit 12 of the Interrupt Status register. Pause frames received with zero quantum are indicated on bit 13 of the Interrupt Status register. The loading of a new pause time only occurs when the GMAC is configured for full duplex operation. If the GMAC is configured for half duplex, the pause time counters will not be loaded, but the pause frame received interrupt will still be triggered. A valid pause frame is defined as having a destination address that matches either the address stored in Specific Address 1 register or if it matches the reserved address of 0x0180C2000001. It must also have the MAC control frame type ID of 0x8808 and have the pause opcode of 0x0101.

Pause frames that have frame check sequence (FCS) or other errors will be treated as invalid and will be discarded. Valid pause frames received will increment the Pause Frames Received Statistic register.

The Pause Time registers decrement every 512 bit times immediately following the PFC frame reception. For test purposes, the retry test bit can be set (bit 12 in the Network Configuration register) which causes the Pause Time register to decrement every GRXCK cycle once transmission has stopped.

The interrupt (bit 13 in the Interrupt Status register) is asserted whenever the Pause Time register decrements to zero (assuming it has been enabled by bit 13 in the Interrupt Mask register). This interrupt is also set when a zero quantum pause frame is received.

37.6.17.2 PFC Pause Frame Transmission

Automatic transmission of pause frames is supported through the transmit priority-based pause frame bit of the Network Control register. If bit 17 of the Network Control register is written with logic 1, a PFC pause frame will be transmitted providing full duplex is selected in the Network Configuration register and the transmit block is enabled in the Network Control register. When bit 17 of the Network Control register is set, the fields of the priority-based pause frame will be built using the values stored in the Transmit PFC Pause register.

Pause frame transmission will happen immediately if transmit is inactive or if transmit is active between the current frame and the next frame due to be transmitted.

Transmitted pause frames comprise the following:

- A destination address of 01-80-C2-00-00-01
- A source address taken from Specific Address 1 register
- A type ID of 88-08 (MAC control frame)
- A pause opcode of 01-01
- A priority enable vector taken from Transmit PFC Pause register
- 8 Pause Quantum registers
- Fill of 00 to take the frame to minimum frame length
- Valid FCS

The Pause Quantum registers used in the generated frame will depend on the trigger source for the frame as follows:

- If bit 17 of the Network Control register is written with a one, then the priority enable vector of the priority-based pause frame will be set equal to the value stored in the Transmit PFC Pause register [7:0]. For each entry equal to zero in the Transmit PFC Pause register [15:8], the pause quantum field of the pause frame associated with that entry will be taken from the Transmit Pause Quantum register. For each entry equal to one in the Transmit PFC Pause register [15:8], the pause quantum associated with that entry will be zero.
- The Transmit Pause Quantum register resets to a value of 0xFFFF giving maximum pause quantum as default.

After transmission, a pause frame transmitted interrupt will be generated (bit 14 of the Interrupt Status register) and the only statistics register that will be incremented will be the Pause Frames Transmitted register.

PFC Pause frames can also be transmitted by the MAC using normal frame transmission methods.

37.6.18 802.1Qav Support - Credit-based Shaping

A credit-based shaping algorithm is available on the two highest priority queues and is defined in the standard 802.1Qav: Forwarding and Queuing Enhancements for Time-Sensitive Streams. This allows traffic on these queues to be limited and to allow other queues to transmit.

Traffic shaping is enabled via the CBS (Credit Based Shaping) Control register. This enables a counter which stores the amount of transmit 'credit', measured in bytes that a particular queue has. A queue may only transmit if it has non-negative credit. If a queue has data to send, but is held off from doing as another queue is transmitting, then credit will accumulate in the credit counter at the rate defined in the IdleSlope register (GMAC_CBSISQx) for that queue. IdleSlope is the rate of change of credit when waiting to transmit and must be less than the value of the portTransmitRate. When this queue is transmitting the credit counter is decremented at the rate of sendSlope which is defined as the portTransmitRate - IdleSlope. A queue can accumulate negative credit when transmitting which will hold off any other transfers from that queue until credit returns to a non-negative value. No transfers are halted when a queue's credit becomes negative; it will accumulate negative credit until the transfer completes.

If both queues have positive credit, when the next queue to transfer is about to be selected, the queue with the most positive credit will be allowed to transfer first. The queue with the largest positive credit is the queue that had been prevented from transmitting for the longest time.

37.6.19 PHY Interface

Different PHY interfaces are supported by the Ethernet MAC:

- MII
- RMII

The MII interface is provided for 10/100 operation and uses txd[3:0] and rxd[3:0]. The RMII interface is provided for 10/100 operation and uses txd[1:0] and rxd[1:0].

37.6.20 10/100 Operation

The 10/100 Mbps speed bit in the Network Configuration register is used to select between 10 Mbps and 100 Mbps.

37.6.21 Jumbo Frames

The jumbo frames enable bit in the Network Configuration register allows the GMAC, in its default configuration, to receive jumbo frames up to 10240 bytes in size. This operation does not form part of the IEEE 802.3 specification and is normally disabled. When jumbo frames are enabled, frames received with a frame size greater than 10240 bytes are discarded.

37.7 Programming Interface

37.7.1 Initialization

37.7.1.1 Configuration

Initialization of the GMAC configuration (e.g., loop back mode, frequency ratios) must be done while the transmit and receive circuits are disabled. See the description of the Network Control register and Network Configuration register earlier in this document.

To change loop back mode, the following sequence of operations must be followed:

1. Write to Network Control register to disable transmit and receive circuits.
2. Write to Network Control register to change loop back mode.
3. Write to Network Control register to re-enable transmit or receive circuits.

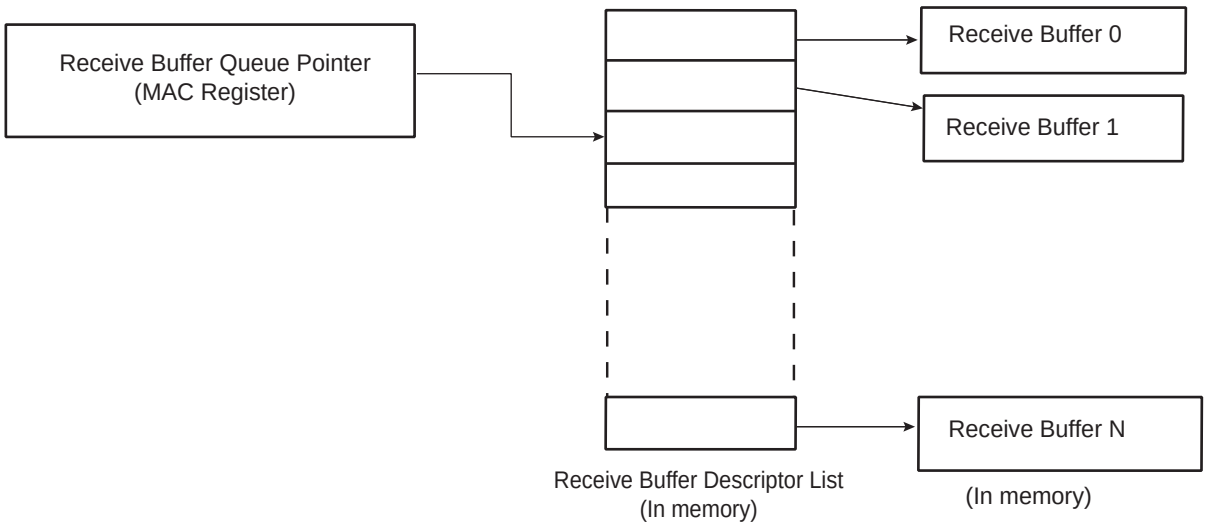
Note: These writes to the Network Control register cannot be combined in any way.

37.7.1.2 Receive Buffer List

Receive data is written to areas of data (i.e., buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (receive buffer queue) is a sequence of descriptor entries as defined in Table 37-4 “Receive Buffer Descriptor Entry”.

The Receive Buffer Queue Pointer register points to this data structure.

Figure 37-3. Receive Buffer List



To create the list of buffers:

1. Allocate a number (N) of buffers of X bytes in system memory, where X is the DMA buffer length programmed in the DMA Configuration register.
2. Allocate an area 8N bytes for the receive buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 0 of word 0 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 1 in word 0 set to 1).
4. Write address of receive buffer descriptor list and control information to GMAC register receive buffer queue pointer
5. The receive circuits can then be enabled by writing to the address recognition registers and the Network Control register.

Note: The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

37.7.1.3 Transmit Buffer List

Transmit data is read from areas of data (the buffers) in system memory. These buffers are listed in another data structure that also resides in main memory. This data structure (Transmit Buffer Queue) is a sequence of descriptor entries as defined in Table 37-5 “Transmit Buffer Descriptor Entry”.

The Transmit Buffer Queue Pointer register points to this data structure.

To create this list of buffers:

1. Allocate a number (N) of buffers of between 1 and 2047 bytes of data to be transmitted in system memory. Up to 128 buffers per frame are allowed.
2. Allocate an area 8N bytes for the transmit buffer descriptor list in system memory and create N entries in this list. Mark all entries in this list as owned by GMAC, i.e., bit 31 of word 1 set to 0.
3. Mark the last descriptor in the queue with the wrap bit (bit 30 in word 1 set to 1).
4. Write address of transmit buffer descriptor list and control information to GMAC register transmit buffer queue pointer.
5. The transmit circuits can then be enabled by writing to the Network Control register.

Note: The queue pointers must be initialized and point to USED descriptors for all queues including those not intended for use.

37.7.1.4 Address Matching

The GMAC Hash register pair and the four Specific Address register pairs must be written with the required values. Each register pair comprises of a bottom register and top register, with the bottom register being written first. The address matching is disabled for a particular register pair after the bottom register has been written and re-enabled when the top register is written. Each register pair may be written at any time, regardless of whether the receive circuits are enabled or disabled.

As an example, to set Specific Address 1 register to recognize destination address 21:43:65:87:A9:CB, the following values are written to Specific Address 1 Bottom register and Specific Address 1 Top register:

- Specific Address 1 Bottom register bits 31:0 (0x98): 0x8765_4321.
- Specific Address 1 Top register bits 31:0 (0x9C): 0x0000_CBA9.

Note: The address matching is the first level of filtering. If there is a match, the screeners are the next level of filtering for routing the data to the appropriate queue. See Section 37.6.3.9 ”Priority Queueing in the DMA” for more details.

37.7.1.5 PHY Maintenance

The PHY Maintenance register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit two is set in the Network Status register (about 2000 MCK cycles later when bits 18:16 are set to 010 in the Network Configuration register). An interrupt is generated as this bit is set.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each Management Data Clock (MDC) cycle. This causes the transmission of a PHY management frame on MDIO. See section 22.2.4.5 of the IEEE 802.3 standard.

Reading during the shift operation will return the current contents of the shift register. At the end of the management operation the bits will have shifted back to their original locations. For a read operation the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The Management Data Clock (MDC) should not toggle faster than 2.5 MHz (minimum period of 400 ns), as defined by the IEEE 802.3 standard. MDC is generated by dividing down MCK. Three bits in the Network Configuration register determine by how much MCK should be divided to produce MDC.

37.7.1.6 Interrupts

There are 18 interrupt conditions that are detected within the GMAC. The conditions are ORed to make multiple interrupts. Depending on the overall system design this may be passed through a further level of interrupt collection (interrupt controller). On receipt of the interrupt signal, the CPU enters the interrupt handler. Refer to the device interrupt controller documentation to identify that it is the GMAC that is generating the interrupt. To ascertain which interrupt, read the Interrupt Status register. Note that in the default configuration this register will clear itself after being read, though this may be configured to be write-one-to-clear if desired.

At reset all interrupts are disabled. To enable an interrupt, write to Interrupt Enable register with the pertinent interrupt bit set to 1. To disable an interrupt, write to Interrupt Disable register with the pertinent interrupt bit set to 1. To check whether an interrupt is enabled or disabled, read Interrupt Mask register. If the bit is set to 1, the interrupt is disabled.

37.7.1.7 Transmitting Frames

The procedure to set up a frame for transmission is the following:

1. Enable transmit in the Network Control register.
2. Allocate an area of system memory for transmit data. This does not have to be contiguous, varying byte lengths can be used if they conclude on byte borders.
3. Set-up the transmit buffer list by writing buffer addresses to word zero of the transmit buffer descriptor entries and control and length to word one.
4. Write data for transmission into the buffers pointed to by the descriptors.
5. Write the address of the first buffer descriptor to transmit buffer descriptor queue pointer.
6. Enable appropriate interrupts.
7. Write to the transmit start bit (TSTART) in the Network Control register.

37.7.1.8 Receiving Frames

When a frame is received and the receive circuits are enabled, the GMAC checks the address and, in the following cases, the frame is written to system memory:

- If it matches one of the four Specific Address registers.
- If it matches one of the four Type ID registers.
- If it matches the hash address function.
- If it is a broadcast address (0xFFFFFFFFFFFF) and broadcasts are allowed.
- If the GMAC is configured to “copy all frames”.

The register receive buffer queue pointer points to the next entry in the receive buffer descriptor list and the GMAC uses this as the address in system memory to write the frame to.

Once the frame has been completely and successfully received and written to system memory, the GMAC then updates the receive buffer descriptor entry (see Table 37-4 “Receive Buffer Descriptor Entry”) with the reason for the address match and marks the area as being owned by software. Once this is complete, a receive complete interrupt is set. Software is then responsible for copying the data to the application area and releasing the buffer (by writing the ownership bit back to 0).

If the GMAC is unable to write the data at a rate to match the incoming frame, then a receive overrun interrupt is set. If there is no receive buffer available, i.e., the next buffer is still owned by software, a receive buffer not available interrupt is set. If the frame is not successfully received, a statistics register is incremented and the frame is discarded without informing software.

37.7.2 Statistics Registers

Statistics registers are described in the User Interface beginning with Section 37.8.47 ”GMAC Octets Transmitted Low Register” and ending with Section 37.8.91 ”GMAC UDP Checksum Errors Register”.

The statistics register block begins at 0x100 and runs to 0x1B0, and comprises the registers listed below.

- Octets Transmitted Low Register
- Octets Transmitted High Register
- Frames Transmitted Register
- Broadcast Frames Transmitted Register
- Multicast Frames Transmitted Register
- Pause Frames Transmitted Register
- 64 Byte Frames Transmitted Register
- 65 to 127 Byte Frames Transmitted Register
- 128 to 255 Byte Frames Transmitted Register
- 256 to 511 Byte Frames Transmitted Register
- 512 to 1023 Byte Frames Transmitted Register
- 1024 to 1518 Byte Frames Transmitted Register
- Greater Than 1518 Byte Frames Transmitted Register
- Transmit Underruns Register
- Single Collision Frames Register
- Multiple Collision Frames Register
- Excessive Collisions Register
- Late Collisions Register
- Deferred Transmission Frames Register
- Carrier Sense Errors Register
- Octets Received Low Register
- Octets Received High Register
- Frames Received Register
- Broadcast Frames Received Register
- Multicast Frames Received Register
- Pause Frames Received Register
- 64 Byte Frames Received Register
- 65 to 127 Byte Frames Received Register
- 128 to 255 Byte Frames Received Register
- 256 to 511 Byte Frames Received Register
- 512 to 1023 Byte Frames Received Register
- 1024 to 1518 Byte Frames Received Register
- 1519 to Maximum Byte Frames Received Register
- Undersize Frames Received Register
- Oversize Frames Received Register
- Jabbers Received Register
- Frame Check Sequence Errors Register
- Length Field Frame Errors Register
- Receive Symbol Errors Register
- Alignment Errors Register
- Receive Resource Errors Register
- Receive Overrun Register
- IP Header Checksum Errors Register
- TCP Checksum Errors Register
- UDP Checksum Errors Register

These registers reset to zero on a read and stick at all ones when they count to their maximum value. They should be read frequently enough to prevent loss of data.

The receive statistics registers are only incremented when the receive enable bit (RXEN) is set in the Network Control register.

Once a statistics register has been read, it is automatically cleared. When reading the Octets Transmitted and Octets Received registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

37.8 Ethernet MAC (GMAC) User Interface

Table 37-17. Register Mapping

Offset[1] [2]	Register	Name	Access	Reset
0x000	Network Control Register	GMAC_NCR	Read/Write	0x0000_0000
0x004	Network Configuration Register	GMAC_NCFGR	Read/Write	0x0008_0000
0x008	Network Status Register	GMAC_NSR	Read-only	0b01x0
0x00C	User Register	GMAC_UR	Read/Write	0x0000_0000
0x010	DMA Configuration Register	GMAC_DCFGR	Read/Write	0x0002_0004
0x014	Transmit Status Register	GMAC_TSR	Read/Write	0x0000_0000
0x018	Receive Buffer Queue Base Address Register	GMAC_RBQB	Read/Write	0x0000_0000
0x01C	Transmit Buffer Queue Base Address Register	GMAC_TBQB	Read/Write	0x0000_0000
0x020	Receive Status Register	GMAC_RSR	Read/Write	0x0000_0000
0x024	Interrupt Status Register	GMAC_ISR	Read-only	0x0000_0000
0x028	Interrupt Enable Register	GMAC_IER	Write-only	–
0x02C	Interrupt Disable Register	GMAC_IDR	Write-only	–
0x030	Interrupt Mask Register	GMAC_IMR	Read/Write	0x07FF_FFFF
0x034	PHY Maintenance Register	GMAC_MAN	Read/Write	0x0000_0000
0x038	Received Pause Quantum Register	GMAC_RPQ	Read-only	0x0000_0000
0x03C	Transmit Pause Quantum Register	GMAC_TPQ	Read/Write	0x0000_FFFF
0x040	TX Partial Store and Forward Register	GMAC_TPSF	Read/Write	0x0000_0FFF
0x044	RX Partial Store and Forward Register	GMAC_RPSF	Read/Write	0x0000_0FFF
0x048	RX Jumbo Frame Max Length Register	GMAC_RJFML	Read/Write	0x0000_3FFF
0x4C–0x07C	Reserved	–	–	–
0x080	Hash Register Bottom	GMAC_HRB	Read/Write	0x0000_0000
0x084	Hash Register Top	GMAC_HRT	Read/Write	0x0000_0000
0x088	Specific Address 1 Bottom Register	GMAC_SAB1	Read/Write	0x0000_0000
0x08C	Specific Address 1 Top Register	GMAC_SAT1	Read/Write	0x0000_0000
0x090	Specific Address 2 Bottom Register	GMAC_SAB2	Read/Write	0x0000_0000
0x094	Specific Address 2 Top Register	GMAC_SAT2	Read/Write	0x0000_0000
0x098	Specific Address 3 Bottom Register	GMAC_SAB3	Read/Write	0x0000_0000
0x09C	Specific Address 3 Top Register	GMAC_SAT3	Read/Write	0x0000_0000
0x0A0	Specific Address 4 Bottom Register	GMAC_SAB4	Read/Write	0x0000_0000
0x0A4	Specific Address 4 Top Register	GMAC_SAT4	Read/Write	0x0000_0000
0x0A8	Type ID Match 1 Register	GMAC_TIDM1	Read/Write	0x0000_0000
0x0AC	Type ID Match 2 Register	GMAC_TIDM2	Read/Write	0x0000_0000
0x0B0	Type ID Match 3 Register	GMAC_TIDM3	Read/Write	0x0000_0000
0x0B4	Type ID Match 4 Register	GMAC_TIDM4	Read/Write	0x0000_0000
0x0B8	Wake on LAN Register	GMAC_WOL	Read/Write	0x0000_0000

Table 37-17. Register Mapping (Continued)

Offset[1] [2]	Register	Name	Access	Reset
0x0BC	IPG Stretch Register	GMAC_IPGS	Read/Write	0x0000_0000
0x0C0	Stacked VLAN Register	GMAC_SVLAN	Read/Write	0x0000_0000
0x0C4	Transmit PFC Pause Register	GMAC_TPFCP	Read/Write	0x0000_0000
0x0C8	Specific Address 1 Mask Bottom Register	GMAC_SAMB1	Read/Write	0x0000_0000
0x0CC	Specific Address 1 Mask Top Register	GMAC_SAMT1	Read/Write	0x0000_0000
0x0D0–0x0D8	Reserved	–	–	–
0x0DC	1588 Timer Nanosecond Comparison Register	GMAC_NSC	Read/Write	0x0000_0000
0x0E0	1588 Timer Second Comparison Low Register	GMAC_SCL	Read/Write	0x0000_0000
0x0E4	1588 Timer Second Comparison High Register	GMAC_SCH	Read/Write	0x0000_0000
0x0E8	PTP Event Frame Transmitted Seconds High Register	GMAC_EFTSH	Read-only	0x0000_0000
0x0EC	PTP Event Frame Received Seconds High Register	GMAC_EFRSH	Read-only	0x0000_0000
0x0F0	PTP Peer Event Frame Transmitted Seconds High Register	GMAC_PEFTSH	Read-only	0x0000_0000
0x0F4	PTP Peer Event Frame Received Seconds High Register	GMAC_PEFRSH	Read-only	0x0000_0000
0x0F8–0x0FC	Reserved	–	–	–
0x100	Octets Transmitted Low Register	GMAC_OTLO	Read-only	0x0000_0000
0x104	Octets Transmitted High Register	GMAC_OTHI	Read-only	0x0000_0000
0x108	Frames Transmitted Register	GMAC_FT	Read-only	0x0000_0000
0x10C	Broadcast Frames Transmitted Register	GMAC_BCFT	Read-only	0x0000_0000
0x110	Multicast Frames Transmitted Register	GMAC_MFT	Read-only	0x0000_0000
0x114	Pause Frames Transmitted Register	GMAC_PFT	Read-only	0x0000_0000
0x118	64 Byte Frames Transmitted Register	GMAC_BFT64	Read-only	0x0000_0000
0x11C	65 to 127 Byte Frames Transmitted Register	GMAC_TBFT127	Read-only	0x0000_0000
0x120	128 to 255 Byte Frames Transmitted Register	GMAC_TBFT255	Read-only	0x0000_0000
0x124	256 to 511 Byte Frames Transmitted Register	GMAC_TBFT511	Read-only	0x0000_0000
0x128	512 to 1023 Byte Frames Transmitted Register	GMAC_TBFT1023	Read-only	0x0000_0000
0x12C	1024 to 1518 Byte Frames Transmitted Register	GMAC_TBFT1518	Read-only	0x0000_0000
0x130	Greater Than 1518 Byte Frames Transmitted Register	GMAC_GTBFT1518	Read-only	0x0000_0000
0x134	Transmit Underruns Register	GMAC_TUR	Read-only	0x0000_0000
0x138	Single Collision Frames Register	GMAC_SCF	Read-only	0x0000_0000
0x13C	Multiple Collision Frames Register	GMAC_MCF	Read-only	0x0000_0000
0x140	Excessive Collisions Register	GMAC_EC	Read-only	0x0000_0000
0x144	Late Collisions Register	GMAC_LC	Read-only	0x0000_0000
0x148	Deferred Transmission Frames Register	GMAC_DTF	Read-only	0x0000_0000
0x14C	Carrier Sense Errors Register	GMAC_CSE	Read-only	0x0000_0000

Table 37-17. Register Mapping (Continued)

Offset[1] [2]	Register	Name	Access	Reset
0x150	Octets Received Low Received Register	GMAC_ORLO	Read-only	0x0000_0000
0x154	Octets Received High Received Register	GMAC_ORHI	Read-only	0x0000_0000
0x158	Frames Received Register	GMAC_FR	Read-only	0x0000_0000
0x15C	Broadcast Frames Received Register	GMAC_BCFR	Read-only	0x0000_0000
0x160	Multicast Frames Received Register	GMAC_MFR	Read-only	0x0000_0000
0x164	Pause Frames Received Register	GMAC_PFR	Read-only	0x0000_0000
0x168	64 Byte Frames Received Register	GMAC_BFR64	Read-only	0x0000_0000
0x16C	65 to 127 Byte Frames Received Register	GMAC_TBFR127	Read-only	0x0000_0000
0x170	128 to 255 Byte Frames Received Register	GMAC_TBFR255	Read-only	0x0000_0000
0x174	256 to 511 Byte Frames Received Register	GMAC_TBFR511	Read-only	0x0000_0000
0x178	512 to 1023 Byte Frames Received Register	GMAC_TBFR1023	Read-only	0x0000_0000
0x17C	1024 to 1518 Byte Frames Received Register	GMAC_TBFR1518	Read-only	0x0000_0000
0x180	1519 to Maximum Byte Frames Received Register	GMAC_TMXBFR	Read-only	0x0000_0000
0x184	Undersize Frames Received Register	GMAC_UFR	Read-only	0x0000_0000
0x188	Oversize Frames Received Register	GMAC_OFR	Read-only	0x0000_0000
0x18C	Jabbers Received Register	GMAC_JR	Read-only	0x0000_0000
0x190	Frame Check Sequence Errors Register	GMAC_FCSE	Read-only	0x0000_0000
0x194	Length Field Frame Errors Register	GMAC_LFFE	Read-only	0x0000_0000
0x198	Receive Symbol Errors Register	GMAC_RSE	Read-only	0x0000_0000
0x19C	Alignment Errors Register	GMAC_AE	Read-only	0x0000_0000
0x1A0	Receive Resource Errors Register	GMAC_RRE	Read-only	0x0000_0000
0x1A4	Receive Overrun Register	GMAC_ROE	Read-only	0x0000_0000
0x1A8	IP Header Checksum Errors Register	GMAC_IHCE	Read-only	0x0000_0000
0x1AC	TCP Checksum Errors Register	GMAC_TCE	Read-only	0x0000_0000
0x1B0	UDP Checksum Errors Register	GMAC_UCE	Read-only	0x0000_0000
0x1B4–0x1B8	Reserved	–	–	–
0x1BC	1588 Timer Increment Sub-nanoseconds Register	GMAC_TISUBN	Read/Write	0x0000_0000
0x1C0	1588 Timer Seconds High Register	GMAC_TSH	Read/Write	0x0000_0000
0x1C4–0x1CC	Reserved	–	–	–
0x1D0	1588 Timer Seconds Low Register	GMAC_TSL	Read/Write	0x0000_0000
0x1D4	1588 Timer Nanoseconds Register	GMAC_TN	Read/Write	0x0000_0000
0x1D8	1588 Timer Adjust Register	GMAC_TA	Write-only	–
0x1DC	1588 Timer Increment Register	GMAC_TI	Read/Write	0x0000_0000
0x1E0	PTP Event Frame Transmitted Seconds Low Register	GMAC_EFTSL	Read-only	0x0000_0000
0x1E4	PTP Event Frame Transmitted Nanoseconds Register	GMAC_EFTN	Read-only	0x0000_0000
0x1E8	PTP Event Frame Received Seconds Low Register	GMAC_EFRSL	Read-only	0x0000_0000

Table 37-17. Register Mapping (Continued)

Offset[1] [2]	Register	Name	Access	Reset
0x1EC	PTP Event Frame Received Nanoseconds Register	GMAC_EFRN	Read-only	0x0000_0000
0x1F0	PTP Peer Event Frame Transmitted Seconds Low Register	GMAC_PEFTSL	Read-only	0x0000_0000
0x1F4	PTP Peer Event Frame Transmitted Nanoseconds Register	GMAC_PEFTN	Read-only	0x0000_0000
0x1F8	PTP Peer Event Frame Received Seconds Low Register	GMAC_PEFRSL	Read-only	0x0000_0000
0x1FC	PTP Peer Event Frame Received Nanoseconds Register	GMAC_PEFRN	Read-only	0x0000_0000
0x200–0x3FC	Reserved	–	–	–
0x3FC + (index * 0x04)	Interrupt Status Register Priority Queue [3]	GMAC_ISRPQ	Read-only	0x0000_0000
0x43C + (index * 0x04)	Transmit Buffer Queue Base Address Register Priority Queue [3]	GMAC_TBQBAPQ	Read/Write	0x0000_0000
0x47C + (index * 0x04)	Receive Buffer Queue Base Address Register Priority Queue [3]	GMAC_RBQBAPQ	Read/Write	0x0000_0000
0x49C + (index * 0x04)	Receive Buffer Size Register Priority Queue [3]	GMAC_RBSRPQ	Read/Write	0x0000_0002
0x4BC	Credit-Based Shaping Control Register	GMAC_CBSCR	Read/Write	0x0000_0000
0x4C0	Credit-Based Shaping IdleSlope Register for Queue A	GMAC_CBSISQA	Read/Write	0x0000_0000
0x4C4	Credit-Based Shaping IdleSlope Register for Queue B	GMAC_CBSISQB	Read/Write	0x0000_0000
0x500 + (index * 0x04)	Screening Type 1 Register Priority Queue [4]	GMAC_ST1RPQ	Read/Write	0x0000_0000
0x540 + (index * 0x04)	Screening Type 2 Register Priority Queue [5]	GMAC_ST2RPQ	Read/Write	0x0000_0000
0x5FC + (index * 0x04)	Interrupt Enable Register Priority Queue [3]	GMAC_IERPQ	Write-only	–
0x61C + (index * 0x04)	Interrupt Disable Register Priority Queue [3]	GMAC_IDRPQ	Write-only	–
0x63C + (index * 0x04)	Interrupt Mask Register Priority Queue [3]	GMAC_IMRPQ	Read/Write	0x0000_0000
0x6E0 + (index * 0x04)	Screening Type 2 Ethertype Register [6]	GMAC_ST2ER	Read/Write	0x0000_0000
0x700 + (index * 0x08)	Screening Type 2 Compare Word 0 Register [7]	GMAC_ST2CW0	Read/Write	0x0000_0000
0x704 + (index * 0x08)	Screening Type 2 Compare Word 1 Register [7]	GMAC_ST2CW1	Read/Write	0x0000_0000

Notes:
1. If an offset is not listed in the Register Mapping, it must be considered as ‘reserved’.
2. Some register groups are not continuous in memory.
3. The index range for the following registers is from 1 to 2:
   - GMAC_ISRPQ
   - GMAC_TBQBAPQ
   - GMAC_RBQBAPQ
   - GMAC_RBSRPQ
   - GMAC_IERPQ
   - GMAC_IDRPQ
   - GMAC_IMRPQ
4. The index for GMAC_ST1RPQ registers ranges from 0 to 3.
5. The index for GMAC_ST2RPQ registers ranges from 0 to 7.
6. The index for GMAC_ST2ER registers ranges from 0 to 3.
7. The index for GMAC_ST2CW0 and GMAC_ST2CW1 registers ranges from 0 to 23.

37.8.1 GMAC Network Control Register

Name: GMAC_NCR

Address: 0xF8008000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	FNP	TXPBPF	ENPBPR
15	14	13	12	11	10	9	8
SRTSM	–	–	TXZQPF	TXPF	THALT	TSTART	BP
7	6	5	4	3	2	1	0
WESTAT	INCSTAT	CLRSTAT	MPE	TXEN	RXEN	LBL	–

- **LBL: Loop Back Local**

Connects GTX to GRX，GTXEN to GRXDV and forces full duplex mode. GRXCK and GTXCK may malfunction as the GMAC is switched into and out of internal loop back. It is important that receive and transmit circuits have already been disabled when making the switch into and out of internal loop back.

- **RXEN: Receive Enable**

When set, RXEN enables the GMAC to receive data. When reset frame reception stops immediately and the receive pipeline will be cleared. The Receive Queue Pointer Register is unaffected.

- **TXEN: Transmit Enable**

When set, TXEN enables the GMAC transmitter to send data. When reset transmission will stop immediately, the transmit pipeline and control registers will be cleared and the Transmit Queue Pointer Register will reset to point to the start of the transmit descriptor list.

- **MPE: Management Port Enable**

Set to one to enable the management port. When zero, forces MDIO to high impedance state and MDC low.

- **CLRSTAT: Clear Statistics Registers**

This bit is write-only. Writing a one clears the statistics registers.

- **INCSTAT: Increment Statistics Registers**

This bit is write-only. Writing a one increments all the statistics registers by one for test purposes.

- **WESTAT: Write Enable for Statistics Registers**

Setting this bit to one makes the statistics registers writable for functional test purposes.

- **BP: Back pressure**

If set in 10M or 100M half duplex mode, forces collisions on all received frames.

- **TSTART: Start Transmission**

Writing one to this bit starts transmission.

- **THALT: Transmit Halt**

Writing one to this bit halts transmission as soon as any ongoing frame transmission ends.

- **TXPF: Transmit Pause Frame**

Writing one to this bit causes a pause frame to be transmitted.

- **TXZQPF: Transmit Zero Quantum Pause Frame**

Writing one to this bit causes a pause frame with zero quantum to be transmitted.

- **SRTSM: Store Receive Time Stamp to Memory**

0: Normal operation.

1: Causes the CRC of every received frame to be replaced with the value of the nanoseconds field of the 1588 timer that was captured as the receive frame passed the message time stamp point.

- **ENPBPR: Enable PFC Priority-based Pause Reception**

Enables PFC Priority Based Pause Reception capabilities. Setting this bit enables PFC negotiation and recognition of priority-based pause frames.

- **TXPBPF: Transmit PFC Priority-based Pause Frame**

Takes the values stored in the Transmit PFC Pause Register.

- **FNP: Flush Next Packet**

Flush the next packet from the external RX DPRAM. Writing one to this bit will only have an effect if the DMA is not currently writing a packet already stored in the DPRAM to memory.

37.8.2 GMAC Network Configuration Register

Name: GMAC_NCFGR

Address: 0xF8008004

Access: Read/Write

31	30	29	28	27	26	25	24
–	IRXER	RXBP	IPGSEN	–	IRXFCS	EFRHD	RXCOEN
23	22	21	20	19	18	17	16
DCPF	DBW		CLK			RFCS	LFERD
15	14	13	12	11	10	9	8
RXBUFO		PEN	RTY	–	–	–	MAXFS
7	6	5	4	3	2	1	0
UNIHEN	MTI HEN	NBC	CAF	JFRAME	DNVLAN	FD	SPD

- **SPD: Speed**

Set to logic one to indicate 100 Mbps operation, logic zero for 10 Mbps.

- **FD: Full Duplex**

If set to logic one, the transmit block ignores the state of collision and carrier sense and allows receive while transmitting.

- **DNVLAN: Discard Non-VLAN FRAMES**

When set only VLAN tagged frames will be passed to the address matching logic.

- **JFRAME: Jumbo Frame Size**

Set to one to enable jumbo frames up to 10240 bytes to be accepted. The default length is 10240 bytes.

- **CAF: Copy All Frames**

When set to logic one, all valid frames will be accepted.

- **NBC: No Broadcast**

When set to logic one, frames addressed to the broadcast address of all ones will not be accepted.

- **MTIHEN: Multicast Hash Enable**

When set, multicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.

- **UNIHEN: Unicast Hash Enable**

When set, unicast frames will be accepted when the 6-bit hash function of the destination address points to a bit that is set in the Hash Register.

- **MAXFS: 1536 Maximum Frame Size**

Setting this bit means the GMAC will accept frames up to 1536 bytes in length. Normally the GMAC would reject any frame above 1518 bytes.

- **RTY: Retry Test**

Must be set to zero for normal operation. If set to one the backoff between collisions will always be one slot time. Setting this bit to one helps test the too many retries condition. Also used in the pause frame tests to reduce the pause counter's decrement time from 512 bit times, to every GRXCK cycle.

- **PEN: Pause Enable**

When set, transmission will pause if a non-zero 802.3 classic pause frame is received and PFC has not been negotiated.

- **RXBUFO: Receive Buffer Offset**

Indicates the number of bytes by which the received data is offset from the start of the receive buffer

- **LFERD: Length Field Error Frame Discard**

Setting this bit causes frames with a measured length shorter than the extracted length field (as indicated by bytes 13 and 14 in a non-VLAN tagged frame) to be discarded. This only applies to frames with a length field less than 0x0600.

- **RFCS: Remove FCS**

Setting this bit will cause received frames to be written to memory without their frame check sequence (last 4 bytes). The frame length indicated will be reduced by four bytes in this mode.

- **CLK: MDC CLock Division**

Set according to MCK speed. These three bits determine the number MCK will be divided by to generate Management Data Clock (MDC). For conformance with the 802.3 specification, MDC must not exceed 2.5 MHz (MDC is only active during MDIO read and write operations).

Value	Name	Description
0	MCK_8	MCK divided by 8 (MCK up to 20 MHz)
1	MCK_16	MCK divided by 16 (MCK up to 40 MHz)
2	MCK_32	MCK divided by 32 (MCK up to 80 MHz)
3	MCK_48	MCK divided by 48 (MCK up to 120 MHz)
4	MCK_64	MCK divided by 64 (MCK up to 160 MHz)
5	MCK_96	MCK divided by 96 (MCK up to 240 MHz)

- **DBW: Data Bus Width**

Should always be written to ‘0’.

- **DCPF: Disable Copy of Pause Frames**

Set to one to prevent valid pause frames being copied to memory. When set, pause frames are not copied to memory regardless of the state of the Copy All Frames bit, whether a hash match is found or whether a type ID match is identified. If a destination address match is found, the pause frame will be copied to memory. Note that valid pause frames received will still increment pause statistics and pause the transmission of frames as required.

- **RXCOEN: Receive Checksum Offload Enable**

When set, the receive checksum engine is enabled. Frames with bad IP, TCP or UDP checksums are discarded.

- **EFRHD: Enable Frames Received in Half Duplex**

Enable frames to be received in half-duplex mode while transmitting.

- **IRXFCS: Ignore RX FCS**

When set, frames with FCS/CRC errors will not be rejected. FCS error statistics will still be collected for frames with bad FCS and FCS status will be recorded in frame’s DMA descriptor. For normal operation this bit must be set to zero.

- **IPGSEN: IP Stretch Enable**

When set, the transmit IPG can be increased above 96 bit times depending on the previous frame length using the IPG Stretch Register.

- **RXBP: Receive Bad Preamble**

When set, frames with non-standard preamble are not rejected.

- **IRXER: Ignore IPG GRXER**

When set, GRXER has no effect on the GMAC's operation when GRXDV is low.

37.8.3 GMAC Network Status Register

Name: GMAC_NSR

Address: 0xF8008008

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	IDLE	MDIO	–

- **MDIO: MDIO Input Status**

Returns status of the MDIO pin.

- **IDLE: PHY Management Logic Idle**

The PHY management logic is idle (i.e., has completed).

37.8.4 GMAC User Register

Name: GMAC_UR

Address: 0xF800800C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	RMII

- **RMII: Reduced MII Mode**

0: MII mode is selected (default).

1: RMII mode is selected.

37.8.5 GMAC DMA Configuration Register

Name: GMAC_DCFGR

Address: 0xF8008010

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	DDRP

23	22	21	20	19	18	17	16
DRBS							

15	14	13	12	11	10	9	8
–	–	–	–	TXCOEN	TXPBMS	RXBMS	

7	6	5	4	3	2	1	0
ESPA	ESMA	–	FBLDO				

- **FBLDO: Fixed Burst Length for DMA Data Operations:**

Selects the burst length to attempt to use on the AHB when transferring frame data. Not used for DMA management operations and only used where space and data size allow. Otherwise SINGLE type AHB transfers are used.

One-hot priority encoding enforced automatically on register writes as follows, where ‘x’ represents don’t care:

Value	Name	Description
0	–	Reserved
1	SINGLE	00001: Always use SINGLE AHB bursts
2	–	Reserved
4	INCR4	001xx: Attempt to use INCR4 AHB bursts (Default)
8	INCR8	01xxx: Attempt to use INCR8 AHB bursts
16	INCR16	1xxxx: Attempt to use INCR16 AHB bursts

- **ESMA: Endian Swap Mode Enable for Management Descriptor Accesses**

When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

- **ESPA: Endian Swap Mode Enable for Packet Data Accesses**

When set, selects swapped endianism for AHB transfers. When clear, selects little endian mode.

- **RXBMS: Receiver Packet Buffer Memory Size Select**

The default receive packet buffer size is 4 Kbytes. The table below shows how to configure this memory to FULL, HALF, QUARTER or EIGHTH of the default size.

Value	Name	Description
0	EIGHTH	4/8 Kbyte Memory Size
1	QUARTER	4/4 Kbytes Memory Size
2	HALF	4/2 Kbytes Memory Size
3	FULL	4 Kbytes Memory Size

- **TXPBMS: Transmitter Packet Buffer Memory Size Select**

Having this bit at zero halves the amount of memory used for the transmit packet buffer. This reduces the amount of memory used by the GMAC. It is important to set this bit to one if the full configured physical memory is available. The value in brackets below represents the size that would result for the default maximum configured memory size of 4 Kbytes.

0: Do not use top address bit (2 Kbytes).

1: Use full configured addressable space (4 Kbytes).

- **TXCOEN: Transmitter Checksum Generation Offload Enable**

Transmitter IP, TCP and UDP checksum generation offload enable. When set, the transmitter checksum generation engine is enabled to calculate and substitute checksums for transmit frames. When clear, frame data is unaffected.

- **DRBS: DMA Receive Buffer Size**

DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.

The value is defined in multiples of 64 bytes, thus a value of 0x01 corresponds to buffers of 64 bytes, 0x02 corresponds to 128 bytes etc.

For example:

– 0x02: 128 bytes

– 0x18: 1536 bytes (1 × max length frame/buffer)

– 0xA0: 10240 bytes (1 × 10K jumbo frame/buffer)

Note that this value should never be written as zero.

- **DDRP: DMA Discard Receive Packets**

When set, the GMAC DMA will automatically discard receive packets from the receiver packet buffer memory when no AHB resource is available.

When low, the received packets will remain to be stored in the SRAM based packet buffer until AHB buffer resource next becomes available.

A write to this bit is ignored if the DMA is not configured in the packet buffer full store and forward mode.

37.8.6 GMAC Transmit Status Register

Name: GMAC_TSR

Address: 0xF8008014

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	HRESP
7	6	5	4	3	2	1	0
–	–	TXCOMP	TFC	TXGO	RLE	COL	UBR

- **UBR: Used Bit Read**

Set when a transmit buffer descriptor is read with its used bit set. Writing a one clears this bit.

- **COL: Collision Occurred**

Set by the assertion of collision. Writing a one clears this bit. When operating in 10/100 mode, this status indicates either a collision or a late collision.

- **RLE: Retry Limit Exceeded**

Writing a one clears this bit.

- **TXGO: Transmit Go**

Transmit go, if high transmit is active. When using the DMA interface this bit represents the TXGO variable as specified in the transmit buffer description.

- **TFC: Transmit Frame Corruption Due to AHB Error**

Transmit frame corruption due to AHB error. Set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame (if the buffers run out during transmission of a frame then transmission stops, FCS shall be bad and GTXER asserted).

Also set in DMA packet buffer mode if single frame is too large for configured packet buffer memory size.

Writing a one clears this bit.

- **TXCOMP: Transmit Complete**

Set when a frame has been transmitted. Writing a one clears this bit.

- **HRESP: HRESP Not OK**

Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

37.8.7 GMAC Receive Buffer Queue Base Address Register

Name: GMAC_RBQB

Address: 0xF8008018

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR						–	–

This register holds the start address of the receive buffer queue (receive buffers descriptor list). The receive buffer queue base address must be initialized before receive is enabled through bit 2 of the Network Control Register. Once reception is enabled, any write to the Receive Buffer Queue Base Address Register is ignored. Reading this register returns the location of the descriptor currently being accessed. This value increments as buffers are used. Software should not use this register for determining where to remove received frames from the queue as it constantly changes as new frames are received. Software should instead work its way through the buffer descriptor queue checking the “used” bits.

In terms of AMBA AHB operation, the descriptors are read from memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are written to using two individual nonsequential accesses.

- **ADDR: Receive Buffer Queue Base Address**

Written with the address of the start of the receive queue.

37.8.8 GMAC Transmit Buffer Queue Base Address Register

Name: GMAC_TBQB

Address: 0xF800801C

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR						–	–

This register holds the start address of the transmit buffer queue (transmit buffers descriptor list). The Transmit Buffer Queue Base Address Register must be initialized before transmit is started through bit 9 of the Network Control Register. Once transmission has started, any write to the Transmit Buffer Queue Base Address Register is illegal and therefore ignored.

Note that due to clock boundary synchronization, it takes a maximum of four MCK cycles from the writing of the transmit start bit before the transmitter is active. Writing to the Transmit Buffer Queue Base Address Register during this time may produce unpredictable results.

Reading this register returns the location of the descriptor currently being accessed. Since the DMA handles two frames at once, this may not necessarily be pointing to the current frame being transmitted.

In terms of AMBA AHB operation, the descriptors are written to memory using a single 32-bit AHB access. The descriptors should be aligned at 32-bit boundaries and the descriptors are read from memory using two individual nonsequential accesses.

- **ADDR: Transmit Buffer Queue Base Address**

Written with the address of the start of the transmit queue.

37.8.9 GMAC Receive Status Register

Name: GMAC_RSR

Address: 0xF8008020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	HNO	RXOVR	REC	BNA

This register, when read, provides receive status details. Once read, individual bits may be cleared by writing a one to them. It is not possible to set a bit to 1 by writing to the register.

- **BNA: Buffer Not Available**

An attempt was made to get a new buffer and the pointer indicated that it was owned by the processor. The DMA will re-read the pointer each time an end of frame is received until a valid pointer is found. This bit is set following each descriptor read attempt that fails, even if consecutive pointers are unsuccessful and software has in the mean time cleared the status flag. Writing a one clears this bit.

- **REC: Frame Received**

One or more frames have been received and placed in memory. Writing a one clears this bit.

- **RXOVR: Receive Overrun**

This bit is set if the receive status was not taken at the end of the frame. This bit is also set if the packet buffer overflows. The buffer will be recovered if an overrun occurs. Writing a one clears this bit.

- **HNO: HRESP Not OK**

Set when the DMA block sees HRESP not OK. Writing a one clears this bit.

37.8.10 GMAC Interrupt Status Register

Name: GMAC_ISR

Address: 0xF8008024

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	WOL	–	SRI	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
–	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

This register indicates the source of the interrupt. In order that the bits of this register read 1, the corresponding interrupt source must be enabled in the mask register. If any bit is set in this register, the GMAC interrupt signal will be asserted in the system.

- **MFS: Management Frame Sent**

The PHY Maintenance Register has completed its operation. Cleared on read.

- **RCOMP: Receive Complete**

A frame has been stored in memory. Cleared on read.

- **RXUBR: RX Used Bit Read**

Set when a receive buffer descriptor is read with its used bit set. Cleared on read.

- **TXUBR: TX Used Bit Read**

Set when a transmit buffer descriptor is read with its used bit set. Cleared on read.

- **TUR: Transmit Underrun**

This interrupt is set if the transmitter was forced to terminate a frame that it has already began transmitting due to further data being unavailable.

This interrupt is set if a transmitter status write back has not completed when another status write back is attempted.

This interrupt is also set when the transmit DMA has written the SOP data into the FIFO and either the AHB bus was not granted in time for further data, or because an AHB not OK response was returned, or because the used bit was read.

- **RLEX: Retry Limit Exceeded**

Transmit error. Cleared on read.

- **TFC: Transmit Frame Corruption Due to AHB Error**

Transmit frame corruption due to AHB error. Set if an error occurs while midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

- **TCOMP: Transmit Complete**

Set when a frame has been transmitted. Cleared on read.

- **ROVR: Receive Overrun**

Set when the receive overrun status bit is set. Cleared on read.

- **HRESP: HRESP Not OK**

Set when the DMA block sees HRESP not OK. Cleared on read.

- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**

Indicates a valid pause has been received that has a non-zero pause quantum field. Cleared on read.

- **PTZ: Pause Time Zero**

Set when either the Pause Time register at address 0x38 decrements to zero, or when a valid pause frame is received with a zero pause quantum field. Cleared on read.

- **PFTR: Pause Frame Transmitted**

Indicates a pause frame has been successfully transmitted after being initiated from the Network Control register. Cleared on read.

- **DRQFR: PTP Delay Request Frame Received**

Indicates a PTP delay_req frame has been received. Cleared on read.

- **SFR: PTP Sync Frame Received**

Indicates a PTP sync frame has been received. Cleared on read.

- **DRQFT: PTP Delay Request Frame Transmitted**

Indicates a PTP delay_req frame has been transmitted. Cleared on read.

- **SFT: PTP Sync Frame Transmitted**

Indicates a PTP sync frame has been transmitted. Cleared on read.

- **PDRQFR: PDelay Request Frame Received**

Indicates a PTP pdelay_req frame has been received. Cleared on read.

- **PDRSFR: PDelay Response Frame Received**

Indicates a PTP pdelay_resp frame has been received. Cleared on read.

- **PDRQFT: PDelay Request Frame Transmitted**

Indicates a PTP pdelay_req frame has been transmitted. Cleared on read.

- **PDRSFT: PDelay Response Frame Transmitted**

Indicates a PTP pdelay_resp frame has been transmitted. Cleared on read.

- **SRI: TSU Seconds Register Increment**

Indicates the register has incremented. Cleared on read.

- **WOL: Wake On LAN**

WOL interrupt. Indicates a WOL event has been received.

37.8.11 GMAC Interrupt Enable Register

Name: GMAC_IER

Address: 0xF8008028

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	WOL	–	SRI	PDRSFT	PDRQFT

23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–

15	14	13	12	11	10	9	8
EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–

7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

This register is write-only and when read will return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **MFS: Management Frame Sent**
- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **TXUBR: TX Used Bit Read**
- **TUR: Transmit Underrun**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**
- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**
- **PTZ: Pause Time Zero**
- **PFTR: Pause Frame Transmitted**
- **EXINT: External Interrupt**
- **DRQFR: PTP Delay Request Frame Received**

- **SFR: PTP Sync Frame Received**
- **DRQFT: PTP Delay Request Frame Transmitted**
- **SFT: PTP Sync Frame Transmitted**
- **PDRQFR: PDelay Request Frame Received**
- **PDRSFR: PDelay Response Frame Received**
- **PDRQFT: PDelay Request Frame Transmitted**
- **PDRSFT: PDelay Response Frame Transmitted**
- **SRI: TSU Seconds Register Increment**
- **WOL: Wake On LAN**

37.8.12 GMAC Interrupt Disable Register

Name: GMAC_IDR

Address: 0xF800802C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	WOL	–	SRI	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

This register is write-only and when read will return zero.

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **MFS: Management Frame Sent**
- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **TXUBR: TX Used Bit Read**
- **TUR: Transmit Underrun**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**
- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**
- **PTZ: Pause Time Zero**
- **PFTR: Pause Frame Transmitted**
- **EXINT: External Interrupt**
- **DRQFR: PTP Delay Request Frame Received**

- **SFR: PTP Sync Frame Received**
- **DRQFT: PTP Delay Request Frame Transmitted**
- **SFT: PTP Sync Frame Transmitted**
- **PDRQFR: PDelay Request Frame Received**
- **PDRSFR: PDelay Response Frame Received**
- **PDRQFT: PDelay Request Frame Transmitted**
- **PDRSFT: PDelay Response Frame Transmitted**
- **SRI: TSU Seconds Register Increment**
- **WOL: Wake On LAN**

37.8.13 GMAC Interrupt Mask Register

Name: GMAC_IMR

Address: 0xF8008030

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	PDRSFT	PDRQFT
23	22	21	20	19	18	17	16
PDRSFR	PDRQFR	SFT	DRQFT	SFR	DRQFR	–	–
15	14	13	12	11	10	9	8
EXINT	PFTR	PTZ	PFNZ	HRESP	ROVR	–	–
7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	TUR	TXUBR	RXUBR	RCOMP	MFS

The Interrupt Mask Register is a read-only register indicating which interrupts are masked. All bits are set at reset and can be reset individually by writing to the Interrupt Enable Register or set individually by writing to the Interrupt Disable Register. Having separate address locations for enable and disable saves the need for performing a read modify write when updating the Interrupt Mask Register.

For test purposes there is a write-only function to this register that allows the bits in the Interrupt Status Register to be set or cleared, regardless of the state of the mask register. A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register when read:

0: The corresponding interrupt is enabled.

1: The corresponding interrupt is not enabled.

- **MFS: Management Frame Sent**
- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **TXUBR: TX Used Bit Read**
- **TUR: Transmit Underrun**
- **RLEX: Retry Limit Exceeded**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**
- **PFNZ: Pause Frame with Non-zero Pause Quantum Received**
- **PTZ: Pause Time Zero**

- **PFTR: Pause Frame Transmitted**
- **EXINT: External Interrupt**
- **DRQFR: PTP Delay Request Frame Received**
- **SFR: PTP Sync Frame Received**
- **DRQFT: PTP Delay Request Frame Transmitted**
- **SFT: PTP Sync Frame Transmitted**
- **PDRQFR: PDelay Request Frame Received**
- **PDRSFR: PDelay Response Frame Received**
- **PDRQFT: PDelay Request Frame Transmitted**
- **PDRSFT: PDelay Response Frame Transmitted**

37.8.14 GMAC PHY Maintenance Register

Name: GMAC_MAN

Address: 0xF8008034

Access: Read/Write

31	30	29	28	27	26	25	24
WZO	CLTTO	OP		PHYA			
23	22	21	20	19	18	17	16
PHYA	REGA					WTN	
15	14	13	12	11	10	9	8
DATA							
7	6	5	4	3	2	1	0
DATA							

The PHY Maintenance Register is implemented as a shift register. Writing to the register starts a shift operation which is signalled as complete when bit 2 is set in the Network Status Register. It takes about 2000 MCK cycles to complete, when MDC is set for MCK divide by 32 in the Network Configuration Register. An interrupt is generated upon completion.

During this time, the MSB of the register is output on the MDIO pin and the LSB updated from the MDIO pin with each MDC cycle. This causes transmission of a PHY management frame on MDIO. *See Section 22.2.4.5 of the IEEE 802.3 standard*.

Reading during the shift operation returns the current contents of the shift register. At the end of management operation, the bits will have shifted back to their original locations. For a read operation, the data bits are updated with data read from the PHY. It is important to write the correct values to the register to ensure a valid PHY management frame is produced.

The MDIO interface can read IEEE 802.3 clause 45 PHYs as well as clause 22 PHYs. To read clause 45 PHYs, bit 30 should be written with a 0 rather than a 1. To write clause 45 PHYs, bits 31:28 should be written as 0x0001. See Table 37-18.

Table 37-18. Clause 22/Clause 45 PHYs Read/Write Access Configuration (GMAC_MAN Bits 31:28)

PHY	Access	Bit Value			
		WZO	CLTTO	OP[1]	OP[0]
Clause 22	Read	0	1	1	0
	Write	0	1	0	1
Clause 45	Read	0	0	1	1
	Write	0	0	0	1
	Read + Address	0	0	1	0

For a description of MDC generation, see Section 37.8.2 ”GMAC Network Configuration Register”.

- **DATA: PHY Data**

For a write operation this field is written with the data to be written to the PHY. After a read operation this field contains the data read from the PHY.

- **WTN: Write Ten**

Must be written to 10.

- **REGA: Register Address**

Specifies the register in the PHY to access.

- **PHYA: PHY Address**

- **OP: Operation**

01: Write

10: Read

- **CLTTO: Clause 22 Operation**

0: Clause 45 operation

1: Clause 22 operation

- **WZO: Write ZERO**

Must be written with 0.

37.8.15 GMAC Receive Pause Quantum Register

Name: GMAC_RPQ

Address: 0xF8008038

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RPQ							

7	6	5	4	3	2	1	0
RPQ							

- **RPQ: Received Pause Quantum**

Stores the current value of the Receive Pause Quantum Register which is decremented every 512 bit times.

37.8.16 GMAC Transmit Pause Quantum Register

Name: GMAC_TPQ

Address: 0xF800803C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TPQ							

7	6	5	4	3	2	1	0
TPQ							

- **TPQ: Transmit Pause Quantum**

Written with the pause quantum value for pause frame transmission.

37.8.17 GMAC TX Partial Store and Forward Register

Name: GMAC_TPSF

Address: 0xF8008040

Access: Read/Write

31	30	29	28	27	26	25	24
ENTXP	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	TPB1ADR			
7	6	5	4	3	2	1	0
TPB1ADR							

- **TPB1ADR: Transmit Partial Store and Forward Address**

Watermark value. Reset = 1.

- **ENTXP: Enable TX Partial Store and Forward Operation**

37.8.18 GMAC RX Partial Store and Forward Register

Name: GMAC_RPSF

Address: 0xF8008044

Access: Read/Write

31	30	29	28	27	26	25	24
ENRXP	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RPB1ADR			
7	6	5	4	3	2	1	0
RPB1ADR							

- **RPB1ADR: Receive Partial Store and Forward Address**

Watermark value. Reset = 1.

- **ENRXP: Enable RX Partial Store and Forward Operation**

37.8.19 GMAC RX Jumbo Frame Max Length Register

Name: GMAC_RJFML

Address: 0xF8008048

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	FML					

7	6	5	4	3	2	1	0
FML							

- **FML: Frame Max Length**

Rx jumbo frame maximum length.

37.8.20 GMAC Hash Register Bottom

Name: GMAC_HRB

Address: 0xF8008080

Access: Read-only

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the Network Configuration Register (Section 37.8.2 ”GMAC Network Configuration Register”) enable the reception of hash matched frames. See Section 37.6.9 ”Hash Addressing”.

- **ADDR: Hash Address**

The first 32 bits of the Hash Address Register.

37.8.21 GMAC Hash Register Top

Name: GMAC_HRT

Address: 0xF8008084

Access: Read-only

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The unicast hash enable (UNIHEN) and the multicast hash enable (MITIHEN) bits in the GMAC Network Configuration Register enable the reception of hash matched frames. See Section 37.6.9 ”Hash Addressing”.

- **ADDR: Hash Address**

Bits 63 to 32 of the Hash Address Register.

37.8.22 GMAC Specific Address 1 Bottom Register

Name: GMAC_SAB1

Address: 0xF8008088

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 1**

Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

37.8.23 GMAC Specific Address 1 Top Register

Name: GMAC_SAT1

Address: 0xF800808C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 1**

The most significant bits of the destination address, that is, bits 47:32.

37.8.24 GMAC Specific Address 2 Bottom Register

Name: GMAC_SAB2

Address: 0xF8008090

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 2**

Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

37.8.25 GMAC Specific Address 2 Top Register

Name: GMAC_SAT2

Address: 0xF8008094

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 2**

The most significant bits of the destination address, that is, bits 47:32.

37.8.26 GMAC Specific Address 3 Bottom Register

Name: GMAC_SAB3

Address: 0xF8008098

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 3**

Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

37.8.27 GMAC Specific Address 3 Top Register

Name: GMAC_SAT3

Address: 0xF800809C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 3**

The most significant bits of the destination address, that is, bits 47:32.

37.8.28 GMAC Specific Address 4 Bottom Register

Name: GMAC_SAB4

Address: 0xF80080A0

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 4**

Least significant 32 bits of the destination address, that is, bits 31:0. Bit zero indicates whether the address is multicast or unicast and corresponds to the least significant bit of the first byte received.

37.8.29 GMAC Specific Address 4 Top Register

Name: GMAC_SAT4

Address: 0xF80080A4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

The addresses stored in the Specific Address Registers are deactivated at reset or when their corresponding Specific Address Register Bottom is written. They are activated when Specific Address Register Top is written.

- **ADDR: Specific Address 4**

The most significant bits of the destination address, that is, bits 47:32.

37.8.30 GMAC Type ID Match 1 Register

Name: GMAC_TIDM1

Address: 0xF80080A8

Access: Read/Write

31	30	29	28	27	26	25	24
ENID1	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TID							

7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 1**

For use in comparisons with received frames type ID/length frames.

- **ENID1: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

37.8.31 GMAC Type ID Match 2 Register

Name: GMAC_TIDM2

Address: 0xF80080AC

Access: Read/Write

31	30	29	28	27	26	25	24
ENID2	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TID							
7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 2**

For use in comparisons with received frames type ID/length frames.

- **ENID2: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

37.8.32 GMAC Type ID Match 3 Register

Name: GMAC_TIDM3

Address: 0xF80080B0

Access: Read/Write

31	30	29	28	27	26	25	24
ENID3	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TID							

7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 3**

For use in comparisons with received frames type ID/length frames.

- **ENID3: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

37.8.33 GMAC Type ID Match 4 Register

Name: GMAC_TIDM4

Address: 0xF80080B4

Access: Read/Write

31	30	29	28	27	26	25	24
ENID4	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TID							

7	6	5	4	3	2	1	0
TID							

- **TID: Type ID Match 4**

For use in comparisons with received frames type ID/length frames.

- **ENID4: Enable Copying of TID Matched Frames**

0: TID is not part of the comparison match.

1: TID is processed for the comparison match.

37.8.34 GMAC Wake on LAN Register

Name: GMAC_WOL

Address: 0xF80080B8

Access: Read/Write

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–				MTI	SA1	ARP	MAG

15	14	13	12	11	10	9	8
IP							

7	6	5	4	3	2	1	0
IP							

- **IP: ARP Request IP Address**

Wake on LAN ARP request IP address. Written to define the least significant 16 bits of the target IP address that is matched to generate a Wake on LAN event. A value of zero will not generate an event, even if this is matched by the received frame.

- **MAG: Magic Packet Event Enable**

Wake on LAN magic packet event enable.

- **ARP: ARP Request Event Enable**

Wake on LAN ARP request event enable.

- **SA1: Specific Address Register 1 Event Enable**

Wake on LAN Specific Address Register 1 event enable.

- **MTI: Multicast Hash Event Enable**

Wake on LAN multicast hash event enable.

37.8.35 GMAC IPG Stretch Register

Name: GMAC_IPGS

Address: 0xF80080BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
FL							

7	6	5	4	3	2	1	0
FL							

- **FL: Frame Length**

Bits 7:0 are multiplied with the previously transmitted frame length (including preamble). Bits 15:8 +1 divide the frame length. If the resulting number is greater than 96 and bit 28 is set in the Network Configuration Register then the resulting number is used for the transmit inter-packet-gap. 1 is added to bits 15:8 to prevent a divide by zero. See Section 37.6.4 ”MAC Transmit Block”.

37.8.36 GMAC Stacked VLAN Register

Name: GMAC_SVLAN

Address: 0xF80080C0

Access: Read/Write

31	30	29	28	27	26	25	24
ESVLAN	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
VLAN_TYPE							

7	6	5	4	3	2	1	0
VLAN_TYPE							

- **VLAN_TYPE: User Defined VLAN_TYPE Field**

User defined VLAN_TYPE field. When Stacked VLAN is enabled, the first VLAN tag in a received frame will only be accepted if the VLAN type field is equal to this user defined VLAN_TYPE, OR equal to the standard VLAN type (0x8100). Note that the second VLAN tag of a Stacked VLAN packet will only be matched correctly if its VLAN_TYPE field equals 0x8100.

- **ESVLAN: Enable Stacked VLAN Processing Mode**

0: Disable the stacked VLAN processing mode

1: Enable the stacked VLAN processing mode

37.8.37 GMAC Transmit PFC Pause Register

Name: GMAC_TPFCP

Address: 0xF80080C4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PQ							

7	6	5	4	3	2	1	0
PEV							

- **PEV: Priority Enable Vector**

If bit 17 of the Network Control Register is written with a one then the priority enable vector of the PFC priority based pause frame will be set equal to the value stored in this register [7:0].

- **PQ: Pause Quantum**

If bit 17 of the Network Control Register is written with a one then for each entry equal to zero in the Transmit PFC Pause Register[15:8], the PFC pause frame's pause quantum field associated with that entry will be taken from the Transmit Pause Quantum Register. For each entry equal to one in the Transmit PFC Pause Register [15:8], the pause quantum associated with that entry will be zero.

37.8.38 GMAC Specific Address 1 Mask Bottom Register

Name: GMAC_SAMB1

Address: 0xF80080C8

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: Specific Address 1 Mask**

Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

37.8.39 GMAC Specific Address Mask 1 Top Register

Name: GMAC_SAMT1

Address: 0xF80080CC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: Specific Address 1 Mask**

Setting a bit to one masks the corresponding bit in the Specific Address 1 Register.

37.8.40 GMAC 1588 Timer Nanosecond Comparison Register

Name: GMAC_NSC

Address: 0xF80080DC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	NANOSEC					

15	14	13	12	11	10	9	8
NANOSEC							

7	6	5	4	3	2	1	0
NANOSEC							

- **NANOSEC: 1588 Timer Nanosecond Comparison Value**

Value is compared to the bits [45:24] of the TSU timer count value (upper 22 bits of nanosecond value).

37.8.41 GMAC 1588 Timer Second Comparison Low Register

Name: GMAC_SCL

Address: 0xF80080E0

Access: Read/Write

31	30	29	28	27	26	25	24
SEC							

23	22	21	20	19	18	17	16
SEC							

15	14	13	12	11	10	9	8
SEC							

7	6	5	4	3	2	1	0
SEC							

- **SEC: 1588 Timer Second Comparison Value**

Value is compared to seconds value bits [31:0] of the TSU timer count value.

37.8.42 GMAC 1588 Timer Second Comparison High Register

Name: GMAC_SCH

Address: 0xF80080E4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
SEC							

7	6	5	4	3	2	1	0
SEC							

- **SEC: 1588 Timer Second Comparison Value**

Value is compared to the top 16 bits (most significant 16 bits [47:32] of seconds value) of the TSU timer count value.

37.8.43 GMAC PTP Event Frame Transmitted Seconds High Register

Name: GMAC_EFTSH

Address: 0xF80080E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.44 GMAC PTP Event Frame Received Seconds High Register

Name: GMAC_EFRSH

Address: 0xF80080EC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.45 GMAC PTP Peer Event Frame Transmitted Seconds High Register

Name: GMAC_PEFTSH

Address: 0xF80080F0

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.46 GMAC PTP Peer Event Frame Received Seconds High Register

Name: GMAC_PEFRSH

Address: 0xF80080F4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 timer seconds register held when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.47 GMAC Octets Transmitted Low Register

Name: GMAC_OTLO

Address: 0xF8008100

Access: Read-only

31	30	29	28	27	26	25	24
TXO							

23	22	21	20	19	18	17	16
TXO							

15	14	13	12	11	10	9	8
TXO							

7	6	5	4	3	2	1	0
TXO							

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

- **TXO: Transmitted Octets**

Transmitted octets in frame without errors [31:0]. The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

37.8.48 GMAC Octets Transmitted High Register

Name: GMAC_OTHI

Address: 0xF8008104

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TXO							

7	6	5	4	3	2	1	0
TXO							

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

- **TXO: Transmitted Octets**

Transmitted octets in frame without errors [47:32]. The number of octets transmitted in valid frames of any type. This counter is 48-bits, and is read through two registers. This count does not include octets from automatically generated pause frames.

37.8.49 GMAC Frames Transmitted Register

Name: GMAC_FT

Address: 0xF8008108

Access: Read-only

31	30	29	28	27	26	25	24
FTX							

23	22	21	20	19	18	17	16
FTX							

15	14	13	12	11	10	9	8
FTX							

7	6	5	4	3	2	1	0
FTX							

- **FTX: Frames Transmitted without Error**

Frames transmitted without error. This register counts the number of frames successfully transmitted, i.e., no underrun and not too many retries. Excludes pause frames.

37.8.50 GMAC Broadcast Frames Transmitted Register

Name: GMAC_BCFT

Address: 0xF800810C

Access: Read-only

31	30	29	28	27	26	25	24
BFTX							

23	22	21	20	19	18	17	16
BFTX							

15	14	13	12	11	10	9	8
BFTX							

7	6	5	4	3	2	1	0
BFTX							

- **BFTX: Broadcast Frames Transmitted without Error**

Broadcast frames transmitted without error. This register counts the number of broadcast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

37.8.51 GMAC Multicast Frames Transmitted Register

Name: GMAC_MFT

Address: 0xF8008110

Access: Read-only

31	30	29	28	27	26	25	24
MFTX							

23	22	21	20	19	18	17	16
MFTX							

15	14	13	12	11	10	9	8
MFTX							

7	6	5	4	3	2	1	0
MFTX							

- **MFTX: Multicast Frames Transmitted without Error**

This register counts the number of multicast frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

37.8.52 GMAC Pause Frames Transmitted Register

Name: GMAC_PFT

Address: 0xF8008114

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PFTX							

7	6	5	4	3	2	1	0
PFTX							

- **PFTX: Pause Frames Transmitted Register**

This register counts the number of pause frames transmitted. Only pause frames triggered by the register interface or through the external pause pins are counted as pause frames. Pause frames received through the FIFO interface are counted in the frames transmitted counter.

37.8.53 GMAC 64 Byte Frames Transmitted Register

Name: GMAC_BFT64

Address: 0xF8008118

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 64 Byte Frames Transmitted without Error**

This register counts the number of 64 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

37.8.54 GMAC 65 to 127 Byte Frames Transmitted Register

Name: GMAC_TBFT127

Address: 0xF800811C

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 65 to 127 Byte Frames Transmitted without Error**

This register counts the number of 65 to 127 byte frames successfully transmitted without error, i.e., no underrun and not too many retries. Excludes pause frames.

37.8.55 GMAC 128 to 255 Byte Frames Transmitted Register

Name: GMAC_TBFT255

Address: 0xF8008120

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 128 to 255 Byte Frames Transmitted without Error**

This register counts the number of 128 to 255 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

37.8.56 GMAC 256 to 511 Byte Frames Transmitted Register

Name: GMAC_TBFT511

Address: 0xF8008124

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 256 to 511 Byte Frames Transmitted without Error**

This register counts the number of 256 to 511 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

37.8.57 GMAC 512 to 1023 Byte Frames Transmitted Register

Name: GMAC_TBFT1023

Address: 0xF8008128

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 512 to 1023 Byte Frames Transmitted without Error**

This register counts the number of 512 to 1023 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

37.8.58 GMAC 1024 to 1518 Byte Frames Transmitted Register

Name: GMAC_TBFT1518

Address: 0xF800812C

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: 1024 to 1518 Byte Frames Transmitted without Error**

This register counts the number of 1024 to 1518 byte frames successfully transmitted without error, i.e., no underrun and not too many retries.

37.8.59 GMAC Greater Than 1518 Byte Frames Transmitted Register

Name: GMAC_GTBFT1518

Address: 0xF8008130

Access: Read-only

31	30	29	28	27	26	25	24
NFTX							

23	22	21	20	19	18	17	16
NFTX							

15	14	13	12	11	10	9	8
NFTX							

7	6	5	4	3	2	1	0
NFTX							

- **NFTX: Greater than 1518 Byte Frames Transmitted without Error**

This register counts the number of 1518 or above byte frames successfully transmitted without error i.e., no underrun and not too many retries.

37.8.60 GMAC Transmit Underruns Register

Name: GMAC_TUR

Address: 0xF8008134

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	TXUNR	

7	6	5	4	3	2	1	0
TXUNR							

- **TXUNR: Transmit Underruns**

This register counts the number of frames not transmitted due to a transmit underrun. If this register is incremented then no other statistics register is incremented.

37.8.61 GMAC Single Collision Frames Register

Name: GMAC_SCF

Address: 0xF8008138

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	SCOL	

15	14	13	12	11	10	9	8
SCOL							

7	6	5	4	3	2	1	0
SCOL							

- **SCOL: Single Collision**

This register counts the number of frames experiencing a single collision before being successfully transmitted i.e., no underrun.

37.8.62 GMAC Multiple Collision Frames Register

Name: GMAC_MCF

Address: 0xF800813C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	MCOL	

15	14	13	12	11	10	9	8
MCOL							

7	6	5	4	3	2	1	0
MCOL							

- **MCOL: Multiple Collision**

This register counts the number of frames experiencing between two and fifteen collisions prior to being successfully transmitted, i.e., no underrun and not too many retries.

37.8.63 GMAC Excessive Collisions Register

Name: GMAC_EC

Address: 0xF8008140

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	XCOL	
7	6	5	4	3	2	1	0
XCOL							

- **XCOL: Excessive Collisions**

This register counts the number of frames that failed to be transmitted because they experienced 16 collisions.

37.8.64 GMAC Late Collisions Register

Name: GMAC_LC

Address: 0xF8008144

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	LCOL	

7	6	5	4	3	2	1	0
LCOL							

- **LCOL: Late Collisions**

This register counts the number of late collisions occurring after the slot time (512 bits) has expired. In 10/100 mode, late collisions are counted twice i.e., both as a collision and a late collision.

37.8.65 GMAC Deferred Transmission Frames Register

Name: GMAC_DTF

Address: 0xF8008148

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	DEFT	

15	14	13	12	11	10	9	8
DEFT							

7	6	5	4	3	2	1	0
DEFT							

- **DEFT: Deferred Transmission**

This register counts the number of frames experiencing deferral due to carrier sense being active on their first attempt at transmission. Frames involved in any collision are not counted nor are frames that experienced a transmit underrun.

37.8.66 GMAC Carrier Sense Errors Register

Name: GMAC_CSE

Address: 0xF800814C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	CSR	

7	6	5	4	3	2	1	0
CSR							

- **CSR: Carrier Sense Error**

This register counts the number of frames transmitted where carrier sense was not seen during transmission or where carrier sense was deasserted after being asserted in a transmit frame without collision (no underrun). Only incremented in half duplex mode. The only effect of a carrier sense error is to increment this register. The behavior of the other statistics registers is unaffected by the detection of a carrier sense error.

37.8.67 GMAC Octets Received Low Register

Name: GMAC_ORLO

Address: 0xF8008150

Access: Read-only

31	30	29	28	27	26	25	24
RXO							

23	22	21	20	19	18	17	16
RXO							

15	14	13	12	11	10	9	8
RXO							

7	6	5	4	3	2	1	0
RXO							

When reading the Octets Transmitted and Octets Received Registers, bits [31:0] should be read prior to bits [47:32] to ensure reliable operation.

- **RXO: Received Octets**

Received octets in frame without errors [31:0]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.68 GMAC Octets Received High Register

Name: GMAC_ORHI

Address: 0xF8008154

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RXO							

7	6	5	4	3	2	1	0
RXO							

When reading the Octets Transmitted and Octets Received Registers, bits 31:0 should be read prior to bits 47:32 to ensure reliable operation.

- **RXO: Received Octets**

Received octets in frame without errors [47:32]. The number of octets received in valid frames of any type. This counter is 48-bits and is read through two registers. This count does not include octets from pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.69 GMAC Frames Received Register

Name: GMAC_FR

Address: 0xF8008158

Access: Read-only

31	30	29	28	27	26	25	24
FRX							

23	22	21	20	19	18	17	16
FRX							

15	14	13	12	11	10	9	8
FRX							

7	6	5	4	3	2	1	0
FRX							

- **FRX: Frames Received without Error**

Frames received without error. This register counts the number of frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.70 GMAC Broadcast Frames Received Register

Name: GMAC_BCFR

Address: 0xF800815C

Access: Read-only

31	30	29	28	27	26	25	24
BFRX							

23	22	21	20	19	18	17	16
BFRX							

15	14	13	12	11	10	9	8
BFRX							

7	6	5	4	3	2	1	0
BFRX							

- **BFRX: Broadcast Frames Received without Error**

Broadcast frames received without error. This register counts the number of broadcast frames successfully received. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.71 GMAC Multicast Frames Received Register

Name: GMAC_MFR

Address: 0xF8008160

Access: Read-only

31	30	29	28	27	26	25	24
MFRX							

23	22	21	20	19	18	17	16
MFRX							

15	14	13	12	11	10	9	8
MFRX							

7	6	5	4	3	2	1	0
MFRX							

- **MFRX: Multicast Frames Received without Error**

This register counts the number of multicast frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.72 GMAC Pause Frames Received Register

Name: GMAC_PFR

Address: 0xF8008164

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
PFRX							

7	6	5	4	3	2	1	0
PFRX							

- **PFRX: Pause Frames Received Register**

This register counts the number of pause frames received without error.

37.8.73 GMAC 64 Byte Frames Received Register

Name: GMAC_BFR64

Address: 0xF8008168

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 64 Byte Frames Received without Error**

This register counts the number of 64 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.74 GMAC 65 to 127 Byte Frames Received Register

Name: GMAC_TBFR127

Address: 0xF800816C

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 65 to 127 Byte Frames Received without Error**

This register counts the number of 65 to 127 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.75 GMAC 128 to 255 Byte Frames Received Register

Name: GMAC_TBFR255

Address: 0xF8008170

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 128 to 255 Byte Frames Received without Error**

This register counts the number of 128 to 255 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.76 GMAC 256 to 511 Byte Frames Received Register

Name: GMAC_TBFR511

Address: 0xF8008174

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 256 to 511 Byte Frames Received without Error**

This register counts the number of 256 to 511 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.77 GMAC 512 to 1023 Byte Frames Received Register

Name: GMAC_TBFR1023

Address: 0xF8008178

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 512 to 1023 Byte Frames Received without Error**

This register counts the number of 512 to 1023 byte frames successfully received without error. Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory.

37.8.78 GMAC 1024 to 1518 Byte Frames Received Register

Name: GMAC_TBFR1518

Address: 0xF800817C

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 1024 to 1518 Byte Frames Received without Error**

This register counts the number of 1024 to 1518 byte frames successfully received without error, i.e., no underrun and not too many retries.

37.8.79 GMAC 1519 to Maximum Byte Frames Received Register

Name: GMAC_TMXBFR

Address: 0xF8008180

Access: Read-only

31	30	29	28	27	26	25	24
NFRX							

23	22	21	20	19	18	17	16
NFRX							

15	14	13	12	11	10	9	8
NFRX							

7	6	5	4	3	2	1	0
NFRX							

- **NFRX: 1519 to Maximum Byte Frames Received without Error**

This register counts the number of 1519 byte or above frames successfully received without error. Maximum frame size is determined by the Network Configuration Register bit 8 (1536 maximum frame size) or bit 3 (jumbo frame size). Excludes pause frames, and is only incremented if the frame is successfully filtered and copied to memory. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.80 GMAC Undersized Frames Received Register

Name: GMAC_UFR

Address: 0xF8008184

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	UFRX	

7	6	5	4	3	2	1	0
UFRX							

- **UFRX: Undersize Frames Received**

This register counts the number of frames received less than 64 bytes in length (10/100 mode, full duplex) that do not have either a CRC error or an alignment error.

37.8.81 GMAC Oversized Frames Received Register

Name: GMAC_OFR

Address: 0xF8008188

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	OFRX	
7	6	5	4	3	2	1	0
OFRX							

- **OFRX: Oversized Frames Received**

This register counts the number of frames received exceeding 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) in length but do not have either a CRC error, an alignment error nor a receive symbol error. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.82 GMAC Jabbers Received Register

Name: GMAC_JR

Address: 0xF800818C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	JRX	

7	6	5	4	3	2	1	0
JRX							

- **JRX: Jabbers Received**

The register counts the number of frames received exceeding 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register) and have either a CRC error, an alignment error or a receive symbol error. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.83 GMAC Frame Check Sequence Errors Register

Name: GMAC_FCSE

Address: 0xF8008190

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	FCKR	

7	6	5	4	3	2	1	0
FCKR							

- **FCKR: Frame Check Sequence Errors**

The register counts frames that are an integral number of bytes, have bad CRC and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and has an integral number of bytes.

This register is incremented for a frame with bad FCS, regardless of whether it is copied to memory due to ignore FCS mode being enabled in bit 26 of the Network Configuration Register. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.84 GMAC Length Field Frame Errors Register

Name: GMAC_LFFE

Address: 0xF8008194

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	LFER	

7	6	5	4	3	2	1	0
LFER							

- **LFER: Length Field Frame Errors**

This register counts the number of frames received that have a measured length shorter than that extracted from the length field (bytes 13 and 14). This condition is only counted if the value of the length field is less than 0x0600, the frame is not of excessive length and checking is enabled through bit 16 of the Network Configuration Register. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.85 GMAC Receive Symbol Errors Register

Name: GMAC_RSE

Address: 0xF8008198

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXSE	

7	6	5	4	3	2	1	0
RXSE							

- **RXSE: Receive Symbol Errors**

This register counts the number of frames that had GRXER asserted during reception. For 10/100 mode symbol errors are counted regardless of frame length checks. Receive symbol errors will also be counted as an FCS or alignment error if the frame is between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register). If the frame is larger it will be recorded as a jabber error. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.86 GMAC Alignment Errors Register

Name: GMAC_AE

Address: 0xF800819C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	AER	

7	6	5	4	3	2	1	0
AER							

- **AER: Alignment Errors**

This register counts the frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.87 GMAC Receive Resource Errors Register

Name: GMAC_RRE

Address: 0xF80081A0

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	RXRER	

15	14	13	12	11	10	9	8
RXRER							

7	6	5	4	3	2	1	0
RXRER							

- **RXRER: Receive Resource Errors**

This register counts frames that are not an integral number of bytes long and have bad CRC when their length is truncated to an integral number of bytes and are between 64 and 1518 bytes in length (1536 if bit 8 is set in Network Configuration Register). This register is also incremented if a symbol error is detected and the frame is of valid length and does not have an integral number of bytes. See Section 37.8.2 ”GMAC Network Configuration Register”.

37.8.88 GMAC Receive Overruns Register

Name: GMAC_ROE

Address: 0xF80081A4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXOVR	

7	6	5	4	3	2	1	0
RXOVR							

- **RXOVR: Receive Overruns**

This register counts the number of frames that are address recognized but were not copied to memory due to a receive overrun.

37.8.89 GMAC IP Header Checksum Errors Register

Name: GMAC_IHCE

Address: 0xF80081A8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
HCKER							

- **HCKER: IP Header Checksum Errors**

This register counts the number of frames discarded due to an incorrect IP header checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

37.8.90 GMAC TCP Checksum Errors Register

Name: GMAC_TCE

Address: 0xF80081AC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
TCKER							

- **TCKER: TCP Checksum Errors**

This register counts the number of frames discarded due to an incorrect TCP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

37.8.91 GMAC UDP Checksum Errors Register

Name: GMAC_UCE

Address: 0xF80081B0

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
UCKER							

- **UCKER: UDP Checksum Errors**

This register counts the number of frames discarded due to an incorrect UDP checksum, but are between 64 and 1518 bytes (1536 bytes if bit 8 is set in the Network Configuration Register) and do not have a CRC error, an alignment error, nor a symbol error.

37.8.92 GMAC 1588 Timer Increment Sub-nanoseconds Register

Name: GMAC_TISUBN

Address: 0xF80081BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
LSBTIR							

7	6	5	4	3	2	1	0
LSBTIR							

- **LSBTIR: Lower Significant Bits of Timer Increment Register**

Lower significant bits of Timer Increment Register[15:0] giving a 24-bit timer_increment counter. These bits are the sub-ns value which the 1588 timer will be incremented each clock cycle. Bit n = $2^{(n-16)}$ nsec giving a resolution of approximately $15.2E^{-15}$ sec.

37.8.93 GMAC 1588 Timer Seconds High Register

Name: GMAC_TSH

Address: 0xF80081C0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TCS							

7	6	5	4	3	2	1	0
TCS							

- **TCS: Timer Count in Seconds**

This register is writable. It increments by one when the 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

37.8.94 GMAC 1588 Timer Seconds Low Register

Name: GMAC_TSL

Address: 0xF80081D0

Access: Read/Write

31	30	29	28	27	26	25	24
TCS							

23	22	21	20	19	18	17	16
TCS							

15	14	13	12	11	10	9	8
TCS							

7	6	5	4	3	2	1	0
TCS							

- **TCS: Timer Count in Seconds**

This register is writable. It increments by one when the 1588 nanoseconds counter counts to one second. It may also be incremented when the Timer Adjust Register is written.

37.8.95 GMAC 1588 Timer Nanoseconds Register

Name: GMAC_TN

Address: 0xF80081D4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	TNS					

23	22	21	20	19	18	17	16
TNS							

15	14	13	12	11	10	9	8
TNS							

7	6	5	4	3	2	1	0
TNS							

- **TNS: Timer Count in Nanoseconds**

This register is writable. It can also be adjusted by writes to the 1588 Timer Adjust Register. It increments by the value of the 1588 Timer Increment Register each clock cycle.

37.8.96 GMAC 1588 Timer Adjust Register

Name: GMAC_TA

Address: 0xF80081D8

Access: Write-only

31	30	29	28	27	26	25	24
ADJ	–	ITDT					

23	22	21	20	19	18	17	16
ITDT							

15	14	13	12	11	10	9	8
ITDT							

7	6	5	4	3	2	1	0
ITDT							

- **ITDT: Increment/Decrement**

The number of nanoseconds to increment or decrement the 1588 Timer Nanoseconds Register. If necessary, the 1588 Seconds Register will be incremented or decremented.

- **ADJ: Adjust 1588 Timer**

Write as one to subtract from the 1588 timer. Write as zero to add to it.

37.8.97 GMAC 1588 Timer Increment Register

Name: GMAC_TI

Address: 0xF80081DC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
NIT							
15	14	13	12	11	10	9	8
ACNS							
7	6	5	4	3	2	1	0
CNS							

- **CNS: Count Nanoseconds**

A count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

- **ACNS: Alternative Count Nanoseconds**

Alternative count of nanoseconds by which the 1588 Timer Nanoseconds Register will be incremented each clock cycle.

- **NIT: Number of Increments**

The number of increments after which the alternative increment is used.

37.8.98 GMAC PTP Event Frame Transmitted Seconds Low Register

Name: GMAC_EFTSL

Address: 0xF80081E0

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.99 GMAC PTP Event Frame Transmitted Nanoseconds Register

Name: GMAC_EFTN

Address: 0xF80081E4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.100 GMAC PTP Event Frame Received Seconds Low Register

Name: GMAC_EFRSL

Address: 0xF80081E8

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.101 GMAC PTP Event Frame Received Nanoseconds Register

Name: GMAC_EFRN

Address: 0xF80081EC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.102 GMAC PTP Peer Event Frame Transmitted Seconds Low Register

Name: GMAC_PEFTSL

Address: 0xF80081F0

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.103 GMAC PTP Peer Event Frame Transmitted Nanoseconds Register

Name: GMAC_PEFTN

Address: 0xF80081F4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP transmit peer event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.104 GMAC PTP Peer Event Frame Received Seconds Low Register

Name: GMAC_PEFRSL

Address: 0xF80081F8

Access: Read-only

31	30	29	28	27	26	25	24
RUD							

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Seconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.105 GMAC PTP Peer Event Frame Received Nanoseconds Register

Name: GMAC_PEFRN

Address: 0xF80081FC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	RUD					

23	22	21	20	19	18	17	16
RUD							

15	14	13	12	11	10	9	8
RUD							

7	6	5	4	3	2	1	0
RUD							

- **RUD: Register Update**

The register is updated with the value that the 1588 Timer Nanoseconds Register holds when the SFD of a PTP receive primary event crosses the MII interface. An interrupt is issued when the register is updated.

37.8.106 GMAC Interrupt Status Register Priority Queue x

Name: GMAC_ISRPQx[x=1..2]

Address: 0xF8008400

Access: Read-only

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
–				HRESP	ROVR	–	–

7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	–	–	RXUBR	RCOMP	–

- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**

Transmit frame corruption due to AHB error—set if an error occurs whilst midway through reading transmit frame from the AHB, including HRESP errors and buffers exhausted mid frame.

- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**

37.8.107 GMAC Transmit Buffer Queue Base Address Register Priority Queue x

Name: GMAC_TBQBAPQx[x=1..2]

Address: 0xF8008440

Access: Read/Write

31	30	29	28	27	26	25	24
TXBQBA							

23	22	21	20	19	18	17	16
TXBQBA							

15	14	13	12	11	10	9	8
TXBQBA							

7	6	5	4	3	2	1	0
TXBQBA						–	–

These registers hold the start address of the transmit buffer queues (transmit buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

- **TXBQBA: Transmit Buffer Queue Base Address**

Written with the address of the start of the transmit queue.

37.8.108 GMAC Receive Buffer Queue Base Address Register Priority Queue x

Name: GMAC_RBQBAPQx[x=1..2]

Address: 0xF8008480

Access: Read/Write

31	30	29	28	27	26	25	24
RXBQBA							

23	22	21	20	19	18	17	16
RXBQBA							

15	14	13	12	11	10	9	8
RXBQBA							

7	6	5	4	3	2	1	0
RXBQBA						–	–

These registers hold the start address of the receive buffer queues (receive buffers descriptor lists) for the additional queues and must be initialized to the address of valid descriptors, even if the priority queues are not used.

- **RXBQBA: Receive Buffer Queue Base Address**

Written with the address of the start of the receive queue.

37.8.109 GMAC Receive Buffer Size Register Priority Queue x

Name: GMAC_RBSRPQx[x=1..2]

Address: 0xF80084A0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RBS							
7	6	5	4	3	2	1	0
RBS							

- **RBS: Receive Buffer Size**

DMA receive buffer size in AHB system memory. The value defined by these bits determines the size of buffer to use in main AHB system memory when writing received data.

The value is defined in multiples of 64 bytes such that a value of 0x01 corresponds to buffers of 64 bytes, 0x02 corresponds to 128 bytes etc.

For example:

0x02: 128 bytes

0x18: 1536 bytes (1 × max length frame/buffer)

0xA0: 10240 bytes (1 × 10K jumbo frame/buffer)

Note that this value should never be written as zero.

37.8.110 GMAC Credit-Based Shaping Control Register

Name: GMAC_CBSCR

Address: 0xF80084BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	QAE	QBE

- **• QAE: Queue A CBS Enable**

0: Credit-based shaping on the second highest priority queue (queue A) is disabled.

1: Credit-based shaping on the second highest priority queue (queue A) is enabled.

- **• QBE: Queue B CBS Enable**

0: Credit-based shaping on the highest priority queue (queue B) is disabled.

1: Credit-based shaping on the highest priority queue (queue B) is enabled.

37.8.111 GMAC Credit-Based Shaping IdleSlope Register for Queue A

Name: GMAC_CBSISQA

Address: 0xF80084C0

Access: Read/Write

31	30	29	28	27	26	25	24
IS							

23	22	21	20	19	18	17	16
IS							

15	14	13	12	11	10	9	8
IS							

7	6	5	4	3	2	1	0
IS							

Credit-based shaping must be disabled in GMAC_CBSCR before updating this register.

- **IS: IdleSlope**

IdleSlope value for queue A in bytes/second.

The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 1Gb/second portTransmitRate = 32'h07735940

If 50% of bandwidth was to be allocated to a particular queue in 1Gb/second mode, then the IdleSlope value for that queue would be calculated as 32'h07735940 / 2.

37.8.112 GMAC Credit-Based Shaping IdleSlope Register for Queue B

Name: GMAC_CBSISQB

Address: 0xF80084C4

Access: Read/Write

31	30	29	28	27	26	25	24
IS							

23	22	21	20	19	18	17	16
IS							

15	14	13	12	11	10	9	8
IS							

7	6	5	4	3	2	1	0
IS							

Credit-based shaping must be disabled in GMAC_CBSCR before updating this register.

- **IS: IdleSlope**

IdleSlope value for queue B in bytes/second.

The IdleSlope value is defined as the rate of change of credit when a packet is waiting to be sent. This must not exceed the port transmit rate which is dependent on the speed of operation, e.g., 1Gb/second portTransmitRate = 32'h07735940

If 50% of bandwidth was to be allocated to a particular queue in 1Gb/sec mode, then the IdleSlope value for that queue would be calculated as 32'h07735940 / 2

37.8.113 GMAC Screening Type 1 Register x Priority Queue

Name: GMAC_ST1RPQx[x=0..3]

Address: 0xF8008500

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	UDPE	DSTCE	UDPM			

23	22	21	20	19	18	17	16
UDPM							

15	14	13	12	11	10	9	8
UDPM				DSTCM			

7	6	5	4	3	2	1	0
DSTCM				–	QNB		

Screening type 1 registers are used to allocate up to 3 priority queues to received frames based on certain IP or UDP fields of incoming frames.

- **QNB: Queue Number (0–2)**

If a match is successful, then the queue value programmed in bits 2:0 is allocated to the frame.

- **DSTCM: Differentiated Services or Traffic Class Match**

When DS/TC match enable is set (bit 28), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against bits 11:4.

- **UDPM: UDP Port Match**

When UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits 27:12.

- **DSTCE: Differentiated Services or Traffic Class Match Enable**

When DS/TC match enable is set (bit 28), the DS (differentiated services) field of the received IPv4 header or TC field (traffic class) of IPv6 headers are matched against bits 11:4.

- **UDPE: UDP Port Match Enable**

When UDP port match enable is set (bit 29), the UDP Destination Port of the received UDP frame is matched against bits 27:12.

37.8.114 GMAC Screening Type 2 Register x Priority Queue

Name: GMAC_ST2RPQx[x=0..7]

Address: 0xF8008540

Access: Read/Write

31	30	29	28	27	26	25	24
–	COMPCE	COMPC					COMPBE

23	22	21	20	19	18	17	16
COMPB					COMPAE	COMPA	

15	14	13	12	11	10	9	8
COMPA			ETHE	I2ETH			VLANE

7	6	5	4	3	2	1	0
–	VLANP			–	QNB		

Screening type 2 registers are used to allocate up to 3 priority queues to received frames based on the VLAN priority field of received ethernet frames.

- **QNB: Queue Number (0–2)**

If a match is successful, then the queue value programmed in QNB is allocated to the frame.

- **VLANP: VLAN Priority**

When VLAN match enable is set (bit 8), the VLAN priority field of the received frame is matched against bits 7:4 of this register.

- **VLANE: VLAN Enable**

0: VLAN match is disabled.

1: VLAN match is enabled.

- **I2ETH: Index of Screening Type 2 EtherType register x**

When ETHE is set (bit 12), the field EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits 15:0 in the register designated by the value of I2ETH.

- **ETHE: EtherType Enable**

0: EtherType match with bits 15:0 in the register designated by the value of I2ETH is disabled.

1: EtherType match with bits 15:0 in the register designated by the value of I2ETH is enabled.

- **COMPA: Index of Screening Type 2 Compare Word 0/Word 1 register x**

COMPA is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPAE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

- **COMPAE: Compare A Enable**

0: Comparison via the register designated by index COMPA is disabled.

1: Comparison via the register designated by index COMPA is enabled.

- **COMPB: Index of Screening Type 2 Compare Word 0/Word 1 register x**

COMPB is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPBE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

- **COMPBE: Compare B Enable**

0: Comparison via the register designated by index COMPB is disabled.

1: Comparison via the register designated by index COMPB is enabled.

- **COMPC: Index of Screening Type 2 Compare Word 0/Word 1 register x**

COMPC is a pointer to the compare registers GMAC_ST2CW0x and GMAC_ST2CW1x. When COMPCE is set, the compare is true if the data at the frame offset ANDed with the value MASKVAL is equal to the value of COMPVAL ANDed with the value of MASKVAL.

- **COMPCE: Compare C Enable**

0: Comparison via the register designated by index COMPC is disabled.

1: Comparison via the register designated by index COMPC is enabled.

37.8.115 GMAC Interrupt Enable Register Priority Queue x

Name: GMAC_IERPQx[x=1..2]

Address: 0xF8008600

Access: Write-only

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
–				HRESP	ROVR	–	–

7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	–	–	RXUBR	RCOMP	–

The following values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**

37.8.116 GMAC Interrupt Disable Register Priority Queue x

Name: GMAC_IDRPQx[x=1..2]

Address: 0xF8008620

Access: Write-only

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
–				HRESP	ROVR	–	–

7	6	5	4	3	2	1	0
TCOMP	TFC	RLEX	–	–	RXUBR	RCOMP	–

The following values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **TFC: Transmit Frame Corruption Due to AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**

37.8.117 GMAC Interrupt Mask Register Priority Queue x

Name: GMAC_IMRPQx[x=1..2]

Address: 0xF8008640

Access: Read/Write

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
–				HRESP	ROVR	–	–

7	6	5	4	3	2	1	0
TCOMP	AHB	RLEX	–	–	RXUBR	RCOMP	–

A read of this register returns the value of the receive complete interrupt mask.

A write to this register directly affects the state of the corresponding bit in the Interrupt Status Register, causing an interrupt to be generated if a 1 is written.

The following values are valid for all listed bit names of this register:

0: Corresponding interrupt is enabled.

1: Corresponding interrupt is disabled.

- **RCOMP: Receive Complete**
- **RXUBR: RX Used Bit Read**
- **RLEX: Retry Limit Exceeded or Late Collision**
- **AHB: AHB Error**
- **TCOMP: Transmit Complete**
- **ROVR: Receive Overrun**
- **HRESP: HRESP Not OK**

37.8.118 GMAC Screening Type 2 EtherType Register x

Name: GMAC_ST2ERx[x=0..3]

Address: 0xF80086E0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
COMPVAL							

7	6	5	4	3	2	1	0
COMPVAL							

- **COMPVAL: Ethertype Compare Value**

When the bit GMAC_ST2RPQ.ETHE is enabled, the EtherType (last EtherType in the header if the frame is VLAN tagged) is compared with bits 15:0 in the register designated by GMAC_ST2RPQ.I2ETH.

37.8.119 GMAC Screening Type 2 Compare Word 0 Register x

Name: GMAC_ST2CW0x[x=0..23]

Access: Read/Write

31	30	29	28	27	26	25	24
COMPVAL							
23	22	21	20	19	18	17	16
COMPVAL							
15	14	13	12	11	10	9	8
MASKVAL							
7	6	5	4	3	2	1	0
MASKVAL							

- **MASKVAL: Mask Value**

The value of MASKVAL ANDed with the 2 bytes extracted from the frame is compared to the value of MASKVAL ANDed with the value of COMPVAL.

- **COMPVAL: Compare Value**

The byte stored in bits [23:16] is compared against the first byte of the 2 bytes extracted from the frame.

The byte stored in bits [31:24] is compared against the second byte of the 2 bytes extracted from the frame.

37.8.120 GMAC Screening Type 2 Compare Word 1 Register x

Name: GMAC_ST2CW1x[x=0..23]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	OFFSSTRT

7	6	5	4	3	2	1	0
OFFSSTRT	OFFSVAL						

- **OFFSVAL: Offset Value in Bytes**

The value of OFFSVAL ranges from 0 to 127 bytes, and is counted from either the start of the frame, the byte after the EtherType field (last EtherType in the header if the frame is VLAN tagged), the byte after the IP header (IPv4 or IPv6) or the byte after the TCP/UDP header.

- **OFFSSTRT: Ethernet Frame Offset Start**

Value	Name	Description
0	FRAMESTART	Offset from the start of the frame
1	ETHERTYPE	Offset from the byte after the EtherType field
2	IP	Offset from the byte after the IP header field
3	TCP_UDP	Offset from the byte after the TCP/UDP header field

38. USB High Speed Device Port (UDPHS)

38.1 Description

The USB High Speed Device Port (UDPHS) is compliant with the Universal Serial Bus (USB), rev 2.0 High Speed device specification.

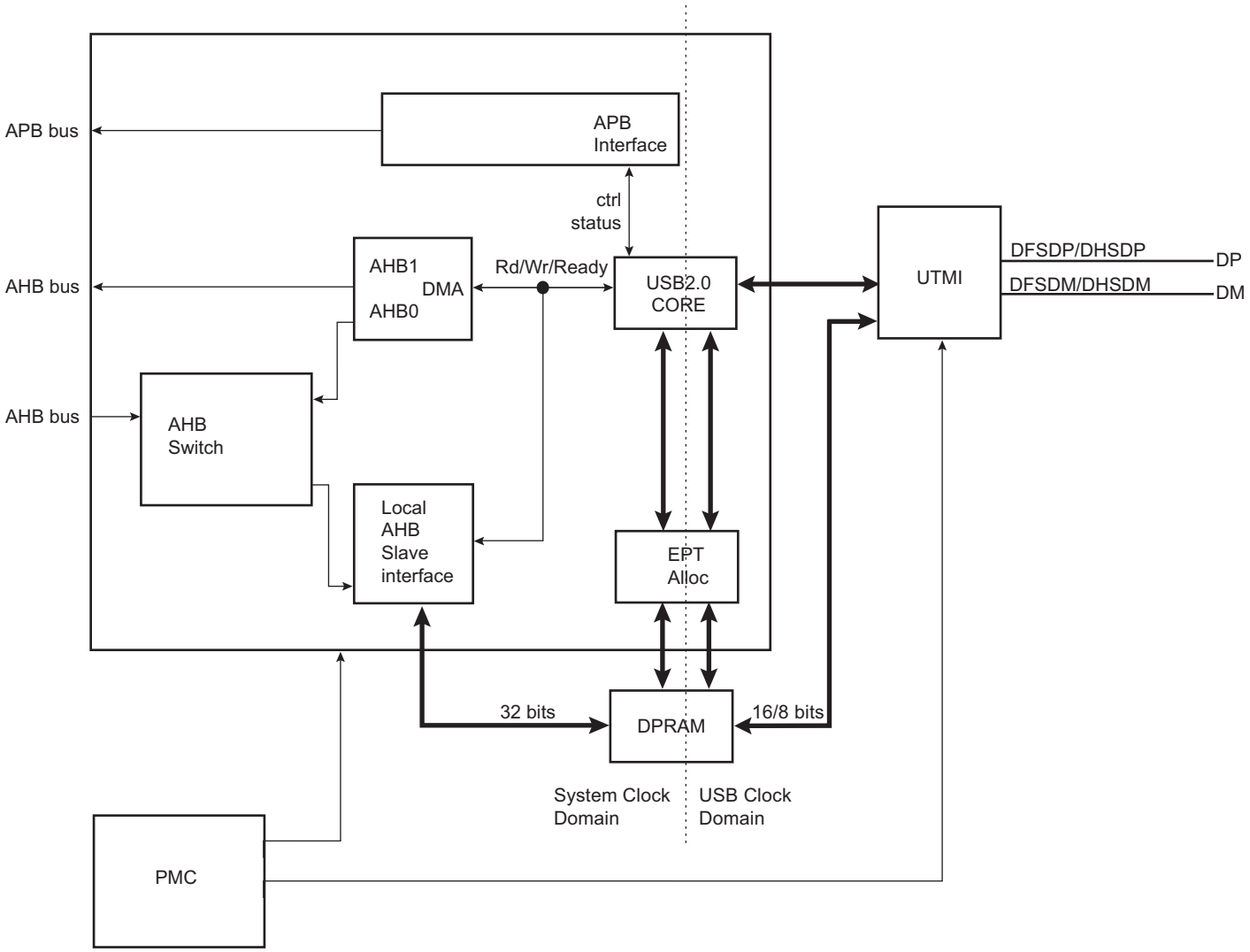
Each endpoint can be configured in one of several USB transfer types. It can be associated with one, two or three banks of a Dual-port RAM used to store the current data payload. If two or three banks are used, one DPR bank is read or written by the processor, while the other is read or written by the USB device peripheral. This feature is mandatory for isochronous endpoints.

38.2 Embedded Characteristics

- 1 Device High Speed
- 1 UTMI transceiver shared between Host and Device
- USB v2.0 High Speed Compliant, 480 Mbit/s
- 16 Endpoints up to 1024 bytes
- Embedded Dual-port RAM for Endpoints
- Suspend/Resume Logic (Command of UTMI)
- Up to Three Memory Banks for Endpoints (Not for Control Endpoint)
- 8 Kbytes of DPRAM

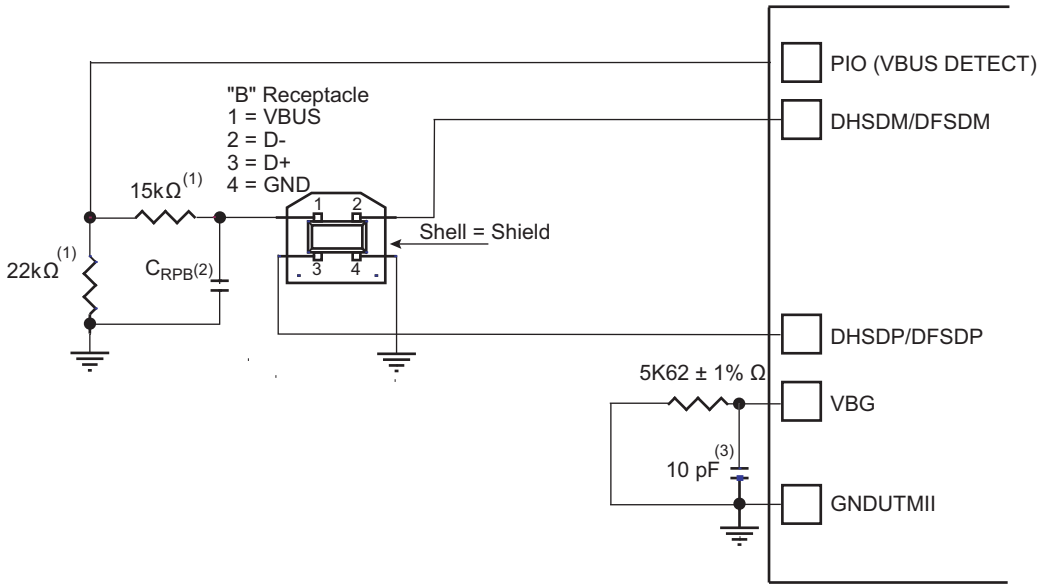
38.3 Block Diagram

Figure 38-1. Block Diagram



38.4 Typical Connection

Figure 38-2. Board Schematic



Notes:
1. The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3V3-supplied PIOs.
2. CRPB: Upstream Facing Port Bypass Capacitance of 1 µF to 10 µF (refer to “DC Electrical Characteristics” in Universal Serial Bus Specification Rev. 2)
3. 10 pF capacitor on VBG is a provision and may not be populated.

38.5 Product Dependencies

38.5.1 Power Management

The UDPHS is not continuously clocked.

For using the UDPHS, the programmer must first enable the UDPHS Clock in the Power Management Controller Peripheral Clock Enable Register (PMC_PCER). Then enable the PLL in the PMC UTMI Clock Configuration Register (CKGR_UCKR). Finally, enable BIAS in CKGR_UCKR.

However, if the application does not require UDPHS operations, the UDPHS clock can be stopped when not needed and restarted later.

38.5.2 Interrupt Sources

The UDPHS interrupt line is connected on one of the internal sources of the interrupt controller. Using the UDPHS interrupt requires the interrupt controller to be programmed first.

Table 38-1. Peripheral IDs

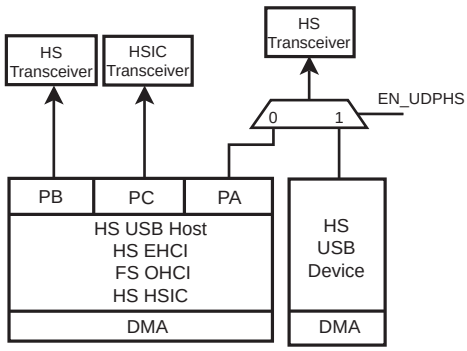
Instance	ID
UDPHS	42

38.6 Functional Description

38.6.1 UTMI transceivers Sharing

The High Speed USB Host Port A is shared with the High Speed USB Device port and connected to the second UTMI transceiver. The selection between Host Port A and USB Device is controlled by the UDPHS enable bit (EN_UDPHS) located in the UDPHS_CTRL register.

Figure 38-3. USB Selection



38.6.2 USB V2.0 High Speed Device Port Introduction

The USB V2.0 High Speed Device Port provides communication services between host and attached USB devices. Each device is offered with a collection of communication flows (pipes) associated with each endpoint. Software on the host communicates with a USB Device through a set of communication flows.

38.6.3 USB V2.0 High Speed Transfer Types

A communication flow is carried over one of four transfer types defined by the USB device.

A device provides several logical communication pipes with the host. To each logical pipe is associated an endpoint. Transfer through a pipe belongs to one of the four transfer types:

- Control Transfers: Used to configure a device at attach time and can be used for other device-specific purposes, including control of other pipes on the device.
- Bulk Data Transfers: Generated or consumed in relatively large burst quantities and have wide dynamic latitude in transmission constraints.
- Interrupt Data Transfers: Used for timely but reliable delivery of data, for example, characters or coordinates with human-perceptible echo or feedback response characteristics.
- Isochronous Data Transfers: Occupy a prenegotiated amount of USB bandwidth with a prenegotiated delivery latency. (Also called streaming real time transfers.)

As indicated below, transfers are sequential events carried out on the USB bus.

Endpoints must be configured according to the transfer type they handle.

Table 38-2. USB Communication Flow

Transfer	Direction	Bandwidth	Endpoint Size	Error Detection	Retrying
Control	Bidirectional	Not guaranteed	8, 16, 32, 64	Yes	Automatic
Isochronous	Unidirectional	Guaranteed	8–1024	Yes	No
Interrupt	Unidirectional	Not guaranteed	8–1024	Yes	Yes
Bulk	Unidirectional	Not guaranteed	8–512	Yes	Yes

38.6.4 USB Transfer Event Definitions

A transfer is composed of one or several transactions as shown in the following table.

Table 38-3. USB Transfer Events

Transfer		Transaction
Direction	Type	
CONTROL (bidirectional)	Control Transfer (1)	• Setup transaction → Data IN transactions → Status OUT transaction • Setup transaction → Data OUT transactions → Status IN transaction • Setup transaction → Status IN transaction
IN (device toward host)	Bulk IN Transfer	• Data IN transaction → Data IN transaction
	Interrupt IN Transfer	• Data IN transaction → Data IN transaction
	Isochronous IN Transfer (2)	• Data IN transaction → Data IN transaction
OUT (host toward device)	Bulk OUT Transfer	• Data OUT transaction → Data OUT transaction
	Interrupt OUT Transfer	• Data OUT transaction → Data OUT transaction
	Isochronous OUT Transfer (2)	• Data OUT transaction → Data OUT transaction

Notes: 1. Control transfer must use endpoints with one bank and can be aborted using a stall handshake.
2. Isochronous transfers must use endpoints configured with two or three banks.

An endpoint handles all transactions related to the type of transfer for which it has been configured.

Table 38-4. UDPHS Endpoint Description

Endpoint #	Mnemonic	Nb Bank	DMA	High Band Width	Max. Endpoint Size	Endpoint Type
0	EPT_0	1	N	N	64	Control
1	EPT_1	3	Y	Y	1024	Ctrl/Bulk/Iso(1)/Interrupt
2	EPT_2	3	Y	Y	1024	Ctrl/Bulk/Iso(1)/Interrupt
3	EPT_3	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
4	EPT_4	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
5	EPT_5	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
6	EPT_6	2	Y	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
7	EPT_7	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
8	EPT_8	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
9	EPT_9	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
10	EPT_10	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
11	EPT_11	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt

Table 38-4. UDPHS Endpoint Description (Continued)

Endpoint #	Mnemonic	Nb Bank	DMA	High Band Width	Max. Endpoint Size	Endpoint Type
12	EPT_12	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
13	EPT_13	2	N	N	1024	Ctrl/Bulk/Isov/Interrupt
14	EPT_14	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt
15	EPT_15	2	N	N	1024	Ctrl/Bulk/Iso(1)/Interrupt

Note: 1. In Isochronous (Iso) mode, it is preferable that High Band Width capability is available.

The size of internal DPRAM is 8 KB.

Suspend and resume are automatically detected by the UDPHS device, which notifies the processor by raising an interrupt.

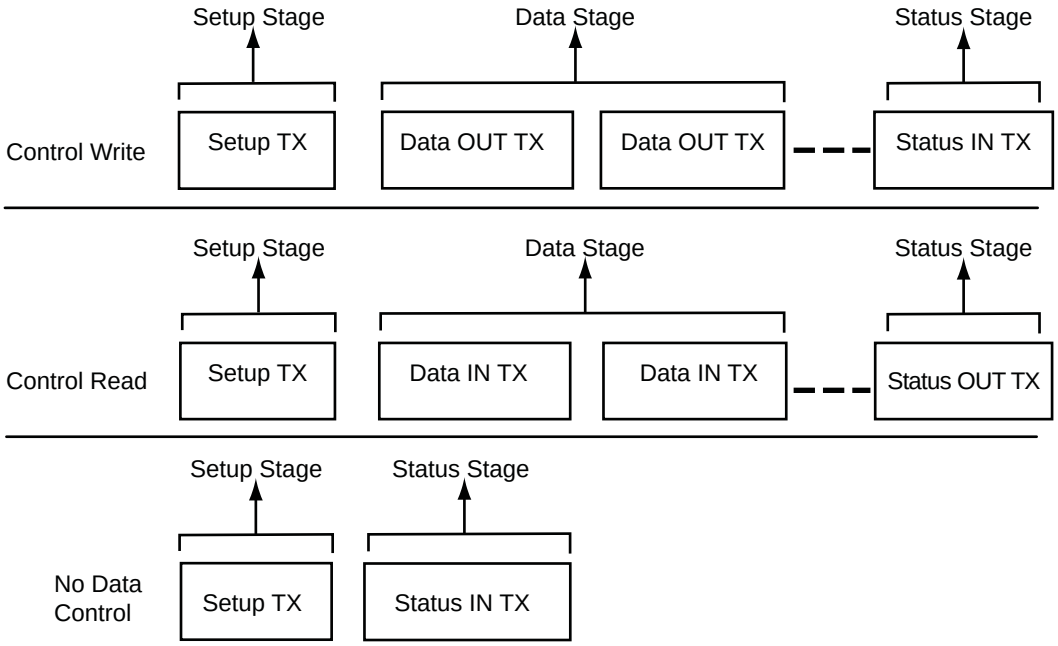
38.6.5 USB V2.0 High Speed BUS Transactions

Each transfer results in one or more transactions over the USB bus.

There are five kinds of transactions flowing across the bus in packets:

1. Setup Transaction
2. Data IN Transaction
3. Data OUT Transaction
4. Status IN Transaction
5. Status OUT Transaction

Figure 38-4. Control Read and Write Sequences



A status IN or OUT transaction is identical to a data IN or OUT transaction.

38.6.6 Endpoint Configuration

The endpoint 0 is always a control endpoint, it must be programmed and active in order to be enabled when the End Of Reset interrupt occurs.

To configure the endpoints:

- Fill the configuration register (UDPHS_EPTCFG) with the endpoint size, direction (IN or OUT), type (CTRL, Bulk, IT, ISO) and the number of banks.
- Fill the number of transactions (NB_TRANS) for isochronous endpoints.

Note: For control endpoints the direction has no effect.

- Verify that the EPT_MAPD flag is set. This flag is set if the endpoint size and the number of banks are correct compared to the FIFO maximum capacity and the maximum number of allowed banks.
- Configure control flags of the endpoint and enable it in UDPHS_EPTCTLENBx according to Section 38.7.16 “UDPHS Endpoint Control Disable Register (Isochronous Endpoint)”.

Control endpoints can generate interrupts and use only 1 bank.

All endpoints (except endpoint 0) can be configured either as Bulk, Interrupt or Isochronous. See Table 38-4. UDPHS Endpoint Description.

The maximum packet size they can accept corresponds to the maximum endpoint size.

Note: The endpoint size of 1024 is reserved for isochronous endpoints.

The size of the DPRAM is 8 KB. The DPR is shared by all active endpoints. The memory size required by the active endpoints must not exceed the size of the DPRAM.

SIZE_DPRAM = SIZE _EPT0

+ NB_BANK_EPT1 x SIZE_EPT1

+ NB_BANK_EPT2 x SIZE_EPT2

+ NB_BANK_EPT3 x SIZE_EPT3

+ NB_BANK_EPT4 x SIZE_EPT4

+ NB_BANK_EPT5 x SIZE_EPT5

+ NB_BANK_EPT6 x SIZE_EPT6

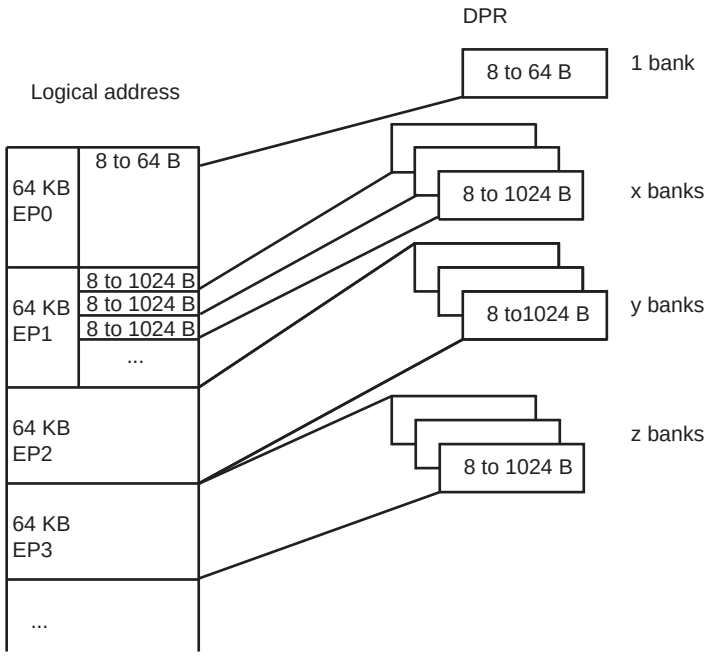
+... (refer to 38.7.12 UDPHS Endpoint Configuration Register)

If a user tries to configure endpoints with a size the sum of which is greater than the DPRAM, then the EPT_MAPD is not set.

The application has access to the physical block of DPR reserved for the endpoint through a 64 KB logical address space.

The physical block of DPR allocated for the endpoint is remapped all along the 64 KB logical address space. The application can write a 64 KB buffer linearly.

Figure 38-5. Logical Address Space for DPR Access



Configuration examples of UDPHS_EPTCTLx (UDPHS Endpoint Control Disable Register (Isochronous Endpoint)) for Bulk IN endpoint type follow below.

- With DMA
  - AUTO_VALID: Automatically validate the packet and switch to the next bank.
  - EPT_ENABL: Enable endpoint.
- Without DMA:
  - TXRDY: An interrupt is generated after each transmission.
  - EPT_ENABL: Enable endpoint.

Configuration examples of Bulk OUT endpoint type follow below.

- With DMA
  - AUTO_VALID: Automatically validate the packet and switch to the next bank.
  - EPT_ENABL: Enable endpoint.
- Without DMA
  - RXRDY_TXKL: An interrupt is sent after a new packet has been stored in the endpoint FIFO.
  - EPT_ENABL: Enable endpoint.

38.6.7 DPRAM Management

Endpoints can only be allocated in ascending order, from the endpoint 0 to the last endpoint to be allocated. The user shall therefore configure them in the same order.

The allocation of an endpoint x starts when the Number of Banks field in the UDPHS Endpoint Configuration Register (UDPHS_EPTCFGx.BK_NUMBER) is different from zero. Then, the hardware allocates a memory area in the DPRAM and inserts it between the x-1 and x+1 endpoints. The x+1 endpoint memory window slides up and its data is lost. Note that the following endpoint memory windows (from x+2) do not slide.

Disabling an endpoint, by writing a one to the Endpoint Disable bit in the UDPHS Endpoint Control Disable Register (UDPHS_EPTCTLDISx.EPT_DISABL), does not reset its configuration:

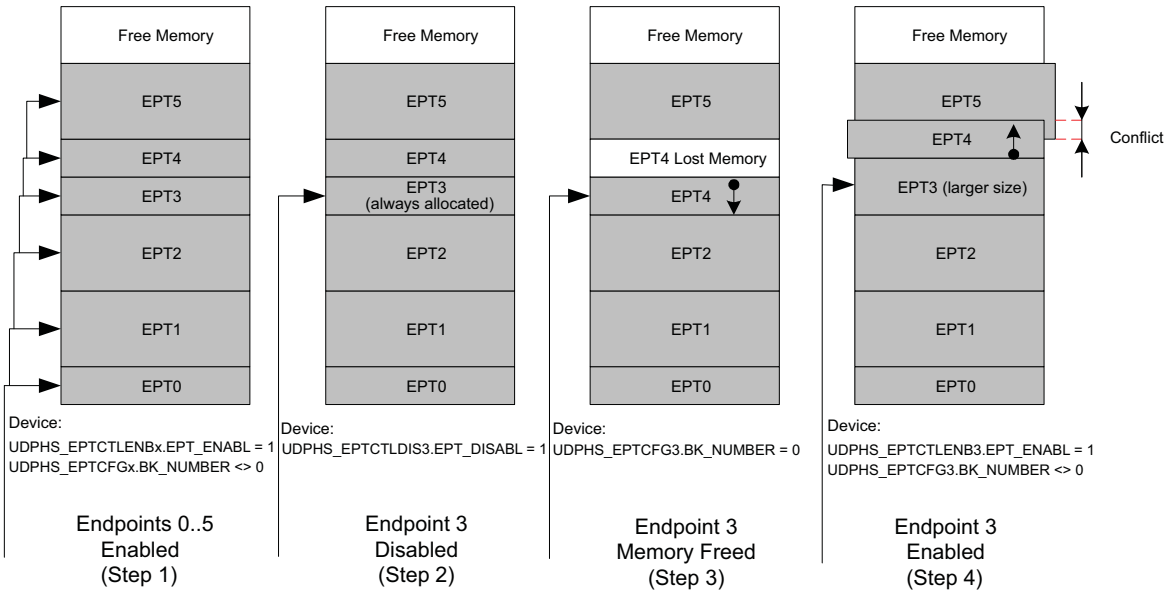
- Endpoint Banks (UDPHS_EPTCFGx.BK_NUMBER)

- Endpoint Size (UDPHS_EPTCFGx.EPT_SIZE)
- Endpoint Direction (UDPHS_EPTCFGx.EPT_DIR)
- Endpoint Type (UDPHS_EPTCFGx.EPT_TYPE)

To free its memory, the user shall write a zero to the UDPHS_EPTCFGx.BK_NUMBER field. The x+1 endpoint memory window then slides down and its data is lost. Note that the following endpoint memory windows (from x+2) do not slide.

Figure 38-6 illustrates the allocation and reorganization of the DPRAM in a typical example.

Figure 38-6. Example of DPRAM Allocation and Reorganization



DPRAM allocation sequence:

1. The endpoints 0 to 5 are enabled, configured and allocated in ascending order. Each endpoint then owns a memory area in the DPRAM.
2. The endpoint 3 is disabled, but its memory is kept allocated by the controller.
3. In order to free its memory, its UDPHS_EPTCFGx.BK_NUMBER field is written to zero. The endpoint 4 memory window slides down, but the endpoint 5 does not move.
4. If the user chooses to reconfigure the endpoint 3 with a larger size, the controller allocates a memory area after the endpoint 2 memory area and automatically slides up the endpoint 4 memory window. The endpoint 5 does not move and a memory conflict appears as the memory windows of the endpoints 4 and 5 overlap. The data of these endpoints is potentially lost.

Notes:
1. There is no way the data of the endpoint 0 can be lost (except if it is de-allocated) as the memory allocation and de-allocation may affect only higher endpoints.
2. Deactivating then reactivating the same endpoint with the same configuration only modifies temporarily the controller DPRAM pointer and size for this endpoint. Nothing changes in the DPRAM, higher endpoints seem not to have been moved and their data is preserved as far as nothing has been written or received into them while changing the allocation state of the first endpoint.
3. When the user writes a value different from zero to the UDPHS_EPTCFGx.BK_NUMBER field, the Endpoint Mapped bit (UDPHS_EPTCFGx.EPT_MAPD) is set only if the configured size and number of banks are correct as compared to the endpoint maximal allowed values and to the maximal FIFO size (i.e., the DPRAM size). The UDPHS_EPTCFGx.EPT_MAPD value does not consider memory allocation conflicts.

38.6.8 Transfer With DMA

USB packets of any length may be transferred when required by the UDPHS device. These transfers always feature sequential addressing.

Packet data AHB bursts may be locked on a DMA buffer basis for drastic overall AHB bus bandwidth performance boost with paged memories. These clock-cycle consuming memory row (or bank) changes will then likely not occur, or occur only once instead of several times, during a single big USB packet DMA transfer in case another AHB master addresses the memory. The locked bursts result in up to 128-word single-cycle unbroken AHB bursts for bulk endpoints and 256-word single-cycle unbroken bursts for isochronous endpoints.

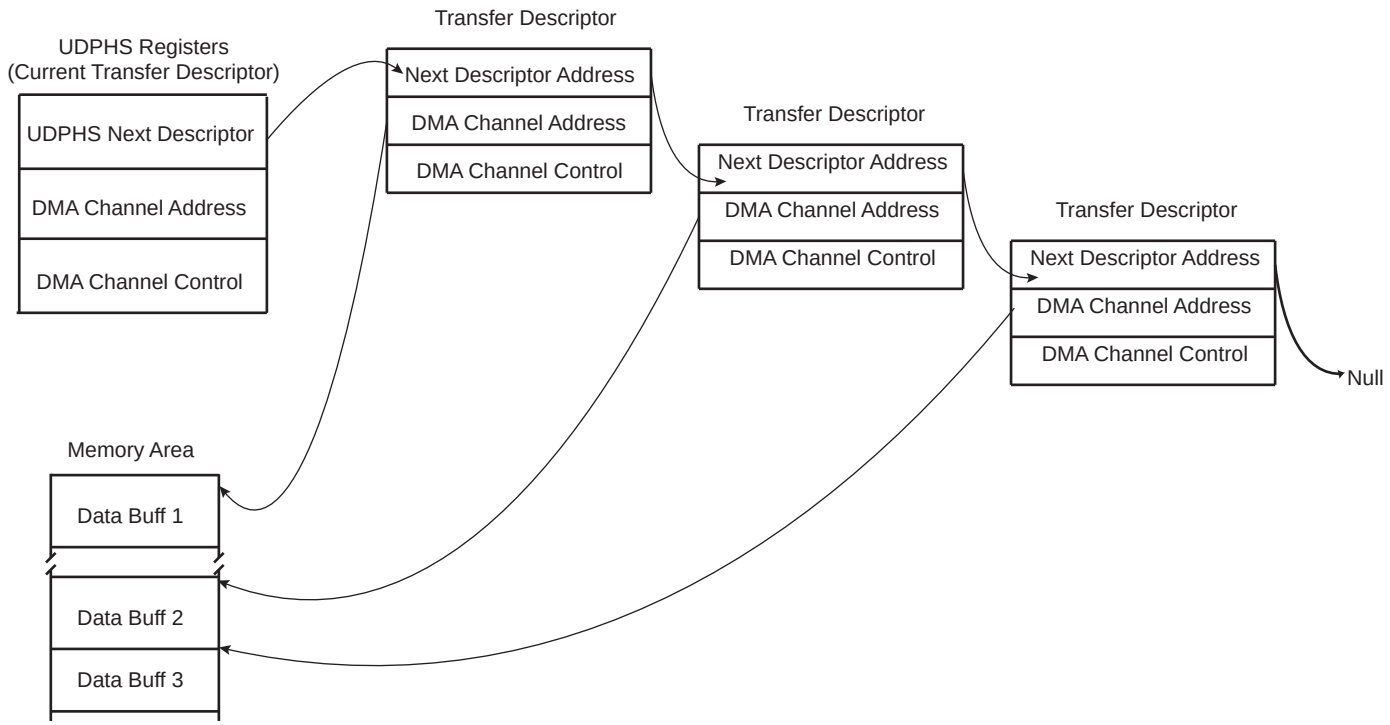
This maximum burst length is then controlled by the lowest programmed USB endpoint size (EPT_SIZE field in the UDPHS_EPTCFGx register) and DMA Size (BUFF_LENGTH field in the UDPHS_DMACONTROLx register).

The USB 2.0 device average throughput may be up to nearly 60 Mbyte/s. Its internal slave average access latency decreases as burst length increases due to the 0 wait-state side effect of unchanged endpoints. If at least 0 wait-state word burst capability is also provided by the external DMA AHB bus slaves, each of both DMA AHB busses need less than 50% bandwidth allocation for full USB 2.0 bandwidth usage at 30 MHz, and less than 25% at 60 MHz.

The UDPHS DMA Channel Transfer Descriptor is described in Section 38.7.25 “UDPHS DMA Channel Transfer Descriptor”.

Note: In case of debug, be careful to address the DMA to an SRAM address even if a remap is done.

Figure 38-7. Example of DMA Chained List



38.6.9 Transfer Without DMA

Important: If the DMA is not to be used, it is necessary to disable it, otherwise it can be enabled by previous versions of software **without warning**. If this should occur, the DMA can process data before an interrupt without knowledge of the user.

The recommended means to disable DMA are as follows:

```
// Reset IP UDPHS
    AT91C_BASE_UDPHS->UDPHS_CTRL &= ~AT91C_UDPHS_EN_UDPHS;
    AT91C_BASE_UDPHS->UDPHS_CTRL |= AT91C_UDPHS_EN_UDPHS;
// With OR without DMA !!!
    for( i=1; i<=((AT91C_BASE_UDPHS->UDPHS_IPFEATURES &
AT91C_UDPHS_DMA_CHANNEL_NBR)>>4); i++ ) {
// RESET endpoint canal DMA:
        // DMA stop channel command
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0;  // STOP
command
// Disable endpoint
        AT91C_BASE_UDPHS->UDPHS_EPT[i].UDPHS_EPTCTLDIS |= 0XFFFFFFFF;
// Reset endpoint config
        AT91C_BASE_UDPHS->UDPHS_EPT[i].UDPHS_EPTCTLCFG = 0;
// Reset DMA channel (Buff count and Control field)
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0x02;  // NON
STOP command
// Reset DMA channel 0 (STOP)
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMACONTROL = 0;  // STOP
command
// Clear DMA channel status (read the register for clear it)
        AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMASTATUS =
AT91C_BASE_UDPHS->UDPHS_DMA[i].UDPHS_DMASTATUS;
}
```

38.6.10 Handling Transactions with USB V2.0 Device Peripheral

38.6.10.1 Setup Transaction

The setup packet is valid in the DPR while RX_SETUP is set. Once RX_SETUP is cleared by the application, the UDPHS accepts the next packets sent over the device endpoint.

When a valid setup packet is accepted by the UDPHS:

- The UDPHS device automatically acknowledges the setup packet (sends an ACK response)
- Payload data is written in the endpoint
- Sets the RX_SETUP interrupt
- The BYTE_COUNT field in the UDPHS_EPTSTAx register is updated

An endpoint interrupt is generated while RX_SETUP in the UDPHS_EPTSTAx register is not cleared. This interrupt is carried out to the microcontroller if interrupts are enabled for this endpoint.

Thus, firmware must detect RX_SETUP polling UDPHS_EPTSTAx or catching an interrupt, read the setup packet in the FIFO, then clear the RX_SETUP bit in the UDPHS_EPTCLRSTA register to acknowledge the setup stage.

If STALL_SNT was set to 1, then this bit is automatically reset when a setup token is detected by the device. Then, the device still accepts the setup stage. (See Section 38.6.10.5 “STALL”).

38.6.10.2 NYET

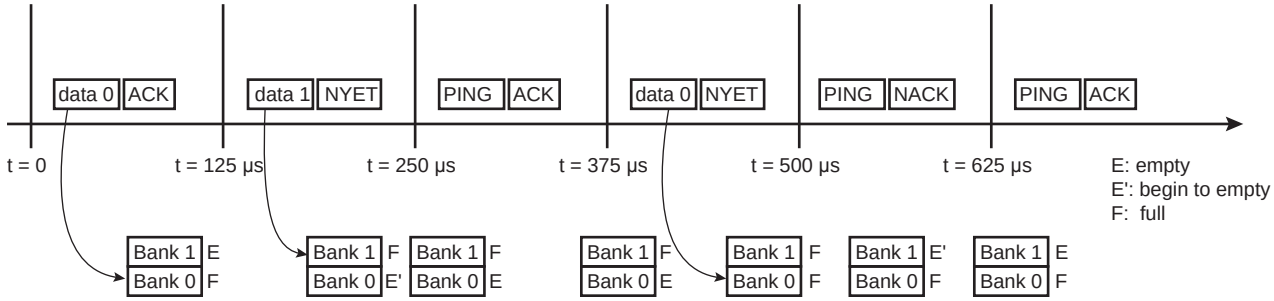
NYET is a High Speed only handshake. It is returned by a High Speed endpoint as part of the PING protocol.

High Speed devices must support an improved NAK mechanism for Bulk OUT and control endpoints (except setup stage). This mechanism allows the device to tell the host whether it has sufficient endpoint space for the next OUT transfer (see USB 2.0 spec 8.5.1 NAK Limiting via Ping Flow Control).

The NYET/ACK response to a High Speed Bulk OUT transfer and the PING response are automatically handled by hardware in the UDPHS_EPTCTLx register (except when the user wants to force a NAK response by using the NYET_DIS bit).

If the endpoint responds instead to the OUT/DATA transaction with an NYET handshake, this means that the endpoint accepted the data but does not have room for another data payload. The host controller must return to using a PING token until the endpoint indicates it has space available.

Figure 38-8. NYET Example with Two Endpoint Banks



38.6.10.3 Data IN

Bulk IN or Interrupt IN

Data IN packets are sent by the device during the data or the status stage of a control transfer or during an (interrupt/bulk/isochronous) IN transfer. Data buffers are sent packet by packet under the control of the application or under the control of the DMA channel.

There are three ways for an application to transfer a buffer in several packets over the USB:

- packet by packet (see below)
- 64 KB (see below)
- DMA (see below)

Bulk IN or Interrupt IN: Sending a Packet Under Application Control (Device to Host)

The application can write one or several banks.

A simple algorithm can be used by the application to send packets regardless of the number of banks associated to the endpoint.

Algorithm Description for Each Packet:

- The application waits for TXRDY flag to be cleared in the UDPHS_EPTSTAx register before it can perform a write access to the DPR.
- The application writes one USB packet of data in the DPR through the 64 KB endpoint logical memory window.
- The application sets TXRDY flag in the UDPHS_EPTSETSTAx register.

The application is notified that it is possible to write a new packet to the DPR by the TXRDY interrupt. This interrupt can be enabled or masked by setting the TXRDY bit in the UDPHS_EPTCTLENB/UDPHS_EPTCTLDIS register.

Algorithm Description to Fill Several Packets:

Using the previous algorithm, the application is interrupted for each packet. It is possible to reduce the application overhead by writing linearly several banks at the same time. The AUTO_VALID bit in the UDPHS_EPTCTLx must be set by writing the AUTO_VALID bit in the UDPHS_EPTCTLENBx register.

The auto-valid-bank mechanism allows the transfer of data (IN and OUT) without the intervention of the CPU. This means that bank validation (set TXRDY or clear the RXRDY_TXKL bit) is done by hardware.

- The application checks the BUSY_BANK_STA field in the UDPHS_EPTSTAx register. The application must wait that at least one bank is free.
- The application writes a number of bytes inferior to the number of free DPR banks for the endpoint. Each time the application writes the last byte of a bank, the TXRDY signal is automatically set by the UDPHS.

- If the last packet is incomplete (i.e., the last byte of the bank has not been written) the application must set the TXRDY bit in the UDPHS_EPTSETSTAx register.

The application is notified that all banks are free, so that it is possible to write another burst of packets by the BUSY_BANK interrupt. This interrupt can be enabled or masked by setting the BUSY_BANK flag in the UDPHS_EPTCTLENB and UDPHS_EPTCTLDIS registers.

This algorithm must not be used for isochronous transfer. In this case, the ping-pong mechanism does not operate.

A Zero Length Packet can be sent by setting just the TXRDY flag in the UDPHS_EPTSETSTAx register.

Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host)

The UDPHS integrates a DMA host controller. This DMA controller can be used to transfer a buffer from the memory to the DPR or from the DPR to the processor memory under the UDPHS control. The DMA can be used for all transfer types except control transfer.

Example DMA configuration:

1. Program UDPHS_DMAADDRESS x with the address of the buffer that should be transferred.
2. Enable the interrupt of the DMA in UDPHS_IEN
3. Program UDPHS_ DMACONTROLx:
   - Size of buffer to send: size of the buffer to be sent to the host.
   - END_B_EN: The endpoint can validate the packet (according to the values programmed in the AUTO_VALID and SHRT_PCKT fields of UDPHS_EPTCTLx.) (See Section 38.7.16 “UDPHS Endpoint Control Disable Register (Isochronous Endpoint)” and Figure 38-13)
   - END_BUFFIT: generate an interrupt when the BUFF_COUNT in UDPHS_DMASTATUSx reaches 0.
   - CHANN_ENB: Run and stop at end of buffer

The auto-valid-bank mechanism allows the transfer of data (IN & OUT) without the intervention of the CPU. This means that bank validation (set TXRDY or clear the RXRDY_TXKL bit) is done by hardware.

A transfer descriptor can be used. Instead of programming the register directly, a descriptor should be programmed and the address of this descriptor is then given to UDPHS_DMANXTDSC to be processed after setting the LDNXT_DSC field (Load Next Descriptor Now) in UDPHS_DMACONTROLx register.

The structure that defines this transfer descriptor must be aligned.

Each buffer to be transferred must be described by a DMA Transfer descriptor (see Section 38.7.25 “UDPHS DMA Channel Transfer Descriptor”). Transfer descriptors are chained. Before executing transfer of the buffer, the UDPHS may fetch a new transfer descriptor from the memory address pointed by the UDPHS_DMANXTDSCx register. Once the transfer is complete, the transfer status is updated in the UDPHS_DMASTATUSx register.

To chain a new transfer descriptor with the current DMA transfer, the DMA channel must be stopped. To do so, INTDIS_DMA and TXRDY may be set in the UDPHS_EPTCTLENBx register. It is also possible for the application to wait for the completion of all transfers. In this case the LDNXT_DSC bit in the last transfer descriptor UDPHS_DMACONTROLx register must be set to 0 and the CHANN_ENB bit set to 1.

Then the application can chain a new transfer descriptor.

The INTDIS_DMA can be used to stop the current DMA transfer if an enabled interrupt is triggered. This can be used to stop DMA transfers in case of errors.

The application can be notified at the end of any buffer transfer (ENB_BUFFIT bit in the UDPHS_DMACONTROLx register).

Figure 38-9. Data IN Transfer for Endpoint with One Bank

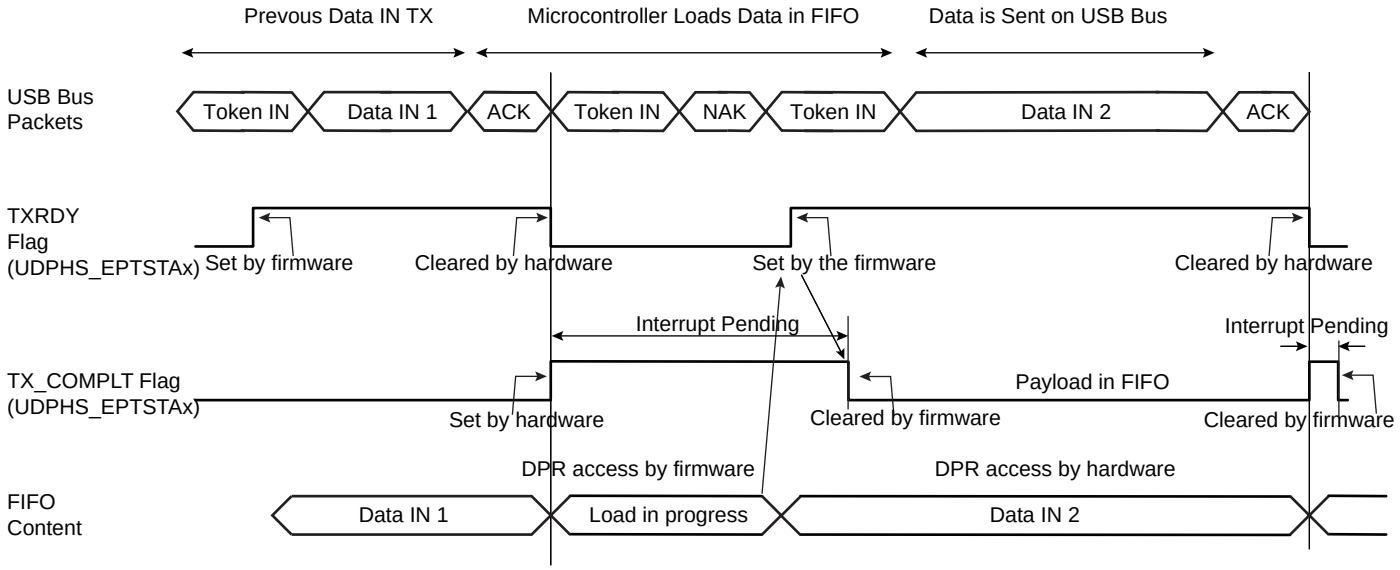


Figure 38-10. Data IN Transfer for Endpoint with Two Banks

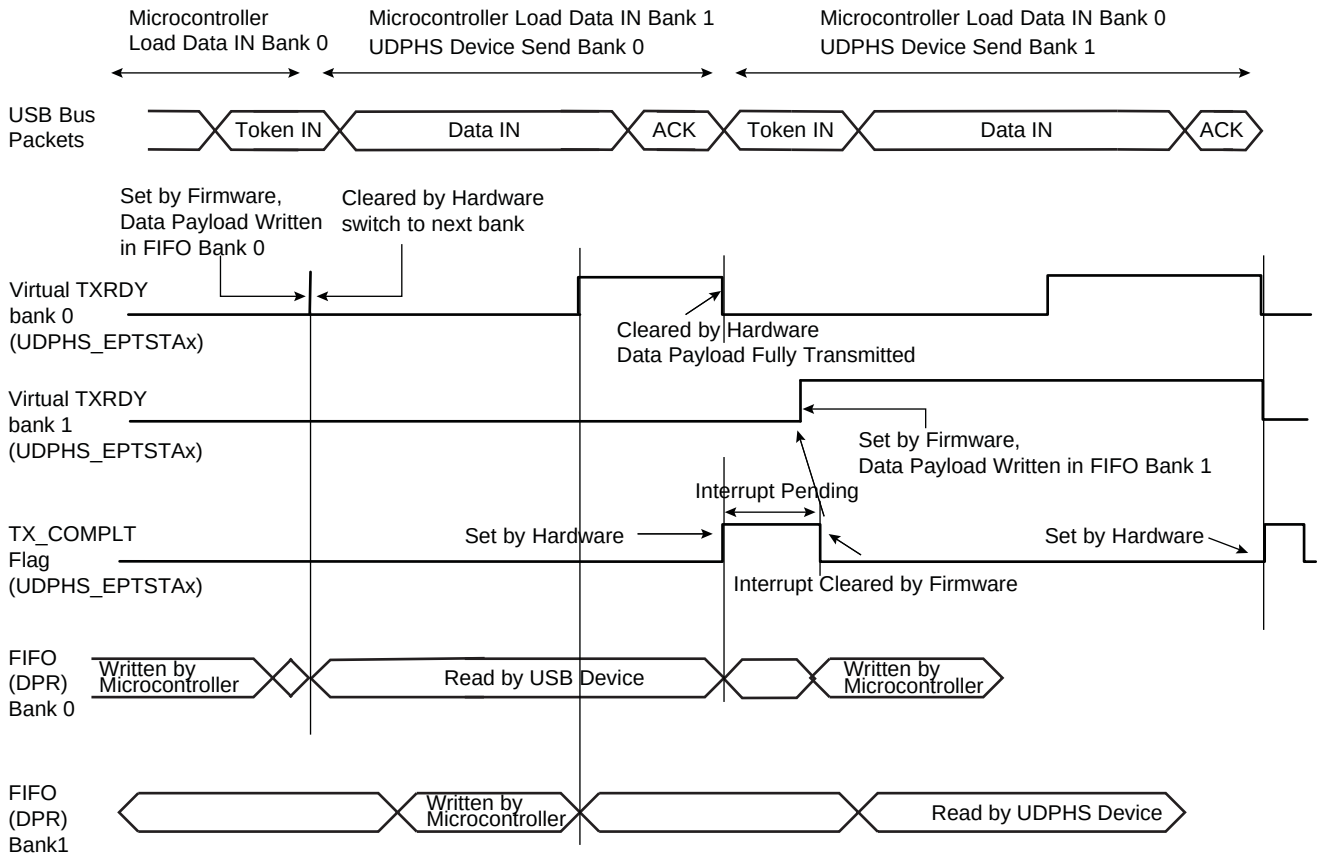
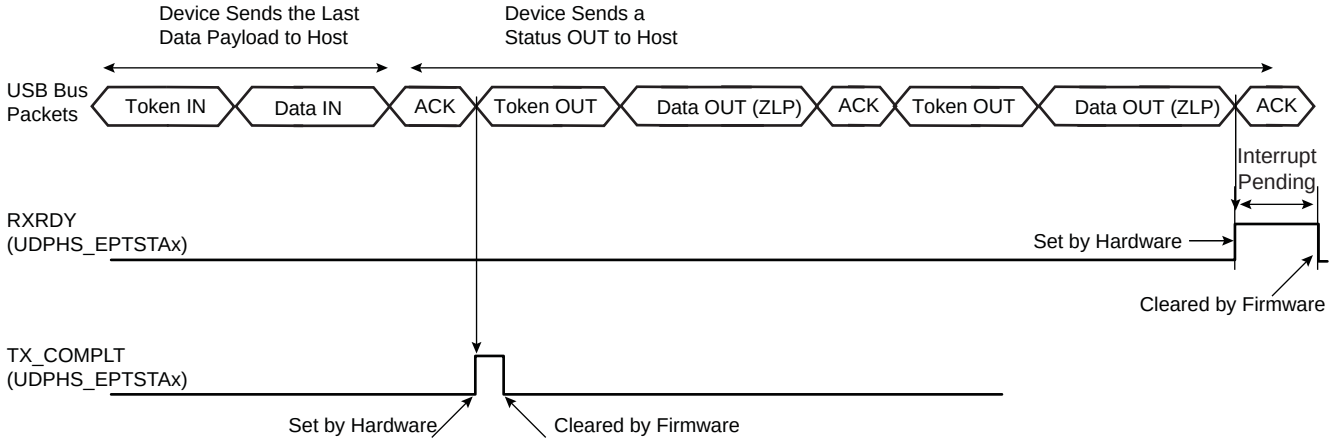
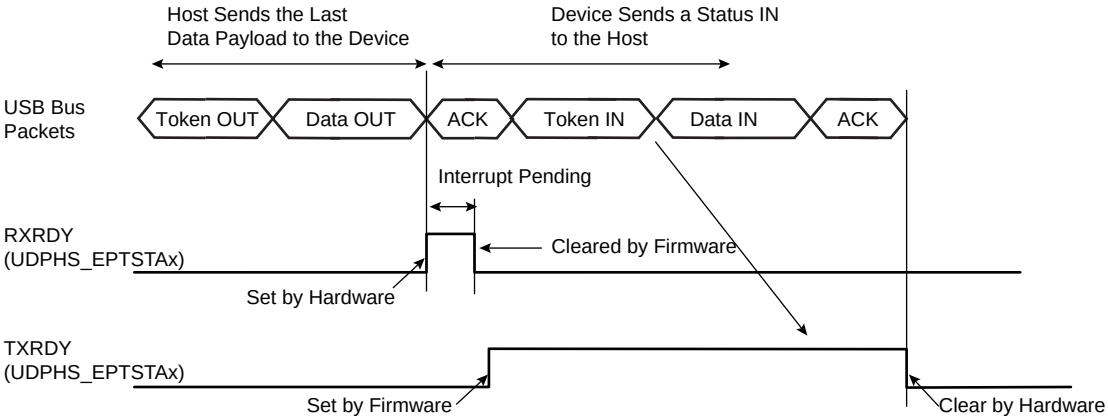


Figure 38-11. Data IN Followed By Status OUT Transfer at the End of a Control Transfer



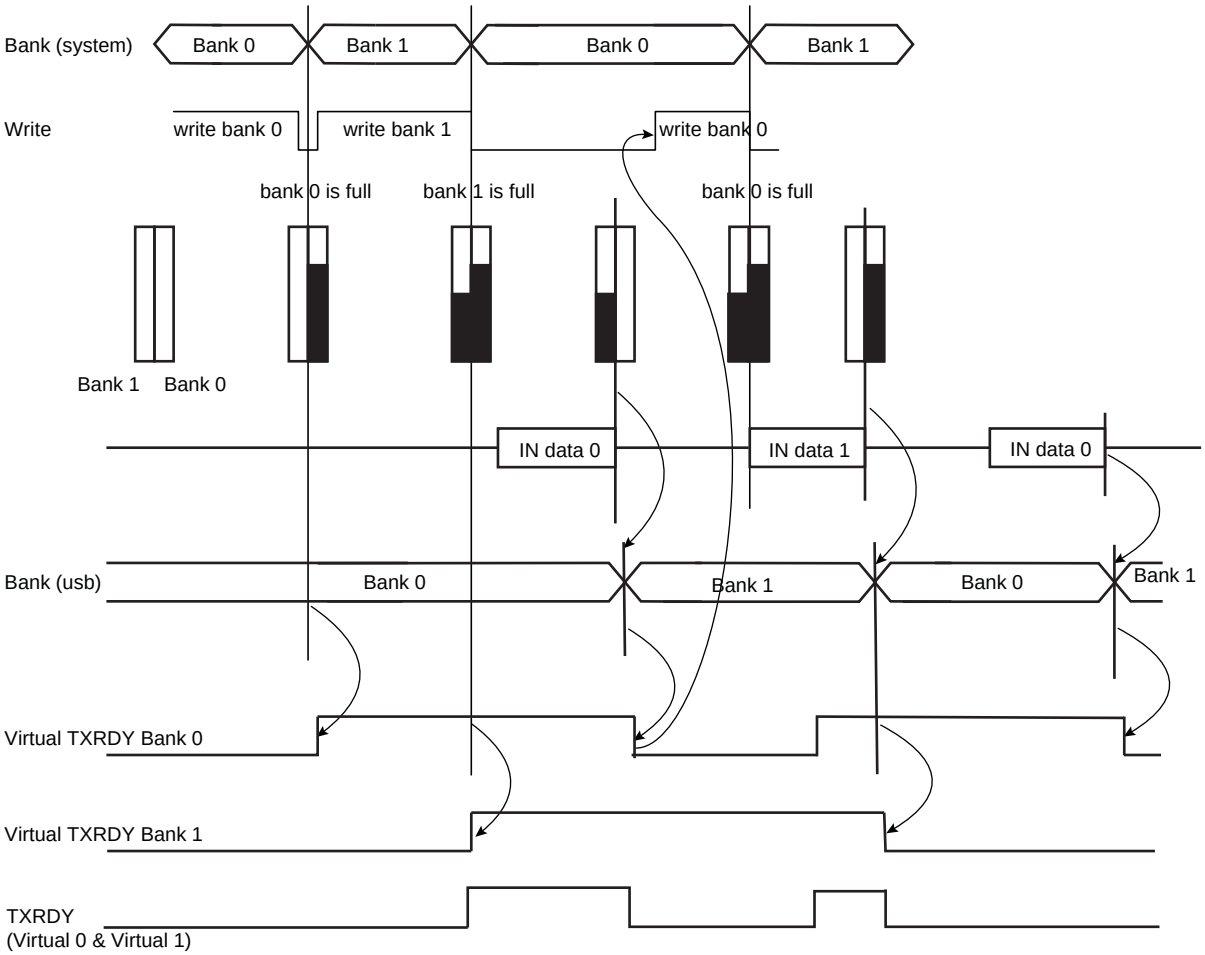
Note: A NAK handshake is always generated at the first status stage token.

Figure 38-12. Data OUT Followed by Status IN Transfer



Note: Before proceeding to the status stage, the software should determine that there is no risk of extra data from the host (data stage). If not certain (non-predictable data stage length), then the software should wait for a NAK-IN interrupt before proceeding to the status stage. This precaution should be taken to avoid collision in the FIFO.

Figure 38-13. Autovalid with DMA



Note: In the illustration above Autovalid validates a bank as full, although this might not be the case, in order to continue processing data and to send to DMA.

Isochronous IN

Isochronous-IN is used to transmit a stream of data whose timing is implied by the delivery rate. Isochronous transfer provides periodic, continuous communication between host and device.

It guarantees bandwidth and low latencies appropriate for telephony, audio, video, etc.

If the endpoint is not available (TXRDY_TRER = 0), then the device does not answer to the host. An ERR_FL_ISO interrupt is generated in the UDPHS_EPTSTAx register and once enabled, then sent to the CPU.

The STALL_SNT command bit is not used for an ISO-IN endpoint.

High Bandwidth Isochronous Endpoint Handling: IN Example

For high bandwidth isochronous endpoints, the DMA can be programmed with the number of transactions (BUFF_LENGTH field in UDPHS_DMACONTROLx) and the system should provide the required number of packets per microframe, otherwise, the host will notice a sequencing problem.

A response should be made to the first token IN recognized inside a microframe under the following conditions:

- If at least one bank has been validated, the correct DATAx corresponding to the programmed Number Of Transactions per Microframe (NB_TRANS) should be answered. In case of a subsequent missed or corrupted token IN inside the microframe, the USB 2.0 Core available data bank(s) that should normally have been transmitted during that microframe shall be flushed at its end. If this flush occurs, an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).

- If no bank is validated yet, the default DATA0 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). Then, no data bank is flushed at microframe end.
- If no data bank has been validated at the time when a response should be made for the second transaction of NB_TRANS = 3 transactions microframe, a DATA1 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). If and only if remaining untransmitted banks for that microframe are available at its end, they are flushed and an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If no data bank has been validated at the time when a response should be made for the last programmed transaction of a microframe, a DATA0 ZLP is answered and underflow is flagged (ERR_FL_ISO is set in UDPHS_EPTSTAx). If and only if the remaining untransmitted data bank for that microframe is available at its end, it is flushed and an error condition is flagged (ERR_FLUSH is set in UDPHS_EPTSTAx).
- If at the end of a microframe no valid token IN has been recognized, no data bank is flushed and no error condition is reported.

At the end of a microframe in which at least one data bank has been transmitted, if less than NB_TRANS banks have been validated for that microframe, an error condition is flagged (ERR_TRANS is set in UDPHS_EPTSTAx).

Cases of Error (in UDPHS_EPTSTAx)

- ERR_FL_ISO: There was no data to transmit inside a microframe, so a ZLP is answered by default.
- ERR_FLUSH: At least one packet has been sent inside the microframe, but the number of token IN received is lesser than the number of transactions actually validated (TXRDY_TRER) and likewise with the NB_TRANS programmed.
- ERR_TRANS: At least one packet has been sent inside the microframe, but the number of token IN received is lesser than the number of programmed NB_TRANS transactions and the packets not requested were not validated.
- ERR_FL_ISO + ERR_FLUSH: At least one packet has been sent inside the microframe, but the data has not been validated in time to answer one of the following token IN.
- ERR_FL_ISO + ERR_TRANS: At least one packet has been sent inside the microframe, but the data has not been validated in time to answer one of the following token IN and the data can be discarded at the microframe end.
- ERR_FLUSH + ERR_TRANS: The first token IN has been answered and it was the only one received, a second bank has been validated but not the third, whereas NB_TRANS was waiting for three transactions.
- ERR_FL_ISO + ERR_FLUSH + ERR_TRANS: The first token IN has been treated, the data for the second Token IN was not available in time, but the second bank has been validated before the end of the microframe. The third bank has not been validated, but three transactions have been set in NB_TRANS.

38.6.10.4 Data OUT

Bulk OUT or Interrupt OUT

Like data IN, data OUT packets are sent by the host during the data or the status stage of control transfer or during an interrupt/bulk/isochronous OUT transfer. Data buffers are sent packet by packet under the control of the application or under the control of the DMA channel.

Bulk OUT or Interrupt OUT: Receiving a Packet Under Application Control (Host to Device)

Algorithm Description for Each Packet:

- The application enables an interrupt on RXRDY_TXKL.
- When an interrupt on RXRDY_TXKL is received, the application knows that UDPHS_EPTSTAx register BYTE_COUNT bytes have been received.
- The application reads the BYTE_COUNT bytes from the endpoint.
- The application clears RXRDY_TXKL.

Note: If the application does not know the size of the transfer, it may **not** be a good option to use AUTO_VALID. Because if a zero-length-packet is received, the RXRDY_TXKL is automatically cleared by the AUTO_VALID hardware and if the endpoint interrupt is triggered, the software will not find its originating flag when reading the UDPHS_EPTSTAx register.

Algorithm to Fill Several Packets:

- The application enables the interrupts of BUSY_BANK and AUTO_VALID.
- When a BUSY_BANK interrupt is received, the application knows that all banks available for the endpoint have been filled. Thus, the application can read all banks available.

If the application does not know the size of the receive buffer, instead of using the BUSY_BANK interrupt, the application must use RXRDY_TXKL.

Bulk OUT or Interrupt OUT: Sending a Buffer Using DMA (Host To Device)

To use the DMA setting, the AUTO_VALID field is mandatory.

See Bulk IN or Interrupt IN: Sending a Buffer Using DMA (Device to Host) for more information.

DMA Configuration Example:

1. First program UDPHS_DMAADDRESSx with the address of the buffer that should be transferred.
2. Enable the interrupt of the DMA in UDPHS_IEN
3. Program the DMA Channelx Control Register:
   - Size of buffer to be sent.
   - END_B_EN: Can be used for OUT packet truncation (discarding of unbuffered packet data) at the end of DMA buffer.
   - END_BUFFIT: Generate an interrupt when BUFF_COUNT in the UDPHS_DMASTATUSx register reaches 0.
   - END_TR_EN: End of transfer enable, the UDPHS device can put an end to the current DMA transfer, in case of a short packet.
   - END_TR_IT: End of transfer interrupt enable, an interrupt is sent after the last USB packet has been transferred by the DMA, if the USB transfer ended with a short packet. (Beneficial when the receive size is unknown.)
   - CHANN_ENB: Run and stop at end of buffer.

For OUT transfer, the bank will be automatically cleared by hardware when the application has read all the bytes in the bank (the bank is empty).

Notes:
1. When a zero-length-packet is received, RXRDY_TXKL bit in UDPHS_EPTSTAx is cleared automatically by AUTO_VALID, and the application knows of the end of buffer by the presence of the END_TR_IT.
2. If the host sends a zero-length packet, and the endpoint is free, then the device sends an ACK. No data is written in the endpoint, the RXRDY_TXKL interrupt is generated, and the BYTE_COUNT field in UDPHS_EPTSTAx is null.

Figure 38-14. Data OUT Transfer for Endpoint with One Bank

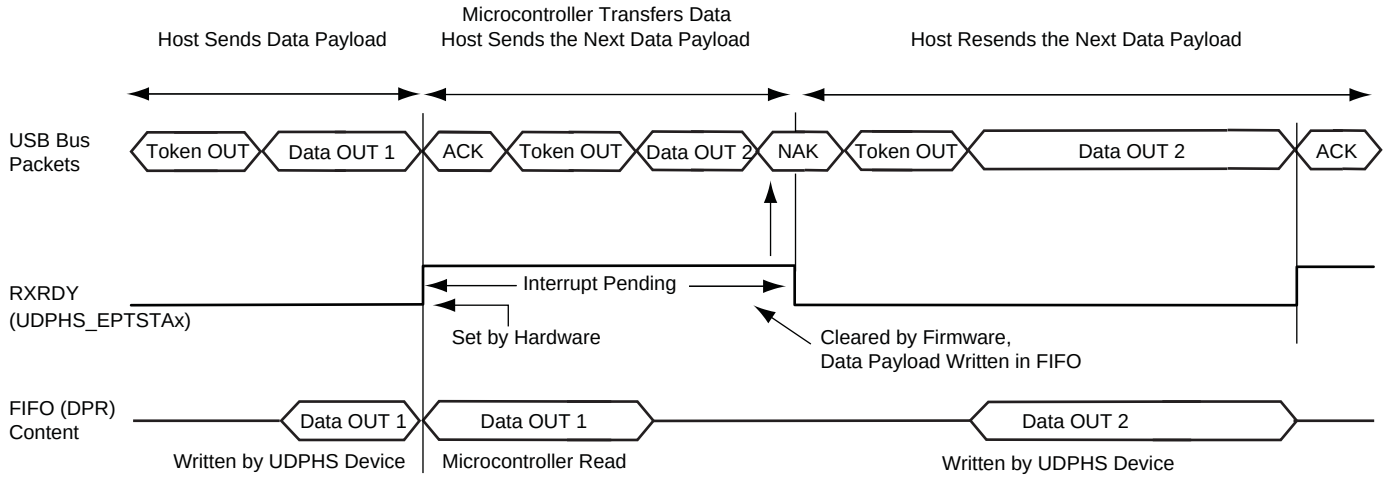
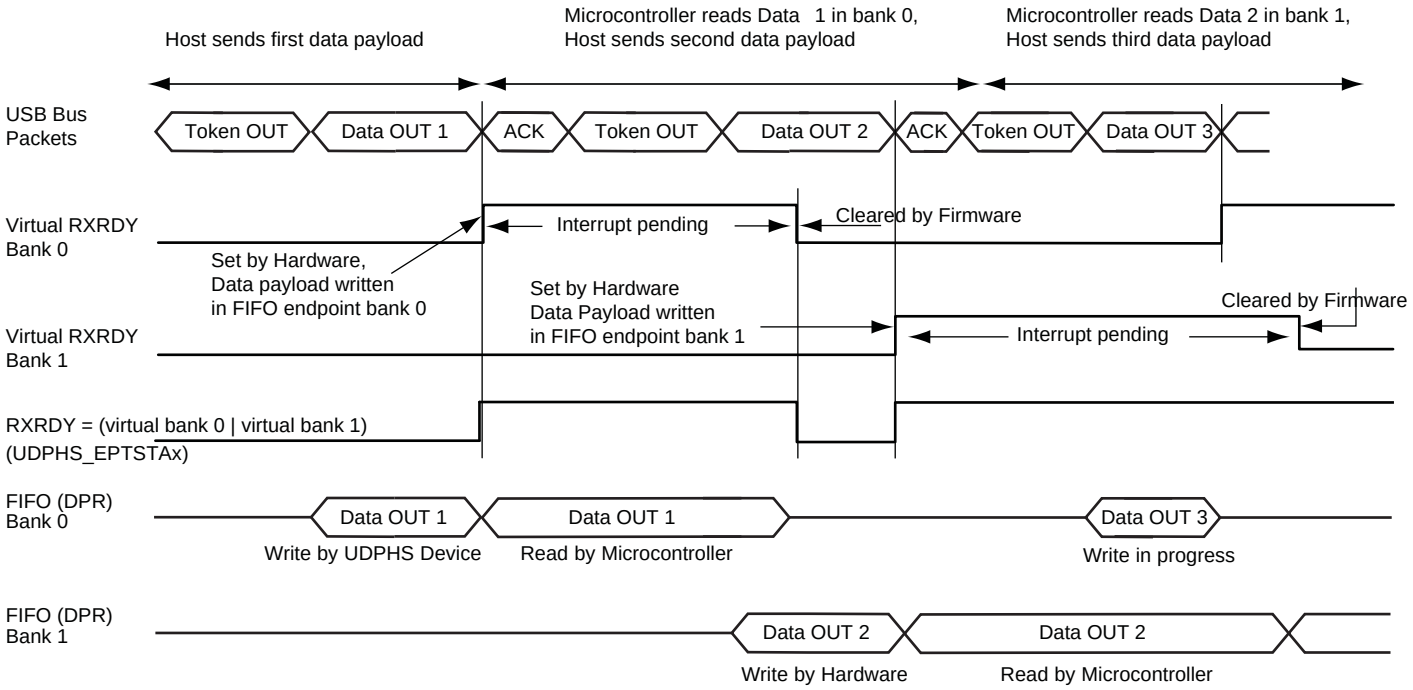


Figure 38-15. Data OUT Transfer for an Endpoint with Two Banks



High Bandwidth Isochronous Endpoint OUT

USB 2.0 supports individual High Speed isochronous endpoints that require data rates up to 192 Mb/s (24 MB/s): 3x1024 data bytes per microframe.

To support such a rate, two or three banks may be used to buffer the three consecutive data packets. The microcontroller (or the DMA) should be able to empty the banks very rapidly (at least 24 MB/s on average).

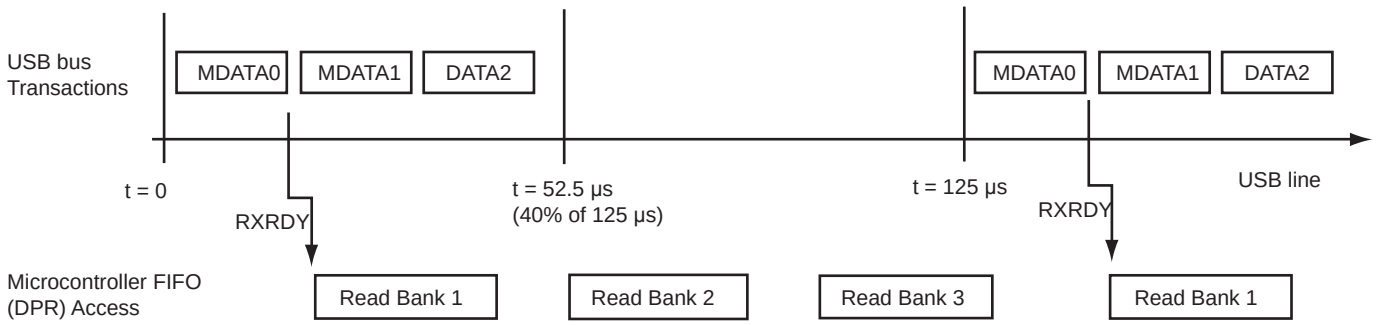
NB_TRANS field in UDPHS_EPTCFGx register = Number Of Transactions per Microframe.

If NB_TRANS > 1 then it is High Bandwidth.

Example:

- If NB_TRANS = 3, the sequence should be either
  - MData0
  - MData0/Data1
  - MData0/Data1/Data2
- If NB_TRANS = 2, the sequence should be either
  - MData0
  - MData0/Data1
- If NB_TRANS = 1, the sequence should be
  - Data0

Figure 38-16. Bank Management, Example of Three Transactions per Microframe



Isochronous Endpoint Handling: OUT Example

The user can ascertain the bank status (free or busy), and the toggle sequencing of the data packet for each bank with the UDPHS_EPTSTAx register in the three fields as follows:

- TOGGLESQ_STA: PID of the data stored in the current bank
- CURBK: Number of the bank currently being accessed by the microcontroller.
- BUSY_BANK_STA: Number of busy bank

This is particularly useful in case of a missing data packet.

If the inter-packet delay between the OUT token and the Data is greater than the USB standard, then the ISO-OUT transaction is ignored. (Payload data is not written, no interrupt is generated to the CPU.)

If there is a data CRC (Cyclic Redundancy Check) error, the payload is, none the less, written in the endpoint. The ERR_CRC_NTR flag is set in UDPHS_EPTSTAx register.

If the endpoint is already full, the packet is not written in the DPRAM. The ERR_FL_ISO flag is set in UDPHS_EPTSTAx.

If the payload data is greater than the maximum size of the endpoint, then the ERR_OVFLW flag is set. It is the task of the CPU to manage this error. The data packet is written in the endpoint (except the extra data).

If the host sends a Zero Length Packet, and the endpoint is free, no data is written in the endpoint, the RXRDY_TXKL flag is set, and the BYTE_COUNT field in UDPHS_EPTSTAx register is null.

The FRCESTALL command bit is unused for an isochronous endpoint.

Otherwise, payload data is written in the endpoint, the RXRDY_TXKL interrupt is generated and the BYTE_COUNT in UDPHS_EPTSTAx register is updated.

38.6.10.5 STALL

STALL is returned by a function in response to an IN token or after the data phase of an OUT or in response to a PING transaction. STALL indicates that a function is unable to transmit or receive data, or that a control pipe request is not supported.

- OUT

To stall an endpoint, set the FRCESTALL bit in UDPHS_EPTSETSTAx register and after the STALL_SNT flag has been set, set the TOGGLE_SEG bit in the UDPHS_EPTCLRSTAx register.

- IN

Set the FRCESTALL bit in UDPHS_EPTSETSTAx register.

Figure 38-17. Stall Handshake Data OUT Transfer

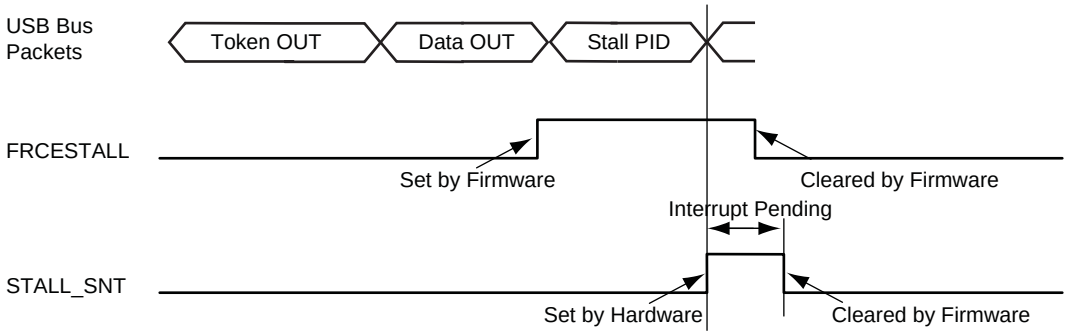
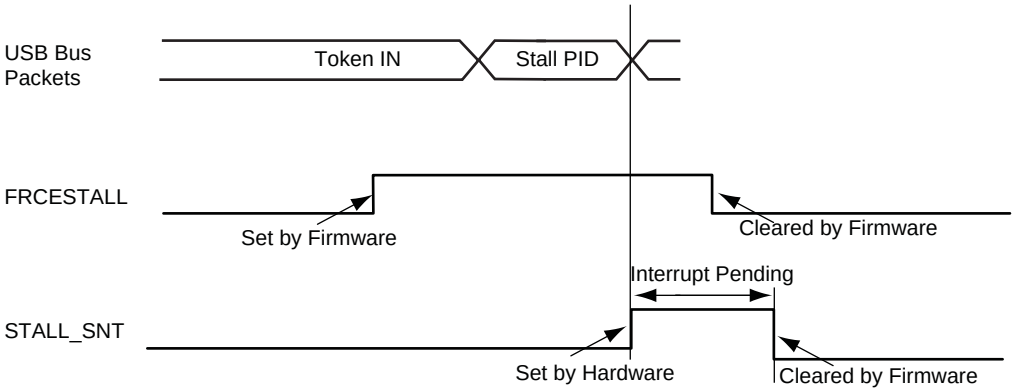


Figure 38-18. Stall Handshake Data IN Transfer



38.6.11 Speed Identification

The high speed reset is managed by hardware.

At the connection, the host makes a reset which could be a classic reset (full speed) or a high speed reset.

At the end of the reset process (full or high), the ENDRESET interrupt is generated.

Then the CPU should read the SPEED bit in UDPHS_INTSTAx to ascertain the speed mode of the device.

38.6.12 USB V2.0 High Speed Global Interrupt

Interrupts are defined in Section 38.7.3 “UDPHS Interrupt Enable Register” (UDPHS_IEN) and in Section 38.7.4 “UDPHS Interrupt Status Register” (UDPHS_INTSTA).

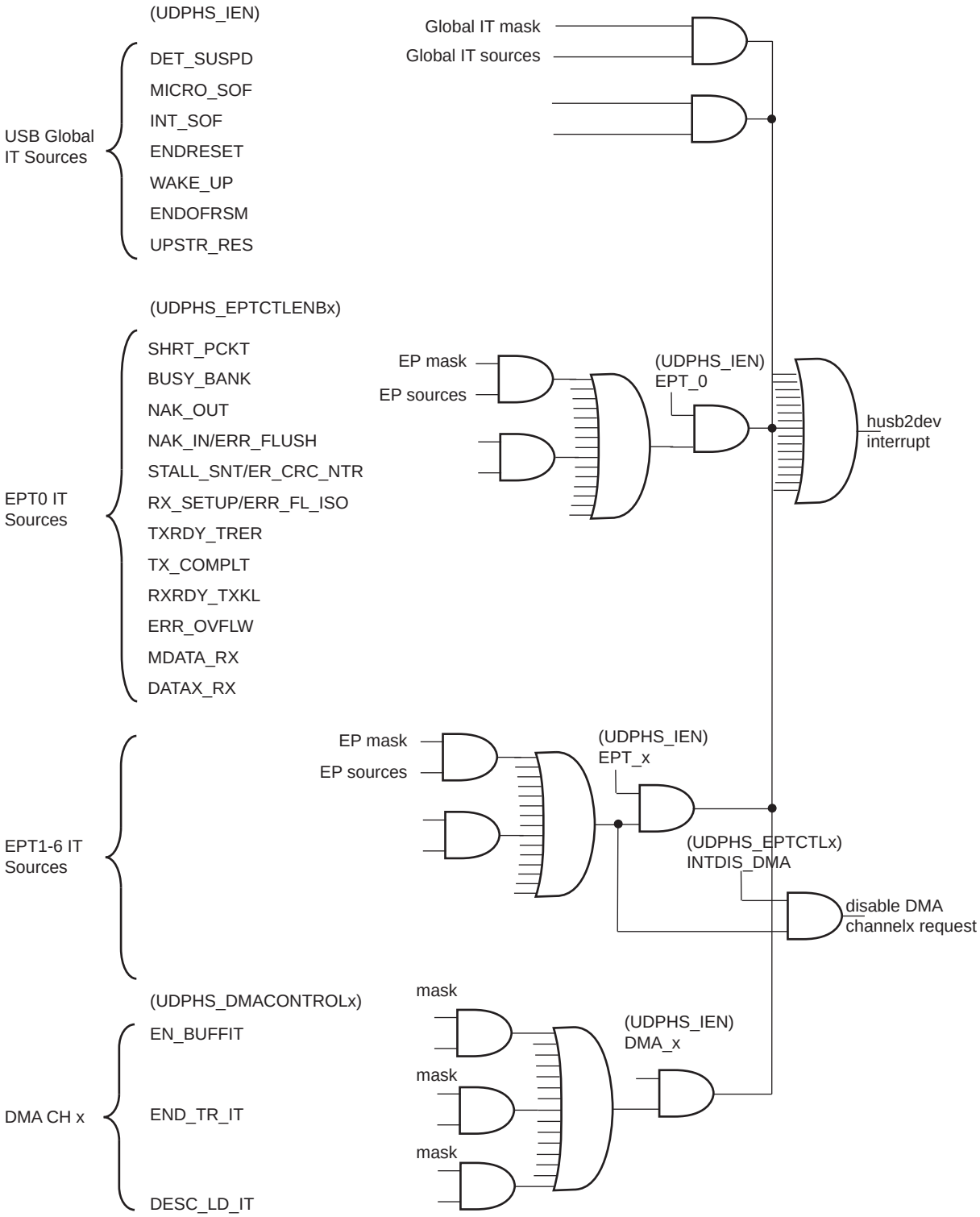
38.6.13 Endpoint Interrupts

Interrupts are enabled in UDPHS_IEN (see Section 38.7.3 “UDPHS Interrupt Enable Register”) and individually masked in UDPHS_EPTCTLENBx (see Section 38.7.13 “UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)”).

Table 38-5. Endpoint Interrupt Source Masks

SHRT_PCKT	Short Packet Interrupt
BUSY_BANK	Busy Bank Interrupt
NAK_OUT	NAKOUT Interrupt
NAK_IN/ERR_FLUSH	NAKIN/Error Flush Interrupt
STALL_SNT/ERR_CRC_NTR	Stall Sent/CRC error/Number of Transaction Error Interrupt
RX_SETUP/ERR_FL_ISO	Received SETUP/Error Flow Interrupt
TXRDY_TRER	TX Packet Read/Transaction Error Interrupt
TX_COMPLT	Transmitted IN Data Complete Interrupt
RXRDY_TXKL	Received OUT Data Interrupt
ERR_OVFLW	Overflow Error Interrupt
MDATA_RX	MDATA Interrupt
DATAX_RX	DATAx Interrupt

Figure 38-19. UDPHS Interrupt Control Interface

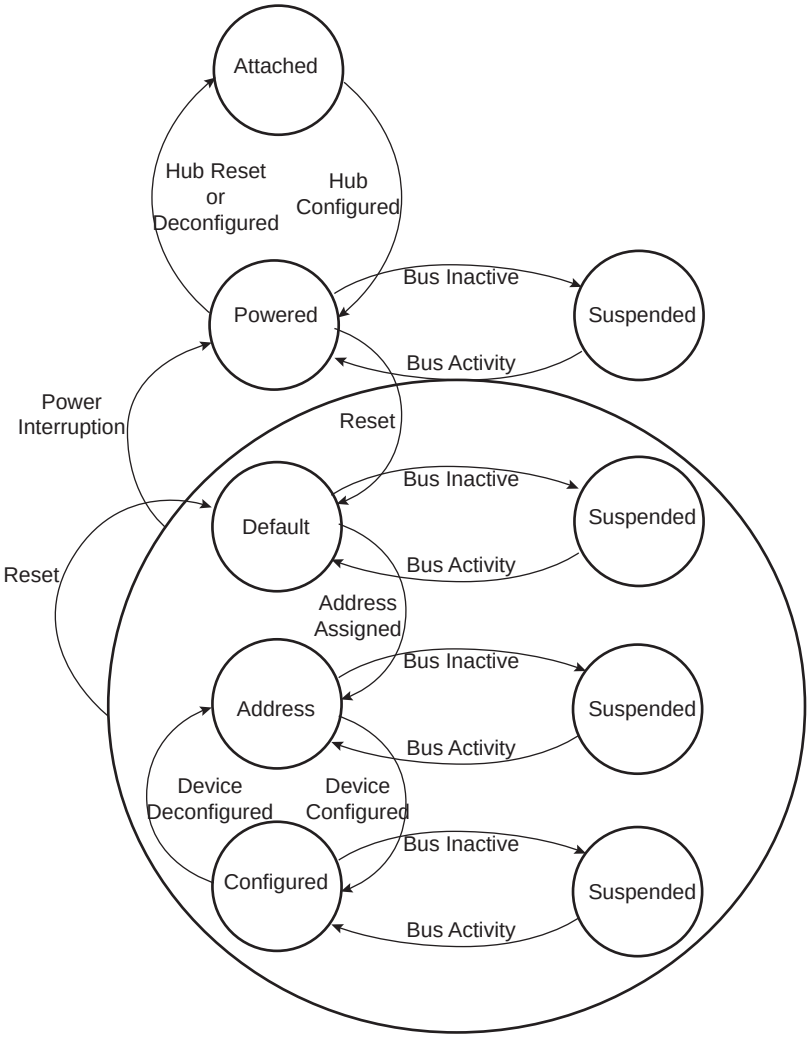


38.6.14 Power Modes

38.6.14.1 Controlling Device States

A USB device has several possible states. Refer to Chapter 9 (USB Device Framework) of the Universal Serial Bus Specification, Rev 2.0.

Figure 38-20. UDPHS Device State Diagram



Movement from one state to another depends on the USB bus state or on standard requests sent through control transactions via the default endpoint (endpoint 0).

After a period of bus inactivity, the USB device enters Suspend mode. Accepting Suspend/Resume requests from the USB host is mandatory. Constraints in Suspend mode are very strict for bus-powered applications; devices may not consume more than 500 µA on the USB bus.

While in Suspend mode, the host may wake up a device by sending a resume signal (bus activity) or a USB device may send a wakeup request to the host, e.g., waking up a PC by moving a USB mouse.

The wakeup feature is not mandatory for all devices and must be negotiated with the host.

38.6.14.2 Not Powered State

Self powered devices can detect 5V VBUS using a PIO. When the device is not connected to a host, device power consumption can be reduced by the DETACH bit in UDPHS_CTRL. Disabling the transceiver is automatically done. HSDM, HSDP, FSDP and FSDP lines are tied to GND pull-downs integrated in the hub downstream ports.

38.6.14.3 Entering Attached State

When no device is connected, the USB FSDP and FSDM signals are tied to GND by 15 KΩ pull-downs integrated in the hub downstream ports. When a device is attached to an hub downstream port, the device connects a 1.5 KΩ pull-up on FSDP. The USB bus line goes into IDLE state, FSDP is pulled up by the device 1.5 KΩ resistor to 3.3V and FSDM is pulled-down by the 15 KΩ resistor to GND of the host.

After pull-up connection, the device enters the powered state. The transceiver remains disabled until bus activity is detected.

In case of low power consumption need, the device can be stopped. When the device detects the VBUS, the software must enable the USB transceiver by enabling the EN_UDPHS bit in UDPHS_CTRL register.

The software can detach the pull-up by setting DETACH bit in UDPHS_CTRL register.

38.6.14.4 From Powered State to Default State (Reset)

After its connection to a USB host, the USB device waits for an end-of-bus reset. The unmasked flag ENDRESET is set in the UDPHS_IEN register and an interrupt is triggered.

Once the ENDRESET interrupt has been triggered, the device enters Default State. In this state, the UDPHS software must:

- Enable the default endpoint, setting the EPT_ENABL flag in the UDPHS_EPTCTLENB[0] register and, optionally, enabling the interrupt for endpoint 0 by writing 1 in EPT_0 of the UDPHS_IEN register. The enumeration then begins by a control transfer.
- Configure the Interrupt Mask Register which has been reset by the USB reset detection
- Enable the transceiver.

In this state, the EN_UDPHS bit in UDPHS_CTRL register must be enabled.

38.6.14.5 From Default State to Address State (Address Assigned)

After a Set Address standard device request, the USB host peripheral enters the address state.

Warning: before the device enters address state, it must achieve the Status IN transaction of the control transfer, i.e., the UDPHS device sets its new address once the TX_COMPLT flag in the UDPHS_EPTCTL[0] register has been received and cleared.

To move to address state, the driver software sets the DEV_ADDR field and the FADDR_EN flag in the UDPHS_CTRL register.

38.6.14.6 From Address State to Configured State (Device Configured)

Once a valid Set Configuration standard request has been received and acknowledged, the device enables endpoints corresponding to the current configuration. This is done by setting the BK_NUMBER, EPT_TYPE, EPT_DIR and EPT_SIZE fields in the UDPHS_EPTCFGx registers and enabling them by setting the EPT_ENABL flag in the UDPHS_EPTCTLENBx registers, and, optionally, enabling corresponding interrupts in the UDPHS_IEN register.

38.6.14.7 Entering Suspend State (Bus Activity)

When a Suspend (no bus activity on the USB bus) is detected, the DET_SUSPD signal in the UDPHS_STA register is set. This triggers an interrupt if the corresponding bit is set in the UDPHS_IEN register. This flag is cleared by writing to the UDPHS_CLRINT register. Then the device enters Suspend mode.

In this state bus powered devices must drain less than 500 µA from the 5V VBUS. As an example, the microcontroller switches to slow clock, disables the PLL and main oscillator, and goes into Idle mode. It may also switch off other devices on the board.

The UDPHS device peripheral clocks can be switched off. Resume event is asynchronously detected.

38.6.14.8 Receiving a Host Resume

In Suspend mode, a resume event on the USB bus line is detected asynchronously, transceiver and clocks disabled (however the pull-up should not be removed).

Once the resume is detected on the bus, the signal WAKE_UP in the UDPHS_INTSTA is set. It may generate an interrupt if the corresponding bit in the UDPHS_IEN register is set. This interrupt may be used to wake up the core, enable PLL and main oscillators and configure clocks.

38.6.14.9 Sending an External Resume

In Suspend State it is possible to wake up the host by sending an external resume.

The device waits at least 5 ms after being entered in Suspend State before sending an external resume.

The device must force a K state from 1 to 15 ms to resume the host.

38.6.15 Test Mode

A device must support the TEST_MODE feature when in the Default, Address or Configured High Speed device states.

TEST_MODE can be:

- Test_J
- Test_K
- Test_Packet
- Test_SEO_NAK

(See Section 38.7.11 “UDPHS Test Register” for definitions of each test mode.)

```
const char test_packet_buffer[] = {
  0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,0x00,                // JKJKJKJK * 9
  0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,0xAA,                     // JJKKJJKK * 8
  0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,0xEE,                     // JJKKJJKK * 8
  0xFE,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF,0xFF, //
JJJJJJJKKKKKKK * 8
  0x7F,0xBF,0xDF,0xEF,0xF7,0xFB,0xFD,                          // JJJJJJK * 8
  0xFC,0x7E,0xBF,0xDF,0xEF,0xF7,0xFB,0xFD,0x7E                 // {JKKKKKKK *
10}, JK
};
```

38.7 USB High Speed Device Port (UDPHS) User Interface

Table 38-6. Register Mapping

Offset	Register	Name	Access	Reset
0x00	UDPHS Control Register	UDPHS_CTRL	Read/Write	0x0000_0200
0x04	UDPHS Frame Number Register	UDPHS_FNUM	Read-only	0x0000_0000
0x08–0x0C	Reserved	–	–	–
0x10	UDPHS Interrupt Enable Register	UDPHS_IEN	Read/Write	0x0000_0010
0x14	UDPHS Interrupt Status Register	UDPHS_INTSTA	Read-only	0x0000_0000
0x18	UDPHS Clear Interrupt Register	UDPHS_CLRINT	Write-only	–
0x1C	UDPHS Endpoints Reset Register	UDPHS_EPTRST	Write-only	–
0x20–0xCC	Reserved	–	–	–
0xD0	UDPHS Test SOF Counter Register	UDPHS_TSTSOFCNT	Read/Write	0x0000_0000
0xD4	UDPHS Test A Counter Register	UDPHS_TSTCNTA	Read/Write	0x0000_0000
0xD8	UDPHS Test B Counter Register	UDPHS_TSTCNTB	Read/Write	0x0000_0000
0xDC	UDPHS Test Mode Register	UDPHS_TSTMODEREG	Read/Write	0x0000_0000
0xE0	UDPHS Test Register	UDPHS_TST	Read/Write	0x0000_0000
0xE4–0xFC	Reserved	–	–	–
0x100 + endpoint * 0x20 + 0x00	UDPHS Endpoint Configuration Register	UDPHS_EPTCFG	Read/Write	0x0000_0000
0x100 + endpoint * 0x20 + 0x04	UDPHS Endpoint Control Enable Register	UDPHS_EPTCTLENB	Write-only	–
0x100 + endpoint * 0x20 + 0x08	UDPHS Endpoint Control Disable Register	UDPHS_EPTCTLDIS	Write-only	–
0x100 + endpoint * 0x20 + 0x0C	UDPHS Endpoint Control Register	UDPHS_EPTCTL	Read-only	0x0000_0000(1)
0x100 + endpoint * 0x20 + 0x10	Reserved (for endpoint)	–	–	–
0x100 + endpoint * 0x20 + 0x14	UDPHS Endpoint Set Status Register	UDPHS_EPTSETSTA	Write-only	–
0x100 + endpoint * 0x20 + 0x18	UDPHS Endpoint Clear Status Register	UDPHS_EPTCLRSTA	Write-only	–
0x100 + endpoint * 0x20 + 0x1C	UDPHS Endpoint Status Register	UDPHS_EPTSTA	Read-only	0x0000_0040
0x120–0x2FC	UDPHS Endpoint1 to 15 (2) Registers	–	–	–
0x300 + channel * 0x10 + 0x00	UDPHS DMA Next Descriptor Address Register	UDPHS_DMANXTDSC	Read/Write	0x0000_0000
0x300 + channel * 0x10 + 0x04	UDPHS DMA Channel Address Register	UDPHS_DMAADDRESS	Read/Write	0x0000_0000
0x300 + channel * 0x10 + 0x08	UDPHS DMA Channel Control Register	UDPHS_DMACONTROL	Read/Write	0x0000_0000
0x300 + channel * 0x10 + 0x0C	UDPHS DMA Channel Status Register	UDPHS_DMASTATUS	Read/Write	0x0000_0000
0x310–0x36C	DMA Channel1 to 6 (3) Registers	–	–	–

Notes:
1. The reset value for UDPHS_EPTCTL0 is 0x0000_0001.
2. The addresses for the UDPHS Endpoint registers shown here are for UDPHS Endpoint0. The structure of this group of registers is repeated successively for each endpoint according to the sequence of endpoint registers located between 0x120 and 0x2FC.
3. The DMA channel index refers to the corresponding EP number. When no DMA channel is assigned to one EP, the associated registers are reserved. This is the case for EP0, so DMA Channel 0 registers are reserved.

38.7.1 UDPHS Control Register

Name: UDPHS_CTRL

Address: 0xFC02C000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	PULLD_DIS	REWAKEUP	DETACH	EN_UDPHS
7	6	5	4	3	2	1	0
FADDR_EN	DEV_ADDR						

- **DEV_ADDR: UDPHS Address (cleared upon USB reset)**

This field contains the default address (0) after powerup or UDPHS bus reset (read), or it is written with the value set by a SET_ADDRESS request received by the device firmware (write).

- **FADDR_EN: Function Address Enable (cleared upon USB reset)**

0: Device is not in address state (read), or only the default function address is used (write).

1: Device is in address state (read), or this bit is set by the device firmware after a successful status phase of a SET_ADDRESS transaction (write). When set, the only address accepted by the UDPHS controller is the one stored in the UDPHS Address field. It will not be cleared afterwards by the device firmware. It is cleared by hardware on hardware reset, or when UDPHS bus reset is received.

- **EN_UDPHS: UDPHS Enable**

0: UDPHS is disabled (read), or this bit disables and resets the UDPHS controller (write). Switch the host to UTMI.

1: UDPHS is enabled (read), or this bit enables the UDPHS controller (write). Switch the host to UTMI.

- **DETACH: Detach Command**

0: UDPHS is attached (read), or this bit pulls up the DP line (attach command) (write).

1: UDPHS is detached, UTMI transceiver is suspended (read), or this bit simulates a detach on the UDPHS line and forces the UTMI transceiver into suspend state (Suspend M = 0) (write).

See PULLD_DIS description below.

- **REWAKEUP: Send Remote Wakeup (cleared upon USB reset)**

0: Remote Wakeup is disabled (read), or this bit has no effect (write).

1: Remote Wakeup is enabled (read), or this bit forces an external interrupt on the UDPHS controller for Remote Wakeup purposes.

An Upstream Resume is sent only after the UDPHS bus has been in SUSPEND state for at least 5 ms.

This bit is automatically cleared by hardware at the end of the Upstream Resume.

- **PULLD_DIS: Pull-Down Disable (cleared upon USB reset)**

When set, there is no pull-down on DP & DM. (DM Pull-Down = DP Pull-Down = 0).

Note: If the DETACH bit is also set, device DP & DM are left in high impedance state.

(See DETACH description above.)

DETACH	PULLD_DIS	DP	DM	Condition
0	0	Pull up	Pull down	Not recommended
0	1	Pull up	High impedance state	VBUS present
1	0	Pull down	Pull down	No VBUS
1	1	High impedance state	High impedance state	VBUS present & software disconnect

38.7.2 UDPHS Frame Number Register

Name: UDPHS_FNUM

Address: 0xFC02C004

Access: Read-only

31	30	29	28	27	26	25	24
FNUM_ERR	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	FRAME_NUMBER					
7	6	5	4	3	2	1	0
FRAME_NUMBER					MICRO_FRAME_NUM		

- **MICRO_FRAME_NUM: Microframe Number (cleared upon USB reset)**

Number of the received microframe (0 to 7) in one frame.This field is reset at the beginning of each new frame (1 ms). One microframe is received each 125 microseconds (1 ms/8).

- **FRAME_NUMBER: Frame Number as defined in the Packet Field Formats (cleared upon USB reset)**

This field is provided in the last received SOF packet (see INT_SOF in the UDPHS Interrupt Status Register).

- **FNUM_ERR: Frame Number CRC Error (cleared upon USB reset)**

This bit is set by hardware when a corrupted Frame Number in Start of Frame packet (or Micro SOF) is received. This bit and the INT_SOF (or MICRO_SOF) interrupt are updated at the same time.

38.7.3 UDPHS Interrupt Enable Register

Name: UDPHS_IEN

Address: 0xFC02C010

Access: Read/Write

31	30	29	28	27	26	25	24
DMA_7	DMA_6	DMA_5	DMA_4	DMA_3	DMA_2	DMA_1	–
23	22	21	20	19	18	17	16
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
15	14	13	12	11	10	9	8
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0
7	6	5	4	3	2	1	0
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	–

- **DET_SUSPD: Suspend Interrupt Enable (cleared upon USB reset)**

0: Disable Suspend Interrupt.

1: Enable Suspend Interrupt.

- **MICRO_SOF: Micro-SOF Interrupt Enable (cleared upon USB reset)**

0: Disable Micro-SOF Interrupt.

1: Enable Micro-SOF Interrupt.

- **INT_SOF: SOF Interrupt Enable (cleared upon USB reset)**

0: Disable SOF Interrupt.

1: Enable SOF Interrupt.

- **ENDRESET: End Of Reset Interrupt Enable (cleared upon USB reset)**

0: Disable End Of Reset Interrupt.

1: Enable End Of Reset Interrupt. Automatically enabled after USB reset.

- **WAKE_UP: Wake Up CPU Interrupt Enable (cleared upon USB reset)**

0: Disable Wake Up CPU Interrupt.

1: Enable Wake Up CPU Interrupt.

- **ENDOFRSM: End Of Resume Interrupt Enable (cleared upon USB reset)**

0: Disable Resume Interrupt.

1: Enable Resume Interrupt.

- **UPSTR_RES: Upstream Resume Interrupt Enable (cleared upon USB reset)**

0: Disable Upstream Resume Interrupt.

1: Enable Upstream Resume Interrupt.

- **EPT_x: Endpoint x Interrupt Enable (cleared upon USB reset)**

0: Disable the interrupts for this endpoint.

1: Enable the interrupts for this endpoint.

- **DMA_x: DMA Channel x Interrupt Enable (cleared upon USB reset)**

0: Disable the interrupts for this channel.

1: Enable the interrupts for this channel.

38.7.4 UDPHS Interrupt Status Register

Name: UDPHS_INTSTA

Address: 0xFC02C014

Access: Read-only

31	30	29	28	27	26	25	24
DMA_7	DMA_6	DMA_5	DMA_4	DMA_3	DMA_2	DMA_1	–
23	22	21	20	19	18	17	16
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
15	14	13	12	11	10	9	8
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0
7	6	5	4	3	2	1	0
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	SPEED

- **SPEED: Speed Status**

0: Reset by hardware when the hardware is in Full Speed mode.

1: Set by hardware when the hardware is in High Speed mode.

- **DET_SUSPD: Suspend Interrupt**

0: Cleared by setting the DET_SUSPD bit in UDPHS_CLRINT register.

1: Set by hardware when a UDPHS Suspend (Idle bus for three frame periods, a J state for 3 ms) is detected. This triggers a UDPHS interrupt when the DET_SUSPD bit is set in UDPHS_IEN register.

- **MICRO_SOF: Micro Start Of Frame Interrupt**

0: Cleared by setting the MICRO_SOF bit in UDPHS_CLRINT register.

1: Set by hardware when an UDPHS micro start of frame PID (SOF) has been detected (every 125 us) or synthesized by the macro. This triggers a UDPHS interrupt when the MICRO_SOF bit is set in UDPHS_IEN. In case of detected SOF, the MICRO_FRAME_NUM field in UDPHS_FNUM register is incremented and the FRAME_NUMBER field does not change.

Note: The Micro Start Of Frame Interrupt (MICRO_SOF), and the Start Of Frame Interrupt (INT_SOF) are not generated at the same time.

- **INT_SOF: Start Of Frame Interrupt**

0: Cleared by setting the INT_SOF bit in UDPHS_CLRINT.

1: Set by hardware when an UDPHS Start Of Frame PID (SOF) has been detected (every 1 ms) or synthesized by the macro. This triggers a UDPHS interrupt when the INT_SOF bit is set in UDPHS_IEN register. In case of detected SOF, in High Speed mode, the MICRO_FRAME_NUMBER field is cleared in UDPHS_FNUM register and the FRAME_NUMBER field is updated.

- **ENDRESET: End Of Reset Interrupt**

0: Cleared by setting the ENDRESET bit in UDPHS_CLRINT.

1: Set by hardware when an End Of Reset has been detected by the UDPHS controller. This triggers a UDPHS interrupt when the ENDRESET bit is set in UDPHS_IEN.

- **WAKE_UP: Wake Up CPU Interrupt**

0: Cleared by setting the WAKE_UP bit in UDPHS_CLRINT.

1: Set by hardware when the UDPHS controller is in SUSPEND state and is re-activated by a filtered non-idle signal from the UDPHS line (not by an upstream resume). This triggers a UDPHS interrupt when the WAKE_UP bit is set in UDPHS_IEN register. When receiving this interrupt, the user has to enable the device controller clock prior to operation.

Note: this interrupt is generated even if the device controller clock is disabled.

- **ENDOFRSM: End Of Resume Interrupt**

0: Cleared by setting the ENDOFRSM bit in UDPHS_CLRINT.

1: Set by hardware when the UDPHS controller detects a good end of resume signal initiated by the host. This triggers a UDPHS interrupt when the ENDOFRSM bit is set in UDPHS_IEN.

- **UPSTR_RES: Upstream Resume Interrupt**

0: Cleared by setting the UPSTR_RES bit in UDPHS_CLRINT.

1: Set by hardware when the UDPHS controller is sending a resume signal called “upstream resume”. This triggers a UDPHS interrupt when the UPSTR_RES bit is set in UDPHS_IEN.

- **EPT_x: Endpoint x Interrupt (cleared upon USB reset)**

0: Reset when the UDPHS_EPTSTAx interrupt source is cleared.

1: Set by hardware when an interrupt is triggered by the UDPHS_EPTSTAx register and this endpoint interrupt is enabled by the EPT_x bit in UDPHS_IEN.

- **DMA_x: DMA Channel x Interrupt**

0: Reset when the UDPHS_DMASTATUSx interrupt source is cleared.

1: Set by hardware when an interrupt is triggered by the DMA Channelx and this endpoint interrupt is enabled by the DMA_x bit in UDPHS_IEN.

38.7.5 UDPHS Clear Interrupt Register

Name: UDPHS_CLRINT

Address: 0xFC02C018

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
UPSTR_RES	ENDOFRSM	WAKE_UP	ENDRESET	INT_SOF	MICRO_SOF	DET_SUSPD	–

- **DET_SUSPD: Suspend Interrupt Clear**

0: No effect.

1: Clear the DET_SUSPD bit in UDPHS_INTSTA.

- **MICRO_SOF: Micro Start Of Frame Interrupt Clear**

0: No effect.

1: Clear the MICRO_SOF bit in UDPHS_INTSTA.

- **INT_SOF: Start Of Frame Interrupt Clear**

0: No effect.

1: Clear the INT_SOF bit in UDPHS_INTSTA.

- **ENDRESET: End Of Reset Interrupt Clear**

0: No effect.

1: Clear the ENDRESET bit in UDPHS_INTSTA.

- **WAKE_UP: Wake Up CPU Interrupt Clear**

0: No effect.

1: Clear the WAKE_UP bit in UDPHS_INTSTA.

- **ENDOFRSM: End Of Resume Interrupt Clear**

0: No effect.

1: Clear the ENDOFRSM bit in UDPHS_INTSTA.

- **UPSTR_RES: Upstream Resume Interrupt Clear**

0: No effect.

1: Clear the UPSTR_RES bit in UDPHS_INTSTA.

38.7.6 UDPHS Endpoints Reset Register

Name: UDPHS_EPTRST

Address: 0xFC02C01C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
EPT_15	EPT_14	EPT_13	EPT_12	EPT_11	EPT_10	EPT_9	EPT_8
7	6	5	4	3	2	1	0
EPT_7	EPT_6	EPT_5	EPT_4	EPT_3	EPT_2	EPT_1	EPT_0

- **EPT_x: Endpoint x Reset**

0: No effect.

1: Reset the Endpointx state.

Setting this bit clears all bits in the Endpoint status UDPHS_EPTSTAx register except the TOGGLESQ_STA field.

38.7.7 UDPHS Test SOF Counter Register

Name: UDPHS_TSTSOFCNT

Address: 0xFC02C0D0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
SOFCTLOAD	SOFCNTMAX						

- **SOFCNTMAX: SOF Counter Max Value**
- **SOFCTLOAD: SOF Counter Load**

38.7.8 UDPHS Test A Counter Register

Name: UDPHS_TSTCNTA

Address: 0xFC02C0D4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
CNTALOAD	CNTAMAX						

7	6	5	4	3	2	1	0
CNTAMAX							

- **CNTALOAD: A Counter Load**
- **CNTAMAX: A Counter Max Value**

38.7.9 UDPHS Test B Counter Register

Name: UDPHS_TSTCNTB

Address: 0xFC02C0D8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
CNTBLOAD	CNTBMAX						

7	6	5	4	3	2	1	0
CNTBMAX							

- **CNTBLOAD: B Counter Load**
- **CNTBMAX: B Counter Max Value**

38.7.10 UDPHS Test Mode Register

Name: UDPHS_TSTMODEREG

Address: 0xFC02C0DC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	TSTMODE					–

- **TSTMODE: UDPHS Core TestModeReg**

38.7.11 UDPHS Test Register

Name: UDPHS_TST

Address: 0xFC02C0E0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	OPMODE2	TST_PKT	TST_K	TST_J	SPEED_CFG	

- **SPEED_CFG: Speed Configuration**

Value	Name	Description
0	NORMAL	Normal mode: The macro is in Full Speed mode, ready to make a High Speed identification, if the host supports it and then to automatically switch to High Speed mode.
1	–	Reserved
2	HIGH_SPEED	Force High Speed: Set this value to force the hardware to work in High Speed mode. Only for debug or test purpose.
3	FULL_SPEED	Force Full Speed: Set this value to force the hardware to work only in Full Speed mode. In this configuration, the macro will not respond to a High Speed reset handshake.

- **TST_J: Test J Mode**

0: No effect.

1: Set to send the J state on the UDPHS line. This enables the testing of the high output drive level on the D+ line.

- **TST_K: Test K Mode**

0: No effect.

1: Set to send the K state on the UDPHS line. This enables the testing of the high output drive level on the D- line.

- **TST_PKT: Test Packet Mode**

0: No effect.

1: Set to repetitively transmit the packet stored in the current bank. This enables the testing of rise and fall times, eye patterns, jitter, and any other dynamic waveform specifications.

- **OPMODE2: OpMode2**

0: No effect.

1: Set to force the OpMode signal (UTMI interface) to “10”, to disable the bit-stuffing and the NRZI encoding.

Note: For the Test mode, Test_SE0_NAK (see Universal Serial Bus Specification, Revision 2.0: 7.1.20, Test Mode Support). Force the device in High Speed mode, and configure a bulk-type endpoint. Do not fill this endpoint for sending NAK to the host. Upon command, a port’s transceiver must enter the High Speed Receive mode and remain in that mode until the exit action is taken. This enables the testing of output impedance, low level output voltage and loading characteristics. In addition, while in this mode, upstream facing ports (and only upstream facing ports) must respond to any IN token packet with a NAK handshake (only if the packet CRC is determined to be correct) within the normal allowed device response time. This enables testing of the device squelch level circuitry and, additionally, provides a general purpose stimulus/response test for basic functional testing.

38.7.12 UDPHS Endpoint Configuration Register

Name: UDPHS_EPTCFGx [x=0..15]

Address: 0xFC02C100 [0], 0xFC02C120 [1], 0xFC02C140 [2], 0xFC02C160 [3], 0xFC02C180 [4], 0xFC02C1A0 [5], 0xFC02C1C0 [6], 0xFC02C1E0 [7], 0xFC02C200 [8], 0xFC02C220 [9], 0xFC02C240 [10], 0xFC02C260 [11], 0xFC02C280 [12], 0xFC02C2A0 [13], 0xFC02C2C0 [14], 0xFC02C2E0 [15]

Access: Read/Write

31	30	29	28	27	26	25	24
EPT_MAPD	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	NB_TRANS	
7	6	5	4	3	2	1	0
BK_NUMBER		EPT_TYPE		EPT_DIR	EPT_SIZE		

- **EPT_SIZE: Endpoint Size (cleared upon USB reset)**

Set this field according to the endpoint size[1] in bytes (see Section 38.6.6 “Endpoint Configuration”).

Value	Name	Description
0	8	8 bytes
1	16	16 bytes
2	32	32 bytes
3	64	64 bytes
4	128	128 bytes
5	256	256 bytes
6	512	512 bytes
7	1024	1024 bytes

Note: 1. 1024 bytes is only for isochronous endpoint.

- **EPT_DIR: Endpoint Direction (cleared upon USB reset)**

0: Clear this bit to configure OUT direction for Bulk, Interrupt and Isochronous endpoints.

1: Set this bit to configure IN direction for Bulk, Interrupt and Isochronous endpoints.

For Control endpoints this bit has no effect and should be left at zero.

- **EPT_TYPE: Endpoint Type (cleared upon USB reset)**

Set this field according to the endpoint type (see Section 38.6.6 “Endpoint Configuration”).

(Endpoint 0 should always be configured as control)

Value	Name	Description
0	CTRL8	Control endpoint
1	ISO	Isochronous endpoint
2	BULK	Bulk endpoint
3	INT	Interrupt endpoint

- **BK_NUMBER: Number of Banks (cleared upon USB reset)**

Set this field according to the endpoint's number of banks (see Section 38.6.6 "Endpoint Configuration").

Value	Name	Description
0	0	Zero bank, the endpoint is not mapped in memory
1	1	One bank (bank 0)
2	2	Double bank (Ping-Pong: bank0/bank1)
3	3	Triple bank (bank0/bank1/bank2)

- **NB_TRANS: Number Of Transaction per Microframe (cleared upon USB reset)**

The Number of transactions per microframe is set by software.

Note: Meaningful for high bandwidth isochronous endpoint only.

- **EPT_MAPD: Endpoint Mapped (cleared upon USB reset)**

0: The user should reprogram the register with correct values.

1: Set by hardware when the endpoint size (EPT_SIZE) and the number of banks (BK_NUMBER) are correct regarding:

  – The FIFO max capacity (FIFO_MAX_SIZE in UDPHS_IPFEATURES register)
  – The number of endpoints/banks already allocated
  – The number of allowed banks for this endpoint

38.7.13 UDPHS Endpoint Control Enable Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCTLENBx [x=0..15]

Address: 0xFC02C104 [0], 0xFC02C124 [1], 0xFC02C144 [2], 0xFC02C164 [3], 0xFC02C184 [4], 0xFC02C1A4 [5], 0xFC02C1C4 [6], 0xFC02C1E4 [7], 0xFC02C204 [8], 0xFC02C224 [9], 0xFC02C244 [10], 0xFC02C264 [11], 0xFC02C284 [12], 0xFC02C2A4 [13], 0xFC02C2C4 [14], 0xFC02C2E4 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
–	–	–	NYET_DIS	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)” .

- **EPT_ENABL: Endpoint Enable**

0: No effect.

1: Enable endpoint according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enable**

0: No effect.

1: Enable this bit to automatically validate the current packet and switch to the next bank for both IN and OUT transfers.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled.

- **NYET_DIS: NYET Disable (Only for High Speed Bulk OUT endpoints)**

0: No effect.

1: Forces an ACK response to the next High Speed Bulk OUT transfer instead of a NYET response.

- **ERR_OVFLW: Overflow Error Interrupt Enable**

0: No effect.

1: Enable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Enable**

0: No effect.

1: Enable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enable**

0: No effect.

1: Enable Transmitted IN Data Complete Interrupt.

- **TXRDY: TX Packet Ready Interrupt Enable**

0: No effect.

1: Enable TX Packet Ready/Transaction Error Interrupt.

- **RX_SETUP: Received SETUP**

0: No effect.

1: Enable RX_SETUP Interrupt.

- **STALL_SNT: Stall Sent Interrupt Enable**

0: No effect.

1: Enable Stall Sent Interrupt.

- **NAK_IN: NAKIN Interrupt Enable**

0: No effect.

1: Enable NAKIN Interrupt.

- **NAK_OUT: NAKOUT Interrupt Enable**

0: No effect.

1: Enable NAKOUT Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Enable**

0: No effect.

1: Enable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Send/Short Packet Interrupt Enable**

For OUT endpoints:

0: No effect.

1: Enable Short Packet Interrupt.

For IN endpoints: Guarantees short packet at end of DMA Transfer if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTOVALID bits are also set.

38.7.14 UDPHS Endpoint Control Enable Register (Isochronous Endpoints)

Name: UDPHS_EPTCTLENBx [x=0..15] (ISOENDPT)

Address: 0xFC02C104 [0], 0xFC02C124 [1], 0xFC02C144 [2], 0xFC02C164 [3], 0xFC02C184 [4], 0xFC02C1A4 [5], 0xFC02C1C4 [6], 0xFC02C1E4 [7], 0xFC02C204 [8], 0xFC02C224 [9], 0xFC02C244 [10], 0xFC02C264 [11], 0xFC02C284 [12], 0xFC02C2A4 [13], 0xFC02C2C4 [14], 0xFC02C2E4 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
MDATA_RX	DATAX_RX	–	–	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Control Register (Isochronous Endpoint)” .

- **EPT_ENABL: Endpoint Enable**

0: No effect.

1: Enable endpoint according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enable**

0: No effect.

1: Enable this bit to automatically validate the current packet and switch to the next bank for both IN and OUT transfers.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled.

- **DATAX_RX: DATAx Interrupt Enable (Only for high bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Enable DATAx Interrupt.

- **MDATA_RX: MDATA Interrupt Enable (Only for high bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Enable MDATA Interrupt.

- **ERR_OVFLW: Overflow Error Interrupt Enable**

0: No effect.

1: Enable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Enable**

0: No effect.

1: Enable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enable**

0: No effect.

1: Enable Transmitted IN Data Complete Interrupt.

- **TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Enable**

0: No effect.

1: Enable TX Packet Ready/Transaction Error Interrupt.

- **ERR_FL_ISO: Error Flow Interrupt Enable**

0: No effect.

1: Enable Error Flow ISO Interrupt.

- **ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Enable**

0: No effect.

1: Enable Error CRC ISO/Error Number of Transaction Interrupt.

- **ERR_FLUSH: Bank Flush Error Interrupt Enable**

0: No effect.

1: Enable Bank Flush Error Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Enable**

0: No effect.

1: Enable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Send/Short Packet Interrupt Enable**

For OUT endpoints:

0: No effect.

1: Enable Short Packet Interrupt.

For IN endpoints: Guarantees short packet at end of DMA Transfer if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTOVALID bits are also set.

38.7.15 UDPHS Endpoint Control Disable Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCTLDISx [x=0..15]

Address: 0xFC02C108 [0], 0xFC02C128 [1], 0xFC02C148 [2], 0xFC02C168 [3], 0xFC02C188 [4], 0xFC02C1A8 [5], 0xFC02C1C8 [6], 0xFC02C1E8 [7], 0xFC02C208 [8], 0xFC02C228 [9], 0xFC02C248 [10], 0xFC02C268 [11], 0xFC02C288 [12], 0xFC02C2A8 [13], 0xFC02C2C8 [14], 0xFC02C2E8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
–	–	–	NYET_DIS	INTDIS_DMA	–	AUTO_VALID	EPT_DISABL

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)” .

- **EPT_DISABL: Endpoint Disable**

0: No effect.

1: Disable endpoint.

- **AUTO_VALID: Packet Auto-Valid Disable**

0: No effect.

1: Disable this bit to not automatically validate the current packet.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: Disable the “Interrupts Disable DMA”.

- **NYET_DIS: NYET Enable (Only for High Speed Bulk OUT endpoints)**

0: No effect.

1: Let the hardware handle the handshake response for the High Speed Bulk OUT transfer.

- **ERR_OVFLW: Overflow Error Interrupt Disable**

0: No effect.

1: Disable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Disable**

0: No effect.

1: Disable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Disable**

0: No effect.

1: Disable Transmitted IN Data Complete Interrupt.

- **TXRDY: TX Packet Ready Interrupt Disable**

0: No effect.

1: Disable TX Packet Ready/Transaction Error Interrupt.

- **RX_SETUP: Received SETUP Interrupt Disable**

0: No effect.

1: Disable RX_SETUP Interrupt.

- **STALL_SNT: Stall Sent Interrupt Disable**

0: No effect.

1: Disable Stall Sent Interrupt.

- **NAK_IN: NAKIN Interrupt Disable**

0: No effect.

1: Disable NAKIN Interrupt.

- **NAK_OUT: NAKOUT Interrupt Disable**

0: No effect.

1: Disable NAKOUT Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Disable**

0: No effect.

1: Disable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Interrupt Disable**

For OUT endpoints:

0: No effect.

1: Disable Short Packet Interrupt.

For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

38.7.16 UDPHS Endpoint Control Disable Register (Isochronous Endpoint)

Name: UDPHS_EPTCTLDISx [x=0..15] (ISOENDPT)

Address: 0xFC02C108 [0], 0xFC02C128 [1], 0xFC02C148 [2], 0xFC02C168 [3], 0xFC02C188 [4], 0xFC02C1A8 [5], 0xFC02C1C8 [6], 0xFC02C1E8 [7], 0xFC02C208 [8], 0xFC02C228 [9], 0xFC02C248 [10], 0xFC02C268 [11], 0xFC02C288 [12], 0xFC02C2A8 [13], 0xFC02C2C8 [14], 0xFC02C2E8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
MDATA_RX	DATAX_RX	–	–	INTDIS_DMA	–	AUTO_VALID	EPT_DISABL

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Control Register (Isochronous Endpoint)” .

- **EPT_DISABL: Endpoint Disable**

0: No effect.

1: Disable endpoint.

- **AUTO_VALID: Packet Auto-Valid Disable**

0: No effect.

1: Disable this bit to not automatically validate the current packet.

- **INTDIS_DMA: Interrupts Disable DMA**

0: No effect.

1: Disable the “Interrupts Disable DMA”.

- **DATAX_RX: DATAx Interrupt Disable (Only for High Bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Disable DATAx Interrupt.

- **MDATA_RX: MDATA Interrupt Disable (Only for High Bandwidth Isochronous OUT endpoints)**

0: No effect.

1: Disable MDATA Interrupt.

- **ERR_OVFLW: Overflow Error Interrupt Disable**

0: No effect.

1: Disable Overflow Error Interrupt.

- **RXRDY_TXKL: Received OUT Data Interrupt Disable**

0: No effect.

1: Disable Received OUT Data Interrupt.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Disable**

0: No effect.

1: Disable Transmitted IN Data Complete Interrupt.

- **TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Disable**

0: No effect.

1: Disable TX Packet Ready/Transaction Error Interrupt.

- **ERR_FL_ISO: Error Flow Interrupt Disable**

0: No effect.

1: Disable Error Flow ISO Interrupt.

- **ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Disable**

0: No effect.

1: Disable Error CRC ISO/Error Number of Transaction Interrupt.

- **ERR_FLUSH: bank flush error Interrupt Disable**

0: No effect.

1: Disable Bank Flush Error Interrupt.

- **BUSY_BANK: Busy Bank Interrupt Disable**

0: No effect.

1: Disable Busy Bank Interrupt.

- **SHRT_PCKT: Short Packet Interrupt Disable**

For OUT endpoints:

0: No effect.

1: Disable Short Packet Interrupt.

For IN endpoints: Never automatically add a zero length packet at end of DMA transfer.

38.7.17 UDPHS Endpoint Control Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCTLx [x=0..15]

Address: 0xFC02C10C [0], 0xFC02C12C [1], 0xFC02C14C [2], 0xFC02C16C [3], 0xFC02C18C [4], 0xFC02C1AC [5], 0xFC02C1CC [6], 0xFC02C1EC [7], 0xFC02C20C [8], 0xFC02C22C [9], 0xFC02C24C [10], 0xFC02C26C [11], 0xFC02C28C [12], 0xFC02C2AC [13], 0xFC02C2CC [14], 0xFC02C2EC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
–	–	–	NYET_DIS	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register” .

- **EPT_ENABL: Endpoint Enable (cleared upon USB reset)**

0: The endpoint is disabled according to the device configuration. Endpoint 0 should always be enabled after a hardware or UDPHS bus reset and participate in the device configuration.

1: The endpoint is enabled according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enabled (Not for CONTROL Endpoints) (cleared upon USB reset)**

Set this bit to automatically validate the current packet and switch to the next bank for both IN and OUT endpoints.

For IN Transfer:

If this bit is set, the UDPHS_EPTSTAx register TXRDY bit is set automatically when the current bank is full and at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set.

The user may still set the UDPHS_EPTSTAx register TXRDY bit if the current bank is not full, unless the user needs to send a Zero Length Packet by software.

For OUT Transfer:

If this bit is set, the UDPHS_EPTSTAx register RXRDY_TXKL bit is automatically reset for the current bank when the last packet byte has been read from the bank FIFO or at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set. For example, to truncate a padded data packet when the actual data transfer size is reached.

The user may still clear the UDPHS_EPTSTAx register RXRDY_TXKL bit, for example, after completing a DMA buffer by software if UDPHS_DMACONTROLx register END_B_EN bit was disabled or in order to cancel the read of the remaining data bank(s).

- **INTDIS_DMA: Interrupt Disables DMA (cleared upon USB reset)**

If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled regardless of the UDPHS_IEN register EPT_x bit for this endpoint. Then, the firmware will have to clear or disable the interrupt source or clear this bit if transfer completion is needed.

If the exception raised is associated with the new system bank packet, then the previous DMA packet transfer is normally completed, but the new DMA packet transfer is not started (not requested).

If the exception raised is not associated to a new system bank packet (NAK_IN, NAK_OUT, etc.), then the request cancellation may happen at any time and may immediately stop the current DMA transfer.

This may be used, for example, to identify or prevent an erroneous packet to be transferred into a buffer or to complete a DMA buffer by software after reception of a short packet.

- **NYET_DIS: NYET Disable (Only for High Speed Bulk OUT Endpoints) (cleared upon USB reset)**

0: Lets the hardware handle the handshake response for the High Speed Bulk OUT transfer.

1: Forces an ACK response to the next High Speed Bulk OUT transfer instead of a NYET response.

Note: According to the *Universal Serial Bus Specification, Rev 2.0* (8.5.1.1 NAK Responses to OUT/DATA During PING Protocol), a NAK response to an HS Bulk OUT transfer is expected to be an unusual occurrence.

- **ERR_OVFLW: Overflow Error Interrupt Enabled (cleared upon USB reset)**

0: Overflow Error Interrupt is masked.

1: Overflow Error Interrupt is enabled.

- **RXRDY_TXKL: Received OUT Data Interrupt Enabled (cleared upon USB reset)**

0: Received OUT Data Interrupt is masked.

1: Received OUT Data Interrupt is enabled.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enabled (cleared upon USB reset)**

0: Transmitted IN Data Complete Interrupt is masked.

1: Transmitted IN Data Complete Interrupt is enabled.

- **TXRDY: TX Packet Ready Interrupt Enabled (cleared upon USB reset)**

0: TX Packet Ready Interrupt is masked.

1: TX Packet Ready Interrupt is enabled.

Caution: Interrupt source is active as long as the corresponding UDPHS_EPTSTAx register TXRDY flag remains low. If there are no more banks available for transmitting after the software has set UDPHS_EPTSTAx/TXRDY for the last transmit packet, then the interrupt source remains inactive until the first bank becomes free again to transmit at UDPHS_EPTSTAx/TXRDY hardware clear.

- **RX_SETUP: Received SETUP Interrupt Enabled (cleared upon USB reset)**

0: Received SETUP is masked.

1: Received SETUP is enabled.

- **STALL_SNT: Stall Sent Interrupt Enabled (cleared upon USB reset)**

0: Stall Sent Interrupt is masked.

1: Stall Sent Interrupt is enabled.

- **NAK_IN: NAKIN Interrupt Enabled (cleared upon USB reset)**

0: NAKIN Interrupt is masked.

1: NAKIN Interrupt is enabled.

- **NAK_OUT: NAKOUT Interrupt Enabled (cleared upon USB reset)**

0: NAKOUT Interrupt is masked.

1: NAKOUT Interrupt is enabled.

- **BUSY_BANK: Busy Bank Interrupt Enabled (cleared upon USB reset)**

0: BUSY_BANK Interrupt is masked.

1: BUSY_BANK Interrupt is enabled.

For OUT endpoints: an interrupt is sent when all banks are busy.

For IN endpoints: an interrupt is sent when all banks are free.

- **SHRT_PCKT: Short Packet Interrupt Enabled (cleared upon USB reset)**

For OUT endpoints: send an Interrupt when a Short Packet has been received.

0: Short Packet Interrupt is masked.

1: Short Packet Interrupt is enabled.

For IN endpoints: a Short Packet transmission is guaranteed upon end of the DMA Transfer, thus signaling a BULK or INTERRUPT end of transfer, but only if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTO_VALID bits are also set.

38.7.18 UDPHS Endpoint Control Register (Isochronous Endpoint)

Name: UDPHS_EPTCTLx [x=0..15] (ISOENDPT)

Address: 0xFC02C10C [0], 0xFC02C12C [1], 0xFC02C14C [2], 0xFC02C16C [3], 0xFC02C18C [4], 0xFC02C1AC [5], 0xFC02C1CC [6], 0xFC02C1EC [7], 0xFC02C20C [8], 0xFC02C22C [9], 0xFC02C24C [10], 0xFC02C26C [11], 0xFC02C28C [12], 0xFC02C2AC [13], 0xFC02C2CC [14], 0xFC02C2EC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	BUSY_BANK	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
MDATA_RX	DATAX_RX	–	–	INTDIS_DMA	–	AUTO_VALID	EPT_ENABL

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register” .

- **EPT_ENABL: Endpoint Enable (cleared upon USB reset)**

0: The endpoint is disabled according to the device configuration. Endpoint 0 should always be enabled after a hardware or UDPHS bus reset and participate in the device configuration.

1: The endpoint is enabled according to the device configuration.

- **AUTO_VALID: Packet Auto-Valid Enabled (cleared upon USB reset)**

Set this bit to automatically validate the current packet and switch to the next bank for both IN and OUT endpoints.

For IN Transfer:

If this bit is set, the UDPHS_EPTSTAx register TXRDY_TRER bit is set automatically when the current bank is full and at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set.

The user may still set the UDPHS_EPTSTAx register TXRDY_TRER bit if the current bank is not full, unless the user needs to send a Zero Length Packet by software.

For OUT Transfer:

If this bit is set, the UDPHS_EPTSTAx register RXRDY_TXKL bit is automatically reset for the current bank when the last packet byte has been read from the bank FIFO or at the end of DMA buffer if the UDPHS_DMACONTROLx register END_B_EN bit is set. For example, to truncate a padded data packet when the actual data transfer size is reached.

The user may still clear the UDPHS_EPTSTAx register RXRDY_TXKL bit, for example, after completing a DMA buffer by software if UDPHS_DMACONTROLx register END_B_EN bit was disabled or in order to cancel the read of the remaining data bank(s).

- **INTDIS_DMA: Interrupt Disables DMA (cleared upon USB reset)**

If set, when an enabled endpoint-originated interrupt is triggered, the DMA request is disabled regardless of the UDPHS_IEN register EPT_x bit for this endpoint. Then, the firmware will have to clear or disable the interrupt source or clear this bit if transfer completion is needed.

If the exception raised is associated with the new system bank packet, then the previous DMA packet transfer is normally completed, but the new DMA packet transfer is not started (not requested).

If the exception raised is not associated to a new system bank packet (ex: ERR_FL_ISO), then the request cancellation may happen at any time and may immediately stop the current DMA transfer.

This may be used, for example, to identify or prevent an erroneous packet to be transferred into a buffer or to complete a DMA buffer by software after reception of a short packet, or to perform buffer truncation on ERR_FL_ISO interrupt for adaptive rate.

- **DATAX_RX: DATAx Interrupt Enabled (Only for High Bandwidth Isochronous OUT endpoints) (cleared upon USB reset)**

0: No effect.

1: Send an interrupt when a DATA2, DATA1 or DATA0 packet has been received meaning the whole microframe data payload has been received.

- **MDATA_RX: MDATA Interrupt Enabled (Only for High Bandwidth Isochronous OUT endpoints) (cleared upon USB reset)**

0: No effect.

1: Send an interrupt when an MDATA packet has been received and so at least one packet of the microframe data payload has been received.

- **ERR_OVFLW: Overflow Error Interrupt Enabled (cleared upon USB reset)**

0: Overflow Error Interrupt is masked.

1: Overflow Error Interrupt is enabled.

- **RXRDY_TXKL: Received OUT Data Interrupt Enabled (cleared upon USB reset)**

0: Received OUT Data Interrupt is masked.

1: Received OUT Data Interrupt is enabled.

- **TX_COMPLT: Transmitted IN Data Complete Interrupt Enabled (cleared upon USB reset)**

0: Transmitted IN Data Complete Interrupt is masked.

1: Transmitted IN Data Complete Interrupt is enabled.

- **TXRDY_TRER: TX Packet Ready/Transaction Error Interrupt Enabled (cleared upon USB reset)**

0: TX Packet Ready/Transaction Error Interrupt is masked.

1: TX Packet Ready/Transaction Error Interrupt is enabled.

Caution: Interrupt source is active as long as the corresponding UDPHS_EPTSTAx register TXRDY_TRER flag remains low. If there are no more banks available for transmitting after the software has set UDPHS_EPTSTAx/TXRDY_TRER for the last transmit packet, then the interrupt source remains inactive until the first bank becomes free again to transmit at UDPHS_EPTSTAx/TXRDY_TRER hardware clear.

- **ERR_FL_ISO: Error Flow Interrupt Enabled (cleared upon USB reset)**

0: Error Flow Interrupt is masked.

1: Error Flow Interrupt is enabled.

- **ERR_CRC_NTR: ISO CRC Error/Number of Transaction Error Interrupt Enabled (cleared upon USB reset)**

0: ISO CRC error/number of Transaction Error Interrupt is masked.

1: ISO CRC error/number of Transaction Error Interrupt is enabled.

- **ERR_FLUSH: Bank Flush Error Interrupt Enabled (cleared upon USB reset)**

0: Bank Flush Error Interrupt is masked.

1: Bank Flush Error Interrupt is enabled.

- **BUSY_BANK: Busy Bank Interrupt Enabled (cleared upon USB reset)**

0: BUSY_BANK Interrupt is masked.

1: BUSY_BANK Interrupt is enabled.

For OUT endpoints: An interrupt is sent when all banks are busy.

For IN endpoints: An interrupt is sent when all banks are free.

- **SHRT_PCKT: Short Packet Interrupt Enabled (cleared upon USB reset)**

For OUT endpoints: send an Interrupt when a Short Packet has been received.

0: Short Packet Interrupt is masked.

1: Short Packet Interrupt is enabled.

For IN endpoints: A Short Packet transmission is guaranteed upon end of the DMA Transfer, thus signaling an end of isochronous (micro-)frame data, but only if the UDPHS_DMACONTROLx register END_B_EN and UDPHS_EPTCTLx register AUTO_VALID bits are also set.

38.7.19 UDPHS Endpoint Set Status Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTSETSTAx [x=0..15]

Address: 0xFC02C114 [0], 0xFC02C134 [1], 0xFC02C154 [2], 0xFC02C174 [3], 0xFC02C194 [4], 0xFC02C1B4 [5], 0xFC02C1D4 [6], 0xFC02C1F4 [7], 0xFC02C214 [8], 0xFC02C234 [9], 0xFC02C254 [10], 0xFC02C274 [11], 0xFC02C294 [12], 0xFC02C2B4 [13], 0xFC02C2D4 [14], 0xFC02C2F4 [15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	TXRDY	–	RXRDY_TXKL	–
7	6	5	4	3	2	1	0
–	–	FRCESTALL	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)” .

- **FRCESTALL: Stall Handshake Request Set**

0: No effect.

1: Set this bit to request a STALL answer to the host for the next handshake

Refer to chapters 8.4.5 (Handshake Packets) and 9.4.5 (Get Status) of the *Universal Serial Bus Specification, Rev 2.0* for more information on the STALL handshake.

- **RXRDY_TXKL: KILL Bank Set (for IN Endpoint)**

0: No effect.

1: Kill the last written bank.

- **TXRDY: TX Packet Ready Set**

0: No effect.

1: Set this bit after a packet has been written into the endpoint FIFO for IN data transfers

  – This flag is used to generate a Data IN transaction (device to host).
  – Device firmware checks that it can write a data payload in the FIFO, checking that TXRDY is cleared.
  – Transfer to the FIFO is done by writing in the “Buffer Address” register.
  – Once the data payload has been transferred to the FIFO, the firmware notifies the UDPHS device setting TXRDY to one.
  – UDPHS bus transactions can start.
  – TXCOMP is set once the data payload has been received by the host.
  – Data should be written into the endpoint FIFO only after this bit has been cleared.
  – Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet.

38.7.20 UDPHS Endpoint Set Status Register (Isochronous Endpoint)

Name: UDPHS_EPTSETSTAx [x=0..15] (ISOENDPT)

Address: 0xFC02C114 [0], 0xFC02C134 [1], 0xFC02C154 [2], 0xFC02C174 [3], 0xFC02C194 [4], 0xFC02C1B4 [5], 0xFC02C1D4 [6], 0xFC02C1F4 [7], 0xFC02C214 [8], 0xFC02C234 [9], 0xFC02C254 [10] 0xFC02C274 [11], 0xFC02C294 [12], 0xFC02C2B4 [13], 0xFC02C2D4 [14], 0xFC02C2F4 [15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	TXRDY_TRER	–	RXRDY_TXKL	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Status Register (Isochronous Endpoint)” .

- **RXRDY_TXKL: KILL Bank Set (for IN Endpoint)**

0: No effect.

1: Kill the last written bank.

- **TXRDY_TRER: TX Packet Ready Set**

0: No effect.

1: Set this bit after a packet has been written into the endpoint FIFO for IN data transfers

  – This flag is used to generate a Data IN transaction (device to host).
  – Device firmware checks that it can write a data payload in the FIFO, checking that TXRDY_TRER is cleared.
  – Transfer to the FIFO is done by writing in the “Buffer Address” register.
  – Once the data payload has been transferred to the FIFO, the firmware notifies the UDPHS device setting TXRDY_TRER to one.
  – UDPHS bus transactions can start.
  – TXCOMP is set once the data payload has been sent.
  – Data should be written into the endpoint FIFO only after this bit has been cleared.
  – Set this bit without writing data to the endpoint FIFO to send a Zero Length Packet.

38.7.21 UDPHS Endpoint Clear Status Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTCLRSTAx [x=0..15]

Address: 0xFC02C118 [0], 0xFC02C138 [1], 0xFC02C158 [2], 0xFC02C178 [3], 0xFC02C198 [4], 0xFC02C1B8 [5], 0xFC02C1D8 [6], 0xFC02C1F8 [7], 0xFC02C218 [8], 0xFC02C238 [9], 0xFC02C258 [10], 0xFC02C278 [11], 0xFC02C298 [12], 0xFC02C2B8 [13], 0xFC02C2D8 [14], 0xFC02C2F8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	–	TX_COMPLT	RXRDY_TXKL	–
7	6	5	4	3	2	1	0
–	TOGGLESQ	FRCESTALL	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)” .

- **FRCESTALL: Stall Handshake Request Clear**

0: No effect.

1: Clear the STALL request. The next packets from host will not be STALLed.

- **TOGGLESQ: Data Toggle Clear**

0: No effect.

1: Clear the PID data of the current bank

For OUT endpoints, the next received packet should be a DATA0.

For IN endpoints, the next packet will be sent with a DATA0 PID.

- **RXRDY_TXKL: Received OUT Data Clear**

0: No effect.

1: Clear the RXRDY_TXKL flag of UDPHS_EPTSTAx.

- **TX_COMPLT: Transmitted IN Data Complete Clear**

0: No effect.

1: Clear the TX_COMPLT flag of UDPHS_EPTSTAx.

- **RX_SETUP: Received SETUP Clear**

0: No effect.

1: Clear the RX_SETUP flags of UDPHS_EPTSTAx.

- **STALL_SNT: Stall Sent Clear**

0: No effect.

1: Clear the STALL_SNT flags of UDPHS_EPTSTAx.

- **NAK_IN: NAKIN Clear**

0: No effect.

1: Clear the NAK_IN flags of UDPHS_EPTSTAx.

- **NAK_OUT: NAKOUT Clear**

0: No effect.

1: Clear the NAK_OUT flag of UDPHS_EPTSTAx.

38.7.22 UDPHS Endpoint Clear Status Register (Isochronous Endpoint)

Name: UDPHS_EPTCLRSTAx [x=0..15] (ISOENDPT)

Address: 0xFC02C118 [0], 0xFC02C138 [1], 0xFC02C158 [2], 0xFC02C178 [3], 0xFC02C198 [4], 0xFC02C1B8 [5], 0xFC02C1D8 [6], 0xFC02C1F8 [7], 0xFC02C218 [8], 0xFC02C238 [9], 0xFC02C258 [10], 0xFC02C278 [11], 0xFC02C298 [12], 0xFC02C2B8 [13], 0xFC02C2D8 [14], 0xFC02C2F8 [15]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	–	TX_COMPLT	RXRDY_TXKL	–
7	6	5	4	3	2	1	0
–	TOGGLESQ	–	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register” .

For additional information, see “UDPHS Endpoint Status Register (Isochronous Endpoint)” .

- **TOGGLESQ: Data Toggle Clear**

0: No effect.

1: Clear the PID data of the current bank

For OUT endpoints, the next received packet should be a DATA0.

For IN endpoints, the next packet will be sent with a DATA0 PID.

- **RXRDY_TXKL: Received OUT Data Clear**

0: No effect.

1: Clear the RXRDY_TXKL flag of UDPHS_EPTSTAx.

- **TX_COMPLT: Transmitted IN Data Complete Clear**

0: No effect.

1: Clear the TX_COMPLT flag of UDPHS_EPTSTAx.

- **ERR_FL_ISO: Error Flow Clear**

0: No effect.

1: Clear the ERR_FL_ISO flags of UDPHS_EPTSTAx.

- **ERR_CRC_NTR: Number of Transaction Error Clear**

0: No effect.

1: Clear the ERR_CRC_NTR flags of UDPHS_EPTSTAx.

- **ERR_FLUSH: Bank Flush Error Clear**

0: No effect.

1: Clear the ERR_FLUSH flags of UDPHS_EPTSTAx.

38.7.23 UDPHS Endpoint Status Register (Control, Bulk, Interrupt Endpoints)

Name: UDPHS_EPTSTAx [x=0..15]

Address: 0xFC02C11C [0], 0xFC02C13C [1], 0xFC02C15C [2], 0xFC02C17C [3], 0xFC02C19C [4], 0xFC02C1BC [5], 0xFC02C1DC [6], 0xFC02C1FC [7], 0xFC02C21C [8], 0xFC02C23C [9], 0xFC02C25C [10], 0xFC02C27C [11], 0xFC02C29C [12], 0xFC02C2BC [13], 0xFC02C2DC [14], 0xFC02C2FC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	BYTE_COUNT						
23	22	21	20	19	18	17	16
BYTE_COUNT				BUSY_BANK_STA		CURBK_CTLDIR	
15	14	13	12	11	10	9	8
NAK_OUT	NAK_IN	STALL_SNT	RX_SETUP	TXRDY	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
TOGGLESQ_STA		FRCESTALL	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x0, 0x2 or 0x3 in “UDPHS Endpoint Configuration Register” .

- **FRCESTALL: Stall Handshake Request (cleared upon USB reset)**

0: No effect.

1: If set a STALL answer will be done to the host for the next handshake.

This bit is reset by hardware upon received SETUP.

- **TOGGLESQ_STA: Toggle Sequencing (cleared upon USB reset)**

Toggle Sequencing:

– **IN endpoint:** It indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.

– CONTROL and OUT endpoint:

These bits are set by hardware to indicate the PID data of the current bank:

Value	Name	Description
0	DATA0	DATA0
1	DATA1	DATA1
2	DATA2	Reserved for High Bandwidth Isochronous Endpoint
3	MDATA	Reserved for High Bandwidth Isochronous Endpoint

Notes:
1. In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).
2. These bits are updated for OUT transfer:
   - A new data has been written into the current bank.
   - The user has just cleared the Received OUT Data bit to switch to the next bank.
3. This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_OVFLW: Overflow Error (cleared upon USB reset)**

This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **RXRDY_TXKL: Received OUT Data/KILL Bank (cleared upon USB reset)**
  - **Received OUT Data** (for OUT endpoint or Control endpoint):

This bit is set by hardware after a new packet has been stored in the endpoint FIFO.

This bit is cleared by the device firmware after reading the OUT data from the endpoint.

For multibank endpoints, this bit may remain active even when cleared by the device firmware, this if an other packet has been received meanwhile.

Hardware assertion of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register RXRDY_TXKL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

  - **KILL Bank** (for IN endpoint):
  - The bank is really cleared or the bank is sent, BUSY_BANK_STA is decremented.
  - The bank is not cleared but sent on the IN transfer, TX_COMPLT
  - The bank is not cleared because it was empty. The user should wait that this bit is cleared before trying to clear another packet.

Note: “Kill a packet” may be refused if at the same time, an IN token is coming and the current packet is sent on the UDPHS line. In this case, the TX_COMPLT bit is set. Take notice however, that if at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. In fact, in that case, the current bank is sent (IN transfer) and the last bank is killed.

- **TX_COMPLT: Transmitted IN Data Complete (cleared upon USB reset)**

This bit is set by hardware after an IN packet has been accepted (ACK’ed) by the host.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **TXRDY: TX Packet Ready (cleared upon USB reset)**

This bit is cleared by hardware after the host has acknowledged the packet.

For Multibank endpoints, this bit may remain clear even after software is set if another bank is available to transmit.

Hardware clear of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register TXRDY bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **RX_SETUP: Received SETUP (cleared upon USB reset)**
  - (for Control endpoint only)

This bit is set by hardware when a valid SETUP packet has been received from the host.

It is cleared by the device firmware after reading the SETUP data from the endpoint FIFO.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **STALL_SNT: Stall Sent (cleared upon USB reset)**
  – (for Control, Bulk and Interrupt endpoints)

This bit is set by hardware after a STALL handshake has been sent as requested by the UDPHS_EPTSTAx register FRCESTALL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **NAK_IN: NAK IN (cleared upon USB reset)**

This bit is set by hardware when a NAK handshake has been sent in response to an IN request from the Host.

This bit is cleared by software.

- **NAK_OUT: NAK OUT (cleared upon USB reset)**

This bit is set by hardware when a NAK handshake has been sent in response to an OUT or PING request from the Host.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by EPT_CTL_DISx (disable endpoint).

- **CURBK_CTLDIR: Current Bank/Control Direction (cleared upon USB reset)**
  – **Current Bank** (not relevant for Control endpoint):

These bits are set by hardware to indicate the number of the current bank.

Value	Name	Description
0	BANK0	Bank 0 (or single bank)
1	BANK1	Bank 1
2	BANK2	Bank 2

Note: The current bank is updated each time the user:
- Sets the TX Packet Ready bit to prepare the next IN transfer and to switch to the next bank.
- Clears the received OUT data bit to access the next bank.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

  – **Control Direction** (for Control endpoint only):

0: A Control Write is requested by the Host.

1: A Control Read is requested by the Host.

Notes: 1. This bit corresponds with the 7th bit of the bmRequestType (Byte 0 of the Setup Data).
2. This bit is updated after receiving new setup data.

- **BUSY_BANK_STA: Busy Bank Number (cleared upon USB reset)**

These bits are set by hardware to indicate the number of busy banks.

IN endpoint: It indicates the number of busy banks filled by the user, ready for IN transfer.

OUT endpoint: It indicates the number of busy banks filled by OUT transaction from the Host.

Value	Name	Description
0	0BUSYBANK	All banks are free
1	1BUSYBANK	1 busy bank
2	2BUSYBANKS	2 busy banks
3	3BUSYBANKS	3 busy banks

- **BYTE_COUNT: UDPHS Byte Count (cleared upon USB reset)**

Byte count of a received data packet.

This field is incremented after each write into the endpoint (to prepare an IN transfer).

This field is decremented after each reading into the endpoint (OUT transfer).

This field is also updated at RXRDY_TXKL flag clear with the next bank.

This field is also updated at TXRDY flag set with the next bank.

This field is reset by EPT_x of UDPHS_EPTRST register.

- **SHRT_PCKT: Short Packet (cleared upon USB reset)**

An OUT Short Packet is detected when the receive byte count is less than the configured UDPHS_EPTCFGx register EPT_Size.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

38.7.24 UDPHS Endpoint Status Register (Isochronous Endpoint)

Name: UDPHS_EPTSTAx [x=0..15] (ISOENDPT)

Address: 0xFC02C11C [0], 0xFC02C13C [1], 0xFC02C15C [2], 0xFC02C17C [3], 0xFC02C19C [4], 0xFC02C1BC [5], 0xFC02C1DC [6], 0xFC02C1FC [7], 0xFC02C21C [8], 0xFC02C23C [9], 0xFC02C25C [10], 0xFC02C27C [11], 0xFC02C29C [12], 0xFC02C2BC [13], 0xFC02C2DC [14], 0xFC02C2FC [15]

Access: Read-only

31	30	29	28	27	26	25	24
SHRT_PCKT	BYTE_COUNT						
23	22	21	20	19	18	17	16
BYTE_COUNT				BUSY_BANK_STA		CURBK	
15	14	13	12	11	10	9	8
–	ERR_FLUSH	ERR_CRC_NTR	ERR_FL_ISO	TXRDY_TRER	TX_COMPLT	RXRDY_TXKL	ERR_OVFLW
7	6	5	4	3	2	1	0
TOGGLESQ_STA		–	–	–	–	–	–

This register view is relevant only if EPT_TYPE = 0x1 in “UDPHS Endpoint Configuration Register” .

- **TOGGLESQ_STA: Toggle Sequencing (cleared upon USB reset)**

Toggle Sequencing:

– **IN endpoint:** It indicates the PID Data Toggle that will be used for the next packet sent. This is not relative to the current bank.

– OUT endpoint:

These bits are set by hardware to indicate the PID data of the current bank:

Value	Name	Description
0	DATA0	DATA0
1	DATA1	DATA1
2	DATA2	Data2 (only for High Bandwidth Isochronous Endpoint)
3	MDATA	MData (only for High Bandwidth Isochronous Endpoint)

Notes:
1. In OUT transfer, the Toggle information is meaningful only when the current bank is busy (Received OUT Data = 1).
2. These bits are updated for OUT transfer:
   - A new data has been written into the current bank.
   - The user has just cleared the Received OUT Data bit to switch to the next bank.
3. For High Bandwidth Isochronous Out endpoint, it is recommended to check the UDPHS_EPTSTAx/TXRDY_TRER bit to know if the toggle sequencing is correct or not.
4. This field is reset to DATA1 by the UDPHS_EPTCLRSTAx register TOGGLESQ bit, and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_OVFLW: Overflow Error (cleared upon USB reset)**

This bit is set by hardware when a new too-long packet is received.

Example: If the user programs an endpoint 64 bytes wide and the host sends 128 bytes in an OUT transfer, then the Overflow Error bit is set.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **RXRDY_TXKL: Received OUT Data/KILL Bank (cleared upon USB reset)**
  - **Received OUT Data** (for OUT endpoint or Control endpoint):

This bit is set by hardware after a new packet has been stored in the endpoint FIFO.

This bit is cleared by the device firmware after reading the OUT data from the endpoint.

For multibank endpoints, this bit may remain active even when cleared by the device firmware, this if an other packet has been received meanwhile.

Hardware assertion of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register RXRDY_TXKL bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

  - **KILL Bank** (for IN endpoint):
  - The bank is really cleared or the bank is sent, BUSY_BANK_STA is decremented.
  - The bank is not cleared but sent on the IN transfer, TX_COMPLT
  - The bank is not cleared because it was empty. The user should wait that this bit is cleared before trying to clear another packet.

Note: “Kill a packet” may be refused if at the same time, an IN token is coming and the current packet is sent on the UDPHS line. In this case, the TX_COMPLT bit is set. Take notice however, that if at least two banks are ready to be sent, there is no problem to kill a packet even if an IN token is coming. In fact, in that case, the current bank is sent (IN transfer) and the last bank is killed.

- **TX_COMPLT: Transmitted IN Data Complete (cleared upon USB reset)**

This bit is set by hardware after an IN packet has been sent.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **TXRDY_TRER: TX Packet Ready/Transaction Error (cleared upon USB reset)**
  - **TX Packet Ready**:

This bit is cleared by hardware, as soon as the packet has been sent.

For Multibank endpoints, this bit may remain clear even after software is set if another bank is available to transmit.

Hardware clear of this bit may generate an interrupt if enabled by the UDPHS_EPTCTLx register TXRDY_TRER bit.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

  - **Transaction Error** (for high bandwidth isochronous OUT endpoints) (Read-Only):

This bit is set by hardware when a transaction error occurs inside one microframe.

If one toggle sequencing problem occurs among the n-transactions (n = 1, 2 or 3) inside a microframe, then this bit is still set as long as the current bank contains one “bad” n-transaction (see “CURBK: Current Bank (cleared upon USB reset)”). As soon as the current bank is relative to a new “good” n-transactions, then this bit is reset.

Notes:
1. A transaction error occurs when the toggle sequencing does not comply with the *Universal Serial Bus Specification, Rev 2.0* (5.9.2 High Bandwidth Isochronous endpoints) (bad PID, missing data, etc.).
2. When a transaction error occurs, the user may empty all the “bad” transactions by clearing the Received OUT Data flag (RXRDY_TXKL).

If this bit is reset, then the user should consider that a new n-transaction is coming.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint), and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_FL_ISO: Error Flow (cleared upon USB reset)**

This bit is set by hardware when a transaction error occurs.

– Isochronous IN transaction is missed, the micro has no time to fill the endpoint (underflow).

– Isochronous OUT data is dropped because the bank is busy (overflow).

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_CRC_NTR: CRC ISO Error/Number of Transaction Error (cleared upon USB reset)**

– **CRC ISO Error** (for Isochronous OUT endpoints) (Read-only):

This bit is set by hardware if the last received data is corrupted (CRC error on data).

This bit is updated by hardware when new data is received (Received OUT Data bit).

– **Number of Transaction Error** (for High Bandwidth Isochronous IN endpoints):

This bit is set at the end of a microframe in which at least one data bank has been transmitted, if less than the number of transactions per micro-frame banks (UDPHS_EPTCFGx register NB_TRANS) have been validated for transmission inside this microframe.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **ERR_FLUSH: Bank Flush Error (cleared upon USB reset)**

– (for High Bandwidth Isochronous IN endpoints)

This bit is set when flushing unsent banks at the end of a microframe.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by EPT_CTL_DISx (disable endpoint).

- **CURBK: Current Bank (cleared upon USB reset)**

– **Current Bank**:

These bits are set by hardware to indicate the number of the current bank.

Value	Name	Description
0	BANK0	Bank 0 (or single bank)
1	BANK1	Bank 1
2	BANK2	Bank 2

Note: The current bank is updated each time the user:
- Sets the TX Packet Ready bit to prepare the next IN transfer and to switch to the next bank.
- Clears the received OUT data bit to access the next bank.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

- **BUSY_BANK_STA: Busy Bank Number (cleared upon USB reset)**

These bits are set by hardware to indicate the number of busy banks.

– **IN endpoint**: It indicates the number of busy banks filled by the user, ready for IN transfer.

– **OUT endpoint**: It indicates the number of busy banks filled by OUT transaction from the Host.

Value	Name	Description
0	0BUSYBANK	All banks are free
1	1BUSYBANK	1 busy bank
2	2BUSYBANKS	2 busy banks
3	3BUSYBANKS	3 busy banks

- **BYTE_COUNT: UDPHS Byte Count (cleared upon USB reset)**

Byte count of a received data packet.

This field is incremented after each write into the endpoint (to prepare an IN transfer).

This field is decremented after each reading into the endpoint (OUT transfer).

This field is also updated at RXRDY_TXKL flag clear with the next bank.

This field is also updated at TXRDY_TRER flag set with the next bank.

This field is reset by EPT_x of UDPHS_EPTRST register.

- **SHRT_PCKT: Short Packet (cleared upon USB reset)**

An OUT Short Packet is detected when the receive byte count is less than the configured UDPHS_EPTCFGx register EPT_Size.

This bit is updated at the same time as the BYTE_COUNT field.

This bit is reset by UDPHS_EPTRST register EPT_x (reset endpoint) and by UDPHS_EPTCTLDISx (disable endpoint).

38.7.25 UDPHS DMA Channel Transfer Descriptor

The DMA channel transfer descriptor is loaded from the memory.

Be careful with the alignment of this buffer.

The structure of the DMA channel transfer descriptor is defined by three parameters as described below:

Offset 0:

The address must be aligned: 0xXXXX0

Next Descriptor Address Register: UDPHS_DMANXTDSCx

Offset 4:

The address must be aligned: 0xXXXX4

DMA Channelx Address Register: UDPHS_DMAADDRESSx

Offset 8:

The address must be aligned: 0xXXXX8

DMA Channelx Control Register: UDPHS_DMACONTROLx

To use the DMA channel transfer descriptor, fill the structures with the correct value (as described in the following pages).

Then write directly in UDPHS_DMANXTDSCx the address of the descriptor to be used first.

Then write 1 in the LDNXT_DSC bit of UDPHS_DMACONTROLx (load next channel transfer descriptor). The descriptor is automatically loaded upon Endpointx request for packet transfer.

38.7.26 UDPHS DMA Next Descriptor Address Register

Name: UDPHS_DMANXTDSCx [x = 0..6]

Address: 0xFC02C300 [0], 0xFC02C310 [1], 0xFC02C320 [2], 0xFC02C330 [3], 0xFC02C340 [4], 0xFC02C350 [5], 0xFC02C360 [6]

Access: Read/Write

31	30	29	28	27	26	25	24
NXT_DSC_ADD							

23	22	21	20	19	18	17	16
NXT_DSC_ADD							

15	14	13	12	11	10	9	8
NXT_DSC_ADD							

7	6	5	4	3	2	1	0
NXT_DSC_ADD							

Note: Channel 0 is not used.

- **NXT_DSC_ADD: Next Descriptor Address**

This field points to the next channel descriptor to be processed. This channel descriptor must be aligned, so bits 0 to 3 of the address must be equal to zero.

38.7.27 UDPHS DMA Channel Address Register

Name: UDPHS_DMAADDRESSx [x = 0..6]

Address: 0xFC02C304 [0], 0xFC02C314 [1], 0xFC02C324 [2], 0xFC02C334 [3], 0xFC02C344 [4], 0xFC02C354 [5], 0xFC02C364 [6]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_ADD							
23	22	21	20	19	18	17	16
BUFF_ADD							
15	14	13	12	11	10	9	8
BUFF_ADD							
7	6	5	4	3	2	1	0
BUFF_ADD							

Note: Channel 0 is not used.

- **BUFF_ADD: Buffer Address**

This field determines the AHB bus starting address of a DMA channel transfer.

Channel start and end addresses may be aligned on any byte boundary.

The firmware may write this field only when the UDPHS_DMASTATUS register CHANN_ENB bit is clear.

This field is updated at the end of the address phase of the current access to the AHB bus. It is incrementing of the access byte width. The access width is 4 bytes (or less) at packet start or end, if the start or end address is not aligned on a word boundary.

The packet start address is either the channel start address or the next channel address to be accessed in the channel buffer.

The packet end address is either the channel end address or the latest channel address accessed in the channel buffer.

The channel start address is written by software or loaded from the descriptor, whereas the channel end address is either determined by the end of buffer or the UDPHS device, USB end of transfer if the UDPHS_DMACONTROLx register END_TR_EN bit is set.

38.7.28 UDPHS DMA Channel Control Register

Name: UDPHS_DMACONTROLx [x = 0..6]

Address: 0xFC02C308 [0], 0xFC02C318 [1], 0xFC02C328 [2], 0xFC02C338 [3], 0xFC02C348 [4], 0xFC02C358 [5], 0xFC02C368 [6]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_LENGTH							
23	22	21	20	19	18	17	16
BUFF_LENGTH							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
BURST_LCK	DESC_LD_IT	END_BUFFIT	END_TR_IT	END_B_EN	END_TR_EN	LDNXT_DSC	CHANN_ENB

Note: Channel 0 is not used.

- **CHANN_ENB: (Channel Enable Command)**

0: DMA channel is disabled at and no transfer will occur upon request. This bit is also cleared by hardware when the channel source bus is disabled at end of buffer.

If the UDPHS_DMACONTROL register LDNXT_DSC bit has been cleared by descriptor loading, the firmware will have to set the corresponding CHANN_ENB bit to start the described transfer, if needed.

If the UDPHS_DMACONTROL register LDNXT_DSC bit is cleared, the channel is frozen and the channel registers may then be read and/or written reliably as soon as both UDPHS_DMASTATUS register CHANN_ENB and CHANN_ACT flags read as 0.

If a channel request is currently serviced when this bit is cleared, the DMA FIFO buffer is drained until it is empty, then the UDPHS_DMASTATUS register CHANN_ENB bit is cleared.

If the LDNXT_DSC bit is set at or after this bit clearing, then the currently loaded descriptor is skipped (no data transfer occurs) and the next descriptor is immediately loaded.

1: UDPHS_DMASTATUS register CHANN_ENB bit will be set, thus enabling DMA channel data transfer. Then any pending request will start the transfer. This may be used to start or resume any requested transfer.

- **LDNXT_DSC: Load Next Channel Transfer Descriptor Enable (Command)**

0: No channel register is loaded after the end of the channel transfer.

1: The channel controller loads the next descriptor after the end of the current transfer, i.e., when the UDPHS_DMASTATUS/CHANN_ENB bit is reset.

If the UDPHS_DMA CONTROL/CHANN_ENB bit is cleared, the next descriptor is immediately loaded upon transfer request.

DMA Channel Control Command Summary

LDNXT_DSC	CHANN_ENB	Description
0	0	Stop now
0	1	Run and stop at end of buffer
1	0	Load next descriptor now
1	1	Run and link at end of buffer

- **END_TR_EN: End of Transfer Enable (Control)**

Used for OUT transfers only.

0: USB end of transfer is ignored.

1: UDPHS device can put an end to the current buffer transfer.

When set, a BULK or INTERRUPT short packet or the last packet of an ISOCHRONOUS (micro) frame (DATAX) will close the current buffer and the UDPHS_DMASTATUSx register END_TR_ST flag will be raised.

This is intended for UDPHS non-prenegotiated end of transfer (BULK or INTERRUPT) or ISOCHRONOUS microframe data buffer closure.

- **END_B_EN: End of Buffer Enable (Control)**

0: DMA Buffer End has no impact on USB packet transfer.

1: Endpoint can validate the packet (according to the values programmed in the UDPHS_EPTCTLx register AUTO_VALID and SHRT_PCKT fields) at DMA Buffer End, i.e., when the UDPHS_DMASTATUS register BUFF_COUNT reaches 0.

This is mainly for short packet IN validation initiated by the DMA reaching end of buffer, but could be used for OUT packet truncation (discarding of unwanted packet data) at the end of DMA buffer.

- **END_TR_IT: End of Transfer Interrupt Enable**

0: UDPHS device initiated buffer transfer completion will not trigger any interrupt at UDPHS_STATUSx/END_TR_ST rising.

1: An interrupt is sent after the buffer transfer is complete, if the UDPHS device has ended the buffer transfer.

Use when the receive size is unknown.

- **END_BUFFIT: End of Buffer Interrupt Enable**

0: UDPHS_DMA_STATUSx/END_BF_ST rising will not trigger any interrupt.

1: An interrupt is generated when the UDPHS_DMASTATUSx register BUFF_COUNT reaches zero.

- **DESC_LD_IT: Descriptor Loaded Interrupt Enable**

0: UDPHS_DMASTATUSx/DESC_LDST rising will not trigger any interrupt.

1: An interrupt is generated when a descriptor has been loaded from the bus.

- **BURST_LCK: Burst Lock Enable**

0: The DMA never locks bus access.

1: USB packets AHB data bursts are locked for maximum optimization of the bus bandwidth usage and maximization of fly-by AHB burst duration.

- **BUFF_LENGTH: Buffer Byte Length (Write-only)**

This field determines the number of bytes to be transferred until end of buffer. The maximum channel transfer size (64 KBytes) is reached when this field is 0 (default value). If the transfer size is unknown, this field should be set to 0, but the transfer end may occur earlier under UDPHS device control.

When this field is written, The UDPHS_DMASTATUSx register BUFF_COUNT field is updated with the write value.

Notes:
1. Bits [31:2] are only writable when issuing a channel Control Command other than “Stop Now”.
2. For reliability it is highly recommended to wait for both UDPHS_DMASTATUSx register CHAN_ACT and CHAN_ENB flags are at 0, thus ensuring the channel has been stopped before issuing a command other than “Stop Now”.

38.7.29 UDPHS DMA Channel Status Register

Name: UDPHS_DMASTATUSx [x = 0..6]

Address: 0xFC02C30C [0], 0xFC02C31C [1], 0xFC02C32C [2], 0xFC02C33C [3], 0xFC02C34C [4], 0xFC02C35C [5], 0xFC02C36C [6]

Access: Read/Write

31	30	29	28	27	26	25	24
BUFF_COUNT							
23	22	21	20	19	18	17	16
BUFF_COUNT							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	DESC_LDST	END_BF_ST	END_TR_ST	–	–	CHANN_ACT	CHANN_ENB

Note: Channel 0 is not used.

- **CHANN_ENB: Channel Enable Status**

0: The DMA channel no longer transfers data, and may load the next descriptor if the UDPHS_DMACONTROLx register LDNXT_DSC bit is set.

When any transfer is ended either due to an elapsed byte count or a UDPHS device initiated transfer end, this bit is automatically reset.

1: The DMA channel is currently enabled and transfers data upon request.

This bit is normally set or cleared by writing into the UDPHS_DMACONTROLx register CHANN_ENB bit either by software or descriptor loading.

If a channel request is currently serviced when the UDPHS_DMACONTROLx register CHANN_ENB bit is cleared, the DMA FIFO buffer is drained until it is empty, then this status bit is cleared.

- **CHANN_ACT: Channel Active Status**

0: The DMA channel is no longer trying to source the packet data.

When a packet transfer is ended this bit is automatically reset.

1: The DMA channel is currently trying to source packet data, i.e., selected as the highest-priority requesting channel.

When a packet transfer cannot be completed due to an END_BF_ST, this flag stays set during the next channel descriptor load (if any) and potentially until UDPHS packet transfer completion, if allowed by the new descriptor.

- **END_TR_ST: End of Channel Transfer Status**

0: Cleared automatically when read by software.

1: Set by hardware when the last packet transfer is complete, if the UDPHS device has ended the transfer.

Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

- **END_BF_ST: End of Channel Buffer Status**

0: Cleared automatically when read by software.

1: Set by hardware when the BUFF_COUNT countdown reaches zero.

Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

- **DESC_LDST: Descriptor Loaded Status**

0: Cleared automatically when read by software.

1: Set by hardware when a descriptor has been loaded from the system bus.

Valid until the CHANN_ENB flag is cleared at the end of the next buffer transfer.

- **BUFF_COUNT: Buffer Byte Count**

This field determines the current number of bytes still to be transferred for this buffer.

This field is decremented from the AHB source bus access byte width at the end of this bus address phase.

The access byte width is 4 by default, or less, at DMA start or end, if the start or end address is not aligned on a word boundary.

At the end of buffer, the DMA accesses the UDPHS device only for the number of bytes needed to complete it.

This field value is reliable (stable) only if the channel has been stopped or frozen (UDPHS_EPTCTLx register NT_DIS_DMA bit is used to disable the channel request) and the channel is no longer active CHANN_ACT flag is 0.

Note: For OUT endpoints, if the receive buffer byte length (BUFF_LENGTH) has been defaulted to zero because the USB transfer length is unknown, the actual buffer byte length received will be 0x10000-BUFF_COUNT.

39. USB Host High Speed Port (UHPHS)

39.1 Description

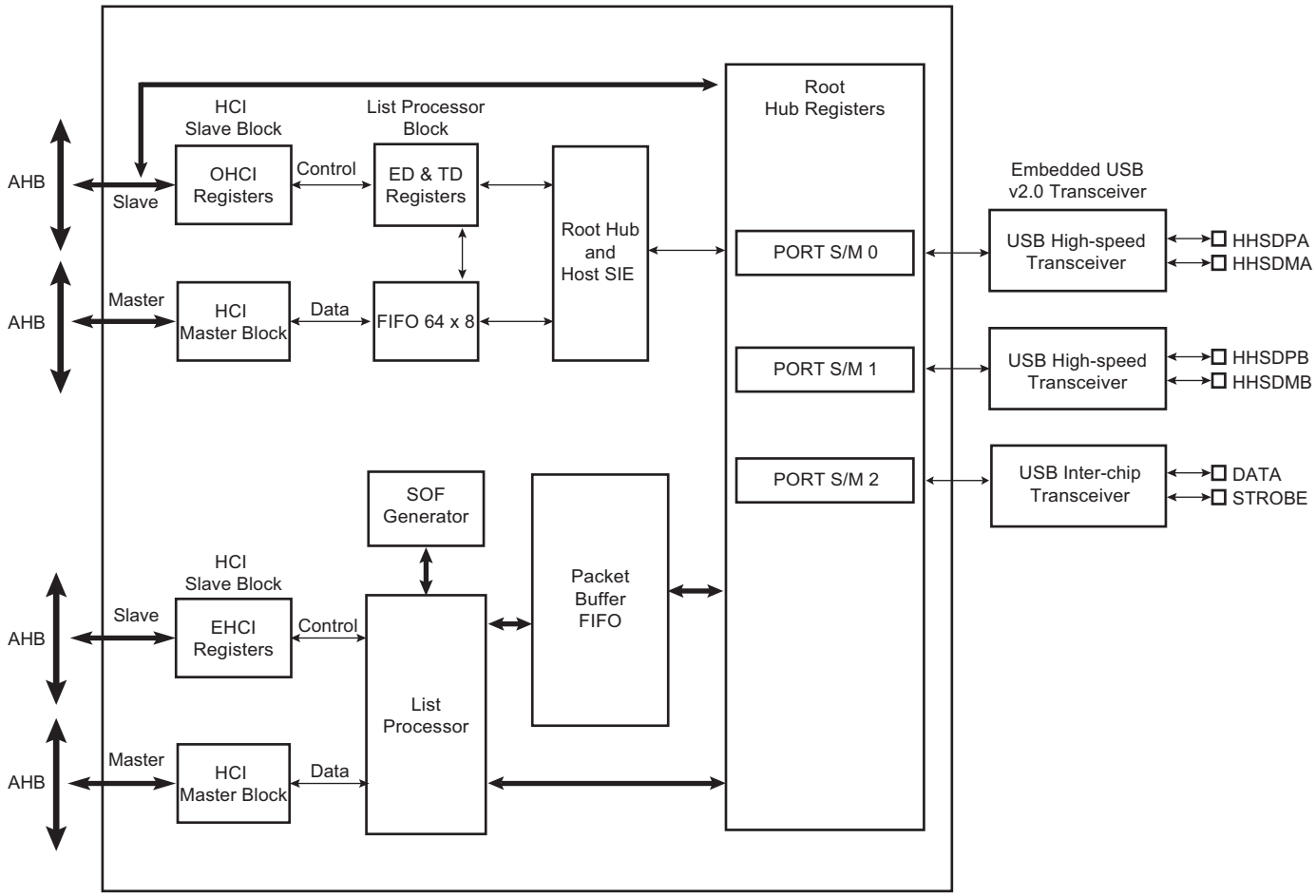
The USB Host High Speed Port (UHPHS) interfaces the USB with the host application. It handles Open HCI protocol (Open Host Controller Interface) as well as Enhanced HCI protocol (Enhanced Host Controller Interface).

39.2 Embedded Characteristics

- Compliant with Enhanced HCI Rev 1.0 Specification
  - Compliant with USB V2.0 High-speed
  - Supports High-speed 480 Mbps
- Compliant with OpenHCI Rev 1.0 Specification
  - Compliant with USB V2.0 Full-speed and Low-speed Specification
  - Supports both Low-speed 1.5 Mbps and Full-speed 12 Mbps USB devices
- Root Hub Integrated with 3 Downstream USB HS Ports
- Embedded USB Transceivers
- Supports Power Management
- 2 Hosts (A and B) High Speed (EHCI), Port A shared with UDPHS
- 1 Host (C) High Speed only (HSIC)

39.3 Block Diagram

Figure 39-1. Block Diagram



Access to the USB host operational registers is achieved through the AHB bus slave interface. The Open HCI host controller and Enhanced HCI host controller initialize master DMA transfers through the AHB bus master interface as follows:

- Fetches endpoint descriptors and transfer descriptors
- Access to endpoint data from system memory
- Access to the HC communication area
- Write status and retire transfer descriptor

Memory access errors (abort, misalignment) lead to an “Unrecoverable Error” indicated by the corresponding flag in the host controller operational registers.

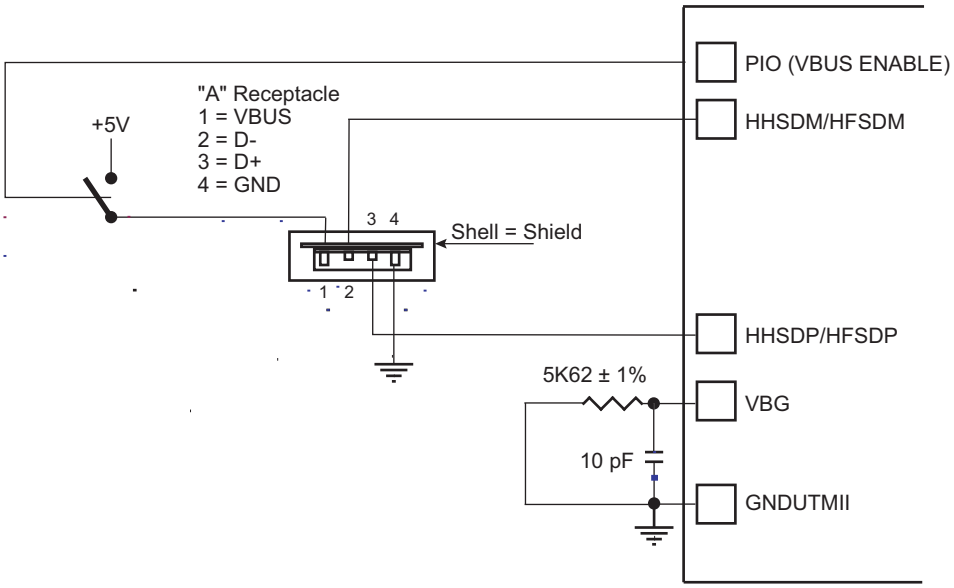
The USB root hub is integrated in the USB host. Several USB downstream ports are available. The number of downstream ports can be determined by the software driver reading the root hub’s operational registers. Device connection is automatically detected by the USB host port logic.

USB physical transceivers are integrated in the product and driven by the root hub’s ports.

Over current protection on ports can be activated by the USB host controller. Atmel’s standard product does not dedicate pads to external over current protection.

39.4 Typical Connection

Figure 39-2. Board Schematic to Interface UHP High-speed Host Controller



Note: 1. 10 pF capacitor on VBG is a provision and may not be populated.

39.5 Product Dependencies

39.5.1 I/O Lines

HFSDPs, HFSDMs, HHSDPs and HHSDMs are not controlled by any PIO controllers. The embedded USB High Speed physical transceivers are controlled by the USB host controller.

39.5.2 Power Management

The system embeds 3 transceivers.

The USB Host High Speed requires a 480 MHz clock for the embedded High-speed transceivers. This clock (UPLLCK) is provided by the UTMI PLL.

In case power consumption is saved by stopping the UTMI PLL, high-speed operations are not possible. Nevertheless, OHCI Full-speed operations remain possible by selecting PLLACK as the input clock of OHCI.

The High-speed transceiver returns a 30 MHz clock to the USB Host controller.

The USB Host controller requires 48 MHz and 12 MHz clocks for OHCI full-speed operations. These clocks must be generated by a PLL with a correct accuracy of ± 0.25% using the USBDIV field.

Thus the USB Host peripheral receives three clocks from the Power Management Controller (PMC): the Peripheral Clock (MCK domain), the UHP48M and the UHP12M (built-in UHP48M divided by four) used by the OHCI to interface with the bus USB signals (recovered 12 MHz domain) in Full-speed operations.

For High-speed operations, the user has to perform the following:

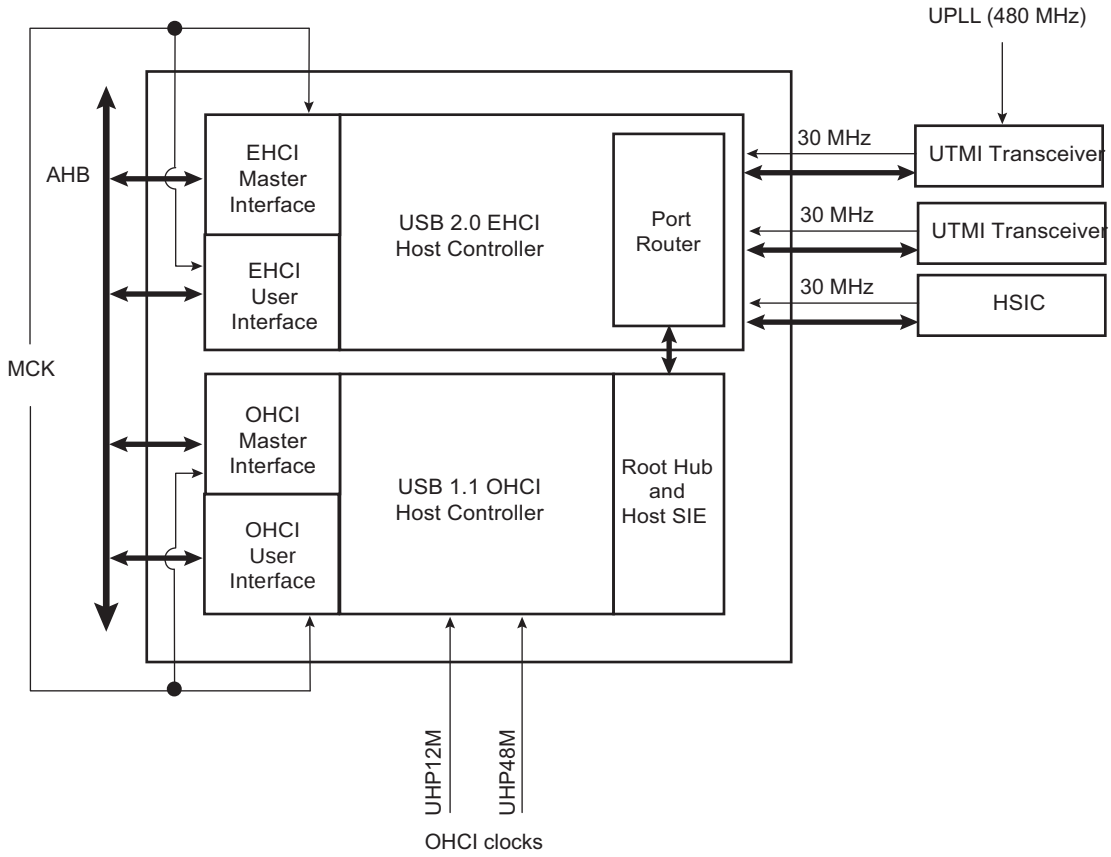
- Enable UHP peripheral clock in PMC_PCER.
- Write PLLCOUNT field in CKGR_UCKR.
- Enable UPLL with UPLLEN bit in CKGR_UCKR.
- Wait until UTMI_PLL is locked (LOCKU bit in PMC_SR).
- Enable BIAS with BIASEN bit in CKGR_UCKR.

- Select UPLLCK as Input clock of OHCI part (USBS bit in PMC_USB register).
- Program OHCI clocks (UHP48M and UHP12M) with USBDIV field in PMC_USB register. USBDIV must be 9 (division by 10) if UPLLCK is selected.
- Enable OHCI clocks with UHP bit in PMC_SCER.

For OHCI Full-speed operations only, the user has to perform the following:

- Enable UHP peripheral clock in PMC_PCER.
- Select PLLACK as Input clock of OHCI part (USBS bit in PMC_USB register).
- Program OHCI clocks (UHP48M and UHP12M) with USBDIV field in PMC_USB register. USBDIV value is to be calculated according to the PLLACK value and USB Full-speed accuracy.
- Enable the OHCI clocks with UHP bit in PMC_SCER.

Figure 39-3. UHP Clock Trees



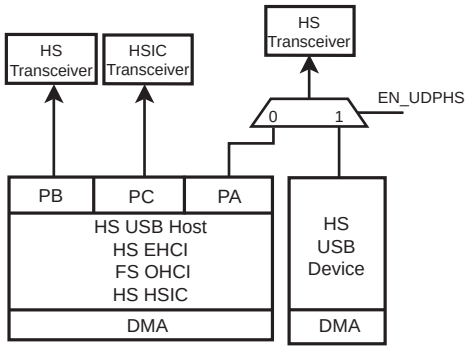
39.5.3 Interrupt Sources

The USB host interface has an interrupt line connected to the interrupt controller.

Handling USB host interrupts requires programming the interrupt controller before configuring the UHPHS.

39.6 Functional Description

Figure 39-4. USB Selection



39.6.1 EHCI

The USB Host Port controller is fully compliant with the Enhanced HCI specification. The USB Host Port User Interface (registers description) can be found in the Enhanced HCI Rev 1.0 Specification available on www.usb.org. The standard EHCI USB stack driver can be easily ported to Atmel's architecture in the same way all existing class drivers run, without hardware specialization.

39.6.2 OHCI

The USB Host Port integrates a root hub and transceivers on downstream ports. It provides several Full-speed half-duplex serial communication ports at a baud rate of 12 Mbit/s. Up to 127 USB devices (printer, camera, mouse, keyboard, disk, etc.) and the USB hub can be connected to the USB host in the USB "tiered star" topology.

The USB Host Port controller is fully compliant with the Open HCI specification. The USB Host Port User Interface (registers description) can be found in the Open HCI Rev 1.0 Specification available on www.usb.org. The standard OHCI USB stack driver can be easily ported to Atmel's architecture, in the same way all existing class drivers run without hardware specialization.

This means that all standard class devices are automatically detected and available to the user's application. As an example, integrating an HID (Human Interface Device) class driver provides a plug & play feature for all USB keyboards and mouses.

39.6.3 HSIC

The High-Speed Inter-Chip (HSIC) is a standard for USB chip-to-chip interconnect with a 2-signal (strobe, data) source synchronous serial interface using 240 MHz DDR signaling to provide only high-speed 480 Mbps data rate.

External cables, connectors and hot plug & play are not supported.

The HSIC interface operates at high speed, 480 Mbps, and is fully compatible with existing USB software stacks. It meets all data transfer needs through a single unified USB software stack.

39.7 USB Host High Speed Port (UHPHS) User Interface

The Enhanced USB Host Controller contains two sets of software-accessible hardware registers – Memory-mapped Host Controller Registers and optional PCI configuration registers. Note that the PCI configuration registers are only needed for PCI devices that implement the Host Controller.

- Memory-mapped USB Host Controller Registers. This block of registers is memory-mapped into non-cacheable memory. This memory space must begin on a DWord (32-bit) boundary. This register space is divided into two sections: a set of read-only capability registers and a set of read/write operational registers. Table 39-1 describes each register space.

Table 39-1. Enhanced Interface Register Sets

Offset	Register Set	Explanation
0 to N-1	Capability Registers	The capability registers specify the limits, restrictions, and capabilities of a host controller implementation. These values are used as parameters to the host controller driver.
N to N+M-1	Operational Registers	The operational registers are used by system software to control and monitor the operational state of the host controller.

Note: Host controllers are not required to support exclusive-access mechanisms (such as PCI LOCK) for accesses to the memory-mapped register space. Therefore, if software attempts exclusive-access mechanisms to the host controller memory-mapped register space, the results are undefined.

- PCI Configuration Registers (for PCI devices). In addition to the normal PCI header, power management, and device-specific registers, two registers are needed in the PCI configuration space to support USB. The normal PCI header and device-specific registers are beyond the scope of this document (the UHPHS_CLASSC register is shown in this document). Note that HCD does not interact with the PCI configuration space. This space is used only by the PCI enumerator to identify the USB Host Controller, and assign the appropriate system resources.

Table 39-2. Register Mapping

Offset	Register	Name	Access	Reset
Host Controller Capability Registers				
0x00	UHPHS Host Controller Capability Register	UHPHS_HCCAPBASE	Read-only	0x0100 0010
0x04	UHPHS Host Controller Structural Parameters Register	UHPHS_HCSPARAMS	Read-only	0x0000 1116
0x08	UHPHS Host Controller Capability Parameters Register	UHPHS_HCCPARAMS	Read-only	0x0000 A010
0x0C	Reserved	–	–	–
Host Controller Operational Registers				
0x10	UHPHS USB Command Register	UHPHS_USBCMD	Read/Write(1)	0x0008 0000 or 0x0008 0B00(2)
0x14	UHPHS USB Status Register	UHPHS_USBSTS	Read/Write(1)	0x0000 1000
0x18	UHPHS USB Interrupt Enable Register	UHPHS_USBINTR	Read/Write	0x0000 0000
0x1C	UHPHS USB Frame Index Register	UHPHS_FRINDEX	Read/Write	0x0000 0000
0x20	UHPHS Control Data Structure Segment Register	UHPHS_CTRLDSSEGMENT	Read/Write	0x0000 0000
0x24	UHPHS Periodic Frame List Base Address Register	UHPHS_PERIODICLISTBASE	Read/Write	0x0000 0000

Table 39-2. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x28	UHPHS Asynchronous List Address Register	UHPHS_ASYNCLISTADDR	Read/Write	0x0000 0000
0x2C - 0x4F	Reserved	–	–	–
0x50	UHPHS Configured Flag Register	UHPHS_CONFIGFLAG	Read/Write	0x0000 0000
0x54	UHPHS Port Status and Control Register 0	UHPHS_PORTSC_0	Read/Write(1)	0x0000 2000 or 0x0000 3000(3)
0x58	UHPHS Port Status and Control Register 1	UHPHS_PORTSC_1	Read/Write(1)	0x0000 2000 or 0x0000 3000(3)
0x5C	UHPHS Port Status and Control Register 2	UHPHS_PORTSC_2	Read/Write(1)	0x0000 2000 or 0x0000 3000(3)
0x90	EHCI Synopsys-Specific Registers 00	UHPHS_INSNREG00	Read/Write(1)	0x0000 0000
0x94	EHCI Synopsys-Specific Registers 01	UHPHS_INSNREG01	Read/Write(1)	0x0020 0020
0x98	EHCI Synopsys-Specific Registers 02	UHPHS_INSNREG02	Read/Write(1)	(5)
0x9C	EHCI Synopsys-Specific Registers 03	UHPHS_INSNREG03	Read/Write(1)	0x0000 0001
0xA0	EHCI Synopsys-Specific Registers 04	UHPHS_INSNREG04	Read/Write(1)	0x0000 0000
0xA4	EHCI Synopsys-Specific Registers 05	UHPHS_INSNREG05	Read/Write(1)	0x0000 1000
0xA8	EHCI Synopsys-Specific Registers 06	UHPHS_INSNREG06	Read/Write(1)	0x0000 0000
0xAC	EHCI Synopsys-Specific Registers 07	UHPHS_INSNREG07	Read/Write(1)	0x0000 0000
0xB0	EHCI Synopsys-Specific Registers 08	UHPHS_INSNREG08	Read/Write(1)	0x0000 0000

Notes:
1. Field-dependent.
2. The default value depends on whether the Asynchronous Schedule Park Capability (ASPC) field in the UHPHS_HCCPARAMS register is enabled. Disabled (set to 0) = 0x0008 0000h; Enabled (set to 1) = 0x0008 0B00h.
3. The default value depends on the value of the Port Power Control (PPC) field in the UHPHS_HCSPARAMS register. 0x0000 2000h (with PPC set to 1); 0x0000 3000h (with PPC set to 0).
4. Software should not assume reserved bits are always 0 and should preserve these bits when writing to modifiable registers.
5. This value is determined by coreConsultant.

39.7.1 UHPHS Host Controller Capability Register

Name: UHPHS_HCCAPBASE

Access: Read-only

31	30	29	28	27	26	25	24
HCIVERSION							
23	22	21	20	19	18	17	16
HCIVERSION							
15	14	13	12	11	10	9	8
–							
7	6	5	4	3	2	1	0
CAPLENGTH							

- **CAPLENGTH: Capability Registers Length**

10h: Default value.

This field is used as an offset to add to register base to find the beginning of the Operational Register Space.

- **HCIVERSION: Host Controller Interface Version Number**

0100h: Default value.

This is a two-byte field containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this field represents a major revision and the least significant byte is the minor revision.

39.7.2 UHPHS Host Controller Structural Parameters Register

Name: UHPHS_HCSPARAMS

Access: Read-only

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
N_DP				–			P_INDICATOR
15	14	13	12	11	10	9	8
N_CC				N_PCC			
7	6	5	4	3	2	1	0
–			PPC	N_PORTS			

This is a set of fields that are structural parameters: number of downstream ports, etc.

- **N_PORTS: Number of Ports**

This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register Space. Valid values are in the range of 1H to FH.

A zero in this field is undefined.

- **PPC: Port Power Control**

This field indicates whether the host controller implementation includes port power control. A one in this bit indicates the ports have port power switches. A zero in this bit indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register (see Section 39.7.12).

- **N_PCC: Number of Ports per Companion Controller**

This field indicates the number of ports supported per companion host controller. It is used to indicate the port routing configuration to system software.

For example, if N_PORTS has a value of 6 and N_CC has a value of 2, then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first four are routed to companion controller 1 and the last two are routed to companion controller 2.

The number in this field must be consistent with N_PORTS and N_CC.

- **N_CC: Number of Companion Controllers**

This field indicates the number of companion controllers associated with this USB 2.0 host controller.

A zero in this field indicates there are no companion host controllers. Port-ownership hand-off is not supported. Only high-speed devices are supported on the host controller root ports.

A value larger than zero in this field indicates there are companion USB 1.1 host controller(s). Port-ownership hand-offs are supported. High, Full- and Low-speed devices are supported on the host controller root ports.

- **P_INDICATOR: Port Indicators**

This bit indicates whether the ports support port indicator control. When this bit is a 1, the port status and control registers include a read/writeable field for controlling the state of the port indicator. See Section 39.7.12 for definition of the port indicator control field.

- **N_DP: Debug Port Number**

Optional. This register identifies which of the host controller ports is the debug port. The value is the port number (1-based) of the debug port. A non-zero value in this field indicates the presence of a debug port. The value in this register must not be greater than N_PORTS (see above).

39.7.3 UHPHS Host Controller Capability Parameters Register

Name: UHPHS_HCCPARAMS

Access: Read-only

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
EECP							
7	6	5	4	3	2	1	0
IST				–	ASPC	PFLF	AC

This is a set of fields that are capability parameters: Multiple Mode control (time-base bit functionality), addressing capability, etc.

- **AC: 64-bit Addressing Capability**

This field documents the addressing range capability of this implementation. The value of this field determines whether software should use 32-bit or 64-bit data structures.

Values for this field have the following interpretation:

0: Data structures using 32-bit address memory pointers

1: Data structures using 64-bit address memory pointers

Note: This is not tightly coupled with the UHPHS_USBBASE address register mapping control. The 64-bit Addressing Capability bit indicates whether the host controller can generate 64-bit addresses as a master. The UHPHS_USBBASE register indicates the host controller only needs to decode 32-bit addresses as a slave.

- **PFLF: Programmable Frame List Flag**

The default value is implementation-dependent.

If this bit is set to 0, then system software must use a frame list length of 1024 elements with this host controller. The UHPHS_USBCMD register Frame List Size field is a read-only register and should be set to 0.

If set to 1, then system software can specify and use a smaller frame list and configure the host controller via the UHPHS_USBCMD register Frame List Size field. The frame list must always be aligned on a 4-kbyte page boundary. This requirement ensures that the frame list is always physically contiguous.

- **ASPC: Asynchronous Schedule Park Capability**

The default value is Implementation dependent.

If this bit is set to 1, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the Asynchronous Schedule Park Mode Enable and Asynchronous Schedule Park Mode Count fields in the UHPHS_USBCMD register.

- **IST: Isochronous Scheduling Threshold**

The default value is Implementation dependent.

This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is 0, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is set to 1, then host software assumes the host controller may cache an isochronous data structure for an entire frame.

- **EECP: EHCI Extended Capabilities Pointer**

The default value is Implementation dependent.

This optional field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device.

39.7.4 UHPHS USB Command Register

Name: UHPHS_USBCMD

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
ITC							
15	14	13	12	11	10	9	8
–				ASPME	–	ASPMC	
7	6	5	4	3	2	1	0
LHCR	IAAD	ASE	PSE	FLS		HCRESET	RS

The Command Register indicates the command to be executed by the serial bus host controller. Writing to the register causes a command to be executed.

- **RS: Run/Stop (read/write)**

0: Stop (default value).

1: Run.

When set to 1, the Host Controller proceeds with execution of the schedule. The Host Controller continues execution as long as this bit is set to 1. When this bit is set to 0, the Host Controller completes the current and any actively pipelined transactions on the USB and then halts. The Host Controller must halt within 16 micro-frames after software clears the Run bit. The HC Halted bit in the status register indicates when the Host Controller has finished its pending pipelined transactions and has entered the stopped state. Software must not write 1 to this field unless the host controller is in the Halted state (i.e., HCHalted in the UHPHS_USBSTS register is 1). Doing so will yield undefined results.

- **HCRESET: Host Controller Reset (read/write)**

This control bit is used by software to reset the host controller. The effects of this on Root Hub registers are similar to a Chip Hardware Reset.

When software writes a 1 to this bit, the Host Controller resets its internal pipelines, timers, counters, state machines, etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports.

PCI Configuration registers are not affected by this reset. All operational registers, including port registers and port state machines, are set to their initial values. Port ownership reverts to the companion host controller(s) with side effects. Software must reinitialize the host controller in order to return the host controller to an operational state.

This bit is set to 0 by the Host Controller when the reset process is complete. Software cannot terminate the reset process early by writing a 0 to this register.

Software should not set this bit to 1 when the HCHalted bit in the UHPHS_USBSTS register is 0. Attempting to reset an actively running host controller will result in undefined behavior.

- **FLS: Frame List Size (read/write or read-only)**

This field is R/W only if Programmable Frame List Flag in the UHPHS_HCCPARAMS registers is set to 1. This field specifies the size of the frame list. The size of the frame list controls which bits in the Frame Index Register should be used for the Frame List Current index.

00b: 1024 elements (4096 bytes) (default value).

01b: 512 elements (2048 bytes).

10b: 256 elements (1024 bytes), for resource-constrained environments.

11b: Reserved.

- **PSE: Periodic Schedule Enable (read/write)**

This bit controls whether the host controller skips processing the Periodic Schedule.

0: Do not process the Periodic Schedule (default value).

1: Use the UHPHS_PERIODICLISTBASE register to access the Periodic Schedule.

- **ASE: Asynchronous Schedule Enable (read/write)**

This bit controls whether the host controller skips processing the Asynchronous Schedule.

0: Do not process the Asynchronous Schedule (default value).

1: Use the UHPHS_ASYNCLISTADDR register to access the Asynchronous Schedule.

- **IAAD: Interrupt on Async Advance Doorbell (read/write)**

This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell.

When the host controller has evicted all appropriate cached schedule state, it sets the Interrupt on Async Advance status bit in the UHPHS_USBSTS register. If the Interrupt on Async Advance Enable bit in the UHPHS_USBINTR register is set to 1, then the host controller will assert an interrupt at the next interrupt threshold.

The host controller sets this bit to 0 after it has set the Interrupt on Async Advance status bit in the UHPHS_USBSTS register to 1.

Software should not write a 1 to this bit when the asynchronous schedule is disabled. Doing so will yield undefined results.

- **LHCR: Light Host Controller Reset (optional) (read/write)**

This control bit is not required. If implemented, it allows the driver to reset the EHCI controller without affecting the state of the ports or the relationship to the companion host controllers. For example, the UHPHS_PORTSC registers should not be reset to their default values and the CF bit setting should not go to 0 (retaining port ownership relationships).

A host software read of this bit as 0 indicates the Light Host Controller Reset has completed and it is safe for host software to reinitialize the host controller. A host software read of this bit as 1 indicates the Light Host Controller Reset has not yet completed.

If not implemented, a read of this field will always return a 0.

- **ASPMC: Asynchronous Schedule Park Mode Count (optional) (read/write or read-only)**

If the Asynchronous Park Capability bit in the UHPHS_HCCPARAMS register is set to 1, then this field defaults to 3h and is R/W. Otherwise it defaults to 0 and is RO. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a 0 to this bit when Park Mode Enable is set to 1 as this will result in undefined behavior.

- **ASPME: Asynchronous Schedule Park Mode Enable (optional) (read/write or read-only)**

If the Asynchronous Park Capability bit in the UHPHS_HCCPARAMS register is set to 1, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a 0 and is RO. Software uses this bit to enable or disable Park mode. When this bit is set to 1, Park mode is enabled. When this bit is set to 0, Park mode is disabled.

- **ITC: Interrupt Threshold Control (read/write)**

This field is used by system software to select the maximum rate at which the host controller will issue interrupts. The only valid values are defined below. If software writes an invalid value to this register, the results are undefined.

Value	Maximum Interrupt Interval
00h	Reserved
01h	1 micro-frame
02h	2 micro-frames
04h	4 micro-frames
08h	8 micro-frames (default, equates to 1 ms)
10h	16 micro-frames (2 ms)
20h	32 micro-frames (4 ms)
40h	64 micro-frames (8 ms)

Any other value in this register yields undefined results.

Software modifications to this bit while HCHalted bit is equal to 0 results in undefined behavior.

39.7.5 UHPHS USB Status Register

Name: UHPHS_USBSTS

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
ASS	PSS	RCM	HCHLT	–			
7	6	5	4	3	2	1	0
–		IAA	HSE	FLR	PCD	USBERRINT	USBINT

This register indicates pending interrupts and various states of the Host Controller. The status resulting from a transaction on the serial bus is not indicated in this register. Software sets a bit to 0 in this register by writing a 1 to it.

- **USBINT: USB Interrupt (read/write clear)**

The Host Controller sets this bit to 1 on the completion of a USB transaction, which results in the retirement of a Transfer Descriptor that had its IOC bit set.

The Host Controller also sets this bit to 1 when a short packet is detected (the actual number of bytes received was less than the expected number of bytes).

- **USBERRINT: USB Error Interrupt (read/write clear)**

The Host Controller sets this bit to 1 when completion of a USB transaction results in an error condition (e.g., error counter underflow). If the TD on which the error interrupt occurred also had its IOC bit set, both this bit and USBINT bit are set.

- **PCD: Port Change Detect (read/write clear)**

The Host Controller sets this bit to 1 when any port for which the Port Owner bit is set to 0 (see Section 39.7.12) has a change bit transition from 0 to 1 or a Force Port Resume bit transition from 0 to 1 as a result of a J-K transition detected on a suspended port. This bit will also be set as a result of the Connect Status Change being set to 1 after system software has relinquished ownership of a connected port by writing 1 to a port's Port Owner bit.

This bit is allowed to be maintained in the Auxiliary power well. Alternatively, it is also acceptable that on a D3 to D0 transition of the EHCI HC device, this bit is loaded with the OR of all of the PORTSC change bits (including: Force Port Resume, Over-Current Change, Enable/Disable Change and Connect Status Change).

- **FLR: Frame List Rollover (read/write clear)**

The Host Controller sets this bit to 1 when the Frame List Index (see Section 39.7.7) rolls over from its maximum value to 0. The exact value at which the rollover occurs depends on the frame list size. For example, if the frame list size (as programmed in the Frame List Size field of the UHPHS_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX[13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to 1 every time FRINDEX[12] toggles.

- **HSE: Host System Error (read/write clear)**

The Host Controller sets this bit to 1 when a serious error occurs during a host system access involving the Host Controller module. In a PCI system, conditions that set this bit to 1 include PCI Parity error, PCI Master Abort, and PCI Target Abort. When this error occurs, the Host Controller clears the Run/Stop bit in the Command register to prevent further execution of the scheduled TDs.

- **IAA: Interrupt on Async Advance (read/write clear)**

0: Default.

System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing 1 to the Interrupt on the Async Advance Doorbell bit in the UHPHS_USBCMD register. This status bit indicates the assertion of that interrupt source.

- **HCHLT: HCHalted (read-only)**

1: Default.

This bit is 0 whenever the Run/Stop bit is 1. The Host Controller sets this bit to 1 after it has stopped executing as a result of the Run/Stop bit being set to 0, either by software or by the Host Controller hardware (e.g. internal error).

- **RCM: Reclamation (read-only)**

0: Default.

This is a read-only status bit used to detect any empty asynchronous schedule.

- **PSS: Periodic Schedule Status (read-only)**

0: Default.

The bit reports the current real status of the Periodic Schedule. If this bit is set to 0, then the status of the Periodic Schedule is disabled. If this bit is set to 1, then the status of the Periodic Schedule is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the UHPHS_USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).

- **ASS: Asynchronous Schedule Status (read-only)**

0: Default.

The bit reports the current real status of the Asynchronous Schedule. If this bit is set to 0, then the status of the Asynchronous Schedule is disabled. If this bit is set to 1, then the status of the Asynchronous Schedule is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the UHPHS_USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0).

39.7.6 UHPHS USB Interrupt Enable Register

Name: UHPHS_USBINTR

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–							
7	6	5	4	3	2	1	0
–		IAAE	HSEE	FLRE	PCIE	USBEIE	USBIE

This register enables and disables reporting of the corresponding interrupt to the software. When a bit is set and the corresponding interrupt is active, an interrupt is generated to the host. Interrupt sources that are disabled in this register still appear in the UHPHS_USBSTS to allow the software to poll for events.

Each interrupt enable bit description indicates whether it is dependent on the interrupt threshold mechanism.

For all enable register bits, 1= Enabled, 0= Disabled.

- **USBIE: USB Interrupt Enable**

When this bit is set to 1, and the USBINT bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBINT bit.

- **USBEIE: USB Error Interrupt Enable**

When this bit is set to 1, and the USBERRINT bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the USBERRINT bit.

- **PCIE: Port Change Interrupt Enable**

When this bit is set to 1, and the Port Change Detect bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Port Change Detect bit.

- **FLRE: Frame List Rollover Enable**

When this bit is set to 1, and the Frame List Rollover bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Frame List Rollover bit.

- **HSEE: Host System Error Enable**

When this bit is set to 1, and the Host System Error Status bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt. The interrupt is acknowledged by software clearing the Host System Error bit.

- **IAAE: Interrupt on Async Advance Enable**

When this bit is set to 1, and the Interrupt on Async Advance bit in the UHPHS_USBSTS register is 1, the host controller will issue an interrupt at the next interrupt threshold. The interrupt is acknowledged by software clearing the Interrupt on Async Advance bit.

39.7.7 UHPHS USB Frame Index Register

Name: UHPHS_FRINDEX

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–		FI					
7	6	5	4	3	2	1	0
FI							

This register is used by the host controller to index into the periodic frame list. The register updates every 125 µs (once each micro-frame). Bits [N:3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the UHPHS_USBCMD register (see Section 39.7.4).

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the Halted state as indicated by the HCHalted bit (UHPHS_USBSTS register, Section 39.7.5). A write to this register while the Run/Stop bit is set to 1 (UHPHS_USBCMD register, Section 39.7.4) produces undefined results. Writes to this register also affect the SOF value.

- **FI: Frame Index**

The value in this register increments at the end of each time frame (e.g. micro-frame). Bits [N:3] are used for the Frame List current index. This means that each location of the frame list is accessed eight times (frames or micro-frames) before moving to the next index. The following illustrates values of N based on the value of the Frame List Size field in the UHPHS_USBCMD register.

USBCMD [Frame List Size]	Number Elements	N
00b	(1024)	12
01b	(512)	11
10b	(256)	10
11b	Reserved	–

The SOF frame number value for the bus SOF token is derived or alternatively managed from this register. The value of FRINDEX must be 125 µs (1 micro-frame) ahead of the SOF token value. The SOF value may be implemented as an 11-bit shadow register. For this discussion, this shadow register is 11 bits and is named SOFV. SOFV updates every eight micro-frames (1 millisecond). An example implementation to achieve this behavior is to increment SOFV each time the FRINDEX[2:0] increments from 0 to 1.

Software must use the value of FRINDEX to derive the current micro-frame number, both for high-speed isochronous scheduling purposes and to provide the “get micro-frame number” function required for client drivers. Therefore, the value of FRINDEX and the value of SOFV must be kept consistent if chip is reset or software writes to FRINDEX. Writes to FRINDEX must also write-through FRINDEX[13:3] to SOFV[10:0]. In order to keep the update as simple as possible, software should never write a FRINDEX value where the three least significant bits are 111b or 000b.

39.7.8 UHPHS Control Data Structure Segment Register

Name: UHPHS_CTRLDSSEGMENT

Access: Read/Write

This 32-bit register corresponds to the most significant address bits [63:32] for all EHCI data structures. If the 64-bit Addressing Capability field in UHPHS_HCCPARAMS is set to 0, then this register is not used. Software cannot write to it and a read from this register will return zeros.

If the 64-bit Addressing Capability field in UHPHS_HCCPARAMS is 1, then this register is used with the link pointers to construct 64-bit addresses to EHCI control data structures. This register is concatenated with the link pointer from either the UHPHS_PERIODICLISTBASE, UHPHS_ASYNCLISTADDR, or any control data structure link field to construct a 64-bit address.

This register must be written as a DWord. Byte writes produce undefined results. This register allows the host software to locate all control data structures within the same 4-Gigabyte memory segment.

39.7.9 UHPHS Periodic Frame List Base Address Register

Name: UHPHS_PERIODICLISTBASE

Access: Read/Write

31	30	29	28	27	26	25	24
BA							
23	22	21	20	19	18	17	16
BA							
15	14	13	12	11	10	9	8
BA				–			
7	6	5	4	3	2	1	0
–							

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. If the host controller is in 64-bit mode (as indicated by a 1 in the 64-bit Addressing Capability field in the UHPHS_HCCSPARAMS register), then the most significant 32 bits of every control data structure address comes from the UHPHS_CTRLDSSEGMENT register (see Section 39.7.8). System software loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (UHPHS_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence. This register must be written as a DWord. Byte writes produce undefined results.

- **BA: Base Address (Low)**

These bits correspond to memory address signals [31:12], respectively.

39.7.10 UHPHS Asynchronous List Address Register

Name: UHPHS_ASYNCLISTADDR

Access: Read/Write

31	30	29	28	27	26	25	24
LPL							
23	22	21	20	19	18	17	16
LPL							
15	14	13	12	11	10	9	8
LPL							
7	6	5	4	3	2	1	0
LPL				–			

This 32-bit register contains the address of the next asynchronous queue head to be executed. If the host controller is in 64-bit mode (as indicated by a 1 in the 64-bit Addressing Capability field in the UHPHS_HCCPARAMS register), then the most significant 32 bits of every control data structure address comes from the UHPHS_CTRLDSSEGMENT register (See Section 39.7.8). Bits [4:0] of this register cannot be modified by system software and will always return a zero when read. The memory structure referenced by this physical memory pointer is assumed to be 32-byte (cache line) aligned. This register must be written as a DWord. Byte writes produce undefined results.

- **LPL: Link Pointer Low**

These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH).

39.7.11 UHPHS Configure Flag Register

Name: UHPHS_CONFIGFLAG

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–							
7	6	5	4	3	2	1	0
–							CF

This register is in the auxiliary power well. It is only reset by hardware when the auxiliary power is initially applied or in response to a host controller reset.

- **CF: Configure Flag (read/write)**

Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic. Bit values and side-effects are listed below.

0: Port routing control logic default-routes each port to an implementation-dependent classic host controller (default value).

1: Port routing control logic default-routes all ports to this host controller.

39.7.12 UHPHS Port Status and Control Register

Name: UHPHS_PORTSC_x[x = 0..2]

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–	WKOC_E	WKDSCNNT_E	WKCNNT_E	PTC			
15	14	13	12	11	10	9	8
PIC		PO	PP	LS		–	PR
7	6	5	4	3	2	1	0
SUS	FPR	OCC	OCA	PEDC	PED	CSC	CCS

A host controller must implement one or more port registers. The number of port registers implemented by a particular instantiation of a host controller is documented in the UHPHS_HCSPARAMS register (Section 39.7.2). Software uses this information as an input parameter to determine how many ports need to be serviced. All ports have the structure defined below.

This register is in the auxiliary power well. It is only reset by hardware when the auxiliary power is initially applied or in response to a host controller reset. The initial conditions of a port are:

- No device connected
- Port disabled

If the port has port power control, software cannot change the state of the port until after it applies power to the port by setting port power to a 1. Software must not attempt to change the state of the port until after power is stable on the port. The host is required to have power stable to the port within 20 milliseconds of the 0 to 1 transition.

Notes:
1. When a device is attached, the port state transitions to the connected state and system software will process this as with any status change notification.
2. If a port is being used as the Debug Port, then the port may report device connected and enabled when the Configured Flag is set to 0.

- **CCS: Current Connect Status (read-only)**

0: No device is present (default value).

1: Device is present on port.

This value reflects the current state of the port, and may not correspond directly to the event that caused the Connect Status Change bit (Bit 1) to be set.

This field is 0 if Port Power is 0.

- **CSC: Connect Status Change (read/write clear)**

0: No change (default value).

1: Change in Current Connect Status.

Indicates a change has occurred in the port's Current Connect Status. The host controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be "setting" an already-set bit (i.e., the bit will remain set). Software sets this bit to 0 by writing a 1 to it.

This field is 0 if Port Power is 0.

- **PED: Port Enabled/Disabled (read/write)**

0: Disable (default value).

1: Enable.

Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a 1 to this field. The host controller will only set this bit to 1 when the reset sequence determines that the attached device is a high-speed device.

Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events.

When the port is disabled (0b), downstream propagation of data is blocked on this port, except for reset.

This field is 0 if Port Power is 0.

- **PEDC: Port Enable/Disable Change (read/write clear)**

0: No change (default value).

1: Port enabled/disabled status has changed.

For the root hub, this bit gets set to 1 only when a port is disabled due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification for the definition of a Port Error). Software clears this bit by writing a 1 to it.

This field is 0 if Port Power is 0.

- **OCA: Over-current Active (read-only)**

0: This port does not have an over-current condition (default value).

1: This port currently has an over-current condition.

This bit will automatically transition from 1 to 0 when the over current condition is removed.

- **OCC: Over-current Change (read/write clear)**

0: Default value.

1: This bit gets set to 1 when there is a change to Over-current Active.

Software clears this bit by writing 1 to this bit position.

- **FPR: Force Port Resume (read/write)**

0: No resume (K-state) detected/driven on port (default value).

1: Resume detected/driven on port.

This functionality defined for manipulating this bit depends on the value of the Suspend bit. For example, if the port is not suspended (Suspend and Enabled bits are set to 1) and software transitions this bit to 1, then the effects on the bus are undefined.

Software sets this bit to a 1 to drive resume signaling. The Host Controller sets this bit to 1 if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to 1 because a J-to-K transition is detected, the Port Change Detect bit in the UHPHS_USBSTS register is also set to 1. If software sets this bit to 1, the host controller must not set the Port Change Detect bit.

Note that when the EHCI controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains set to 1. Software must appropriately time the Resume and set this bit to 0 when the appropriate amount of time has elapsed. Writing a 0 (from 1) causes the port to return to High-Speed mode (forcing the bus below the port into a high-speed idle).This bit will remain set to 1 until the port has switched to the high-speed idle. The host controller must complete this transition within 2 milliseconds of software setting this bit to 0.

This field is 0 if Port Power is 0.

- **SUS: Suspend (read/write)**

0: Port not in suspend state (default value).

1: Port in suspend state.

Port Enabled Bit and Suspend bit of this register define the port states as follows:

Bits [Port Enabled, Suspend]	Port State
0X	Disable
10	Enable
11	Suspend

When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction, if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the USB.

A write of 0 to this bit is ignored by the host controller. The host controller will unconditionally set this bit to 0 when:

- Software sets the Force Port Resume bit to 0 (from 1).
- Software sets the Port Reset bit to 1 (from 0).

If host software sets this bit to 1 when the port is not enabled (i.e., Port Enabled bit set to 0), the results are undefined.

This field is 0 if Port Power is set to 0.

- **PR: Port Reset (read/write)**

0: Port is not in Reset (default value).

1: Port is in Reset.

When software writes a 1 to this bit (from 0), the bus reset sequence as defined in the USB Specification Revision 2.0 is started. Software writes a 0 to this bit to terminate the bus reset sequence. Software must keep this bit set to 1 long enough to ensure the reset sequence, as specified in the USB Specification Revision 2.0, completes. Note: when software writes this bit to 1, it must also write 0 to the Port Enable bit.

When software writes a 0 to this bit, there may be a delay before the bit status changes to 0. The bit status will not read as 0 until after the reset has completed. If the port is in High-Speed mode after reset is complete, the host controller will automatically enable this port (e.g. set the Port Enable bit to 1). A host controller must terminate the reset and stabilize the state of the port within 2 milliseconds of software transitioning this bit from 1 to 0. For example: if the port detects that the attached device is high-speed during reset, then the host controller must have the port in the enabled state within 2 ms of software writing this bit to 0.

The HCHalted bit in the UHPHS_USBSTS register should be set to 0 before software attempts to use this bit. The host controller may hold Port Reset asserted to 1 when the HCHalted bit is 1.

This field is 0 if Port Power is 0.

- **LS: Line Status (read-only)**

These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines. These bits are used for detection of low-speed USB devices prior to the port reset and enable sequence. This field is valid only when the port enable bit is 0 and the current connect status bit is set to 1.

Bits are encoded as follows:

Value	USB State	Interpretation
00b	SE0	Not a low-speed device, perform EHCI reset
10b	J-state	Not a low-speed device, perform EHCI reset
01b	K-state	Low-speed device, release ownership of port
11b	Undefined	Not a low-speed device, perform EHCI reset

This value of this field is undefined if Port Power is 0.

- **PP: Port Power (read/write or read-only)**

The function of this bit depends on the value of the Port Power Control (PPC) field in the UHPHS_HCSPARAMS register. The behavior is as follows:

PPC	PP	Operation
0b	1b	Read-only. Host controller does not have port power control switches. Each port is hard-wired to power.
1b	1b/0b	Read/write. Host controller has port power control switches. This bit represents the current setting of the switch (0 = off, 1 = on). When power is not available on a port (i.e., PP at 0), the port is non-functional and will not report attaches, detaches, etc.

When an over-current condition is detected on a powered port and PPC is set to 1, the PP bit in each affected port may be transitioned by the host controller from 1 to 0 (removing power from the port).

- **PO: Port Owner (read/write)**

0: This bit unconditionally goes to a 0 when the Configured bit in the UHPHS_CONFIGFLAG register makes a 0 to 1 transition.

1: This bit unconditionally goes to 1 whenever the Configured bit is 0 (default value).

System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes 1 to this bit when the attached device is not a high-speed device. A 1 in this bit means that a companion host controller owns and controls the port.

- **PIC: Port Indicator Control (read/write)**

00b: Default value.

Writing to these bits has no effect if the P_INDICATOR bit in the UHPHS_HCSPARAMS register is set to 0. If the P_INDICATOR bit is set to 1, then the bits are encoded as follows:

Value	Meaning
00b	Port indicators are off
01b	Amber
10b	Green
11b	Undefined

Refer to the USB Specification Revision 2.0 for a description on how these bits are to be used.

This field is 0 if Port Power is 0.

- **PTC: Port Test Control (read/write)**

0000b: Default value.

When this field is set to 0, the port is NOT operating in a test mode. A non-zero value indicates that it is operating in test mode and the specific test mode is indicated by the specific value.

Test mode bits are encoded as follows (0110b - 1111b are reserved):

Value	Test Mode
0000b	Test mode not enabled
0001b	Test J_STATE
0010b	Test K_STATE
0011b	Test SE0_NAK
0100b	Test Packet
0101b	Test FORCE_ENABLE

Refer to the USB Specification Revision 2.0, Chapter 7, for details on each test mode.

- **WKCNNT_E: Wake on Connect Enable (read/write)**

0: Default value.

Writing this bit to 1 enables the port to be sensitive to device connects as wakeup events.

This field is 0 if Port Power is 0.

- **WKDSCNNT_E: Wake on Disconnect Enable (read/write)**

0: Default value.

Writing this bit to 1 enables the port to be sensitive to device disconnects as wakeup events.

This field is 0 if Port Power is 0.

- **WKOC_E: Wake on Over-current Enable (read/write)**

0: Default value.

Writing this bit to 1 enables the port to be sensitive to over-current conditions as wakeup events.

This field is 0 if Port Power is 0.

39.7.13 EHCI: REG00 - Programmable Microframe Base Value

Name: UHPHS_INSNREG00

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–				Debug			
15	14	13	12	11	10	9	8
Debug		MFC_8		MFC_16			
7	6	5	4	3	2	1	0
MFC_16							En

The Programmable Microframe Base Value is used to change the microframe length value (default is microframe SOF = 125 µs) in order to reduce simulation time.

- **En: Enable this Register**

0: Register disabled (default value).

1: Register enabled.

Note: Do not enable this register for the gate-level netlist.

- **MFC_16: Microframe Counter with Word Byte Interface**

This value is used as the 1-microframe counter with 16-bit interface.

- **MFC_8: Microframe Counter with Byte Interface**

This value is used as the 1-microframe counter with 8-bit interface.

- **Debug: Debug Purposes**

This field is used for debug purposes only.

In Heterogeneous mode, if the per port clock gets out of sync (but still within the ppm limits) of the phy_clk, then the per port SOF counter needs some correction relative to the global SOF counter. The RTL corrects itself if this happens.

This field controls the SOF correction, in case some debugging is required for the correction.

If bit 14 is set to 1, then it enables the RTL to use the value in bits 19:15 to perform the correction.

In normal operating mode, these bits should not be written.

Note:

The “value” in bits [31:1] must be programmed as follows: (value + 32/64) * Clock Period = microframe timer duration Factor 32 is used for a 16-bit interface and factor 64 is used for an 8-bit interface. For example, for the full (125 µs) microframe duration:

- In 8-bit, 60-MHz mode, the value is h1D0C (=7436), so (7436 + 64) * 16.67 ns = 125 µs
- In 16-bit, 30-MHz mode, the value is hE86 (=3718), so (3718 + 32) * 33.33 ns = 125 µs

For a 50 µs microframe duration:

- In 8-bit, 60-MHz mode, the value is hB77 (=2395), so (2395 + 64) * 16.67 ns = 50 µs
- In 16-bit, 30-MHz mode, the value is h5BC (=1468), so (1468 + 32) * 33.33 ns = 50 µs

39.7.14 EHCI: REG01 - Programmable Packet Buffer OUT/IN Thresholds

Name: UHPHS_INSNREG01

Access: Read/Write

31	30	29	28	27	26	25	24
Out_Threshold							
23	22	21	20	19	18	17	16
Out_Threshold							
15	14	13	12	11	10	9	8
In_Threshold							
7	6	5	4	3	2	1	0
In_Threshold							

Programmable Packet Buffer OUT/IN thresholds (in CONFIG1 mode only, not applicable in Config2 mode).

The value specified here is the number of DWORDs (32-bit entries).

- **In_Threshold: Amount of Data Available in the IN Packet Buffer**

The IN threshold is used to start the memory transfer as soon as the IN threshold amount of data is available in the Packet Buffer. It is also used to disconnect the data write, if the threshold amount of data is not available in the Packet Buffer.

- **Out_Threshold: Amount of Data Available in the OUT Packet Buffer**

The OUT threshold is used to start the USB transfer as soon as the OUT threshold amount of data is fetched from system memory. It is also used to disconnect the data fetch, if the threshold amount of space is not available in the Packet Buffer.

The minimum OUT and IN threshold amount that can be programmed through INSN registers is 16 bytes.

For INCRX configurations, the minimum threshold amount that can be programmed is the highest possible INCRX burst value. For example, if the value of the strap signals {ss_ena_incr16_i, ss_ena_incr8_i, ss_ena_incr4_i} is 3'b011 (for example, INCR16 burst is disabled, INCR8/INCR4 bursts are enabled), then the minimum OUT and IN threshold values should be 32 bytes (8 DWords).

OUT and IN threshold values can be equal to the packet buffer depth only when one of the following conditions is met:

- The packet buffer depth is equal to 512 bytes and isochronous/interrupt transactions are not initiated by the host controller.
- The packet buffer depth is equal to 1024 bytes.

The threshold default value depends on one of the following packet buffer configurations:

- 1024 bytes depth, 256 bytes IN and OUT thresholds
- 512 bytes depth, 128 bytes IN and OUT thresholds
- 256 bytes depth, 64 bytes IN and OUT thresholds
- 128 bytes depth, 64 bytes IN and OUT thresholds
- 64 bytes depth, 60 bytes IN and OUT thresholds

For INCRX configurations, the Break Memory Transfer bit is always enabled.

Depending on the different packet buffer settings, not all MSB bits are used.

39.7.15 EHCI: REG02 - Programmable Packet Buffer Depth

Name: UHPHS_INSNREG02

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–				Dwords			
7	6	5	4	3	2	1	0
Dwords							

Programmable Packet Buffer Depth (in CONFIG1 mode only, not applicable in Config2 mode).

The value specified here is the number of DWORDs (32-bit entries).

- **Dwords: Number of Entries**

For a maximum 256 entries for 1-Kbyte packet buffer, bits [8:0] are sufficient.

39.7.16 EHCI: REG03

Name: UHPHS_INSNREG03

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–	EN_CK256	Ignore_LS	Tx_Tx			Per_Frame	TA_Offset
7	6	5	4	3	2	1	0
TA_Offset							Break_Mem

The default value for INSNREG03[0] depends on the host core configuration. So, if INCRx support is enabled, this bit is 1 after reset. Otherwise, it should stay at 0.

- **Break_Mem: Break Memory Transfer (in CONFIG1 mode only, not applicable in CONFIG2 mode)**

0: Disables this function.

1: Enables this function.

Used in conjunction with INSNREG01 to enable breaking memory transactions into chunks once the OUT/IN threshold value is reached.

- **TA_Offset: Time-Available Offset**

This value indicates the additional number of bytes to be accommodated for the time-available calculation. The USB traffic on the bus can be started only when sufficient time is available to complete the packet within the EOF1 point.

Refer to the USB 2.0 specification for details of the EOF1 point. This time-available calculation is done in the hardware, and can be further offset by programming a value in this location.

Note: Time-available calculation is added for future flexibility. The application is not required to program this field by default.

- **Per_Frame: Periodic Frame List Fetch**

In CONFIG1 mode only (“EHCI Descriptor/Data Prefetching” is disabled in core configuration), setting this bit forces the host controller to fetch the periodic frame list in every microframe of a frame. If not set, then the periodic frame list is fetched only in microframe 0 of every frame.

The default is 0 (not set). This bit can be changed only during core initialization and should not be changed afterwards.

- **Tx_Tx: Tx-Tx turnaround Delay Add-on**

This field specifies the extra delays in phy_clks to be added to the “Transmit to Transmit turnaround delay” value maintained in the core. The default value of this register field is 0. This default value of 0 is sufficient for most PHYs. But for some PHYs which enter wait states during the token packet, it may be required to program a value greater than 0 to meet the transmit-to-transmit minimum turnaround time.

It is recommended to use default value 0 and to change it only if there is an issue with minimum transmit-to-transmit turnaround time.

This value should be programmed during core initialization and should not be changed afterwards.

- **Ignore_LS: Ignore Linestate during TestSE0 Nak**

When set to 1 (default), the core ignores the linestate checking when transmitting SOF in SE0_NAK Test mode.

When set to 0, the port state machine disables the port if it does not find the linestate to be in SE0 when transmitting SOF during the SE0_NAK test.

While performing impedance measurement during the SE0_NAK test, the linestate could go to non SE0 forcing the core to disable the port. This bit is used to control the port behavior during this operation.

- **EN_CK256: Enable 256 Clock Checking**

This bit controls the End of Resume sequence of the EHCI host controller.

By default, the value of this bit is 0 and during the End of Resume sequence, the host controller waits for SE0 on the linestate before switching the PHY to High-Speed.

When set to 1, during the End of Resume sequence, the controller waits for SE0 or 256 clocks before switching the PHY to High-Speed.

Setting this bit to 1 enables the 256-clock check. Some of the UTMI PHYs do not present SE0 on the linestate during the End of Resume sequence. For such PHYs, this bit should be set, so that the core does not wait forever for SE0.

This bit should be set only during initialization.

39.7.17 EHCI: REG04

Name: UHPHS_INSNREG04

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–							
7	6	5	4	3	2	1	0
–		EN_AutoFunc	NAK_RF	–	SDPE_TIME	HCCPARAMS_BW	HCSPARAMS_W

Bits [2:0] are used for debug purposes. Bits [(5+UHC2_N_PORTS):4] are functional bits where UHC2_N_PORTS indicates the number of physical USB ports.

- **HCSPARAMS_W: HCSPARAMS Write**

When set, the HCSPARAMS register becomes writable. Upon system reset, this bit is 0.

- **HCCPARAMS_BW: HCCPARAMS Bits Write**

When set, the HCCPARAMS register's bits 17, 15:4, and 2:0 become writable. Upon system reset, these bits are 0.

- **SDPE_TIME: Scales Down Port Enumeration Time**

When set, Scales Down Port Enumeration Time is enabled. Reset value is 1’b0.

Note: This bit can be used for both RTL and Gate level simulations.

- **NAK_RF: NAK Reload Fix (Read/Write)**

0: Enables this function.

1: Disables this function

Incorrect NAK reload transition at the end of a microframe for backward compatibility with Release 2.40c. For more information, see the *USB 2.0 Host-AHB Release Notes*. Reset value is 1’b0.

- **EN_AutoFunc: Enable Automatic Feature**

0: Enables the automatic feature.

The Suspend signal is deasserted (logic level 1'b1) when run/stop is reset by software, but the hchalted bit is not set yet.

1: Disables the automatic feature, which takes all ports out of suspend when software clears the run/stop bit. This is for backward compatibility.

Bit [5] has an added functionality in release 2.80a and later. For systems where the host is halted without waking up all ports out of suspend, the port can remain suspended because the PHYCLK is not running when the halt is programmed. To avoid this, the DWC H20AHB host core automatically pulls ports out of suspend when the host is halted by software.

This bit is used to disable this automatic function.

Reset value is 0.

39.7.18 EHCI: REG05 - UTMI Configuration

Name: UHPHS_INSNREG05

Access: Read/Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–						VBusy	VPort
15	14	13	12	11	10	9	8
VPort			VControlLoadM	VControl			
7	6	5	4	3	2	1	0
VStatus							

Control and Status Register, used to read the UTMI registers from the following signals:

- **VStatus: Vendor Status (Software RO)**
- **VControl: Vendor Control (Software R/W)**
- **VControlLoadM: Vendor Control Load Microframe**

0: Load.

1: NOP (software R/W)

- **VPort: Vendor Port (Software R/W)**

Valid values range from 1 to 15 depending on coreConsultant configuration.

For example, if the number of ports is 3, then software should only write values 1, 2, and 3 to this field and not any other values in the range, that is, 0 or 4 to 15. For example, if the software writes value 4 to VPort, from that write onwards, any write to this register is ignored and the read value will always be 4.

- **VBusy: Vendor Busy (Software RO)**

Hardware indicator that a write to this register has occurred and the hardware is currently processing the operation defined by the data written. When processing is finished, this bit is cleared.

39.7.19 EHCI: REG06 - AHB Error Status

Name: UHPHS_INSNREG06

Access: Read/Write

31	30	29	28	27	26	25	24
AHB_ERR							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–				HBURST			Nb_Burst
7	6	5	4	3	2	1	0
Nb_Burst				Nb_Success_Burst			

Control and Status Register, used to read the UTMI registers from the following signals:

- **Nb_Success_Burst: Number of Successful Bursts (read-only)**(1)

Number of successfully completed beats in the current burst before the AHB error occurred.

- **Nb_Burst: Number of Bursts (read-only)**(1)

Number of beats expected in the burst at which the AHB error occurred. Valid values are 0 to 16.

5’b10001–5b11111: Reserved

5’b00000–5b10000: Valid

- **HBURST: Burst Value (read-only)**(1)

Value of the control phase at which the AHB error occurred.

Note: 1. This field applies to AHB INCRX-enabled configurations only.

- **AHB_ERR: AHB Error**

AHB Error Captured Indicator that an AHB error was encountered and values were captured. To clear this field the application must write a 0 to it.

EHCI:

– When no error, 0 is written to INSNREG06[8:4].

– When INCR4 and an error occur, 4 is written to INSNREG06[8:4].

– When INCR8 and an error occur, 8 is written to INSNREG06[8:4].

– When INCR16 and an error occur, 16 is written to INSNREG06[8:4].

– Other values except 4, 8, and 16 are not written to INSNREG06[8:4].

OHCI:

– When no error, 0 is written to INSNREG06[8:4].

– When INCR4 and error occur, 4 is written to INSNREG06[8:4].

– Other values except 4 are not written to INSNREG06[8:4].

39.7.20 EHCI: REG07 - AHB Master Error Address

Name: UHPHS_INSNREG07

Access: Read Only

31	30	29	28	27	26	25	24
AHB_ADDR							
23	22	21	20	19	18	17	16
AHB_ADDR							
15	14	13	12	11	10	9	8
AHB_ADDR							
7	6	5	4	3	2	1	0
AHB_ADDR							

- **AHB_ADDR: AHB Address (read only)**

AHB address of the control phase at which the AHB error occurred.

39.7.21 EHCI: REG08 - HSIC Enable/Disable

Name: UHPHS_INSNREG08

Access: Read / Write

31	30	29	28	27	26	25	24
–							
23	22	21	20	19	18	17	16
–							
15	14	13	12	11	10	9	8
–							
7	6	5	4	3	2	1	0
–					HSIC_EN	–	

- **HSIC_EN: HSIC Enable/Disable**

This register has R/W access to the host driver and gives control to the host driver to enable/disable the HSIC interface of PORT C.

0: PORT C is in the HSIC Disable state (see *High-Speed Inter-Chip USB Electrical Specification*, Version 1.0, Section 3.1.2).). HSIC is in the Disabled state after a power-on reset.

1: PORT C is in the HSIC Enable state (see High-Speed Inter-Chip USB Electrical Specification, Version 1.0, Section 3.1.2).

40. Audio Class D Amplifier (CLASSD)

40.1 Description

The Audio Class D Amplifier (CLASSD) is a digital input, Pulse Width Modulated (PWM) output stereo Class D amplifier. It features a high quality interpolation filter embedding a digitally controlled gain, an equalizer and a de-emphasis filter.

On its input side, the CLASSD is compatible with most common audio data rates. and on the output side, its PWM output can drive either:

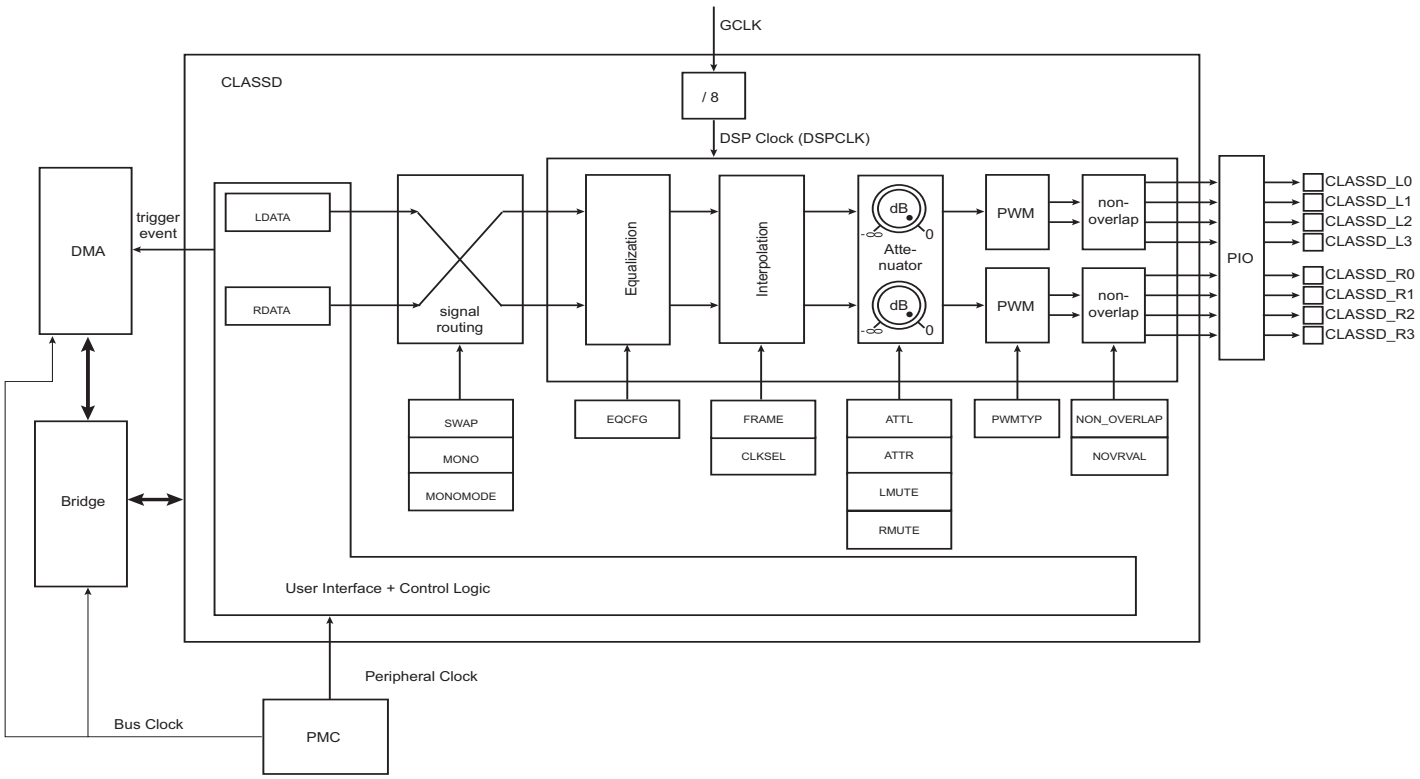
- high-impedance single-ended or differential output loads (Audio DAC application) or,
- external MOSFETs through an integrated non-overlapping circuit (Class D power amplifier application).

40.2 Embedded Characteristics

- Stereo PWM Class D amplifier
- 16-bit audio data
- DSP clocks: 12.288 and 11.2896 MHz
- Input sampling rates: 8, 16, 32, 48, 96, 22.05, 44.1, 88.2 kHz
- 3-band equalizer
- De-emphasis filter
- Digital volume control
- Differential or single-ended outputs
- Non-overlapping circuit to control external MOSFETs
- Supports DMA

40.3 Block Diagram

Figure 40-1. CLASSD Block Diagram



40.4 Pin Name List

Table 40-1. Output Pins Assignment Versus Application Use Cases

	External MOS Driver (NON_OVERLAP = 1)		Direct Load (NON_OVERLAP = 0)		
	Full H-Bridge (PWMTYP = 1)	Half H-Bridge (PWMTYP = 0)	Differential Load (PWMTYP = 1)	Single-Ended Load (PWMTYP = 0)	
Pin	Use Case 1	Use Case 2	Use Case 3A & 3B	Use Case 4A & 4B	Type
CLASSD_L0	gate_pmos_leftp	gate_pmos_left	leftp	left	Output
CLASSD_L1	gate_nmos_leftp	gate_nmos_left	Not used (fixed to 0)	Not used (fixed to 0)	Output
CLASSD_L2	gate_pmos_leftn	Not used (fixed to 1)	leftn	Not used (fixed to 0)	Output
CLASSD_L3	gate_nmos_leftn	Not used (fixed to 1)	Not used (fixed to 0)	Not used (fixed to 0)	Output
CLASSD_R0	gate_pmos_rightp	gate_pmos_right	rightp	right	Output
CLASSD_R1	gate_nmos_rightp	gate_nmos_right	Not used (fixed to 0)	Not used (fixed to 0)	Output
CLASSD_R2	gate_pmos_rightn	Not used (fixed to 1)	rightn	Not used (fixed to 0)	Output
CLASSD_R3	gate_nmos_rightn	Not used (fixed to 1)	Not used (fixed to 0)	Not used (fixed to 0)	Output

40.5 Product Dependencies

40.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the CLASSD pins to their peripheral functions.

Table 40-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
CLASSD	CLASSD_L0	PA28	F
CLASSD	CLASSD_L1	PA29	F
CLASSD	CLASSD_L2	PA30	F
CLASSD	CLASSD_L3	PA31	F
CLASSD	CLASSD_R0	PB1	F
CLASSD	CLASSD_R1	PB2	F
CLASSD	CLASSD_R2	PB3	F
CLASSD	CLASSD_R3	PB4	F

40.5.2 Power Management

The CLASSD is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the CLASSD Peripheral Clock and provide a generic clock (GCLK).

The fields NOVRVAL, NON_OVERLAP, PWMTYP in CLASSD_MR, and DSPCLKFREQ and FREQ in CLASSD_INTPMR, must be configured prior to applying the GCLK.

40.5.3 Interrupt

The CLASSD has an interrupt line connected to the interrupt controller. Handling the CLASSD interrupt requires programming the interrupt controller before configuring the CLASSD.

Table 40-3. Peripheral IDs

Instance	ID
CLASSD	59

40.6 Functional Description

This section describes the functionalities of the CLASSD interpolator, equalizer filters and the de-emphasis filter.

40.6.1 Interpolator

40.6.1.1 Clock Configuration

The interpolator accepts input sampling frequencies (f_s) and the input DSP clock (DSPCLK) that can be configured in the Interpolator Mode Register. GCLK must be configured in the PMC according to the desired DSPCLK so that DSPCLK = GCLK / 8.

The following table provides authorized DSPCLK / f_s ratios and associated filter types.

Table 40-4. Authorized DSPCLK / f_s Ratios & Filter Types

f_s	DSPCLK	
	12.288 MHz	11.2896 MHz
8 kHz	2	–[1]
16 kHz	2	–[1]
32 kHz	2	–[1]
48 kHz	1	–[1]
96 kHz	3	–[1]
22.05 kHz	–[1]	1
44.1 kHz	–[1]	1
88.2 kHz	–[1]	3

Note: 1. This configuration is not authorized and raises the CFGERR flag in the Interpolator Status Register.

40.6.1.2 CLASSD Frequency Response

Interpolation is performed with a combination of Infinite Impulse Response (IIR) and Cascaded Integrator-Comb (CIC) filters. Given the input configuration, filters' coefficients are redefined to optimize the filters' transfer function in order to optimize the audio bandwidth. The different types of filters are defined in Section 40.6.1.1 "Clock Configuration".

Figure 40-2. Type 1 Frequency Response

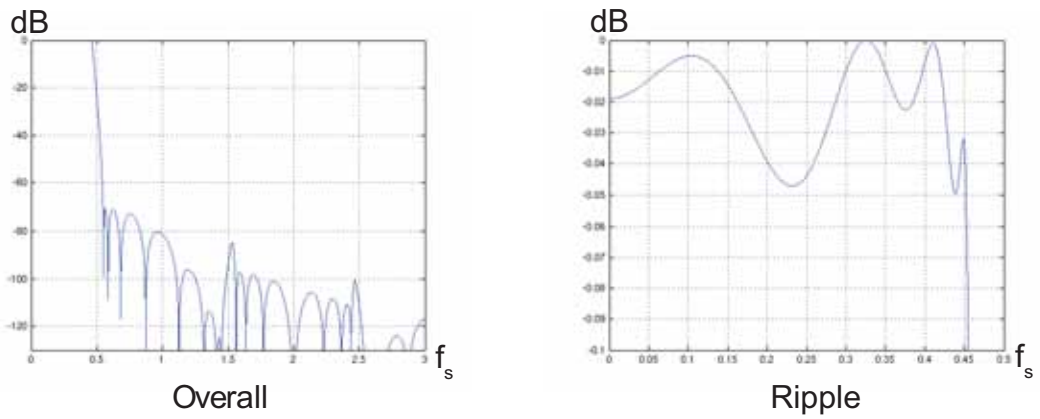


Figure 40-3. Type 2 Frequency Response

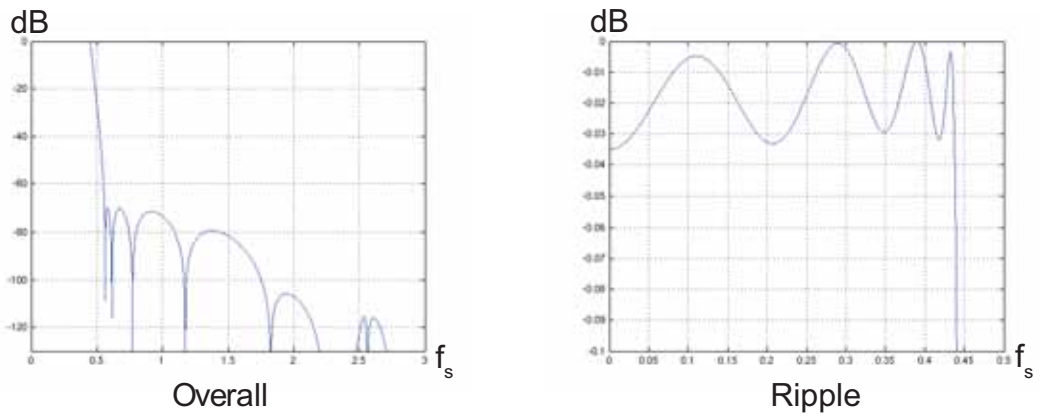
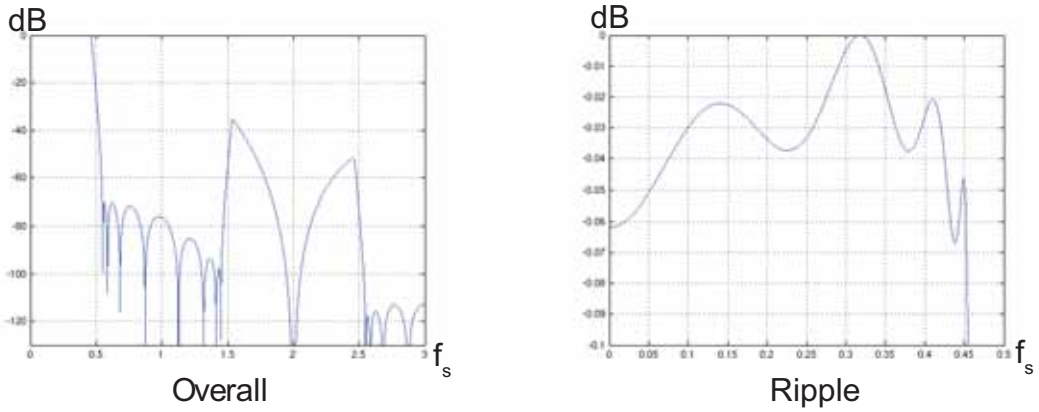


Figure 40-4. Type 3 Frequency Response



40.6.2 Equalizer

The CLASSD offers 12 preprogrammed equalization filters.

A zero-cross detection system allows to modify the equalizer on-the-fly with minimum perturbation on the output signal.

Section 40.7.3 “Interpolator Mode Register” details the programming of the equalization filter.

The following figures show the frequency response of the equalizer function implemented in the D/A channels.

Figure 40-5. Bass Filters Response

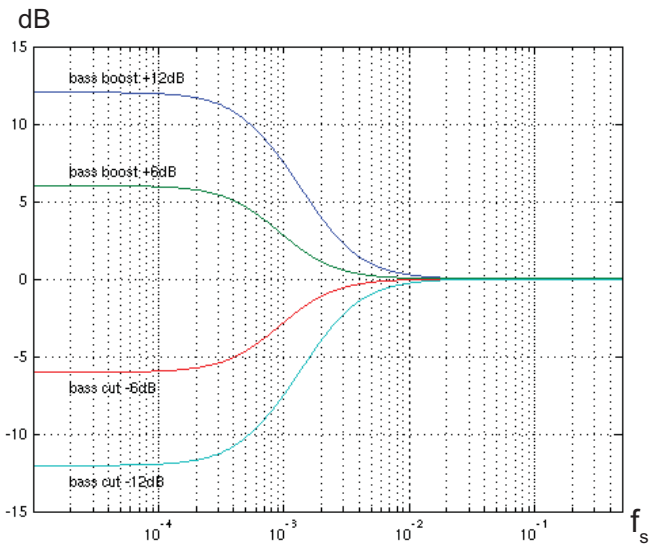


Figure 40-6. Medium Filters Response

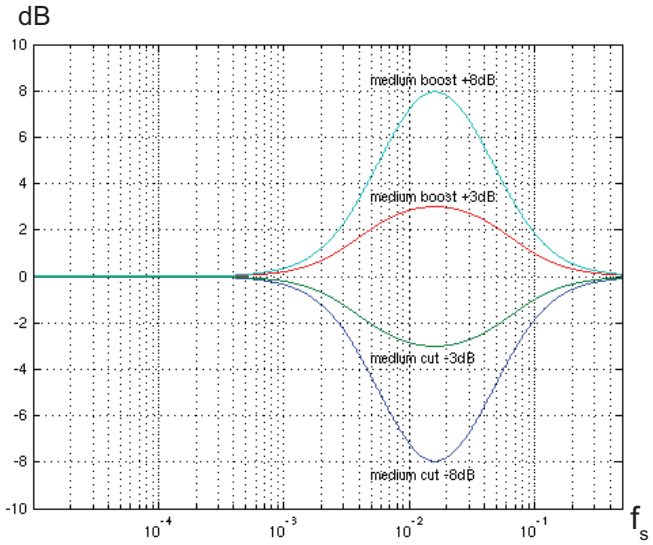
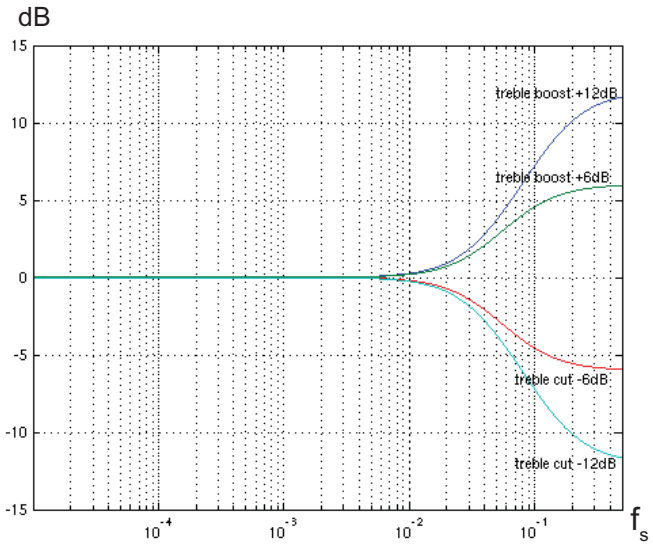


Figure 40-7. Treble Filters Response



40.6.3 De-emphasis Filter Frequency Response

The CLASSD includes a de-emphasis filter which can be enabled for 32, 44.1 or 48 kHz sampling frequencies.

The response and the error generated by the digital approximation of the filter are illustrated in the following figures.

Figure 40-8. De-emphasis Filter: Frequency Response & Error (f_s = 32 kHz)

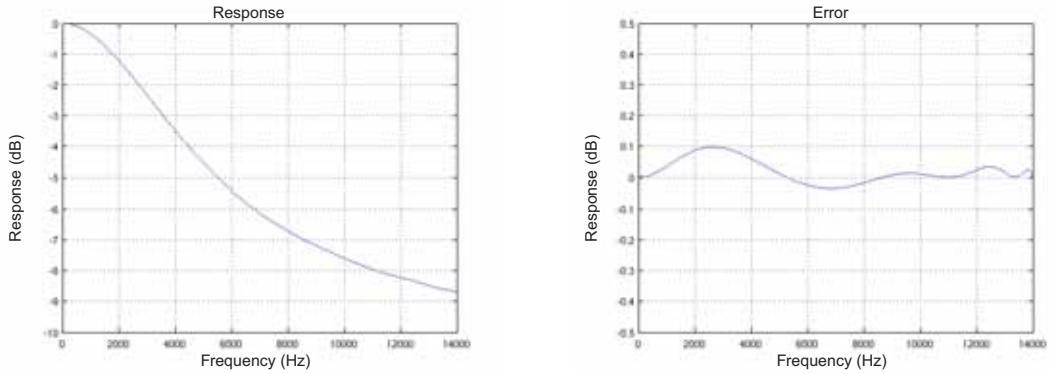


Figure 40-9. De-emphasis Filter: Frequency Response & Error (f_s = 44.1 kHz)

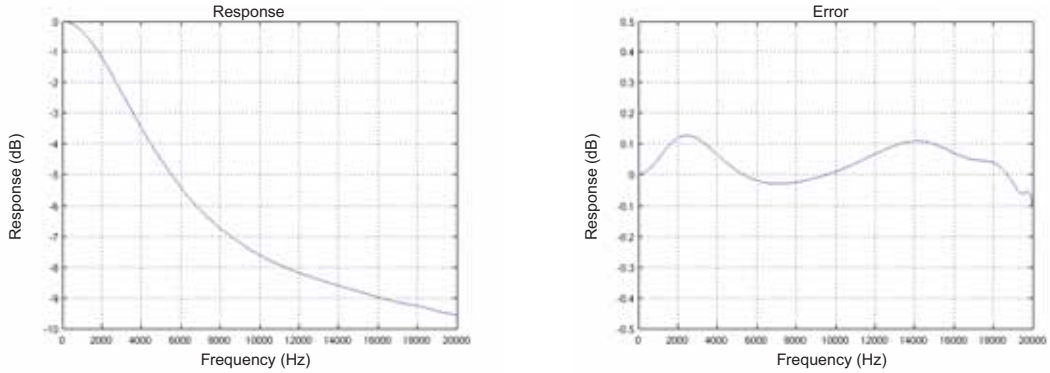
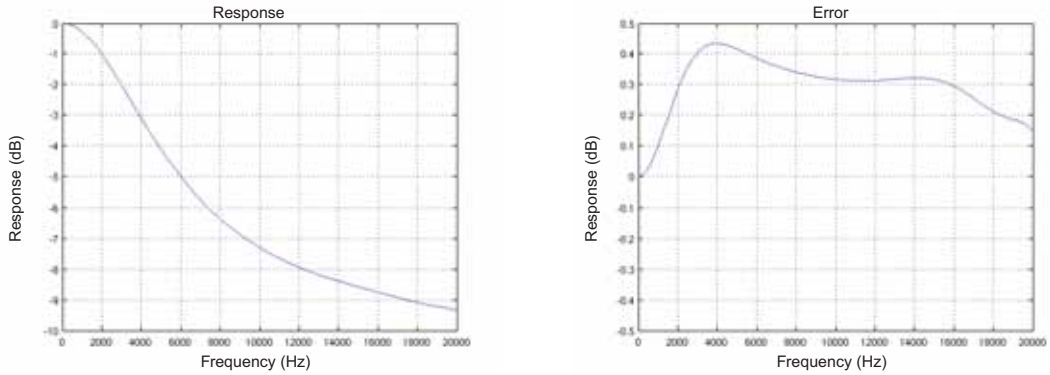


Figure 40-10. De-emphasis Filter: Frequency Response & Error (f_s = 48 kHz)



40.6.4 Attenuator and Recommended Input Levels

The CLASSD features a digital attenuator with an attenuation range of 0–77 dB and a step size of 1 dB. When a greater than 77 dB attenuation is programmed, the attenuator mutes the channel.

To avoid saturations in the PWM stage, it is recommended avoid input levels greater than 1 dB below the digital full scale (-1 dBFS). This can done by programming a minimum attenuation of 1 dB.

40.6.5 Pulse Width Modulator (PWM) Description

The CLASSD Pulse Width Modulator generates fixed frequency pulse width modulated output signals. For the 44.1 kS/s and 48 kS/s standard audio sample rates, the PWM output frequency is set to 16 × f_s: 705.6 kHz and 768 kHz respectively. For 8, 16, 24 and 96 kS/s, the 16× (interpolation) ratio is adapted to keep the output frequency at 768 kHz. By the same mechanism, the output frequency is 705.6 kHz for the 22.05 and 88.2 kS/s cases.

The CLASSD can work either as a DAC loaded by a medium to high resistive load (e.g., 1 kΩ to 100 kΩ) or as a Class D power amplifier controller driving an external power stage. Depending on the NON_OVERLAP bit value in the Mode Register (CLASSD_MR), the CLASSD will drive:

- Single-ended or differential resistive loads (NON_OVERLAP = 0)
- Full or Half MOSFET H-bridges (NON_OVERLAP = 1)

When driving an external power stage (NON_OVERLAP = 1), the CLASSD generates the signals to control complementary MOSFET pairs (PMOS and NMOS) with a non-overlapping delay between the NMOS and PMOS controls to avoid short circuit current. The non-overlapping delay can be adjusted in the CLASSD_MR.NOVRVAL field.

The CLASSD can have a single-ended or a differential output. A specific pulse width modulation type is associated to each case. For single-ended output (CLASSD_MR.PWMTYP = 0), the PWM acts only on the falling edge of the PWM waveform (trailing edge PWM). For differential output (CLASSD_MR.PWMTYP = 1), both the rising and the falling edges of the PWM waveform are modulated (symmetric PWM). Modulation principles are illustrated in Figure 40-11 and Figure 40-12 for both types of PWM. In particular, when describing a null input, if PWMTYP = 0 (trailing edge PWM), the output waveform is a square wave with 50% duty cycle. With the same input and PWMTYP = 1, the differential output waveform is zero. This difference allows to remove the classical L-C low pass filter when PWMTYP = 1.

Figure 40-11. Output Waveform Modulation Principle for PWMTYP = 0

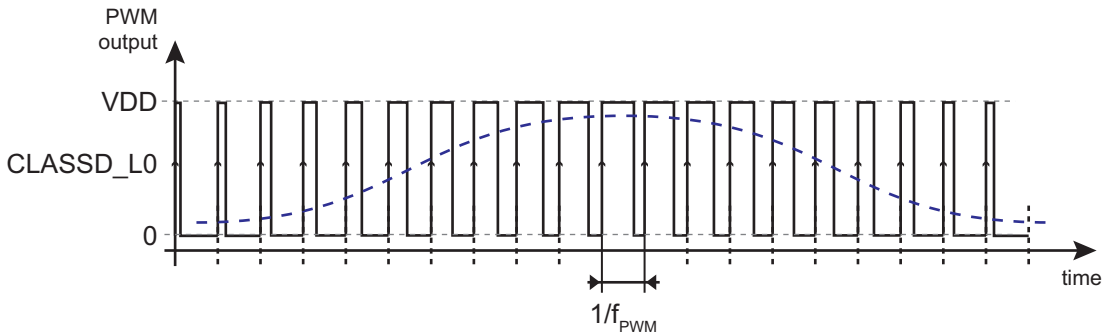
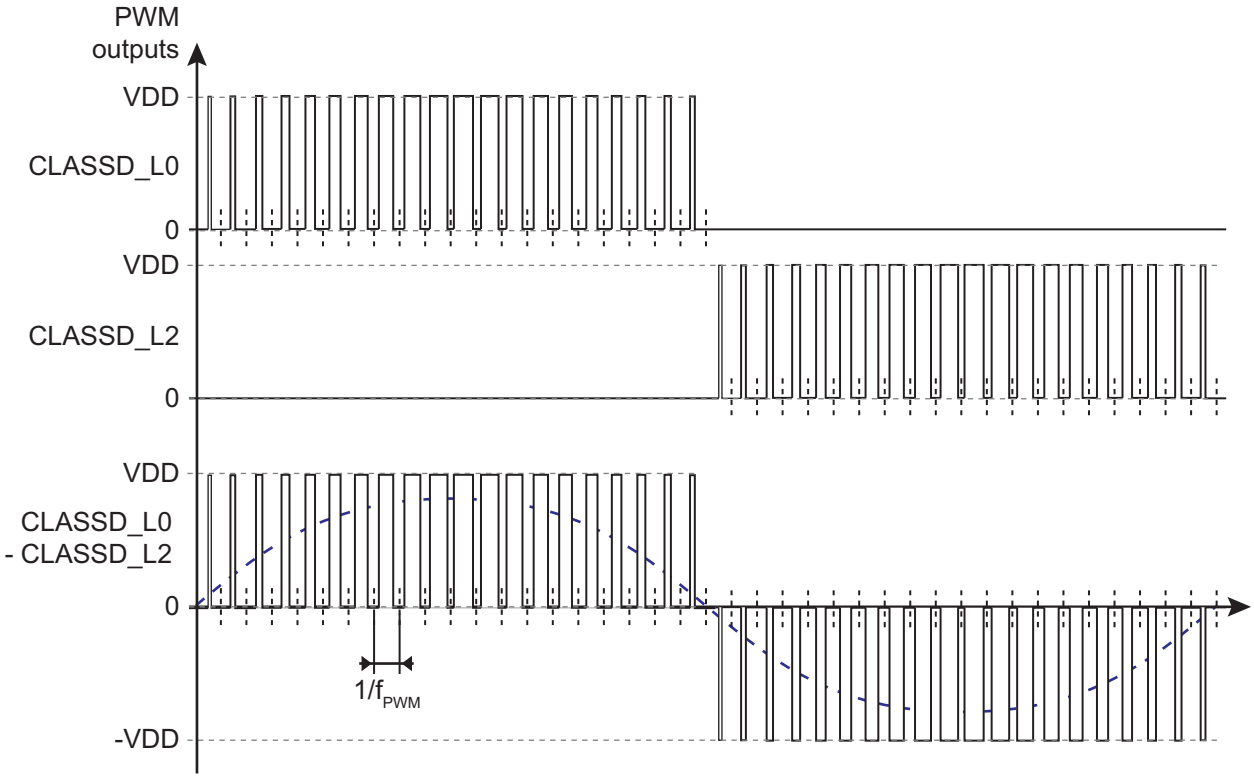
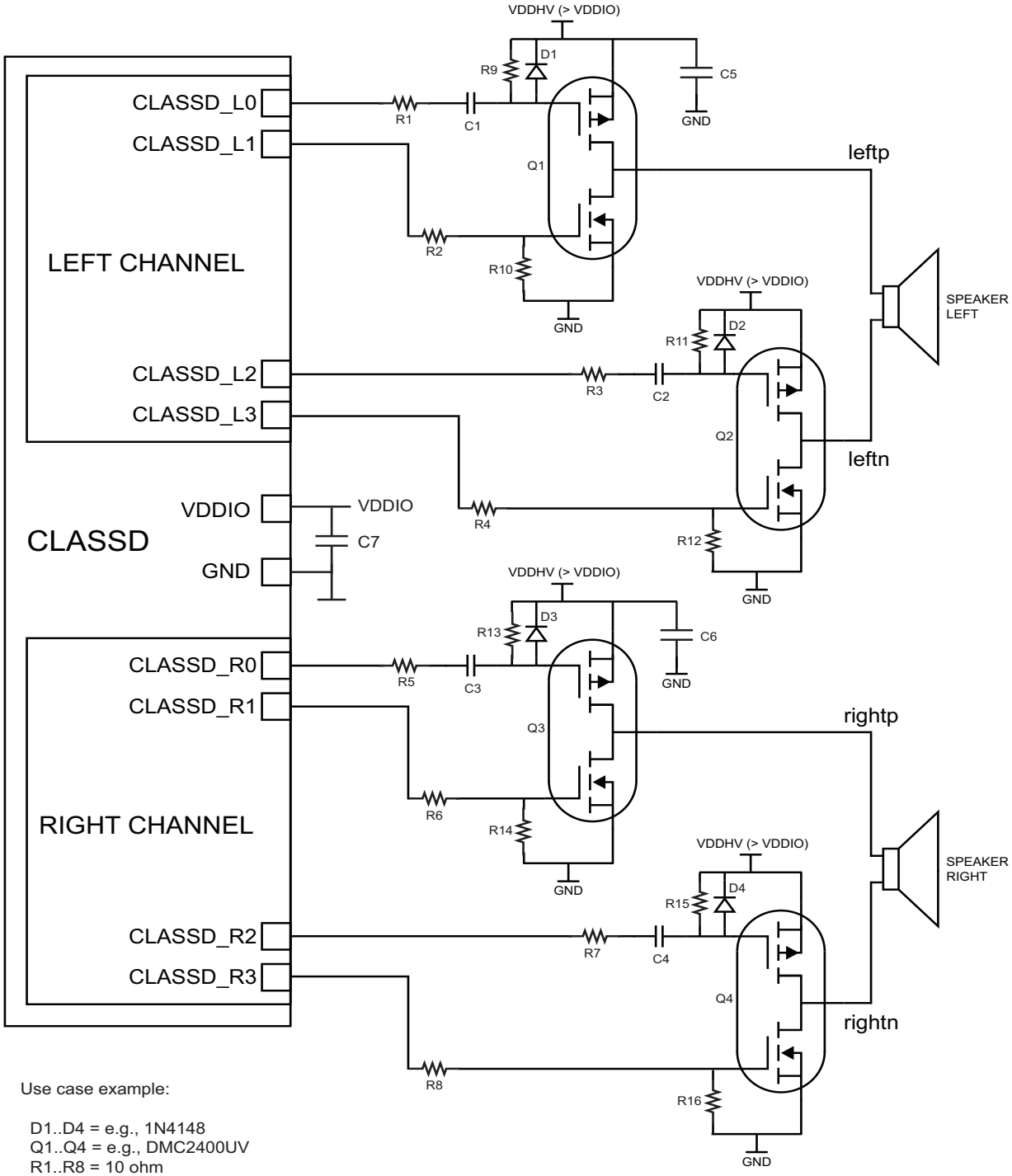


Figure 40-12. Output Waveform Modulation Principle for PWMTYP = 1 (Only Left Channel Pins Shown)



40.6.6 Application Schematics For Use Case Examples

Figure 40-13. Use Case 1: Stereo Class D Amplifier With External Differential Power Stage

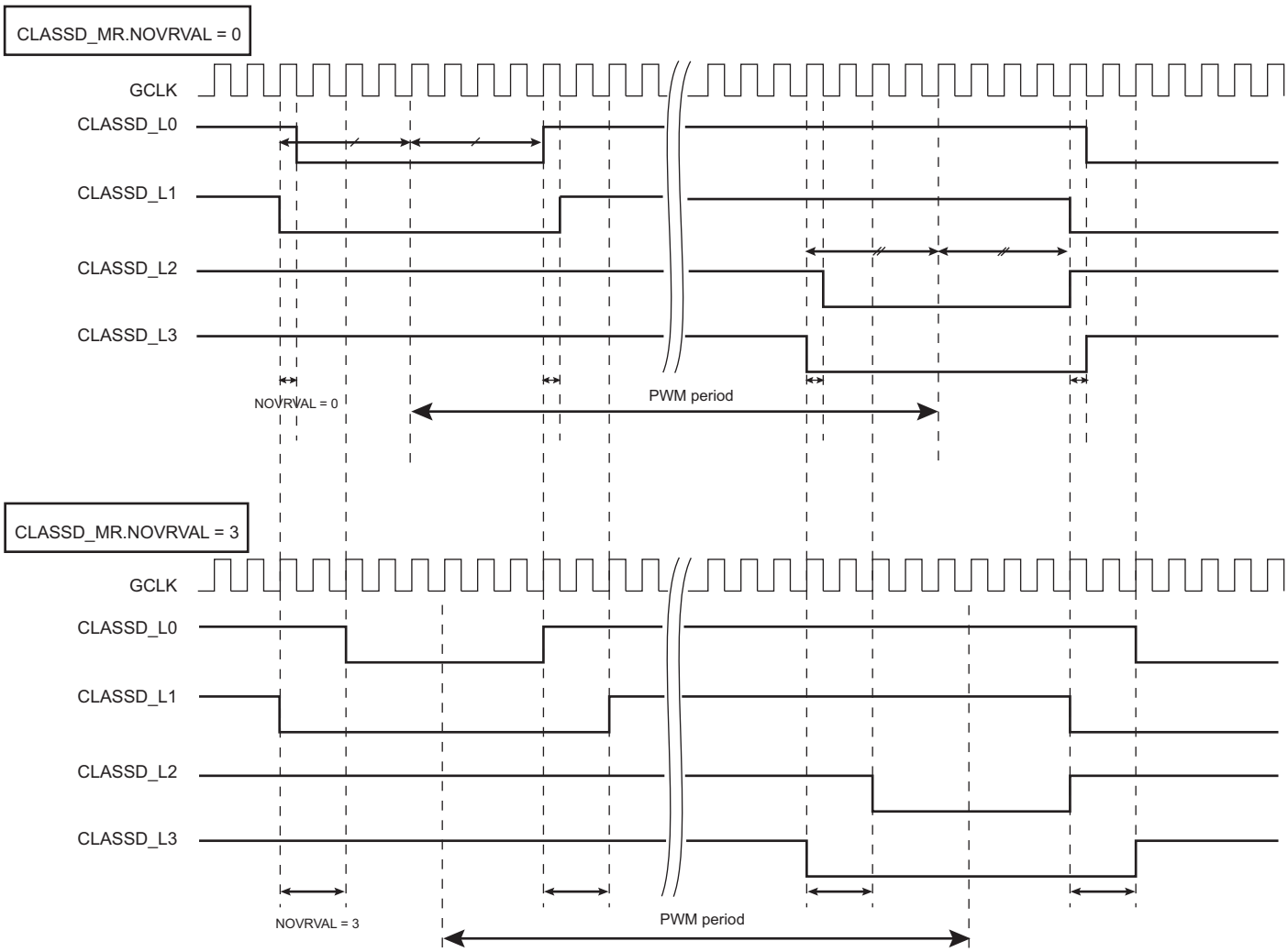


Use case example:

D1..D4 = e.g., 1N4148
Q1..Q4 = e.g., DMC2400UV
R1..R8 = 10 ohm
R9..R16 = 10 kohm
C1..C4 = 10 nF
C5..C6 = 10 µF
C7 = 1 µF

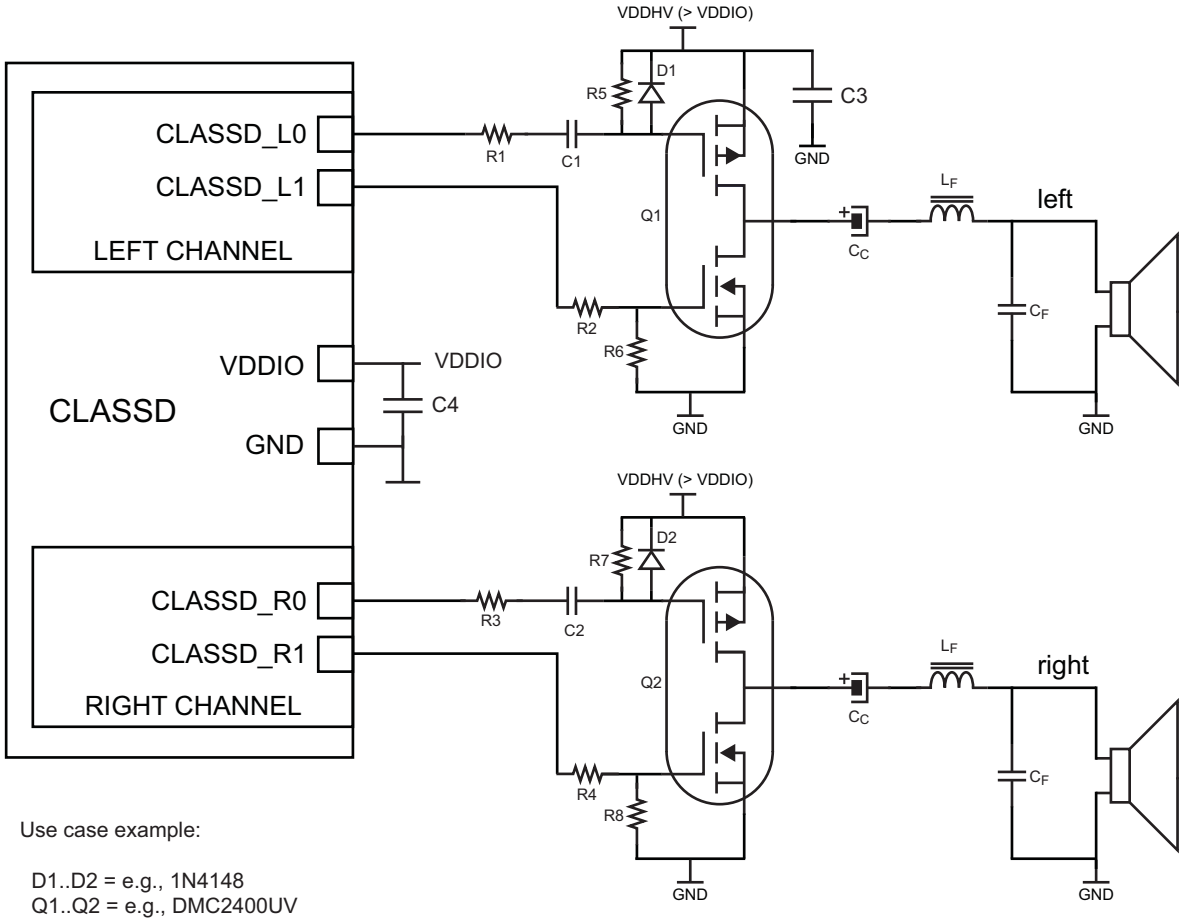
Figure 40-14. Use Case 1: Waveforms

CLASSD_MR.PWMTYP = 1, CLASSD_MR.NON_OVERLAP = 1



In use case 1, the external power stages are made of complementary low cost MOSFETs. On top of the usual R_{DSON} and drain breakdown voltage characteristics, the choice of these components is driven by a low gate threshold voltage and a low input capacitance characteristics. Series resistance (10 Ω) added to the gates of the MOSFETs are optional and may be adjusted to optimize the gate drive. They help to limit the output current peaks driven by the I/Os into the MOSFET gates in some cases. The 10k resistors ensure an OFF condition when not driven and the capacitor / diode network (C1..C2 / D1..D2) shifts the PMOS drive from the typical V_{DDIO} level (3.3V) to a higher supply voltage (e.g., a 5V power domain).

Figure 40-15. Use Case 2: Stereo Class D Amplifier With External Single-ended Power Stage

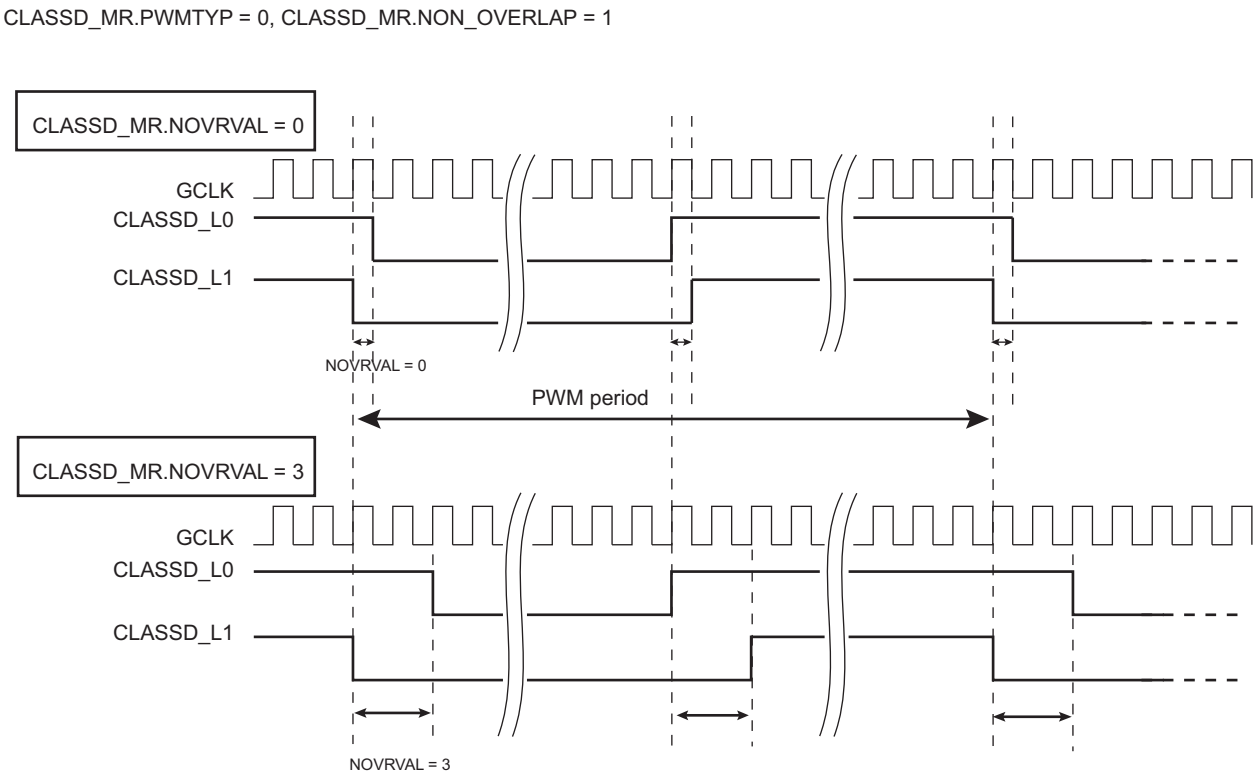


Use case example:

D1..D2 = e.g., 1N4148
Q1..Q2 = e.g., DMC2400UV
R1..R4 = 10 ohm
R5..R8 = 10 kohm
C1..C2 = 10 nF
C3 = 10 µF
C4 = 1 µF

In the use case 2 application schematic, the drive network of the MOSFETs gates follows the principles described in use case 1.

Figure 40-16. Use Case 2: Waveforms



A coupling capacitor (C_C) and an L-C low pass filter (L_F, C_F) are added to the output of the power stage to remove both the DC and the high frequency components of the PWM signal. C_C with the resistive part of the speaker (R_{SPK}) forms a C-R high pass filter with a corner frequency of $f_{HP} = 1 / (2 \times PI \times C_C \times R_{SPK})$.

L_F, C_F and R_{SPK} form a second order low pass filter of corner frequency $f_C = 1 / (2 \times PI \times sqrt (L_F \times C_F))$ and of quality factor $Q = R_{SPK} \times sqrt (C_F / L_F)$. As a numerical example, consider the case f_{HP} = 200 Hz, f_C = 30 kHz, Q = 0.707 (maximally flat response) with R_{SPK} = 8 Ω. This leads to C_C = 100 µF, L_F = 60 µH, C_F = 470 nF.

Figure 40-17. Use Case 3A: Stereo Audio DAC With Active Low Pass Filter and Single-ended Outputs

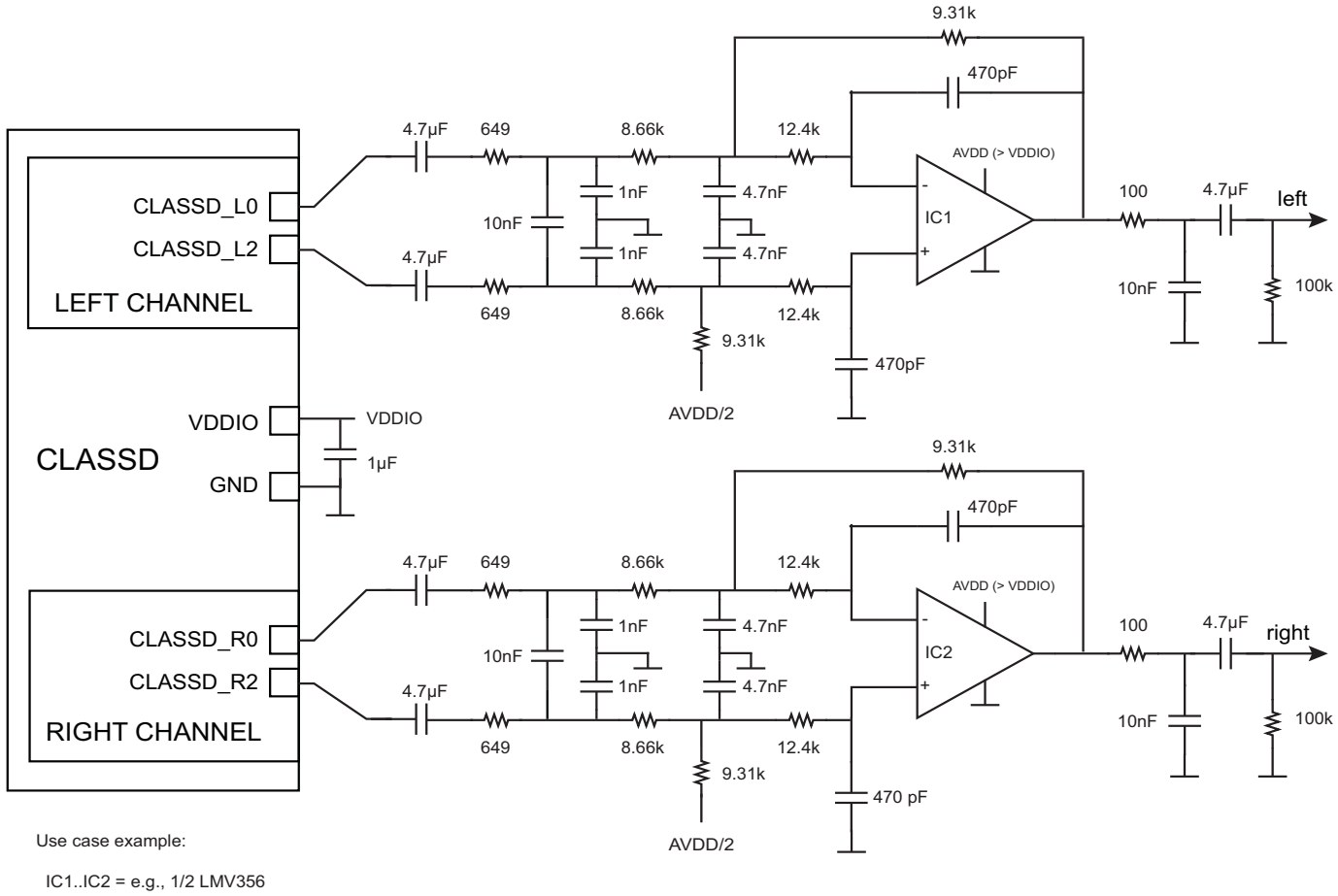


Figure 40-18. Use Case 3B: Stereo Audio DAC With Simple Passive Low Pass Filter and Differential Outputs

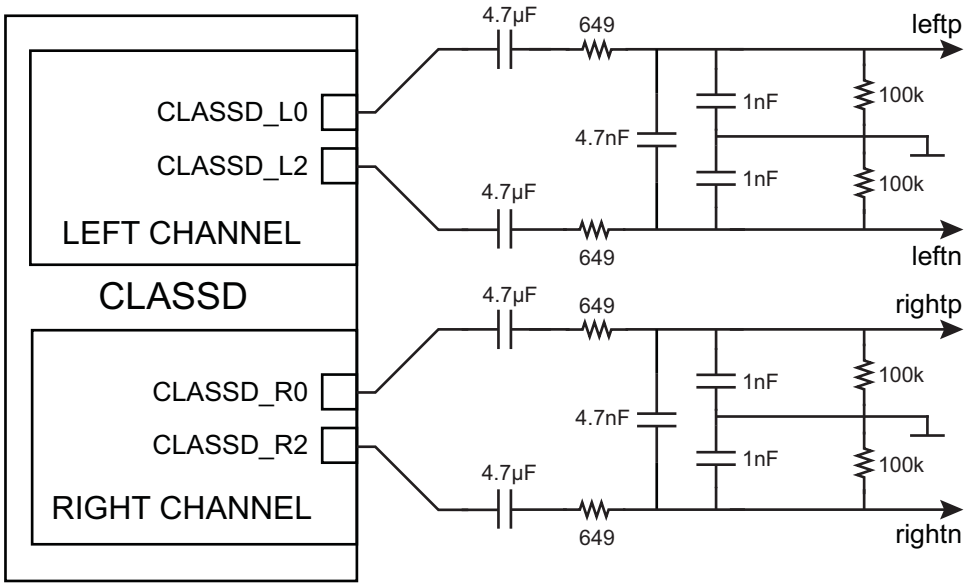
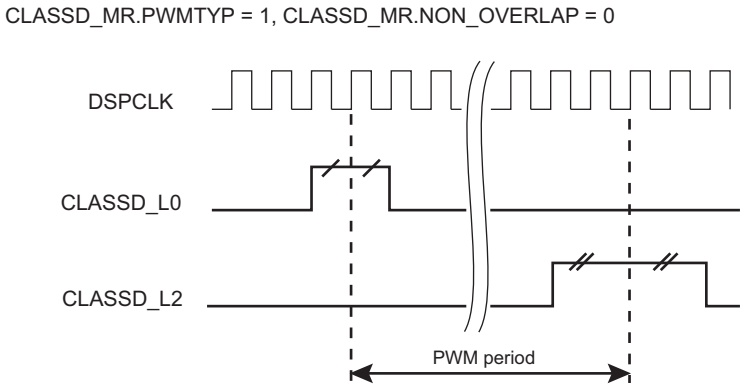


Figure 40-19. Use Case 3A and 3B: Waveforms



In use case 3A, the CLASSD is used as an audio DAC. In this case, the differential outputs of the CLASSD are used. The application schematic suggested in Figure 40-17 implements a third order 10 kHz low pass Butterworth filter and makes the differential to single-ended conversion. Note that in this schematic, the AVDD/2 point needs to be fed at low impedance (e.g., a buffered voltage). A simpler schematic (use case 3B) may also be possible as shown in Figure 40-18 at the cost of higher out-of band noise and differential outputs which may be acceptable in some applications.

Figure 40-20. Use Case 4A: Stereo Audio DAC With Active Low Pass Filter and Single-ended Outputs

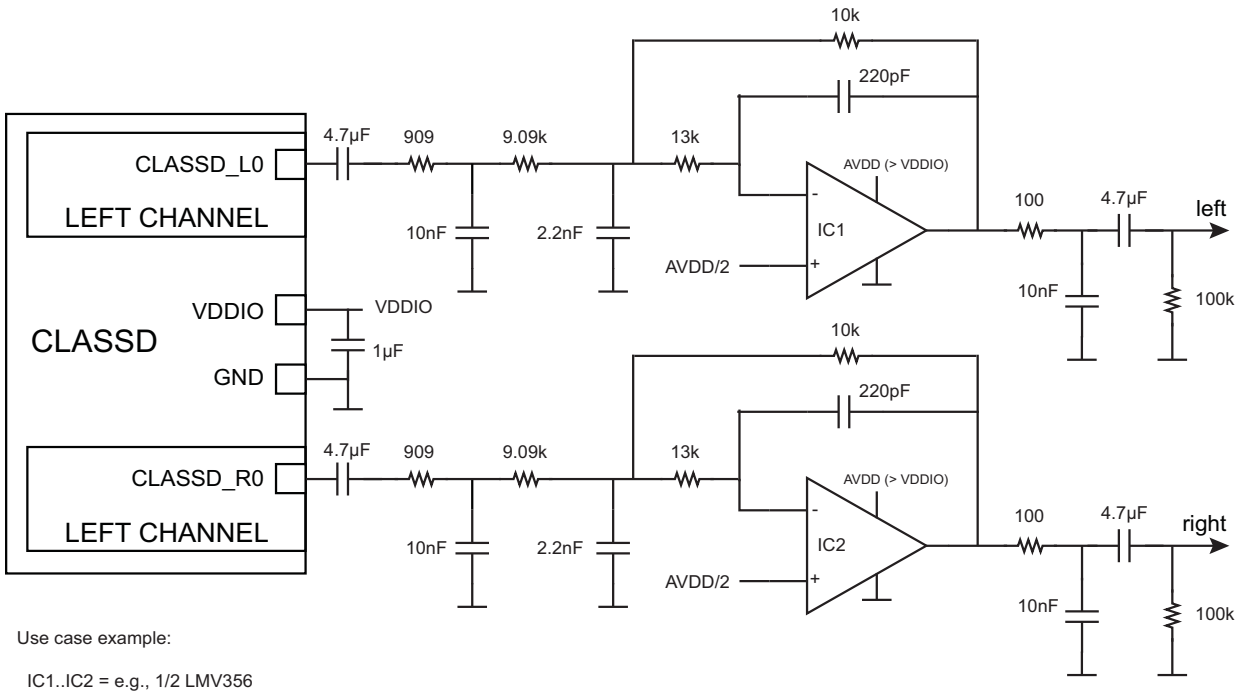


Figure 40-21. Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Single-ended Outputs

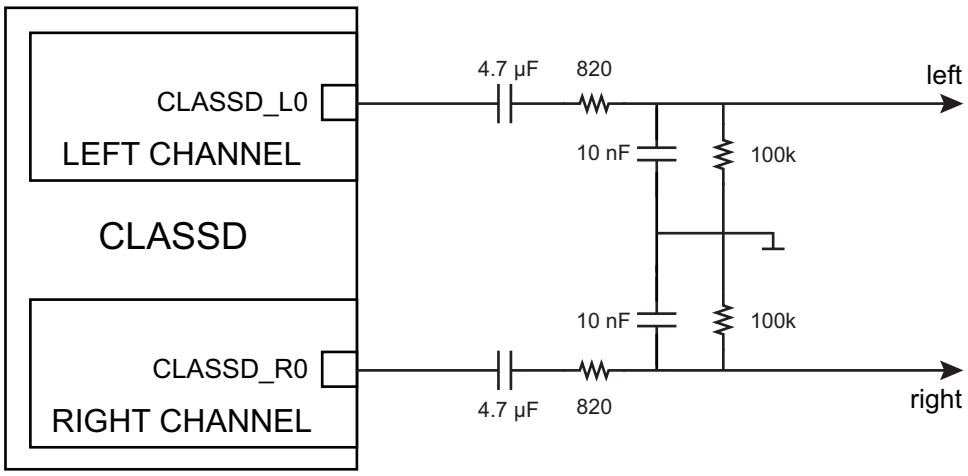
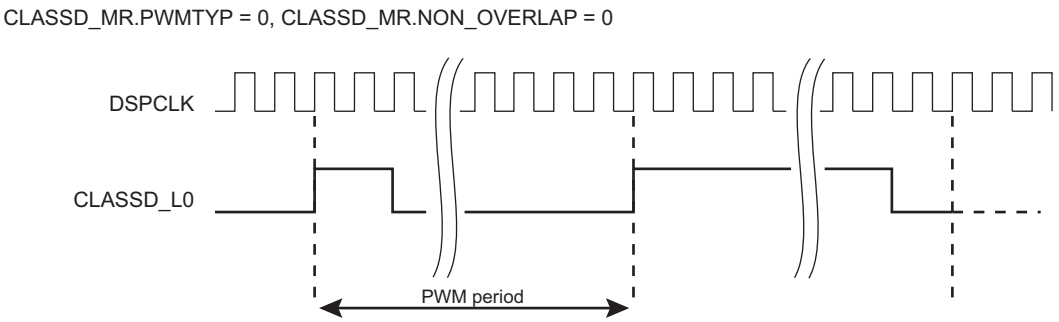


Figure 40-22. Use Case 4A and 4B: Waveforms



In use case 4A, the CLASSD is used as an audio DAC with active low pass filter. In this case, the single-ended-outputs of the CLASSD are selected (PWMTYP = 0, trailing edge PWM) which leaves more I/Os to the application. A third order 30 kHz low pass Butterworth filter is shown in Figure 40-20. The AVDD/2 point can be fed at relatively high impedance as no current is drawn from this point (a simple resistive divider properly decoupled is acceptable). A reduced complexity schematic is presented in Figure 40-21 for less constrained applications.

40.6.7 Register Write Protection

To prevent any single software error from corrupting CLASSD behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the Write Protection Mode Register (CLASSD_WPMR).

The following registers can be write-protected:

- Mode Register
- Interpolator Mode Register

40.7 Audio Class D Amplifier (CLASSD) User Interface

Table 40-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	CLASSD_CR	Write-only	–
0x04	Mode Register	CLASSD_MR	Read/Write	0x00010022
0x08	Interpolator Mode Register	CLASSD_INTPMR	Read/Write	0x00304E4E
0x0C	Interpolator Status Register	CLASSD_INTSR	Read-only	0x00000000
0x10	Transmit Holding Register	CLASSD_THR	Read/Write	0x00000000
0x14	Interrupt Enable Register	CLASSD_IER	Write-only	–
0x18	Interrupt Disable Register	CLASSD_IDR	Write-only	–
0x1C	Interrupt Mask Register	CLASSD_IMR	Read/Write	0x00000000
0x20	Interrupt Status Register	CLASSD_ISR	Read-only	0x00000000
0x24–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	CLASSD_WPMR	Read/Write	0x00000000
0xE8–0xFC	Reserved	–	–	–

40.7.1 Control Register

Name: CLASSD_CR

Address: 0xFC048000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SWRST

- **SWRST: Software Reset**

0: No effect

1: Reset the CLASSD simulating a hardware reset

40.7.2 Mode Register

Name: CLASSD_MR

Address: 0xFC048004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	NOVRVAL		–	–	–	NON_OVERLAP
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	PWMTYP
7	6	5	4	3	2	1	0
–	–	RMUTE	REN	–	–	LMUTE	LEN

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **LEN: Left Channel Enable**

0: Left channel is disabled

1: Left channel is enabled

- **LMUTE: Left Channel Mute**

0: Left channel is unmuted

1: Left channel is muted

- **REN: Right Channel Enable**

0: Right channel is disabled

1: Right channel is enabled

- **RMUTE: Right Channel Mute**

0: Right channel is unmuted

1: Right channel is muted

- **PWMTYP: PWM Modulation Type**

0 (TRAILING_EDGE): The signal is single-ended.

If bit NON_OVERLAP is cleared, the signal is sent to CLASSD_L0 and CLASSD_R0 (see Figure 40-20 or Figure 40-21).

If bit NON_OVERLAP is set, the signal is sent to CLASSD_L0/L1 and CLASSD_R0/R1 (see Figure 40-15).

1 (UNIFORM): The signal is differential.

If bit NON_OVERLAP is cleared, the signal is sent to CLASSD_L0/L2 and CLASSD_R0/R2 (see Figure 40-17 or Figure 40-18).

If bit NON_OVERLAP is set, the signal is sent to CLASSD_L0/L1/L2/L3 and CLASSD_R0/R1/R2/R3 (see Figure 40-13).

- **NON_OVERLAP: Non-Overlapping Enable**

0: Non-overlapping circuit is disabled

1: Non-overlapping circuit is enabled

- **NOVRVAL: Non-Overlapping Value**

Value	Name	Description
0	5NS	Non-overlapping time is 5 ns
1	10NS	Non-overlapping time is 10 ns
2	15NS	Non-overlapping time is 15 ns
3	20NS	Non-overlapping time is 20 ns

Note: This field has no effect when NON_OVERLAP = 0.

40.7.3 Interpolator Mode Register

Name: CLASSD_INTPMR

Address: 0xFC048008

Access: Read/Write

31	30	29	28	27	26	25	24
–	MONOMODE		MONO	EQCFG			
23	22	21	20	19	18	17	16
–	FRAME			SWAP	DEEMP	–	DSPCLKFREQ
15	14	13	12	11	10	9	8
–	ATTR						
7	6	5	4	3	2	1	0
–	ATTL						

This register can only be written if the WPEN bit is cleared in the Write Protection Mode Register.

- **ATTL: Left Channel Attenuation**

Left channel attenuation is defined as follows:

if ATTL ≤ 77 the attenuation is -ATTL dB

else the left signal is muted

- **ATTR: Right Channel Attenuation**

Right channel attenuation is defined as follows:

if ATTR ≤ 77 the attenuation is -ATTR dB

else the right signal is muted

- **DSPCLKFREQ: DSP Clock Frequency**

0 (12M288): DSP Clock (DSPCLK) is 12.288 MHz

1 (11M2896): DSP Clock (DSPCLK) is 11.2896 MHz

- **DEEMP: Enable De-emphasis Filter**

0 (DISABLED): De-emphasis filter is disabled

1 (ENABLED): De-emphasis filter is enabled

- **SWAP: Swap Left and Right Channels**

0 (LEFT_ON_LSB): Left channel is on CLASSD_THR[15:0], right channel is on CLASSD_THR[31:16]

1 (RIGHT_ON_LSB): Right channel is on CLASSD_THR[15:0], left channel is on CLASSD_THR[31:16]

- **FRAME: CLASSD Incoming Data Sampling Frequency**

Value	Name	Description
0	FRAME_8K	8 kHz
1	FRAME_16K	16 kHz
2	FRAME_32K	32 kHz
3	FRAME_48K	48 kHz
4	FRAME_96K	96 kHz
5	FRAME_22K	22.05 kHz
6	FRAME_44K	44.1 kHz
7	FRAME_88K	88.2 kHz

- **EQCFG: Equalization Selection**

Value	Name	Description
0	FLAT	Flat Response
1	BBOOST12	Bass boost +12 dB
2	BBOOST6	Bass boost +6 dB
3	BCUT12	Bass cut -12 dB
4	BCUT6	Bass cut -6 dB
5	MBOOST3	Medium boost +3 dB
6	MBOOST8	Medium boost +8 dB
7	MCUT3	Medium cut -3 dB
8	MCUT8	Medium cut -8 dB
9	TBOOST12	Treble boost +12 dB
10	TBOOST6	Treble boost +6 dB
11	TCUT12	Treble cut -12 dB
12	TCUT6	Treble cut -6 dB

Note: EQCFG field values 13–15 = Flat Response

- **MONO: Mono Signal**

0 (DISABLED): The signal is sent stereo to the left and right channels.

1 (ENABLED): The same signal is sent on both left and right channels. The sent signal is defined by the MONOMODE field value.

- **MONOMODE: Mono Mode Selection**

This field defines which signal is sent on both channels when the MONO bit is set.

Value	Name	Description
0	MONOMIX	(left + right) / 2 is sent on both channels
1	MONOSAT	(left + right) is sent to both channels. If the sum is too high, the result is saturated.
2	MONOLEFT	THR[15:0] is sent on both left and right channels
3	MONORIGHT	THR[31:16] is sent on both left and right channels

40.7.4 Interpolator Status Register

Name: CLASSD_INTSR

Address: 0xFC04800C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CFGERR

- **CFGERR: Configuration Error**

0: The frame and clock configuration are correct.

1: The frame and clock configuration are wrong (see Section 40.6.1.1 “Clock Configuration” for information about allowed configurations).

40.7.5 Transmit Holding Register

Name: CLASSD_THR

Address: 0xFC048010

Access: Read/Write

31	30	29	28	27	26	25	24
RDATA							

23	22	21	20	19	18	17	16
RDATA							

15	14	13	12	11	10	9	8
LDATA							

7	6	5	4	3	2	1	0
LDATA							

- **LDATA: Left Channel Data**
- **RDATA: Right Channel Data**

40.7.6 Interrupt Enable Register

Name: CLASSD_IER

Address: 0xFC048014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: No effect

1: Enables the interrupt when the CLASSD is ready to receive a new data to convert

40.7.7 Interrupt Disable Register

Name: CLASSD_IDR

Address: 0xFC048018

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: No effect

1: Disables the interrupt when the CLASSD is ready to receive a new data to convert

40.7.8 Interrupt Mask Register

Name: CLASSD_IMR

Address: 0xFC04801C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: The interrupt is disabled.

1: The interrupt is enabled.

40.7.9 Interrupt Status Register

Name: CLASSD_ISR

Address: 0xFC048020

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: CLASSD has not been ready to convert a value since the last read of CLASSD_ISR.

1: CLASSD is ready to convert a value since the last read of CLASSD_ISR.

40.7.10 Write Protection Mode Register

Name: CLASSD_WPMR

Address: 0xFC0480E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x434C44 (“CLD” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x434C44 (“CLD” in ASCII).

See Section 40.6.7 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x434C44	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

41. Inter-IC Sound Controller (I2SC)

41.1 Description

The Inter-IC Sound Controller (I2SC) provides a 5-wire, bidirectional, synchronous, digital audio link to external audio devices: I2SDI, I2SDO, I2SWS, I2SCK, and I2SMCK pins.

The I2SC is compliant with the Inter-IC Sound (I^2S) bus specification.

The I2SC consists of a receiver, a transmitter and a common clock generator that can be enabled separately to provide Master, Slave or Controller modes with receiver and/or transmitter active.

DMA Controller channels, separate for the receiver and for the transmitter, allow a continuous high bit rate data transfer without processor intervention to the following:

- Audio CODECs in Master, Slave, or Controller mode
- Stereo DAC or ADC through a dedicated I^2S serial interface

The I2SC uses a single DMA Controller channel for both audio channels.

The 8- and 16-bit compact stereo format reduces the required DMA Controller bandwidth by transferring the left and right samples within the same data word.

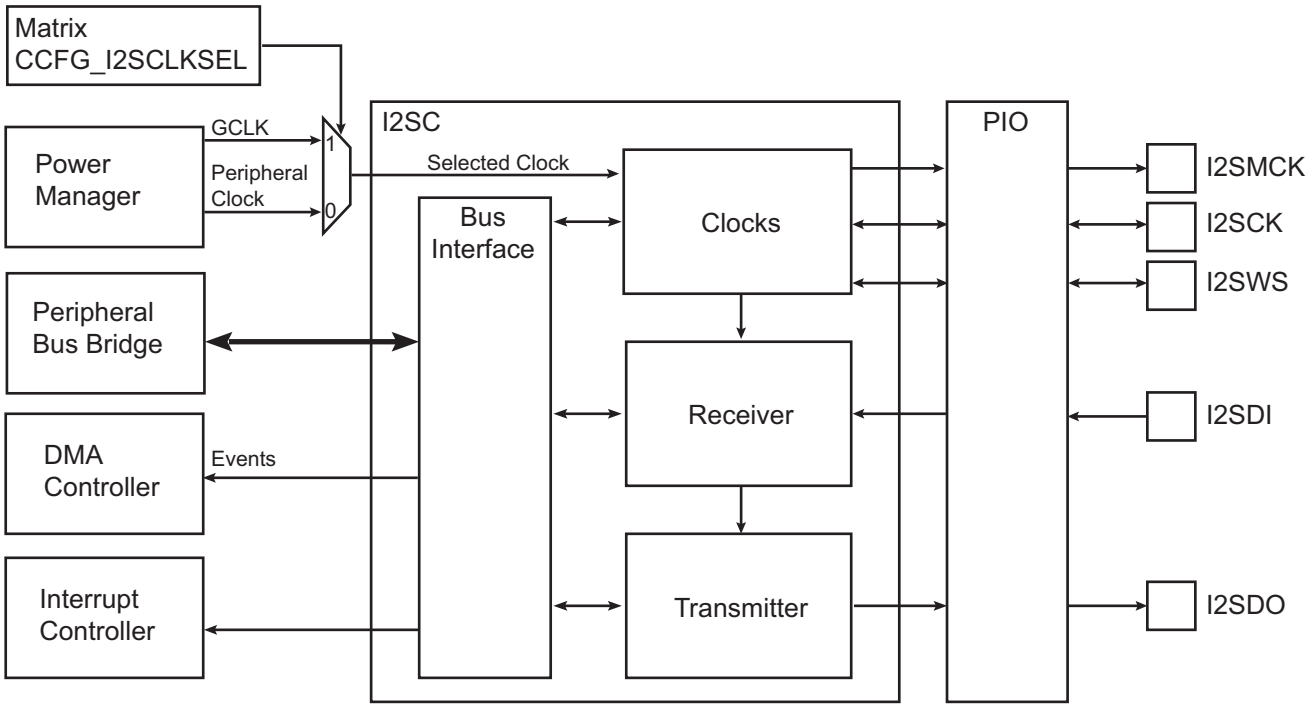
In Master mode, the I2SC can produce a 32 f_s to 1024 f_s master clock that provides an over-sampling clock to an external audio codec or digital signal processor (DSP).

41.2 Embedded Characteristics

- Compliant with Inter-IC Sound (I^2S) Bus Specification
- Master, Slave, and Controller Modes
  - Slave: Data Received/Transmitted
  - Master: Data Received/Transmitted And Clocks Generated
  - Controller: Clocks Generated
- Individual Enable and Disable of Receiver, Transmitter and Clocks
- Configurable Clock Generator Common to Receiver and Transmitter
  - Suitable for a Wide Range of Sample Frequencies (f_s), Including 32 kHz, 44.1 kHz, 48 kHz, 88.2 kHz, 96 kHz, and 192 kHz
  - 32 f_s to 1024 f_s Master Clock Generated for External Oversampling Data Converters
- Support for Multiple Data Formats
  - 32-, 24-, 20-, 18-, 16-, and 8-bit Mono or Stereo Format
  - 16- and 8-bit Compact Stereo Format, with Left and Right Samples Packed in the Same Word to Reduce Data Transfers
- DMA Controller Interfaces the Receiver and Transmitter to Reduce Processor Overhead
  - One DMA Controller Channel for Both Audio Channels
- Smart Holding Registers Management to Avoid Audio Channels Mix After Overrun or Underrun

41.3 Block Diagram

Figure 41-1. I2SC Block Diagram



41.4 I/O Lines Description

Table 41-1. I/O Lines Description

Pin Name	Pin Description	Type
I2SMCK	Master Clock	Output
I2SCK	Serial Clock	Input/Output
I2SWS	I^2S Word Select	Input/Output
I2SDI	Serial Data Input	Input
I2SDO	Serial Data Output	Output

41.5 Product Dependencies

To use the I2SC, other parts of the system must be configured correctly, as described below.

41.5.1 I/O Lines

The I2SC pins may be multiplexed with I/O Controller lines. The user must first program the PIO Controller to assign the required I2SC pins to their peripheral function. If the I2SC I/O lines are not used by the application, they can be used for other purposes by the PIO Controller. The user must enable the I2SC inputs and outputs that are used.

Table 41-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
I2SC0	I2SC0_CK	PC1	E
I2SC0	I2SC0_CK	PD19	E
I2SC0	I2SC0_DI0	PC4	E
I2SC0	I2SC0_DI0	PD22	E
I2SC0	I2SC0_DO0	PC5	E
I2SC0	I2SC0_DO0	PD23	E
I2SC0	I2SC0_MCK	PC2	E
I2SC0	I2SC0_MCK	PD20	E
I2SC0	I2SC0_WS	PC3	E
I2SC0	I2SC0_WS	PD21	E
I2SC1	I2SC1_CK	PA15	D
I2SC1	I2SC1_CK	PB15	D
I2SC1	I2SC1_DI0	PA17	D
I2SC1	I2SC1_DI0	PB17	D
I2SC1	I2SC1_DO0	PA18	D
I2SC1	I2SC1_DO0	PB18	D
I2SC1	I2SC1_MCK	PA14	D
I2SC1	I2SC1_MCK	PB14	D
I2SC1	I2SC1_WS	PA16	D
I2SC1	I2SC1_WS	PB16	D

41.5.2 Power Management

If the CPU enters a Sleep mode that disables clocks used by the I2SC, the I2SC stops functioning and resumes operation after the system wakes up from Sleep mode.

41.5.3 Clocks

The clock for the I2SC bus interface is generated by the Power Management Controller (PMC). I2SC must be disabled before disabling the clock to avoid freezing the I2SC in an undefined state.

41.5.4 DMA Controller

The I2SC interfaces to the DMA Controller. Using the I2SC DMA functionality requires the DMA Controller to be programmed first.

41.5.5 Interrupt Sources

The I2SC interrupt line is connected to the Interrupt Controller. Using the I2SC interrupt requiresthe Interrupt Controller to be programmed first.

Table 41-3. Peripheral IDs

Instance	ID
I2SC0	54
I2SC1	55

41.6 Functional Description

41.6.1 Initialization

The I2SC features a receiver, a transmitter and a clock generator for Master and Controller modes. Receiver and transmitter share the same serial clock and word select.

Before enabling the I2SC, the selected configuration must be written to the I2SC Mode Register (I2SC_MR). If the I2SC_MR.IMCKMODE bit is set, the I2SC_MR.IMCKFS field must be configured as described in Section 41.6.5 “Serial Clock and Word Select Generation”.

Once the I2SC_MR has been written, the I2SC clock generator, receiver, and transmitter can be enabled by writing a ’1’ to the CKEN, RXEN, and TXEN bits in the Control Register (I2SC_CR). The clock generator can be enabled alone in Controller mode to output clocks to the I2SMCK, I2SCK, and I2SWS pins. The clock generator must also be enabled if the receiver or the transmitter is enabled.

The clock generator, receiver, and transmitter can be disabled independently by writing a ’1’ to I2SC_CR.CXDIS, I2SC_CR.RXDIS and/or I2SC_CR.TXDIS, respectively. Once requested to stop, they stop only when the transmission of the pending frame transmission is completed.

41.6.2 Basic Operation

The receiver can be operated by reading the Receiver Holding Register (I2SC_RHR), whenever the Receive Ready (RXRDY) bit in the Status Register (I2SC_SR) is set. Successive values read from RHR correspond to the samples from the left and right audio channels for the successive frames.

The transmitter can be operated by writing to the Transmitter Holding Register (I2SC_THR), whenever the Transmit Ready (TXRDY) bit in the I2SC_SR is set. Successive values written to THR correspond to the samples from the left and right audio channels for the successive frames.

The RXRDY and TXRDY bits can be polled by reading the I2SC_SR.

The I2SC processor load can be reduced by enabling interrupt-driven operation. The RXRDY and/or TXRDY interrupt requests can be enabled by writing a ’1’ to the corresponding bit in the Interrupt Enable Register (I2SC_IER). The interrupt service routine associated to the I2SC interrupt request is executed whenever the Receive Ready or the Transmit Ready status bit is set.

41.6.3 Master, Controller and Slave Modes

In Master and Controller modes, the I2SC provides the master clock, the serial clock and the word select. I2SMCK, I2SCK, and I2SWS pins are outputs.

In Controller mode, the I2SC receiver and transmitter are disabled. Only the clocks are enabled and used by an external receiver and/or transmitter.

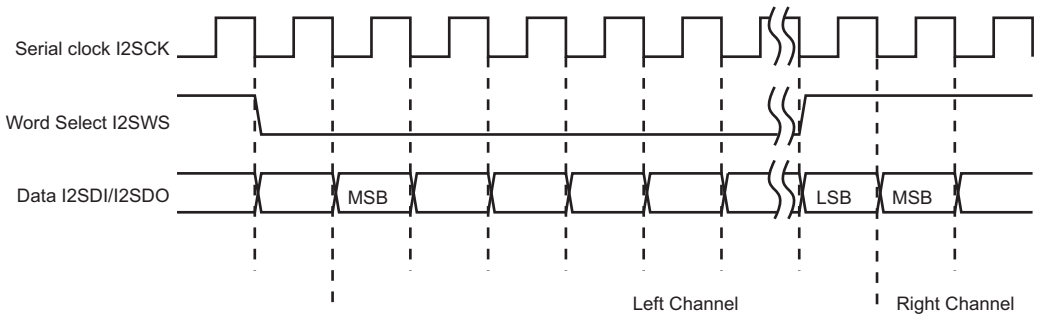
In Slave mode, the I2SC receives the serial clock and the word select from an external master. I2SCK and I2SWS pins are inputs.

The mode is selected by writing the MODE field in the I2SC_MR. Since the MODE field changes the direction of the I2SWS and I2SSCK pins, the I2SC_MR must be written when the I2SC is stopped.

41.6.4 I²S Reception and Transmission Sequence

As specified in the I²S protocol, data bits are left-justified in the word select time slot, with the MSB transmitted first, starting one clock period after the transition on the word select line.

Figure 41-2. I²S Reception and Transmission Sequence



Data bits are sent on the falling edge of the serial clock and sampled on the rising edge of the serial clock. the word select line indicates the channel in transmission, a low level for the left channel and a high level for the right channel.

The length of transmitted words can be chosen among 8, 16, 18, 20, 24, and 32 bits by writing the I2SC_MR.DATALENGTH field.

If the time slot allows for more data bits than written in the I2SC_MR.DATALENGTH field, zeroes are appended to the transmitted data word or extra received bits are discarded.

41.6.5 Serial Clock and Word Select Generation

The generation of clocks in the I2SC is described in Figure 41-3.

In Slave mode, the serial clock and word select clock are driven by an external master. I2SCK and I2SWS pins are inputs.

In Master mode, the user can configure the master clock, serial clock, and word select clock through the I2SC_MR. I2SMCK, I2SCK, and I2SWS pins are outputs and MCK is used to derive the I2SC clocks.

In Master mode, if the Peripheral clock frequency is higher than 96 MHz, the GCLK clock from PMC must be selected as I2SC input clock by writing a ’1’ in the CLKSELx bit of the CCFG_I2CLKSEL register located in Matrix (See Figure 41-3).

Audio codecs connected to the I2SC pins may require a master clock (I2SMCK) signal with a frequency multiple of the audio sample frequency (f_s), such as $256f_s$. When the I2SC is in Master mode, writing a ’1’ to I2SC_MR.IMCKMODE outputs MCK as master clock to the I2SMCK pin, and divides MCK to create the internal bit clock, output on the I2SCK pin. The clock division factor is defined by writing to I2SC_MR.IMCKFS and I2SC_MR.DATALENGTH, as described in the I2SC_MR.IMCKFS field description.

The master clock (I2SMCK) frequency is [2×16 × (IMCKFS + 1)] / (IMCKDIV + 1) times the sample frequency (f_s), i.e., I2SWS frequency.

Example: If the sampling rate is 44.1 kHz with an I2S master clock (I2SMCK) ratio of 256, the core frequency must be an integer multiple of 11.2896 MHz. Assuming an integer multiple of 4, the IMCKDIV field must be configured to 4; the field IMCKFS must then be set to 31.

The serial clock (I2SCK) frequency is 2 × Slot Length times the sample frequency (f_s), where Slot Length is defined in Table 41-4.

Table 41-4. Slot Length

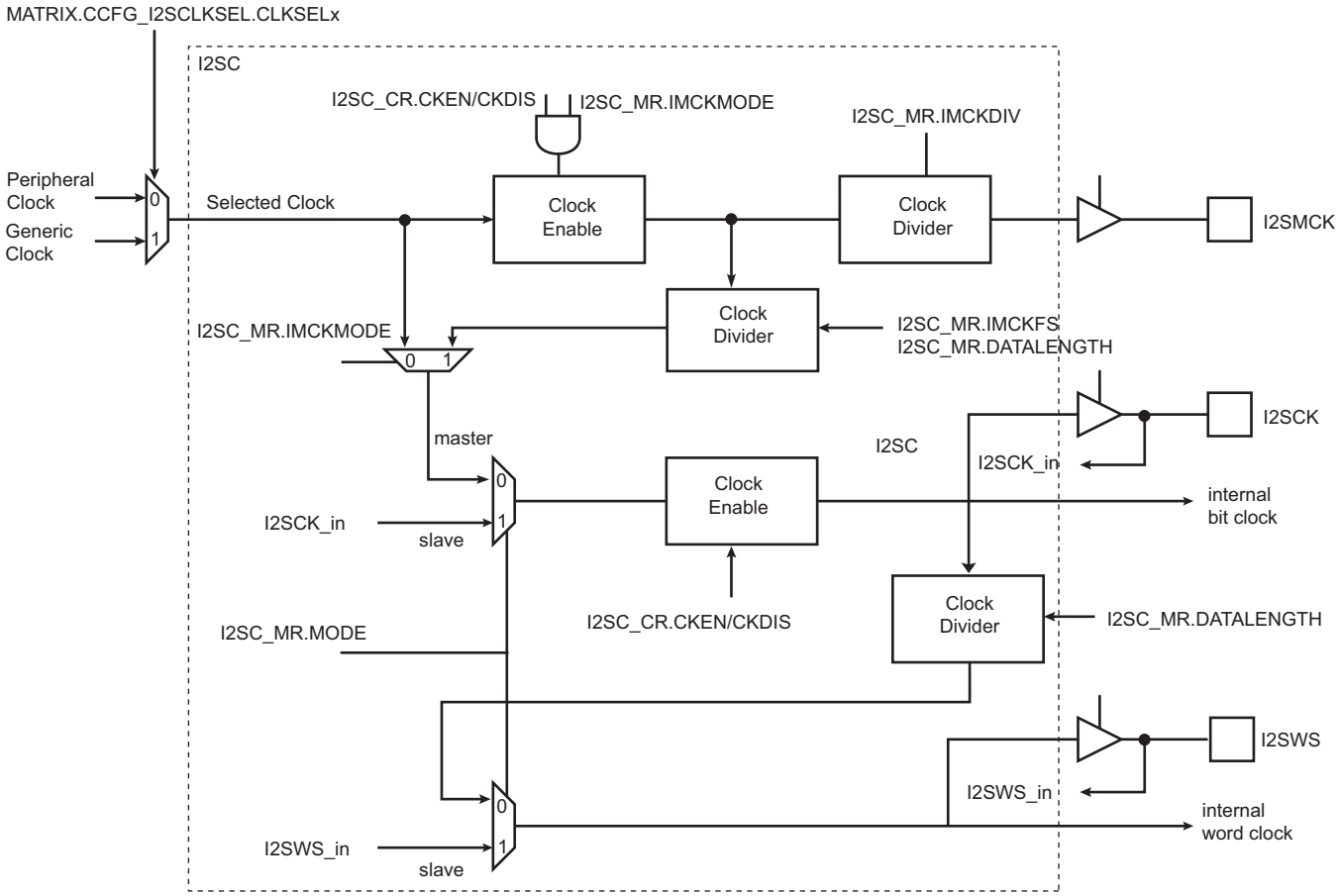
I2SC_MR.DATALENGTH	Word Length	Slot Length
0	32 bits	32
1	24 bits	32 if I2SC_MR.IWS = 0 24 if I2SC_MR.IWS = 1
2	20 bits	
3	18 bits	
4	16 bits	16
5	16 bits compact stereo	
6	8 bits	8
7	8 bits compact stereo	

Warning: I2SC_MR.IMCKMODE must be written to ’1’ if the master clock frequency is strictly higher than the serial clock.

If a master clock output is not required, the MCK clock is used as I2SCK by clearing I2SC_MR.IMCKMODE. Alternatively, if the frequency of the MCK clock used is a multiple of the required I2SCK frequency, the I2SMCK to I2SCK divider can be used with the ratio defined by writing the I2SC_MR.IMCKFS field.

The I2SWS pin is used as word select as described in Section 41.6.4 “I2S Reception and Transmission Sequence”.

Figure 41-3. I2SC Clock Generation



41.6.6 Mono

When the Transmit Mono bit (TXMONO) in I2SC_MR is set, data written to the left channel is duplicated to the right output channel.

When the Receive Mono bit (RXMONO) in I2SC_MR is set, data received from the left channel is duplicated to the right channel.

41.6.7 Holding Registers

The I2SC user interface includes a Receive Holding Register (I2SC_RHR) and a Transmit Holding Register (I2SC_THR). These registers are used to access audio samples for both audio channels.

When a new data word is available in I2SC_RHR, the Receive Ready bit (RXRDY) in I2SC_SR is set. Reading I2SC_RHR clears this bit.

A receive overrun condition occurs if a new data word becomes available before the previous data word has been read from I2SC_RHR. In this case, the Receive Overrun bit in I2SC_SR and bit i of the RXORCH field in I2SC_SR are set, where i is the current receive channel number.

When I2SC_THR is empty, the Transmit Ready bit (TXRDY) in I2SC_SR is set. Writing to I2SC_THR clears this bit.

A transmit underrun condition occurs if a new data word needs to be transmitted before it has been written to I2SC_THR. In this case, the Transmit Underrun (TXUR) bit and bit i of the TXORCH field in I2SC_SR are set, where i is the current transmit channel number. If the TXSAME bit in I2SC_MR is ’0’, then a zero data word is

transmitted in case of underrun. If I2SC_MR.TXSAME is ’1’, then the previous data word for the current transmit channel number is transmitted.

Data words are right-justified in I2SC_RHR and I2SC_THR. For the 16-bit compact stereo data format, the left sample uses bits 15:0 and the right sample uses bits 31:16 of the same data word. For the 8-bit compact stereo data format, the left sample uses bits 7:0 and the right sample uses bits 15:8 of the same data word.

41.6.8 DMA Controller Operation

All receiver audio channels are assigned to a single DMA Controller.

The DMA Controller reads from the I2SC_RHR and writes to the I2SC_THR for both audio channels successively.

The DMA Controller transfers may use 32-bit word, 16-bit halfword, or 8-bit byte depending on the value of the I2SC_MR.DATALENGTH field.

41.6.9 Loop-back Mode

For debug purposes, the I2SC can be configured to loop back the transmitter to the Receiver. Writing a ’1’ to the I2SC_MR.LOOP bit internally connects I2SDO to I2SDI, so that the transmitted data is also received. Writing a ’0’ to I2SC_MR.LOOP restores the normal behavior with independent Receiver and Transmitter. As for other changes to the Receiver or Transmitter configuration, the I2SC Receiver and Transmitter must be disabled before writing to I2SC_MR to update I2SC_MR.LOOP.

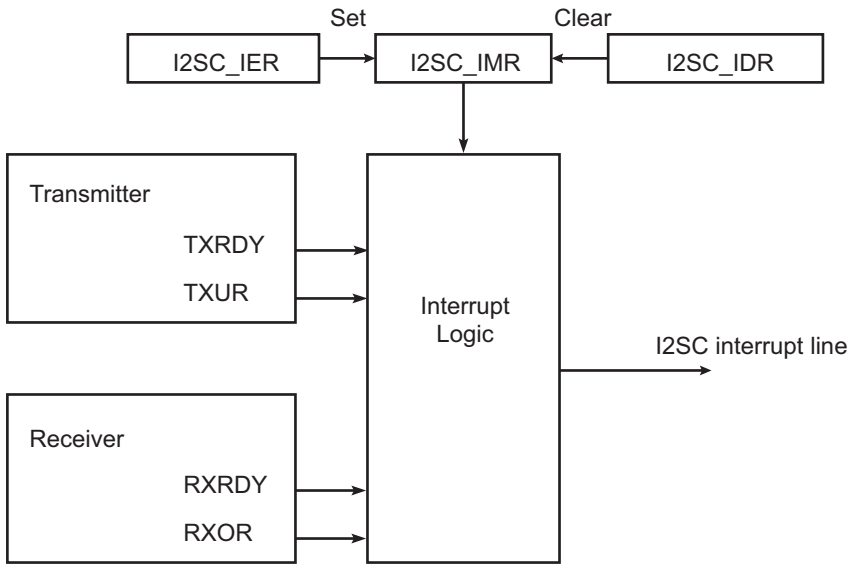
41.6.10 Interrupts

An I2SC interrupt request can be triggered whenever one or several of the following bits are set in I2SC_SR: Receive Ready (RXRDY), Receive Overrun (RXOR), Transmit Ready (TXRDY) or Transmit Underrun (TXUR).

The interrupt request is generated if the corresponding bit in the Interrupt Mask Register (I2SC_IMR) is set. Bits in I2SC_IMR are set by writing a ’1’ to the corresponding bit in I2SC_IER and cleared by writing a ’1’ to the corresponding bit in the Interrupt Disable Register (I2SC_IDR). The interrupt request remains active until the corresponding bit in I2SC_SR is cleared by writing a ’1’ to the corresponding bit in the Status Clear Register (I2SC_SCR).

For debug purposes, interrupt requests can be simulated by writing a ’1’ to the corresponding bit in the Status Set Register (I2SC_SSR).

Figure 41-4. Interrupt Block Diagram



41.7 I2SC Application Examples

The I2SC supports several serial communication modes used in audio or high-speed serial links. Examples of standard applications are shown in the following figures. All serial link applications supported by the I2SC are not listed here.

Figure 41-5. Slave Transmitter I2SC Application Example

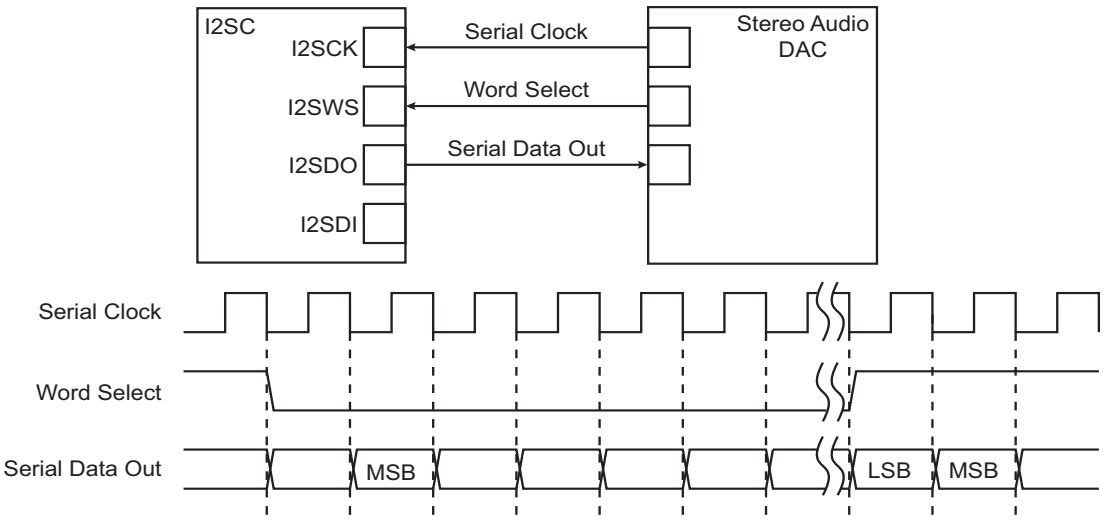


Figure 41-6. Dual Microphone Application Block Diagram

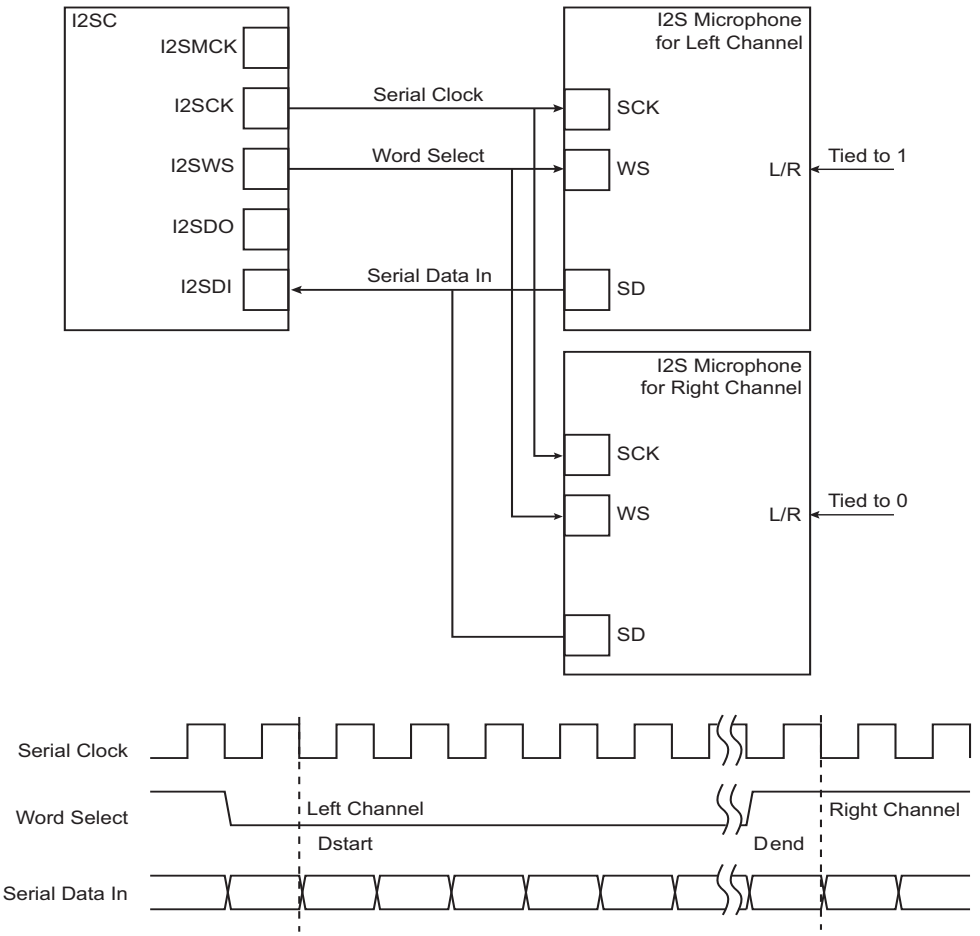
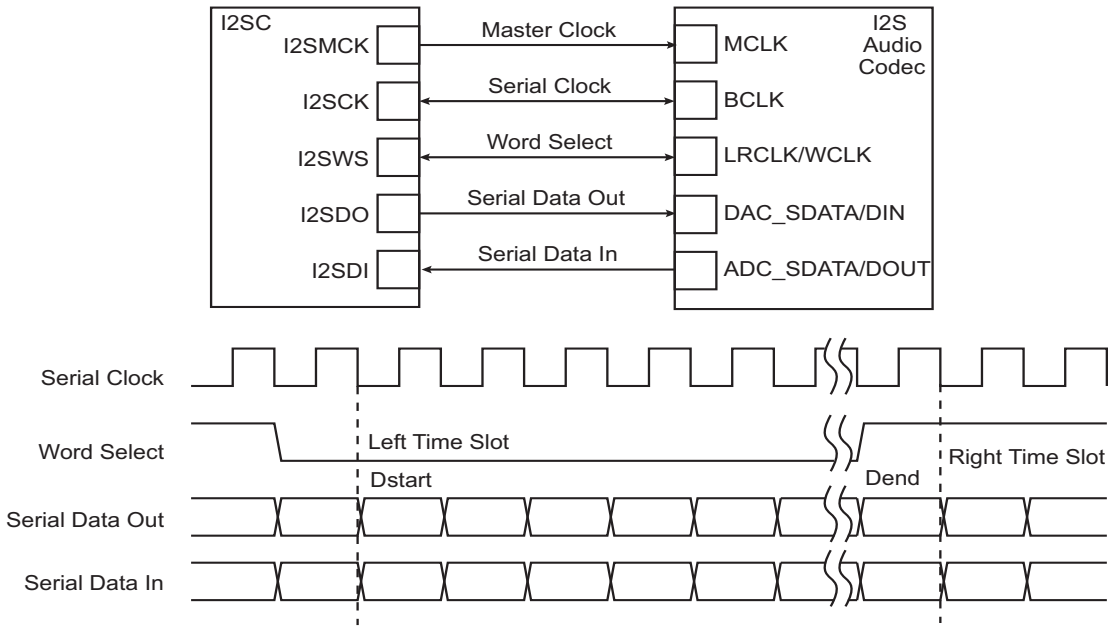


Figure 41-7. Codec Application Block Diagram



41.8 Inter-IC Sound Controller (I2SC) User Interface

Table 41-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	I2SC_CR	Write-only	–
0x04	Mode Register	I2SC_MR	Read/Write	0x00000000
0x08	Status Register	I2SC_SR	Read-only	0x00000000
0x0C	Status Clear Register	I2SC_SCR	Write-only	–
0x10	Status Set Register	I2SC_SSR	Write-only	–
0x14	Interrupt Enable Register	I2SC_IER	Write-only	–
0x18	Interrupt Disable Register	I2SC_IDR	Write-only	–
0x1C	Interrupt Mask Register	I2SC_IMR	Read-only	0x00000000
0x20	Receiver Holding Register	I2SC_RHR	Read-only	0x00000000
0x24	Transmitter Holding Register	I2SC_THR	Write-only	–
0x28–0xFC	Reserved	–	–	–

41.8.1 Inter-IC Sound Controller Control Register

Name: I2SC_CR

Address: 0xF8050000 (0), 0xFC04C000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
SWRST	–	TXDIS	TXEN	CKDIS	CKEN	RXDIS	RXEN

- **RXEN: Receiver Enable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit enables the I2SC receiver, if RXDIS is not one. Bit I2SC_SR.RXEN is set when the receiver is activated.

- **RXDIS: Receiver Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit disables the I2SC receiver. Bit I2SC_SR.RXEN is cleared when the receiver is stopped.

- **CKEN: Clocks Enable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit enables the I2SC clocks generation, if CKDIS is not one.

- **CKDIS: Clocks Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a zone to this bit disables the I2SC clock generation.

- **TXEN: Transmitter Enable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit enables the I2SC transmitter, if TXDIS is not one. Bit I2SC_SR.TXEN is set when the Transmitter is started.

- **TXDIS: Transmitter Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit disables the I2SC transmitter. Bit I2SC_SR.TXEN is cleared when the Transmitter is stopped.

- **SWRST: Software Reset**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit resets all the registers in the I2SC. The I2SC is disabled after the reset.

41.8.2 Inter-IC Sound Controller Mode Register

Name: I2SC_MR

Address: 0xF8050004 (0), 0xFC04C004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
IWS	IMCKMODE	IMCKFS					
23	22	21	20	19	18	17	16
–	–	IMCKDIV					
15	14	13	12	11	10	9	8
–	TXSAME	–	TXMONO	–	RXLOOP	–	RXMONO
7	6	5	4	3	2	1	0
FORMAT		–	DATALENGTH			–	MODE

The I2SC_MR must be written when the I2SC is stopped. The proper sequence is to write to I2SC_MR, then write to I2SC_CR to enable the I2SC or to disable the I2SC before writing a new value to I2SC_MR.

- **MODE: Inter-IC Sound Controller Mode**

Value	Name	Description
0	SLAVE	I2SCK and i2SWS pin inputs used as bit clock and word select/frame synchronization.
1	MASTER	Bit clock and word select/frame synchronization generated by I2SC from MCK and output to I2SCK and I2SWS pins. MCK is output as master clock on I2SMCK if I2SC_MR.IMCKMODE is set.

- **DATALENGTH: Data Word Length**

Value	Name	Description
0	32_BITS	Data length is set to 32 bits
1	24_BITS	Data length is set to 24 bits
2	20_BITS	Data length is set to 20 bits
3	18_BITS	Data length is set to 18 bits
4	16_BITS	Data length is set to 16 bits
5	16_BITS_COMPACT	Data length is set to 16-bit compact stereo. Left sample in bits 15:0 and right sample in bits 31:16 of same word.
6	8_BITS	Data length is set to 8 bits
7	8_BITS_COMPACT	Data length is set to 8-bit compact stereo. Left sample in bits 7:0 and right sample in bits 15:8 of the same word.

- **FORMAT: Data Format**

Value	Name	Description
0	I2S	I^2S format, stereo with I2SWS low for left channel, and MSB of sample starting one I2SCK period after I2SWS edge
1	LJ	Left-justified format, stereo with I2SWS high for left channel, and MSB of sample starting on I2SWS edge
2	–	Reserved
3	–	Reserved

- **RXMONO: Receive Mono**

0: Stereo

1: Mono, with left audio samples duplicated to right audio channel by the I2SC.

- **RXLOOP: Loop-back Test Mode**

0: Normal mode

1: I2SDO output of I2SC is internally connected to I2SDI input.

- **TXMONO: Transmit Mono**

0: Stereo

1: Mono, with left audio samples duplicated to right audio channel by the I2SC.

- **TXSAME: Transmit Data when Underrun**

0: Zero sample transmitted when underrun.

1: Previous sample transmitted when underrun

- **IMCKDIV: Clock to I2SC Master Clock Ratio**

I2SMCK Master clock output frequency is Clock divided by (IMCKDIV + 1). Refer to the IMCKFS field description.

Notes: 1. This field is write-only. Always read as '0'.
2. Do not write a '0' to this field.

- **IMCKFS: Master Clock to f_s Ratio**

Master clock frequency is [2 x 16 × (IMCKFS + 1)] / (IMCKDIV + 1) times the sample rate, i.e., I2SWS frequency.

Value	Name	Description
0	M2SF32	Sample frequency ratio set to 32
1	M2SF64	Sample frequency ratio set to 64
2	M2SF96	Sample frequency ratio set to 96
3	M2SF128	Sample frequency ratio set to 128
5	M2SF192	Sample frequency ratio set to 192
7	M2SF256	Sample frequency ratio set to 256
11	M2SF384	Sample frequency ratio set to 384
15	M2SF512	Sample frequency ratio set to 512
23	M2SF768	Sample frequency ratio set to 768
31	M2SF1024	Sample frequency ratio set to 1024
47	M2SF1536	Sample frequency ratio set to 1536
63	M2SF2048	Sample frequency ratio set to 2048

- **IMCKMODE: Master Clock Mode**

0: No master clock generated (Clock drives I2SCK output).

1: Master clock generated (internally generated clock is used as I2SMCK output).

Warning: If I2SMCK frequency is the same as I2SCK, IMCKMODE must be cleared. Refer to Section 41.6.5 "Serial Clock and Word Select Generation" and Table 41-4 "Slot Length".

- **IWS: I2SWS Slot Width**

0: I2SWS slot is 32 bits wide for DATALENGTH = 18/20/24 bits.

1: I2SWS slot is 24 bits wide for DATALENGTH = 18/20/24 bits.

Refer to Table 41-4 “Slot Length”.

41.8.3 Inter-IC Sound Controller Status Register

Name: I2SC_SR

Address: 0xF8050008 (0), 0xFC04C008 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	TXURCH		–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXORCH	
7	6	5	4	3	2	1	0
–	TXUR	TXRDY	TXEN	–	RXOR	RXRDY	RXEN

- **RXEN: Receiver Enabled**

0: This bit is cleared when the receiver is disabled, following a RXDIS or SWRST request in I2SC_CR.

1: This bit is set when the receiver is enabled, following a RXEN request in I2SC_CR.

- **RXRDY: Receive Ready**

0: This bit is cleared when I2SC_RHR is read.

1: This bit is set when received data is present in I2SC_RHR.

- **RXOR: Receive Overrun**

0: This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’.

1: This bit is set when an overrun error occurs on I2SC_RHR or when the corresponding bit in I2SC_SSR is written to ’1’.

- **TXEN: Transmitter Enabled**

0: This bit is cleared when the transmitter is disabled, following a I2SC_CR.TXDIS or I2SC_CR.SWRST request.

1: This bit is set when the transmitter is enabled, following a I2SC_CR.TXEN request.

- **TXRDY: Transmit Ready**

0: This bit is cleared when data is written to I2SC_THR.

1: This bit is set when I2SC_THR is empty and can be written with new data to be transmitted.

- **TXUR: Transmit Underrun**

0: This bit is cleared when the corresponding bit in I2SC_SCR is written to ’1’.

1: This bit is set when an underrun error occurs on I2SC_THR or when the corresponding bit in I2SC_SSR is written to ’1’.

- **RXORCH: Receive Overrun Channel**

This field is cleared when I2SC_SCR.RXOR is written to ’1’.

Bit i of this field is set when a receive overrun error occurred in channel i (i = 0 for first channel of the frame).

- **TXURCH: Transmit Underrun Channel**

0: This field is cleared when I2SC_SCR.TXUR is written to ’1’.

1: Bit i of this field is set when a transmit underrun error occurred in channel i (i = 0 for first channel of the frame).

41.8.4 Inter-IC Sound Controller Status Clear Register

Name: I2SC_SCR

Address: 0xF805000C (0), 0xFC04C00C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	TXURCH		–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXORCH	
7	6	5	4	3	2	1	0
–	TXUR	–	–	–	RXOR	–	–

- **RXOR: Receive Overrun Status Clear**

Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit clears the status bit.

- **TXUR: Transmit Underrun Status Clear**

Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit clears the status bit.

- **RXORCH: Receive Overrun Per Channel Status Clear**

Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

- **TXURCH: Transmit Underrun Per Channel Status Clear**

Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field clears the corresponding bit in the I2SC_SR and the corresponding interrupt request.

41.8.5 Inter-IC Sound Controller Status Set Register

Name: I2SC_SSR

Address: 0xF8050010 (0), 0xFC04C010 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	TXURCH		–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXORCH	
7	6	5	4	3	2	1	0
–	TXUR	–	–	–	RXOR	–	–

- **RXOR: Receive Overrun Status Set**

Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit sets the status bit.

- **TXUR: Transmit Underrun Status Set**

Writing a ’0’ to this bit has no effect.

Writing a ’1’ to this bit sets the status bit.

- **RXORCH: Receive Overrun Per Channel Status Set**

Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field sets the corresponding bit in I2SC_SR and the corresponding interrupt request.

- **TXURCH: Transmit Underrun Per Channel Status Set**

Writing a ’0’ has no effect.

Writing a ’1’ to any bit in this field sets the corresponding bit in I2SC_SR and the corresponding interrupt request.

41.8.6 Inter-IC Sound Controller Interrupt Enable Register

Name: I2SC_IER

Address: 0xF8050014 (0), 0xFC04C014 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	TXUR	TXRDY	–	–	RXOR	RXRDY	–

- **RXRDY: Receiver Ready Interrupt Enable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR.

- **RXOR: Receiver Overrun Interrupt Enable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR.

- **TXRDY: Transmit Ready Interrupt Enable**

0: Writing a ’0’ to this bit as no effect.

1: Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR.

- **TXUR: Transmit Underflow Interrupt Enable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit sets the corresponding bit in I2SC_IMR.

41.8.7 Inter-IC Sound Controller Interrupt Disable Register

Name: I2SC_IDR

Address: 0xF8050018 (0), 0xFC04C018 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	TXUR	TXRDY	–	–	RXOR	RXRDY	–

- **RXRDY: Receiver Ready Interrupt Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR.

- **RXOR: Receiver Overrun Interrupt Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR.

- **TXRDY: Transmit Ready Interrupt Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR.

- **TXUR: Transmit Underflow Interrupt Disable**

0: Writing a ’0’ to this bit has no effect.

1: Writing a ’1’ to this bit clears the corresponding bit in I2SC_IMR.

41.8.8 Inter-IC Sound Controller Interrupt Mask Register

Name: I2SC_IMR

Address: 0xF805001C (0), 0xFC04C01C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	TXUR	TXRDY	–	–	RXOR	RXRDY	–

- **RXRDY: Receiver Ready Interrupt Disable**

0: The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’.

1: The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’.

- **RXOR: Receiver Overrun Interrupt Disable**

0: The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’.

1: The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’.

- **TXRDY: Transmit Ready Interrupt Disable**

0: The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’.

1: The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’.

- **TXUR: Transmit Underflow Interrupt Disable**

0: The corresponding interrupt is disabled. This bit is cleared when the corresponding bit in I2SC_IDR is written to ’1’.

1: The corresponding interrupt is enabled. This bit is set when the corresponding bit in I2SC_IER is written to ’1’.

41.8.9 Inter-IC Sound Controller Receiver Holding Register

Name: I2SC_RHR

Address: 0xF8050020 (0), 0xFC04C020 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RHR							
23	22	21	20	19	18	17	16
RHR							
15	14	13	12	11	10	9	8
RHR							
7	6	5	4	3	2	1	0
RHR							

- **RHR: Receiver Holding Register**

This field is set by hardware to the last received data word. If I2SC_MR.DATALENGTH specifies fewer than 32 bits, data is right-justified in the RHR field.

41.8.10 Inter-IC Sound Controller Transmitter Holding Register

Name: I2SC_THR

Address: 0xF8050024 (0), 0xFC04C024 (1)

Access: Write-only

31	30	29	28	27	26	25	24
THR							
23	22	21	20	19	18	17	16
THR							
15	14	13	12	11	10	9	8
THR							
7	6	5	4	3	2	1	0
THR							

- **THR: Transmitter Holding Register**

Next data word to be transmitted after the current word if TXRDY is not set. If I2SC_MR.DATALENGTH specifies fewer than 32 bits, data is right-justified in the THR field.

42. Synchronous Serial Controller (SSC)

42.1 Description

The Synchronous Serial Controller (SSC) provides a synchronous communication link with external devices. It supports many serial synchronous communication protocols generally used in audio and telecom applications such as I2S, Short Frame Sync, Long Frame Sync, etc.

The SSC contains an independent receiver and transmitter and a common clock divider. The receiver and the transmitter each interface with three signals: the TD/RD signal for data, the TK/RK signal for the clock and the TF/RF signal for the Frame Sync. The transfers can be programmed to start automatically or on different events detected on the Frame Sync signal.

The SSC high-level of programmability and its use of DMA enable a continuous high bit rate data transfer without processor intervention.

Featuring connection to the DMA, the SSC enables interfacing with low processor overhead to:

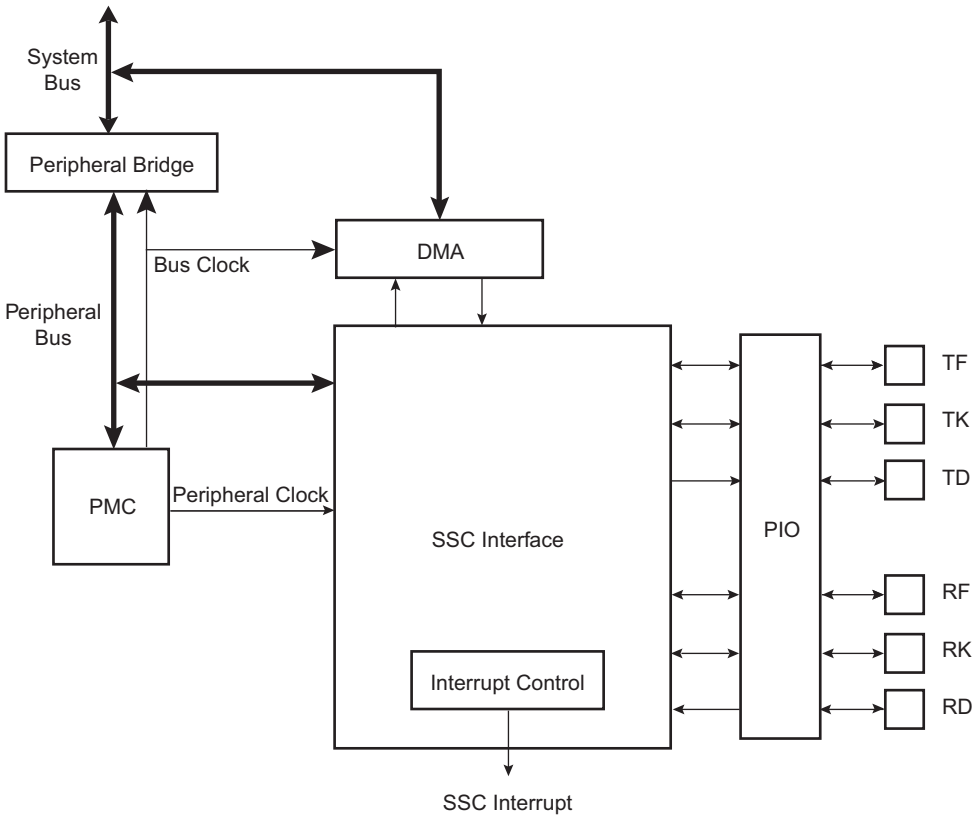
- Codecs in Master or Slave mode,
- DAC through dedicated serial interface, particularly I2S,
- Magnetic card reader.

42.2 Embedded Characteristics

- Provides Serial Synchronous Communication Links Used in Audio and Telecom Applications
- Contains an Independent Receiver and Transmitter and a Common Clock Divider
- Interfaced with the DMA Controller (DMAC) to Reduce Processor Overhead
- Offers a Configurable Frame Sync and Data Length
- Receiver and Transmitter Can be Programmed to Start Automatically or on Detection of Different Events on the Frame Sync Signal
- Receiver and Transmitter Include a Data Signal, a Clock Signal and a Frame Synchronization Signal

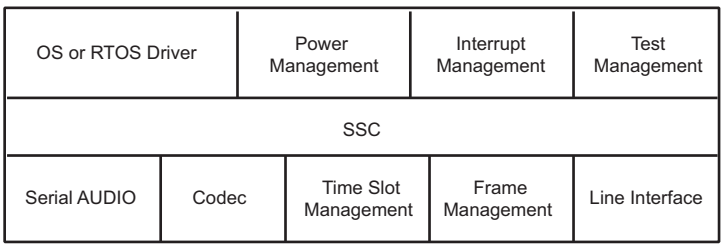
42.3 Block Diagram

Figure 42-1. Block Diagram



42.4 Application Block Diagram

Figure 42-2. Application Block Diagram



42.5 SSC Application Examples

The SSC can support several serial communication modes used in audio or high speed serial links. Some standard applications are shown in the following figures. All serial link applications supported by the SSC are not listed here.

Figure 42-3. Audio Application Block Diagram

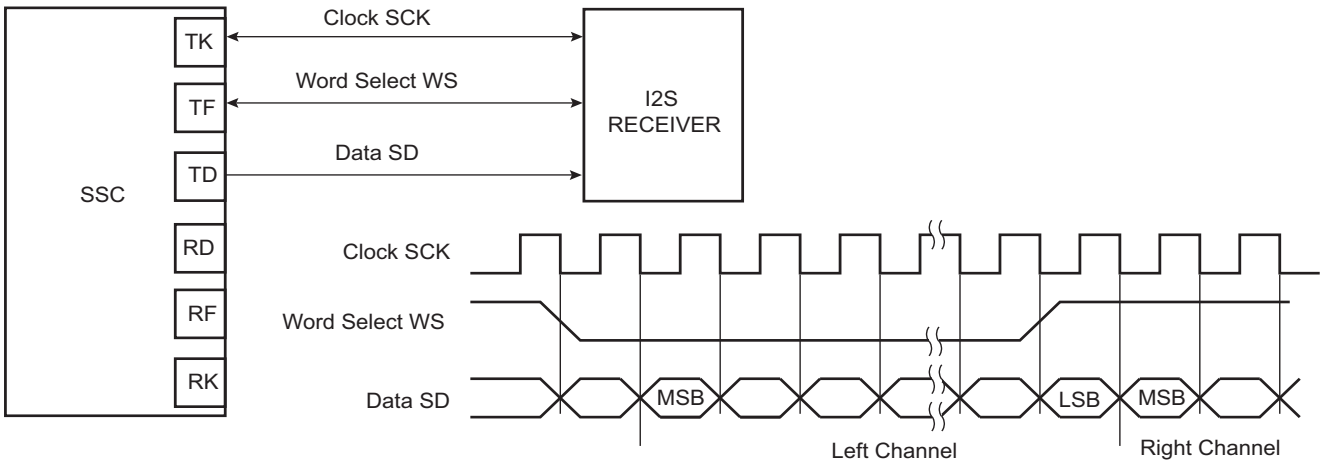


Figure 42-4. Codec Application Block Diagram

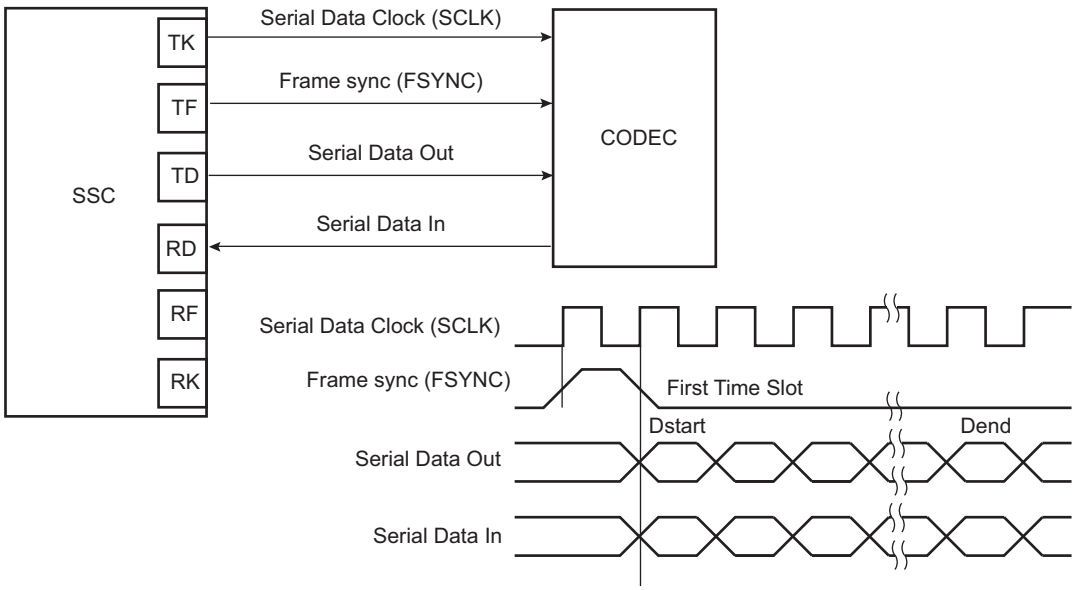
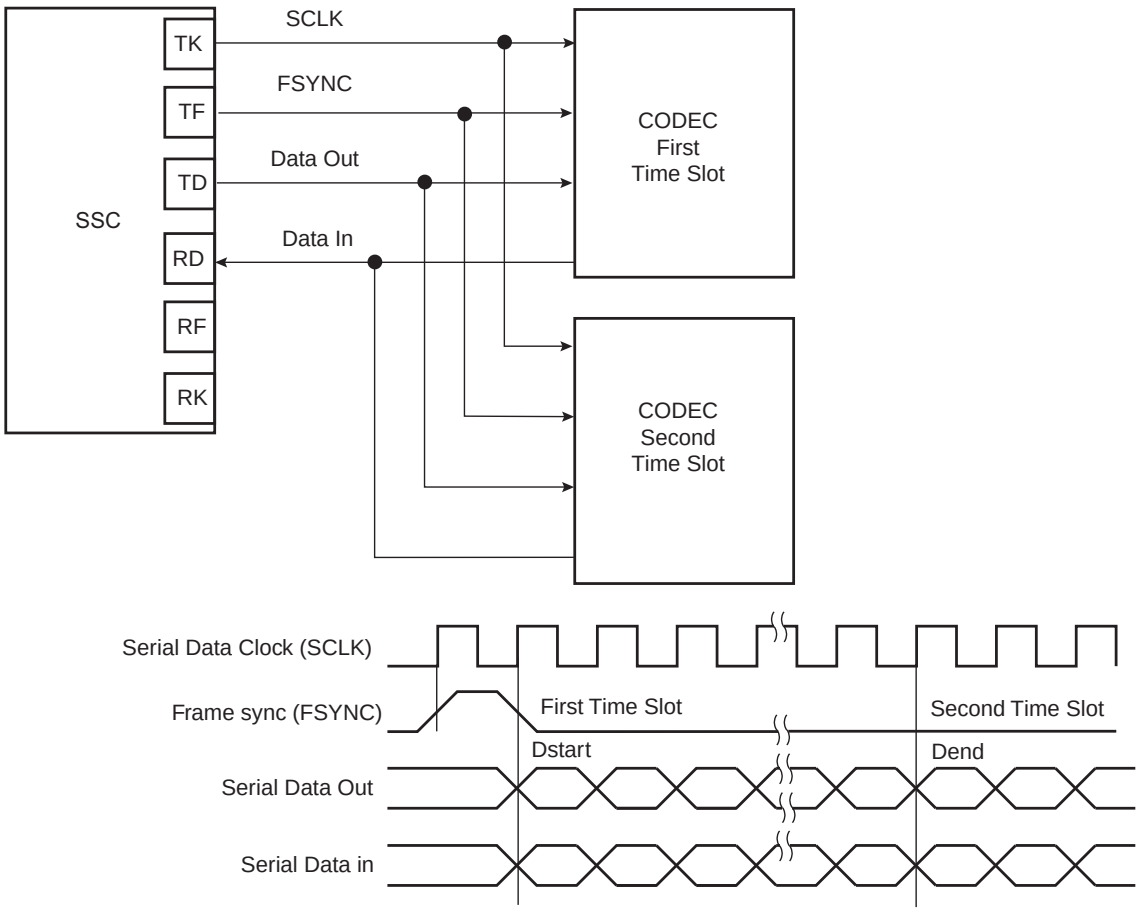


Figure 42-5. Time Slot Application Block Diagram



42.6 Pin Name List

Table 42-1. I/O Lines Description

Pin Name	Pin Description	Type
RF	Receiver Frame Synchro	Input/Output
RK	Receiver Clock	Input/Output
RD	Receiver Data	Input
TF	Transmitter Frame Synchro	Input/Output
TK	Transmitter Clock	Input/Output
TD	Transmitter Data	Output

42.7 Product Dependencies

42.7.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines.

Before using the SSC receiver, the PIO controller must be configured to dedicate the SSC receiver I/O lines to the SSC Peripheral mode.

Before using the SSC transmitter, the PIO controller must be configured to dedicate the SSC transmitter I/O lines to the SSC Peripheral mode.

Table 42-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
SSC0	RD0	PB23	C
SSC0	RD0	PC15	E
SSC0	RF0	PB25	C
SSC0	RF0	PC17	E
SSC0	RK0	PB24	C
SSC0	RK0	PC16	E
SSC0	TD0	PB22	C
SSC0	TD0	PC14	E
SSC0	TF0	PB21	C
SSC0	TF0	PC13	E
SSC0	TK0	PB20	C
SSC0	TK0	PC12	E
SSC1	RD1	PA17	B
SSC1	RD1	PB17	C
SSC1	RF1	PA19	B
SSC1	RF1	PB19	C
SSC1	RK1	PA18	B
SSC1	RK1	PB18	C
SSC1	TD1	PA16	B
SSC1	TD1	PB16	C
SSC1	TF1	PA15	B
SSC1	TF1	PB15	C
SSC1	TK1	PA14	B
SSC1	TK1	PB14	C

42.7.2 Power Management

The SSC is not continuously clocked. The SSC interface may be clocked through the Power Management Controller (PMC), therefore the programmer must first configure the PMC to enable the SSC clock.

42.7.3 Interrupt

The SSC interface has an interrupt line connected to the interrupt controller. Handling interrupts requires programming the interrupt controller before configuring the SSC.

All SSC interrupts can be enabled/disabled configuring the SSC Interrupt Mask Register. Each pending and unmasked SSC interrupt asserts the SSC interrupt line. The SSC interrupt service routine can get the interrupt origin by reading the SSC Interrupt Status Register.

Table 42-3. Peripheral IDs

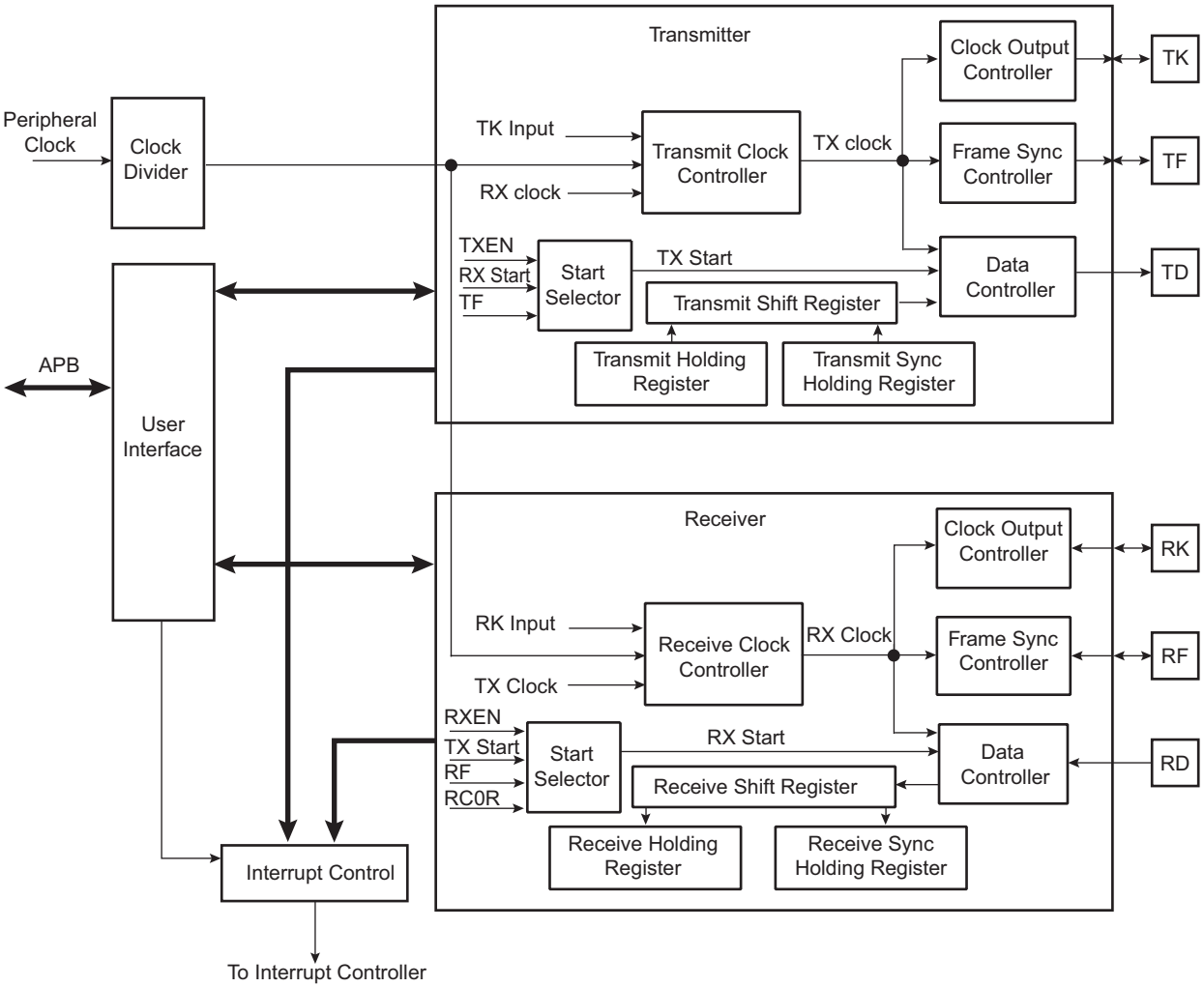
Instance	ID
SSC0	43
SSC1	44

42.8 Functional Description

This section contains the functional description of the following: SSC Functional Block, Clock Management, Data format, Start, Transmitter, Receiver and Frame Sync.

The receiver and transmitter operate separately. However, they can work synchronously by programming the receiver to use the transmit clock and/or to start a data transfer when transmission starts. Alternatively, this can be done by programming the transmitter to use the receive clock and/or to start a data transfer when reception starts. The transmitter and the receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. The maximum clock speed allowed on the TK and RK pins is the peripheral clock divided by 2.

Figure 42-6. SSC Functional Block Diagram



42.8.1 Clock Management

The transmitter clock can be generated by:

- an external clock received on the TK I/O pad
- the receiver clock
- the internal clock divider

The receiver clock can be generated by:

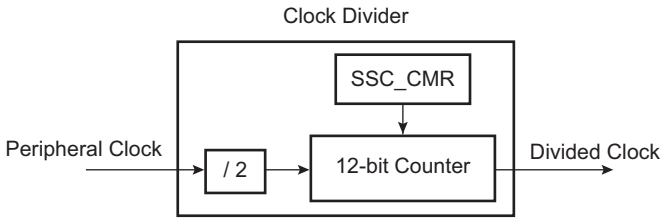
- an external clock received on the RK I/O pad
- the transmitter clock
- the internal clock divider

Furthermore, the transmitter block can generate an external clock on the TK I/O pad, and the receiver block can generate an external clock on the RK I/O pad.

This allows the SSC to support many Master and Slave mode data transfers.

42.8.1.1 Clock Divider

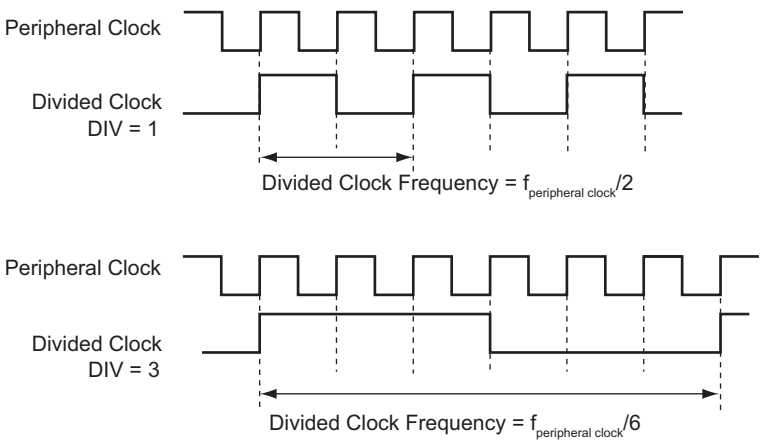
Figure 42-7. Divided Clock Block Diagram



The peripheral clock divider is determined by the 12-bit field DIV counter and comparator (so its maximal value is 4095) in the Clock Mode Register (SSC_CMR), allowing a peripheral clock division by up to 8190. The Divided Clock is provided to both the Receiver and Transmitter. When this field is programmed to 0, the Clock Divider is not used and remains inactive.

When DIV is set to a value equal to or greater than 1, the Divided Clock has a frequency of peripheral clock divided by 2 times DIV. Each level of the Divided Clock has a duration of the peripheral clock multiplied by DIV. This ensures a 50% duty cycle for the Divided Clock regardless of whether the DIV value is even or odd.

Figure 42-8. Divided Clock Generation

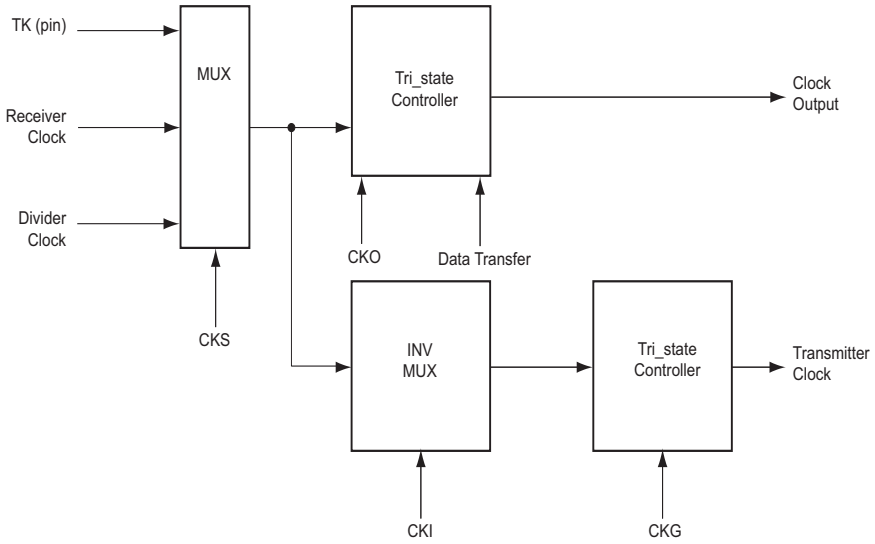


42.8.1.2 Transmitter Clock Management

The transmitter clock is generated from the receiver clock or the divider clock or an external clock scanned on the TK I/O pad. The transmitter clock is selected by the CKS field in the Transmit Clock Mode Register (SSC_TCMR). Transmit Clock can be inverted independently by the CKI bits in the SSC_TCMR.

The transmitter can also drive the TK I/O pad continuously or be limited to the actual data transfer. The clock output is configured by the SSC_TCMR. The Transmit Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_TCMR to select TK pin (CKS field) and at the same time Continuous Transmit Clock (CKO field) can lead to unpredictable results.

Figure 42-9. Transmitter Clock Management

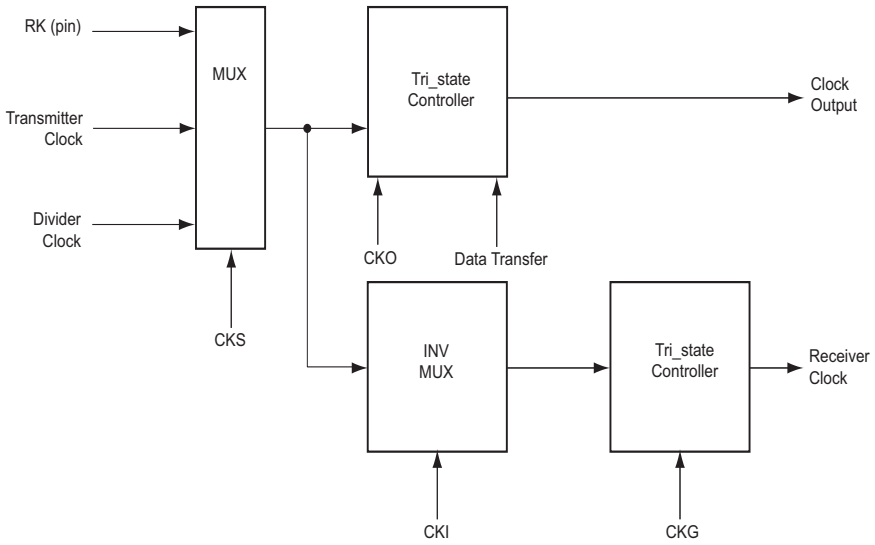


42.8.1.3 Receiver Clock Management

The receiver clock is generated from the transmitter clock or the divider clock or an external clock scanned on the RK I/O pad. The Receive Clock is selected by the CKS field in SSC_RCMR (Receive Clock Mode Register). Receive Clocks can be inverted independently by the CKI bits in SSC_RCMR.

The receiver can also drive the RK I/O pad continuously or be limited to the actual data transfer. The clock output is configured by the SSC_RCMR. The Receive Clock Inversion (CKI) bits have no effect on the clock outputs. Programming the SSC_RCMR to select RK pin (CKS field) and at the same time Continuous Receive Clock (CKO field) can lead to unpredictable results.

Figure 42-10. Receiver Clock Management



42.8.1.4 Serial Clock Ratio Considerations

The Transmitter and the Receiver can be programmed to operate with the clock signals provided on either the TK or RK pins. This allows the SSC to support many Slave mode data transfers. In this case, the maximum clock speed allowed on the RK pin is:

- Peripheral clock divided by 2 if Receiver Frame Synchro is input
- Peripheral clock divided by 3 if Receiver Frame Synchro is output

In addition, the maximum clock speed allowed on the TK pin is:

- Peripheral clock divided by 6 if Transmit Frame Synchro is input
- Peripheral clock divided by 2 if Transmit Frame Synchro is output

42.8.2 Transmitter Operations

A transmitted frame is triggered by a start event and can be followed by synchronization data before data transmission.

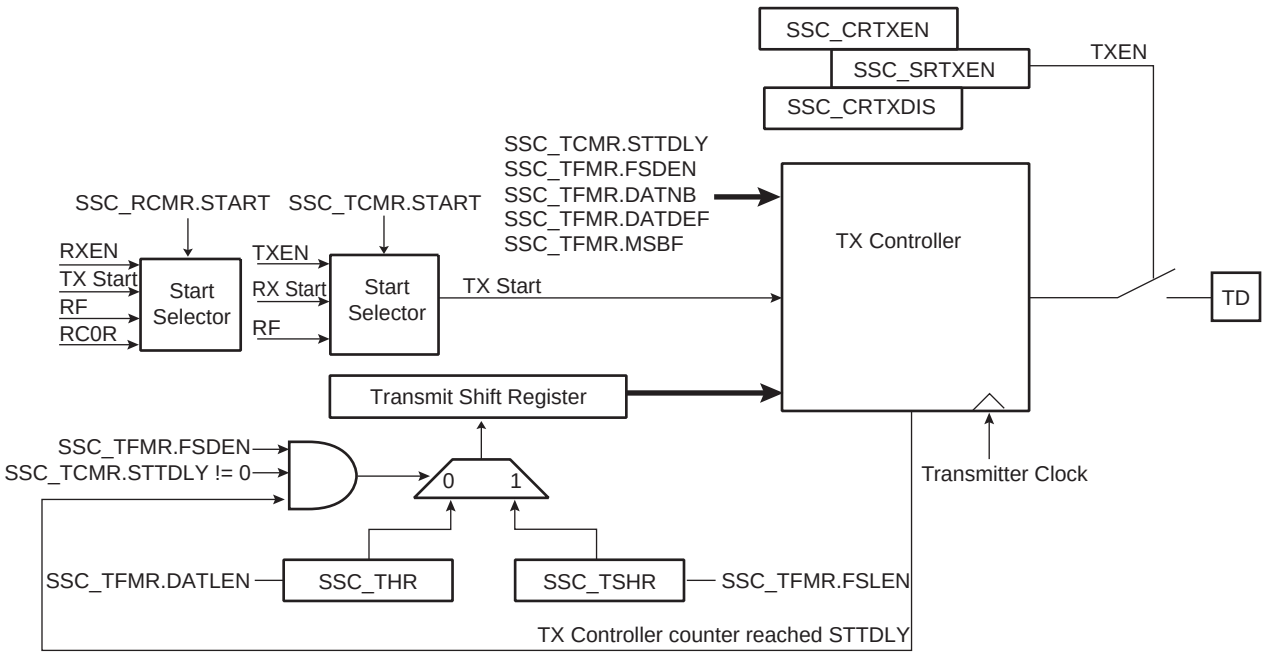
The start event is configured by setting the SSC_TCMR. See Section 42.8.4 “Start”.

The frame synchronization is configured setting the Transmit Frame Mode Register (SSC_TFMR). See Section 42.8.5 “Frame Sync”.

To transmit data, the transmitter uses a shift register clocked by the transmitter clock signal and the start mode selected in the SSC_TCMR. Data is written by the application to the SSC_THR then transferred to the shift register according to the data format selected.

When both the SSC_THR and the transmit shift register are empty, the status flag TXEMPTY is set in the SSC_SR. When the Transmit Holding register is transferred in the transmit shift register, the status flag TXRDY is set in the SSC_SR and additional data can be loaded in the holding register.

Figure 42-11. Transmitter Block Diagram



42.8.3 Receiver Operations

A received frame is triggered by a start event and can be followed by synchronization data before data transmission.

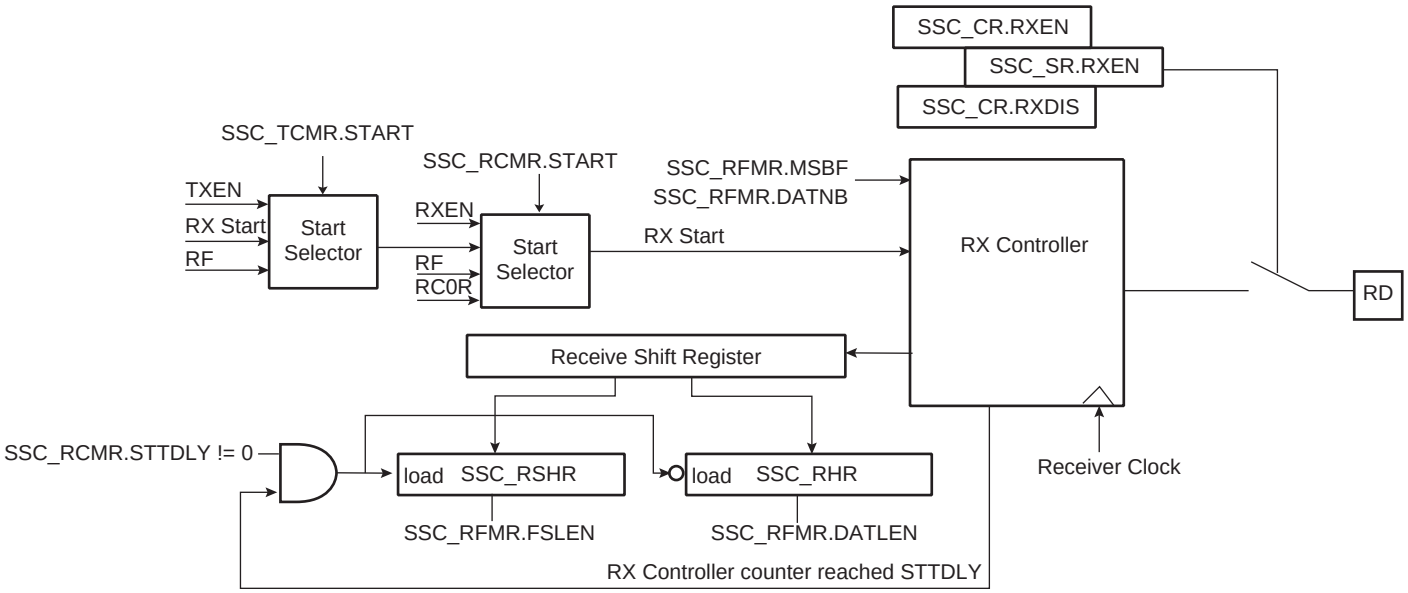
The start event is configured setting the Receive Clock Mode Register (SSC_RCMR). See Section 42.8.4 “Start”.

The frame synchronization is configured setting the Receive Frame Mode Register (SSC_RFMR). See Section 42.8.5 “Frame Sync”.

The receiver uses a shift register clocked by the receiver clock signal and the start mode selected in the SSC_RCMR. The data is transferred from the shift register depending on the data format selected.

When the receiver shift register is full, the SSC transfers this data in the holding register, the status flag RXRDY is set in the SSC_SR and the data can be read in the receiver holding register. If another transfer occurs before read of the Receive Holding Register (SSC_RHR), the status flag OVERUN is set in the SSC_SR and the receiver shift register is transferred in the SSC_RHR.

Figure 42-12. Receiver Block Diagram



42.8.4 Start

The transmitter and receiver can both be programmed to start their operations when an event occurs, respectively in the Transmit Start Selection (START) field of SSC_TCMR and in the Receive Start Selection (START) field of SSC_RCMR.

Under the following conditions the start event is independently programmable:

- Continuous. In this case, the transmission starts as soon as a word is written in SSC_THR and the reception starts as soon as the Receiver is enabled.
- Synchronously with the transmitter/receiver
- On detection of a falling/rising edge on TF/RF
- On detection of a low level/high level on TF/RF
- On detection of a level change or an edge on TF/RF

A start can be programmed in the same manner on either side of the Transmit/Receive Clock Register (SSC_RCMR/SSC_TCMR). Thus, the start could be on TF (Transmit) or RF (Receive).

Moreover, the Receiver can start when data is detected in the bit stream with the Compare Functions.

Detection on TF/RF input/output is done by the field FSOS of the Transmit/Receive Frame Mode Register (SSC_TFMR/SSC_RFMR).

Figure 42-13. Transmit Start Mode

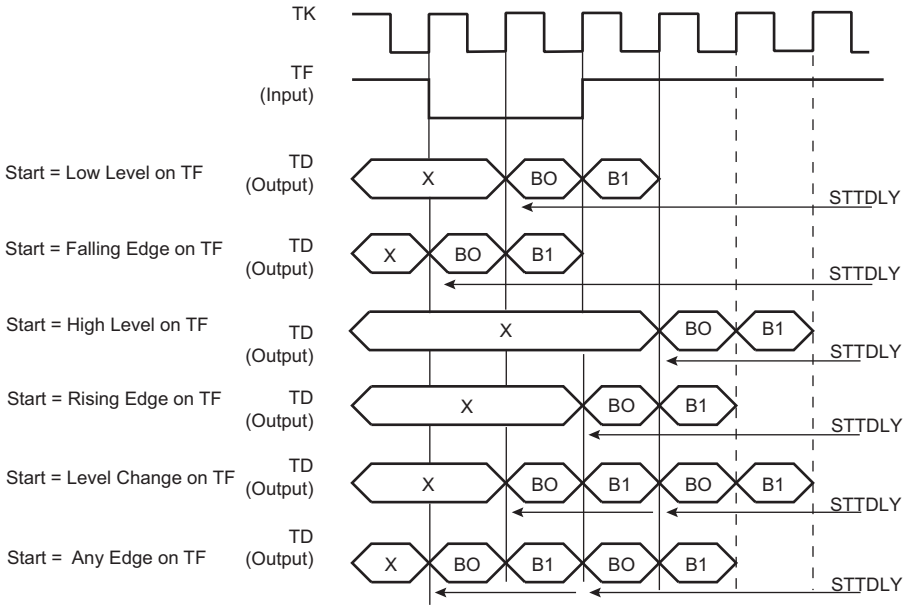
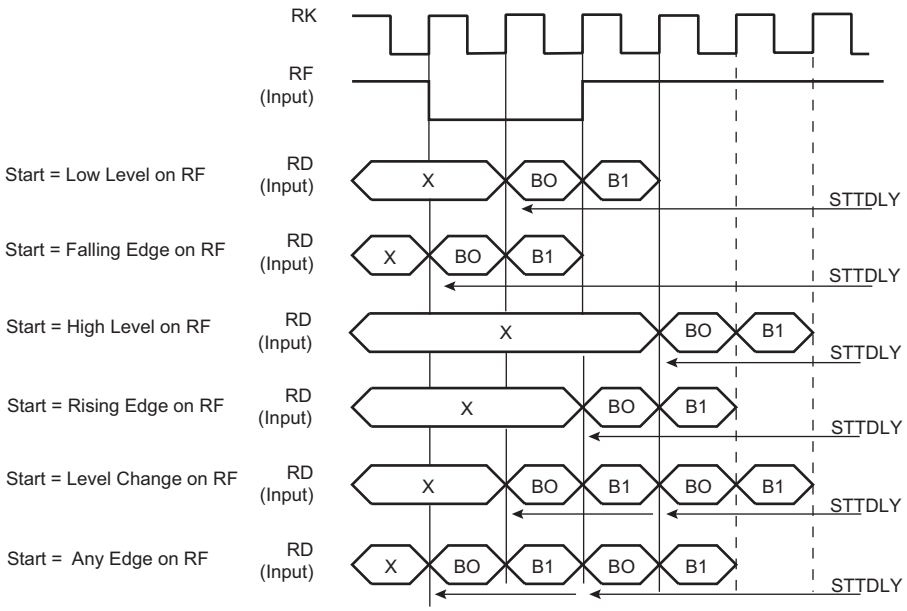


Figure 42-14. Receive Pulse/Edge Start Modes



42.8.5 Frame Sync

The Transmitter and Receiver Frame Sync pins, TF and RF, can be programmed to generate different kinds of frame synchronization signals. The Frame Sync Output Selection (FSOS) field in the Receive Frame Mode Register (SSC_RFMR) and in the Transmit Frame Mode Register (SSC_TFMR) are used to select the required waveform.

- Programmable low or high levels during data transfer are supported.
- Programmable high levels before the start of data transfers or toggling are also supported.

If a pulse waveform is selected, the Frame Sync Length (FSLEN) field in SSC_RFMR and SSC_TFMR programs the length of the pulse, from 1 bit time up to 256 bit times.

The periodicity of the Receive and Transmit Frame Sync pulse output can be programmed through the Period Divider Selection (PERIOD) field in SSC_RCMR and SSC_TCMR.

42.8.5.1 Frame Sync Data

Frame Sync Data transmits or receives a specific tag during the Frame Sync signal.

During the Frame Sync signal, the Receiver can sample the RD line and store the data in the Receive Sync Holding Register and the transmitter can transfer Transmit Sync Holding Register in the shift register. The data length to be sampled/shifted out during the Frame Sync signal is programmed by the FSLEN field in SSC_RFMR/SSC_TFMR and has a maximum value of 256.

Concerning the Receive Frame Sync Data operation, if the Frame Sync Length is equal to or lower than the delay between the start event and the actual data reception, the data sampling operation is performed in the Receive Sync Holding Register through the receive shift register.

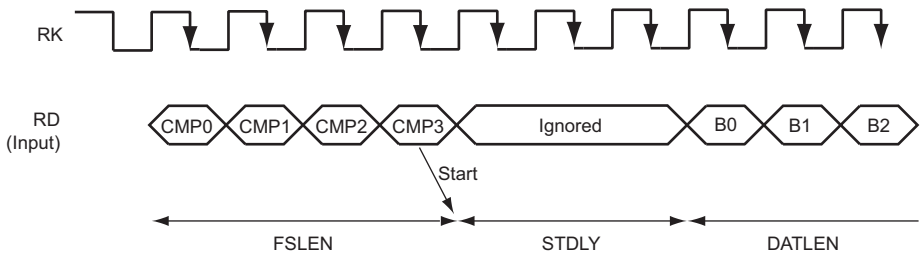
The Transmit Frame Sync Operation is performed by the transmitter only if the bit Frame Sync Data Enable (FSDEN) in SSC_TFMR is set. If the Frame Sync length is equal to or lower than the delay between the start event and the actual data transmission, the normal transmission has priority and the data contained in the Transmit Sync Holding Register is transferred in the Transmit Register, then shifted out.

42.8.5.2 Frame Sync Edge Detection

The Frame Sync Edge detection is programmed by the FSEDGE field in SSC_RFMR/SSC_TFMR. This sets the corresponding flags RXSYN/TXSYN in the SSC Status Register (SSC_SR) on frame synchro edge detection (signals RF/TF).

42.8.6 Receive Compare Modes

Figure 42-15. Receive Compare Modes



42.8.6.1 Compare Functions

The length of the comparison patterns (Compare 0, Compare 1) and thus the number of bits they are compared to is defined by FSLEN, but with a maximum value of 256 bits. Comparison is always done by comparing the last bits received with the comparison pattern. Compare 0 can be one start event of the Receiver. In this case, the receiver compares at each new sample the last bits received at the Compare 0 pattern contained in the Compare 0 Register (SSC_RC0R). When this start event is selected, the user can program the Receiver to start a new data transfer either by writing a new Compare 0, or by receiving continuously until Compare 1 occurs. This selection is done with the STOP bit in the SSC_RCMR.

42.8.7 Data Format

The data framing format of both the transmitter and the receiver are programmable through the Transmitter Frame Mode Register (SSC_TFMR) and the Receiver Frame Mode Register (SSC_RFMR). In either case, the user can independently select the following parameters:

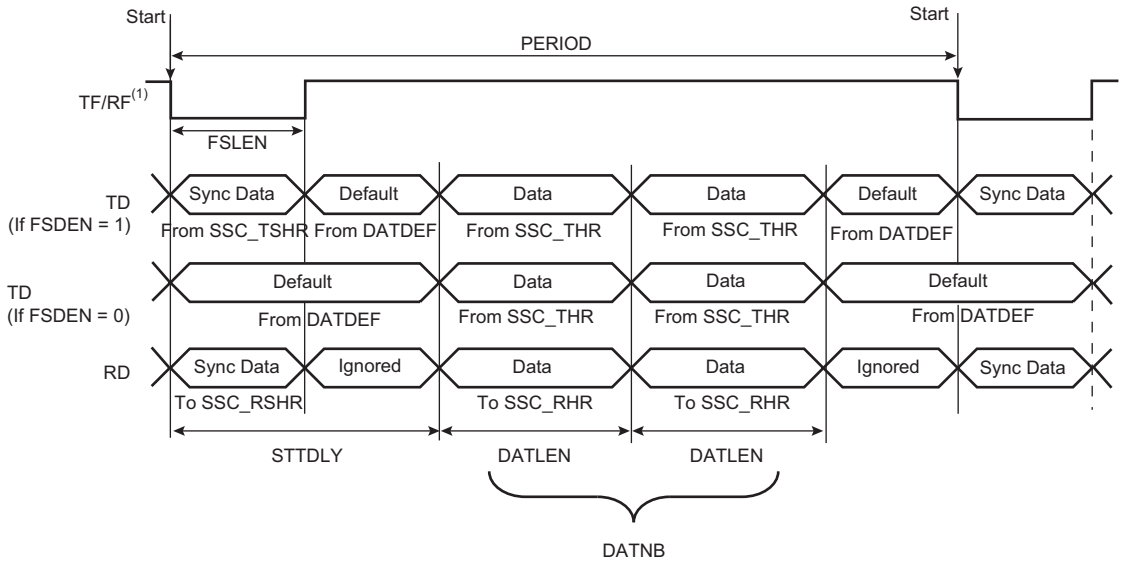
- Event that starts the data transfer (START)
- Delay in number of bit periods between the start event and the first data bit (STTDLY)
- Length of the data (DATLEN)
- Number of data to be transferred for each start event (DATNB)
- Length of synchronization transferred for each start event (FSLEN)
- Bit sense: most or lowest significant bit first (MSBF)

Additionally, the transmitter can be used to transfer synchronization and select the level driven on the TD pin while not in data transfer operation. This is done respectively by the Frame Sync Data Enable (FSDEN) and by the Data Default Value (DATDEF) bits in SSC_TFMR.

Table 42-4. Data Frame Registers

Transmitter	Receiver	Field	Length	Comment
SSC_TFMR	SSC_RFMR	DATLEN	Up to 32	Size of word
SSC_TFMR	SSC_RFMR	DATNB	Up to 16	Number of words transmitted in frame
SSC_TFMR	SSC_RFMR	MSBF	–	Most significant bit first
SSC_TFMR	SSC_RFMR	FSLEN	Up to 256	Size of Synchro data register
SSC_TFMR	–	DATDEF	0 or 1	Data default value ended
SSC_TFMR	–	FSDEN	–	Enable send SSC_TSHR
SSC_TCMR	SSC_RCMR	PERIOD	Up to 512	Frame size
SSC_TCMR	SSC_RCMR	STTDLY	Up to 255	Size of transmit start delay

Figure 42-16. Transmit and Receive Frame Format in Edge/Pulse Start Modes



Note: 1. Example of input on falling edge of TF/RF.

In the example illustrated in Figure 42-17, the SSC_THR is loaded twice. The FSDEN value has no effect on the transmission. SyncData cannot be output in Continuous mode.

Figure 42-17. Transmit Frame Format in Continuous Mode (STTDLY = 0)

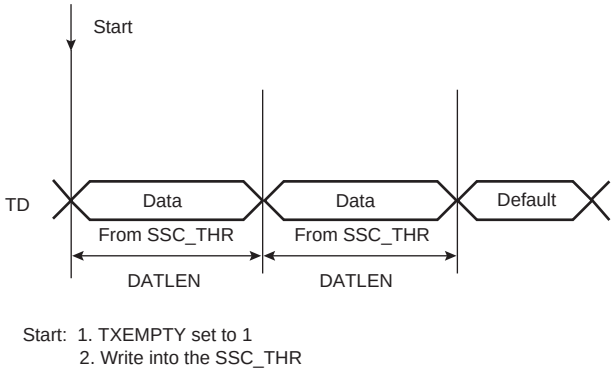
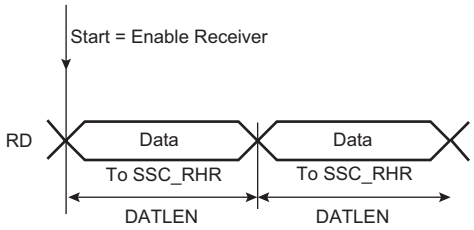


Figure 42-18. Receive Frame Format in Continuous Mode (STTDLY = 0)



42.8.8 Loop Mode

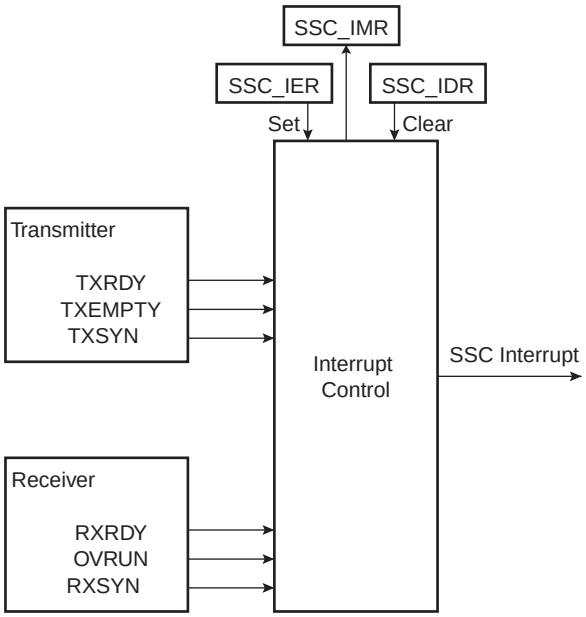
The receiver can be programmed to receive transmissions from the transmitter. This is done by setting the Loop Mode (LOOP) bit in the SSC_RFMR. In this case, RD is connected to TD, RF is connected to TF and RK is connected to TK.

42.8.9 Interrupt

Most bits in the SSC_SR have a corresponding bit in interrupt management registers.

The SSC can be programmed to generate an interrupt when it detects an event. The interrupt is controlled by writing the Interrupt Enable Register (SSC_IER) and Interrupt Disable Register (SSC_IDR). These registers enable and disable, respectively, the corresponding interrupt by setting and clearing the corresponding bit in the Interrupt Mask Register (SSC_IMR), which controls the generation of interrupts by asserting the SSC interrupt line connected to the interrupt controller.

Figure 42-19. Interrupt Block Diagram



42.8.10 Register Write Protection

To prevent any single software error from corrupting SSC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the SSC Write Protection Mode Register (SSC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the SSC Write Protection Status Register (SSC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the SSC_WPSR.

The following registers can be write-protected:

- SSC Clock Mode Register
- SSC Receive Clock Mode Register
- SSC Receive Frame Mode Register
- SSC Transmit Clock Mode Register
- SSC Transmit Frame Mode Register
- SSC Receive Compare 0 Register
- SSC Receive Compare 1 Register

42.9 Synchronous Serial Controller (SSC) User Interface

Table 42-5. Register Mapping

Offset	Register	Name	Access	Reset
0x0	Control Register	SSC_CR	Write-only	–
0x4	Clock Mode Register	SSC_CMR	Read/Write	0x0
0x8–0xC	Reserved	–	–	–
0x10	Receive Clock Mode Register	SSC_RCMR	Read/Write	0x0
0x14	Receive Frame Mode Register	SSC_RFMR	Read/Write	0x0
0x18	Transmit Clock Mode Register	SSC_TCMR	Read/Write	0x0
0x1C	Transmit Frame Mode Register	SSC_TFMR	Read/Write	0x0
0x20	Receive Holding Register	SSC_RHR	Read-only	0x0
0x24	Transmit Holding Register	SSC_THR	Write-only	–
0x28–0x2C	Reserved	–	–	–
0x30	Receive Sync. Holding Register	SSC_RSHR	Read-only	0x0
0x34	Transmit Sync. Holding Register	SSC_TSHR	Read/Write	0x0
0x38	Receive Compare 0 Register	SSC_RC0R	Read/Write	0x0
0x3C	Receive Compare 1 Register	SSC_RC1R	Read/Write	0x0
0x40	Status Register	SSC_SR	Read-only	0x000000CC
0x44	Interrupt Enable Register	SSC_IER	Write-only	–
0x48	Interrupt Disable Register	SSC_IDR	Write-only	–
0x4C	Interrupt Mask Register	SSC_IMR	Read-only	0x0
0x50–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	SSC_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	SSC_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–
0x100–0x124	Reserved	–	–	–

42.9.1 SSC Control Register

Name: SSC_CR

Address: 0xF8004000 (0), 0xFC004000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
SWRST	–	–	–	–	–	TXDIS	TXEN
7	6	5	4	3	2	1	0
–	–	–	–	–	–	RXDIS	RXEN

- **RXEN: Receive Enable**

0: No effect.

1: Enables Receive if RXDIS is not set.

- **RXDIS: Receive Disable**

0: No effect.

1: Disables Receive. If a character is currently being received, disables at end of current character reception.

- **TXEN: Transmit Enable**

0: No effect.

1: Enables Transmit if TXDIS is not set.

- **TXDIS: Transmit Disable**

0: No effect.

1: Disables Transmit. If a character is currently being transmitted, disables at end of current character transmission.

- **SWRST: Software Reset**

0: No effect.

1: Performs a software reset. Has priority on any other bit in SSC_CR.

42.9.2 SSC Clock Mode Register

Name: SSC_CMR

Address: 0xF8004004 (0), 0xFC004004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	DIV			

7	6	5	4	3	2	1	0
DIV							

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **DIV: Clock Divider**

0: The Clock Divider is not active.

Any other value: The divided clock equals the peripheral clock divided by 2 times DIV.
The maximum bit rate is $f_{peripheral\ clock}/2$. The minimum bit rate is $f_{peripheral\ clock}/2 \times 4095 = f_{peripheral\ clock}/8190$.

42.9.3 SSC Receive Clock Mode Register

Name: SSC_RCMR

Address: 0xF8004010 (0), 0xFC004010 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
PERIOD							

23	22	21	20	19	18	17	16
STTDLY							

15	14	13	12	11	10	9	8
–	–	–	STOP	START			

7	6	5	4	3	2	1	0
CKG		CKI	CKO			CKS	

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **CKS: Receive Clock Selection**

Value	Name	Description
0	MCK	Divided Clock
1	TK	TK Clock signal
2	RK	RK pin

- **CKO: Receive Clock Output Mode Selection**

Value	Name	Description
0	NONE	None, RK pin is an input
1	CONTINUOUS	Continuous Receive Clock, RK pin is an output
2	TRANSFER	Receive Clock only during data transfers, RK pin is an output

- **CKI: Receive Clock Inversion**

0: The data inputs (Data and Frame Sync signals) are sampled on Receive Clock falling edge. The Frame Sync signal output is shifted out on Receive Clock rising edge.

1: The data inputs (Data and Frame Sync signals) are sampled on Receive Clock rising edge. The Frame Sync signal output is shifted out on Receive Clock falling edge.

CKI affects only the Receive Clock and not the output clock signal.

- **CKG: Receive Clock Gating Selection**

Value	Name	Description
0	CONTINUOUS	None
1	EN_RF_LOW	Receive Clock enabled only if RF Low
2	EN_RF_HIGH	Receive Clock enabled only if RF High

- **START: Receive Start Selection**

Value	Name	Description
0	CONTINUOUS	Continuous, as soon as the receiver is enabled, and immediately after the end of transfer of the previous data.
1	TRANSMIT	Transmit start
2	RF_LOW	Detection of a low level on RF signal
3	RF_HIGH	Detection of a high level on RF signal
4	RF_FALLING	Detection of a falling edge on RF signal
5	RF_RISING	Detection of a rising edge on RF signal
6	RF_LEVEL	Detection of any level change on RF signal
7	RF_EDGE	Detection of any edge on RF signal
8	CMP_0	Compare 0

- **STOP: Receive Stop Selection**

0: After completion of a data transfer when starting with a Compare 0, the receiver stops the data transfer and waits for a new compare 0.

1: After starting a receive with a Compare 0, the receiver operates in a continuous mode until a Compare 1 is detected.

- **STTDLY: Receive Start Delay**

If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the actual start of reception. When the Receiver is programmed to start synchronously with the Transmitter, the delay is also applied.

Note: It is very important that STTDLY be set carefully. If STTDLY must be set, it should be done in relation to TAG (Receive Sync Data) reception.

- **PERIOD: Receive Period Divider Selection**

This field selects the divider to apply to the selected Receive Clock in order to generate a new Frame Sync Signal. If 0, no PERIOD signal is generated. If not 0, a PERIOD signal is generated each 2 x (PERIOD + 1) Receive Clock.

42.9.4 SSC Receive Frame Mode Register

Name: SSC_RFMR

Address: 0xF8004014 (0), 0xFC004014 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
FSLEN_EXT				–	–	–	FSEDGE

23	22	21	20	19	18	17	16
–	FSOS			FSLEN			

15	14	13	12	11	10	9	8
–	–	–	–	DATNB			

7	6	5	4	3	2	1	0
MSBF	–	LOOP	DATLEN				

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **DATLEN: Data Length**

0: Forbidden value (1-bit data length not supported).

Any other value: The bit stream contains DATLEN + 1 data bits.

- **LOOP: Loop Mode**

0: Normal operating mode.

1: RD is driven by TD, RF is driven by TF and TK drives RK.

- **MSBF: Most Significant Bit First**

0: The lowest significant bit of the data register is sampled first in the bit stream.

1: The most significant bit of the data register is sampled first in the bit stream.

- **DATNB: Data Number per Frame**

This field defines the number of data words to be received after each transfer start, which is equal to (DATNB + 1).

- **FSLEN: Receive Frame Sync Length**

This field defines the number of bits sampled and stored in the Receive Sync Data Register. When this mode is selected by the START field in the Receive Clock Mode Register, it also determines the length of the sampled data to be compared to the Compare 0 or Compare 1 register.

This field is used with FSLEN_EXT to determine the pulse length of the Receive Frame Sync signal.

Pulse length is equal to FSLEN + (FSLEN_EXT $\times$ 16) + 1 Receive Clock periods.

- **FSOS: Receive Frame Sync Output Selection**

Value	Name	Description
0	NONE	None, RF pin is an input
1	NEGATIVE	Negative Pulse, RF pin is an output
2	POSITIVE	Positive Pulse, RF pin is an output
3	LOW	Driven Low during data transfer, RF pin is an output
4	HIGH	Driven High during data transfer, RF pin is an output
5	TOGGLING	Toggling at each start of data transfer, RF pin is an output

- **FSEDGE: Frame Sync Edge Detection**

Determines which edge on Frame Sync will generate the interrupt RXSYN in the SSC Status Register.

Value	Name	Description
0	POSITIVE	Positive Edge Detection
1	NEGATIVE	Negative Edge Detection

- **FSLEN_EXT: FSLEN Field Extension**

Extends FSLEN field. For details, refer to FSLEN bit description above.

42.9.5 SSC Transmit Clock Mode Register

Name: SSC_TCMR

Address: 0xF8004018 (0), 0xFC004018 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
PERIOD							
23	22	21	20	19	18	17	16
STTDLY							
15	14	13	12	11	10	9	8
–	–	–	–	START			
7	6	5	4	3	2	1	0
CKG		CKI	CKO			CKS	

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **CKS: Transmit Clock Selection**

Value	Name	Description
0	MCK	Divided Clock
1	RK	RK Clock signal
2	TK	TK pin

- **CKO: Transmit Clock Output Mode Selection**

Value	Name	Description
0	NONE	None, TK pin is an input
1	CONTINUOUS	Continuous Transmit Clock, TK pin is an output
2	TRANSFER	Transmit Clock only during data transfers, TK pin is an output

- **CKI: Transmit Clock Inversion**

0: The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock falling edge. The Frame sync signal input is sampled on Transmit clock rising edge.

1: The data outputs (Data and Frame Sync signals) are shifted out on Transmit Clock rising edge. The Frame sync signal input is sampled on Transmit clock falling edge.

CKI affects only the Transmit Clock and not the output clock signal.

- **CKG: Transmit Clock Gating Selection**

Value	Name	Description
0	CONTINUOUS	None
1	EN_TF_LOW	Transmit Clock enabled only if TF Low
2	EN_TF_HIGH	Transmit Clock enabled only if TF High

- **START: Transmit Start Selection**

Value	Name	Description
0	CONTINUOUS	Continuous, as soon as a word is written in the SSC_THR (if Transmit is enabled), and immediately after the end of transfer of the previous data
1	RECEIVE	Receive start
2	TF_LOW	Detection of a low level on TF signal
3	TF_HIGH	Detection of a high level on TF signal
4	TF_FALLING	Detection of a falling edge on TF signal
5	TF_RISING	Detection of a rising edge on TF signal
6	TF_LEVEL	Detection of any level change on TF signal
7	TF_EDGE	Detection of any edge on TF signal

- **STTDLY: Transmit Start Delay**

If STTDLY is not 0, a delay of STTDLY clock cycles is inserted between the start event and the actual start of transmission of data. When the Transmitter is programmed to start synchronously with the Receiver, the delay is also applied.

Note: STTDLY must be set carefully. If STTDLY is too short in respect to TAG (Transmit Sync Data) transmission, data is transmitted instead of the end of TAG.

- **PERIOD: Transmit Period Divider Selection**

This field selects the divider to apply to the selected Transmit Clock to generate a new Frame Sync Signal. If 0, no period signal is generated. If not 0, a period signal is generated at each 2 × (PERIOD + 1) Transmit Clock.

42.9.6 SSC Transmit Frame Mode Register

Name: SSC_TFMR

Address: 0xF800401C (0), 0xFC00401C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
FSLEN_EXT				–	–	–	FSEDGE
23	22	21	20	19	18	17	16
FSDEN	FSOS			FSLEN			
15	14	13	12	11	10	9	8
–	–	–	–	DATNB			
7	6	5	4	3	2	1	0
MSBF	–	DATDEF	DATLEN				

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **DATLEN: Data Length**

0: Forbidden value (1-bit data length not supported).

Any other value: The bit stream contains DATLEN + 1 data bits.

- **DATDEF: Data Default Value**

This bit defines the level driven on the TD pin while out of transmission. Note that if the pin is defined as multidrive by the PIO Controller, the pin is enabled only if the SCC TD output is 1.

- **MSBF: Most Significant Bit First**

0: The lowest significant bit of the data register is shifted out first in the bit stream.

1: The most significant bit of the data register is shifted out first in the bit stream.

- **DATNB: Data Number per Frame**

This field defines the number of data words to be transferred after each transfer start, which is equal to (DATNB + 1).

- **FSLEN: Transmit Frame Sync Length**

This field defines the length of the Transmit Frame Sync signal and the number of bits shifted out from the Transmit Sync Data Register if FSDEN is 1.

This field is used with FSLEN_EXT to determine the pulse length of the Transmit Frame Sync signal.

Pulse length is equal to FSLEN + (FSLEN_EXT $\times$ 16) + 1 Transmit Clock period.

- **FSOS: Transmit Frame Sync Output Selection**

Value	Name	Description
0	NONE	None, TF pin is an input
1	NEGATIVE	Negative Pulse, TF pin is an output
2	POSITIVE	Positive Pulse, TF pin is an output
3	LOW	Driven Low during data transfer
4	HIGH	Driven High during data transfer
5	TOGGLING	Toggling at each start of data transfer

- **FSDEN: Frame Sync Data Enable**

0: The TD line is driven with the default value during the Transmit Frame Sync signal.

1: SSC_TSHR value is shifted out during the transmission of the Transmit Frame Sync signal.

- **FSEDGE: Frame Sync Edge Detection**

Determines which edge on frame sync will generate the interrupt TXSYN (Status Register).

Value	Name	Description
0	POSITIVE	Positive Edge Detection
1	NEGATIVE	Negative Edge Detection

- **FSLEN_EXT: FSLEN Field Extension**

Extends FSLEN field. For details, refer to FSLEN bit description above.

42.9.7 SSC Receive Holding Register

Name: SSC_RHR

Address: 0xF8004020 (0), 0xFC004020 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RDAT							

23	22	21	20	19	18	17	16
RDAT							

15	14	13	12	11	10	9	8
RDAT							

7	6	5	4	3	2	1	0
RDAT							

- **RDAT: Receive Data**

Right aligned regardless of the number of data bits defined by DATLEN in SSC_RFMR.

42.9.8 SSC Transmit Holding Register

Name: SSC_THR

Address: 0xF8004024 (0), 0xFC004024 (1)

Access: Write-only

31	30	29	28	27	26	25	24
TDAT							

23	22	21	20	19	18	17	16
TDAT							

15	14	13	12	11	10	9	8
TDAT							

7	6	5	4	3	2	1	0
TDAT							

- **TDAT: Transmit Data**

Right aligned regardless of the number of data bits defined by DATLEN in SSC_TFMR.

42.9.9 SSC Receive Synchronization Holding Register

Name: SSC_RSHR

Address: 0xF8004030 (0), 0xFC004030 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RSDAT							

7	6	5	4	3	2	1	0
RSDAT							

- **RSDAT: Receive Synchronization Data**

42.9.10 SSC Transmit Synchronization Holding Register

Name: SSC_TSHR

Address: 0xF8004034 (0), 0xFC004034 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TSDAT							

7	6	5	4	3	2	1	0
TSDAT							

- **TSDAT: Transmit Synchronization Data**

42.9.11 SSC Receive Compare 0 Register

Name: SSC_RC0R

Address: 0xF8004038 (0), 0xFC004038 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CP0							
7	6	5	4	3	2	1	0
CP0							

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **CP0: Receive Compare Data 0**

42.9.12 SSC Receive Compare 1 Register

Name: SSC_RC1R

Address: 0xF800403C (0), 0xFC00403C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CP1							
7	6	5	4	3	2	1	0
CP1							

This register can only be written if the WPEN bit is cleared in the SSC Write Protection Mode Register.

- **CP1: Receive Compare Data 1**

42.9.13 SSC Status Register

Name: SSC_SR

Address: 0xF8004040 (0), 0xFC004040 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	RXEN	TXEN
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready**

0: Data has been loaded in SSC_THR and is waiting to be loaded in the transmit shift register (TSR).

1: SSC_THR is empty.

- **TXEMPTY: Transmit Empty**

0: Data remains in SSC_THR or is currently transmitted from TSR.

1: Last data written in SSC_THR has been loaded in TSR and last data loaded in TSR has been transmitted.

- **RXRDY: Receive Ready**

0: SSC_RHR is empty.

1: Data has been received and loaded in SSC_RHR.

- **OVRUN: Receive Overrun**

0: No data has been loaded in SSC_RHR while previous data has not been read since the last read of the Status Register.

1: Data has been loaded in SSC_RHR while previous data has not yet been read since the last read of the Status Register.

- **CP0: Compare 0**

0: A compare 0 has not occurred since the last read of the Status Register.

1: A compare 0 has occurred since the last read of the Status Register.

- **CP1: Compare 1**

0: A compare 1 has not occurred since the last read of the Status Register.

1: A compare 1 has occurred since the last read of the Status Register.

- **TXSYN: Transmit Sync**

0: A Tx Sync has not occurred since the last read of the Status Register.

1: A Tx Sync has occurred since the last read of the Status Register.

- **RXSYN: Receive Sync**

0: An Rx Sync has not occurred since the last read of the Status Register.

1: An Rx Sync has occurred since the last read of the Status Register.

- **TXEN: Transmit Enable**

0: Transmit is disabled.

1: Transmit is enabled.

- **RXEN: Receive Enable**

0: Receive is disabled.

1: Receive is enabled.

42.9.14 SSC Interrupt Enable Register

Name: SSC_IER

Address: 0xF8004044 (0), 0xFC004044 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready Interrupt Enable**

0: No effect.

1: Enables the Transmit Ready Interrupt.

- **TXEMPTY: Transmit Empty Interrupt Enable**

0: No effect.

1: Enables the Transmit Empty Interrupt.

- **RXRDY: Receive Ready Interrupt Enable**

0: No effect.

1: Enables the Receive Ready Interrupt.

- **OVRUN: Receive Overrun Interrupt Enable**

0: No effect.

1: Enables the Receive Overrun Interrupt.

- **CP0: Compare 0 Interrupt Enable**

0: No effect.

1: Enables the Compare 0 Interrupt.

- **CP1: Compare 1 Interrupt Enable**

0: No effect.

1: Enables the Compare 1 Interrupt.

- **TXSYN: Tx Sync Interrupt Enable**

0: No effect.

1: Enables the Tx Sync Interrupt.

- **RXSYN: Rx Sync Interrupt Enable**

0: No effect.

1: Enables the Rx Sync Interrupt.

42.9.15 SSC Interrupt Disable Register

Name: SSC_IDR

Address: 0xF8004048 (0), 0xFC004048 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready Interrupt Disable**

0: No effect.

1: Disables the Transmit Ready Interrupt.

- **TXEMPTY: Transmit Empty Interrupt Disable**

0: No effect.

1: Disables the Transmit Empty Interrupt.

- **RXRDY: Receive Ready Interrupt Disable**

0: No effect.

1: Disables the Receive Ready Interrupt.

- **OVRUN: Receive Overrun Interrupt Disable**

0: No effect.

1: Disables the Receive Overrun Interrupt.

- **CP0: Compare 0 Interrupt Disable**

0: No effect.

1: Disables the Compare 0 Interrupt.

- **CP1: Compare 1 Interrupt Disable**

0: No effect.

1: Disables the Compare 1 Interrupt.

- **TXSYN: Tx Sync Interrupt Enable**

0: No effect.

1: Disables the Tx Sync Interrupt.

- **RXSYN: Rx Sync Interrupt Enable**

0: No effect.

1: Disables the Rx Sync Interrupt.

42.9.16 SSC Interrupt Mask Register

Name: SSC_IMR

Address: 0xF800404C (0), 0xFC00404C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	RXSYN	TXSYN	CP1	CP0
7	6	5	4	3	2	1	0
–	–	OVRUN	RXRDY	–	–	TXEMPTY	TXRDY

- **TXRDY: Transmit Ready Interrupt Mask**

0: The Transmit Ready Interrupt is disabled.

1: The Transmit Ready Interrupt is enabled.

- **TXEMPTY: Transmit Empty Interrupt Mask**

0: The Transmit Empty Interrupt is disabled.

1: The Transmit Empty Interrupt is enabled.

- **RXRDY: Receive Ready Interrupt Mask**

0: The Receive Ready Interrupt is disabled.

1: The Receive Ready Interrupt is enabled.

- **OVRUN: Receive Overrun Interrupt Mask**

0: The Receive Overrun Interrupt is disabled.

1: The Receive Overrun Interrupt is enabled.

- **CP0: Compare 0 Interrupt Mask**

0: The Compare 0 Interrupt is disabled.

1: The Compare 0 Interrupt is enabled.

- **CP1: Compare 1 Interrupt Mask**

0: The Compare 1 Interrupt is disabled.

1: The Compare 1 Interrupt is enabled.

- **TXSYN: Tx Sync Interrupt Mask**

0: The Tx Sync Interrupt is disabled.

1: The Tx Sync Interrupt is enabled.

- **RXSYN: Rx Sync Interrupt Mask**

0: The Rx Sync Interrupt is disabled.

1: The Rx Sync Interrupt is enabled.

42.9.17 SSC Write Protection Mode Register

Name: SSC_WPMR

Address: 0xF80040E4 (0), 0xFC0040E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x535343 (“SSC” in ASCII).

See Section 42.8.10 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x535343	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

42.9.18 SSC Write Protection Status Register

Name: SSC_WPSR

Address: 0xF80040E8 (0), 0xFC0040E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the SSC_WPSR.

1: A write protection violation has occurred since the last read of the SSC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protect Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

43. Two-wire Interface (TWIHS)

43.1 Description

The Atmel Two-wire Interface (TWIHS) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbit/s in Fast mode and up to 3.4 Mbit/s in High-speed slave mode only, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus Serial EEPROM and I²C-compatible devices, such as a Real-Time Clock (RTC), Dot Matrix/Graphic LCD Controller and temperature sensor. The TWIHS is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

Table 43-1 lists the compatibility level of the Atmel Two-wire Interface in Master mode and a full I^2C compatible device.

Table 43-1. Atmel TWI Compatibility with I^2C Standard

I^2C Standard	Atmel TWI
Standard Mode Speed (100 kHz)	Supported
Fast Mode Speed (400 kHz)	Supported
High-speed Mode (Slave only, 3.4 MHz)	Supported
7- or 10-bit(1) Slave Addressing	Supported
START Byte(2)	Not Supported
Repeated Start (Sr) Condition	Supported
ACK and NACK Management	Supported
Input Filtering	Supported
Slope Control	Not Supported
Clock Stretching	Supported
Multi Master Capability	Supported

Notes: 1. 10-bit support in Master mode only
2. START + b000000001 + Ack + Sr

43.2 Embedded Characteristics

- 2 TWIHSs
- 16-byte Transmit and Receive FIFOs
- Compatible with Atmel Two-wire Interface Serial Memory and I²C Compatible Devices[1]
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations
- Master and Multimaster Operation (Standard and Fast Modes Only)
- Slave Mode Operation (Standard, Fast and High-Speed Modes)
- Bit Rate: Up to 400 Kbit/s in Fast Mode and 3.4 Mbit/s in High-Speed Mode (Slave Mode Only)
- General Call Supported in Slave Mode
- SleepWalking (Asynchronous and Partial Wakeup)
- SMBus Support
- Connection to DMA Controller (DMA) Channel Capabilities Optimizes Data Transfers
- Register Write Protection

Note: 1. See Table 43-1 for details on compatibility with I²C Standard.

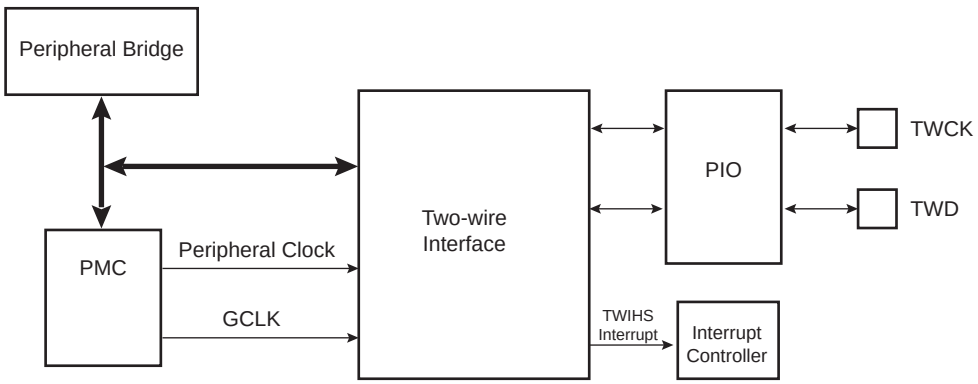
43.3 List of Abbreviations

Table 43-2. Abbreviations

Abbreviation	Description
TWI	Two-wire Interface
A	Acknowledge
NA	Non Acknowledge
P	Stop
S	Start
Sr	Repeated Start
SADR	Slave Address
ADR	Any address except SADR
R	Read
W	Write

43.4 Block Diagram

Figure 43-1. Block Diagram



43.4.1 I/O Lines Description

Table 43-3. I/O Lines Description

Pin Name	Pin Description	Type
TWD	Two-wire Serial Data	Input/Output
TWCK	Two-wire Serial Clock	Input/Output

43.5 Product Dependencies

43.5.1 I/O Lines

Both TWD and TWCK are bidirectional lines, connected to a positive supply voltage via a current source or pull-up resistor. When the bus is free, both lines are high. The output stages of devices connected to the bus must have an open-drain or open-collector to perform the wired-AND function.

TWD and TWCK pins may be multiplexed with PIO lines. To enable the TWIHS, the user must program the PIO Controller to dedicate TWD and TWCK as peripheral lines. When High-speed Slave mode is enabled, the analog pad filter must be enabled.

The user must not program TWD and TWCK as open-drain. This is already done by the hardware.

Table 43-4. I/O Lines

Instance	Signal	I/O Line	Peripheral
TWIHS0	TWCK0	PC0	D
TWIHS0	TWCK0	PC28	E
TWIHS0	TWCK0	PD22	B
TWIHS0	TWCK0	PD30	E
TWIHS0	TWD0	PB31	D
TWIHS0	TWD0	PC27	E
TWIHS0	TWD0	PD21	B
TWIHS0	TWD0	PD29	E
TWIHS1	TWCK1	PC7	C
TWIHS1	TWCK1	PD5	A
TWIHS1	TWCK1	PD20	B
TWIHS1	TWD1	PC6	C
TWIHS1	TWD1	PD4	A
TWIHS1	TWD1	PD19	B

43.5.2 Power Management

Enable the peripheral clock.

The TWIHS may be clocked through the Power Management Controller (PMC), thus the user must first configure the PMC to enable the TWIHS clock.

43.5.3 Interrupt Sources

The TWIHS has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TWIHS.

Table 43-5. Peripheral IDs

Instance	ID
TWIHS0	29
TWIHS1	30

43.6 Functional Description

43.6.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data is transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited. See Figure 43-3.

Each transfer begins with a START condition and terminates with a STOP condition. See Figure 43-2.

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines the STOP condition.

Figure 43-2. START and STOP Conditions

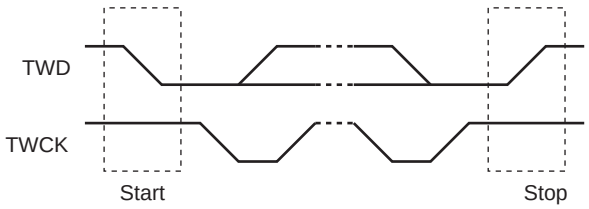
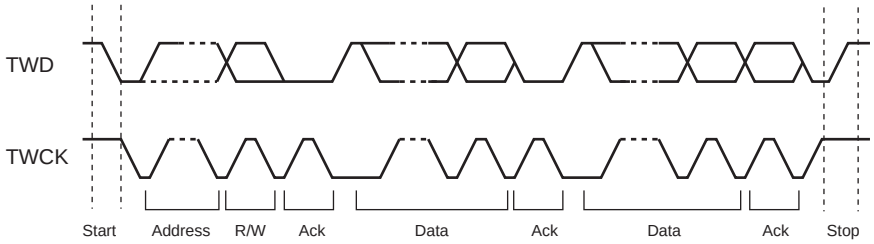


Figure 43-3. Transfer Format



43.6.2 Modes of Operation

The TWIHS has different modes of operation:

- Master Transmitter mode (Standard and Fast modes only)
- Master Receiver mode (Standard and Fast modes only)
- Multimaster Transmitter mode (Standard and Fast modes only)
- Multimaster Receiver mode (Standard and Fast modes only)
- Slave Transmitter mode (Standard, Fast and High-speed modes)
- Slave Receiver mode (Standard, Fast and High-speed modes)

These modes are described in the following sections.

43.6.3 Master Mode

43.6.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it. This operating mode is not available if High-speed mode is selected.

43.6.3.2 Programming Master Mode

The following registers must be programmed before entering Master mode:

1. TWIHS_MMR.DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.
2. TWIHS_CWGR.CKDIV + CHDIV + CLDIV: Clock Waveform register
3. TWIHS_CR.SVDIS: Disables the Slave mode
4. TWIHS_CR.MSEN: Enables the Master mode

Note: If the TWIHS is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

43.6.3.3 Transfer Rate Clock Source

The TWIHS speed is defined in the TWIHS_CWGR. The TWIHS baud rate can be based either on the peripheral clock if the CKSRC bit value is ‘0’ or on a GCLK clock if the CKSRC bit value is ‘1’.

If CKSRC = 1, the baud rate is independent of the system/core clock (MCK) and thus the MCK frequency can be changed without affecting the TWIHS transfer rate.

The GCLK frequency must always be three times lower than the peripheral clock frequency.

43.6.3.4 Master Transmitter Mode

This operating mode is not available if High-speed mode is selected.

After the master initiates a START condition when writing into the Transmit Holding register (TWIHS_THR), it sends a 7-bit slave address, configured in the Master Mode register (DADR in TWIHS_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction, 0 in this case (MREAD = 0 in TWIHS_MMR).

The TWIHS transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWIHS Status Register (TWIHS_SR) of the master and a STOP condition is sent. The NACK flag must be cleared by reading TWIHS_SR before the next write into TWIHS_THR. As with the other status bits, an interrupt can be generated if enabled in the Interrupt Enable register (TWIHS_IER). If the slave acknowledges the byte, the data written in the TWIHS_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in the TWIHS_THR.

TXRDY is used as Transmit Ready for the DMA transmit channel.

While no new data is written in the TWIHS_THR, the serial clock line is tied low. When new data is written in the TWIHS_THR, the SCL is released and the data is sent. Setting the STOP bit in TWIHS_CR generates a STOP condition.

After a master write transfer, the serial clock line is stretched (tied low) as long as no new data is written in the TWIHS_THR or until a STOP command is performed.

To clear the TXRDY flag, first set the bit TWIHS_CR.MSDIS, then set the bit TWIHS_CR.MSEN.

See Figure 43-4, Figure 43-5, and Figure 43-6.

Figure 43-4. Master Write with One Data Byte

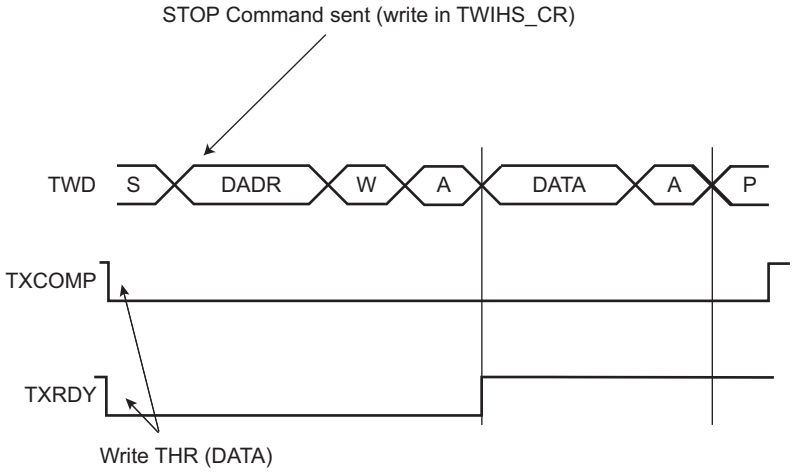


Figure 43-5. Master Write with Multiple Data Bytes

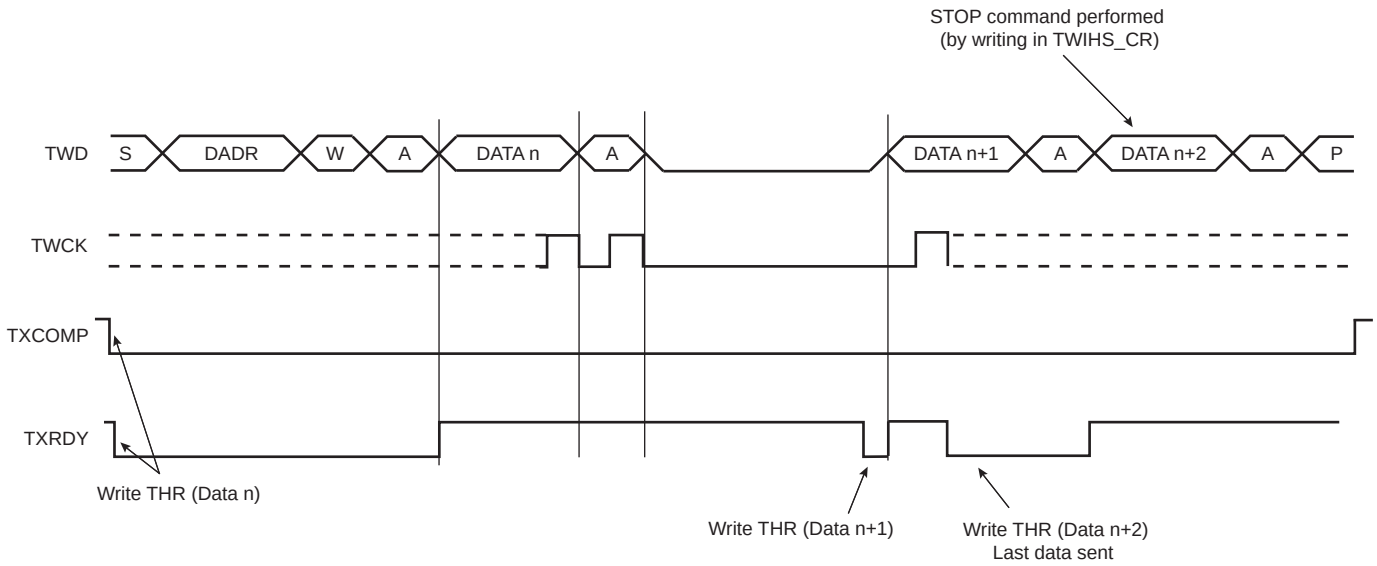
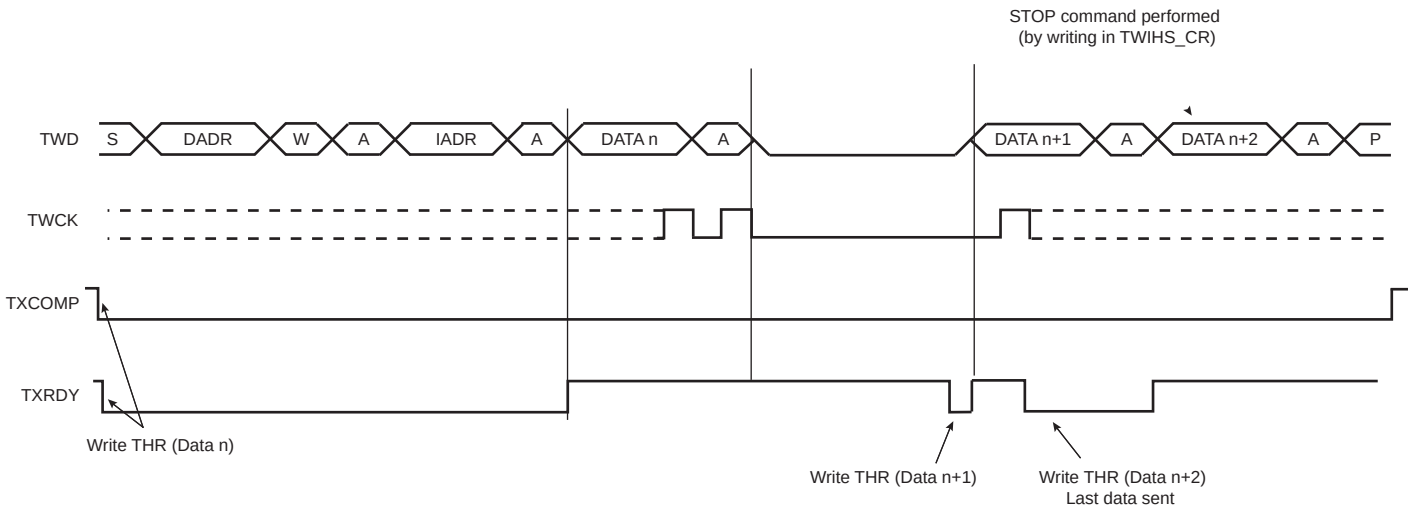


Figure 43-6. Master Write with One-Byte Internal Address and Multiple Data Bytes



43.6.3.5 Master Receiver Mode

Master Receiver mode is not available if High-speed mode is selected.

The read sequence begins by setting the START bit. After the START condition has been sent, the master sends a 7-bit slave address to notify the slave device. The bit following the slave address indicates the transfer direction, 1 in this case (MREAD = 1 in TWIHS_MMR). During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets the NACK bit in the TWIHS_SR if the slave does not acknowledge the byte.

If an acknowledge is received, the master is then ready to receive data from the slave. After data has been received, the master sends an acknowledge condition to notify the slave that the data has been received except for the last data (see Figure 43-7). When the RXRDY bit is set in the TWIHS_SR, a character has been received in the Receive Holding register (TWIHS_RHR). The RXRDY bit is reset when reading the TWIHS_RHR.

When a single data byte read is performed, with or without internal address **(**IADR), the START and STOP bits must be set at the same time. See Figure 43-7. When a multiple data byte read is performed, with or without internal address **(**IADR), the STOP bit must be set after the next-to-last data received (same condition applies for START bit to generate a REPEATED START). See Figure 43-8. For internal address usage, see Section 43.6.3.6 “Internal Address”.

If TWIHS_RHR is full (RXRDY high) and the master is receiving data, the serial clock line is tied low before receiving the last bit of the data and until the TWIHS_RHR is read. Once the TWIHS_RHR is read, the master stops stretching the serial clock line and ends the data reception. See Figure 43-9.

Warning: When receiving multiple bytes in Master Read mode, if the next-to-last access is not read (the RXRDY flag remains high), the last access is not completed until TWIHS_RHR is read. The last access stops on the next-to-last bit (clock stretching). When the TWIHS_RHR is read, there is only half a bit period to send the STOP (or START) command, else another read access might occur (spurious access).

A possible workaround is to set the STOP (or START) bit before reading the TWIHS_RHR on the next-to-last access (within IT handler).

Figure 43-7. Master Read with One Data Byte

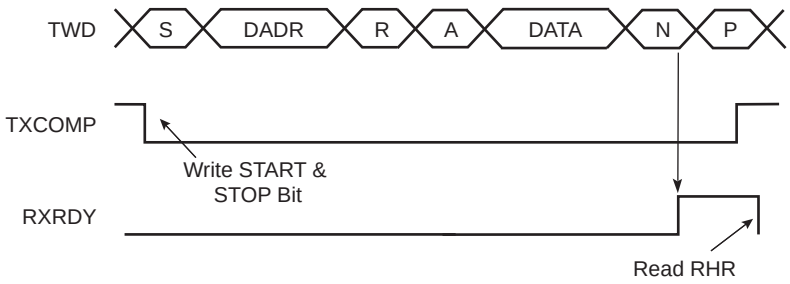


Figure 43-8. Master Read with Multiple Data Bytes

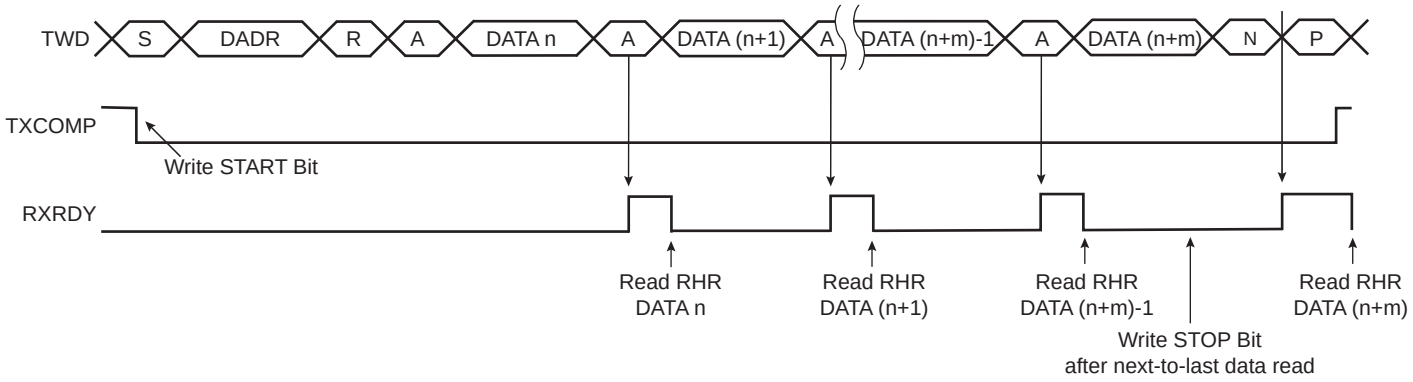
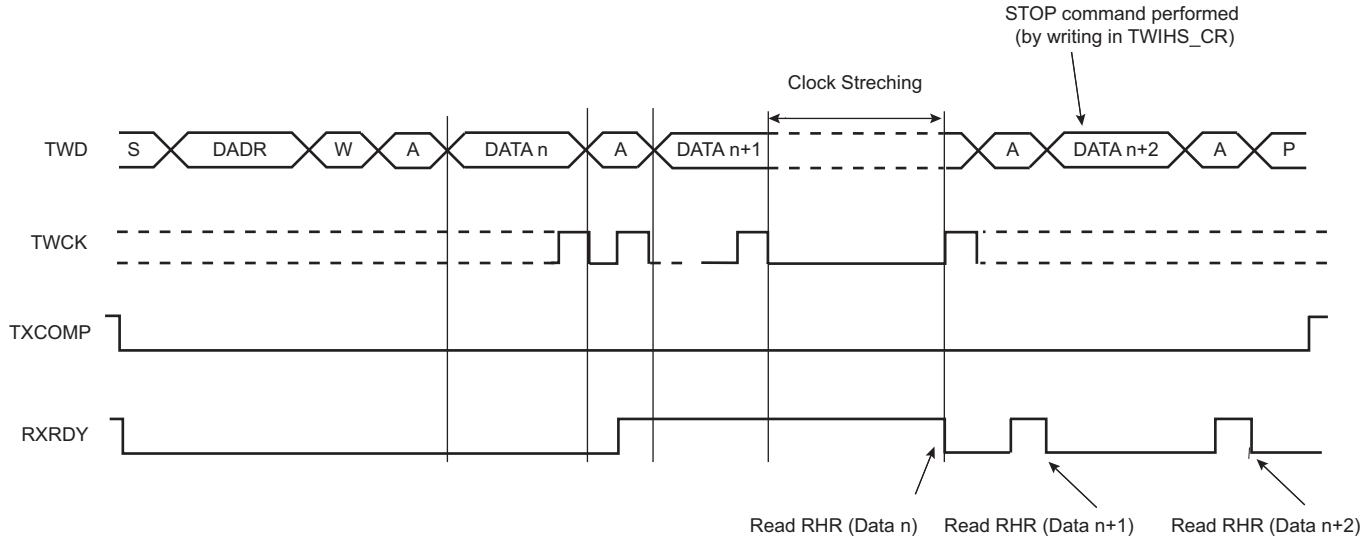


Figure 43-9. Master Read Clock Stretching with Multiple Data Bytes



RXRDY is used as receive ready for the DMA receive channel.

43.6.3.6 Internal Address

The TWIHS can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g. within a memory page location in a serial memory. When performing read operations with an internal address, the TWIHS performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second START condition (after sending the IADR) is sometimes called “repeated start” (Sr) in I^2C fully-compatible devices. See Figure 43-11.

See Figure 43-10 and Figure 43-12 for the master write operation with internal address.

The three internal address bytes are configurable through TWIHS_MMR.

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be set to 0.

Table 43-6 shows the abbreviations used in Figure 43-10 and Figure 43-11.

Table 43-6. Abbreviations

Abbreviation	Definition
S	Start
Sr	Repeated Start
P	Stop
W	Write
R	Read
A	Acknowledge
NA	Not Acknowledge
DADR	Device Address
IADR	Internal Address

Figure 43-10. Master Write with One-, Two- or Three-Byte Internal Address and One Data Byte

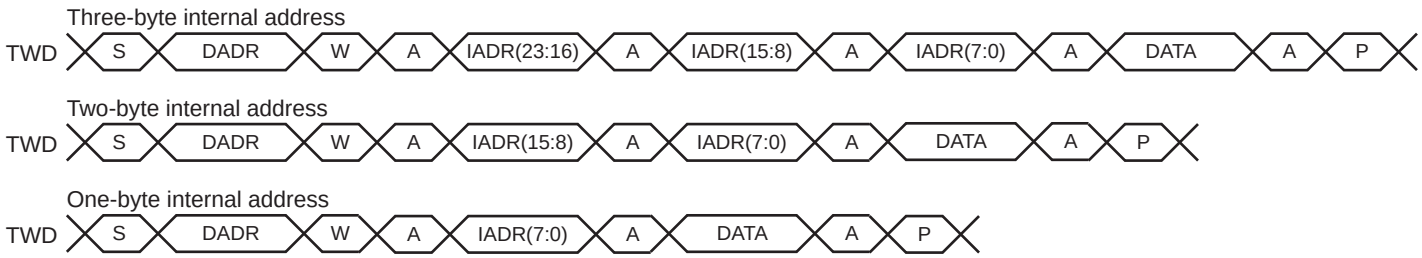
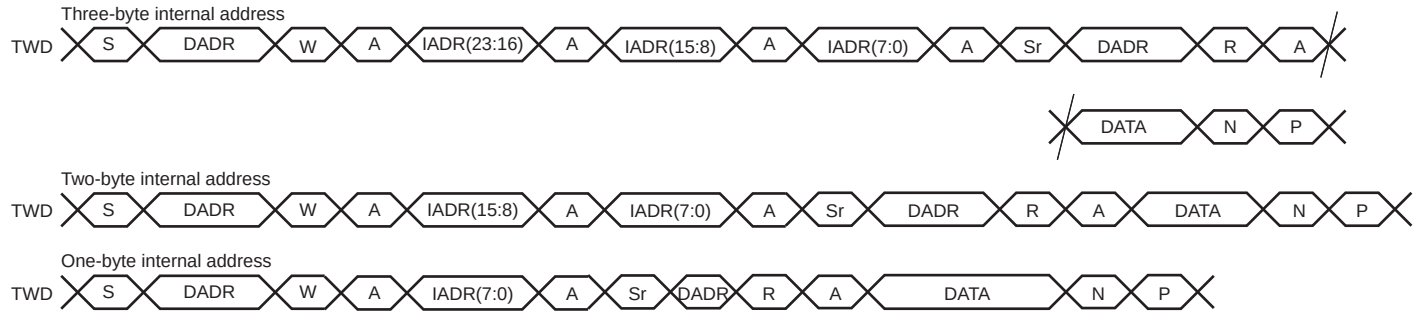


Figure 43-11. Master Read with One-, Two- or Three-Byte Internal Address and One Data Byte



10-bit Slave Addressing

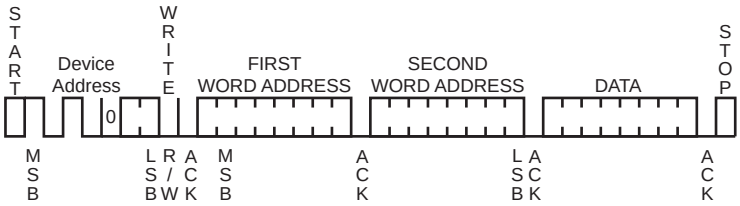
For a slave address higher than seven bits, configure the address size (IADRSZ) and set the other slave address bits in the Internal Address register (TWIHS_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16], can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1,
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program TWIHS_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

Figure 43-12 shows a byte write to a memory device. This demonstrates the use of internal addresses to access the device.

Figure 43-12. Internal Address Usage



43.6.3.7 Repeated Start

In addition to Internal Address mode, REPEATED START (Sr) can be generated manually by writing the START bit at the end of a transfer instead of the STOP bit. In such case, the parameters of the next transfer (direction, SADR, etc.) need to be set before writing the START bit at the end of the previous transfer.

See Section 43.6.3.14 “Read/Write Flowcharts” for detailed flowcharts.

Note that generating a REPEATED START after a single data read is not supported.

43.6.3.8 Bus Clear Command

The TWIHS can perform a Bus Clear command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Start the transfer by setting the CLEAR bit in the TWIHS_CR.

Note: If alternative command is used (ACMEN bit set to ‘1’) DATAL field must be set to 0.

43.6.3.9 Using the DMA Controller (DMAC) in Master Mode

The use of the DMA significantly reduces the CPU load.

To ensure correct implementation, follow the programming sequences below:

Data Transmit with the DMA in Master Mode

If Alternative Command mode is disabled (ACMEN bit set to ‘0’):

The DMA transfer size must be defined with the buffer size minus 1. The remaining character must be managed without DMA to ensure that the exact number of bytes are transmitted regardless of system bus latency conditions during the end of the buffer transfer period.

1. Initialize the DMA (channels, memory pointers, size - 1, etc.);
2. Configure the Master mode (DADR, CKDIV, MREAD = 0, etc.) or Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. Wait for the TXRDY flag in TWIHS_SR.
7. Set the STOP bit in TWIHS_CR.
8. Write the last character in TWIHS_THR.
9. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

If Alternative Command mode is enabled (ACMEN bit set to ‘1’):

1. Initialize the transmit DMA (memory pointers, transfer size).
2. Configure the Master mode (DADR, CKDIV, etc.) and TWIHS_ACR.
3. Start the transfer by setting the DMA TXTEN bit.
4. Wait for the DMA ENDTX flag either by using the polling method or ENDTX interrupt.
5. Disable the DMA by setting the DMA TXTDIS bit.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

Data Receive with the DMA in Master Mode

If Alternative Command mode is disabled (ACMEN bit set to ‘0’):

The DMA transfer size must be defined with the buffer size minus 2. The two remaining characters must be managed without DMA to ensure that the exact number of bytes are received regardless of system bus latency conditions encountered during the end of buffer transfer period.

1. Initialize the DMA (channels, memory pointers, size - 2, etc.);
2. Configure the Master mode (DADR, CKDIV, MREAD = 1, etc.) or Slave mode.
3. Enable the DMA.
4. (Master Only) Write the START bit in the TWIHS_CR to start the transfer.
5. Wait for the DMA status flag indicating that the buffer transfer is complete.
6. Disable the DMA.
7. Wait for the RXRDY flag in the TWIHS_SR.
8. Set the STOP bit in TWIHS_CR.
9. Read the penultimate character in TWIHS_RHR.
10. Wait for the RXRDY flag in the TWIHS_SR.
11. Read the last character in TWIHS_RHR.
12. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

If Alternative Command mode is enabled (ACMEN bit set to ‘1’):

1. Initialize the transmit DMA (memory pointers, transfer size).
2. Configure the Master mode (DADR, CKDIV, etc.) and TWIHS_ACR.
3. Set the DMA RXTEN bit.
4. (Master Only) Write the START bit in the TWIHS_CR to start the transfer.
5. Wait for the DMA ENDTX Flag either by using the polling method or ENDTX interrupt.
6. Disable the DMA by setting the DMA TXTDIS bit.
7. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

43.6.3.10 SMBus Mode

SMBus mode is enabled when a one is written to the SMEN bit in the TWIHS_CR. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into TWIHS_SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses has been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring the TWIHS_CR.

Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to the PECEN bit in TWIHS_CR enables automatic PEC handling in the current transfer. Transfers with and without PEC can be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers is correct.

In Master Transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave compares it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave returns an ACK to the master. If the PEC values differ, data was corrupted, and the slave returns a NACK value. Some slaves may not be able to check the

received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and another method must be used to verify that the transmission was received correctly.

In Master Receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master compares it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the PECERR bit in TWIHS_SR is set. In Master Receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

In combined transfers, the PECRQ bit should only be set in the last of the combined transfers.
If the Alternative Command mode is enabled, only the NPEC bit should be set.

Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

See Section 43.6.3.14 “Read/Write Flowcharts” for detailed flowcharts.

Timeouts

The TLOWS and TLOWM fields in TWIHS_SMBTR configure the SMBus timeout values. If a timeout occurs, the master transmits a STOP condition and leaves the bus. The TOUT bit is also set in TWIHS_SR.

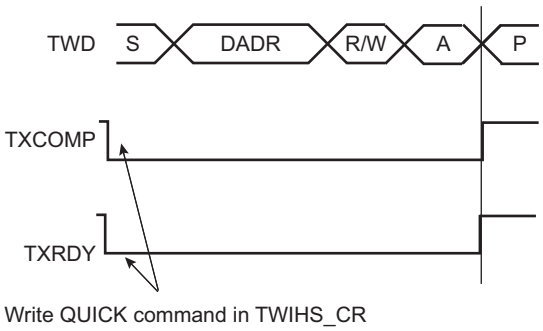
43.6.3.11 SMBus Quick Command (Master Mode Only)

The TWIHS can perform a quick command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Write the MREAD bit in the TWIHS_MMR at the value of the one-bit command to be sent.
3. Start the transfer by setting the QUICK bit in the TWIHS_CR.

Note: If alternative command is used (ACMEN bit set to ‘1’) DATAL field must be set to 0.

Figure 43-13. SMBus Quick Command



43.6.3.12 Alternative Command

Another way to configure the transfer is to enable the Alternative Command mode with the ACMEN bit of the TWIHS Control Register.

In this mode, the transfer is configured through the TWIHS Alternative Command Register. It is possible to define a simple read or write transfer or a combined transfer with a repeated start.

In order to set a simple transfer, the DATAL field and the DIR field of the TWIHS Alternative Command Register must be filled accordingly and the NDATAL field must be cleared. To begin the transfer, either set the START bit in the TWIHS Control Register in case of a read transfer, or write the TWIHS Transmit Holding Register in case of a write transfer.

For a combined transfer linked by a repeated start, the NDATAL field must be filled with the length of the second transfer and NDIR with the corresponding direction.

The PEC and NPEC bits are used to set a PEC field. In the case of a single transfer with PEC, the PEC bit must be set. In the case of a combined transfer, the NPEC bit must be set.

Note: If Alternative Command mode is used, IADRSZ in TWIHS_MMR must be set to 0.

See Section 43.6.3.14 “Read/Write Flowcharts” for detailed flowcharts.

43.6.3.13 Handling Errors in Alternative Command

If a NACK is generated by a slave device or SMBus timeout error, the TWIHS stops the frame immediately, although the DMA transfer may still be active. To prevent a new frame from being restarted with the remaining DMA data (transmit), the TWIHS prevents any start of frame until the LOCK flag is cleared in the TWIHS_SR.

The LOCK bit in the TWIHS_SR indicates the state of the TWIHS (locked or not locked).

When the TWIHS is locked, no transfer begins until the LOCK is cleared by the LOCKCLR bit in the TWIHS_CR and error flags are cleared by reading the TWIHS_SR.

In case of error, the TWIHS_THR may have been loaded with a new data. The THRCLR bit in the TWIHS_CR can be used to flush the TWIHS_THR. If the THRCLR bit is set, the TXRDY and TXCOMP flags are set.

43.6.3.14 Read/Write Flowcharts

The flowcharts give examples for read and write operations. A polling or interrupt method can be used to check the status bits. The interrupt method requires that TWIHS_IER be configured first.

Figure 43-14. TWIHS Write Operation with Single Data Byte without Internal Address

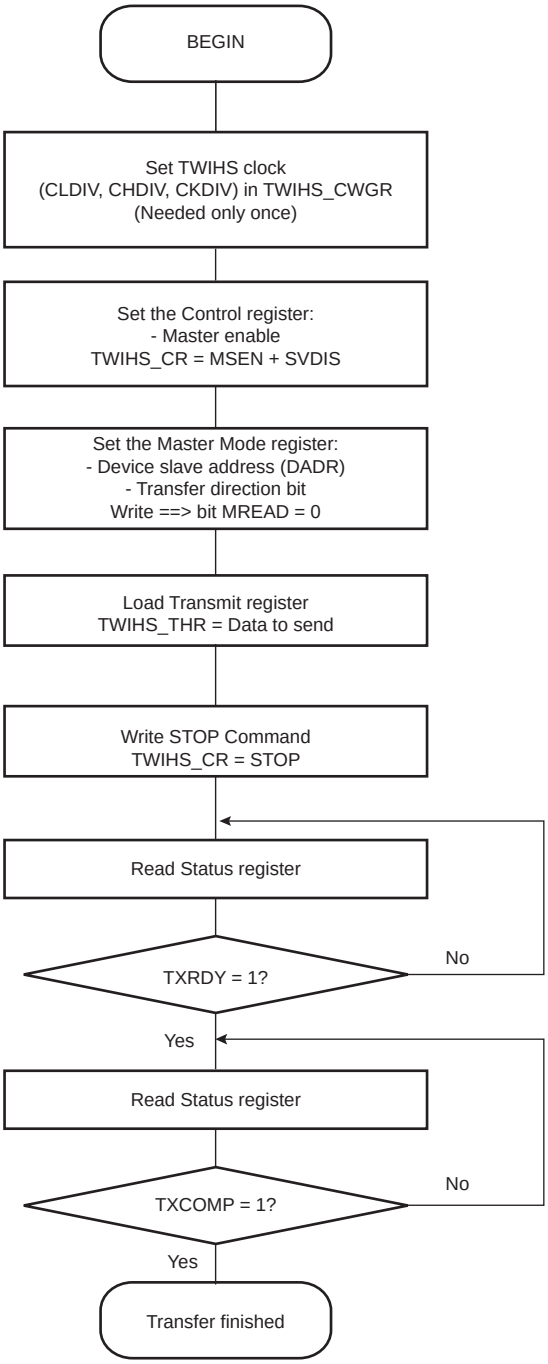


Figure 43-15. TWIHS Write Operation with Single Data Byte and Internal Address

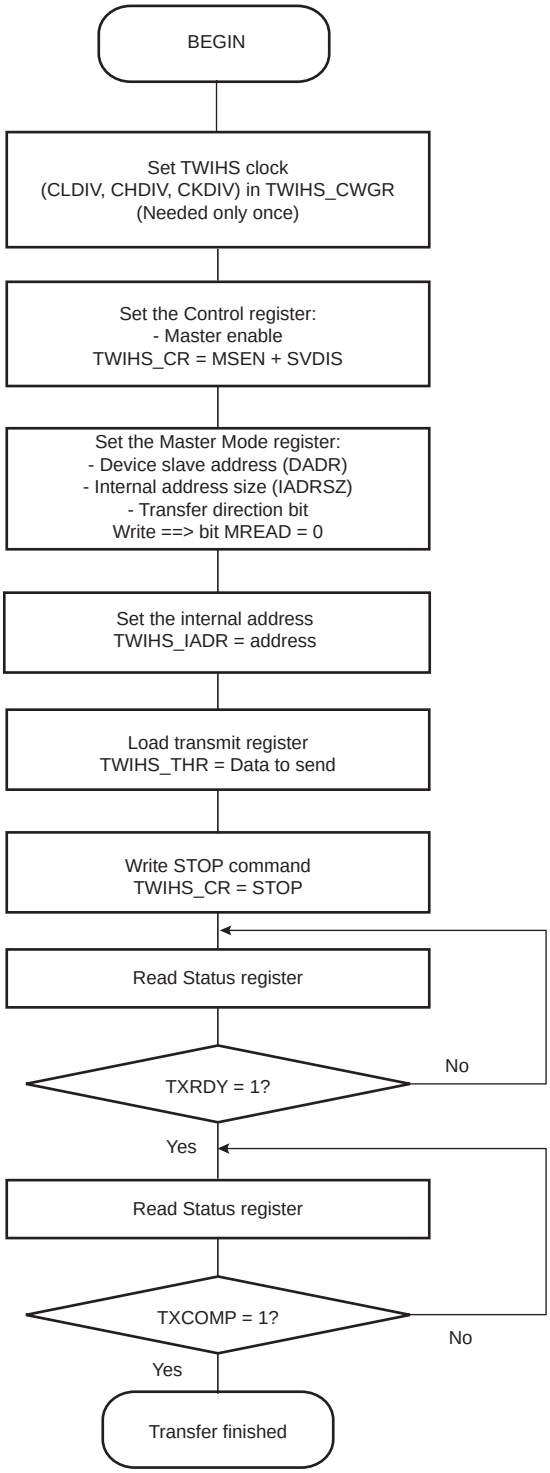


Figure 43-16. TWIHS Write Operation with Multiple Data Bytes with or without Internal Address

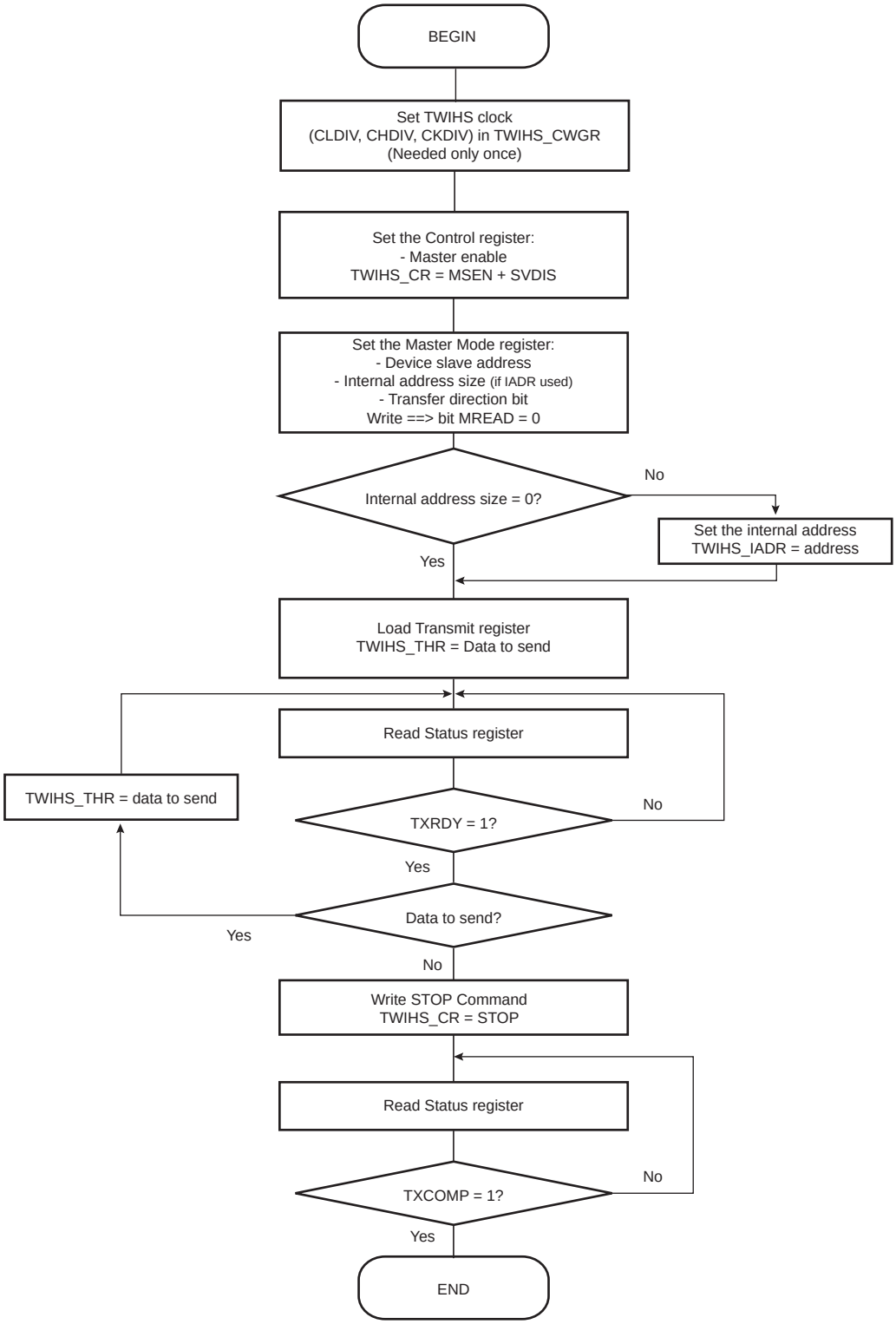


Figure 43-17. SMBus Write Operation with Multiple Data Bytes with or without Internal Address and PEC Sending

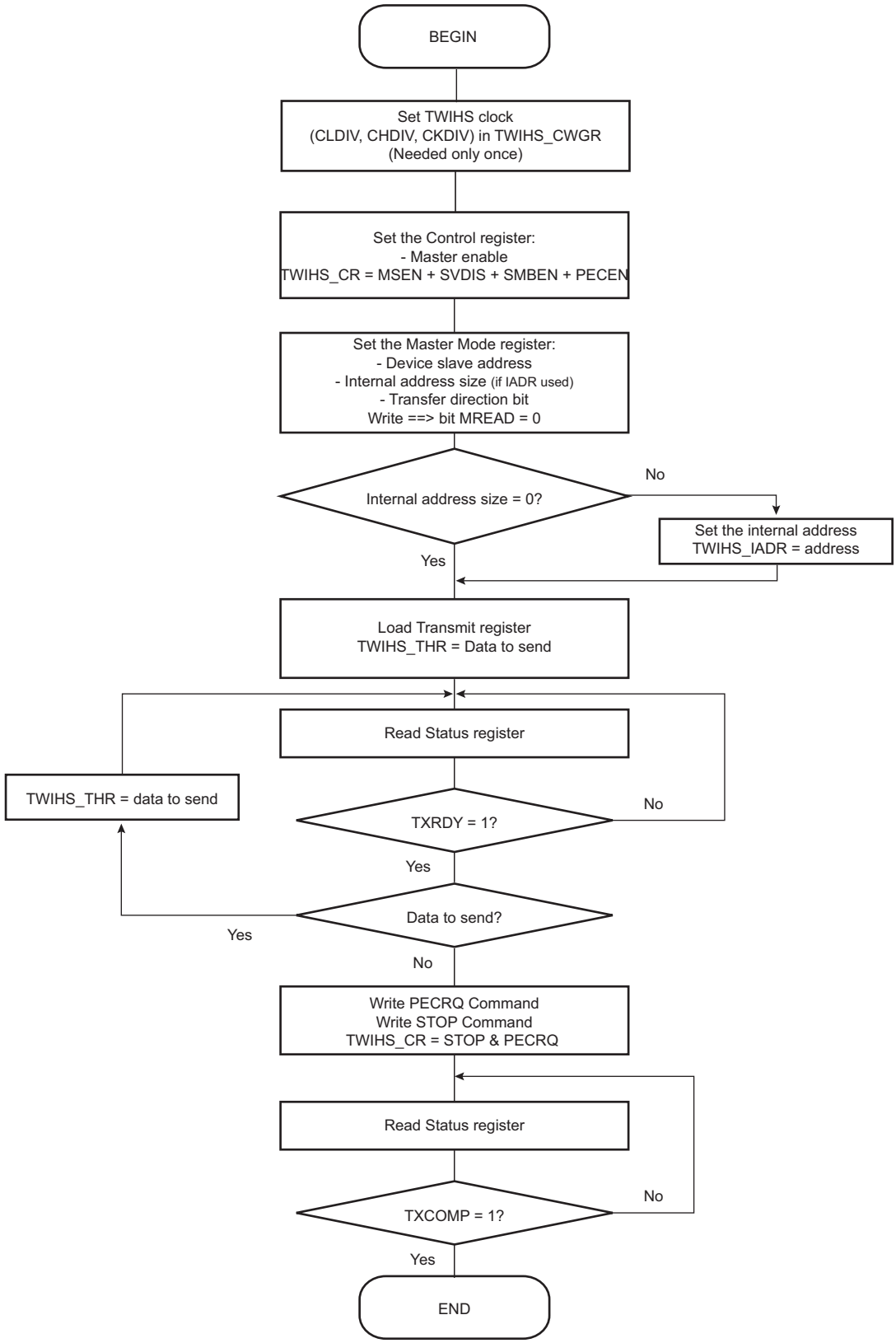


Figure 43-18. SMBus Write Operation with Multiple Data Bytes with PEC and Alternative Command Mode

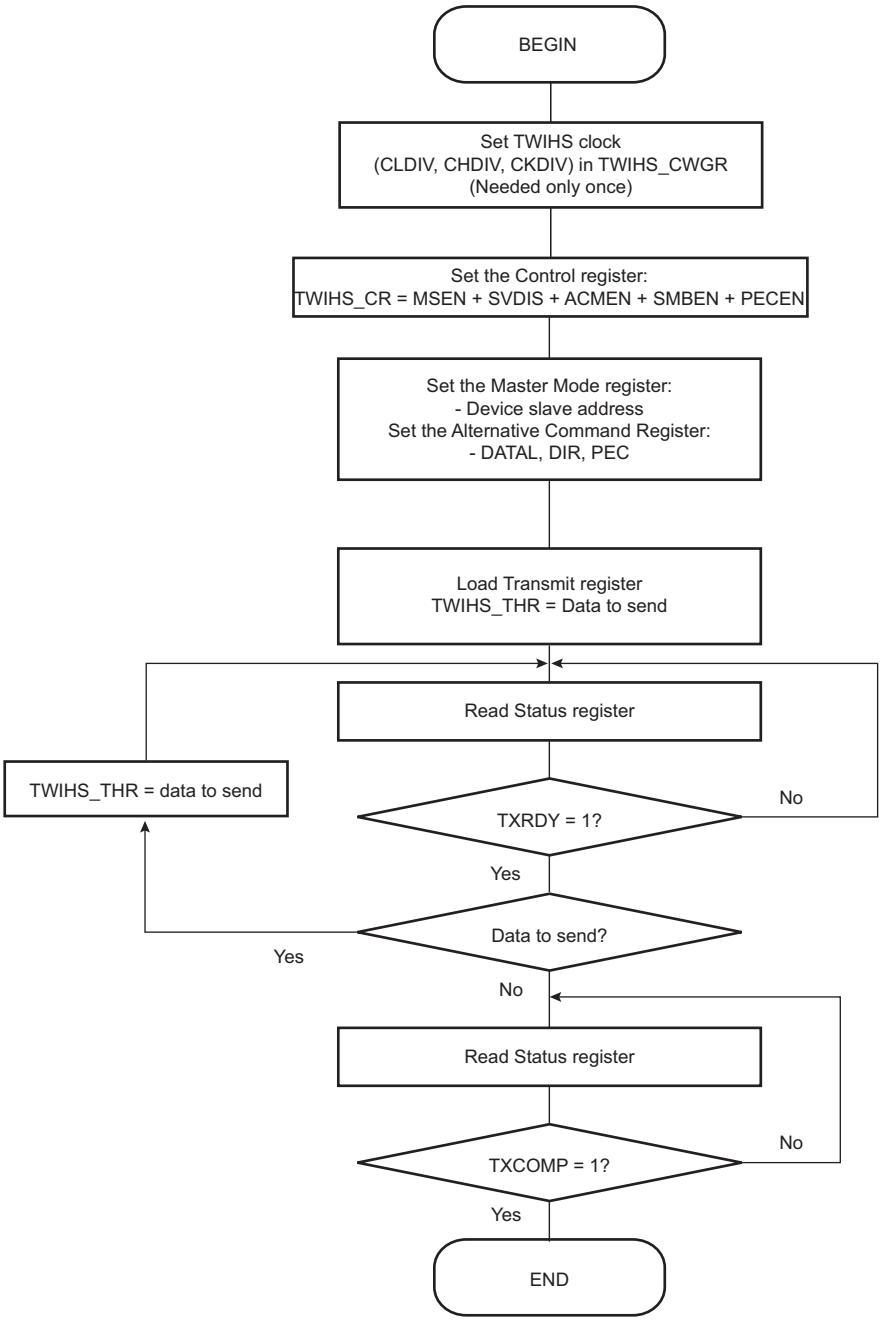


Figure 43-19. TWIHS Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)

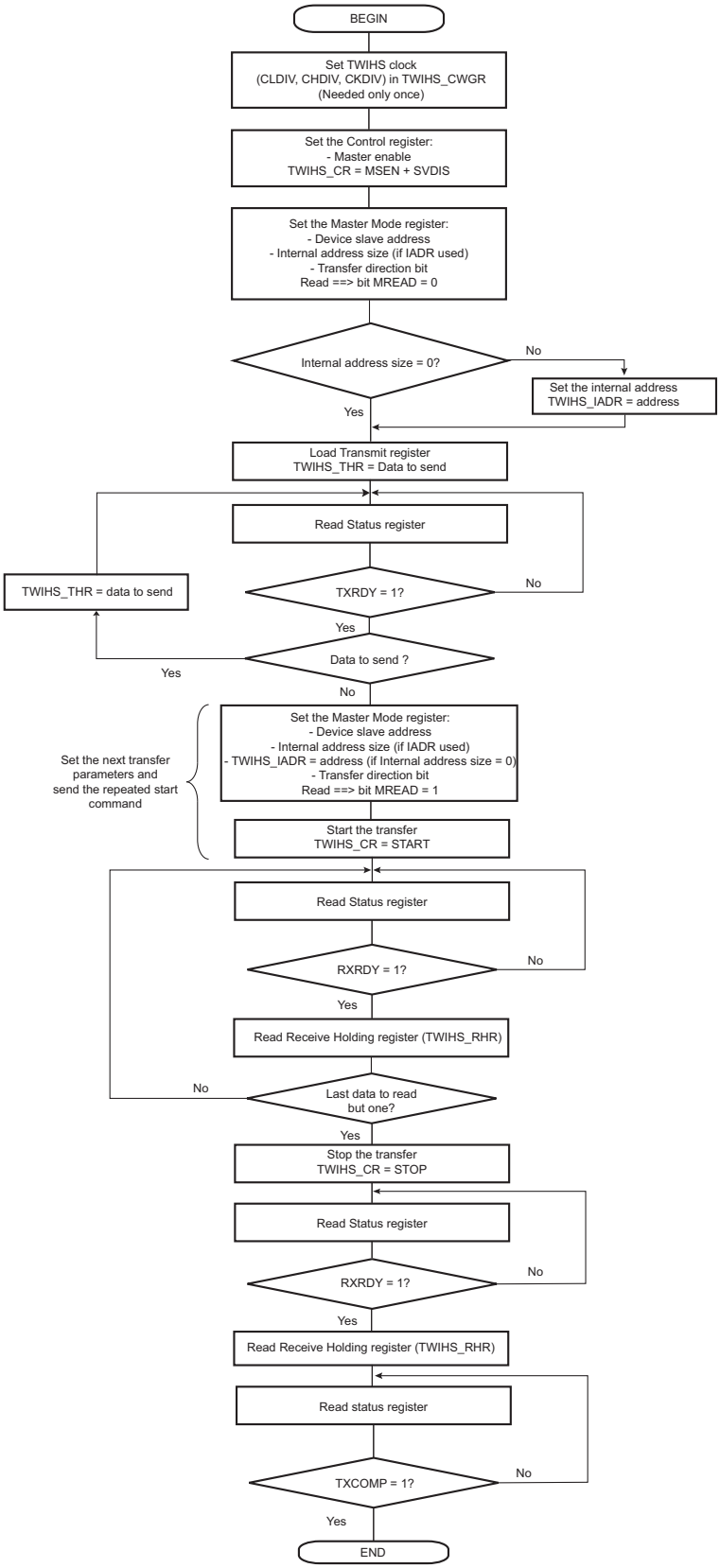


Figure 43-20. TWIHS Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC

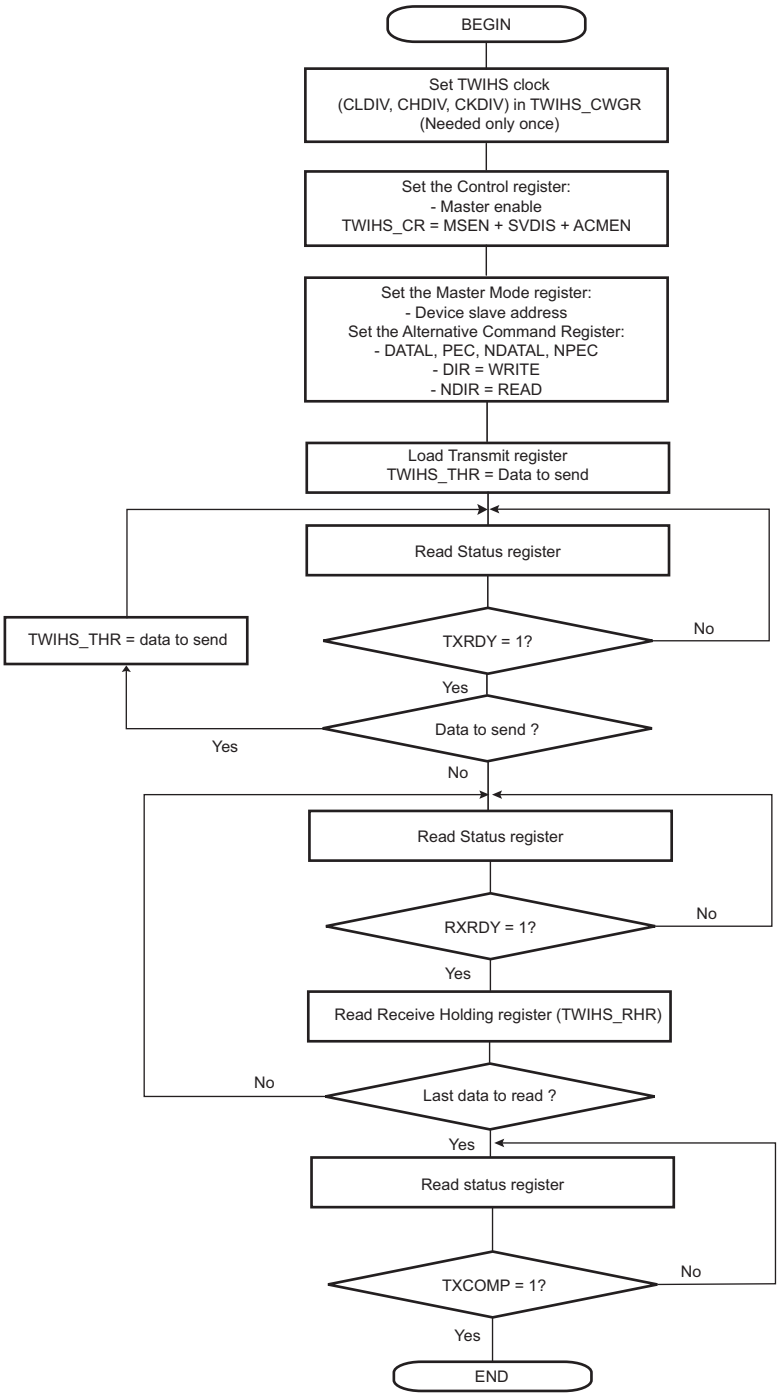


Figure 43-21. TWIHS Read Operation with Single Data Byte without Internal Address

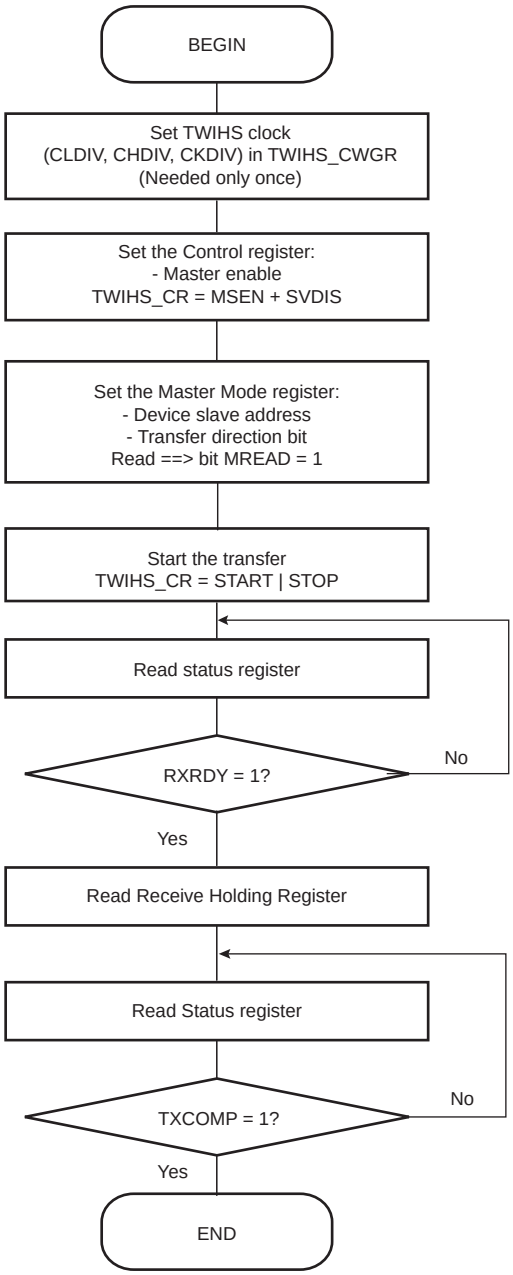


Figure 43-22. TWIHS Read Operation with Single Data Byte and Internal Address

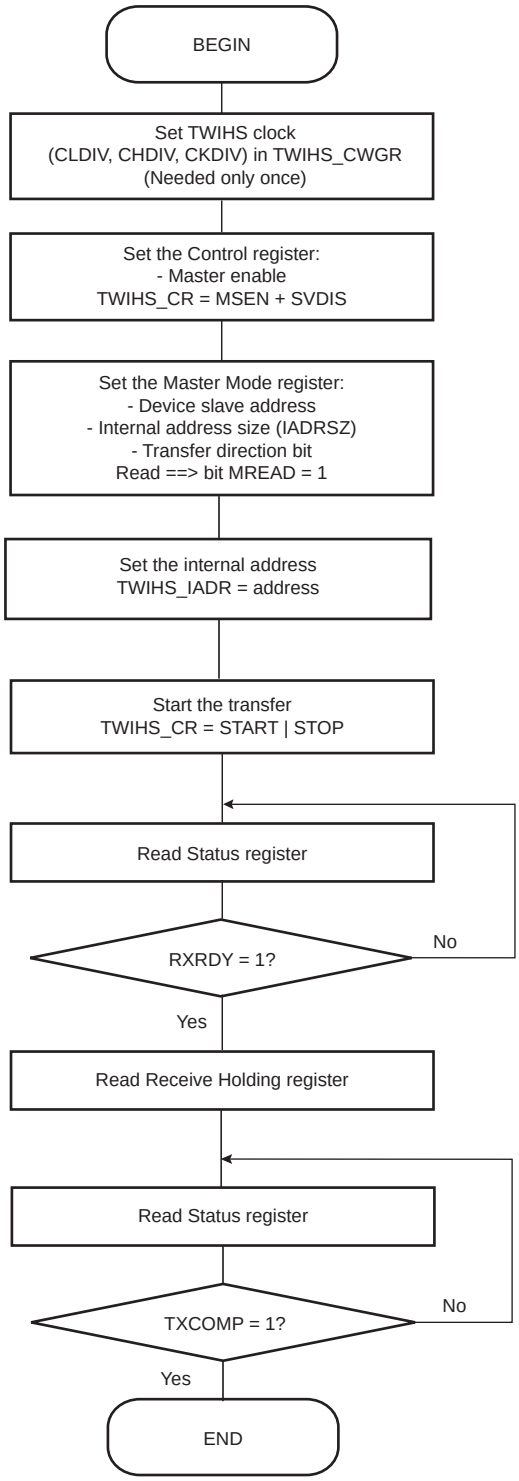


Figure 43-23. TWIHS Read Operation with Multiple Data Bytes with or without Internal Address

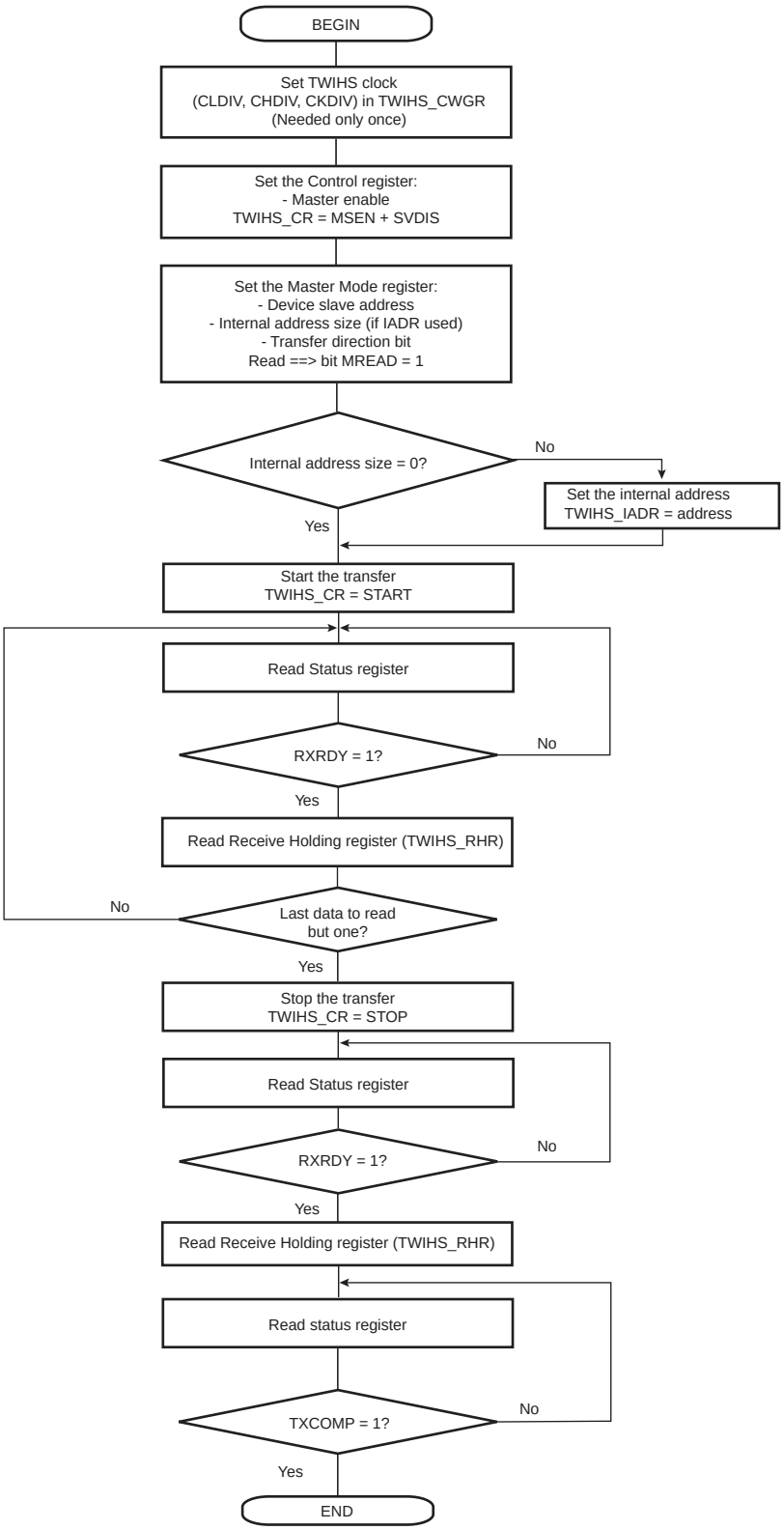


Figure 43-24. TWIHS Read Operation with Multiple Data Bytes with or without Internal Address with PEC

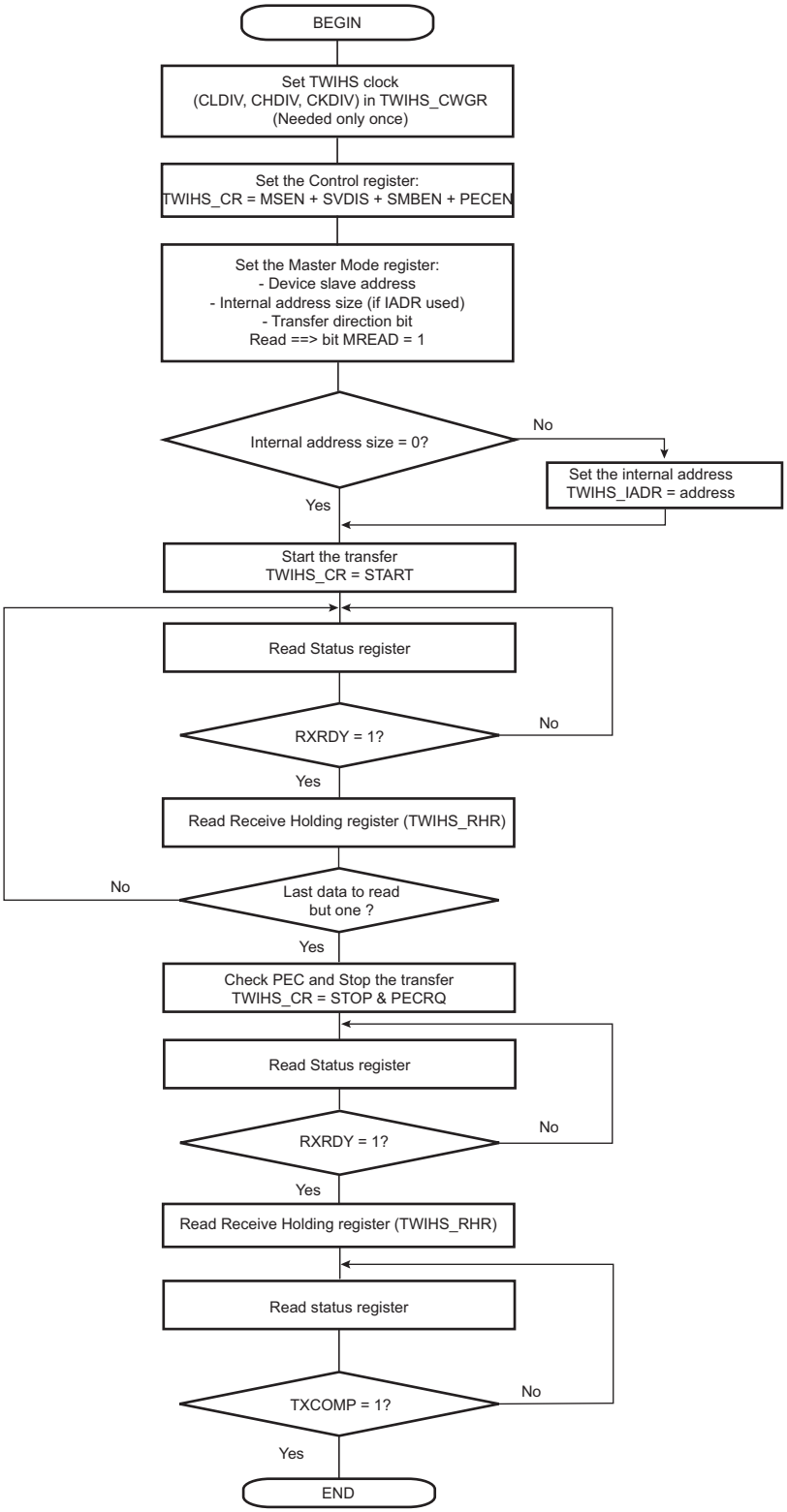


Figure 43-25. TWIHS Read Operation with Multiple Data Bytes with Alternative Command Mode with PEC

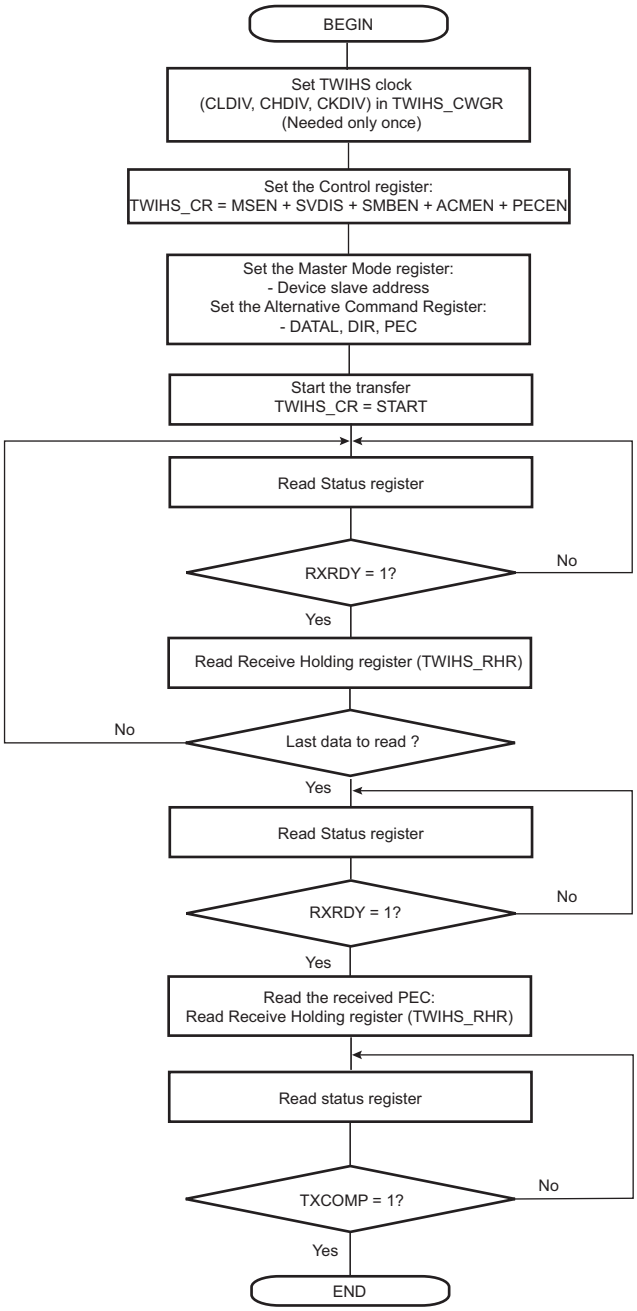


Figure 43-26. TWIHS Read Operation with Multiple Data Bytes + Write Operation with Multiple Data Bytes (Sr)

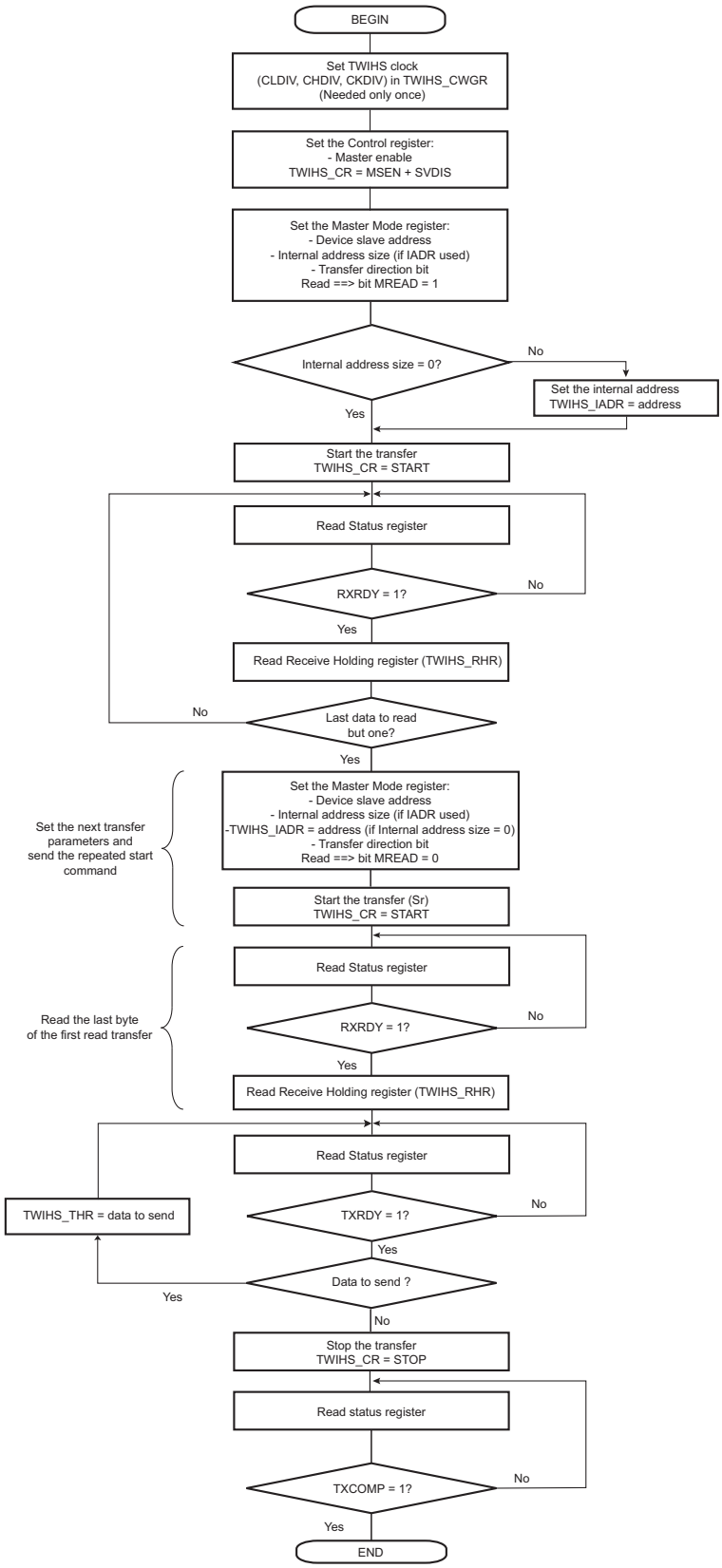
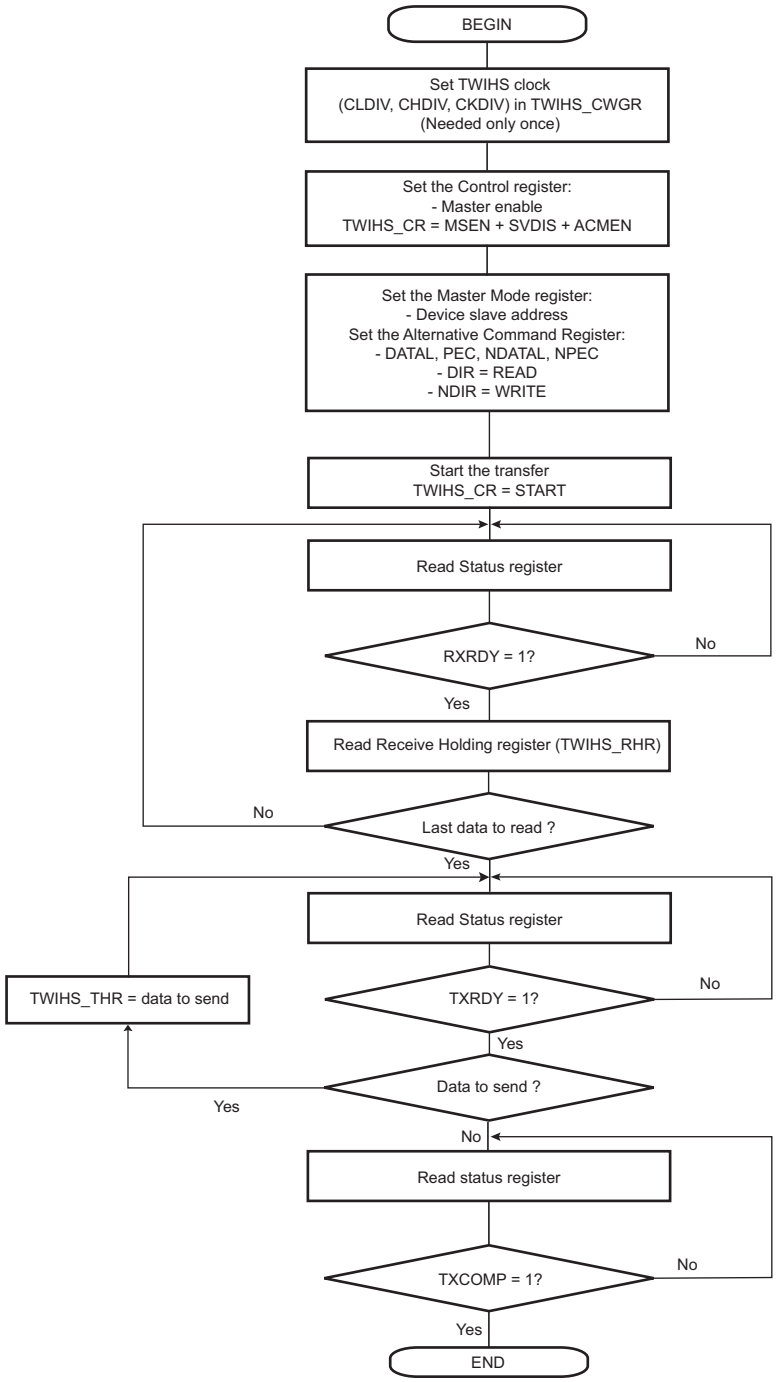


Figure 43-27. TWIHS Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC

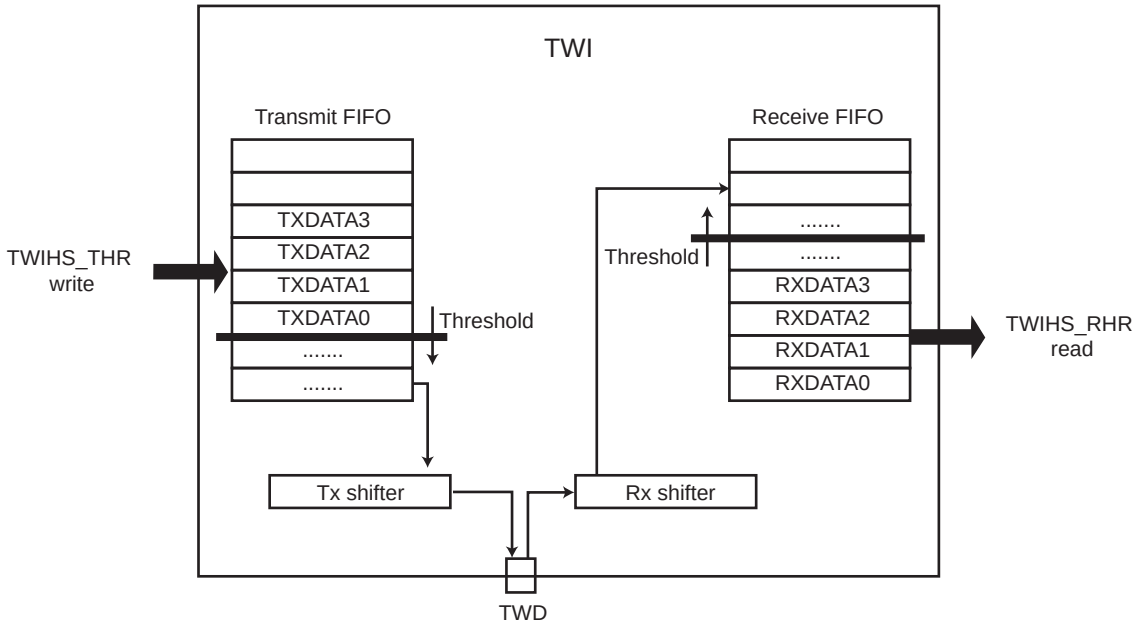


43.6.3.15 FIFOs

The TWIHS includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in the TWIHS_CR. It is recommended to disable both Master and Slave modes before enabling or disabling the FIFO (MSDIS and SVDIS bit in TWIHS_CR).

Writing the FIFOEN bit to ‘1’ will enable a 16-data Transmit FIFO and a 16-data Receive FIFO.

Figure 43-28. FIFOs Block Diagram

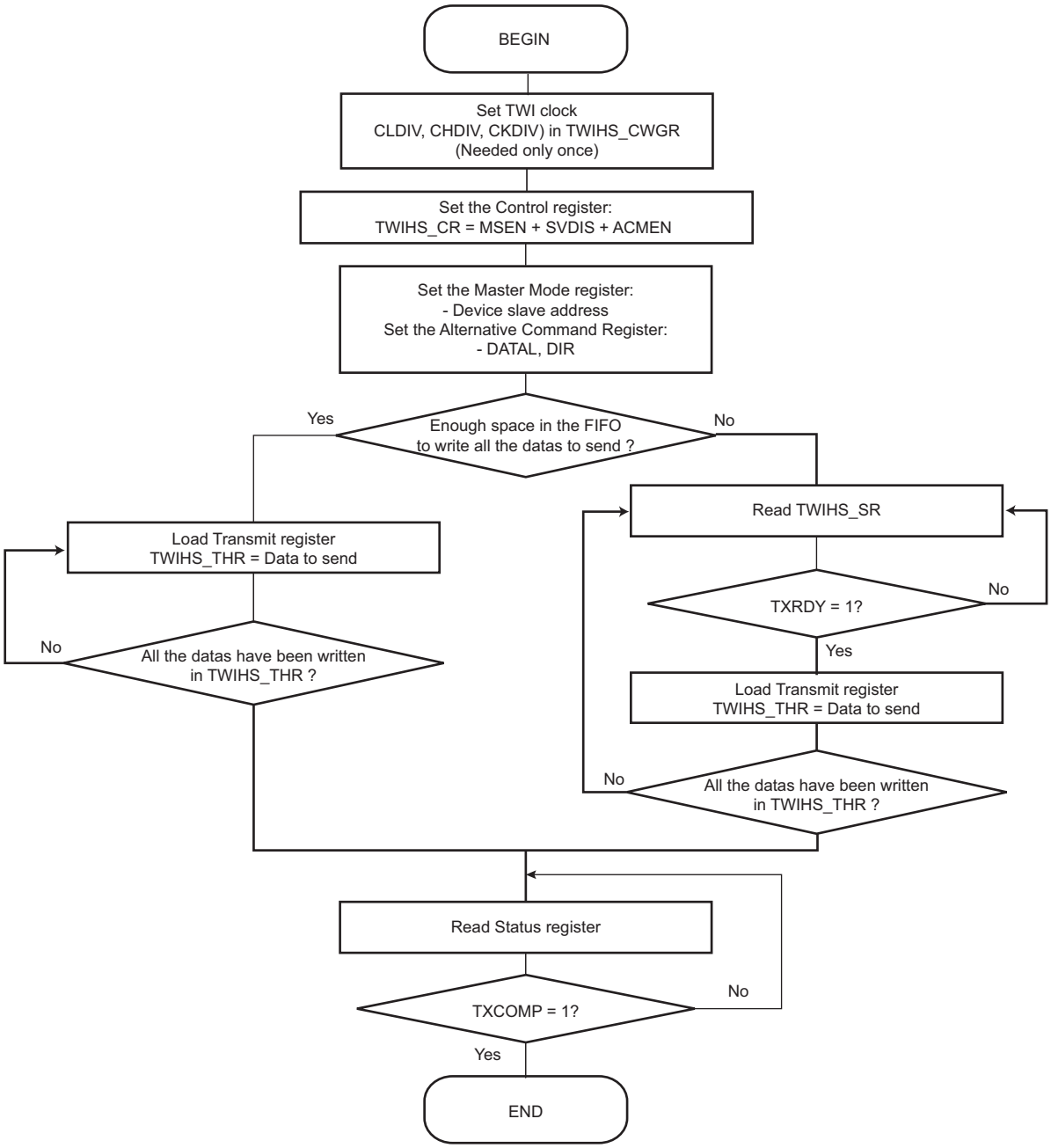


Sending Data with FIFO Enabled

With the Transmit FIFO enabled, any write access to the TWIHS Transmit Holding Register (TWIHS_THR) brings the written data to the Transmit FIFO. As a consequence, it is not mandatory any more to monitor the TXRDY flag state to send multiple data without DMAC.

Knowing the number of data to send and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in the TWIHS_THR without checking the TXRDY flag between each access. The Transmit FIFO state can be checked reading the TXFL field in the TWIHS FIFO Level Register (TWIHS_FLR).

Figure 43-29. Sending Data with FIFO Flowchart

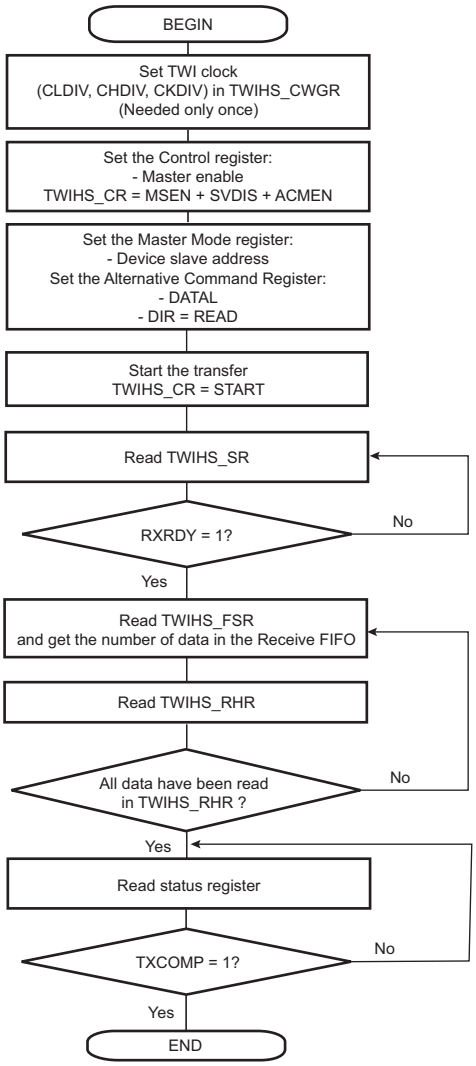


Receiving Data with FIFO Enabled

With Receive FIFO enabled, any read access on TWIHS_RHR will pull out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RXRDY flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with the RXFL field in the TWIHS_FLR and all the data read successively in the TWIHS_RHR without checking the RXRDY flag between each access.

Figure 43-30. Receiving Data with FIFO Flowchart



Clearing/Flushing FIFOs

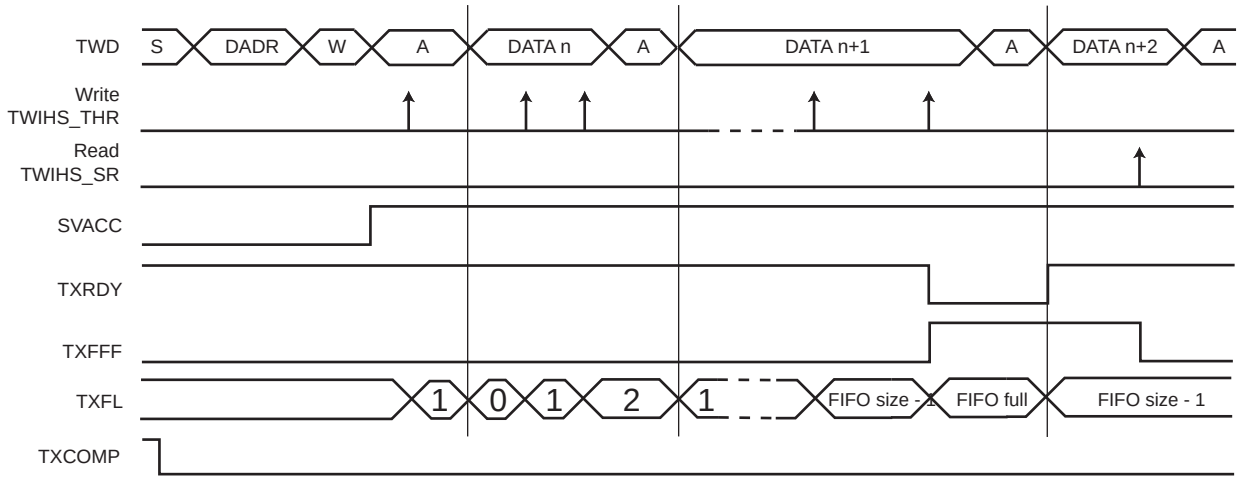
Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in the TWIHS_CR.

TXRDY and RXRDY Behavior

If FIFOs are enabled, the behavior of the TXRDY and RXRDY flags will be slightly different.

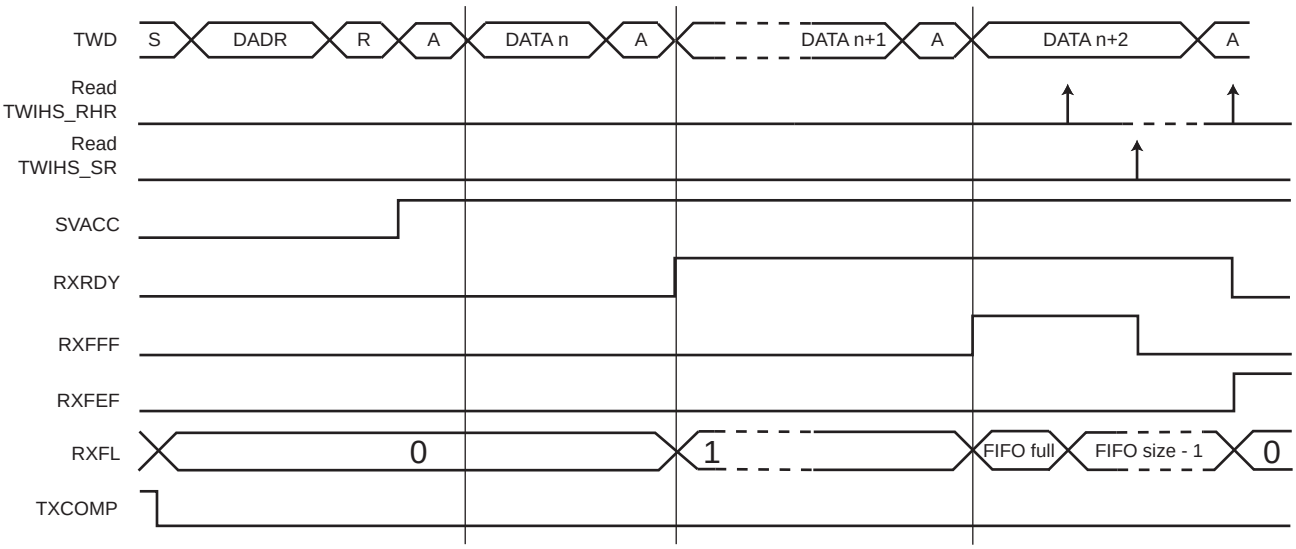
TXRDY will indicate if a data can be written in the Transmit FIFO. By default, the TXRDY flag will then stay at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 43-31. TXRDY in Single Data Mode and TXRDYM = 0x0



RXRDY will indicate if an unread data is present in the Receive FIFO. By default, the RXRDY flag will then be at level ‘1’ as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 43-32. RXRDY in Single Data Mode and RXRDYM = 0x0



TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the TWIHS FIFO Mode Register (TWIHS_FMR). In Single Data mode, there is no need to modify the TXRDY and RXRDY behavior; however, it may be useful in Multiple Data Mode.

Master Multiple Data Mode

When the FIFOs are enabled, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one TWIHS_THR/TWIHS_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte size register access, only one data will be written/read; if the access is a halfword-size register access, then two data will be written/read; finally, if the access is a word-size register access, then four data will be written/read.

Written/Read data are always right-aligned, as shown in Section 43.7.14 ”TWIHS Receive Holding Register (FIFO Enabled)” and Section 43.7.17 ”TWIHS Transmit Holding Register (FIFO Enabled)”.

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six TWIHS_THR-byte write accesses.
- Perform three TWIHS_THR-halfword write accesses.
- Perform one TWIHS_THR-word write access and one TWIHS_THR halfword write access.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six TWIHS_RHR-byte read accesses.
- Perform three TWIHS_RHR-halfword read accesses.
- Perform one TWIHS_RHR-word read access and one TWIHS_RHR-halfword read access.

This mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

- TXRDY and RXRDY Configuration

  In Multiple Data mode, the TXRDYM and RXRDYM fields in the TWIHS_FMR become useful.

  As in Multiple Data mode, it is possible to write several data in the same access it might be useful to configure TXRDY flag behavior to indicate if 1, 2 or 4 data can be written in the FIFO depending on the access to perform on TWIHS_THR.

  If for instance four data are written each time in the TWIHS_THR it might be useful to configure TXRDYM field to 0x2 value so that TXRDY flag will be at ‘1’ only when at least four data can be written in the Transmit FIFO.

  In the same way if four data are read each time in the TWIHS_RHR it might be useful to configure RXRDYM field to 0x2 value so that RXRDY flag will be at ‘1’ only when at least four unread data are in the Receive FIFO.

- DMAC

  If DMAC transfer is used it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, reset DMAC channels, etc., without any risk.

The LOCK bit in the TWIHS_SR is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared setting the TXFLCLR bit to ‘1’ in the TWIHS_CR.

FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

If the Transmit FIFO is full and a write access is performed on the TWIHS_THR, it will generate a Transmit FIFO pointer error and set the TXFPTEF flag in TWIHS_FSR.

In Multiple Data mode, if the number of data written in the TWIHS_THR (according to the register access size) is bigger than the Transmit FIFO free space, it will generate a Transmit FIFO pointer error and set the TXFPTEF flag in TWIHS_FSR.

- Receive FIFO:

In Multiple Data mode, if the number of data read in the TWIHS_RHR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it will generate a Receive FIFO pointer error and set the RXFPTEF flag in TWIHS_FSR.

Pointer error should not happen if FIFO state is checked before writing/reading in TWIHS_THR/TWIHS_RHR. FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using the SWRST bit in the TWIHS_CR. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent afterwards might not be received properly by the slave).

FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO:

The Transmit FIFO threshold can be set using the TXFTHRES field in TWIHS_FMR. Each time the Transmit FIFO goes from the ‘above threshold’ to the ‘equal or below threshold’ state, the TXFTHF flag in TWIHS_FSR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.

- Receive FIFO:

The Receive FIFO threshold can be set using the RXFTHRES field in TWIHS_FMR. Each time the Receive FIFO goes from the ‘below threshold’ to the ‘equal or above threshold’ state, the RXFTHF flag in TWIHS_FSR is set. This enables the application to know that the Receive FIFO reached the defined threshold and to read some data before it becomes full.

The TXFTHF and RXFTHF flags can be both configured to generate an interrupt using TWIHS_FIER and TWIHS_FIDR.

FIFO Flags

FIFOs come with a set of flags which can be configured each to generate an interrupt through TWIHS_FIER and TWIHS_FIDR.

FIFO flags state can be read in TWIHS_FSR. They are cleared on TWIHS_FSR read.

43.6.4 Multimaster Mode

43.6.4.1 Definition

In Multimaster mode, more than one master may handle the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a stop. When the stop is detected, the master that has lost arbitration may put its data on the bus by respecting arbitration.

Arbitration is illustrated in Figure 43-34.

43.6.4.2 Different Multimaster Modes

Two Multimaster modes may be distinguished:

1. The TWIHS is considered as a master only and is never addressed.
2. The TWIHS may be either a master or a slave and may be addressed.

Note: Arbitration in supported in both Multimaster modes.

TWIHS as Master Only

In this mode, the TWIHS is considered as a master only (MSEN is always at one) and must be driven like a master with the ARBLST (Arbitration Lost) flag in addition.

If arbitration is lost (ARBLST = 1), the user must reinitiate the data transfer.

If starting a transfer (ex.: DADR + START + W + Write in THR) and if the bus is busy, the TWIHS automatically waits for a STOP condition on the bus to initiate the transfer (see Figure 43-33).

Note: The state of the bus (busy or free) is not indicated in the user interface.

TWIHS as Master or Slave

The automatic reversal from master to slave is not supported in case of a lost arbitration.

Then, in the case where TWIHS may be either a master or a slave, the user must manage the pseudo Multimaster mode described in the steps below:

1. Program the TWIHS in Slave mode (SADR + MSDIS + SVEN) and perform a slave access (if TWIHS is addressed).
2. If the TWIHS has to be set in Master mode, wait until TXCOMP flag is at 1.
3. Program the Master mode (DADR + SVDIS + MSEN) and start the transfer (ex: START + Write in THR).
4. As soon as the Master mode is enabled, the TWIHS scans the bus in order to detect if it is busy or free. When the bus is considered free, the TWIHS initiates the transfer.
5. As soon as the transfer is initiated and until a STOP condition is sent, the arbitration becomes relevant and the user must monitor the ARBLST flag.
6. If the arbitration is lost (ARBLST is set to 1), the user must program the TWIHS in Slave mode in case the master that won the arbitration needs to access the TWIHS.
7. If the TWIHS has to be set in Slave mode, wait until the TXCOMP flag is at 1 and then program the Slave mode.

Note: If the arbitration is lost and the TWIHS is addressed, the TWIHS does not acknowledge, even if it is programmed in Slave mode as soon as ARBLST is set to 1. Then the master must repeat SADR.

Figure 43-33. User Sends Data While the Bus is Busy

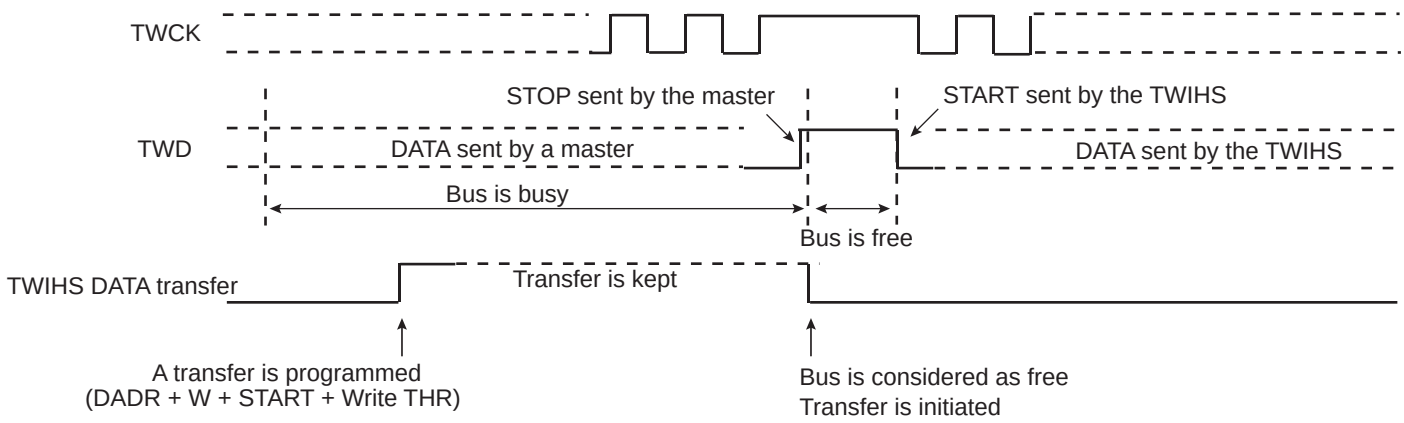
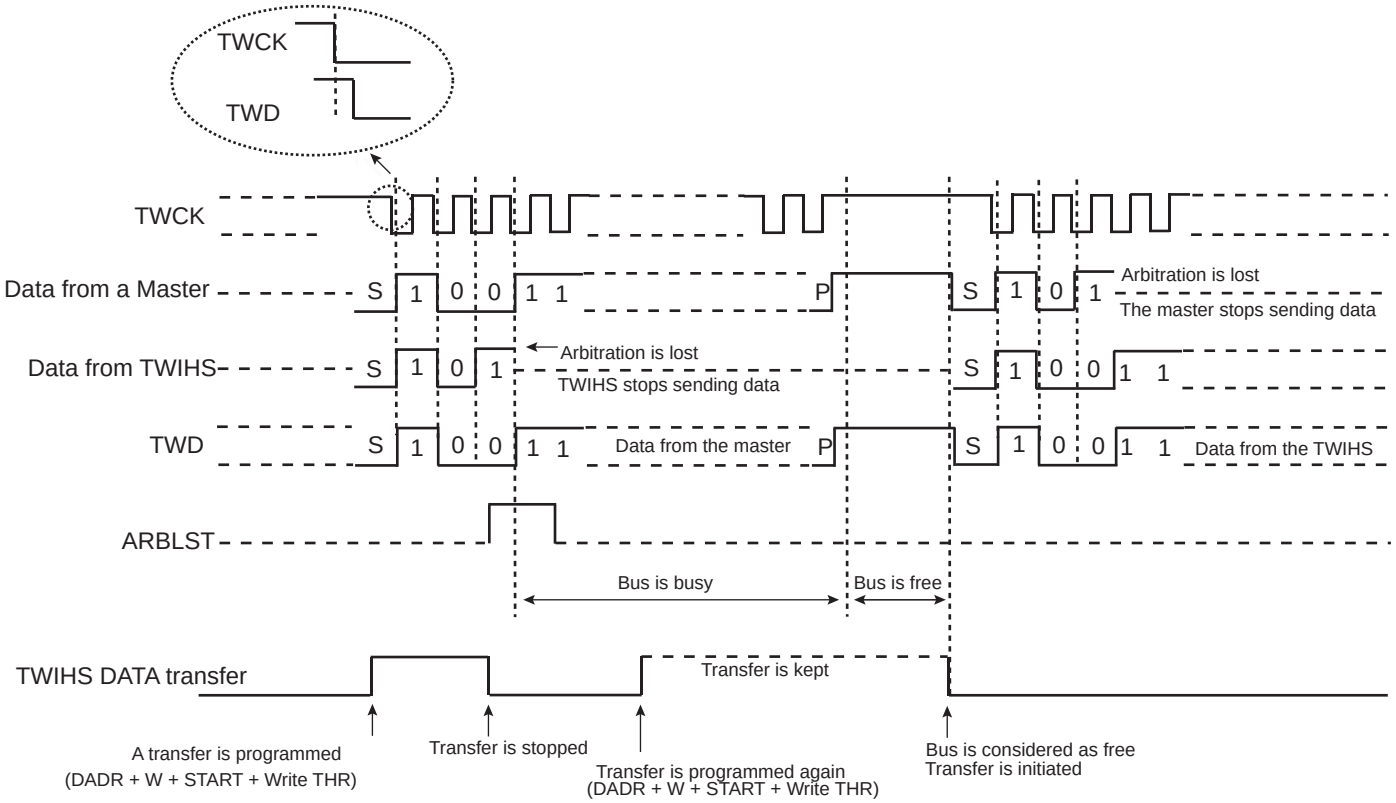
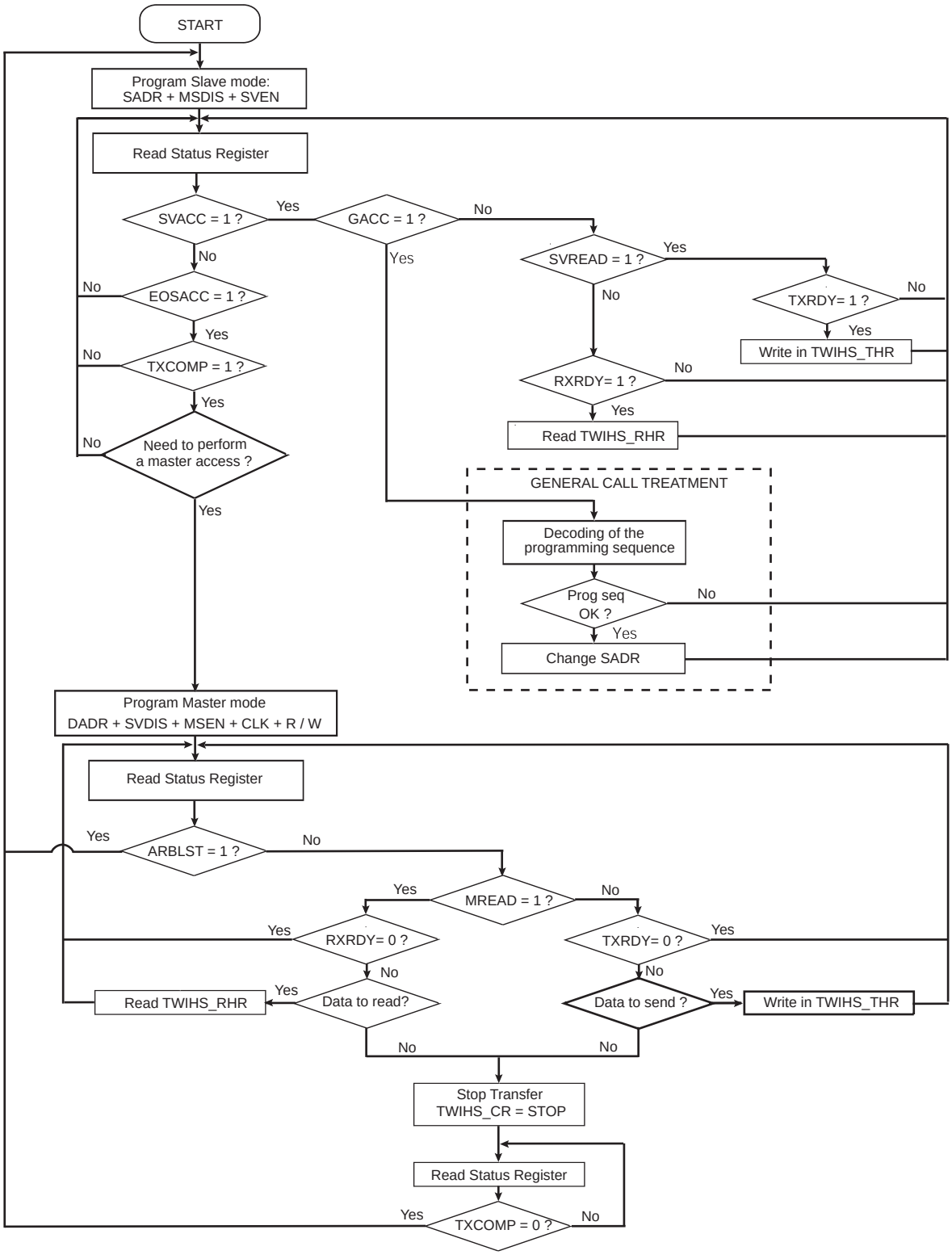


Figure 43-34. Arbitration Cases



The flowchart shown in Figure 43-35 gives an example of read and write operations in Multimaster mode.

Figure 43-35. Multimaster Flowchart



43.6.5 Slave Mode

43.6.5.1 Definition

Slave mode is defined as a mode where the device receives the clock and the address from another device called the master.

In this mode, the device never initiates and never completes the transmission (START, REPEATED_START and STOP conditions are always provided by the master).

43.6.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. TWIHS_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. (Optional) TWIHS_SMR.MASK can be set to mask some SADR address bits and thus allow multiple address matching.
3. TWIHS_CR.MSDIS: Disables the Master mode.
4. TWIHS_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in TWIHS_CWGR are ignored.

43.6.5.3 Receiving Data

After a START or REPEATED START condition is detected, and if the address sent by the master matches the slave address programmed in the SADR (Slave Address) field, the SVACC (Slave Access) flag is set and SVREAD (Slave Read) indicates the direction of the transfer.

SVACC remains high until a STOP condition or a REPEATED START is detected. When such a condition is detected, the EOSACC (End Of Slave Access) flag is set.

Read Sequence

In the case of a read sequence (SVREAD is high), the TWIHS transfers data written in the TWIHS_THR until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC reset.

As soon as data is written in the TWIHS_THR, TXRDY (Transmit Holding Register Ready) flag is reset, and it is set when the internal shifter is empty and the sent data acknowledged or not. If the data is not acknowledged, the NACK flag is set.

Note that a STOP or a REPEATED START always follows a NACK.

To clear the TXRDY flag, first set the bit TWIHS_CR.SVDIS, then set the bit TWIHS_CR.SVEN.

See Figure 43-36.

Write Sequence

In the case of a write sequence (SVREAD is low), the RXRDY (Receive Holding Register Ready) flag is set as soon as a character has been received in the TWIHS_RHR. RXRDY is reset when reading the TWIHS_RHR.

The TWIHS continues receiving data until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the write sequence, the TXCOMP flag is set and SVACC is reset.

See Figure 43-37.

Clock Stretching Sequence

If TWIHS_THR or TWIHS_RHR is not written/read in time, the TWIHS performs a clock stretching.

Clock stretching information is given by the SCLWS (Clock Wait State) bit.

See Figure 43-39 and Figure 43-40.

Note: Clock stretching can be disabled by configuring the SCLWSDIS bit in the TWIHS_SMR. In that case, the UNRE and OVRE flags indicate an underrun (when TWIHS_THR is not filled on time) or an overrun (when TWIHS_RHR is not read on time).

General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, the user must interpret the meaning of the GENERAL CALL and decode the new address programming sequence.

See Figure 43-38.

43.6.5.4 Data Transfer

Read Operation

The Read mode is defined as a data requirement from the master.

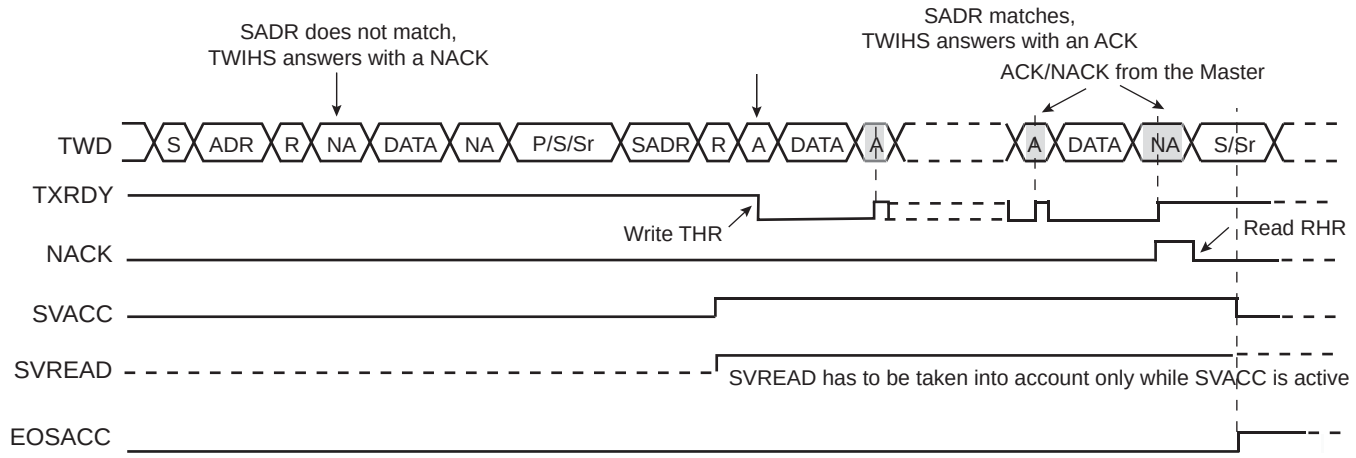
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, the TWIHS continues sending data loaded in the TWIHS_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

Figure 43-36 describes the read operation.

Figure 43-36. Read Access Ordered by a Master



Notes:
1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from TWIHS_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

Write Operation

The Write mode is defined as a data transmission from the master.

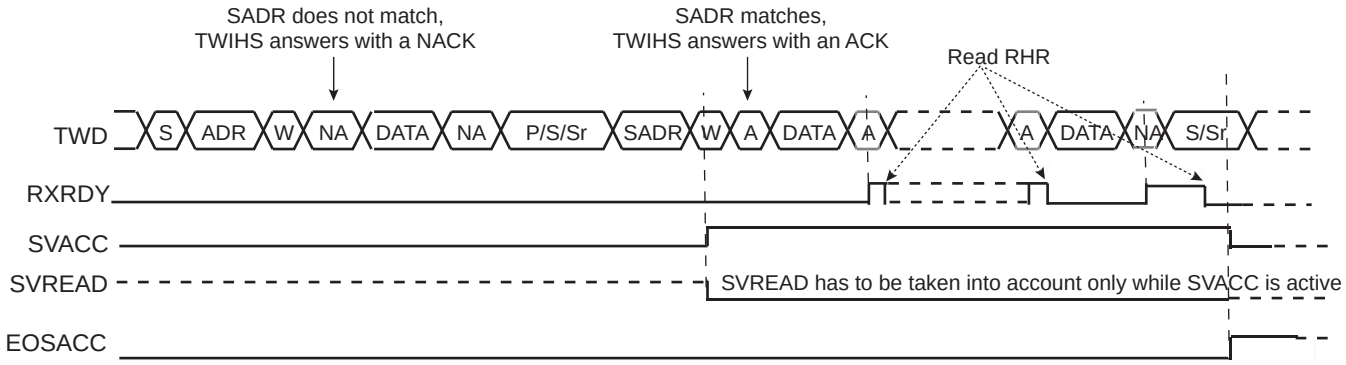
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, the TWIHS stores the received data in the TWIHS_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

Figure 43-37 describes the write operation.

Figure 43-37. Write Access Ordered by a Master



Notes:
1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. RXRDY is set when data has been transmitted from the internal shifter to the TWIHS_RHR and reset when this data is read.

General Call

The general call is performed in order to change the address of the slave.

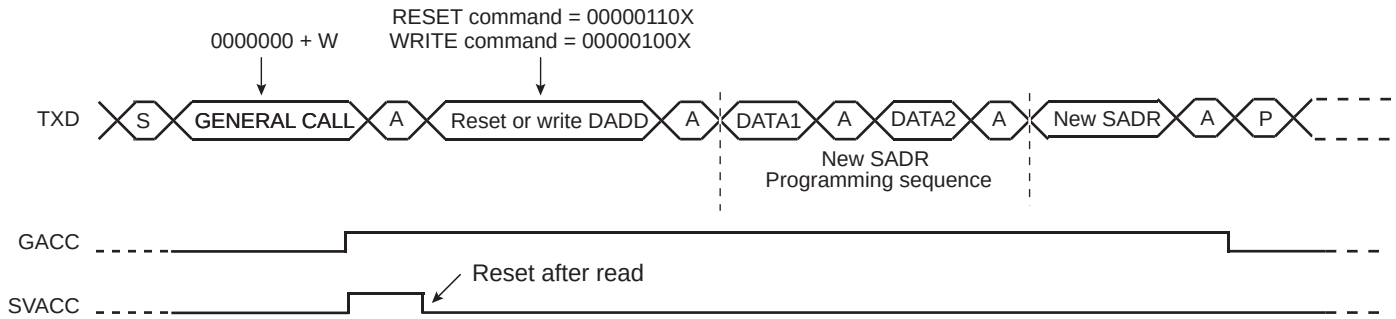
If a GENERAL CALL is detected, GACC is set.

After the detection of general call, decode the commands that follow.

In case of a WRITE command, decode the programming sequence and program a new SADR if the programming sequence matches.

Figure 43-38 describes the general call access.

Figure 43-38. Master Performs a General Call



Note: This method enables the user to create a personal programming sequence by choosing the programming bytes and their number. The programming sequence has to be provided to the master.

Clock Stretching

In both Read and Write modes, it may occur that TWIHS_THR/TWIHS_RHR buffer is not filled/emptied before the transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching mechanism is implemented.

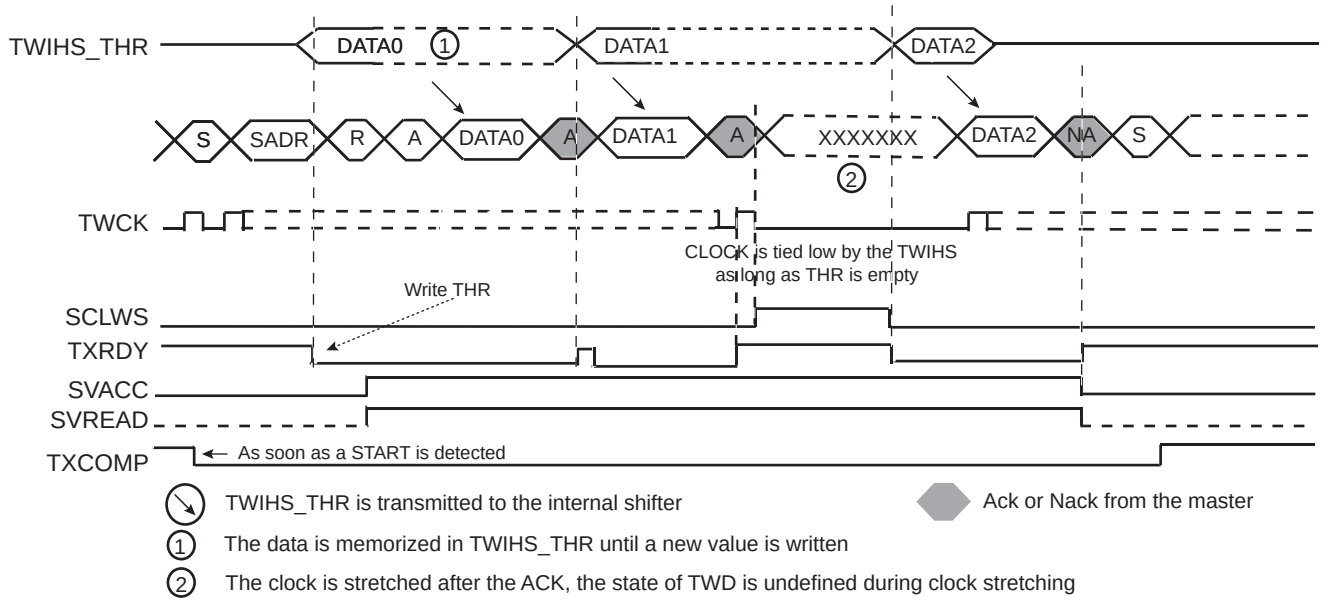
Note: Clock stretching can be disabled by setting the SCLWSDIS bit in the TWIHS_SMR. In that case the UNRE and OVRE flags indicate an underrun (when TWIHS_THR is not filled on time) or an overrun (when TWIHS_RHR is not read on time).

Clock Stretching in Read Mode

The clock is tied low if the internal shifter is empty and if a STOP or REPEATED START condition was not detected. It is tied low until the internal shifter is loaded.

Figure 43-39 describes the clock stretching in Read mode.

Figure 43-39. Clock Stretching in Read Mode



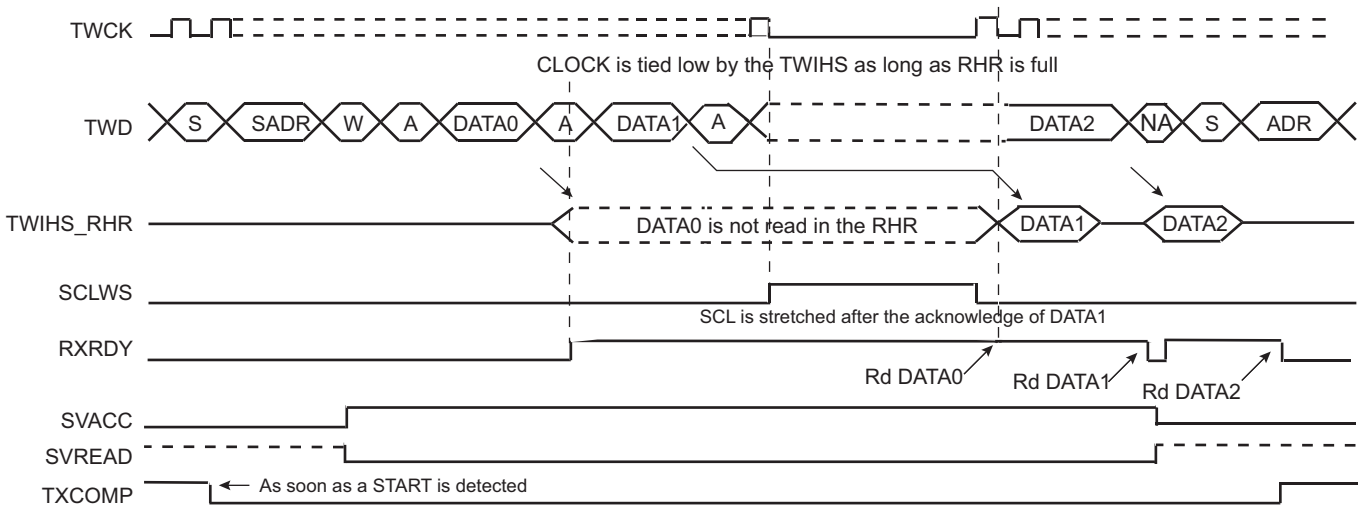
Notes:
1. TXRDY is reset when data has been written in the TWIHS_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

Clock Stretching in Write Mode

The clock is tied low if the internal shifter and the TWIHS_RHR is full. If a STOP or REPEATED_START condition was not detected, it is tied low until TWIHS_RHR is read.

Figure 43-40 describes the clock stretching in Write mode.

Figure 43-40. Clock Stretching in Write Mode



Notes:
1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.

2. SCLWS is automatically set when the clock stretching mechanism is started and automatically reset when the mechanism is finished.

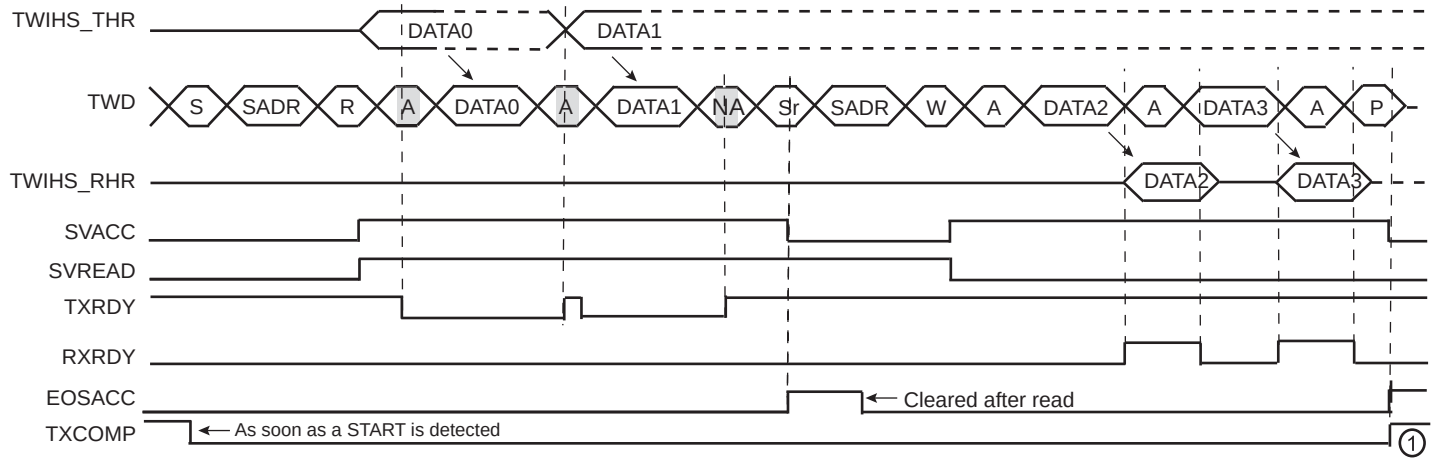
Reversal after a Repeated Start

Reversal of Read to Write

The master initiates the communication by a read command and finishes it by a write command.

Figure 43-41 describes the REPEATED START and the reversal from Read mode to Write mode.

Figure 43-41. Repeated Start and Reversal from Read Mode to Write Mode

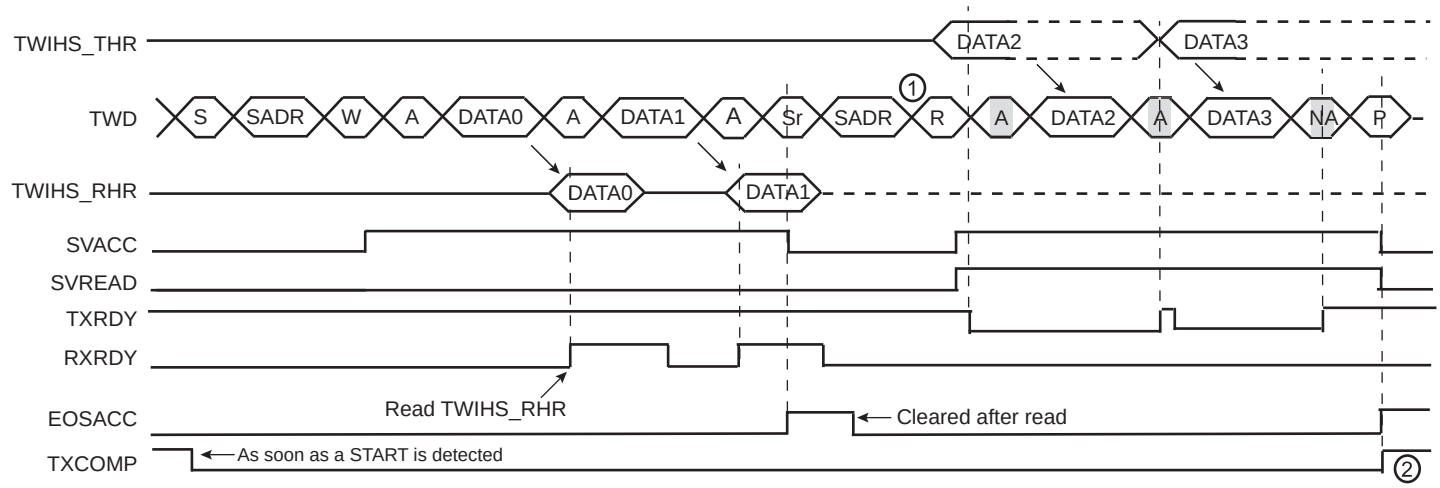


Note: TXCOMP is only set at the end of the transmission. This is because after the REPEATED START, SADR is detected again.

Reversal of Write to Read

The master initiates the communication by a write command and finishes it by a read command. Figure 43-42 describes the REPEATED START and the reversal from Write mode to Read mode.

Figure 43-42. Repeated Start and Reversal from Write Mode to Read Mode



Notes:
1. In this case, if TWIHS_THR has not been written at the end of the read command, the clock is automatically stretched before the ACK.
2. TXCOMP is only set at the end of the transmission. This is because after the REPEATED START, SADR is detected again.

43.6.5.5 Using the DMA Controller (DMAC) in Slave Mode

The use of the DMAC significantly reduces the CPU load.

Data Transmit with the DMA in Slave Mode

The following procedure shows an example to transmit data with DMA.

1. Initialize the transmit DMA (memory pointers, transfer size, etc).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

Data Receive with the DMA in Slave Mode

The following procedure shows an example to transmit data with DMA where the number of characters to receive is known.

1. Initialize the DMA (channels, memory pointers, size, etc.).
2. Configure the Slave mode.
3. Enable the DMA.
4. Wait for the DMA status flag indicating that the buffer transfer is complete.
5. Disable the DMA.
6. (Only if peripheral clock must be disabled) Wait for the TXCOMP flag to be raised in TWIHS_SR.

43.6.5.6 SMBus Mode

SMBus mode is enabled when a one is written to the SMEN bit in the TWIHS_CR. SMBus mode operation is similar to I²C operation with the following exceptions:

- Only 7-bit addressing can be used.
- The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into the TWIHS_SMBTR.
- Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
- A set of addresses have been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring the TWIHS_CR.

Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing a one to the PECEN bit in TWIHS_CR will send/check the PEC field in the current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on the following linked transfers is correct.

In Slave Receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave compares it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave returns an ACK to the master. If the PEC values differ, data was corrupted, and the slave returns a NACK value. The PECERR bit in TWIHS_SR is set automatically if a PEC error occurred.

In Slave Transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master compares it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

See Section 43.6.5.10 “Slave Read Write Flowcharts” for detailed flowcharts.

Timeouts

The TWIHS SMBus Timing Register (TWIHS_SMBTR) configures the SMBus timeout values. If a timeout occurs, the slave leaves the bus. The TOUT bit is also set in TWIHS_SR.

43.6.5.7 High-Speed Slave Mode

High-speed mode is enabled when a one is written to the HSEN bit in the TWIHS_CR. Furthermore, the analog pad filter must be enabled, a one must be written to the PADFEN bit in the TWIHS_FILTR and the FILT bit must be cleared. TWIHS High-speed mode operation is similar to TWIHS operation with the following exceptions:

1. A master code is received first at normal speed before entering High-speed mode period.
2. When TWIHS High-speed mode is active, clock stretching is only allowed after acknowledge (ACK), not-acknowledge (NACK), START (S) or REPEATED START (Sr) (as consequence OVF may happen).

TWIHS High-speed mode allows transfers of up to 3.4 Mbit/s.

The TWIHS slave in High-speed mode requires that the peripheral clock runs at a minimum of 14 MHz if slave clock stretching is enabled (SCLWSDIS bit at ‘0’). If slave clock stretching is disabled (SCLWSDIS bit at ‘1’), the peripheral clock must run at a minimum of 11 MHz (assuming the system has no latency).

Note: When slave clock stretching is disabled, the TWIHS_RHR must always be read before receiving the next data (MASTER write frame). It is strongly recommended to use either the polling method on the RXRDY flag in TWIHS_SR, or the DMA. If the receive is managed by an interrupt, the TWIHS interrupt priority must be set to the right level and its latency minimized to avoid receive overrun.

Note: When slave clock stretching is disabled, the TWIHS_THR must be filled with the first data to send before the beginning of the frame (MASTER read frame). It is strongly recommended to use either the polling method on the TXRDY flag in TWIHS_SR, or the DMA. If the transmit is managed by an interrupt, the TWIHS interrupt priority must be set to the right level and its latency minimized to avoid transmit underrun.

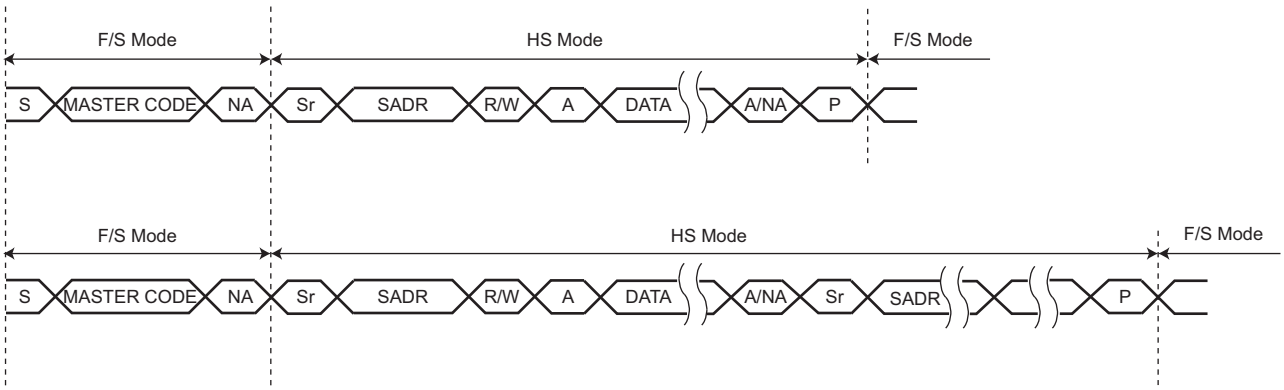
Read/Write Operation

A TWIHS high-speed frame always begins with the following sequence:

1. START condition (S)
2. Master Code (0000 1XXX)
3. Not-acknowledge (NACK)

When the TWIHS is programmed in Slave mode and TWIHS High-speed mode is activated, master code matching is activated and internal timings are set to match the TWIHS High-speed mode requirements.

Figure 43-43. High-Speed Mode Read/Write



Usage

TWIHS High-speed mode usage is the same as the standard TWIHS (See Section 43.6.3.14 “Read/Write Flowcharts”).

43.6.5.8 Alternative Command

In Slave mode, Alternative Command mode is useful when SMBus mode is enabled to send or check the PEC byte.

Alternative Command mode is enabled by setting the ACMEN bit of the TWIHS Control Register and the transfer is configured in TWIHS_ACR.

For a combined transfer with PEC, only the NPEC bit in TWIHS_ACR must be set as the PEC byte is sent once at the end of the frame.

See Section 43.6.5.10 “Slave Read Write Flowcharts” for detailed flowcharts.

43.6.5.9 Asynchronous Partial Wakeup (SleepWalking)

The TWIHS includes an asynchronous start condition detector. It is capable of waking the device up from a Sleep mode upon an address match (and optionally an additional data match), including Sleep modes where the TWIHS peripheral clock is stopped.

After detecting the START condition on the bus, the TWIHS stretches TWCK until the TWIHS peripheral clock has started. The time required for starting the TWIHS depends on which Sleep mode the device is in. After the TWIHS peripheral clock has started, the TWIHS releases its TWCK stretching and receives one byte of data (slave address) on the bus. At this time, only a limited part of the device, including the TWIHS module, receives a clock, thus saving power. If the address phase causes a TWIHS address match (and, optionally, if the first data byte causes data match as well), the entire device is woken up and normal TWIHS address matching actions are performed. Normal TWIHS transfer then follows. If the TWIHS is not addressed (or if the optional data match fails), the TWIHS peripheral clock is automatically stopped and the device returns to its original Sleep mode.

The TWIHS has the capability to match on more than one address. The SADR1EN, SADR2EN and SADR3EN bits in TWIHS_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in the TWIHS_SWMR. The SleepWalking matching process can be extended to the first received data byte if DATAMEN bit in TWIHS_SMR is set and, in this case, a complete matching includes address matching and first received data matching. The field DATAM in TWIHS_SWMR configures the data to match on the first received byte.

When the system is in Active mode and the TWIHS enters Asynchronous Partial Wakeup mode, the flag SVACC must be programmed as the unique source of the TWIHS interrupt and the data match comparison must be disabled.

When the system exits Wait mode as the result of a matching condition, the SVACC flag is used to determine if the TWIHS is the source of exit.

Figure 43-44. Address Match Only (Data Matching Disabled)

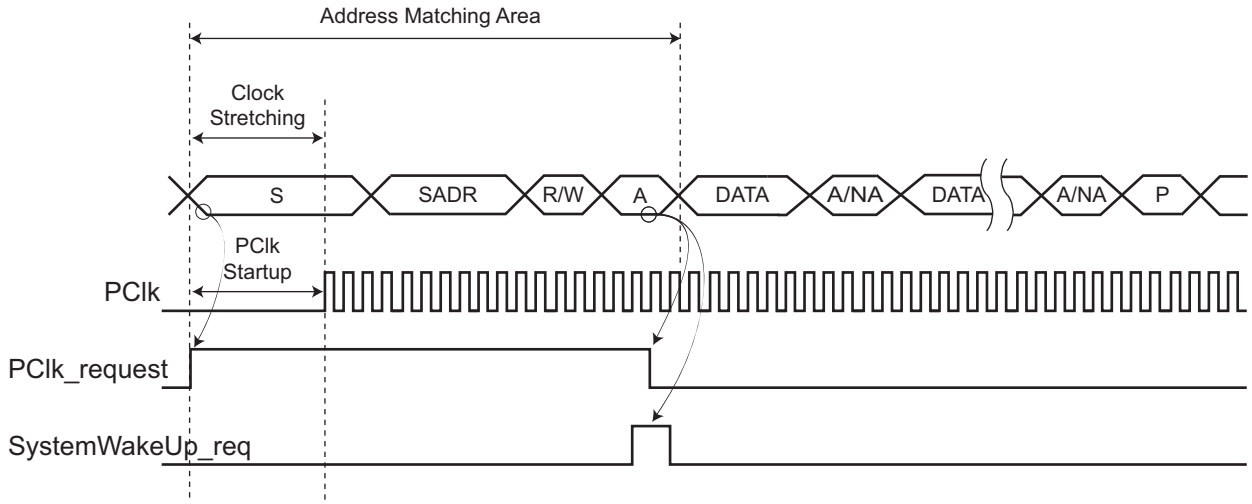


Figure 43-45. No Address Match (Data Matching Disabled)

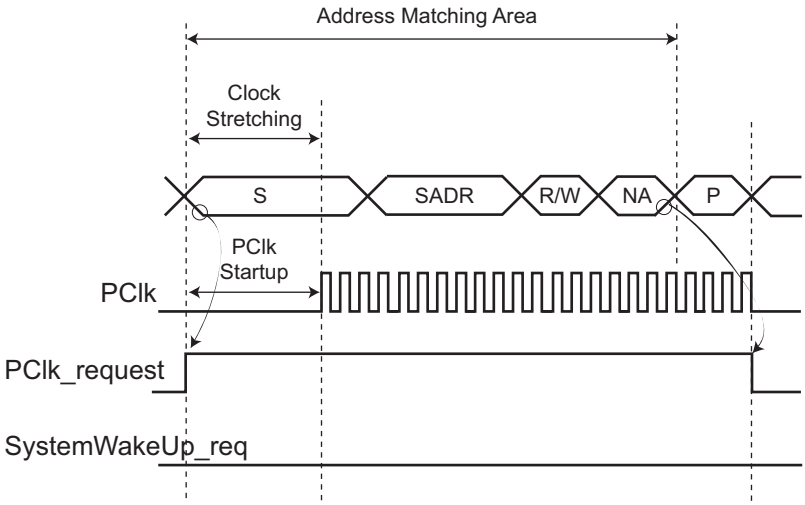


Figure 43-46. Address Match and Data Match (Data Matching Enabled)

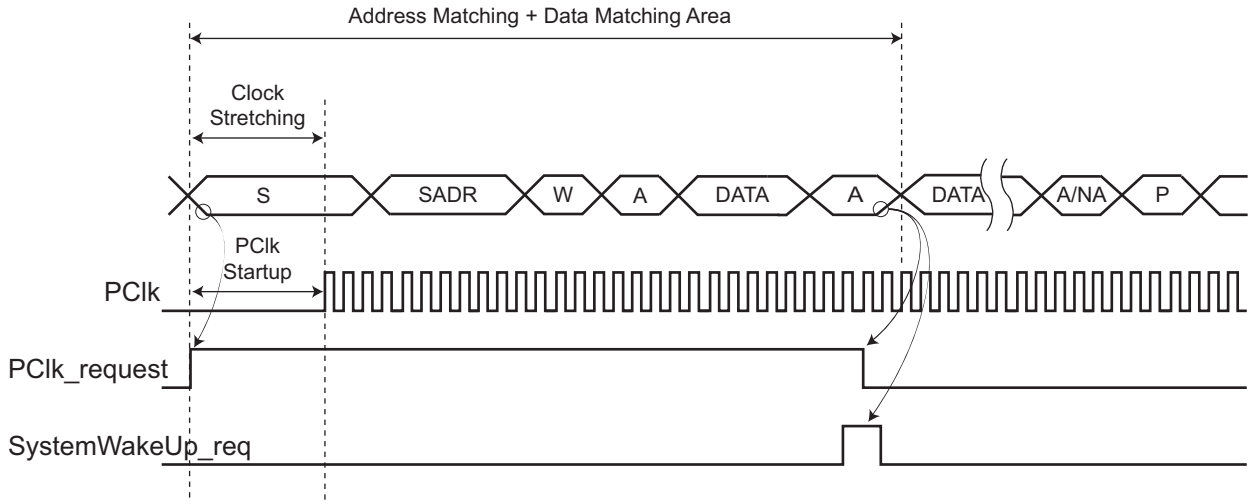
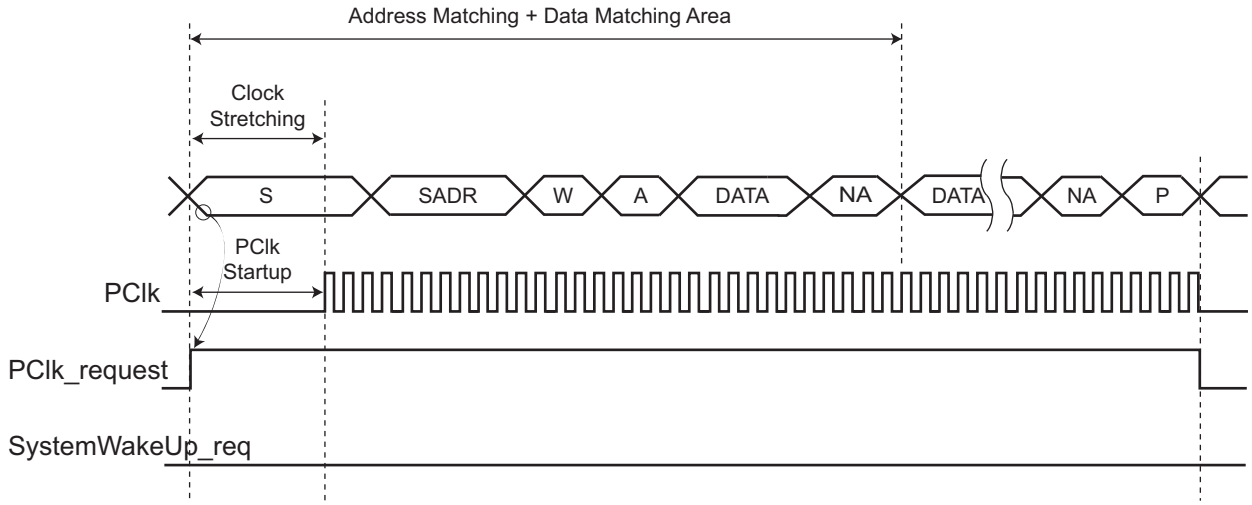


Figure 43-47. Address Match and No Data Match (Data Matching Enabled)



43.6.5.10 Slave Read Write Flowcharts

The flowchart shown in Figure 43-48 gives an example of read and write operations in Slave mode. A polling or interrupt method can be used to check the status bits. The interrupt method requires that TWIHS_IER be configured first.

Figure 43-48. Read Write Flowchart in Slave Mode

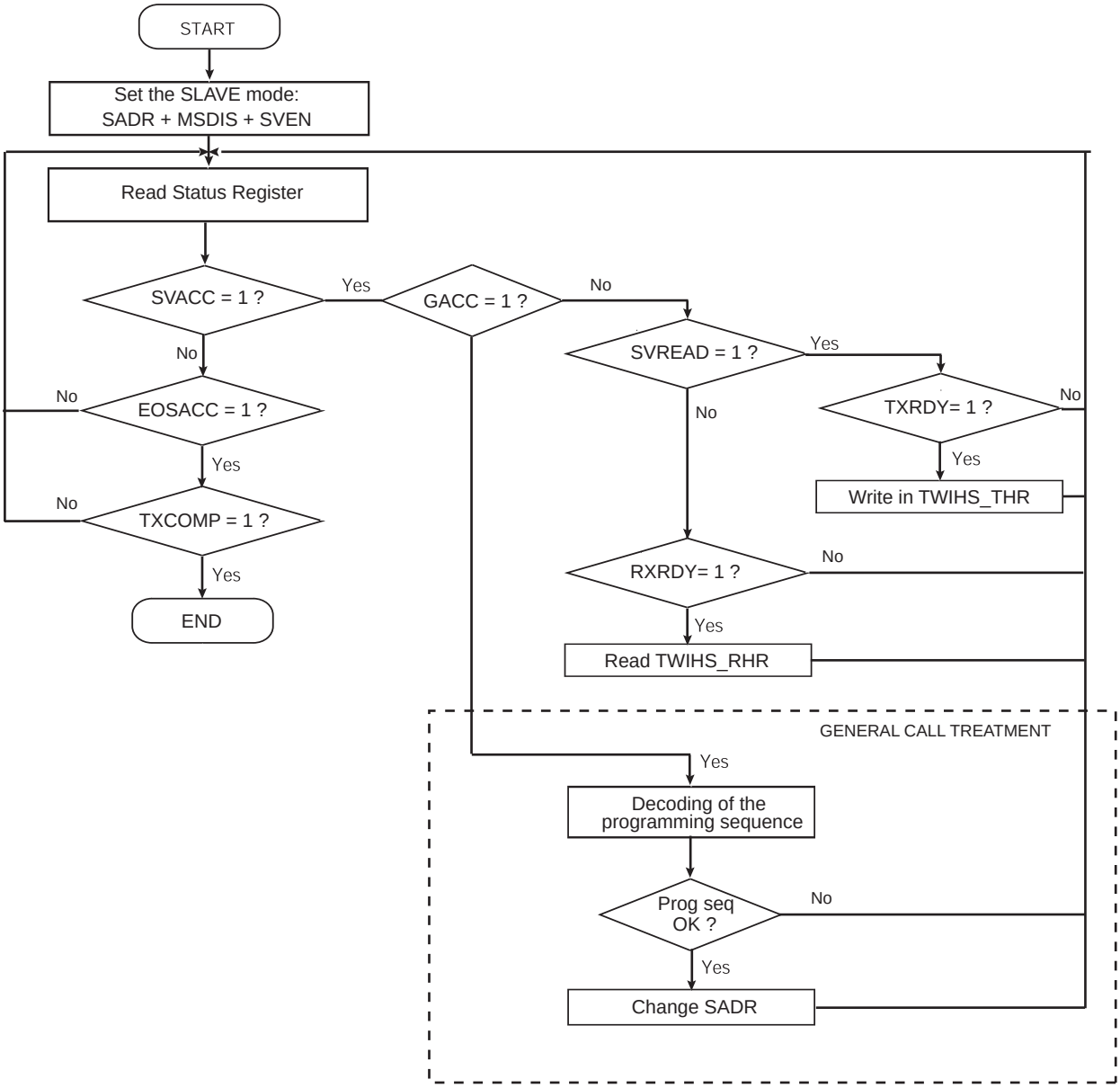


Figure 43-49. Read Write Flowchart in Slave Mode with SMBus PEC

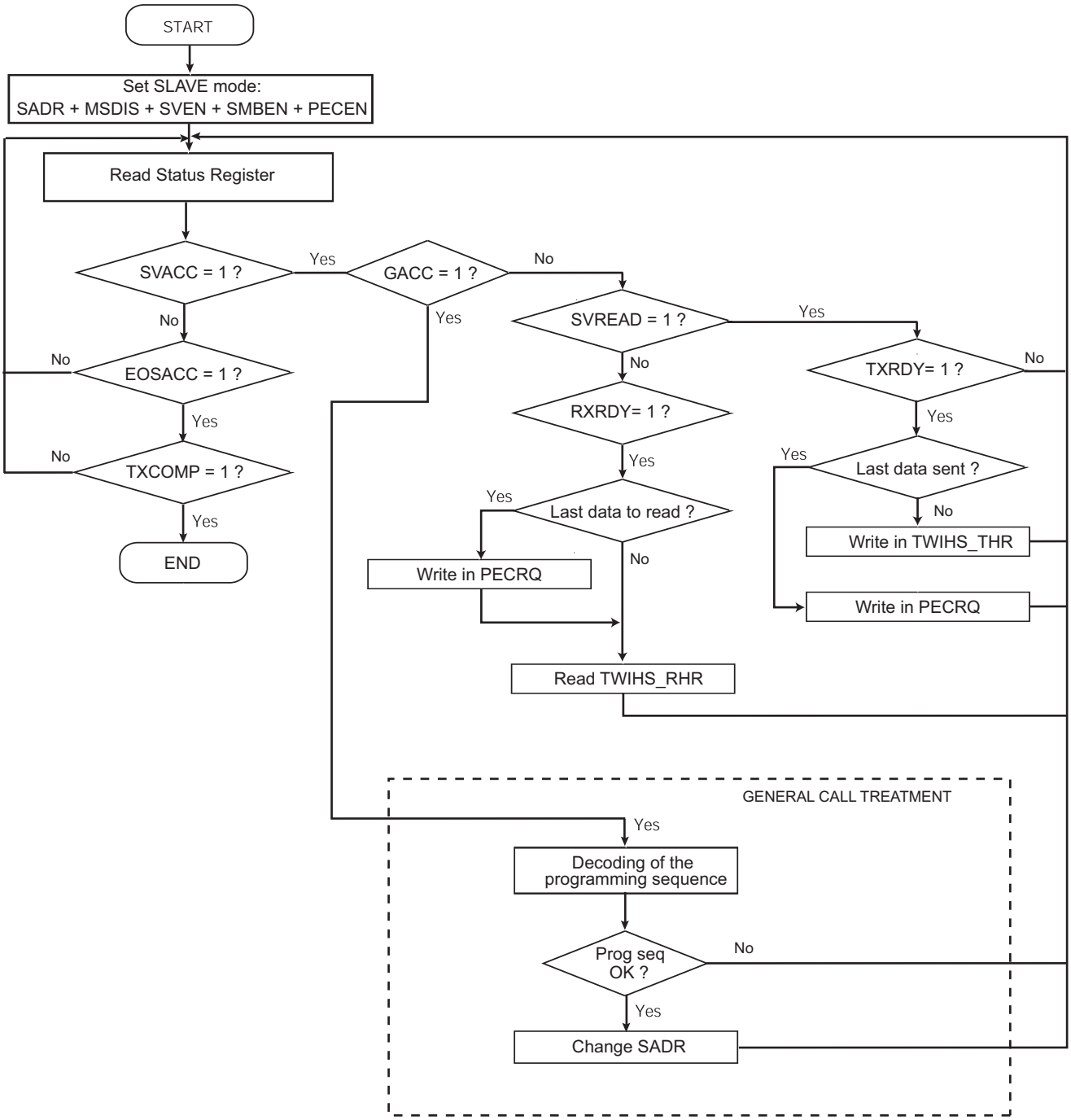
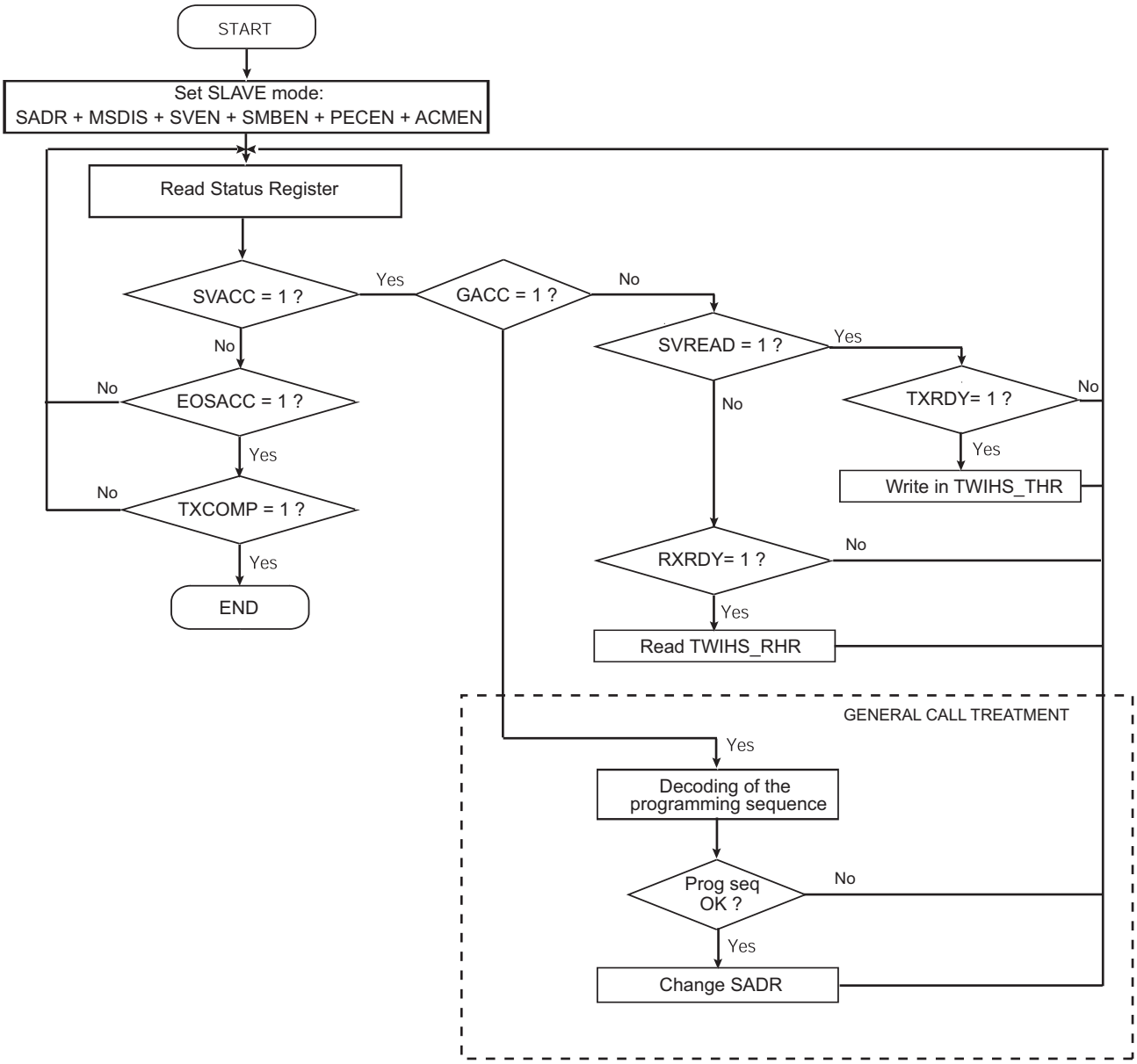


Figure 43-50. Read Write Flowchart in Slave Mode with SMBus PEC and Alternative Command Mode

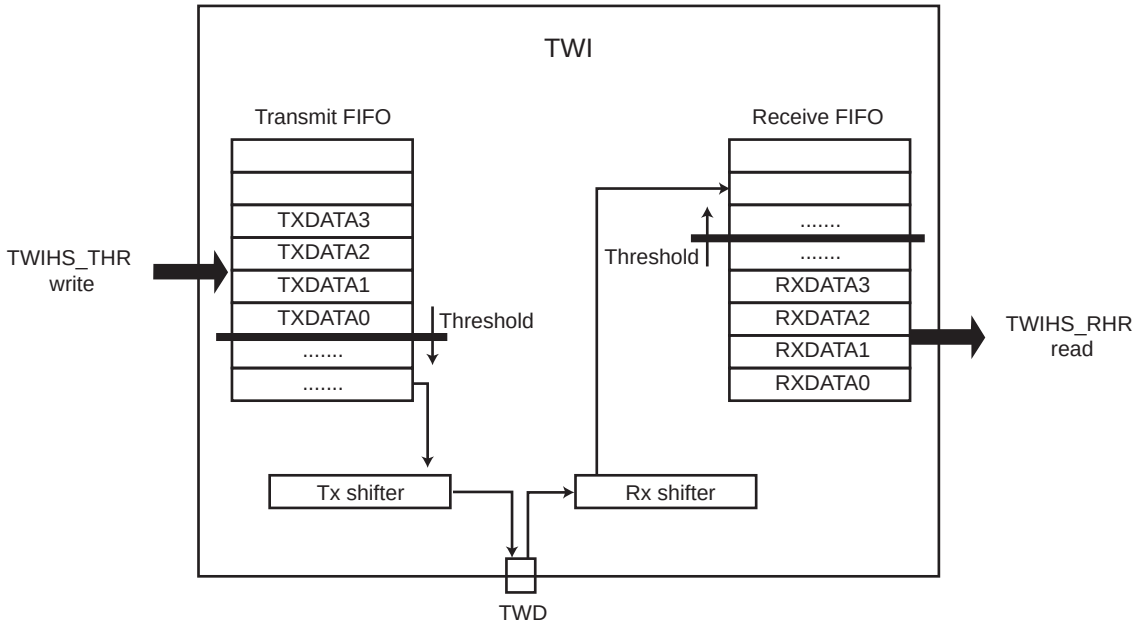


43.6.5.11 FIFOs

The TWIHS includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in the TWIHS_CR. It is recommended to disable both Master and Slave modes before enabling or disabling the FIFO (MSDIS and SVDIS bit in TWIHS_CR).

Writing the FIFOEN bit to ‘1’ will enable a 16-data Transmit FIFO and a 16-data Receive FIFO.

Figure 43-51. FIFOs Block Diagram



Sending/Receiving Data with FIFO Enabled

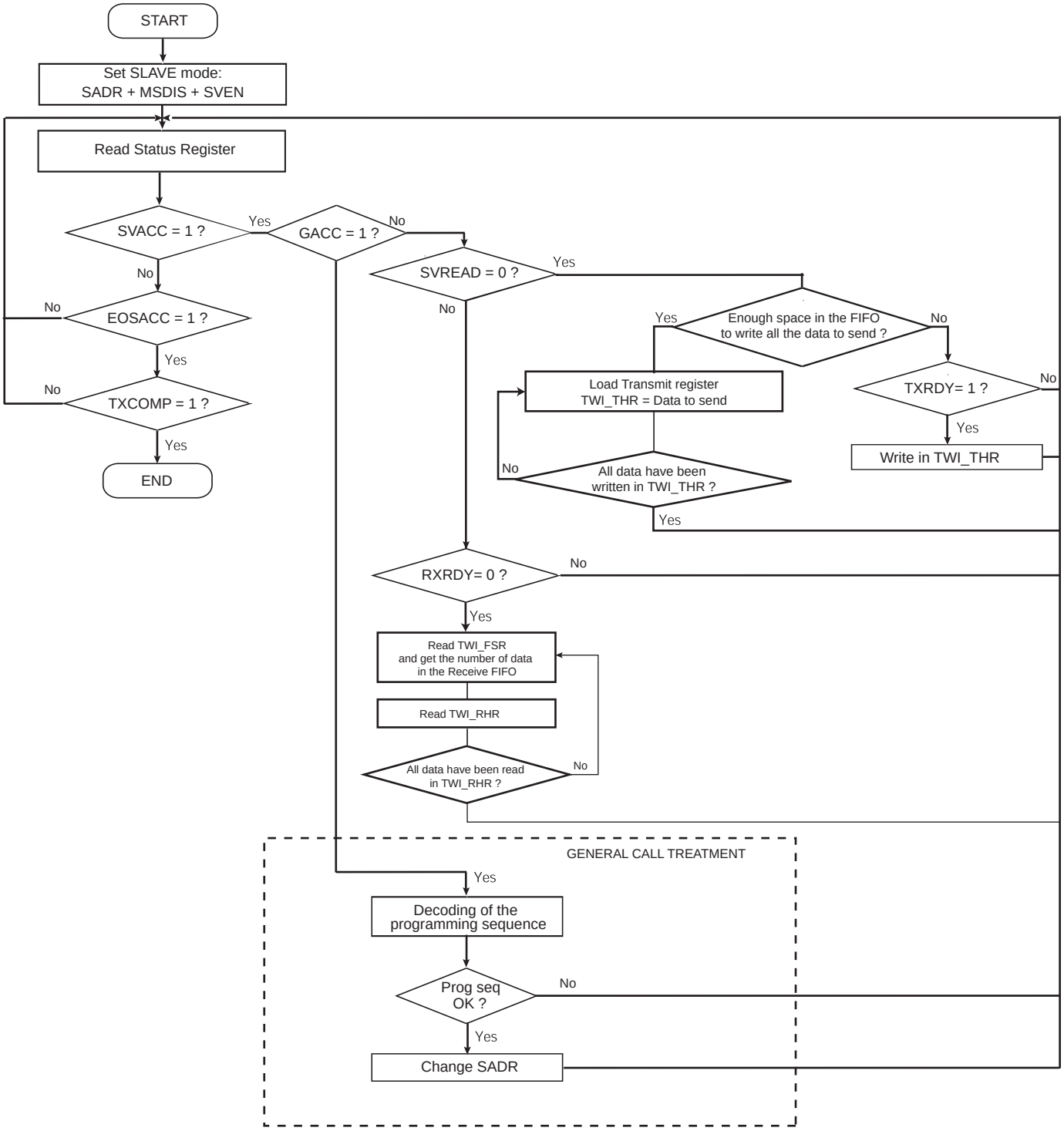
With the Transmit FIFO enabled, any write access to the TWIHS_THR will bring the written data to the Transmit FIFO. As a consequence, it is not mandatory any more to monitor the TXRDY flag state to send multiple data without DMAC.

Knowing the number of data to send and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in the TWIHS_THR without checking the TXRDY flag between each access. The Transmit FIFO state can be checked reading the TXFL field in the TWIHS_FLR.

With Receive FIFO enabled, any read access on TWIHS_RHR will pull out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RXRDY flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with the RXFL field in the TWIHS_FLR and all the data read successively in the TWIHS_RHR without checking the RXRDY flag between each access.

Figure 43-52. Sending/Receiving Data with FIFO Flowchart



Clearing/Flushing FIFOs

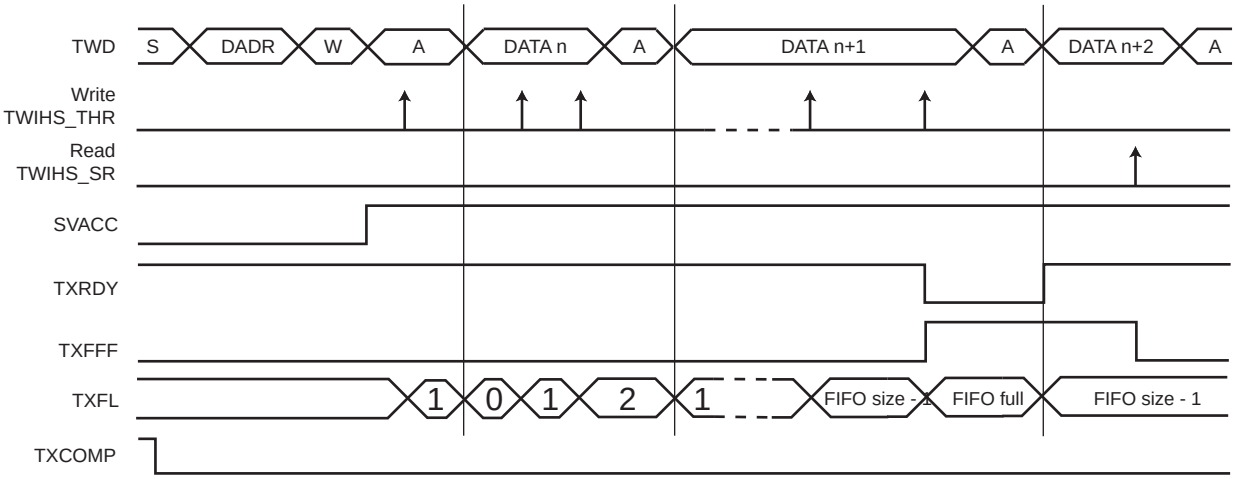
Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in the TWIHS_CR.

TXRDY and RXRDY Behavior

If FIFOs are enabled, the behavior of the TXRDY and RXRDY flags will be slightly different.

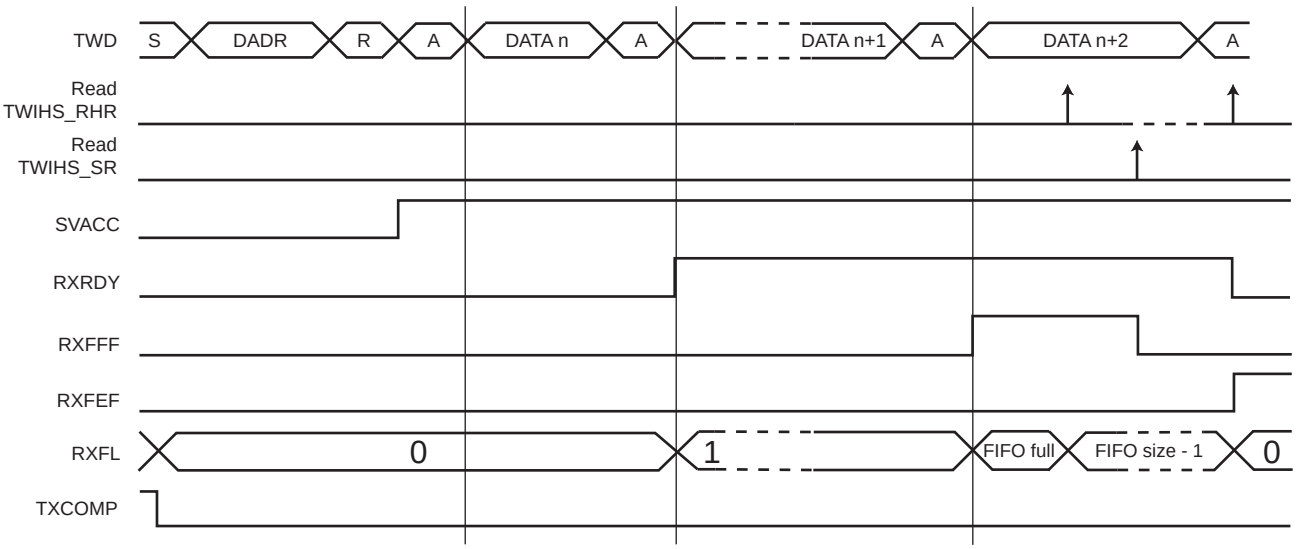
TXRDY will indicate if a data can be written in the Transmit FIFO. By default, the TXRDY flag will then stay at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 43-53. TXRDY in Single Data Mode and TXRDYM = 0x0



RXRDY will indicate if an unread data is present in the Receive FIFO. By default, the RXRDY flag will then be at level ‘1’ as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 43-54. RXRDY in Single Data Mode and RXRDYM = 0x0



TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the TWIHS_FMR. In Single Data mode, there is no need to modify the TXRDY and RXRDY behavior; however, it may be useful in Multiple Data mode.

Slave Multiple Data Mode

When the FIFOs are enabled, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one TWIHS_THR/TWIHS_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data will be written/read; if the access is a halfword-size register access, then two data will be written/read and finally, if the access is a word-size register access, then four data will be written/read.

Written/Read data are always right-aligned, as shown in Section 43.7.14 "TWIHS Receive Holding Register (FIFO Enabled)" and Section 43.7.17 "TWIHS Transmit Holding Register (FIFO Enabled)".

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six TWIHS_THR-byte write accesses.
- Perform three TWIHS_THR-halfword write accesses.
- Perform one TWIHS_THR-word write access and one TWIHS_THR-halfword write access.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six TWIHS_RHR-byte read accesses.
- Perform three TWIHS_RHR-halfword read accesses.
- Perform one TWIHS_RHR-word read access and one TWIHS_RHR-halfword read access.

Multiple Data mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

- TXRDY and RXRDY configuration

  In Multiple Data mode, the TXRDYM and RXRDYM fields in TWIHS_FMR become useful.

  As in Multiple Data mode, it is possible to write several data in the same access; it might be useful to configure the TXRDY flag behavior to indicate if 1, 2 or 4 data can be written in the FIFO depending on the access to perform on TWIHS_THR.

  If for instance four data are written each time in the TWIHS_THR it might be useful to configure the TXRDYM field to 0x2 value so that the TXRDY flag will be at '1' only when at least four data can be written in the Transmit FIFO.

  In the same way, if four data are read each time in the TWIHS_RHR, it might be useful to configure the RXRDYM field to 0x2 value so that the RXRDY flag will be at '1' only when at least four unread data are in the Receive FIFO.

- DMAC

  If DMAC transfer is used, it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, reset DMAC channels, etc., without any risk.

The LOCK bit in the TWIHS_SR is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting the TXFLCLR bit to '1' in the TWIHS_CR.

FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

  If the Transmit FIFO is full and a write access is performed on the TWIHS_THR it will generate a Transmit FIFO pointer error and set the TXFPTEF flag in TWIHS_FSR.

  In Multiple Data mode, if the number of data written in the TWIHS_THR (according to the register access size) is bigger than the Transmit FIFO free space, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in TWIHS_FSR.

- Receive FIFO:

  In Multiple Data mode, if the number of data read in the TWIHS_RHR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it generates a Receive FIFO pointer error and sets the RXFPTEF flag in TWIHS_FSR.

  Pointer error should not happen if the FIFO state is checked before writing/reading in the TWIHS_THR/TWIHS_RHR registers. The FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using the SWRST bit in the TWIHS_CR. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent afterwards might not be received properly by the slave).

FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO:

  The Transmit FIFO threshold can be set using the TXFTHRES field in TWIHS_FMR. Each time the Transmit FIFO goes from the ‘above threshold’ to the ‘equal or below threshold’ state, the TXFTHF flag in TWIHS_FSR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.

- Receive FIFO:

  The Receive FIFO threshold can be set using the RXFTHRES field in TWIHS_FMR. Each time the Receive FIFO goes from the ‘below threshold’ to the ‘equal or above threshold’ state, the RXFTHF flag in TWIHS_FSR is set. This enables the application to know that the Receive FIFO reached the defined threshold and to read some data before it becomes full.

The TXFTHF and RXFTHF flags can be both configured to generate an interrupt using TWIHS_FIER and TWIHS_FIDR.

FIFO Flags

FIFOs come with a set of flags which can be configured each to generate an interrupt through TWIHS_FIER and TWIHS_FIDR.

FIFO flags state can be read in TWIHS_FSR. They are cleared on TWIHS_FSR read.

43.6.6 TWIHS Comparison Function on Received Character

The comparison function differs if asynchronous partial wakeup (SleepWalking) is enabled or not.

If asynchronous partial wakeup is disabled (see the section “Power Management Controller (PMC)”), the TWIHS can extend the address matching on up to three slave addresses. The SADR1EN, SADR2EN and SADR3EN bits in TWIHS_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in the TWIHS_SWMR. The DATAMEN bit in the TWIHS_SMR has no effect.

The SVACC bit is set when there is a comparison match with the received slave address.

43.6.7 Register Write Protection

To prevent any single software error from corrupting TWIHS behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TWIHS Write Protection Mode Register (TWIHS_WPMR).

If a write access to a write-protected register is detected, the WPVS bit in the TWIHS Write Protection Status Register (TWIHS_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading TWIHS_WPSR.

The following registers can be write-protected:

- TWIHS Slave Mode Register
- TWIHS Clock Waveform Generator Register
- TWIHS SMBus Timing Register
- TWIHS SleepWalking Matching Register

43.7 Two-wire Interface High Speed (TWIHS) User Interface

Table 43-7. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TWIHS_CR	Write-only	–
0x04	Master Mode Register	TWIHS_MMR	Read/Write	0x00000000
0x08	Slave Mode Register	TWIHS_SMR	Read/Write	0x00000000
0x0C	Internal Address Register	TWIHS_IADR	Read/Write	0x00000000
0x10	Clock Waveform Generator Register	TWIHS_CWGR	Read/Write	0x00000000
0x14–0x1C	Reserved	–	–	–
0x20	Status Register	TWIHS_SR	Read-only	0x0300F009
0x24	Interrupt Enable Register	TWIHS_IER	Write-only	–
0x28	Interrupt Disable Register	TWIHS_IDR	Write-only	–
0x2C	Interrupt Mask Register	TWIHS_IMR	Read-only	0x00000000
0x30	Receive Holding Register	TWIHS_RHR	Read-only	0x00000000
0x34	Transmit Holding Register	TWIHS_THR	Write-only	0x00000000
0x38	SMBus Timing Register	TWIHS_SMBTR	Read/Write	0x00000000
0x3C	Reserved	–	–	–
0x40	Alternative Command Register	TWIHS_ACR	Read/Write	–
0x44	Filter Register	TWIHS_FILTR	Read/Write	0x00000000
0x48	Reserved	–	–	–
0x4C	SleepWalking Matching Register	TWIHS_SWMR	Read/Write	0x00000000
0x50	FIFO Mode Register	TWIHS_FMR	Read/Write	0x00000000
0x54	FIFO Level Register	TWIHS_FLR	Read-only	0x00000000
0x58–0x5C	Reserved	–	–	–
0x60	FIFO Status Register	TWIHS_FSR	Read-only	0x00000000
0x64	FIFO Interrupt Enable Register	TWIHS_FIER	Write-only	–
0x68	FIFO Interrupt Disable Register	TWIHS_FIDR	Write-only	–
0x6C	FIFO Interrupt Mask Register	TWIHS_FIMR	Read-only	0x00000000
0x70–0xCC	Reserved	–	–	–
0xD0	Reserved	–	–	–
0xD4–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	TWIHS_WPMR	Read/Write	0x00000000
0xE8	Write Protection Status Register	TWIHS_WPSR	Read-only	0x00000000
0xEC–0xFC[1]	Reserved	–	–	–
0x100–0x128	Reserved	–	–	–

Note: 1. All unlisted offset values are considered as “reserved”.

43.7.1 TWIHS Control Register

Name: TWIHS_CR

Address: 0xF8028000 (0), 0xFC028000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	FIFODIS	FIFOEN	–	LOCKCLR	–	THRCLR
23	22	21	20	19	18	17	16
–	–	–	–	–	–	ACMDIS	ACMEN
15	14	13	12	11	10	9	8
CLEAR	PECRQ	PECDIS	PECEN	SMBDIS	SMBEN	HSDIS	HSEN
7	6	5	4	3	2	1	0
SWRST	QUICK	SVDIS	SVEN	MSDIS	MSEN	STOP	START

- **START: Send a START Condition**

0: No effect.

1: A frame beginning with a START bit is transmitted according to the features defined in the TWIHS Master Mode Register (TWIHS_MMR).

This action is necessary when the TWIHS peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (TWIHS_THR).

- **STOP: Send a STOP Condition**

0: No effect.

1: STOP condition is sent just after completing the current byte transmission in Master Read mode.

  – In single data byte master read, both START and STOP must be set.
  – In multiple data bytes master read, the STOP must be set after the last data received but one.
  – In Master Read mode, if a NACK bit is received, the STOP is automatically performed.
  – In master data write operation, a STOP condition will be sent after the transmission of the current data is finished.

- **MSEN: TWIHS Master Mode Enabled**

0: No effect.

1: Enables the Master mode (MSDIS must be written to 0).

Note: Switching from Slave to Master mode is only permitted when TXCOMP = 1.

- **MSDIS: TWIHS Master Mode Disabled**

0: No effect.

1: The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling.

- **SVEN: TWIHS Slave Mode Enabled**

0: No effect.

1: Enables the Slave mode (SVDIS must be written to 0).

Note: Switching from Master to Slave mode is only permitted when TXCOMP = 1.

- **SVDIS: TWIHS Slave Mode Disabled**

0: No effect.

1: The Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in case of read operation. In write operation, the character being transferred must be completely received before disabling.

- **QUICK: SMBus Quick Command**

0: No effect.

1: If Master mode is enabled, a SMBus Quick Command is sent.

- **SWRST: Software Reset**

0: No effect.

1: Equivalent to a system reset.

- **HSEN: TWIHS High-Speed Mode Enabled**

0: No effect.

1: High-speed mode enabled.

- **HSDIS: TWIHS High-Speed Mode Disabled**

0: No effect.

1: High-speed mode disabled.

- **SMBEN: SMBus Mode Enabled**

0: No effect.

1: If SMBDIS = 0, SMBus mode enabled.

- **SMBDIS: SMBus Mode Disabled**

0: No effect.

1: SMBus mode disabled.

- **PECEN: Packet Error Checking Enable**

0: No effect.

1: SMBus PEC (CRC) generation and check enabled.

- **PECDIS: Packet Error Checking Disable**

0: No effect.

1: SMBus PEC (CRC) generation and check disabled.

- **PECRQ: PEC Request**

0: No effect.

1: A PEC check or transmission is requested.

- **CLEAR: Bus CLEAR Command**

0: No effect.

1: If Master mode is enabled, sends a bus clear command.

- **ACMEN: Alternative Command Mode Enable**

0: No effect.

1: Alternative Command mode enabled.

- **ACMDIS: Alternative Command Mode Disable**

0: No effect.

1: Alternative Command mode disabled.

- **THRCLR: Transmit Holding Register Clear**

0: No effect.

1: Clears the Transmit Holding Register and set TXRDY, TXCOMP flags.

- **LOCKCLR: Lock Clear**

0: No effect.

1: Clears the TWIHS FSM lock.

- **FIFOEN: FIFO Enable**

0: No effect.

1: Enables the Transmit and Receive FIFOs

- **FIFODIS: FIFO Disable**

0: No effect.

1: Disables the Transmit and Receive FIFOs

43.7.2 TWIHS Master Mode Register

Name: TWIHS_MMR

Address: 0xF8028004 (0), 0xFC028004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	DADR						

15	14	13	12	11	10	9	8
–	–	–	MREAD	–	–	IADRSZ	

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **IADRSZ: Internal Device Address Size**

Value	Name	Description
0	NONE	No internal device address
1	1_BYTE	One-byte internal device address
2	2_BYTE	Two-byte internal device address
3	3_BYTE	Three-byte internal device address

- **MREAD: Master Read Direction**

0: Master write direction.

1: Master read direction.

- **DADR: Device Address**

The device address is used to access slave devices in Read or Write mode. These bits are only used in Master mode.

43.7.3 TWIHS Slave Mode Register

Name: TWIHS_SMR

Address: 0xF8028008 (0), 0xFC028008 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
DATAMEN	SADR3EN	SADR2EN	SADR1EN	–	–	–	–
23	22	21	20	19	18	17	16
–	SADR						
15	14	13	12	11	10	9	8
–	MASK						
7	6	5	4	3	2	1	0
–	SCLWSDIS	–	–	SMHH	SMDA	–	NACKEN

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

- **NACKEN: Slave Receiver Data Phase NACK enable**

0: Normal value to be returned in the ACK cycle of the data phase in Slave Receiver mode.

1: NACK value to be returned in the ACK cycle of the data phase in Slave Receiver mode.

- **SMDA: SMBus Default Address**

0: Acknowledge of the SMBus default address disabled.

1: Acknowledge of the SMBus default address enabled.

- **SMHH: SMBus Host Header**

0: Acknowledge of the SMBus host header disabled.

1: Acknowledge of the SMBus host header enabled.

- **SCLWSDIS: Clock Wait State Disable**

0: No effect.

1: Clock stretching disabled in Slave mode, OVRE and UNRE indicate an overrun/underrun.

- **MASK: Slave Address Mask**

A mask can be applied on the slave device address in Slave mode in order to allow multiple address answer. For each bit of the MASK field set to 1, the corresponding SADR bit is masked.

If the MASK field value is 0, no mask is applied to the SADR field.

- **SADR: Slave Address**

The slave device address is used in Slave mode in order to be accessed by master devices in Read or Write mode.

SADR must be programmed before enabling the Slave mode or after a general call. Writes at other times have no effect.

- **SADR1EN: Slave Address 1 Enable**

0: Slave address 1 matching is disabled.

1: Slave address 1 matching is enabled.

- **SADR2EN: Slave Address 2 Enable**

0: Slave address 2 matching is disabled.

1: Slave address 2 matching is enabled.

- **SADR3EN: Slave Address 3 Enable**

0: Slave address 3 matching is disabled.

1: Slave address 3 matching is enabled.

- **DATAMEN: Data Matching Enable**

0: Data matching on first received data is disabled.

1: Data matching on first received data is enabled.

43.7.4 TWIHS Internal Address Register

Name: TWIHS_IADR

Address: 0xF802800C (0), 0xFC02800C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
IADR							

15	14	13	12	11	10	9	8
IADR							

7	6	5	4	3	2	1	0
IADR							

- **IADR: Internal Address**

0, 1, 2 or 3 bytes depending on IADRSZ.

43.7.5 TWIHS Clock Waveform Generator Register

Name: TWIHS_CWGR

Address: 0xF8028010 (0), 0xFC028010 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	HOLD				

23	22	21	20	19	18	17	16
–	–	–	CKSRC	–	CKDIV		

15	14	13	12	11	10	9	8
CHDIV							

7	6	5	4	3	2	1	0
CLDIV							

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

TWIHS_CWGR is used in Master mode only.

- **CLDIV: Clock Low Divider**

The SCL low period is defined as follows:

If CKSRC = 0

$t_{low} = ((CLDIV \times 2^{CKDIV}) + 3) \times t_{peripheral\ clock}$

If CKSRC = 1

$t_{low} = (CLDIV \times 2^{CKDIV}) \times t_{external\ clock}$

- **CHDIV: Clock High Divider**

The SCL high period is defined as follows:

If CKSRC = 0

$t_{high} = ((CHDIV \times 2^{CKDIV}) + 3) \times t_{peripheral\ clock}$

If CKSRC = 1

$t_{high} = (CHDIV \times 2^{CKDIV}) \times t_{external\ clock}$

- **CKDIV: Clock Divider**

The CKDIV is used to increase both SCL high and low periods.

- **HOLD: TWD Hold Time Versus TWCK Falling**

If High-speed mode is selected TWD is internally modified on the TWCK falling edge to meet the I2C specified maximum hold time, else if High-speed mode is not configured TWD is kept unchanged after TWCK falling edge for a period of $(HOLD + 3) \times t_{peripheral\ clock}$.

- **CKSRC: Transfer Rate Clock Source**

Value	Name	Description
0	PERIPH_CK	Peripheral clock is used to generate the TWIHS baud rate.
1	GCLK	GCLK is used to generate the TWIHS baud rate.

43.7.6 TWIHS Status Register

Name: TWIHS_SR

Address: 0xF8028020 (0), 0xFC028020 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	SDA	SCL
23	22	21	20	19	18	17	16
LOCK	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK
15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCLWS	ARBLST	NACK
7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP

- **TXCOMP: Transmission Completed (cleared by writing TWIHS_THR)**

TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both holding register and internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in Figure 43-6 and in Figure 43-8.

TXCOMP used in Slave mode:

0: As soon as a START is detected.

1: After a STOP or a REPEATED START + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in Figure 43-39, Figure 43-40, Figure 43-41 and Figure 43-42.

- **RXRDY: Receive Holding Register Ready (cleared by reading TWIHS_RHR)**

0: No character has been received since the last TWIHS_RHR read operation.

1: A byte has been received in the TWIHS_RHR since the last read.

RXRDY behavior in Master mode can be seen in Figure 43-7, Figure 43-8 and Figure 43-9.

RXRDY behavior in Slave mode can be seen in Figure 43-37, Figure 43-40, Figure 43-41 and Figure 43-42.

- **TXRDY: Transmit Holding Register Ready (cleared by writing TWIHS_THR)**

TXRDY used in Master mode:

0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into TWIHS_THR.

1: As soon as a data byte is transferred from TWIHS_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWIHS).

TXRDY behavior in Master mode can be seen in Figure 43-4, Figure 43-5 and Figure 43-6.

TXRDY used in Slave mode:

0: As soon as data is written in the TWIHS_THR, until this data has been transmitted and acknowledged (ACK or NACK).

1: Indicates that the TWIHS_THR is empty and that data has been transmitted and acknowledged.

If TXRDY is high and if a NACK has been detected, the transmission is stopped. Thus when TRDY = NACK = 1, the user must not fill TWIHS_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in Figure 43-36, Figure 43-39, Figure 43-41 and Figure 43-42.

- **SVREAD: Slave Read**

This bit is used in Slave mode only. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.

0: Indicates that a write access is performed by a master.

1: Indicates that a read access is performed by a master.

SVREAD behavior can be seen in Figure 43-36, Figure 43-37, Figure 43-41 and Figure 43-42.

- **SVACC: Slave Access**

This bit is used in Slave mode only.

0: TWIHS is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected.

1: Indicates that the address decoding sequence has matched (A master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected.

SVACC behavior can be seen in Figure 43-36, Figure 43-37, Figure 43-41 and Figure 43-42.

- **GACC: General Call Access (cleared on read)**

This bit is used in Slave mode only.

0: No general call has been detected.

1: A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes.

GACC behavior can be seen in Figure 43-38.

- **OVRE: Overrun Error (cleared on read)**

This bit is used only if clock stretching is disabled.

0: TWIHS_RHR has not been loaded while RXRDY was set.

1: TWIHS_RHR has been loaded while RXRDY was set. Reset by read in TWIHS_SR when TXCOMP is set.

- **UNRE: Underrun Error (cleared on read)**

This bit is used only if clock stretching is disabled.

0: TWIHS_THR has been filled on time.

1: TWIHS_THR has not been filled on time.

- **NACK: Not Acknowledged (cleared on read)**

NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWIHS slave component.

1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.

1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set, the user must not fill TWIHS_THR even if TXRDY is set, because it means that the master stops the data transfer or reinitiate it.

Note: in Slave Write mode, all data are acknowledged by the TWIHS.

- **ARBLST: Arbitration Lost (cleared on read)**

This bit is used in Master mode only.

0: Arbitration won.

1: Arbitration lost. Another master of the TWIHS bus has won the multimaster arbitration. TXCOMP is set at the same time.

- **SCLWS: Clock Wait State**

This bit is used in Slave mode only.

0: The clock is not stretched.

1: The clock is stretched. TWIHS_THR / TWIHS_RHR buffer is not filled / emptied before the transmission / reception of a new character.

SCLWS behavior can be seen in Figure 43-39 and Figure 43-40.

- **EOSACC: End Of Slave Access (cleared on read)**

This bit is used in Slave mode only.

0: A slave access is being performing.

1: The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset.

EOSACC behavior can be seen in Figure 43-41 and Figure 43-42.

- **MCACK: Master Code Acknowledge (cleared on read)**

MACK used in Slave mode:

0: No Master Code has been received since the last read of TWIHS_SR.

1: A Master Code has been received since the last read of TWIHS_SR.

- **TOUT: Timeout Error (cleared on read)**

0: No SMBus timeout occurred since the last read of TWIHS_SR.

1: An SMBus timeout occurred since the last read of TWIHS_SR.

- **PECERR: PEC Error (cleared on read)**

0: No SMBus PEC error occurred since the last read of TWIHS_SR.

1: An SMBus PEC error occurred since the last read of TWIHS_SR.

- **SMBDAM: SMBus Default Address Match (cleared on read)**

0: No SMBus Default Address received since the last read of TWIHS_SR.

1: An SMBus Default Address was received since the last read of TWIHS_SR.

- **SMBHHM: SMBus Host Header Address Match (cleared on read)**

0: No SMBus Host Header Address received since the last read of TWIHS_SR.

1: An SMBus Host Header Address was received since the last read of TWIHS_SR.

- **LOCK: TWIHS Lock due to Frame Errors (cleared by writing a one to bit LOCKCLR in TWIHS_CR)**

0: The TWIHS is not locked or LOCKCLR command issued in TWIHS_CR.

1: The TWIHS is locked due to frame errors (see Section 43.6.3.13 “Handling Errors in Alternative Command”).

- **SCL: SCL Line Value**

0: SCL line sampled value is ‘0’.

1: SCL line sampled value is ‘1.’

- **SDA: SDA Line Value**

0: SDA line sampled value is ‘0’.

1: SDA line sampled value is ‘1’.

43.7.7 TWIHS SMBus Timing Register

Name: TWIHS_SMBTR

Address: 0xF8028038 (0), 0xFC028038 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
THMAX							

23	22	21	20	19	18	17	16
TLOWM							

15	14	13	12	11	10	9	8
TLOWS							

7	6	5	4	3	2	1	0
–	–	–	–	PRESC			

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

- **PRESC: SMBus Clock Prescaler**

Used to specify how to prescale the TLOWS, TLOWM and THMAX counters in SMBTR. Counters are prescaled according to the following formula:

$$f_{Prescaled} = \frac{f_{\text{peripheral clock}}}{2^{(PRESC+1)}}$$

- **TLOWS: Slave Clock Stretch Maximum Cycles**

0: TLOW:SEXT timeout check disabled.

1–255: Clock cycles in slave maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:SEXT.

- **TLOWM: Master Clock Stretch Maximum Cycles**

0: TLOW:MEXT timeout check disabled.

1–255: Clock cycles in master maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:MEXT.

- **THMAX: Clock High Maximum Cycles**

Clock cycles in clock high maximum count. Prescaled by PRESC. Used for bus free detection. Used to time THIGH:MAX.

43.7.8 TWIHS Alternative Command Register

Name: TWIHS_ACR

Address: 0xF8028040 (0), 0xFC028040 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	NPEC	NDIR

23	22	21	20	19	18	17	16
NDATAL							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	PEC	DIR

7	6	5	4	3	2	1	0
DATAL							

- **DATAL: Data Length**

0: No data to send (see Section 43.6.3.12 “Alternative Command”).

1–255: Number of bytes to send during the transfer.

- **DIR: Transfer Direction**

0: Write direction.

1: Read direction.

- **PEC: PEC Request (SMBus Mode only)**

0: The transfer does not use a PEC byte.

1: The transfer uses a PEC byte.

- **NDATAL: Next Data Length**

0: No data to send (see Section 43.6.3.12 “Alternative Command”.

1–255: Number of bytes to send for the next transfer.

- **NDIR: Next Transfer Direction**

0: Write direction.

1: Read direction.

- **NPEC: Next PEC Request (SMBus Mode only)**

0: The next transfer does not use a PEC byte.

1: The next transfer uses a PEC byte.

43.7.9 TWIHS Filter Register

Name: TWIHS_FILTR

Address: 0xF8028044 (0), 0xFC028044 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	THRES		
7	6	5	4	3	2	1	0
–	–	–	–	–	PADFCFG	PADFEN	FILT

- **FILT: RX Digital Filter**

0: No filtering applied on TWIHS inputs.

1: TWIHS input filtering is active (only in Standard and Fast modes)

Note: TWIHS digital input filtering follows a majority decision based on three samples from SDA/SCL lines at peripheral clock frequency.

- **PADFEN: PAD Filter Enable**

0: PAD analog filter is disabled.

1: PAD analog filter is enabled. (The analog filter must be enabled if High-speed mode is enabled.)

- **PADFCFG: PAD Filter Config**

See the electrical characteristics section for filter configuration details.

- **THRES: Digital Filter Threshold**

0: No filtering applied on TWIHS inputs.

1–7: Maximum pulse width of spikes to be suppressed by the input filter, defined in peripheral clock cycles.

43.7.10 TWIHS Interrupt Enable Register

Name: TWIHS_IER

Address: 0xF8028024 (0), 0xFC028024 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK

15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCL_WS	ARBLST	NACK

7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Enable**
- **RXRDY: Receive Holding Register Ready Interrupt Enable**
- **TXRDY: Transmit Holding Register Ready Interrupt Enable**
- **SVACC: Slave Access Interrupt Enable**
- **GACC: General Call Access Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **UNRE: Underrun Error Interrupt Enable**
- **NACK: Not Acknowledge Interrupt Enable**
- **ARBLST: Arbitration Lost Interrupt Enable**
- **SCL_WS: Clock Wait State Interrupt Enable**
- **EOSACC: End Of Slave Access Interrupt Enable**
- **MCACK: Master Code Acknowledge Interrupt Enable**
- **TOUT: Timeout Error Interrupt Enable**
- **PECERR: PEC Error Interrupt Enable**
- **SMBDAM: SMBus Default Address Match Interrupt Enable**
- **SMBHHM: SMBus Host Header Address Match Interrupt Enable**

43.7.11 TWIHS Interrupt Disable Register

Name: TWIHS_IDR

Address: 0xF8028028 (0), 0xFC028028 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK

15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCL_WS	ARBLST	NACK

7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Disable**
- **RXRDY: Receive Holding Register Ready Interrupt Disable**
- **TXRDY: Transmit Holding Register Ready Interrupt Disable**
- **SVACC: Slave Access Interrupt Disable**
- **GACC: General Call Access Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **UNRE: Underrun Error Interrupt Disable**
- **NACK: Not Acknowledge Interrupt Disable**
- **ARBLST: Arbitration Lost Interrupt Disable**
- **SCL_WS: Clock Wait State Interrupt Disable**
- **EOSACC: End Of Slave Access Interrupt Disable**
- **MCACK: Master Code Acknowledge Interrupt Disable**
- **TOUT: Timeout Error Interrupt Disable**
- **PECERR: PEC Error Interrupt Disable**
- **SMBDAM: SMBus Default Address Match Interrupt Disable**
- **SMBHHM: SMBus Host Header Address Match Interrupt Disable**

43.7.12 TWIHS Interrupt Mask Register

Name: TWIHS_IMR

Address: 0xF802802C (0), 0xFC02802C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK

15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCL_WS	ARBLST	NACK

7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

- **TXCOMP: Transmission Completed Interrupt Mask**
- **RXRDY: Receive Holding Register Ready Interrupt Mask**
- **TXRDY: Transmit Holding Register Ready Interrupt Mask**
- **SVACC: Slave Access Interrupt Mask**
- **GACC: General Call Access Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **UNRE: Underrun Error Interrupt Mask**
- **NACK: Not Acknowledge Interrupt Mask**
- **ARBLST: Arbitration Lost Interrupt Mask**
- **SCL_WS: Clock Wait State Interrupt Mask**
- **EOSACC: End Of Slave Access Interrupt Mask**
- **MCACK: Master Code Acknowledge Interrupt Mask**
- **TOUT: Timeout Error Interrupt Mask**
- **PECERR: PEC Error Interrupt Mask**
- **SMBDAM: SMBus Default Address Match Interrupt Mask**
- **SMBHHM: SMBus Host Header Address Match Interrupt Mask**

43.7.13 TWIHS Receive Holding Register

Name: TWIHS_RHR

Address: 0xF8028030 (0), 0xFC028030 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
RXDATA							

- **RXDATA: Master or Slave Receive Holding Data**

43.7.14 TWIHS Receive Holding Register (FIFO Enabled)

Name: TWIHS_RHR (FIFO_ENABLED)

Address: 0xF8028030 (0), 0xFC028030 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RXDATA3							
23	22	21	20	19	18	17	16
RXDATA2							
15	14	13	12	11	10	9	8
RXDATA1							
7	6	5	4	3	2	1	0
RXDATA0							

Note: If FIFO is enabled (FIFOEN bit in TWIHS_CR), refer to Section "Master Multiple Data Mode" for details.

- **RXDATA0: Master or Slave Receive Holding Data 0**
- **RXDATA1: Master or Slave Receive Holding Data 1**
- **RXDATA2: Master or Slave Receive Holding Data 2**
- **RXDATA3: Master or Slave Receive Holding Data 3**

43.7.15 TWIHS SleepWalking Matching Register

Name: TWIHS_SWMR

Address: 0xF802804C (0), 0xFC02804C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
DATAM							

23	22	21	20	19	18	17	16
–	SADR3						

15	14	13	12	11	10	9	8
–	SADR2						

7	6	5	4	3	2	1	0
–	SADR1						

This register can only be written if the WPEN bit is cleared in the TWIHS Write Protection Mode Register.

- **SADR1: Slave Address 1**

Slave address 1. The TWIHS module matches on this additional address if SADR1EN bit is enabled.

- **SADR2: Slave Address 2**

Slave address 2. The TWIHS module matches on this additional address if SADR2EN bit is enabled.

- **SADR3: Slave Address 3**

Slave address 3. The TWIHS module matches on this additional address if SADR3EN bit is enabled.

- **DATAM: Data Match**

The TWIHS module extends the SleepWalking matching process to the first received data, comparing it with DATAM if DATAMEN bit is enabled.

43.7.16 TWIHS Transmit Holding Register

Name: TWIHS_THR

Address: 0xF8028034 (0), 0xFC028034 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
TXDATA							

- **TXDATA: Master or Slave Transmit Holding Data**

43.7.17 TWIHS Transmit Holding Register (FIFO Enabled)

Name: TWIHS_THR (FIFO_ENABLED)

Address: 0xF8028034 (0), 0xFC028034 (1)

Access: Write-only

31	30	29	28	27	26	25	24
TXDATA3							
23	22	21	20	19	18	17	16
TXDATA2							
15	14	13	12	11	10	9	8
TXDATA1							
7	6	5	4	3	2	1	0
TXDATA0							

Note: If FIFO is enabled (FIFOEN bit in TWIHS_CR), refer to Section "Master Multiple Data Mode" for details.

- **TXDATA0: Master or Slave Transmit Holding Data 02**
- **TXDATA1: Master or Slave Transmit Holding Data 1**
- **TXDATA2: Master or Slave Transmit Holding Data 2**
- **TXDATA3: Master or Slave Transmit Holding Data 3**

43.7.18 TWIHS FIFO Mode Register

Name: TWIHS_FMR

Address: 0xF8028050 (0), 0xFC028050 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	RXFTHRES					

23	22	21	20	19	18	17	16
–	–	TXFTHRES					

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	RXRDYM		–	–	TXRDYM	

- **TXRDYM: Transmitter Ready Mode**

If FIFOs are enabled, the TXRDY flag (in TWIHS_SR) behaves as follows.

Value	Name	Description
0x0	ONE_DATA	TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO
0x1	TWO_DATA	TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO
0x2	FOUR_DATA	TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO

- **RXRDYM: Receiver Ready Mode**

If FIFOs are enabled, the RXRDY flag (in TWIHS_SR) behaves as follows.

Value	Name	Description
0x0	ONE_DATA	RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO
0x1	TWO_DATA	RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO
0x2	FOUR_DATA	RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO

- **TXFTHRES: Transmit FIFO Threshold**

0–16: Defines the Transmit FIFO threshold value (number of data). TXFTH flag in TWIHS_FSR will be set when Transmit FIFO goes from “above” threshold state to “equal or below” threshold state.

- **RXFTHRES: Receive FIFO Threshold**

0–16: Defines the Receive FIFO threshold value (number of data). RXFTH flag in TWIHS_FSR will be set when Receive FIFO goes from “below” threshold state to “equal or above” threshold state.

43.7.19 TWIHS FIFO Level Register

Name: TWIHS_FLR

Address: 0xF8028054 (0), 0xFC028054 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	RXFL					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TXFL					

- **TXFL: Transmit FIFO Level**

0: There is no data in the Transmit FIFO

1–16: Indicates the number of DATA in the Transmit FIFO

- **RXFL: Receive FIFO Level**

0: There is no unread data in the Receive FIFO

1–16: Indicates the number of unread DATA in the Receive FIFO

43.7.20 TWIHS FIFO Status Register

Name: TWIHS_FSR

Address: 0xF8028060 (0), 0xFC028060 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: Transmit FIFO Empty Flag (cleared on read)**

0: Transmit FIFO is not empty.

1: Transmit FIFO has been emptied since the last read of TWIHS_FSR.

- **TXFFF: Transmit FIFO Full Flag (cleared on read)**

0: Transmit FIFO is not full.

1: Transmit FIFO has been filled since the last read of TWIHS_FSR.

- **TXFTHF: Transmit FIFO Threshold Flag (cleared on read)**

0: Number of DATA in Transmit FIFO is above TXFTHRES threshold.

1: Number of DATA in Transmit FIFO has reached TXFTHRES threshold since the last read of TWIHS_FSR.

- **RXFEF: Receive FIFO Empty Flag**

0: Receive FIFO is not empty.

1: Receive FIFO has been emptied since the last read of TWIHS_FSR.

- **RXFFF: Receive FIFO Full Flag**

0: Receive FIFO is not empty.

1: Receive FIFO has been filled since the last read of TWIHS_FSR.

- **RXFTHF: Receive FIFO Threshold Flag**

0: Number of unread DATA in Receive FIFO is below RXFTHRES threshold.

1: Number of unread DATA in Receive FIFO has reached RXFTHRES threshold since the last read of TWIHS_FSR.

- **TXFPTEF: Transmit FIFO Pointer Error Flag**

0: No Transmit FIFO pointer occurred

1: Transmit FIFO pointer error occurred. Transceiver must be reset

See Section "FIFO Pointer Error" for details.

- **RXFPTEF: Receive FIFO Pointer Error Flag**

0: No Receive FIFO pointer occurred

1: Receive FIFO pointer error occurred. Receiver must be reset

See Section ”FIFO Pointer Error” for details.

43.7.21 TWIHS FIFO Interrupt Enable Register

Name: TWIHS_FIER

Address: 0xF8028064 (0), 0xFC028064 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **TXFEF: TXFEF Interrupt Enable**
- **TXFFF: TXFFF Interrupt Enable**
- **TXFTHF: TXFTHF Interrupt Enable**
- **RXFEF: RXFEF Interrupt Enable**
- **RXFFF: RXFFF Interrupt Enable**
- **RXFTHF: RXFTHF Interrupt Enable**
- **TXFPTEF: TXFPTEF Interrupt Enable**
- **RXFPTEF: RXFPTEF Interrupt Enable**

43.7.22 TWIHS FIFO Interrupt Disable Register

Name: TWIHS_FIDR

Address: 0xF8028068 (0), 0xFC028068 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **TXFEF: TXFEF Interrupt Disable**
- **TXFFF: TXFFF Interrupt Disable**
- **TXFTHF: TXFTHF Interrupt Disable**
- **RXFEF: RXFEF Interrupt Disable**
- **RXFFF: RXFFF Interrupt Disable**
- **RXFTHF: RXFTHF Interrupt Disable**
- **TXFPTEF: TXFPTEF Interrupt Disable**
- **RXFPTEF: RXFPTEF Interrupt Disable**

43.7.23 TWIHS FIFO Interrupt Mask Register

Name: TWIHS_FIMR

Address: 0xF802806C (0), 0xFC02806C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **TXFEF: TXFEF Interrupt Mask**
- **TXFFF: TXFFF Interrupt Mask**
- **TXFTHF: TXFTHF Interrupt Mask**
- **RXFEF: RXFEF Interrupt Mask**
- **RXFFF: RXFFF Interrupt Mask**
- **RXFTHF: RXFTHF Interrupt Mask**
- **TXFPTEF: TXFPTEF Interrupt Mask**
- **RXFPTEF: RXFPTEF Interrupt Mask**

43.7.24 TWIHS Write Protection Mode Register

Name: TWIHS_WPMR

Address: 0xF80280E4 (0), 0xFC0280E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

See Section 43.6.7 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x545749	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0

43.7.25 TWIHS Write Protection Status Register

Name: TWIHS_WPSR

Address: 0xF80280E8 (0), 0xFC0280E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
WPVSRC							

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the TWIHS_WPSR.

1: A write protection violation has occurred since the last read of the TWIHS_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

44. Flexible Serial Communication Controller (FLEXCOM)

44.1 Description

The Flexible Serial Communication Controller (FLEXCOM) offers several serial communication protocols that are managed by the three submodules USART, SPI, and TWI.

The Universal Synchronous Asynchronous Receiver Transceiver (USART) provides one full duplex universal synchronous asynchronous serial link. Data frame format is widely programmable (data length, parity, number of stop bits) to support a maximum of standards. The receiver implements parity error, framing error and overrun error detection. The receiver timeout enables handling variable-length frames and the transmitter timeguard facilitates communications with slow remote devices. Multidrop communications are also supported through address bit handling in reception and transmission.

The USART features three test modes: Remote Loopback, Local Loopback and Automatic Echo.

The USART supports specific operating modes providing interfaces on RS485, LIN, and SPI, with ISO7816 T = 0 or T = 1 smart card slots, and infrared transceivers. The hardware handshaking feature enables an out-of-band flow control by automatic management of the pins RTS and CTS.

The USART supports the connection to the DMA Controller, which enables data transfers to the transmitter and from the receiver. The DMAC provides chained buffer management without any intervention of the processor.

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a Shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master” which controls the data flow, while the other devices act as “slaves” which have data shifted into and out by the master. Different CPUs can take turn being masters (multiple master protocol, contrary to single master protocol where one CPU is always the master while all of the others are always slaves). One master can simultaneously shift data into multiple slaves. However, only one slave can drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK)—This control line is driven by the master and regulates the flow of the data bits. The master can transmit data at a variety of baud rates; there is one SPCK pulse for each bit that is transmitted.
- Slave Select (NSS)—This control line allows slaves to be turned on and off by hardware.

The Two-wire Interface (TWI) interconnects components on a unique two-wire bus, made up of one clock line and one data line with speeds of up to 400 kbits per second in Fast mode and up to 3.4 Mbits per second in High-speed Slave mode only, based on a byte-oriented transfer format. It can be used with any Atmel Two-wire Interface bus Serial EEPROM and I^2C-compatible devices, such as a Real-Time Clock (RTC), Dot Matrix/Graphic LCD Controller and temperature sensor. The TWI is programmable as a master or a slave with sequential or single-byte access. Multiple master capability is supported.

Arbitration of the bus is performed internally and puts the TWI in Slave mode automatically if the bus arbitration is lost.

A configurable baud rate generator permits the output data rate to be adapted to a wide range of core clock frequencies.

Table 44-1 lists the compatibility level of the Atmel Two-wire Interface in Master mode and a full I^2C compatible device.

Table 44-1. Atmel TWI Compatibility with I^2C Standard

I^2C Standard	Atmel TWI
Standard Mode Speed (100 kHz)	Supported
Fast Mode Speed (400 kHz)	Supported
High-speed Mode (Slave only, 3.4 MHz)	Supported
7- or 10-bit(1) Slave Addressing	Supported
Repeated Start (Sr) Condition	Supported
ACK and NACK Management	Supported
Input Filtering	Supported
Slope Control	Not Supported
Clock Stretching	Supported
Multi Master Capability	Supported

Notes: 1. 10-bit support in Master mode only

44.2 Embedded Characteristics

44.2.1 USART/UART Characteristics

- 32-byte Transmit and Receive FIFOs
- Programmable Baud Rate Generator
- Baud Rate can be Independent of the Processor/Peripheral Clock
- Comparison Function on Received Character
- 5-bit to 9-bit Full-duplex Synchronous or Asynchronous Serial Communications
  - 1, 1.5 or 2 Stop Bits in Asynchronous Mode or 1 or 2 Stop Bits in Synchronous Mode
  - Parity Generation and Error Detection
  - Framing Error Detection, Overrun Error Detection
  - Digital Filter on Receive Line
  - MSB- or LSB-first
  - Optional Break Generation and Detection
  - By 8 or by 16 Over-sampling Receiver Frequency
  - Optional Hardware Handshaking RTS-CTS
  - Receiver Timeout and Transmitter Timeguard
  - Optional Multidrop Mode with Address Generation and Detection
- RS485 with Driver Control Signal
- ISO7816, T = 0 or T = 1 Protocols for Interfacing with Smart Cards
  - NACK Handling, Error Counter with Repetition and Iteration Limit
- IrDA Modulation and Demodulation
  - Communication at up to 115.2 kbit/s
- SPI Mode
  - MASTER or Slave

  - Serial Clock Programmable Phase and Polarity
  - SPI Serial Clock (SCK) Frequency up to $f_{peripheral\ clock}/6$
- LIN Mode
  - Compliant with LIN 1.3 and LIN 2.0 specifications
  - Master or Slave
  - Processing of Frames with Up to 256 Data Bytes
  - Response Data Length can be Configurable or Defined Automatically by the Identifier
  - Self-synchronization in Slave Node Configuration
  - Automatic Processing and Verification of the “Synch Break” and the “Synch Field”
  - “Synch Break” Detection Even When Partially Superimposed with a Data Byte
  - Automatic Identifier Parity Calculation/Sending and Verification
  - Parity Sending and Verification Can be Disabled
  - Automatic Checksum Calculation/sending and Verification
  - Checksum Sending and Verification Can be Disabled
  - Support Both “Classic” and “Enhanced” Checksum Types
  - Full LIN Error Checking and Reporting
  - Frame Slot Mode: Master Allocates Slots to the Scheduled Frames Automatically
  - Generation of the Wakeup Signal
- Test Modes
  - Remote Loopback, Local Loopback, Automatic Echo
- Supports Connection of:
  - Two DMA Controller (DMAC) Channels
  - Offers Buffer Transfer without Processor Intervention
- Register Write Protection

44.2.2 SPI Characteristics

- 32-byte Transmit and Receive FIFOs
- Master or Slave Serial Peripheral Bus Interface
  - 8-bit to 16-bit programmable data length per chip select
  - Programmable phase and polarity per chip select
  - Programmable transfer delay between consecutive transfers and delay before SPI clock per chip select
  - Programmable delay between chip selects
- Selectable mode fault detection
- Master Mode can drive SPCK up to Peripheral Clock
- Master Mode Bit Rate can be Independent of the Processor/Peripheral Clock
- Slave mode operates on SPCK, asynchronously with core and bus clock
- Two chip selects with external decoder support allow communication with up to 3 peripherals
- Communication with Serial External Devices Supported
  - Serial memories, such as DataFlash and 3-wire EEPROMs
  - Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - External coprocessors
- Connection to DMA Channel Capabilities, Optimizing Data Transfers
  - One channel for the receiver

- One channel for the transmitter
- Register Write Protection

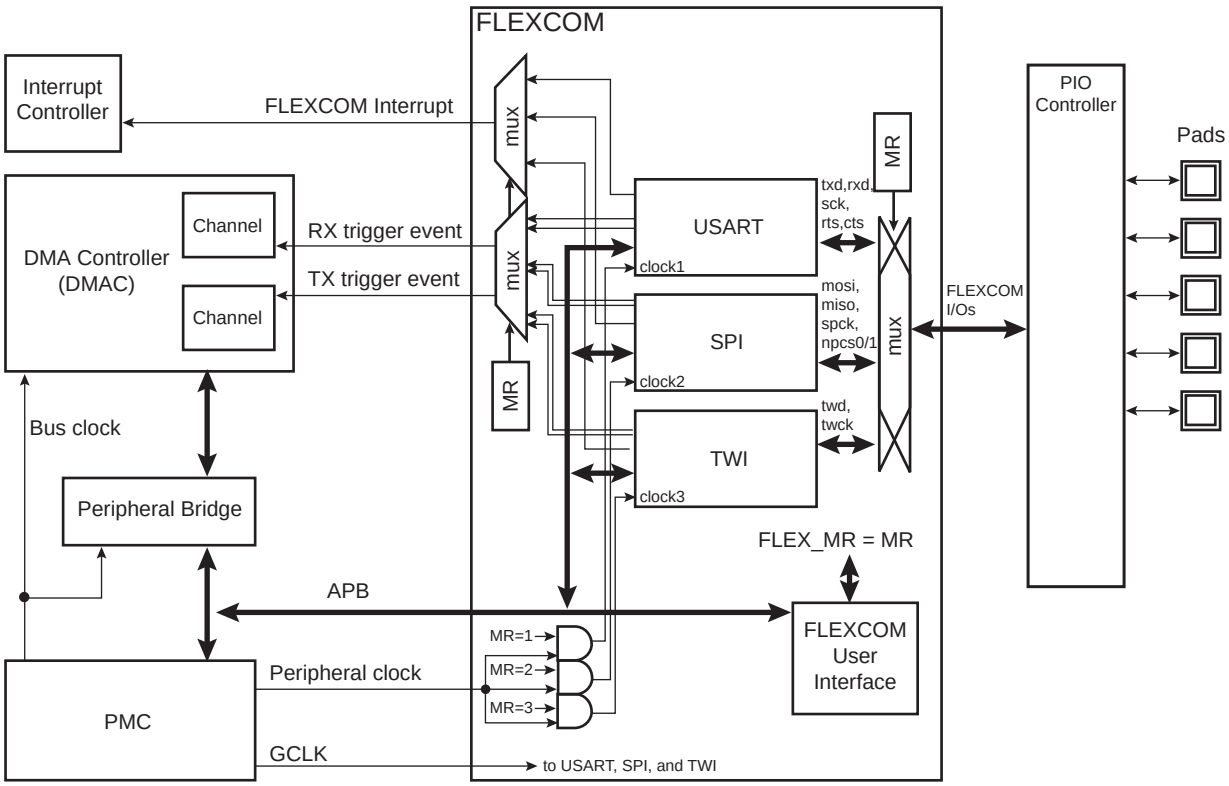
44.2.3 TWI/SMBus Characteristics

- 16-byte Transmit and Receive FIFOs
- Bit Rate: Up to 400 kbit/s in Fast Mode and 3.4 Mbit/s in High-Speed Mode (Slave Only)
- Bit Rate can be Independent of the Processor/Peripheral Clock
- SMBus support
- Compatible with Atmel Two-wire Interface Serial Memory and I^2C Compatible Devices(1)
- Master and Multi-Master Operation (Standard and Fast Mode Only)
- Slave Mode Operation (Standard, Fast and High-Speed Mode)
- One, Two or Three Bytes for Slave Address
- Sequential Read/Write Operations
- General Call Supported in Slave Mode
- Connection to DMA Controller Channels Optimizes Data Transfers
  - One Channel for the Receiver
  - One Channel for the Transmitter
- Register Write Protection

Note: 1. See Table 44-1 for details on compatibility with I^2C Standard.

44.3 Block Diagram

Figure 44-1. FLEXCOM Block Diagram



44.4 I/O Lines Description

Table 44-2. I/O Lines Description

Name	Description			Type
	USART/UART	SPI	TWI	
FLEXCOM_IO0	TXD	MOSI	TWD	I/O
FLEXCOM_IO1	RXD	MISO	TWCK	I/O
FLEXCOM_IO2	SCK	SPCK	–	I/O
FLEXCOM_IO3	CTS	NPCS0/NSS	–	I/O
FLEXCOM_IO4	RTS	NPCS1	–	O

44.5 Product Dependencies

44.5.1 I/O Lines

The pins used for interfacing the FLEXCOM are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the desired FLEXCOM pins to their peripheral function. If I/O lines of the FLEXCOM are not used by the application, they can be used for other purposes by the PIO Controller.

Table 44-3. I/O Lines

Instance	Signal	I/O Line	Peripheral
FLEXCOM0	FLEXCOM0_IO0	PB28	C
FLEXCOM0	FLEXCOM0_IO1	PB29	C
FLEXCOM0	FLEXCOM0_IO2	PB30	C
FLEXCOM0	FLEXCOM0_IO3	PB31	C
FLEXCOM0	FLEXCOM0_IO4	PC0	C
FLEXCOM1	FLEXCOM1_IO0	PA24	A
FLEXCOM1	FLEXCOM1_IO1	PA23	A
FLEXCOM1	FLEXCOM1_IO2	PA22	A
FLEXCOM1	FLEXCOM1_IO3	PA25	A
FLEXCOM1	FLEXCOM1_IO4	PA26	A
FLEXCOM2	FLEXCOM2_IO0	PA6	E
FLEXCOM2	FLEXCOM2_IO0	PD26	C
FLEXCOM2	FLEXCOM2_IO1	PA7	E
FLEXCOM2	FLEXCOM2_IO1	PD27	C
FLEXCOM2	FLEXCOM2_IO2	PA8	E
FLEXCOM2	FLEXCOM2_IO2	PD28	C
FLEXCOM2	FLEXCOM2_IO3	PA9	E
FLEXCOM2	FLEXCOM2_IO3	PD29	C
FLEXCOM2	FLEXCOM2_IO4	PA10	E
FLEXCOM2	FLEXCOM2_IO4	PD30	C
FLEXCOM3	FLEXCOM3_IO0	PA15	E

Table 44-3. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
FLEXCOM3	FLEXCOM3_IO0	PB23	E
FLEXCOM3	FLEXCOM3_IO0	PC20	E
FLEXCOM3	FLEXCOM3_IO1	PA13	E
FLEXCOM3	FLEXCOM3_IO1	PB22	E
FLEXCOM3	FLEXCOM3_IO1	PC19	E
FLEXCOM3	FLEXCOM3_IO2	PA14	E
FLEXCOM3	FLEXCOM3_IO2	PB21	E
FLEXCOM3	FLEXCOM3_IO2	PC18	E
FLEXCOM3	FLEXCOM3_IO3	PA16	E
FLEXCOM3	FLEXCOM3_IO3	PB24	E
FLEXCOM3	FLEXCOM3_IO3	PC21	E
FLEXCOM3	FLEXCOM3_IO4	PA17	E
FLEXCOM3	FLEXCOM3_IO4	PB25	E
FLEXCOM3	FLEXCOM3_IO4	PC22	E
FLEXCOM4	FLEXCOM4_IO0	PC28	B
FLEXCOM4	FLEXCOM4_IO0	PD12	B
FLEXCOM4	FLEXCOM4_IO0	PD21	C
FLEXCOM4	FLEXCOM4_IO1	PC29	B
FLEXCOM4	FLEXCOM4_IO1	PD13	B
FLEXCOM4	FLEXCOM4_IO1	PD22	C
FLEXCOM4	FLEXCOM4_IO2	PC30	B
FLEXCOM4	FLEXCOM4_IO2	PD14	B
FLEXCOM4	FLEXCOM4_IO2	PD23	C
FLEXCOM4	FLEXCOM4_IO3	PC31	B
FLEXCOM4	FLEXCOM4_IO3	PD15	B
FLEXCOM4	FLEXCOM4_IO3	PD24	C
FLEXCOM4	FLEXCOM4_IO4	PD0	B
FLEXCOM4	FLEXCOM4_IO4	PD16	B
FLEXCOM4	FLEXCOM4_IO4	PD25	C

44.5.2 Power Management

The peripheral clock is not continuously provided to the FLEXCOM. The programmer must first enable the FLEXCOM Clock in the Power Management Controller (PMC) before using the USART or SPI or TWI.

In SleepWalking mode (asynchronous partial wakeup), the PMC must be configured to enable SleepWalking for the FLEXCOM in the Sleepwalking Enable Register (PMC_SLPWK_ER). The peripheral clock can be automatically provided to the FLEXCOM, depending on the instructions (requests) provided by the FLEXCOM to the PMC.

44.5.3 Interrupt Sources

The FLEXCOM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the FLEXCOM interrupt requires the Interrupt Controller to be programmed first.

Table 44-4. Peripheral IDs

Instance	ID
FLEXCOM0	19
FLEXCOM1	20
FLEXCOM2	21
FLEXCOM3	22
FLEXCOM4	23

44.6 Register Accesses

Register accesses supports 8-bit, 16-bit and 32-bit accesses which means that only an 8-bit part of a 32-bit register can be written in one access for instance. For this the access must be done with the right size at the right address.

8-bit, 16-bit and 32-bit accesses are supported for register accesses however a field in a register cannot be partially written (e.g., if a field is bigger than 8 bits, the whole field must be written).

This feature helps avoiding a read-modify-write process if only a small part of the register is to be modified.

44.7 USART Functional Description

44.7.1 Baud Rate Generator

The baud rate generator provides the bit period clock named “baud rate clock” to both the receiver and the transmitter.

Configuring the USCLKS field in FLEX_US_MR selects the baud rate generator clock from one of the following sources:

- the peripheral clock
- a division of the peripheral clock, the divider being product dependent, but generally set to 8
- a fully programmable generic clock (GCLK) provided by PMC and independent of processor/peripheral clock
- the external clock, available on the SCK pin

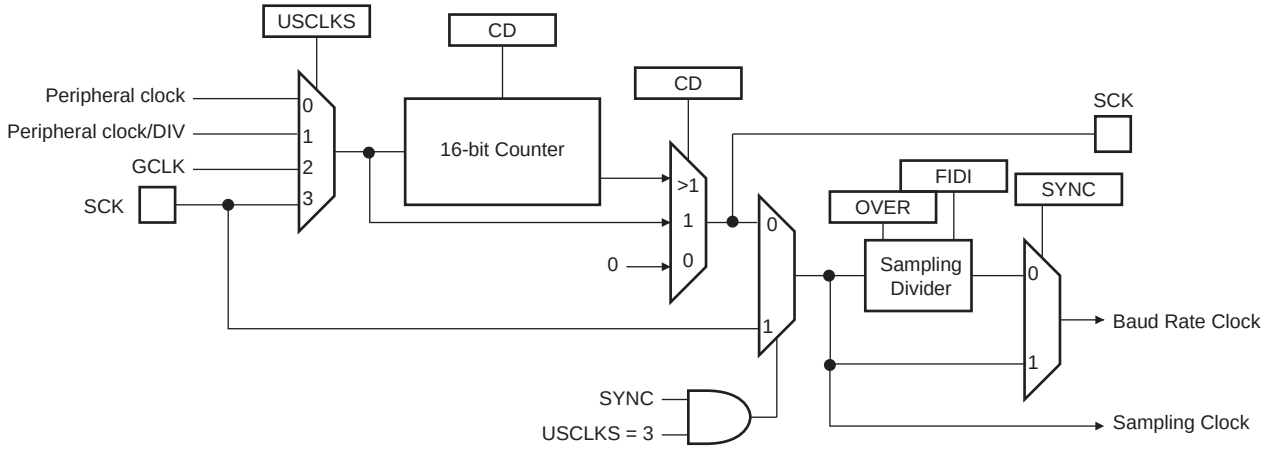
The baud rate generator is based upon a 16-bit divider, which is programmed with the CD field of the Baud Rate Generator Register (FLEX_US_BRGR). If a zero is written to CD, the baud rate generator does not generate any clock. If a one is written to CD, the divider is bypassed and becomes inactive.

If the external SCK clock is selected, the duration of the low and high levels of the signal provided on the SCK pin must be longer than a peripheral clock period. The frequency of the signal provided on SCK must be at least three times lower than peripheral clock in USART mode (field USART_MODE differs from 0xE or 0xF) or six times lower in SPI mode (field USART_MODE equals 0xE or 0xF).

If GCLK is selected, the baud rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the USART transfer. The GCLK frequency must be at least three times lower than peripheral clock frequency.

If GCLK is selected (USCLKS = 2) and the SCK pin is driven (CLKO = 1), the CD field must be greater than 1.

Figure 44-2. Baud Rate Generator



44.7.1.1 Baud Rate in Asynchronous Mode

If the USART is programmed to operate in Asynchronous mode, the selected clock is first divided by CD, which is field-programmed in FLEX_US_BRGR. The resulting clock is provided to the receiver as a sampling clock and then divided by 16 or 8, depending on the programming of the OVER bit in FLEX_US_MR.

If OVER is set, the receiver sampling is eight times higher than the baud rate clock. If OVER is cleared, the sampling is performed at 16 times the baud rate clock.

The baud rate is calculated as per the following formula:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{(8(2-OVER)CD)}$$

This gives a maximum baud rate of peripheral clock divided by 8, assuming that peripheral clock is the highest possible clock and that the OVER bit is set.

Baud Rate Calculation Example

Table 44-5 shows calculations of CD to obtain a baud rate at 38,400 bit/s for different source clock frequencies. This table also shows the actual resulting baud rate and the error.

Table 44-5. Baud Rate Example (OVER = 0)

Source Clock (MHz)	Expected Baud Rate (bit/s)	Calculation Result	CD	Actual Baud Rate (bit/s)	Error
3,686,400	38,400	6.00	6	38,400.00	0.00%
4,915,200	38,400	8.00	8	38,400.00	0.00%
5,000,000	38,400	8.14	8	39,062.50	1.70%
7,372,800	38,400	12.00	12	38,400.00	0.00%
8,000,000	38,400	13.02	13	38,461.54	0.16%
12,000,000	38,400	19.53	20	37,500.00	2.40%
12,288,000	38,400	20.00	20	38,400.00	0.00%
14,318,180	38,400	23.30	23	38,908.10	1.31%
14,745,600	38,400	24.00	24	38,400.00	0.00%
18,432,000	38,400	30.00	30	38,400.00	0.00%
24,000,000	38,400	39.06	39	38,461.54	0.16%

Table 44-5. Baud Rate Example (OVER = 0) (Continued)

Source Clock (MHz)	Expected Baud Rate (bit/s)	Calculation Result	CD	Actual Baud Rate (bit/s)	Error
24,576,000	38,400	40.00	40	38,400.00	0.00%
25,000,000	38,400	40.69	40	38,109.76	0.76%
32,000,000	38,400	52.08	52	38,461.54	0.16%
32,768,000	38,400	53.33	53	38,641.51	0.63%
33,000,000	38,400	53.71	54	38,194.44	0.54%
40,000,000	38,400	65.10	65	38,461.54	0.16%
50,000,000	38,400	81.38	81	38,580.25	0.47%

The baud rate is calculated with the following formula:

Baud rate = MCK / CD × 16

The baud rate error is calculated with the following formula. It is not recommended to work with an error higher than 5%.

$$Error = 1 - \left(\frac{\text{Expected Baud Rate}}{\text{Actual Baud Rate}}\right)$$

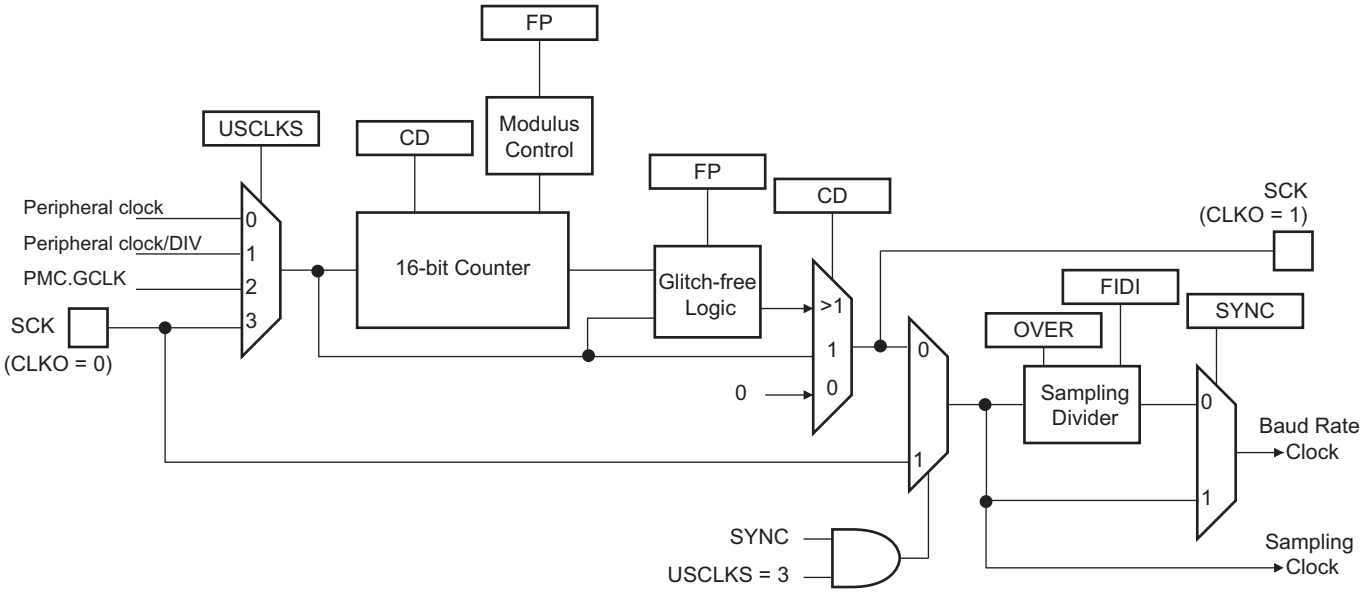
44.7.1.2 Fractional Baud Rate in Asynchronous Mode

The baud rate generator previously defined is subject to the following limitation: the output frequency changes by only integer multiples of the reference frequency. An approach to this problem is to integrate a fractional N clock generator that has a high resolution. The generator architecture is modified to obtain baud rate changes by a fraction of the reference source clock. This fractional part is programmed with the FP field in FLEX_US_BRGR. If FP is not 0, the fractional part is activated. The resolution is one eighth of the clock divider. The fractional baud rate is calculated using the following formula:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{\left(8(2 - OVER)\left(CD + \frac{FP}{8}\right)\right)}$$

The modified architecture is presented in Figure 44-3.

Figure 44-3. Fractional Baud Rate Generator



Warning: When the value of field FP is greater than 0, the SCK (oversampling clock) generates non-constant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of the CD field.

44.7.1.3 Baud Rate in Synchronous Mode or SPI Mode

If the USART is programmed to operate in Synchronous mode, the selected clock is simply divided by the CD field in FLEX_US_BRGR:

$$\text{Baud rate} = \frac{\text{Selected Clock}}{CD}$$

In Synchronous mode, if the external clock is selected (USCLKS = 3) and CLKO = 0 (Slave mode), the clock is provided directly by the signal on the USART SCK pin. No division is active. The value written in FLEX_US_BRGR has no effect. The external clock frequency must be at least three times lower than the system clock. In Synchronous mode master (USCLKS = 0 or 1, CLKO = 1), the receive part limits the SCK maximum frequency to $f_{peripheral\ clock}/3$ in USART mode, or $f_{peripheral\ clock}/6$ in SPI mode.

When either the external clock SCK or the internal clock divided (peripheral clock/DIV or GCLK) is selected, the value programmed in CD must be even if the user has to ensure a 50:50 mark/space ratio on the SCK pin. If the peripheral clock is selected, the baud rate generator ensures a 50:50 duty cycle on the SCK pin, even if the value programmed in CD is odd.

44.7.1.4 Baud Rate in ISO 7816 Mode

The ISO7816 specification defines the bit rate with the following formula:

$$B = \frac{Di}{Fi} \times f$$

where:

- B is the bit rate
- Di is the bit-rate adjustment factor
- Fi is the clock frequency division factor
- f is the ISO7816 clock frequency (Hz)

Di is a binary value encoded on a 4-bit field, named DI, as represented in Table 44-6.

Table 44-6. Binary and Decimal Values for Di

DI field	0001	0010	0011	0100	0101	0110	1000	1001
Di (decimal)	1	2	4	8	16	32	12	20

Fi is a binary value encoded on a 4-bit field, named FI, as represented in Table 44-7.

Table 44-7. Binary and Decimal Values for Fi

FI field	0000	0001	0010	0011	0100	0101	0110	1001	1010	1011	1100	1101
Fi (decimal)	372	372	558	744	1116	1488	1860	512	768	1024	1536	2048

Table 44-8 shows the resulting Fi/Di Ratio, which is the ratio between the ISO7816 clock and the baud rate clock.

Table 44-8. Possible Values for the Fi/Di Ratio

Fi/Di	372	558	744	1116	1488	1806	512	768	1024	1536	2048
1	372	558	744	1116	1488	1860	512	768	1024	1536	2048
2	186	279	372	558	744	930	256	384	512	768	1024
4	93	139.5	186	279	372	465	128	192	256	384	512
8	46.5	69.75	93	139.5	186	232.5	64	96	128	192	256
16	23.25	34.87	46.5	69.75	93	116.2	32	48	64	96	128
32	11.62	17.43	23.25	34.87	46.5	58.13	16	24	32	48	64
12	31	46.5	62	93	124	155	42.66	64	85.33	128	170.6
20	18.6	27.9	37.2	55.8	74.4	93	25.6	38.4	51.2	76.8	102.4

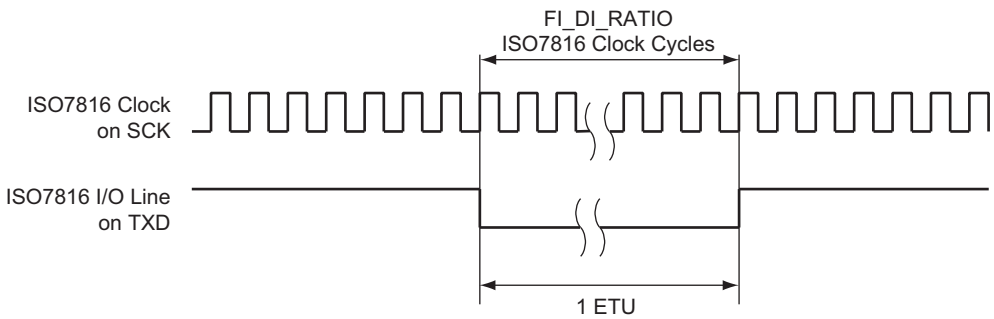
If the USART is configured in ISO7816 mode, the clock selected by the USCLKS field in FLEX_US_MR is first divided by the value programmed in field CD field in FLEX_US_BRGR. The resulting clock can be provided to the SCK pin to feed the smart card clock inputs. This means that the CLKO bit can be set in FLEX_US_MR.

This clock is then divided by the value programmed in the FI_DI_RATIO field in the FI DI Ratio Register (FLEX_US_FIDI). This is performed by the Sampling Divider, which performs a division by up to 65535 in ISO7816 mode. The non-integer values of the Fi/Di Ratio are not supported and the user must program the FI_DI_RATIO field to a value as close as possible to the expected value.

The FI_DI_RATIO field resets to the value 0x174 (372 in decimal) and is the most common divider between the ISO7816 clock and the bit rate (Fi = 372, Di = 1).

Figure 44-4 shows the relation between the Elementary Time Unit, corresponding to a bit time, and the ISO 7816 clock.

Figure 44-4. Elementary Time Unit (ETU)



44.7.2 Receiver and Transmitter Control

After reset, the receiver is disabled. The user must enable the receiver by setting the RXEN bit in the USART Control Register (FLEX_US_CR). However, the receiver registers can be programmed before the receiver clock is enabled.

After reset, the transmitter is disabled. The user must enable it by setting the TXEN bit in FLEX_US_CR. However, the transmitter registers can be programmed before being enabled.

The receiver and the transmitter can be enabled together or independently.

At any time, the software can perform a reset on the receiver or the transmitter of the USART by setting the corresponding bit, RSTRX and RSTTX respectively, in FLEX_US_CR. The software resets clear the status flag and reset internal state machines but the user interface configuration registers hold the value configured prior to software reset. Regardless of what the receiver or the transmitter is performing, the communication is immediately stopped.

The user can also independently disable the receiver or the transmitter by setting RXDIS and TXDIS respectively in FLEX_US_CR. If the receiver is disabled during a character reception, the USART waits until the end of reception of the current character, then the reception is stopped. If the transmitter is disabled while it is operating, the USART waits the end of transmission of both the current character and character being stored in the USART Transmit Holding Register (FLEX_US_THR). If a timeguard is programmed, it is handled normally.

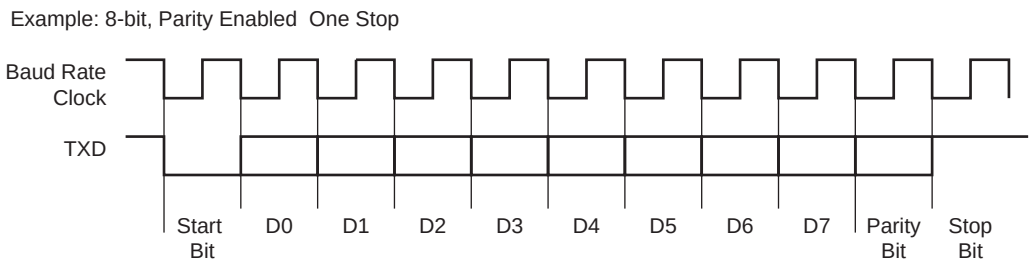
44.7.3 Synchronous and Asynchronous Modes

44.7.3.1 Transmitter Operations

The transmitter performs the same in both Synchronous and Asynchronous operating modes (SYNC = 0 or SYNC = 1). One start bit, up to 9 data bits, 1 optional parity bit and up to 2 stop bits are successively shifted out on the TXD pin at each falling edge of the programmed serial clock.

The number of data bits is selected by the CHRL field and the MODE 9 bit in FLEX_US_MR. Nine bits are selected by setting the MODE 9 bit regardless of the CHRL field. The parity bit is set according to the PAR field in FLEX_US_MR. The even, odd, space, marked or none parity bit can be configured. The MSBF bit in FLEX_US_MR configures which data bit is sent first. If written to 1, the most significant bit is sent first. If written to 0, the less significant bit is sent first. The number of stop bits is selected by the NBSTOP field in FLEX_US_MR. The 1.5 stop bit is supported in Asynchronous mode only.

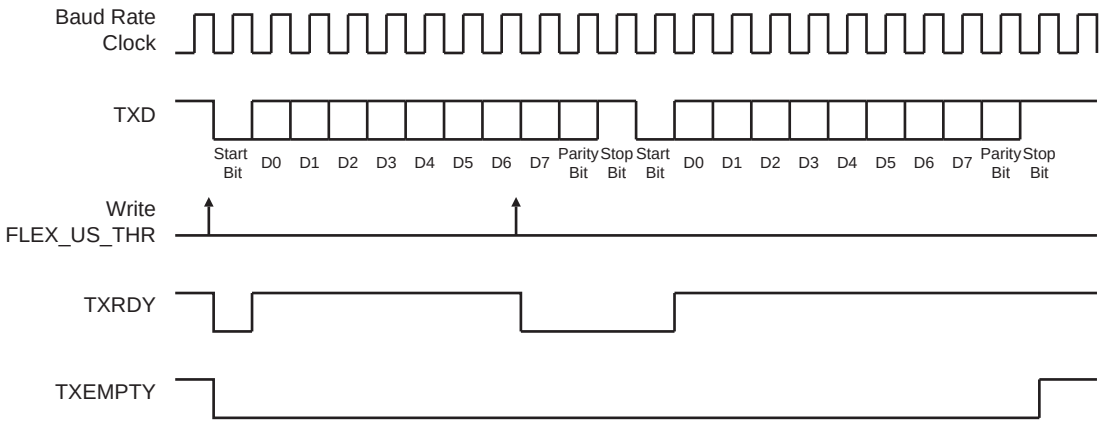
Figure 44-5. Character Transmit



The characters are sent by writing in FLEX_US_THR. The transmitter reports two status bits in the USART Channel Status Register (FLEX_US_CSR): TXRDY (Transmitter Ready), which indicates that FLEX_US_THR is empty and TXEMPTY, which indicates that all the characters written in FLEX_US_THR have been processed. When the current character processing is completed, the last character written in FLEX_US_THR is transferred into the Shift register of the transmitter and FLEX_US_THR becomes empty, thus TXRDY rises.

Both TXRDY and TXEMPTY bits are low when the transmitter is disabled. Writing a character in FLEX_US_THR while TXRDY is low has no effect and the written character is lost.

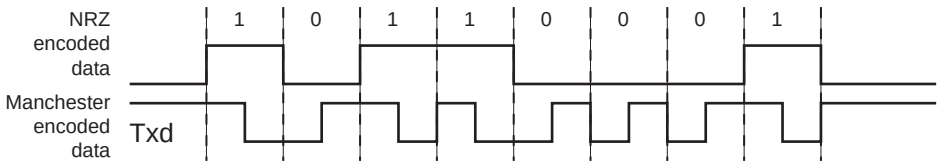
Figure 44-6. Transmitter Status



44.7.3.2 Manchester Encoder

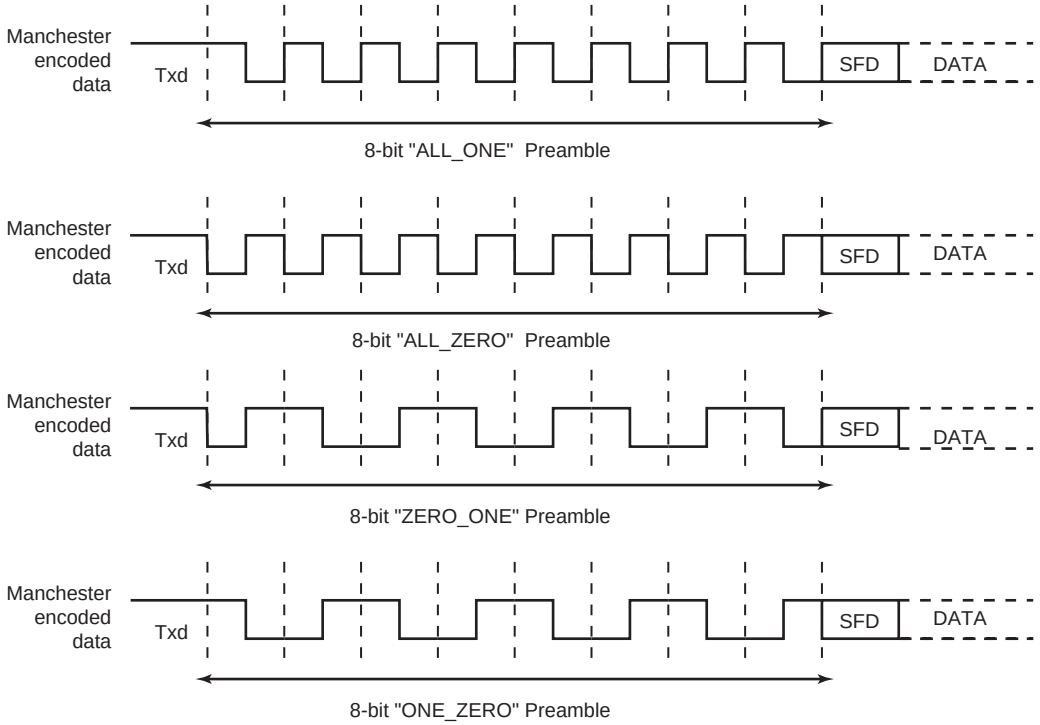
When the Manchester encoder is in use, characters transmitted through the USART are encoded based on biphase Manchester II format. To enable this mode, set the MAN bit in FLEX_US_MR to 1. Depending on polarity configuration, a logic level (zero or one), is transmitted as a coded signal one-to-zero or zero-to-one. Thus, a transition always occurs at the midpoint of each bit time. It consumes more bandwidth than the original NRZ signal (2x) but the receiver has more error control since the expected input must show a change at the center of a bit cell. An example of Manchester encoded sequence is: the byte 0xB1 or 10110001 encodes to 10 01 10 10 01 01 01 10, assuming the default polarity of the encoder. Figure 44-7 illustrates this coding scheme.

Figure 44-7. NRZ to Manchester Encoding



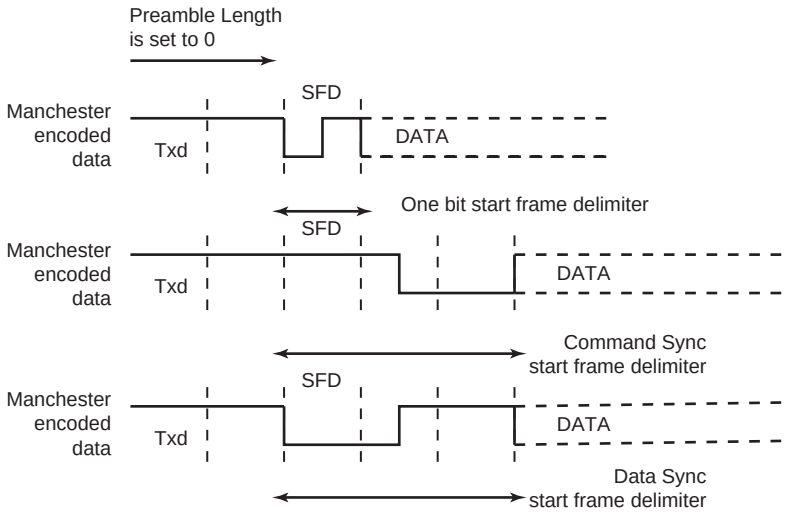
The Manchester encoded character can also be encapsulated by adding both a configurable preamble and a start frame delimiter pattern. Depending on the configuration, the preamble is a training sequence, composed of a predefined pattern with a programmable length from 1 to 15 bit times. If the preamble length is set to 0, the preamble waveform is not generated prior to any character. The preamble pattern is chosen among the following sequences: ALL_ONE, ALL_ZERO, ONE_ZERO or ZERO_ONE, writing the TX_PP field in FLEX_US_MAN register. The TX_PL field is used to configure the preamble length. Figure 44-8 illustrates and defines the valid patterns. To improve flexibility, the encoding scheme can be configured using the TX_MPOL bit in the FLEX_US_MAN register. If the TX_MPOL bit is set to zero (default), a logic zero is encoded with a zero-to-one transition and a logic one is encoded with a one-to-zero transition. If the TX_MPOL bit is set to one, a logic one is encoded with a one-to-zero transition and a logic zero is encoded with a zero-to-one transition.

Figure 44-8. Preamble Patterns, Default Polarity Assumed



A start frame delimiter is to be configured using the ONEBIT bit in FLEX_US_MR. It consists of a user-defined pattern that indicates the beginning of a valid data. Figure 44-9 illustrates these patterns. If the start frame delimiter, also known as the start bit, is one bit, (ONEBIT = 1), a logic zero is Manchester encoded and indicates that a new character is being sent serially on the line. If the start frame delimiter is a synchronization pattern also referred to as sync (ONEBIT = 0), a sequence of three bit times is sent serially on the line to indicate the start of a new character. The sync waveform is in itself an invalid Manchester waveform as the transition occurs at the middle of the second bit time. Two distinct sync patterns are used: the command sync and the data sync. The command sync has a logic one level for one and a half bit times, then a transition to logic zero for the second one and a half bit times. If the MODSYNC bit in FLEX_US_MR is set to 1, the next character is a command. If it is set to 0, the next character is a data. When direct memory access is used, the MODSYNC bit can be immediately updated with a modified character located in memory. To enable this mode, VAR_SYNC bit in FLEX_US_MR must be set. In this case, the MODSYNC bit in FLEX_US_MR is bypassed and the sync configuration is held in the TXSYNH in FLEX_US_THR. The USART character format is modified and includes sync information.

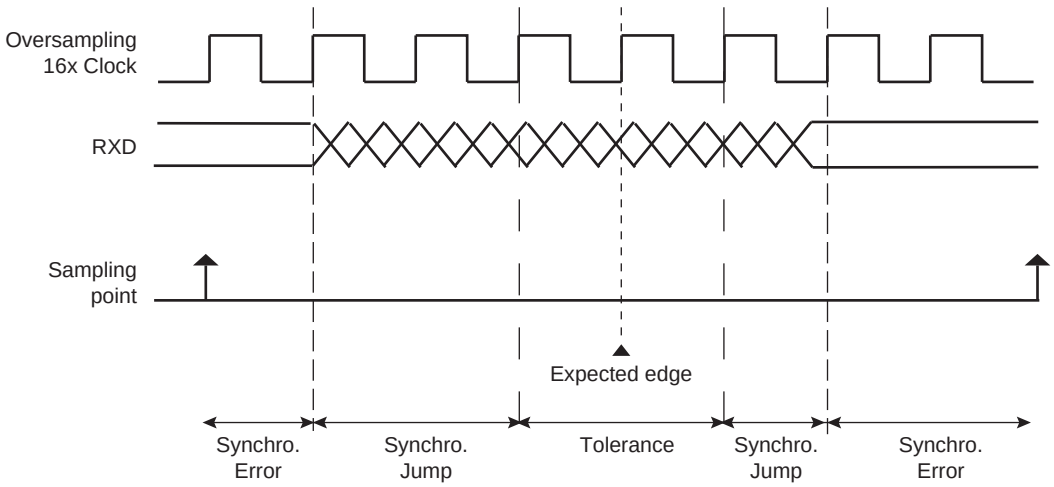
Figure 44-9. Start Frame Delimiter



Drift Compensation

Drift compensation is available only in 16X Oversampling mode. An hardware recovery system allows a larger clock drift. To enable the hardware system, the bit in the USART_MAN register must be set. If the RXD edge is one 16X clock cycle from the expected edge, this is considered as normal jitter and no corrective actions is taken. If the RXD event is between 4 and 2 clock cycles before the expected edge, then the current period is shortened by one clock cycle. If the RXD event is between 2 and 3 clock cycles after the expected edge, then the current period is lengthened by one clock cycle. These intervals are considered to be drift and so corrective actions are automatically taken.

Figure 44-10. Bit Resynchronization



44.7.3.3 Asynchronous Receiver

If the USART is programmed in Asynchronous operating mode (SYNC = 0), the receiver oversamples the RXD input line. The oversampling is either 16 or 8 times the baud rate clock, depending on the OVER bit in FLEX_US_MR.

The receiver samples the RXD line. If the line is sampled during one half of a bit time to 0, a start bit is detected and data, parity and stop bits are successively sampled on the bit rate clock.

If the oversampling is 16 (OVER = 0), a start is detected at the eighth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 16 oversampling clock cycles. If the oversampling is 8 (OVER = 1), a start bit is detected at the fourth sample to 0. Data bits, parity bit and stop bit are assumed to have a duration corresponding to 8 oversampling clock cycles.

The number of data bits, first bit sent and Parity mode are selected by the same fields and bits as the transmitter, i.e., respectively CHRL, MODE9, MSBF and PAR. For the synchronization mechanism **only**, the number of stop bits has no effect on the receiver as it considers only one stop bit, regardless of the NBSTOP field, so that resynchronization between the receiver and the transmitter can occur. Moreover, as soon as the stop bit is sampled, the receiver starts looking for a new start bit so that resynchronization can also be accomplished when the transmitter is operating with one stop bit.

Figure 44-11 and Figure 44-12 illustrate start detection and character reception when USART operates in Asynchronous mode.

Figure 44-11. Asynchronous Start Detection

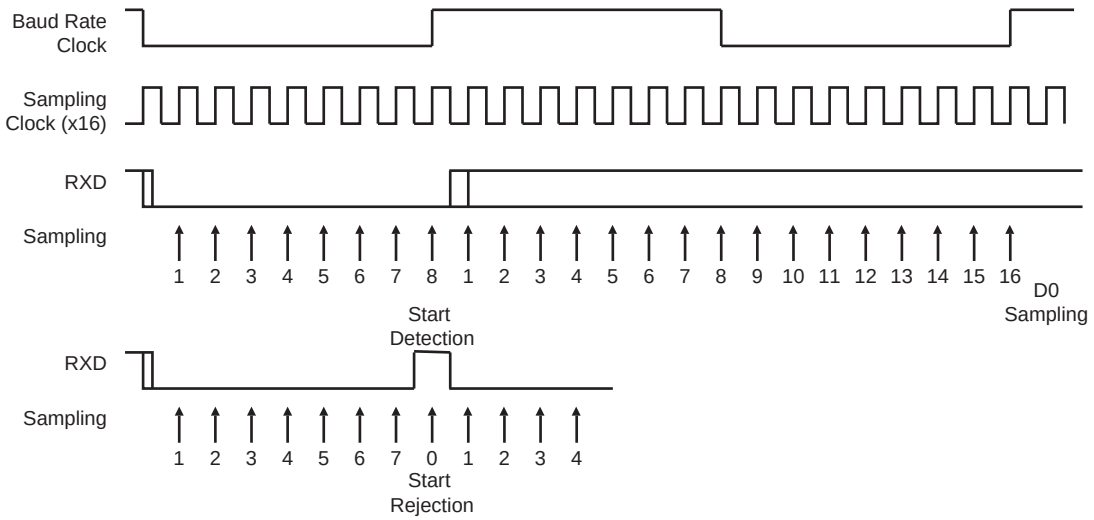
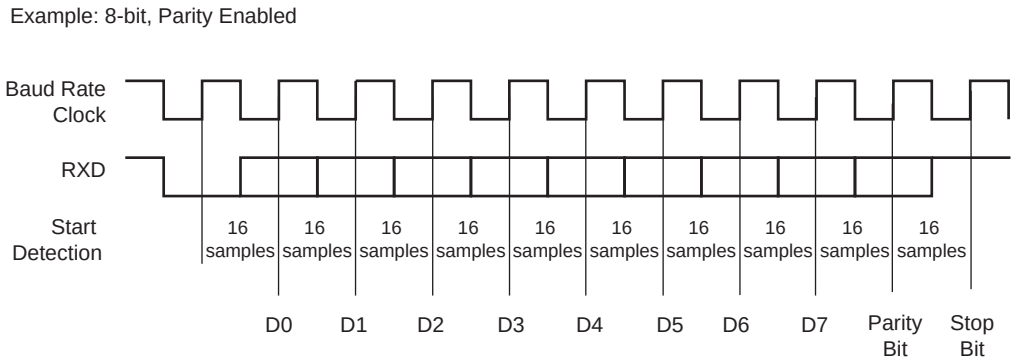


Figure 44-12. Asynchronous Character Reception



44.7.3.4 Manchester Decoder

When the MAN bit in FLEX_US_MR is set, the Manchester decoder is enabled. The decoder performs both preamble and start frame delimiter detection. One input line is dedicated to Manchester encoded input data.

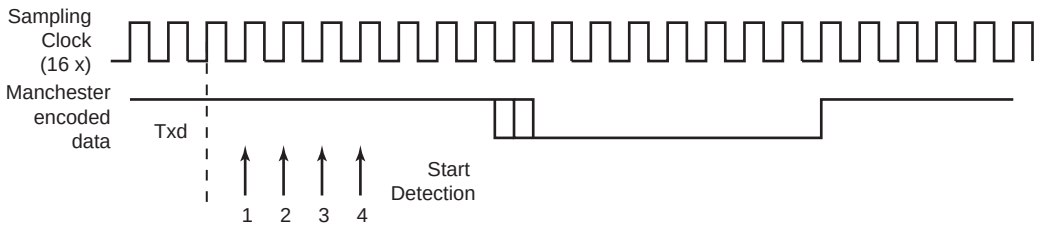
An optional preamble sequence can be defined. Its length is user-defined and totally independent of the transmitter side. Use RX_PL in FLEX_US_MAN register to configure the length of the preamble sequence. If the length is set to 0, no preamble is detected and the function is disabled. In addition, the polarity of the input stream

is programmable with RX_MPOL bit in FLEX_US_MAN register. Depending on the desired application the preamble pattern matching is to be defined via the RX_PP field in FLEX_US_MAN. See Figure 44-8 for available preamble patterns.

Unlike preamble, the start frame delimiter is shared between Manchester Encoder and Decoder. So, if ONEBIT bit = 1, only a zero encoded Manchester can be detected as a valid start frame delimiter. If ONEBIT = 0, only a sync pattern is detected as a valid start frame delimiter. Decoder operates by detecting transition on incoming stream. If RXD is sampled during one quarter of a bit time to zero, a start bit is detected. See Figure 44-13. The sample pulse rejection mechanism applies.

The RXIDLEV bit in FLEX_US_MAN informs the USART of the receiver line idle state value (receiver line inactive). The user must define RXIDLEV to ensure reliable synchronization. By default, RXIDLEV is set to one (receiver line is at level 1 when there is no activity).

Figure 44-13. Asynchronous Start Bit Detection



The receiver is activated and starts preamble and frame delimiter detection, sampling the data at one quarter and then three quarters. If a valid preamble pattern or start frame delimiter is detected, the receiver continues decoding with the same synchronization. If the stream does not match a valid pattern or a valid start frame delimiter, the receiver resynchronizes on the next valid edge.The minimum time threshold to estimate the bit value is three quarters of a bit time.

If a valid preamble (if used) followed with a valid start frame delimiter is detected, the incoming stream is decoded into NRZ data and passed to USART for processing. Figure 44-14 illustrates Manchester pattern mismatch. When incoming data stream is passed to the USART, the receiver is also able to detect Manchester code violation. A code violation is a lack of transition in the middle of a bit cell. In this case, the MANE flag in FLEX_US_CSR is raised. It is cleared by writing a one to the RSTSTA bit in FLEX_US_CR. See Figure 44-15 for an example of Manchester error detection during the data phase.

Figure 44-14. Preamble Pattern Mismatch

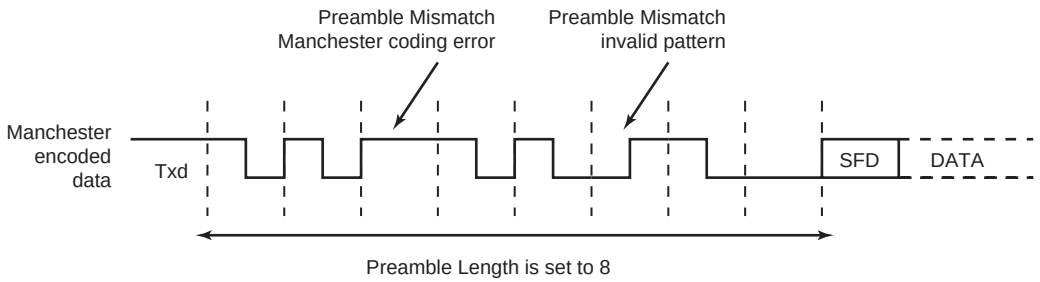
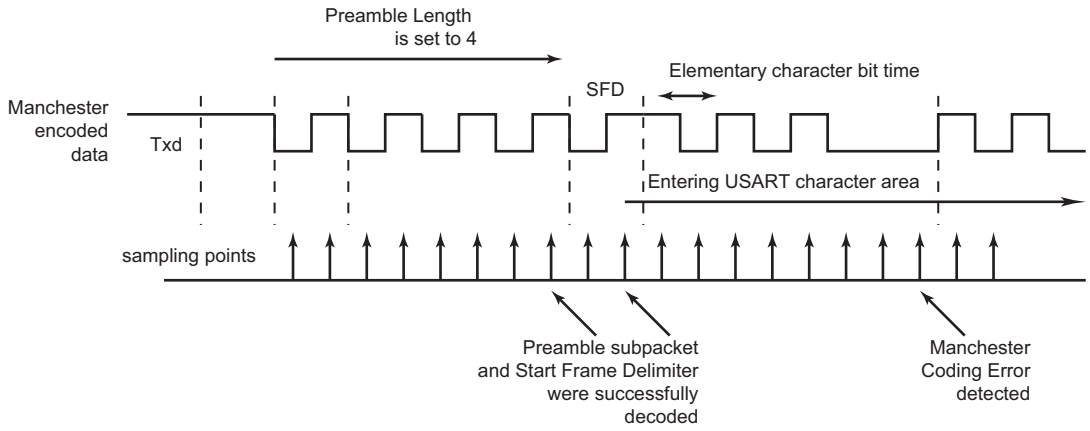


Figure 44-15. Manchester Error Flag



When the start frame delimiter is a sync pattern (ONEBIT = 0), both command and data delimiter are supported. If a valid sync is detected, the received character is written as RXCHR field in the Receive Holding Register (FLEX_US_RHR) and the RXSYNH is updated. RXCHR is set to 1 when the received character is a command, and it is set to 0 if the received character is a data. This mechanism alleviates and simplifies the direct memory access as the character contains its own sync field in the same register.

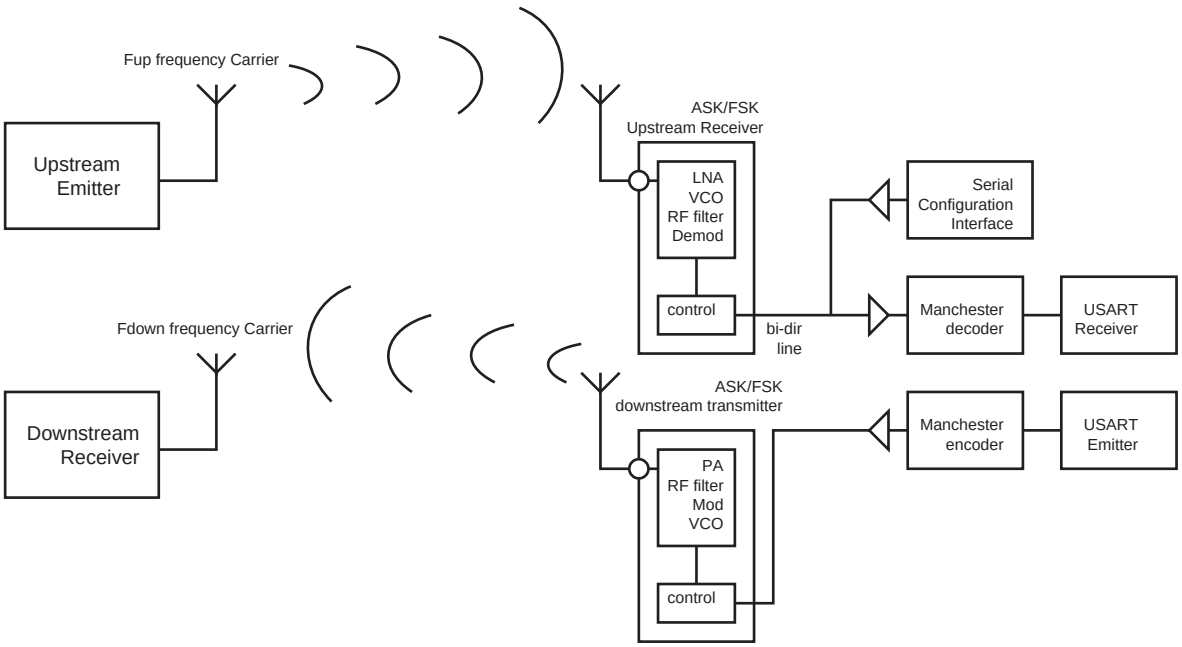
As the decoder is setup to be used in Unipolar mode, the first bit of the frame has to be a zero-to-one transition.

44.7.3.5 Radio Interface: Manchester Encoded USART Application

This section describes low data rate RF transmission systems and their integration with a Manchester encoded USART. These systems are based on transmitter and receiver ICs that support ASK and FSK modulation schemes.

The goal is to perform full duplex radio transmission of characters using two different frequency carriers. See the configuration in Figure 44-16.

Figure 44-16. Manchester Encoded Characters RF Transmission



The USART peripheral is configured as a Manchester encoder/decoder. Looking at the downstream communication channel, Manchester encoded characters are serially sent to the RF transmitter. This may also include a user defined preamble and a start frame delimiter. Mostly, preamble is used in the RF receiver to distinguish between a valid data from a transmitter and signals due to noise. The Manchester stream is then modulated. See Figure 44-17 for an example of ASK modulation scheme. When a logic one is sent to the ASK modulator, the power amplifier, referred to as PA, is enabled and transmits an RF signal at downstream frequency. When a logic zero is transmitted, the RF signal is turned off. If the FSK modulator is activated, two different frequencies are used to transmit data. When a logic 1 is sent, the modulator outputs an RF signal at frequency F0 and switches to F1 if the data sent is a 0. See Figure 44-18.

From the receiver side, another carrier frequency is used. The RF receiver performs a bit check operation examining demodulated data stream. If a valid pattern is detected, the receiver switches to Receiving mode. The demodulated stream is sent to the Manchester decoder. Because of bit checking inside RF IC, the data transferred to the microcontroller is reduced by a user-defined number of bits. The Manchester preamble length is to be defined in accordance with the RF IC configuration.

Figure 44-17. ASK Modulator Output

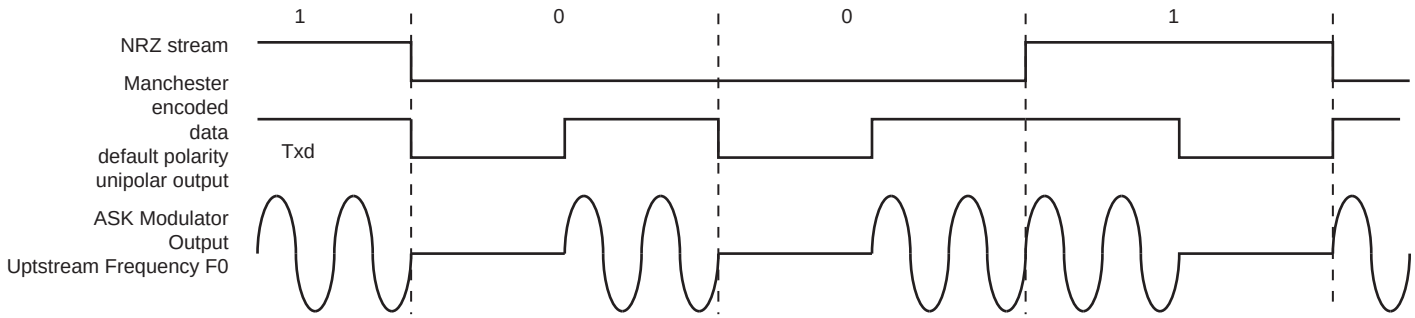
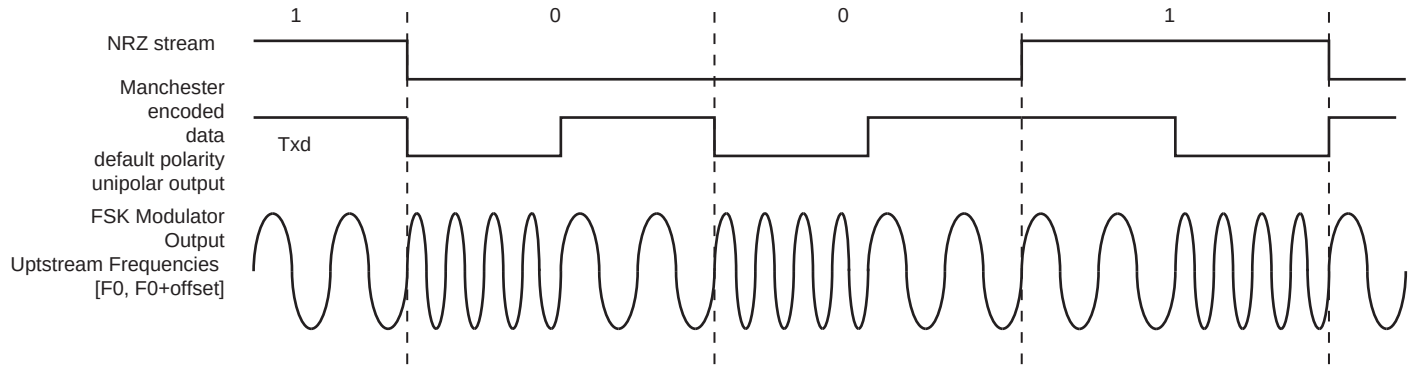


Figure 44-18. FSK Modulator Output



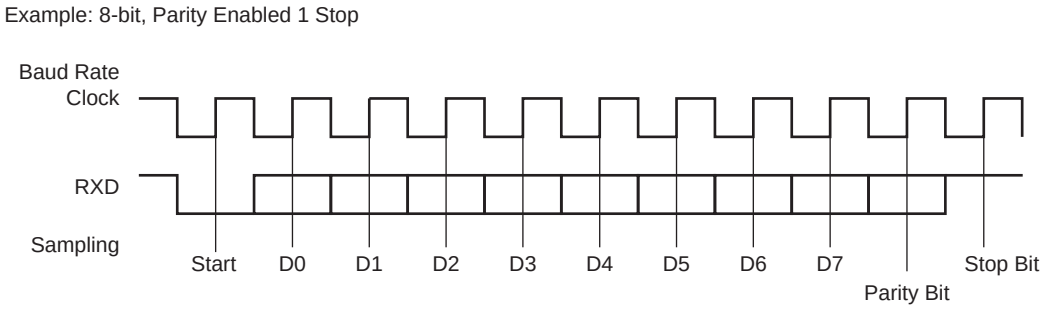
44.7.3.6 Synchronous Receiver

In Synchronous mode (SYNC = 1), the receiver samples the RXD signal on each rising edge of the baud rate clock. If a low level is detected, it is considered as a start. All data bits, the parity bit and the stop bits are sampled and the receiver waits for the next start bit. Synchronous mode operations provide a high-speed transfer capability.

Configuration fields and bits are the same as in Asynchronous mode.

Figure 44-19 illustrates a character reception in Synchronous mode.

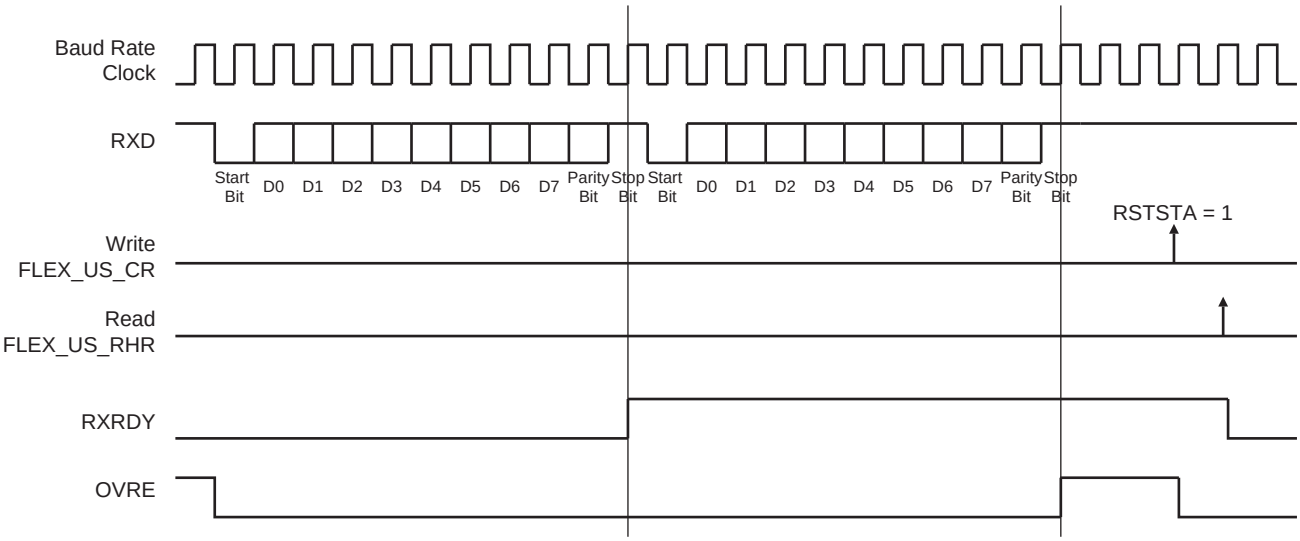
Figure 44-19. Synchronous Mode Character Reception



44.7.3.7 Receiver Operations

When a character reception is completed, it is transferred to the Receive Holding Register (FLEX_US_RHR) and the RXRDY bit in FLEX_US_CSR rises. If a character is completed while the RXRDY is set, the Overrun Error (OVRE) bit is set. The last character is transferred into FLEX_US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a one to the Reset Status (RSTSTA) bit in FLEX_US_CR.

Figure 44-20. Receiver Status



44.7.3.8 Parity

The USART supports five parity modes that are selected by writing to the PAR field in FLEX_US_MR. The PAR field also enables the Multidrop mode (see Section 44.7.3.9 “Multidrop Mode”). Even and odd parity bit generation and error detection are supported.

If even parity is selected, the parity generator of the transmitter drives the parity bit to 0 if a number of 1s in the character data bit is even, and to 1 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If odd parity is selected, the parity generator of the transmitter drives the parity bit to 1 if a number of 1s in the character data bit is even, and to 0 if the number of 1s is odd. Accordingly, the receiver parity checker counts the number of received 1s and reports a parity error if the sampled parity bit does not correspond. If the mark parity is used, the parity generator of the transmitter drives the parity bit to 1 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 0. If the space parity is used, the parity generator of the transmitter drives the parity bit to 0 for all characters. The receiver parity checker reports an error if the parity bit is sampled to 1. If parity is disabled, the transmitter does not generate any parity bit and the receiver does not report any parity error.

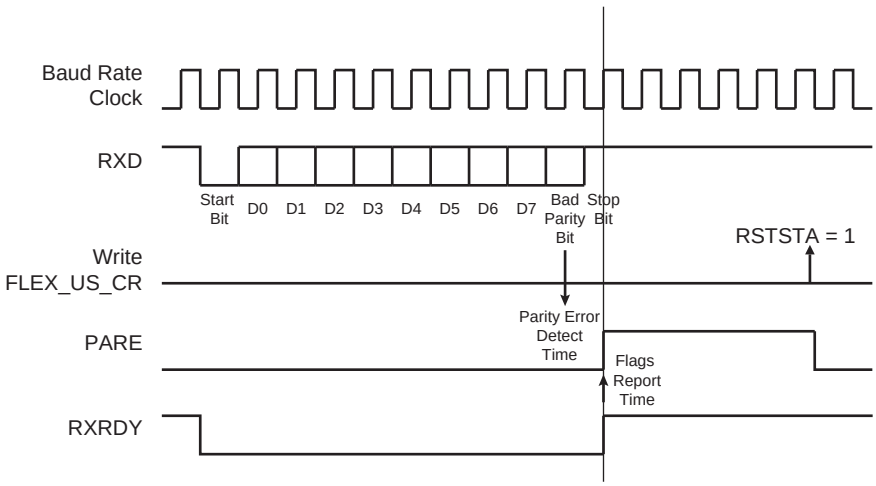
Table 44-9 shows an example of the parity bit for the character 0x41 (character ASCII “A”) depending on the configuration of the USART. Because there are two bits set to 1 in the character value, the parity bit is set to 1 when the parity is odd, or configured to 0 when the parity is even.

Table 44-9. Parity Bit Examples

Character	Hexadecimal	Binary	Parity Bit	Parity Mode
A	0x41	0100 0001	1	Odd
A	0x41	0100 0001	0	Even
A	0x41	0100 0001	1	Mark
A	0x41	0100 0001	0	Space
A	0x41	0100 0001	None	None

When the receiver detects a parity error, it sets the PARE (Parity Error) bit in FLEX_US_CSR. The PARE bit can be cleared by writing a one to the RSTSTA bit in FLEX_US_CR. Figure 44-21 illustrates the parity bit status setting and clearing.

Figure 44-21. Parity Error



44.7.3.9 Multidrop Mode

If the value 0x6 or 0x07 is written to the PAR field in FLEX_US_MR, the USART runs in Multidrop mode. This mode differentiates the data characters and the address characters. Data are transmitted with the parity bit to 0 and addresses are transmitted with the parity bit to 1.

If the USART is configured in Multidrop mode, the receiver sets the PARE parity error bit when the parity bit is high and the transmitter is able to send a character with the parity bit high when a one is written to the SENTA bit in FLEX_US_CR.

To handle parity error, the PARE bit is cleared by writing a one to the RSTSTA bit in FLEX_US_CR.

The transmitter sends an address byte (parity bit set) when SENDA is written to 1 in FLEX_US_CR. In this case, the next byte written to FLEX_US_THR is transmitted as an address. Any character written in FLEX_US_THR when the SENDA command is not written is transmitted normally with parity to 0.

44.7.3.10 Transmitter Timeguard

The timeguard feature enables the USART interface with slow remote devices.

The timeguard function enables the transmitter to insert an idle state on the TXD line between two characters. This idle state actually acts as a long stop bit.

The duration of the idle state is programmed in the TG field of the Transmitter Timeguard Register (FLEX_US_TTGR). When this field is written to zero no timeguard is generated. Otherwise, the transmitter holds a high level on TXD after each transmitted byte during the number of bit periods programmed in TG in addition to the number of stop bits.

As illustrated in Figure 44-22, the behavior of TXRDY and TXEMPTY status bits is modified by the programming of a timeguard. TXRDY rises only when the start bit of the next character is sent, and thus remains to 0 during the timeguard transmission if a character has been written in FLEX_US_THR. TXEMPTY remains low until the timeguard transmission is completed as the timeguard is part of the current character being transmitted.

Figure 44-22. Timeguard Operations

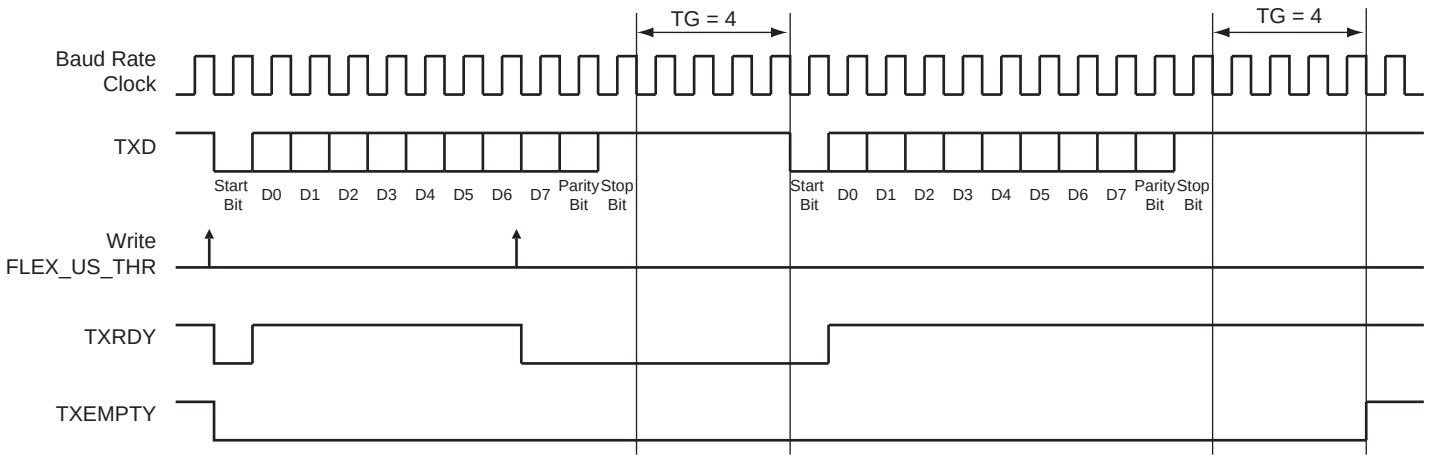


Table 44-10 indicates the maximum length of a timeguard period that the transmitter can handle in relation to the function of the baud rate.

Table 44-10. Maximum Timeguard Length Depending on Baud Rate

Baud Rate (bit/s)	Bit Time (µs)	Timeguard (ms)
1,200	833	212.50
9,600	104	26.56
14,400	69.4	17.71
19,200	52.1	13.28
28,800	34.7	8.85
38,400	26	6.63
56,000	17.9	4.55
57,600	17.4	4.43
115,200	8.7	2.21

44.7.3.11 Receiver Timeout

The Receiver Timeout provides support in handling variable-length frames. This feature detects an idle condition on the RXD line. When a timeout is detected, the TIMEOUT bit in FLEX_US_CSR rises and can generate an interrupt, thus indicating to the driver an end of frame.

The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Timeout Register (FLEX_US_RTOR). If the TO field is written to 0, the Receiver Timeout is disabled and no timeout is detected. The TIMEOUT bit in FLEX_US_CSR remains at 0. Otherwise, the receiver loads a 16-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time

a new character is received. If the counter reaches 0, the TIMEOUT bit in FLEX_US_CSR rises. Then, the user can either:

- Stop the counter clock until a new character is received. This is performed by writing a ‘1’ to FLEX_US_CR.STTTO. In this case, the idle state on RXD before a new character is received does not provide a timeout. This prevents having to handle an interrupt before a character is received and enables waiting for the next idle state on RXD after a frame is received.
- Obtain an interrupt while no character is received. This is performed by writing a ‘1’ to FLEX_US_CR.RETTO. In this case, the counter starts counting down immediately from the value TO. This generates a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

Figure 44-23 shows the block diagram of the Receiver Timeout feature.

Figure 44-23. Receiver Timeout Block Diagram

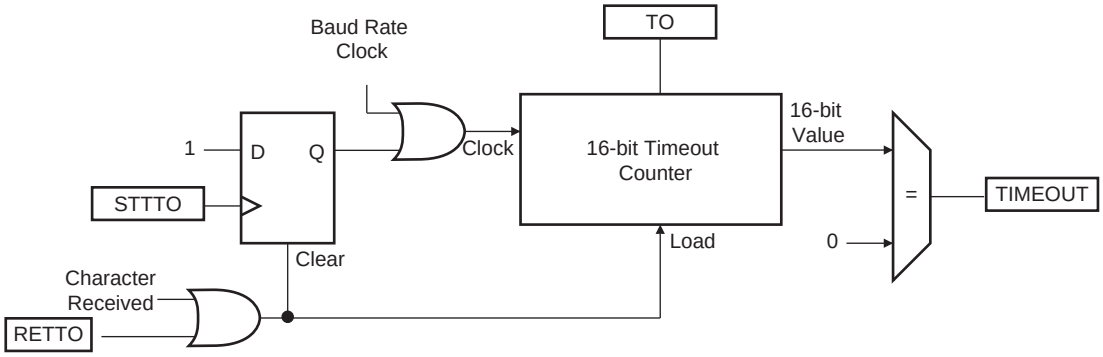


Table 44-11 gives the maximum timeout period for some standard baud rates.

Table 44-11. Maximum Timeout Period

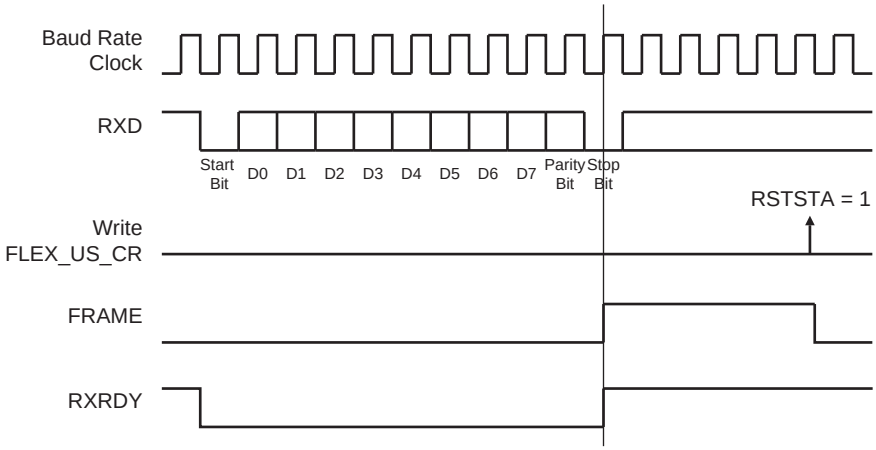
Baud Rate (bit/s)	Bit Time (µs)	Timeout (ms)
600	1,667	109,225
1,200	833	54,613
2,400	417	27,306
4,800	208	13,653
9,600	104	6,827
14,400	69	4,551
19,200	52	3,413
28,800	35	2,276
38,400	26	1,704
56,000	18	1,170
57,600	17	1,138
200,000	5	328

44.7.3.12 Framing Error

The receiver is capable of detecting framing errors. A framing error happens when the stop bit of a received character is detected at level 0. This can occur if the receiver and the transmitter are fully desynchronized.

A framing error is reported on the FRAME bit of FLEX_US_CSR. The FRAME bit is asserted in the middle of the stop bit as soon as the framing error is detected. It is cleared by writing a one to the RSTSTA bit in FLEX_US_CR.

Figure 44-24. Framing Error Status



44.7.3.13 Transmit Break

The user can request the transmitter to generate a break condition on the TXD line. A break condition drives the TXD line low during at least one complete character. It appears the same as a 0x00 character sent with the parity and the stop bits to 0. However, the transmitter holds the TXD line at least during one character until the user requests the break condition to be removed.

A break is transmitted by setting the STTBRK bit in FLEX_US_CR. This can be done at any time, either while the transmitter is empty (no character in either the Shift register or in FLEX_US_THR) or when a character is being transmitted. If a break is requested while a character is being shifted out, the character is first completed before the TXD line is held low.

Once the Start Break command is requested, further Start Break commands are ignored until the end of the break is completed.

The break condition is removed by setting the STPBRK bit in FLEX_US_CR. If the Stop Break command is requested before the end of the minimum break duration (one character, including start, data, parity and stop bits), the transmitter ensures that the break condition completes.

The transmitter considers the break as though it is a character, i.e., the Start Break and Stop Break commands are processed only if the TXRDY bit = 1 in FLEX_US_CSR and the start of the break condition clears the TXRDY and TXEMPTY bits as if a character is processed.

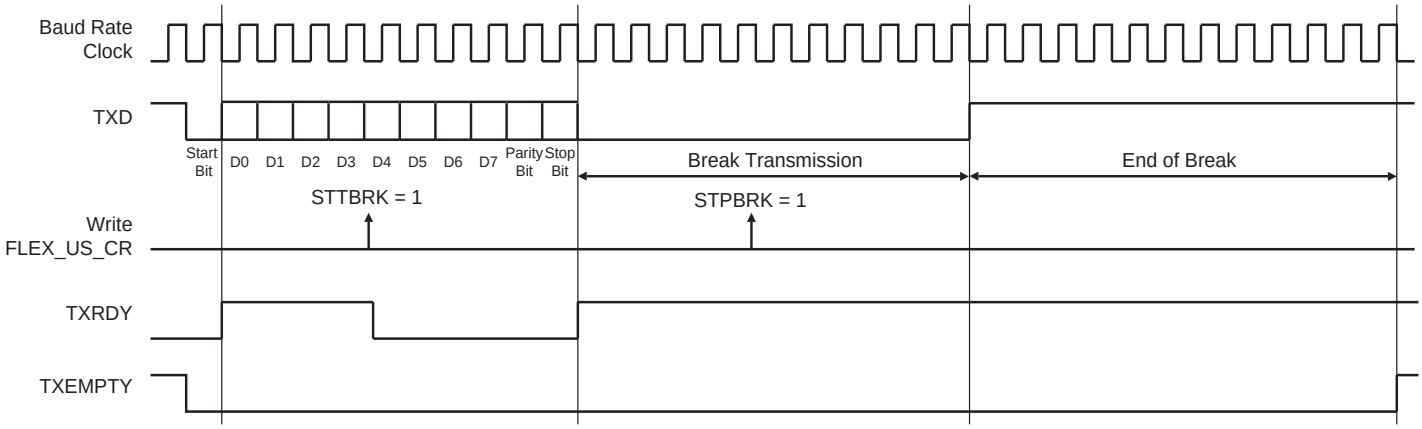
Setting both the STTBRK and STPBRK bits in FLEX_US_CR can lead to an unpredictable result. All Stop Break commands requested without a previous Start Break command are ignored. A byte written into the Transmit Holding register while a break is pending, but not started, is ignored.

After the break condition, the transmitter returns the TXD line to 1 for a minimum of 12 bit times. Thus, the transmitter ensures that the remote receiver detects correctly the end of break and the start of the next character. If the timeguard is programmed with a value higher than 12, the TXD line is held high for the timeguard period.

After holding the TXD line for this period, the transmitter resumes normal operations.

Figure 44-25 illustrates the effect of both the Start Break (STTBRK) and Stop Break (STPBRK) commands on the TXD line.

Figure 44-25. Break Transmission



44.7.3.14 Receive Break

The receiver detects a break condition when all data, parity and stop bits are low. This corresponds to detecting a framing error with data to 0x00, but FRAME remains low.

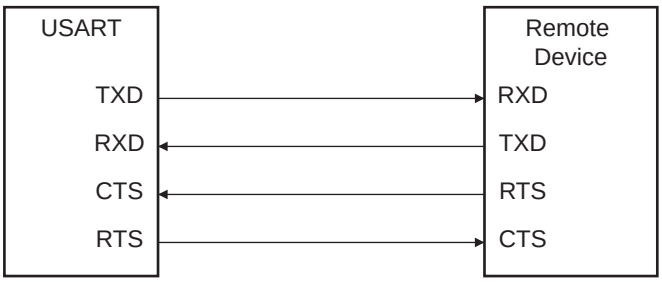
When the low stop bit is detected, the receiver asserts the RXBRK bit in FLEX_US_CSR. FLEX_US_CSR.RXBRK may be cleared by setting the RSTSTA bit in FLEX_US_CR.

An end of receive break is detected by a high level for at least 2/16ths of a bit period in Asynchronous operating mode or one sample at high level in Synchronous operating mode. The end of break detection also asserts the RXBRK bit.

44.7.3.15 Hardware Handshaking

The USART features a hardware handshaking out-of-band flow control. The RTS and CTS pins are used to connect with the remote device, as shown in Figure 44-26.

Figure 44-26. Connection with a Remote Device for Hardware Handshaking



Setting the USART to operate with hardware handshaking is performed by writing the USART_MODE field in FLEX_US_MR to the value 0x2.

The USART behavior when hardware handshaking is enabled is the same as the behavior in standard Synchronous or Asynchronous mode, except that the receiver drives the RTS pin as described below and the level on the CTS pin modifies the behavior of the transmitter as described below. Using this mode requires using the DMAC channel for reception. The transmitter can handle hardware handshaking in any case.

Figure 44-27. RTS line software control when FLEX_US_MR.USART_MODE = 2

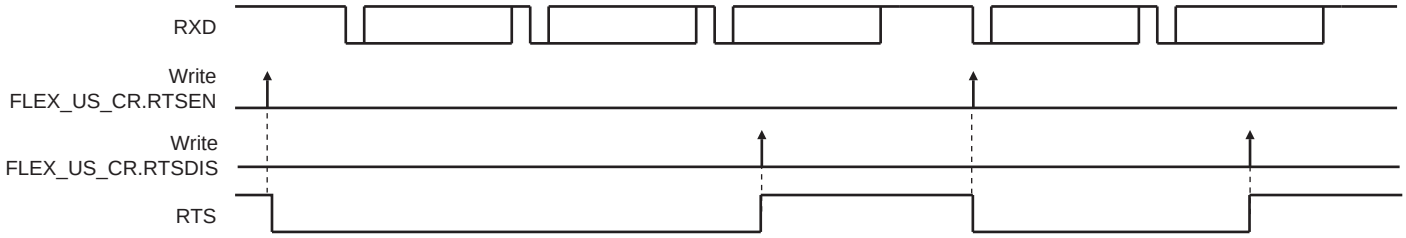
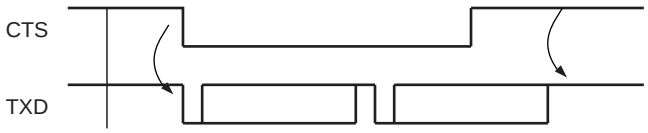


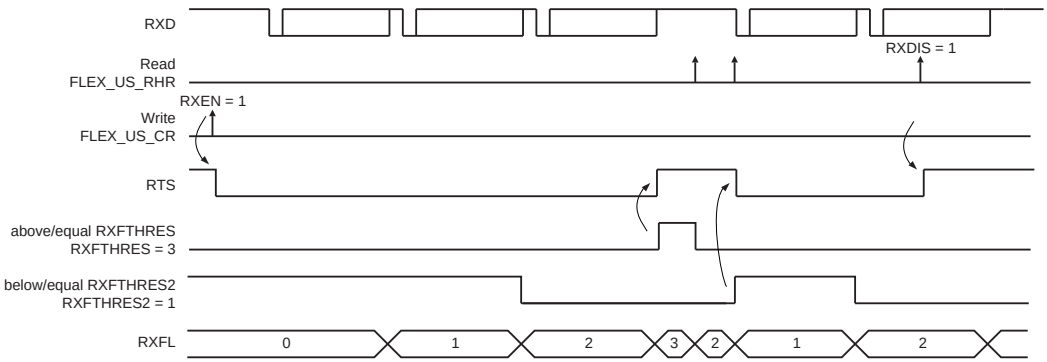
Figure 44-28 shows how the transmitter operates if hardware handshaking is enabled. The CTS pin disables the transmitter. If a character is being processed, the transmitter is disabled only after the completion of the current character and transmission of the next character happens as soon as the pin CTS falls.

Figure 44-28. Transmitter Behavior when Operating with Hardware Handshaking



If USART FIFOs are enabled (FIFOEN bit in FLEX_US_CR), the RTS pin can be controlled by the USART Receive FIFO thresholds. The RTS pin control through Receive FIFO thresholds can be activated with the FRTSC bit in FLEX_US_FMR. Once activated, the RTS pin will be controlled by Receive FIFO thresholds, set to level ‘1’ each time RXFTHRES is reached and set to level ‘0’ each time RXFTHRES2 is reached (and RXFTHRES is not reached).

Figure 44-29. Receiver Behavior When FIFO Enabled and FRTSC Set to ‘1’



Note: In this mode, RXFTHRES must be > RXFTHRES2.

44.7.4 ISO7816 Mode

The USART features an ISO7816-compatible operating mode. This mode permits interfacing with smart cards and Security Access Modules (SAM) communicating through an ISO7816 link. Both T = 0 and T = 1 protocols defined by the ISO7816 specification are supported.

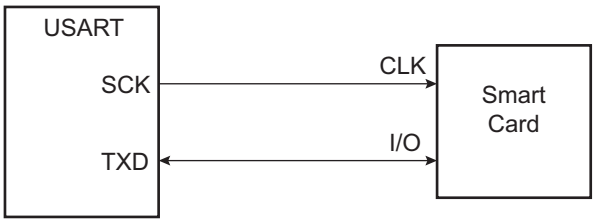
Setting the USART in ISO7816 mode is performed by writing the USART_MODE field in FLEX_US_MR to the value 0x4 for protocol T = 0 and to the value 0x6 for protocol T = 1.

44.7.4.1 ISO7816 Mode Overview

The ISO7816 is a half duplex communication on only one bidirectional line. The baud rate is determined by a division of the clock provided to the remote device (see Figure 44.7.1.1).

The USART connects to a smart card as shown in Figure 44-30. The TXD line becomes bidirectional and the baud rate generator feeds the ISO7816 clock on the SCK pin. As the TXD pin becomes bidirectional, its output remains driven by the output of the transmitter but only when the transmitter is active while its input is directed to the input of the receiver. The USART is considered as the master of the communication as it generates the clock.

Figure 44-30. Connection of a Smart Card to the USART



When operating in ISO7816, either in T = 0 or T = 1 modes, the character format is fixed. The configuration is 8 data bits, even parity and 1 or 2 stop bits, regardless of the values programmed in the CHRL, MODE9, PAR and CHMODE fields. MSBF can be used to transmit LSB or MSB first. Parity Bit (PAR) can be used to transmit in Normal or Inverse mode. Refer to Section 44.10.6 “USART Mode Register” and PAR: Parity Type.

The USART cannot operate concurrently in both Receiver and Transmitter modes as the communication is unidirectional at a time. It has to be configured according to the required mode by enabling or disabling either the receiver or the transmitter as desired. Enabling both the receiver and the transmitter at the same time in ISO7816 mode may lead to unpredictable results.

The ISO7816 specification defines an inverse transmission format. Data bits of the character must be transmitted on the I/O line at their negative value.

44.7.4.2 Protocol T = 0

In T = 0 protocol, a character is made up of 1 start bit, 8 data bits, 1 parity bit and 1 guard time, which lasts two bit times. The transmitter shifts out the bits and does not drive the I/O line during the guard time.

If no parity error is detected, the I/O line remains at 1 during the guard time and the transmitter can continue with the transmission of the next character, as shown in Figure 44-31.

If a parity error is detected by the receiver, it drives the I/O line to 0 during the guard time, as shown in Figure 44-32. This error bit is also named NACK, for Non Acknowledge. In this case, the character lasts 1 bit time more, as the guard time length is the same and is added to the error bit time which lasts 1 bit time.

When the USART is the receiver and it detects an error, it does not load the erroneous character in the Receive Holding Register (FLEX_US_RHR). It appropriately sets the PARE bit in the Status Register (FLEX_US_SR) so that the software can handle the error.

Figure 44-31. T = 0 Protocol without Parity Error

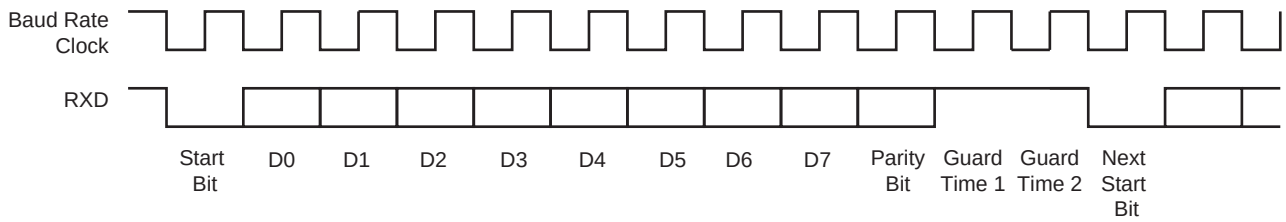
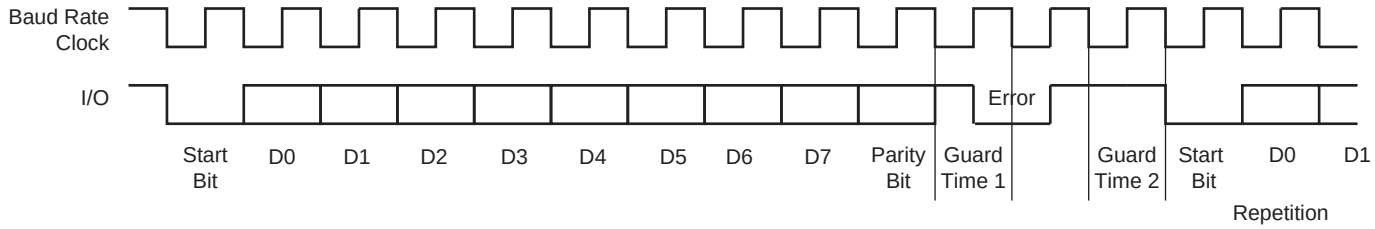


Figure 44-32. T = 0 Protocol with Parity Error



Receive Error Counter

The USART receiver also records the total number of errors. This can be read in the Number of Error (FLEX_US_NER) register. The NB_ERRORS field can record up to 255 errors. Reading FLEX_US_NER automatically clears the NB_ERRORS field.

Receive NACK Inhibit

The USART can be configured to inhibit an error. This is done by writing a '1' to FLEX_US_MR.INACK. In this case, no error signal is driven on the I/O line even if a parity bit is detected.

Moreover, if INACK = 1, the erroneous received character is stored in the Receive Holding register as if no error occurred, and the RXRDY bit rises.

Transmit Character Repetition

When the USART is transmitting a character and gets a NACK, it can automatically repeat the character before moving on to the next one. Repetition is enabled by writing the MAX_ITERATION field in FLEX_US_MR at a value higher than 0. Each character can be transmitted up to eight times; the first transmission plus seven repetitions.

If MAX_ITERATION does not equal zero, the USART repeats the character as many times as the value loaded in MAX_ITERATION.

When the USART repetition number reaches MAX_ITERATION, and the last repeated character is not acknowledged, the ITER bit is set in FLEX_US_CSR. If the repetition of the character is acknowledged by the receiver, the repetitions are stopped and the iteration counter is cleared.

The ITER bit in FLEX_US_CSR can be cleared by writing FLEX_US_CR with the RSTIT bit to 1.

Disable Successive Receive NACK

The receiver can limit the number of successive NACKs sent back to the remote transmitter. This is programmed by setting the DSNACK bit in FLEX_US_MR. The maximum number of NACKs transmitted is programmed in the MAX_ITERATION field. As soon as MAX_ITERATION is reached, no error signal is driven on the I/O line and the ITER bit in FLEX_US_CSR is set.

44.7.4.3 Protocol T = 1

When operating in ISO7816 protocol T = 1, the transmission is similar to an asynchronous format with only one stop bit. The parity is generated when transmitting and checked when receiving. Parity error detection sets the PARE bit in FLEX_US_CSR.

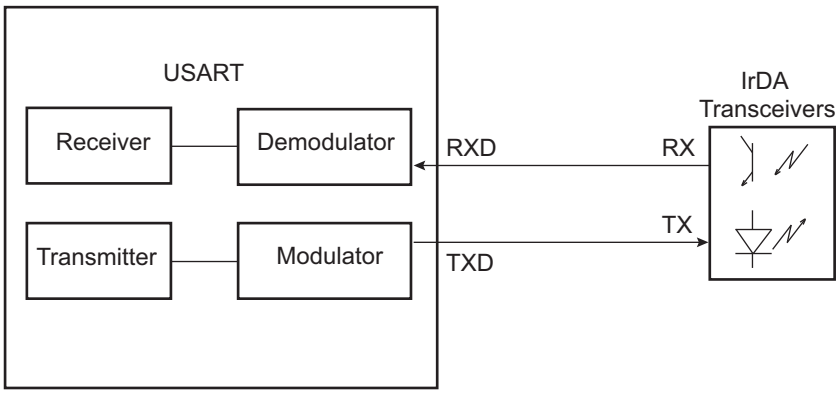
44.7.5 IrDA Mode

The USART features an IrDA mode supplying half-duplex point-to-point wireless communication. It embeds the modulator and demodulator which allows a glueless connection to the infrared transceivers, as shown in Figure 44-33. The modulator and demodulator are compliant with the IrDA specification version 1.1 and support data transfer speeds ranging from 2.4 kbit/s to 115.2 kbit/s.

The USART IrDA mode is enabled by setting the USART_MODE field in FLEX_US_MR to the value 0x8. The IrDA Filter Register (FLEX_US_IF) allows configuring the demodulator filter. The USART transmitter and receiver

operate in a normal Asynchronous mode and all parameters are accessible. Note that the modulator and the demodulator are activated.

Figure 44-33. Connection to IrDA Transceivers



The receiver and the transmitter must be enabled or disabled according to the direction of the transmission to be managed.

To receive IrDA signals, the following needs to be done:

- Disable TX and Enable RX
- Configure the TXD pin as PIO and set it as an output to 0 (to avoid LED transmission). Disable the internal pull-up (better for power consumption).
- Receive data

44.7.5.1 IrDA Modulation

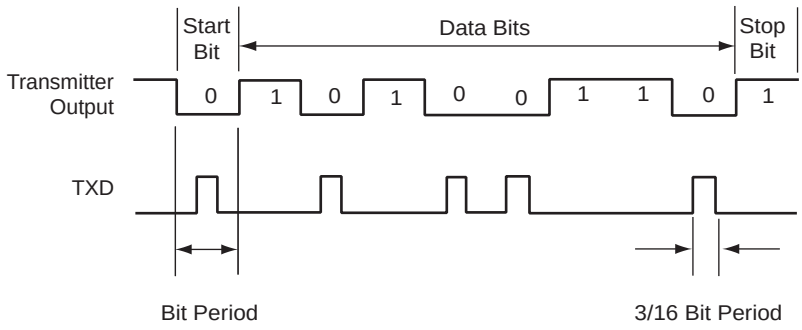
For baud rates up to and including 115.2 kbit/s, the RZI modulation scheme is used. “0” is represented by a light pulse of 3/16th of a bit time. Some examples of signal pulse duration are shown in Table 44-12.

Table 44-12. IrDA Pulse Duration

Baud Rate	Pulse Duration (3/16)
2.4 kbit/s	78.13 µs
9.6 kbit/s	19.53 µs
19.2 kbit/s	9.77 µs
38.4 kbit/s	4.88 µs
57.6 kbit/s	3.26 µs
115.2 kbit/s	1.63 µs

Figure 44-34 shows an example of character transmission.

Figure 44-34. IrDA Modulation



44.7.5.2 IrDA Baud Rate

Table 44-13 gives some examples of CD values, baud rate error and pulse duration. Note that the requirement on the maximum acceptable error of ±1.87% must be met.

Table 44-13. IrDA Baud Rate Error

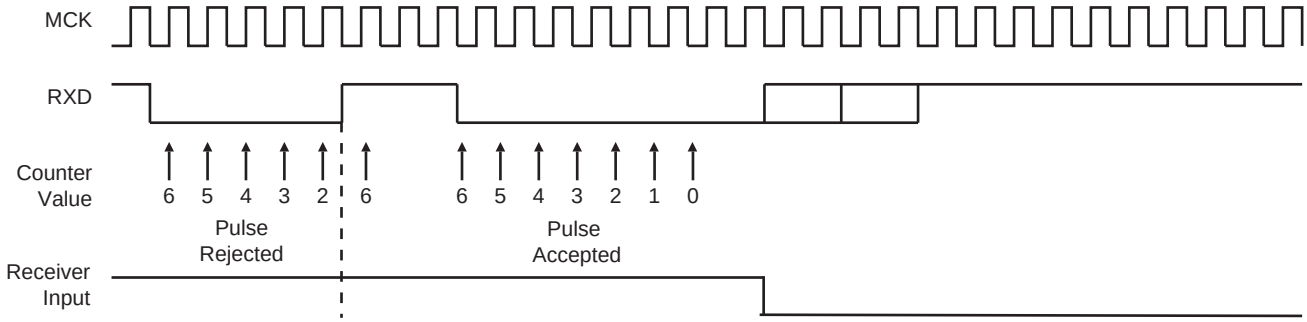
Peripheral Clock	Baud Rate (bit/s)	CD	Baud Rate Error	Pulse Time (µs)
3,686,400	115,200	2	0.00%	1.63
20,000,000	115,200	11	1.38%	1.63
32,768,000	115,200	18	1.25%	1.63
40,000,000	115,200	22	1.38%	1.63
3,686,400	57,600	4	0.00%	3.26
20,000,000	57,600	22	1.38%	3.26
32,768,000	57,600	36	1.25%	3.26
40,000,000	57,600	43	0.93%	3.26
3,686,400	38,400	6	0.00%	4.88
20,000,000	38,400	33	1.38%	4.88
32,768,000	38,400	53	0.63%	4.88
40,000,000	38,400	65	0.16%	4.88
3,686,400	19,200	12	0.00%	9.77
20,000,000	19,200	65	0.16%	9.77
32,768,000	19,200	107	0.31%	9.77
40,000,000	19,200	130	0.16%	9.77
3,686,400	9,600	24	0.00%	19.53
20,000,000	9,600	130	0.16%	19.53
32,768,000	9,600	213	0.16%	19.53
40,000,000	9,600	260	0.16%	19.53
3,686,400	2,400	96	0.00%	78.13
20,000,000	2,400	521	0.03%	78.13
32,768,000	2,400	853	0.04%	78.13

44.7.5.3 IrDA Demodulator

The demodulator is based on the IrDA Receive filter comprised of an 8-bit down counter which is loaded with the value programmed in FLEX_US_IF. When a falling edge is detected on the RXD pin, the Filter Counter starts counting down at the peripheral clock speed. If a rising edge is detected on the RXD pin, the counter stops and is reloaded with FLEX_US_IF. If no rising edge is detected when the counter reaches 0, the input of the receiver is driven low during one bit time.

Figure 44-35 illustrates the operations of the IrDA demodulator.

Figure 44-35. IrDA Demodulator Operations



The programmed value in the FLEX_US_IF register must always meet the following criteria:

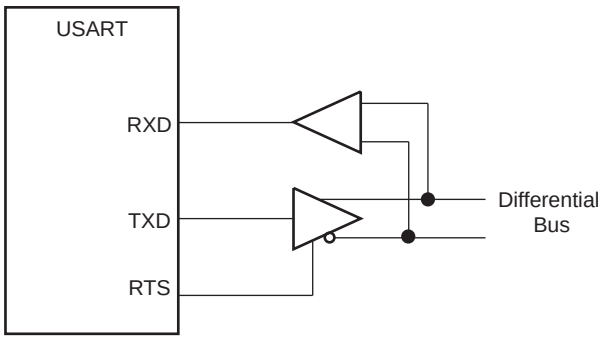
$$t_{peripheral\ clock} \times (IRDA_FILTER + 3) < 1.41\ \mu s$$

As the IrDA mode uses the same logic as the ISO7816, note that the FI_DI_RATIO field in FLEX_US_FIDI must be set to a value higher than 0 to make sure IrDA communications operate correctly.

44.7.6 RS485 Mode

The USART features the RS485 mode to enable line driver control. While operating in RS485 mode, the USART behaves as though in Asynchronous or Synchronous mode and configuration of all the parameters is possible. The difference is that the RTS pin is driven high when the transmitter is operating. The behavior of the RTS pin is controlled by the TXEMPTY bit. A typical connection of the USART to an RS485 bus is shown in Figure 44-36.

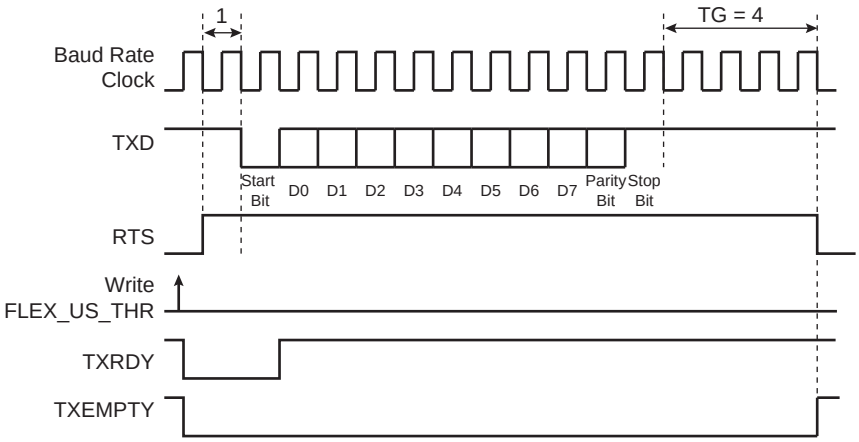
Figure 44-36. Typical Connection to an RS485 Bus



The USART is set in RS485 mode by writing the value 0x1 to the USART_MODE field in FLEX_US_MR.

The RTS pin is at a level inverse to the TXEMPTY bit. Significantly, the RTS pin remains high when a timeguard is programmed so that the line can remain driven after the last character completion. Figure 44-37 gives an example of the RTS waveform during a character transmission when the timeguard is enabled.

Figure 44-37. Example of RTS Drive with Timeguard



44.7.7 USART Comparison Function on Received Character

The CMP flag in FLEX_US_CSR is set when the received character matches the conditions programmed in FLEX_US_CMPR. The CMP flag is set as soon as FLEX_US_RHR is loaded with the new received character. The CMP flag is cleared by writing a one to the RSTSTA bit in FLEX_US_CR.

FLEX_US_CMPR (see Section 44.10.34 “USART Comparison Register”) can be programmed to provide different comparison methods:

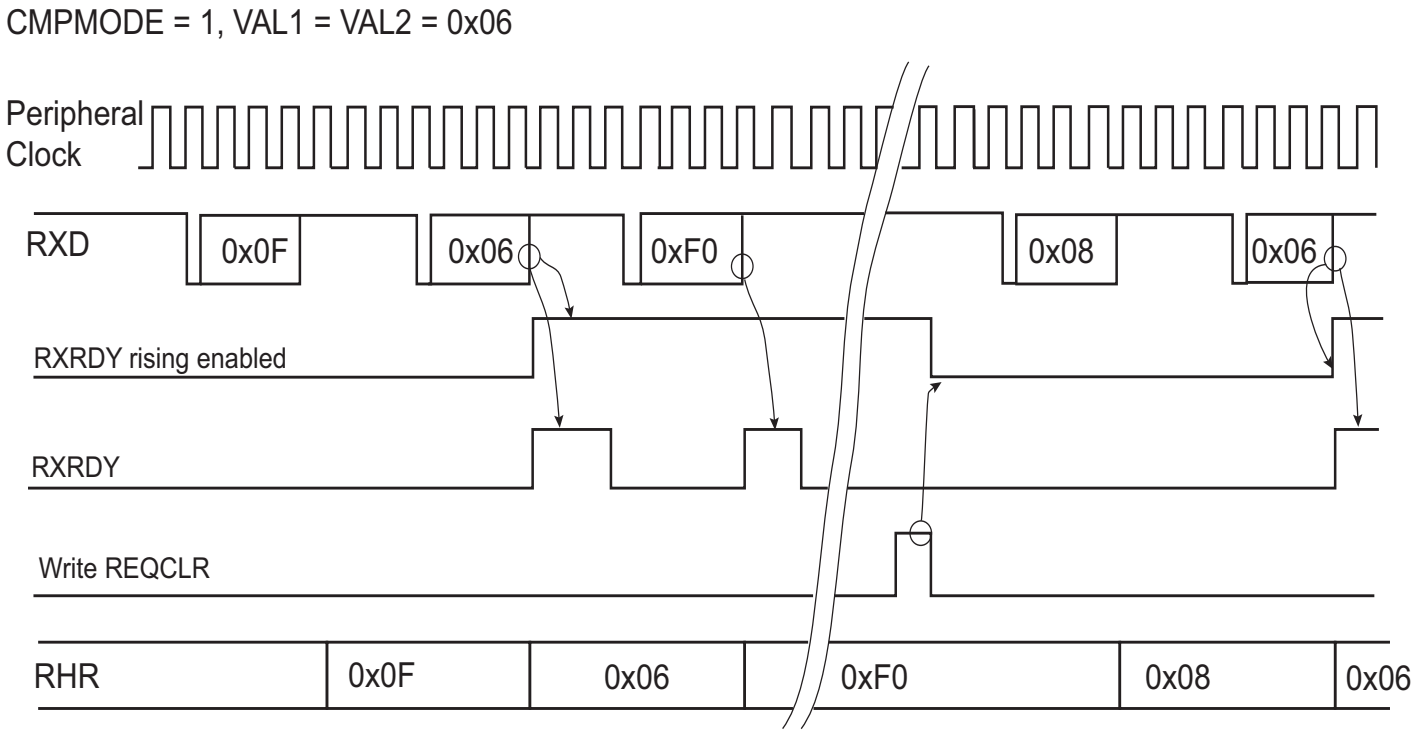
- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When the CMPMODE bit is set to FLAG_ONLY (value 0) in FLEX_US_CMPR, all received data are loaded in FLEX_US_RHR and the CMP flag provides the status of the comparison result.

By programming the CMPMODE bit to START_CONDITION (value 1), the comparison function result triggers the start of the loading of FLEX_US_RHR (see Figure 44-38). The trigger condition exists as soon as the received character value matches the condition defined by the programming of VAL1, VAL2 and CMPPAR in FLEX_US_CMPR. The comparison trigger event is restarted by writing a 1 to the REQCLR bit in FLEX_US_CR.

The value programmed in the VAL1 and VAL2 fields must not exceed the maximum value of the received character (see CHRL field in FLEX_US_MR).

Figure 44-38. Receive Holding Register Management



44.7.8 SPI Mode

The Serial Peripheral Interface (SPI) mode is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master” which controls the data flow, while the other devices act as “slaves" which have data shifted into and out by the master. Different CPUs can take turns being masters and one master may simultaneously shift data into multiple slaves. (Multiple master protocol is the opposite of single master protocol, where one CPU is always the master while all of the others are always slaves.) However, only one slave may drive its output to write data back to the master at any given time.

A slave device is selected when its NSS signal is asserted by the master. The USART in SPI Master mode can address only one SPI slave because it can generate only one NSS signal.

The SPI system consists of two data lines and two control lines:

- Master Out Slave In (MOSI): This data line supplies the output data from the master shifted into the input of the slave.
- Master In Slave Out (MISO): This data line supplies the output data from a slave to the input of the master.
- Serial Clock (SCK): This control line is driven by the master and regulates the flow of the data bits. The master may transmit data at a variety of bit rates. The SCK line cycles once for each bit that is transmitted.
- Slave Select (NSS): This control line allows the master to select or deselect the slave.

44.7.8.1 Modes of Operation

The USART can operate in SPI Master mode or in SPI Slave mode.

Operation in SPI Master mode is programmed by writing 0xE to the USART_MODE field in FLEX_US_MR. In this case the SPI lines must be connected as described below:

- The MOSI line is driven by the output pin TXD
- The MISO line drives the input pin RXD
- The SCK line is driven by the output pin SCK
- The NSS line is driven by the output pin RTS

Operation in SPI Slave mode is programmed by writing to 0xF the USART_MODE field in FLEX_US_MR. In this case the SPI lines must be connected as described below:

- The MOSI line drives the input pin RXD
- The MISO line is driven by the output pin TXD
- The SCK line drives the input pin SCK
- The NSS line drives the input pin CTS

In order to avoid unpredictable behavior, any change of the SPI mode must be followed by a software reset of the transmitter and of the receiver (except the initial configuration after a hardware reset). (See Section 44.7.8.4 “Receiver and Transmitter Control”.)

44.7.8.2 Bit Rate

In SPI mode, the bit rate generator operates in the same way as in USART Synchronous mode: Section 44.7.1.3 “Baud Rate in Synchronous Mode or SPI Mode” However, there are some restrictions:

In SPI Master mode:

- The external clock SCK must not be selected (USCLKS ≠ 0x3), and the CLKO bit in FLEX_US_MR must be set in order to generate correctly the serial clock on the SCK pin.
- To obtain correct behavior of the receiver and the transmitter, the value programmed in CD must be ≥ 6.
- If the divided peripheral clock is selected, the value programmed in CD must be even to ensure a 50:50 mark/space ratio on the SCK pin, this value can be odd if the peripheral clock is selected.

In SPI Slave mode:

- The external clock (SCK) selection is forced regardless of the value of the USCLKS field in FLEX_US_MR. Likewise, the value written in FLEX_US_BRGR has no effect, because the clock is provided directly by the signal on the USART SCK pin.
- To obtain correct behavior of the receiver and the transmitter, the external clock (SCK) frequency must be at least six times lower than the system clock.

44.7.8.3 Data Transfer

Up to nine data bits are successively shifted out on the TXD pin at each rising or falling edge (depending of CPOL and CPHA) of the programmed serial clock. There is no Start bit, no Parity bit and no Stop bit.

The number of data bits is selected by the CHRL field and the MODE 9 bit in FLEX_US_MR. The nine bits are selected by setting the MODE 9 bit regardless of the CHRL field. The MSB data bit is always sent first in SPI mode (Master or Slave).

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in FLEX_US_MR. The clock phase is programmed with the CPHA bit. These two parameters determine the edges of the clock signal upon which data are driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are used and fixed in different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

Table 44-14. SPI Bus Protocol Mode

SPI Bus Protocol Mode	CPOL	CPHA
0	0	1
1	0	0
2	1	1
3	1	0

Figure 44-39. SPI Transfer Format (CPHA = 1, 8 bits per transfer)

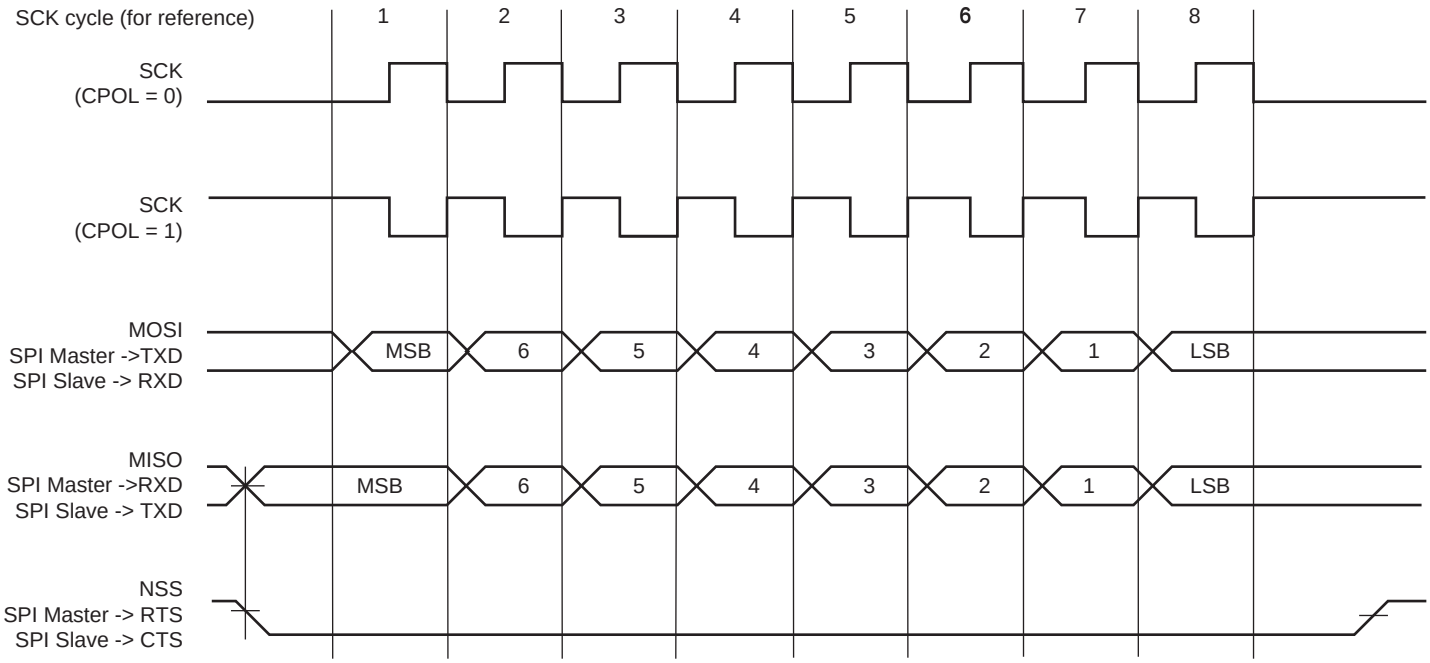
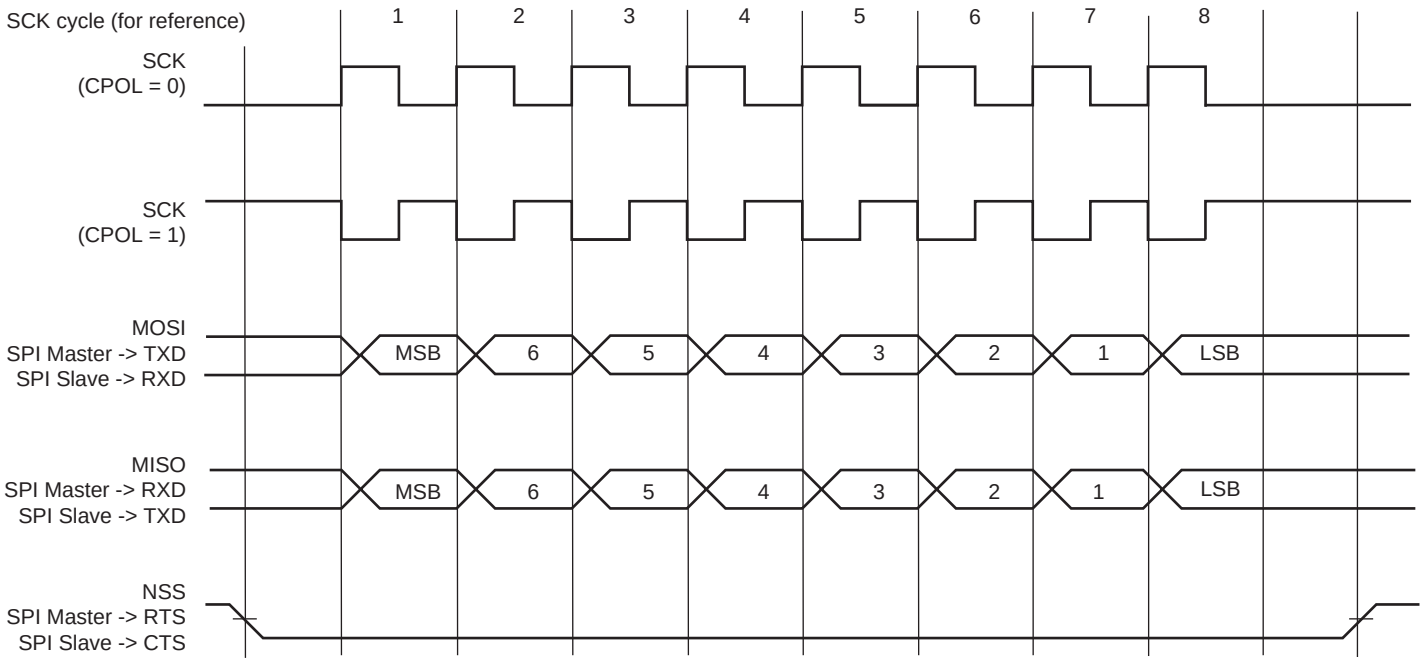


Figure 44-40. SPI Transfer Format (CPHA = 0, 8 bits per transfer)



44.7.8.4 Receiver and Transmitter Control

See Section 44.7.2 “Receiver and Transmitter Control”.

44.7.8.5 Character Transmission

The characters are sent by writing in the Transmit Holding Register (FLEX_US_THR). An additional condition for transmitting a character can be added when the USART is configured in SPI Master mode. In the “USART Mode Register (SPI_MODE)” (USART_MR), the value configured on the WRDBT bit can prevent any character transmission (even if FLEX_US_THR has been written) while the receiver side is not ready (character not read). When WRDBT = 0, the character is transmitted whatever the receiver status. If WRDBT = 1, the transmitter waits for the Receive Holding Register (FLEX_US_RHR) to be read before transmitting the character (RXRDY flag cleared), thus preventing any overflow (character loss) on the receiver side.

The chip select line is de-asserted for a period equivalent to 3 bits between the transmission of two data.

The transmitter reports two status bits in FLEX_US_CSR: TXRDY (Transmitter Ready), which indicates that FLEX_US_THR is empty and TXEMPTY, which indicates that all the characters written in FLEX_US_THR have been processed. When the current character processing is completed, the last character written in FLEX_US_THR is transferred into the Shift register of the transmitter and FLEX_US_THR becomes empty, thus TXRDY rises.

Both TXRDY and TXEMPTY bits are low when the transmitter is disabled. Writing a character in FLEX_US_THR while TXRDY is low has no effect and the written character is lost.

If the USART is in SPI Slave mode and if a character must be sent while FLEX_US_THR is empty, the UNRE (Underrun Error) bit is set. The TXD transmission line stays at high level during all this time. The UNRE bit is cleared by writing a one to the RSTSTA bit in FLEX_US_CR.

In SPI Master mode, the slave select line (NSS) is asserted at low level one t_{bit} (t_{bit} being the

nominal time required to transmit a bit) before the transmission of the MSB bit and released at high level one t_{bit} after the transmission of the LSB bit. So, the slave select line (NSS) is always released between each character transmission and a minimum delay of three t_{bit} always inserted. However, in order to address slave devices supporting the CSAAT mode (Chip Select Active After Transfer), the slave select line (NSS) can be forced at low level by writing a one to the RTSEN bit in FLEX_US_CR. The slave select line (NSS) can be released at high level

only by writing a one to the RTSDIS bit in FLEX_US_CR (for example, when all data have been transferred to the slave device).

In SPI Slave mode, the transmitter does not require a falling edge of the slave select line (NSS) to initiate a character transmission but only a low level. However, this low level must be present on the slave select line (NSS) at least one t_{bit} before the first serial clock cycle corresponding to the MSB bit.

44.7.8.6 Character Reception

When a character reception is completed, it is transferred to the Receive Holding Register (FLEX_US_RHR) and the RXRDY bit in the Status Register (FLEX_US_CSR) rises. If a character is completed while RXRDY is set, the OVRE (Overrun Error) bit is set. The last character is transferred into FLEX_US_RHR and overwrites the previous one. The OVRE bit is cleared by writing a one to the RSTSTA bit in FLEX_US_CR.

To ensure correct behavior of the receiver in SPI Slave mode, the master device sending the frame must ensure a minimum delay of one t_{bit} between each character transmission. The receiver does not require a falling edge of the slave select line (NSS) to initiate a character reception but only a low level. However, this low level must be present on the slave select line (NSS) at least one t_{bit} before the first serial clock cycle corresponding to the MSB bit.

44.7.8.7 Receiver Timeout

Because the receiver bit rate clock is active only during data transfers in SPI mode, a receiver timeout is impossible in this mode, whatever the timeout value is (field TO) in FLEX_US_RTOR.

44.7.9 LIN Mode

The LIN mode provides master node and slave node connectivity on a LIN bus.

The LIN (Local Interconnect Network) is a serial communication protocol which efficiently supports the control of mechatronic nodes in distributed automotive applications.

The main properties of the LIN bus are:

- Single master/multiple slaves concept
- Low-cost silicon implementation based on common UART/SCI interface hardware, an equivalent in software, or as a pure state machine.
- Self synchronization without quartz or ceramic resonator in the slave nodes
- Deterministic signal transmission
- Low cost single-wire implementation
- Speed up to 20 kbit/s

LIN provides cost efficient bus communication where the bandwidth and versatility of CAN are not required.

The LIN mode enables processing LIN frames with a minimum of action from the microprocessor.

44.7.9.1 Modes of Operation

The USART can act either as a LIN master node or as a LIN slave node.

The node configuration is chosen by setting the USART_MODE field in the USART Mode Register (FLEX_US_MR):

- LIN master node (USART_MODE = 0xA)
- LIN slave node (USART_MODE = 0xB)

In order to avoid unpredictable behavior, any change of the LIN node configuration must be followed by a software reset of the transmitter and of the receiver (except the initial node configuration after a hardware reset). (See Section 44.7.9.3 “Receiver and Transmitter Control”.)

44.7.9.2 Baud Rate Configuration

See Section 44.7.1.1 “Baud Rate in Asynchronous Mode”.

- LIN master node: The baud rate is configured in FLEX_US_BRGR.
- LIN slave node: The initial baud rate is configured in FLEX_US_BRGR. This configuration is automatically copied in the LIN Baud Rate Register (FLEX_US_LINBRR) when writing FLEX_US_BRGR. After the synchronization procedure, the baud rate is updated in FLEX_US_LINBRR.

44.7.9.3 Receiver and Transmitter Control

See Section 44.7.2 “Receiver and Transmitter Control”.

44.7.9.4 Character Transmission

See Section 44.7.3.1 “Transmitter Operations”.

44.7.9.5 Character Reception

See Section 44.7.3.7 “Receiver Operations”.

44.7.9.6 Header Transmission (Master Node Configuration)

All the LIN Frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

So in master node configuration, the frame handling starts with the sending of the header.

The header is transmitted as soon as the identifier is written in the LIN Identifier Register (FLEX_US_LINIR). At this moment the flag TXRDY falls.

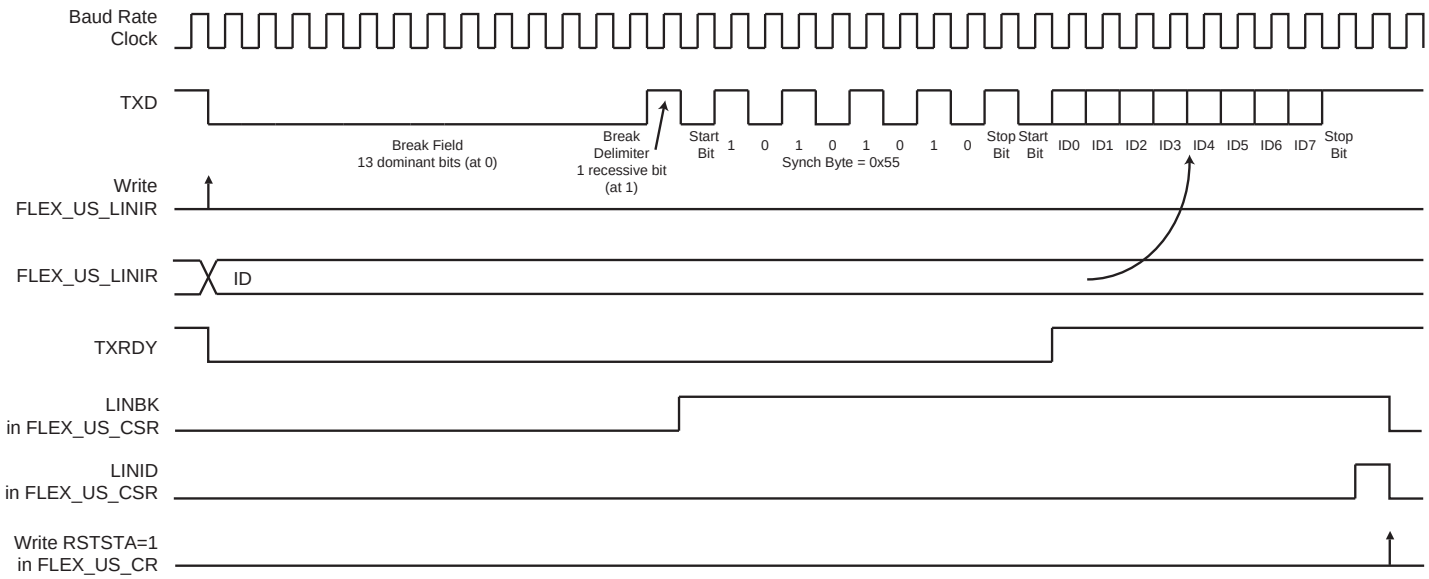
The Break Field, the Synch Field and the Identifier Field are sent automatically one after the other.

The Break Field consists of 13 dominant bits and 1 recessive bit, the Synch Field is the character 0x55 and the Identifier corresponds to the character written in the LIN Identifier Register (FLEX_US_LINIR). The Identifier parity bits can be automatically computed and sent (see Section 44.7.9.9).

The flag TXRDY rises when the identifier character is transferred into the Shift register of the transmitter.

As soon as the Synch Break Field is transmitted, the flag bit LINBK in FLEX_US_CSR is set. Likewise, as soon as the Identifier Field is sent, the flag bit LINID in FLEX_US_CSR is set. These flags are reset by writing a one to the RSTSTA bit in FLEX_US_CR.

Figure 44-41. Header Transmission



44.7.9.7 Header Reception (Slave Node Configuration)

All the LIN frames start with a header which is sent by the master node and consists of a Synch Break Field, Synch Field and Identifier Field.

In slave node configuration, the frame handling starts with the reception of the header.

The USART uses a break detection threshold of 11 nominal bit times at the actual baud rate. At any time, if 11 consecutive recessive bits are detected on the bus, the USART detects a Break Field. As long as a Break Field has not been detected, the USART stays idle and the received data are not taken in account.

When a Break Field has been detected, the flag LINBK in FLEX_US_CSR is set and the USART expects the Synch Field character to be 0x55. This field is used to update the actual baud rate in order to stay synchronized (see Section 44.7.9.8). If the received Synch character is not 0x55, an Inconsistent Synch Field error is generated (see Section 44.7.9.14).

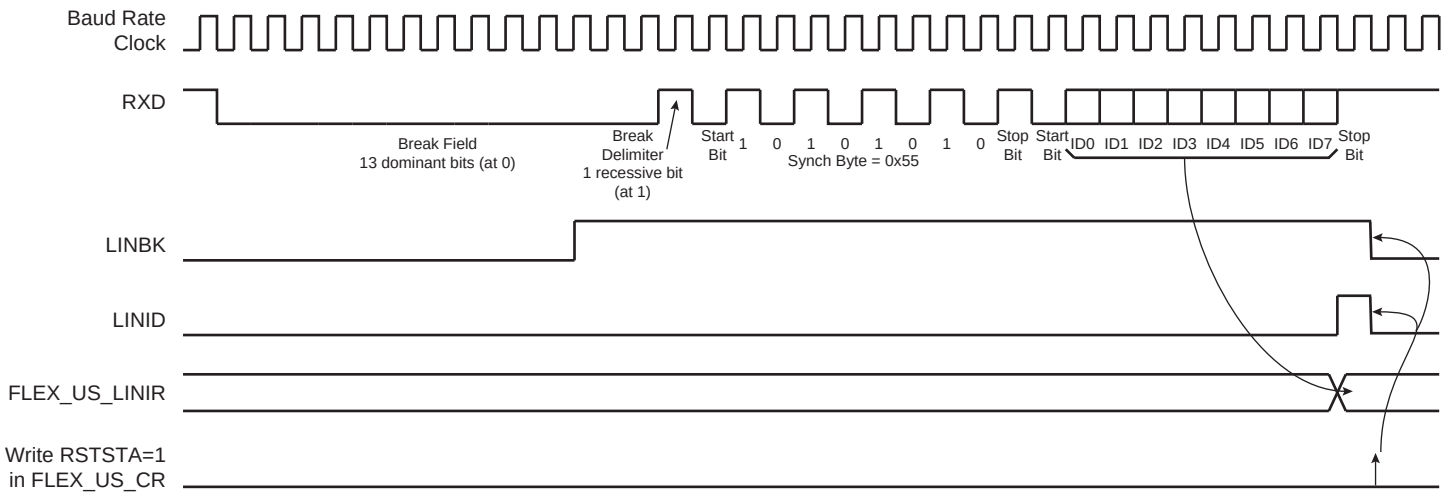
After receiving the Synch Field, the USART expects to receive the Identifier Field.

When the Identifier Field has been received, the flag bit LINID in FLEX_US_CSR is set. At this moment the IDCHR field in the LIN Identifier Register (FLEX_US_LINIR) is updated with the received character. The Identifier parity bits can be automatically computed and checked (see Section 44.7.9.9).

If the Header is not entirely received within the time given by the maximum length of the header $t_{Header_Maximum}$, the error flag bit LINHTE in FLEX_US_CSR is set.

The flag bits LINID, LINBK and LINHTE are reset by writing a one to the RSTSTA bit in FLEX_US_CR.

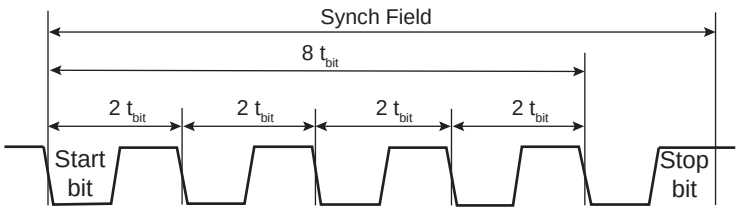
Figure 44-42. Header Reception



44.7.9.8 Slave Node Synchronization

The synchronization is done only in slave node configuration. The procedure is based on time measurement between falling edges of the Synch Field. The falling edges are available in distances of 2, 4, 6 and 8 bit times.

Figure 44-43. Synch Field



The time measurement is made by a 19-bit counter driven by the sampling clock (see Section 44.7.1).

When the start bit of the Synch Field is detected, the counter is reset. Then during the next eight t_{bit} of the Synch Field, the counter is incremented. At the end of these eight t_{bit}, the counter is stopped. At this moment, the 16 most significant bits of the counter (value divided by 8) give the new clock divider (LINCD) and the 3 least significant bits of this value (the remainder) give the new fractional part (LINFP).

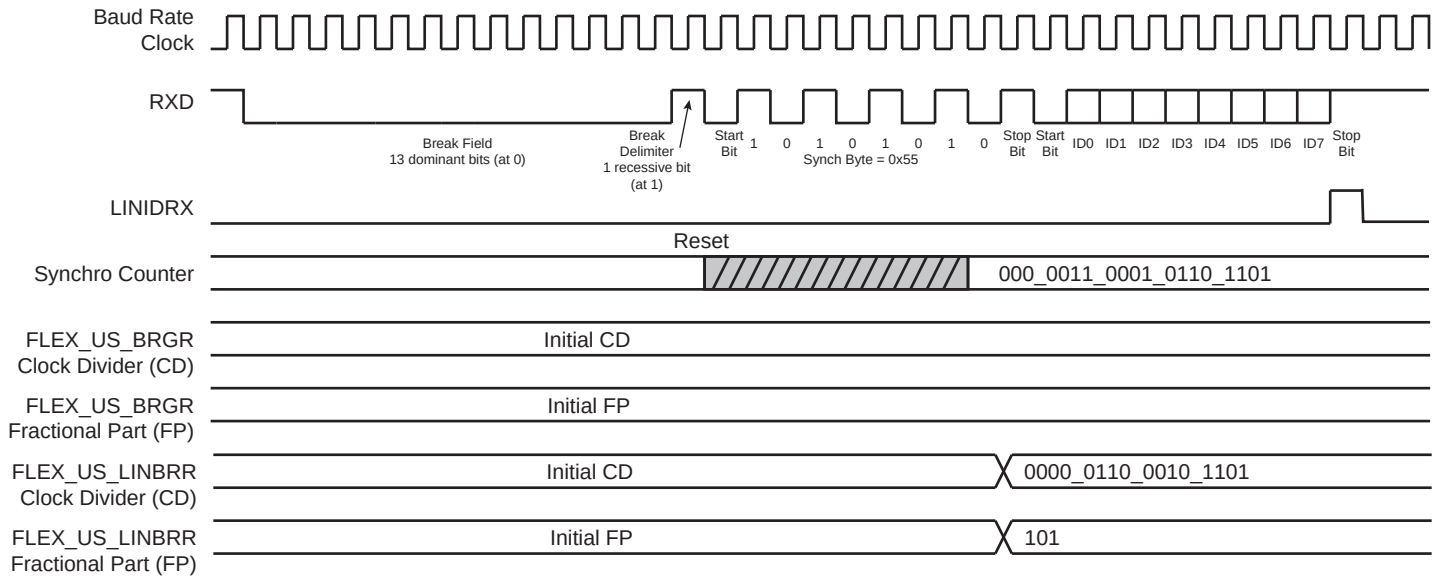
Once the Synch Field has been entirely received, the clock divider (LINCD) and the fractional part (LINFP) are updated in the LIN Baud Rate Register (FLEX_US_LINBRR) with the computed values, if the Synchronization is not disabled by the SYNCDIS bit in the LIN Mode Register (FLEX_US_LINMR).

After reception of the Synch Field:

- If it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%), then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the error flag bit LINSTE in FLEX_US_CSR is set.
- If it appears that the sampled Synch character is not equal to 0x55, then the clock divider (LINCD) and the fractional part (LINFP) are not updated, and the error flag bit LINISFE in FLEX_US_CSR is set.

Flags LINSTE and LINISFE are reset by writing a one to the RSTSTA bit in FLEX_US_CR.

Figure 44-44. Slave Node Synchronization



The accuracy of the synchronization depends on several parameters:

- The nominal clock frequency (f_{Nom}) (the theoretical slave node clock frequency)
- The Baud Rate
- The oversampling (OVER = 0 => 16X or OVER = 1 => 8X)

The following formula is used to compute the deviation of the slave bit rate relative to the master bit rate after synchronization (f_{SLAVE} is the real slave node clock frequency).

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{Over}) + \beta] \times \text{Baud rate}}{8 \times f_{SLAVE}}\right)\%$$

$$\text{Baud rate deviation} = \left(100 \times \frac{[\alpha \times 8 \times (2 - \text{Over}) + \beta] \times \text{Baud rate}}{8 \times \left(\frac{f_{TOL_UNSYNCH}}{100}\right) \times f_{Nom}}\right)\%$$

$$-0.5 \le \alpha \le +0.5 \quad -1 < \beta < +1$$

$f_{TOL_UNSYNCH}$ is the deviation of the real slave node clock from the nominal clock frequency. The LIN Standard imposes that it must not exceed ±15%. The LIN Standard imposes also that for communication between two nodes, their bit rate must not differ by more than ±2%. This means that the baud rate deviation must not exceed ±1%.

It follows from that, a minimum value for the nominal clock frequency:

$$f_{Nom}(\text{min}) = \left(100 \times \frac{[0.5 \times 8 \times (2 - \text{Over}) + 1] \times \text{Baud rate}}{8 \times \left(\frac{-15}{100} + 1\right) \times 1\%}\right)\text{Hz}$$

Examples:

- Baud rate = 20 kbit/s, OVER = 0 (Oversampling 16X) => f_{Nom}(min) = 2.64 MHz
- Baud rate = 20 kbit/s, OVER = 1 (Oversampling 8X) => f_{Nom}(min) = 1.47 MHz
- Baud rate = 1 kbit/s, OVER = 0 (Oversampling 16X) => f_{Nom}(min) = 132 kHz
- Baud rate = 1 kbit/s, OVER = 1 (Oversampling 8X) => f_{Nom}(min) = 74 kHz

44.7.9.9 Identifier Parity

A protected identifier consists of two subfields; the identifier and the identifier parity. Bits 0 to 5 are assigned to the identifier and bits 6 and 7 are assigned to the parity.

The USART interface can generate/check these parity bits, but this feature can also be disabled. The user can choose between two modes by the PARDIS bit of FLEX_US_LINMR:

- PARDIS = 0:
  - During header transmission, the parity bits are computed and sent with the six least significant bits of the IDCHR field of the LIN Identifier Register (FLEX_US_LINIR). The bits 6 and 7 of this register are discarded.
  - During header reception, the parity bits of the identifier are checked. If the parity bits are wrong, an Identifier Parity error occurs (see Section 44.7.3.8). Only the six least significant bits of the IDCHR field are updated with the received Identifier. The bits 6 and 7 are stuck to 0.
- PARDIS = 1:
  - During header transmission, all the bits of the IDCHR field of the LIN Identifier Register (FLEX_US_LINIR) are sent on the bus.
  - During header reception, all the bits of the IDCHR field are updated with the received Identifier.

44.7.9.10 Node Action

Depending on the identifier, the node is affected—or not—by the LIN response. Consequently, after sending or receiving the identifier, the USART must be configured. There are three possible configurations:

- PUBLISH: the node sends the response.
- SUBSCRIBE: the node receives the response.
- IGNORE: the node is not concerned by the response, it does not send and does not receive the response.

This configuration is made by the LIN Node Action (NACT) field in FLEX_US_LINMR (see Section 44.10.31).

Example: a LIN cluster that contains a master and two slaves:

- Data transfer from the master to slave 1 and to slave 2:

  NACT(master) = PUBLISH

  NACT(slave 1) = SUBSCRIBE

  NACT(slave 2) = SUBSCRIBE
- Data transfer from the master to slave 1 only:

  NACT(master) = PUBLISH

  NACT(slave 1) = SUBSCRIBE

  NACT(slave 2) = IGNORE
- Data transfer from slave 1 to the master:

  NACT(master) = SUBSCRIBE

  NACT(slave 1) = PUBLISH

  NACT(slave 2) = IGNORE
- Data transfer from slave 1 to slave 2:

  NACT(master) = IGNORE

  NACT(slave 1) = PUBLISH

  NACT(slave 2) = SUBSCRIBE
- Data transfer from slave 2 to the master and to slave 1:

  NACT(master) = SUBSCRIBE

  NACT(slave 1) = SUBSCRIBE

  NACT(slave 2) = PUBLISH

44.7.9.11 Response Data Length

The LIN response data length is the number of data fields (bytes) of the response excluding the checksum.

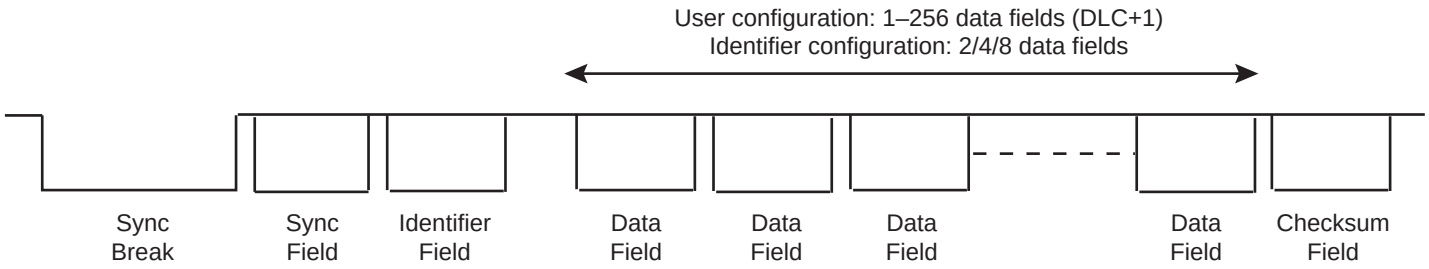
The response data length can either be configured by the user or be defined automatically by bits 4 and 5 of the Identifier (compatibility to LIN Specification 1.1). The user can choose between these two modes by the DLM bit of FLEX_US_LINMR:

- DLM = 0: The response data length is configured by the user via the DLC field of FLEX_US_LINMR. The response data length is equal to (DLC + 1) bytes. DLC can be programmed from 0 to 255, so the response can contain from 1 data byte up to 256 data bytes.
- DLM = 1: The response data length is defined by the Identifier (IDCHR in FLEX_US_LINIR) according to the table below. The DLC field of FLEX_US_LINMR is discarded. The response can contain 2 or 4 or 8 data bytes.

Table 44-15. Response Data Length if DLM = 1

IDCHR[5]	IDCHR[4]	Response Data Length (bytes)
0	0	2
0	1	2
1	0	4
1	1	8

Figure 44-45. Response Data Length



44.7.9.12 Checksum

The last field of a frame is the checksum. The checksum contains the inverted 8- bit sum with carry, over all data bytes or all data bytes and the protected identifier. Checksum calculation over the data bytes only is called classic checksum and it is used for communication with LIN 1.3 slaves. Checksum calculation over the data bytes and the protected identifier byte is called enhanced checksum and it is used for communication with LIN 2.0 slaves.

The USART can be configured to:

- Send/Check an Enhanced checksum automatically (CHKDIS = 0 & CHKTYP = 0)
- Send/Check a Classic checksum automatically (CHKDIS = 0 & CHKTYP = 1)
- Not send/check a checksum (CHKDIS = 1)

This configuration is made by the Checksum Type (CHKTYP) and Checksum Disable (CHKDIS) bits of FLEX_US_LINMR.

If the checksum feature is disabled, the user can send it manually all the same, by considering the checksum as a normal data byte and by adding 1 to the response data length (see Section 44.7.9.11).

44.7.9.13 Frame Slot Mode

This mode is useful only for master nodes. It respects the following rule: each frame slot shall be longer than or equal to $t_{Frame_Maximum}$.

If the Frame Slot mode is enabled (FSDIS = 0) and a frame transfer has been completed, the TXRDY flag is set again only after $t_{Frame_Maximum}$ delay, from the start of frame. So the master node cannot send a new header if the frame slot duration of the previous frame is inferior to $t_{Frame_Maximum}$.

If the Frame Slot mode is disabled (FSDIS = 1) and a frame transfer has been completed, the TXRDY flag is set again immediately.

The $t_{Frame_Maximum}$ is calculated as follows:

If the Checksum is sent (CHKDIS = 0):

- $t_{Header_Nominal} = 34 \times t_{bit}$
- $t_{Response_Nominal} = 10 \times (NData + 1) \times t_{bit}$
- $t_{Frame_Maximum} = 1.4 \times (t_{Header_Nominal} + t_{Response_Nominal} + 1)$ (1)
- $t_{Frame_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1 + 1) + 1) \times t_{bit}$

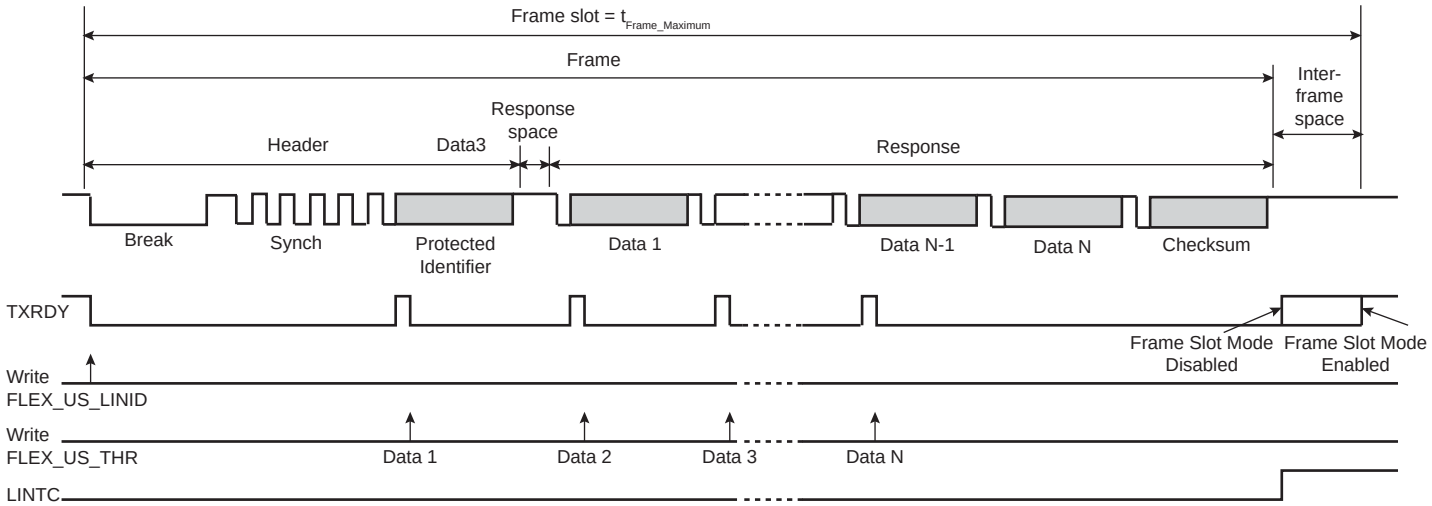
- $t_{Frame_Maximum} = (77 + 14 \times DLC) \times t_{bit}$

If the Checksum is not sent (CHKDIS = 1):

- $t_{Header_Nominal} = 34 \times t_{bit}$
- $t_{Response_Nominal} = 10 \times NData \times t_{bit}$
- $t_{Frame_Maximum} = 1.4 \times (t_{Header_Nominal} + t_{Response_Nominal} + 1)$(1)
- $t_{Frame_Maximum} = 1.4 \times (34 + 10 \times (DLC + 1) + 1) \times t_{bit}$
- $t_{Frame_Maximum} = (63 + 14 \times DLC) \times t_{bit}$

Note: 1. The term “+1” leads to an integer result for $t_{Frame_Maximum}$ (LIN Specification 1.3).

Figure 44-46. Frame Slot Mode



44.7.9.14 LIN Errors

Bit Error

This error is generated in master of slave node configuration, when the USART is transmitting and if the transmitted value on the Tx line is different from the value sampled on the Rx line. If a bit error is detected, the transmission is aborted at the next byte border.

This error is reported by flag LINBE in FLEX_US_CSR.

Inconsistent Synch Field Error

This error is generated in slave node configuration, if the Synch Field character received is other than 0x55.

This error is reported by flag LINISFE in FLEX_US_CSR.

Identifier Parity Error

This error is generated in slave node configuration, if the parity of the identifier is wrong. This error can be generated only if the parity feature is enabled (PARDIS = 0).

This error is reported by flag LINIPE in FLEX_US_CSR.

Checksum Error

This error is generated in master of slave node configuration, if the received checksum is wrong. This flag can be set to 1 only if the checksum feature is enabled (CHKDIS = 0).

This error is reported by flag LINCE in FLEX_US_CSR.

Slave Not Responding Error

This error is generated in master of slave node configuration, when the USART expects a response from another node (NACT = SUBSCRIBE) but no valid message appears on the bus within the time given by the maximum length of the message frame, $t_{Frame_Maximum}$ (see Section 44.7.9.13). This error is disabled if the USART does not expect any message (NACT = PUBLISH or NACT = IGNORE).

This error is reported by flag LINSNRE in FLEX_US_CSR.

Synch Tolerance Error

This error is generated in slave node configuration if, after the clock synchronization procedure, it appears that the computed baud rate deviation compared to the initial baud rate is superior to the maximum tolerance FTol_Unsynch (±15%).

This error is reported by flag LINSTE in FLEX_US_CSR.

Header Timeout Error

This error is generated in slave node configuration, if the Header is not entirely received within the time given by the maximum length of the Header, $t_{Header_Maximum}$.

This error is reported by flag LINHTE in FLEX_US_CSR.

44.7.9.15 LIN Frame Handling

Master Node Configuration

- Write TXEN and RXEN in FLEX_US_CR to enable both the transmitter and the receiver.
- Write USART_MODE in FLEX_US_MR to select the LIN mode and the master node configuration.
- Write CD and FP in FLEX_US_BRGR to configure the baud rate.
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM, FSDIS and DLC in FLEX_US_LINMR to configure the frame transfer.
- Check that TXRDY in FLEX_US_CSR is set to 1
- Write IDCHR in FLEX_US_LINIR to send the header

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the USART sends the response
  - Wait until TXRDY in FLEX_US_CSR rises
  - Write TCHR in FLEX_US_THR to send a byte
  - If all the data have not been written, redo the two previous steps
  - Wait until LINTC in FLEX_US_CSR rises
  - Check the LIN errors
- Case 2: NACT = SUBSCRIBE, the USART receives the response
  - Wait until RXRDY in FLEX_US_CSR rises
  - Read RCHR in FLEX_US_RHR
  - If all the data have not been read, redo the two previous steps
  - Wait until LINTC in FLEX_US_CSR rises
  - Check the LIN errors
- Case 3: NACT = IGNORE, the USART is not concerned by the response
  - Wait until LINTC in FLEX_US_CSR rises
  - Check the LIN errors

Figure 44-47. Master Node Configuration, NACT = PUBLISH

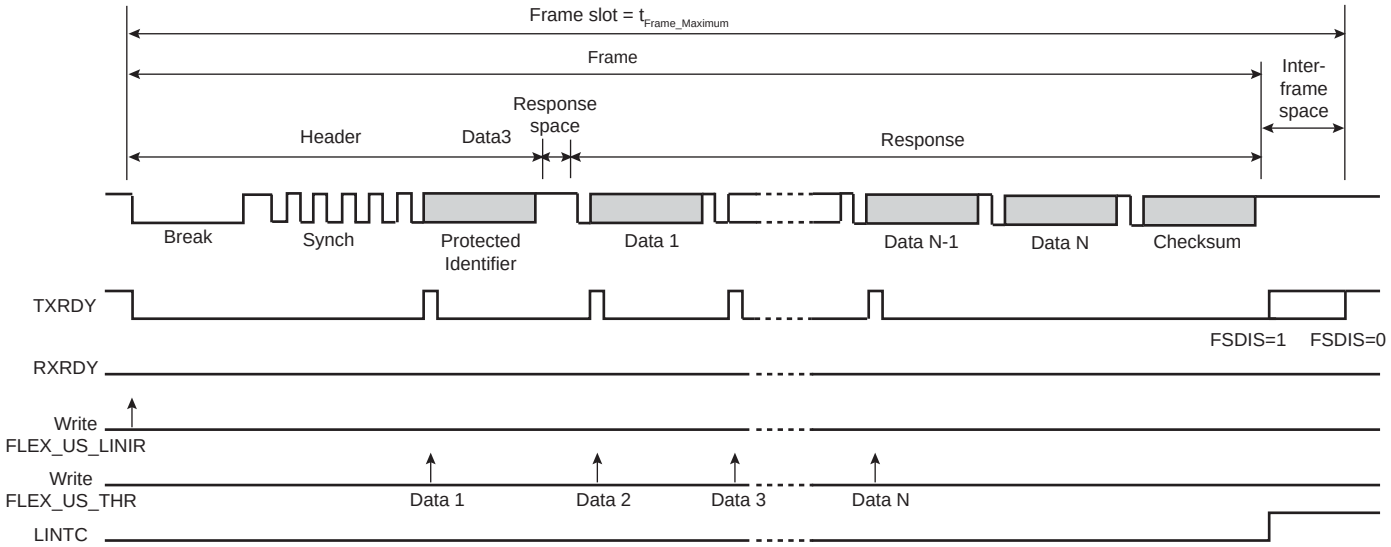


Figure 44-48. Master Node Configuration, NACT = SUBSCRIBE

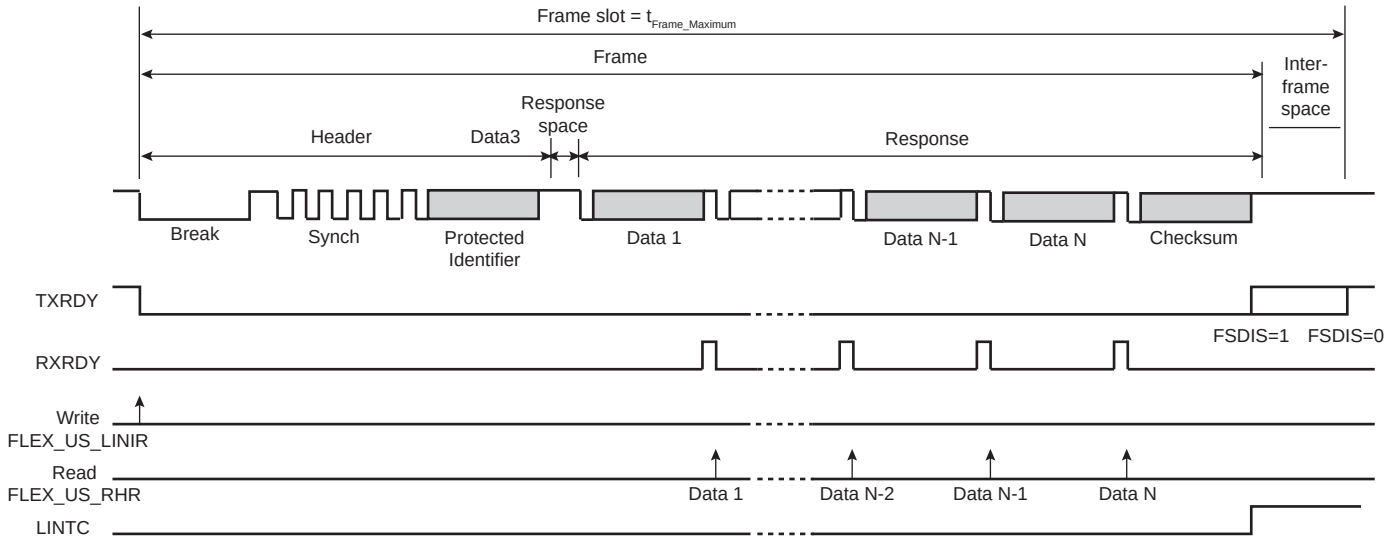
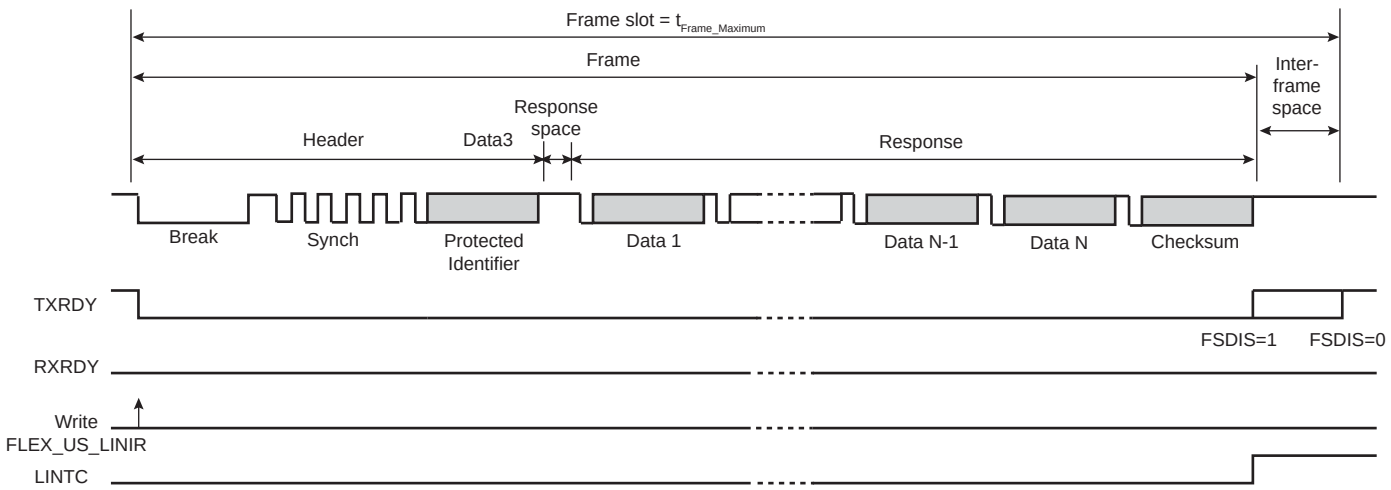


Figure 44-49. Master Node Configuration, NACT = IGNORE



Slave Node Configuration

- Write TXEN and RXEN in FLEX_US_CR to enable both the transmitter and the receiver.
- Write USART_MODE in FLEX_US_MR to select the LIN mode and the slave node configuration.
- Write CD and FP in FLEX_US_BRGR to configure the baud rate.
- Wait until LINID in FLEX_US_CSR rises
- Check LINISFE and LINPE errors
- Read IDCHR in FLEX_US_RHR
- Write NACT, PARDIS, CHKDIS, CHKTYPE, DLCM and DLC in FLEX_US_LINMR to configure the frame transfer.

IMPORTANT: If the NACT configuration for this frame is PUBLISH, FLEX_US_LINMR must be written with NACT = PUBLISH even if this field is already correctly configured, in order to set the TXREADY flag and the corresponding write transfer request.

What comes next depends on the NACT configuration:

- Case 1: NACT = PUBLISH, the LIN controller sends the response
  - Wait until TXRDY in FLEX_US_CSR rises
  - Write TCHR in FLEX_US_THR to send a byte
  - If all the data have not been written, redo the two previous steps
  - Wait until LINTC in FLEX_US_CSR rises
  - Check the LIN errors
- Case 2: NACT = SUBSCRIBE, the USART receives the response
  - Wait until RXRDY in FLEX_US_CSR rises
  - Read RCHR in FLEX_US_RHR
  - If all the data have not been read, redo the two previous steps
  - Wait until LINTC in FLEX_US_CSR rises
  - Check the LIN errors
- Case 3: NACT = IGNORE, the USART is not concerned by the response
  - Wait until LINTC in FLEX_US_CSR rises
  - Check the LIN errors

Figure 44-50. Slave Node Configuration, NACT = PUBLISH

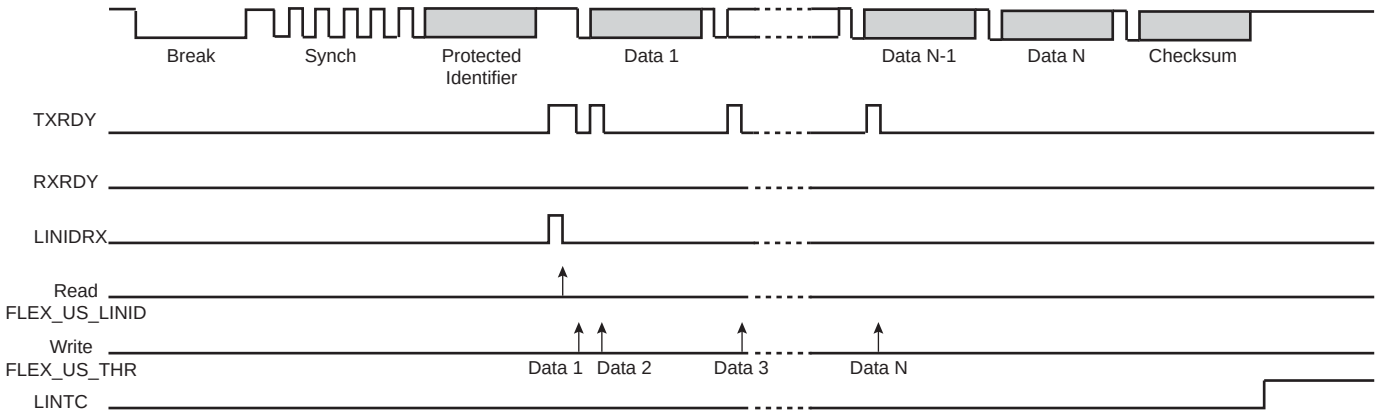


Figure 44-51. Slave Node Configuration, NACT = SUBSCRIBE

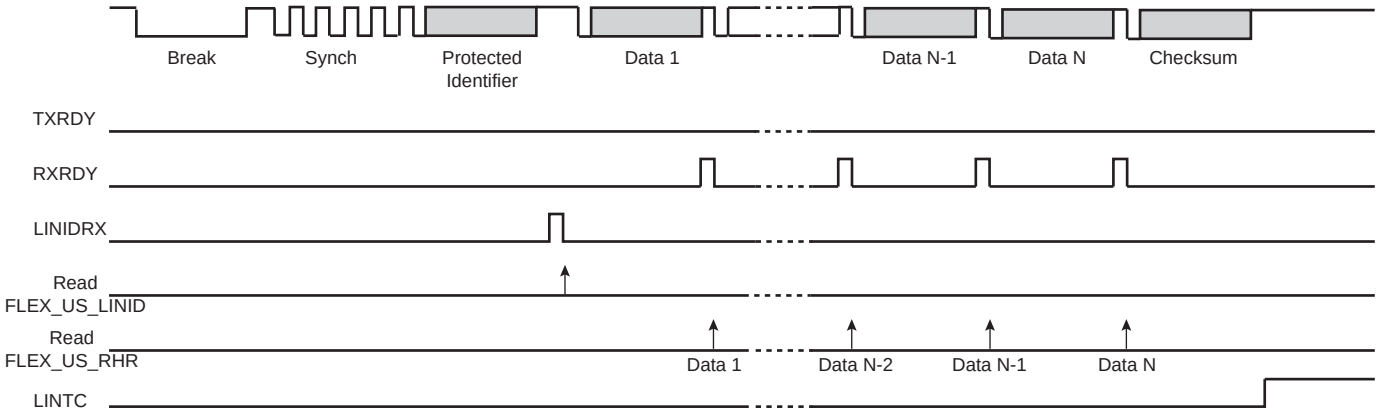
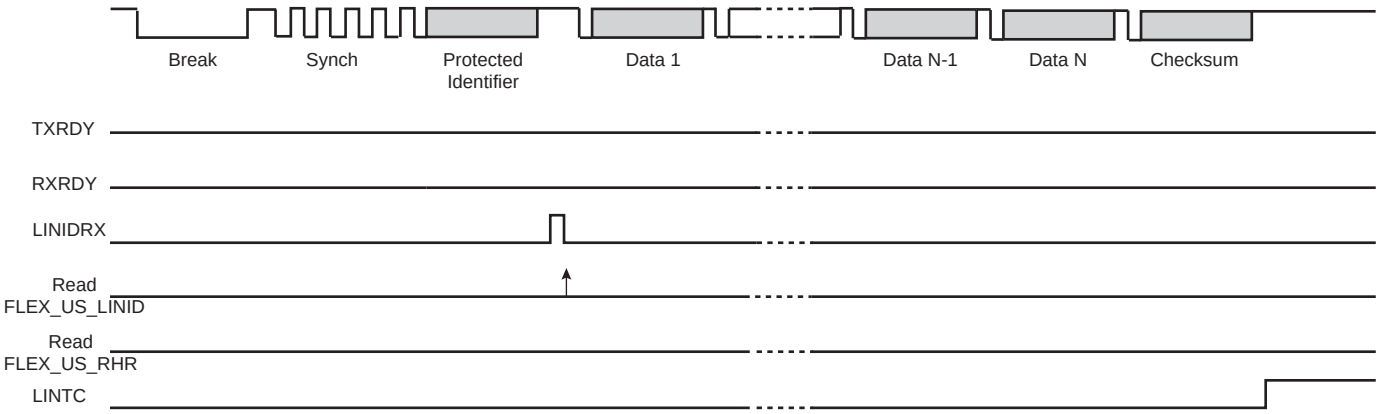


Figure 44-52. Slave Node Configuration, NACT = IGNORE



44.7.9.16 LIN Frame Handling With The DMAC

The USART can be used in association with the DMAC in order to transfer data directly into/from the on- and off-chip memories without any processor intervention.

The DMAC uses the trigger flags, TXRDY and RXRDY, to write or read into the USART. The DMAC always writes in the Transmit Holding Register (FLEX_US_THR) and it always reads in the Receive Holding Register (FLEX_US_RHR). The size of the data written or read by the DMAC in the USART is always a byte.

Master Node Configuration

The user can choose between two DMAC modes by configuring the PDCM bit in FLEX_US_LINMR:

- PDCM = 1: The LIN configuration is stored in the WRITE buffer and it is written by the DMAC in the Transmit Holding register FLEX_US_THR (instead of the LIN Mode register FLEX_US_LINMR). Because the DMAC transfer size is limited to a byte, the transfer is split into two accesses. During the first access the bits, NACT, PARDIS, CHKDIS, CHKTYP, DLM and FSDIS are written. During the second access the 8-bit DLC field is written.
- PDCM = 0: The LIN configuration is not stored in the WRITE buffer and it must be written by the user in FLEX_US_LINMR.

The WRITE buffer also contains the Identifier and the DATA, if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the DATA if the USART receives the response (NACT = SUBSCRIBE).

Figure 44-53. Master Node with DMAC (PDCM = 1)

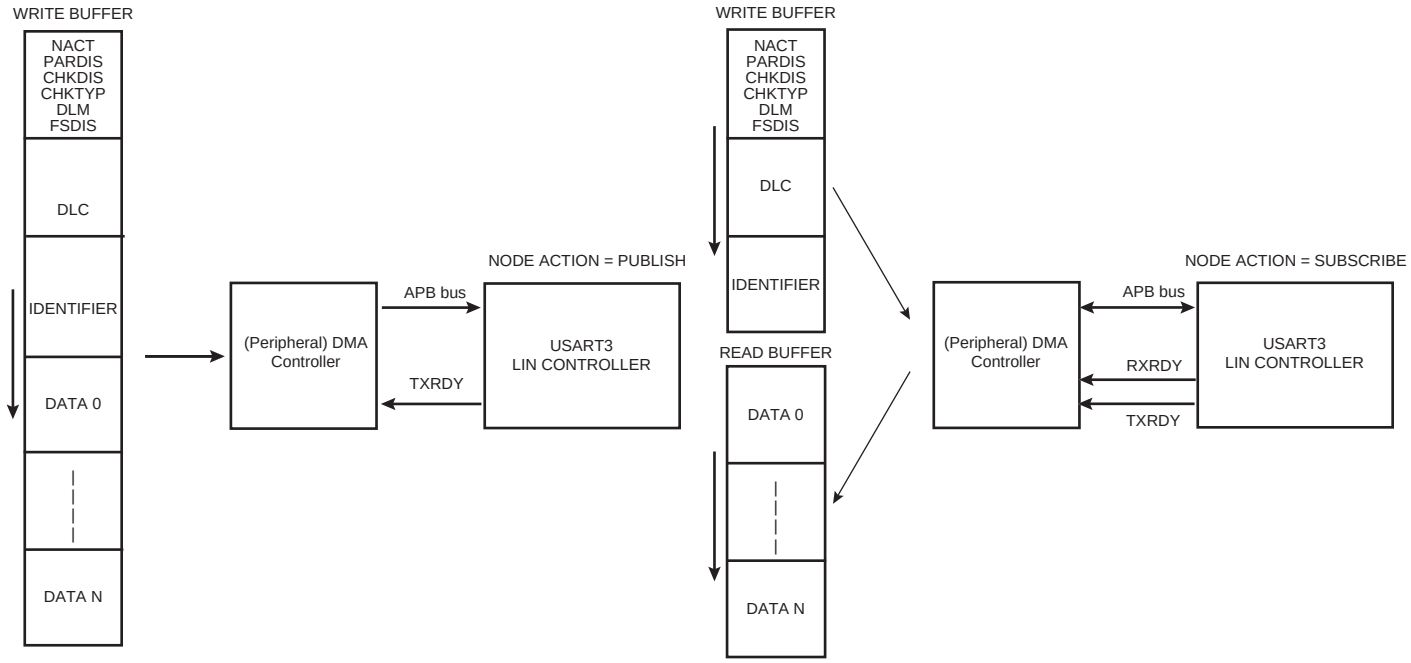
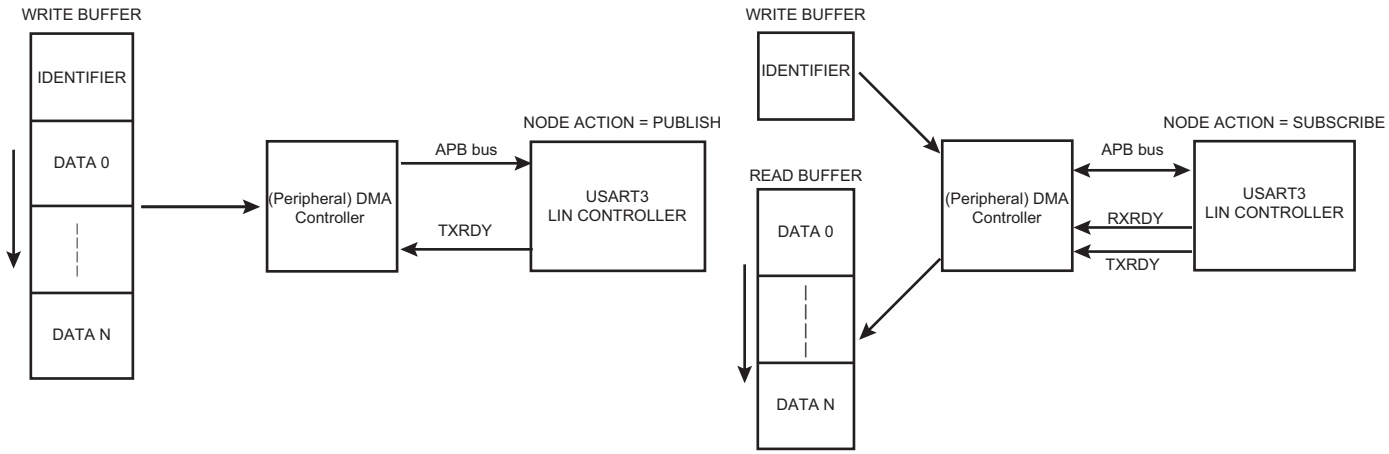


Figure 44-54. Master Node with DMAC (PDCM = 0)



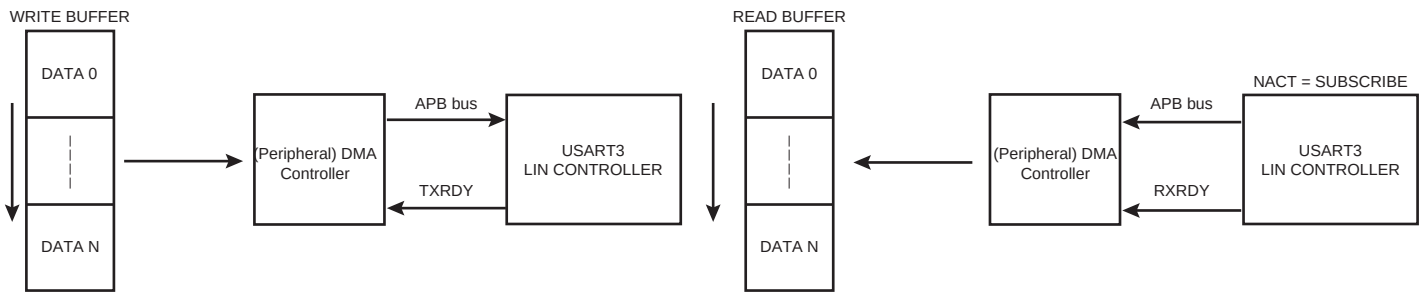
Slave Node Configuration

In this configuration, the DMAC transfers only the DATA. The Identifier must be read by the user in the LIN Identifier Register (FLEX_US_LINIR). The LIN mode must be written by the user in FLEX_US_LINMR.

The WRITE buffer contains the DATA if the USART sends the response (NACT = PUBLISH).

The READ buffer contains the DATA if the USART receives the response (NACT = SUBSCRIBE).

Figure 44-55. Slave Node with DMAC



44.7.9.17 Wakeup Request

Any node in a sleeping LIN cluster may request a wakeup.

In the LIN 2.0 specification, the wakeup request is issued by forcing the bus to the dominant state from 250 µs to 5 ms. For this, it is necessary to send the character 0xF0 in order to impose five successive dominant bits. Whatever the baud rate is, this character respects the specified timings.

- Baud rate min = 1 kbit/s -> t_{bit} = 1 ms -> 5 t_{bit} = 5 ms
- Baud rate max = 20 kbit/s -> t_{bit} = 50 µs -> 5 t_{bit} = 250 µs

In the LIN 1.3 specification, the wakeup request should be generated with the character 0x80 in order to impose eight successive dominant bits.

The user can choose by the WKUPTYP bit in FLEX_US_LINMR either to send a LIN 2.0 wakeup request (WKUPTYP = 0) or to send a LIN 1.3 wakeup request (WKUPTYP = 1).

A wakeup request is transmitted by writing FLEX_US_CR with the LINWKUP bit to 1. Once the transfer is completed, the LINTC flag is asserted in the Status Register (FLEX_US_SR). It is cleared by writing a one to the RSTSTA bit in FLEX_US_CR.

44.7.9.18 Bus Idle Timeout

If the LIN bus is inactive for a certain duration, the slave nodes shall automatically enter in Sleep mode. In the LIN 2.0 specification, this timeout is defined as 4 seconds. In the LIN 1.3 specification, it is defined as 25,000 t_{bit}.

In slave Node configuration, the receiver timeout detects an idle condition on the RXD line. When a timeout is detected, the TIMEOUT bit in FLEX_US_CSR rises and can generate an interrupt, thus indicating to the driver to go into Sleep mode.

The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of FLEX_US_RTOR. If a zero is written to the TO field, the Receiver Timeout is disabled and no timeout is detected. The TIMEOUT bit in FLEX_US_CSR remains at 0. Otherwise, the receiver loads a 17-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the TIMEOUT bit in FLEX_US_CSR rises.

If STTTO is performed, the counter clock is stopped until a first character is received.

If RETTO is performed, the counter starts counting down immediately from the value TO.

Table 44-16. Receiver Timeout Programming

LIN Specification	Baud Rate	Timeout period	TO
2.0	1,000 bit/s	4s	4,000
	2,400 bit/s		9,600
	9,600 bit/s		38,400
	19,200 bit/s		76,800
	20,000 bit/s		80,000
1.3	–	25,000 t_{bit}	25,000

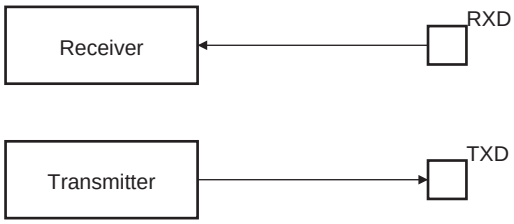
44.7.10 Test Modes

The USART can be programmed to operate in three different test modes. The internal loopback capability allows on-board diagnostics. In Loopback mode, the USART interface pins are disconnected or not and reconfigured for loopback internally or externally.

44.7.10.1 Normal Mode

Normal mode connects the RXD pin on the receiver input and the transmitter output on the TXD pin.

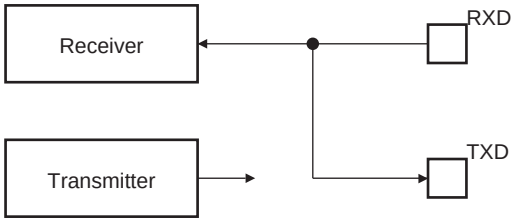
Figure 44-56. Normal Mode Configuration



44.7.10.2 Automatic Echo Mode

Automatic Echo mode allows bit-by-bit retransmission. When a bit is received on the RXD pin, it is sent to the TXD pin, as shown in Figure 44-57. Programming the transmitter has no effect on the TXD pin. The RXD pin is still connected to the receiver input, thus the receiver remains active.

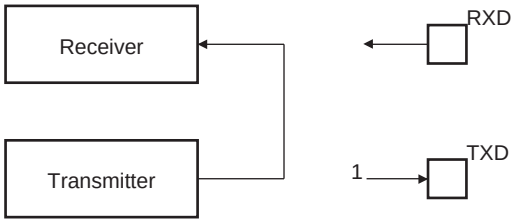
Figure 44-57. Automatic Echo Mode Configuration



44.7.10.3 Local Loopback Mode

Local Loopback mode connects the output of the transmitter directly to the input of the receiver, as shown in Figure 44-58. The TXD and RXD pins are not used. The RXD pin has no effect on the receiver and the TXD pin is continuously driven high, as in idle state.

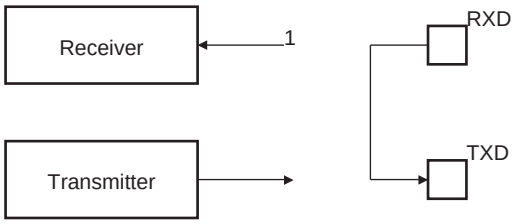
Figure 44-58. Local Loopback Mode Configuration



44.7.10.4 Remote Loopback Mode

Remote Loopback mode directly connects the RXD pin to the TXD pin, as shown in Figure 44-59. The transmitter and the receiver are disabled and have no effect. This mode allows bit-by-bit retransmission.

Figure 44-59. Remote Loopback Mode Configuration

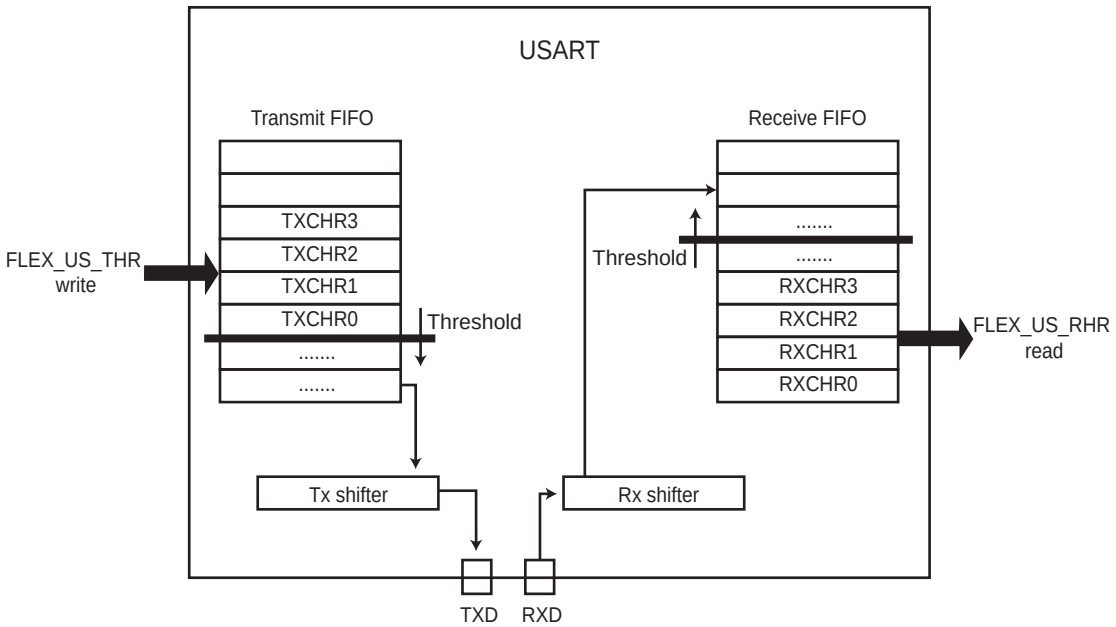


44.7.11 FIFOs

The USART includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in FLEX_US_CR. It is recommended to disable both the transmitter and the receiver before enabling or disabling the FIFOs (TXDIS and RXDIS bit in FLEX_US_CR).

Writing a ‘1’ to the FIFOEN bit enables a 32-data Transmit FIFO and a 32-data Receive FIFO.

Figure 44-60. FIFOs Block Diagram

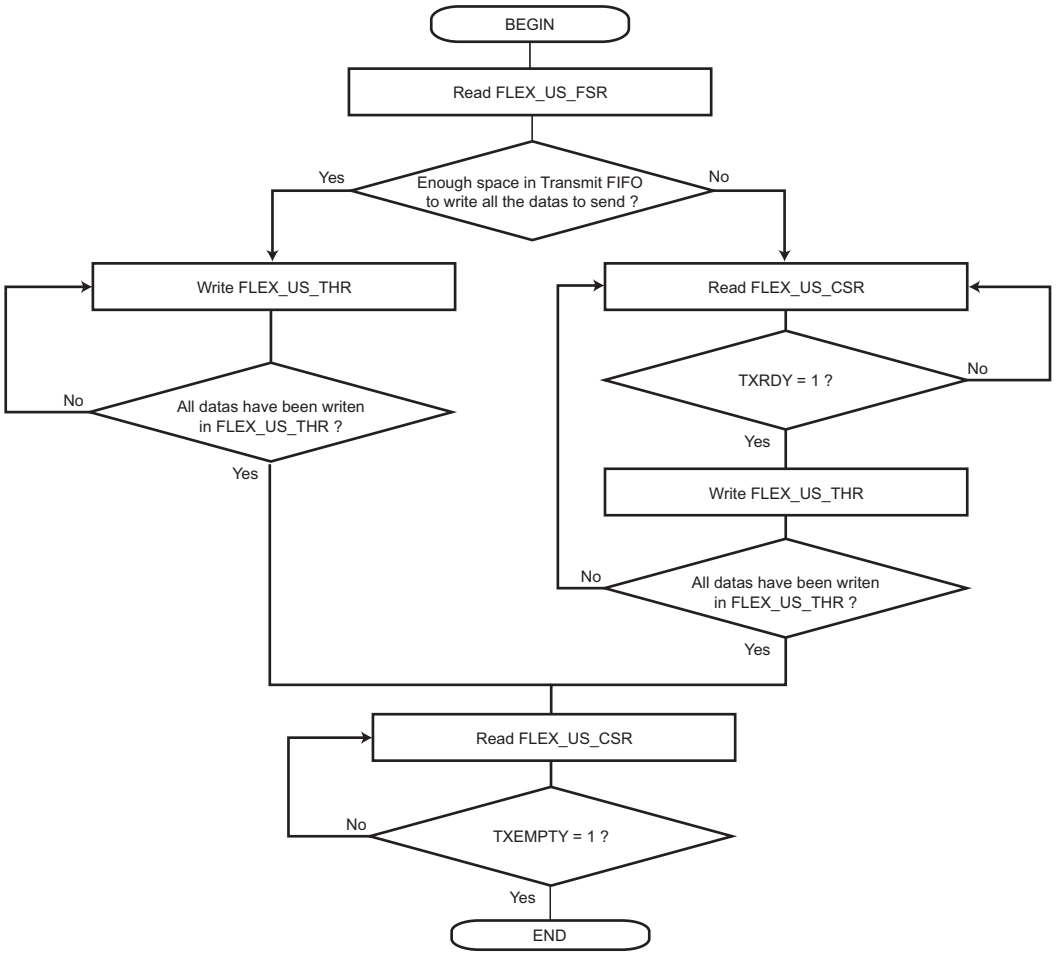


44.7.11.1 Sending Data with FIFO Enabled

With the Transmit FIFO enabled, any write access to FLEX_US_THR loads the written data to the Transmit FIFO. As a consequence it is not mandatory any more to monitor the TXRDY flag state to send multiple data without DMAC.

Knowing the number of data to send, and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in FLEX_US_THR without checking the TXRDY flag between each access. The Transmit FIFO state can be checked reading the TXFL field in the USART FIFO Level Register (FLEX_US_FLR).

Figure 44-61. Sending Data with FIFO Flowchart

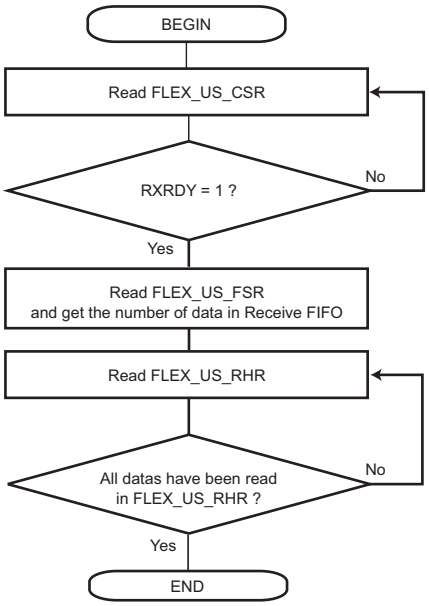


44.7.11.2 Receiving Data with FIFO Enabled

With Receive FIFO enabled, any read access on FLEX_US_RHR pulls out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RXRDY flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with the RXFL field in FLEX_US_FLR and all the data read successively in FLEX_US_RHR without checking the RXRDY flag between each access.

Figure 44-62. Receiving Data with FIFO Flowchart



44.7.11.3 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in FLEX_US_CR.

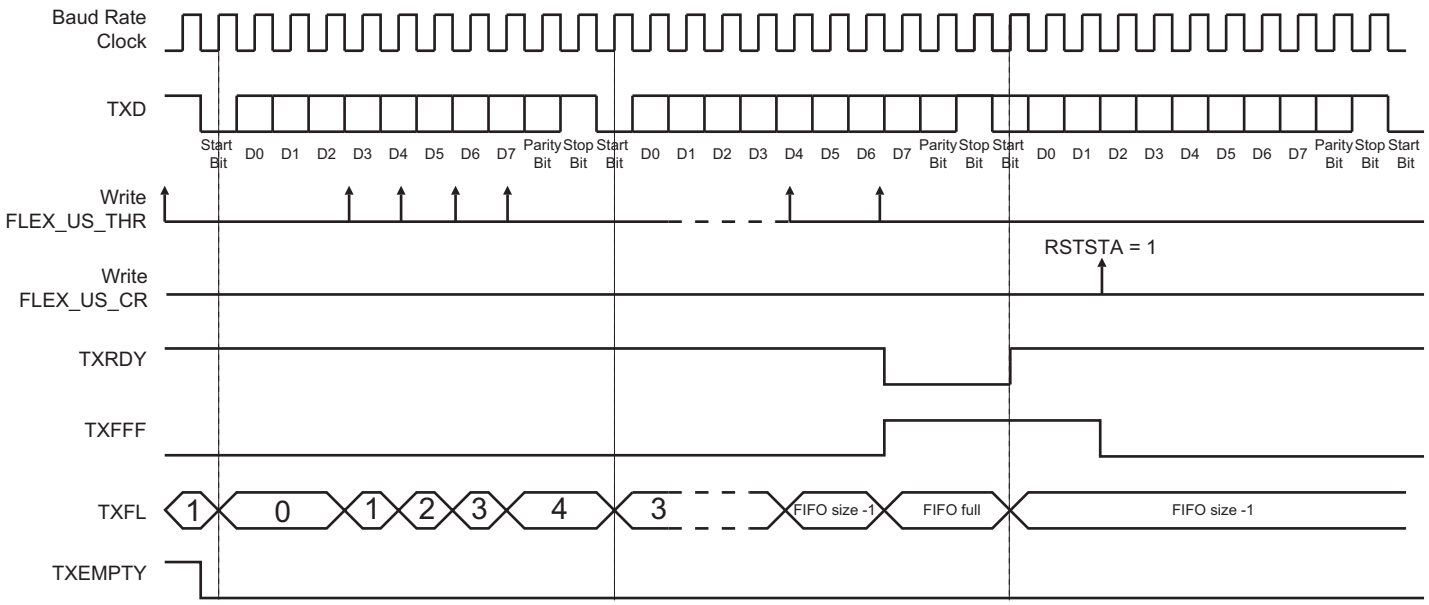
44.7.11.4 TXRDY, RXRDY and TXEMPTY Behavior

If FIFOs are enabled, the behavior of the TXRDY, RXRDY and TXEMPTY flags is slightly different.

With FIFO enabled, the TXEMPTY flag remains at ‘0’ state as long as there are characters in the Transmit FIFO or in the Transmit Shift Register. It is at ‘1’ state when there are no character in the Transmit FIFO and no character in the Transmit Shift Register.

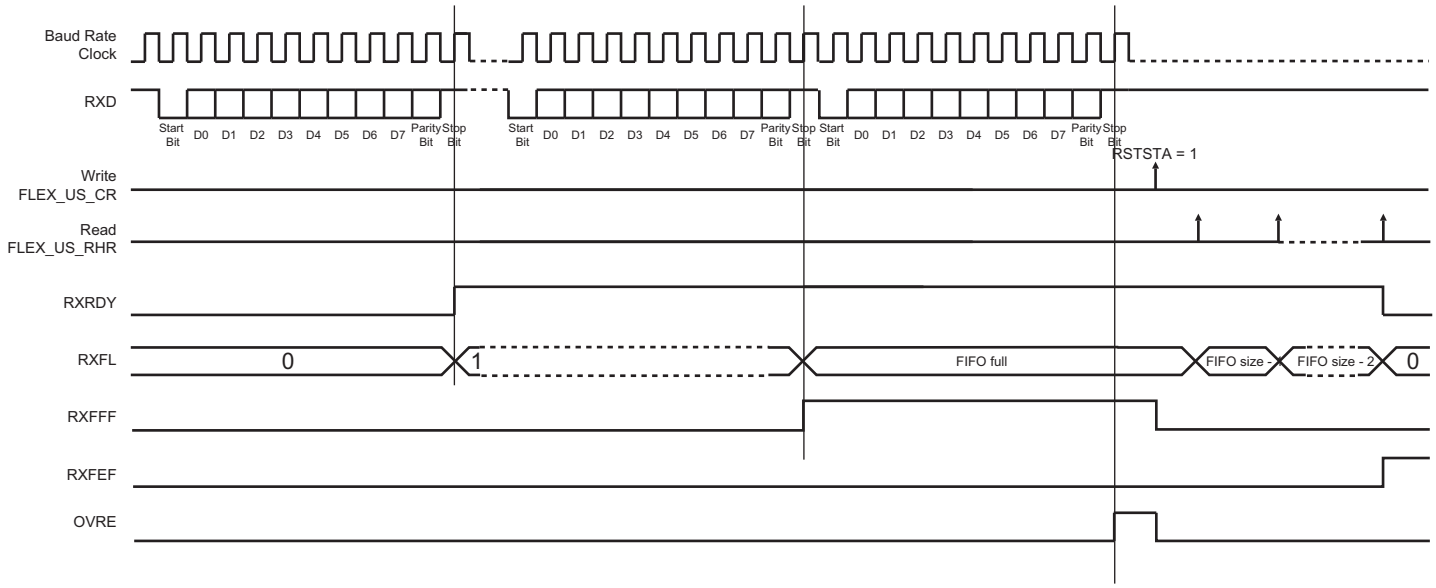
TXRDY indicates if a data can be written in the Transmit FIFO. Then, by default, the TXRDY flag remains at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 44-63. TXRDY in Single Data Mode and TXRDYM = 0x0



RXRDY indicates if an unread data is present in the Receive FIFO. Then, by default, the RXRDY flag is at level ‘1’ as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 44-64. RXRDY in Single Data Mode and RXRDYM = 0x0



TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the USART FIFO Mode Register (FLEX_US_MR). In Single Data mode, there is no need to modify the TXRDY and RXRDY behavior; however it may be useful in Multiple Data mode.

44.7.11.5 Single Data Mode

If the MODE9 bit is set, or if the USART_MODE field is set to either LIN_MASTER or LIN_SLAVE, the FIFOs operate in Single Data mode.

In this mode, only one data can be written/read per write/read access of FLEX_US_THR/FLEX_US_RHR (see Section 44.10.20 “USART Receive Holding Register” and Section 44.10.22 “USART Transmit Holding Register”). On the other hand TXRDYM and RXRDYM fields should be configured with 0x0 value in this mode.

DMAC

The TXRDYM and RXRDYM fields must be configured with 0x0 value when working with DMAC transfer in Single Data mode.

The same DMAC procedure applies as with FIFO disabled.

44.7.11.6 Multiple Data Mode

When the FIFOs do not operate in Single Data mode, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_US_THR/FLEX_US_RHR access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read, if the access is a halfword size register access, then two data are written/read and, finally, if the access is a word-size register access, four data are written/read.

Written/Read data are always right-aligned, as shown in Section 44.10.21 “USART Receive Holding Register (FIFO Multi Data)” and Section 44.10.23 “USART Transmit Holding Register (FIFO Multi Data)”.

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six FLEX_US_THR byte write accesses.
- Perform three FLEX_US_THR halfword write accesses.

- Perform one FLEX_US_THR word write access and one FLEX_US_THR halfword write access.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six FLEX_US_RHR byte read accesses.
- Perform three FLEX_US_RHR halfword read accesses.
- Perform one FLEX_US_RHR word read access and one FLEX_US_RHR halfword read access.

This mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

TXRDY and RXRDY Configuration

In Multiple Data mode, the TXRDYM and RXRDYM fields in FLEX_US_FMR become useful.

As in Multiple Data mode, it is possible to write several data in the same access it might be useful to configure TXRDY flag behavior to indicate if 1, 2 or 4 data can be written in the FIFO depending on the access to perform on FLEX_US_THR.

If, for instance, four data are written each time in FLEX_US_THR, it might be useful to configure the TXRDYM field to value 0x2 so that the TXRDY flag is at ‘1’ only when at least four data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_US_RHR, it might be useful to configure the RXRDYM field to value 0x2 so that the RXRDY flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

DMAC

If DMAC transfer is used, it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

44.7.11.7 Transmit FIFO Lock

- LIN Mode:

If a frame is aborted using the Abort LIN Transmission bit (FLEX_US_CR.LINABT), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, reset DMAC channels, etc., without any risk.

The TXFLOCK bit in the USART FIFO Event Status Register (FLEX_US_FESR) is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting the TXFLCLR bit in FLEX_US_CR.

44.7.11.8 FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

If the Transmit FIFO is full and a write access is performed on FLEX_US_THR, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in FLEX_US_FESR.

In Multiple Data mode, if the number of data written in FLEX_US_THR (according to the register access size) is bigger than the Transmit FIFO free space, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in FLEX_US_FESR.

- Receive FIFO:

In Multiple Data mode, if the number of data read in FLEX_US_RHR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it generates a Receive FIFO pointer error and sets the RXFPTEF flag in FLEX_US_FESR.

No pointer error should happen if the FIFO state is checked before writing/reading in FLEX_US_THR/FLEX_US_RHR. The FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a Transmit pointer error occurs, the transmitter must be reset through the RSTTX bit in FLEX_US_CR.

If a Receive pointer error occurs, the receiver must be reset through the RSTRX bit in FLEX_US_CR.

44.7.11.9 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO:

The Transmit FIFO threshold can be set using the TXFTHRES field in FLEX_US_FMR. Each time the Transmit FIFO goes from the 'above threshold' to the 'equal or below threshold' state, the TXFTHF flag in FLEX_US_FESR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.

- Receive FIFO:

The Receive FIFO threshold can be set using the RXFTHRES field in FLEX_US_FMR. Each time the Receive FIFO goes from the 'below threshold' to the 'equal to or above threshold' state, the RXFTHF flag in FLEX_US_FESR is set. This enables the application to know that the Receive FIFO reached the defined threshold and read some data before it becomes full. The Receive FIFO threshold 2 can be set with RXFTHRES2 field in FLEX_US_FMR. Each time the Receive FIFO goes from the 'above threshold 2' to the 'equal or below threshold 2' state, the RXFTHF2 flag in FLEX_US_FESR is set. This enables the application to know that the Receive FIFO reached the defined threshold.

The RXFTHF, RXFTHF and RXTHF2 flags can be configured to generate an interrupt using FLEX_US_FIER and FLEX_US_FIDR.

44.7.11.10 FIFO Flags

FIFOs come with a set of flags which can be configured each to generate an interrupt through FLEX_US_FIER and FLEX_US_FIDR.

FIFO flags state can be read in FLEX_US_FESR. They are cleared writing to '1' the RSTSTA bit in FLEX_US_CR.

44.7.12 USART Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_USART to enable access to the write protection registers.

To prevent any single software error from corrupting USART behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable) bit in the USART Write Protection Mode Register (FLEX_US_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the USART Write Protection Status Register (FLEX_US_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_US_WPSR.

The following registers can be write-protected when WPEN is set:

- USART Mode Register
- USART Baud Rate Generator Register
- USART Receiver Timeout Register
- USART Transmitter Timeguard Register
- USART FI DI RATIO Register
- USART IrDA FILTER Register
- USART Manchester Configuration Register
- USART Comparison Register

44.8 SPI Functional Description

44.8.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by writing a 1 to the MSTR bit in the SPI Mode Register (FLEX_SPI_MR):
  - The pins NPCS0 to NPCS1 are all configured as outputs
  - The SPCK pin is driven
  - The MISO line is wired on the receiver input
  - The MOSI line is driven as an output by the transmitter.
- The SPI operates in Slave mode if the MSTR bit in FLEX_SPI_MR is written to 0:
  - The MISO line is driven by the transmitter output
  - The MOSI line is wired on the receiver input
  - The SPCK pin is driven by the transmitter to synchronize the receiver.
  - The NPCS0 pin becomes an input, and is used as a slave select signal (NSS)
  - Pin NPCS1 is not are not are not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operation. The bit rate generator is activated only in Master mode.

44.8.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the SPI Chip Select Register (FLEX_SPI_CSR). The clock phase is programmed with the NCPHA bit. These two parameters determine the edges of the clock signal on which data are driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Consequently, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are connected and require different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

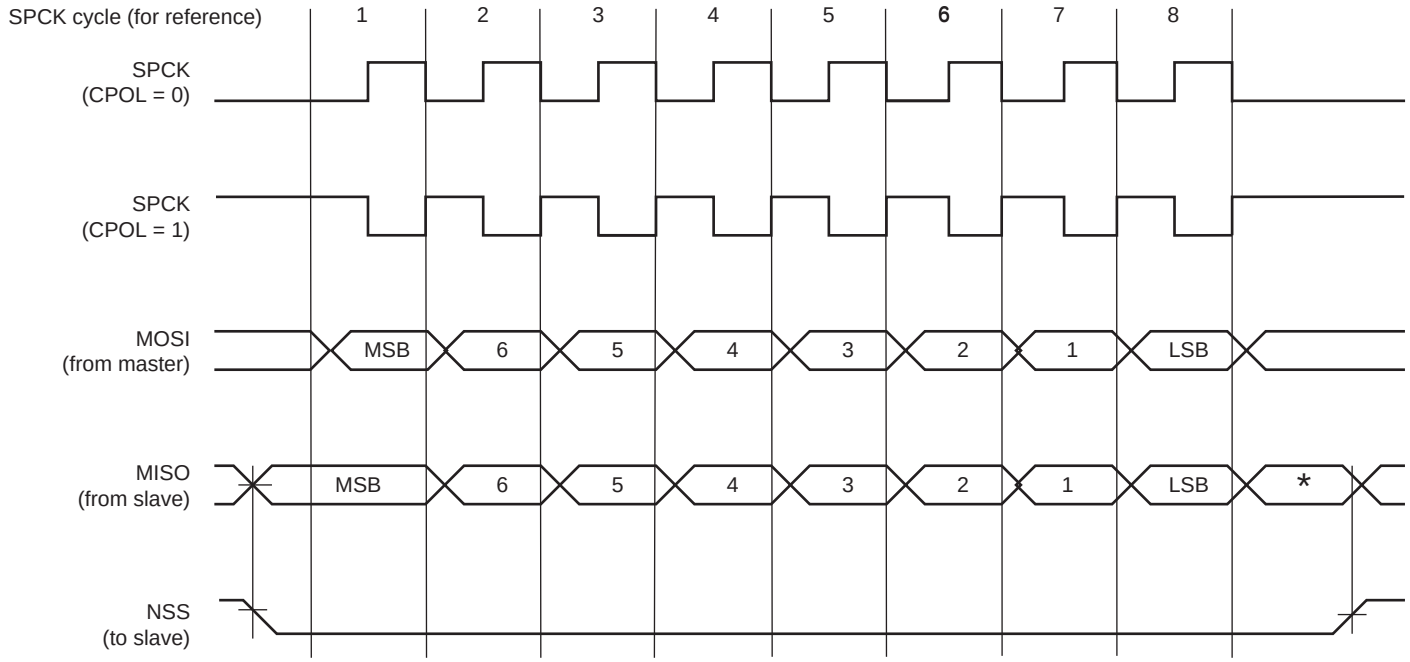
Table 44-17 shows the four modes and corresponding parameter settings.

Table 44-17. SPI Bus Protocol Mode

SPI Mode	CPOL	NCPHA	Shift SPCK Edge	Capture SPCK Edge	SPCK Inactive Level
0	0	1	Falling	Rising	Low
1	0	0	Rising	Falling	Low
2	1	1	Rising	Falling	High
3	1	0	Falling	Rising	High

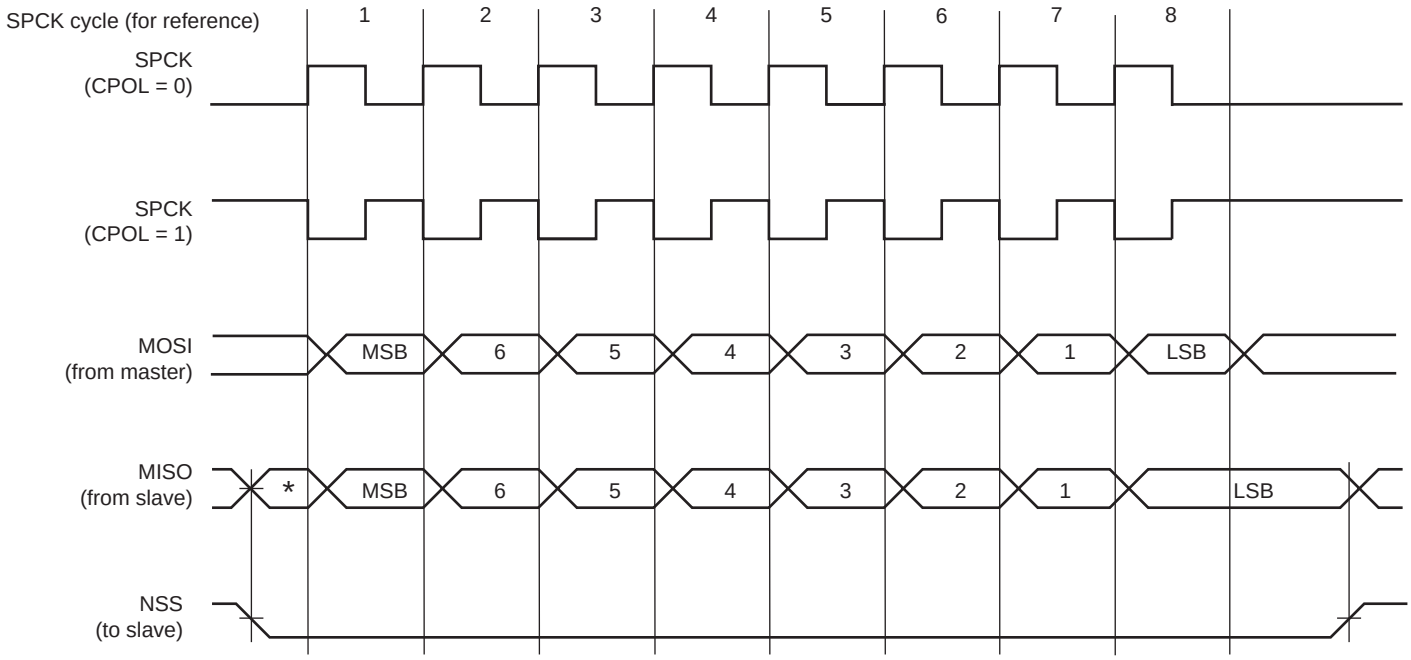
Figure 44-65 and Figure 44-66 show examples of data transfers.

Figure 44-65. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)



* Not defined.

Figure 44-66. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)



* Not defined.

44.8.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable bit rate generator. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (FLEX_SPI_TDR) and the Receive Data Register (FLEX_SPI_RDR), and a single shift register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer starts when the processor writes to FLEX_SPI_TDR. The written data are immediately transferred in the Shift register and the transfer on the SPI bus starts. While the data in the Shift register is shifted on the MOSI line, the MISO line is sampled and shifted in the Shift register. Data cannot be loaded in FLEX_SPI_RDR without transmitting data. If there is no data to transmit, a dummy data can be used (FLEX_SPI_TDR filled with ones). When the WDRBT bit is set, a new data cannot be transmitted if FLEX_SPI_RDR has not been read. If Receiving mode is not required, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the SPI Status Register (FLEX_SPI_SR) can be discarded.

Before writing the TDR, the PCS field in FLEX_SPI_MR must be set in order to select a slave.

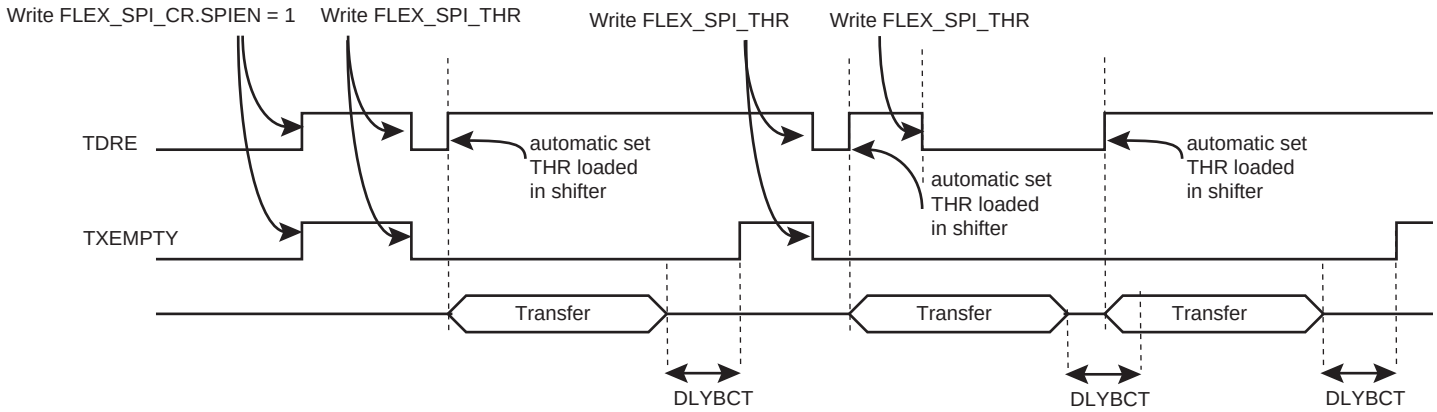
If new data are written in FLEX_SPI_TDR during the transfer, it is kept in FLEX_SPI_TDR until the current transfer is completed. Then, the received data are transferred from the Shift register to FLEX_SPI_RDR, the data in FLEX_SPI_TDR is loaded in the Shift register and a new transfer starts.

As soon as the FLEX_SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in the FLEX_SPI_SR is cleared. When the data written in the FLEX_SPI_TDR is loaded into the Shift register, the TDRE flag in the FLEX_SPI_SR is set. The TDRE bit is used to trigger the Transmit DMA channel (see Figure 44-67).

The end of transfer is indicated by the TXEMPTY flag in FLEX_US_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

Notes:
1. When the SPI is enabled, the TDRE and TXEMPTY flags are set.
2. The TXEMPTY flag alone cannot be used to detect the end of the buffer DMA transfer.

Figure 44-67. TDRE and TXEMPTY Flag Behavior



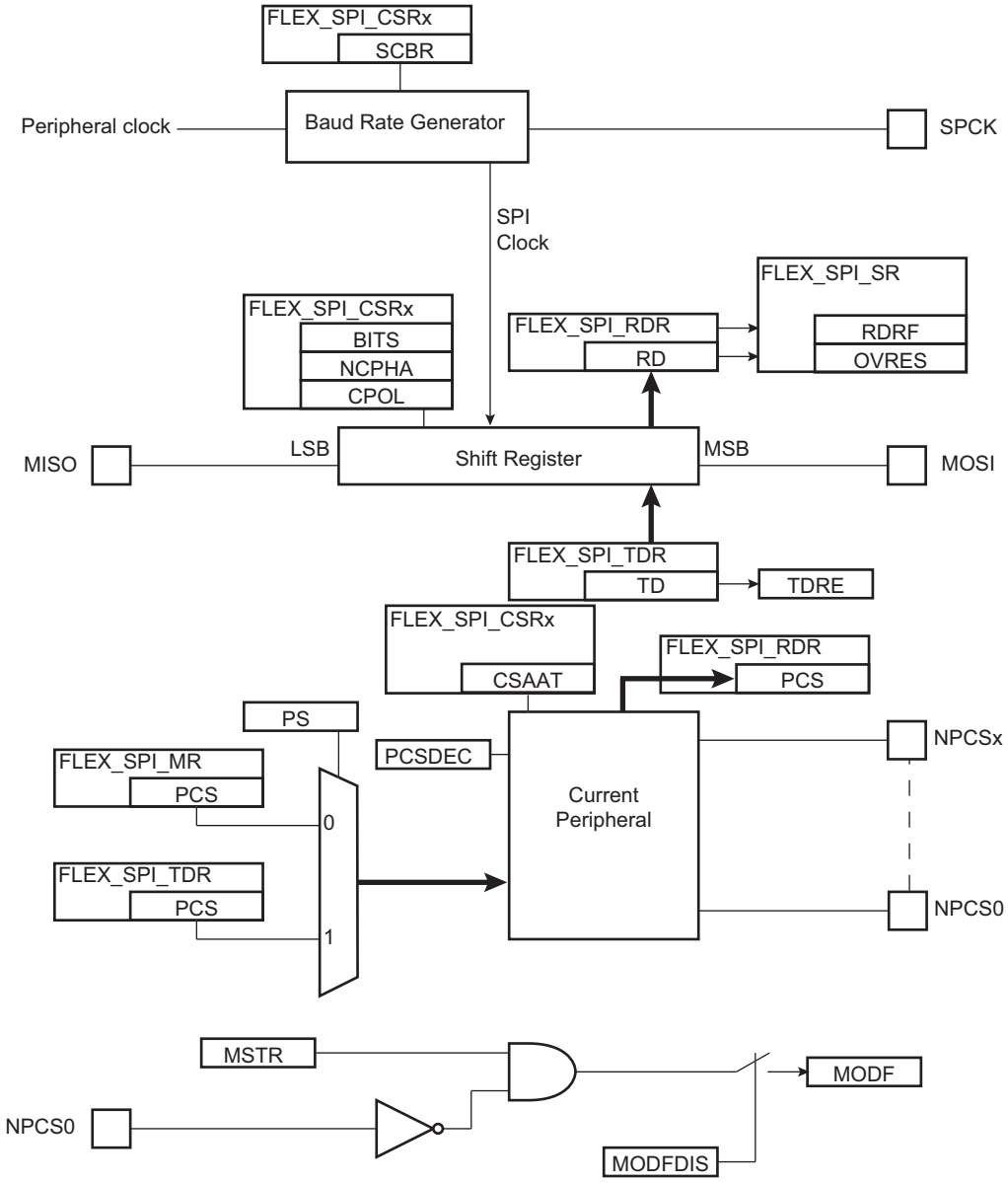
The transfer of received data from the Shift register to FLEX_SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in FLEX_SPI_SR. When the received data are read, the RDRF bit is cleared.

If FLEX_SPI_RDR has not been read before new data are received, the Overrun Error bit (OVRES) in FLEX_SPI_SR is set. As long as this flag is set, data are loaded in FLEX_SPI_RDR. The user has to read the status register to clear the OVRES bit.

Figure 44-68 shows a block diagram of the SPI when operating in Master mode. Figure 44-69 shows a flow chart describing how transfers are handled.

44.8.3.1 Master Mode Block Diagram

Figure 44-68. Master Mode Block Diagram



44.8.3.2 Master Mode Flow Diagram

Figure 44-69. Master Mode Flow Diagram

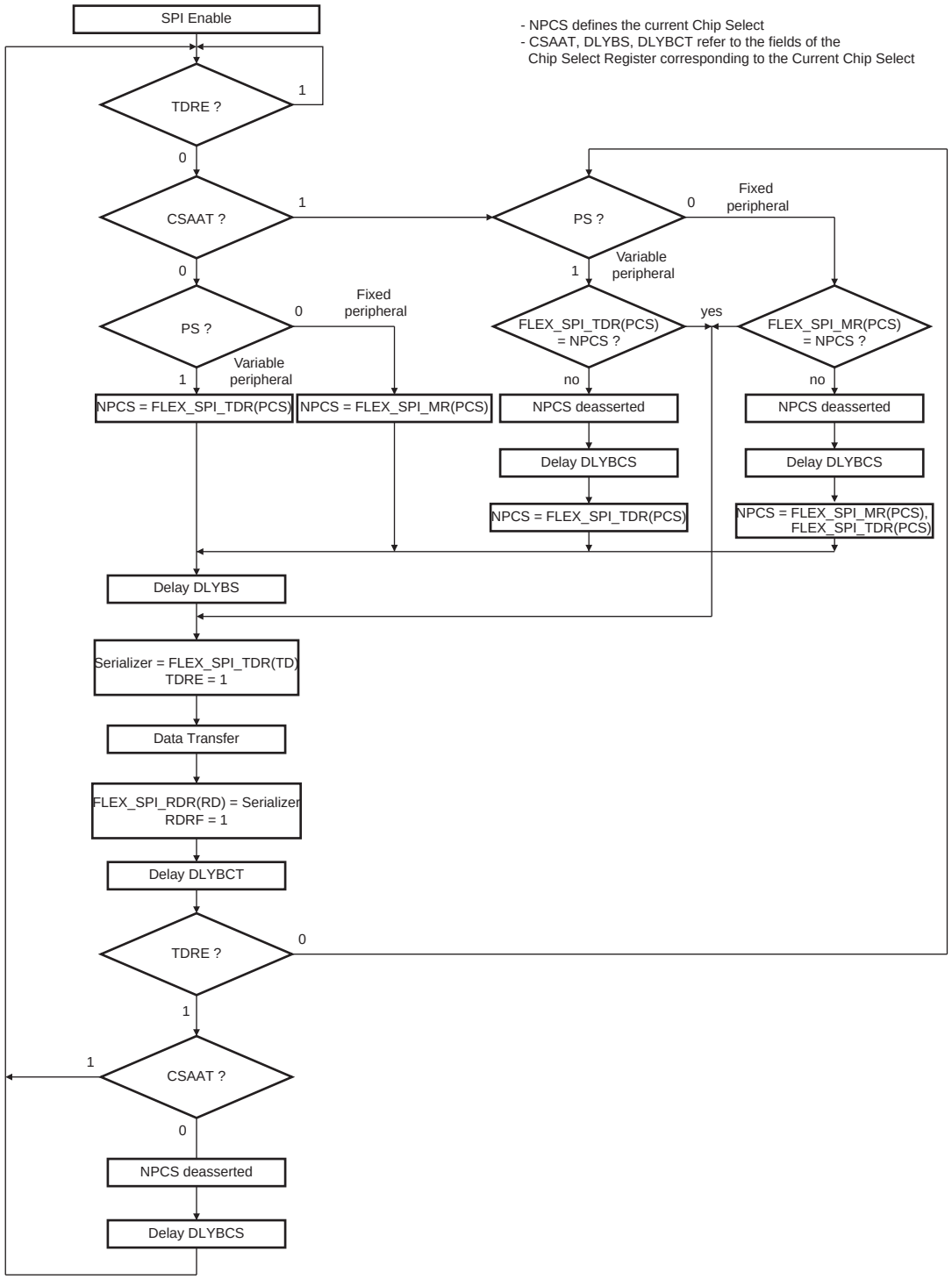
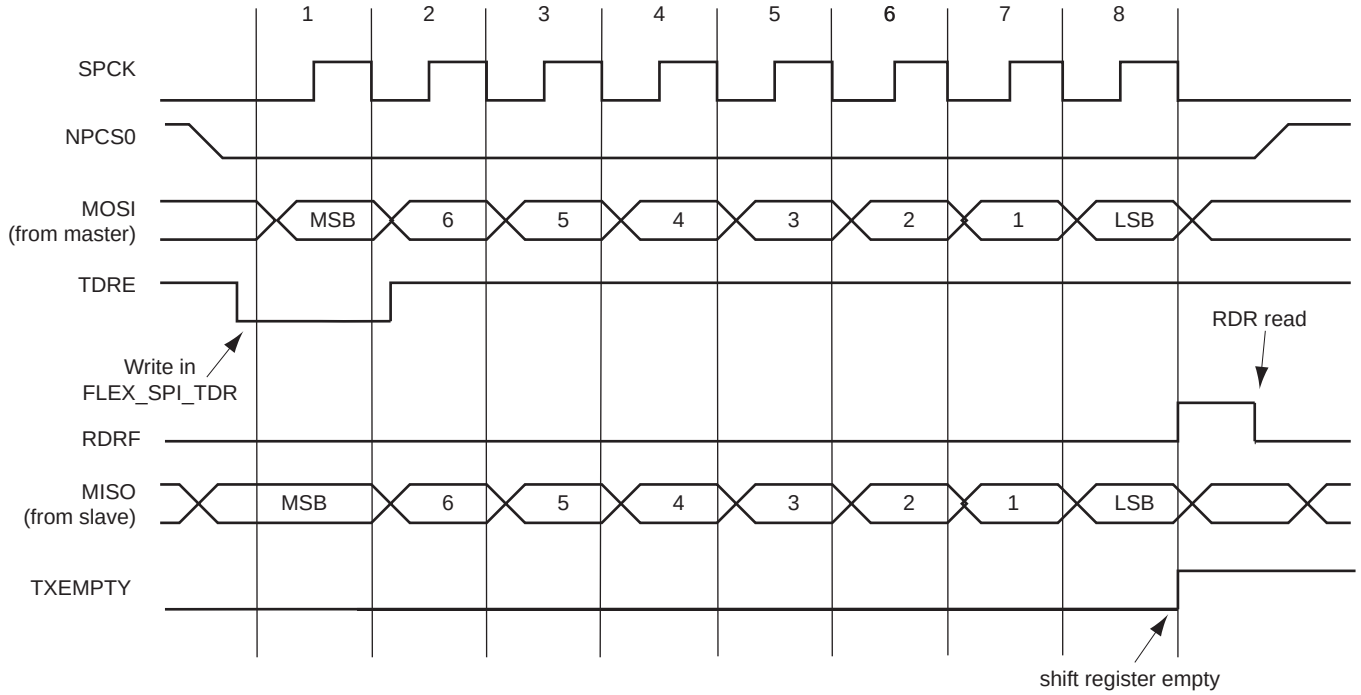


Figure 44-70 shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within FLEX_SPI_SR during an 8-bit data transfer in Fixed mode without the DMAC involved.

Figure 44-70. Status Register Flags Behavior



44.8.3.3 Clock Generation

The SPI bit rate clock is generated by dividing a source clock which can be the peripheral clock or a programmable clock from the GCLK. The divider can be a value between 1 and 255.

If the SCBR field is programmed to 1 and the clock source is GCLK, the operating bit rate is peripheral clock (see the electrical characteristics section for the SPCK maximum frequency). Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

At reset, SCBR is 0 and the user has to program it to a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be programmed in the SCBR field in FLEX_SPI_CSR. This allows the SPI to automatically adapt the bit rate for each interfaced peripheral without reprogramming.

If GCLK is selected as source clock (BRSRCCLK = 1 in FLEX_SPI_MR), the bit rate is independent of the processor/bus clock. Thus the processor clock can be changed while SPI is enabled. The processor clock frequency changes must be performed only by programming the PRES field in PMC_MCKR (see PMC section). Other methods to modify the processor/bus clock frequency (PLL multiplier, etc.) are forbidden when SPI is enabled.

The peripheral clock frequency must be at least three times higher than GCLK.

44.8.3.4 Transfer Delays

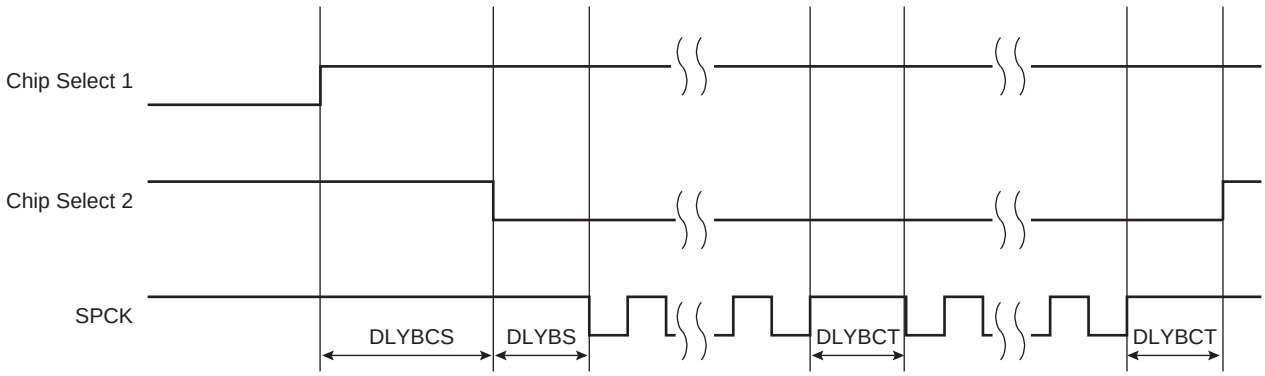
Figure 44-71 shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- The delay between the chip selects. It is programmable only once for all chip selects by writing the DLYBCS field in FLEX_SPI_MR. The SPI slave device deactivation delay is managed through DLYBCS. If there is only one SPI slave device connected to the master, the DLYBCS field does not need to be configured. If several slave devices are connected to a master, DLYBCS must be configured depending on the highest deactivation delay. Refer to the SPI slave device electrical characteristics.

- The delay before SPCK, independently programmable for each chip select by writing the DLYBS field. The SPI slave device activation delay is managed through DLYBS. Refer to the SPI slave device electrical characteristics to define DLYBS.
- The delay between consecutive transfers, independently programmable for each chip select by writing the DLYBCT field. The time required by the SPI slave device to process received data is managed through DLYBCT. This time depends on the SPI slave system activity.

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

Figure 44-71. Programmable Delays



44.8.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS1 signals. By default, all NPCS signals are high before and after each transfer.

- **Fixed Peripheral Select Mode**: SPI exchanges data with only one peripheral.
  Fixed Peripheral Select mode is enabled by writing the PS bit to zero in FLEX_SPI_MR. In this case, the current peripheral is defined by the PCS field in FLEX_SPI_MR and the PCS field in FLEX_SPI_TDR has no effect.
- **Variable Peripheral Select Mode:** Data can be exchanged with more than one peripheral without having to reprogram the NPCS field in FLEX_SPI_MR.
  Variable Peripheral Select Mode is enabled by setting the PS bit to one in FLEX_SPI_MR. The PCS field in FLEX_SPI_TDR is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value to write in FLEX_SPI_TDR has the following format:

  [xxxxxxx(7-bit) + LASTXFER(1-bit)[(1)]+ xxxx(4-bit) + PCS (4-bit) + DATA (8 to 16-bit)] with PCS equals the chip select to assert, as defined in Section 44.10.48 “SPI Transmit Data Register” and LASTXFER bit at 0 or 1 depending on the CSAAT bit.

Note: 1. Optional

The CSAAT, LASTXFER and CSNAAT bits are discussed in Section 44.8.3.9 “Peripheral Deselection with DMA”.

If LASTXFER is used, the command must be issued after writing the last character. Instead of LASTXFER, the user can use the SPIDIS command. After the end of the DMA transfer, it is necessary to wait for the TXEMPTY flag and then write SPIDIS into the SPI Control Register (FLEX_SPI_CR). This does not change the configuration register values). The NPCS is disabled after the last character transfer. Then, another DMA transfer can be started if the SPIEN has previously been written in FLEX_SPI_CR.

44.8.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, FLEX_SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming FLEX_SPI_MR. Data written in FLEX_SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

44.8.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 3 slave peripherals by decoding the two chip select lines, NPCS0 to NPCS1 with an external decoder/demultiplexer (refer to Figure 44-72). This can be enabled by setting the PCSDEC bit in FLEX_SPI_MR.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

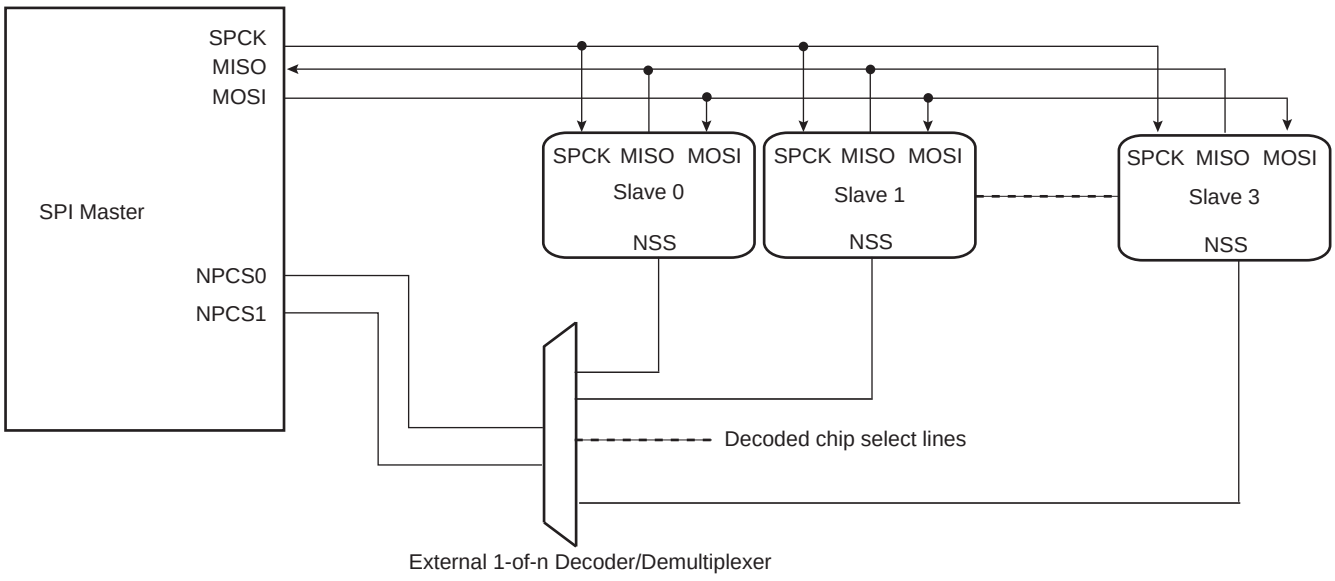
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either FLEX_SPI_MR or FLEX_SPI_TDR (depending on PS).

As the SPI sets a default value of 0x3 on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 3 peripherals can be decoded.

The SPI has only two Chip Select registers. As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to two peripherals. As an example, FLEX_SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 1, corresponding to the PCS values 0x0 to 0x1. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 1 and 2. Figure 44-72 shows this type of implementation.

If the CSAAT bit is used, with or without the DMAC, the mode fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since mode fault detection is only on NPCS0.

Figure 44-72. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation



44.8.3.8 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, FLEX_SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of FLEX_SPI_TDR is transferred into the internal Shift register. When this flag is detected high, FLEX_SPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not de-asserted between the two transfers. But depending on the application software handling the SPI status register flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload FLEX_SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in FLEX_SPI_CSR, gives even less time for the processor to reload FLEX_SPI_TDR. With some SPI slave peripherals, if the chip select line must remain active (low) during a full set of transfers, communication errors can occur.

To facilitate interfacing with such devices, the Chip Select registers [CSR0...CSR1] can be programmed with the Chip Select Active After Transfer (CSAAT) bit to 1. This allows the chip select lines to remain in their current state (low = active) until a transfer to another chip select is required. Even if FLEX_SPI_TDR is not reloaded, the chip select remains active. To de-assert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in FLEX_SPI_CR must be set after writing the last data to transmit into FLEX_SPI_TDR.

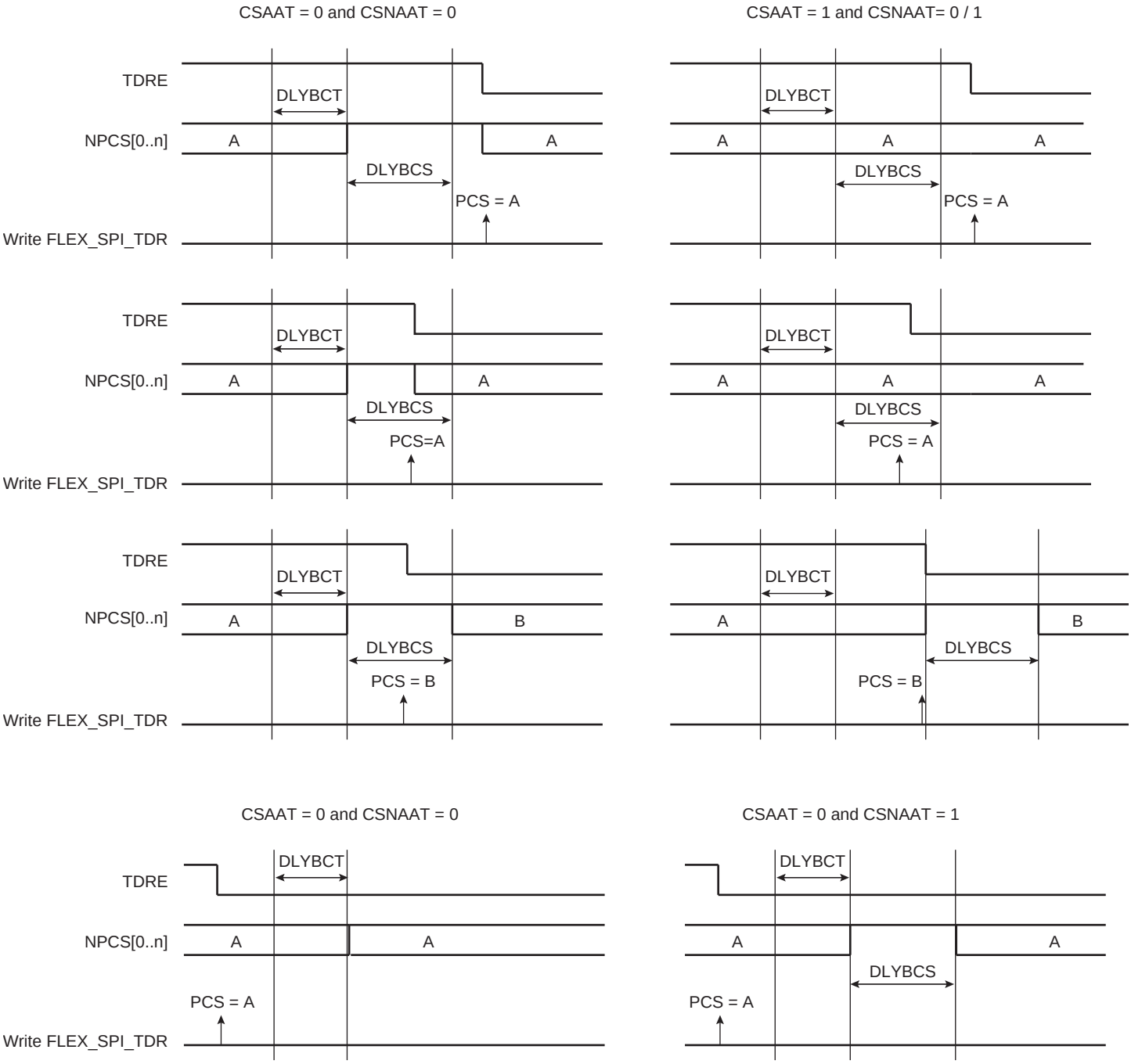
44.8.3.9 Peripheral Deselection with DMA

DMA provides faster reloads of FLEX_SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that FLEX_SPI_TDR is written with the next data before the end of the current transfer. Consequently, a data can be lost by the de-assertion of the NPCS line for SPI slave peripherals requiring the chip select line to remain active between two transfers. The only way to guarantee a safe transfer in this case is the use of the CSAAT and LASTXFER bits.

When the CSAAT bit is cleared, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the TDRE flag rises as soon as the content of FLEX_SPI_TDR is transferred into the internal shift register. When this flag is detected, FLEX_SPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not de-asserted between the two transfers. This can lead to difficulties to interface with some serial peripherals requiring the chip select to be de-asserted after each transfer. To facilitate interfacing with such devices, FLEX_SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit to 1. This allows the chip select lines to be de-asserted systematically during a time “DLYBCS” (the value of the CSNAAT bit is processed only if the CSAAT bit is cleared for the same chip select).

Figure 44-73 shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

Figure 44-73. Peripheral Deselection



44.8.3.10 Mode Fault Detection

The SPI has the capability to operate in multi-master environment. Consequently, the NPCS0/NSS line must be monitored. If one of the masters on the SPI bus is currently transmitting, the NPCS0/NSS line is low and the SPI must not transmit a data. A mode fault is detected when the SPI is programmed in Master mode and a low level is driven by an external master on the NPCS0/NSS signal. In multi-master environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, the MODF bit in FLEX_SPI_SR is set until FLEX_SPI_SR is read and the SPI is automatically disabled until it is re-enabled by writing the SPIEN bit in FLEX_SPI_CR to 1.

By default, the mode fault detection is enabled. The user can disable it by setting the MODFDIS bit in FLEX_SPI_MR.

44.8.4 SPI Slave Mode

When operating in Slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits until NSS goes active before receiving the serial clock from an external master. When NSS falls, the clock is validated and the data are loaded in FLEX_SPI_RDR according to the configuration value of the BITS field in FLEX_SPI_CSR0. These bits are processed following a phase and a polarity defined respectively by the NCPHA and CPOL bits in FLEX_SPI_CSR0. Note that the BITS field, CPOL bit and NCPHA bit of the other Chip Select registers have no effect when the SPI is programmed in Slave mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

Note: For more information on the BITS field, see also the note below the FLEX_SPI_CSRx register bitmap in Section 44.10.54 “SPI Chip Select Register”.

When all bits are processed, the received data are transferred in FLEX_SPI_RDR and the RDRF bit rises. If FLEX_SPI_RDR has not been read before new data are received, the Overrun Error bit (OVRES) in FLEX_SPI_SR is set. As long as this flag is set, data are loaded in FLEX_SPI_RDR. The user must read FLEX_SPI_SR to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the Shift register. If no data has been written in FLEX_SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the Shift register resets to 0.

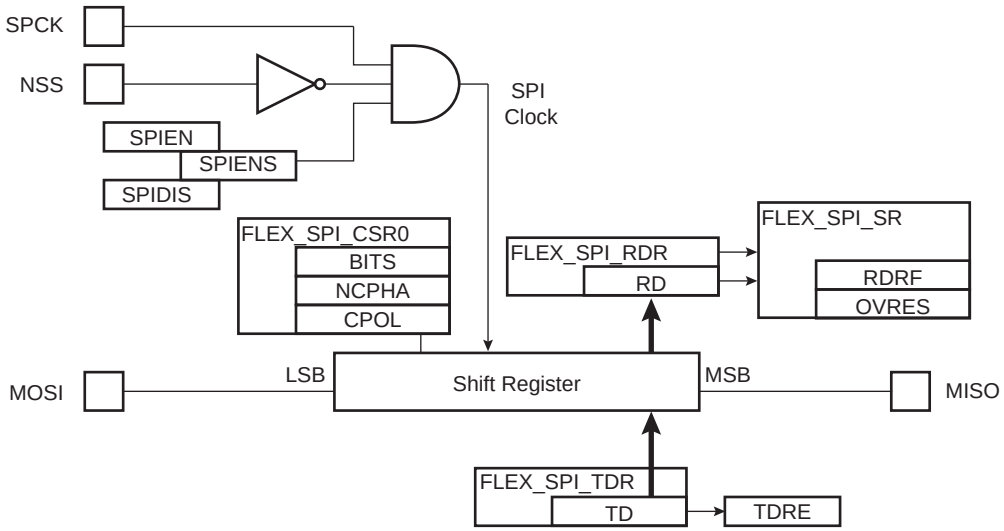
When a first data is written in FLEX_SPI_TDR, it is transferred immediately in the Shift register and the TDRE flag rises. If new data is written, it remains in FLEX_SPI_TDR until a transfer occurs, i.e., NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in FLEX_SPI_TDR is transferred in the Shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

Then, a new data is loaded in the Shift register from FLEX_SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in FLEX_SPI_TDR since the last load from FLEX_SPI_TDR to the Shift register, FLEX_SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in FLEX_SPI_SR.

If NSS rises between two characters, it must be kept high for two MCK clock periods or more and the next SPCK capture edge must not occur less than four MCK periods after NSS rise.

Figure 44-74 shows a block diagram of the SPI when operating in Slave mode.

Figure 44-74. Slave Mode Functional Block Diagram



44.8.5 SPI Comparison Function on Received Character

The comparison is only relevant for SPI Slave mode (MSTR = 0 in FLEX_US_MR).

The effect of a comparison match changes if the system is in Wait or Active mode.

In Wait mode, if asynchronous partial wakeup is enabled, a system wakeup is performed (see Section 44.8.6 “SPI Asynchronous and Partial Wakeup (SleepWalking)”).

In Active mode, the CMP flag in FLEX_SPI_SR is raised. It is set when the received character matches the conditions programmed in the SPI Comparison Register (FLEX_SPI_CMPR). The CMP flag is set as soon as FLEX_SPI_RDR is loaded with the new received character. The CMP flag is cleared by reading FLEX_SPI_SR.

FLEX_SPI_CMPR (see Section 44.10.57) can be programmed to provide different comparison methods. These are listed below:

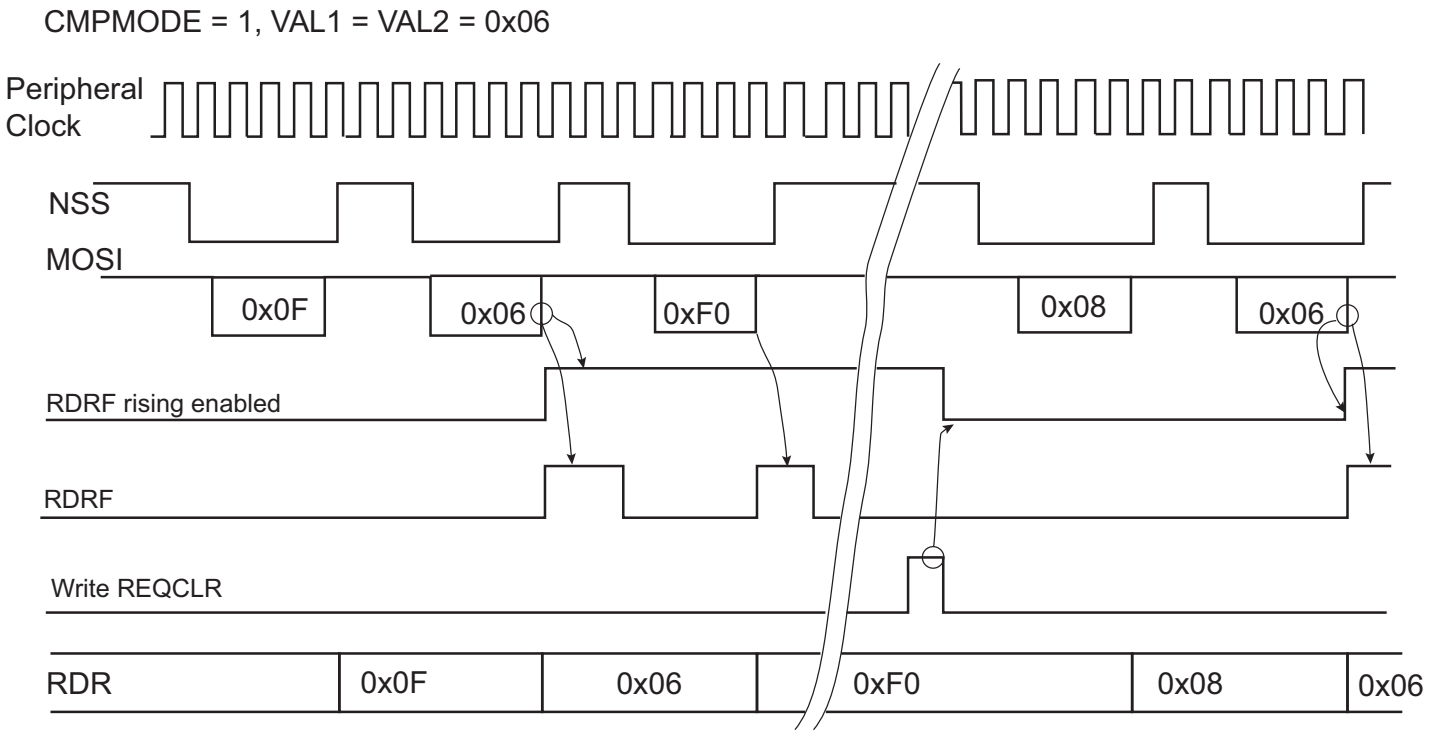
- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When the CMPMODE bit is cleared in FLEX_SPI_CMPR, all received data is loaded in FLEX_SPI_RDR and the CMP flag provides the status of the comparison result.

By setting the CMPMODE bit, the comparison result triggers the start of FLEX_SPI_RDR loading (see Figure 44-75). The trigger condition exists as soon as the received character value matches the conditions defined by VAL1 and VAL2 in FLEX_SPI_CMPR. The comparison trigger event is restarted by writing a 1 to the REQCLR bit in FLEX_SPI_CR.

The value programmed in VAL1 and VAL2 fields must not exceed the maximum value of the received character (see BITS field in FLEX_SPI_CSR0).

Figure 44-75. Receive Data Register Management



44.8.6 SPI Asynchronous and Partial Wakeup (SleepWalking)

This operating mode is a means of data pre-processing that qualifies an incoming event, thus allowing the SPI to decide whether or not to wake up the system. Asynchronous and partial wakeup is mainly used when the system is in Wait mode (see the PMC section for further details). It can also be enabled when the system is fully running.

Asynchronous and partial wakeup can be used only when SPI is configured in Slave mode (MSTR is cleared in FLEX_SPI_MR).

The maximum SPI clock (SPCK) frequency that can be provided by the SPI master is bounded by the peripheral clock frequency. The SPCK frequency must be lower than or equal to the peripheral clock. The NSS line must be de-asserted by the SPI master between two characters. The NSS de-assertion duration time must be greater than or equal to six peripheral clock periods. The time between the assertion of NSS line (falling edge) and the first edge of the SPI clock must be higher than 15 µs.

The FLEX_SPI_RDR register must be read before enabling the asynchronous and partial wakeup.

When asynchronous and partial wakeup is enabled for the SPI (see the PMC section), the PMC decodes a clock request from the SPI. The request is generated as soon as there is a falling edge on the NSS line as this may indicate the beginning of a frame. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the SPI.

The SPI processes the received frame and compares the received character with VAL1 and VAL2 in FLEX_SPI_CMPR (Section 44.10.57).

The SPI instructs the PMC to disable the peripheral clock if the received character value does not meet the conditions defined by VAL1 and VAL2 fields in FLEX_SPI_CMPR (see Figure 44-77).

If the received character value meets the conditions, the SPI instructs the PMC to exit the system from Wait mode (see Figure 44-77).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 equals VAL2, then the comparison is performed on a single value and the wakeup is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, the wakeup is triggered if any received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 65535, the wakeup is triggered as soon as a character is received.

If the processor and peripherals are running, the SPI can be configured in Asynchronous and Partial Wakeup mode by enabling the PMC_SLPWK_ER (see PMC section). When activity is detected on the receive line, the SPI requests the clock from the PMC and the comparison is performed. If there is a comparison match, the SPI continues to request the clock. If there is no match, the clock is switched off for the SPI only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wakeup mode is enabled for the SPI (see PMC_SLPWK_ER in PMC section).

When the system is in Active mode and the SPI enters Asynchronous and Partial Wakeup mode, the flag RDRF must be programmed as the unique source of the SPI interrupt.

When the system exits Wait mode as the result of a matching condition, the RDRF flag is used to determine if the SPI is the source for the exit from Wait mode.

Figure 44-76. Asynchronous Wakeup Use Case Example

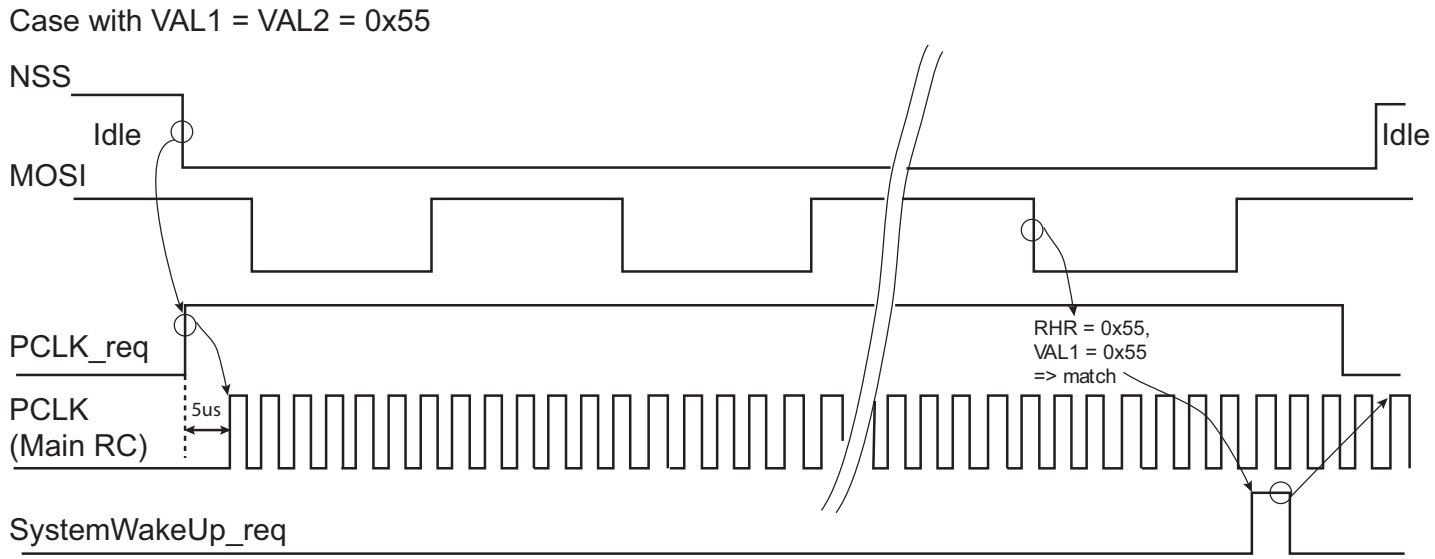
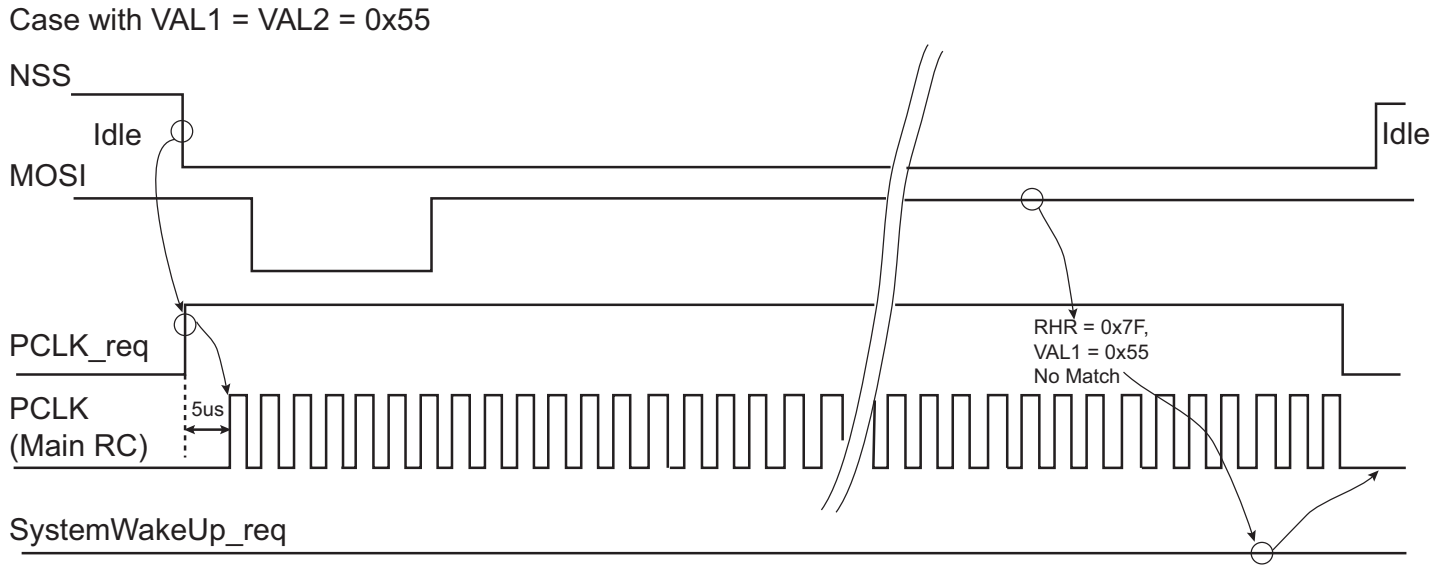


Figure 44-77. Asynchronous Event Generating Only Partial Wakeup

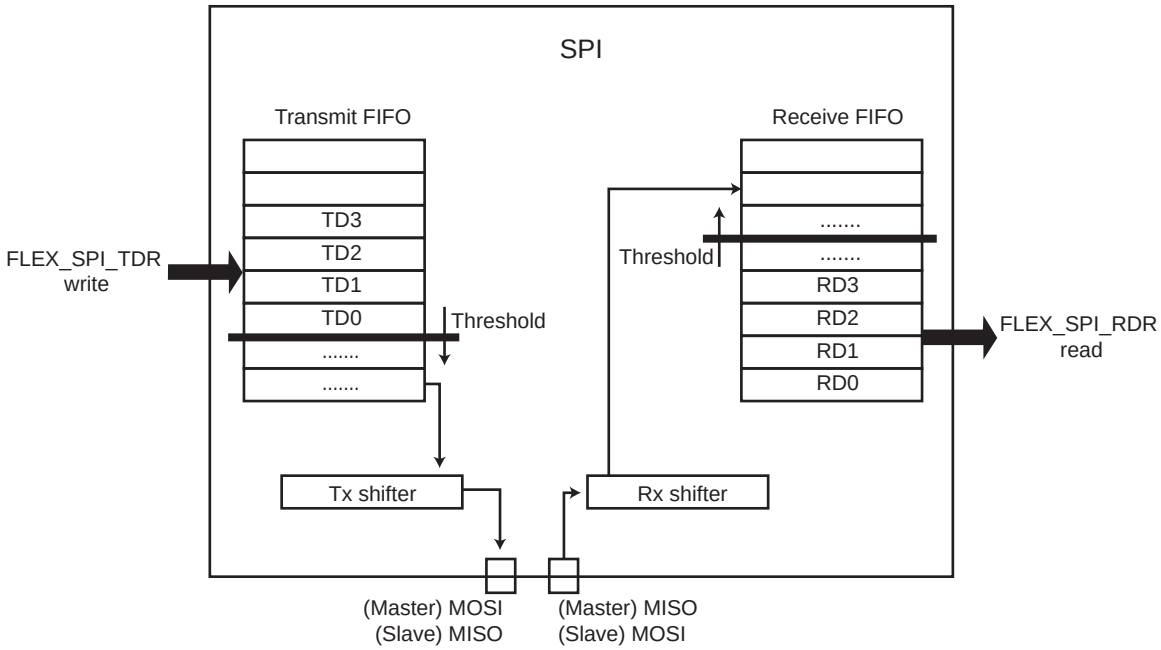


44.8.7 FIFOs

The SPI includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in FLEX_SPI_CR. It is recommended to disable the SPI module before enabling or disabling the SPI FIFOs (TXDIS and RXDIS bit in FLEX_SPI_CR).

Writing the FIFOEN bit to ‘1’ enables a 32-data Transmit FIFO and a 32-data Receive FIFO.

Figure 44-78. FIFOs Block Diagram

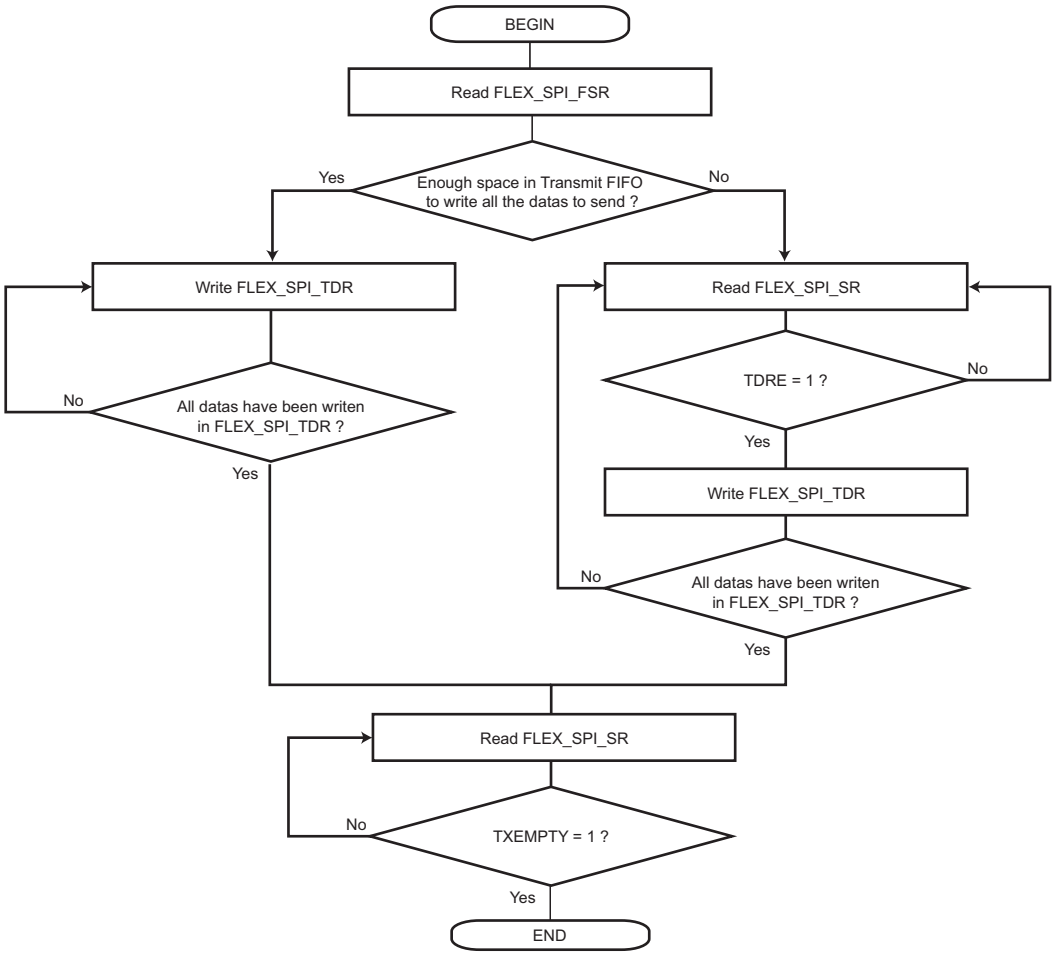


44.8.7.1 Sending Data with FIFO Enabled

With the Transmit FIFO enabled, any write access to FLEX_SPI_TDR brings the written data to the Transmit FIFO. As a consequence, it is not mandatory any more to monitor the TDRE flag state to send multiple data without DMAC.

Knowing the number of data to send, and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in FLEX_SPI_TDR without checking the TDRE flag between each access. The Transmit FIFO state can be checked reading the TXFL field in the SPI FIFO Level Register (FLEX_SPI_FLR).

Figure 44-79. Sending Data with FIFO Flowchart

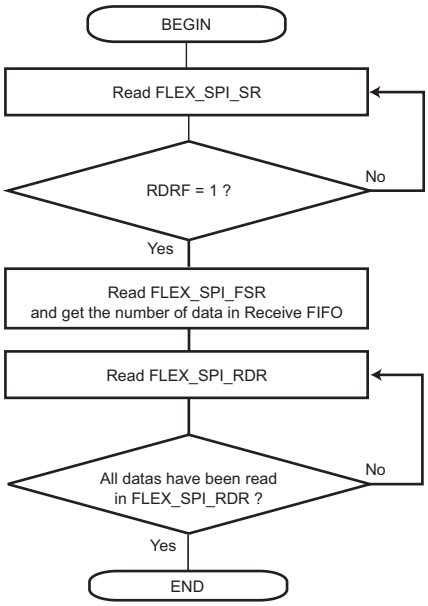


44.8.7.2 Receiving Data with FIFO Enabled

With Receive FIFO enabled, any read access on FLEX_SPI_RDR pulls out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RDRF flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RDRF flag set to ‘1’), the exact number of data can be checked with the RXFL field in FLEX_SPI_FLR and all the data read successively in FLEX_SPI_RDR without checking the RDRF flag between each access.

Figure 44-80. Receiving Data with FIFO Flowchart



44.8.7.3 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in FLEX_SPI_CR.

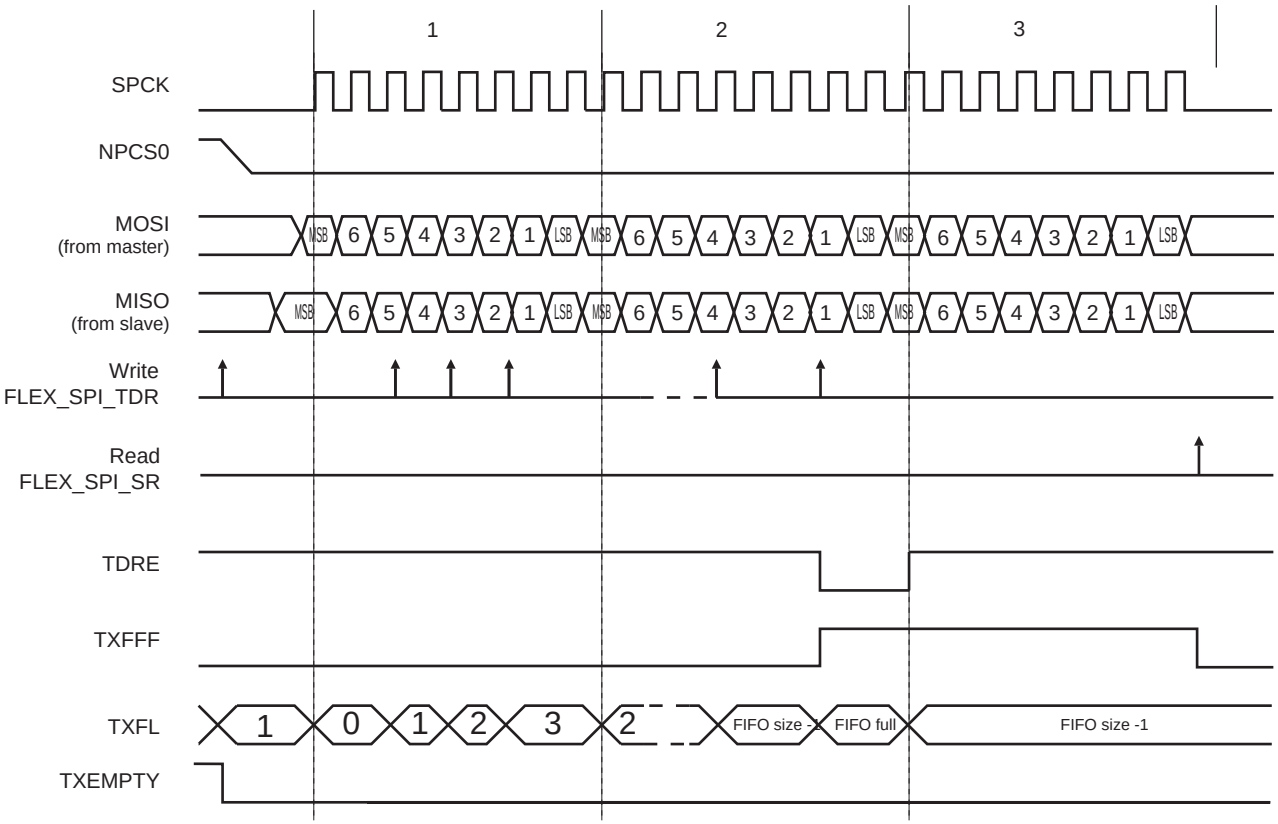
44.8.7.4 TDRE, RDRF and TXEMPTY behavior

If FIFOs are enabled, the behavior of the TDRE, RDRF and TXEMPTY flags is slightly different.

With FIFO enabled, the TXEMPTY flag remains at ‘0’ state as long as there are characters in the Transmit FIFO or in the Transmit Shift Register. It is at ‘1’ state when there are no character in the Transmit FIFO and no character in the Transmit Shift Register.

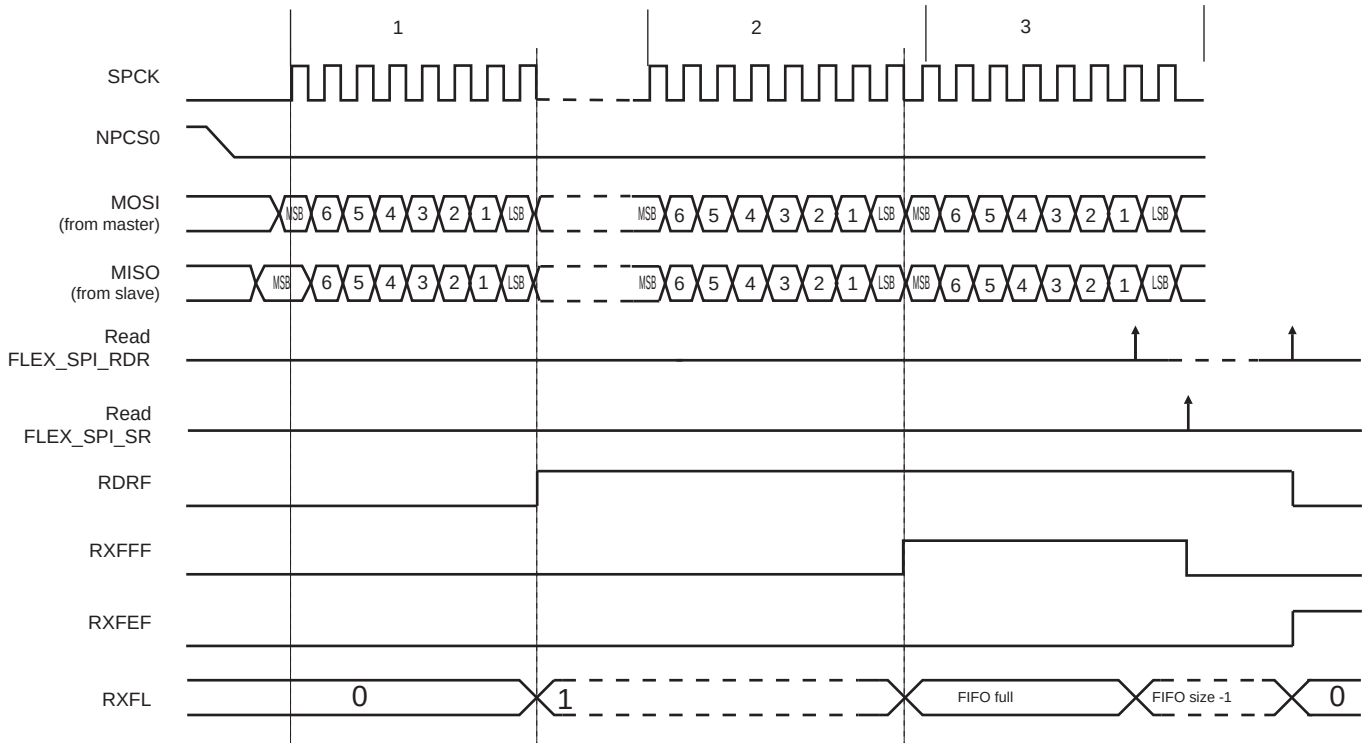
TDRE indicates if a data can be written in the Transmit FIFO. By default, the TDRE flag then stays at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 44-81. TDRE in Single Data Mode and TXRDYM = 0x0



RDRF indicates if an unread data is present in the Receive FIFO. By default, the RDRF flag is then at level ‘1’ as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 44-82. RDRF in Single Data Mode and RXRDYM = 0x0



TDRE and RDRF behavior can be modified using the TXRDYM and RXRDYM fields in the SPI FIFO Mode Register (FLEX_SPI_FMR). In Single Data mode, there is no need to modify the TDRE and RDRF behavior; however, it may be useful in Multiple Data mode.

44.8.7.5 Single Data Mode

If Variable Peripheral Select mode is used (PS bit set to ‘1’ in FLEX_SPI_MR), the Transmit FIFO operates in Single Data mode.

If Master mode is set (MSTR bit set to ‘1’ in FLEX_SPI_MR), the Receive FIFO operates in Single Data mode.

In this mode, only one data can be written/read per write/read access of FLEX_SPI_TDR/FLEX_SPI_RDR (see Section 44.10.45 “SPI Receive Data Register”). On the other hand TXRDYM and RXRDYM fields should be configured with 0x0 value in this mode.

DMAC

TXRDYM and RXRDYM fields must be configured with 0x0 value when working with DMAC transfer in Single Data mode.

The same DMAC procedure applies as with FIFO disabled.

44.8.7.6 Multiple Data Mode

When the FIFOs do not operate in Single Data mode, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write up to two data in one FLEX_SPI_TDR write access. It is possible, in this mode, to read up to four data in one FLEX_SPI_RDR access if the BITS field is configured to 0 (8-bit data size) and up to two data if the BITS field value is other than 0 (more than 8-bit data size).

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read/only one data is written and finally, if the access is a word-size register access, then up to four data are read/up to two data are written.

Written/Read data are always right-aligned, as shown in Section 44.10.46 “SPI Receive Data Register (FIFO Multiple Data, 8-bit)”, Section 44.10.47 “SPI Receive Data Register (FIFO Multiple Data, 16-bit)” and Section 44.10.49 “SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)”.

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six FLEX_SPI_TDR-byte write accesses.
- Perform three FLEX_SPI_TDR-halfword write accesses.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six FLEX_SPI_RDR-byte read accesses.
- Perform three FLEX_SPI_RDR-halfword read accesses.
- Perform one FLEX_SPI_RDR-word read access and one FLEX_SPI_RDR-halfword read access.

This mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

TDRE and RDRF Configuration

In Multiple Data mode the TXRDYM and RXRDYM fields in FLEX_SPI_FMR become useful.

As in Multiple Data mode, it is possible to write several data in the same access it might be useful to configure TDRE flag behavior to indicate if one or two data can be written in the FIFO depending on the access to perform on FLEX_SPI_TDR.

If, for instance, two data are written each time in FLEX_SPI_TDR, it might be useful to configure the TXRDYM field to 0x1 value so that the TDRE flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_SPI_RDR, it might be useful to configure the RXRDYM field to 0x2 value so that the RDRF flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

DMAC

If DMAC transfer is used, it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

44.8.7.7 FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

If the Transmit FIFO is full and a write access is performed on FLEX_SPI_TDR, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in FLEX_SPI_SR.

In Multiple Data mode, if the number of data written in FLEX_SPI_TDR (according to the register access size) is bigger than the Transmit FIFO free space, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in FLEX_SPI_SR.

- Receive FIFO:

In Multiple Data mode, if the number of data read in FLEX_SPI_RDR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it generates a Receive FIFO pointer error and sets the RXFPTEF flag in FLEX_SPI_SR.

No pointer error should happen if the FIFO state is checked before writing/reading in FLEX_SPI_TDR/FLEX_SPI_RDR. The FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a pointer error occurs, a software reset must be performed through the SWRST bit in FLEX_SPI_CR (configuration will be lost).

44.8.7.8 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO:

The Transmit FIFO threshold can be set using the TXFTHRES field in FLEX_SPI_FMR. Each time the Transmit FIFO goes from the 'above threshold' to the 'equal or below threshold' state, the TXFTHF flag in FLEX_SPI_SR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.

- Receive FIFO:

The Receive FIFO threshold can be set using the RXFTHRES field in FLEX_SPI_FMR. Each time the Receive FIFO goes from the 'below threshold' to the 'equal to or above threshold' state, the RXFTHF flag in FLEX_SPI_SR is set. This enables the application to know that the Receive FIFO reached the defined threshold and to read some data before it becomes full.

The TXFTHF and RXFTHF flags can be both configured to generate an interrupt using FLEX_SPI_IER and FLEX_SPI_IDR.

44.8.7.9 FIFO Flags

FIFOs come with a set of flags which can be configured each to generate interrupt through FLEX_SPI_IER and FLEX_SPI_IDR.

FIFO flags state can be read in FLEX_SPI_SR. They are cleared on FLEX_SPI_SR read.

44.8.8 SPI Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_SPI to enable access to the write protection registers.

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable) bit in the SPI Write Protection Mode Register (FLEX_SPI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the SPI Write Protection Status Register (FLEX_SPI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_SPI_WPSR.

The following register(s) can be write-protected when WPEN is set:

- SPI Mode Register
- SPI Chip Select Register
- SPI Comparison Register

44.9 TWI Functional Description

44.9.1 Transfer Format

The data put on the TWD line must be 8 bits long. Data are transferred MSB first; each byte must be followed by an acknowledgement. The number of bytes per transfer is unlimited (see Figure 44-84).

Each transfer begins with a START condition and terminates with a STOP condition (see Figure 44-83).

- A high-to-low transition on the TWD line while TWCK is high defines the START condition.
- A low-to-high transition on the TWD line while TWCK is high defines a STOP condition.

Figure 44-83. START and STOP Conditions

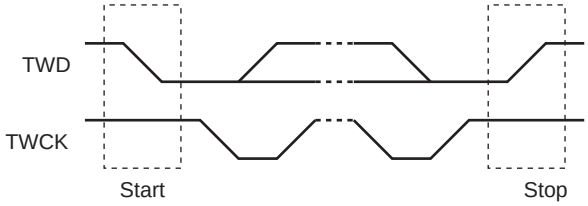
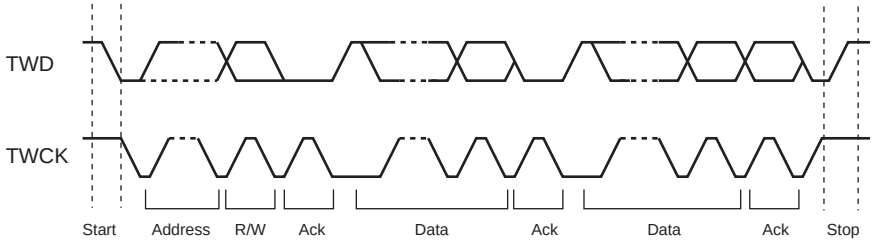


Figure 44-84. Transfer Format



44.9.2 Modes of Operation

The TWI has different modes of operation:

- Master Transmitter mode (Standard and Fast modes only)
- Master Receiver mode (Standard and Fast modes only)
- Multi-master Transmitter mode (Standard and Fast modes only)
- Multi-master Receiver mode (Standard and Fast modes only)
- Slave Transmitter mode (Standard, Fast and High-speed modes)
- Slave Receiver mode (Standard, Fast and High-speed modes)

These modes are described in the following sections.

44.9.3 Master Mode

44.9.3.1 Definition

The master is the device that starts a transfer, generates a clock and stops it. This operating mode is not available if High-speed mode is selected.

44.9.3.2 Programming Master Mode

The following fields must be programmed before entering Master mode:

1. DADR (+ IADRSZ + IADR if a 10-bit device is addressed): The device address is used to access slave devices in Read or Write mode.
2. CWGR + CKDIV + CHDIV + CLDIV: Clock waveform.
3. SVDIS: Disables Slave mode.
4. MSEN: Enables Master mode.

Note: If the TWI is already in Master mode, the device address (DADR) can be configured without disabling the Master mode.

44.9.3.3 Transfer Speed/Bit Rate

The TWI speed is defined in FLEX_TWI_CWGR. The TWI bit rate can be based either on the peripheral clock if the BRSRCCLK bit value is 0 or on a programmable clock source provided by the GCLK if the BRSRCCLK bit value is 1.

If BRSRCCLK = 1, the bit rate is independent of the processor/peripheral clock and thus processor/peripheral clock frequency can be changed without affecting the TWI transfer rate.

The GCLK frequency must be at least three times lower than the peripheral clock frequency.

44.9.3.4 Master Transmitter Mode

This operating mode is not available if High-speed mode is selected.

After the master initiates a START condition when writing into the Transmit Holding register FLEX_TWI_THR, it sends a 7-bit slave address, configured in the Master Mode Register (DADR in FLEX_TWI_MMR), to notify the slave device. The bit following the slave address indicates the transfer direction, 0 in this case (MREAD = 0 in FLEX_TWI_MMR).

The TWI transfers require the slave to acknowledge each received byte. During the acknowledge clock pulse (ninth pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. If the slave does not acknowledge the byte, then the Not Acknowledge flag (NACK) is set in the TWI Status Register (FLEX_TWI_SR) of the master and a STOP condition is sent. The NACK flag must be cleared by reading the TWI Status Register (FLEX_TWI_SR) before the next write into the TWI Transmit Holding Register (FLEX_TWI_THR). As with the other status bits, an interrupt can be generated if enabled in the interrupt enable Register (FLEX_TWI_IER). If the slave acknowledges the byte, the data written in FLEX_TWI_THR is then shifted in the internal shifter and transferred. When an acknowledge is detected, the TXRDY bit is set until a new write in FLEX_TWI_THR.

TXRDY is used as transmit ready for the DMA transmit channel.

Note: To clear the TXRDY flag in Master mode, write bit MSDIS to 1, then write bit MSEN to 1 in FLEX_TWI_CR.

While no new data is written in FLEX_TWI_THR, the serial clock line is tied low. When new data is written in FLEX_TWI_THR, the SCL is released and the data is sent. To generate a STOP event, the STOP command must be performed by writing in the STOP field of the TWI Control Register (FLEX_TWI_CR).

After a master write transfer, the Serial Clock line is stretched (tied low) while no new data is written in FLEX_TWI_THR or until a STOP command is performed.

See Figure 44-85, Figure 44-86, and Figure 44-87.

Figure 44-85. Master Write with One Data Byte

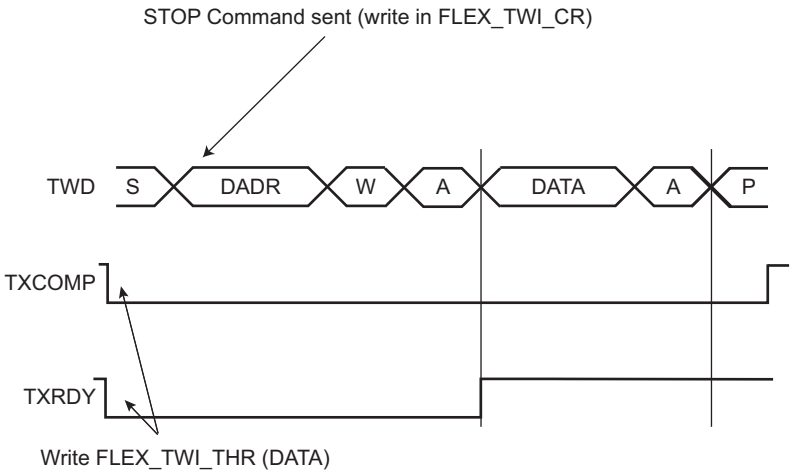


Figure 44-86. Master Write with Multiple Data Bytes

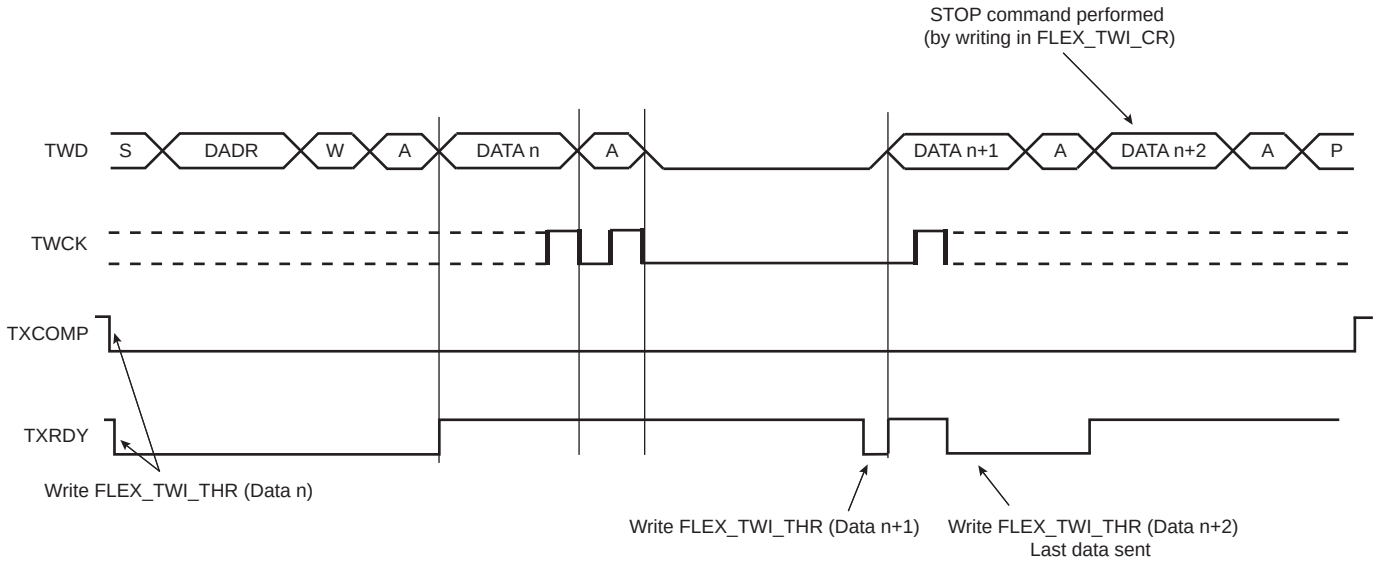
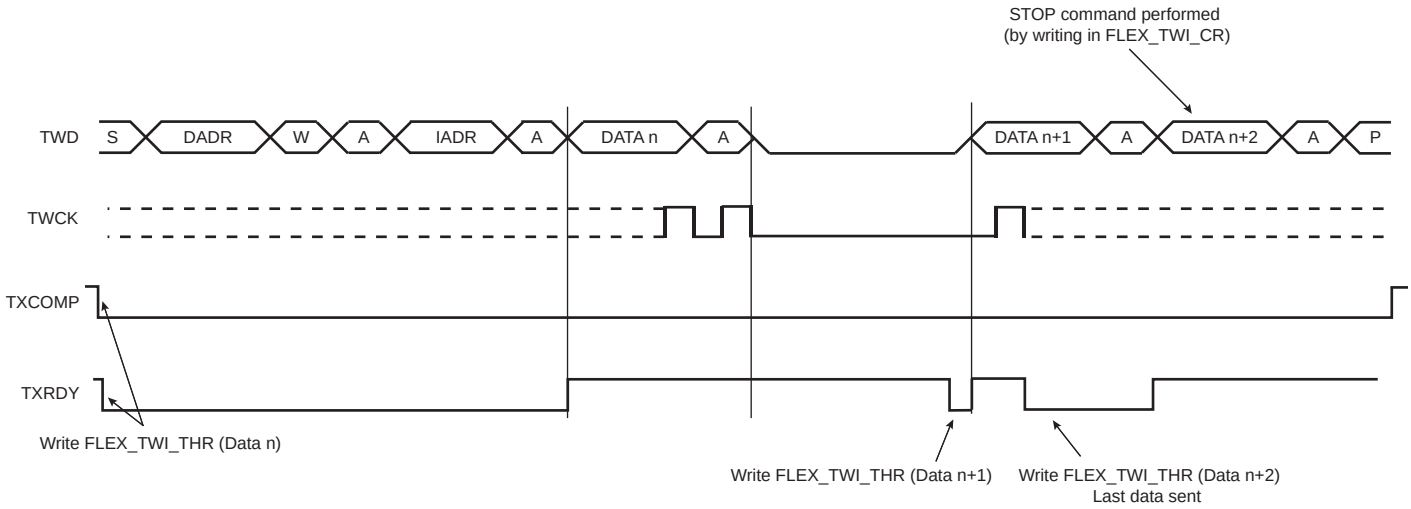


Figure 44-87. Master Write with One Byte Internal Address and Multiple Data Bytes



44.9.3.5 Master Receiver Mode

Master Receiver mode is not available if High-speed mode is selected.

The read sequence begins by setting the START bit. After the start condition has been sent, the master sends a 7-bit slave address to notify the slave device. The bit following the slave address indicates the transfer direction, 1 in this case (MREAD = 1 in FLEX_TWI_MMR). During the acknowledge clock pulse (9th pulse), the master releases the data line (HIGH), enabling the slave to pull it down in order to generate the acknowledge. The master polls the data line during this clock pulse and sets the NACK bit in FLEX_TWI_SR if the slave does not acknowledge the byte.

If an acknowledge is received, the master is then ready to receive data from the slave. After data has been received, the master sends an acknowledge condition to notify the slave that the data has been received except for the last data (see Figure 44-88). When the RXRDY bit is set in FLEX_TWI_SR, a character has been received in the Receive Holding Register (FLEX_TWI_RHR). The RXRDY bit is reset when reading FLEX_TWI_RHR.

When a single data byte read is performed, with or without internal address **(**IADR), the START and STOP bits must be set at the same time. See Figure 44-88. When a multiple data byte read is performed, with or without

internal address **(**IADR), the STOP bit must be set after the next-to-last data received (same condition applies for START bit to generate a repeated start). See Figure 44-89. For internal address usage, see Section 44.9.3.6 “Internal Address”.

If FLEX_TWI_RHR is full (RXRDY high) and the master is receiving data, the serial clock line will be tied low before receiving the last bit of the data and until FLEX_TWI_RHR is read. Once FLEX_TWI_RHR is read, the master will stop stretching the serial clock line and end the data reception. See Figure 44-90.

Warning: When receiving multiple bytes in Master Read mode, if the next-to-last access is not read (the RXRDY flag remains high), the last access will not be completed until FLEX_TWI_RHR is read. The last access stops on the next-to-last bit (clock stretching). When FLEX_TWI_RHR is read there is only half a bit period to send the STOP bit (or START bit) command, else another read access might occur (spurious access).

A possible workaround is to set the STOP bit (or START bit) before reading FLEX_TWI_RHR on the next-to-last access (within IT handler).

Figure 44-88. Master Read with One Data Byte

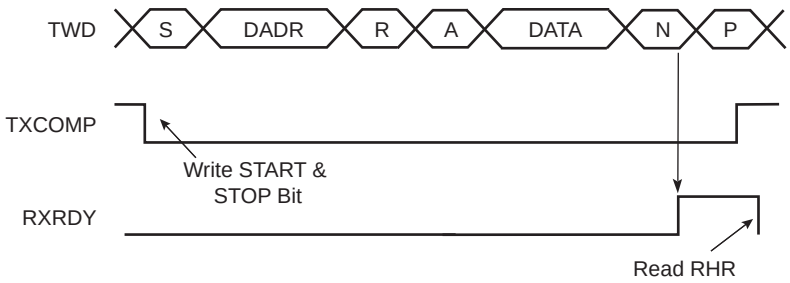


Figure 44-89. Master Read with Multiple Data Bytes

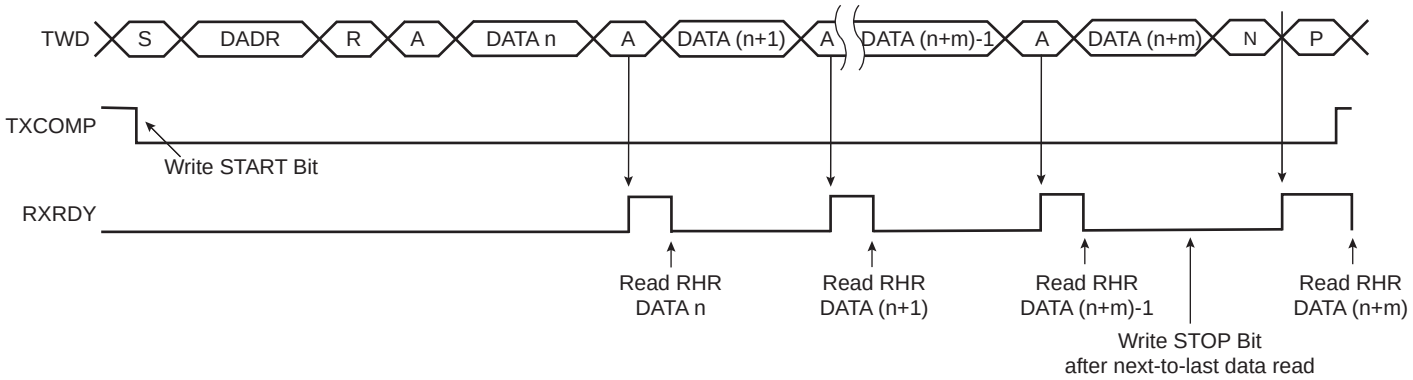
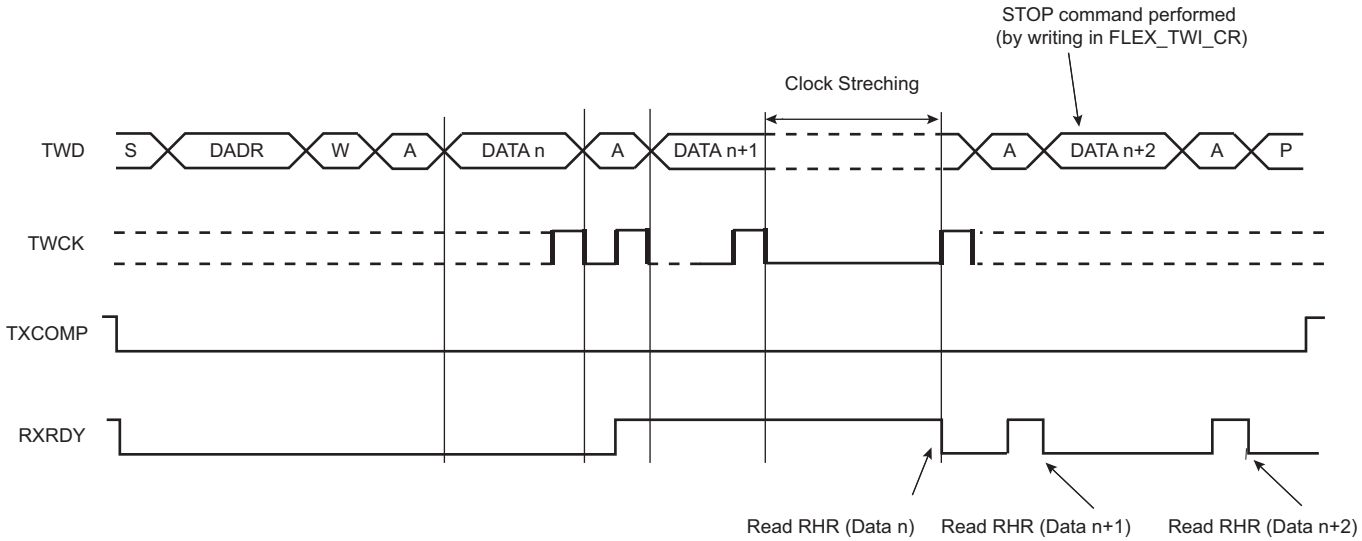


Figure 44-90. Master Read Clock Stretching with Multiple Data Bytes



RXRDY is used as receive ready trigger event for the DMA receive channel.

44.9.3.6 Internal Address

The TWI interface can perform transfers with 7-bit slave address devices and with 10-bit slave address devices.

7-bit Slave Addressing

When addressing 7-bit slave devices, the internal address bytes are used to perform random address (read or write) accesses to reach one or more data bytes, e.g., within a memory page location in a serial memory. When performing read operations with an internal address, the TWI performs a write operation to set the internal address into the slave device, and then switch to Master Receiver mode. Note that the second start condition (after sending the IADR) is sometimes called “repeated start” (Sr) in I^2C fully-compatible devices. See Figure 44-92.

See Figure 44-91 and Figure 44-93 for the master write operation with internal address.

The three internal address bytes are configurable through the Master Mode Register (FLEX_TWI_MMR).

If the slave device supports only a 7-bit address, i.e., no internal address, IADRSZ must be configured to 0.

The abbreviations listed below are used in Figure 44-91 and Figure 44-92:

S	Start
Sr	Repeated Start
P	Stop
W	Write
R	Read
A	Acknowledge
N	Not Acknowledge
DADR	Device Address
IADR	Internal Address

Figure 44-91. Master Write with One, Two or Three Bytes Internal Address and One Data Byte

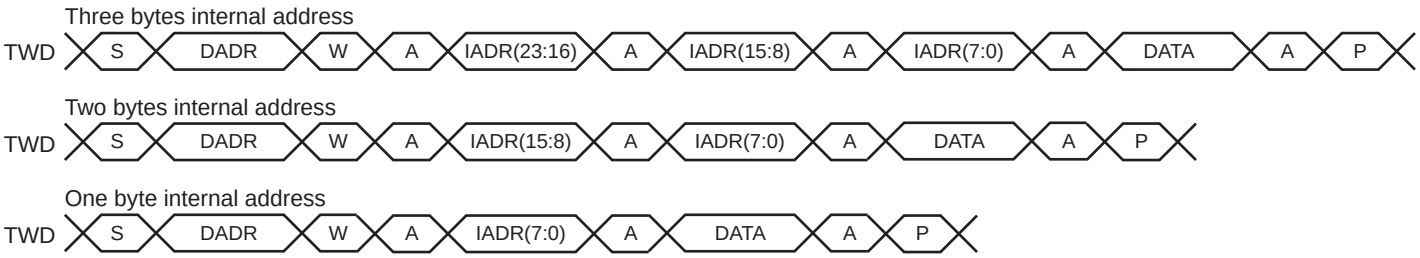
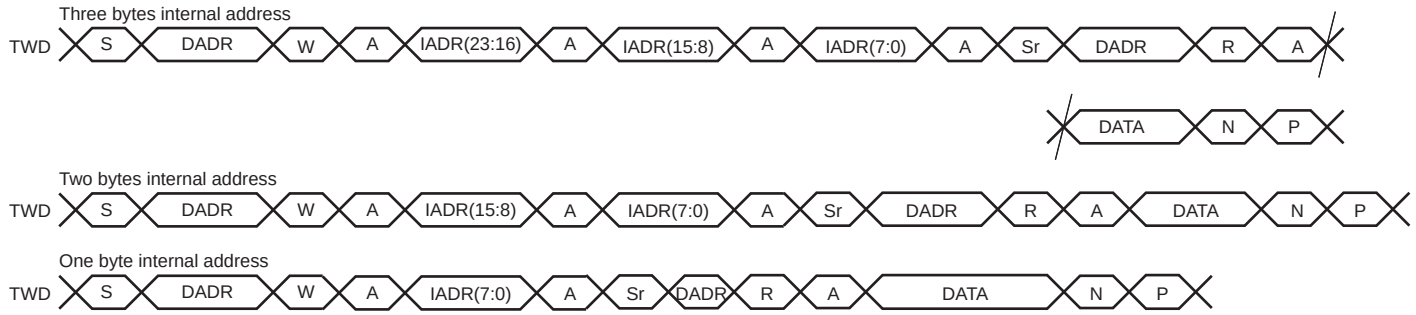


Figure 44-92. Master Read with One, Two or Three Bytes Internal Address and One Data Byte



10-bit Slave Addressing

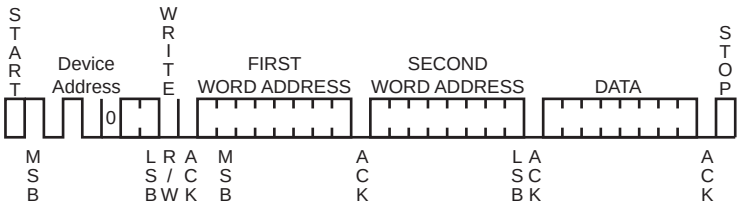
For a slave address higher than seven bits, the user must configure the address size **(**IADRSZ) and set the other slave address bits in the Internal Address Register (FLEX_TWI_IADR). The two remaining internal address bytes, IADR[15:8] and IADR[23:16], can be used the same way as in 7-bit slave addressing.

Example: Address a 10-bit device (10-bit device address is b1 b2 b3 b4 b5 b6 b7 b8 b9 b10)

1. Program IADRSZ = 1,
2. Program DADR with 1 1 1 1 0 b1 b2 (b1 is the MSB of the 10-bit address, b2, etc.)
3. Program FLEX_TWI_IADR with b3 b4 b5 b6 b7 b8 b9 b10 (b10 is the LSB of the 10-bit address)

Figure 44-93 shows a byte write to an Atmel AT24LC512 EEPROM. This demonstrates the use of internal addresses to access the device.

Figure 44-93. Internal Address Usage



44.9.3.7 Repeated Start

In addition to Internal Address mode, repeated start (Sr) can be generated manually by writing the START bit at the end of a transfer instead of the STOP bit. In such case the parameters of the next transfer (direction, SADR, etc.) will need to be set before writing the START bit at the end of the previous transfer.

See Section 44.9.3.13 for detailed flowcharts.

Note that generating a repeated start after a single data read is not supported.

44.9.3.8 Bus Clear Command

The TWI interface can perform a Bus Clear Command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Start the transfer by setting the CLEAR bit in FLEX_TWI_CR.

Note: If an alternative command is used (ACMEN bit = 1), the DATAL field must be cleared.

44.9.3.9 SMBus Mode

SMBus mode is enabled when the SMEN bit is written to one in FLEX_TWI_CR. SMBus mode operation is similar to I²C operation with the following exceptions:

1. Only 7-bit addressing can be used.
2. The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into FLEX_TWI_SMBTR.
3. Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).
4. A set of addresses has been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring FLEX_TWI_CR appropriately.

Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing the PECEN bit in FLEX_TWI_CR to one enables automatic PEC handling in the current transfer. Transfers with and without PEC can freely be intermixed in the same system, since some slaves may not support PEC. The PEC LFSR is always updated on every bit transmitted or received, so that PEC handling on combined transfers will be correct.

In Master Transmitter mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. Some slaves may not be able to check the received PEC in time to return a NACK if an error occurred. In this case, the slave should always return an ACK after the PEC byte, and some other mechanism must be implemented to verify that the transmission was received correctly.

In Master Receiver mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the PECERR bit in FLEX_TWI_SR is set. In Master Receiver mode, the PEC byte is always followed by a NACK transmitted by the master, since it is the last byte in the transfer.

In combined transfers, the PECRQ bit should only be set in the last of the combined transfers. If Alternative Command mode is enabled, only the NPEC bit should be set.

Consider the following transfer:

S, ADR+W, COMMAND_BYTE, ACK, SR, ADR+R, DATA_BYTE, ACK, PEC_BYTE, NACK, P

See Section 44.9.3.13 for detailed flowcharts.

Timeouts

The TLOWS and TLOWM fields in FLEX_TWI_SMBTR configure the SMBus timeout values. If a timeout occurs, the master will transmit a STOP condition and leave the bus. The TOUT bit is also set in FLEX_TWI_SR.

44.9.3.10 SMBus Quick Command (Master Mode Only)

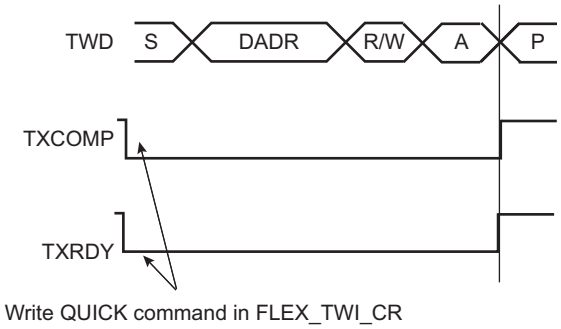
The TWI interface can perform a quick command:

1. Configure the Master mode (DADR, CKDIV, etc).
2. Write the MREAD bit in FLEX_TWI_MMR at the value of the one-bit command to be sent.

3. Start the transfer by setting the QUICK bit in FLEX_TWI_CR.

Note: If an alternative command is used (ACMEN bit = 1) DATAL field must be cleared.

Figure 44-94. SMBus Quick Command



44.9.3.11 Alternative Command

Another way to configure the transfer is to enable the Alternative Command mode with the ACMEN bit of the TWI Control Register.

In this mode, the transfer is configured through the TWI Alternative Command Register. It is possible to define a simple read or write transfer or a combined transfer with a repeated start.

In order to set a simple transfer, the DATAL field and the DIR field of the TWI Alternative Command Register must be filled accordingly and the NDATAL field must be cleared. To begin the transfer, either set the START bit in the TWI Control Register in case of a read transfer, or write the TWI Transmit Holding Register in case of a write transfer.

For a combined transfer linked by a repeated start, the NDATAL field must be filled with the length of the second transfer and NDIR with the corresponding direction.

The PEC and NPEC bits are used to set a PEC field. In the case of a single transfer with PEC, the PEC bit must be set. In the case of a combined transfer, the NPEC bit must be set.

Note: If the Alternative Command mode is used, IADRSZ in TWIHS_MMR must be set to 0.

See Section 44.9.3.13 for detailed flowcharts.

44.9.3.12 Handling Errors in Alternative Command

In case of NACK generated by a slave device or SMBus timeout error, the TWI stops immediately the frame but the DMA transfer may still be active. To prevent a new frame to be restarted with remaining DMA data (transmit), the TWI prevents any start of frame until the LOCK flag is cleared in FLEX_TWI_SR.

The LOCK bit in FLEX_TWI_SR indicates the state of the TWI (locked or not locked).

When the TWI is locked, no transfer will begin until the LOCK is cleared using the LOCKCLR bit in FLEX_TWI_CR and error flags cleared reading FLEX_TWI_SR.

In case of error, FLEX_TWI_THR may have been loaded with a new data. The THRCLR bit in FLEX_TWI_CR can be used to flush FLEX_TWI_THR. If the THRCLR bit is set, the TXRDY and TXCOMP flags are set.

44.9.3.13 Read/Write Flowcharts

The following flowcharts shown in Figure 44-96, Figure 44-97, Figure 44-102, Figure 44-103 and Figure 44-104 give examples for read and write operations. A polling or interrupt method can be used to check the status bits. The interrupt method requires that the Interrupt Enable Register (FLEX_TWI_IER) be configured first.

Figure 44-95. TWI Write Operation with Single Data Byte without Internal Address

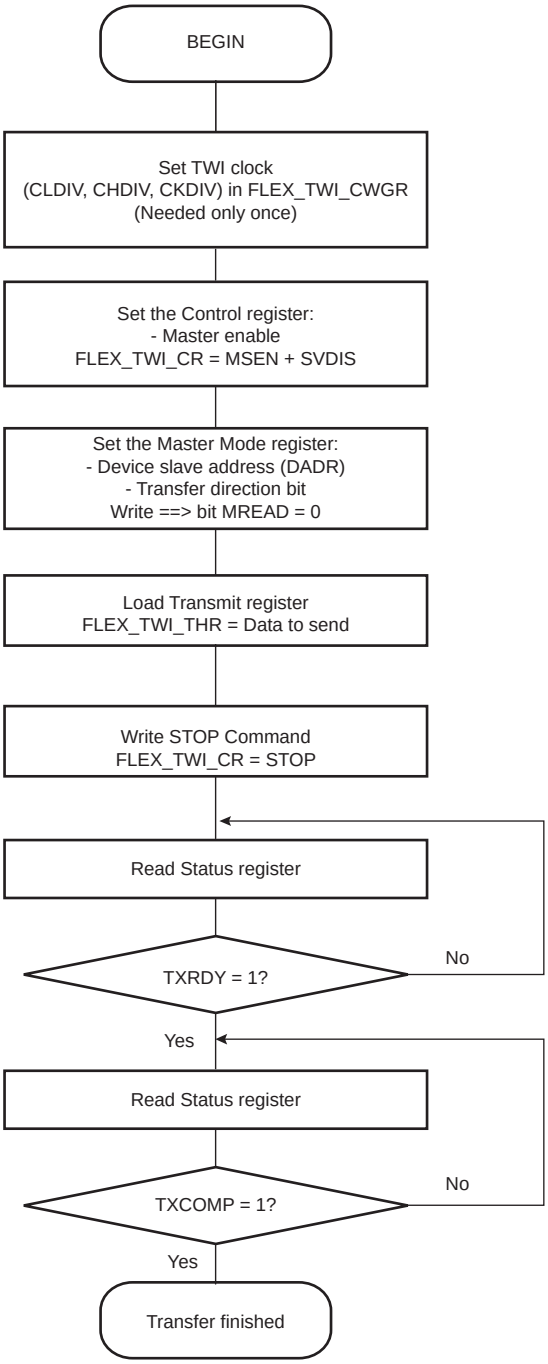


Figure 44-96. TWI Write Operation with Single Data Byte and Internal Address

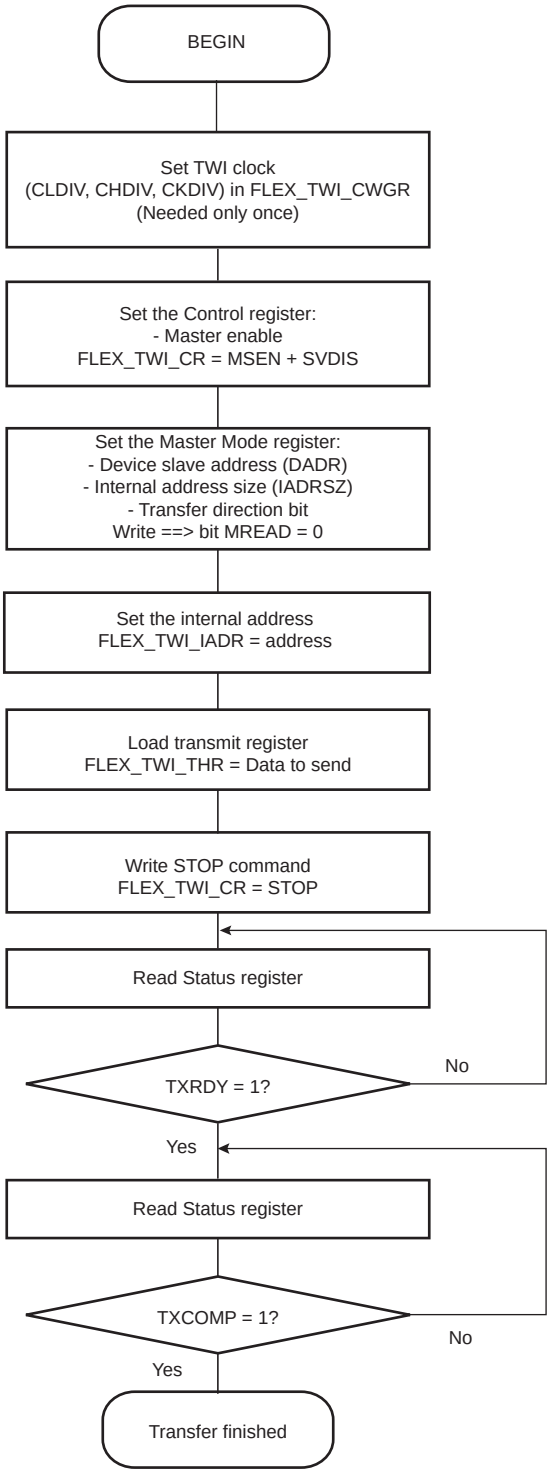


Figure 44-97. TWI Write Operation with Multiple Data Bytes with or without Internal Address

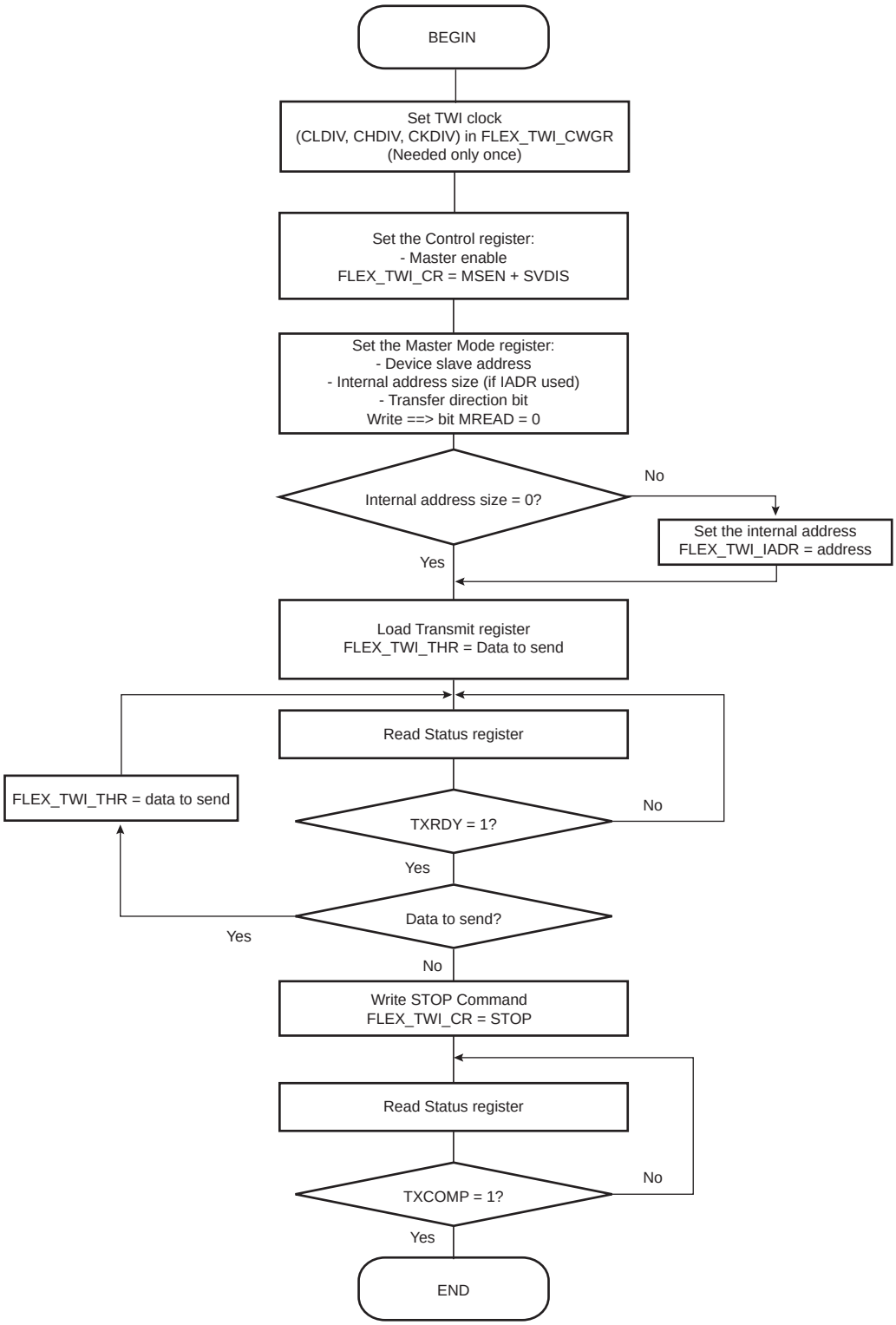


Figure 44-98. SMBus Write Operation with Multiple Data Bytes with or without Internal Address and PEC Sending

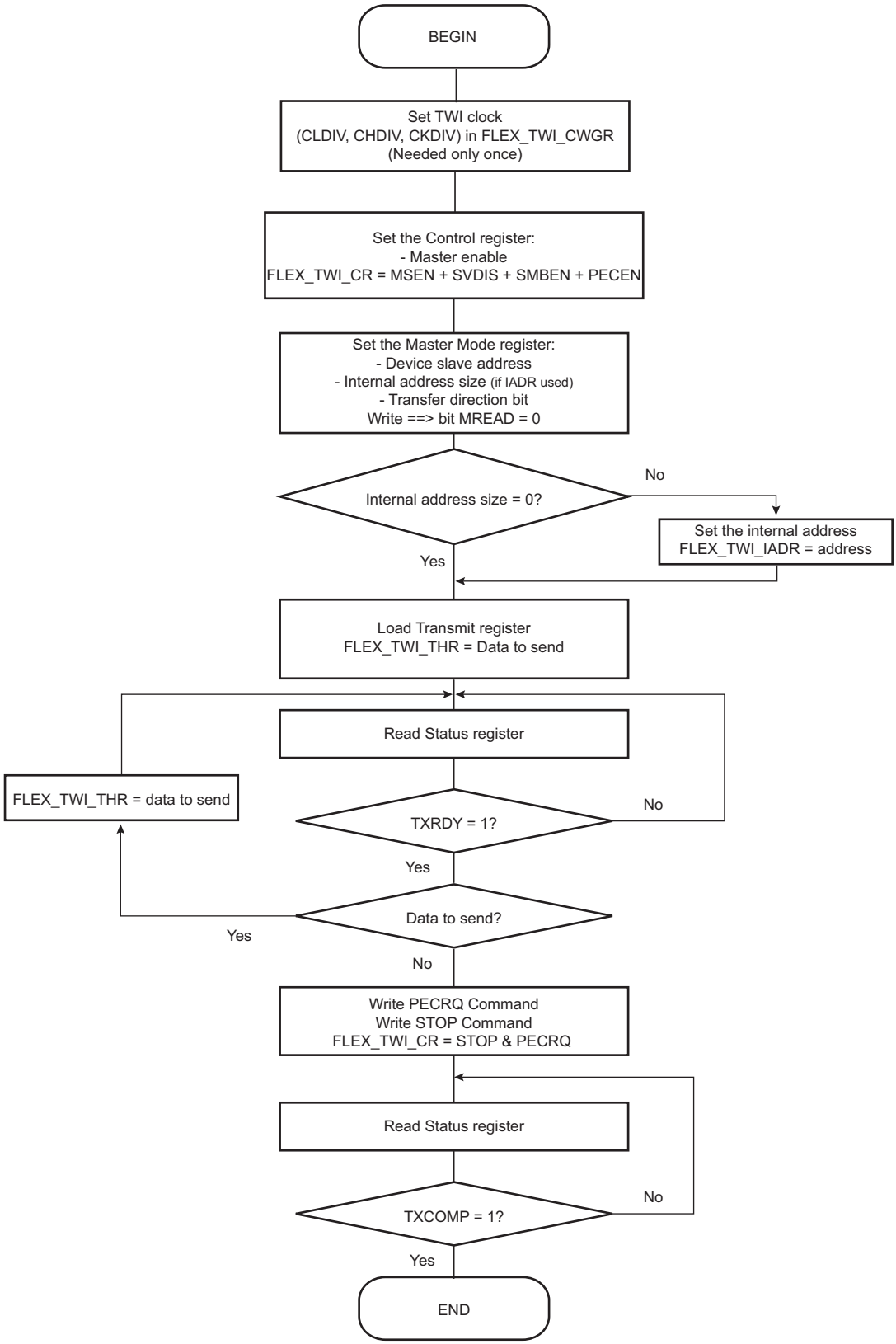


Figure 44-99. SMBus Write Operation with Multiple Data Bytes with PEC and Alternative Command Mode

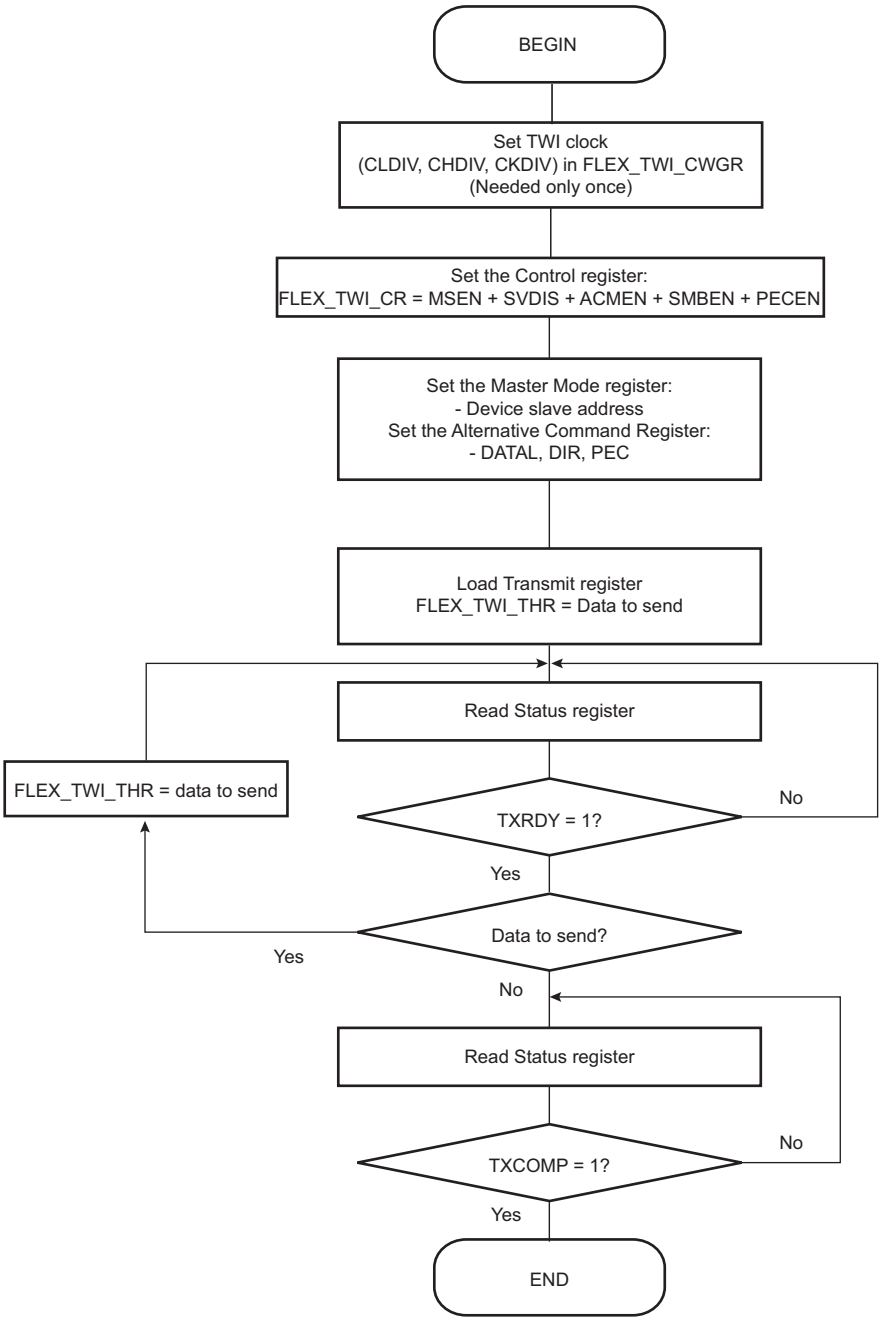


Figure 44-100. TWI Write Operation with Multiple Data Bytes and Read Operation with Multiple Data Bytes (Sr)

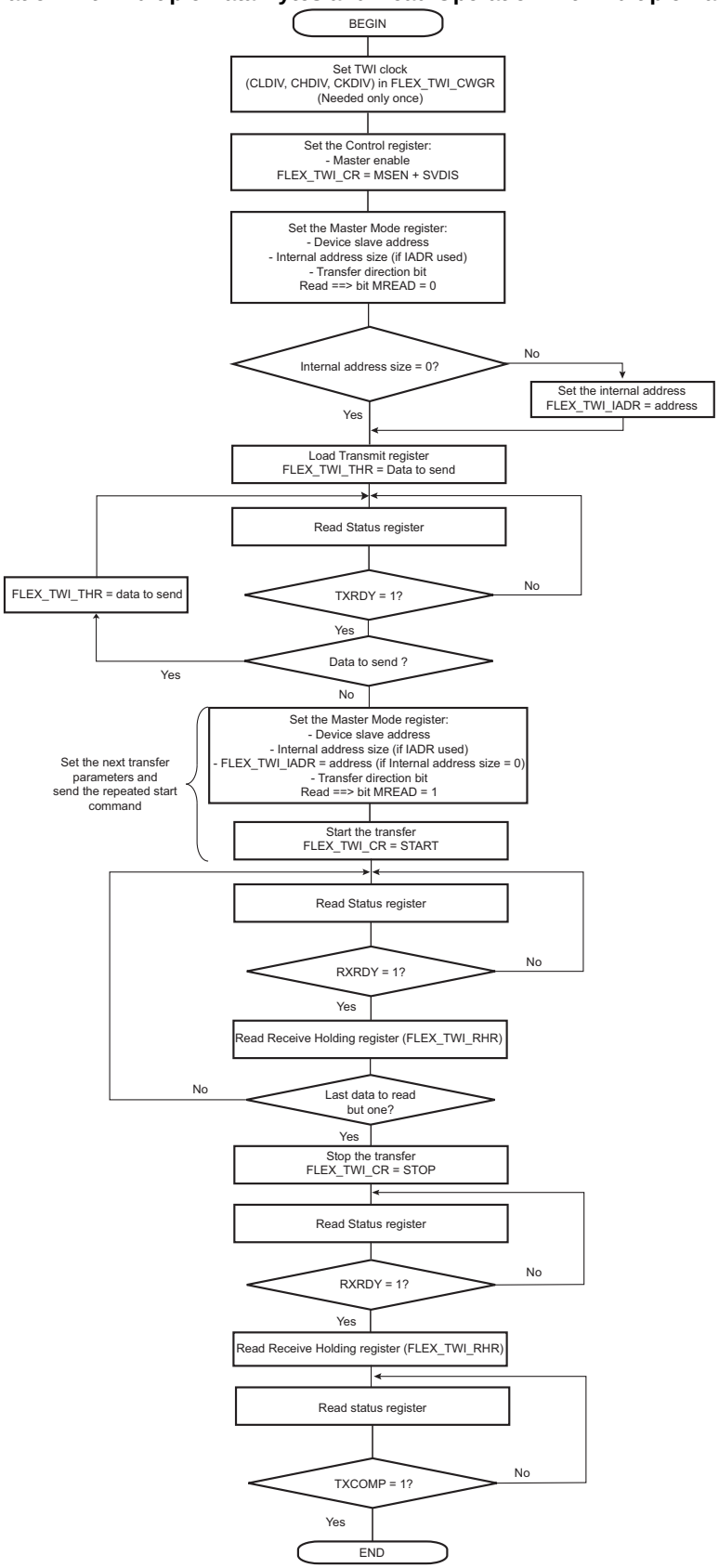


Figure 44-101. TWI Write Operation with Multiple Data Bytes + Read Operation and Alternative Command Mode + PEC

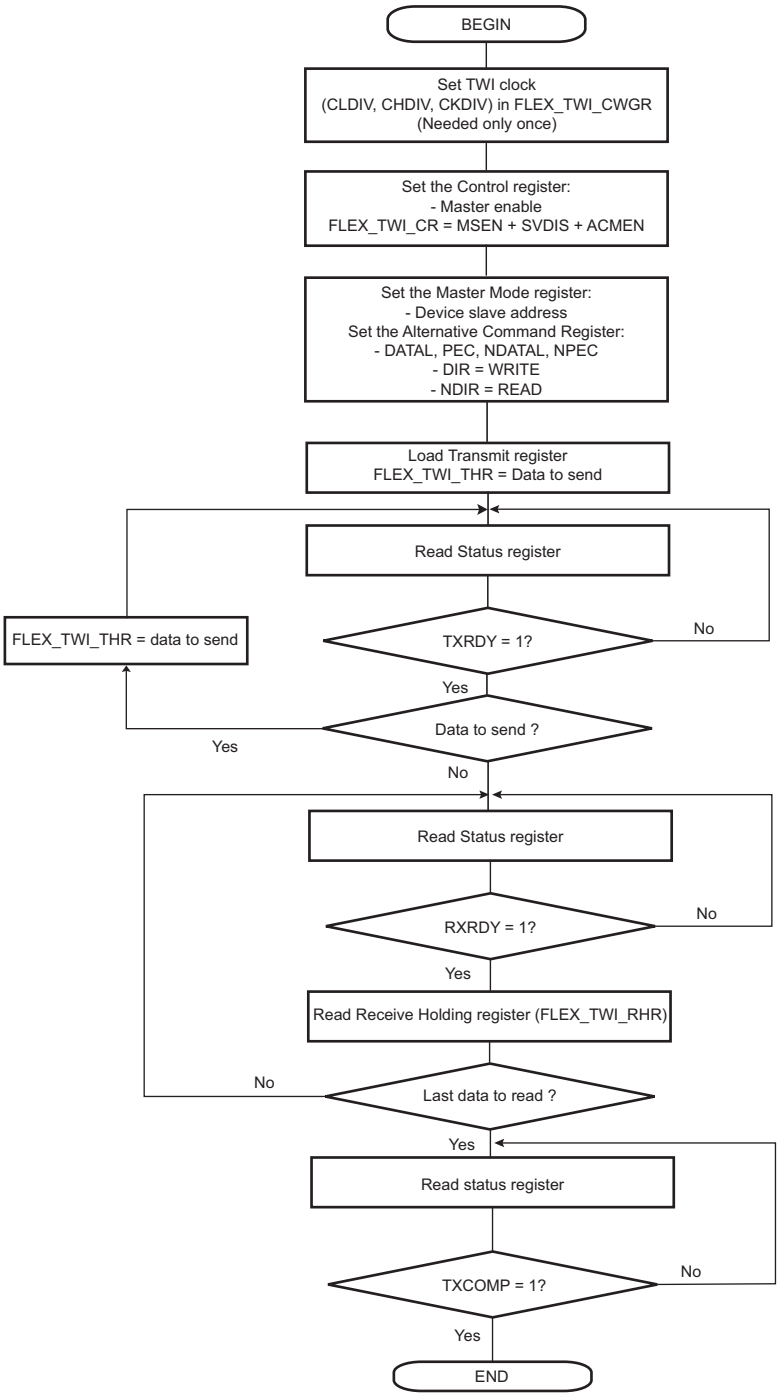


Figure 44-102. TWI Read Operation with Single Data Byte without Internal Address

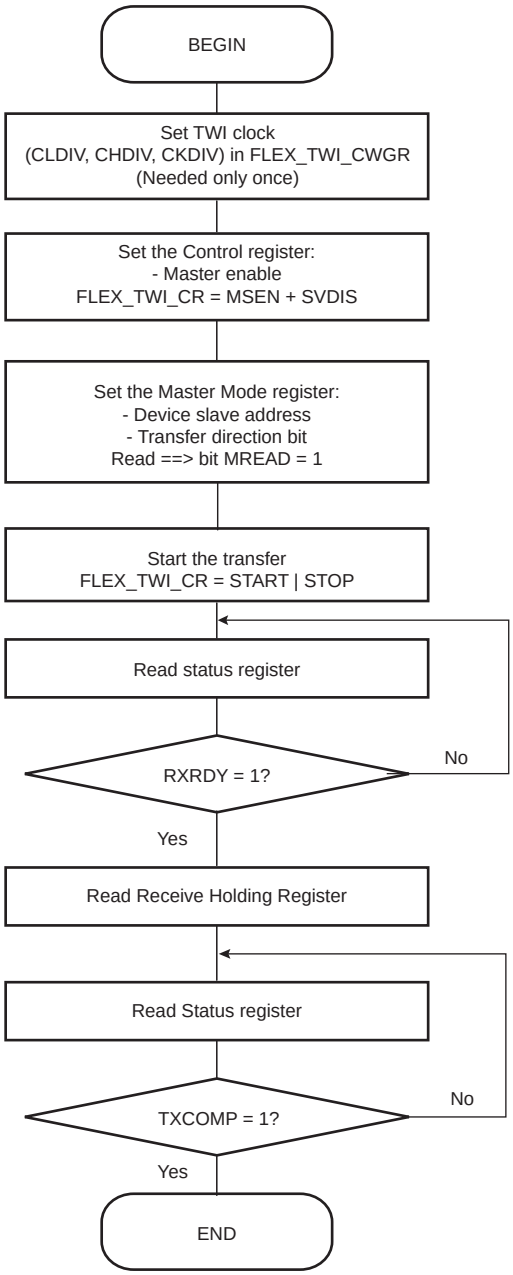


Figure 44-103. TWI Read Operation with Single Data Byte and Internal Address

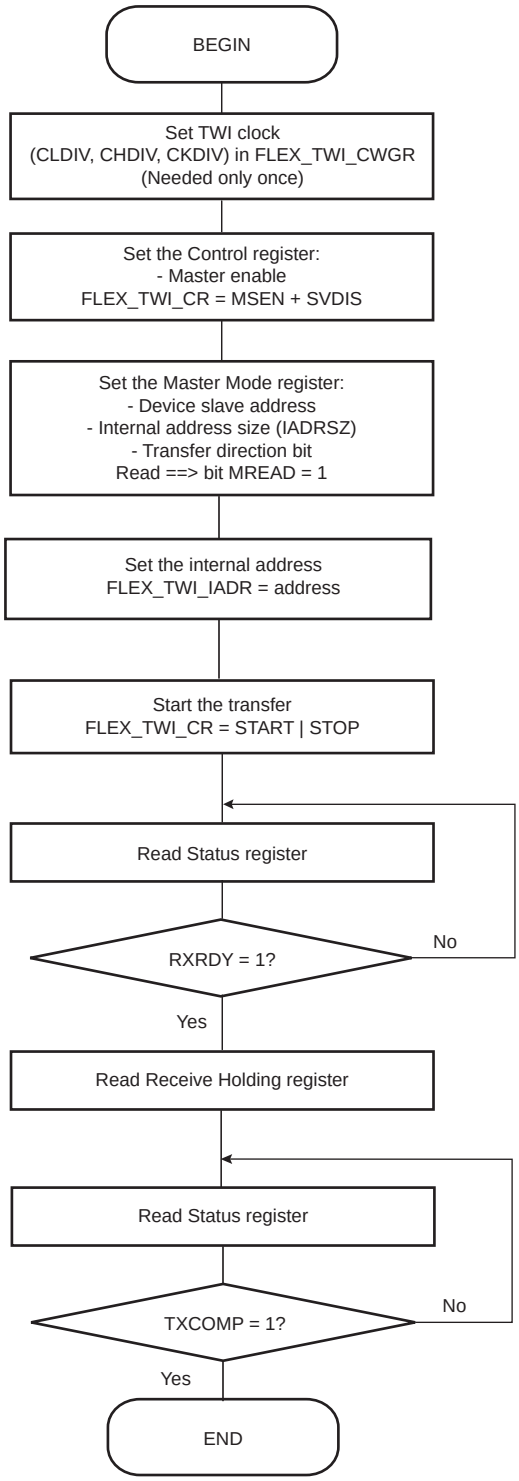


Figure 44-104. TWI Read Operation with Multiple Data Bytes with or without Internal Address

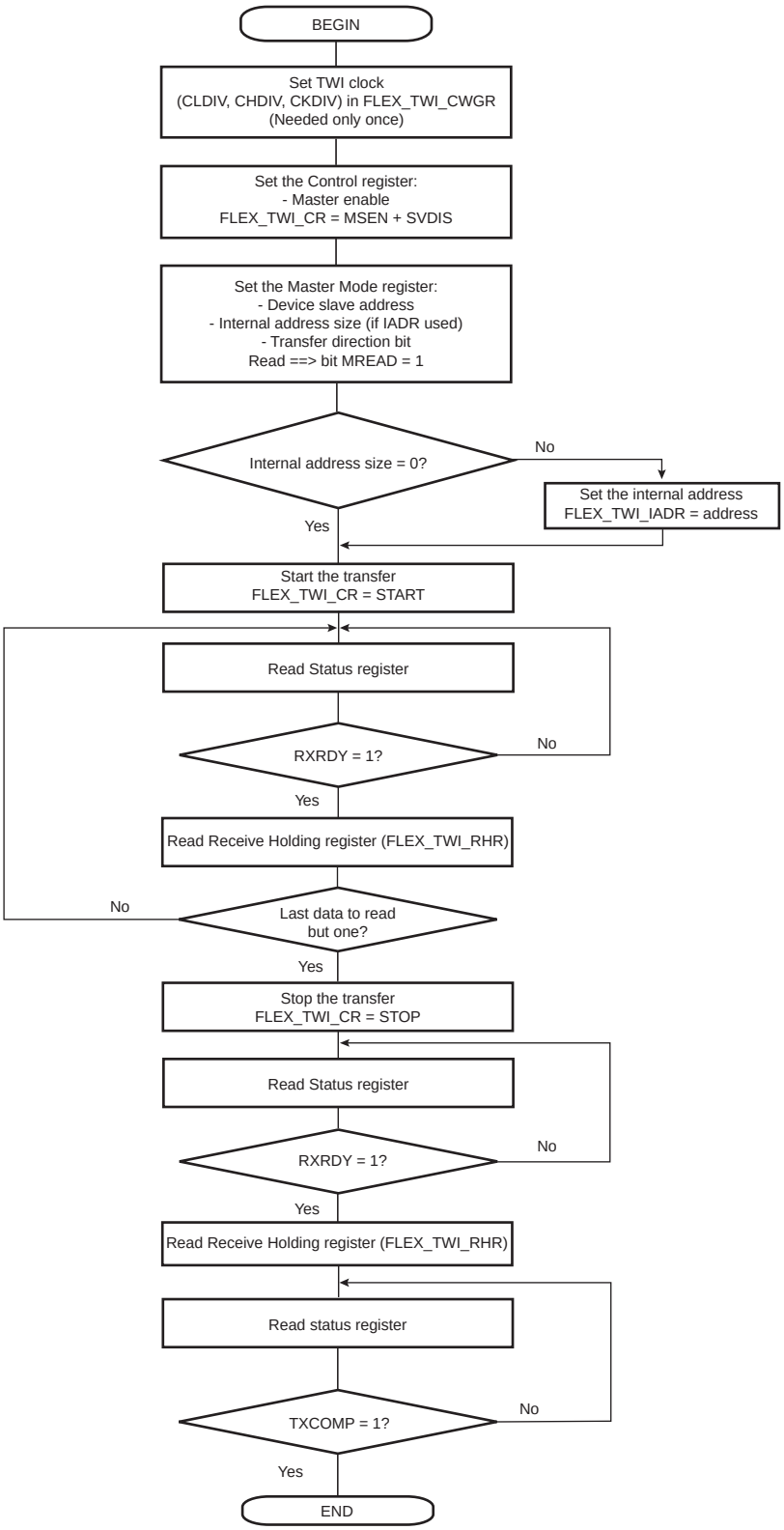


Figure 44-105. TWI Read Operation with Multiple Data Bytes with or without Internal Address with PEC

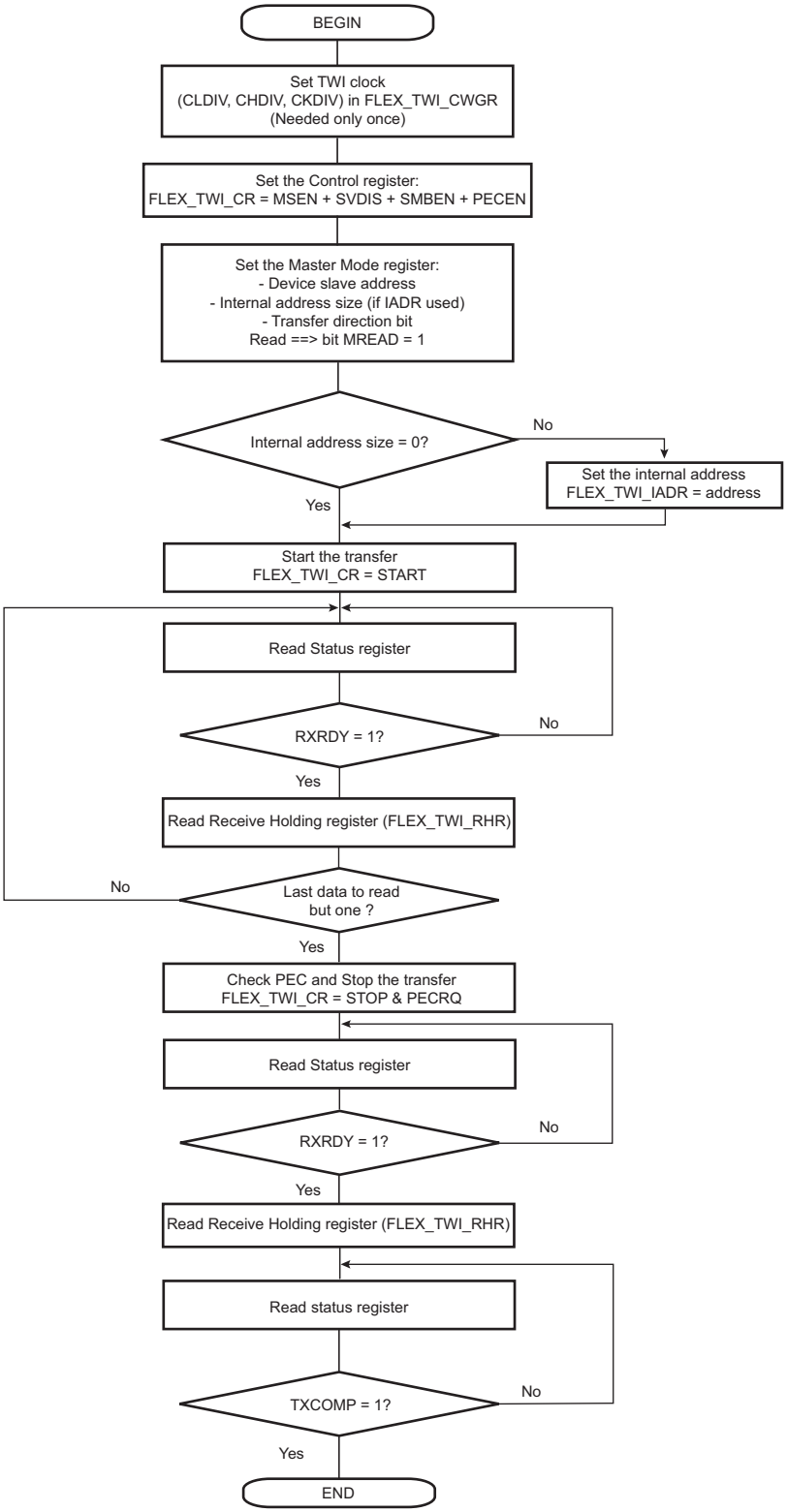


Figure 44-106. TWI Read Operation with Multiple Data Bytes with Alternative Command Mode with PEC

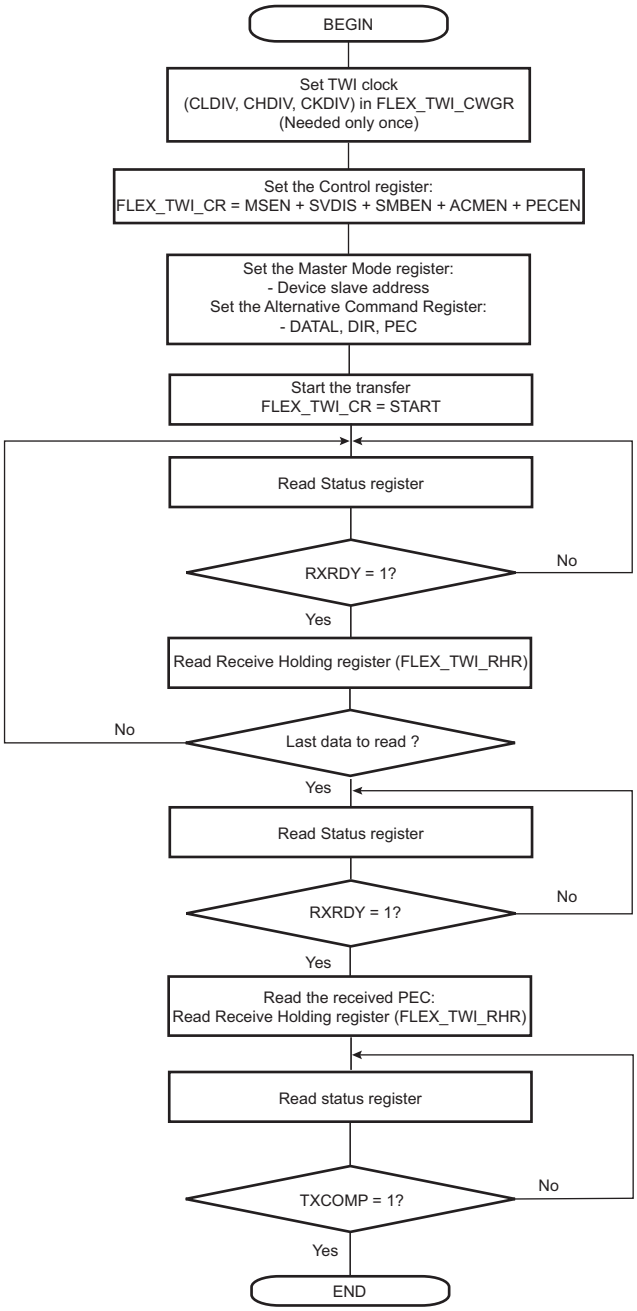


Figure 44-107. TWI Read Operation with Multiple Data Bytes + Write Operation with Multiple Data Bytes (Sr)

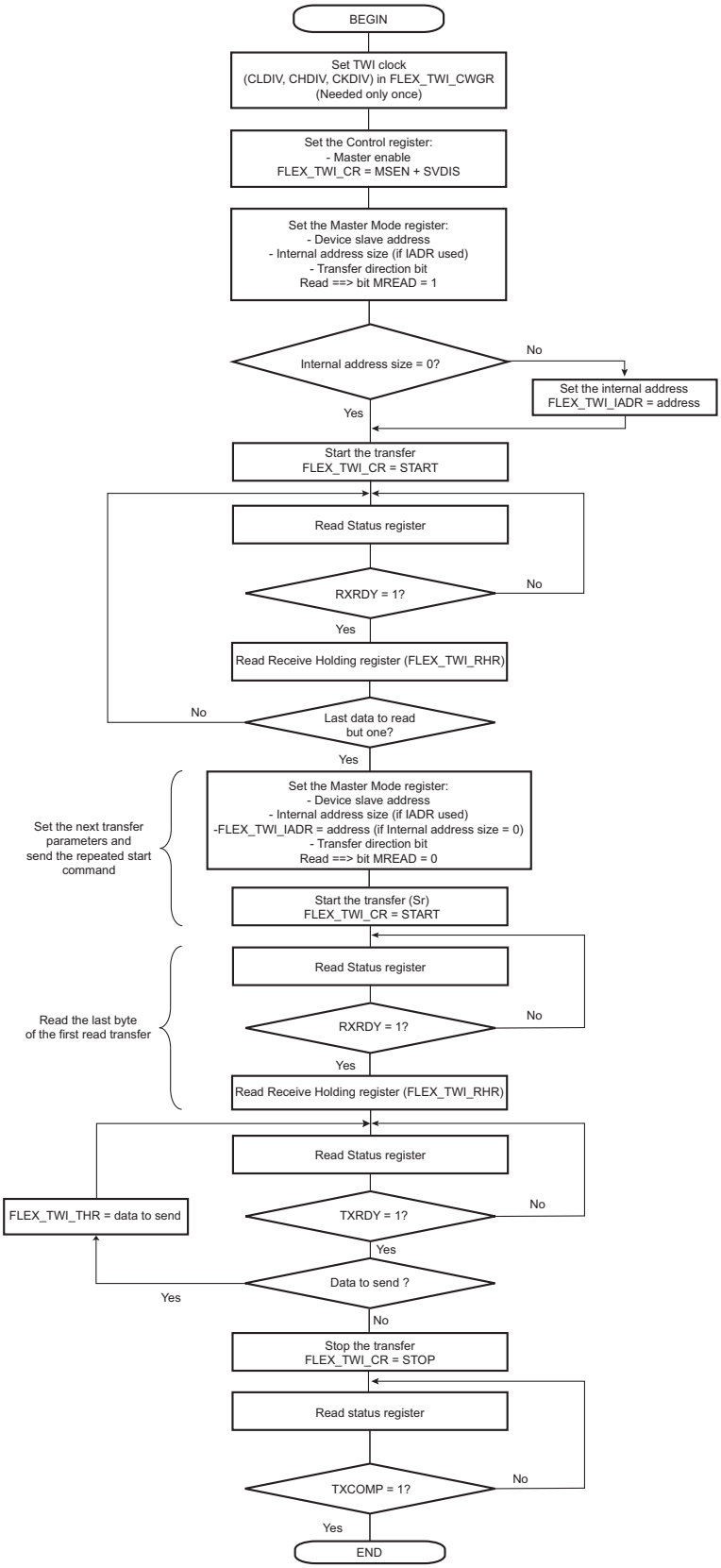


Figure 44-108. TWI Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC

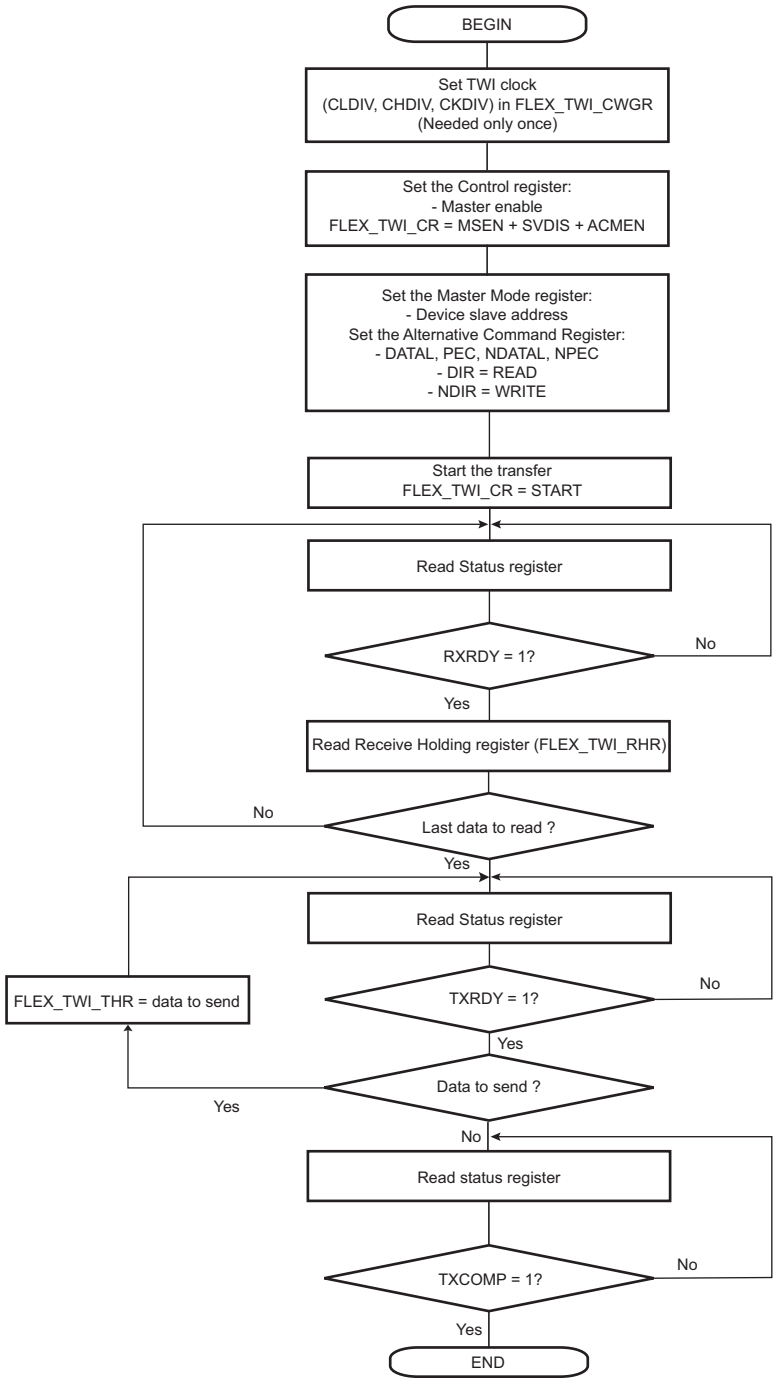
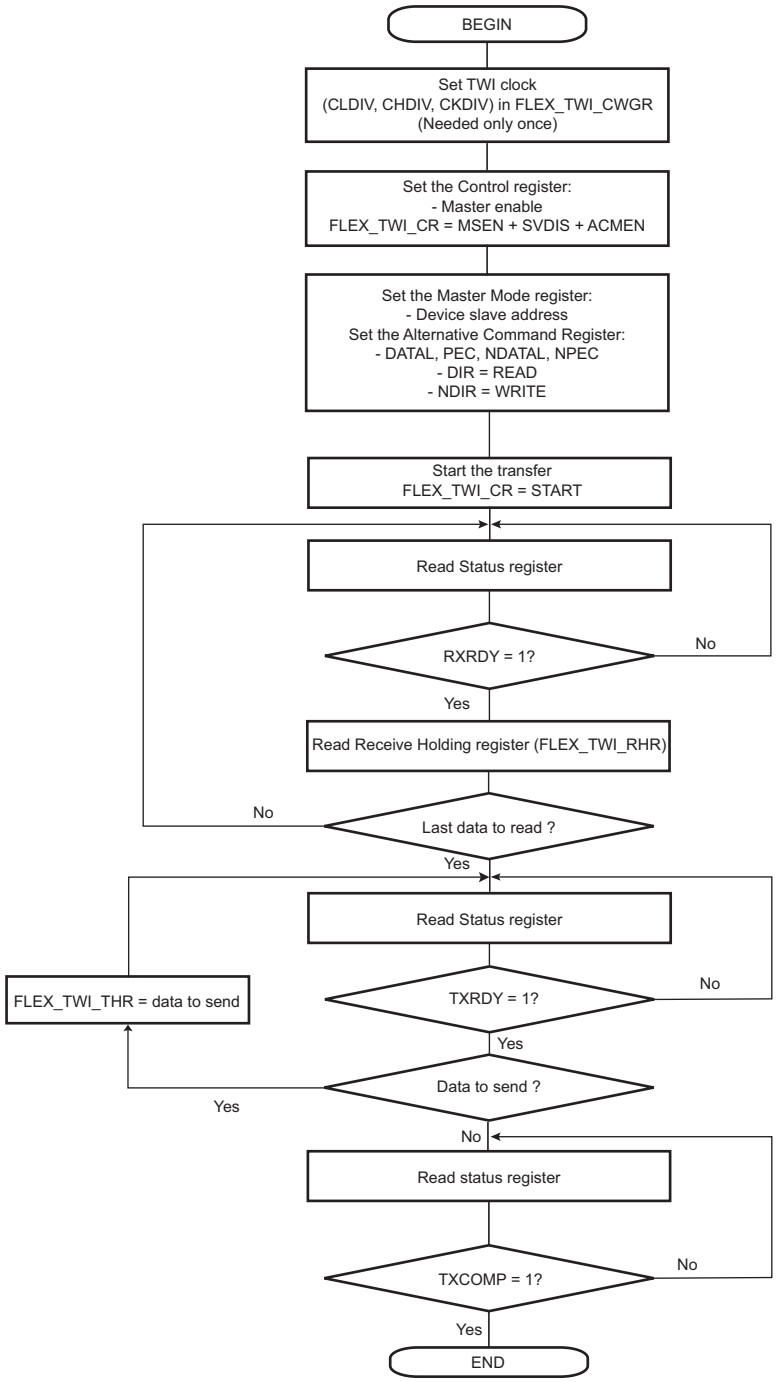


Figure 44-109. TWI Read Operation with Multiple Data Bytes + Write with Alternative Command Mode with PEC

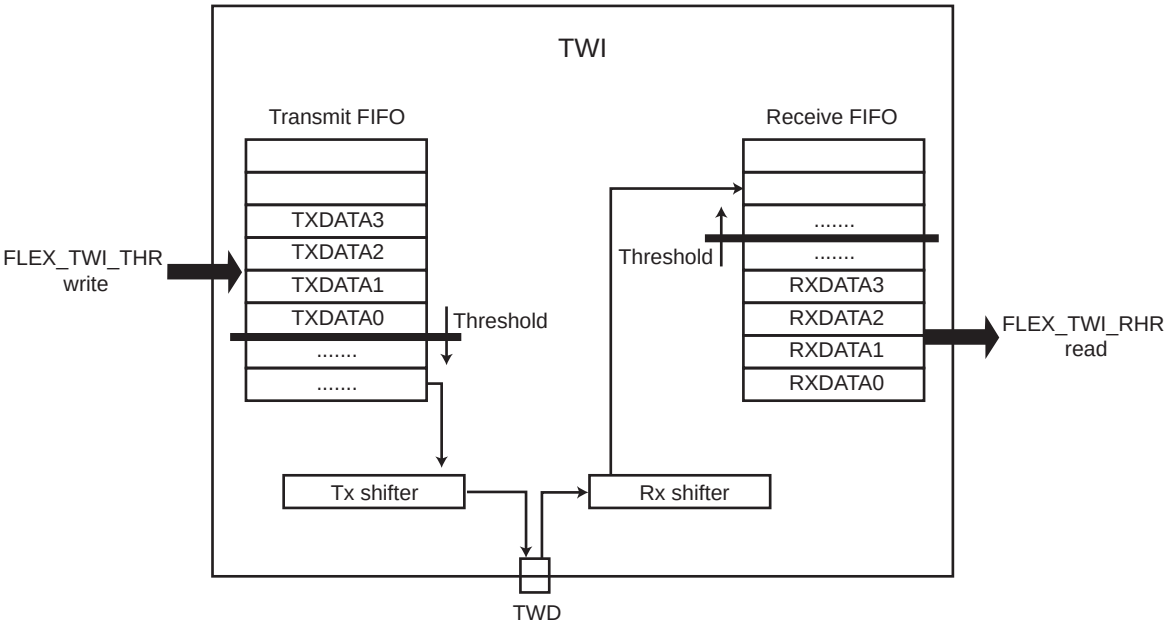


44.9.3.14 FIFOs

The TWI includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in FLEX_TWI_CR. It is recommended to disable both Master and Slave modes before enabling or disabling the FIFO (MSDIS and SVDIS bit in FLEX_TWI_CR).

Writing the FIFOEN bit to ‘1’ will enable a 16-data Transmit FIFO and a 16-data Receive FIFO.

Figure 44-110. FIFOs Block Diagram

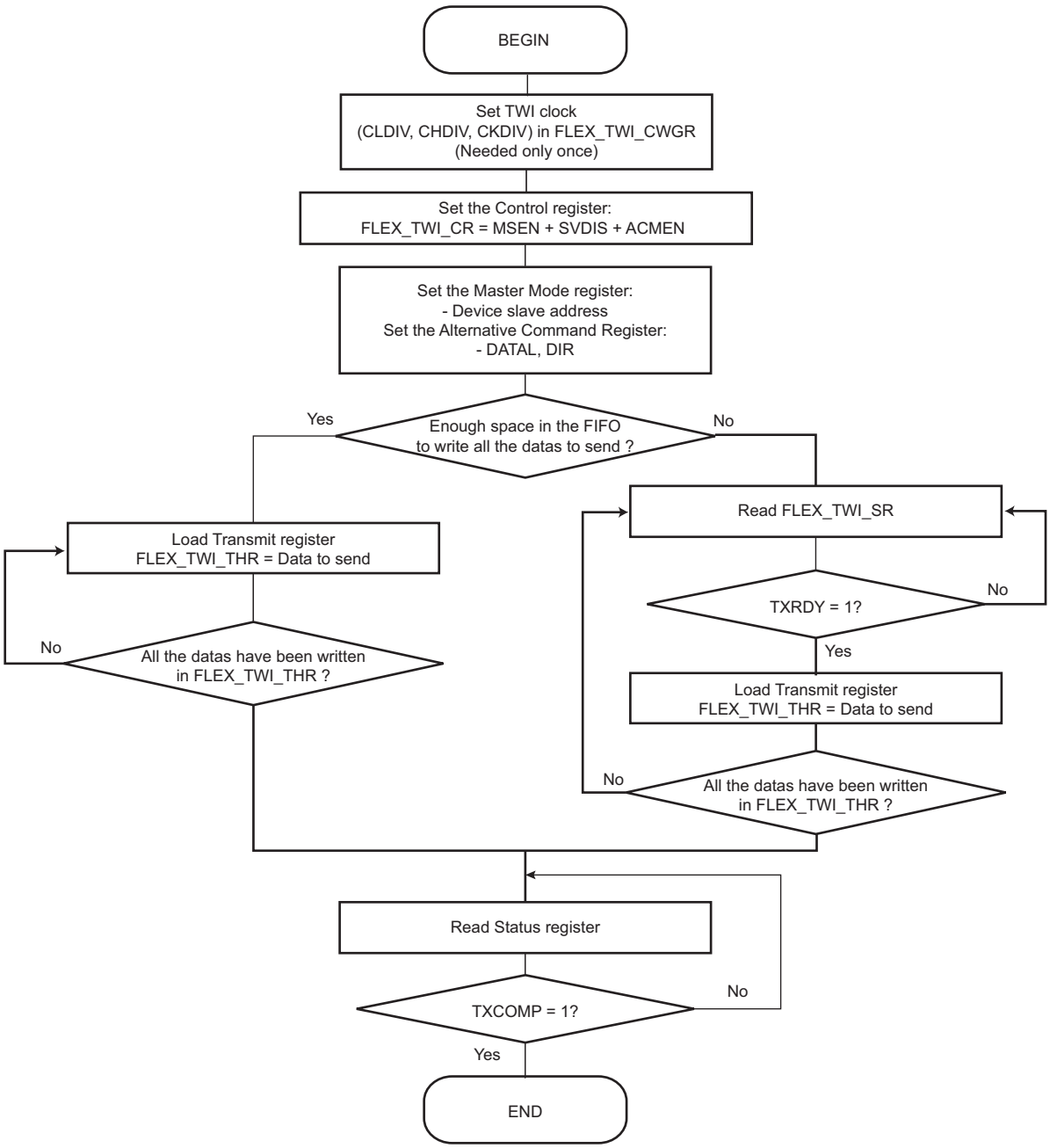


Sending Data with FIFO Enabled

With the Transmit FIFO enabled, any write access to the TWI Transmit Holding Register (FLEX_TWI_THR) brings the written data to the Transmit FIFO. As a consequence, it is not mandatory any more to monitor the TXRDY flag state to send multiple data without DMAC.

Knowing the number of data to send and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in FLEX_TWI_THR without checking the TXRDY flag between each access. The Transmit FIFO state can be checked reading the TXFL field in the TWI FIFO Level Register (FLEX_TWI_FLR).

Figure 44-111. Sending Data with FIFO Flowchart

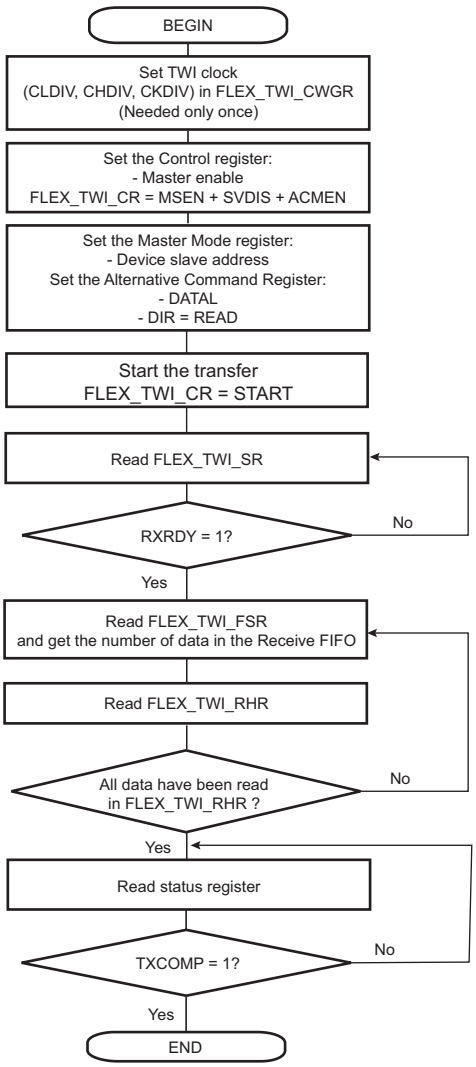


Receiving Data with FIFO Enabled

With Receive FIFO enabled, any read access on FLEX_TWI_RHR will pull out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RXRDY flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with the RXFL field in FLEX_TWI_FLR and all the data read successively in FLEX_TWI_RHR without checking the RXRDY flag between each access.

Figure 44-112. Receiving Data with FIFO Flowchart



Clearing/Flushing FIFOs

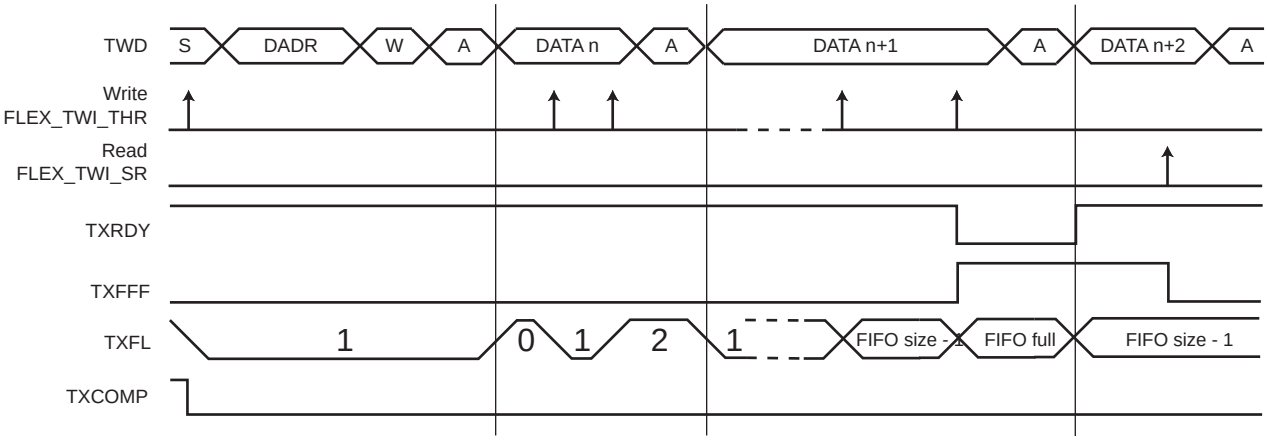
Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in FLEX_TWI_CR.

TXRDY and RXRDY Behavior

If FIFOs are enabled, the behavior of the TXRDY and RXRDY flags will be slightly different.

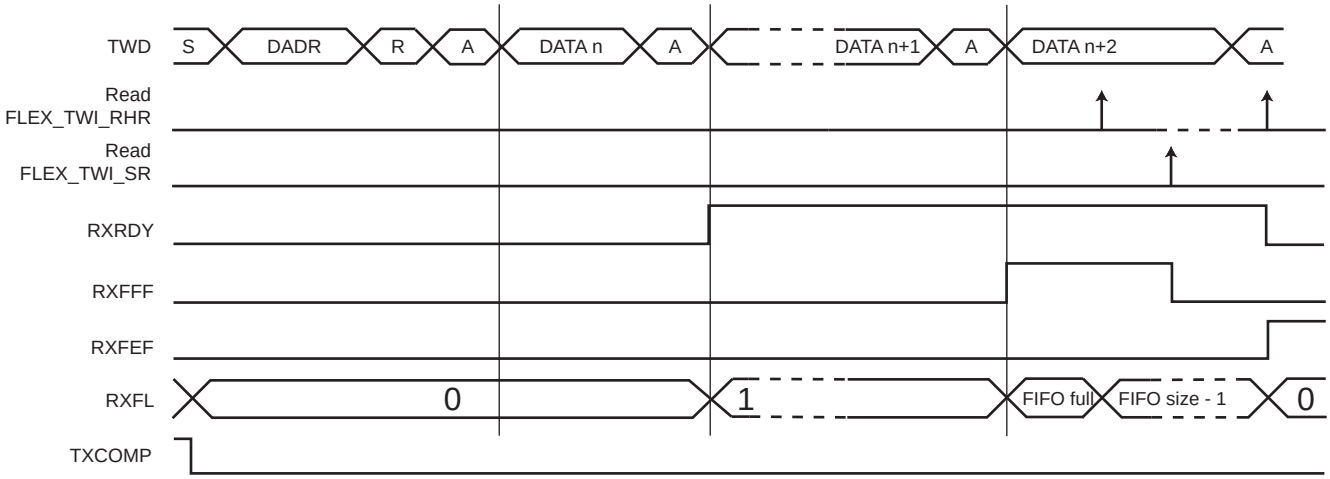
TXRDY will indicate if a data can be written in the Transmit FIFO. By default, the TXRDY flag will then stay at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 44-113. TXRDY in Single Data Mode and TXRDYM = 0x0



RXRDY will indicate if an unread data is present in the Receive FIFO. By default, the RXRDY flag will then be at level '1' as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 44-114. RXRDY in Single Data Mode and RXRDYM = 0x0



TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in the TWI FIFO Mode Register (FLEX_TWI_FMR). In Single Data mode, there is no need to modify the TXRDY and RXRDY behavior; however, it may be useful in Multiple Data Mode.

Master Multiple Data Mode

When the FIFOs are enabled, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_TWI_THR/FLEX_TWI_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte size register access, only one data will be written/read; if the access is a halfword-size register access, then two data will be written/read; finally, if the access is a word-size register access, then four data will be written/read.

Written/Read data are always right-aligned, as shown in Section 44.10.70 "TWI Receive Holding Register (FIFO Enabled)" and Section 44.10.72 "TWI Transmit Holding Register (FIFO Enabled)".

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six FLEX_TWI_THR-byte write accesses.
- Perform three FLEX_TWI_THR-halfword write accesses.

- Perform one FLEX_TWI_THR-word write access and one FLEX_TWI_THR halfword write access.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six FLEX_TWI_RHR-byte read accesses.
- Perform three FLEX_TWI_RHR-halfword read accesses.
- Perform one FLEX_TWI_RHR-word read access and one FLEX_TWI_RHR-halfword read access.

This mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

• TXRDY and RXRDY Configuration

In Multiple Data mode, the TXRDYM and RXRDYM fields in FLEX_TWI_FMR become useful.

As in Multiple Data mode, it is possible to write several data in the same access it might be useful to configure TXRDY flag behavior to indicate if 1, 2 or 4 data can be written in the FIFO depending on the access to perform on FLEX_TWI_THR.

If, for instance, four data are written each time in FLEX_TWI_THR it might be useful to configure TXRDYM field to 0x2 value so that TXRDY flag will be at ‘1’ only when at least four data can be written in the Transmit FIFO.

In the same way, if four data are read each time in FLEX_TWI_RHR it might be useful to configure RXRDYM field to 0x2 value so that RXRDY flag will be at ‘1’ only when at least four unread data are in the Receive FIFO.

• DMAC

If DMAC transfer is used it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, reset DMAC channels, etc., without any risk.

The LOCK bit in FLEX_TWI_SR is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared setting the TXFLCLR bit to ‘1’ in FLEX_TWI_CR.

FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

If the Transmit FIFO is full and a write access is performed on FLEX_TWI_THR, it will generate a Transmit FIFO pointer error and set the TXFPTEF flag in FLEX_TWI_FSR.

In Multiple Data mode, if the number of data written in FLEX_TWI_THR (according to the register access size) is bigger than the Transmit FIFO free space, it will generate a Transmit FIFO pointer error and set the TXFPTEF flag in FLEX_TWI_FSR.

- Receive FIFO:

In Multiple Data mode, if the number of data read in FLEX_TWI_RHR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it will generate a Receive FIFO pointer error and set the RXFPTEF flag in FLEX_TWI_FSR.

Pointer error should not happen if FIFO state is checked before writing/reading in FLEX_TWI_THR/FLEX_TWI_RHR. FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using the SWRST bit in FLEX_TWI_CR. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent afterwards might not be received properly by the slave).

FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO: The Transmit FIFO threshold can be set using the TXFTHRES field in FLEX_TWI_FMR. Each time the Transmit FIFO goes from the ‘above threshold’ to the ‘equal or below threshold’ state, the TXFTHF flag in FLEX_TWI_FSR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.
- Receive FIFO: The Receive FIFO threshold can be set using the RXFTHRES field in FLEX_TWI_FMR. Each time the Receive FIFO goes from the ‘below threshold’ to the ‘equal to or above threshold’ state, the RXFTHF flag in FLEX_TWI_FSR is set. This enables the application to know that the Receive FIFO reached the defined threshold and to read some data before it becomes full.

The TXFTHF and RXFTHF flags can be both configured to generate an interrupt using FLEX_TWI_FIER and FLEX_TWI_FIDR.

FIFO Flags

FIFOs come with a set of flags which can be configured each to generate an interrupt through FLEX_TWI_FIER and FLEX_TWI_FIDR.

FIFO flags state can be read in FLEX_TWI_FSR. They are cleared on FLEX_TWI_FSR read.

44.9.4 Multi-Master Mode

44.9.4.1 Definition

In Multi-Master mode, more than one master may handle the bus at the same time without data corruption by using arbitration.

Arbitration starts as soon as two or more masters place information on the bus at the same time, and stops (arbitration is lost) for the master that intends to send a logical one while the other master sends a logical zero.

As soon as arbitration is lost by a master, it stops sending data and listens to the bus in order to detect a STOP. When the STOP is detected, the master that has lost arbitration may put its data on the bus by respecting arbitration.

Arbitration is illustrated in Figure 44-116.

44.9.4.2 Different Multi-Master Modes

Two Multi-Master modes may be distinguished:

- TWI as Master Only—TWI is considered as a master only and will never be addressed.
- TWI as Master or Slave—TWI may be either a master or a slave and may be addressed.

Note: Arbitration in supported in both Multi-Master modes.

TWI as Master Only

In this mode, the TWI is considered as a master only (MSEN is always at one) and must be driven like a master with the ARBLST (ARBitration Lost) flag in addition.

If arbitration is lost (ARBLST = 1), the user must reinitiate the data transfer.

If the user starts a transfer (ex.: DADR + START + W + Write in THR) and if the bus is busy, the TWI automatically waits for a STOP condition on the bus to initiate the transfer (see Figure 44-115).

Note: The state of the bus (busy or free) is not indicated in the user interface.

TWI as Master or Slave

The automatic reversal from master to slave is not supported in case of a lost arbitration.

Then, in the case where TWI may be either a master or a slave, the user must manage the pseudo Multi-Master mode described in the steps below:

1. Program the TWI in Slave mode (SADR + MSDIS + SVEN) and perform a slave access (if TWI is addressed).
2. If the TWI has to be set in Master mode, wait until TXCOMP flag is at 1.
3. Program the Master mode (DADR + SVDIS + MSEN) and start the transfer (ex: START + Write in THR).
4. As soon as the Master mode is enabled, the TWI scans the bus in order to detect if it is busy or free. When the bus is considered as free, the TWI initiates the transfer.
5. As soon as the transfer is initiated and until a STOP condition is sent, the arbitration becomes relevant and the user must monitor the ARBLST flag.
6. If the arbitration is lost (ARBLST is = 1), the user must program the TWI in Slave mode in case the master that won the arbitration needs to access the TWI.
7. If the TWI has to be set in Slave mode, wait until TXCOMP flag is at 1 and then program the Slave mode.

Note: In case the arbitration is lost and the TWI is addressed, the TWI will not acknowledge even if it is programmed in Slave mode as soon as ARBLST = 1. Then the master must repeat SADR.

Figure 44-115. User Sends Data While the Bus is Busy

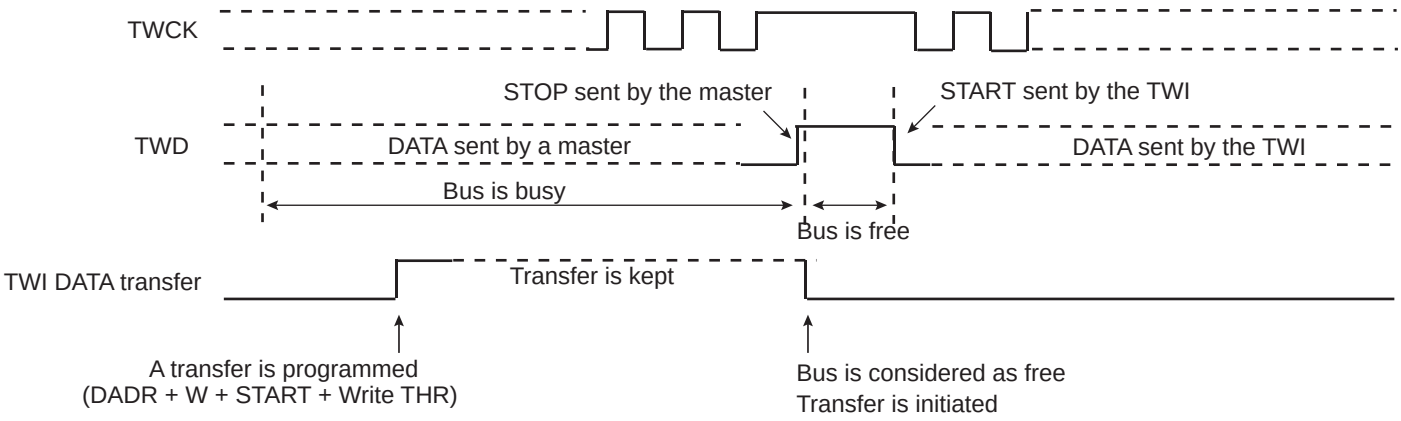
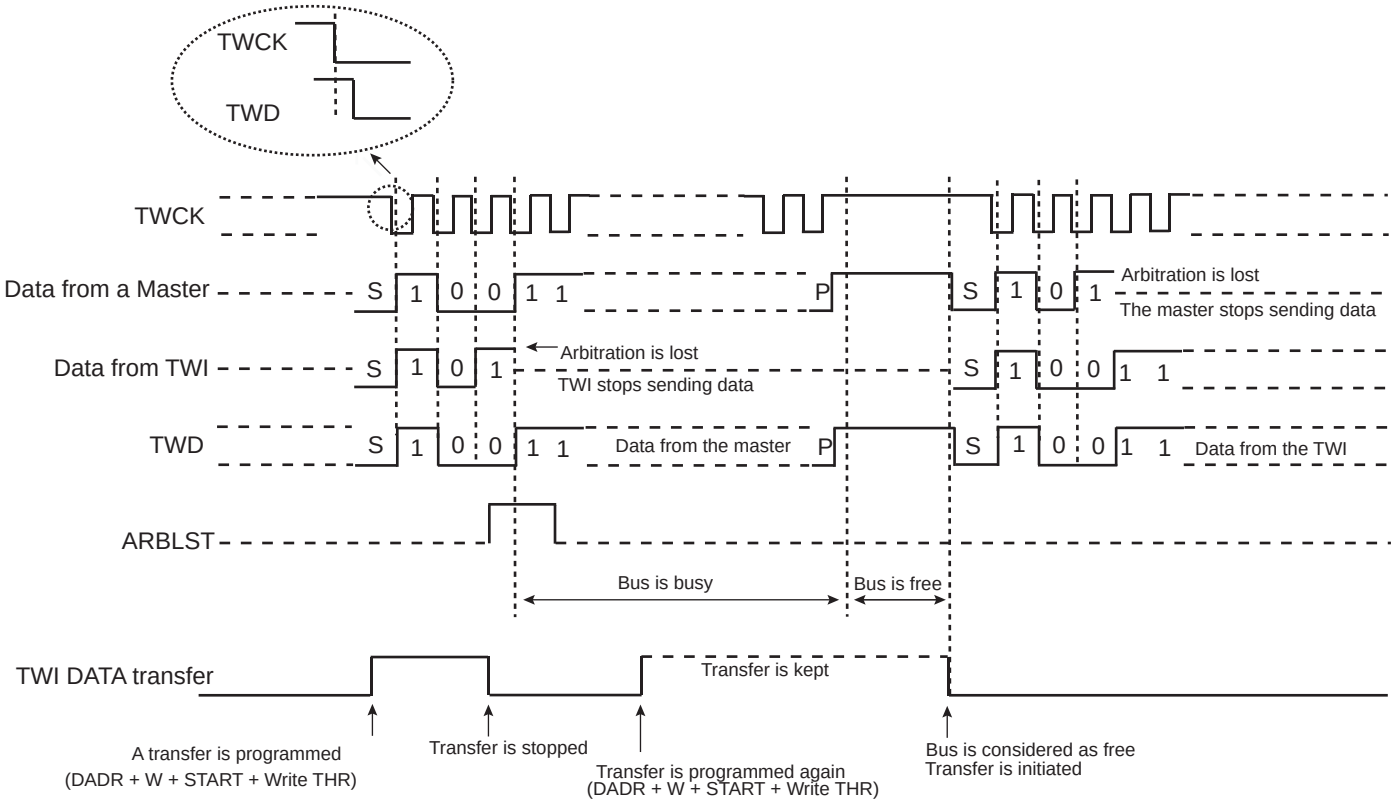
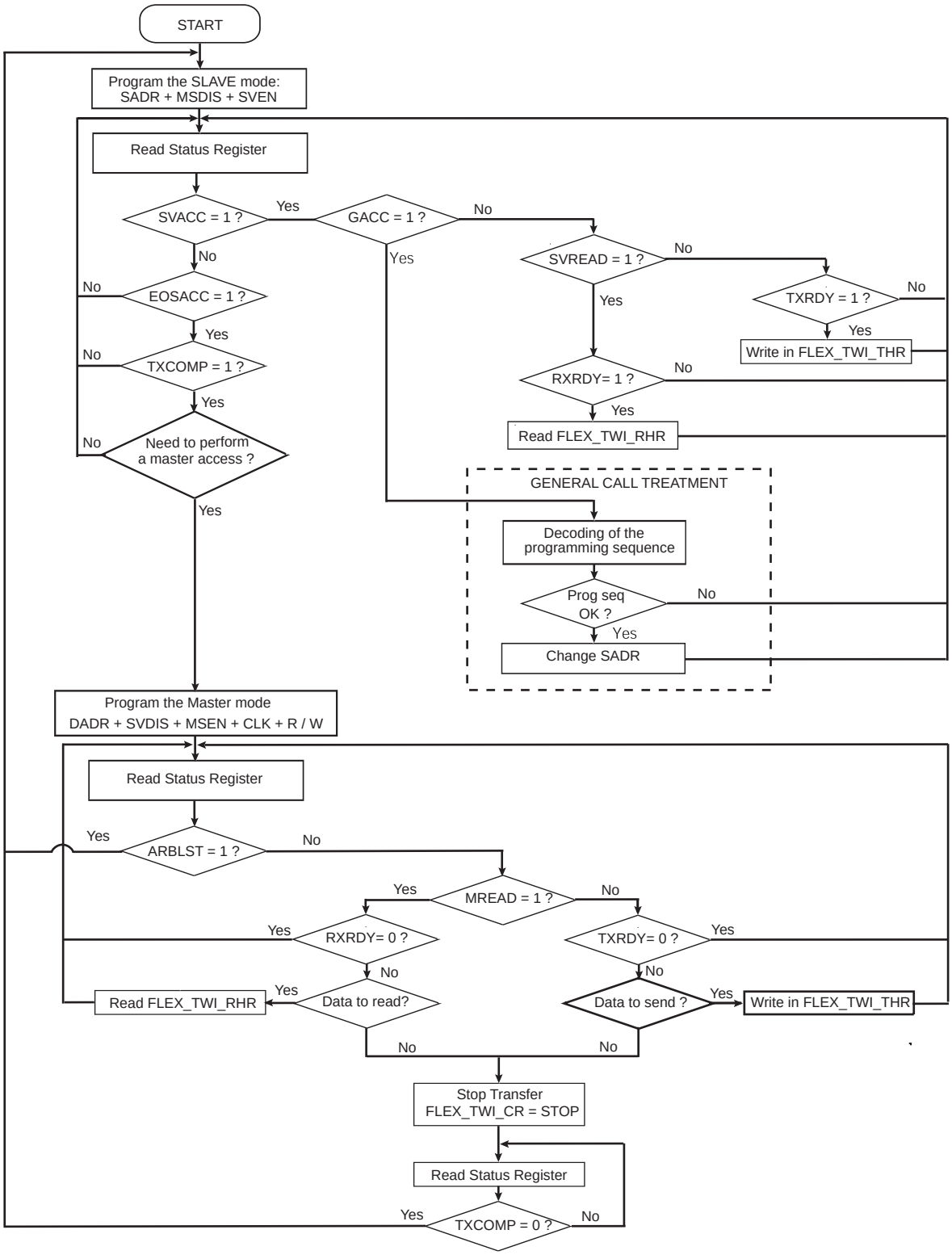


Figure 44-116. Arbitration Cases



The flowchart shown in Figure 44-117 gives an example of read and write operations in Multi-Master mode.

Figure 44-117. Multi-Master Flowchart



44.9.5 Slave Modes

44.9.5.1 Definition

Slave mode is defined as a mode where the device receives the clock and the address from another device called the master.

In this mode, the device never initiates and never completes the transmission (START, REPEATED_START and STOP conditions are always provided by the master).

44.9.5.2 Programming Slave Mode

The following fields must be programmed before entering Slave mode:

1. FLEX_TWI_SMR.SADR: The slave device address is used in order to be accessed by master devices in Read or Write mode.
2. (Optional) FLEX_TWI_SMR.MASK can be set to mask some SADR address bits and thus allow multiple address matching.
3. FLEX_TWI_CR.MSDIS: Disables the Master mode.
4. FLEX_TWI_CR.SVEN: Enables the Slave mode.

As the device receives the clock, values written in FLEX_TWI_CWGR are not processed.

44.9.5.3 Receiving Data

After a START or repeated START condition is detected, and if the address sent by the master matches the slave address programmed in the SADR (Slave Address) field, the SVACC (Slave Access) flag is set and SVREAD (Slave Read) indicates the direction of the transfer.

SVACC remains high until a STOP condition or a repeated START is detected. When such a condition is detected, EOSACC (End Of Slave Access) flag is set.

Read Sequence

In the case of a read sequence (SVREAD is high), the TWI transfers data written in FLEX_TWI_THR (TWI Transmit Holding Register) until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the read sequence TXCOMP (Transmission Complete) flag is set and SVACC reset.

As soon as data is written in FLEX_TWI_THR, the TXRDY (Transmit Holding Register Ready) flag is reset, and it is set when the internal shifter is empty and the sent data acknowledged or not. If the data is not acknowledged, the NACK flag is set.

Note that a STOP or a repeated START always follows a NACK.

See Figure 44-118.

Note: To clear the TXRDY flag in Slave mode, in the TW_CR, write bit SVDIS to 1, then write bit SVEN to 1.

Write Sequence

In the case of a write sequence (SVREAD is low), the RXRDY (Receive Holding Register Ready) flag is set as soon as a character has been received in FLEX_TWI_RHR (TWI Receive Holding Register). RXRDY is reset when reading FLEX_TWI_RHR.

TWI continues receiving data until a STOP condition or a REPEATED_START + an address different from SADR is detected. Note that at the end of the write sequence TXCOMP flag is set and SVACC reset.

See Figure 44-119.

Clock Stretching Sequence

If FLEX_TWI_THR or FLEX_TWI_RHR is not written/read in time, the TWI performs a clock stretching.

Clock stretching information is given by the SCLWS (Clock Wait State) bit.

See Figure 44-121 and Figure 44-122.

Note: Clock stretching can be disabled by configuring the SCLWSDIS bit in FLEX_TWI_SMR. In that case, UNRE and OVRE flags will indicate underrun (when FLEX_TWI_THR is not filled on time) or overrun (when FLEX_TWI_RHR is not read on time).

General Call

In the case where a GENERAL CALL is performed, the GACC (General Call Access) flag is set.

After GACC is set, it is up to the user to interpret the meaning of the GENERAL CALL and to decode the new address programming sequence.

See Figure 44-120.

44.9.5.4 Data Transfer

Read Operation

The Read mode is defined as a data requirement from the master.

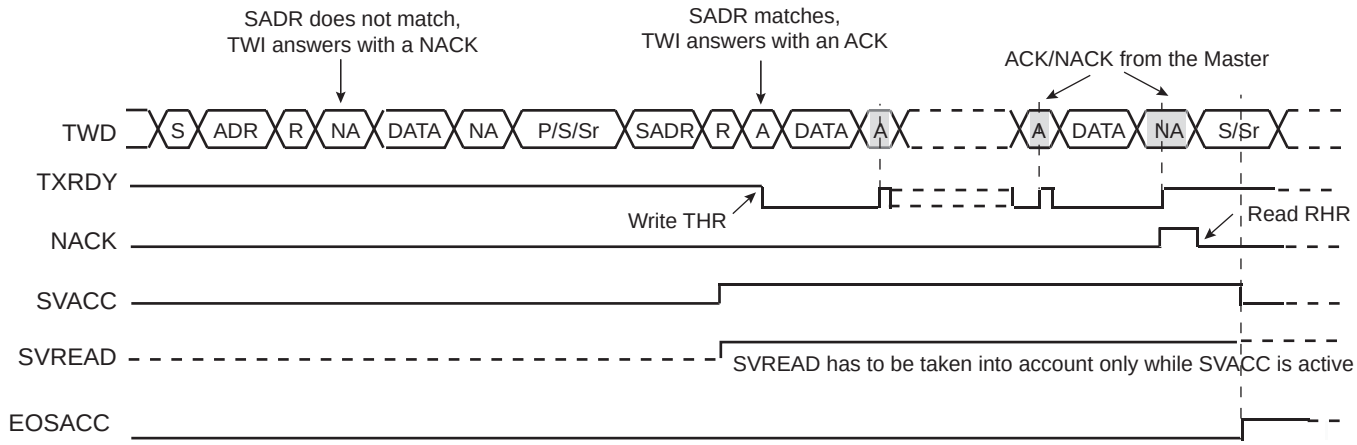
After a START or a REPEATED START condition is detected, the decoding of the address starts. If the slave address (SADR) is decoded, SVACC is set and SVREAD indicates the direction of the transfer.

Until a STOP or REPEATED START condition is detected, TWI continues sending data loaded in FLEX_TWI_THR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

Figure 44-118 describes the read operation.

Figure 44-118. Read Access Ordered by a Master



Notes:
1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. TXRDY is reset when data has been transmitted from FLEX_TWI_THR to the internal shifter and set when this data has been acknowledged or non acknowledged.

Write Operation

The Write mode is defined as a data transmission from the master.

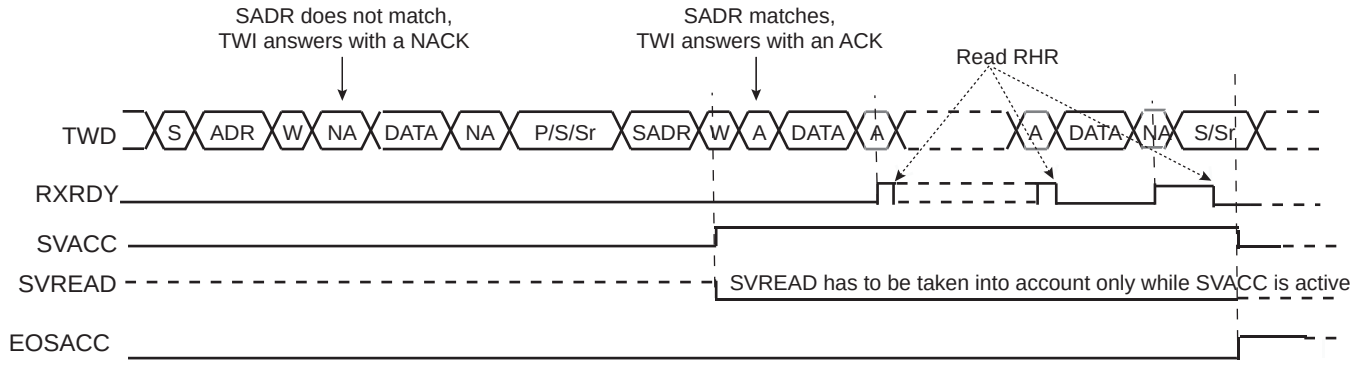
After a START or a REPEATED START, the decoding of the address starts. If the slave address is decoded, SVACC is set and SVREAD indicates the direction of the transfer (SVREAD is low in this case).

Until a STOP or REPEATED START condition is detected, TWI stores the received data in FLEX_TWI_RHR.

If a STOP condition or a REPEATED START + an address different from SADR is detected, SVACC is reset.

Figure 44-119 describes the write operation.

Figure 44-119. Write Access Ordered by a Master



Notes:
1. When SVACC is low, the state of SVREAD becomes irrelevant.
2. RXRDY is set when data has been transmitted from the internal shifter to FLEX_TWI_RHR, and reset when this data is read.

General Call

The general call is performed in order to change the address of the slave.

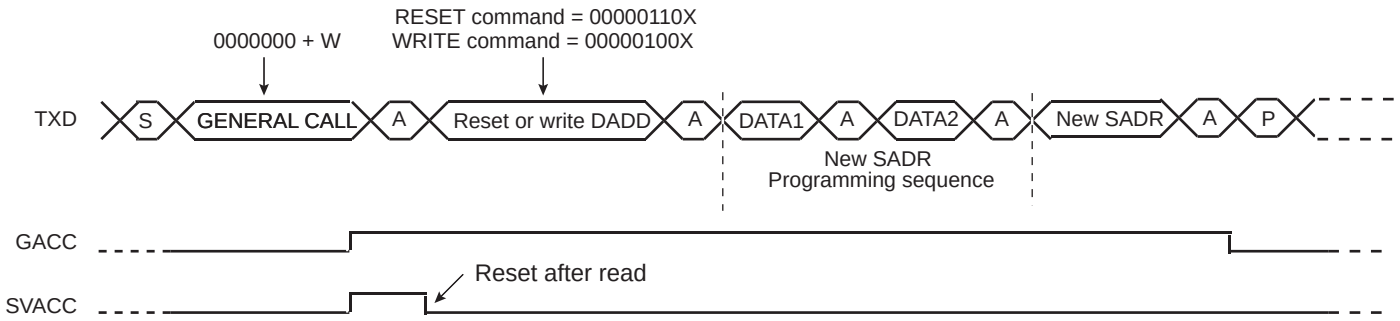
If a GENERAL CALL is detected, GACC is set.

After the detection of general call, it is up to the user to decode the commands which follow.

In case of a WRITE command, the user has to decode the programming sequence and program a new SADR if the programming sequence matches.

Figure 44-120 describes the general call access.

Figure 44-120. Master Performs a General Call



Note: This method allows to create a user-specific programming sequence by choosing the number of programming bytes. The programming sequence has to be provided to the master.

Clock Stretching

In both Read and Write modes, it may happen that FLEX_TWI_THR/FLEX_TWI_RHR buffer is not filled/emptied before the transmission/reception of a new character. In this case, to avoid sending/receiving undesired data, a clock stretching mechanism is implemented.

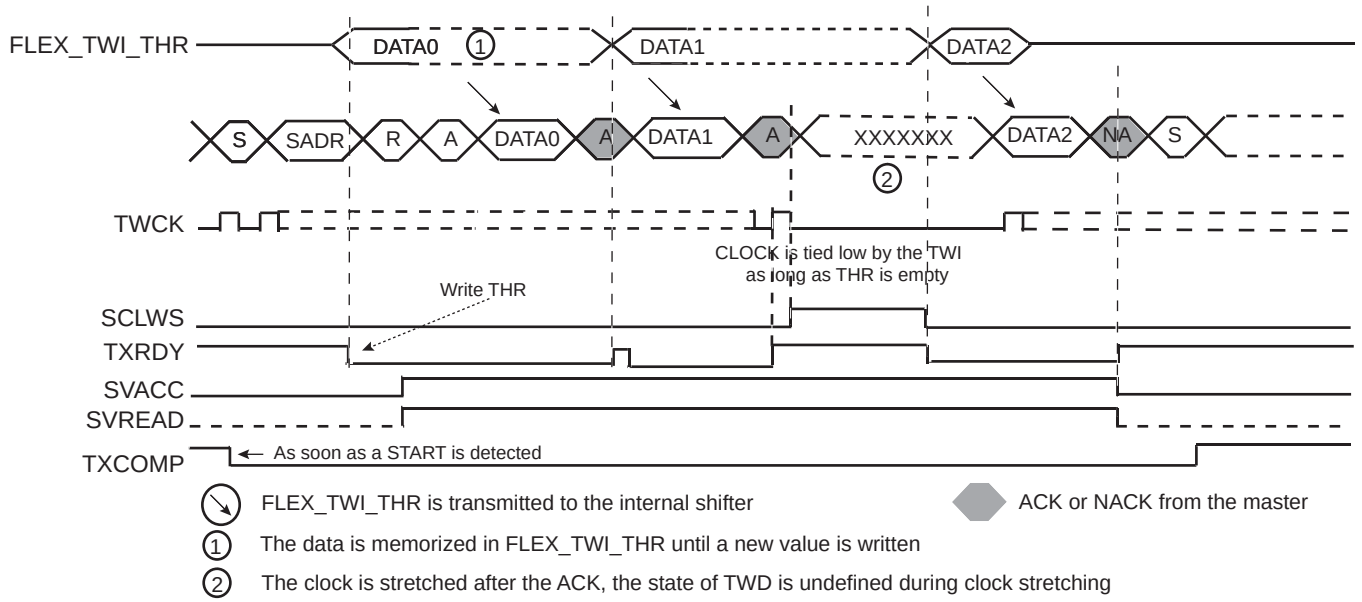
Note: Clock stretching can be disabled by setting the SCLWSDIS bit in FLEX_TWI_SMR. In that case, the UNRE and OVRE flags will indicate underrun (when FLEX_TWI_THR is not filled on time) or overrun (when FLEX_TWI_RHR is not read on time).

Clock Stretching in Read Mode

The clock is tied low if the internal shifter is empty and if a STOP or REPEATED START condition was not detected. It is tied low until the internal shifter is loaded.

Figure 44-121 describes the clock stretching in Read mode.

Figure 44-121. Clock Stretching in Read Mode



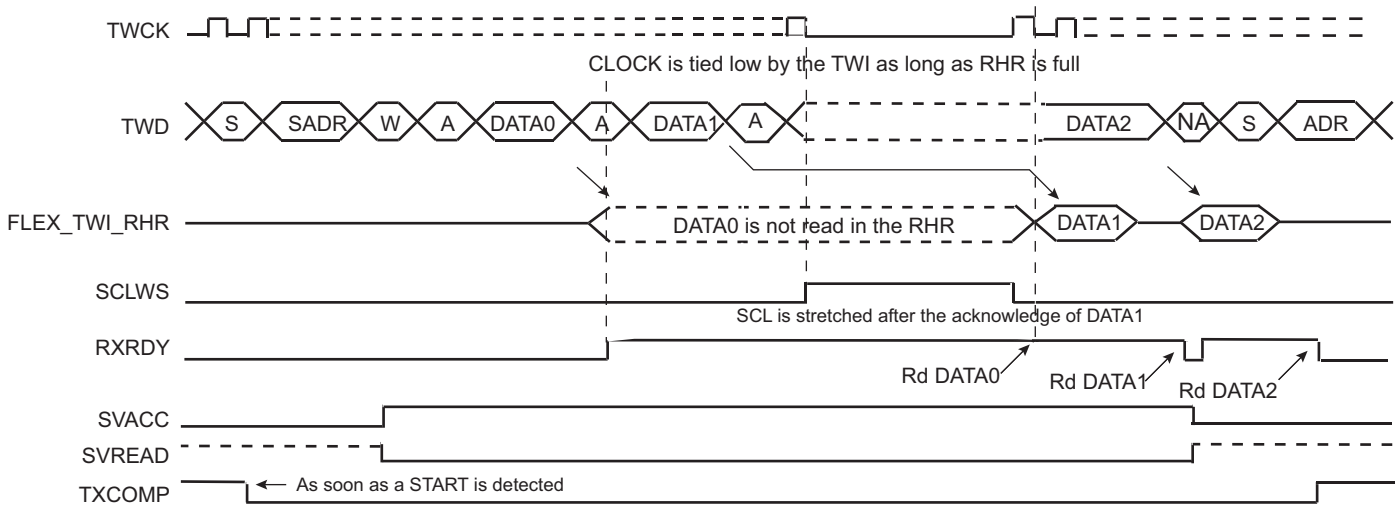
Notes:
1. TXRDY is reset when data has been written in FLEX_TWI_THR to the internal shifter, and set when this data has been acknowledged or non acknowledged.
2. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
3. SCLWS is automatically set when the clock stretching mechanism is started.

Clock Stretching in Write Mode

The clock is tied low if the internal shifter and FLEX_TWI_RHR are full. If a STOP or REPEATED_START condition was not detected, it is tied low until FLEX_TWI_RHR is read.

Figure 44-122 describes the clock stretching in Write mode.

Figure 44-122. Clock Stretching in Write Mode



Notes:
1. At the end of the read sequence, TXCOMP is set after a STOP or after a REPEATED_START + an address different from SADR.
2. SCLWS is automatically set when the clock stretching mechanism is started and automatically reset when the mechanism is finished.

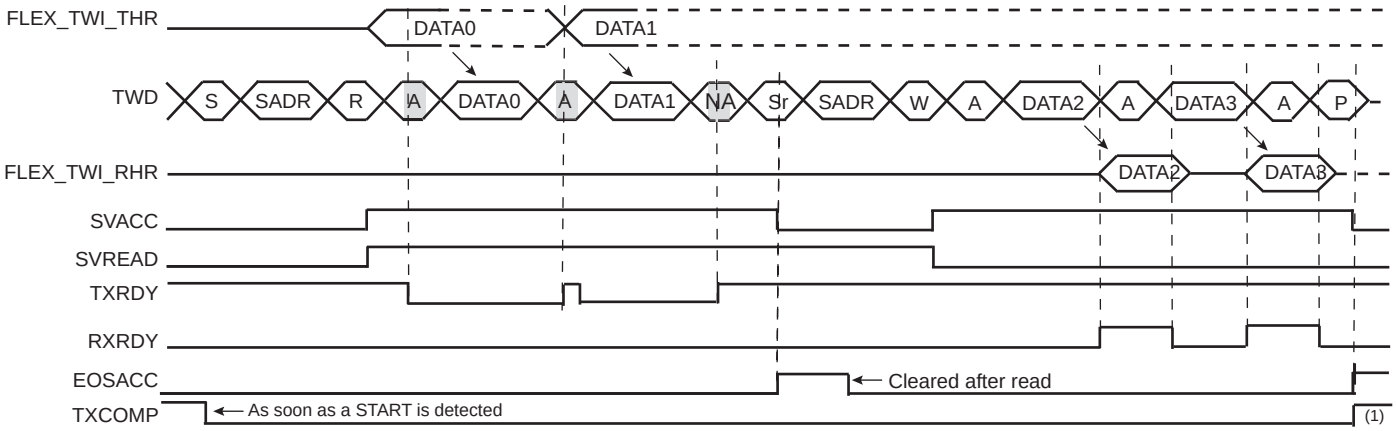
Reversal after a Repeated Start

Reversal of Read to Write

The master initiates the communication by a read command and finishes it by a write command.

Figure 44-123 describes the repeated start and the reversal from Read mode to Write mode.

Figure 44-123. Repeated Start and Reversal from Read Mode to Write Mode

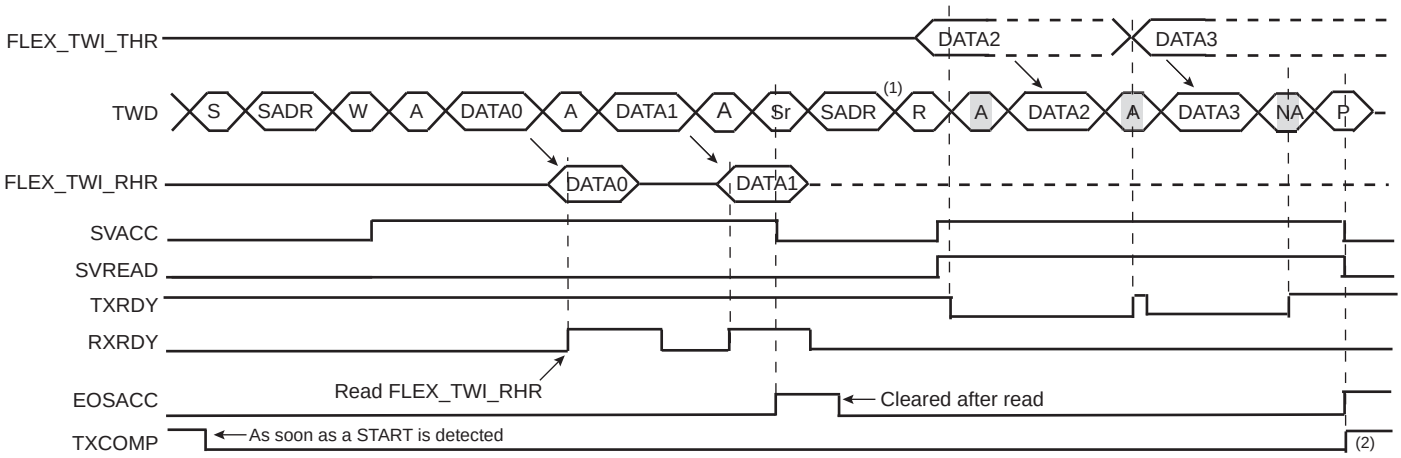


Note:
1. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

Reversal of Write to Read

The master initiates the communication by a write command and finishes it by a read command. Figure 44-124 describes the repeated start and the reversal from Write mode to Read mode.

Figure 44-124. Repeated Start and Reversal from Write Mode to Read Mode



Notes:
1. In this case, if FLEX_TWI_THR has not been written at the end of the read command, the clock is automatically stretched before the ACK.
2. TXCOMP is only set at the end of the transmission because after the repeated start, SADR is detected again.

SMBus Mode

SMBus mode is enabled when SMEN bit is written to one in FLEX_TWI_CR. SMBus mode operation is similar to I²C operation with the following exceptions:

1. Only 7-bit addressing can be used.
2. The SMBus standard describes a set of timeout values to ensure progress and throughput on the bus. These timeout values must be programmed into FLEX_TWI_SMBTR.
3. Transmissions can optionally include a CRC byte, called Packet Error Check (PEC).

4. A set of addresses have been reserved for protocol handling, such as alert response address (ARA) and host header (HH) address. Address matching on these addresses can be enabled by configuring FLEX_TWI_CR appropriately.

Packet Error Checking

Each SMBus transfer can optionally end with a CRC byte, called the PEC byte. Writing the PECEN bit in FLEX_TWI_CR to one will send/check the FLEX_TWI_ACR.PEC field in the current transfer. The PEC generator is always updated on every bit transmitted or received, so that PEC handling on following linked transfers will be correct.

In Slave Receiver mode, the master calculates a PEC value and transmits it to the slave after all data bytes have been transmitted. Upon reception of this PEC byte, the slave will compare it to the PEC value it has computed itself. If the values match, the data was received correctly, and the slave will return an ACK to the master. If the PEC values differ, data was corrupted, and the slave will return a NACK value. The PECERR bit in FLEX_TWI_SR is set automatically if a PEC error occurred.

In Slave Transmitter mode, the slave calculates a PEC value and transmits it to the master after all data bytes have been transmitted. Upon reception of this PEC byte, the master will compare it to the PEC value it has computed itself. If the values match, the data was received correctly. If the PEC values differ, data was corrupted, and the master must take appropriate action.

See Section 44.9.5.8 for detailed flowcharts.

Timeouts

The TWI SMBus Timing Register (FLEX_TWI_SMBTR) configures the SMBus timeout values. If a timeout occurs, the slave will leave the bus. The TOUT bit is also set in FLEX_TWI_SR.

44.9.5.5 High-Speed Slave Mode

High-speed mode is enabled when the HSEN bit is written to one in FLEX_TWI_CR. Furthermore, the analog pad filter must be enabled, the PADFEN bit must be written to one in FLEX_TWI_FILTR and the FILT bit must be cleared. TWI High-speed mode operation is similar to TWI operation with the following exceptions:

1. A master code is received first at normal speed before entering High-speed mode period.
2. When TWI High-speed mode is active, clock stretching is only allowed after acknowledge (ACK), not-acknowledge (NACK), START (S) or repeated START (Sr) (asa consequence, OVF may happen).

TWI High-speed mode allows transfers of up to 3.4 Mbit/s.

The TWI slave in High-speed mode requires that the peripheral clock runs at a minimum of 14 MHz if slave clock stretching is enabled (SCLWSDIS bit at ‘0’). If slave clock stretching is disabled (SCLWSDIS bit at ‘1’), the peripheral clock must run at a minimum of 11 MHz (assuming the system has no latency).

Notes:
1. When slave clock stretching is disabled, FLEX_TWI_RHR must always be read before receiving the next data (MASTER write frame). It is strongly recommended to use either the polling method on the RXRDY flag in FLEX_TWI_SR, or the DMA. If the receive is managed by an interrupt, the TWI interrupt priority must be set to the right level and its latency minimized to avoid receive overrun.
2. When slave clock stretching is disabled, FLEX_TWI_THR must be filled with the first data to send before the beginning of the frame (MASTER read frame). It is strongly recommended to use either the polling method on the TXRDY flag in FLEX_TWI_SR, or the DMA. If the transmit is managed by an interrupt, the TWI interrupt priority must be set to the right level and its latency minimized to avoid transmit underrun.

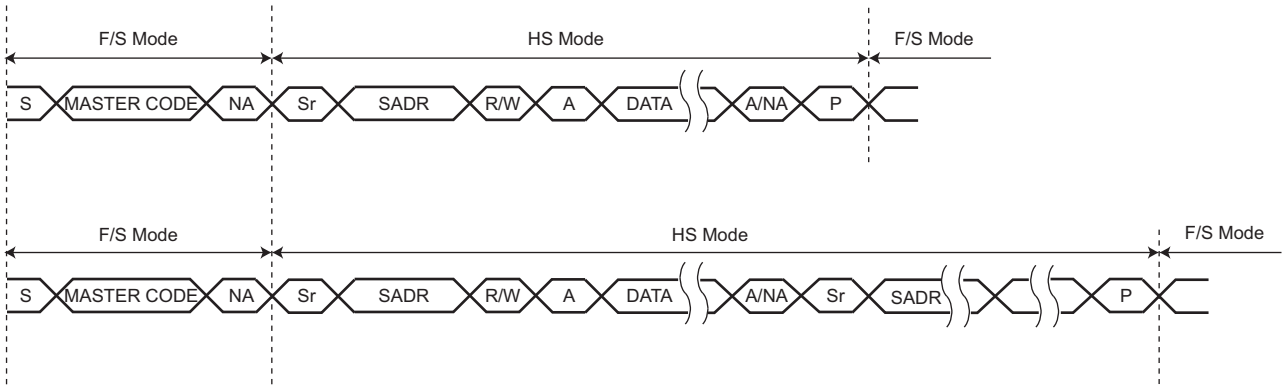
Read/Write Operation

A TWI high-speed frame always begins with the following sequence:

1. START condition (S)
2. Master Code (0000 1XXX)
3. Not-acknowledge (NACK)

When the TWI is programmed in Slave mode and TWI High-speed mode is activated, master code matching is activated and internal timings are set to match the TWI High-speed mode requirements.

Figure 44-125. High-Speed Mode Read/Write



Usage

TWI High-speed mode usage is the same as the standard TWI (see Section 44.9.3.13).

44.9.5.6 Alternative Command

In Slave mode, the Alternative Command mode is used when the SMBus mode is enabled to send or check the PEC byte.

The Alternative Command mode is enabled by setting the ACMEN bit of the TWIHS Control Register, and the transfer is configured in TWIHS_ACR.

For a combined transfer with PEC, only the NPEC bit in TWIHS_ACR must be set as the PEC byte is sent once at the end of the frame.

See Section 44.9.5.8 “Slave Read Write Flowcharts” for detailed flowcharts.

44.9.5.7 TWI Asynchronous and Partial Wakeup (SleepWalking)

The TWI module includes an asynchronous start condition detector, it is capable of waking the device up from a Sleep mode upon an address match (and optionally an additional data match), including Sleep modes where the TWI peripheral clock is stopped.

The FLEX_TWI_RHR register must be read prior to enable the asynchronous and partial wakeup.

After detecting the START condition on the bus, the TWI will stretch TWCK until the TWI peripheral clock has started. The time required for starting the TWI peripheral depends on which Sleep mode the device is in. After the TWI peripheral clock has started, the TWI releases its TWCK stretching and receives one byte of data (slave address) on the bus. At this time, only a limited part of the device, including the TWI module, receives a clock, thus saving power. If the address phase causes a TWIS address match (and optionally if the first data byte causes data match as well), the entire device is wakened and normal TWI address matching actions are performed. Normal TWI transfer then follows. If the TWI module is not addressed (or if the optional data match fails), the TWI peripheral clock is automatically stopped and the device returns to its original Sleep mode.

The TWI module has the capability to match on more than one address. The SADR1EN, SADR2EN and SADR3EN bits in FLEX_TWI_SMR allow to enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in FLEX_TWI_SWMR. The SleepWalking matching process can be extended to the first received data byte if DATAMEN bit in FLEX_TWI_SMR is set, in that case a complete matching includes address matching and first received data matching. The DATAM field in FLEX_TWI_SWMR allows to configure the data to match on the first received byte.

When the system is in Active mode and the TWI enters asynchronous partial Wakeup mode, the flag SVACC must be programmed as the unique source of the TWI interrupt and the data match comparison must be disabled.

When the system exits Wait mode as the result of a matching condition, the SVACC flag is used to determine if the TWI is the source of the exit from Wait mode.

Figure 44-126. Address Match and Data Matching Disabled

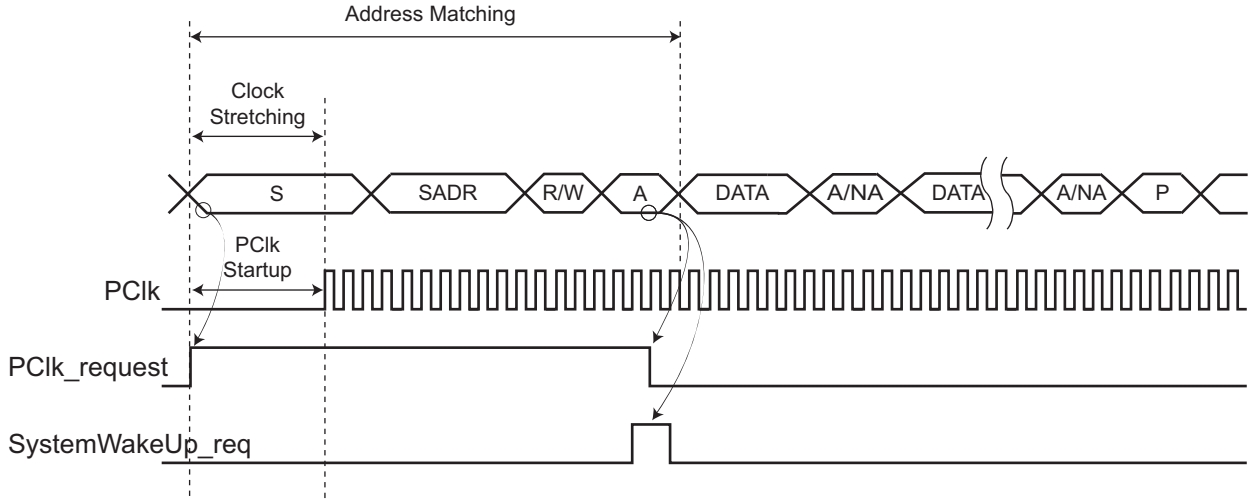


Figure 44-127. Address Does Not Match and Data Matching Disabled

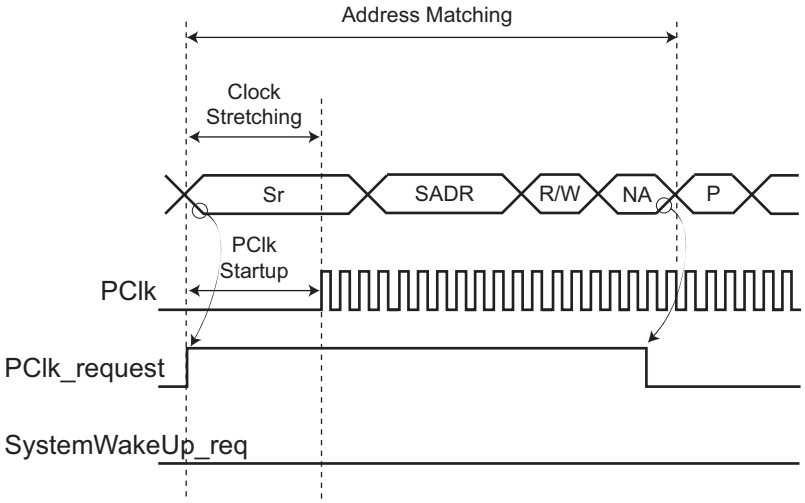


Figure 44-128. Address and Data Match (Data Matching Enabled)

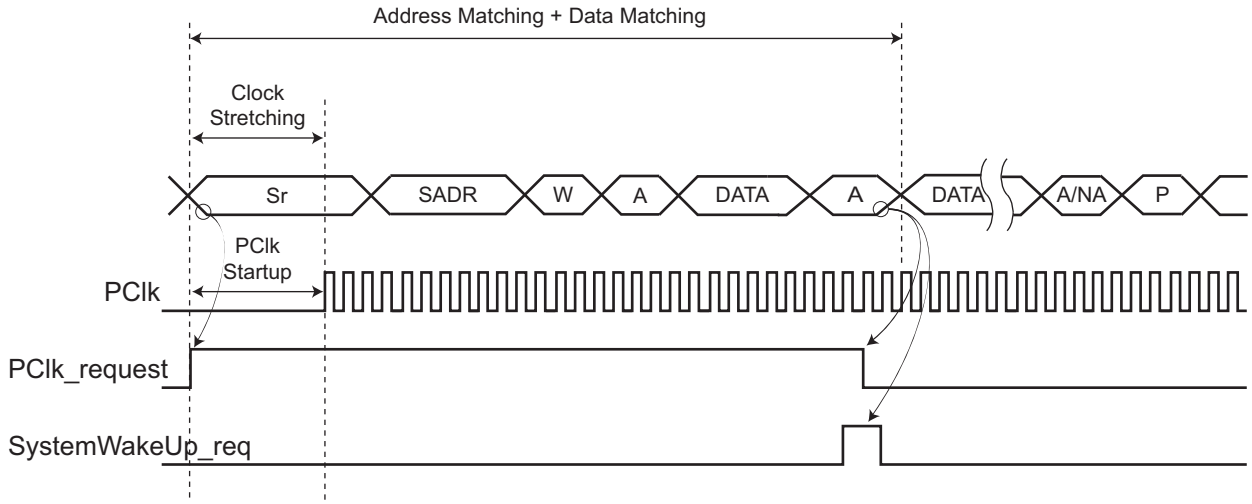
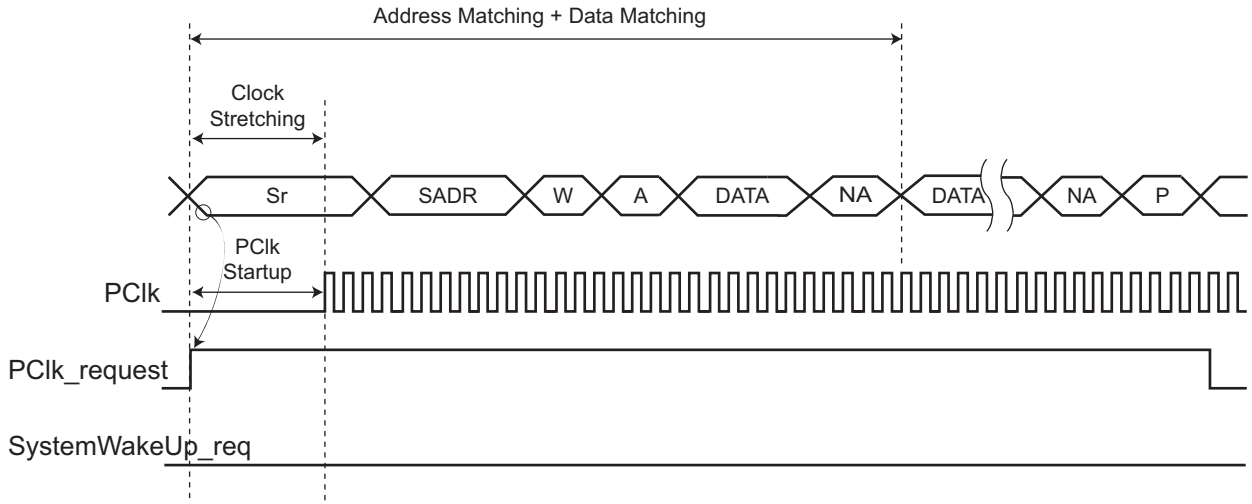


Figure 44-129. Address Matches and Data Do Not Match (Data Matching Enabled)



44.9.5.8 Slave Read Write Flowcharts

The flowchart shown in Figure 44-130 gives an example of read and write operations in Slave mode. A polling or interrupt method can be used to check the status bits. The interrupt method requires that the Interrupt Enable Register (FLEX_TWI_IER) be configured first.

Figure 44-130. Read Write Flowchart in Slave Mode

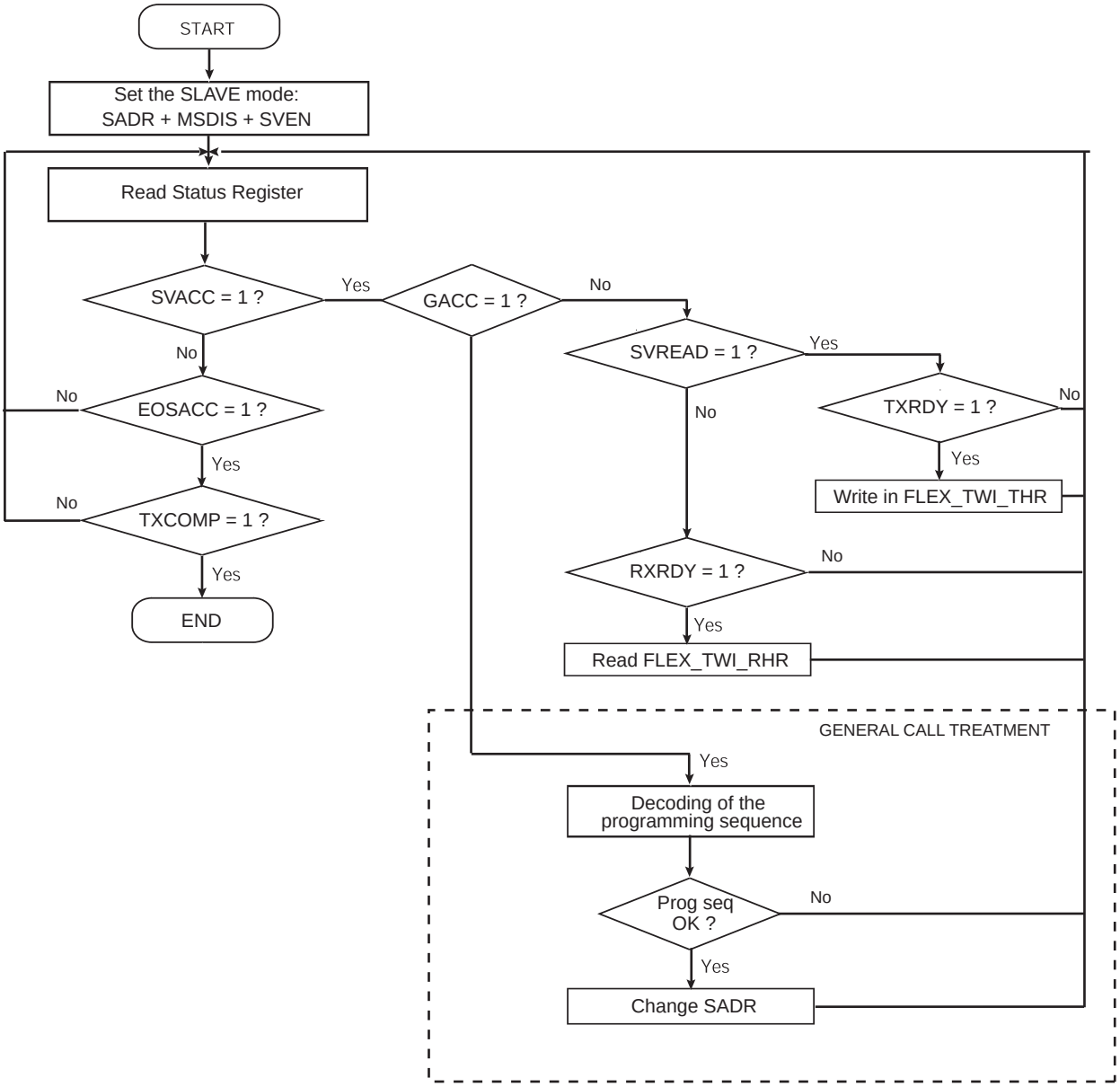


Figure 44-131. Read Write Flowchart in Slave Mode with SMBus PEC

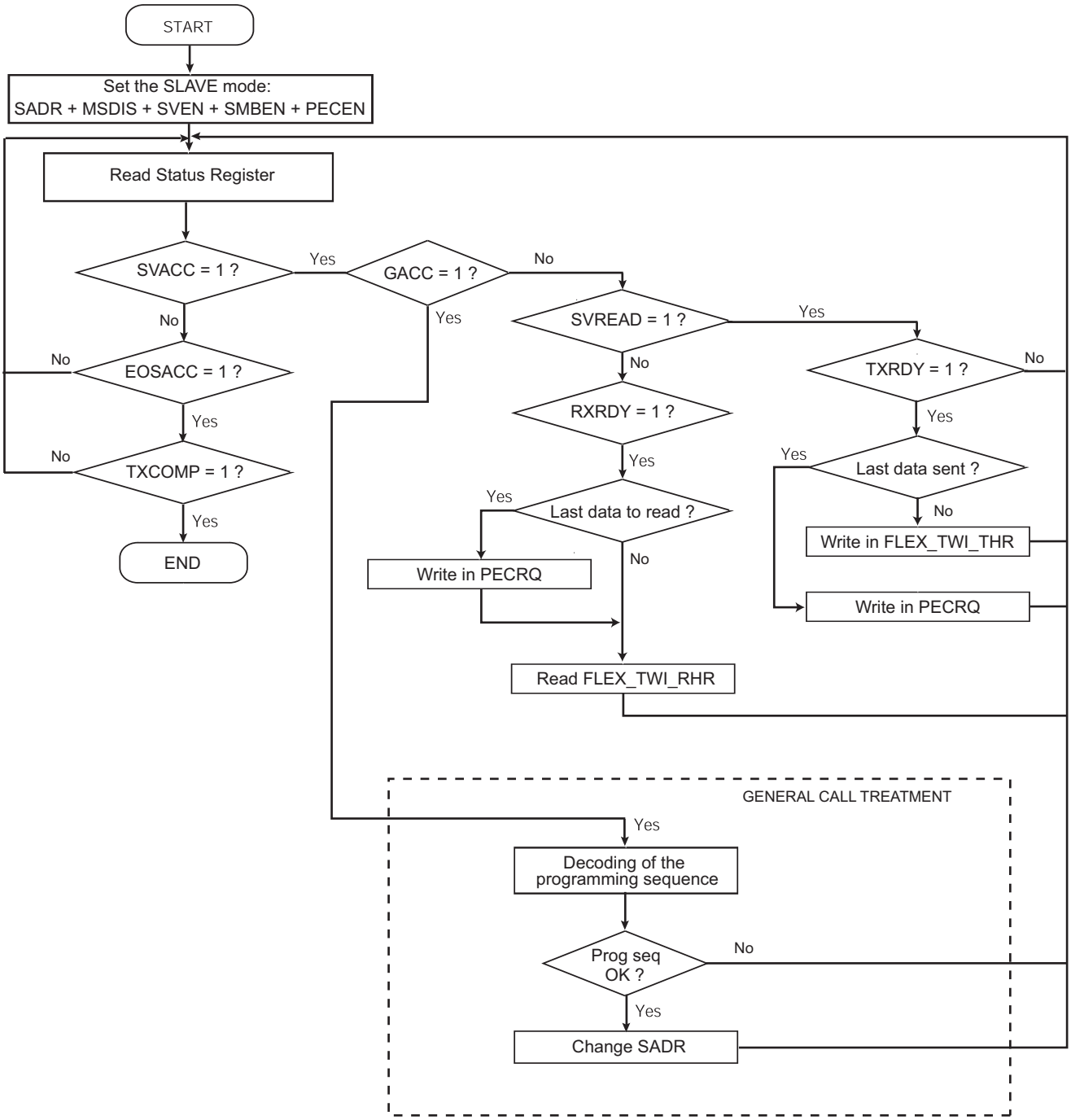
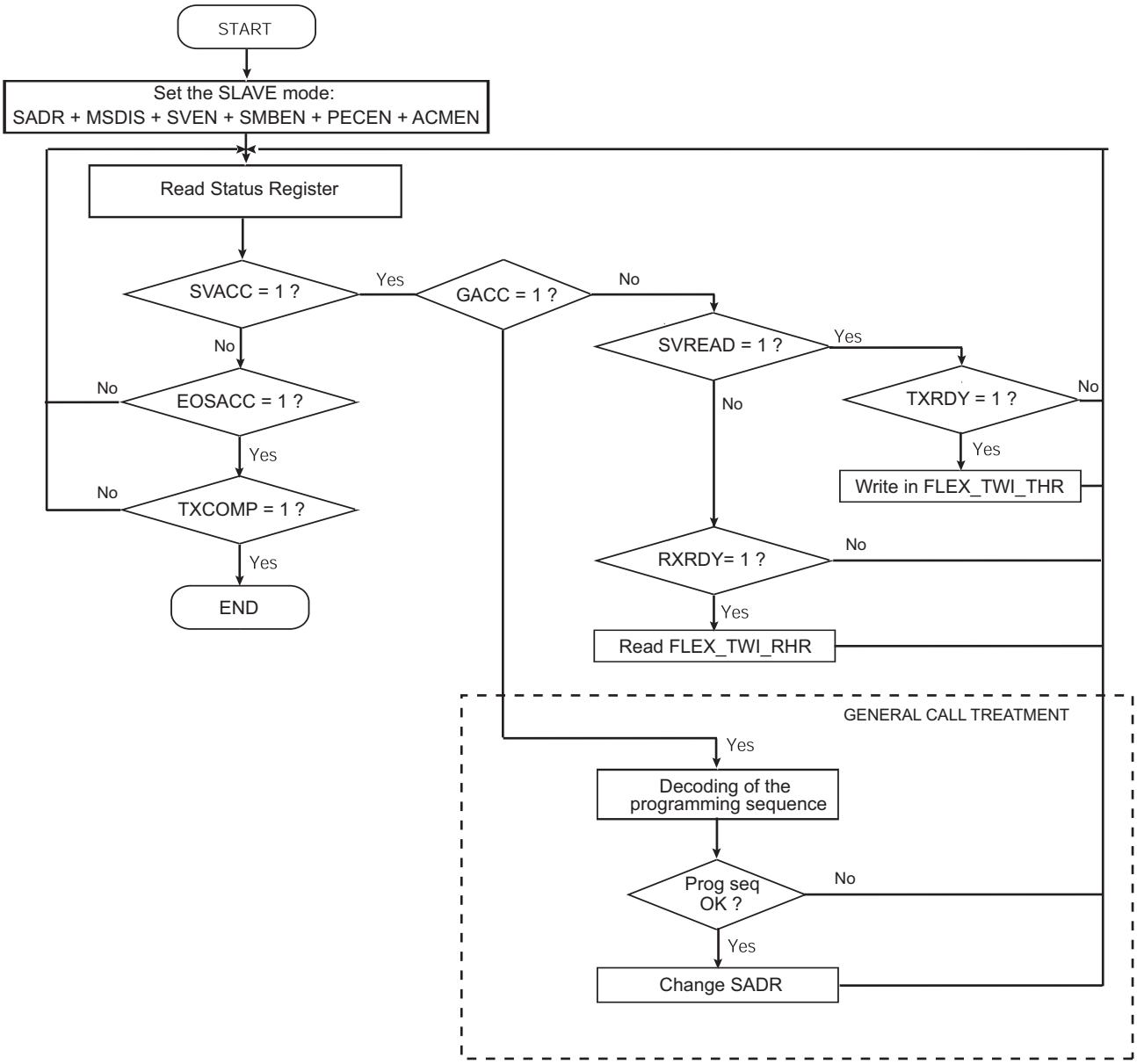


Figure 44-132. Read Write Flowchart in Slave Mode with SMBus PEC and Alternative Command Mode

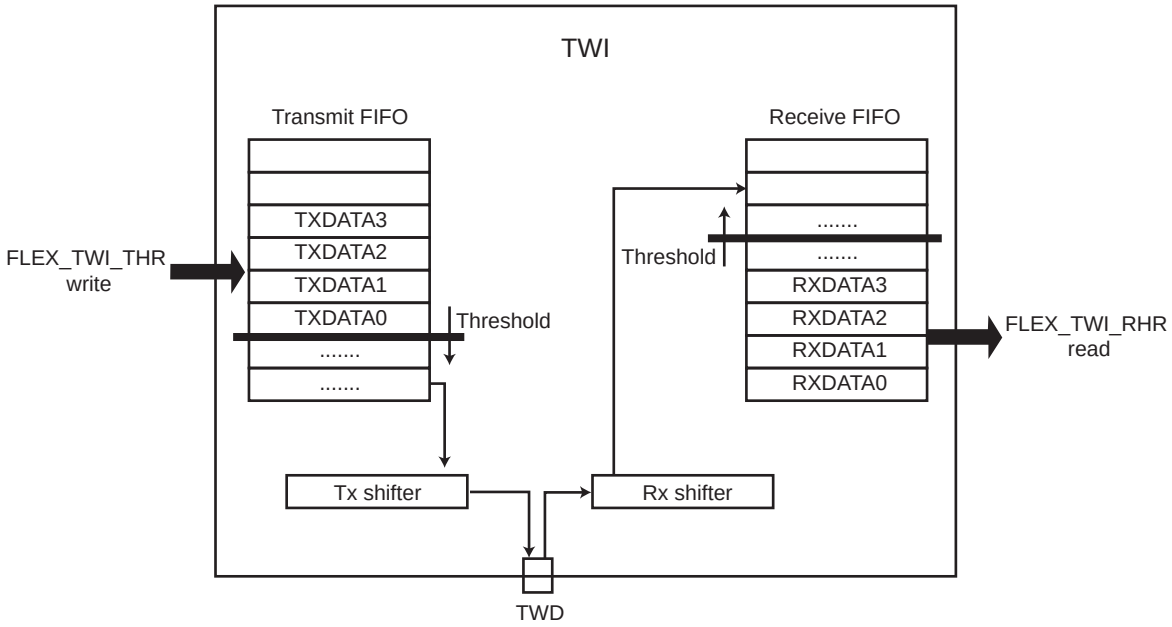


44.9.5.9 FIFOs

The TWI includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in FLEX_TWI_CR. It is recommended to disable both Master and Slave modes before enabling or disabling the FIFO (MSDIS and SVDIS bit in FLEX_TWI_CR).

Writing the FIFOEN bit to ‘1’ will enable a 16-data Transmit FIFO and a 16-data Receive FIFO.

Figure 44-133. FIFOs Block Diagram



Sending/Receiving Data with FIFO Enabled

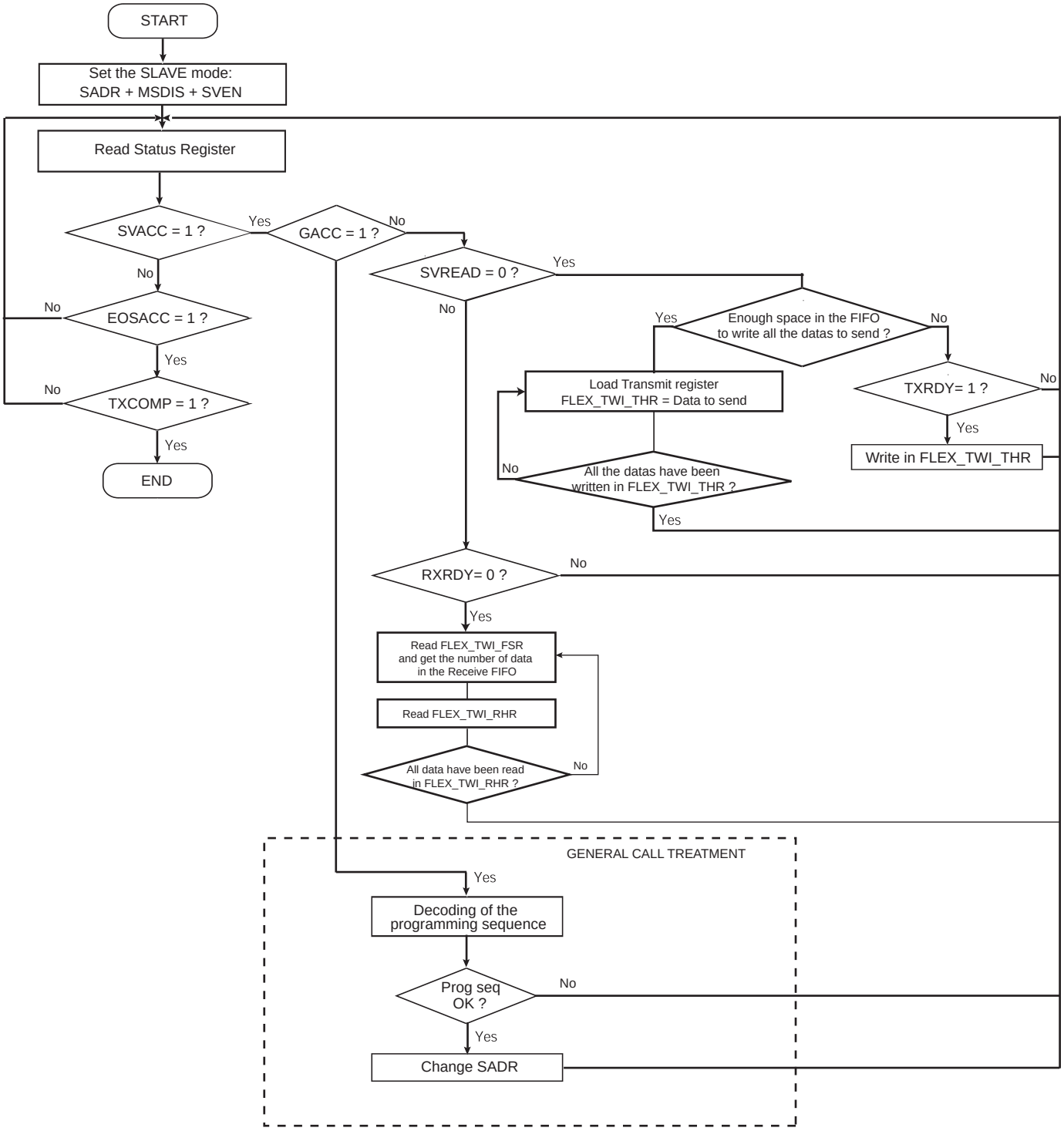
With the Transmit FIFO enabled, any write access to FLEX_TWI_THR will bring the written data to the Transmit FIFO. As a consequence, it is not mandatory any more to monitor the TXRDY flag state to send multiple data without DMAC.

Knowing the number of data to send and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in FLEX_TWI_THR without checking the TXRDY flag between each access. The Transmit FIFO state can be checked reading the TXFL field in FLEX_TWI_FLR.

With Receive FIFO enabled, any read access on FLEX_TWI_RHR will pull out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RXRDY flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RXRDY flag set to ‘1’), the exact number of data can be checked with the RXFL field in FLEX_TWI_FLR and all the data read successively in FLEX_TWI_RHR without checking the RXRDY flag between each access.

Figure 44-134. Sending/Receiving Data with FIFO Flowchart



Clearing/Flushing FIFOs

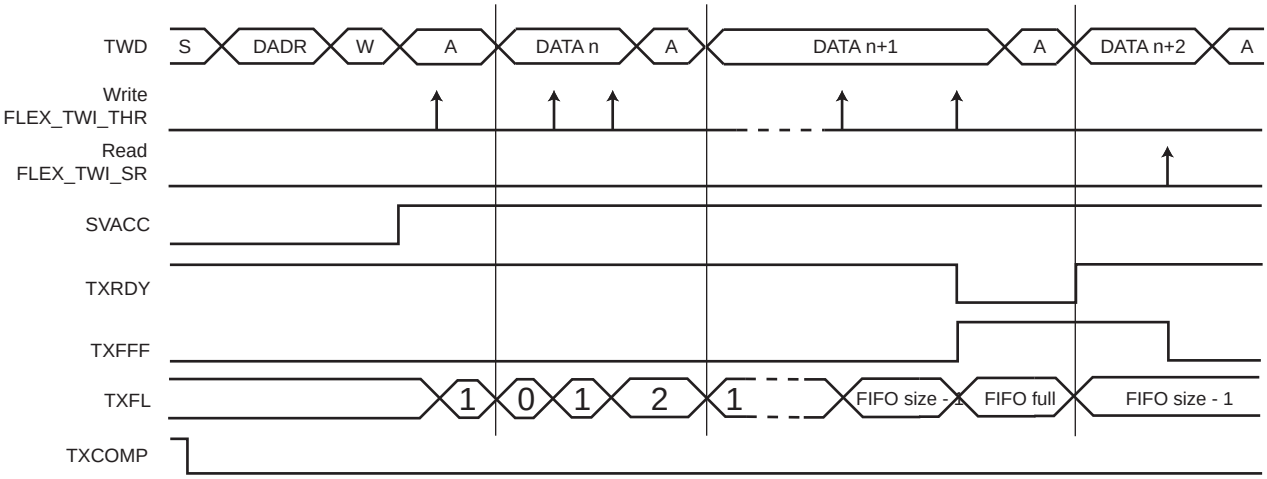
Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in FLEX_TWI_CR.

TXRDY and RXRDY Behavior

If FIFOs are enabled, the behavior of the TXRDY and RXRDY flags will be slightly different.

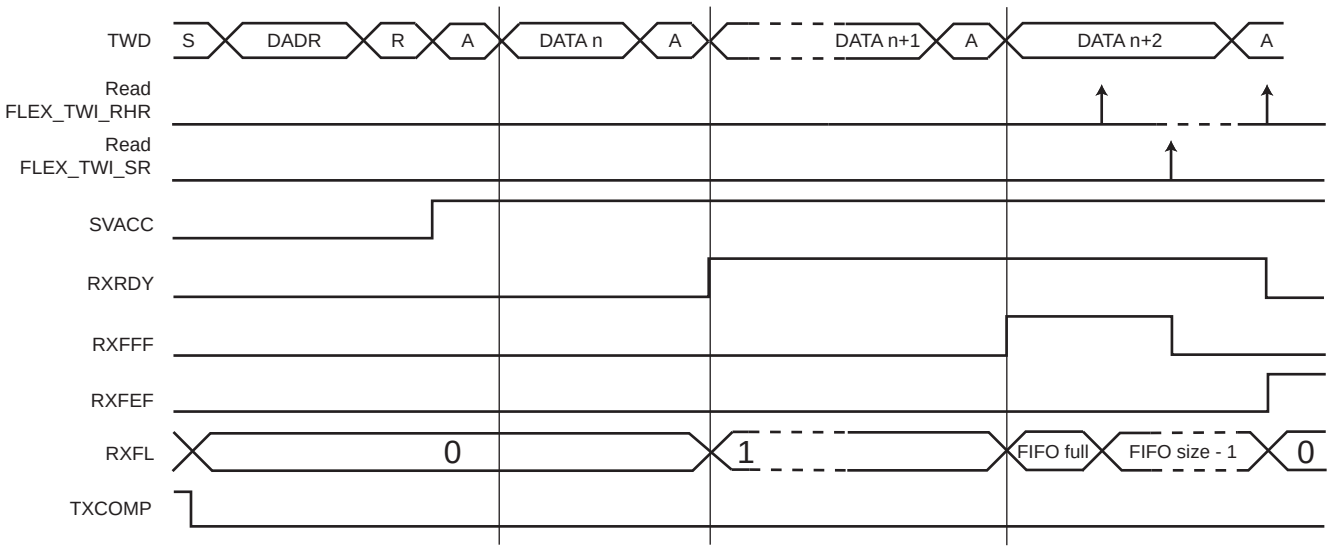
TXRDY will indicate if a data can be written in the Transmit FIFO. By default, the TXRDY flag will then stay at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 44-135. TXRDY in Single Data Mode and TXRDYM = 0x0



RXRDY will indicate if an unread data is present in the Receive FIFO. By default, the RXRDY flag will then be at level ‘1’ as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 44-136. RXRDY in Single Data Mode and RXRDYM = 0x0



TXRDY and RXRDY behavior can be modified using the TXRDYM and RXRDYM fields in FLEX_TWI_FMR. In Single Data Mode, there is no need to modify the TXRDY and RXRDY behavior; however, it may be useful in Multiple Data mode.

Slave Multiple Data Mode

When the FIFOs are enabled, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write/read up to four data in one FLEX_TWI_THR/FLEX_TWI_RHR register access.

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data will be written/read; if the access is a halfword-size register access, then two data will be written/read and finally, if the access is a word-size register access, then four data will be written/read.

Written/Read data are always right-aligned, as shown in Section 44.10.70 “TWI Receive Holding Register (FIFO Enabled)” and Section 44.10.72 “TWI Transmit Holding Register (FIFO Enabled)”.

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six FLEX_TWI_THR-byte write accesses.
- Perform three FLEX_TWI_THR-halfword write accesses.
- Perform one FLEX_TWI_THR-word write access and one FLEX_TWI_THR-halfword write access.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six FLEX_TWI_RHR-byte read accesses.
- Perform three FLEX_TWI_RHR-halfword read accesses.
- Perform one FLEX_TWI_RHR-word read access and one FLEX_TWI_RHR-halfword read access.

Multiple Data mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

- TXRDY and RXRDY configuration

  In Multiple Data mode, the TXRDYM and RXRDYM fields in FLEX_TWI_FMR become useful.

  As in Multiple Data mode, it is possible to write several data in the same access; it might be useful to configure the TXRDY flag behavior to indicate if 1, 2 or 4 data can be written in the FIFO depending on the access to perform on FLEX_TWI_THR.

  If, for instance, four data are written each time in FLEX_TWI_THR it might be useful to configure the TXRDYM field to 0x2 value so that the TXRDY flag will be at ‘1’ only when at least four data can be written in the Transmit FIFO.

  In the same way, if four data are read each time in FLEX_TWI_RHR, it might be useful to configure the RXRDYM field to 0x2 value so that the RXRDY flag will be at ‘1’ only when at least four unread data are in the Receive FIFO.

- DMAC

  If DMAC transfer is used, it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

Transmit FIFO Lock

If a frame is terminated early due to a not-acknowledge error (NACK flag), SMBus timeout error (TOUT flag) or master code acknowledge error (MACK flag), a lock is set on the Transmit FIFO preventing any new frame from being sent until it is cleared. This allows clearing the FIFO if needed, reset DMAC channels, etc., without any risk.

The LOCK bit in FLEX_TWI_SR is used to check the state of the Transmit FIFO lock.

The Transmit FIFO lock can be cleared by setting the TXFLCLR bit to ‘1’ in FLEX_TWI_CR.

FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

  If the Transmit FIFO is full and a write access is performed on FLEX_TWI_THR it will generate a Transmit FIFO pointer error and set the TXFPTEF flag in FLEX_TWI_FSR.

  In Multiple Data mode, if the number of data written in FLEX_TWI_THR (according to the register access size) is bigger than the Transmit FIFO free space, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in FLEX_TWI_FSR.

- Receive FIFO:

  In Multiple Data mode, if the number of data read in FLEX_TWI_RHR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it generates a Receive FIFO pointer error and sets the RXFPTEF flag in FLEX_TWI_FSR.

  Pointer error should not happen if the FIFO state is checked before writing/reading in the FLEX_TWI_THR/FLEX_TWI_RHR registers. The FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a Transmit or Receive pointer error occurs, a software reset must be performed using the SWRST bit in FLEX_TWI_CR. Note that issuing a software while transmitting might leave a slave in an unknown state holding the TWD line. In such case, a Bus Clear Command will allow to make the slave release the TWD line (the first frame sent afterwards might not be received properly by the slave).

FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO:

  The Transmit FIFO threshold can be set using the TXFTHRES field in FLEX_TWI_FMR. Each time the Transmit FIFO goes from the ‘above threshold’ to the ‘equal or below threshold’ state, the TXFTHF flag in FLEX_TWI_FSR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.

- Receive FIFO:

  The Receive FIFO threshold can be set using the RXFTHRES field in FLEX_TWI_FMR. Each time the Receive FIFO goes from the ‘below threshold’ to the ‘equal to or above threshold’ state, the RXFTHF flag in FLEX_TWI_FSR is set. This enables the application to know that the Receive FIFO reached the defined threshold and to read some data before it becomes full.

The TXFTHF and RXFTHF flags can be both configured to generate an interrupt using FLEX_TWI_FIER and FLEX_TWI_FIDR.

FIFO Flags

FIFOs come with a set of flags which can be configured each to generate an interrupt through FLEX_TWI_FIER and FLEX_TWI_FIDR.

FIFO flags state can be read in FLEX_TWI_FSR. They are cleared on FLEX_TWI_FSR read.

44.9.6 TWI Comparison Function on Received Character

The comparison function differs if the asynchronous partial wakeup (Sleepwalking) is enabled or not.

If asynchronous partial wakeup is disabled (see PMC section), the TWI has the capability to extend the address matching on up to three slave addresses. The SADR1EN, SADR2EN and SADR3EN bits in FLEX_TWI_SMR enable address matching on additional addresses which can be configured through SADR1, SADR2 and SADR3 fields in FLEX_TWI_SWMR. The DATAMEN bit had no effect.

The SVACC bit is set when there is a comparison match with the received slave address.

44.9.7 TWI Register Write Protection

The FLEXCOM operating mode (FLEX_MR.OPMODE) must be set to FLEX_MR_OPMODE_TWI to enable access to the write protection registers.

To prevent any single software error from corrupting TWI behavior, certain registers in the address space can be write-protected by setting the WPEN (Write Protection Enable) bit in the TWI Write Protection Mode Register (FLEX_TWI_WPMR).

If a write access to a write-protected register is detected, the Write Protection Violation Status (WPVS) flag in the TWI Write Protection Status Register (FLEX_TWI_WPSR) is set and the Write Protection Violation Source (WPVSRC) field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading FLEX_TWI_WPSR.

The following register(s) can be write-protected when WPEN is set:

- TWI Slave Mode Register
- TWI Clock Waveform Generator Register
- TWI SMBus Timing Register
- TWI SleepWalking Matching Register

44.10 Flexible Serial Communication Unit (FLEXCOM) User Interface

Table 44-18. Register Mapping

Offset	Register	Name	Access	Reset
0x000	FLEXCOM Mode Register	FLEX_MR	Read/Write	0x0
0x004–0x00C	Reserved	–	–	–
0x010	FLEXCOM Receive Holding Register	FLEX_RHR	Read-only	0x0
0x014–0x01C	Reserved	–	–	–
0x020	FLEXCOM Transmit Holding Register	FLEX_THR	Read/Write	0x0
0x024–0x0FC	Reserved	–	–	–
0x100–0x1FC	Reserved	–	–	–
0x200	USART Control Register	FLEX_US_CR	Write-only	–
0x204	USART Mode Register	FLEX_US_MR	Read/Write	–
0x208	USART Interrupt Enable Register	FLEX_US_IER	Write-only	–
0x20C	USART Interrupt Disable Register	FLEX_US_IDR	Write-only	–
0x210	USART Interrupt Mask Register	FLEX_US_IMR	Read-only	0x0
0x214	USART Channel Status Register	FLEX_US_CSR	Read-only	–
0x218	USART Receive Holding Register	FLEX_US_RHR	Read-only	0x0
0x21C	USART Transmit Holding Register	FLEX_US_THR	Write-only	–
0x220	USART Baud Rate Generator Register	FLEX_US_BRGR	Read/Write	0x0
0x224	USART Receiver Timeout Register	FLEX_US_RTOR	Read/Write	0x0
0x228	USART Transmitter Timeguard Register	FLEX_US_TTGR	Read/Write	0x0
0x22C–0x23C	Reserved	–	–	–
0x240	USART FI DI Ratio Register	FLEX_US_FIDI	Read/Write	0x174
0x244	USART Number of Errors Register	FLEX_US_NER	Read-only	–
0x248	Reserved	–	–	–
0x24C	USART IrDA Filter Register	FLEX_US_IF	Read/Write	0x0
0x250	USART Manchester Configuration Register	FLEX_US_MAN	Read/Write	0xB0011004
0x254	USART LIN Mode Register	FLEX_US_LINMR	Read/Write	0x0
0x258	USART LIN Identifier Register	FLEX_US_LINIR	Read/Write(1)	0x0
0x25C	USART LIN Baud Rate Register	FLEX_US_LINBRR	Read-only	0x0
0x260–0x288	Reserved	–	–	–
0x290	USART Comparison Register	FLEX_US_CMPR	Read/Write	0x0
0x2A0	USART FIFO Mode Register	FLEX_US_FMR	Read/Write	0x0
0x2A4	USART FIFO Level Register	FLEX_US_FLR	Read-only	0x0
0x2A8	USART FIFO Interrupt Enable Register	FLEX_US_FIER	Write-only	–
0x2AC	USART FIFO Interrupt Disable Register	FLEX_US_FIDR	Write-only	–
0x2B0	USART FIFO Interrupt Mask Register	FLEX_US_FIMR	Read-only	0x0
0x2B4	USART FIFO Event Status Register	FLEX_US_FESR	Read-only	0x0

Table 44-18. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x2B8–0x2E0	Reserved	–	–	–
0x2E4	USART Write Protection Mode Register	FLEX_US_WPMR	Read/Write	0x0
0x2E8	USART Write Protection Status Register	FLEX_US_WPSR	Read-only	0x0
0x2EC–0x2F8	Reserved	–	–	–
0x300–0x3FC	Reserved	–	–	–
0x400	SPI Control Register	FLEX_SPI_CR	Write-only	–
0x404	SPI Mode Register	FLEX_SPI_MR	Read/Write	0x0
0x408	SPI Receive Data Register	FLEX_SPI_RDR	Read-only	0x0
0x40C	SPI Transmit Data Register	FLEX_SPI_TDR	Write-only	–
0x410	SPI Status Register	FLEX_SPI_SR	Read-only	0x0
0x414	SPI Interrupt Enable Register	FLEX_SPI_IER	Write-only	–
0x418	SPI Interrupt Disable Register	FLEX_SPI_IDR	Write-only	–
0x41C	SPI Interrupt Mask Register	FLEX_SPI_IMR	Read-only	0x0
0x420–0x42C	Reserved	–	–	–
0x430	SPI Chip Select Register 0	FLEX_SPI_CSR0	Read/Write	0x0
0x434	SPI Chip Select Register 1	FLEX_SPI_CSR1	Read/Write	0x0
0x438–0x43C	Reserved	–	–	–
0x440	SPI FIFO Mode Register	FLEX_SPI_FMR	Read/Write	0x0
0x444	SPI FIFO Level Register	FLEX_SPI_FLR	Read-only	0x0
0x448	SPI Comparison Register	FLEX_SPI_CMPR	Read/Write	0x0
0x44C–0x4E0	Reserved	–	–	–
0x4E4	SPI Write Protection Mode Register	FLEX_SPI_WPMR	Read/Write	0x0
0x4E8	SPI Write Protection Status Register	FLEX_SPI_WPSR	Read-only	0x0
0x4EC–0x4F8	Reserved	–	–	–
0x500–0x5FC	Reserved	–	–	–
0x600	TWI Control Register	FLEX_TWI_CR	Write-only	–
0x604	TWI Master Mode Register	FLEX_TWI_MMR	Read/Write	0x00000000
0x608	TWI Slave Mode Register	FLEX_TWI_SMR	Read/Write	0x00000000
0x60C	TWI Internal Address Register	FLEX_TWI_IADR	Read/Write	0x00000000
0x610	TWI Clock Waveform Generator Register	FLEX_TWI_CWGR	Read/Write	0x00000000
0x614–0x61C	Reserved	–	–	–
0x620	TWI Status Register	FLEX_TWI_SR	Read-only	0x0300F009
0x624	TWI Interrupt Enable Register	FLEX_TWI_IER	Write-only	–
0x628	TWI Interrupt Disable Register	FLEX_TWI_IDR	Write-only	–
0x62C	TWI Interrupt Mask Register	FLEX_TWI_IMR	Read-only	0x00000000
0x630	TWI Receive Holding Register	FLEX_TWI_RHR	Read-only	0x00000000
0x634	TWI Transmit Holding Register	FLEX_TWI_THR	Write-only	–

Table 44-18. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x638	TWI SMBus Timing Register	FLEX_TWI_SMBTR	Read/Write	0x00000000
0x63C	Reserved	–	–	–
0x640	TWI Alternative Command Register	FLEX_TWI_ACR	Read/Write	0x0
0x644	TWI Filter Register	FLEX_TWI_FILTR	Read/Write	0x00000000
0x648	Reserved	–	–	–
0x64C	TWI SleepWalking Matching Register	FLEX_TWI_SWMR	Read/Write	0x00000000
0x650	TWI FIFO Mode Register	FLEX_TWI_FMR	Read/Write	0x0
0x654	TWI FIFO Level Register	FLEX_TWI_FLR	Read-only	0x0
0x658–0x65C	Reserved	–	–	–
0x660	TWI FIFO Status Register	FLEX_TWI_FSR	Read-only	0x0
0x664	TWI FIFO Interrupt Enable Register	FLEX_TWI_FIER	Write-only	–
0x668	TWI FIFO Interrupt Disable Register	FLEX_TWI_FIDR	Write-only	–
0x66C	TWI FIFO Interrupt Mask Register	FLEX_TWI_FIMR	Read-only	0x0
0x670–0x6CC	Reserved	–	–	–
0x6D0	Reserved	–	–	–
0x6D4–0x6E0	Reserved	–	–	–
0x6E4	TWI Write Protection Mode Register	FLEX_TWI_WPMR	Read/Write	0x00000000
0x6E8	TWI Write Protection Status Register	FLEX_TWI_WPSR	Read-only	0x00000000
0x6EC–0x6FC	Reserved	–	–	–
0x700–0x7FC	Reserved	–	–	–

Notes: 1. Write is possible only in LIN master node configuration.

44.10.1 FLEXCOM Mode Register

Name: FLEX_MR

Address: 0xF8034000 (0), 0xF8038000 (1), 0xFC010000 (2), 0xFC014000 (3), 0xFC018000 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	OPMODE	

- **OPMODE: FLEXCOM Operating Mode**

Value	Name	Description
0	NO_COM	No communication
1	USART	All UART related protocols are selected (RS232, RS485, IrDA, ISO7816, LIN,) SPI/TWI related registers are not accessible and have no impact on IOs.
2	SPI	SPI operating mode is selected. USART/TWI related registers are not accessible and have no impact on IOs.
3	TWI	All TWI related protocols are selected (TWI, SMBus). USART/SPI related registers are not accessible and have no impact on IOs.

44.10.2 FLEXCOM Transmit Holding Register

Name: FLEX_THR

Address: 0xF8034020 (0), 0xF8038020 (1), 0xFC010020 (2), 0xFC014020 (3), 0xFC018020 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TXDATA							

7	6	5	4	3	2	1	0
TXDATA							

- **TXDATA: Transmit Data**

This register is a mirror of:

- USART Transmit Holding Register (FLEX_US_THR) if FLEX_MR.OPMODE field equals 1
- SPI Transmit Data Register (FLEX_SPI_TDR) if FLEX_MR.OPMODE field equals 2
- TWI Transmit Holding Register (FLEX_TWI_THR) if FLEX_MR.OPMODE field equals 3

44.10.3 FLEXCOM Receive Holding Register

Name: FLEX_RHR

Address: 0xF8034010 (0), 0xF8038010 (1), 0xFC010010 (2), 0xFC014010 (3), 0xFC018010 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
RXDATA							

7	6	5	4	3	2	1	0
RXDATA							

- **RXDATA: Receive Data**

This register is a mirror of:

- USART Receive Holding Register (FLEX_US_RHR) if FLEX_MR.OPMODE field equals 1
- SPI Receive Data Register (FLEX_SPI_RDR) if FLEX_MR.OPMODE field equals 2
- TWI Transmit Holding Register (FLEX_TWI_RHR) if FLEX_MR.OPMODE field equals 3

44.10.4 USART Control Register

Name: FLEX_US_CR

Address: 0xF8034200 (0), 0xF8038200 (1), 0xFC010200 (2), 0xFC014200 (3), 0xFC018200 (4)

Access: Write-only

31	30	29	28	27	26	25	24
FIFODIS	FIFOEN	–	REQCLR	–	TXFLCLR	RXFCLR	TXFCLR
23	22	21	20	19	18	17	16
–	–	LINWKUP	LINABT	RTSDIS	RTSEN	–	–
15	14	13	12	11	10	9	8
RETTO	RSTNACK	RSTIT	SENDA	STTTO	STPBRK	STTBRK	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

For SPI control, see Section 44.10.5 “USART Control Register (SPI_MODE)”.

- **RSTRX: Reset Receiver**

0: No effect.

1: Resets the receiver.

- **RSTTX: Reset Transmitter**

0: No effect.

1: Resets the transmitter.

- **RXEN: Receiver Enable**

0: No effect.

1: Enables the receiver, if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: Disables the receiver.

- **TXEN: Transmitter Enable**

0: No effect.

1: Enables the transmitter if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: Disables the transmitter.

- **RSTSTA: Reset Status Bits**

0: No effect.

1: Resets the status bits PARE, FRAME, OVRE, MANE, LINBE, LINISFE, LINIPE, LINCE, LINSNRE, LINSTE, LINHTE, LINID, LINTC, LINBK, CMP and RXBRK in FLEX_US_CSR. Also resets the status bits TXFEF, TXFFF, TXFTHF, RXFEF, RXFFF, RXFTHF, TXFPTEF, RXFPTEF in FLEX_US_FESR.

- **STTBRK: Start Break**

0: No effect.

1: Starts transmission of a break after the characters present in FLEX_US_THR and the Transmit Shift Register have been transmitted. No effect if a break is already being transmitted.

- **STPBRK: Stop Break**

0: No effect.

1: Stops transmission of the break after a minimum of one character length and transmits a high level during 12-bit periods. No effect if no break is being transmitted.

- **STTTO: Clear TIMEOUT Flag and Start Timeout After Next Character Received**

0: No effect.

1: Starts waiting for a character before clocking the timeout counter. Immediately disables a timeout period in progress. Resets the status bit TIMEOUT in FLEX_US_CSR.

- **SENDA: Send Address**

0: No effect.

1: In Multidrop mode only, the next character written to FLEX_US_THR is sent with the address bit set.

- **RSTIT: Reset Iterations**

0: No effect.

1: Resets ITER in FLEX_US_CSR. No effect if the ISO7816 is not enabled.

- **RSTNACK: Reset Non Acknowledge**

0: No effect

1: Resets NACK in FLEX_US_CSR.

- **RETTO: Start Timeout Immediately**

0: No effect

1: Immediately restarts timeout period.

- **RTSEN: Request to Send Enable**

0: No effect.

1: Drives the RTS pin to 1 if FLEX_US_MR.USART_MODE field = 2, else drives the RTS pin to 0 if FLEX_US_MR.USART_MODE field = 0.

- **RTSDIS: Request to Send Disable**

0: No effect.

1: Drives the RTS pin to 0 if FLEX_US_MR.USART_MODE field = 2, else drives the RTS pin to 1 if FLEX_US_MR.USART_MODE field = 0.

- **LINABT: Abort LIN Transmission**

0: No effect.

1: Aborts the current LIN transmission.

- **LINWKUP: Send LIN Wakeup Signal**

0: No effect:

1: Sends a wakeup signal on the LIN bus.

- **TXFCLR: Transmit FIFO Clear**

0: No effect.

1: Clears the Transmit FIFO, Transmit FIFO will become empty.

- **RXFCLR: Receive FIFO Clear**

0: No effect.

1: Clears the Receive FIFO, Receive FIFO will become empty.

- **TXFLCLR: Transmit FIFO Lock CLEAR**

0: No effect.

1: Clears the Transmit FIFO Lock

- **REQCLR: Request to Clear the Comparison Trigger**

- **FIFOEN: FIFO Enable**

0: No effect.

1: Enables the Transmit and Receive FIFOs

- **FIFODIS: FIFO Disable**

0: No effect.

1: Disables the Transmit and Receive FIFOs

44.10.5 USART Control Register (SPI_MODE)

Name: FLEX_US_CR (SPI_MODE)

Address: 0xF8034200 (0), 0xF8038200 (1), 0xFC010200 (2), 0xFC014200 (3), 0xFC018200 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	RCS	FCS	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

- **RSTRX: Reset Receiver**

0: No effect.

1: Resets the receiver.

- **RSTTX: Reset Transmitter**

0: No effect.

1: Resets the transmitter.

- **RXEN: Receiver Enable**

0: No effect.

1: Enables the receiver, if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: Disables the receiver.

- **TXEN: Transmitter Enable**

0: No effect.

1: Enables the transmitter if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: Disables the transmitter.

- **RSTSTA: Reset Status Bits**

0: No effect.

1: Resets the status bits OVRE, UNRE in FLEX_US_CSR.

- **FCS: Force SPI Chip Select**

Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

0: No effect.

1: Forces the Slave Select Line NSS (RTS pin) to 0, even if USART is not transmitting, in order to address SPI slave devices supporting the CSAAT mode (Chip Select Active After Transfer).

- **RCS: Release SPI Chip Select**

Applicable if USART operates in SPI Master mode (USART_MODE = 0xE):

0: No effect.

1: Releases the Slave Select Line NSS (RTS pin).

44.10.6 USART Mode Register

Name: FLEX_US_MR

Address: 0xF8034204 (0), 0xF8038204 (1), 0xFC010204 (2), 0xFC014204 (3), 0xFC018204 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
ONEBIT	MODSYNC	MAN	FILTER	–	MAX_ITERATION		

23	22	21	20	19	18	17	16
INVDATA	VAR_SYNC	DSNACK	INACK	OVER	CLKO	MODE9	MSBF

15	14	13	12	11	10	9	8
CHMODE		NBSTOP		PAR			SYNC

7	6	5	4	3	2	1	0
CHRL		USCLKS		USART_MODE			

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

For SPI configuration, see Section 44.10.7 “USART Mode Register (SPI_MODE)”.

- **USART_MODE: USART Mode of Operation**

Value	Name	Description
0x0	NORMAL	Normal mode
0x1	RS485	RS485
0x2	HW_HANDSHAKING	Hardware handshaking
0x4	IS07816_T_0	IS07816 Protocol: T = 0
0x6	IS07816_T_1	IS07816 Protocol: T = 1
0x8	IRDA	IrDA
0xA	LIN_MASTER	LIN Master mode
0xB	LIN_SLAVE	LIN Slave mode
0xE	SPI_MASTER	SPI Master mode (CLKO must be written to 1 and USCLKS = 0, 1 or 2)
0xF	SPI_SLAVE	SPI Slave mode

- **USCLKS: Clock Selection**

Value	Name	Description
0	MCK	Peripheral clock is selected
1	DIV	Peripheral clock divided (DIV = 8) is selected
2	GCLK	PMC generic clock is selected. If the SCK pin is driven (CLKO = 1), the CD field must be greater than 1.
3	SCK	External pin SCK is selected

- **CHRL: Character Length**

Value	Name	Description
0	5_BIT	Character length is 5 bits
1	6_BIT	Character length is 6 bits
2	7_BIT	Character length is 7 bits
3	8_BIT	Character length is 8 bits

- **SYNC: Synchronous Mode Select**

0: USART operates in Asynchronous mode (UART).

1: USART operates in Synchronous mode.

- **PAR: Parity Type**

Value	Name	Description
0	EVEN	Even parity
1	ODD	Odd parity
2	SPACE	Parity forced to 0 (Space)
3	MARK	Parity forced to 1 (Mark)
4	NO	No parity
6	MULTIDROP	Multidrop mode

- **NBSTOP: Number of Stop Bits**

Value	Name	Description
0	1_BIT	1 stop bit
1	1_5_BIT	1.5 stop bit (SYNC = 0) or reserved (SYNC = 1)
2	2_BIT	2 stop bits

- **CHMODE: Channel Mode**

Value	Name	Description
0	NORMAL	Normal mode
1	AUTOMATIC	Automatic Echo. Receiver input is connected to the TXD pin.
2	LOCAL_LOOPBACK	Local Loopback. Transmitter output is connected to the Receiver Input.
3	REMOTE_LOOPBACK	Remote Loopback. RXD pin is internally connected to the TXD pin.

- **MSBF: Bit Order**

0: Least significant bit is sent/received first.

1: Most significant bit is sent/received first.

- **MODE9: 9-bit Character Length**

0: CHRL defines character length.

1: 9-bit character length.

- **CLKO: Clock Output Select**

0: The USART does not drive the SCK pin (Synchronous Slave mode or Asynchronous mode with external baud rate clock source).

1: The USART drives the SCK pin if USCLKS does not select the external clock SCK (USART Synchronous Master mode).

- **OVER: Oversampling Mode**

0: 16x Oversampling.

1: 8x Oversampling.

- **INACK: Inhibit Non Acknowledge**

0: The NACK is generated.

1: The NACK is not generated.

- **DSNACK: Disable Successive NACK**

0: NACK is sent on the ISO line as soon as a parity error occurs in the received character (unless INACK is set).

1: Successive parity errors are counted up to the value specified in the MAX_ITERATION field. These parity errors generate a NACK on the ISO line. As soon as this value is reached, no additional NACK is sent on the ISO line. The flag ITER is asserted.

Note: The MAX_ITERATION field must be cleared if DSNACK is cleared.

- **INVDATA: Inverted Data**

0: The data field transmitted on TXD line is the same as the one written in FLEX_US_THR or the content read in FLEX_US_RHR is the same as RXD line. Normal mode of operation.

1: The data field transmitted on TXD line is inverted (voltage polarity only) compared to the value written in FLEX_US_THR or the content read in FLEX_US_RHR is inverted compared to what is received on RXD line (or ISO7816 IO line). Inverted mode of operation, useful for contactless card application. To be used with configuration bit MSBF.

- **VAR_SYNC: Variable Synchronization of Command/Data Sync Start Frame Delimiter**

0: User defined configuration of command or data sync field depending on MODSYNC value.

1: The sync field is updated when a character is written into FLEX_US_THR.

- **MAX_ITERATION: Maximum Number of Automatic Iteration**

0–7: Defines the maximum number of iterations in mode ISO7816, protocol T = 0.

- **FILTER: Receive Line Filter**

0: The USART does not filter the receive line.

1: The USART filters the receive line using a three-sample filter (1/16-bit clock) (2 over 3 majority).

- **MAN: Manchester Encoder/Decoder Enable**

0: Manchester encoder/decoder are disabled.

1: Manchester encoder/decoder are enabled.

- **MODSYNC: Manchester Synchronization Mode**

0:The Manchester start bit is a 0 to 1 transition

1: The Manchester start bit is a 1 to 0 transition.

- **ONEBIT: Start Frame Delimiter Selector**

0: Start frame delimiter is COMMAND or DATA SYNC.

1: Start frame delimiter is one bit.

44.10.7 USART Mode Register (SPI_MODE)

Name: FLEX_US_MR (SPI_MODE)

Address: 0xF8034204 (0), 0xF8038204 (1), 0xFC010204 (2), 0xFC014204 (3), 0xFC018204 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	WRDBT	–	–	–	CPOL
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CPHA
7	6	5	4	3	2	1	0
CHRL		USCLKS		USART_MODE			

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **USART_MODE: USART Mode of Operation**

Value	Name	Description
0xE	SPI_MASTER	SPI master
0xF	SPI_SLAVE	SPI slave

- **USCLKS: Clock Selection**

Value	Name	Description
0	MCK	Peripheral clock is selected
1	DIV	Peripheral clock Divided (DIV= 8) is selected
2	GCLK	A PMC generic clock is selected
3	SCK	External pin SCK is selected

- **CHRL: Character Length**

Value	Name	Description
3	8_BIT	Character length is 8 bits

- **CPHA: SPI Clock Phase**
  – Applicable if USART operates in SPI mode (USART_MODE = 0xE or 0xF):

0: Data are changed on the leading edge of SPCK and captured on the following edge of SPCK.

1: Data are captured on the leading edge of SPCK and changed on the following edge of SPCK.

CPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

- **CHMODE: Channel Mode**

Value	Name	Description
0	NORMAL	Normal mode
1	AUTOMATIC	Automatic Echo mode. Receiver input is connected to the TXD pin.
2	LOCAL_LOOPBACK	Local Loopback mode. Transmitter output is connected to the Receiver Input.
3	REMOTE_LOOPBACK	Remote Loopback mode. RXD pin is internally connected to the TXD pin.

- **CPOL: SPI Clock Polarity**

Applicable if USART operates in SPI mode (slave or master, USART_MODE = 0xE or 0xF):

0: The inactive state value of SPCK is logic level zero.

1: The inactive state value of SPCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

- **WRDBT: Wait Read Data Before Transfer**

0: The character transmission starts as soon as a character is written into FLEX_US_THR (assuming TXRDY was set).

1: The character transmission starts when a character is written and only if RXRDY flag is cleared (Receive Holding Register has been read).

44.10.8 USART Interrupt Enable Register

Name: FLEX_US_IER

Address: 0xF8034208 (0), 0xF8038208 (1), 0xFC010208 (2), 0xFC014208 (3), 0xFC018208 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
–	CMP	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI-s-specific configurations, see Section 44.10.9 “USART Interrupt Enable Register (SPI_MODE)”.

For LIN-specific configurations, see Section 44.10.10 “USART Interrupt Enable Register (LIN_MODE)”.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Enable**
- **TXRDY: TXRDY Interrupt Enable**
- **RXBRK: Receiver Break Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **FRAME: Framing Error Interrupt Enable**
- **PARE: Parity Error Interrupt Enable**
- **TIMEOUT: Timeout Interrupt Enable**
- **TXEMPTY: TXEMPTY Interrupt Enable**
- **ITER: Max number of Repetitions Reached Interrupt Enable**
- **NACK: Non Acknowledge Interrupt Enable**
- **CTSIC: Clear to Send Input Change Interrupt Enable**
- **CMP: Comparison Interrupt Enable**
- **MANE: Manchester Error Interrupt Enable**

44.10.9 USART Interrupt Enable Register (SPI_MODE)

Name: FLEX_US_IER (SPI_MODE)

Address: 0xF8034208 (0), 0xF8038208 (1), 0xFC010208 (2), 0xFC014208 (3), 0xFC018208 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	CMP	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Enable**
- **TXRDY: TXRDY Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **TXEMPTY: TXEMPTY Interrupt Enable**
- **UNRE: SPI Underrun Error Interrupt Enable**
- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event**
- **CMP: Comparison Interrupt Enable**

44.10.10 USART Interrupt Enable Register (LIN_MODE)

Name: FLEX_US_IER (LIN_MODE)

Address: 0xF8034208 (0), 0xF8038208 (1), 0xFC010208 (2), 0xFC014208 (3), 0xFC018208 (4)

Access: Write-only

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
LINTC	LINID	LINBK	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Enable**
- **TXRDY: TXRDY Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **FRAME: Framing Error Interrupt Enable**
- **PARE: Parity Error Interrupt Enable**
- **TIMEOUT: Timeout Interrupt Enable**
- **TXEMPTY: TXEMPTY Interrupt Enable**
- **LINBK: LIN Break Sent or LIN Break Received Interrupt Enable**
- **LINID: LIN Identifier Sent or LIN Identifier Received Interrupt Enable**
- **LINTC: LIN Transfer Completed Interrupt Enable**
- **LINBE: LIN Bus Error Interrupt Enable**
- **LINISFE: LIN Inconsistent Synch Field Error Interrupt Enable**
- **LINIPE: LIN Identifier Parity Interrupt Enable**
- **LINCE: LIN Checksum Error Interrupt Enable**
- **LINSNRE: LIN Slave Not Responding Error Interrupt Enable**

- **LINSTE: LIN Synch Tolerance Error Interrupt Enable**
- **LINHTE: LIN Header Timeout Error Interrupt Enable**

44.10.11 USART Interrupt Disable Register

Name: FLEX_US_IDR

Address: 0xF803420C (0), 0xF803820C (1), 0xFC01020C (2), 0xFC01420C (3), 0xFC01820C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
–	CMP	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI-specific configurations, see Section 44.10.12 “USART Interrupt Disable Register (SPI_MODE)”.

For LIN-specific configurations, see Section 44.10.13 “USART Interrupt Disable Register (LIN_MODE)”.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Disable**
- **TXRDY: TXRDY Interrupt Disable**
- **RXBRK: Receiver Break Interrupt Disable**
- **OVRE: Overrun Error Interrupt Enable**
- **FRAME: Framing Error Interrupt Disable**
- **PARE: Parity Error Interrupt Disable**
- **TIMEOUT: Timeout Interrupt Disable**
- **TXEMPTY: TXEMPTY Interrupt Disable**
- **ITER: Max Number of Repetitions Reached Interrupt Disable**
- **NACK: Non Acknowledge Interrupt Disable**
- **CTSIC: Clear to Send Input Change Interrupt Disable**
- **CMP: Comparison Interrupt Disable**
- **MANE: Manchester Error Interrupt Disable**

44.10.12 USART Interrupt Disable Register (SPI_MODE)

Name: FLEX_US_IDR (SPI_MODE)

Address: 0xF803420C (0), 0xF803820C (1), 0xFC01020C (2), 0xFC01420C (3), 0xFC01820C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	CMP	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Disable**
- **TXRDY: TXRDY Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **TXEMPTY: TXEMPTY Interrupt Disable**
- **UNRE: SPI Underrun Error Interrupt Disable**
- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event**
- **CMP: Comparison Interrupt Disable**

44.10.13 USART Interrupt Disable Register (LIN_MODE)

Name: FLEX_US_IDR (LIN_MODE)

Address: 0xF803420C (0), 0xF803820C (1), 0xFC01020C (2), 0xFC01420C (3), 0xFC01820C (4)

Access: Write-only

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
LINTC	LINID	LINBK	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt.

- **RXRDY: RXRDY Interrupt Disable**
- **TXRDY: TXRDY Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **FRAME: Framing Error Interrupt Disable**
- **PARE: Parity Error Interrupt Disable**
- **TIMEOUT: Timeout Interrupt Disable**
- **TXEMPTY: TXEMPTY Interrupt Disable**
- **LINBK: LIN Break Sent or LIN Break Received Interrupt Disable**
- **LINID: LIN Identifier Sent or LIN Identifier Received Interrupt Disable**
- **LINTC: LIN Transfer Completed Interrupt Disable**
- **LINBE: LIN Bus Error Interrupt Disable**
- **LINISFE: LIN Inconsistent Synch Field Error Interrupt Disable**
- **LINIPE: LIN Identifier Parity Interrupt Disable**
- **LINCE: LIN Checksum Error Interrupt Disable**
- **LINSNRE: LIN Slave Not Responding Error Interrupt Disable**

- **LINSTE: LIN Synch Tolerance Error Interrupt Disable**
- **LINHTE: LIN Header Timeout Error Interrupt Disable**

44.10.14 USART Interrupt Mask Register

Name: FLEX_US_IMR

Address: 0xF8034210 (0), 0xF8038210 (1), 0xFC010210 (2), 0xFC014210 (3), 0xFC018210 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
–	CMP	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI-specific configurations, see Section 44.10.15 “USART Interrupt Mask Register (SPI_MODE)”.

For LIN-specific configurations, see Section 44.10.16 “USART Interrupt Mask Register (LIN_MODE)”.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **RXBRK: Receiver Break Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **FRAME: Framing Error Interrupt Mask**
- **PARE: Parity Error Interrupt Mask**
- **TIMEOUT: Timeout Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **ITER: Max Number of Repetitions Reached Interrupt Mask**
- **NACK: Non Acknowledge Interrupt Mask**
- **CTSIC: Clear to Send Input Change Interrupt Mask**
- **CMP: Comparison Interrupt Mask**
- **MANE: Manchester Error Interrupt Mask**

44.10.15 USART Interrupt Mask Register (SPI_MODE)

Name: FLEX_US_IMR (SPI_MODE)

Address: 0xF8034210 (0), 0xF8038210 (1), 0xFC010210 (2), 0xFC014210 (3), 0xFC018210 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	CMP	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **UNRE: SPI Underrun Error Interrupt Mask**
- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event**
- **CMP: Comparison Interrupt Mask**

44.10.16 USART Interrupt Mask Register (LIN_MODE)

Name: FLEX_US_IMR (LIN_MODE)

Address: 0xF8034210 (0), 0xF8038210 (1), 0xFC010210 (2), 0xFC014210 (3), 0xFC018210 (4)

Access: Read-only

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
LINTC	LINID	LINBK	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RXRDY: RXRDY Interrupt Mask**
- **TXRDY: TXRDY Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **FRAME: Framing Error Interrupt Mask**
- **PARE: Parity Error Interrupt Mask**
- **TIMEOUT: Timeout Interrupt Mask**
- **TXEMPTY: TXEMPTY Interrupt Mask**
- **LINBK: LIN Break Sent or LIN Break Received Interrupt Mask**
- **LINID: LIN Identifier Sent or LIN Identifier Received Interrupt Mask**
- **LINTC: LIN Transfer Completed Interrupt Mask**
- **LINBE: LIN Bus Error Interrupt Mask**
- **LINISFE: LIN Inconsistent Synch Field Error Interrupt Mask**
- **LINIPE: LIN Identifier Parity Interrupt Mask**
- **LINCE: LIN Checksum Error Interrupt Mask**
- **LINSNRE: LIN Slave Not Responding Error Interrupt Mask**

- **LINSTE: LIN Synch Tolerance Error Interrupt Mask**
- **LINHTE: LIN Header Timeout Error Interrupt Mask**

44.10.17 USART Channel Status Register

Name: FLEX_US_CSR

Address: 0xF8034214 (0), 0xF8038214 (1), 0xFC010214 (2), 0xFC014214 (3), 0xFC018214 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	MANE
23	22	21	20	19	18	17	16
CTS	CMP	–	–	CTSIC	–	–	–
15	14	13	12	11	10	9	8
–	–	NACK	–	–	ITER	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	RXBRK	TXRDY	RXRDY

For SPI-specific configurations, see Section 44.10.18 “USART Channel Status Register (SPI_MODE)”.

For LIN-specific configurations, see Section 44.10.19 “USART Channel Status Register (LIN_MODE)”.

- **RXRDY: Receiver Ready (cleared by reading FLEX_US_RHR)**

When FIFOs are disabled:

0: No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.

1: At least one complete character has been received and FLEX_US_RHR has not yet been read.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read

1: At least one unread data is in the Receive FIFO

RXRDY behavior with FIFO enabled is illustrated in Section 44.7.11.4 “TXRDY, RXRDY and TXEMPTY Behavior”.

- **TXRDY: Transmitter Ready (cleared by writing FLEX_US_THR)**

When FIFOs are disabled:

0: A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or an STTBRK command has been requested, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.

1: There is no character in FLEX_US_THR.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration

TXRDY behavior with FIFO enabled is illustrated in Section 44.7.11.4 “TXRDY, RXRDY and TXEMPTY Behavior”.

- **RXBRK: Break Received/End of Break**

0: No break received or end of break detected since the last RSTSTA command was issued.

1: Break received or end of break detected since the last RSTSTA command was issued.

- **OVRE: Overrun Error**

0: No overrun error has occurred since the last RSTSTA command was issued.

1: At least one overrun error has occurred since the last RSTSTA command was issued.

- **FRAME: Framing Error**

0: No stop bit has been detected low since the last RSTSTA command was issued.

1: At least one stop bit has been detected low since the last RSTSTA command was issued.

- **PARE: Parity Error**

0: No parity error has been detected since the last RSTSTA command was issued.

1: At least one parity error has been detected since the last RSTSTA command was issued.

- **TIMEOUT: Receiver Timeout**

0: There has not been a timeout since the last Start Timeout command (STTTO in FLEX_US_CR) or the Timeout Register is 0.

1: There has been a timeout since the last Start Timeout command (STTTO in FLEX_US_CR).

- **TXEMPTY: Transmitter Empty (cleared by writing FLEX_US_THR)**

0: There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled.

1: There are no characters in FLEX_US_THR, nor in the Transmit Shift Register.

- **ITER: Max Number of Repetitions Reached**

0: Maximum number of repetitions has not been reached since the last RSTIT command was issued.

1: Maximum number of repetitions has been reached since the last RSTIT command was issued.

- **NACK: Non Acknowledge Interrupt**

0: Non acknowledge has not been detected since the last RSTNACK.

1: At least one non acknowledge has been detected since the last RSTNACK.

- **CTSIC: Clear to Send Input Change Flag**

0: No input change has been detected on the CTS pin since the last read of FLEX_US_CSR.

1: At least one input change has been detected on the CTS pin since the last read of FLEX_US_CSR.

- **CMP: Comparison Status**

0: No received character matched the comparison criteria programmed in VAL1, VAL2 fields and CMPPAR bit in since the last RSTSTA command was issued.

1: A received character matched the comparison criteria since the last RSTSTA command was issued.

- **CTS: Image of CTS Input**

0: CTS input is driven low.

1: CTS input is driven high.

- **MANE: Manchester Error**

0: No Manchester error has been detected since the last RSTSTA command was issued.

1: At least one Manchester error has been detected since the last RSTSTA command was issued.

44.10.18 USART Channel Status Register (SPI_MODE)

Name: FLEX_US_CSR (SPI_MODE)

Address: 0xF8034214 (0), 0xF8038214 (1), 0xFC010214 (2), 0xFC014214 (3), 0xFC018214 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
NSS	CMP	–	–	NSSE	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	UNRE	TXEMPTY	–
7	6	5	4	3	2	1	0
–	–	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xE or 0xF in the USART Mode Register.

- **RXRDY: Receiver Ready (cleared by reading FLEX_US_RHR)**

0: No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.

1: At least one complete character has been received and FLEX_US_RHR has not yet been read.

- **TXRDY: Transmitter Ready (cleared by writing FLEX_US_THR)**

0: A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.

1: There is no character in FLEX_US_THR.

- **OVRE: Overrun Error**

0: No overrun error has occurred since the last RSTSTA command was issued.

1: At least one overrun error has occurred since the last RSTSTA command was issued.

- **TXEMPTY: Transmitter Empty (cleared by writing FLEX_US_THR)**

0: There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled.

1: There are no characters in FLEX_US_THR, nor in the Transmit Shift Register.

- **UNRE: Underrun Error**

0: No SPI underrun error has occurred since the last RSTSTA command was issued.

1: At least one SPI underrun error has occurred since the last RSTSTA command was issued.

- **NSSE: NSS Line (Driving CTS Pin) Rising or Falling Edge Event (cleared on read)**

0: No NSS line event has been detected since the last read of FLEX_US_CSR.

1: A rising or falling edge has been detected on the NSS line since the last read of FLEX_US_CSR.

- **CMP: Comparison Match**

0: No received character matched the comparison criteria programmed in VAL1, VAL2 fields and CMPPAR bit in FLEX_US_CMPR since the last RSTSTA command was issued.

1: A received character matched the comparison criteria since the last RSTSTA command was issued.

- **NSS: Image of NSS Line**

0: NSS line is driven low (if NSSE = 1, falling edge occurred on NSS line).

1: NSS line is driven high (if NSSE = 1, rising edge occurred on NSS line).

44.10.19 USART Channel Status Register (LIN_MODE)

Name: FLEX_US_CSR (LIN_MODE)

Address: 0xF8034214 (0), 0xF8038214 (1), 0xFC010214 (2), 0xFC014214 (3), 0xFC018214 (4)

Access: Read-only

31	30	29	28	27	26	25	24
LINHTE	LINSTE	LINSNRE	LINCE	LINIPE	LINISFE	LINBE	–
23	22	21	20	19	18	17	16
LINBLS	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
LINTC	LINID	LINBK	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

This configuration is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

- **RXRDY: Receiver Ready (cleared by reading FLEX_US_RHR)**

0: No complete character has been received since the last read of FLEX_US_RHR or the receiver is disabled. If characters were being received when the receiver was disabled, RXRDY changes to 1 when the receiver is enabled.

1: At least one complete character has been received and FLEX_US_RHR has not yet been read.

- **TXRDY: Transmitter Ready (cleared by writing FLEX_US_THR)**

0: A character in FLEX_US_THR is waiting to be transferred to the Transmit Shift Register, or the transmitter is disabled. As soon as the transmitter is enabled, TXRDY becomes 1.

1: There is no character in FLEX_US_THR.

- **OVRE: Overrun Error**

0: No overrun error has occurred since the last RSTSTA command was issued.

1: At least one overrun error has occurred since the last RSTSTA command was issued.

- **FRAME: Framing Error**

0: No stop bit has been detected low since the last RSTSTA command was issued.

1: At least one stop bit has been detected low since the last RSTSTA command was issued.

- **PARE: Parity Error**

0: No parity error has been detected since the last RSTSTA command was issued.

1: At least one parity error has been detected since the last RSTSTA command was issued.

- **TIMEOUT: Receiver Timeout**

0: There has not been a timeout since the last start timeout command (STTTO in FLEX_US_CR) or the Timeout Register is 0.

1: There has been a timeout since the last start timeout command (STTTO in FLEX_US_CR).

- **TXEMPTY: Transmitter Empty (cleared by writing FLEX_US_THR)**

0: There are characters in either FLEX_US_THR or the Transmit Shift Register, or the transmitter is disabled.

1: There are no characters in FLEX_US_THR, nor in the Transmit Shift Register.

- **LINBK: LIN Break Sent or LIN Break Received**

Applicable if USART operates in LIN Master mode (USART_MODE = 0xA):

0: No LIN break has been sent since the last RSTSTA command was issued.

1:At least one LIN break has been sent since the last RSTSTA

If USART operates in LIN Slave mode (USART_MODE = 0xB):

0: No LIN break has received sent since the last RSTSTA command was issued.

1:At least one LIN break has been received since the last RSTSTA command was issued.

- **LINID: LIN Identifier Sent or LIN Identifier Received**

If USART operates in LIN Master mode (USART_MODE = 0xA):

0: No LIN identifier has been sent since the last RSTSTA command was issued.

1:At least one LIN identifier has been sent since the last RSTSTA command was issued.

If USART operates in LIN Slave mode (USART_MODE = 0xB):

0: No LIN identifier has been received since the last RSTSTA command was issued.

1:At least one LIN identifier has been received since the last RSTSTA

- **LINTC: LIN Transfer Completed**

0: The USART is idle or a LIN transfer is ongoing.

1: A LIN transfer has been completed since the last RSTSTA command was issued.

- **LINBLS: LIN Bus Line Status**

0: LIN bus line is set to 0.

1: LIN bus line is set to 1.

- **LINBE: LIN Bit Error**

0: No bit error has been detected since the last RSTSTA command was issued.

1: A bit error has been detected since the last RSTSTA command was issued.

- **LINISFE: LIN Inconsistent Synch Field Error**

0: No LIN inconsistent synch field error has been detected since the last RSTSTA

1: The USART is configured as a slave node and a LIN Inconsistent synch field error has been detected since the last RSTSTA command was issued.

- **LINIPE: LIN Identifier Parity Error**

0: No LIN identifier parity error has been detected since the last RSTSTA command was issued.

1: A LIN identifier parity error has been detected since the last RSTSTA command was issued.

- **LINCE: LIN Checksum Error**

0: No LIN checksum error has been detected since the last RSTSTA command was issued.

1: A LIN checksum error has been detected since the last RSTSTA command was issued.

- **LINSNRE: LIN Slave Not Responding Error**

0: No LIN slave not responding error has been detected since the last RSTSTA command was issued.

1: A LIN slave not responding error has been detected since the last RSTSTA command was issued.

- **LINSTE: LIN Synch Tolerance Error**

0: No LIN synch tolerance error has been detected since the last RSTSTA command was issued.

1: A LIN synch tolerance error has been detected since the last RSTSTA command was issued.

- **LINHTE: LIN Header Timeout Error**

0: No LIN header timeout error has been detected since the last RSTSTA command was issued.

1: A LIN header timeout error has been detected since the last RSTSTA command was issued.

44.10.20 USART Receive Holding Register

Name: FLEX_US_RHR

Address: 0xF8034218 (0), 0xF8038218 (1), 0xFC010218 (2), 0xFC014218 (3), 0xFC018218 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RXSYNH	–	–	–	–	–	–	RXCHR
7	6	5	4	3	2	1	0
RXCHR							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.7.11.5 “Single Data Mode” for details.

- **RXCHR: Received Character**

Last character received if RXRDY is set.

- **RXSYNH: Received Sync**

0: Last character received is a data.

1: Last character received is a command.

44.10.21 USART Receive Holding Register (FIFO Multi Data)

Name: FLEX_US_RHR (FIFO_MULTI_DATA)

Address: 0xF8034218 (0), 0xF8038218 (1), 0xFC010218 (2), 0xFC014218 (3), 0xFC018218 (4)

Access: Read-only

31	30	29	28	27	26	25	24
RXCHR3							
23	22	21	20	19	18	17	16
RXCHR2							
15	14	13	12	11	10	9	8
RXCHR1							
7	6	5	4	3	2	1	0
RXCHR0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.7.11.6 “Multiple Data Mode” for details.

- **RXCHRx: Received Character**

First unread character in the Receive FIFO if RXRDY is set.

44.10.22 USART Transmit Holding Register

Name: FLEX_US_THR

Address: 0xF803421C (0), 0xF803821C (1), 0xFC01021C (2), 0xFC01421C (3), 0xFC01821C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
TXSYNH	–	–	–	–	–	–	TXCHR

7	6	5	4	3	2	1	0
TXCHR							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.7.11.5 “Single Data Mode” for details.

- **TXCHR: Character to be Transmitted**

Next character to be transmitted after the current character if TXRDY is not set.

- **TXSYNH: Sync Field to be Transmitted**

0: The next character sent is encoded as a data. Start frame delimiter is DATA SYNC.

1: The next character sent is encoded as a command. Start frame delimiter is COMMAND SYNC.

44.10.23 USART Transmit Holding Register (FIFO Multi Data)

Name: FLEX_US_THR (FIFO_MULTI_DATA)

Address: 0xF803421C (0), 0xF803821C (1), 0xFC01021C (2), 0xFC01421C (3), 0xFC01821C (4)

Access: Write-only

31	30	29	28	27	26	25	24
TXCHR3							
23	22	21	20	19	18	17	16
TXCHR2							
15	14	13	12	11	10	9	8
TXCHR1							
7	6	5	4	3	2	1	0
TXCHR0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.7.11.6 “Multiple Data Mode” for details.

- **TXCHRx: Character to be Transmitted**

Next character to be transmitted.

44.10.24 USART Baud Rate Generator Register

Name: FLEX_US_BRGR

Address: 0xF8034220 (0), 0xF8038220 (1), 0xFC010220 (2), 0xFC014220 (3), 0xFC018220 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	FP		

15	14	13	12	11	10	9	8
CD							

7	6	5	4	3	2	1	0
CD							

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **CD: Clock Divider**

	USART_MODE ≠ ISO7816			
	SYNC = 0		**SYNC = 1 or USART_MODE = SPI (master or Slave)**	**USART_MODE = ISO7816**
CD	**OVER = 0**	**OVER = 1**		
0	Baud Rate Clock Disabled			
1 to 65535	CD = Selected Clock / (16 × Baud Rate)	CD = Selected Clock / (8 × Baud Rate)	CD = Selected Clock / Baud Rate	CD = Selected Clock / (FI_DI_RATIO × Baud Rate)

- **FP: Fractional Part**

0: Fractional divider is disabled.

1–7: Baud rate resolution, defined by FP × 1/8.

Warning: When the value of field FP is greater than 0, the SCK (oversampling clock) generates non-constant duty cycles. The SCK high duration is increased by “selected clock” period from time to time. The duty cycle depends on the value of the CD field.

44.10.25 USART Receiver Timeout Register

Name: FLEX_US_RTOR

Address: 0xF8034224 (0), 0xF8038224 (1), 0xFC010224 (2), 0xFC014224 (3), 0xFC018224 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	TO

15	14	13	12	11	10	9	8
TO							

7	6	5	4	3	2	1	0
TO							

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **TO: Timeout Value**

0: The receiver timeout is disabled.

1–131071: The receiver timeout is enabled and the timeout delay is TO × bit period.

44.10.26 USART Transmitter Timeguard Register

Name: FLEX_US_TTGR

Address: 0xF8034228 (0), 0xF8038228 (1), 0xFC010228 (2), 0xFC014228 (3), 0xFC018228 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
TG							

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **TG: Timeguard Value**

0: The transmitter timeguard is disabled.

1–255: The transmitter timeguard is enabled and TG is timeguard delay / bit period.

44.10.27 USART FI DI RATIO Register

Name: FLEX_US_FIDI

Address: 0xF8034240 (0), 0xF8038240 (1), 0xFC010240 (2), 0xFC014240 (3), 0xFC018240 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
FI_DI_RATIO							
7	6	5	4	3	2	1	0
FI_DI_RATIO							

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **FI_DI_RATIO: FI Over DI Ratio Value**

0: If ISO7816 mode is selected, the baud rate generator generates no signal.

1–2: Do not use.

3–65535: If ISO7816 mode is selected, the baud rate is the clock provided on SCK divided by FI_DI_RATIO.

44.10.28 USART Number of Errors Register

Name: FLEX_US_NER

Address: 0xF8034244 (0), 0xF8038244 (1), 0xFC010244 (2), 0xFC014244 (3), 0xFC018244 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
NB_ERRORS							

This register is relevant only if USART_MODE = 0x4 or 0x6 in the USART Mode Register.

- **NB_ERRORS: Number of Errors**

Total number of errors that occurred during an ISO7816 transfer. This register automatically clears when read.

44.10.29 USART IrDA FILTER Register

Name: FLEX_US_IF

Address: 0xF803424C (0), 0xF803824C (1), 0xFC01024C (2), 0xFC01424C (3), 0xFC01824C (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
IRDA_FILTER							

This register is relevant only if USART_MODE = 0x8 in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **IRDA_FILTER: IrDA Filter**

The IRDA_FILTER value must be defined to meet the following criteria:

$t_{peripheral\ clock} \times (IRDA_FILTER + 3) < 1.41\ \mu s$

44.10.30 USART Manchester Configuration Register

Name: FLEX_US_MAN

Address: 0xF8034250 (0), 0xF8038250 (1), 0xFC010250 (2), 0xFC014250 (3), 0xFC018250 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
RXIDLEV	DRIFT	ONE	RX_MPOL	–	–	RX_PP	

23	22	21	20	19	18	17	16
–	–	–	–	RX_PL			

15	14	13	12	11	10	9	8
–	–	–	TX_MPOL	–	–	TX_PP	

7	6	5	4	3	2	1	0
–	–	–	–	TX_PL			

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **TX_PL: Transmitter Preamble Length**

0: The transmitter preamble pattern generation is disabled

1–15: The preamble length is TX_PL × Bit Period

- **TX_PP: Transmitter Preamble Pattern**

The following values assume that TX_MPOL field is not set:

Value	Name	Description
0	ALL_ONE	The preamble is composed of ‘1’s
1	ALL_ZERO	The preamble is composed of ‘0’s
2	ZERO_ONE	The preamble is composed of ‘01’s
3	ONE_ZERO	The preamble is composed of ‘10’s

- **TX_MPOL: Transmitter Manchester Polarity**

0: Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition.

1: Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition.

- **RX_PL: Receiver Preamble Length**

0: The receiver preamble pattern detection is disabled

1–15: The detected preamble length is RX_PL × Bit Period

- **RX_PP: Receiver Preamble Pattern detected**

The following values assume that RX_MPOL field is not set:

Value	Name	Description
0	ALL_ONE	The preamble is composed of ‘1’s
1	ALL_ZERO	The preamble is composed of ‘0’s
2	ZERO_ONE	The preamble is composed of ‘01’s
3	ONE_ZERO	The preamble is composed of ‘10’s

- **RX_MPOL: Receiver Manchester Polarity**

0: Logic zero is coded as a zero-to-one transition, Logic one is coded as a one-to-zero transition.

1: Logic zero is coded as a one-to-zero transition, Logic one is coded as a zero-to-one transition.

- **ONE: Must Be Set to 1**

Bit 29 must always be set to 1 when programming the FLEX_US_MAN register.

- **DRIFT: Drift Compensation**

0: The USART cannot recover from an important clock drift

1: The USART can recover from clock drift. The 16X Clock mode must be enabled.

- **RXIDLEV: Receiver Idle Value**

0: Receiver line idle value is 0.

1: Receiver line idle value is 1.

44.10.31 USART LIN Mode Register

Name: FLEX_US_LINMR

Address: 0xF8034254 (0), 0xF8038254 (1), 0xFC010254 (2), 0xFC014254 (3), 0xFC018254 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	SYNCDIS	PDCM

15	14	13	12	11	10	9	8
DLC							

7	6	5	4	3	2	1	0
WKUPTYP	FSDIS	DLM	CHKTYP	CHKDIS	PARDIS	NACT	

This register is relevant only if USART_MODE = 0xA or 0xB in the USART Mode Register.

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **NACT: LIN Node Action**

Value	Name	Description
0	PUBLISH	The USART transmits the response.
1	SUBSCRIBE	The USART receives the response.
2	IGNORE	The USART does not transmit and does not receive the response.

Values which are not listed in the table must be considered as “reserved”.

- **PARDIS: Parity Disable**

0: In master node configuration, the identifier parity is computed and sent automatically. In master node and slave node configuration, the parity is checked automatically.

1: Whatever the node configuration is, the Identifier parity is not computed/sent and it is not checked.

- **CHKDIS: Checksum Disable**

0: In master node configuration, the checksum is computed and sent automatically. In slave node configuration, the checksum is checked automatically.

1: Whatever the node configuration is, the checksum is not computed/sent and it is not checked.

- **CHKTYP: Checksum Type**

0: LIN 2.0 “enhanced” checksum

1: LIN 1.3 “classic” checksum

- **DLM: Data Length Mode**

0: The response data length is defined by the DLC field of this register.

1: The response data length is defined by the bits 5 and 6 of the identifier (IDCHR in FLEX_US_LINIR).

- **FSDIS: Frame Slot Mode Disable**

0: The Frame Slot mode is enabled.

1: The Frame Slot mode is disabled.

- **WKUPTYP: Wakeup Signal Type**

0: Setting the LINWKUP bit in the control register sends a LIN 2.0 wakeup signal.

1: Setting the LINWKUP bit in the control register sends a LIN 1.3 wakeup signal.

- **DLC: Data Length Control**

0–255: Defines the response data length if DLM = 0,in that case the response data length is equal to DLC+1 bytes.

- **PDCM: DMAC Mode**

0: The LIN mode register FLEX_US_LINMR is not written by the DMAC.

1: The LIN mode register FLEX_US_LINMR (excepting that flag) is written by the DMAC.

- **SYNCDIS: Synchronization Disable**

0: The synchronization procedure is performed in LIN slave node configuration.

1: The synchronization procedure is not performed in LIN slave node configuration.

44.10.32 USART LIN Identifier Register

Name: FLEX_US_LINIR

Address: 0xF8034258 (0), 0xF8038258 (1), 0xFC010258 (2), 0xFC014258 (3), 0xFC018258 (4)

Access: Read/Write or Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
IDCHR							

This register is relevant only if USART_MODE = 0xA or 0xB in Section 44.10.6 “USART Mode Register”.

- **IDCHR: Identifier Character**

If USART_MODE = 0xA (master node configuration):

IDCHR is Read/Write and its value is the identifier character to be transmitted.

If USART_MODE = 0xB (slave node configuration):

IDCHR is Read-only and its value is the last identifier character that has been received.

44.10.33 USART LIN Baud Rate Register

Name: FLEX_US_LINBRR

Address: 0xF803425C (0), 0xF803825C (1), 0xFC01025C (2), 0xFC01425C (3), 0xFC01825C (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	LINFP		

15	14	13	12	11	10	9	8
LINCD							

7	6	5	4	3	2	1	0
LINCD							

This register is relevant only if USART_MODE = 0xA or 0xB in Section 44.10.6 “USART Mode Register”.

Returns the baud rate value after the synchronization process completion.

- **LINCD: Clock Divider after Synchronization**
- **LINFP: Fractional Part after Synchronization**

44.10.34 USART Comparison Register

Name: FLEX_US_CMPR

Address: 0xF8034290 (0), 0xF8038290 (1), 0xFC010290 (2), 0xFC014290 (3), 0xFC018290 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	VAL2
23	22	21	20	19	18	17	16
VAL2							
15	14	13	12	11	10	9	8
–	CMPPAR	–	CMPMODE	–	–	–	VAL1
7	6	5	4	3	2	1	0
VAL1							

This register can only be written if the WPEN bit is cleared in the USART Write Protection Mode Register.

- **VAL1: First Comparison Value for Received Character**

0–511: The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in FLEX_US_CSR.

- **CMPMODE: Comparison Mode**

Value	Name	Description
0	FLAG_ONLY	Any character is received and comparison function drives CMP flag.
1	START_CONDITION	Comparison condition must be met to start reception.

- **CMPPAR: Compare Parity**

0: The parity is not checked and a bad parity cannot prevent from waking up the system.

1: The parity is checked and a matching condition on data can be cancelled by an error on parity bit, so no wakeup is performed.

- **VAL2: Second Comparison Value for Received Character**

0–511: The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in FLEX_US_CSR.

44.10.35 USART FIFO Mode Register

Name: FLEX_US_FMR

Address: 0xF80342A0 (0), 0xF80382A0 (1), 0xFC0102A0 (2), 0xFC0142A0 (3), 0xFC0182A0 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	RXFTHRES2					
23	22	21	20	19	18	17	16
–	–	RXFTHRES					
15	14	13	12	11	10	9	8
–	–	TXFTHRES					
7	6	5	4	3	2	1	0
FRTSC	–	RXRDYM		–	–	TXRDYM	

- **TXRDYM: Transmitter Ready Mode**

If FIFOs are enabled, the TXRDY flag (in FLEX_US_CSR) behaves as follows.

Value	Name	Description
0	ONE_DATA	TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO
1	TWO_DATA	TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO
2	FOUR_DATA	TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO

- **RXRDYM: Receiver Ready Mode**

If FIFOs are enabled, the RXRDY flag (in FLEX_US_CSR) behaves as follows.

Value	Name	Description
0	ONE_DATA	RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO
1	TWO_DATA	RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO
2	FOUR_DATA	RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO

- **FRTSC: FIFO RTS pin Control enable (Hardware Handshaking mode only)**

0: RTS pin is not controlled by Receive FIFO thresholds.

1: RTS pin is controlled by Receive FIFO thresholds.

See Section 44.7.3.15 “Hardware Handshaking” for details.

- **TXFTHRES: Transmit FIFO Threshold**

0–32: Defines the Transmit FIFO threshold value (number of data). TXFTHF flag in FLEX_US_FESR will be set when Transmit FIFO goes from “above” threshold state to “equal or below” threshold state.

- **RXFTHRES: Receive FIFO Threshold**

0–32: Defines the Receive FIFO threshold value (number of data). RXFTHF flag in FLEX_US_FESR will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state.

- **RXFTHRES2: Receive FIFO Threshold 2**

0–32: Defines the Receive FIFO threshold 2 value (number of data). RXFTHF2 flag in FLEX_US_FESR will be set when Receive FIFO goes from “above” threshold state to “equal or below” threshold state.

44.10.36 USART FIFO Level Register

Name: FLEX_US_FLR

Address: 0xF80342A4 (0), 0xF80382A4 (1), 0xFC0102A4 (2), 0xFC0142A4 (3), 0xFC0182A4 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	RXFL					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TXFL					

- **TXFL: Transmit FIFO Level**

0: There is no data in the Transmit FIFO

1–32: Indicates the number of DATA in the Transmit FIFO

- **RXFL: Receive FIFO Level**

0: There is no unread data in the Receive FIFO

1–32: Indicates the number of unread DATA in the Receive FIFO

44.10.37 USART FIFO Interrupt Enable Register

Name: FLEX_US_FIER

Address: 0xF80342A8 (0), 0xF80382A8 (1), 0xFC0102A8 (2), 0xFC0142A8 (3), 0xFC0182A8 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXFTHF2	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: TXFEF Interrupt Enable**
- **TXFFF: TXFFF Interrupt Enable**
- **TXFTHF: TXFTHF Interrupt Enable**
- **RXFEF: RXFEF Interrupt Enable**
- **RXFFF: RXFFF Interrupt Enable**
- **RXFTHF: RXFTHF Interrupt Enable**
- **TXFPTEF: TXFPTEF Interrupt Enable**
- **RXFPTEF: RXFPTEF Interrupt Enable**
- **RXFTHF2: RXFTHF2 Interrupt Enable**

44.10.38 USART FIFO Interrupt Disable Register

Name: FLEX_US_FIDR

Address: 0xF80342AC (0), 0xF80382AC (1), 0xFC0102AC (2), 0xFC0142AC (3), 0xFC0182AC (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXFTHF2	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: TXFEF Interrupt Disable**
- **TXFFF: TXFFF Interrupt Disable**
- **TXFTHF: TXFTHF Interrupt Disable**
- **RXFEF: RXFEF Interrupt Disable**
- **RXFFF: RXFFF Interrupt Disable**
- **RXFTHF: RXFTHF Interrupt Disable**
- **TXFPTEF: TXFPTEF Interrupt Disable**
- **RXFPTEF: RXFPTEF Interrupt Disable**
- **RXFTHF2: RXFTHF2 Interrupt Disable**

44.10.39 USART FIFO Interrupt Mask Register

Name: FLEX_US_FIMR

Address: 0xF80342B0 (0), 0xF80382B0 (1), 0xFC0102B0 (2), 0xFC0142B0 (3), 0xFC0182B0 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXFTHF2	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: TXFEF Interrupt Mask**
- **TXFFF: TXFFF Interrupt Mask**
- **TXFTHF: TXFTHF Interrupt Mask**
- **RXFEF: RXFEF Interrupt Mask**
- **RXFFF: RXFFF Interrupt Mask**
- **RXFTHF: RXFTHF Interrupt Mask**
- **TXFPTEF: TXFPTEF Interrupt Mask**
- **RXFPTEF: RXFPTEF Interrupt Mask**
- **RXFTHF2: RXFTHF2 Interrupt Mask**

44.10.40 USART FIFO Event Status Register

Name: FLEX_US_FESR

Address: 0xF80342B4 (0), 0xF80382B4 (1), 0xFC0102B4 (2), 0xFC0142B4 (3), 0xFC0182B4 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	RXFTHF2	TXFLOCK
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: Transmit FIFO Empty Flag (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Transmit FIFO is not empty.

1: Transmit FIFO has been emptied since the last RSTSTA command was issued.

- **TXFFF: Transmit FIFO Full Flag (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Transmit FIFO is not full.

1: Transmit FIFO has been filled since the last RSTSTA command was issued.

- **TXFTHF: Transmit FIFO Threshold Flag (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Number of DATA in Transmit FIFO is above TXFTHRES threshold.

1: Number of DATA in Transmit FIFO has reached TXFTHRES threshold since the last RSTSTA command was issued.

- **RXFEF: Receive FIFO Empty Flag (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Receive FIFO is not empty.

1: Receive FIFO has been emptied since the last RSTSTA command was issued.

- **RXFFF: Receive FIFO Full Flag (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Receive FIFO is not empty.

1: Receive FIFO has been filled since the last RSTSTA command was issued.

- **RXFTHF: Receive FIFO Threshold Flag (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Number of unread DATA in Receive FIFO is below RXFTHRES threshold.

1: Number of unread DATA in Receive FIFO has reached RXFTHRES threshold since the last RSTSTA command was issued.

- **TXFPTEF: Transmit FIFO Pointer Error Flag**

0: No Transmit FIFO pointer occurred

1: Transmit FIFO pointer error occurred. Transceiver must be reset

See Section 44.7.11.8 “FIFO Pointer Error” for details.

- **RXFPTEF: Receive FIFO Pointer Error Flag**

0: No Receive FIFO pointer occurred

1: Receive FIFO pointer error occurred. Receiver must be reset

See Section 44.7.11.8 “FIFO Pointer Error” for details.

- **TXFLOCK: Transmit FIFO Lock**

0: The Transmit FIFO is not locked.

1: The Transmit FIFO is locked.

- **RXFTHF2: Receive FIFO Threshold Flag 2 (cleared by writing RSTSTA bit in FLEX_US_CR)**

0: Number of unread DATA in Receive FIFO is above RXFTHRES threshold.

1: Number of unread DATA in Receive FIFO has reached RXFTHRES2 threshold since the last RSTSTA command was issued.

44.10.41 USART Write Protection Mode Register

Name: FLEX_US_WPMR

Address: 0xF80342E4 (0), 0xF80382E4 (1), 0xFC0102E4 (2), 0xFC0142E4 (3), 0xFC0182E4 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection on configuration registers if WPKEY corresponds to 0x555341 (“USA” in ASCII).

1: Enables the write protection on configuration registers if WPKEY corresponds to 0x555341 (“USA” in ASCII).

See Section 44.7.12 “USART Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x555341	PASSWD	Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0.

44.10.42 USART Write Protection Status Register

Name: FLEX_US_WPSR

Address: 0xF80342E8 (0), 0xF80382E8 (1), 0xFC0102E8 (2), 0xFC0142E8 (3), 0xFC0182E8 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of FLEX_US_WPSR.

1: A write protection violation has occurred since the last read of FLEX_US_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

44.10.43 SPI Control Register

Name: FLEX_SPI_CR

Address: 0xF8034400 (0), 0xF8038400 (1), 0xFC010400 (2), 0xFC014400 (3), 0xFC018400 (4)

Access: Write-only

31	30	29	28	27	26	25	24
FIFODIS	FIFOEN	–	–	–	–	–	LASTXFER
23	22	21	20	19	18	17	16
–	–	–	–	–	–	RXFCLR	TXFCLR
15	14	13	12	11	10	9	8
–	–	–	REQCLR	–	–	–	–
7	6	5	4	3	2	1	0
SWRST	–	–	–	–	–	SPIDIS	SPIEN

- **SPIEN: SPI Enable**

0: No effect.

1: Enables the SPI to transfer and receive data.

- **SPIDIS: SPI Disable**

0: No effect.

1: Disables the SPI.

If a transfer is in progress when SPIDIS is set, the SPI completes the transmission of the shifter register and does not start any new transfer, even if the FLEX_US_THR is loaded.

All pins are set in Input mode after completion of the transmission in progress, if any.

- **SWRST: SPI Software Reset**

0: No effect.

1: Reset the SPI. A software-triggered hardware reset of the SPI interface is performed.

The SPI is in Slave mode after software reset.

- **REQCLR: Request to Clear the Comparison Trigger**

SleepWalking enabled:

0: No effect.

1: Clears the potential clock request currently issued by SPI, thus the potential system wakeup is cancelled.

SleepWalking disabled:

0: No effect.

1: Restarts the comparison trigger to enable FLEX_SPI_RDR loading.

- **TXFCLR: Transmit FIFO Clear**

0: No effect.

1: Clears the Transmit FIFO, Transmit FIFO will become empty.

- **RXFCLR: Receive FIFO Clear**

0: No effect.

1: Clears the Receive FIFO, Receive FIFO will become empty.

- **LASTXFER: Last Transfer**

0: No effect.

1: The current NPCS will be de-asserted after the character written in TD has been transferred. When CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

Refer to Section 44.8.3.5 “Peripheral Selection” for more details.

- **FIFOEN: FIFO Enable**

0: No effect.

1: Enables the Transmit and Receive FIFOs

- **FIFODIS: FIFO Disable**

0: No effect.

1: Disables the Transmit and Receive FIFOs

44.10.44 SPI Mode Register

Name: FLEX_SPI_MR

Address: 0xF8034404 (0), 0xF8038404 (1), 0xFC010404 (2), 0xFC014404 (3), 0xFC018404 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYBCS							
23	22	21	20	19	18	17	16
–	–	–	–	–	–	PCS	
15	14	13	12	11	10	9	8
–	–	–	CMPMODE	–	–	–	–
7	6	5	4	3	2	1	0
LLB	–	WDRBT	MODFDIS	BRSRCCLK	PCSDEC	PS	MSTR

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

- **MSTR: Master/Slave Mode**

0: SPI is in Slave mode.

1: SPI is in Master mode.

- **PS: Peripheral Select**

0: Fixed Peripheral Select

1: Variable Peripheral Select

- **PCSDEC: Chip Select Decode**

0: The chip selects are directly connected to a peripheral device.

1: The two NPCS chip select lines are connected to a 2- to 4-bit decoder.

When PCSDEC equals one, up to 3 Chip Select signals can be generated with the two NPCS lines using an external 2- to 4-bit decoder. The Chip Select registers define the characteristics of the 3 chip selects, with the following rules:

FLEX_SPI_CSR0 defines peripheral chip select signals 0 to 1.

FLEX_SPI_CSR1 defines peripheral chip select signal 2.

- **BRSRCCLK: Bit Rate Source Clock**

Value	Name	Description
0	PERIPH_CLK	The peripheral clock is the source clock for the bit rate generation.
1	GCLK	GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock.

Note: If the bit BRSRCCLK = 1, the SCBR field in FLEX_US_CSRx must be programmed with a value greater than 1.

- **MODFDIS: Mode Fault Detection**

0: Mode fault detection is enabled.

1: Mode fault detection is disabled.

- **WDRBT: Wait Data Read Before Transfer**

0: No Effect. In Master mode, a transfer can be initiated regardless of the FLEX_SPI_RDR state.

1: In Master mode, a transfer can start only if FLEX_SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

- **CMPMODE: Comparison Mode**

Value	Name	Description
0	FLAG_ONLY	Any character is received and comparison function drives CMP flag.
1	START_CONDITION	Comparison condition must be met to start reception of all incoming characters until REQCLR is set.

- **LLB: Local Loopback Enable**

0: Local loopback path disabled.

1: Local loopback path enabled.

LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

- **PCS: Peripheral Chip Select**

This field is only used if fixed peripheral select is active (PS = 0).

If PCSDEC = 0:

PCS = x0 NPCS[1:0] = 10

PCS = 01 NPCS[1:0] = 01

PCS = 11 forbidden (no peripheral is selected)

(x = don’t care)

If PCSDEC = 1:

NPCS[1:0] output signals = PCS

- **DLYBCS: Delay Between Chip Selects**

This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees non-overlapping chip selects and solves bus contentions in case of peripherals having long data float times.

If DLYBCS is ≤ 6, six peripheral clock periods are inserted by default.

Otherwise, the following equations determine the delay:

If FLEX_SPI_MR.BRSRCCLK = 0: DLYBCS = Delay Between Chip Selects × $f_{peripheral\ clock}$

If FLEX_SPI_MR.BRSRCCLK = 1: DLYBCS = Delay Between Chip Selects × f_{GCLK}

44.10.45 SPI Receive Data Register

Name: FLEX_SPI_RDR

Address: 0xF8034408 (0), 0xF8038408 (1), 0xFC010408 (2), 0xFC014408 (3), 0xFC018408 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
RD							
7	6	5	4	3	2	1	0
RD							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.8.7.5 “Single Data Mode” for details.

- **RD: Receive Data**

Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

- **PCS: Peripheral Chip Select**

In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.

Note: When using Variable Peripheral Select mode (PS = 1 in FLEX_SPI_MR), it is mandatory to set the FLEX_SPI_MR.WDRBT bit to 1 if the PCS field must be processed in FLEX_SPI_RDR.

44.10.46 SPI Receive Data Register (FIFO Multiple Data, 8-bit)

Name: FLEX_SPI_RDR (FIFO_MULTI_DATA_8)

Address: 0xF8034408 (0), 0xF8038408 (1), 0xFC010408 (2), 0xFC014408 (3), 0xFC018408 (4)

Access: Read-only

31	30	29	28	27	26	25	24
RD3							
23	22	21	20	19	18	17	16
RD2							
15	14	13	12	11	10	9	8
RD1							
7	6	5	4	3	2	1	0
RD0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.8.7.6 “Multiple Data Mode” for details.

- **RDx: Receive Data**

First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

44.10.47 SPI Receive Data Register (FIFO Multiple Data, 16-bit)

Name: FLEX_SPI_RDR (FIFO_MULTI_DATA_16)

Address: 0xF8034408 (0), 0xF8038408 (1), 0xFC010408 (2), 0xFC014408 (3), 0xFC018408 (4)

Access: Read-only

31	30	29	28	27	26	25	24
RD1							
23	22	21	20	19	18	17	16
RD1							
15	14	13	12	11	10	9	8
RD0							
7	6	5	4	3	2	1	0
RD0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.8.7.6 “Multiple Data Mode” for details.

- **RDx: Receive Data**

First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

44.10.48 SPI Transmit Data Register

Name: FLEX_SPI_TDR

Address: 0xF803440C (0), 0xF803840C (1), 0xFC01040C (2), 0xFC01440C (3), 0xFC01840C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	LASTXFER

23	22	21	20	19	18	17	16
–	–	–	–	PCS			

15	14	13	12	11	10	9	8
TD							

7	6	5	4	3	2	1	0
TD							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.8.7.5 “Single Data Mode” for details.

- **TD: Transmit Data**

Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the transmit data register in a right-justified format.

- **PCS: Peripheral Chip Select**

This field is only used if variable peripheral select is active (PS = 1).

If PCSDEC = 0:

PCS = x0 NPCS[1:0] = 10

PCS = 01 NPCS[1:0] = 01

PCS = 11 forbidden (no peripheral is selected)

(x = don’t care)

If PCSDEC = 1:

NPCS[1:0] output signals = PCS

- **LASTXFER: Last Transfer**

0: No effect.

1: The current NPCS is de-asserted after the transfer of the character written in TD. When CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

This field is only used if variable peripheral select is active (PS = 1).

44.10.49 SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)

Name: FLEX_SPI_TDR (FIFO_MULTI_DATA)

Address: 0xF803440C (0), 0xF803840C (1), 0xFC01040C (2), 0xFC01440C (3), 0xFC01840C (4)

Access: Write-only

31	30	29	28	27	26	25	24
TD1							
23	22	21	20	19	18	17	16
TD1							
15	14	13	12	11	10	9	8
TD0							
7	6	5	4	3	2	1	0
TD0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section 44.8.7.6 “Multiple Data Mode” for details.

- **TDx: Transmit Data**

Next data to write in the Transmit FIFO. Information to be transmitted must be written to the transmit data register in a right-justified format.

44.10.50 SPI Status Register

Name: FLEX_SPI_SR

Address: 0xF8034410 (0), 0xF8038410 (1), 0xFC010410 (2), 0xFC014410 (3), 0xFC018410 (4)

Access: Read-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	SPIENS
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full (cleared by reading FLEX_SPI_RDR)**

When FIFOs are disabled:

0: No data has been received since the last read of FLEX_SPI_RDR.

1: Data has been received and the received data has been transferred from the shift register to FLEX_SPI_RDR since the last read of FLEX_SPI_RDR.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read

1: At least one unread data is in the Receive FIFO

RDRF behavior with FIFOs enabled is illustrated in Section 44.8.7.4 “TDRE, RDRF and TXEMPTY behavior”.

- **TDRE: Transmit Data Register Empty (cleared by writing FLEX_SPI_TDR)**

0: Data has been written to FLEX_SPI_TDR and not yet transferred to the shift register.

1: The last data written to FLEX_SPI_TDR has been transferred to the shift register.

TDRE = 0 when the SPI is disabled or at reset. The SPI enable command sets this bit to 1.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration

TDRE behavior with FIFOs enabled is illustrated in Section 44.8.7.4 “TDRE, RDRF and TXEMPTY behavior”.

- **MODF: Mode Fault Error (cleared on read)**

0: No mode fault has been detected since the last read of FLEX_SPI_SR.

1: A mode fault occurred since the last read of FLEX_SPI_SR.

- **OVRES: Overrun Error Status (cleared on read)**

0: No overrun has been detected since the last read of FLEX_SPI_SR.

1: An overrun has occurred since the last read of FLEX_SPI_SR.

An overrun occurs when FLEX_SPI_RDR is loaded at least twice from the shift register since the last read of FLEX_SPI_RDR.

- **NSSR: NSS Rising (cleared on read)**

0: No rising edge detected on NSS pin since the last read of FLEX_SPI_SR.

1: A rising edge occurred on NSS pin since the last read of FLEX_SPI_SR.

- **TXEMPTY: Transmission Registers Empty (cleared by writing FLEX_SPI_TDR)**

0: As soon as data is written in FLEX_SPI_TDR.

1: FLEX_SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay.

- **UNDES: Underrun Error Status (Slave mode Only) (cleared on read)**

0: No underrun has been detected since the last read of FLEX_SPI_SR.

1: A transfer starts whereas no data has been loaded in FLEX_SPI_TDR, cleared when FLEX_SPI_SR is read.

- **SPIENS: SPI Enable Status**

0: SPI is disabled.

1: SPI is enabled.

- **CMP: Comparison Status (cleared on read)**

0: No received character matched the comparison criteria programmed in VAL1 and VAL2 fields in FLEX_SPI_CMPR since the last read of FLEX_SPI_SR.

1: A received character matched the comparison criteria since the last read of FLEX_SPI_SR.

- **TXFEF: Transmit FIFO Empty Flag (cleared on read)**

0: Transmit FIFO is not empty.

1: Transmit FIFO has been emptied since the last read of FLEX_SPI_SR.

- **TXFFF: Transmit FIFO Full Flag (cleared on read)**

0: Transmit FIFO is not full or TXFF flag has been cleared.

1: Transmit FIFO has been filled since the last read of FLEX_SPI_SR.

- **TXFTHF: Transmit FIFO Threshold Flag (cleared on read)**

0: Number of DATA in Transmit FIFO is above TXFTHRES threshold.

1: Number of DATA in Transmit FIFO has reached TXFTHRES threshold since the last read of FLEX_SPI_SR.

- **RXFEF: Receive FIFO Empty Flag**

0: Receive FIFO is not empty or RXFE flag has been cleared.

1: Receive FIFO has become empty (coming from “not empty” state to “empty” state).

- **RXFFF: Receive FIFO Full Flag**

0: Receive FIFO is not empty or RXFE flag has been cleared.

1: Receive FIFO has become full (coming from “not full” state to “full” state).

- **RXFTHF: Receive FIFO Threshold Flag**

0: Number of unread DATA in Receive FIFO is below RXFTHRES threshold or RXFTH flag has been cleared.

1: Number of unread DATA in Receive FIFO has reached RXFTHRES threshold (coming from “below threshold” state to “equal to or above threshold” state).

- **TXFPTEF: Transmit FIFO Pointer Error Flag**

0: No Transmit FIFO pointer occurred

1: Transmit FIFO pointer error occurred. Transceiver must be reset

See Section 44.8.7.7 “FIFO Pointer Error” for details.

- **RXFPTEF: Receive FIFO Pointer Error Flag**

0: No Receive FIFO pointer occurred

1: Receive FIFO pointer error occurred. Receiver must be reset

See Section 44.8.7.7 “FIFO Pointer Error” for details.

44.10.51 SPI Interrupt Enable Register

Name: FLEX_SPI_IER

Address: 0xF8034414 (0), 0xF8038414 (1), 0xFC010414 (2), 0xFC014414 (3), 0xFC018414 (4)

Access: Write-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Enable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Enable**
- **MODF: Mode Fault Error Interrupt Enable**
- **OVRES: Overrun Error Interrupt Enable**
- **NSSR: NSS Rising Interrupt Enable**
- **TXEMPTY: Transmission Registers Empty Enable**
- **UNDES: Underrun Error Interrupt Enable**
- **CMP: Comparison Interrupt Enable**
- **TXFEF: TXFEF Interrupt Enable**
- **TXFFF: TXFFF Interrupt Enable**
- **TXFTHF: TXFTHF Interrupt Enable**
- **RXFEF: RXFEF Interrupt Enable**
- **RXFFF: RXFFF Interrupt Enable**
- **RXFTHF: RXFTHF Interrupt Enable**
- **TXFPTEF: TXFPTEF Interrupt Enable**
- **RXFPTEF: RXFPTEF Interrupt Enable**

44.10.52 SPI Interrupt Disable Register

Name: FLEX_SPI_IDR

Address: 0xF8034418 (0), 0xF8038418 (1), 0xFC010418 (2), 0xFC014418 (3), 0xFC018418 (4)

Access: Write-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Disable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Disable**
- **MODF: Mode Fault Error Interrupt Disable**
- **OVRES: Overrun Error Interrupt Disable**
- **NSSR: NSS Rising Interrupt Disable**
- **TXEMPTY: Transmission Registers Empty Disable**
- **UNDES: Underrun Error Interrupt Disable**
- **CMP: Comparison Interrupt Disable**
- **TXFEF: TXFEF Interrupt Disable**
- **TXFFF: TXFFF Interrupt Disable**
- **TXFTHF: TXFTHF Interrupt Disable**
- **RXFEF: RXFEF Interrupt Disable**
- **RXFFF: RXFFF Interrupt Disable**
- **RXFTHF: RXFTHF Interrupt Disable**
- **TXFPTEF: TXFPTEF Interrupt Disable**
- **RXFPTEF: RXFPTEF Interrupt Disable**

44.10.53 SPI Interrupt Mask Register

Name: FLEX_SPI_IMR

Address: 0xF803441C (0), 0xF803841C (1), 0xFC01041C (2), 0xFC01441C (3), 0xFC01841C (4)

Access: Read-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RDRF: Receive Data Register Full Interrupt Mask**
- **TDRE: SPI Transmit Data Register Empty Interrupt Mask**
- **MODF: Mode Fault Error Interrupt Mask**
- **OVRES: Overrun Error Interrupt Mask**
- **NSSR: NSS Rising Interrupt Mask**
- **TXEMPTY: Transmission Registers Empty Mask**
- **UNDES: Underrun Error Interrupt Mask**
- **CMP: Comparison Interrupt Mask**
- **TXFEF: TXFEF Interrupt Mask**
- **TXFFF: TXFFF Interrupt Mask**
- **TXFTHF: TXFTHF Interrupt Mask**
- **RXFEF: RXFEF Interrupt Mask**
- **RXFFF: RXFFF Interrupt Mask**
- **RXFTHF: RXFTHF Interrupt Mask**
- **TXFPTEF: TXFPTEF Interrupt Mask**
- **RXFPTEF: RXFPTEF Interrupt Mask**

44.10.54 SPI Chip Select Register

Name: FLEX_SPI_CSRx[x = 0..1]

Address: 0xF8034430 (0), 0xF8038430 (1), 0xFC010430 (2), 0xFC014430 (3), 0xFC018430 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYBCT							
23	22	21	20	19	18	17	16
DLYBS							
15	14	13	12	11	10	9	8
SCBR							
7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

Note: FLEX_SPI_CSRx must be written even if the user wants to use the default reset values. The BITS field is not updated with the translated value unless the register is written.

- **CPOL: Clock Polarity**

0: The inactive state value of SPCK is logic level zero.

1: The inactive state value of SPCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

- **NCPHA: Clock Phase**

0: Data are changed on the leading edge of SPCK and captured on the following edge of SPCK.

1: Data are captured on the leading edge of SPCK and changed on the following edge of SPCK.

NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**

0: The Peripheral Chip Select does not rise between two transfers if the FLEX_US_TDR is reloaded before the end of the first transfer and if the two transfers occur on the same Chip Select.

1: The Peripheral Chip Select rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of:

If FLEX_SPI_MR.BRSRCCLK = 0: $\frac{DLYBCS}{f_{\text{peripheral clock}}}$ (if DLYBCS ≠ 0)

If FLEX_SPI_MR.BRSRCCLK = 1: $\frac{DLYBCS}{f_{GCLK}}$

If DLYBCS < 6, a minimum of six periods is introduced.

- **CSAAT: Chip Select Active After Transfer**

0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.

1: The Peripheral Chip Select does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.

- **BITS: Bits Per Transfer**

(See the note below the FLEX_SPI_CSRx bitmap.)

The BITS field determines the number of data bits transferred. Reserved values should not be used.

Value	Name	Description
0	8_BIT	8 bits for transfer
1	9_BIT	9 bits for transfer
2	10_BIT	10 bits for transfer
3	11_BIT	11 bits for transfer
4	12_BIT	12 bits for transfer
5	13_BIT	13 bits for transfer
6	14_BIT	14 bits for transfer
7	15_BIT	15 bits for transfer
8	16_BIT	16 bits for transfer
9–15	–	Reserved

- **SCBR: Serial Clock Bit Rate**

In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the clock defined by the bit BRSRCCLK. The bit rate is selected by writing a value from 1 to 255 in the SCBR field. The following equations determine the SPCK bit rate:

If FLEX_SPI_MR.BRSRCCLK = 0: SCBR = $f_{peripheral\ clock}$ / SPCK Bit Rate

If FLEX_SPI_MR.BRSRCCLK = 1: SCBR = f_{GCLK} / SPCK Bit Rate

Programming the SCBR field to 0 is forbidden. Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

If BRSRCCLK = 1 in FLEX_SPI_MR, SCBR must be programmed with a value greater than 1.

At reset, SCBR is 0 and the user has to program it at a valid value before performing the first transfer.

Note: If one of the SCBR fields in FLEX_SPI_CSRx is set to 1, the other SCBR fields in FLEX_SPI_CSRx must be set to 1 as well, if they are used to process transfers. If they are not used to transfer data, they can be set at any value.

- **DLYBS: Delay Before SPCK**

This field defines the delay from NPCS falling edge (activation) to the first valid SPCK transition.

When DLYBS = 0, the delay is half the SPCK clock period.

Otherwise, the following equations determine the delay:

If FLEX_SPI_MR.BRSRCCLK = 0: DLYBS = Delay Before SPCK $\times$ $f_{peripheral\ clock}$

If FLEX_SPI_MR.BRSRCCLK = 1: DLYBS = Delay Before SPCK $\times$ f_{GCLK}

- **DLYBCT: Delay Between Consecutive Transfers**

This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT = 0, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

Otherwise, the following equations determine the delay:

If FLEX_SPI_MR.BRSRCCLK = 0: DLYBCT = Delay Between Consecutive Transfers $\times$ $f_{peripheral\ clock}$ / 32

If FLEX_SPI_MR.BRSRCCLK = 1: DLYBCT = Delay Between Consecutive Transfers $\times$ f_{GCLK} / 32

44.10.55 SPI FIFO Mode Register

Name: FLEX_SPI_FMR

Address: 0xF8034440 (0), 0xF8038440 (1), 0xFC010440 (2), 0xFC014440 (3), 0xFC018440 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	RXFTHRES					
23	22	21	20	19	18	17	16
–	–	TXFTHRES					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RXRDYM		–	–	TXRDYM	

- **TXRDYM: Transmitter Data Register Empty Mode**

If FIFOs are enabled, the TDRE flag (in FLEX_SPI_SR) behaves as follows.

Value	Name	Description
0x0	ONE_DATA	TDRE will be at level ‘1’ when at least one data can be written in the Transmit FIFO.
0x1	TWO_DATA	TDRE will be at level ‘1’ when at least two data can be written in the Transmit FIFO.
0x2	FOUR_DATA	TDRE will be at level ‘1’ when at least four data can be written in the Transmit FIFO.

- **RXRDYM: Receiver Data Register Full Mode**

If FIFOs are enabled, the RDRF flag (in FLEX_SPI_SR) behaves as follows.

Value	Name	Description
0x0	ONE_DATA	RDRF will be at level ‘1’ when at least one unread data is in the Receive FIFO.
0x1	TWO_DATA	RDRF will be at level ‘1’ when at least two unread data are in the Receive FIFO.
0x2	FOUR_DATA	RDRF will be at level ‘1’ when at least four unread data are in the Receive FIFO.

- **TXFTHRES: Transmit FIFO Threshold**

0–32: Defines the Transmit FIFO threshold value (number of data). TXFTH flag in FLEX_SPI_SR will be set when Transmit FIFO goes from “above” threshold state to “equal or below” threshold state.

- **RXFTHRES: Receive FIFO Threshold**

0–32: Defines the Receive FIFO threshold value (number of data). RXFTH flag in FLEX_SPI_SR will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state.

44.10.56 SPI FIFO Level Register

Name: FLEX_SPI_FLR

Address: 0xF8034444 (0), 0xF8038444 (1), 0xFC010444 (2), 0xFC014444 (3), 0xFC018444 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	RXFL					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TXFL					

- **TXFL: Transmit FIFO Level**

0: There is no data in the Transmit FIFO

1–32: Indicates the number of DATA in the Transmit FIFO

- **RXFL: Receive FIFO Level**

0: There is no unread data in the Receive FIFO

1–32: Indicates the number of unread DATA in the Receive FIFO

44.10.57 SPI Comparison Register

Name: FLEX_SPI_CMPR

Address: 0xF8034448 (0), 0xF8038448 (1), 0xFC010448 (2), 0xFC014448 (3), 0xFC018448 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
VAL2							
23	22	21	20	19	18	17	16
VAL2							
15	14	13	12	11	10	9	8
VAL1							
7	6	5	4	3	2	1	0
VAL1							

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

- **VAL1: First Comparison Value for Received Character**

0–65535: The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in FLEX_SPI_SR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if the condition is met.

- **VAL2: Second Comparison Value for Received Character**

0–65535: The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in FLEX_SPI_CSR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if condition is met.

44.10.58 SPI Write Protection Mode Register

Name: FLEX_SPI_WPMR

Address: 0xF80344E4 (0), 0xF80384E4 (1), 0xFC0104E4 (2), 0xFC0144E4 (3), 0xFC0184E4 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII)

1: Enables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII)

See Section 44.8.8 “SPI Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x535049	PASSWD	Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0

44.10.59 SPI Write Protection Status Register

Name: FLEX_SPI_WPSR

Address: 0xF80344E8 (0), 0xF80384E8 (1), 0xFC0104E8 (2), 0xFC0144E8 (3), 0xFC0184E8 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protect violation has occurred since the last read of FLEX_SPI_WPSR.

1: A write protect violation has occurred since the last read of FLEX_SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

44.10.60 TWI Control Register

Name: FLEX_TWI_CR

Address: 0xF8034600 (0), 0xF8038600 (1), 0xFC010600 (2), 0xFC014600 (3), 0xFC018600 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	FIFODIS	FIFOEN	–	LOCKCLR	–	THRCLR
23	22	21	20	19	18	17	16
–	–	–	–	–	–	ACMDIS	ACMEN
15	14	13	12	11	10	9	8
CLEAR	PECRQ	PECDIS	PECEN	SMBDIS	SMBEN	HSDIS	HSEN
7	6	5	4	3	2	1	0
SWRST	QUICK	SVDIS	SVEN	MSDIS	MSEN	STOP	START

- **START: Send a START Condition**

0: No effect.

1: A frame beginning with a START bit is transmitted according to the features defined in the TWI Master Mode Register (FLEX_TWI_MMR).

This action is necessary when the TWI peripheral needs to read data from a slave. When configured in Master mode with a write operation, a frame is sent as soon as the user writes a character in the Transmit Holding Register (FLEX_TWI_THR).

- **STOP: Send a STOP Condition**

0: No effect.

1: STOP condition is sent just after completing the current byte transmission in Master Read mode.

  – In single data byte master read, both START and STOP must be set.
  – In multiple data bytes master read, the STOP must be set after the last data received but one.
  – In Master Read mode, if a NACK bit is received, the STOP is automatically performed.
  – In master data write operation, a STOP condition will be sent after the transmission of the current data is finished.

- **MSEN: TWI Master Mode Enabled**

0: No effect.

1: Enables the Master mode (MSDIS must be written to 0).

Note: Switching from Slave to Master mode is only permitted when TXCOMP = 1.

- **MSDIS: TWI Master Mode Disabled**

0: No effect.

1: The Master mode is disabled, all pending data is transmitted. The shifter and holding characters (if it contains data) are transmitted in case of write operation. In read operation, the character being transferred must be completely received before disabling.

- **SVEN: TWI Slave Mode Enabled**

0: No effect.

1: Enables the Slave mode (SVDIS must be written to 0).

Note: Switching from Master to Slave mode is only permitted when TXCOMP = 1.

- **SVDIS: TWI Slave Mode Disabled**

0: No effect.

1: The Slave mode is disabled. The shifter and holding characters (if it contains data) are transmitted in case of read operation. In write operation, the character being transferred must be completely received before disabling.

- **QUICK: SMBus Quick Command**

0: No effect.

1: If Master mode is enabled, a SMBus Quick Command is sent.

- **SWRST: Software Reset**

0: No effect.

1: Equivalent to a system reset.

- **HSEN: TWI High-Speed Mode Enabled**

0: No effect.

1: High-speed mode enabled.

- **HSDIS: TWI High-Speed Mode Disabled**

0: No effect.

1: High-speed mode disabled.

- **SMBEN: SMBus Mode Enabled**

0: No effect.

1: If SMBDIS = 0, SMBus mode enabled.

- **SMBDIS: SMBus Mode Disabled**

0: No effect.

1: SMBus mode disabled.

- **PECEN: Packet Error Checking Enable**

0: No effect.

1: SMBus PEC (CRC) generation and check enabled.

- **PECDIS: Packet Error Checking Disable**

0: No effect.

1: SMBus PEC (CRC) generation and check disabled.

- **PECRQ: PEC Request**

0: No effect.

1: A PEC check or transmission is requested.

- **CLEAR: Bus CLEAR Command**

0: No effect.

1: If Master mode is enabled, send a bus clear command.

- **ACMEN: Alternative Command Mode Enable**

0: No effect.

1: Alternative Command mode enabled.

- **ACMDIS: Alternative Command Mode Disable**

0: No effect.

1: Alternative Command mode disabled.

- **THRCLR: Transmit Holding Register Clear**

0: No effect.

1: Clear the Transmit Holding Register and set TXRDY, TXCOMP flags.

- **LOCKCLR: Lock Clear**

0: No effect.

1: Clear the TWI FSM lock.

- **FIFOEN: FIFO Enable**

0: No effect.

1: Enable the Transmit and Receive FIFOs

- **FIFODIS: FIFO Disable**

0: No effect.

1: Disable the Transmit and Receive FIFOs

44.10.61 TWI Master Mode Register

Name: FLEX_TWI_MMR

Address: 0xF8034604 (0), 0xF8038604 (1), 0xFC010604 (2), 0xFC014604 (3), 0xFC018604 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	DADR						

15	14	13	12	11	10	9	8
–	–	–	MREAD	–	–	IADRSZ	

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **IADRSZ: Internal Device Address Size**

Value	Name	Description
0	NONE	No internal device address
1	1_BYTE	One-byte internal device address
2	2_BYTE	Two-byte internal device address
3	3_BYTE	Three-byte internal device address

- **MREAD: Master Read Direction**

0: Master write direction.

1: Master read direction.

- **DADR: Device Address**

The device address is used to access slave devices in Read or Write mode. Those bits are only used in Master mode.

44.10.62 TWI Slave Mode Register

Name: FLEX_TWI_SMR

Address: 0xF8034608 (0), 0xF8038608 (1), 0xFC010608 (2), 0xFC014608 (3), 0xFC018608 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
DATAMEN	SADR3EN	SADR2EN	SADR1EN	–	–	–	–
23	22	21	20	19	18	17	16
–	SADR						
15	14	13	12	11	10	9	8
–	MASK						
7	6	5	4	3	2	1	0
–	SCLWSDIS	–	–	SMHH	SMDA	–	NACKEN

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

- **NACKEN: Slave Receiver Data Phase NACK Enable**

0: Normal value to be returned in the ACK cycle of the data phase in Slave Receiver mode.

1: NACK value to be returned in the ACK cycle of the data phase in Slave Receiver mode.

- **SMDA: SMBus Default Address**

0: Acknowledge of the SMBus Default Address disabled.

1: Acknowledge of the SMBus Default Address enabled.

- **SMHH: SMBus Host Header**

0: Acknowledge of the SMBus Host Header disabled.

1: Acknowledge of the SMBus Host Header enabled.

- **SCLWSDIS: Clock Wait State Disable**

0: No effect.

1: Clock stretching disabled in Slave mode, OVRE and UNRE will indicate overrun and underrun.

- **MASK: Slave Address Mask**

A mask can be applied on the slave device address in Slave mode in order to allow multiple address answer. For each bit of the MASK field set to one the corresponding SADR bit will be masked.

If MASK field is set to 0 no mask is applied to SADR field.

- **SADR: Slave Address**

The slave device address is used in Slave mode in order to be accessed by master devices in Read or Write mode.

SADR must be programmed before enabling the Slave mode or after a general call. Writes at other times have no effect.

- **SADR1EN: Slave Address 1 Enable**

0: Slave address 1 matching is disabled.

1: Slave address 1 matching is enabled.

- **SADR2EN: Slave Address 2 Enable**

0: Slave address 2 matching is disabled.

1: Slave address 2 matching is enabled.

- **SADR3EN: Slave Address 3 Enable**

0: Slave address 3 matching is disabled.

1: Slave address 3 matching is enabled.

- **DATAMEN: Data Matching Enable**

0: Data matching on first received data is disabled.

1: Data matching on first received data is enabled.

44.10.63 TWI Internal Address Register

Name: FLEX_TWI_IADR

Address: 0xF803460C (0), 0xF803860C (1), 0xFC01060C (2), 0xFC01460C (3), 0xFC01860C (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
IADR							
15	14	13	12	11	10	9	8
IADR							
7	6	5	4	3	2	1	0
IADR							

- **IADR: Internal Address**

0, 1, 2 or 3 bytes depending on IADRSZ.

44.10.64 TWI Clock Waveform Generator Register

Name: FLEX_TWI_CWGR

Address: 0xF8034610 (0), 0xF8038610 (1), 0xFC010610 (2), 0xFC014610 (3), 0xFC018610 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	HOLD				

23	22	21	20	19	18	17	16
–	–	–	BRSRCCLK	–	CKDIV		

15	14	13	12	11	10	9	8
CHDIV							

7	6	5	4	3	2	1	0
CLDIV							

This register can only be written if the WPEN bit is cleared in the TWI Write Protection Mode Register.

FLEX_TWI_CWGR is only used in Master mode.

- **CLDIV: Clock Low Divider**

The SCL low period is defined as follows:

If BRSRCCLK = 0: CLDIV = $((t_{low} / t_{peripheral\ clock}) - 3) / 2^{CKDIV}$

If BRSRCCLK = 1: CLDIV = $(t_{low} / t_{ext_ck}) / 2^{CKDIV}$

- **CHDIV: Clock High Divider**

The SCL high period is defined as follows:

If BRSRCCLK = 0: CHDIV = $((t_{high} / t_{peripheral\ clock}) - 3) / 2^{CKDIV}$

If BRSRCCLK = 1: CHDIV = $(t_{high} / t_{ext_ck}) / 2^{CKDIV}$

- **CKDIV: Clock Divider**

The CKDIV is used to increase both SCL high and low periods.

- **BRSRCCLK: Bit Rate Source Clock**

Value	Name	Description
0	PERIPH_CLK	The peripheral clock is the source clock for the bit rate generation.
1	GCLK	GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock.

- **HOLD: TWD Hold Time Versus TWCK Falling**

If High-speed mode is selected TWD is internally modified on the TWCK falling edge to meet the I^2C specified maximum hold time, else if High-speed mode is not configured TWD is kept unchanged after TWCK falling edge for a period of $(HOLD + 3) \times t_{peripheral\ clock}$.

- **CKSRC: Transfer Rate Clock Source**

Value	Name	Description
0	PERIPH_CLK	The peripheral clock is used to generate the TWI bitrate.
1	GCLK	GCLK is used to generate the TWI bitrate.

44.10.65 TWI Status Register

Name: FLEX_TWI_SR

Address: 0xF8034620 (0), 0xF8038620 (1), 0xFC010620 (2), 0xFC014620 (3), 0xFC018620 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	SDA	SCL

23	22	21	20	19	18	17	16
LOCK	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK

15	14	13	12	11	10	9	8
–	–	–	–	EOSACC	SCLWS	ARBLST	NACK

7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	SVREAD	TXRDY	RXRDY	TXCOMP

- **TXCOMP: Transmission Completed (cleared by writing FLEX_TWI_THR)**

TXCOMP used in Master mode:

0: During the length of the current frame.

1: When both holding register and internal shifter are empty and STOP condition has been sent.

TXCOMP behavior in Master mode can be seen in Figure 44-87 and in Figure 44-89.

TXCOMP used in Slave mode:

0: As soon as a Start is detected.

1: After a Stop or a Repeated Start + an address different from SADR is detected.

TXCOMP behavior in Slave mode can be seen in Figure 44-121, Figure 44-122, Figure 44-123 and Figure 44-124.

- **RXRDY: Receive Holding Register Ready (cleared when reading FLEX_TWI_RHR)**

When FIFOs are disabled:

0: No character has been received since the last FLEX_TWI_RHR read operation.

1: A byte has been received in FLEX_TWI_RHR since the last read.

RXRDY behavior in Master mode can be seen in Figure 44-89.

RXRDY behavior in Slave mode can be seen in Figure 44-119, Figure 44-122, Figure 44-123 and Figure 44-124.

When FIFOs are enabled:

0: Receive FIFO is empty; no data to read

1: At least one unread data is in the Receive FIFO

RXRDY behavior with FIFO enabled is illustrated in Section "TXRDY and RXRDY Behavior" and Section "TXRDY and RXRDY Behavior".

- **TXRDY: Transmit Holding Register Ready (cleared by writing FLEX_TWI_THR)**

TXRDY used in Master mode:

0: The transmit holding register has not been transferred into the internal shifter. Set to 0 when writing into FLEX_TWI_THR.

1: As soon as a data byte is transferred from FLEX_TWI_THR to internal shifter or if a NACK error is detected, TXRDY is set at the same time as TXCOMP and NACK. TXRDY is also set when MSEN is set (enables TWI).

TXRDY behavior in Master mode can be seen in Figure 44-85, Figure 44-86 and Figure 44-87.

TXRDY used in Slave mode:

0: As soon as data is written in FLEX_TWI_THR, until this data has been transmitted and acknowledged (ACK or NACK).

1: Indicates that FLEX_TWI_THR is empty and that data has been transmitted and acknowledged.

If TXRDY is high and if a NACK has been detected, the transmission will be stopped. Thus when TRDY = NACK = 1, the user must not fill FLEX_TWI_THR to avoid losing it.

TXRDY behavior in Slave mode can be seen in Figure 44-118, Figure 44-121, Figure 44-123 and Figure 44-124.

When FIFOs are enabled:

0: Transmit FIFO is full and cannot accept more data

1: Transmit FIFO is not full; one or more data can be written according to TXRDYM field configuration

TXRDY behavior with FIFOs enabled is illustrated in Section "TXRDY and RXRDY Behavior" and Section "TXRDY and RXRDY Behavior".

- **SVREAD: Slave Read**

This bit is only used in Slave mode. When SVACC is low (no slave access has been detected) SVREAD is irrelevant.

0: Indicates that a write access is performed by a master.

1: Indicates that a read access is performed by a master.

SVREAD behavior can be seen in Figure 44-118, Figure 44-119, Figure 44-123 and Figure 44-124.

- **SVACC: Slave Access**

This bit is only used in Slave mode.

0: TWI is not addressed. SVACC is automatically cleared after a NACK or a STOP condition is detected.

1: Indicates that the address decoding sequence has matched (A master has sent SADR). SVACC remains high until a NACK or a STOP condition is detected.

SVACC behavior can be seen in Figure 44-118, Figure 44-119, Figure 44-123 and Figure 44-124.

- **GACC: General Call Access (cleared on read)**

This bit is only used in Slave mode.

0: No general call has been detected.

1: A general call has been detected. After the detection of general call, if need be, the user may acknowledge this access and decode the following bytes and respond according to the value of the bytes.

GACC behavior can be seen in Figure 44-120.

- **OVRE: Overrun Error (cleared on read)**

This bit is only used in Slave mode if clock stretching is disabled.

0: FLEX_TWI_RHR has not been loaded while RXRDY was set.

1: FLEX_TWI_RHR has been loaded while RXRDY was set. Reset by read in FLEX_TWI_SR when TXCOMP is set.

- **UNRE: Underrun Error (cleared on read)**

This bit is only used in Slave mode if clock stretching is disabled.

0: FLEX_TWI_THR has been filled on time.

1: FLEX_TWI_THR has not been filled on time.

- **NACK: Not Acknowledged (cleared on read)**

NACK used in Master mode:

0: Each data byte has been correctly received by the far-end side TWI slave component.

1: A data or address byte has not been acknowledged by the slave component. Set at the same time as TXCOMP.

NACK used in Slave Read mode:

0: Each data byte has been correctly received by the master.

1: In Read mode, a data byte has not been acknowledged by the master. When NACK is set the user must not fill FLEX_TWI_THR even if TXRDY is set, because it means that the master will stop the data transfer or re initiate it.

Note that in Slave Write mode all data are acknowledged by the TWI.

- **ARBLST: Arbitration Lost (cleared on read)**

This bit is only used in Master mode.

0: Arbitration won.

1: Arbitration lost. Another master of the TWI bus has won the multi-master arbitration. TXCOMP is set at the same time.

- **SCLWS: Clock Wait State**

This bit is only used in Slave mode.

0: The clock is not stretched.

1: The clock is stretched. FLEX_TWI_THR / FLEX_TWI_RHR buffer is not filled / emptied before the transmission / reception of a new character.

SCLWS behavior can be seen in Figure 44-121 and Figure 44-122.

- **EOSACC: End Of Slave Access (cleared on read)**

This bit is only used in Slave mode.

0: A slave access is being performing.

1: The Slave Access is finished. End Of Slave Access is automatically set as soon as SVACC is reset.

EOSACC behavior can be seen in Figure 44-123 and Figure 44-124.

- **MCACK: Master Code Acknowledge (cleared on read)**

MACK used in Slave mode:

0: No master code has been received.

1: A master code has been received.

- **TOUT: Timeout Error (cleared on read)**

0: No SMBus timeout occurred.

1: SMBus timeout occurred.

- **PECERR: PEC Error (cleared on read)**

0: No SMBus PEC error occurred.

1: A SMBus PEC error occurred.

- **SMBDAM: SMBus Default Address Match (cleared on read)**

0: No SMBus Default Address received.

1: A SMBus Default Address was received.

- **SMBHHM: SMBus Host Header Address Match (cleared on read)**

0: No SMBus Host Header Address received.

1: A SMBus Host Header Address was received.

- **LOCK: TWI Lock Due to Frame Errors**

0: The TWI is not locked.

1: The TWI is locked due to frame errors (see Section 44.9.3.12 “Handling Errors in Alternative Command” and Section 44.9.3.14 “FIFOs”).

- **SCL: SCL line value**

0: SCL line sampled value is ‘0’.

1: SCL line sampled value is ‘1.’

- **SDA: SDA line value**

0: SDA line sampled value is ‘0’.

1: SDA line sampled value is ‘1’.

44.10.66 TWI Interrupt Enable Register

Name: FLEX_TWI_IER

Address: 0xF8034624 (0), 0xF8038624 (1), 0xFC010624 (2), 0xFC014624 (3), 0xFC018624 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK

15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK

7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Enable**
- **RXRDY: Receive Holding Register Ready Interrupt Enable**
- **TXRDY: Transmit Holding Register Ready Interrupt Enable**
- **SVACC: Slave Access Interrupt Enable**
- **GACC: General Call Access Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**
- **UNRE: Underrun Error Interrupt Enable**
- **NACK: Not Acknowledge Interrupt Enable**
- **ARBLST: Arbitration Lost Interrupt Enable**
- **SCL_WS: Clock Wait State Interrupt Enable**
- **EOSACC: End Of Slave Access Interrupt Enable**
- **ENDRX: End of Receive Buffer Interrupt Enable**
- **ENDTX: End of Transmit Buffer Interrupt Enable**
- **RXBUFF: Receive Buffer Full Interrupt Enable**
- **TXBUFE: Transmit Buffer Empty Interrupt Enable**
- **MCACK: Master Code Acknowledge Interrupt Enable**

- **TOUT: Timeout Error Interrupt Enable**
- **PECERR: PEC Error Interrupt Enable**
- **SMBDAM: SMBus Default Address Match Interrupt Enable**
- **SMBHHM: SMBus Host Header Address Match Interrupt Enable**

44.10.67 TWI Interrupt Disable Register

Name: FLEX_TWI_IDR

Address: 0xF8034628 (0), 0xF8038628 (1), 0xFC010628 (2), 0xFC014628 (3), 0xFC018628 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK
15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK
7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **TXCOMP: Transmission Completed Interrupt Disable**
- **RXRDY: Receive Holding Register Ready Interrupt Disable**
- **TXRDY: Transmit Holding Register Ready Interrupt Disable**
- **SVACC: Slave Access Interrupt Disable**
- **GACC: General Call Access Interrupt Disable**
- **OVRE: Overrun Error Interrupt Disable**
- **UNRE: Underrun Error Interrupt Disable**
- **NACK: Not Acknowledge Interrupt Disable**
- **ARBLST: Arbitration Lost Interrupt Disable**
- **SCL_WS: Clock Wait State Interrupt Disable**
- **EOSACC: End Of Slave Access Interrupt Disable**
- **ENDRX: End of Receive Buffer Interrupt Disable**
- **ENDTX: End of Transmit Buffer Interrupt Disable**
- **RXBUFF: Receive Buffer Full Interrupt Disable**
- **TXBUFE: Transmit Buffer Empty Interrupt Disable**
- **MCACK: Master Code Acknowledge Interrupt Disable**

- **TOUT: Timeout Error Interrupt Disable**
- **PECERR: PEC Error Interrupt Disable**
- **SMBDAM: SMBus Default Address Match Interrupt Disable**
- **SMBHHM: SMBus Host Header Address Match Interrupt Disable**

44.10.68 TWI Interrupt Mask Register

Name: FLEX_TWI_IMR

Address: 0xF803462C (0), 0xF803862C (1), 0xFC01062C (2), 0xFC01462C (3), 0xFC01862C (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	SMBHHM	SMBDAM	PECERR	TOUT	–	MCACK

15	14	13	12	11	10	9	8
TXBUFE	RXBUFF	ENDTX	ENDRX	EOSACC	SCL_WS	ARBLST	NACK

7	6	5	4	3	2	1	0
UNRE	OVRE	GACC	SVACC	–	TXRDY	RXRDY	TXCOMP

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

- **TXCOMP: Transmission Completed Interrupt Mask**
- **RXRDY: Receive Holding Register Ready Interrupt Mask**
- **TXRDY: Transmit Holding Register Ready Interrupt Mask**
- **SVACC: Slave Access Interrupt Mask**
- **GACC: General Call Access Interrupt Mask**
- **OVRE: Overrun Error Interrupt Mask**
- **UNRE: Underrun Error Interrupt Mask**
- **NACK: Not Acknowledge Interrupt Mask**
- **ARBLST: Arbitration Lost Interrupt Mask**
- **SCL_WS: Clock Wait State Interrupt Mask**
- **EOSACC: End Of Slave Access Interrupt Mask**
- **ENDRX: End of Receive Buffer Interrupt Mask**
- **ENDTX: End of Transmit Buffer Interrupt Mask**
- **RXBUFF: Receive Buffer Full Interrupt Mask**
- **TXBUFE: Transmit Buffer Empty Interrupt Mask**
- **MCACK: Master Code Acknowledge Interrupt Mask**

- **TOUT: Timeout Error Interrupt Mask**
- **PECERR: PEC Error Interrupt Mask**
- **SMBDAM: SMBus Default Address Match Interrupt Mask**
- **SMBHHM: SMBus Host Header Address Match Interrupt Mask**

44.10.69 TWI Receive Holding Register

Name: FLEX_TWI_RHR

Address: 0xF8034630 (0), 0xF8038630 (1), 0xFC010630 (2), 0xFC014630 (3), 0xFC018630 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXDATA							

- **RXDATA: Master or Slave Receive Holding Data**

44.10.70 TWI Receive Holding Register (FIFO Enabled)

Name: FLEX_TWI_RHR (FIFO_ENABLED)

Address: 0xF8034630 (0), 0xF8038630 (1), 0xFC010630 (2), 0xFC014630 (3), 0xFC018630 (4)

Access: Read-only

31	30	29	28	27	26	25	24
RXDATA3							
23	22	21	20	19	18	17	16
RXDATA2							
15	14	13	12	11	10	9	8
RXDATA1							
7	6	5	4	3	2	1	0
RXDATA0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section "Master Multiple Data Mode" and Section "Slave Multiple Data Mode" for details.

- **RXDATA0: Master or Slave Receive Holding Data 0**
- **RXDATA1: Master or Slave Receive Holding Data 1**
- **RXDATA2: Master or Slave Receive Holding Data 2**
- **RXDATA3: Master or Slave Receive Holding Data 3**

44.10.71 TWI Transmit Holding Register

Name: FLEX_TWI_THR

Address: 0xF8034634 (0), 0xF8038634 (1), 0xFC010634 (2), 0xFC014634 (3), 0xFC018634 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
TXDATA							

- **TXDATA: Master or Slave Transmit Holding Data**

44.10.72 TWI Transmit Holding Register (FIFO Enabled)

Name: FLEX_TWI_THR (FIFO_ENABLED)

Address: 0xF8034634 (0), 0xF8038634 (1), 0xFC010634 (2), 0xFC014634 (3), 0xFC018634 (4)

Access: Write-only

31	30	29	28	27	26	25	24
TXDATA3							
23	22	21	20	19	18	17	16
TXDATA2							
15	14	13	12	11	10	9	8
TXDATA1							
7	6	5	4	3	2	1	0
TXDATA0							

Note: If FIFO is enabled (FIFOEN bit in FLEX_US_CR), refer to Section "Master Multiple Data Mode" and Section "Slave Multiple Data Mode" for details.

- **TXDATA0: Master or Slave Transmit Holding Data 0**
- **TXDATA1: Master or Slave Transmit Holding Data 1**
- **TXDATA2: Master or Slave Transmit Holding Data 2**
- **TXDATA3: Master or Slave Transmit Holding Data 3**

44.10.73 TWI SMBus Timing Register

Name: FLEX_TWI_SMBTR

Address: 0xF8034638 (0), 0xF8038638 (1), 0xFC010638 (2), 0xFC014638 (3), 0xFC018638 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
THMAX							

23	22	21	20	19	18	17	16
TLOWM							

15	14	13	12	11	10	9	8
TLOWS							

7	6	5	4	3	2	1	0
–	–	–	–	PRESC			

- **PRESC: SMBus Clock Prescaler**

Used to specify how to prescale the TLOWS, TLOWM and THMAX counters in SMBTR. Counters are prescaled according to the following formula: PRESC = Log(fMCK / $f_{Prescaled}$) / Log(2) - 1

- **TLOWS: Slave Clock Stretch Maximum Cycles**

0: TLOW:SEXT timeout check disabled.

1–255: Clock cycles in slave maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:SEXT.

- **TLOWM: Master Clock Stretch Maximum Cycles**

0: TLOW:MEXT timeout check disabled.

1–255: Clock cycles in master maximum clock stretch count. Prescaled by PRESC. Used to time TLOW:MEXT.

- **THMAX: Clock High Maximum Cycles**

Clock cycles in clock high maximum count. Prescaled by PRESC. Used for bus free detection. Used to time THIGH:MAX.

44.10.74 TWI Alternative Command Register

Name: FLEX_TWI_ACR

Address: 0xF8034640 (0), 0xF8038640 (1), 0xFC010640 (2), 0xFC014640 (3), 0xFC018640 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	NPEC	NDIR

23	22	21	20	19	18	17	16
NDATAL							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	PEC	DIR

7	6	5	4	3	2	1	0
DATAL							

- **DATAL: Data Length**

0: No data to send (see Section 44.9.3.11 “Alternative Command”).

1–255: Number of bytes to send during the transfer.

- **DIR: Transfer Direction**

0: Write direction.

1: Read direction.

- **PEC: PEC Request (SMBus Mode only)**

0: The transfer does not use a PEC byte.

1: The transfer uses a PEC byte.

- **NDATAL: Next Data Length**

0: No data to send (see Section 44.9.3.11 “Alternative Command”).

1–255: Number of bytes to send for the next transfer.

- **NDIR: Next Transfer Direction**

0: Write direction.

1: Read direction.

- **NPEC: Next PEC Request (SMBus Mode only)**

0: The next transfer does not use a PEC byte.

1: The next transfer uses a PEC byte.

44.10.75 TWI Filter Register

Name: FLEX_TWI_FILTR

Address: 0xF8034644 (0), 0xF8038644 (1), 0xFC010644 (2), 0xFC014644 (3), 0xFC018644 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	THRES		
7	6	5	4	3	2	1	0
–	–	–	–	–	PADFCFG	PADFEN	FILT

- **FILT: RX Digital Filter**

0: No filtering applied on TWI inputs.

1: TWI input filtering is active. (Only in Standard and Fast modes)

Note: TWI digital input filtering follows a majority decision based on three samples from SDA/SCL lines at peripheral clock frequency.

- **PADFEN: PAD Filter Enable**

0: PAD analog filter is disabled.

1: PAD analog filter is enabled. (The analog filter must be enabled if High-speed mode is enabled.)

- **PADFCFG: PAD Filter Config**

See the electrical characteristics section for filter configuration details.

- **THRES: Digital Filter Threshold**

0: No filtering applied on TWI inputs.

1–7: Maximum pulse width of spikes which will be suppressed by the input filter, defined in peripheral clock cycles.

44.10.76 TWI SleepWalking Matching Register

Name: FLEX_TWI_SWMR

Address: 0xF803464C (0), 0xF803864C (1), 0xFC01064C (2), 0xFC01464C (3), 0xFC01864C (4)

Access: Read/Write

31	30	29	28	27	26	25	24
DATAM							

23	22	21	20	19	18	17	16
–	SADR3						

15	14	13	12	11	10	9	8
–	SADR2						

7	6	5	4	3	2	1	0
–	SADR1						

- **SADR1: Slave Address 1**

Slave address 1. The TWI module will match on this additional address if SADR1EN bit is enabled.

- **SADR2: Slave Address 2**

Slave address 2. The TWI module will match on this additional address if SADR2EN bit is enabled.

- **SADR3: Slave Address 3**

Slave address 3. The TWI module will match on this additional address if SADR3EN bit is enabled.

- **DATAM: Data Match**

The TWI module will extend the SleepWalking matching process to the first received data comparing it with DATAM if DATAMEN bit is enabled.

44.10.77 TWI FIFO Mode Register

Name: FLEX_TWI_FMR

Address: 0xF8034650 (0), 0xF8038650 (1), 0xFC010650 (2), 0xFC014650 (3), 0xFC018650 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	RXFTHRES					
23	22	21	20	19	18	17	16
–	–	TXFTHRES					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RXRDYM		–	–	TXRDYM	

- **TXRDYM: Transmitter Ready Mode**

If FIFOs are enabled, the TXRDY flag (in FLEX_TWI_SR) behaves as follows.

Value	Name	Description
0x0	ONE_DATA	TXRDY will be at level ‘1’ when at least one data can be written in the Transmit FIFO
0x1	TWO_DATA	TXRDY will be at level ‘1’ when at least two data can be written in the Transmit FIFO
0x2	FOUR_DATA	TXRDY will be at level ‘1’ when at least four data can be written in the Transmit FIFO

- **RXRDYM: Receiver Ready Mode**

If FIFOs are enabled, the RXRDY flag (in FLEX_TWI_SR) behaves as follows.

Value	Name	Description
0x0	ONE_DATA	RXRDY will be at level ‘1’ when at least one unread data is in the Receive FIFO
0x1	TWO_DATA	RXRDY will be at level ‘1’ when at least two unread data are in the Receive FIFO
0x2	FOUR_DATA	RXRDY will be at level ‘1’ when at least four unread data are in the Receive FIFO

- **TXFTHRES: Transmit FIFO Threshold**

0–16: Defines the Transmit FIFO threshold value (number of data). TXFTH flag in FLEX_TWI_FSR will be set when Transmit FIFO goes from “above” threshold state to “equal or below” threshold state.

- **RXFTHRES: Receive FIFO Threshold**

0–16: Defines the Receive FIFO threshold value (number of data). RXFTH flag in FLEX_TWI_FSR will be set when Receive FIFO goes from “below” threshold state to “equal to or above” threshold state.

44.10.78 TWI FIFO Level Register

Name: FLEX_TWI_FLR

Address: 0xF8034654 (0), 0xF8038654 (1), 0xFC010654 (2), 0xFC014654 (3), 0xFC018654 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	RXFL					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TXFL					

- **TXFL: Transmit FIFO Level**

0: There is no data in the Transmit FIFO

1–16: Indicates the number of DATA in the Transmit FIFO

- **RXFL: Receive FIFO Level**

0: There is no unread data in the Receive FIFO

1–16: Indicates the number of unread DATA in the Receive FIFO

44.10.79 TWI FIFO Status Register

Name: FLEX_TWI_FSR

Address: 0xF8034660 (0), 0xF8038660 (1), 0xFC010660 (2), 0xFC014660 (3), 0xFC018660 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: Transmit FIFO Empty Flag (cleared on read)**

0: Transmit FIFO is not empty.

1: Transmit FIFO has been emptied since the last read of FLEX_TWI_FSR.

- **TXFFF: Transmit FIFO Full Flag (cleared on read)**

0: Transmit FIFO is not full.

1: Transmit FIFO has been filled since the last read of FLEX_TWI_FSR.

- **TXFTHF: Transmit FIFO Threshold Flag (cleared on read)**

0: Number of DATA in Transmit FIFO is above TXFTHRES threshold.

1: Number of DATA in Transmit FIFO has reached TXFTHRES threshold since the last read of FLEX_TWI_FSR.

- **RXFEF: Receive FIFO Empty Flag**

0: Receive FIFO is not empty.

1: Receive FIFO has been emptied since the last read of FLEX_TWI_FSR.

- **RXFFF: Receive FIFO Full Flag**

0: Receive FIFO is not empty.

1: Receive FIFO has been filled since the last read of FLEX_TWI_FSR.

- **RXFTHF: Receive FIFO Threshold Flag**

0: Number of unread DATA in Receive FIFO is below RXFTHRES threshold.

1: Number of unread DATA in Receive FIFO has reached RXFTHRES threshold since the last read of FLEX_TWI_FSR.

- **TXFPTEF: Transmit FIFO Pointer Error Flag**

0: No Transmit FIFO pointer occurred

1: Transmit FIFO pointer error occurred. Transceiver must be reset

See Section "FIFO Pointer Error" and Section "FIFO Pointer Error" for details.

- **RXFPTEF: Receive FIFO Pointer Error Flag**

0: No Receive FIFO pointer occurred

1: Receive FIFO pointer error occurred. Receiver must be reset

See Section “FIFO Pointer Error” and Section “FIFO Pointer Error” for details.

44.10.80 TWI FIFO Interrupt Enable Register

Name: FLEX_TWI_FIER

Address: 0xF8034664 (0), 0xF8038664 (1), 0xFC010664 (2), 0xFC014664 (3), 0xFC018664 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: TXFEF Interrupt Enable**
- **TXFFF: TXFFF Interrupt Enable**
- **TXFTHF: TXFTHF Interrupt Enable**
- **RXFEF: RXFEF Interrupt Enable**
- **RXFFF: RXFFF Interrupt Enable**
- **RXFTHF: RXFTHF Interrupt Enable**
- **TXFPTEF: TXFPTEF Interrupt Enable**
- **RXFPTEF: RXFPTEF Interrupt Enable**

44.10.81 TWI FIFO Interrupt Disable Register

Name: FLEX_TWI_FIDR

Address: 0xF8034668 (0), 0xF8038668 (1), 0xFC010668 (2), 0xFC014668 (3), 0xFC018668 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: TXFEF Interrupt Disable**
- **TXFFF: TXFFF Interrupt Disable**
- **TXFTHF: TXFTHF Interrupt Disable**
- **RXFEF: RXFEF Interrupt Disable**
- **RXFFF: RXFFF Interrupt Disable**
- **RXFTHF: RXFTHF Interrupt Disable**
- **TXFPTEF: TXFPTEF Interrupt Disable**
- **RXFPTEF: RXFPTEF Interrupt Disable**

44.10.82 TWI FIFO Interrupt Mask Register

Name: FLEX_TWI_FIMR

Address: 0xF803466C (0), 0xF803866C (1), 0xFC01066C (2), 0xFC01466C (3), 0xFC01866C (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF

- **TXFEF: TXFEF Interrupt Mask**
- **TXFFF: TXFFF Interrupt Mask**
- **TXFTHF: TXFTHF Interrupt Mask**
- **RXFEF: RXFEF Interrupt Mask**
- **RXFFF: RXFFF Interrupt Mask**
- **RXFTHF: RXFTHF Interrupt Mask**
- **TXFPTEF: TXFPTEF Interrupt Mask**
- **RXFPTEF: RXFPTEF Interrupt Mask**

44.10.83 TWI Write Protection Mode Register

Name: FLEX_TWI_WPMR

Address: 0xF80346E4 (0), 0xF80386E4 (1), 0xFC0106E4 (2), 0xFC0146E4 (3), 0xFC0186E4 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x545749 (“TWI” in ASCII).

See Section 44.9.7 “TWI Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x545749	PASSWD	Writing any other value in this field aborts the write operation of bit WPEN. Always reads as 0

44.10.84 TWI Write Protection Status Register

Name: FLEX_TWI_WPSR

Address: 0xF80346E8 (0), 0xF80386E8 (1), 0xFC0106E8 (2), 0xFC0146E8 (3), 0xFC0186E8 (4)

Access: Read-only

31	30	29	28	27	26	25	24
WPVSRC							

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
WPVSRC							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protect Violation Status**

0: No Write Protection Violation has occurred since the last read of FLEX_TWI_WPSR.

1: A Write Protection Violation has occurred since the last read of FLEX_TWI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

45. Universal Asynchronous Receiver Transmitter (UART)

45.1 Description

The Universal Asynchronous Receiver Transmitter (UART) features a two-pin UART that can be used for communication and trace purposes and offers an ideal medium for in-situ programming solutions.

Moreover, the association with a DMA controller permits packet handling for these tasks with processor time reduced to a minimum.

45.2 Embedded Characteristics

- Two-pin UART
  - Independent Receiver and Transmitter with a Common Programmable Baud Rate Generator
  - Baud Rate can be Driven by Processor-Independent Generic Source Clock
  - Even, Odd, Mark or Space Parity Generation
  - Parity, Framing and Overrun Error Detection
  - Automatic Echo, Local Loopback and Remote Loopback Channel Modes
  - Digital Filter on Receive Line
  - Interrupt Generation
  - Support for Two DMA Channels with Connection to Receiver and Transmitter
  - Supports Asynchronous Partial Wakeup on Receive Line Activity (SleepWalking)
  - Comparison Function on Received Character
  - Receiver Timeout
  - Register Write Protection

45.3 Block Diagram

Figure 45-1. UART Block Diagram

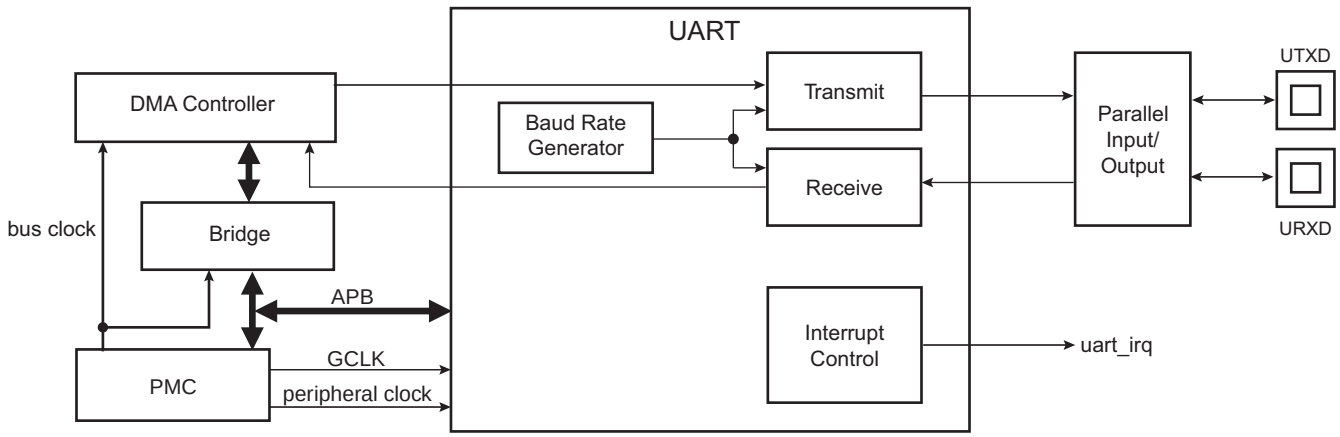


Table 45-1. UART Pin Description

Pin Name	Description	Type
URXD	UART Receive Data	Input
UTXD	UART Transmit Data	Output

45.4 Product Dependencies

45.4.1 I/O Lines

The UART pins are multiplexed with PIO lines. The user must first configure the corresponding PIO Controller to enable I/O line operations of the UART.

Table 45-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
UART0	URXD0	PB26	C
UART0	UTXD0	PB27	C
UART1	URXD1	PC7	E
UART1	URXD1	PD2	A
UART1	UTXD1	PC8	E
UART1	UTXD1	PD3	A
UART2	URXD2	PD4	B
UART2	URXD2	PD19	C
UART2	URXD2	PD23	A
UART2	UTXD2	PD5	B
UART2	UTXD2	PD20	C
UART2	UTXD2	PD24	A
UART3	URXD3	PB11	C
UART3	URXD3	PC12	D
UART3	URXD3	PC31	C
UART3	UTXD3	PB12	C
UART3	UTXD3	PC13	D
UART3	UTXD3	PD0	C
UART4	URXD4	PB3	A
UART4	UTXD4	PB4	A

45.4.2 Power Management

The UART clock can be controlled through the Power Management Controller (PMC). In this case, the user must first configure the PMC to enable the UART clock. Usually, the peripheral identifier used for this purpose is 1.

In SleepWalking mode (asynchronous partial wakeup), the PMC must be configured to enable SleepWalking for the UART in the Sleepwalking Enable Register (PMC_SLPWK_ER). Depending on the instructions (requests) provided by the UART to the PMC, the system clock may or may not be automatically provided to the UART.

45.4.3 Interrupt Sources

The UART interrupt line is connected to one of the interrupt sources of the Interrupt Controller. Interrupt handling requires programming of the Interrupt Controller before configuring the UART.

Table 45-3. Peripheral IDs

Instance	ID
UART0	24
UART1	25
UART2	26
UART3	27
UART4	28

45.5 Functional Description

The UART operates in Asynchronous mode only and supports only 8-bit character handling (with parity). It has no clock pin.

The UART is made up of a receiver and a transmitter that operate independently, and a common baud rate generator. Receiver timeout and transmitter time guard are not implemented. However, all the implemented features are compatible with those of a standard USART.

45.5.1 Baud Rate Generator

The baud rate generator provides the bit period clock named baud rate clock to both the receiver and the transmitter.

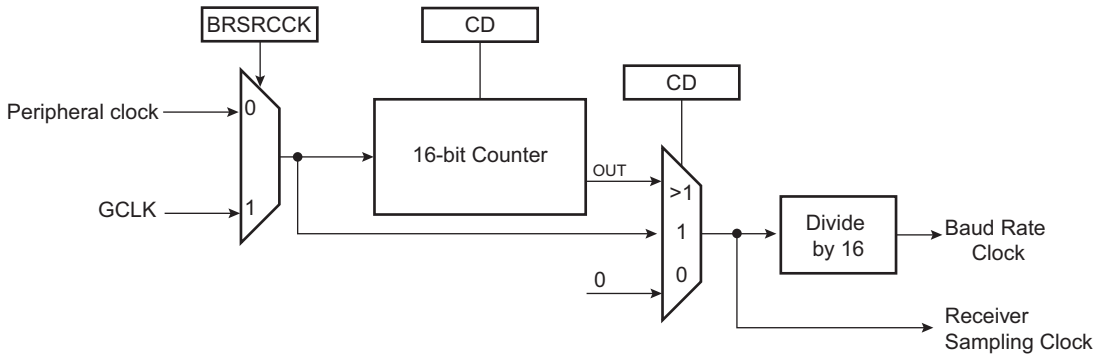
The baud rate clock is the peripheral clock divided by 16 times the clock divisor (CD) value written in the Baud Rate Generator register (UART_BRGR). If UART_BRGR is set to 0, the baud rate clock is disabled and the UART remains inactive. The maximum allowable baud rate is peripheral clock or GCLK divided by 16. The minimum allowable baud rate is peripheral clock divided by (16 x 65536). The clock source driving the baud rate generator (peripheral clock or GCLK) can be selected by writing the bit BRSRCCK in UART_MR.

If GCLK is selected, the baud rate is independent of the processor/bus clock. Thus the processor clock can be changed while UART is enabled. The processor clock frequency changes must be performed only by programming the field PRES in PMC_MCKR (see PMC section). Other methods to modify the processor/bus clock frequency (PLL multiplier, etc.) are forbidden when UART is enabled.

The peripheral clock frequency must be at least three times higher than GCLK.

Figure 45-2.

Figure 45-3. Baud Rate Generator



45.5.2 Receiver

45.5.2.1 Receiver Reset, Enable and Disable

After device reset, the UART receiver is disabled and must be enabled before being used. The receiver can be enabled by writing the Control Register (UART_CR) with the bit RXEN at 1. At this command, the receiver starts looking for a start bit.

The programmer can disable the receiver by writing UART_CR with the bit RXDIS at 1. If the receiver is waiting for a start bit, it is immediately stopped. However, if the receiver has already detected a start bit and is receiving the data, it waits for the stop bit before actually stopping its operation.

The receiver can be put in reset state by writing UART_CR with the bit RSTRX at 1. In this case, the receiver immediately stops its current operations and is disabled, whatever its current state. If RSTRX is applied when data is being processed, this data is lost.

45.5.2.2 Start Detection and Data Sampling

The UART only supports asynchronous operations, and this affects only its receiver. The UART receiver detects the start of a received character by sampling the URXD signal until it detects a valid start bit. A low level (space) on URXD is interpreted as a valid start bit if it is detected for more than seven cycles of the sampling clock, which is 16 times the baud rate. Hence, a space that is longer than 7/16 of the bit period is detected as a valid start bit. A space which is 7/16 of a bit period or shorter is ignored and the receiver continues to wait for a valid start bit.

When a valid start bit has been detected, the receiver samples the URXD at the theoretical midpoint of each bit. It is assumed that each bit lasts 16 cycles of the sampling clock (1-bit period) so the bit sampling point is eight cycles (0.5-bit period) after the start of the bit. The first sampling point is therefore 24 cycles (1.5-bit periods) after detecting the falling edge of the start bit.

Each subsequent bit is sampled 16 cycles (1-bit period) after the previous one.

Figure 45-4. Start Bit Detection

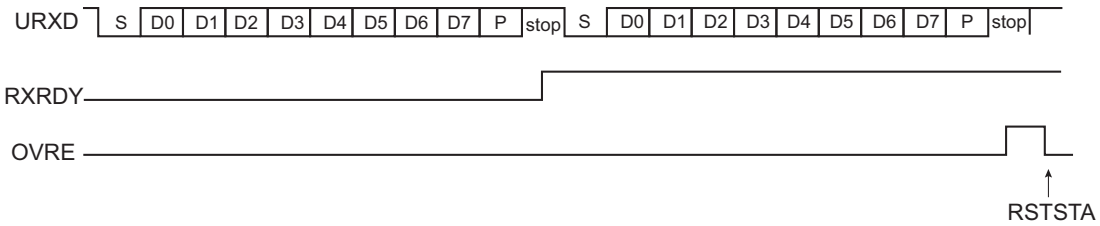
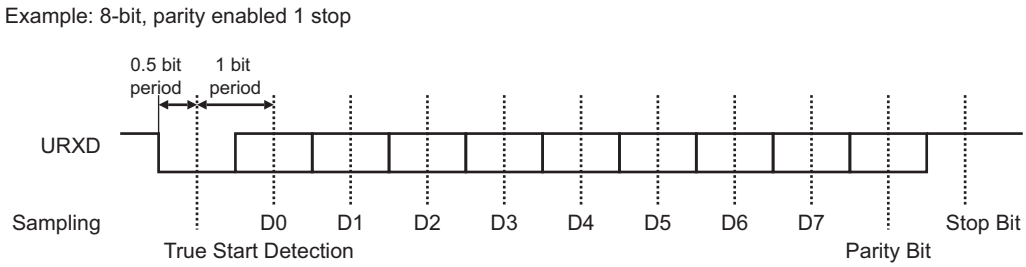


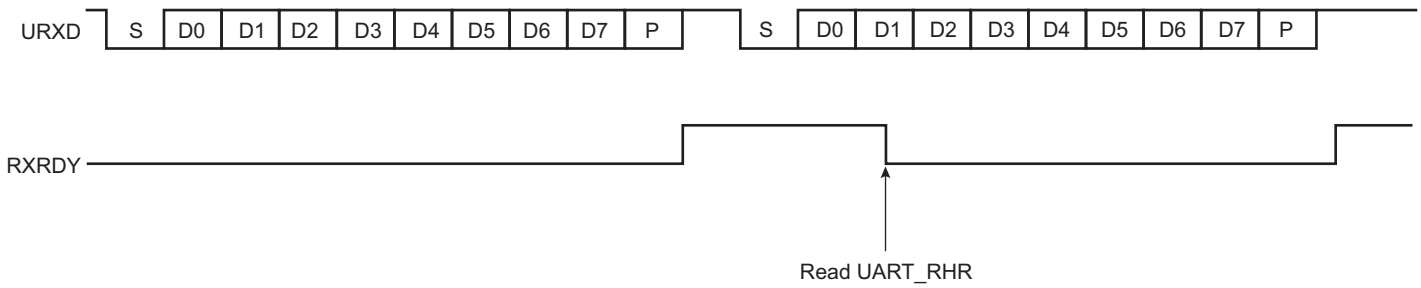
Figure 45-5. Character Reception



45.5.2.3 Receiver Ready

When a complete character is received, it is transferred to the Receive Holding Register (UART_RHR) and the RXRDY status bit in the Status Register (UART_SR) is set. The bit RXRDY is automatically cleared when UART_RHR is read.

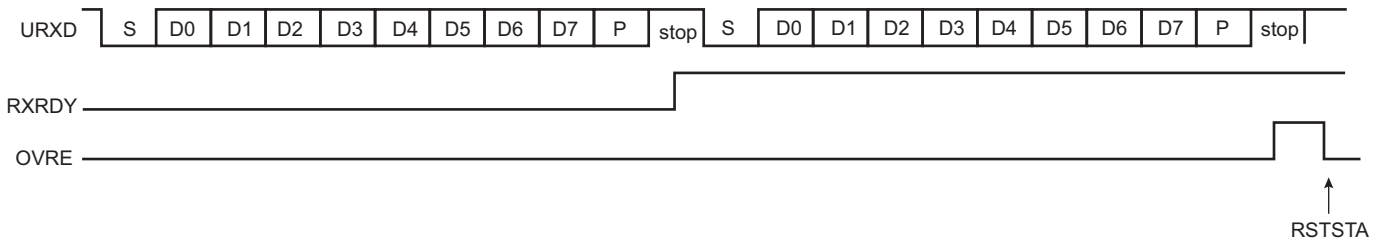
Figure 45-6. Receiver Ready



45.5.2.4 Receiver Overrun

The OVRE status bit in UART_SR is set if UART_RHR has not been read by the software (or the DMA Controller) since the last transfer, the RXRDY bit is still set and a new character is received. OVRE is cleared when the software writes a 1 to the bit RSTSTA (Reset Status) in UART_CR.

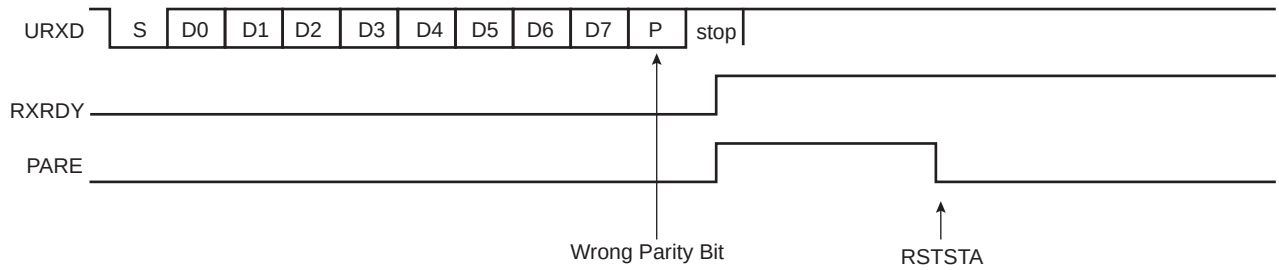
Figure 45-7. Receiver Overrun



45.5.2.5 Parity Error

Each time a character is received, the receiver calculates the parity of the received data bits, in accordance with the field PAR in the Mode Register (UART_MR). It then compares the result with the received parity bit. If different, the parity error bit PARE in UART_SR is set at the same time RXRDY is set. The parity bit is cleared when UART_CR is written with the bit RSTSTA (Reset Status) at 1. If a new character is received before the reset status command is written, the PARE bit remains at 1.

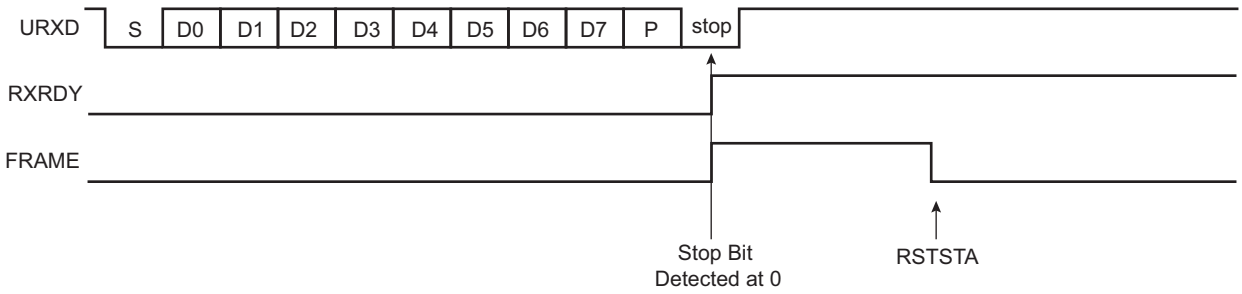
Figure 45-8. Parity Error



45.5.2.6 Receiver Framing Error

When a start bit is detected, it generates a character reception when all the data bits have been sampled. The stop bit is also sampled and when it is detected at 0, the FRAME (Framing Error) bit in UART_SR is set at the same time the RXRDY bit is set. The FRAME bit remains high until the Control Register (UART_CR) is written with the bit RSTSTA at 1.

Figure 45-9. Receiver Framing Error



45.5.2.7 Receiver Digital Filter

The UART embeds a digital filter on the receive line. It is disabled by default and can be enabled by writing a logical 1 in the FILTER bit of UART_MR. When enabled, the receive line is sampled using the 16x bit clock and a three-sample filter (majority 2 over 3) determines the value of the line.

45.5.2.8 Receiver Timeout

The Receiver Timeout provides support in handling variable-length frames. This feature detects an idle condition on the RXD line. When a timeout is detected, the bit TIMEOUT in the UART_SR rises and can generate an interrupt, thus indicating to the driver an end of frame.

The timeout delay period (during which the receiver waits for a new character) is programmed in the TO field of the Receiver Timeout register (UART_RTOR). If the TO field is written to 0, the Receiver Timeout is disabled and no timeout is detected. The TIMEOUT bit in the UART_SR remains at 0. Otherwise, the receiver loads an 8-bit counter with the value programmed in TO. This counter is decremented at each bit period and reloaded each time a new character is received. If the counter reaches 0, the TIMEOUT bit UART_SR rises. Then, the user can either:

- stop the counter clock until a new character is received. This is performed by writing a one to the STTTO (start Timeout) bit in the UART_CR. In this case, the idle state on RXD before a new character is received does not provide a timeout. This prevents having to handle an interrupt before a character is received and allows waiting for the next idle state on RXD after a frame is received, or
- obtain an interrupt while no character is received. This is performed by writing a one to the RETTO (Reload and Start Timeout) bit in the UART_CR. If RETTO is performed, the counter starts counting down immediately from the TO value. This enables generation of a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

If STTTO is performed, the counter clock is stopped until a first character is received. The idle state on RXD before the start of the frame does not provide a timeout. This prevents having to obtain a periodic interrupt and enables a wait of the end of frame when the idle state on RXD is detected.

If RETTO is performed, the counter starts counting down immediately from the TO value. This enables generation of a periodic interrupt so that a user timeout can be handled, for example when no key is pressed on a keyboard.

Figure 45-10 shows the block diagram of the Receiver Timeout feature.

Figure 45-10. Receiver Timeout Block Diagram

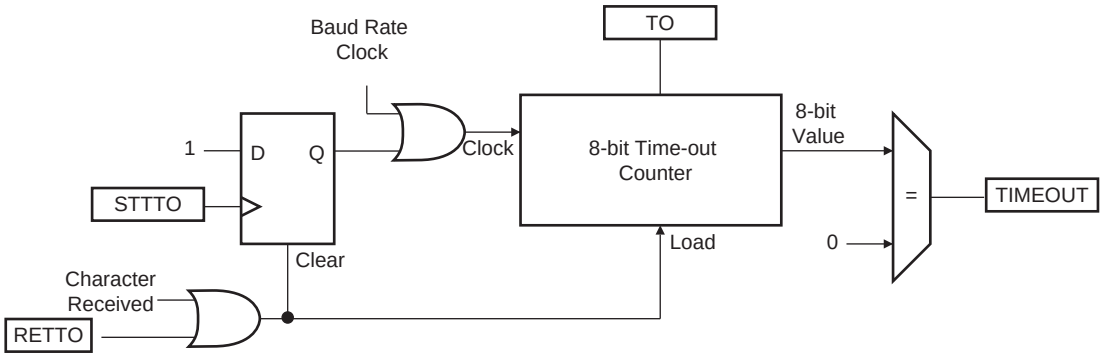


Table 45-4 gives the maximum timeout period for some standard baud rates.

Table 45-4. Maximum Timeout Period

Baud Rate (bit/s)	Bit Time (µs)	Timeout (µs)
600	1,667	425,085
1,200	833	212,415
2,400	417	106,335
4,800	208	53,040
9,600	104	26,520
14,400	69	17,595
19,200	52	13,260
28,800	35	8,925
38,400	26	6,630
56,000	18	4,590
57,600	17	4,335
200,000	5	1,275

45.5.3 Transmitter

45.5.3.1 Transmitter Reset, Enable and Disable

After device reset, the UART transmitter is disabled and must be enabled before being used. The transmitter is enabled by writing UART_CR with the bit TXEN at 1. From this command, the transmitter waits for a character to be written in the Transmit Holding Register (UART_THR) before actually starting the transmission.

The programmer can disable the transmitter by writing UART_CR with the bit TXDIS at 1. If the transmitter is not operating, it is immediately stopped. However, if a character is being processed into the internal shift register and/or a character has been written in the UART_THR, the characters are completed before the transmitter is actually stopped.

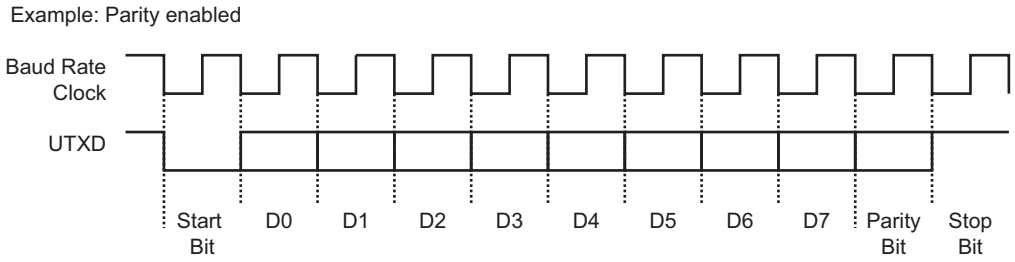
The programmer can also put the transmitter in its reset state by writing the UART_CR with the bit RSTTX at 1. This immediately stops the transmitter, whether or not it is processing characters.

45.5.3.2 Transmit Format

The UART transmitter drives the pin UTXD at the baud rate clock speed. The line is driven depending on the format defined in UART_MR and the data stored in the internal shift register. One start bit at level 0, then the 8 data bits, from the lowest to the highest bit, one optional parity bit and one stop bit at 1 are consecutively shifted

out as shown in the following figure. The field PARE in UART_MR defines whether or not a parity bit is shifted out. When a parity bit is enabled, it can be selected between an odd parity, an even parity, or a fixed space or mark bit.

Figure 45-11. Character Transmission

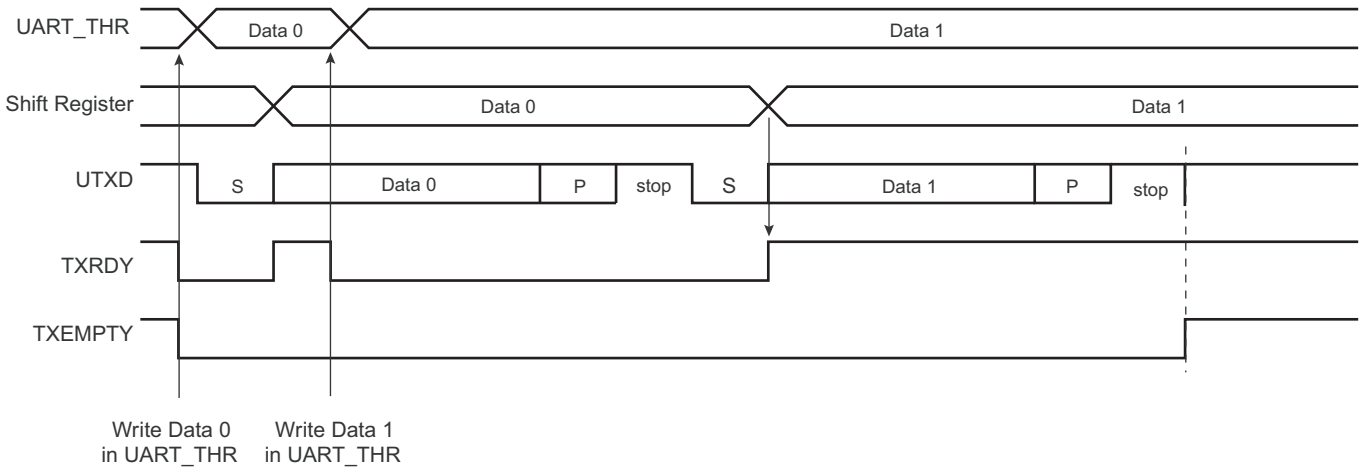


45.5.3.3 Transmitter Control

When the transmitter is enabled, the bit TXRDY (Transmitter Ready) is set in UART_SR. The transmission starts when the programmer writes in the UART_THR, and after the written character is transferred from UART_THR to the internal shift register. The TXRDY bit remains high until a second character is written in UART_THR. As soon as the first character is completed, the last character written in UART_THR is transferred into the internal shift register and TXRDY rises again, showing that the holding register is empty.

When both the internal shift register and UART_THR are empty, i.e., all the characters written in UART_THR have been processed, the TXEMPTY bit rises after the last stop bit has been completed.

Figure 45-12. Transmitter Control



45.5.4 DMA Support

Both the receiver and the transmitter of the UART are connected to a DMA Controller (DMAC) channel.

The DMA Controller channels are programmed via registers that are mapped within the DMAC user interface.

45.5.5 Comparison Function on Received Character

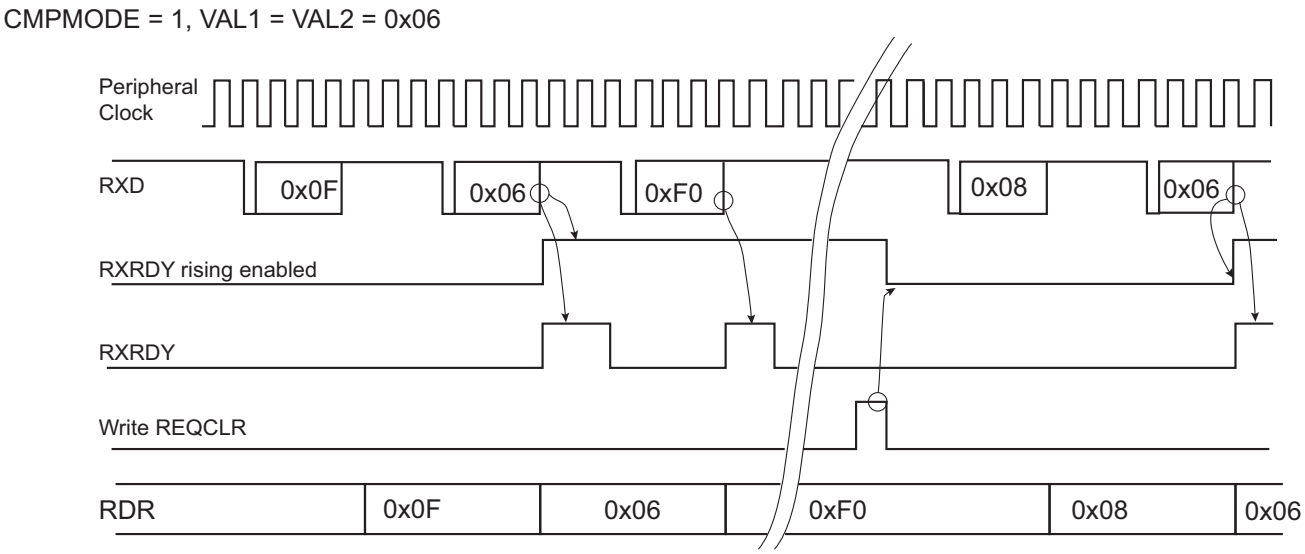
When a comparison is performed on a received character, the result of the comparison is reported on the CMP flag in UART_SR when UART_RHR is loaded with the new received character. The CMP flag is cleared by writing a one to the RSTSTA bit in UART_CR.

UART_CMPR (see Section 45.6.10 “UART Comparison Register”) can be programmed to provide different comparison methods. These are listed below:

- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if either received character equals VAL1 or VAL2.

By programming the CMPMODE bit to 1, the comparison function result triggers the start of the loading of UART_RHR (see Figure 45-13). The trigger condition occurs as soon as the received character value matches the condition defined by the programming of VAL1, VAL2 and CMPPAR in UART_CMPR. The comparison trigger event can be restarted by writing a one to the REQCLR bit in UART_CR.

Figure 45-13. Receive Holding Register Management



45.5.6 Asynchronous and Partial Wakeup (SleepWalking)

Asynchronous and partial wakeup (SleepWalking) is a means of data preprocessing that qualifies an incoming event, thus allowing the UART to decide whether or not to wake up the system. SleepWalking is used primarily when the system is in Wait mode (refer to section “Power Management Controller (PMC)”) but can also be enabled when the system is fully running.

No access must be performed in the UART between the enable of asynchronous partial wakeup and the wakeup performed by the UART.

If the system is in Wait mode and asynchronous and partial wakeup is enabled, the maximum baud rate that can be achieved equals 19200.

If the system is running or in Sleep mode, the maximum baud rate that can be achieved equals 115200 or higher. This limit is bounded by the peripheral clock frequency divided by 16.

The UART_RHR must be read before enabling asynchronous and partial wakeup.

When SleepWalking is enabled for the UART (see the PMC section), the PMC decodes a clock request from the UART. The request is generated as soon as there is a falling edge on the RXD line as this may indicate the beginning of a start bit. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the UART.

As soon as the clock is provided by the PMC, the UART processes the received frame and compares the received character with VAL1 and VAL2 in UART_CMPR (Section 45.6.10 “UART Comparison Register”).

The UART instructs the PMC to disable the clock if the received character value does not meet the conditions defined by VAL1 and VAL2 fields in UART_CMPR (see Figure 45-15).

If the received character value meets the conditions, the UART instructs the PMC to exit the full system from Wait mode (see Figure 45-14).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 equals VAL2, then the comparison is performed on a single value and the wakeup is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, then the wakeup is triggered if the received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 255, the wakeup is triggered as soon as a character is received.

The matching condition can be configured to include the parity bit (CMPPAR in UART_CMPR). Thus, if the received data matches the comparison condition defined by VAL1 and VAL2 but a parity error is encountered, the matching condition is cancelled and the UART instructs the PMC to disable the clock (see Figure 45-15).

If the processor and peripherals are running, the UART can be configured in Asynchronous and partial wakeup mode by enabling the PMC_SLPWK_ER (see PMC section). When activity is detected on the receive line, the UART requests the clock from the PMC and the comparison is performed. If there is a comparison match, the UART continues to request the clock. If there is no match, the clock is switched off for the UART only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wakeup mode is enabled for the UART (see PMC_SLPWK_ER in the PMC section).

When the system is kept in active/running mode and the UART enters Asynchronous and Partial Wakeup mode, the flag CMP must be programmed as the unique source of the UART interrupt.

When the system exits Wait mode as the result of a matching condition, the RXRDY flag is used to determine if the UART is the source of exit.

Note: If the SleepWalking function is enabled on the UART, a divide by 8 of the peripheral clock versus the bus clock is not possible. Other dividers can be used with no constraints.

Figure 45-14. Asynchronous Wakeup Use Case Examples

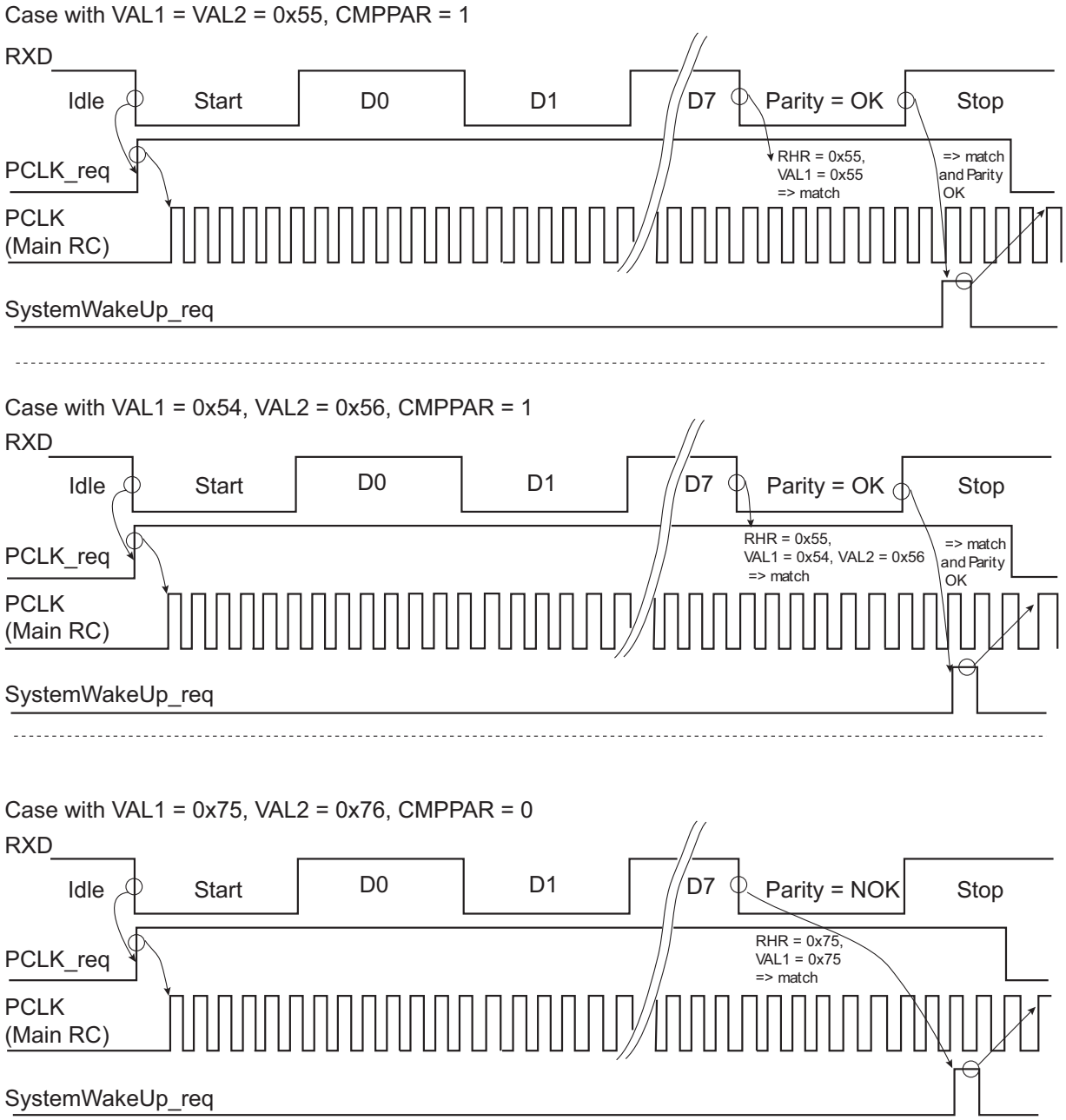
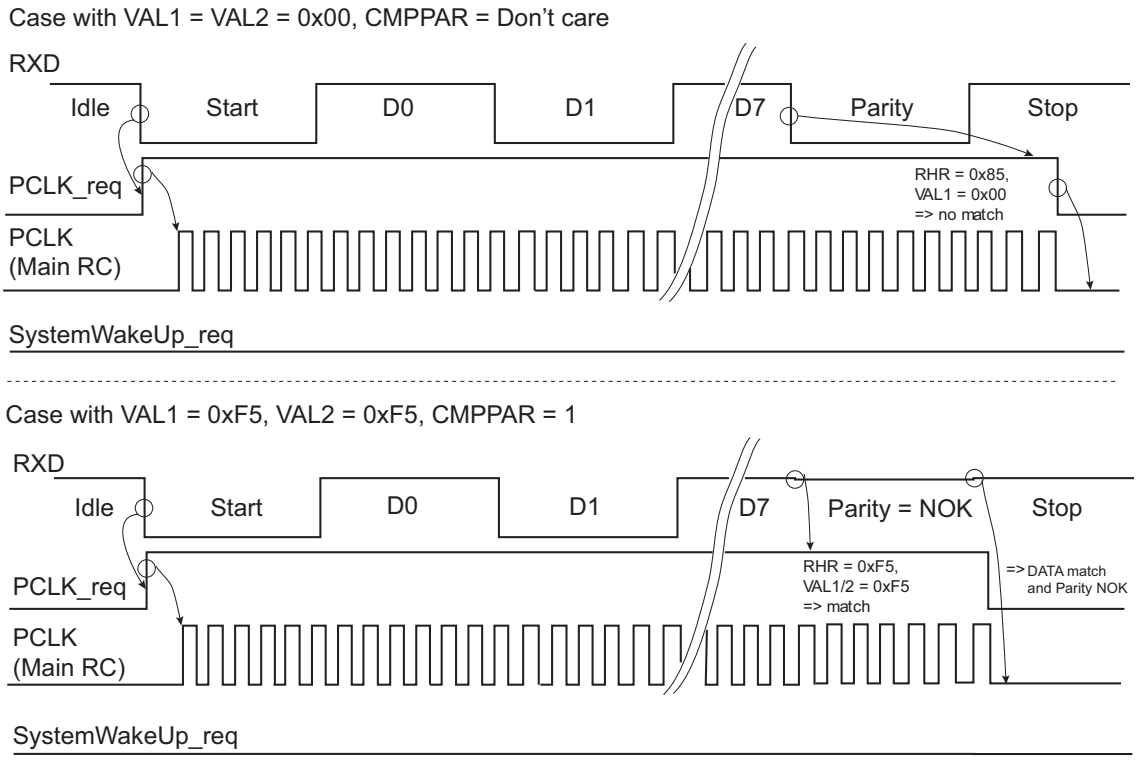


Figure 45-15. Asynchronous Event Generating Only Partial Wakeup



45.5.7 Register Write Protection

To prevent any single software error from corrupting UART behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the UART Write Protection Mode Register (UART_WPMR).

The following registers can be write-protected:

- UART Mode Register
- UART Baud Rate Generator Register
- UART Comparison Register

45.5.8 Test Modes

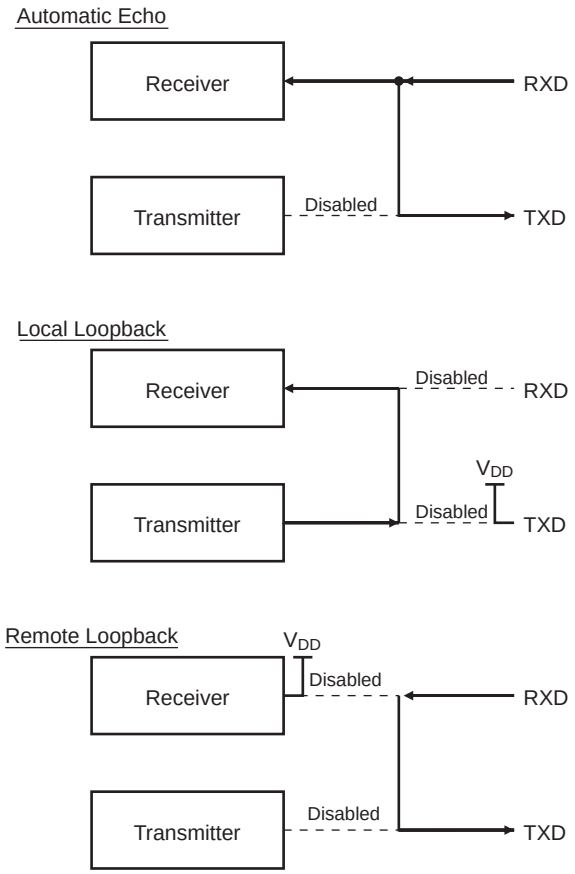
The UART supports three test modes. These modes of operation are programmed by using the CHMODE field in UART_MR.

The Automatic Echo mode allows a bit-by-bit retransmission. When a bit is received on the URXD line, it is sent to the UTXD line. The transmitter operates normally, but has no effect on the UTXD line.

The Local Loopback mode allows the transmitted characters to be received. UTXD and URXD pins are not used and the output of the transmitter is internally connected to the input of the receiver. The URXD pin level has no effect and the UTXD line is held high, as in idle state.

The Remote Loopback mode directly connects the URXD pin to the UTXD line. The transmitter and the receiver are disabled and have no effect. This mode allows a bit-by-bit retransmission.

Figure 45-16. Test Modes



45.6 Universal Asynchronous Receiver Transmitter (UART) User Interface

Table 45-5. Register Mapping

Offset	Register	Name	Access	Reset
0x0000	Control Register	UART_CR	Write-only	–
0x0004	Mode Register	UART_MR	Read/Write	0x0
0x0008	Interrupt Enable Register	UART_IER	Write-only	–
0x000C	Interrupt Disable Register	UART_IDR	Write-only	–
0x0010	Interrupt Mask Register	UART_IMR	Read-only	0x0
0x0014	Status Register	UART_SR	Read-only	–
0x0018	Receive Holding Register	UART_RHR	Read-only	0x0
0x001C	Transmit Holding Register	UART_THR	Write-only	–
0x0020	Baud Rate Generator Register	UART_BRGR	Read/Write	0x0
0x0024	Comparison Register	UART_CMPR	Read/Write	0x0
0x0028	Receiver Timeout Register	UART_RTOR	Read/Write	0x0
0x002C–0x003C	Reserved	–	–	–
0x0040–0x00E0	Reserved	–	–	–
0x00E4	Write Protection Mode Register	UART_WPMR	Read/Write	0x0
0x00E8	Reserved	–	–	–
0x00EC–0x00FC	Reserved	–	–	–

45.6.1 UART Control Register

Name: UART_CR

Address: 0xF801C000 (0), 0xF8020000 (1), 0xF8024000 (2), 0xFC008000 (3), 0xFC00C000 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	REQCLR	STTTO	RETTO	–	RSTSTA
7	6	5	4	3	2	1	0
TXDIS	TXEN	RXDIS	RXEN	RSTTX	RSTRX	–	–

- **RSTRX: Reset Receiver**

0: No effect.

1: The receiver logic is reset and disabled. If a character is being received, the reception is aborted.

- **RSTTX: Reset Transmitter**

0: No effect.

1: The transmitter logic is reset and disabled. If a character is being transmitted, the transmission is aborted.

- **RXEN: Receiver Enable**

0: No effect.

1: The receiver is enabled if RXDIS is 0.

- **RXDIS: Receiver Disable**

0: No effect.

1: The receiver is disabled. If a character is being processed and RSTRX is not set, the character is completed before the receiver is stopped.

- **TXEN: Transmitter Enable**

0: No effect.

1: The transmitter is enabled if TXDIS is 0.

- **TXDIS: Transmitter Disable**

0: No effect.

1: The transmitter is disabled. If a character is being processed and a character has been written in the UART_THR and RSTTX is not set, both characters are completed before the transmitter is stopped.

- **RSTSTA: Reset Status**

0: No effect.

1: Resets the status bits PARE, FRAME, CMP and OVRE in the UART_SR.

- **RETTO: Rearm Timeout**

0: No effect.

1: Restarts timeout.

- **STTTO: Start Timeout**

0: No effect.

1: Starts waiting for a character before clocking the timeout counter. Resets status bit TIMEOUT in UART_SR.

- **REQCLR: Request Clear**

SleepWalking enabled:

0: No effect.

1: Bit REQCLR clears the potential clock request currently issued by UART, thus the potential system wakeup is cancelled.

SleepWalking disabled:

0: No effect.

1: Bit REQCLR restarts the comparison trigger to enable receive holding register loading.

45.6.2 UART Mode Register

Name: UART_MR

Address: 0xF801C004 (0), 0xF8020004 (1), 0xF8024004 (2), 0xFC008004 (3), 0xFC00C004 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CHMODE		–	BRSRCCK	PAR			–
7	6	5	4	3	2	1	0
–	–	–	FILTER	–	–	–	–

- **FILTER: Receiver Digital Filter**

0 (DISABLED): UART does not filter the receive line.

1 (ENABLED): UART filters the receive line using a three-sample filter (16x-bit clock) (2 over 3 majority).

- **PAR: Parity Type**

Value	Name	Description
0	EVEN	Even Parity
1	ODD	Odd Parity
2	SPACE	Space: parity forced to 0
3	MARK	Mark: parity forced to 1
4	NO	No parity

- **BRSRCCK: Baud Rate Source Clock**

0 (PERIPH_CLK): The baud rate is driven by the peripheral clock

1 (GCLK): The baud rate is driven by a PMC-programmable clock GCLK (see section Power Management Controller (PMC)).

- **CHMODE: Channel Mode**

Value	Name	Description
0	NORMAL	Normal mode
1	AUTOMATIC	Automatic echo
2	LOCAL_LOOPBACK	Local loopback
3	REMOTE_LOOPBACK	Remote loopback

45.6.3 UART Interrupt Enable Register

Name: UART_IER

Address: 0xF801C008 (0), 0xF8020008 (1), 0xF8024008 (2), 0xFC008008 (3), 0xFC00C008 (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CMP	–	–	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **RXRDY: Enable RXRDY Interrupt**
- **TXRDY: Enable TXRDY Interrupt**
- **OVRE: Enable Overrun Error Interrupt**
- **FRAME: Enable Framing Error Interrupt**
- **PARE: Enable Parity Error Interrupt**
- **TIMEOUT: Enable Timeout Interrupt**
- **TXEMPTY: Enable TXEMPTY Interrupt**
- **CMP: Enable Comparison Interrupt**

45.6.4 UART Interrupt Disable Register

Name: UART_IDR

Address: 0xF801C00C (0), 0xF802000C (1), 0xF802400C (2), 0xFC00800C (3), 0xFC00C00C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CMP	–	–	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **RXRDY: Disable RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **OVRE: Disable Overrun Error Interrupt**
- **FRAME: Disable Framing Error Interrupt**
- **PARE: Disable Parity Error Interrupt**
- **TIMEOUT: Disable Timeout Interrupt**
- **TXEMPTY: Disable TXEMPTY Interrupt**
- **CMP: Disable Comparison Interrupt**

45.6.5 UART Interrupt Mask Register

Name: UART_IMR

Address: 0xF801C010 (0), 0xF8020010 (1), 0xF8024010 (2), 0xFC008010 (3), 0xFC00C010 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CMP	–	–	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

- **RXRDY: Mask RXRDY Interrupt**
- **TXRDY: Disable TXRDY Interrupt**
- **OVRE: Mask Overrun Error Interrupt**
- **FRAME: Mask Framing Error Interrupt**
- **PARE: Mask Parity Error Interrupt**
- **TIMEOUT: Mask Timeout Interrupt**
- **TXEMPTY: Mask TXEMPTY Interrupt**
- **CMP: Mask Comparison Interrupt**

45.6.6 UART Status Register

Name: UART_SR

Address: 0xF801C014 (0), 0xF8020014 (1), 0xF8024014 (2), 0xFC008014 (3), 0xFC00C014 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CMP	–	–	–	–	–	TXEMPTY	TIMEOUT
7	6	5	4	3	2	1	0
PARE	FRAME	OVRE	–	–	–	TXRDY	RXRDY

- **RXRDY: Receiver Ready**

0: No character has been received since the last read of the UART_RHR, or the receiver is disabled.

1: At least one complete character has been received, transferred to UART_RHR and not yet read.

- **TXRDY: Transmitter Ready**

0: A character has been written to UART_THR and not yet transferred to the internal shift register, or the transmitter is disabled.

1: There is no character written to UART_THR not yet transferred to the internal shift register.

- **OVRE: Overrun Error**

0: No overrun error has occurred since the last RSTSTA.

1: At least one overrun error has occurred since the last RSTSTA.

- **FRAME: Framing Error**

0: No framing error has occurred since the last RSTSTA.

1: At least one framing error has occurred since the last RSTSTA.

- **PARE: Parity Error**

0: No parity error has occurred since the last RSTSTA.

1: At least one parity error has occurred since the last RSTSTA.

- **TIMEOUT: Receiver Timeout**

0: There has not been a timeout since the last Start Timeout command (STTTO in UART_CR) or the Timeout Register is 0.

1: There has been a timeout since the last Start Timeout command (STTTO in UART_CR).

- **TXEMPTY: Transmitter Empty**

0: There are characters in UART_THR, or characters being processed by the transmitter, or the transmitter is disabled.

1: There are no characters in UART_THR and there are no characters being processed by the transmitter.

- **CMP: Comparison Match**

0: No received character matches the comparison criteria programmed in VAL1, VAL2 fields and in CMPPAR bit since the last RSTSTA.

1: The received character matches the comparison criteria.

45.6.7 UART Receiver Holding Register

Name: UART_RHR

Address: 0xF801C018 (0), 0xF8020018 (1), 0xF8024018 (2), 0xFC008018 (3), 0xFC00C018 (4)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RXCHR							

- **RXCHR: Received Character**

Last received character if RXRDY is set.

45.6.8 UART Transmit Holding Register

Name: UART_THR

Address: 0xF801C01C (0), 0xF802001C (1), 0xF802401C (2), 0xFC00801C (3), 0xFC00C01C (4)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
TXCHR							

- **TXCHR: Character to be Transmitted**

Next character to be transmitted after the current character if TXRDY is not set.

45.6.9 UART Baud Rate Generator Register

Name: UART_BRGR

Address: 0xF801C020 (0), 0xF8020020 (1), 0xF8024020 (2), 0xFC008020 (3), 0xFC00C020 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
CD							
7	6	5	4	3	2	1	0
CD							

- **CD: Clock Divisor**

0: Baud rate clock is disabled

1 to 65,535:

If BRSRCCK = 0:

$$CD = \frac{f_{peripheral\ clock}}{16 \times Baud\ Rate}$$

If BRSRCCK = 1:

$$CD = \frac{f_{GCLKx}}{16 \times Baud\ Rate}$$

45.6.10 UART Comparison Register

Name: UART_CMPR

Address: 0xF801C024 (0), 0xF8020024 (1), 0xF8024024 (2), 0xFC008024 (3), 0xFC00C024 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
VAL2							
15	14	13	12	11	10	9	8
–	CMPPAR	–	CMPMODE	–	–	–	–
7	6	5	4	3	2	1	0
VAL1							

- **VAL1: First Comparison Value for Received Character**

0–255: The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in UART_SR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the UART requests a system wakeup if the condition is met.

- **CMPMODE: Comparison Mode**

Value	Name	Description
0	FLAG_ONLY	Any character is received and comparison function drives CMP flag.
1	START_CONDITION	Comparison condition must be met to start reception.

- **CMPPAR: Compare Parity**

0: The parity is not checked and a bad parity cannot prevent from waking up the system.

1: The parity is checked and a matching condition on data can be cancelled by an error on parity bit, so no wakeup is performed.

- **VAL2: Second Comparison Value for Received Character**

0–255: The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in UART_SR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the UART requests a system wakeup if condition is met.

45.6.11 UART Receiver Timeout Register

Name: UART_RTOR

Address: 0xF801C028 (0), 0xF8020028 (1), 0xF8024028 (2), 0xFC008028 (3), 0xFC00C028 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
TO							

- **TO: Timeout Value**

0: The receiver timeout is disabled.

1–255: The receiver timeout is enabled and the timeout delay is TO x bit period.

45.6.12 UART Write Protection Mode Register

Name: UART_WPMR

Address: 0xF801C0E4 (0), 0xF80200E4 (1), 0xF80240E4 (2), 0xFC0080E4 (3), 0xFC00C0E4 (4)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x554152 (UART in ASCII).

See Section 45.5.7 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x554152	PASSWD	Writing any other value in this field aborts the write operation. Always reads as 0.

46. Serial Peripheral Interface (SPI)

46.1 Description

The Serial Peripheral Interface (SPI) circuit is a synchronous serial data link that provides communication with external devices in Master or Slave mode. It also enables communication between processors if an external processor is connected to the system.

The Serial Peripheral Interface is essentially a Shift register that serially transmits data bits to other SPIs. During a data transfer, one SPI system acts as the “master”' which controls the data flow, while the other devices act as “slaves'' which have data shifted into and out by the master. Different CPUs can take turn being masters (multiple master protocol, contrary to single master protocol where one CPU is always the master while all of the others are always slaves). One master can simultaneously shift data into multiple slaves. However, only one slave can drive its output to write data back to the master at any given time.

A slave device is selected when the master asserts its NSS signal. If multiple slave devices exist, the master generates a separate slave select signal for each slave (NPCS).

The SPI system consists of two data lines and two control lines:

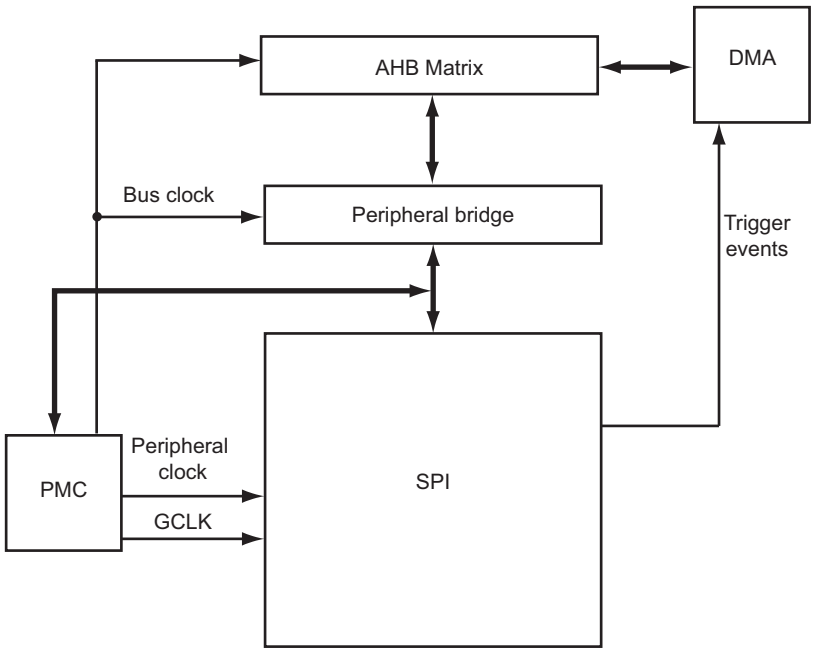
- Master Out Slave In (MOSI)—This data line supplies the output data from the master shifted into the input(s) of the slave(s).
- Master In Slave Out (MISO)—This data line supplies the output data from a slave to the input of the master. There may be no more than one slave transmitting data during any particular transfer.
- Serial Clock (SPCK)—This control line is driven by the master and regulates the flow of the data bits. The master can transmit data at a variety of baud rates; there is one SPCK pulse for each bit that is transmitted.
- Slave Select (NSS)—This control line allows slaves to be turned on and off by hardware.

46.2 Embedded Characteristics

- Master or Slave Serial Peripheral Bus Interface
  - 8-bit to 16-bit programmable data length per chip select
  - Programmable phase and polarity per chip select
  - Programmable transfer delay between consecutive transfers and delay before SPI clock per chip select
  - Programmable delay between chip selects
  - Selectable mode fault detection
- Master Mode can drive SPCK up to Peripheral Clock
- 16-data Transmit and Receive FIFOs
- Master Mode Bit Rate can be Independent of the Processor/Peripheral Clock
- Slave mode operates on SPCK, asynchronously with core and bus clock
- Four chip selects with external decoder support allow communication with up to 15 peripherals
- Communication with Serial External Devices Supported
  - Serial memories, such as DataFlash and 3-wire EEPROMs
  - Serial peripherals, such as ADCs, DACs, LCD controllers, CAN controllers and sensors
  - External coprocessors
- Connection to DMA Channel Capabilities, Optimizing Data Transfers
  - One channel for the receiver
  - One channel for the transmitter
- Register Write Protection

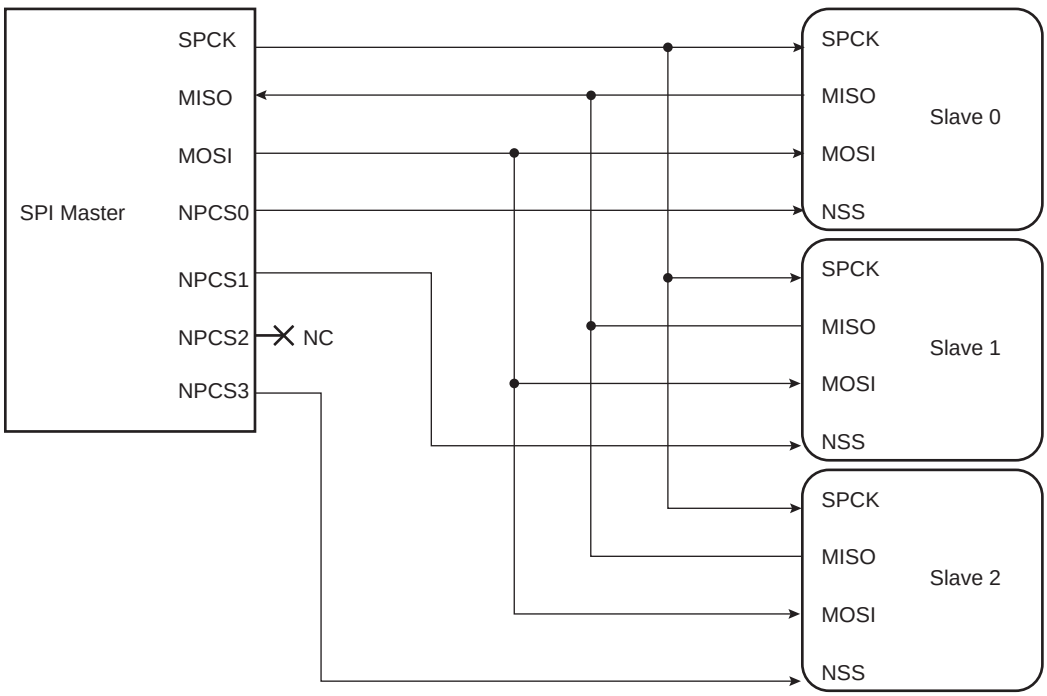
46.3 Block Diagram

Figure 46-1. Block Diagram



46.4 Application Block Diagram

Figure 46-2. Application Block Diagram: Single Master/Multiple Slave Implementation



46.5 Signal Description

Table 46-1. Signal Description

Pin Name	Pin Description	Type	
		Master	Slave
MISO	Master In Slave Out	Input	Output
MOSI	Master Out Slave In	Output	Input
SPCK	Serial Clock	Output	Input
NPCS1–NPCS3	Peripheral Chip Selects	Output	Unused
NPCS0/NSS	Peripheral Chip Select/Slave Select	Output	Input

46.6 Product Dependencies

46.6.1 I/O Lines

The pins used for interfacing the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the SPI pins to their peripheral functions.

Table 46-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
SPI0	SPI0_MISO	PA16	A
SPI0	SPI0_MISO	PA31	C
SPI0	SPI0_MOSI	PA15	A
SPI0	SPI0_MOSI	PB0	C
SPI0	SPI0_NPCS0	PA17	A
SPI0	SPI0_NPCS0	PA30	C
SPI0	SPI0_NPCS1	PA18	A
SPI0	SPI0_NPCS1	PA29	C
SPI0	SPI0_NPCS2	PA19	A
SPI0	SPI0_NPCS2	PA27	C
SPI0	SPI0_NPCS3	PA20	A
SPI0	SPI0_NPCS3	PA28	C
SPI0	SPI0_SPCK	PA14	A
SPI0	SPI0_SPCK	PB1	C
SPI1	SPI1_MISO	PA24	D
SPI1	SPI1_MISO	PC3	D
SPI1	SPI1_MISO	PD27	A
SPI1	SPI1_MOSI	PA23	D
SPI1	SPI1_MOSI	PC2	D
SPI1	SPI1_MOSI	PD26	A
SPI1	SPI1_NPCS0	PA25	D
SPI1	SPI1_NPCS0	PC4	D

Table 46-2. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
SPI1	SPI1_NPCS0	PD28	A
SPI1	SPI1_NPCS1	PA26	D
SPI1	SPI1_NPCS1	PC5	D
SPI1	SPI1_NPCS1	PD29	A
SPI1	SPI1_NPCS2	PA27	D
SPI1	SPI1_NPCS2	PC6	D
SPI1	SPI1_NPCS2	PD30	A
SPI1	SPI1_NPCS3	PA28	D
SPI1	SPI1_NPCS3	PC7	D
SPI1	SPI1_SPCK	PA22	D
SPI1	SPI1_SPCK	PC1	D
SPI1	SPI1_SPCK	PD25	A

46.6.2 Power Management

The SPI can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the SPI clock.

46.6.3 Interrupt

The SPI interface has an interrupt line connected to the interrupt controller. Handling the SPI interrupt requires programming the interrupt controller before configuring the SPI.

Table 46-3. Peripheral IDs

Instance	ID
SPI0	33
SPI1	34

46.6.4 Direct Memory Access Controller (DMAC)

The SPI interface can be used in conjunction with the DMAC in order to reduce processor overhead. For a full description of the DMAC, refer to the relevant section.

46.7 Functional Description

46.7.1 Modes of Operation

The SPI operates in Master mode or in Slave mode.

- The SPI operates in Master mode by setting the MSTR bit in the SPI Mode Register (SPI_MR):
  - Pins NPCS0 to NPCS3 are all configured as outputs
  - The SPCK pin is driven
  - The MISO line is wired on the receiver input
  - The MOSI line is driven as an output by the transmitter.
- The SPI operates in Slave mode if the MSTR bit in the SPI_MR is written to ‘0’:
  - The MISO line is driven by the transmitter output
  - The MOSI line is wired on the receiver input
  - The SPCK pin is driven by the transmitter to synchronize the receiver.
  - The NPCS0 pin becomes an input, and is used as a slave select signal (NSS)
  - NPCS1 to NPCS3 are not driven and can be used for other purposes.

The data transfers are identically programmable for both modes of operation. The baud rate generator is activated only in Master mode.

46.7.2 Data Transfer

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the SPI chip select registers (SPI_CSRx). The clock phase is programmed with the NCPHA bit. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Consequently, a master/slave pair must use the same parameter pair values to communicate. If multiple slaves are connected and require different configurations, the master must reconfigure itself each time it needs to communicate with a different slave.

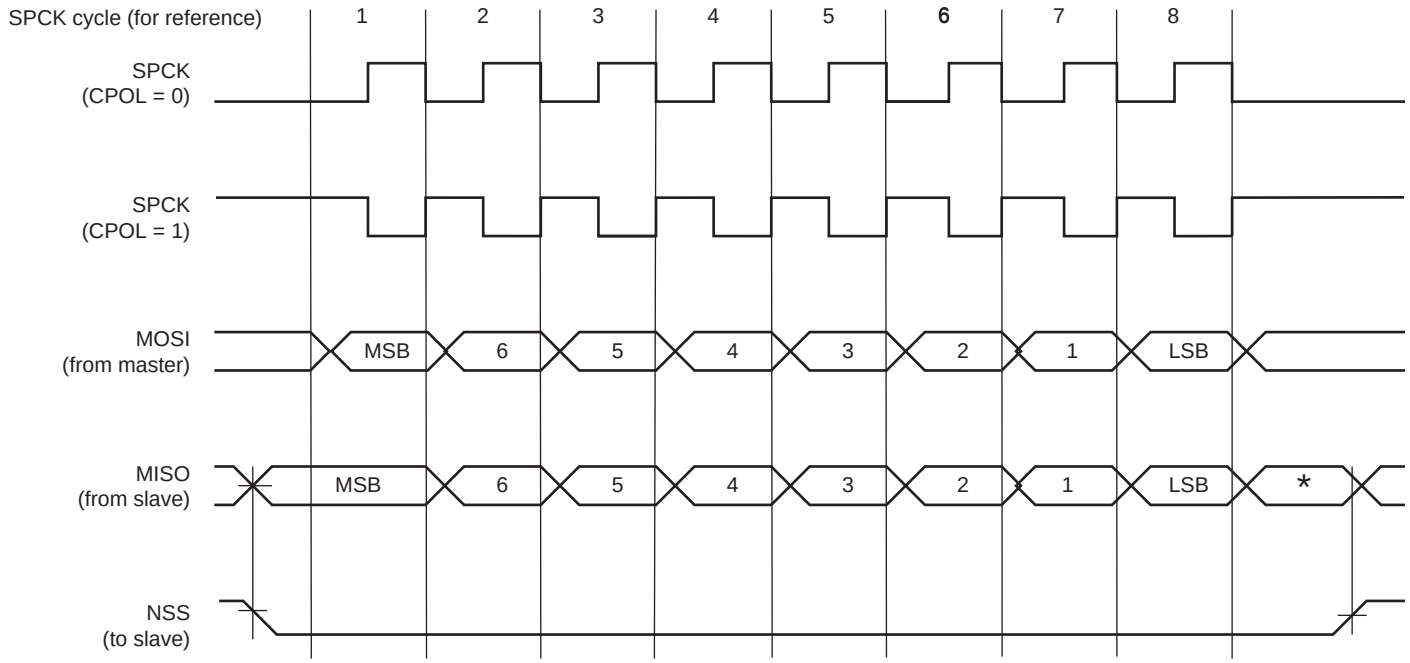
Table 46-4 shows the four modes and corresponding parameter settings.

Table 46-4. SPI Bus Protocol Modes

SPI Mode	CPOL	NCPHA	Shift SPCK Edge	Capture SPCK Edge	SPCK Inactive Level
0	0	1	Falling	Rising	Low
1	0	0	Rising	Falling	Low
2	1	1	Rising	Falling	High
3	1	0	Falling	Rising	High

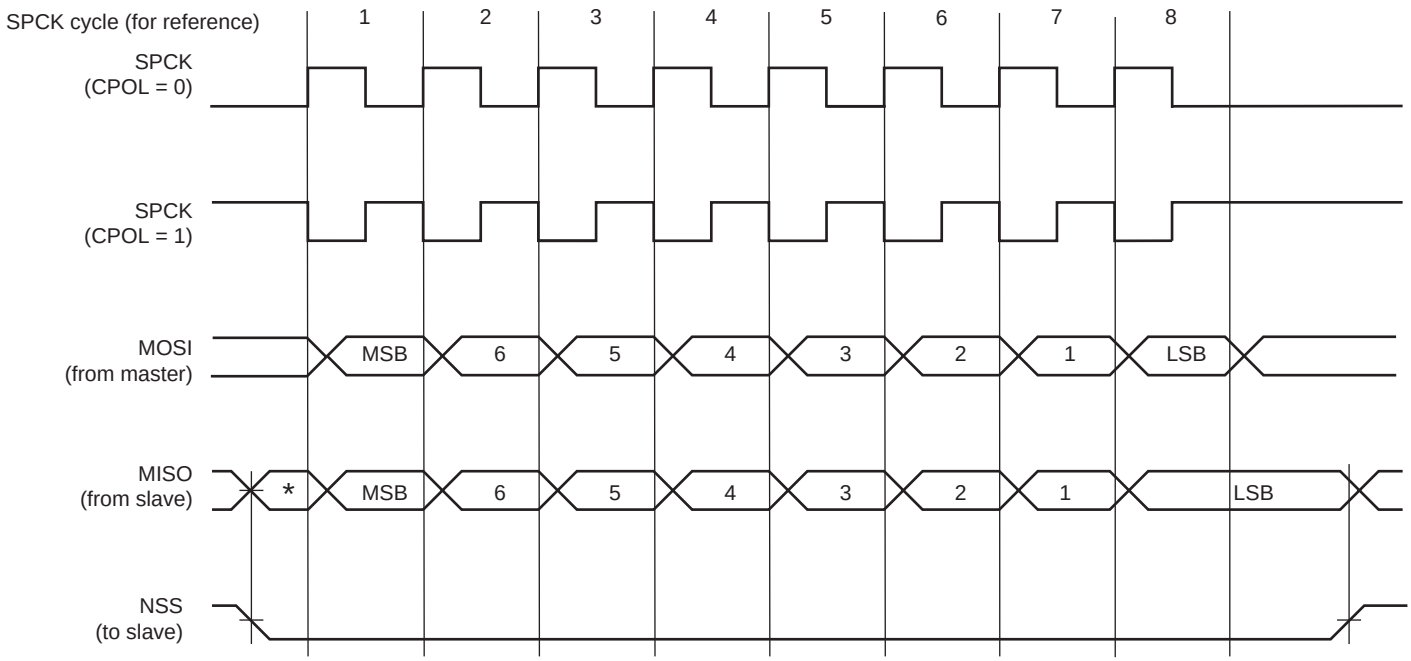
Figure 46-3 and Figure 46-4 show examples of data transfers.

Figure 46-3. SPI Transfer Format (NCPHA = 1, 8 bits per transfer)



* Not defined.

Figure 46-4. SPI Transfer Format (NCPHA = 0, 8 bits per transfer)



* Not defined.

46.7.3 Master Mode Operations

When configured in Master mode, the SPI operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave(s) connected to the SPI bus. The SPI drives the chip select line to the slave and the serial clock signal (SPCK).

The SPI features two holding registers, the Transmit Data Register (SPI_TDR) and the Receive Data Register (SPI_RDR), and a single shift register. The holding registers maintain the data flow at a constant rate.

After enabling the SPI, a data transfer starts when the processor writes to the SPI_TDR. The written data is immediately transferred in the Shift register and the transfer on the SPI bus starts. While the data in the Shift register is shifted on the MOSI line, the MISO line is sampled and shifted in the Shift register. Data cannot be loaded in the SPI_RDR without transmitting data. If there is no data to transmit, dummy data can be used (SPI_TDR filled with ones). If the SPI_MR.WDRBT bit is set, transmission can occur only if the SPI_RDR has been read. If Receiving mode is not required, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the SPI Status register (SPI_SR) can be discarded.

Before writing the SPI_TDR, the PCS field in the SPI_MR must be set in order to select a slave.

If new data is written in the SPI_TDR during the transfer, it is kept in the SPI_TDR until the current transfer is completed. Then, the received data is transferred from the Shift register to the SPI_RDR, the data in the SPI_TDR is loaded in the Shift register and a new transfer starts.

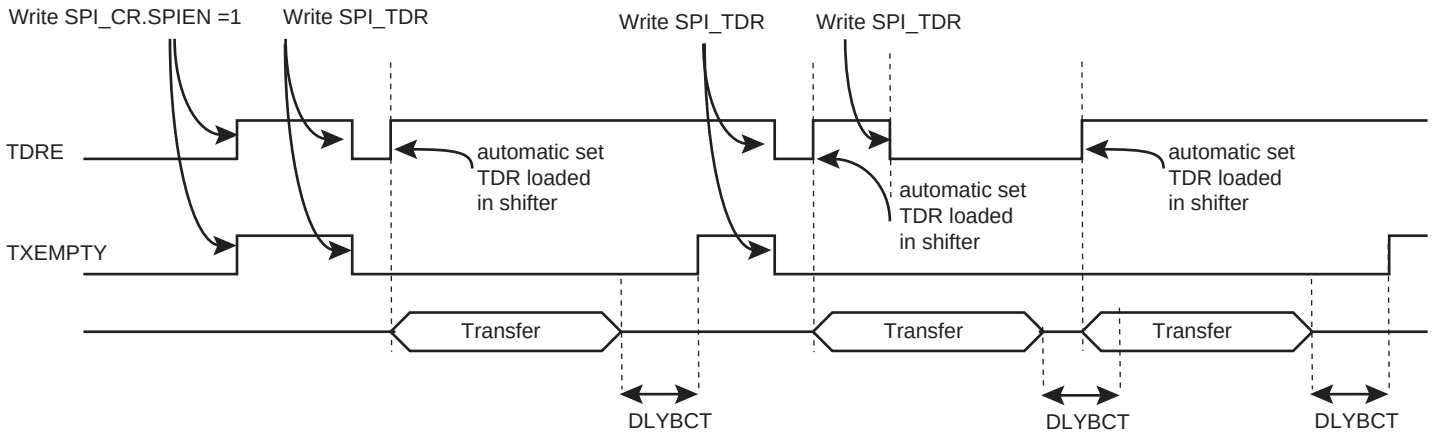
As soon as the SPI_TDR is written, the Transmit Data Register Empty (TDRE) flag in the SPI_SR is cleared. When the data written in the SPI_TDR is loaded into the Shift register, the TDRE flag in the SPI_SR is set. The TDRE bit is used to trigger the Transmit DMA channel.

See Figure 46-5.

The end of transfer is indicated by the TXEMPTY flag in the SPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

Note: When the SPI is enabled, the TDRE and TXEMPTY flags are set.

Figure 46-5. TDRE and TXEMPTY flag behavior



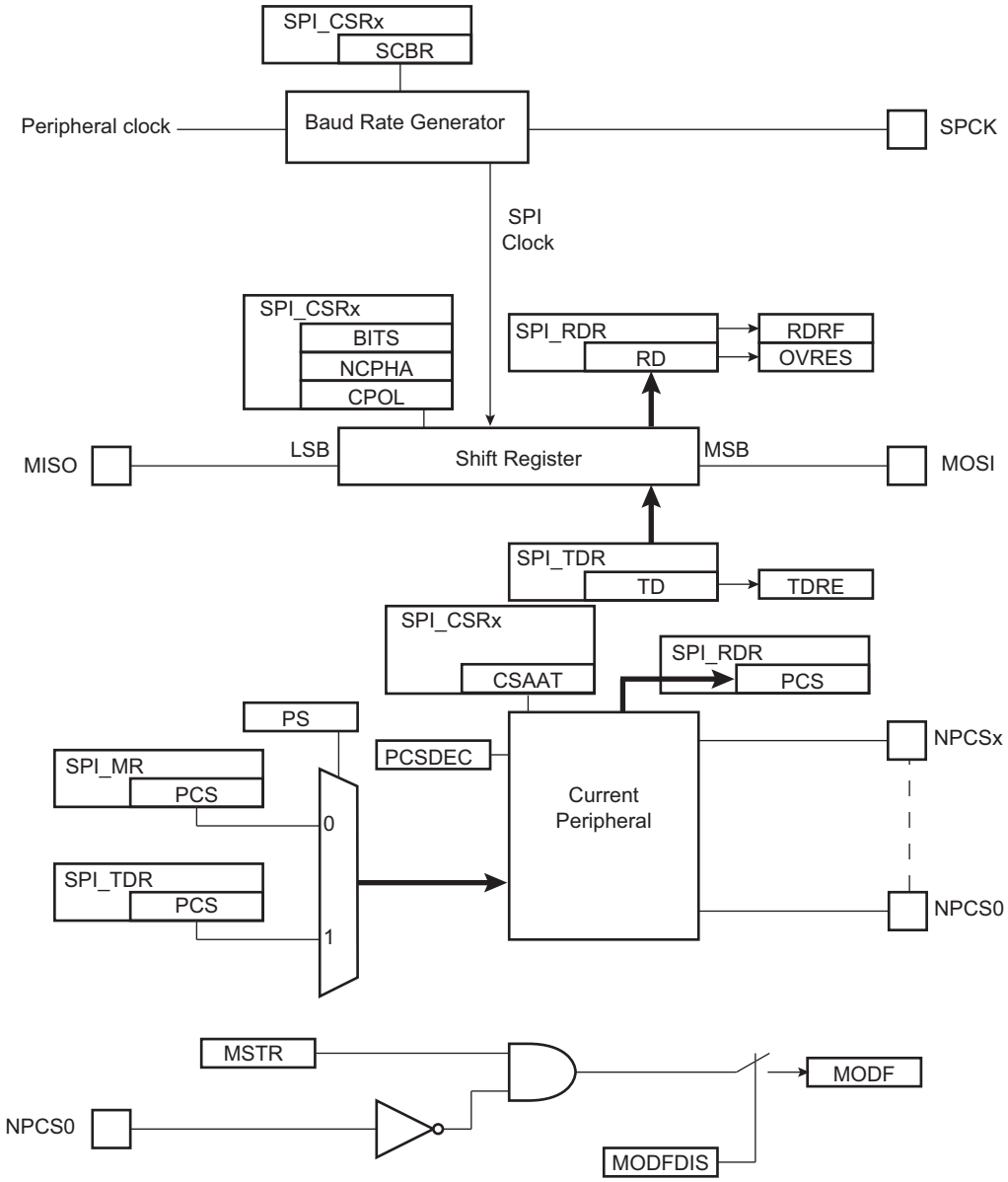
The transfer of received data from the Shift register to the SPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in the SPI_SR. When the received data is read, the RDRF bit is cleared.

If the SPI_RDR has not been read before new data is received, the Overrun Error (OVRES) bit in the SPI_SR is set. As long as this flag is set, data is loaded in the SPI_RDR. The user has to read the SPI_SR to clear the OVRES bit.

Figure 46-6 shows a block diagram of the SPI when operating in Master mode. Figure 46-7 shows a flow chart describing how transfers are handled.

46.7.3.1 Master Mode Block Diagram

Figure 46-6. Master Mode Block Diagram



46.7.3.2 Master Mode Flow Diagram

Figure 46-7. Master Mode Flow Diagram

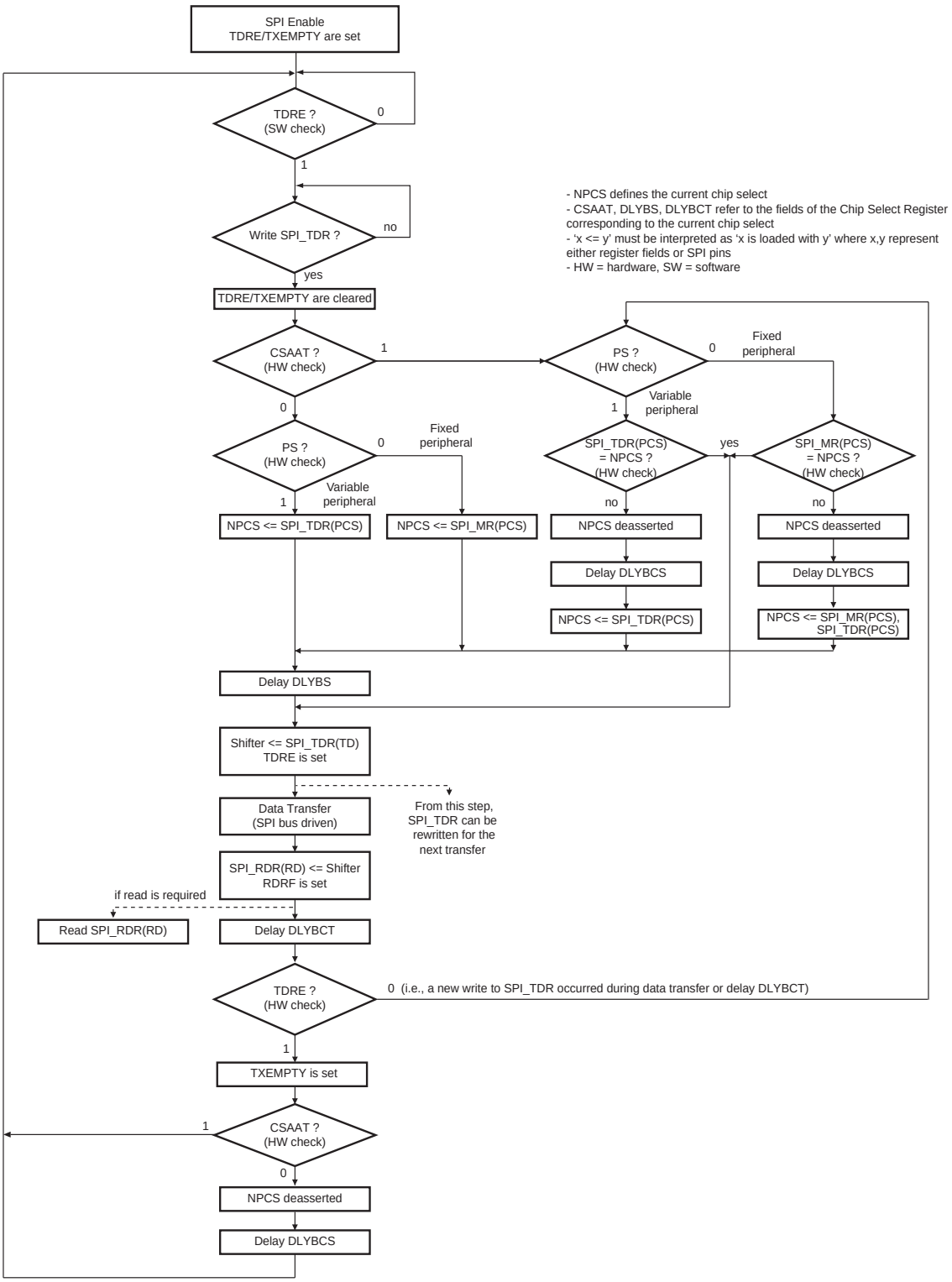
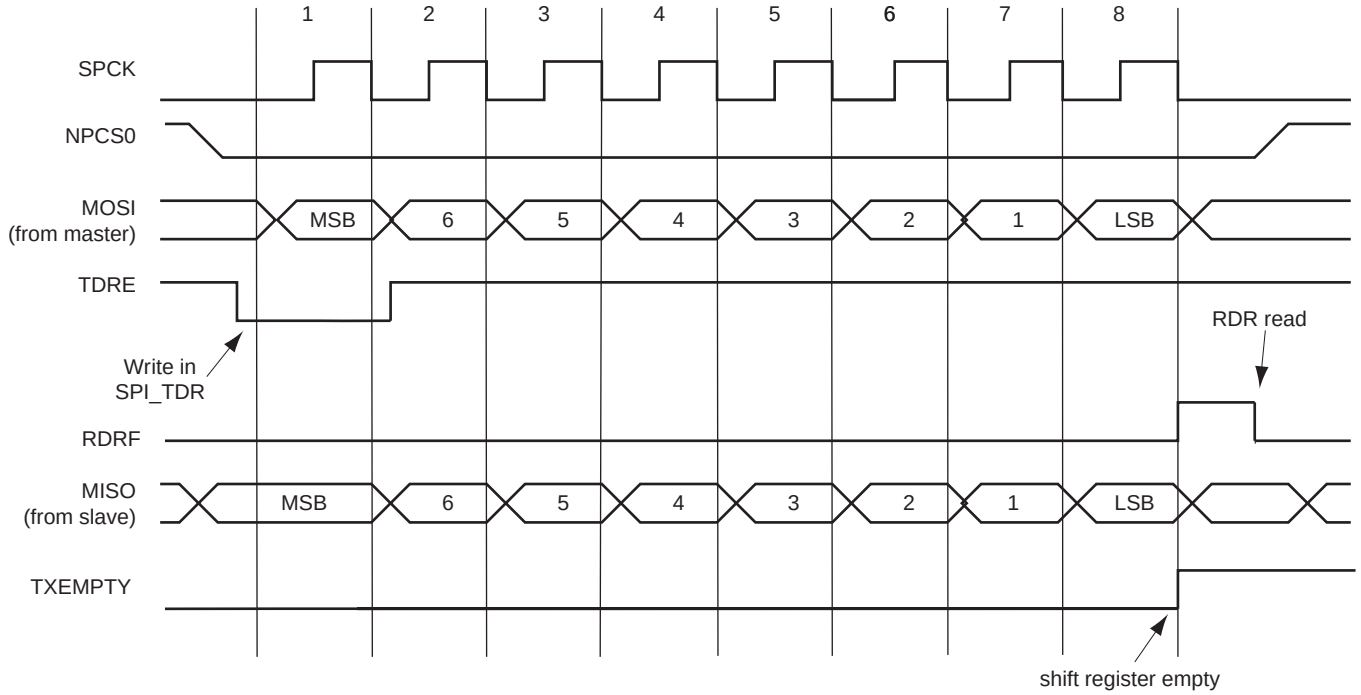


Figure 46-8 shows the behavior of Transmit Data Register Empty (TDRE), Receive Data Register (RDRF) and Transmission Register Empty (TXEMPTY) status flags within the SPI_SR during an 8-bit data transfer in Fixed mode without the DMA involved.

Figure 46-8. Status Register Flags Behavior



46.7.3.3 Clock Generation

The SPI Baud rate clock is generated by dividing the peripheral clock by a value between 1 and 255.

If the SCBR field in the SPI_CSRx is programmed to 1, the operating baud rate is peripheral clock (see the electrical characteristics section for the SPCK maximum frequency). Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

At reset, SCBR is 0 and the user has to program it to a valid value before performing the first transfer.

The divisor can be defined independently for each chip select, as it has to be programmed in the SCBR field. This allows the SPI to automatically adapt the baud rate for each interfaced peripheral without reprogramming.

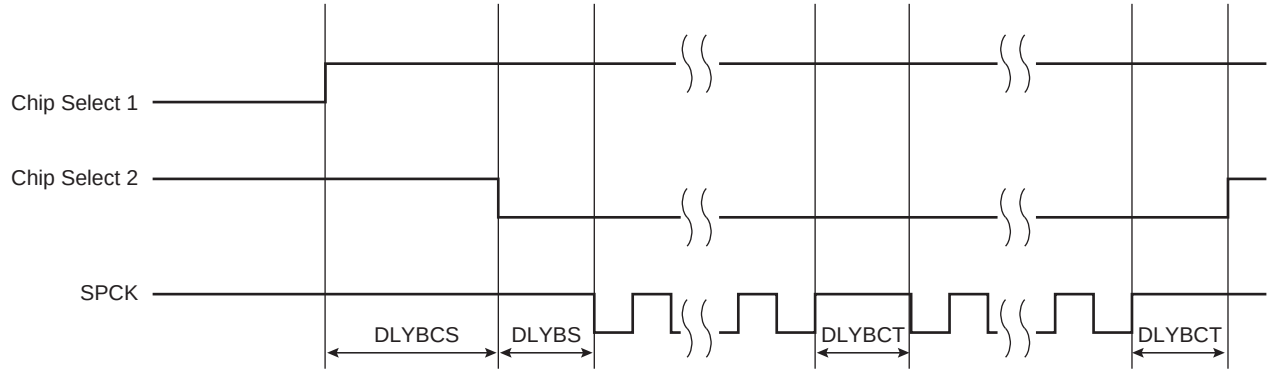
46.7.3.4 Transfer Delays

Figure 46-9 shows a chip select transfer change and consecutive transfers on the same chip select. Three delays can be programmed to modify the transfer waveforms:

- Delay between the chip selects—programmable only once for all chip selects by writing the DLYBCS field in the SPI_MR. The SPI slave device deactivation delay is managed through DLYBCS. If there is only one SPI slave device connected to the master, the DLYBCS field does not need to be configured. If several slave devices are connected to a master, DLYBCS must be configured depending on the highest deactivation delay. Refer to the SPI slave device electrical characteristics.
- Delay before SPCK—independently programmable for each chip select by writing the DLYBS field. The SPI slave device activation delay is managed through DLYBS. Refer to the SPI slave device electrical characteristics to define DLYBS.
- Delay between consecutive transfers—independently programmable for each chip select by writing the DLYBCT field. The time required by the SPI slave device to process received data is managed through DLYBCT. This time depends on the SPI slave system activity.

These delays allow the SPI to be adapted to the interfaced peripherals and their speed and bus release time.

Figure 46-9. Programmable Delays



46.7.3.5 Peripheral Selection

The serial peripherals are selected through the assertion of the NPCS0 to NPCS3 signals. By default, all NPCS signals are high before and after each transfer.

- **Fixed Peripheral Select Mode**: SPI exchanges data with only one peripheral.
  Fixed Peripheral Select mode is enabled by clearing the PS bit in the SPI_MR. In this case, the current peripheral is defined by the PCS field in the SPI_MR and the PCS field in the SPI_TDR has no effect.
- **Variable Peripheral Select Mode:** Data can be exchanged with more than one peripheral without having to reprogram the NPCS field in the SPI_MR.
  Variable Peripheral Select mode is enabled by setting the PS bit in the SPI_MR. The PCS field in the SPI_TDR is used to select the current peripheral. This means that the peripheral selection can be defined for each new data. The value to write in the SPI_TDR has the following format:

  [xxxxxxx(7-bit) + LASTXFER(1-bit)(1)+ xxxx(4-bit) + PCS (4-bit) + DATA (8 to 16-bit)] with PCS equals the chip select to assert, as defined in Section 46.8.6 “SPI Transmit Data Register” and LASTXFER bit at 0 or 1 depending on the CSAAT bit.

Note: 1. Optional

CSAAT, LASTXFER and CSNAAT bits are discussed in Section 46.7.3.9 “Peripheral Deselection with DMA”.

If LASTXFER is used, the command must be issued after writing the last character. Instead of LASTXFER, the user can use the SPIDIS command. After the end of the DMA transfer, it is necessary to wait for the TXEMPTY flag and then write SPIDIS into the SPI Control Register (SPI_CR). This does not change the configuration register values). The NPCS is disabled after the last character transfer. Then, another DMA transfer can be started if the SPIEN has previously been written in the SPI_CR.

46.7.3.6 SPI Direct Access Memory Controller (DMAC)

In both Fixed and Variable modes, the Direct Memory Access Controller (DMAC) can be used to reduce processor overhead.

The fixed peripheral selection allows buffer transfers with a single peripheral. Using the DMAC is an optimal means, as the size of the data transfer between the memory and the SPI is either 8 bits or 16 bits. However, if the peripheral selection is modified, the SPI_MR must be reprogrammed.

The variable peripheral selection allows buffer transfers with multiple peripherals without reprogramming the SPI_MR. Data written in the SPI_TDR is 32 bits wide and defines the real data to be transmitted and the destination peripheral. Using the DMAC in this mode requires 32-bit wide buffers, with the data in the LSBs and the

PCS and LASTXFER fields in the MSBs. However, the SPI still controls the number of bits (8 to 16) to be transferred through MISO and MOSI lines with the chip select configuration registers. This is not the optimal means in terms of memory size for the buffers, but it provides a very effective means to exchange data with several peripherals without any intervention of the processor.

46.7.3.7 Peripheral Chip Select Decoding

The user can program the SPI to operate with up to 15 slave peripherals by decoding the four chip select lines, NPCS0 to NPCS3 with an external decoder/demultiplexer (refer to Figure 46-10). This can be enabled by setting the PCSDEC bit in the SPI_MR.

When operating without decoding, the SPI makes sure that in any case only one chip select line is activated, i.e., one NPCS line driven low at a time. If two bits are defined low in a PCS field, only the lowest numbered chip select is driven low.

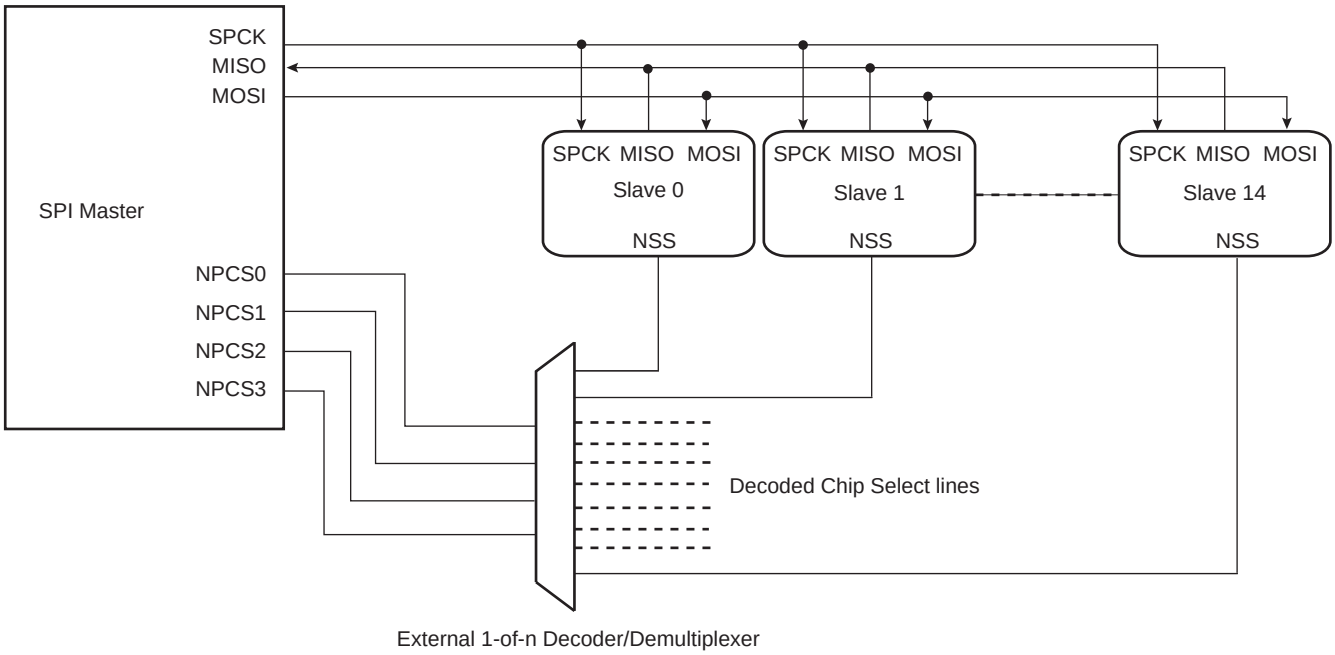
When operating with decoding, the SPI directly outputs the value defined by the PCS field on the NPCS lines of either SPI_MR or SPI_TDR (depending on PS).

As the SPI sets a default value of 0xF on the chip select lines (i.e., all chip select lines at 1) when not processing any transfer, only 15 peripherals can be decoded.

The SPI has four chip select registers (SPI_CSR0...SPI_CSR3). As a result, when external decoding is activated, each NPCS chip select defines the characteristics of up to four peripherals. As an example, SPI_CRS0 defines the characteristics of the externally decoded peripherals 0 to 3, corresponding to the PCS values 0x0 to 0x3. Consequently, the user has to make sure to connect compatible peripherals on the decoded chip select lines 0 to 3, 4 to 7, 8 to 11 and 12 to 14. Figure 46-10 shows this type of implementation.

If the CSAAT bit is used, with or without the DMAC, the Mode Fault detection for NPCS0 line must be disabled. This is not needed for all other chip select lines since Mode Fault detection is only on NPCS0.

Figure 46-10. Chip Select Decoding Application Block Diagram: Single Master/Multiple Slave Implementation



46.7.3.8 Peripheral Deselection without DMA

During a transfer of more than one unit of data on a chip select without the DMA, the SPI_TDR is loaded by the processor, the TDRE flag rises as soon as the content of the SPI_TDR is transferred into the internal Shift register. When this flag is detected high, the SPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the chip select is not deasserted between the two transfers. But depending on the application software handling the SPI status register flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload the SPI_TDR in time to keep the chip select active (low). A null DLYBCT value (delay between consecutive transfers) in the SPI_CSR, gives even less time for the processor to reload the SPI_TDR. With some SPI slave peripherals, if the chip select line must remain active (low) during a full set of transfers, communication errors can occur.

To facilitate interfacing with such devices, the chip select registers [SPI_CSR0...SPI_CSR3] can be programmed with the Chip Select Active After Transfer (CSAAT) bit at 1. This allows the chip select lines to remain in their current state (low = active) until a transfer to another chip select is required. Even if the SPI_TDR is not reloaded, the chip select remains active. To deassert the chip select line at the end of the transfer, the Last Transfer (LASTXFER) bit in SPI_CR must be set after writing the last data to transmit into SPI_TDR.

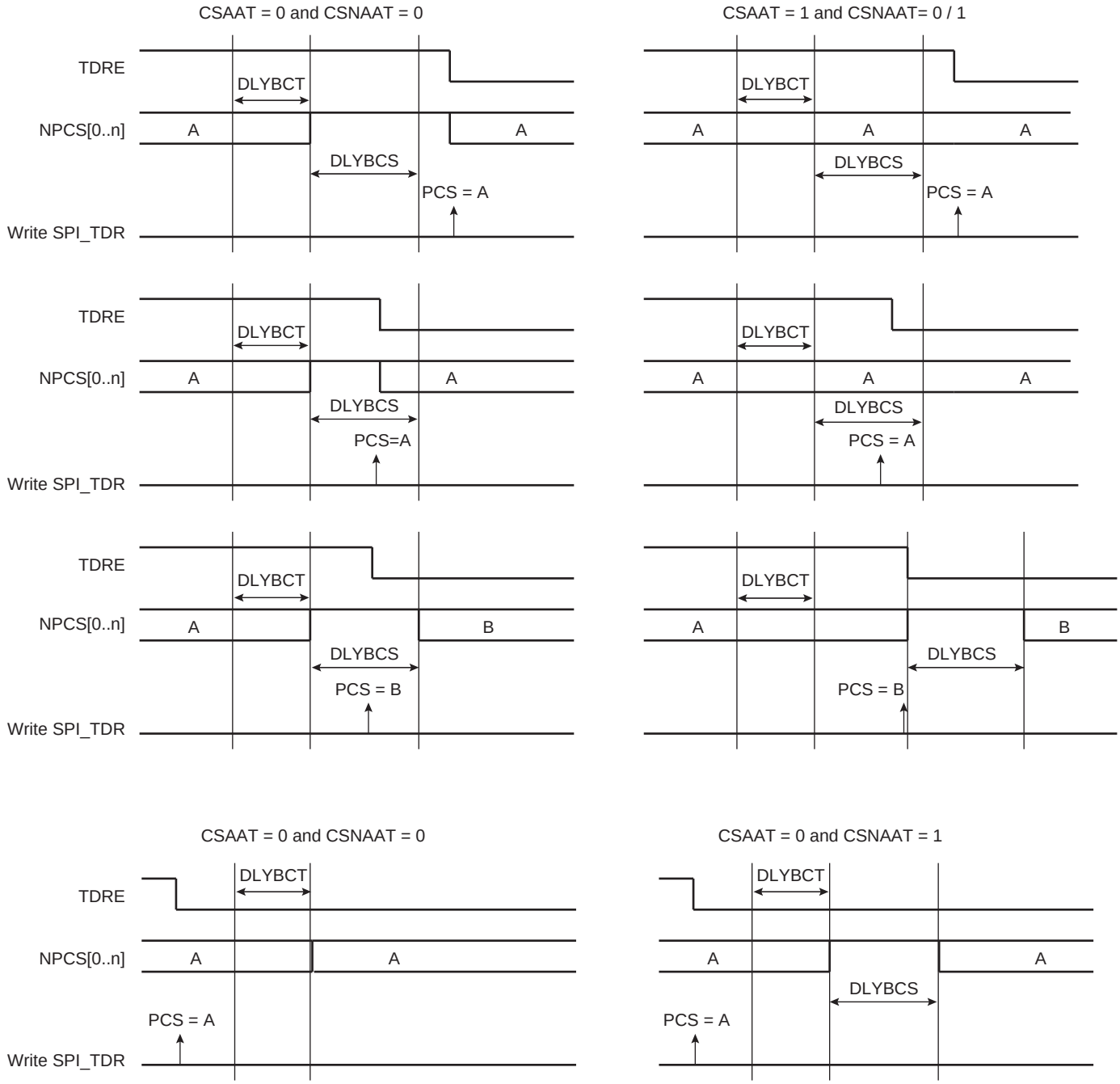
46.7.3.9 Peripheral Deselection with DMA

DMA provides faster reloads of the SPI_TDR compared to software. However, depending on the system activity, it is not guaranteed that the SPI_TDR is written with the next data before the end of the current transfer. Consequently, data can be lost by the deassertion of the NPCS line for SPI slave peripherals requiring the chip select line to remain active between two transfers. The only way to guarantee a safe transfer in this case is the use of the CSAAT and LASTXFER bits.

When the CSAAT bit is configured to 0, the NPCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a chip select, the TDRE flag rises as soon as the content of the SPI_TDR is transferred into the internal shift register. When this flag is detected, the SPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the chip select is not deasserted between the two transfers. This can lead to difficulties to interface with some serial peripherals requiring the chip select to be deasserted after each transfer. To facilitate interfacing with such devices, the SPI_CSR can be programmed with the Chip Select Not Active After Transfer (CSNAAT) bit at 1. This allows the chip select lines to be deasserted systematically during a time “DLYBCS” (the value of the CSNAAT bit is processed only if the CSAAT bit is configured to 0 for the same chip select).

Figure 46-11 shows different peripheral deselection cases and the effect of the CSAAT and CSNAAT bits.

Figure 46-11. Peripheral Deselection



46.7.3.10 Mode Fault Detection

The SPI has the capability to operate in multimaster environment. Consequently, the NPCS0/NSS line must be monitored. If one of the masters on the SPI bus is currently transmitting, the NPCS0/NSS line is low and the SPI must not transmit any data. A mode fault is detected when the SPI is programmed in Master mode and a low level is driven by an external master on the NPCS0/NSS signal. In multimaster environment, NPCS0, MOSI, MISO and SPCK pins must be configured in open drain (through the PIO controller). When a mode fault is detected, the SPI_SR.MODF bit is set until SPI_SR is read and the SPI is automatically disabled until it is reenabled by setting the SPI_CR.SPIEN bit.

By default, the mode fault detection is enabled. The user can disable it by setting the SPI_MR.MODFDIS bit.

46.7.4 SPI Slave Mode

When operating in Slave mode, the SPI processes data bits on the clock provided on the SPI clock pin (SPCK).

The SPI waits until NSS goes active before receiving the serial clock from an external master. When NSS falls, the clock is validated and the data is loaded in the SPI_RDR depending on the BITS field configured in SPI_CSR0. These bits are processed following a phase and a polarity defined respectively by the NCPHA and CPOL bits in SPI_CSR0. Note that the fields BITS, CPOL and NCPHA of the other chip select registers (SPI_CSR1...SPI_CSR3) have no effect when the SPI is programmed in Slave mode.

The bits are shifted out on the MISO line and sampled on the MOSI line.

Note: For more information on the BITS field, see also the note below the SPI_CSRx bitmap (Section 46.8.12 “SPI Chip Select Register”).

When all bits are processed, the received data is transferred in the SPI_RDR and the RDRF bit rises. If the SPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in the SPI_SR is set. As long as this flag is set, data is loaded in the SPI_RDR. The user must read SPI_SR to clear the OVRES bit.

When a transfer starts, the data shifted out is the data present in the Shift register. If no data has been written in the SPI_TDR, the last data received is transferred. If no data has been received since the last reset, all bits are transmitted low, as the Shift register resets to 0.

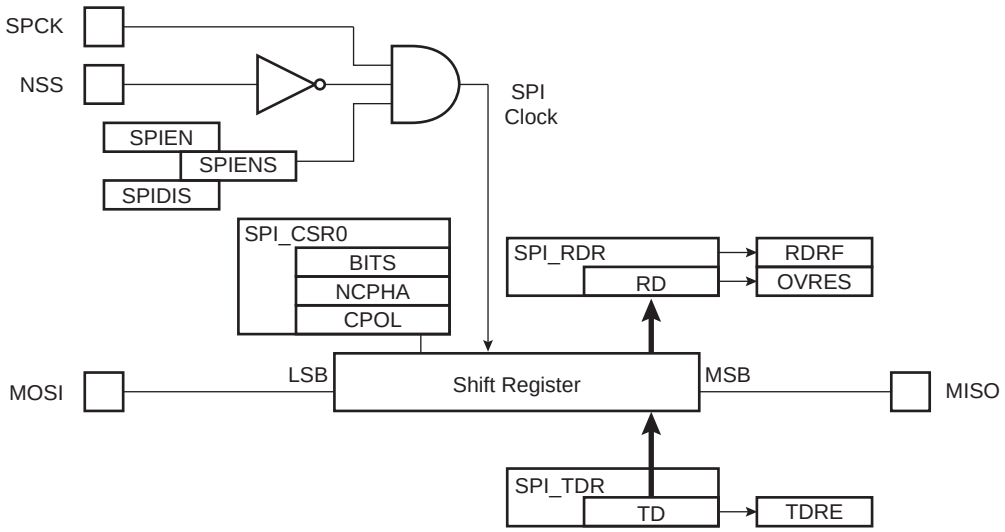
When a first data is written in the SPI_TDR, it is transferred immediately in the Shift register and the TDRE flag rises. If new data is written, it remains in the SPI_TDR until a transfer occurs, i.e., NSS falls and there is a valid clock on the SPCK pin. When the transfer occurs, the last data written in the SPI_TDR is transferred in the Shift register and the TDRE flag rises. This enables frequent updates of critical variables with single transfers.

Then, new data is loaded in the Shift register from the SPI_TDR. If no character is ready to be transmitted, i.e., no character has been written in the SPI_TDR since the last load from the SPI_TDR to the Shift register, the SPI_TDR is retransmitted. In this case the Underrun Error Status Flag (UNDES) is set in the SPI_SR.

If NSS rises between two characters, it must be kept high for two MCK clock periods or more and the next SPCK capture edge must not occur less than four MCK periods after NSS rise.

Figure 46-12 shows a block diagram of the SPI when operating in Slave mode.

Figure 46-12. Slave Mode Functional Block Diagram



46.7.5 SPI Comparison Function on Received Character

The comparison is on ly relevant for SPI Slave mode (MSTR = 0 in SPI_MR).

The effect of a comparison match changes if the system is in Wait or Active mode.

In Wait mode, if asynchronous partial wakeup is enabled, a system wakeup is performed (see Section 46.7.6).

In Active mode, the CMP flag in SPI_SR is raised. It is set when the received character matches the conditions programmed in the SPI Comparison Register (SPI_CMPR). The CMP flag is set as soon as SPI_RDR is loaded with the new received character. The CMP flag is cleared by reading SPI_SR.

SPI_CMPR (see Section 46.8.15) can be programmed to provide different comparison methods. These are listed below:

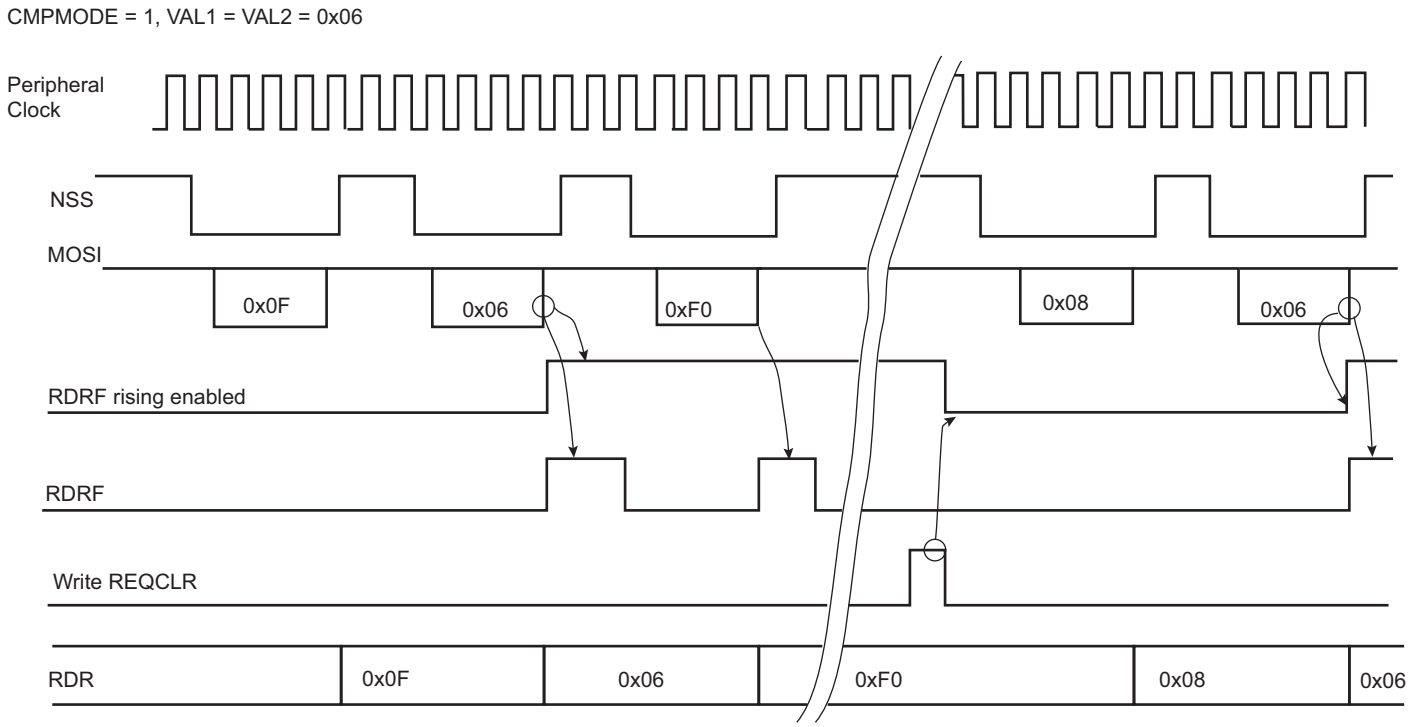
- If VAL1 equals VAL2, then the comparison is performed on a single value and the flag is set to 1 if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 sets the CMP flag.
- If VAL1 is strictly higher than VAL2, then the flag CMP is set to 1 if any received character equals VAL1 or VAL2.

When the CMPMODE bit is cleared in SPI_CMPR, all received data is loaded in SPI_RDR and the CMP flag provides the status of the comparison result.

By setting the CMPMODE bit, the comparison result triggers the start of the SPI_RDR loading (see Figure 46-13). The trigger condition exists as soon as the received character value matches the conditions defined by VAL1 and VAL2 in SPI_CMPR. The comparison trigger event is restarted by setting the REQCLR bit in SPI_CR if SleepWalking is disabled.

The value programmed in VAL1 and VAL2 fields must not exceed the maximum value of the received character (see BITS field in SPI_CSR0).

Figure 46-13. Receive Data Register Management



46.7.6 SPI Asynchronous and Partial Wakeup (SleepWalking)

This operating mode is a means of data preprocessing that qualifies an incoming event, thus allowing the SPI to decide whether or not to wake up the system. Asynchronous and partial wakeup is mainly used when the system is in Wait mode (see the PMC section for further details). It can also be enabled when the system is fully running.

Asynchronous and partial wakeup can be used only when SPI is configured in Slave mode (MSTR is cleared in SPI_MR).

The maximum SPI clock (SPCK) frequency that can be provided by the SPI master is bounded by the peripheral clock frequency. The SPCK frequency must be lower than or equal to the peripheral clock. The NSS line must be deasserted by the SPI master between two characters. The NSS deassertion duration time must be greater than or equal to six peripheral clock periods. The time between the assertion of NSS line (falling edge) and the first edge of the SPI clock must be higher than 5 µs.

The SPI_RDR must be read before enabling the asynchronous and partial wakeup.

When asynchronous and partial wakeup is enabled for the SPI (see the PMC section), the PMC decodes a clock request from the SPI. The request is generated as soon as there is a falling edge on the NSS line as this may indicate the beginning of a frame. If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the SPI.

The SPI processes the received frame and compares the received character with VAL1 and VAL2 in SPI_CMPR (Section 46.8.15).

The SPI instructs the PMC to disable the peripheral clock if the received character value does not meet the conditions defined by VAL1 and VAL2 fields in SPI_CMPR (see Figure 46-15).

If the received character value meets the conditions, the SPI instructs the PMC to exit the system from Wait mode (see Figure 46-14).

The VAL1 and VAL2 fields can be programmed to provide different comparison methods and thus matching conditions.

- If VAL1 = VAL2, then the comparison is performed on a single value and the wakeup is triggered if the received character equals VAL1.
- If VAL1 is strictly lower than VAL2, then any value between VAL1 and VAL2 wakes up the system.
- If VAL1 is strictly higher than VAL2, the wakeup is triggered if any received character equals VAL1 or VAL2.
- If VAL1 = 0 and VAL2 = 65535, the wakeup is triggered as soon as a character is received.

If the processor and peripherals are running, the SPI can be configured in Asynchronous and Partial Wakeup mode by enabling the PMC_SLPWK_ER (see PMC section). When activity is detected on the receive line, the SPI requests the clock from the PMC and the comparison is performed. If there is a comparison match, the SPI continues to request the clock. If there is no match, the clock is switched off for the SPI only, until a new activity is detected.

The CMPMODE configuration has no effect when Asynchronous and Partial Wakeup mode is enabled for the SPI (see PMC_SLPWK_ER in PMC section).

When the system is in Active mode and the SPI enters Asynchronous and Partial Wakeup mode, the flag RDRF must be programmed as the unique source of the SPI interrupt.

When the system exits Wait mode as the result of a matching condition, the RDRF flag is used to determine if the SPI is the source for the exit from Wait mode.

Figure 46-14. Asynchronous Wakeup Use Case Example

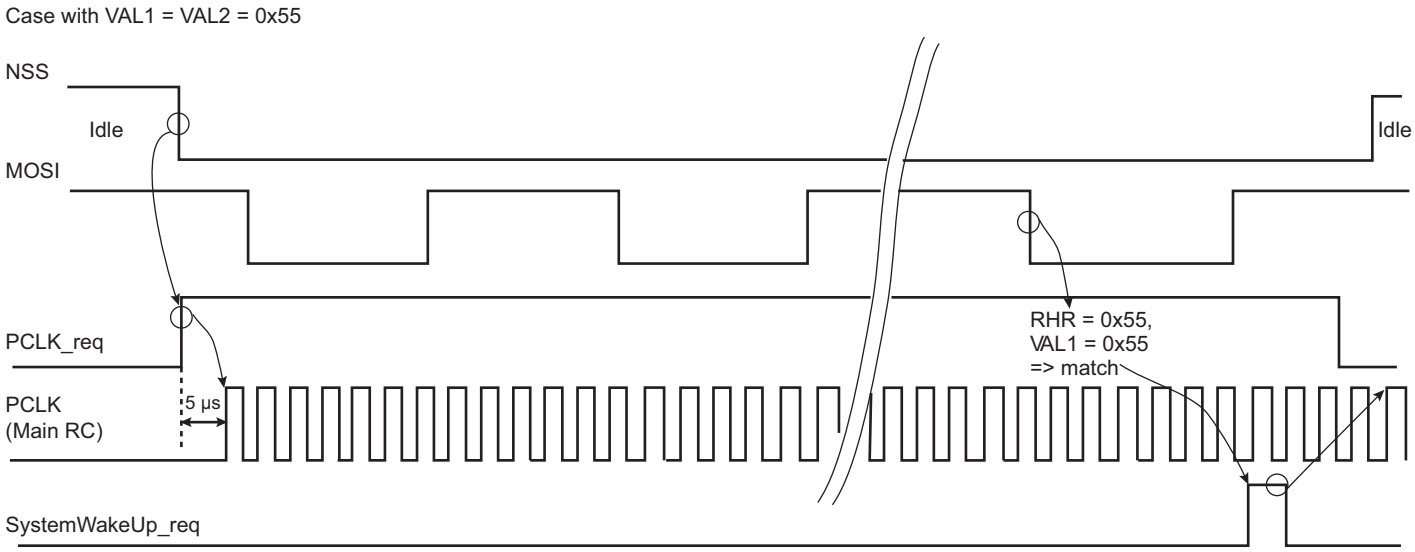
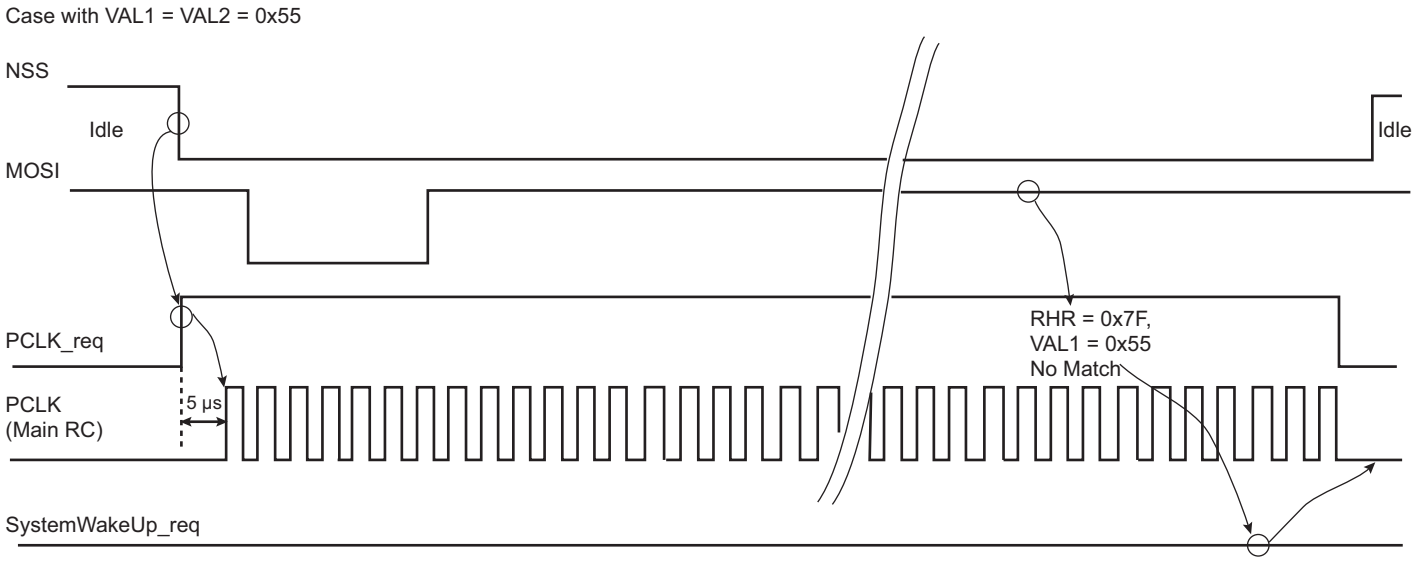


Figure 46-15. Asynchronous Event Generating Only Partial Wakeup

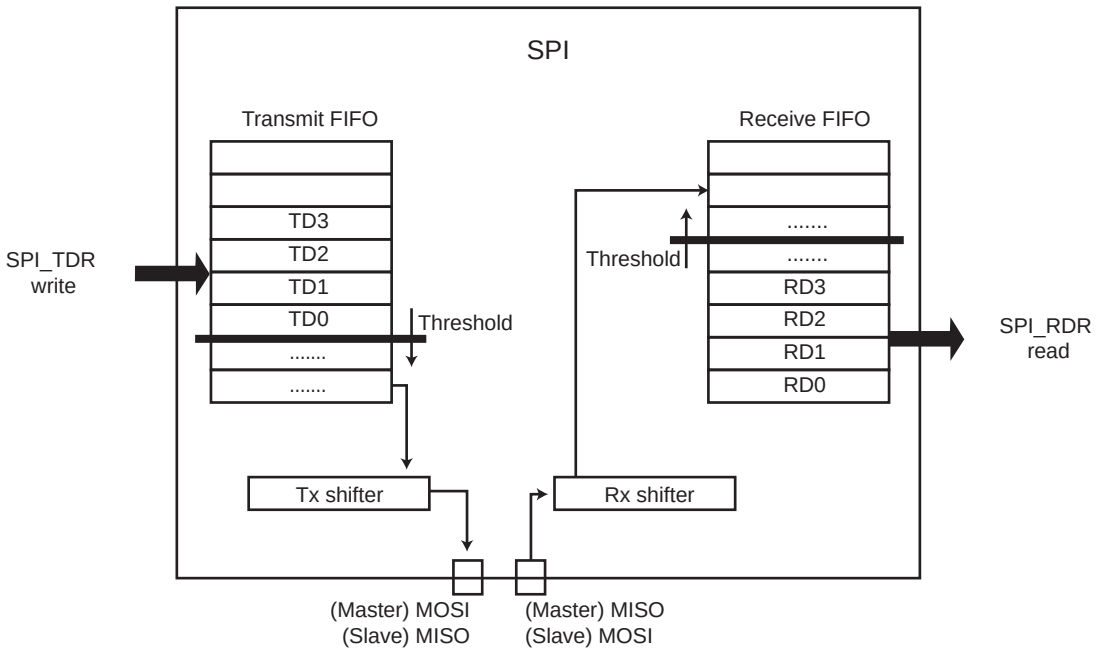


46.7.7 FIFOs

The SPI includes two FIFOs which can be enabled/disabled using the FIFOEN/FIFODIS bits in the SPI_CR. It is recommended to disable the SPI module before enabling or disabling the SPI FIFOs (TXDIS and RXDIS bit in SPI_CR).

Writing the FIFOEN bit to ‘1’ enables a 16-data Transmit FIFO and a 16-data Receive FIFO.

Figure 46-16. FIFOs Block Diagram

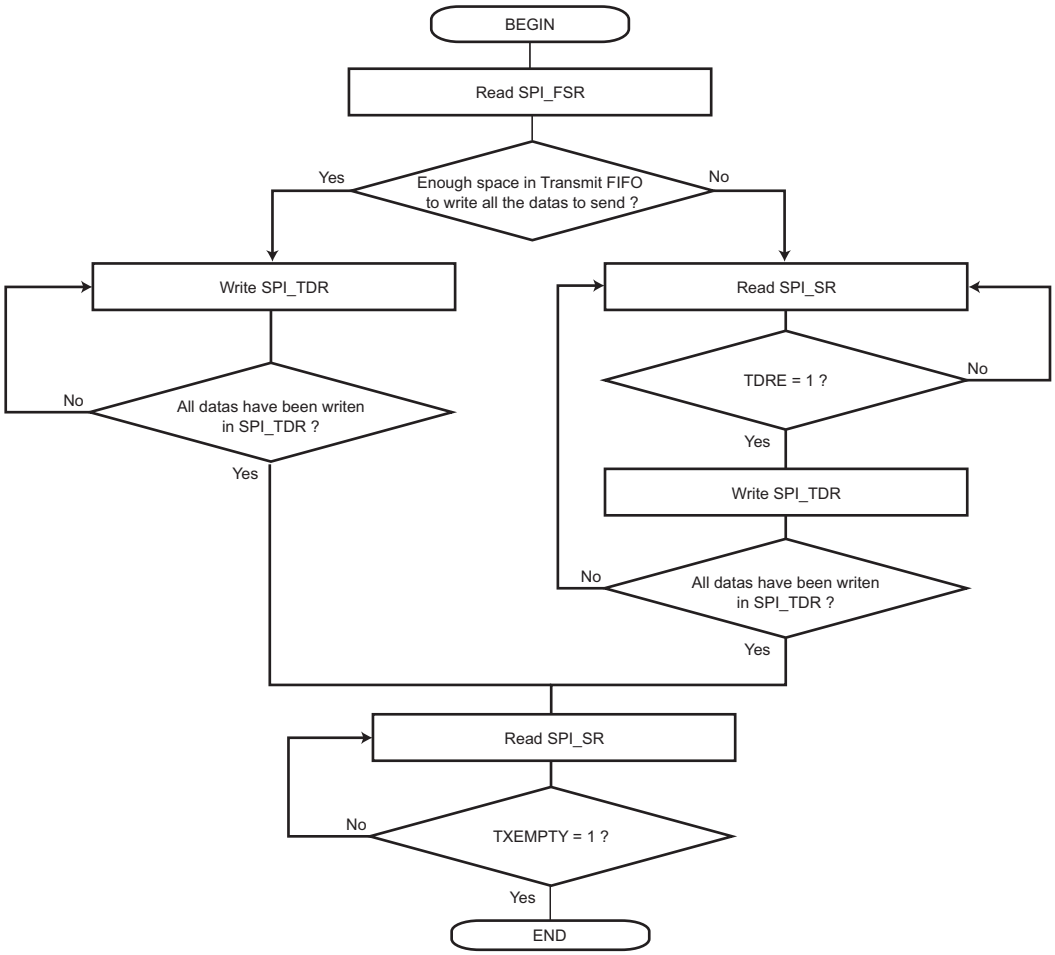


46.7.7.1 Sending Data with FIFO Enabled

With the Transmit FIFO enabled, any write access to the SPI_TDR brings the written data to the Transmit FIFO. As a consequence, it is not mandatory any more to monitor the TDRE flag state to send multiple data without DMAC.

Knowing the number of data to send, and provided there is enough space in the Transmit FIFO, all the data to send can be written successively in the SPI_TDR without checking the TDRE flag between each access. The Transmit FIFO state can be checked reading the TXFL field in the SPI FIFO Level Register (SPI_FLR).

Figure 46-17. Sending Data with FIFO Flowchart

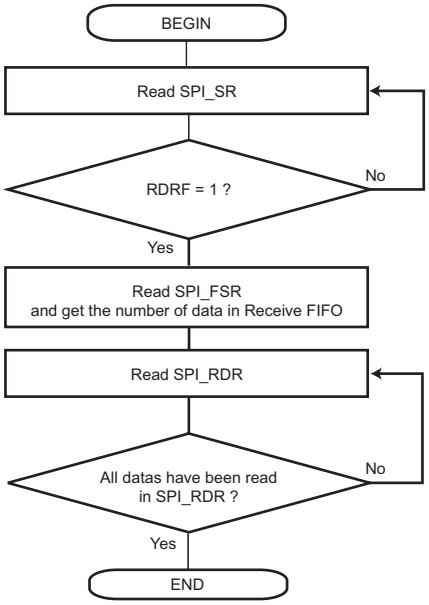


46.7.7.2 Receiving Data with FIFO Enabled

With Receive FIFO enabled, any read access on SPI_RDR pulls out a data from the Receive FIFO. As a consequence, it is not mandatory any more to monitor the RDRF flag when DMAC is not used and there are multiple data to read.

When data are present in the Receive FIFO (RDRF flag set to ‘1’), the exact number of data can be checked with the RXFL field in the SPI_FLR and all the data read successively in the SPI_RDR without checking the RDRF flag between each access.

Figure 46-18. Receiving Data with FIFO Flowchart



46.7.7.3 Clearing/Flushing FIFOs

Each FIFO can be cleared/flushed using the TXFCLR and RXFCLR bits in the SPI_CR.

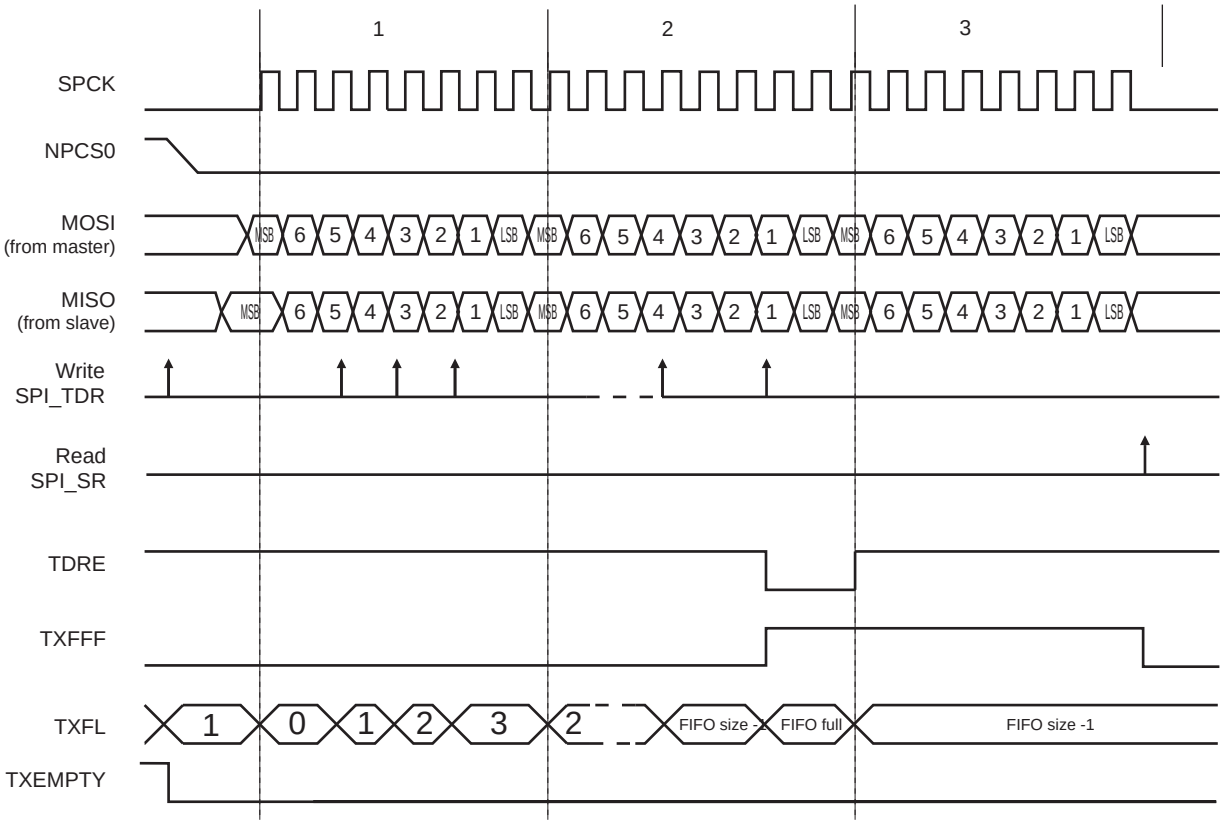
46.7.7.4 TDRE, RDRF and TXEMPTY Behavior

If FIFOs are enabled, the behavior of the TDRE, RDRF and TXEMPTY flags is slightly different.

With FIFO enabled, the TXEMPTY flag remains at ‘0’ state as long as there are characters in the Transmit FIFO or in the Transmit Shift Register. It is at ‘1’ state when there are no character in the Transmit FIFO and no character in the Transmit Shift Register.

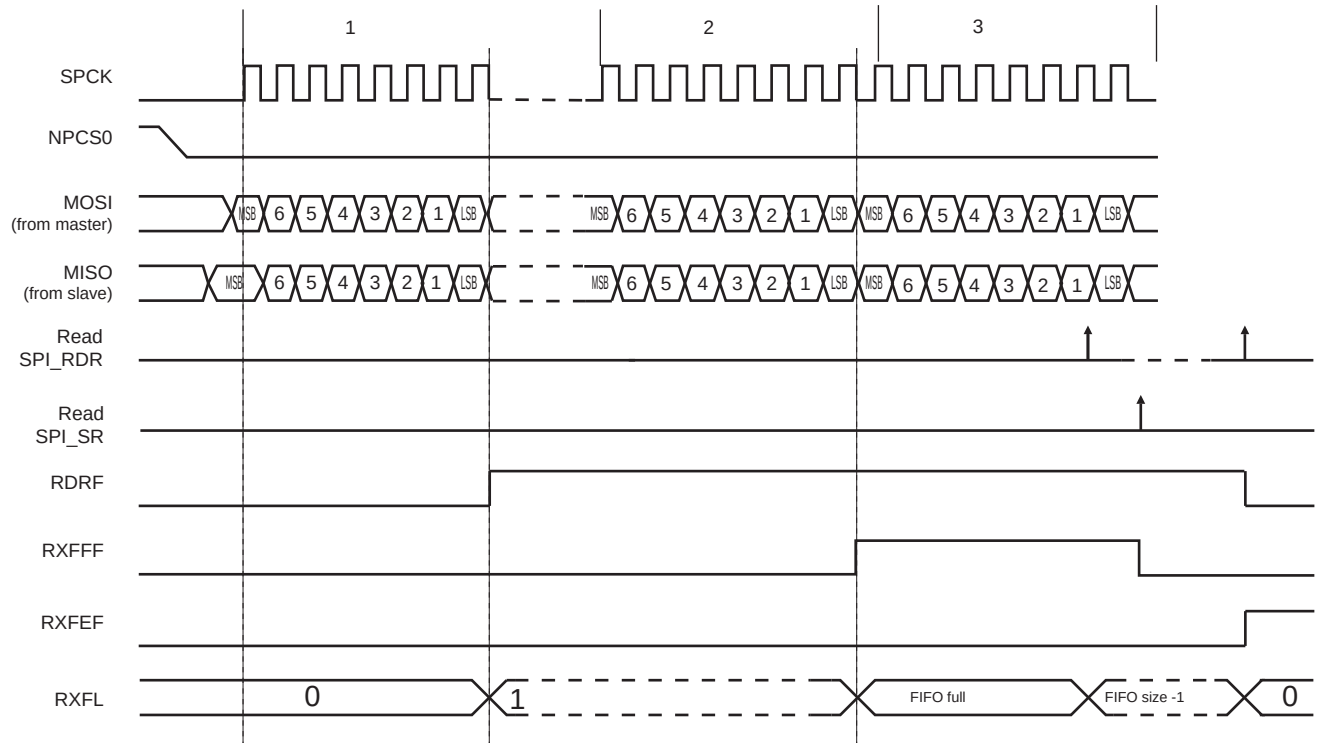
TDRE indicates if a data can be written in the Transmit FIFO. By default, the TDRE flag then stays at level ‘1’ as long as the Transmit FIFO is not full (TXRDYM = 0x0).

Figure 46-19. TDRE in Single Data Mode and TXRDYM = 0x0



RDRF indicates if an unread data is present in the Receive FIFO. By default, the RDRF flag is then at level ‘1’ as soon as one unread data is in the Receive FIFO (RXRDYM = 0x0).

Figure 46-20. RDRF in Single Data Mode and RXRDYM = 0x0



TDRE and RDRF behavior can be modified using the TXRDYM and RXRDYM fields in the SPI FIFO Mode Register (SPI_FMR). In Single Data mode, there is no need to modify the TDRE and RDRF behavior; however, it may be useful in Multiple Data mode.

46.7.7.5 Single Data Mode

If Variable Peripheral Select mode is used (PS bit set to '1' in SPI_MR), the Transmit FIFO operates in Single Data mode.

If Master mode is set (MSTR bit set to '1' in SPI_MR), the Receive FIFO operates in Single Data mode.

In this mode, only one data can be written/read per write/read access of SPI_TDR/SPI_RDR (see Section 46.8.3 "SPI Receive Data Register"). On the other hand, TXRDYM and RXRDYM fields should be configured with 0x0 value in this mode.

DMAC

TXRDYM and RXRDYM fields must be configured with 0x0 value when working with DMAC transfer in Single Data mode.

The same DMAC procedure applies as with FIFO disabled.

46.7.7.6 Multiple Data Mode

When the FIFOs do not operate in Single Data mode, they operate in Multiple Data mode.

In Multiple Data mode, it is possible to write up to two data in one SPI_TDR write access. It is possible, in this mode, to read up to four data in one SPI_RDR access if the BITS field is configured to 0 (8-bit data size) and up to two data if the BITS field value is other than 0 (more than 8-bit data size).

The number of data to write/read is defined by the size of the register access. If the access is a byte-size register access, only one data is written/read. If the access is a halfword size register access, then up to two data are read/only one data is written and finally, if the access is a word-size register access, then up to four data are read/up to two data are written.

Written/Read data are always right-aligned, as shown in Section 46.8.4 “SPI Receive Data Register (FIFO Multiple Data, 8-bit)”, Section 46.8.5 “SPI Receive Data Register (FIFO Multiple Data, 16-bit)” and Section 46.8.7 “SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)”.

For instance, if the Transmit FIFO is empty and there are six data to send, you can either:

- Perform six SPI_TDR-byte write accesses.
- Perform three SPI_TDR-halfword write accesses.

It goes the same with a Receive FIFO containing six data where you can either:

- Perform six SPI_RDR-byte read accesses.
- Perform three SPI_RDR-halfword read accesses.
- Perform one SPI_RDR-word read access and one SPI_RDR-halfword read access.

This mode allows to minimize the number of accesses by concatenating the data to send/read in one access.

TDRE and RDRF Configuration

In Multiple Data mode, the TXRDYM and RXRDYM fields in the SPI_FMR become useful.

As in Multiple Data mode, it is possible to write several data in the same access it might be useful to configure TDRE flag behavior to indicate if one or two data can be written in the FIFO depending on the access to perform on SPI_TDR.

If for instance two data are written each time in the SPI_TDR, it might be useful to configure the TXRDYM field to 0x1 value so that the TDRE flag is at ‘1’ only when at least two data can be written in the Transmit FIFO.

In the same way, if four data are read each time in the SPI_RDR, it might be useful to configure the RXRDYM field to 0x2 value so that the RDRF flag is at ‘1’ only when at least four unread data are in the Receive FIFO.

DMAC

If DMAC transfer is used, it is mandatory to configure TXRDYM/RXRDYM to the right value depending on the DMAC channel size (byte, halfword or word).

46.7.7.7 FIFO Pointer Error

In some specific cases, it is possible to generate a FIFO pointer error.

- Transmit FIFO:

If the Transmit FIFO is full and a write access is performed on the SPI_TDR, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in SPI_SR.

In Multiple Data mode, if the number of data written in the SPI_TDR (according to the register access size) is bigger than the Transmit FIFO free space, it generates a Transmit FIFO pointer error and sets the TXFPTEF flag in SPI_SR.

- Receive FIFO:

In Multiple Data mode, if the number of data read in the SPI_RDR (according to the register access size) is bigger than the number of unread data in the Receive FIFO, it generates a Receive FIFO pointer error and sets the RXFPTEF flag in SPI_SR.

No pointer error should happen if the FIFO state is checked before writing/reading in SPI_TDR/SPI_RDR. The FIFO state can be checked either with TXRDY, RXRDY, TXFL or RXFL. When a pointer error occurs, other FIFO flags might not behave as expected; their state should be ignored.

If a pointer error occurs, a software reset must be performed through the SWRST bit in SPI_CR (configuration will be lost).

46.7.7.8 FIFO Thresholds

Each Transmit and Receive FIFO includes a threshold feature used to set a flag and an interrupt when a FIFO threshold is crossed. Thresholds are defined as a number of data in the FIFO, and the FIFO state (TXFL or RXFL) represents the number of data currently in the FIFO.

- Transmit FIFO:

The Transmit FIFO threshold can be set using the TXFTHRES field in SPI_FMR. Each time the Transmit FIFO goes from the ‘above threshold’ to the ‘equal or below threshold’ state, the TXFTHF flag in SPI_SR is set. This enables the application to know that the Transmit FIFO reached the defined threshold and to refill it before it becomes empty.

- Receive FIFO:

The Receive FIFO threshold can be set using the RXFTHRES field in SPI_FMR. Each time the Receive FIFO goes from the ‘below threshold’ to the ‘equal or above threshold’ state, the RXFTHF flag in SPI_SR is set. This enables the application to know that the Receive FIFO reached the defined threshold and to read some data before it becomes full.

The TXFTHF and RXFTHF flags can be both configured to generate an interrupt using SPI_IER and SPI_IDR.

46.7.7.9 FIFO Flags

FIFOs come with a set of flags which can be configured each to generate interrupt through SPI_IER and SPI_IDR.

FIFO flags state can be read in SPI_SR. They are cleared on SPI_SR read.

46.7.8 Register Write Protection

To prevent any single software error from corrupting SPI behavior, certain registers in the address space can be write-protected in the SPI Write Protection Mode Register (SPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the SPI Write Protection Status Register (SPI_WPSR) is set and the WPVSRC field indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading SPI_WPSR.

The following registers are write-protected when WPEN is set in SPI_WPMR:

- SPI Mode Register
- SPI Chip Select Register

46.8 Serial Peripheral Interface (SPI) User Interface

In the “Offset” column of Table 46-5, ‘CS_number’ denotes the chip select number.

Table 46-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	SPI_CR	Write-only	–
0x04	Mode Register	SPI_MR	Read/Write	0x0
0x08	Receive Data Register	SPI_RDR	Read-only	0x0
0x0C	Transmit Data Register	SPI_TDR	Write-only	–
0x10	Status Register	SPI_SR	Read-only	0x0
0x14	Interrupt Enable Register	SPI_IER	Write-only	–
0x18	Interrupt Disable Register	SPI_IDR	Write-only	–
0x1C	Interrupt Mask Register	SPI_IMR	Read-only	0x0
0x20–0x2C	Reserved	–	–	–
0x30 + (CS_number * 0x04)	Chip Select Register	SPI_CSR	Read/Write	0x0
0x40	FIFO Mode Register	SPI_FMR	Read/Write	0x0
0x44	FIFO Level Register	SPI_FLR	Read-only	0x0
0x48	Comparison Register	SPI_CMPR	Read-only	0x0
0x4C–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	SPI_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	SPI_WPSR	Read-only	0x0
0xEC–0xF8	Reserved	–	–	–
0xFC	Reserved	–	–	–

46.8.1 SPI Control Register

Name: SPI_CR

Address: 0xF8000000 (0), 0xFC000000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
FIFODIS	FIFOEN	–	–	–	–	–	LASTXFER
23	22	21	20	19	18	17	16
–	–	–	–	–	–	RXFCLR	TXFCLR
15	14	13	12	11	10	9	8
–	–	–	REQCLR	–	–	–	–
7	6	5	4	3	2	1	0
SWRST	–	–	–	–	–	SPIDIS	SPIEN

- **SPIEN: SPI Enable**

0: No effect.

1: Enables the SPI to transfer and receive data.

- **SPIDIS: SPI Disable**

0: No effect.f

1: Disables the SPI.

All pins are set in Input mode after completion of the transmission in progress, if any.

If a transfer is in progress when SPIDIS is set, the SPI completes the transmission of the shifter register and does not start any new transfer, even if the SPI_THR is loaded.

Note: If both SPIEN and SPIDIS are equal to one when the SPI_CR is written, the SPI is disabled.

- **SWRST: SPI Software Reset**

0: No effect.

1: Reset the SPI. A software-triggered hardware reset of the SPI interface is performed.

The SPI is in Slave mode after software reset.

- **REQCLR: Request to Clear the Comparison Trigger**

SleepWalking enabled:

0: No effect.

1: Clears the potential clock request currently issued by SPI, thus the potential system wakeup is cancelled.

SleepWalking disabled:

0: No effect.

1: Restarts the comparison trigger to enable SPI_RDR loading.

- **TXFCLR: Transmit FIFO Clear**

0: No effect.

1: Clears the Transmit FIFO, Transmit FIFO will become empty.

- **RXFCLR: Receive FIFO Clear**

0: No effect.

1: Clears the Receive FIFO, Receive FIFO will become empty.

- **LASTXFER: Last Transfer**

0: No effect.

1: The current NPCS is deasserted after the character written in TD has been transferred. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

Refer to Section 46.7.3.5 “Peripheral Selection” for more details.

- **FIFOEN: FIFO Enable**

0: No effect.

1: Enables the Transmit and Receive FIFOs

- **FIFODIS: FIFO Disable**

0: No effect.

1: Disables the Transmit and Receive FIFOs

46.8.2 SPI Mode Register

Name: SPI_MR

Address: 0xF8000004 (0), 0xFC000004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYBCS							
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
–	–	–	CMPMODE	–	–	–	LSBHALF
7	6	5	4	3	2	1	0
LLB	–	WDRBT	MODFDIS	BRSRCCLK	PCSDEC	PS	MSTR

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

- **MSTR: Master/Slave Mode**

0: SPI is in Slave mode

1: SPI is in Master mode

- **PS: Peripheral Select**

0: Fixed Peripheral Select

1: Variable Peripheral Select

- **PCSDEC: Chip Select Decode**

0: The chip select lines are directly connected to a peripheral device.

1: The four NPCS chip select lines are connected to a 4-bit to 16-bit decoder.

When PCSDEC = 1, up to 15 chip select signals can be generated with the four NPCS lines using an external 4-bit to 16-bit decoder. The chip select registers define the characteristics of the 15 chip selects, with the following rules:

SPI_CSR0 defines peripheral chip select signals 0 to 3.

SPI_CSR1 defines peripheral chip select signals 4 to 7.

SPI_CSR2 defines peripheral chip select signals 8 to 11.

SPI_CSR3 defines peripheral chip select signals 12 to 14.

- **BRSRCCLK: Bit Rate Source Clock**

0 (PERIPH_CLK): The peripheral clock is the source clock for the bit rate generation.

1 (GCLK): PMC GCLK is the source clock for the bit rate generation, thus the bit rate can be independent of the core/peripheral clock.

Note: If bit BRSRCCLK = 1, the SCBR field in SPI_CSRx must be programmed with a value greater than 1.

- **MODFDIS: Mode Fault Detection**

0: Mode fault detection enabled

1: Mode fault detection disabled

- **WDRBT: Wait Data Read Before Transfer**

0: No Effect. In Master mode, a transfer can be initiated regardless of the SPI_RDR state.

1: In Master mode, a transfer can start only if the SPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

- **LLB: Local Loopback Enable**

0: Local loopback path disabled.

1: Local loopback path enabled.

LLB controls the local loopback on the data shift register for testing in Master mode only (MISO is internally connected on MOSI).

- **LSBHALF: LSB Timing Selection**

0: To be used only if SPI slave LSB timing is 100% compliant with SPI standard (LSB duration is a full bit time). This value gives the better margin for SPI slave response delay (less than 1 SPCK clock cycle).

1: To be selected if the SPI slave LSB timing does not behave as the SPI standard (not triggered by NPCS deassertion in mode), the slave response delay is limited to less than 1/2 SPCK cycle.

- **CMPMODE: Comparison Mode**

Value	Name	Description
0	FLAG_ONLY	Any character is received and comparison function drives CMP flag.
1	START_CONDITION	Comparison condition must be met to start reception of all incoming characters until REQCLR is set.

- **PCS: Peripheral Chip Select**

This field is only used if fixed peripheral select is active (PS = 0).

If SPI_MR.PCSDEC = 0:

PCS = xxx0 NPCS[3:0] = 1110

PCS = xx01 NPCS[3:0] = 1101

PCS = x011 NPCS[3:0] = 1011

PCS = 0111 NPCS[3:0] = 0111

PCS = 1111 forbidden (no peripheral is selected)

(x = don’t care)

If SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

- **DLYBCS: Delay Between Chip Selects**

This field defines the delay between the inactivation and the activation of NPCS. The DLYBCS time guarantees nonoverlapping chip selects and solves bus contentions in case of peripherals having long data float times.

If DLYBCS is lower than 6, six peripheral clock periods are inserted by default.

Otherwise, the following equations determine the delay:

If BRSRCCLK = 0:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{f_{\text{peripheral clock}}}$$

If BRSRCCLK = 1:

$$\text{Delay Between Chip Selects} = \frac{\text{DLYBCS}}{f_{\text{GCLK}}}$$

46.8.3 SPI Receive Data Register

Name: SPI_RDR

Address: 0xF8000008 (0), 0xFC000008 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	PCS			
15	14	13	12	11	10	9	8
RD							
7	6	5	4	3	2	1	0
RD							

- **RD: Receive Data**

Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

- **PCS: Peripheral Chip Select**

In Master mode only, these bits indicate the value on the NPCS pins at the end of a transfer. Otherwise, these bits are read as zero.

Note: When using Variable Peripheral Select mode (PS = 1 in SPI_MR), it is mandatory to set the SPI_MR.WDRBT bit if the PCS field must be processed in SPI_RDR.

46.8.4 SPI Receive Data Register (FIFO Multiple Data, 8-bit)

Name: SPI_RDR (FIFO_MULTI_DATA_8)

Address: 0xF8000008 (0), 0xFC000008 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RD3							
23	22	21	20	19	18	17	16
RD2							
15	14	13	12	11	10	9	8
RD1							
7	6	5	4	3	2	1	0
RD0							

Note: If FIFO is enabled (FIFOEN bit in SPI_CR), refer to Section 46.7.7.6 “Multiple Data Mode”.

- **RDx: Receive Data**

First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

46.8.5 SPI Receive Data Register (FIFO Multiple Data, 16-bit)

Name: SPI_RDR (FIFO_MULTI_DATA_16)

Address: 0xF8000008 (0), 0xFC000008 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RD1							
23	22	21	20	19	18	17	16
RD1							
15	14	13	12	11	10	9	8
RD0							
7	6	5	4	3	2	1	0
RD0							

Note: If FIFO is enabled (FIFOEN bit in SPI_CR), refer to Section 46.7.7.6 “Multiple Data Mode”.

- **RDx: Receive Data**

First unread data in the Receive FIFO. Data received by the SPI Interface is stored in this register in a right-justified format. Unused bits are read as zero.

46.8.6 SPI Transmit Data Register

Name: SPI_TDR

Address: 0xF800000C (0), 0xFC00000C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	LASTXFER

23	22	21	20	19	18	17	16
–	–	–	–	PCS			

15	14	13	12	11	10	9	8
TD							

7	6	5	4	3	2	1	0
TD							

- **TD: Transmit Data**

Data to be transmitted by the SPI Interface is stored in this register. Information to be transmitted must be written to the transmit data register in a right-justified format.

- **PCS: Peripheral Chip Select**

This field is only used if variable peripheral select is active (PS = 1).

If SPI_MR.PCSDEC = 0:

PCS = xxx0 NPCS[3:0] = 1110

PCS = xx01 NPCS[3:0] = 1101

PCS = x011 NPCS[3:0] = 1011

PCS = 0111 NPCS[3:0] = 0111

PCS = 1111 forbidden (no peripheral is selected)

(x = don’t care)

If SPI_MR.PCSDEC = 1:

NPCS[3:0] output signals = PCS.

- **LASTXFER: Last Transfer**

0: No effect

1: The current NPCS is deasserted after the transfer of the character written in TD. When SPI_CSRx.CSAAT is set, the communication with the current serial peripheral can be closed by raising the corresponding NPCS line as soon as TD transfer is completed.

This field is only used if variable peripheral select is active (SPI_MR.PS = 1).

46.8.7 SPI Transmit Data Register (FIFO Multiple Data, 8- to 16-bit)

Name: SPI_TDR (FIFO_MULTI_DATA)

Address: 0xF800000C (0), 0xFC00000C (1)

Access: Write-only

31	30	29	28	27	26	25	24
TD1							
23	22	21	20	19	18	17	16
TD1							
15	14	13	12	11	10	9	8
TD0							
7	6	5	4	3	2	1	0
TD0							

Note: If FIFO is enabled (FIFOEN bit in SPI_CR), refer to Section 46.7.7.6 “Multiple Data Mode”.

- **TDx: Transmit Data**

Next data to write in the Transmit FIFO. Information to be transmitted must be written to the transmit data register in a right-justified format.

46.8.8 SPI Status Register

Name: SPI_SR

Address: 0xF8000010 (0), 0xFC000010 (1)

Access: Read-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	SPIENS
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

- **RDRF: Receive Data Register Full (cleared by reading SPI_RDR)**

0: No data has been received since the last read of SPI_RDR.

1: Data has been received and the received data has been transferred from the shift register to SPI_RDR since the last read of SPI_RDR.

- **TDRE: Transmit Data Register Empty (cleared by writing SPI_TDR)**

0: Data has been written to SPI_TDR and not yet transferred to the shift register.

1: The last data written in the SPI_TDR has been transferred to the shift register.

TDRE equals zero when the SPI is disabled or at reset. The SPI enable command sets this bit to 1.

- **MODF: Mode Fault Error (cleared on read)**

0: No mode fault has been detected since the last read of SPI_SR.

1: A mode fault occurred since the last read of SPI_SR.

- **OVRES: Overrun Error Status (cleared on read)**

0: No overrun has been detected since the last read of SPI_SR.

1: An overrun has occurred since the last read of SPI_SR.

An overrun occurs when SPI_RDR is loaded at least twice from the shift register since the last read of the SPI_RDR.

- **NSSR: NSS Rising (cleared on read)**

0: No rising edge detected on NSS pin since the last read of SPI_SR.

1: A rising edge occurred on NSS pin since the last read of SPI_SR.

- **TXEMPTY: Transmission Registers Empty (cleared by writing SPI_TDR)**

0: As soon as data is written in SPI_TDR.

1: SPI_TDR and internal shift register are empty. If a transfer delay has been defined, TXEMPTY is set after the end of this delay.

- **UNDES: Underrun Error Status (Slave mode only) (cleared on read)**

0: No underrun has been detected since the last read of SPI_SR.

1: A transfer starts whereas no data has been loaded in SPI_TDR.

- **CMP: Comparison Status (cleared on read)**

0: No received character matched the comparison criteria programmed in VAL1 and VAL2 fields in SPI_CMPR since the last read of SPI_SR.

1: A received character matched the comparison criteria since the last read of SPI_SR.

- **SPIENS: SPI Enable Status**

0: SPI is disabled.

1: SPI is enabled.

- **TXFEF: Transmit FIFO Empty Flag (cleared on read)**

0: Transmit FIFO is not empty.

1: Transmit FIFO has been emptied since the last read of SPI_SR.

- **TXFFF: Transmit FIFO Full Flag (cleared on read)**

0: Transmit FIFO is not full or TXFF flag has been cleared.

1: Transmit FIFO has been filled since the last read of SPI_SR.

- **TXFTHF: Transmit FIFO Threshold Flag (cleared on read)**

0: Number of DATA in Transmit FIFO is above TXFTHRES threshold.

1: Number of DATA in Transmit FIFO has reached TXFTHRES threshold since the last read of SPI_SR.

- **RXFEF: Receive FIFO Empty Flag**

0: Receive FIFO is not empty or RXFE flag has been cleared.

1: Receive FIFO has become empty (coming from “not empty” state to “empty” state).

- **RXFFF: Receive FIFO Full Flag**

0: Receive FIFO is not empty or RXFE flag has been cleared.

1: Receive FIFO has become full (coming from “not full” state to “full” state).

- **RXFTHF: Receive FIFO Threshold Flag**

0: Number of unread DATA in Receive FIFO is below RXFTHRES threshold or RXFTH flag has been cleared.

1: Number of unread DATA in Receive FIFO has reached RXFTHRES threshold (coming from “below threshold” state to “equal or above threshold” state).

- **TXFPTEF: Transmit FIFO Pointer Error Flag**

0: No Transmit FIFO pointer occurred

1: Transmit FIFO pointer error occurred. Transceiver must be reset

See Section 46.7.7.7 “FIFO Pointer Error” for details.

- **RXFPTEF: Receive FIFO Pointer Error Flag**

0: No Receive FIFO pointer occurred

1: Receive FIFO pointer error occurred. Receiver must be reset

See Section 46.7.7.7 “FIFO Pointer Error” for details.

46.8.9 SPI Interrupt Enable Register

Name: SPI_IER

Address: 0xF8000014 (0), 0xFC000014 (1)

Access: Write-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Enable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Enable**
- **MODF: Mode Fault Error Interrupt Enable**
- **OVRES: Overrun Error Interrupt Enable**
- **NSSR: NSS Rising Interrupt Enable**
- **TXEMPTY: Transmission Registers Empty Enable**
- **UNDES: Underrun Error Interrupt Enable**
- **CMP: Comparison Interrupt Enable**
- **TXFEF: TXFEF Interrupt Enable**
- **TXFFF: TXFFF Interrupt Enable**
- **TXFTHF: TXFTHF Interrupt Enable**
- **RXFEF: RXFEF Interrupt Enable**
- **RXFFF: RXFFF Interrupt Enable**
- **RXFTHF: RXFTHF Interrupt Enable**
- **TXFPTEF: TXFPTEF Interrupt Enable**
- **RXFPTEF: RXFPTEF Interrupt Enable**

46.8.10 SPI Interrupt Disable Register

Name: SPI_IDR

Address: 0xF8000018 (0), 0xFC000018 (1)

Access: Write-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Disable**
- **TDRE: SPI Transmit Data Register Empty Interrupt Disable**
- **MODF: Mode Fault Error Interrupt Disable**
- **OVRES: Overrun Error Interrupt Disable**
- **NSSR: NSS Rising Interrupt Disable**
- **TXEMPTY: Transmission Registers Empty Disable**
- **UNDES: Underrun Error Interrupt Disable**
- **CMP: Comparison Interrupt Disable**
- **TXFEF: TXFEF Interrupt Disable**
- **TXFFF: TXFFF Interrupt Disable**
- **TXFTHF: TXFTHF Interrupt Disable**
- **RXFEF: RXFEF Interrupt Disable**
- **RXFFF: RXFFF Interrupt Disable**
- **RXFTHF: RXFTHF Interrupt Disable**
- **TXFPTEF: TXFPTEF Interrupt Disable**
- **RXFPTEF: RXFPTEF Interrupt Disable**

46.8.11 SPI Interrupt Mask Register

Name: SPI_IMR

Address: 0xF800001C (0), 0xFC00001C (1)

Access: Read-only

31	30	29	28	27	26	25	24
RXFPTEF	TXFPTEF	RXFTHF	RXFFF	RXFEF	TXFTHF	TXFFF	TXFEF
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CMP	UNDES	TXEMPTY	NSSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	MODF	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RDRF: Receive Data Register Full Interrupt Mask**
- **TDRE: SPI Transmit Data Register Empty Interrupt Mask**
- **MODF: Mode Fault Error Interrupt Mask**
- **OVRES: Overrun Error Interrupt Mask**
- **NSSR: NSS Rising Interrupt Mask**
- **TXEMPTY: Transmission Registers Empty Mask**
- **UNDES: Underrun Error Interrupt Mask**
- **CMP: Comparison Interrupt Mask**
- **TXFEF: TXFEF Interrupt Mask**
- **TXFFF: TXFFF Interrupt Mask**
- **TXFTHF: TXFTHF Interrupt Mask**
- **RXFEF: RXFEF Interrupt Mask**
- **RXFFF: RXFFF Interrupt Mask**
- **RXFTHF: RXFTHF Interrupt Mask**
- **TXFPTEF: TXFPTEF Interrupt Mask**
- **RXFPTEF: RXFPTEF Interrupt Mask**

46.8.12 SPI Chip Select Register

Name: SPI_CSRx [x=0..3]

Address: 0xF8000030 (0), 0xFC000030 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYBCT							
23	22	21	20	19	18	17	16
DLYBS							
15	14	13	12	11	10	9	8
SCBR							
7	6	5	4	3	2	1	0
BITS				CSAAT	CSNAAT	NCPHA	CPOL

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

Note: SPI_CSRx must be written even if the user wants to use the default reset values. The BITS field is not updated with the translated value unless the register is written.

- **CPOL: Clock Polarity**

0: The inactive state value of SPCK is logic level zero.

1: The inactive state value of SPCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (SPCK). It is used with NCPHA to produce the required clock/data relationship between master and slave devices.

- **NCPHA: Clock Phase**

0: Data is changed on the leading edge of SPCK and captured on the following edge of SPCK.

1: Data is captured on the leading edge of SPCK and changed on the following edge of SPCK.

NCPHA determines which edge of SPCK causes data to change and which edge causes data to be captured. NCPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

- **CSNAAT: Chip Select Not Active After Transfer (Ignored if CSAAT = 1)**

0: The Peripheral Chip Select Line does not rise between two transfers if the SPI_TDR is reloaded before the end of the first transfer and if the two transfers occur on the same chip select.

1: The Peripheral Chip Select Line rises systematically after each transfer performed on the same slave. It remains inactive after the end of transfer for a minimal duration of:

If SPI_MR.BRSRCCLK = 0:

$$\frac{DLYBCS}{f_{\text{peripheral clock}}}$$

If SPI_MR.BRSRCCLK = 1:

$$\frac{DLYBCS}{f_{\text{GCLK}}}$$

If field DLYBCS is lower than 6, a minimum of six periods is introduced.

- **CSAAT: Chip Select Active After Transfer**

0: The Peripheral Chip Select Line rises as soon as the last transfer is achieved.

1: The Peripheral Chip Select Line does not rise after the last transfer is achieved. It remains active until a new transfer is requested on a different chip select.

- **BITS: Bits Per Transfer**

(See the note below the SPI_CSR bitmap.)

The BITS field determines the number of data bits transferred. Reserved values should not be used.

Value	Name	Description
0	8_BIT	8 bits for transfer
1	9_BIT	9 bits for transfer
2	10_BIT	10 bits for transfer
3	11_BIT	11 bits for transfer
4	12_BIT	12 bits for transfer
5	13_BIT	13 bits for transfer
6	14_BIT	14 bits for transfer
7	15_BIT	15 bits for transfer
8	16_BIT	16 bits for transfer
9	–	Reserved
10	–	Reserved
11	–	Reserved
12	–	Reserved
13	–	Reserved
14	–	Reserved
15	–	Reserved

- **SCBR: Serial Clock Bit Rate**

In Master mode, the SPI Interface uses a modulus counter to derive the SPCK bit rate from the clock defined by the SPI_MR.BRSRCCLK bit. The bit rate is selected by writing a value from1 to 255 in the SCBR field. The following equations determine the SPCK bit rate:

If SPI_MR.BRSRCCLK = 0: SCBR = $f_{peripheral\ clock}$ / SPCK Bit Rate

If SPI_MR.BRSRCCLK = 1: SCBR = f_{GCLK} / SPCK Bit Rate

Programming the SCBR field to 0 is forbidden. Triggering a transfer while SCBR is at 0 can lead to unpredictable results.

If BRSRCCLK = 1 in SPI_MR, SCBR must be programmed with a value greater than 1.

At reset, SCBR is 0 and the user has to program it at a valid value before performing the first transfer.

Note: If one of the SCBR fields in SPI_CSRx is set to 1, the other SCBR fields in SPI_CSRx must be set to 1 as well, if they are used to process transfers. If they are not used to transfer data, they can be set at any value.

- **DLYBS: Delay Before SPCK**

This field defines the delay from NPCS falling edge (activation) to the first valid SPCK transition.

When DLYBS = 0, the delay is half the SPCK clock period.

Otherwise, the following equations determine the delay:

If SPI_MR.BRSRCCLK = 0: DLYBS = Delay Before SPCK × $f_{peripheral\ clock}$

If SPI_MR.BRSRCCLK = 1: DLYBS = Delay Before SPCK × f_{GCLK}

- **DLYBCT: Delay Between Consecutive Transfers**

This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT = 0, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers.

Otherwise, the following equations determine the delay:

If SPI_MR.BRSRCCLK = 0: DLYBCT = Delay Between Consecutive Transfers × $f_{peripheral\ clock}$ / 32

If SPI_MR.BRSRCCLK = 1: DLYBCT = Delay Between Consecutive Transfers × f_{GCLK} / 32

46.8.13 SPI FIFO Mode Register

Name: SPI_FMR

Address: 0xF8000040 (0), 0xFC000040 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	RXFTHRES					
23	22	21	20	19	18	17	16
–	–	TXFTHRES					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	RXRDYM		–	–	TXRDYM	

- **TXRDYM: Transmitter Data Register Empty Mode**

If FIFOs are enabled, the TDRE flag (in SPI_SR) will behave as follows.

Value	Name	Description
0x0	ONE_DATA	TDRE will be at level ‘1’ when at least one data can be written in the Transmit FIFO.
0x1	TWO_DATA	TDRE will be at level ‘1’ when at least two data can be written in the Transmit FIFO.
0x2	FOUR_DATA	TDRE will be at level ‘1’ when at least four data can be written in the Transmit FIFO.

- **RXRDYM: Receiver Data Register Full Mode**

If FIFOs are enabled, the RDRF flag (in SPI_SR) will behave as follows.

Value	Name	Description
0x0	ONE_DATA	RDRF will be at level ‘1’ when at least one unread data is in the Receive FIFO.
0x1	TWO_DATA	RDRF will be at level ‘1’ when at least two unread data are in the Receive FIFO.
0x2	FOUR_DATA	RDRF will be at level ‘1’ when at least four unread data are in the Receive FIFO.

- **TXFTHRES: Transmit FIFO Threshold**

0–16: Defines the Transmit FIFO threshold value (number of data). TXFTH flag in SPI_SR will be set when Transmit FIFO goes from “above” threshold state to “equal or below” threshold state.

- **RXFTHRES: Receive FIFO Threshold**

0–16: Defines the Receive FIFO threshold value (number of data). RXFTH flag in SPI_SR will be set when Receive FIFO goes from “below” threshold state to “equal or above” threshold state.

46.8.14 SPI FIFO Level Register

Name: SPI_FLR

Address: 0xF8000044 (0), 0xFC000044 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	RXFL					
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	TXFL					

- **TXFL: Transmit FIFO Level**

0: There is no data in the Transmit FIFO

1–16: Indicates the number of DATA in the Transmit FIFO

- **RXFL: Receive FIFO Level**

0: There is no unread data in the Receive FIFO

1–16: Indicates the number of unread DATA in the Receive FIFO

46.8.15 SPI Comparison Register

Name: SPI_CMPR

Address: 0xF8000048 (0), 0xFC000048 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
VAL2							
23	22	21	20	19	18	17	16
VAL2							
15	14	13	12	11	10	9	8
VAL1							
7	6	5	4	3	2	1	0
VAL1							

This register can only be written if the WPEN bit is cleared in the SPI Write Protection Mode Register.

- **VAL1: First Comparison Value for Received Character**

0–65535: The received character must be higher or equal to the value of VAL1 and lower or equal to VAL2 to set CMP flag in SPI_SR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if the condition is met.

- **VAL2: Second Comparison Value for Received Character**

0–65535: The received character must be lower or equal to the value of VAL2 and higher or equal to VAL1 to set CMP flag in SPI_CSR. If asynchronous partial wakeup (SleepWalking) is enabled in PMC_SLPWK_ER, the SPI requests a system wakeup if condition is met.

46.8.16 SPI Write Protection Mode Register

Name: SPI_WPMR

Address: 0xF80000E4 (0), 0xFC0000E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII)

1: Enables the write protection if WPKEY corresponds to 0x535049 (“SPI” in ASCII)

- **WPKEY: Write Protection Key**

Value	Name	Description
0x535049	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

See Section 46.7.8 “Register Write Protection” for the list of registers that can be write-protected.

46.8.17 SPI Write Protection Status Register

Name: SPI_WPSR

Address: 0xF80000E8 (0), 0xFC0000E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of SPI_WPSR.

1: A write protection violation has occurred since the last read of SPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

47. Quad Serial Peripheral Interface (QSPI)

47.1 Description

The Quad Serial Peripheral Interface (QSPI) is a synchronous serial data link that provides communication with external devices in Master mode.

The QSPI can be used in SPI mode to interface to serial peripherals such as ADCs, DACs, LCD controllers, CAN controllers and sensors, or in Serial Memory mode to interface to serial Flash memories.

The QSPI allows the system to execute code directly from a serial Flash memory (XIP) without code shadowing to RAM. The serial Flash memory mapping is seen in the system as other memories such as ROM, SRAM, DRAM, embedded Flash memory, etc.

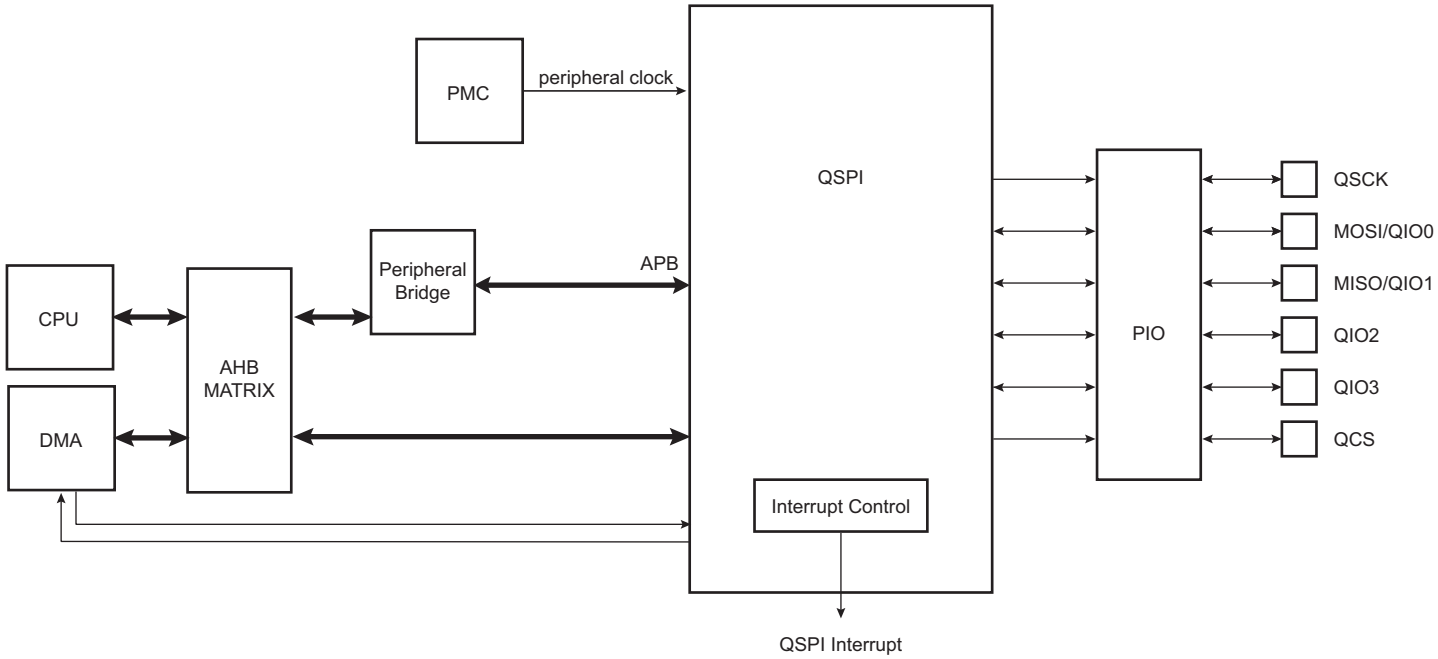
With the support of the Quad SPI protocol, the QSPI allows the system to use high-performance serial Flash memories which are small and inexpensive, in place of larger and more expensive parallel Flash memories.

47.2 Embedded Characteristics

- Master SPI Interface
  - Programmable Clock Phase and Clock Polarity
  - Programmable Transfer Delays Between Consecutive Transfers, Between Clock and Data, Between Deactivation and Activation of Chip Select
- SPI Mode
  - Interface to Serial Peripherals such as ADCs, DACs, LCD Controllers, CAN Controllers and Sensors
  - 8-bit/16-bit/32-bit Programmable Data Length
- Serial Memory Mode
  - Interface to Serial Flash Memories Operating in Single-bit SPI, Dual SPI and Quad SPI
  - Interface to Serial Flash Memories Operating in Single Data Rate or Double Data Rate Modes
  - Supports “Execute In Place” (XIP)— Code Execution by the System Directly from a Serial Flash Memory
  - Flexible Instruction Register for Compatibility with All Serial Flash Memories
  - 32-bit Address Mode (default is 24-bit address) to Support Serial Flash Memories Larger than 128 Mbit
  - Continuous Read Mode
  - Scrambling/Unscrambling “On-The-Fly”
- Connection to DMA Channel Capabilities Optimizes Data Transfers
  - One Channel for the Receiver, One Channel for the Transmitter
- Register Write Protection

47.3 Block Diagram

Figure 47-1. Block Diagram



47.4 Signal Description

Table 47-1. Signal Description

Pin Name	Pin Description	Type
QSCK	Serial Clock	Output
MOSI (QIO0) (1) (2)	Data Output (Data Input Output 0)	Output (Input/Output)
MISO (QIO1) (1) (2)	Data Input (Data Input Output 1)	Input (Input/Output)
QIO2 (3)	Data Input Output 2	Input/Output
QIO3 (3)	Data Input Output 3	Input/Output
QCS	Peripheral Chip Select	Output

Notes:
1. MOSI and MISO are used for single-bit SPI operation.
2. QIO0–QIO1 are used for Dual SPI operation.
3. QIO0–QIO3 are used for Quad SPI operation.

47.5 Product Dependencies

47.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the QSPI pins to their peripheral functions.

Table 47-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
QSPI0	QSPI0_CS	PA1	B
QSPI0	QSPI0_CS	PA15	C
QSPI0	QSPI0_CS	PA23	F
QSPI0	QSPI0_IO0	PA2	B
QSPI0	QSPI0_IO0	PA16	C
QSPI0	QSPI0_IO0	PA24	F
QSPI0	QSPI0_IO1	PA3	B
QSPI0	QSPI0_IO1	PA17	C
QSPI0	QSPI0_IO1	PA25	F
QSPI0	QSPI0_IO2	PA4	B
QSPI0	QSPI0_IO2	PA18	C
QSPI0	QSPI0_IO2	PA26	F
QSPI0	QSPI0_IO3	PA5	B
QSPI0	QSPI0_IO3	PA19	C
QSPI0	QSPI0_IO3	PA27	F
QSPI0	QSPI0_SCK	PA0	B
QSPI0	QSPI0_SCK	PA14	C
QSPI0	QSPI0_SCK	PA22	F
QSPI1	QSPI1_CS	PA11	B
QSPI1	QSPI1_CS	PB6	D
QSPI1	QSPI1_CS	PB15	E
QSPI1	QSPI1_IO0	PA7	B
QSPI1	QSPI1_IO0	PB7	D
QSPI1	QSPI1_IO0	PB16	E
QSPI1	QSPI1_IO1	PA8	B
QSPI1	QSPI1_IO1	PB8	D
QSPI1	QSPI1_IO1	PB17	E
QSPI1	QSPI1_IO2	PA9	B
QSPI1	QSPI1_IO2	PB9	D
QSPI1	QSPI1_IO2	PB18	E
QSPI1	QSPI1_IO3	PA10	B
QSPI1	QSPI1_IO3	PB10	D

Table 47-2. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
QSPI1	QSPI1_IO3	PB19	E
QSPI1	QSPI1_SCK	PA6	B
QSPI1	QSPI1_SCK	PB5	D
QSPI1	QSPI1_SCK	PB14	E

47.5.2 Power Management

The QSPI may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the QSPI clock.

47.5.3 Interrupt Sources

The QSPI has an interrupt line connected to the Interrupt Controller. Handling the QSPI interrupt requires programming the interrupt controller before configuring the QSPI.

Table 47-3. Peripheral IDs

Instance	ID
QSPI0	52
QSPI1	53

47.5.4 Direct Memory Access Controller (DMA)

The QSPI can be used in conjunction with the Direct Memory Access Controller (DMA) in order to reduce processor overhead. For a full description of the DMA, refer to the section “DMA Controller (XDMAC)”.

47.6 Functional Description

47.6.1 Serial Clock Baud Rate

The QSPI baud rate clock is generated by dividing the peripheral clock by a value between 1 and 256.

47.6.2 Serial Clock Phase and Polarity

Four combinations of polarity and phase are available for data transfers. The clock polarity is programmed with the CPOL bit in the QSPI Serial Clock register (QSPI_SCR). The CPHA bit in the QSPI_SCR programs the clock phase. These two parameters determine the edges of the clock signal on which data is driven and sampled. Each of the two parameters has two possible states, resulting in four possible combinations that are incompatible with one another. Thus, the interfaced slave must use the same parameter values to communicate.

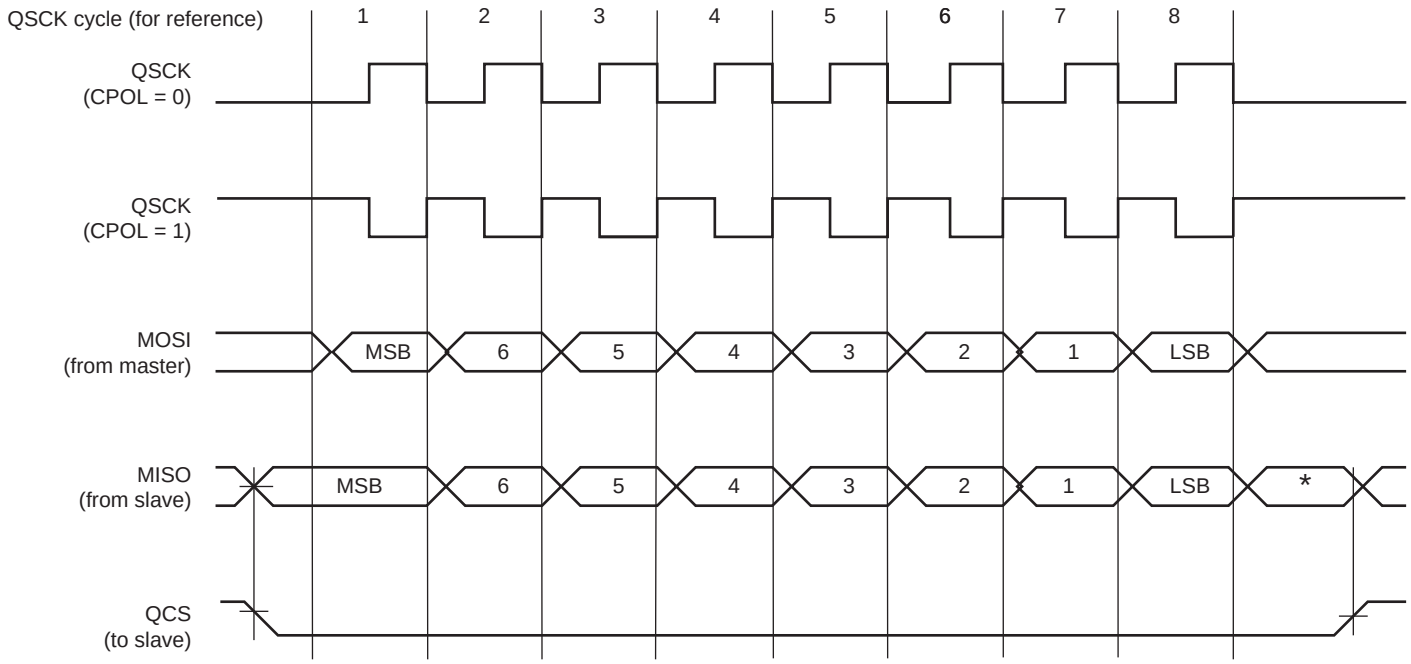
Table 47-4 shows the four modes and corresponding parameter settings.

Table 47-4. QSPI Bus Clock Modes

QSPI Clock Mode	QSPI_SCR.CPOL	QSPI_SCR.CPHA	Shift QSCK Edge	Capture QSCK Edge	QSCK Inactive Level
0	0	0	Falling	Rising	Low
1	0	1	Rising	Falling	Low
2	1	0	Rising	Falling	High
3	1	1	Falling	Rising	High

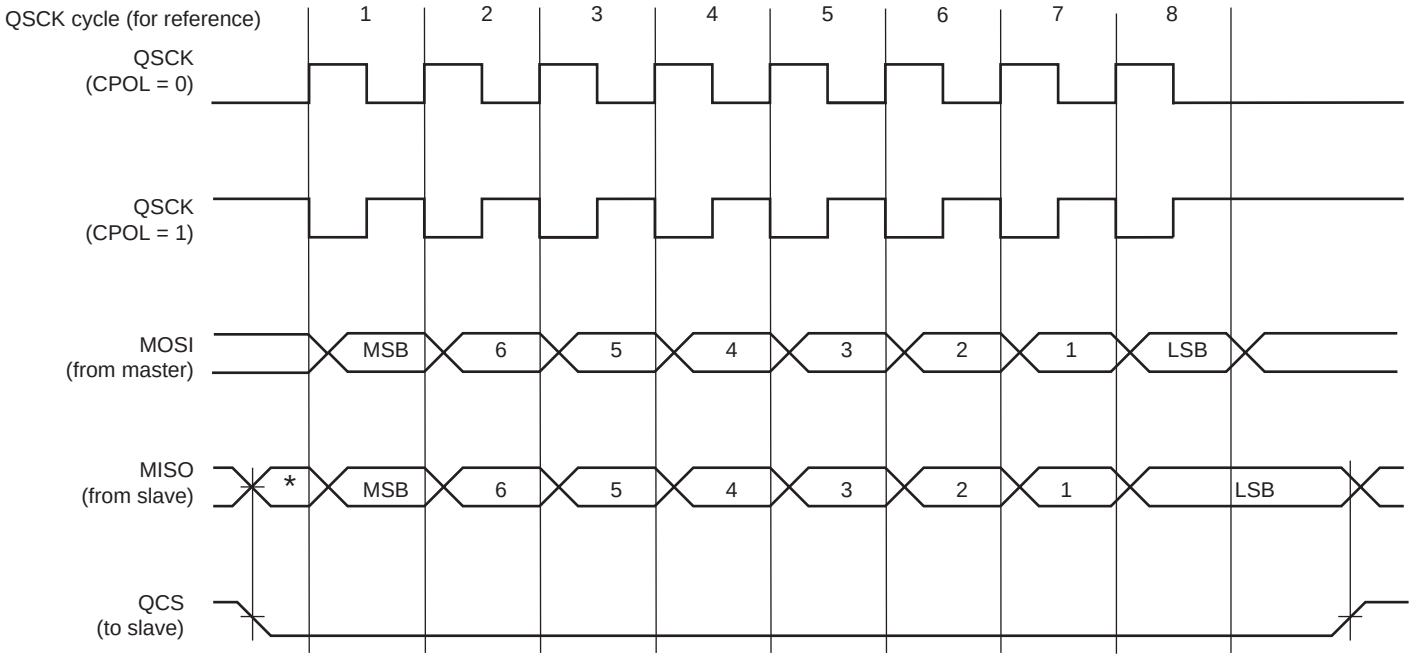
Figure 47-2 and Figure 47-3 show examples of data transfers.

Figure 47-2. QSPI Transfer Format (QSPI_SCR.CPHA = 0, 8 bits per transfer)



* Not defined, but normally MSB of previous character received.

Figure 47-3. QSPI Transfer Format (QSPI_SCR.CPHA = 1, 8 bits per transfer)



* Not defined but normally LSB of previous character transmitted.

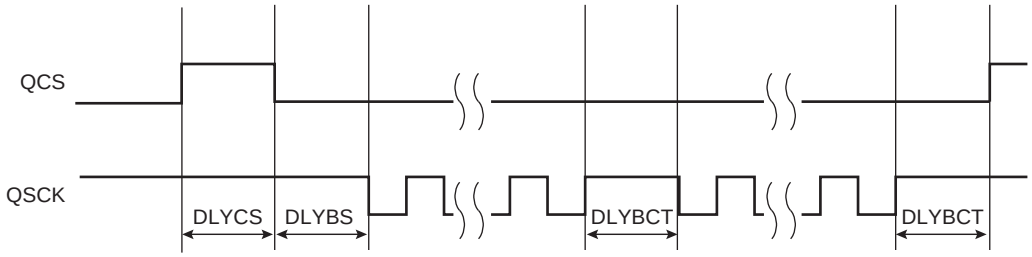
47.6.3 Transfer Delays

Figure 47-4 shows several consecutive transfers while the chip select is active. Three delays can be programmed to modify the transfer waveforms:

- The delay between the deactivation and the activation of QCS, programmed by writing QSPI_MR.DLYCS. Allows to adjust the minimum time of QCS at high level.
- The delay before QSCK, programmed by writing QSPI_SR.DLYBS. Allows the start of QSCK to be delayed after the chip select has been asserted.
- The delay between consecutive transfers, programmed by writing QSPI_MR.DLYBCT. Allows insertion of a delay between two consecutive transfers. In Serial Memory mode, this delay is not programmable and DLYBCT is ignored. In this mode, DLYBCT must be written to ‘0’.

These delays allow the QSPI to be adapted to the interfaced peripherals and their speed and bus release time.

Figure 47-4. Programmable Delays



47.6.4 QSPI SPI Mode

In SPI mode, the QSPI acts as a regular SPI Master.

To activate this mode, QSPI_MR.SMM must be written to ‘0’ in QSPI_MR.

47.6.4.1 SPI Mode Operations

The QSPI in standard SPI mode operates on the clock generated by the internal programmable baud rate generator. It fully controls the data transfers to and from the slave connected to the SPI bus. The QSPI drives the chip select line to the slave (QCS) and the serial clock signal (QSCK).

The QSPI features two holding registers, the Transmit Data register (QSPI_TDR) and the Receive Data register (QSPI_RDR), and a single internal shift register. The holding registers maintain the data flow at a constant rate.

After enabling the QSPI, a data transfer begins when the processor writes to the QSPI_TDR. The written data is immediately transferred to the internal shift register and transfer on the SPI bus starts. While the data in the internal shift register is shifted on the MOSI line, the MISO line is sampled and shifted to the internal shift register. Receiving data cannot occur without transmitting data. If receiving mode is not needed, for example when communicating with a slave receiver only (such as an LCD), the receive status flags in the Status register (QSPI_SR) can be discarded.

If new data is written in QSPI_TDR during the transfer, it is retained there until the current transfer is completed. Then, the received data is transferred from the internal shift register to the QSPI_RDR, the data in QSPI_TDR is loaded in the internal shift register and a new transfer starts.

The transfer of a data written in QSPI_TDR in the internal shift register is indicated by the Transmit Data Register Empty (TDRE) bit in the QSPI_SR. When new data is written in QSPI_TDR, this bit is cleared. QSPI_SR.TDRE is used to trigger the Transmit DMA channel.

The end of transfer is indicated by the TXEMPTY flag in the QSPI_SR. If a transfer delay (DLYBCT) is greater than 0 for the last transfer, QSPI_SR.TXEMPTY is set after the completion of this delay. The peripheral clock can be switched off at this time.

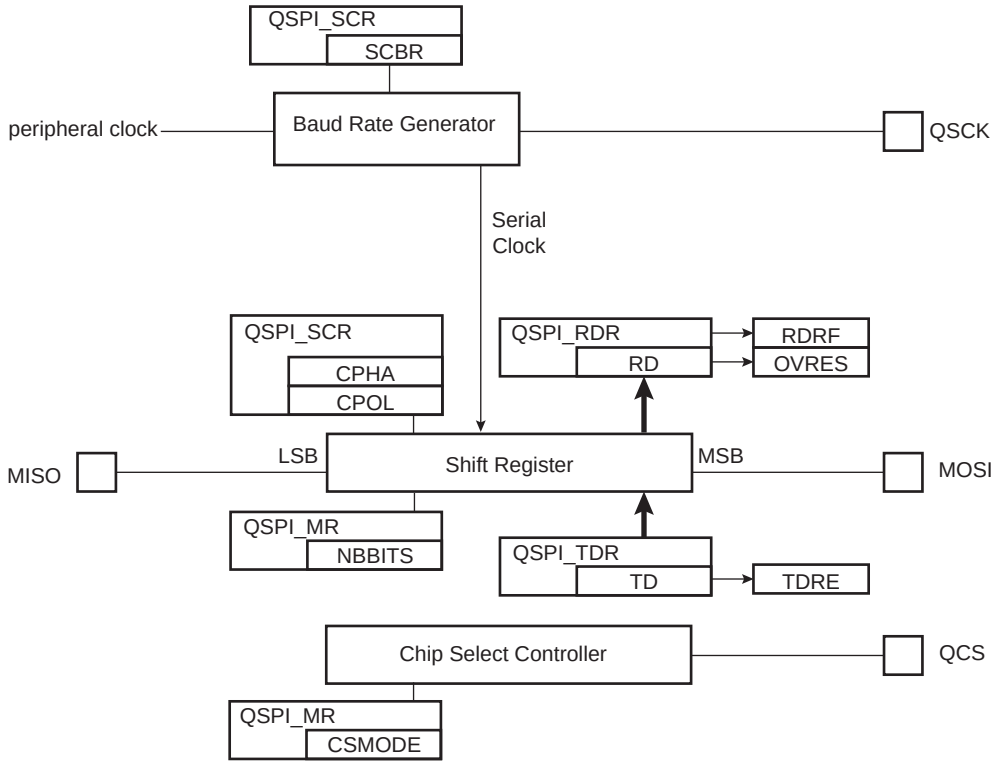
The transfer of received data from the internal shift register in QSPI_RDR is indicated by the Receive Data Register Full (RDRF) bit in the QSPI_SR. When the received data is read, QSPI_SR.RDRF bit is cleared.

If the QSPI_RDR has not been read before new data is received, the Overrun Error Status (OVRES) bit in QSPI_SR is set. As long as this flag is set, data is loaded in QSPI_RDR. The user must read the QSPI_SR to clear the OVRES bit.

Figure 47-5 shows a block diagram of the SPI when operating in Master mode. Figure 47-6 shows a flow chart describing how transfers are handled.

47.6.4.2 SPI Mode Block Diagram

Figure 47-5. SPI Mode Block Diagram



47.6.4.3 SPI Mode Flow Diagram

Figure 47-6. SPI Mode Flow Diagram

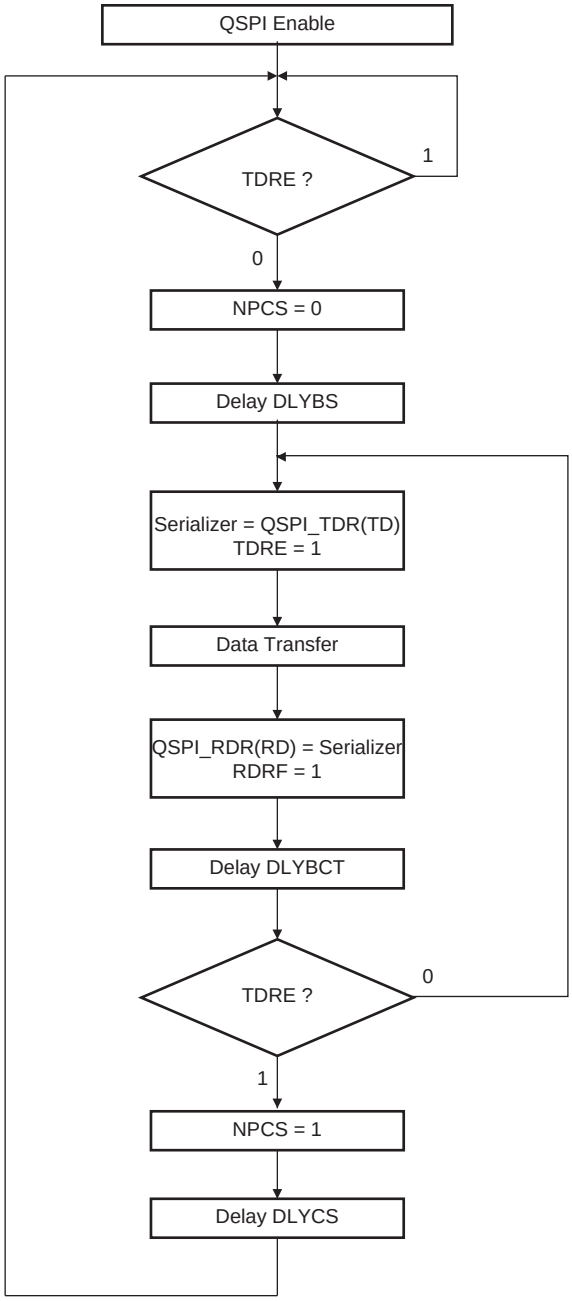
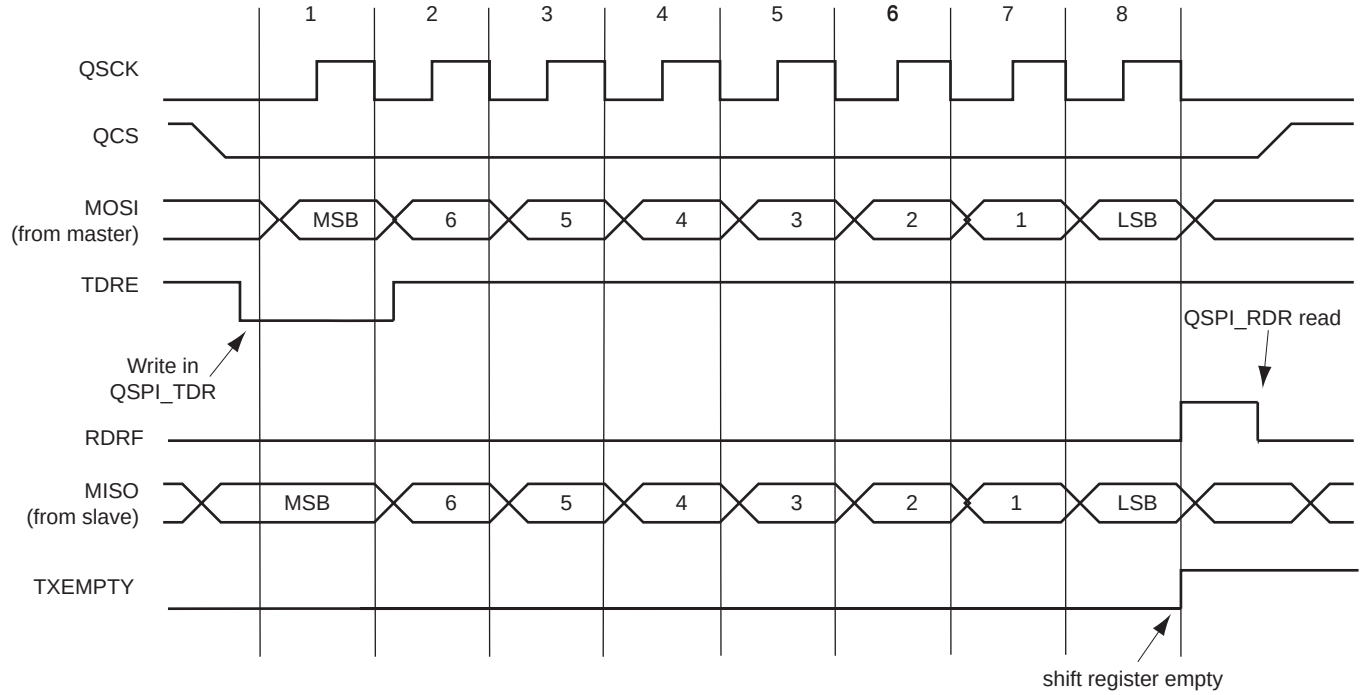


Figure 47-7 shows Transmit Data Register Empty (TDRE), Receive Data Register Full (RDRF) and Transmission Register Empty (TXEMPTY) status flags behavior within the QSPI_SR during an 8-bit data transfer in Fixed mode, without DMA.

Figure 47-7. Status Register Flags Behavior



47.6.4.4 Peripheral Deselection without DMA

During a transfer of more than one data on a Chip Select without the DMA, the QSPI_TDR is loaded by the processor and the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shift register. When this flag is detected high, the QSPI_TDR can be reloaded. If this reload by the processor occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not de-asserted between the two transfers. Depending on the application software handling the QSPI_SR flags (by interrupt or polling method) or servicing other interrupts or other tasks, the processor may not reload the QSPI_TDR in time to keep the chip select active (low). A null Delay Between Consecutive Transfer (DLYBCT) value in the QSPI_MR gives even less time for the processor to reload the QSPI_TDR. With some SPI slave peripherals, requiring the chip select line to remain active (low) during a full set of transfers may lead to communication errors.

To facilitate interfacing with such devices, QSPI_MR.CSMODE may be configured to ‘1’. This allows the chip select lines to remain in their current state (low = active) until the end of transfer is indicated by the Last Transfer (LASTXFER) bit in the Control register (QSPI_CR). Even if the QSPI_TDR is not reloaded, the chip select remains active. To have the chip select line rise at the end of the last data transfer, QSPI_CR.LASTXFER must be written to ‘1’ at the same time or after writing the last data to transmit into the QSPI_TDR.

47.6.4.5 Peripheral Deselection with DMA

When the DMA Controller is used, the Chip Select line remains low during the transfer since the TDRE flag is managed by the DMA itself. Reloading the QSPI_TDR by the DMA is done as soon as the TDRE flag is set. In this case, writing QSPI_MR.CSMODE to ‘1’ may not be needed. However, when other DMA channels connected to other peripherals are also in use, the QSPI DMA could be delayed by another DMA with a higher priority on the bus. Having DMA buffers in slower memories like Flash memory or SDRAM compared to fast internal SRAM, may lengthen the reload time of the QSPI_TDR by the DMA as well. This means that the QSPI_TDR might not be reloaded in time to keep the chip select line low. In this case, the chip select line may toggle between data transfer and according to some SPI Slave devices, the communication might get lost. It may be necessary to configure QSPI_MR.CSMODE to ‘1’.

When QSPI_MR.CSMODE is configured to ‘0’, the QCS does not rise in all cases between two transfers on the same peripheral. During a transfer on a Chip Select, the flag TDRE rises as soon as the content of the QSPI_TDR is transferred into the internal shifter. When this flag is detected, the QSPI_TDR can be reloaded. If this reload occurs before the end of the current transfer and if the next transfer is performed on the same chip select as the current transfer, the Chip Select is not de-asserted between the two transfers. This might lead to difficulties for interfacing with some serial peripherals requiring the chip select to be de-asserted after each transfer. To facilitate interfacing with such devices, the QSPI_MR may be configured with QSPI_MR.CSMODE at ‘2’.

47.6.5 QSPI Serial Memory Mode

In Serial Memory mode, the QSPI acts as a serial Flash memory controller. The QSPI can be used to read data from the serial Flash memory allowing the CPU to execute code from it (XIP execute in place). The QSPI can also be used to control the serial Flash memory (Program, Erase, Lock, etc.) by sending specific commands. In this mode, the QSPI is compatible with single-bit SPI, Dual SPI and Quad SPI protocols.

To activate this mode, QSPI_MR.SMM must be written to ‘1’.

In Serial Memory mode, data cannot be transferred by the QSPI_TDR and the QSPI_RDR, but by writing or reading the QSPI memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).

47.6.5.1 Instruction Frame

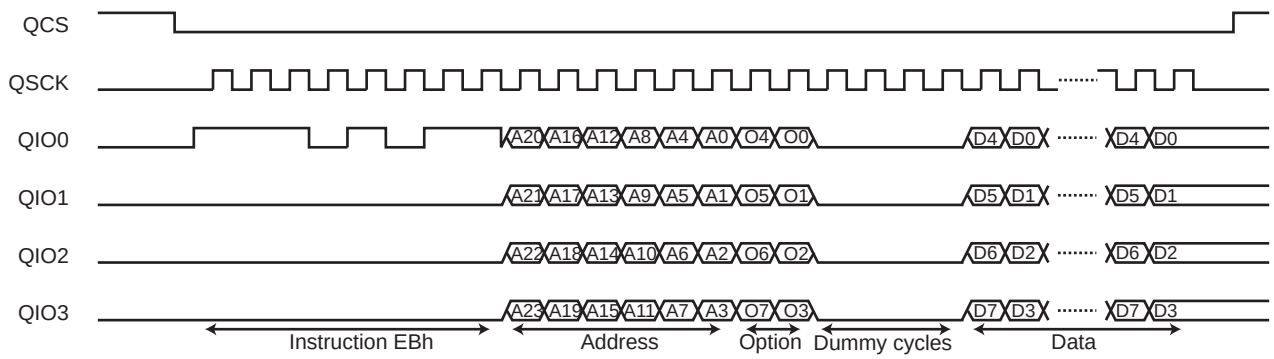
In order to control serial Flash memories, the QSPI is able to send instructions via the SPI bus (ex: READ, PROGRAM, ERASE, LOCK, etc.). Because the instruction set implemented in serial Flash memories is memory vendor dependant, the QSPI includes a complete Instruction Frame register (QSPI_IFR), which makes it very flexible and compatible with all serial Flash memories.

An instruction frame includes:

- An instruction code (size: 8 bits). The instruction is optional in some cases (see Section 47.6.5.4).
- An address (size: 24 bits or 32 bits). The address is optional but is required by instructions such as READ, PROGRAM, ERASE, LOCK. By default the address is 24 bits long, but it can be 32 bits long to support serial Flash memories larger than 128 Mbit (16 Mbyte).
- An option code (size: 1/2/4/8 bits). The option code is not required, but it is useful to activate the XIP mode or the Continuous Read mode (see Section 47.6.5.4) for READ instructions, in some serial Flash memory devices. These modes improve the data read latency.
- Dummy cycles. Dummy cycles are optional but required by some READ instructions.
- Data bytes are optional. Data bytes are present for data transfer instructions such as READ or PROGRAM.

The instruction code, the address/option and the data can be sent with Single-bit SPI, Dual SPI or Quad SPI protocols.

Figure 47-8. Instruction Frame



47.6.5.2 Instruction Frame Transmission

To send an instruction frame, the user must first configure the address to send by writing the field ADDR in the Instruction Address register (QSPI_IAR). This step is required if the instruction frame includes an address and no data. When data is present, the address of the instruction is defined by the address of the data accesses in the QSPI memory space, not by QSPI_IAR.

If the instruction frame includes the instruction code and/or the option code, the user must configure the instruction code and/or the option code to send by writing the fields INST and OPT in the Instruction Code register (QSPI_ICR).

Then, the user must write QSPI_IFR to configure the instruction frame depending on which instruction must be sent. If the instruction frame does not include data, writing in this register triggers the send of the instruction frame in the QSPI. If the instruction frame includes data, the send of the instruction frame is triggered by the first data access in the QSPI memory space.

The instruction frame is configured by the following bits and fields of QSPI_IFR:

- WIDTH field—used to configure which data lanes are used to send the instruction code, the address, the option code and to transfer the data. It is possible to use two unidirectional data lanes (MISO-MOSI Single-bit SPI), two bidirectional data lanes (QIO0-QIO1 Dual SPI) or four bidirectional data lanes (QIO0–QIO3 Quad SPI).
- INSTEN bit—used to enable the send of an instruction code.
- ADDREN bit—used to enable the send of an address after the instruction code.
- OPTEN bit—used to enable the send of an option code after the address.
- DATAEN bit—used to enable the transfer of data (READ or PROGRAM instruction).
- OPTL field—used to configure the option code length. The value written in OPTL must be consistent with the value written in the field WIDTH. For example: OPTL = 0 (1-bit option code) is not consistent with WIDTH = 6 (option code sent with QuadSPI protocol, thus the minimum length of the option code is 4 bits).
- ADDRL bit—used to configure the address length.
- TFRTYP field—used to define which type of data transfer must be performed.
- DDREN bit—used to configure the double data rate mode, instruction code is still transmitted in single data rate mode.
- NBDUM field—used to configure the number of dummy cycles when reading data from the serial Flash memory. Between the address/option and the data, with some instructions, dummy cycles are inserted by the serial Flash memory.

Refer to Section 47.7.12 “QSPI Instruction Frame Register”.

If data transfer is enabled, the user can access the serial memory by reading or writing the QSPI memory space:

- To read in the serial memory, but not a memory data, for example a JEDEC-ID or the QSPI_SR, QSPI_IFR.TFRTYP must be written to ‘0’.
- To read in the serial memory, and particularly a memory data, TFRTYP must be written to ‘1’.
- To write in the serial memory, but not a memory data, for example writing the configuration or the QSPI_SR, TFRTYP must be written to ‘2’.
- If the user wants to write in the serial memory in particular to program a memory data, TFRTYP must be written to ‘3’ .

If QSPI_IFR.TFRTYP has a value other than ‘1’ and QSPI_MR.SMRM = 0, the address sent in the instruction frame is the address of the first system bus accesses. The addresses of the next accesses are not used by the QSPI. At each system bus access, an SPI transfer is performed with the same size. For example, a halfword system bus access leads to a 16-bit SPI transfer, and a byte system bus access leads to an 8-bit SPI transfer.
If SMRM = 1 and TFRTYP= (0 or 2), accesses are made via the QSPI registers and the address sent in the instruction frame is the address defined in QSPI_IAR.

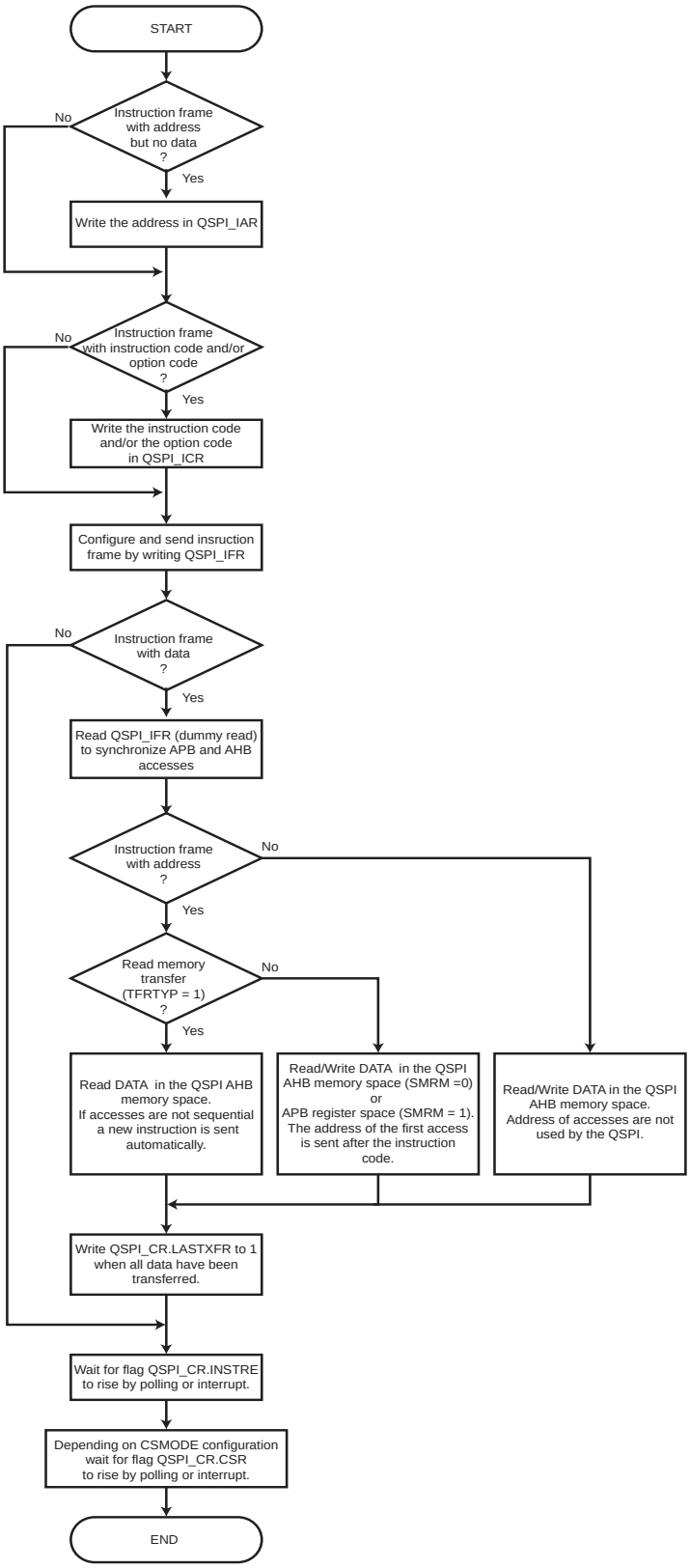
Each time QSPI_IFR is written (in case of read access), or each time QSPI_TDR is written (in case of write transfer), an SPI transfer is performed with a byte size. Another byte is read each time QSPI_RDR is read (flag RDRF shows when a data can be read in QSPI_RDR) or written each time QSPI_TDR is written (flag TDRE shows when a new data can be written). The SPI transfer ends by writing QSPI_CR.LASTXFER.

If TFRTYP = 1, the address of the first instruction frame is the one of the first read access in the QSPI memory space. Each time the read accesses become non-sequential (addresses are not consecutive), a new instruction frame is sent with the last system bus access address. In this way, the system can read data at a random location in the serial memory. The size of the SPI transfers may differ from the size of the system bus read accesses.

When data transfer is not enabled, the end of the instruction frame is indicated when QSPI_SR.INSTRE rises. (The QSPI_SR.CSR flag indicates when chip select rises. A delay between these flags may exist in case of high clock division or a high DLYBCT value). When data transfer is enabled, the user must indicate when the data transfer is completed in the QSPI memory space by setting QSPI_CR.LASTXFR. The end of the instruction frame is indicated when QSPI_SR.INSTRE rises.

Figure 47-9 illustrates instruction transmission management.

Figure 47-9. Instruction Transmission Flow Diagram



47.6.5.3 Read Memory Transfer

The user can access the data of the serial memory by sending an instruction with QSPI_IFR.DATAEN = 1 and QSPI_IFR.TFRTYP = 1.

In this mode, the QSPI is able to read data at random address into the serial Flash memory, allowing the CPU to execute code directly from it (XIP execute-in-place).

In order to fetch data, the user must first configure the instruction frame by writing the QSPI_IFR. Then data can be read at any address in the QSPI address space mapping. The address of the system bus read accesses match the address of the data inside the serial Flash memory.

When Fetch mode is enabled, several instruction frames can be sent before writing QSPI_CR.LASTXFR. Each time the system bus read accesses become non- sequential (addresses are not consecutive), a new instruction frame is sent with the corresponding address.

47.6.5.4 Continuous Read Mode

The QSPI is compatible with the Continuous Read mode which is implemented in some serial Flash memories.

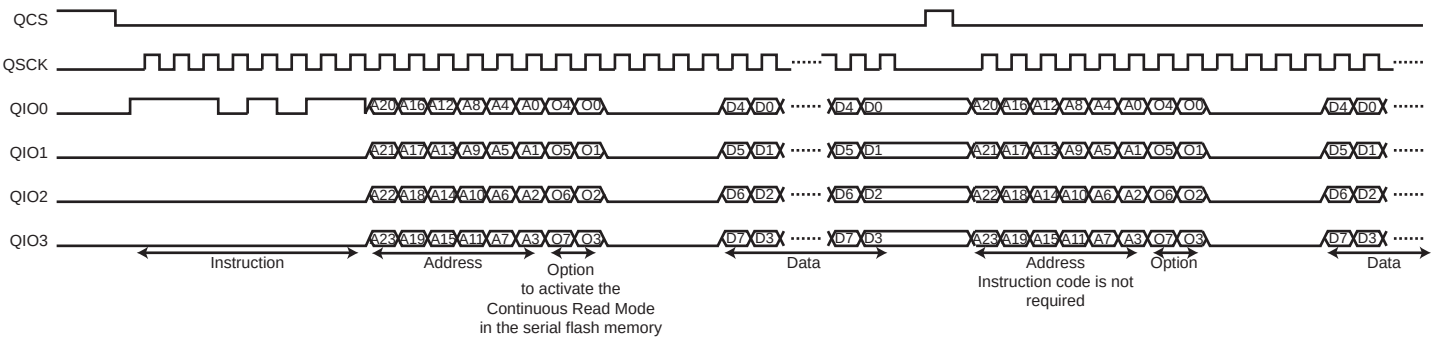
In Continuous Read mode, the instruction overhead is reduced by excluding the instruction code from the instruction frame. When the Continuous Read mode is activated in a serial Flash memory by a specific option code, the instruction code is stored in the memory. For the next instruction frames, the instruction code is not required as the memory uses the stored one.

In the QSPI, Continuous Read mode is used when reading data from the memory (QSPI_IFR.TFRTYP = 1). The addresses of the system bus read accesses are often non-sequential and this leads to many instruction frames that have the same instruction code. By disabling the send of the instruction code, the Continuous Read mode reduces the access time of the data.

To be functional, this mode must be enabled in both the QSPI and the serial Flash memory. The Continuous Read mode is enabled in the QSPI by writing CRM to ‘1’ in the QSPI_IFR (TFRTYP must equal 1). The Continuous Read mode is enabled in the serial Flash memory by sending a specific option code.

CAUTION: If the Continuous Read mode is not supported by the serial Flash memory or disabled, CRM bit must not be written to ‘1’, otherwise data read out the serial Flash memory is unpredictable.

Figure 47-10. Continuous Read Mode



47.6.5.5 Instruction Frame Transmission Examples

All waveforms in the following examples describe SPI transfers in SPI Clock mode 0 (QSPI_SCR.CPOL = 0 and QSPI_SCR.CPHA = 0; see Section 47.6.2 “Serial Clock Phase and Polarity”).

All system bus accesses described below refer to the system bus address phase. System bus wait cycles and system bus data phases are not shown.

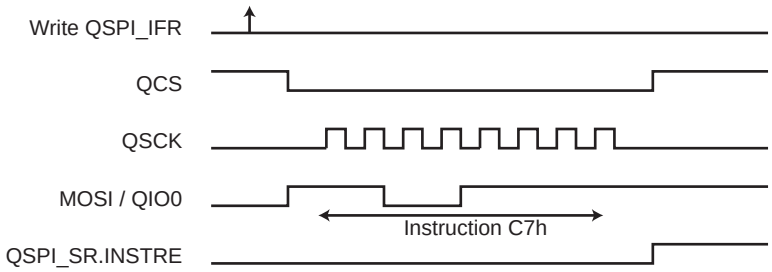
Example 1:

Instruction in Single-bit SPI, without address, without option, without data.

Command: CHIP ERASE (C7h).

- Write 0x0000_00C7 in QSPI_ICR.
- Write 0x0000_0010 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-11. Instruction Transmission Waveform 1



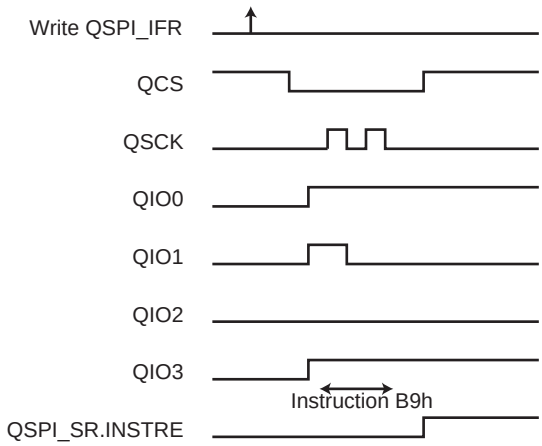
Example 2:

Instruction in Quad SPI, without address, without option, without data.

Command: POWER DOWN (B9h)

- Write 0x0000_00B9 in QSPI_ICR.
- Write 0x0000_0016 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-12. Instruction Transmission Waveform 2



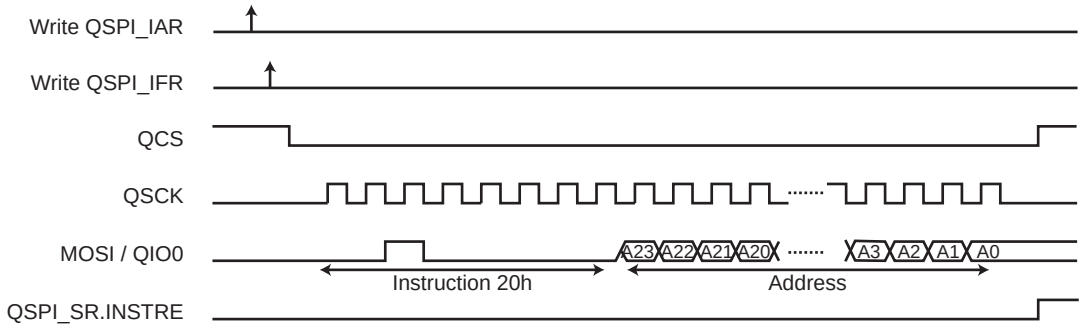
Example 3:

Instruction in Single-bit SPI, with address in Single-bit SPI, without option, without data.

Command: BLOCK ERASE (20h)

- Write the address (of the block to erase) in QSPI_AR.
- Write 0x0000_0020 in QSPI_ICR.
- Write 0x0000_0030 in QSPI_IFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-13. Instruction Transmission Waveform 3



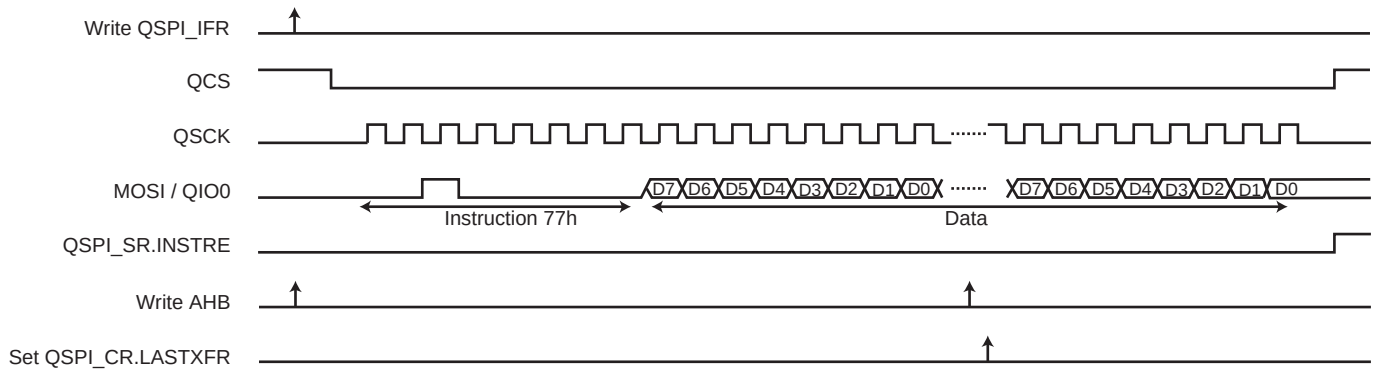
Example 4:

Instruction in Single-bit SPI, without address, without option, with data write in Single-bit SPI.

Command: SET BURST (77h)

- Write 0x0000_0077 in QSPI_ICR.
- Write 0x0000_2090 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the system bus memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).
  The address of system bus write accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-14. Instruction Transmission Waveform 4



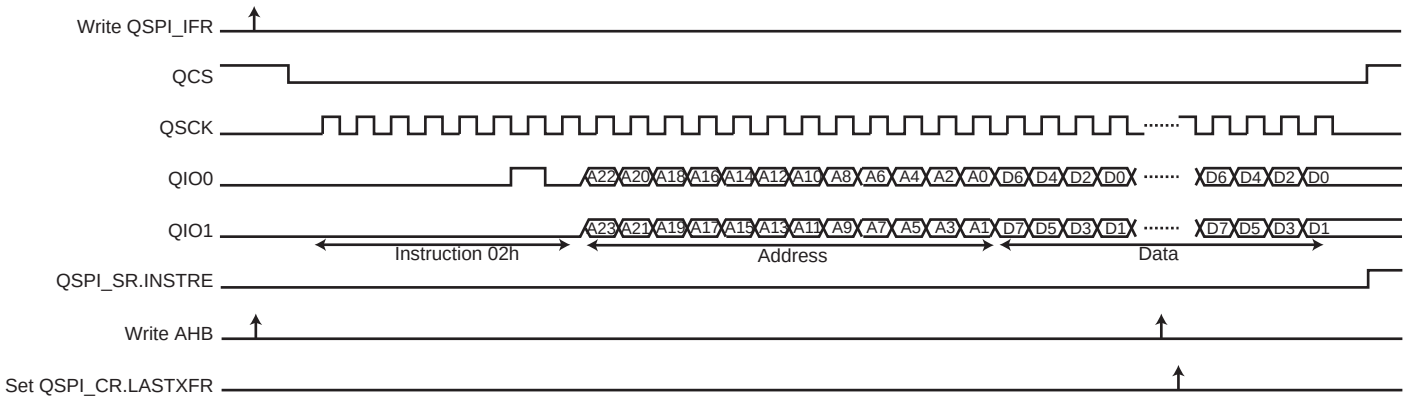
Example 5:

Instruction in Single-bit SPI, with address in Dual SPI, without option, with data write in Dual SPI.

Command: BYTE/PAGE PROGRAM (02h)

- Write 0x0000_0002 in QSPI_ICR.
- Write 0x0000_30B3 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Write data in the QSPI system bus memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).
  The address of the first system bus write access is sent in the instruction frame.
  The address of the next system bus write accesses is not used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-15. Instruction Transmission Waveform 5



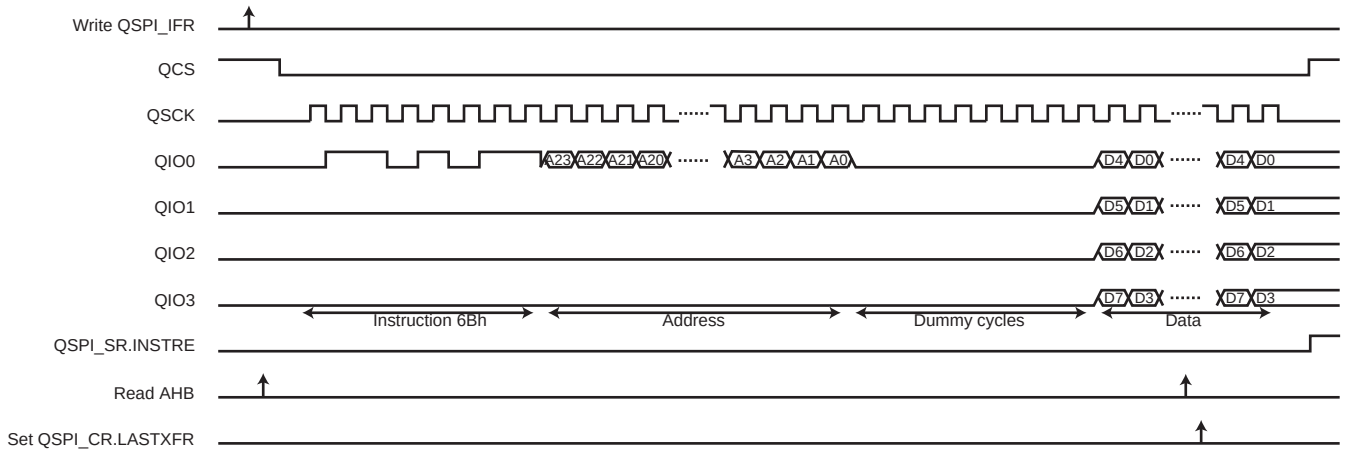
Example 6:

Instruction in Single-bit SPI, with address in Single-bit SPI, without option, with data read in Quad SPI, with eight dummy cycles.

Command: QUAD_OUTPUT READ ARRAY (6Bh)

- Write 0x0000_006B in QSPI_ICR.
- Write 0x0008_10B2 in QSPI_IFR.
- Read QSPI_IR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).
  The address of the first system bus read access is sent in the instruction frame.
  The address of the next system bus read accesses is not used.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-16. Instruction Transmission Waveform 6



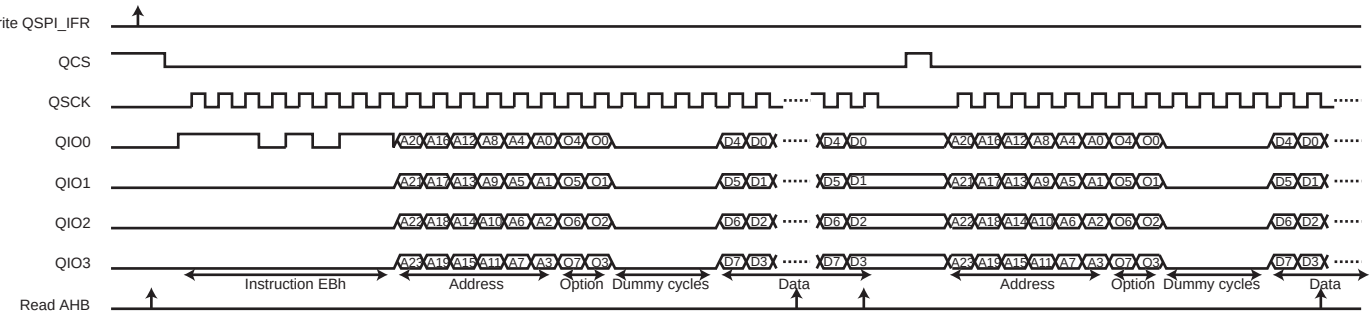
Example 7:

Instruction in Single-bit SPI, with address and option in Quad SPI, with data read in Quad SPI, with four dummy cycles, with fetch and continuous read.

Command: FAST READ QUAD I/O (EBh) - 8-BIT OPTION (0x30h)

- Write 0x0030_00EB in QSPI_ICR.
- Write 0x0004_33F4 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-17. Instruction Transmission Waveform 7



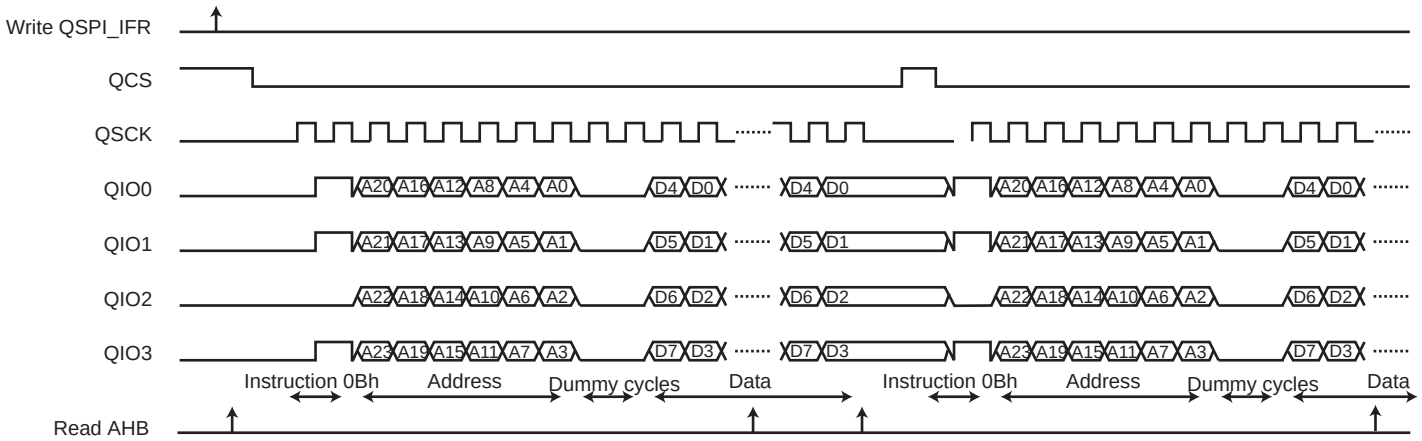
Example 8:

Instruction in Quad SPI, with address in Quad SPI, without option, with data read in Quad SPI, with two dummy cycles, with fetch.

Command: HIGH-SPEED READ (0Bh)

- Write 0x0000_000B in QSPI_ICR.
- Write 0x0002_20B6 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).
  Fetch is enabled, the address of the system bus read accesses is always used.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-18. Instruction Transmission Waveform 8



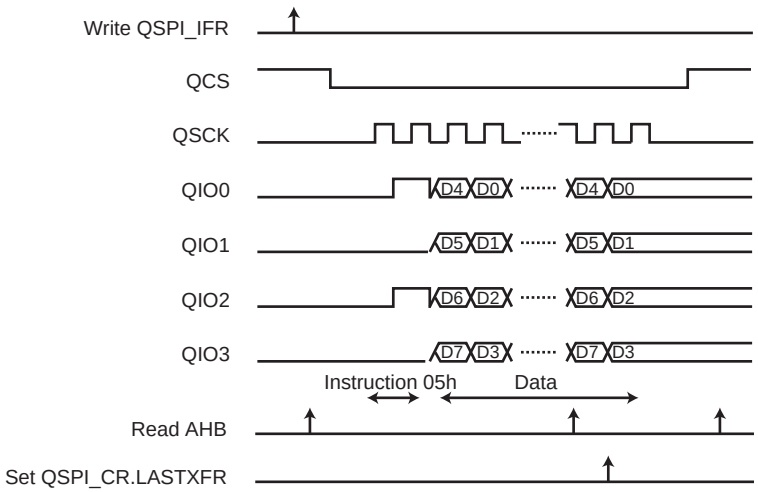
Example 9:

Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch.

Command: HIGH-SPEED READ (05h)

- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0000_0096 in QSPI_IFR.
- Read QSPI_IFR (dummy read) to synchronize system bus accesses.
- Read data in the QSPI system bus memory space (0x9000_00000-0x9800_00000/0XD000_0000--0XD800_0000).
  Fetch is disabled.
- Write a '1' to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-19. Instruction Transmission Waveform 9



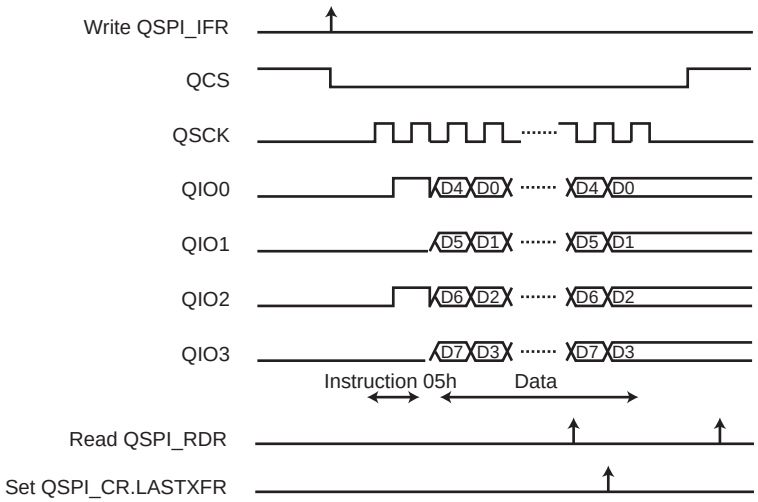
Example 10:

Instruction in Quad SPI, without address, without option, with data read in Quad SPI, without dummy cycles, without fetch, read launched through APB interface.

Command: HIGH-SPEED READ (05h)

- Set SMRM to ‘1’ in QSPI_MR
- Write 0x0000_0005 in QSPI_ICR.
- Write 0x0100_0096 in QSPI_IFR (will start the transfer).
- Wait flag RDRF and Read data in the QSPI_RDR register
  Fetch is disabled.
- Write a ‘1’ to QSPI_CR.LASTXFR.
- Wait for QSPI_SR.INSTRE to rise.

Figure 47-20. Instruction Transmission Waveform 10



47.6.6 Scrambling/Unscrambling Function

The scrambling/unscrambling function cannot be performed on devices other than memories. Data is scrambled when written to memory and unscrambled when data is read.

The external data lines can be scrambled in order to prevent intellectual property data located in off-chip memories from being easily recovered by analyzing data at the package pin level of either the microcontroller or the QSPI slave device (e.g., memory).

The scrambling/unscrambling function can be enabled by writing a '1' to the SCREN bit in the QSPI Scrambling Mode Register (QSPI_SMR).

The scrambling and unscrambling are performed on-the-fly without impacting the throughput.

The scrambling method depends on the user-configurable user scrambling key (field USRK) in the QSPI Scrambling Key Register (QSPI_SKR). QSPI_SKR is only accessible in Write mode.

If QSPI_SMR.RVDIS is written to '0', the scrambling/unscrambling algorithm includes the user scrambling key plus a random value depending on device processing characteristics. Data scrambled by a given microcontroller cannot be unscrambled by another.

If QSPI_SMR.RVDIS is written to '1', the scrambling/unscrambling algorithm includes only the user scrambling key. No random value is part of the key.

The user scrambling key or the seed for key generation must be securely stored in a reliable non-volatile memory in order to recover data from the off-chip memory. Any data scrambled with a given key cannot be recovered if the key is lost.

47.6.7 Register Write Protection

To prevent any single software error from corrupting QSPI behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the QSPI Write Protection Mode Register (QSPI_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the QSPI Write Protection Status Register (QSPI_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading the QSPI_WPSR.

The following registers can be write-protected when WPEN is set in QSPI_WPMR:

- QSPI Mode Register
- QSPI Serial Clock Register
- QSPI Scrambling Mode Register
- QSPI Scrambling Key Register

47.7 Quad Serial Peripheral Interface (QSPI) User Interface

Table 47-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	QSPI_CR	Write-only	–
0x04	Mode Register	QSPI_MR	Read/Write	0x0
0x08	Receive Data Register	QSPI_RDR	Read-only	0x0
0x0C	Transmit Data Register	QSPI_TDR	Write-only	–
0x10	Status Register	QSPI_SR	Read-only	0x0
0x14	Interrupt Enable Register	QSPI_IER	Write-only	–
0x18	Interrupt Disable Register	QSPI_IDR	Write-only	–
0x1C	Interrupt Mask Register	QSPI_IMR	Read-only	0x0
0x20	Serial Clock Register	QSPI_SCR	Read/Write	0x0
0x30	Instruction Address Register	QSPI_IAR	Read/Write	0x0
0x34	Instruction Code Register	QSPI_ICR	Read/Write	0x0
0x38	Instruction Frame Register	QSPI_IFR	Read/Write	0x0
0x3C	Reserved	–	–	–
0x40	Scrambling Mode Register	QSPI_SMR	Read/Write	0x0
0x44	Scrambling Key Register	QSPI_SKR	Write-only	–
0x48–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	QSPI_WPMR	Read/Write	0x0
0xE8	Write Protection Status Register	QSPI_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–

47.7.1 QSPI Control Register

Name: QSPI_CR

Address: 0xF0020000 (0), 0xF0024000 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	LASTXFER
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
SWRST	–	–	–	–	–	QSPIDIS	QSPIEN

- **QSPIEN: QSPI Enable**

0: No effect.

1: Enables the QSPI to transfer and receive data.

- **QSPIDIS: QSPI Disable**

0: No effect.

1: Disables the QSPI.

As soon as QSPIDIS is set, the QSPI finishes its transfer.

All pins are set in Input mode and no data is received or transmitted.

If a transfer is in progress, the transfer is finished before the QSPI is disabled.

If both QSPIEN and QSPIDIS are equal to one when QSPI_CR is written, the QSPI is disabled.

- **SWRST: QSPI Software Reset**

0: No effect.

1: Reset the QSPI. A software-triggered hardware reset of the QSPI interface is performed.

DMA channels are not affected by software reset.

- **LASTXFER: Last Transfer**

0: No effect.

1: The chip select is deasserted after the character written in QSPI_TDR.TD has been transferred.

47.7.2 QSPI Mode Register

Name: QSPI_MR

Address: 0xF0020004 (0), 0xF0024004 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
DLYCS							
23	22	21	20	19	18	17	16
DLYBCT							
15	14	13	12	11	10	9	8
–	–	–	–	NBBITS			
7	6	5	4	3	2	1	0
–	–	CSMODE		SMRM	WDRBT	LLB	SMM

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

- **SMM: Serial Memory Mode**

0 (SPI): The QSPI is in SPI mode.

1 (MEMORY): The QSPI is in Serial Memory mode.

- **LLB: Local Loopback Enable**

0 (DISABLED): Local loopback path disabled.

1 (ENABLED): Local loopback path enabled.

LLB controls the local loopback on the data serializer for testing in SPI mode only. (MISO is internally connected on MOSI).

- **WDRBT: Wait Data Read Before Transfer**

0 (DISABLED): No effect. In SPI mode, a transfer can be initiated whatever the state of the QSPI_RDR is.

1 (ENABLED): In SPI mode, a transfer can start only if the QSPI_RDR is empty, i.e., does not contain any unread data. This mode prevents overrun error in reception.

- **SMRM: Serial Memory Register Mode**

0: Serial Memory registers are written via AHB access. See Section 47.6.5.2 for details.

1: Serial Memory registers are written via APB access. See Section 47.6.5.2 for details.

- **CSMODE: Chip Select Mode**

The CSMODE field determines how the chip select is deasserted

Note: This field is forced to LASTXFER when SMM is written to ‘1’.

Value	Name	Description
0	NOT_RELOADED	The chip select is deasserted if QSPI_TDR.TD has not been reloaded before the end of the current transfer.
1	LASTXFER	The chip select is deasserted when the bit LASTXFER is written to ‘1’ and the character written in QSPI_TDR.TD has been transferred.
2	SYSTEMATICALLY	The chip select is deasserted systematically after each transfer.

- **NBBITS: Number Of Bits Per Transfer**

Value	Name	Description
0	8_BIT	8 bits for transfer
8	16_BIT	16 bits for transfer

- **DLYCS: Minimum Inactive QCS Delay**

This field defines the minimum delay between the deactivation and the activation of QCS. The DLYCS time guarantees the slave minimum deselect time.

If DLYCS written to ‘0’, one peripheral clock period is inserted by default.

Otherwise, the following equation determines the delay:

DLYCS = Minimum inactive $\times f_{peripheral\ clock}$

- **DLYBCT: Delay Between Consecutive Transfers**

This field defines the delay between two consecutive transfers with the same peripheral without removing the chip select. The delay is always inserted after each transfer and before removing the chip select if needed.

When DLYBCT is written to ‘0’, no delay between consecutive transfers is inserted and the clock keeps its duty cycle over the character transfers. In Serial Memory mode (SMM = 1), DLYBCT must be written to ‘0’ and no delay is inserted.

Otherwise, the following equation determines the delay:

DLYBCT = (Delay Between Consecutive Transfers $\times f_{peripheral\ clock}$) / 32

47.7.3 QSPI Receive Data Register

Name: QSPI_RDR

Address: 0xF0020008 (0), 0xF0024008 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RD							
7	6	5	4	3	2	1	0
RD							

- **RD: Receive Data**

Data received by the QSPI is stored in this register right-justified. Unused bits read zero.

47.7.4 QSPI Transmit Data Register

Name: QSPI_TDR

Address: 0xF002000C (0), 0xF002400C (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TD							
7	6	5	4	3	2	1	0
TD							

- **TD: Transmit Data**

Data to be transmitted by the QSPI is stored in this register. Information to be transmitted must be written to the Transmit Data register in a right-justified format.

47.7.5 QSPI Status Register

Name: QSPI_SR

Address: 0xF0020010 (0), 0xF0024010 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	QSPIENS
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	INSTRE	CSS	CSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	TXEMPTY	TDRE	RDRF

- **RDRF: Receive Data Register Full (cleared by reading QSPI_RDR)**

0: No data has been received since the last read of QSPI_RDR.

1: Data has been received and the received data has been transferred from the serializer to QSPI_RDR since the last read of QSPI_RDR.

- **TDRE: Transmit Data Register Empty (cleared by writing QSPI_TDR)**

0: Data has been written to QSPI_TDR and not yet transferred to the serializer.

1: The last data written in the QSPI_TDR has been transferred to the serializer.

TDRE equals zero when the QSPI is disabled or at reset. The QSPI enable command sets this bit to one.

- **TXEMPTY: Transmission Registers Empty (cleared by writing QSPI_TDR)**

0: As soon as data is written in QSPI_TDR.

1: QSPI_TDR and the internal shifter are empty. If a transfer delay has been defined, TXEMPTY is set after the completion of such delay.

- **OVRES: Overrun Error Status (cleared on read)**

0: No overrun has been detected since the last read of QSPI_SR.

1: At least one overrun error has occurred since the last read of QSPI_SR.

An overrun occurs when QSPI_RDR is loaded at least twice from the serializer since the last read of the QSPI_RDR.

- **CSR: Chip Select Rise (cleared on read)**

0: No chip select rise has been detected since the last read of QSPI_SR.

1: At least one chip select rise has been detected since the last read of QSPI_SR.

- **CSS: Chip Select Status**

0: The chip select is asserted.

1: The chip select is not asserted.

- **INSTRE: Instruction End Status (cleared on read)**

0: No instruction end has been detected since the last read of QSPI_SR.

1: At least one instruction end has been detected since the last read of QSPI_SR.

- **QSPIENS: QSPI Enable Status**

0: QSPI is disabled.

1: QSPI is enabled.

47.7.6 QSPI Interrupt Enable Register

Name: QSPI_IER

Address: 0xF0020014 (0), 0xF0024014 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	INSTRE	CSS	CSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	TXEMPTY	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Enable**
- **TDRE: Transmit Data Register Empty Interrupt Enable**
- **TXEMPTY: Transmission Registers Empty Enable**
- **OVRES: Overrun Error Interrupt Enable**
- **CSR: Chip Select Rise Interrupt Enable**
- **CSS: Chip Select Status Interrupt Enable**
- **INSTRE: Instruction End Interrupt Enable**

47.7.7 QSPI Interrupt Disable Register

Name: QSPI_IDR

Address: 0xF0020018 (0), 0xF0024018 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	INSTRE	CSS	CSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	TXEMPTY	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **RDRF: Receive Data Register Full Interrupt Disable**
- **TDRE: Transmit Data Register Empty Interrupt Disable**
- **TXEMPTY: Transmission Registers Empty Disable**
- **OVRES: Overrun Error Interrupt Disable**
- **CSR: Chip Select Rise Interrupt Disable**
- **CSS: Chip Select Status Interrupt Disable**
- **INSTRE: Instruction End Interrupt Disable**

47.7.8 QSPI Interrupt Mask Register

Name: QSPI_IMR

Address: 0xF002001C (0), 0xF002401C (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	INSTRE	CSS	CSR
7	6	5	4	3	2	1	0
–	–	–	–	OVRES	TXEMPTY	TDRE	RDRF

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **RDRF: Receive Data Register Full Interrupt Mask**
- **TDRE: Transmit Data Register Empty Interrupt Mask**
- **TXEMPTY: Transmission Registers Empty Mask**
- **OVRES: Overrun Error Interrupt Mask**
- **CSR: Chip Select Rise Interrupt Mask**
- **CSS: Chip Select Status Interrupt Mask**
- **INSTRE: Instruction End Interrupt Mask**

47.7.9 QSPI Serial Clock Register

Name: QSPI_SCR

Address: 0xF0020020 (0), 0xF0024020 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
DLYBS							
15	14	13	12	11	10	9	8
SCBR							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	CPHA	CPOL

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

- **CPOL: Clock Polarity**

0: The inactive state value of QSCK is logic level zero.

1: The inactive state value of QSCK is logic level one.

CPOL is used to determine the inactive state value of the serial clock (QSCK). It is used with CPHA to produce the required clock/data relationship between master and slave devices.

- **CPHA: Clock Phase**

0: Data is captured on the leading edge of QSCK and changed on the following edge of QSCK.

1: Data is changed on the leading edge of QSCK and captured on the following edge of QSCK.

CPHA determines which edge of QSCK causes data to change and which edge causes data to be captured. CPHA is used with CPOL to produce the required clock/data relationship between master and slave devices.

- **SCBR: Serial Clock Baud Rate**

The QSPI uses a modulus counter to derive the QSCK baud rate from the peripheral clock. The baud rate is selected by writing a value from 0 to 255 in the SCBR field. The following equation determines the QSCK baud rate:

SCBR = ($f_{peripheral\ clock}$ / QSCK Baud rate) - 1

- **DLYBS: Delay Before QSCK**

This field defines the delay from QCS valid to the first valid QSCK transition.

When DLYBS equals zero, the QCS valid to QSCK transition is 1/2 the QSCK clock period.

Otherwise, the following equation determines the delay:

DLYBS = Delay Before QSCK $\times$ $f_{peripheral\ clock}$

47.7.10 QSPI Instruction Address Register

Name: QSPI_IAR

Address: 0xF0020030 (0), 0xF0024030 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR							

23	22	21	20	19	18	17	16
ADDR							

15	14	13	12	11	10	9	8
ADDR							

7	6	5	4	3	2	1	0
ADDR							

- **ADDR: Address**

Address to send to the serial Flash memory in the instruction frame.

47.7.11 QSPI Instruction Code Register

Name: QSPI_ICR

Address: 0xF0020034 (0), 0xF0024034 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
OPT							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
INST							

- **INST: Instruction Code**

Instruction code to send to the serial Flash memory.

- **OPT: Option Code**

Option code to send to the serial Flash memory.

47.7.12 QSPI Instruction Frame Register

Name: QSPI_IFR

Address: 0xF0020038 (0), 0xF0024038 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	NBDUM				

15	14	13	12	11	10	9	8
–	CRM	TFRTYP		–	ADDRL	OPTL	

7	6	5	4	3	2	1	0
DATAEN	OPTEN	ADDREN	INSTEN	–	WIDTH		

- **WIDTH: Width of Instruction Code, Address, Option Code and Data**

Value	Name	Description
0	SINGLE_BIT_SPI	Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Single-bit SPI
1	DUAL_OUTPUT	Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Dual SPI
2	QUAD_OUTPUT	Instruction: Single-bit SPI / Address-Option: Single-bit SPI / Data: Quad SPI
3	DUAL_IO	Instruction: Single-bit SPI / Address-Option: Dual SPI / Data: Dual SPI
4	QUAD_IO	Instruction: Single-bit SPI / Address-Option: Quad SPI / Data: Quad SPI
5	DUAL_CMD	Instruction: Dual SPI / Address-Option: Dual SPI / Data: Dual SPI
6	QUAD_CMD	Instruction: Quad SPI / Address-Option: Quad SPI / Data: Quad SPI

- **INSTEN: Instruction Enable**

0: The instruction is not sent to the serial Flash memory.

1: The instruction is sent to the serial Flash memory.

- **ADDREN: Address Enable**

0: The transfer address is not sent to the serial Flash memory.

1: The transfer address is sent to the serial Flash memory.

- **OPTEN: Option Enable**

0: The option is not sent to the serial Flash memory.

1: The option is sent to the serial Flash memory.

- **DATAEN: Data Enable**

0: No data is sent/received to/from the serial Flash memory.

1: Data is sent/received to/from the serial Flash memory.

- **OPTL: Option Code Length**

The OPTL field determines the length of the option code. The value written in OPTL must be coherent with value written in the field WIDTH. For example: OPTL = 0 (1-bit option code) is not coherent with WIDTH = 6 (option code sent with Quad-SPI protocol, thus the minimum length of the option code is 4 bits).

Value	Name	Description
0	OPTION_1BIT	The option code is 1 bit long.
1	OPTION_2BIT	The option code is 2 bits long.
2	OPTION_4BIT	The option code is 4 bits long.
3	OPTION_8BIT	The option code is 8 bits long.

- **ADDRL: Address Length**

The ADDRL bit determines the length of the address.

0 (24_BIT): The address is 24 bits long.

1 (32_BIT): The address is 32 bits long.

- **TFRTYP: Data Transfer Type**

Value	Name	Description
0	TRSFR_READ	Read transfer from the serial memory. Scrambling is not performed. Read at random location (fetch) in the serial Flash memory is not possible.
1	TRSFR_READ_MEMORY	Read data transfer from the serial memory. If enabled, scrambling is performed. Read at random location (fetch) in the serial Flash memory is possible.
2	TRSFR_WRITE	Write transfer into the serial memory. Scrambling is not performed.
3	TRSFR_WRITE_MEMORY	Write data transfer into the serial memory. If enabled, scrambling is performed.

- **CRM: Continuous Read Mode**

0 (DISABLED): The Continuous Read mode is disabled.

1 (ENABLED): The Continuous Read mode is enabled.

- **NBDUM: Number Of Dummy Cycles**

The NBDUM field defines the number of dummy cycles required by the serial Flash memory before data transfer.

47.7.13 QSPI Scrambling Mode Register

Name: QSPI_SMR

Address: 0xF0020040 (0), 0xF0024040 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	RVDIS	SCREN

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

- **SCREN: Scrambling/Unscrambling Enable**

0 (DISABLED): The scrambling/unscrambling is disabled.

1 (ENABLED): The scrambling/unscrambling is enabled.

- **RVDIS: Scrambling/Unscrambling Random Value Disable**

0: The scrambling/unscrambling algorithm includes the user scrambling key plus a random value that may differ between devices.

1: The scrambling/unscrambling algorithm includes only the user scrambling key.

47.7.14 QSPI Scrambling Key Register

Name: QSPI_SKR

Address: 0xF0020044 (0), 0xF0024044 (1)

Access: Write-only

31	30	29	28	27	26	25	24
USRK							

23	22	21	20	19	18	17	16
USRK							

15	14	13	12	11	10	9	8
USRK							

7	6	5	4	3	2	1	0
USRK							

This register can only be written if bit WPEN is cleared in the QSPI Write Protection Mode Register.

- **USRK: User Scrambling Key**

47.7.15 QSPI Write Protection Mode Register

Name: QSPI_WPMR

Address: 0xF00200E4 (0), 0xF00240E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII)

1: Enables the write protection if WPKEY corresponds to 0x515350 (QSP in ASCII)

See Section 47.6.7 “Register Write Protection” for the list of registers that can be protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x515350	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

47.7.16 QSPI Write Protection Status Register

Name: QSPI_WPSR

Address: 0xF00200E8 (0), 0xF00240E8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of the QSPI_WPSR.

1: A write protection violation has occurred since the last read of the QSPI_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

48. Secure Digital Multimedia Card Controller (SDMMC)

48.1 Description

The Secure Digital Multimedia Card Controller (SDMMC) supports the embedded MultiMedia Card (e.MMC) Specification V4.51, the SD Memory Card Specification V3.0, and the SDIO V3.0 specification. It is compliant with the SD Host Controller Standard V3.0 specification.

The SDMMC includes the register set defined in the "SD Host Controller Simplified Specification V3.00" and additional registers to manage e.MMC devices, sampling tuning procedure, PAD calibration and enhanced features.

The SDMMC is clocked by three asynchronous clocks (see Section 48.5 "Block Diagram") and requires the PMC to be configured first.

48.2 Embedded Characteristics

- Compatible with SD Host Controller Standard Specification Version 3.00
- Compatible with MultiMedia Card Specification Version 4.51
- Compatible with SD Memory Card Specification Version 3.00
- Compatible with SDIO Specification Version 3.00
- Support for 1-bit/ 4-bit SD/SDIO Devices
- Support for 1-bit/4-bit/8-bit e.MMC Devices
- Support for SD/SDIO Default Speed (Maximum SDCLK Frequency = 25 MHz)
- Support for SD/SDIO High Speed (Maximum SDCLK Frequency = 50 MHz)
- Support for SD/SDIO UHS-I SDR12 (Maximum SDCLK Frequency = 25 MHz)
- Support for SD/SDIO UHS-I SDR25 (Maximum SDCLK Frequency = 50 MHz)
- Support for SD/SDIO UHS-I SDR50 (Maximum SDCLK Frequency = 100 MHz)
- Support for SD/SDIO UHS-I SDR104 (Maximum SDCLK Frequency = 120 MHz)
- Support for SD/SDIO UHS-I DDR50 (Maximum SDCLK Frequency = 50 MHz)
- Support for SDSC, SDHC and SDXC
- Support for e.MMC Default Speed (Maximum SDCLK Frequency = 26 MHz)
- Support for e.MMC High Speed (Maximum SDCLK Frequency = 52 MHz)
- Support for e.MMC High Speed DDR (Maximum SDCLK Frequency = 52 MHz)
- Support for e.MMC HS200 (Maximum SDCLK Frequency = 120 MHz)
- e.MMC Boot Operation Mode Support
- Support for Block Size from 1 to 512 bytes
- Support for Stream, Block and Multi-block Data Read and Write
  - Advanced DMA and SDMA Capability
- Internal 1024-byte Dual Port RAM
- Support for both synchronous and asynchronous abort
- Supports for SDIO Card Interrupt

48.3 Embedded Features for SDMMC0 and SDMMC1

The device embeds two SDMMC interfaces; both do not support the same features.

SDMMC0 supports all the features listed in Section 48.2 “Embedded Characteristics”.

SDMMC1 is compatible with SD Memory Card Specification Version 3.00 and does **not** support the following features:

- 8-bit e.MMC Devices
- SD/SDIO UHS-I SDR12 (Maximum SDCLK Frequency = 25 MHz)
- SD/SDIO UHS-I SDR25 (Maximum SDCLK Frequency = 50 MHz)
- SD/SDIO UHS-I SDR50 (Maximum SDCLK Frequency = 100 MHz)
- SD/SDIO UHS-I SDR104 (Maximum SDCLK Frequency = 120 MHz)
- SD/SDIO UHS-I DDR50 (Maximum SDCLK Frequency = 50 MHz)
- e.MMC HS200 (Maximum SDCLK Frequency = 120 MHz)

In addition, the SDMMC1 pin interface does not embed SDMMC_1V8SEL and the SDMMC_DAT is limited to SDMMC_DAT[3..0].

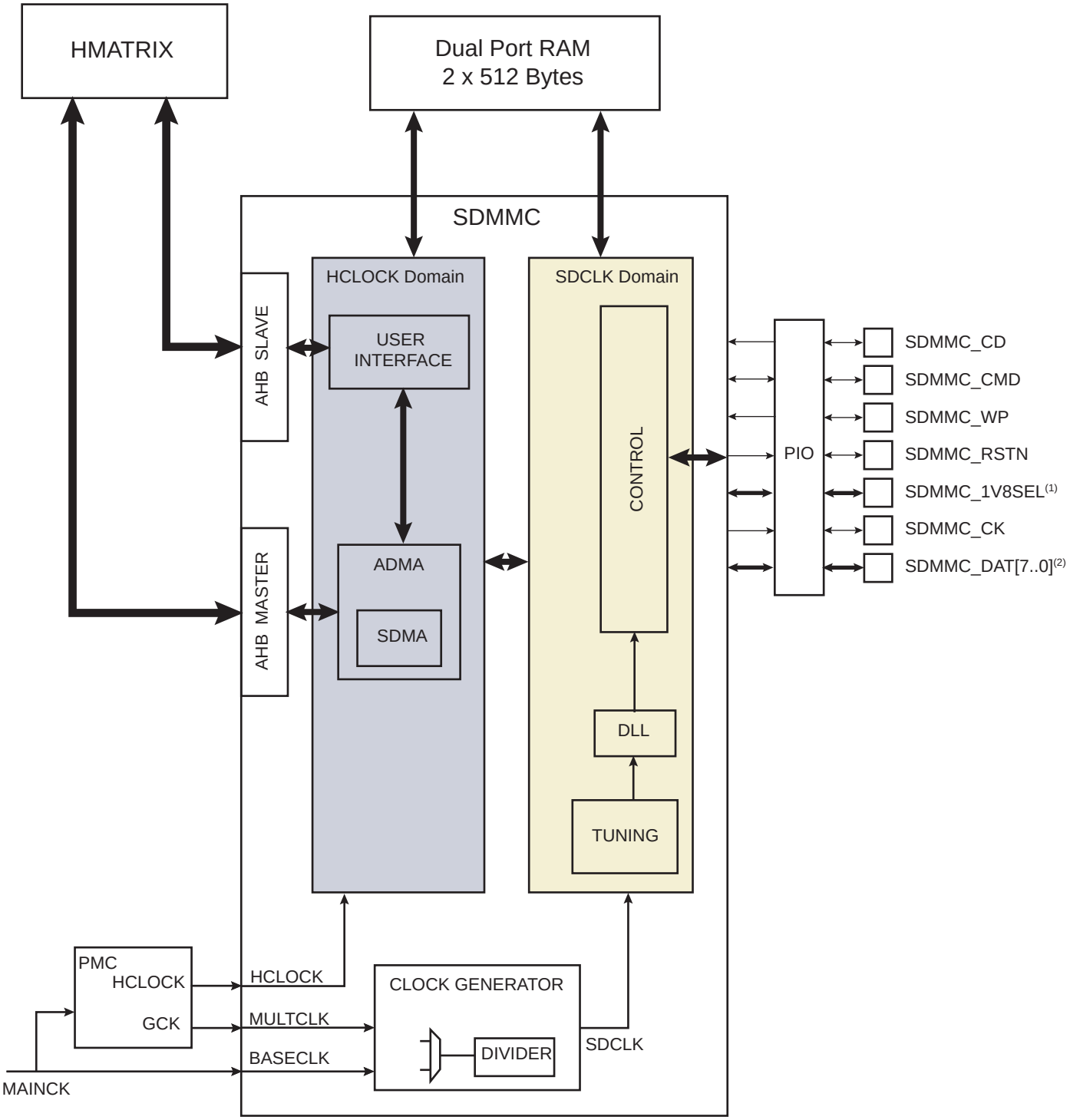
48.4 Reference Documents

Table 48-1. Reference Documents

Name	Link
SD Host Controller Simplified Specification V3.00	https://www.sdcard.org
SDIO Simplified Specification V3.00	
Physical Layer Simplified Specification V3.01	
Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51	http://www.jedec.org

48.5 Block Diagram

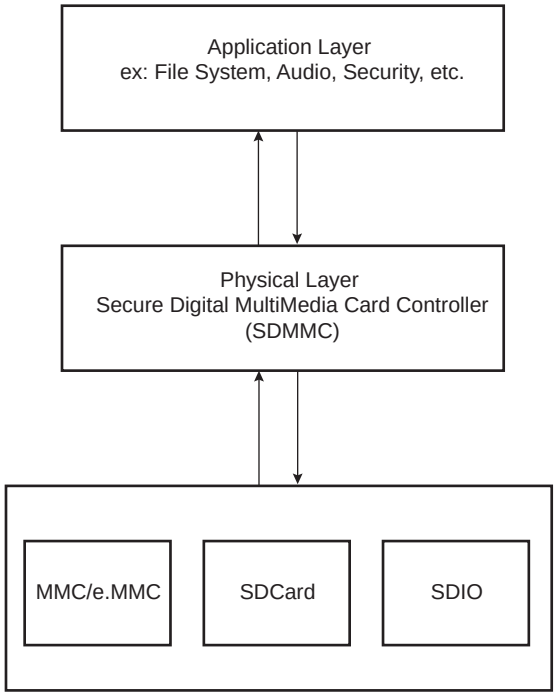
Figure 48-1. Block Diagram



Notes: 1. SDMMC_1V8SEL not available in SDMMC1
2. Limited to SDMMC_DAT[3..0] in SDMMC1

48.6 Application Block Diagram

Figure 48-2. Application Block Diagram



48.7 Pin Name List

Table 48-2. I/O Lines Description for 8-bit Configuration

Pin Name[1]	Pin Description	Type
SDMMC_CD	SDcard / SDIO / e.MMC Card Detect	Input
SDMMC_CMD	SDcard / SDIO / e.MMC Command/Response Line	I/O
SDMMC_WP	SDcard Connector Write Protect Signal	Input
SDMMC_RSTN	e.MMC Reset Signal	Output
SDMMC_1V8SEL	SDcard Signal Voltage Selection	Output
SDMMC_CK	SDcard / SDIO / e.MMC Clock Signal	Output
SDMMC_DAT[7..0]	SDcard / SDIO / e.MMC Data Lines	I/O

Notes: 1. When several SDMMCs are embedded in a product, SDMMC_CK refers to SDMMCx_CK, SDMMC_CMD to SDMMCx_CMD, SDMMC_DATy to SDMMCx_DATy, SDMMC_WP to SDMMCx_WP, SDMMC_1V8SEL to SDMMCx_1V8SEL, SDMMC_CD to SDMMCx_CD and SDMMC_RSTN to SDMMCx_RSTN.

48.8 Product Dependencies

48.8.1 I/O Lines

The pins used for interfacing the Secure Digital MultiMedia Card (SDMMC) Controller are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the peripheral functions to SDMMC pins.

48.8.2 Power Management

The SDMMC is clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the SDMMC clocks.

48.8.3 Interrupt Sources

The SDMMC has an interrupt line connected to the interrupt controller.

Handling the SDMMC interrupt requires programming the interrupt controller before configuring the SDMMC.

48.9 SD/SDIO Operating Mode

The SDMMC is fully compliant with the “SD Host Controller Simplified Specification V3.00” for SD/SDIO devices. Refer to this specification for SDMMC configuration.

Refer to “Physical Layer Simplified Specification V3.01” and “SDIO Simplified Specification V3.00” for SD/SDIO management.

48.10 e.MMC Operating Mode

The SDMMC supports e.MMC devices management. As the “SD Host Controller Simplified Specification V3.00” does not apply to e.MMC devices, some registers have been added to those described in this specification in order to manage e.MMC devices. Most of the registers described in the “SD Host Controller Simplified Specification V3.00” must be used for e.MMC management, but e.MMC-specific features are managed using SDMMC_MC1R and SDMMC_MC2R.

48.10.1 Boot Operation Mode

In Boot Operation mode, the processor can read boot data from the e.MMC device by keeping the CMD line low after power-on before issuing the CMD1. The data can be read from either one of the boot partitions or the user area according to BOOT_PARTITION_ENABLE in the Extended CSD register (see “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51”).

48.10.1.1 Boot Procedure, Processor Mode

1. Configure SDMMC:
    1. Set the data bus width using SDMMC_HC1R.DW and SDMMC_HC1R.EXTDW according to BOOT_BUS_WIDTH in the Extended CSD Register (see “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51”).
    2. Select the speed mode (using SDMMC_HC1R.HSEN or SDMMC_MC1R.DDR) according to BOOT_MODE in the Extended CSD Register.
    3. Set the SDCLK frequency according to the selected speed mode.
    4. If the Boot Acknowledge is sent by the e.MMC device (BOOT_ACK = 1 in the Extended CSD Register), set the Boot Acknowledge Enable to 1 (SDMMC_MC1R.BOOTA = 1).
    5. Enable interrupt on Boot Acknowledge Received (SDMMC_NISTER.BOOTAR = 1 and SDMMC_NISIER.BOOTAR = 1).
    6. Set the e.MMC Command Type to BOOT (SDMMC_MC1R.CMDTYP = 3)

7. Set SDMMC_TMR to read multiple blocks for the e.MMC device (SDMMC_TMR.MSBSEL = 1 and SDMMC_TMR.DTDSEL = 1).
8. Select the Non-DMA transfer (SDMMC_TMR.DMAEN = 0).
9. Optional: select the Auto CMD method (using SDMMC_TMR.ACMDEN).
10. Set the block size to 512 bytes (SDMMC_BSR.BLKSIZE = 512).
11. Set the required number of read blocks (using SDMMC_BCR.BLKCNT). SDMMC_TMR.BCEN must be set to 1.

2. Write SDMMC_CR = 20(hexa) to set the e.MMC in Boot Operation mode.
3. Wait for interrupt on Boot Acknowledge Received (BOOTAR).
4. The user can copy the boot data sequentially as soon as the BRDRDY flag is asserted.
5. When the data transfer is completed, the boot operation must be terminated by setting SDMMC_MC2R.ABOOT to 1.

48.10.1.2 Boot Procedure, SDMA Mode

1. Configure SDMMC:
    1. Set the data bus width using SDMMC_HC1R.DW and SDMMC_HC1R.EXTDW according to BOOT_BUS_WIDTH in the Extended CSD Register (see "Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51").
    2. Select the speed mode (SDMMC_HC1R.HSEN or SDMMC_MC1R.DDR) according to BOOT_MODE in the Extended CSD Register .
    3. Set the SDCLK frequency according to the selected speed mode.
    4. If the Boot Acknowledge is sent by the e.MMC device (BOOT_ACK = 1 in the Extended CSD Register), set the Boot Acknowledge Enable to 1 (SDMMC_MC1R.BOOTA = 1).
    5. Enable interrupt on Boot Acknowledge Received (SDMMC_NISTER.BOOTAR = 1 and SDMMC_NISIER.BOOTAR = 1).
    6. Set the e.MMC Command Type to BOOT (SDMMC_MC1R.CMDTYP = 3).
    7. Set SDMMC_TMR to read multiple blocks for the e.MMC device (SDMMC_TMR.MSBSEL = 1 and SDMMC_TMR.TDSEL = 1).
    8. Select the SDMA transfer (SDMMC_TMR.DMAEN = 1 and SDMMC_HC1R.DMASEL = 0).
    9. Write the SDMA system address where the boot data will be copied (SDMMC_SSAR.ADDR).
    10. Optional: select the Auto CMD method (SDMMC_TMR.ACMDEN).
    Note: Auto CMD23 cannot be used with SDMA.
    11. Set the block size to 512 bytes (SDMMC_BSR.BLKSIZE = 512).
    12. Set the required number of read blocks (SDMMC_BCR.BLKCNT). SDMMC_TMR.BCEN must be set to 1.
2. Write SDMMC_CR = 20(hexa) to set the e.MMC in Boot Operation mode.
3. Wait for interrupt on Boot Acknowledge Received (BOOTAR).
4. The user can copy the boot data sequentially as soon as the BRDRDY flag is asserted.
5. When the data transfer is completed, the boot operation must be terminated by setting SDMMC_MC2R.ABOOT to 1.

48.10.1.3 Boot Procedure, ADMA Mode

1. Configure SDMMC:
    1. Set the data bus width using SDMMC_HC1R.DW and SDMMC_HC1R.EXTDW according to BOOT_BUS_WIDTH in the Extended CSD Register (see "Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51").
    2. Select the speed mode (SDMMC_HC1R.HSEN or SDMMC_MC1R.DDR) according to BOOT_MODE in the Extended CSD Register .
    3. Set the SDCLK frequency according to the selected speed mode.

4. If the Boot Acknowledge is sent by the e.MMC device (BOOT_ACK = 1 in the Extended CSD Register), set the Boot Acknowledge Enable to 1 (SDMMC_MC1R.BOOTA = 1).
5. Enable interrupt on Boot Acknowledge Received (SDMMC_NISTER.BOOTAR = 1 and SDMMC_NISIER.BOOTAR = 1).
6. Set the e.MMC Command Type to BOOT (SDMMC_MC1R.CMDTYP = 3).
7. Set SDMMC_TMR to read multiple blocks for the e.MMC device (SDMMC_TMR.MSBSEL = 1 and SDMMC_TMR.DTDSEL = 1).
8. Select the ADMA transfer (SDMMC_TMR.DMAEN = 1 and SDMMC_HC1R.DMASEL = 2 or 3).
9. Write the address of the descriptor table in the ADMA system address (SDMMC_ASARx [0..1].ADMASA).
10. Optional: select the Auto CMD method (SDMMC_TMR.ACMDEN).
    Note: Auto CMD23 cannot be used with SDMA.
11. Set the block size to 512 bytes (SDMMC_BSR.BLKSIZE = 512).
12. Set the required number of read blocks (SDMMC_BCR.BLKCNT). SDMMC_TMR.BCEN must be set to 1.

2. Write SDMMC_CR = 20(hexa) to set the e.MMC in Boot Operation Mode.
3. Wait for interrupt on Boot Acknowledge Received (BOOTAR).
4. The user can copy the boot data sequentially as soon as the BRDRDY flag is asserted.
5. When the data transfer is completed, the boot operation must be terminated by setting SDMMC_MC2R.ABOOT to 1.

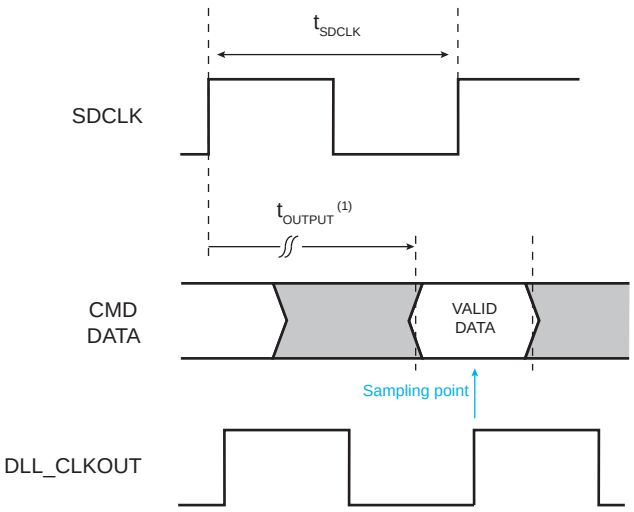
48.11 SDR104 / HS200 Tuning

48.11.1 DLL and Sampling Point

In SD/SDIO SDR104 mode (VS18EN = 1 and UHSMS = 3 in SDMMC_HC2R) or e.MMC HS200 mode (HS200EN = $B_{(hexa)}$), a tuning procedure must be performed first in order to adjust the sampling point for read transactions. For more details regarding the basic tuning procedure, refer to “Sampling Clock Tuning Procedure” in the “SD Host Controller Simplified Specification V3.00”.

As the position of data and command coming from the device varies, a DLL is used to generate an accurate sampling point (DLL_CLKOUT)(see Figure 48-3).

Figure 48-3. DLL Sampling Point

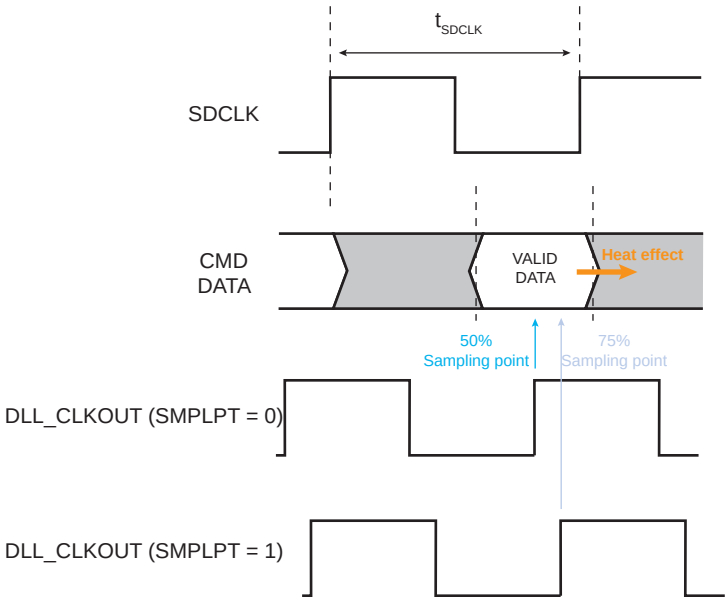


Note: 1. t_{OUTPUT} varies from 0 to 2 x t_{SDCLK}

The minimum SDLCK frequency is 100 MHz when SD/SDIO SDR104 or e.MMC HS200 is selected.

The sampling point can be selected to be located at 50% or 75% of the data window to anticipate the effect of the temperature rise. If the SMPLPT bit is cleared in SDMMC_TUNCR, the sampling point is centered (50% of the data window). If the SMPLPT bit is set to 1 in SDMMC_TUNCR, the sampling point is set at 75% of the data window (see Figure 48-4).

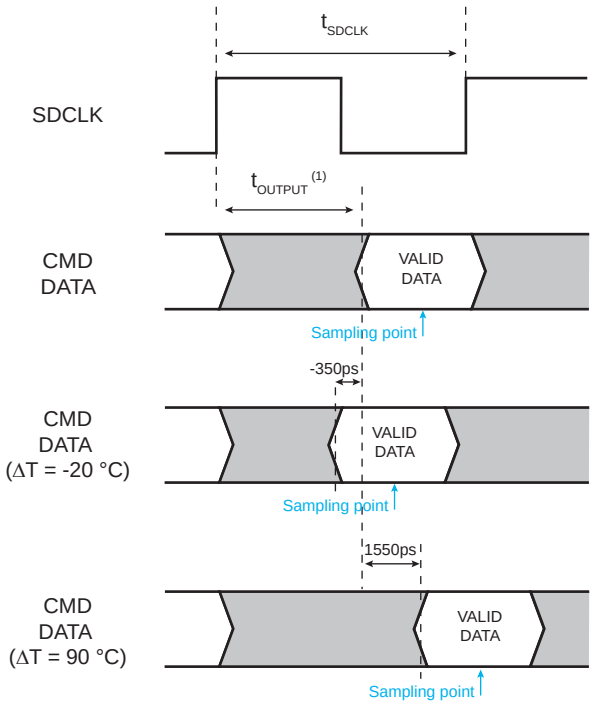
Figure 48-4. SDR104/HS200 Sampling Point Selection



48.11.2 Retuning Method

Once the data window sampling point has been tuned following the tuning procedure, the data window can be shifted by temperature drift. Thus, the tuning procedure must be applied periodically to adjust the sampling point position. The SDMMC implements a retuning timer which periodically instructs the software to restart the tuning procedure.

Figure 48-5. Temperature Effect on Data Window



Note: 1. t_{OUTPUT} varies from 0 to 2 x t_{SDCLK}

48.11.2.1 SDMMC Tuning Sequence

The SDMMC tuning sequence must only be done when SD/SDIO SDR104 or e.MMC HS200 is selected and for a 100-MHz SDCLK frequency or higher.

1. Enable the retuning timer (TMREN = 1 in SDMMC_RTC1R).
2. Configure the retuning period by setting TCVAL in SDMMC_RTCVR.
3. Set ‘Retuning Timer Event’ (TEVT) to 1 in SDMMC_RTISTER so that the TEVT status flag in SDMMC_RTISTR rises each time the retuning timer counter period elapses.
4. Set TEVT to 1 in SDMMC_RTISIER to generate an interrupt on the TEVT status flag assertion (optional).
5. Execute the tuning procedure as defined in “Sampling Clock Tuning Procedure” in the “SD Host Controller Simplified Specification V3.00”.
6. Start the retuning timer count (write RLD to 1 in SDMMC_RTC2R). At this step, data can be read by the SDMMC.
7. Each time TEVT is set to 1 in SDMMC_RTISTR:
    1. Execute the tuning procedure as defined in “Sampling Clock Tuning Procedure” in the “SD Host Controller Simplified Specification V3.00” before issuing the next command.
    2. Re-start the retuning timer count (write RLD to 1 in SDMMC_RTC2R).
    3. Resume data reading from the device.

When several instances of SDMMC are implemented in a product, the TEVT status flag of each SDMMC instance can be checked by reading SDMMC_RTSSR.

48.12 I/O Calibration

The need for output impedance calibration arises with higher data rates. As the data rate increases, some transmission line effects can occur and lead to the generation of undershoots and overshoots, hence degrading the signal quality.

To avoid these transmission problems, an I/O calibration cell is used to adjust the output impedance to the driven I/Os.

The I/O calibration sequence is mandatory when one of the SD/SDIO UHS-I modes (VS18EN = 1 in SDMMC_HC2R) or e.MMC HS200 (HS200EN = $B_{(hexa)}$) is selected. It must be performed periodically to prevent the output impedance drift. Once the calibration is finished, the I/O calibration cell provides two four-bit control words (CALP[3:0] and CALN[3:0] in SDMMC_CALCR) to tune the output impedance, and thus reach the best transmission performances.

The I/O calibration sequence can be started manually by writing the EN bit to 1 in SDMMC_CALCR. The EN bit is cleared automatically at the end of the calibration.

The I/O calibration sequence can also be performed automatically if the TUNDIS bit is cleared in SDMMC_CALCR. In such case, the calibration starts automatically at the beginning of the tuning procedure when writing EXTUN to 1 in SDMMC_HC2R.

The I/O calibration cell requires a startup time defined by the CNTVAL field in SDMMC_CALCR. Thus, CNTVAL must be configured prior to start the calibration sequence. If ALWYSON is set to 1 in SDMMC_CALCR, the startup time is only required for the first calibration sequence as the analog circuitry is not shut down at the end of the calibration. In order to reduce the power consumption, the analog circuitry can be shut down at the end of the calibration sequence by clearing the ALWYSON bit. In this case, the startup time is performed each time a calibration sequence is started.

48.13 Secure Digital MultiMedia Card Controller (SDMMC) User Interface

Table 48-3. Register Mapping

Offset	Register	Name	Access	Reset
0x00	SDMA System Address / Argument 2 Register	SDMMC_SSAR	Read/Write	0x0
0x04	Block Size Register	SDMMC_BSR	Read/Write	0x0
0x06	Block Count Register	SDMMC_BCR	Read/Write	0x0
0x08	Argument 1 Register	SDMMC_ARG1R	Read/Write	0x0
0x0C	Transfer Mode Register	SDMMC_TMR	Read/Write	0x0
0x0E	Command Register	SDMMC_CR	Read/Write	0x0
0x10	Response Register 0	SDMMC_RR0	Read-only	0x0
0x14	Response Register 1	SDMMC_RR1	Read-only	0x0
0x18	Response Register 2	SDMMC_RR2	Read-only	0x0
0x1C	Response Register 3	SDMMC_RR3	Read-only	0x0
0x20	Buffer Data Port Register	SDMMC_BDPR	Read/Write	–(1)
0x24	Present State Register	SDMMC_PSR	Read-only	0x00F8_0000
0x28	Host Control 1 Register	SDMMC_HC1R	Read/Write	0x0
0x29	Power Control Register	SDMMC_PCR	Read/Write	0x0E
0x2A	Block Gap Control Register	SDMMC_BGCR	Read/Write	0x0
0x2B	Wakeup Control Register	SDMMC_WCR	Read/Write	0x0
0x2C	Clock Control Register	SDMMC_CCR	Read/Write	0x0
0x2E	Timeout Control Register	SDMMC_TCR	Read/Write	0x0
0x2F	Software Reset Register	SDMMC_SRR	Read/Write	0x0
0x30	Normal Interrupt Status Register	SDMMC_NISTR	Read/Write	0x0
0x32	Error Interrupt Status Register	SDMMC_EISTR	Read/Write	0x0
0x34	Normal Interrupt Status Enable Register	SDMMC_NISTER	Read/Write	0x0
0x36	Error Interrupt Status Enable Register	SDMMC_EISTER	Read/Write	0x0
0x38	Normal Interrupt Signal Enable Register	SDMMC_NISIER	Read/Write	0x0
0x3A	Error Interrupt Signal Enable Register	SDMMC_EISIER	Read/Write	0x0
0x3C	Auto CMD Error Status Register	SDMMC_ACESR	Read-only	0x0
0x3E	Host Control 2 Register	SDMMC_HC2R	Read/Write	0x0
0x40	Capabilities 0 Register	SDMMC_CA0R	Read-only	0x27EC_0C8C(2) 0x27E8_0C8C(3)
0x44	Capabilities 1 Register	SDMMC_CA1R	Read/Write	0x0020_0F77(2) 0x0020_0070(3)
0x48	Maximum Current Capabilities Register	SDMMC_MCCAR	Read/Write	0x0
0x4C	Reserved	–	–	–
0x50	Force Event Register for Auto CMD Error Status	SDMMC_FERACES	Write-only	–
0x52	Force Event Register for Error Interrupt Status	SDMMC_FEREIS	Write-only	–
0x54	ADMA Error Status Register	SDMMC_AESR	Read-only	0x0

Table 48-3. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x58	ADMA System Address Register 0	SDMMC_ASAR0	Read/Write	0x0
0x5C	Reserved	–	–	–
0x60	Preset Value Register 0 (for initialization)	SDMMC_PVR0	Read/Write	0x0
0x62	Preset Value Register 1 (for Default Speed)	SDMMC_PVR1	Read/Write	0x0
0x64	Preset Value Register 2 (for High Speed)	SDMMC_PVR2	Read/Write	0x0
0x66	Preset Value Register 3 (for SDR12)	SDMMC_PVR3	Read/Write	0x0
0x68	Preset Value Register 4 (for SDR25)	SDMMC_PVR4	Read/Write	0x0
0x6A	Preset Value Register 5 (for SDR50)	SDMMC_PVR5	Read/Write	0x0
0x6C	Preset Value Register 6 (for SDR104)	SDMMC_PVR6	Read/Write	0x0
0x6E	Preset Value Register 7 (for DDR50)	SDMMC_PVR7	Read/Write	0x0
0x70–0xF8	Reserved	–	–	–
0xFC	Slot Interrupt Status Register	SDMMC_SISR	Read-only	0x0
0xFE	Host Controller Version Register	SDMMC_HCVR	Read-only	0x1502
0x100–0x1FC	Reserved	–	–	–
0x200	Additional Present State Register	SDMMC_APSR	Read-only	0x0000_000F(2) 0x0000_0000(3)
0x204	MMC Control 1 Register	SDMMC_MC1R	Read/Write	0x0
0x205	MMC Control 2 Register	SDMMC_MC2R	Write-only	–
0x208	AHB Control Register	SDMMC_ACR	Read/Write	0x0
0x20C	Clock Control 2 Register	SDMMC_CC2R	Read/Write	0x0
0x210	Retuning Timer Control 1 Register	SDMMC_RTC1R	Read/Write	0x0
0x211	Retuning Timer Control 2 Register	SDMMC_RTC2R	Write-only	–
0x214	Retuning Timer Counter Value Register	SDMMC_RTCVR	Read/Write	0x0
0x218	Retuning Timer Interrupt Status Enable Register	SDMMC_RTISTER	Read/Write	0x0
0x219	Retuning Timer Interrupt Signal Enable Register	SDMMC_RTISIER	Read/Write	0x0
0x21C	Retuning Timer Interrupt Status Register	SDMMC_RTISTR	Read/Write	0x0
0x21D	Retuning Timer Status Slots Register	SDMMC_RTSSR	Read-only	0x0
0x220	Tuning Control Register	SDMMC_TUNCR	Read/Write	0x0
0x224–0x22C	Reserved	–	–	–
0x230	Capabilities Control Register	SDMMC_CACR	Read/Write	0x0
0x234–0x23C	Reserved	–	–	–
0x240	Calibration Control Register	SDMMC_CALCR	Read/Write	0x00005000
0x244–0x2FC	Reserved	–	–	–

Notes: 1. Unpredictable value read from the dual port RAM
2. Reset value for SDMMC0 instance
3. Reset value for SDMMC1 instance

48.13.1 SDMMC SDMA System Address / Argument 2 Register

Name: SDMMC_SSAR

Access: Read/Write

31	30	29	28	27	26	25	24
ADDR / ARG2							
23	22	21	20	19	18	17	16
ADDR / ARG2							
15	14	13	12	11	10	9	8
ADDR / ARG2							
7	6	5	4	3	2	1	0
ADDR / ARG2							

This register contains the physical system memory address used for SDMA transfers or the second argument for Auto CMD23.

- **ADDR: SDMA System Address**

This field is the system memory address for a SDMA transfer. When the SDMMC stops an SDMA transfer, this field points to the system address of the next contiguous data position. This field can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value. An interrupt can be generated to instruct the software to update this field. Writing the next system address of the next data position restarts the SDMA transfer.

- **ARG2: Argument 2**

This field is used with Auto CMD23 to set a 32-bit block count value to the CMD23 argument while executing Auto CMD23. If Auto CMD23 is used with ADMA, the full 32-bit block count value can be used. If Auto CMD23 is used without ADMA, the available block count value is limited by SDMMC_BCR. In this case, 65535 blocks is the maximum value.

48.13.2 SDMMC Block Size Register

Name: SDMMC_BSR

Access: Read/Write

15	14	13	12	11	10	9	8
–	BOUNDARY			–	–	BLKSIZE	

7	6	5	4	3	2	1	0
BLKSIZE							

- **BLKSIZE: Transfer Block Size**

This field specifies the block size of data transfers for CMD17, CMD18, CMD24, CMD25 and CMD53. Values ranging from 1 to 512 can be set. It can be accessed only if no transaction is executing (i.e., after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations are ignored.

- **BOUNDARY: SDMA Buffer Boundary**

This field specifies the size of the contiguous buffer in the system memory. The SDMA transfer waits at every boundary specified by this field and the SDMMC generates the DMA Interrupt to instruct the software to update SDMMC_SSAR. If this field is set to 0 (buffer size = 4 Kbytes), the lowest 12 bits of SDMMC_SSAR.ADDRESS point to data in the contiguous buffer, and the upper 20 bits point to the location of the buffer in the system memory. This function is active when the DMAEN bit is set in SDMMC_TMR.

Value	Name	Description
0	4K	4-Kbyte boundary
1	8K	8-Kbyte boundary
2	16K	16-Kbyte boundary
3	32K	32-Kbyte boundary
4	64K	64-Kbyte boundary
5	128K	128-Kbyte boundary
6	256k	256-Kbyte boundary
7	512K	512-Kbyte boundary

48.13.3 SDMMC Block Count Register

Name: SDMMC_BCR

Access: Read/Write

15	14	13	12	11	10	9	8
BLKCNT							
7	6	5	4	3	2	1	0
BLKCNT							

- **BLKCNT: Block Count for Current Transfer**

This field is used only if SDMMC_TMR.BCEN (Block Count Enable) is set to 1 and is valid only for multiple block transfers. BLKCNT is the number of blocks to be transferred and it must be set to a value between 1 and the maximum block count. The SDMMC decrements the block count after each block transfer and stops when the count reaches 0. When this field is set to 0, no data block is transferred.

This register should be accessed only when no transaction is executing (i.e., after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored.

When a suspend command is completed, the number of blocks yet to be transferred can be determined by reading this register. Before issuing a resume command, the previously saved block count is restored.

48.13.4 SDMMC Argument 1 Register

Name: SDMMC_ARG1R

Access: Read/Write

31	30	29	28	27	26	25	24
ARG1							
23	22	21	20	19	18	17	16
ARG1							
15	14	13	12	11	10	9	8
ARG1							
7	6	5	4	3	2	1	0
ARG1							

- **ARG1: Argument 1**

This register contains the SD command argument which is specified as the bit 39-8 of Command-Format in the “Physical Layer Simplified Specification V3.01” or “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51”.

48.13.5 SDMMC Transfer Mode Register

Name: SDMMC_TMR

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	MSBSEL	DTDSEL	ACMDEN		BCEN	DMAEN

This register is used to control data transfers. The user shall set this register before issuing a command which transfers data (refer to bit DPSEL in SDMMC_CR), or before issuing a Resume command. The user must save the value of this register when the data transfer is suspended (as a result of a Suspend command) and restore it before issuing a Resume command. To prevent data loss, this register cannot be written while data transactions are in progress. Writes to this register are ignored when bit SDMMC_PSR.CMDINHD is 1.

Table 48-4. Determining the Transfer Type

MSBSEL	BCEN	BLKCNT (SDMMC_BCR)	Function
0	Don't care	Don't care	Single Transfer
1	0	Don't care	Infinite Transfer
1	1	Not Zero	Multiple Transfer
1	1	Zero	Stop Multiple Transfer

- **DMAEN: DMA Enable**

This bit enables the DMA functionality described in section "Supporting DMA" in "SD Host Controller Simplified Specification V3.00". DMA can be enabled only if it is supported as indicated by the bit SDMMC_CA0R.ADMA2SUP. One of the DMA modes can be selected using the field SDMMC_HC1R.DMASEL. If DMA is not supported, this bit is meaningless and then always reads 0. When this bit is set to 1, a DMA operation begins when the user writes to the upper byte of SDMMC_CR.

0 (DISABLED): DMA functionality is disabled.

1 (ENABLED): DMA functionality is enabled.

- **BCEN: Block Count Enable**

This bit is used to enable SDMMC_BCR, which is only relevant for multiple block transfers. When this bit is 0, SDMMC_BCR is disabled, which is useful when executing an infinite transfer (refer to Table 48-4). If an ADMA2 transfer is more than 65535 blocks, this bit is set to 0 and the data transfer length is designated by the Descriptor Table.

0 (DISABLED): Block count is disabled.

1 (ENABLED): Block count is enabled.

- **ACMDEN: Auto Command Enable**

Two methods can be used to stop Multiple-block read and write operation:

- Auto CMD12: when the ACMDEN field is set to 1, the SDMMC issues CMD12 automatically when the last block transfer is completed. An Auto CMD12 error is indicated to SDMMC_ACESR. Auto CMD12 is not enabled if the command does not require CMD12.
- Auto CMD23: when the ACMDEN field is set to 2, the SDMMC issues a CMD23 automatically before issuing a command specified in SDMMC_CR.

The following conditions are required to use Auto CMD23:

– A memory card that supports CMD23 (SCR[33] = 1)

– If DMA is used, it must be ADMA (SDMA not supported).

– Only CMD18 or CMD25 is issued.

Note: The SDMMC does not check the command index.

Auto CMD23 can be used with or without ADMA. By writing SDMMC_CR, the SDMMC issues a CMD23 first and then issues a command specified by the SDMMC_CR.CMDIDX field. If CMD23 response errors are detected, the second command is not issued. A CMD23 error is indicated in SDMMC_ACESR. The CMD23 argument (32-bit block count value) is set in SDMMC_SSAR.

This field determines the use of auto command functions.

Value	Name	Description
0	DISABLED	Auto Command Disabled
1	CMD12	Auto CMD12 Enabled
2	CMD23	Auto CMD23 Enabled
3	–	Reserved

- **DTDSEL: Data Transfer Direction Selection**

This bit defines the direction of the DAT lines data transfers. Set this bit to 1 to transfer data from the device (SD Card/SDIO/e.MMC) to the SDMMC, and to 0 for all other commands.

0 (WRITE): Writes data from the SDMMC to the device.

1 (READ): Reads data from the device to the SDMMC.

- **MSBSEL: Multi/Single Block Selection**

This bit is set to 1 when issuing multiple-block transfer commands using DAT line(s). For any other commands, set this bit to 0. If this bit is 0, it is not necessary to set SDMMC_BCR (refer to Table 48-4).

48.13.6 SDMMC Command Register

Name: SDMMC_CR

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	CMDIDX					
7	6	5	4	3	2	1	0
CMDTYP		DPSEL	CMDICEN	CMDCCEN	–	RESPTYP	

- **RESPTYP: Response Type**

This field is set according to the response type expected for the command index (CMDIDX).

Value	Name	Description
0	NORESP	No Response
1	RL136	Response Length 136
2	RL48	Response Length 48
3	RL48BUSY	Response Length 48 with Busy

- **CMDCCEN: Command CRC Check Enable**

If this bit is set to 1, the SDMMC checks the CRC field in the response. If an error is detected, it is reported as a Command CRC Error (CMDCRC) in SDMMC_EISTR. If this bit is set to 0, the CRC field is not checked. The position of the CRC field is determined according to the length of the response.

0 (DISABLED): The Command CRC Check is disabled.

1 (ENABLED): The Command CRC Check is enabled.

- **CMDICEN: Command Index Check Enable**

If this bit is set to 1, the SDMMC checks the Index field in the response to see if it has the same value as the command index. If it has not, it is reported as a Command Index Error (CMDIDX) in SDMMC_EISTR. If this bit is set to 0, the Index field of the response is not checked.

0 (DISABLED): The Command Index Check is disabled.

1 (ENABLED): The Command Index Check is enabled.

- **DPSEL: Data Present Select**

This bit is set to 1 to indicate that data is present and shall be transferred using the DAT lines. It is set to 0 for the following:

- Commands using only CMD line (Ex. CMD52)
- Commands with no data transfer but using Busy signal on DAT[0] line (Ex. CMD38)
- Resume command

0: No data present

1: Data present

- **CMDTYP: Command Type**

Value	Name	Description
0	NORMAL	Other commands
1	SUSPEND	CMD52 to write “Bus Suspend” in the Card Common Control Registers (CCCR) (for SDIO only)
2	RESUME	CMD52 to write “Function Select” in the Card Common Control Registers (CCCR) (for SDIO only)
3	ABORT	CMD12, CMD52 to write “I/O Abort” in the Card Common Control Registers (CCCR) (for SDIO only)

- **CMDIDX: Command Index**

This bit shall be set to the command number (CMD0-63, ACMD0-63) that is specified in bits 45-40 of the Command-Format in the “Physical Layer Simplified Specification V3.01”, “SDIO Simplified Specification V3.00”, and “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51”.

48.13.7 SDMMC Response Register

Name: SDMMC_RRx [x=0..3]

Access: Read-only

31	30	29	28	27	26	25	24
CMDRESP							
23	22	21	20	19	18	17	16
CMDRESP							
15	14	13	12	11	10	9	8
CMDRESP							
7	6	5	4	3	2	1	0
CMDRESP							

- **CMDRESP: Command Response**

The table below describes the mapping of command responses from the SD_SDIO/eMMC bus to these registers for each responses type. In this table, R[] refers to a bit range of the response data as transmitted on the SD_SDIO/eMMC bus.

Type of response	Meaning of response	Response field	Response register
R1, R1b (normal response)	Card Status	R[39:8]	SDMMC_RR0[31:0]
R1b (Auto CMD12 response)	Card Status for Auto CMD12	R[39:8]	SDMMC_RR3[31:0]
R1 (Auto CMD23 response)	Card Status for Auto CMD23	R[39:8]	SDMMC_RR3[31:0]
R2 (CID, CSD register)	CID or CSD register	R[127:8]	SDMMC_RR0[31:0] SDMMC_RR1[31:0] SDMMC_RR2[31:0] SDMMC_RR3[23:0]
R3 (OCR register)	OCR register for memory	R[39:8]	SDMMC_RR0[31:0]
R4 (OCR register)	OCR register for I/O	R[39:8]	SDMMC_RR0[31:0]
R5, R5b	SDIO response	R[39:8]	SDMMC_RR0[31:0]
R6 (Published RCA response)	New published RCA[31:16] and Card status bits	R[39:8]	SDMMC_RR0[31:0]

48.13.8 SDMMC Buffer Data Port Register

Name: SDMMC_BDPR

Access: Read/Write

31	30	29	28	27	26	25	24
BUFDATA							
23	22	21	20	19	18	17	16
BUFDATA							
15	14	13	12	11	10	9	8
BUFDATA							
7	6	5	4	3	2	1	0
BUFDATA							

- **BUFDATA: Buffer Data**

The SDMMC data buffer can be accessed through this 32-bit Data Port register.

48.13.9 SDMMC Present State Register

Name: SDMMC_PSR

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	CMDLL
23	22	21	20	19	18	17	16
DATLL				WRPPL	CARDDPL	CARDSS	CARDINS
15	14	13	12	11	10	9	8
–	–	–	–	BUFRDEN	BUFWREN	RTACT	WTACT
7	6	5	4	3	2	1	0
–	–	–	–	–	DLACT	CMDINHD	CMDINHC

- **CMDINHC: Command Inhibit (CMD)**

If this bit is 0, it indicates the CMD line is not in use and the SDMMC can issue a command using the CMD line. This bit is set to 1 immediately after SDMMC_CR is written. This bit is cleared when the command response is received. Auto CMD12 and Auto CMD23 consist of two responses. In this case, this bit is not cleared by the CMD12 or CMD23 response, but by the Read/Write command response.

Status issuing Auto CMD12 is not read from this bit. So, if a command is issued during Auto CMD12 operation, the SDMMC manages to issue both commands: CMD12 and a command set by SDMMC_CR.

Even if the Command Inhibit (DAT) is set to 1, commands using only the CMD line can be issued if this bit is 0.

A change from 1 to 0 raises the Command Complete (CMDC) status flag in SDMMC_NISTR if SDMMC_NISTER.CMDC is set to 1. An interrupt is generated if SDMMC_NISIER.CMDC is set to 1.

If the SDMMC cannot issue the command because of a command conflict error (refer to CMDCRC in SDMMC_EISTR) or because of a 'Command Not Issued By Auto CMD12' error (refer to Section 48.13.31 "SDMMC Auto CMD Error Status Register"), this bit remains 1 and Command Complete is not set.

0: Can issue a command using only CMD line.

1: Cannot issue a command.

- **CMDINHD: Command Inhibit (DAT)**

This status bit is 1 if either the DAT Line Active (DLACT) or the Read Transfer Active (RTACT) is set to 1. If this bit is 0, it indicates that the SDMMC can issue the next command. Commands with a Busy signal belong to Command Inhibit (DAT) (ex. R1b, R5b type). A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1.

Note: The software can save registers in the 000-00Dh range for a suspend transaction after this bit has changed from 1 to 0.

0: Can issue a command which uses the DAT line(s).

1: Cannot issue a command which uses the DAT line(s).

- **DLACT: DAT Line Active**

This bit indicates whether one of the DAT lines on the bus is in use.

In the case of read transactions:

This status indicates whether a read transfer is executing on the bus. A change from 1 to 0 resulting from setting the Stop At Block Gap Request (STPBGR) raises the Block Gap Event (BLKGE) status flag in SDMMC_NISTR if SDMMC_NISTER.BLKGE is set to 1. An interrupt is generated if SDMMC_NISIER.BLKGE is set to 1. Refer to section "Read Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" for details on timing.

This bit is set in either of the following cases:

– After the end bit of the read command.

– When writing 1 to SDMMC_BGCR.CONTR (Continue Request) to restart a read transfer.

This bit is SDMMC cleared in either of the following cases:

– When the end bit of the last data block is sent from the bus to the SDMMC. In case of ADMA2, the last block is designated by the last transfer of the Descriptor Table.

– When a read transfer is stopped at the block gap initiated by a Stop At Block Gap Request (STPBGR).

The SDMMC stops a read operation at the start of the interrupt cycle by driving the Read Wait (DAT[2] line) or by stopping the SD Clock. If the Read Wait signal is already driven (due to the fact that the data buffer cannot receive data), the SDMMC can continue to stop the read operation by driving the Read Wait signal. It is necessary to support the Read Wait in order to use the Suspend/Resume operation.

In the case of write transactions:

This status indicates that a write transfer is executing on the bus. A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1. Refer to section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for details on timing.

This bit is set in either of the following cases:

– After the end bit of the write command.

– When writing 1 to SDMMC_BGCR.CONTR (Continue Request) to continue a write transfer.

This bit is cleared in either of the following cases:

– When the card releases Write Busy of the last data block. If the card does not drive a Busy signal for 8 SDCLK, the SDMMC considers the card drive “Not Busy”. In the case of ADMA2, the last block is designated by the last transfer of the Descriptor Table.

– When the card releases Write Busy prior to wait for write transfer as a result of a Stop At Block Gap Request (STPBGR).

Command with Busy:

This status indicates whether a command that indicates Busy (ex. erase command for memory) is executing on the bus. This bit is set to 1 after the end bit of the command with Busy and cleared when Busy is de-asserted. A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1. Refer to Figures 2.11 to 2.13 in the “SD Host Controller Simplified Specification V3.00”.

0: DAT Line Inactive

1: DAT Line Active

- **WTACT: Write Transfer Active**

This bit indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the SDMMC. Refer to section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit is set to 1 in either of the following conditions:

– After the end bit of the write command.

– When a write operation is restarted by writing a 1 to SDMMC_BGCR.CONTR (Continue Request).

This bit is cleared to 0 in either of the following conditions:

– After getting the CRC status of the last data block as specified by the transfer count (single and multiple). In case of ADMA2, transfer count is designated by the descriptor table.

– After getting the CRC status of any block where a data transmission is about to be stopped by a Stop At Block Gap Request (STPBGR) of SDMMC_BGCR.

During a write transaction and as the result of the Stop At Block Gap Request (STPBGR) being set, a change from 1 to 0 raises the Block Gap Event (BLKGE) status flag in SDMMC_NISTR if SDMMC_NISTER.BLKGE is set to 1. An interrupt is generated if BLKGE is set to 1 in SDMMC_NISIER. This status is useful to determine whether non-DAT line commands can be issued during Write Busy.

- **RTACT: Read Transfer Active**

This bit is used to detect completion of a read transfer. Refer to section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

This bit is set to 1 in either of the following conditions:

– After the end bit of the read command.

– When a read operation is restarted by writing a 1 to SDMMC_BGCR.CONTR (Continue Request).

This bit is cleared to 0 in either of the following conditions:

– When the last data block as specified by Transfer Block Size (BLKSIZE) is transferred to the system.

– In case of ADMA2, end of read is designated by the descriptor table.

– When all valid data blocks in the SDMMC have been transferred to the system and no current block transfers are being sent as a result of the Stop At Block Gap Request (STPBGR) of SDMMC_BGCR being set to 1.

A change from 1 to 0 raises the Transfer Complete (TRFC) status flag in SDMMC_NISTR if SDMMC_NISTER.TRFC is set to 1. An interrupt is generated if SDMMC_NISIER.TRFC is set to 1.

- **BUFWREN: Buffer Write Enable**

This bit is used for non-DMA write transfers. This flag indicates if space is available for write data. If this bit is 1, data can be written to the buffer.

A change from 1 to 0 occurs when all the block data are written to the buffer.

A change from 0 to 1 occurs when top of block data can be written to the buffer. This raises the Buffer Write Ready (BRWRDY) status flag in SDMMC_NISTR if SDMMC_NISTER.BRWRDY is set to 1. An interrupt is generated if SDMMC_NISIER.BRWRDY is set to 1.

- **BUFRDEN: Buffer Read Enable**

This bit is used for non-DMA read transfers. This flag indicates that valid data exists in the SDMMC data buffer. If this bit is 1, readable data exists in the buffer.

A change from 1 to 0 occurs when all the block data is read from the buffer.

A change from 0 to 1 occurs when block data is ready in the buffer. This raises the Buffer Read Ready (BRDRDY) status flag in SDMMC_NISTR if SDMMC_NISTER.BRDRDY is set to 1. An interrupt is generated if SDMMC_NISIER.BRDRDY is set to 1.

- **CARDINS: Card Inserted**

This bit indicates wether a card has been inserted. The SDMMC debounces this signal so that the user does not need to wait for it to stabilize.

A change from 0 to 1 raises the Card Insertion (CINS) status flag in SDMMC_NISTR if SDMMC_NISTER.CINS is set to 1. An interrupt is generated if SDMMC_NISIER.CINS is set to 1.

A change from 1 to 0 raises the Card Removal (CREM) status flag in SDMMC_NISTR if SDMMC_NISTER.CREM is set to 1. An interrupt is generated if SDMMC_NISIER.CREM is set to 1.

The Software Reset For All (SWRSTALL) in SDMMC_SRR does not affect this bit.

- **CARDSS: Card State Stable**

This bit is used for testing. If it is 0, the CARDDPL is not stable. If this bit is set to 1, it means that the CARDDPL is stable. No Card state can be detected if this bit is set to 1 and CARDINS is set to 0.

The Software Reset For All (SWRSTALL) in SDMMC_SRR does not affect this bit.

0: Reset or debouncing

1: No card or card inserted

- **CARDDPL: Card Detect Pin Level**

This bit reflects the inverse value of the SDMMC_CD pin. Debouncing is not performed on this bit. This bit may be valid when CARDSS is set to 1, but it is not guaranteed because of the propagation delay. Use of this bit is limited to testing since it must be debounced by software.

0: No card present (SDMMC_CD = 1)

1: Card present (SDMMC_CD = 0)

- **WRPPL: Write Protect Pin Level**

The Write Protect Switch is supported for memory and combo cards. This bit reflects the SDMMC_WP pin.

0: Write protected (SDMMC_WP = 0)

1: Write enabled (SDMMC_WP = 1)

- **DATLL: DAT[3:0] Line Level**

This status is used to check the DAT line level to recover from errors, and for debugging. This is especially useful in detecting the Busy signal level from DAT[0].

- **CMDLL: CMD Line Level**

This status is used to check the CMD line level to recover from errors, and for debugging.

48.13.10 SDMMC Host Control 1 Register (SD_SDIO)

Name: SDMMC_HC1R (SD_SDIO)

Access: Read/Write

7	6	5	4	3	2	1	0
CARDDSEL	CARDDTL	–	DMASEL		HSEN	DW	LEDCTRL

- **LEDCTRL: LED Control**

This bit is used to caution the user not to remove the card while it is being accessed. If the software is going to issue multiple commands, this bit is set to 1 during all transactions.

0 (OFF): LED off

1 (ON): LED on

- **DW: Data Width**

This bit selects the data width of the SDMMC. It must be set to match the data width of the card.

0 (1_BIT): 1-bit mode

1 (4_BIT): 4-bit mode

Note: If the Extended Data Transfer Width is 1, this bit has no effect and the data width is 8-bit mode.

- **HSEN: High Speed Enable**

Before setting this bit, the user must check the High Speed Support (HSSUP) in SDMMC_CA0R.

If this bit is set to 0 (default), the SDMMC outputs CMD line and DAT lines at the falling edge of the SD clock (up to 25 MHz). If this bit is set to 1, the SDMMC outputs the CMD line and the DAT lines at the rising edge of the SD clock (up to 50 MHz).

If Preset Value Enable (PVALEN) in SDMMC_HC2R is set to 1, the user needs to reset SD Clock Enable (SDCLKEN) before changing this bit to avoid generating clock glitches. After setting this bit to 1, the user sets SDCLEN to 1 again.

0: Normal Speed mode.

1: High Speed mode.

Note: 1. This bit is effective only if SDMMC_MC1R.DDR is set to 0.
2. The clock divider (DIV) in SDMMC_CCR must be set to a value different from 0 when HSEN is 1.

- **DMASEL: DMA Select**

One of the supported DAM modes can be selected. The user must check support of DMA modes by referring the SDMMC_CA0R. Use of selected DMA is determined by DMA Enable (DMAEN) in SDMMC_TMR.

Value	Name	Description
0	SDMA	SDMA is selected
1	–	Reserved
2	ADMA32	32-bit Address ADMA2 is selected
3	–	Reserved

- **CARDDTL: Card Detect Test Level**

This bit is enabled while the Card Detect Signal Selection (CARDDSEL) is set to 1 and it indicates whether the card is inserted or not.

0: No card.

1: Card inserted.

- **CARDDSEL: Card Detect Signal Selection**

This bit selects the source for the card detection.

0: The SDMMC_CD pin is selected.

1: The Card Detect Test Level (CARDDTL) is selected (for test purpose).

48.13.11 SDMMC Host Control 1 Register (eMMC)

Name: SDMMC_HC1R (eMMC)

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	EXTDW	DMASEL		HSEN	DW	–

- **DW: Data Width**

This bit selects the data width of the SDMMC. It must be set to match the data width of the card.

0 (1_BIT): 1-bit mode

1 (4_BIT): 4-bit mode

Note: If the Extended Data Transfer Width is 1, this bit has no effect and the data width is 8-bit mode.

- **HSEN: High Speed Enable**

Before setting this bit, the user must check the High Speed Support (HSSUP) in SDMMC_CA0R.

If this bit is set to 0 (default), the SDMMC outputs CMD line and DAT lines at the falling edge of the SD clock (up to 25 MHz). If this bit is set to 1, the SDMMC outputs the CMD line and the DAT lines at the rising edge of the SD clock (up to 50 MHz).

If Preset Value Enable (PVALEN) in SDMMC_HC2R is set to 1, the user needs to reset the SD Clock Enable (SDCLKEN) before changing this bit to avoid generating clock glitches. After setting this bit to 1, the user sets SDCLEN to 1 again.

0: Normal Speed mode.

1: High Speed mode.

Note: 1. This bit is effective only if SDMMC_MC1R.DDR is set to 0.
2. The clock divider (DIV) in SDMMC_CCR must be set to a value different from 0 when HSEN is 1.

- **DMASEL: DMA Select**

One of the supported DAM modes can be selected. The user must check support of DMA modes by referring the SDMMC_CA0R. Use of selected DMA is determined by DMA Enable (DMAEN) in SDMMC_TMR.

Value	Name	Description
0	SDMA	SDMA is selected
1	–	Reserved
2	ADMA32	32-bit Address ADMA2 is selected
3	–	Reserved

- **EXTDW: Extended Data Width**

This bit controls the 8-bit Bus Width mode for embedded devices. Support of this function is indicated in 8-bit Support for Embedded Device in SDMMC_CA0R. If a device supports the 8-bit mode, this may be set to 1. If this bit is 0, the bus width is controlled by Data Width (DW).

48.13.12 SDMMC Power Control Register

Name: SDMMC_PCR

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SDBPWR

- **SDBPWR: SD Bus Power**

This bit is automatically cleared by the SDMMC if the card is removed. If this bit is cleared, the SDMMC stops driving SDMMC_CMD and SDMMC_DAT[7:0] (tri-state) and drives SDMMC_CK to low level.

48.13.13 SDMMC Block Gap Control Register (SD_SDIO)

Name: SDMMC_BGCR (SD_SDIO)

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	INTBG	RWCTRL	CONTR	STPBGR

- **STPBGR: Stop At Block Gap Request**

This bit is used to stop executing read and write transactions at the next block gap for non-DMA, SDMA, and ADMA transfers. The user must leave this bit set to 1 until Transfer Complete (TRFC) in SDMMC_NISTR. Clearing both Stop At Block Gap Request and Continue Request does not cause the transaction to restart. This bit can be set whether the card supports the Read Wait signal or not.

During read transfers, the SDMMC stops the transaction by using the Read Wait signal (SDMMC_DAT[2]) if supported, or by stopping the SD clock otherwise.

In case of write transfers in which the user writes data to SDMMC_BDPR, this bit must be set to 1 after all the block of data is written. If this bit is set to 1, the user does not write data to SDMMC_BDPR.

This bit affects Read Transfer Active (RTACT), Write Transfer Active (WTACT), DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

Refer to the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

0: Transfer

1: Stop

- **CONTR: Continue Request**

This bit is used to restart a transaction which was stopped using a Stop At Block Gap Request (STPBGR). To cancel stop at the block gap, set STPBGR to 0 and set this bit to 1 to restart the transfer.

The SDMMC automatically clears this bit in either of the following cases:

– In the case of a read transaction, the DAT Line Active (DLACT) changes from 0 to 1 as a read transaction restarts.

– In the case of a write transaction, the Write Transfer Active (WTACT) changes from 0 to 1 as the write transaction restarts.

Therefore, it is not necessary to set this bit to 0. If STPBGR is set to 1, any write to this bit is ignored.

Refer to the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

0: No affect

1: Restart

- **RWCTRL: Read Wait Control**

The Read Wait control is optional for SDIO cards. If the card supports Read Wait, set this bit to enable use of the Read Wait protocol to stop read data using the SDMMC_DAT[2] line. Otherwise, the SDMMC stops the SDCLK to hold read data, which restricts command generation. When the software detects an SD card insertion, this bit must be set according to the CCCR of the SDIO card. If the card does not support Read Wait, this bit shall never be set to 1, otherwise an SDMMC_DAT line conflict may occur. If this bit is set to 0, Suspend/Resume cannot be supported.

0: Disables Read Wait control.

1: Enables Read Wait control.

- **INTBG: Interrupt at Block Gap**

This bit is valid only in 4-bit mode of the SDIO card and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. If the SDIO card cannot signal an interrupt during a multiple block transfer, this bit should be set to 0. When the software detects an SDIO card insertion, it sets this bit according to the CCCR of the SDIO card.

0 (DISABLED): Interrupt detection disabled

1 (ENABLED): Interrupt detection enabled

48.13.14 SDMMC Block Gap Control Register (eMMC)

Name: SDMMC_BGCR (eMMC)

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	CONTR	STPBGR

- **STPBGR: Stop At Block Gap Request**

This bit is used to stop executing read and write transactions at the next block gap for non-DMA, SDMA, and ADMA transfers. The user must leave this bit set to 1 until Transfer Complete (TRFC) in SDMMC_NISTR. Clearing both Stop At Block Gap Request and Continue Request does not cause the transaction to restart. This bit can be set whether the card supports the Read Wait signal or not.

During read transfers, the SDMMC stops the transaction by using the Read Wait signal (SDMMC_DAT[2]) if supported, or by stopping the SD clock otherwise.

In case of write transfers in which the user writes data to SDMMC_BDPR, this bit must be set to 1 after all the block of data is written. If this bit is set to 1, the user does not write data to SDMMC_BDPR.

This bit affects Read Transfer Active (RTACT), Write Transfer Active (WTACT), DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

Refer to the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

0: Transfer

1: Stop

- **CONTR: Continue Request**

This bit is used to restart a transaction which was stopped using a Stop At Block Gap Request (STPBGR). To cancel stop at the block gap, set STPBGR to 0 and set this bit to 1 to restart the transfer.

The SDMMC automatically clears this bit in either of the following cases:

– In the case of a read transaction, the DAT Line Active (DLACT) changes from 0 to 1 as a read transaction restarts.

– In the case of a write transaction, the Write Transfer Active (WTACT) changes from 0 to 1 as the write transaction restarts.

Therefore, it is not necessary to set this bit to 0. If STPBGR is set to 1, any write to this bit is ignored.

Refer to the “Abort Transaction” and “Suspend/Resume” sections in the “SD Host Controller Simplified Specification V3.00” for more details.

0: No affect

1: Restart

48.13.15 SDMMC Wakeup Control Register (SD_SDIO)

Name: SDMMC_WCR (SD_SDIO)

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	WKENCREM	WKENCINS	WKENCINT

- **WKENCINT: Wakeup Event Enable on Card Interrupt**

This bit enables a wakeup event via Card Interrupt (CINT) in SDMMC_NISTR. This bit can be set to 1 if FN_WUS (Wakeup Support) in the CIS (Card Information Structure) is set to 1 in the SDIO card.

0 (DISABLED): Wakeup Event disabled

1 (ENABLED): Wakeup Event enabled

- **WKENCINS: Wakeup Event Enable on Card Insertion**

This bit enables a wakeup event via Card Insertion (CINS) in SDMMC_NISTR. FN_WUS (Wakeup Support) in the CIS (Card Information Structure) does not affect this bit.

0 (DISABLED): Wakeup Event disabled

1 (ENABLED): Wakeup Event enabled

- **WKENCREM: Wakeup Event Enable on Card Removal**

This bit enables a wakeup event via Card Removal (CREM) in SDMMC_NISTR. FN_WUS (Wakeup Support) in the CIS (Card Information Structure) does not affect this bit.

0 (DISABLED): Wakeup Event disabled

1 (ENABLED): Wakeup Event enabled

48.13.16 SDMMC Clock Control Register

Name: SDMMC_CCR

Access: Read/Write

15	14	13	12	11	10	9	8
SDCLKFSEL							
7	6	5	4	3	2	1	0
USDCLKFSEL		CLKGSEL	–	–	SDCLKEN	INTCLKS	INTCLKEN

- **INTCLKEN: Internal Clock Enable**

This bit is set to 0 when the SDMMC is not used or is awaiting a wakeup interrupt. In this case, its internal clock is stopped to reach a very low power state. Registers are still able to be read and written. The clock starts to oscillate when this bit is set to 1. Once the clock oscillation is stable, the SDMMC sets Internal Clock Stable (INTCLKS) in this register to 1.

This bit does not affect card detection.

0: The internal clock stops.

1: The internal clock oscillates.

- **INTCLKS: Internal Clock Stable**

This bit is set to 1 when the SD clock is stable after setting SDMMC_CCR.INTCLKEN (Internal Clock Enable) to 1. The user must wait to set SD Clock Enable (SDCLKEN) until this bit is set to 1.

0: Internal clock not ready

1: Internal clock ready

- **SDCLKEN: SD Clock Enable**

The SDMMC stops the SD Clock when writing this bit to 0. SDCLK Frequency Select (SDCLKFSEL) can be changed when this bit is 0. Then, the SDMMC maintains the same clock frequency until SDCLK is stopped (Stop at SDCLK = 0). If Card Inserted (CARDINS) in SDMMC_PSR is cleared, this bit is also cleared.

0: SD Clock disabled

1: SD Clock enabled

- **CLKGSEL: Clock Generator Select**

This bit is used to select the clock generator mode in the SDCLK Frequency Select field. If the Programmable mode is not supported (SDMMC_CA1R.CLKMULT (Clock Multiplier) set to 0), then this bit cannot be written and is always read at 0.

This bit depends on the setting of Preset Value Enable (PVALEN) in SDMMC_HC2R.

If PVALEN = 0, this bit is set by the user.

If PVALEN = 1, this bit is automatically set to a value specified in one of the SDMMC_PVRx.

0: Divided Clock mode (BASECLK is used to generate SDCLK).

1: Programmable Clock mode (MULTCLK is used to generate SDCLK).

- **USDCLKFSEL: Upper Bits of SDCLK Frequency Select**

These bits expand the SDCLK Frequency Select (SDCLKFSEL) to 10 bits. These two bits are assigned to bit 09-08 of the clock divider as described in SDCLKFSEL.

- **SDCLKFSEL: SDCLK Frequency Select**

This register is used to select the frequency of the SDCLK pin. There are two SDCLK Frequency modes according to Clock Generator Select (CLKGSEL).

The length of the clock divider (DIV) is extended to 10 bits (DIV[9:8] = USDCLKFSEL, DIV[7:0] = SDCLKFSEL)

– 10-bit Divided Clock Mode (CLKGSEL = 0): $f_{SDCLK} = f_{BASECLK}/(2 \times DIV)$. If DIV = 0 then $f_{SDCLK} = f_{BASECLK}$

– Programmable Clock Mode (CLKGSEL = 1): $f_{SDCLK} = f_{MULTCLK}/(DIV + 1)$

This field depends on the setting of Preset Value Enable (PVALEN) in SDMMC_HC2R.

If PVALEN = 0, this field is set by the user.

If PVALEN = 1, this field is automatically set to a value specified in one of the SDMMC_PVR.

48.13.17 SDMMC Timeout Control Register

Name: SDMMC_TCR

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	DTCVAL			

- **DTCVAL: Data Timeout Counter Value**

This value determines the interval at which DAT line timeouts are detected. For more information about timeout generation, refer to Data Timeout Error (DATTEO) in SDMMC_EISTR. When setting this register, the user can prevent inadvertent timeout events by clearing the Data Timeout Error Status Enable (in SDMMC_EISTER).

$$TIMEOUT_{(\mu s)} = \frac{2^{13 + DTCVAL}}{f_{BASECLK(MHz)}}$$

Note: DTCVAL = $f_{(Hexa)}$ is reserved.

48.13.18 SDMMC Software Reset Register

Name: SDMMC_SRR

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	SWRSTDAT	SWRSTCMD	SWRSTALL

- **SWRSTALL: Software reset for All**

This reset affects the entire SDMMC except the card detection circuit. During initialization, the SDMMC must be reset by setting this bit to 1. This bit is automatically cleared to 0 when SDMMC_CA0R and SDMMC_CA1R are valid and the user can read them. If this bit is set to 1, the user should issue a reset command and reinitialize the card.

List of registers cleared to 0:

– "SDMMC SDMA System Address / Argument 2 Register"
– "SDMMC Block Size Register"
– "SDMMC Block Count Register"
– "SDMMC Argument 1 Register"
– "SDMMC Command Register"
– "SDMMC Transfer Mode Register"
– "SDMMC Response Register"
– "SDMMC Buffer Data Port Register"
– "SDMMC Present State Register" (except CMDLL, DATLL, WRPPL, CARDDDPL, CARDSS, CARDINS)
– "SDMMC Host Control 1 Register (SD_SDIO)"
– "SDMMC Host Control 1 Register (eMMC)"
– "SDMMC Power Control Register"
– "SDMMC Block Gap Control Register (SD_SDIO)"
– "SDMMC Block Gap Control Register (eMMC)"
– "SDMMC Wakeup Control Register (SD_SDIO)"
– "SDMMC Clock Control Register"
– "SDMMC Timeout Control Register"
– "SDMMC Normal Interrupt Status Register (SD_SDIO)"
– "SDMMC Error Interrupt Status Register (SD_SDIO)"
– "SDMMC Normal Interrupt Status Enable Register (SD_SDIO)"
– "SDMMC Error Interrupt Status Enable Register (SD_SDIO)"
– "SDMMC Normal Interrupt Signal Enable Register (SD_SDIO)"
– "SDMMC Error Interrupt Signal Enable Register (SD_SDIO)"
– "SDMMC Auto CMD Error Status Register"
– "SDMMC Host Control 2 Register (SD_SDIO)"
– "SDMMC ADMA Error Status Register"
– "SDMMC ADMA System Address Register"
– "SDMMC Slot Interrupt Status Register" (except NBCLKP, NBINTP, BWTPRE)
– "SDMMC Slot Interrupt Status Register"
– "SDMMC e.MMC Control 1 Register"
– "SDMMC e.MMC Control 2 Register"

– “SDMMC AHB Control Register”
– “SDMMC Clock Control 2 Register”
– “SDMMC Retuning Control 1 Register”
– “SDMMC Retuning Counter Value Register”
– “SDMMC Retuning Interrupt Status Enable Register”
– “SDMMC Retuning Interrupt Signal Enable Register”
– “SDMMC Retuning Interrupt Status Register”
– “SDMMC Tuning Control Register”
– “SDMMC Capabilities Control Register” (except KEY)

0: Work

1: Reset

- **SWRSTCMD: Software reset for CMD line**

Only part of a command circuit is reset.

The following registers and bits are cleared by this bit:

“SDMMC Present State Register”

– Command Inhibit (CMD) (CMDINHC)

“SDMMC Normal Interrupt Status Register (SD_SDIO)” and “SDMMC Normal Interrupt Status Register (eMMC)”

– Command Complete (CMDC)

0: Work

1: Reset

- **SWRSTDAT: Software reset for DAT line**

Only part of a data circuit is reset. The DMA circuit is also reset.

The following registers and bits are cleared by this bit:

“SDMMC Buffer Data Port Register”

– Buffer is cleared and initialized.

“SDMMC Present State Register”

– Buffer Read Enable (BUFRDEN)
– Buffer Write Enable (BUFWREN)
– Read Transfer Active (RTACT)
– Write Transfer Active (WTACT)
– DAT Line Active (DATLL)
– Command Inhibit (DAT) (CMDINHD)

“SDMMC Block Gap Control Register (SD_SDIO)”

– Continue Request (CONTR)
– Stop At Block Gap Request (STPBGR)

“SDMMC Normal Interrupt Status Register (SD_SDIO)”

– Buffer Read Ready (BRDRDY)
– Buffer Write Ready (BWRRDY)
– DMA Interrupt (DMAINT)

– Block Gap Event (BLKGE)
– Transfer Complete (TRFC)

0: Work

1: Reset

48.13.19 SDMMC Normal Interrupt Status Register (SD_SDIO)

Name: SDMMC_NISTR (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
ERRINT	–	–	–	–	–	–	CINT
7	6	5	4	3	2	1	0
CREM	CINS	BRDRDY	BWRRDY	DMAINT	BLKGE	TRFC	CMDC

- **CMDC: Command Complete**

This bit is set when getting the end bit of the command response. Auto CMD12 and Auto CMD23 consist of two responses. Command Complete is not generated by the response of CMD12 or CMD23, but it is generated by the response of a read/write command. Refer to Command Inhibit (CMD) in SDMMC_PSR for details on how to control this bit.

This bit can only be set to 1 if SDMMC_NISTER.CMDC is set to 1. An interrupt can only be generated if SDMMC_NISIER.CMDC is set to 1.

Writing this bit to 1 clears this bit.

The table below shows that Command Timeout Error (CMDTEO) has a higher priority than Command Complete (CMDC). If both bits are set to 1, it can be considered that the response was not received correctly.

CMDC	CMDTEO	Meaning of the status
0	0	Interrupted by another factor
Don’t care	1	Response not received within 64 SDCLK cycles
1	0	Response received

0: No command complete

1: Command complete

- **TRFC: Transfer Complete**

This bit is set when a read/write transfer and a command with Busy is completed.

In the case of a Read Transaction:

This bit is set at the falling edge of the Read Transfer Active Status. The interrupt is generated in two cases. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR (after valid data has been read to the system). Refer to section “Read Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

In the case of a Write Transaction:

This bit is set at the falling edge of the DAT Line Active (DLACT) status. This interrupt is generated in two cases. The first is when the last data is written to the card as specified by the data length and the Busy signal is released. The second is when data transfers are stopped at the block gap by setting Stop At Block Gap Request (STPBGR) in SDMMC_BGCR and data transfers are completed. (After valid data is written to the card and the Busy signal is released). Refer to section “Write Transaction Wait / Continue Timing” in the “SD Host Controller Simplified Specification V3.00” for more details on the sequence of events.

In the case of command with Busy:

This bit is set when Busy is de-asserted. Refer to DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

This bit can only be set to 1 if SDMMC_NISTER.TRFC is set to 1. An interrupt can only be generated if SDMMC_NISIER.TRFC is set to 1.

Writing this bit to 1 clears this bit.

The table below shows that Transfer Complete (TRFC) has a higher priority than Data Timeout Error (DATTEO). If both bits are set to 1, execution of a command can be considered to be completed.

TRFC	DATTEO	Meaning of the status
0	0	Interrupted by another factor
0	1	Timeout occurred during transfer
1	Don't Care	Command execution complete

0: Command execution is not complete.

1: Command execution is complete.

- **BLKGE: Block Gap Event**

If the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR is set to 1, this bit is set when either a read or a write transaction is stopped at a block gap. If STPBGR is not set to 1, this bit is not set to 1.

In the case of a Read transaction:

This bit is set at the falling edge of the DAT Line Active (DLACT) status (when the transaction is stopped at SD bus timing). The Read Wait must be supported in order to use this function. Refer to section "Read Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" about the detailed timing.

In the case of a Write transaction:

This bit is set at the falling edge of the Write Transfer Active (WTACT) status (after getting the CRC status at SD bus timing). Refer to section "Write Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" for more details on the sequence of events.

This bit can only be set to 1 if SDMMC_NISTER.BLKGE is set to 1. An interrupt can only be generated if SDMMC_NISIER.BLKGE is set to 1.

Writing this bit to 1 clears this bit.

0: No block gap event

1: Transaction stopped at block gap

- **DMAINT: DMA Interrupt**

This status is set if the SDMMC detects the Host SDMA Buffer boundary during transfer. Refer to SDMA Buffer Boundary (BOUNDARY) in SDMMC_BSR.

In case of ADMA, by setting the "int" field in the descriptor table, the SDMMC raises this status flag when the descriptor line is completed. This status flag does not rise after Transfer Complete (TRFC).

This bit can only be set to 1 if SDMMC_NISTER.DMAINT is set to 1. An interrupt can only be generated if SDMMC_NISIER.DMAINT is set to 1.

Writing this bit to 1 clears this bit.

0: No DMA Interrupt

1: DMA Interrupt

- **BWRRDY: Buffer Write Ready**

This status is set to 1 if the Buffer Write Enable (BUFWREN) changes from 0 to 1. Refer to BUFWREN in SDMMC_PSR.

This bit can only be set to 1 if SDMMC_NISTER.BWRRDY is set to 1. An interrupt can only be generated if SDMMC_NISIER.BWRRDY is set to 1.

Writing this bit to 1 clears this bit.

0: Not ready to write buffer

1: Ready to write buffer

- **BRDRDY: Buffer Read Ready**

This status is set to 1 if the Buffer Read Enable (BUFRDEN) changes from 0 to 1. Refer to BUFRDEN in SDMMC_PSR. While processing the tuning procedure (Execute Tuning (EXTUN) in SDMMC_HC2R is set to 1), BRDRDY is set to 1 for every CMD19 execution.

This bit can only be set to 1 if SDMMC_NISTER.BRDRDY is set to 1. An interrupt can only be generated if SDMMC_NISIER.BRDRDY is set to 1.

Writing this bit to 1 clears this bit.

0: Not ready to read buffer

1: Ready to read buffer

- **CINS: Card Insertion**

This status is set if Card Inserted (CARDINS) in SDMMC_PSR changes from 0 to 1. When the user writes this bit to 1 to clear this status, the status of SDMMC_PSR.CARDINS must be confirmed because the card detect state may possibly be changed when the user clears this bit and no interrupt event can be generated.

This bit can only be set to 1 if SDMMC_NISTER.CINS is set to 1. An interrupt can only be generated if SDMMC_NISIER.CINS is set to 1.

Writing this bit to 1 clears this bit.

0: Card state unstable or card removed

1: Card inserted

- **CREM: Card Removal**

This status is set to 1 if Card Inserted (CARDINS) in SDMMC_PSR changes from 1 to 0. When the user writes this bit to 1 to clear this status, the status of SDMMC_PSR.CARDINS must be confirmed because the card detect state may possibly be changed when the user clears this bit and no interrupt event can be generated.

This bit can only be set to 1 if SDMMC_NISTER.CREM is set to 1. An interrupt can only be generated if SDMMC_NISIER.CREM is set to 1.

Writing this bit to 1 clears this bit.

0: Card state unstable or card inserted

1: Card removed

- **CINT: Card Interrupt**

Writing this bit to 1 does not clear this bit. It is cleared by resetting the SD card interrupt factor. In 1-bit mode, the SDMMC detects the Card Interrupt without SDCLK to support wakeup. In 4-bit mode, the Card Interrupt signal is sampled during the interrupt cycle, so there are some sample delays between the interrupt signal from the SD card and the interrupt to the system.

When this bit has been set to 1 and the user needs to start this interrupt service, Card Interrupt Status Enable (CINT) in SDMMC_NISTER may be set to 0 in order to clear the card interrupt statuses latched in the SDMMC and to stop driving

the interrupt signal to the system. After completion of the card interrupt service (it should reset interrupt factors in the SD card and the interrupt signal may not be asserted), set SDMMC_NISTER.CINT to 1 and start sampling the interrupt signal again.

Interrupt detected by DAT[1] is supported when there is one card per slot.

This bit can only be set to 1 if SDMMC_NISTER.CREM is set to 1. An interrupt can only be generated if SDMMC_NISIER.CREM is set to 1.

0: No card interrupt

1: Card interrupt

- **ERRINT: Error Interrupt**

If any of the bits in SDMMC_EISTR are set, then this bit is set. Therefore, the user can efficiently test for an error by checking this bit first. This bit is read-only.

0: No error

1: Error

48.13.20 SDMMC Normal Interrupt Status Register (eMMC)

Name: SDMMC_NISTR (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
ERRINT	BOOTAR	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	BRDRDY	BWRRDY	DMAINT	BLKGE	TRFC	CMDC

- **CMDC: Command Complete**

This bit is set when getting the end bit of the command response. Auto CMD12 and Auto CMD23 consist of two responses. Command Complete is not generated by the response of CMD12 or CMD23, but it is generated by the response of a read/write command. Refer to CMRINHC in SDMMC_PSR for details on how to control this bit.

This bit can only be set to 1 if SDMMC_NISTER.CMDC is set to 1. An interrupt can only be generated if SDMMC_NISIER.CMDC is set to 1.

Writing this bit to 1 clears this bit.

The table below shows that Command Timeout Error (CMDTEO) has a higher priority than Command Complete (CMDC). If both bits are set to 1, it can be considered that the response was not received correctly.

CMDC	CMDTEO	Meaning of the status
0	0	Interrupted by another factor
Don't care	1	Response not received within 64 SDCLK cycles
1	0	Response received

0: No command complete

1: Command complete

- **TRFC: Transfer Complete**

This bit is set when a read/write transfer and a command with Busy is completed.

In the case of a Read Transaction:

This bit is set at the falling edge of the Read Transfer Active Status. The interrupt is generated in two cases. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the system). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR (after valid data has been read to the system). Refer to section "Read Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" for more details on the sequence of events.

In the case of a Write Transaction:

This bit is set at the falling edge of the DAT Line Active (DLACT) status. This interrupt is generated in two cases. The first is when the last data is written to the card as specified by the data length and the Busy signal is released. The second is when data transfers are stopped at the block gap by setting Stop At Block Gap Request (STPBGR) in SDMMC_BGCR and data transfers are completed. (After valid data is written to the card and the Busy signal is released). Refer to section "Write Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" for more details on the sequence of events.

In the case of command with Busy:

This bit is set when Busy is de-asserted. Refer to DAT Line Active (DLACT) and Command Inhibit (DAT) (CMDINHD) in SDMMC_PSR.

This bit can only be set to 1 if SDMMC_NISTER.TRFC is set to 1. An interrupt can only be generated if SDMMC_NISIER.TRFC is set to 1.

Writing this bit to 1 clears this bit.

The table below shows that Transfer Complete (TRFC) has a higher priority than Data Timeout Error (DATTEO). If both bits are set to 1, execution of a command can be considered to be completed.

TRFC	DATTEO	Meaning of the status
0	0	Interrupted by another factor
0	1	Timeout occurred during transfer
1	Don't Care	Command execution complete

0: Command execution is not complete.

1: Command execution is complete.

- **BLKGE: Block Gap Event**

If the Stop At Block Gap Request (STPBGR) in SDMMC_BGCR is set to 1, this bit is set when either a read or a write transaction is stopped at a block gap. If STPBGR is not set to 1, this bit is not set to 1.

In the case of a Read transaction:

This bit is set at the falling edge of the DAT Line Active (DLACT) status (when the transaction is stopped at SD bus timing). The Read Wait must be supported in order to use this function. Refer to section "Read Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" about the detailed timing.

In the case of a Write transaction:

This bit is set at the falling edge of the Write Transfer Active (WTACT) status (after getting the CRC status at SD bus timing). Refer to section "Write Transaction Wait / Continue Timing" in the "SD Host Controller Simplified Specification V3.00" for more details on the sequence of events.

This bit can only be set to 1 if SDMMC_NISTER.BLKGE is set to 1. An interrupt can only be generated if SDMMC_NISIER.BLKGE is set to 1.

Writing this bit to 1 clears this bit.

0: No block gap event

1: Transaction stopped at block gap

- **DMAINT: DMA Interrupt**

This status is set if the SDMMC detects the Host SDMA Buffer boundary during transfer. Refer to SDMA Buffer Boundary (BOUNDARY) in SDMMC_BSR.

In case of ADMA, by setting "int" field in the descriptor table, the SDMMC raises this status flag when the descriptor line is completed. This status flag does not rise after the Transfer Complete (TRFC).

This bit can only be set to 1 if SDMMC_NISTER.DMAINT is set to 1. An interrupt can only be generated if SDMMC_NISIER.DMAINT is set to 1.

Writing this bit to 1 clears this bit.

0: No DMA interrupt

1: DMA interrupt

- **BWRRDY: Buffer Write Ready**

This status is set to 1 if Buffer Write Enable (BUFWREN) changes from 0 to 1. Refer to Buffer Write Enable (BUFWREN) in SDMMC_PSR.

This bit can only be set to 1 if SDMMC_NISTER.BWRRDY is set to 1. An interrupt can only be generated if SDMMC_NISIER.BWRRDY is set to 1.

Writing this bit to 1 clears this bit.

0: Not ready to write buffer

1: Ready to write buffer

- **BRDRDY: Buffer Read Ready**

This status is set to 1 if Buffer Read Enable (BUFRDEN) changes from 0 to 1. Refer to Buffer Read Enable (BUFRDEN) in SDMMC_PSR. While processing the tuning procedure (Execute Tuning (EXTUN) in SDMMC_HC2R is set to 1), BRDRDY is set to 1 for every CMD19 execution.

This bit can only be set to 1 if SDMMC_NISTER.BRDRDY is set to 1. An interrupt can only be generated if SDMMC_NISIER.BRDRDY is set to 1.

Writing this bit to 1 clears this bit.

0: Not ready to read buffer

1: Ready to read buffer

- **BOOTAR: Boot Acknowledge Received**

This bit is set to 1 when the SDMMC received a Boot Acknowledge pattern from the e.MMC.

This bit can only be set to 1 if SDMMC_NISTER.BOOTAR is set to 1. An interrupt can only be generated if SDMMC_NISIER.BOOTAR is set to 1.

Writing this bit to 1 clears this bit.

0: Boot Acknowledge pattern not received.

1: Boot Acknowledge pattern received.

- **ERRINT: Error Interrupt**

If any of the bits in SDMMC_EISTR are set, then this bit is set. Therefore, the user can efficiently test for an error by checking this bit first. This bit is read only.

0: No error

1: Error

48.13.21 SDMMC Error Interrupt Status Register (SD_SDIO)

Name: SDMMC_EISTR (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Command Timeout Error**

This bit is set to 1 only if no response is returned within 64 SDCLK cycles from the end bit of the command. If the SDMMC detects a CMD line conflict, in which case Command CRC Error (CMDCRC) is also set to 1 as shown in Table 48-5, this bit is set without waiting for 64 SDCLK cycles because the command is aborted by the SDMMC.

This bit can only be set to 1 if SDMMC_EISTER.CMDTEO is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDTEO is set to 1.

Writing this bit to 1 clears this bit.

Table 48-5. Relations between CMDCRC and CMDTEO

CMDCRC	CMDTEO	Types of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

- **CMDCRC: Command CRC Error**

The Command CRC Error is generated in two cases.

If a response is returned and the Command Timeout Error (CMDTEO) is set to 0 (indicating no command timeout), this bit is set to 1 when detecting a CRC error in the command response.

The SDMMC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the SDMMC drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SDCLK edge, then the SDMMC aborts the command (stops driving the CMD line) and sets this bit to 1. CMDTEO is also set to 1 to indicate a CMD line conflict (refer to Table 48-5).

This bit can only be set to 1 if SDMMC_EISTER.CMDCRC is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDCRC is set to 1.

Writing this bit to 1 clears this bit.

- **CMDEND: Command End Bit Error**

This bit is set to 1 when detecting that the end bit of a command response is 0.

This bit can only be set to 1 if SDMMC_EISTER.CMDEND is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDEND is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **CMDIDX: Command Index Error**

This bit is set to 1 if a Command Index error occurs in the command response.

This bit can only be set to 1 if SDMMC_EISTER.CMDIDX is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDIDX is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **DATTEO: Data Timeout Error**

This bit is set to 1 when detecting one of following timeout conditions.

  – Busy timeout for R1b, R5b response type (see “Physical Layer Simplified Specification V3.01” and “SDIO Simplified Specification V3.00”).

  – Busy timeout after Write CRC status.

  – Write CRC Status timeout.

  – Read data timeout

This bit can only be set to 1 if SDMMC_EISTER.DATTEO is set to 1. An interrupt can only be generated if SDMMC_EISIER.DATTEO is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **DATCRC: Data CRC error**

This bit is set to 1 when detecting a CRC error when transferring read data which uses the DAT line or when detecting that the Write CRC Status has a value other than “010”.

This bit can only be set to 1 if SDMMC_EISTER.DATCRC is set to 1. An interrupt can only be generated if SDMMC_EISIER.DATCRC is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **DATEND: Data End Bit Error**

This bit is set to 1 either when detecting 0 at the end bit position of read data which uses the DAT line or at the end bit position of the CRC Status.

This bit can only be set to 1 if SDMMC_EISTER.DATEND is set to 1. An interrupt can only be generated if SDMMC_EISIER.DATEND is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **CURLIM: Current Limit Error**

By setting SD Bus Power (SDBPWR) in SDMMC_PSR, the SDMMC is requested to supply power for the SD Bus. The SDMMC is protected from an illegal card by stopping power supply to the card, in which case this bit indicates a failure status. Reading 1 means the SDMMC is not supplying power to the card due to some failure. Reading 0 means that the SDMMC is supplying power and no error has occurred. The SDMMC may require some sampling time to detect the current limit.

This bit can only be set to 1 if SDMMC_EISTER.CURLIM is set to 1. An interrupt can only be generated if SDMMC_EISIER.CURLIM is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **ACMD: Auto CMD Error**

Auto CMD12 and Auto CMD23 use this error status. This bit is set to 1 when detecting that one of the 0 to 4 bits in SDMMC_AESR (SDMMC_ACESR[4:0]) has changed from 0 to 1. In the case of Auto CMD12, this bit is set to 1, not only when errors occur in Auto CMD12 but also when auto CMD12 is not executed due to the previous command error.

This bit can only be set to 1 if SDMMC_EISTER.ACMD is set to 1. An interrupt can only be generated if SDMMC_EISIER.ACMD is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **ADMA: ADMA Error**

This bit is set to 1 when the SDMMC detects errors during an ADMA-based data transfer. The state of the ADMA at an error occurrence is saved in SDMMC_AESR.

In addition, the SDMMC raises this status flag when it detects some invalid description data (Valid = 0) at the ST_FDS state (refer to section “Advanced DMA” in the “SD Host Controller Simplified Specification V3.00”). ADMA Error Status (ERRST) in SDMMC_AESR indicates that an error occurred in ST_FDS state. The user may find that the Valid bit is not set at the error descriptor.

This bit can only be set to 1 if SDMMC_EISTER.ADMA is set to 1. An interrupt can only be generated if SDMMC_EISIER.ADMA is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

48.13.22 SDMMC Error Interrupt Status Register (eMMC)

Name: SDMMC_EISTR (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	BOOTAE	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Command Timeout Error**

This bit is set to 1 only if no response is returned within 64 SDCLK cycles from the end bit of the command. If the SDMMC detects a CMD line conflict, in which case Command CRC Error (CMDCRC) is also set to 1 as shown in Table 48-5, this bit is set without waiting for 64 SDCLK cycles because the command is aborted by the SDMMC.

This bit can only be set to 1 if SDMMC_EISTER.CMDTEO is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDTEO is set to 1.

Writing this bit to 1 clears this bit.

Table 48-6. Relations between CMDCRC and CMDTEO

CMDCRC	CMDTEO	Types of error
0	0	No error
0	1	Response timeout error
1	0	Response CRC error
1	1	CMD line conflict

- **CMDCRC: Command CRC Error**

The Command CRC Error is generated in two cases.

If a response is returned and Command Timeout Error (CMDTEO) is set to 0 (indicating no command timeout), this bit is set to 1 when detecting a CRC error in the command response.

The SDMMC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the SDMMC drives the CMD line to 1 level, but detects 0 level on the CMD line at the next SDCLK edge, then the SDMMC aborts the command (stops driving the CMD line) and sets this bit to 1. CMDTEO is also set to 1 to indicate a CMD line conflict (refer to Table 48-5).

This bit can only be set to 1 if SDMMC_EISTER.CMDCRC is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDCRC is set to 1.

Writing this bit to 1 clears this bit.

- **CMDEND: Command End Bit Error**

This bit is set to 1 when detecting that the end bit of a command response is 0.

This bit can only be set to 1 if SDMMC_EISTER.CMDEND is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDEND is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **CMDIDX: Command Index Error**

This bit is set to 1 if a Command Index error occurs in the command response.

This bit can only be set to 1 if SDMMC_EISTER.CMDIDX is set to 1. An interrupt can only be generated if SDMMC_EISIER.CMDIDX is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **DATTEO: Data Timeout error**

This bit is set to 1 when detecting one of following timeout conditions.

  – Busy timeout for R1b, R5b response type (see “Physical Layer Simplified Specification V3.01” and “SDIO Simplified Specification V3.00”).
  – Busy timeout after Write CRC Status.
  – Write CRC Status timeout.
  – Read data timeout

This bit can only be set to 1 if SDMMC_EISTER.DATTEO is set to 1. An interrupt can only be generated if SDMMC_EISIER.DATTEO is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **DATCRC: Data CRC Error**

This bit is set to 1 when detecting a CRC error during a transfer of read data which uses the DAT line or when detecting that the Write CRC Status has a value other than “010”.

This bit can only be set to 1 if SDMMC_EISTER.DATCRC is set to 1. An interrupt can only be generated if SDMMC_EISIER. DATCRC is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **DATEND: Data End Bit Error**

This bit is set to 1 either when detecting 0 at the end bit position of read data which uses the DAT line or at the end bit position of the CRC Status.

This bit can only be set to 1 if SDMMC_EISTER.DATEND is set to 1. An interrupt can only be generated if SDMMC_EISIER.DATEND is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **CURLIM: Current Limit Error**

By setting SD Bus Power (SDBPWR) in SDMMC_PSR, the SDMMC is requested to supply power for the SD Bus. The SDMMC is protected from an illegal card by stopping power supply to the card, in which case this bit indicates a failure status. Reading 1 means the SDMMC is not supplying power to the card due to some failure. Reading 0 means that the SDMMC is supplying power and no error has occurred. The SDMMC may require some sampling time to detect the current limit.

This bit can only be set to 1 if SDMMC_EISTER.CURLIM is set to 1. An interrupt can only be generated if SDMMC_EISIER.CURLIM is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **ACMD: Auto CMD Error**

Auto CMD12 and Auto CMD23 use this error status. This bit is set to 1 when detecting that one of the 0 to 4 bits in SDMMC_AESR (SDMMC_ACESR[4:0]) has changed from 0 to 1. In the case of Auto CMD12, this bit is set to 1, not only when errors occur in Auto CMD12, but also when Auto CMD12 is not executed due to the previous command error.

This bit can only be set to 1 if SDMMC_EISTER.ACMD is set to 1. An interrupt can only be generated if SDMMC_EISIER.ACMD is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **ADMA: ADMA Error**

This bit is set to 1 when the SDMMC detects errors during an ADMA-based data transfer. The state of the ADMA at an error occurrence is saved in SDMMC_AESR.

In addition, the SDMMC raises this status flag when it detects some invalid description data (Valid = 0) at the ST_FDS state (refer to section “Advanced DMA” in the “SD Host Controller Simplified Specification V3.00”). ADMA Error Status (ERRST) in SDMMC_AESR indicates that an error occured in ST_FDS state. The user may find that the Valid bit is not set at the error descriptor.

This bit can only be set to 1 if SDMMC_EISTER.ADMA is set to 1. An interrupt can only be generated if SDMMC_EISIER.ADMA is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

- **BOOTAE: Boot Acknowledge Error**

This bit is set to 1 when detecting that the e.MMC Boot Acknowledge Status has a value other than “010”.

This bit can only be set to 1 if SDMMC_EISTER.BOOTAE is set to 1. An interrupt can only be generated if SDMMC_EISIER.BOOTAE is set to 1.

Writing this bit to 1 clears this bit.

0: No error

1: Error

48.13.23 SDMMC Normal Interrupt Status Enable Register (SD_SDIO)

Name: SDMMC_NISTER (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CINT
7	6	5	4	3	2	1	0
CREM	CINS	BRDRDY	BWRRDY	DMAINT	BLKGE	TRFC	CMDC

- **CMDC: Command Complete Status Enable**

0 (MASKED): The CMDC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CMDC status flag in SDMMC_NISTR is enabled.

- **TRFC: Transfer Complete Status Enable**

0 (MASKED): The TRFC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The TRFC status flag in SDMMC_NISTR is enabled.

- **BLKGE: Block Gap Event Status Enable**

0 (MASKED): The BLKGE status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BLKGE status flag in SDMMC_NISTR is enabled.

- **DMAINT: DMA Interrupt Status Enable**

0 (MASKED): The DMAINT status flag in SDMMC_NISTR is masked.

1 (ENABLED): The DMAINT status flag in SDMMC_NISTR is enabled.

- **BWRRDY: Buffer Write Ready Status Enable**

0 (MASKED): The BWRRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BWRRDY status flag in SDMMC_NISTR is enabled.

- **BRDRDY: Buffer Read Ready Status Enable**

0 (MASKED): The BRDRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BRDRDY status flag in SDMMC_NISTR is enabled.

- **CINS: Card Insertion Status Enable**

0 (MASKED): The CINS status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CINS status flag in SDMMC_NISTR is enabled.

- **CREM: Card Removal Status Enable**

0 (MASKED): The CREM status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CREM status flag in SDMMC_NISTR is enabled.

- **CINT: Card Interrupt Status Enable**

If this bit is set to 0, the SDMMC clears interrupt requests to the system. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The user may clear this bit before servicing the Card Interrupt and may set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

0 (MASKED): The CINT status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CINT status flag in SDMMC_NISTR is enabled.

48.13.24 SDMMC Normal Interrupt Status Enable Register (eMMC)

Name: SDMMC_NISTER (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
–	BOOTAR	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	BRDRDY	BWRRDY	DMAINT	BLKGE	TRFC	CMDC

- **CMDC: Command Complete Status Enable**

0 (MASKED): The CMDC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The CMDC status flag in SDMMC_NISTR is enabled.

- **TRFC: Transfer Complete Status Enable**

0 (MASKED): The TRFC status flag in SDMMC_NISTR is masked.

1 (ENABLED): The TRFC status flag in SDMMC_NISTR is enabled.

- **BLKGE: Block Gap Event Status Enable**

0 (MASKED): The BLKGE status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BLKGE status flag in SDMMC_NISTR is enabled.

- **DMAINT: DMA Interrupt Status Enable**

0 (MASKED): The DMAINT status flag in SDMMC_NISTR is masked.

1 (ENABLED): The DMAINT status flag in SDMMC_NISTR is enabled.

- **BWRRDY: Buffer Write Ready Status Enable**

0 (MASKED): The BWRRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BWRRDY status flag in SDMMC_NISTR is enabled.

- **BRDRDY: Buffer Read Ready Status Enable**

0 (MASKED): The BRDRDY status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BRDRDY status flag in SDMMC_NISTR is enabled.

- **BOOTAR: Boot Acknowledge Received Status Enable**

0 (MASKED): The BOOTAR status flag in SDMMC_NISTR is masked.

1 (ENABLED): The BOOTAR status flag in SDMMC_NISTR is enabled.

48.13.25 SDMMC Error Interrupt Status Enable Register (SD_SDIO)

Name: SDMMC_EISTER (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Command Timeout Error Status Enable**

0 (MASKED): The CMDTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDTEO status flag in SDMMC_EISTR is enabled.

- **CMDCRC: Command CRC Error Status Enable**

0 (MASKED): The CMDCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDCRC status flag in SDMMC_EISTR is enabled.

- **CMDEND: Command End Bit Error Status Enable**

0 (MASKED): The CMDEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDEND status flag in SDMMC_EISTR is enabled.

- **CMDIDX: Command Index Error Status Enable**

0 (MASKED): The CMDIDX status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDIDX status flag in SDMMC_EISTR is enabled.

- **DATTEO: Data Timeout Error Status Enable**

0 (MASKED): The DATTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATTEO status flag in SDMMC_EISTR is enabled.

- **DATCRC: Data CRC Error Status Enable**

0 (MASKED): The DATCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATCRC status flag in SDMMC_EISTR is enabled.

- **DATEND: Data End Bit Error Status Enable**

0 (MASKED): The DATEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATEND status flag in SDMMC_EISTR is enabled.

- **CURLIM: Current Limit Error Status Enable**

0 (MASKED): The CURLIM status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CURLIM status flag in SDMMC_EISTR is enabled.

- **ACMD: Auto CMD Error Status Enable**

0 (MASKED): The ACMD status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ACMD status flag in SDMMC_EISTR is enabled.

- **ADMA: ADMA Error Status Enable**

0 (MASKED): The ADMA status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ADMA status flag in SDMMC_EISTR is enabled.

48.13.26 SDMMC Error Interrupt Status Enable Register (eMMC)

Name: SDMMC_EISTER (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	BOOTAE	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Command Timeout Error Status Enable**

0 (MASKED): The CMDTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDTEO status flag in SDMMC_EISTR is enabled.

- **CMDCRC: Command CRC Error Status Enable**

0 (MASKED): The CMDCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDCRC status flag in SDMMC_EISTR is enabled.

- **CMDEND: Command End Bit Error Status Enable**

0 (MASKED): The CMDEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDEND status flag in SDMMC_EISTR is enabled.

- **CMDIDX: Command Index Error Status Enable**

0 (MASKED): The CMDIDX status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CMDIDX status flag in SDMMC_EISTR is enabled.

- **DATTEO: Data Timeout Error Status Enable**

0 (MASKED): The DATTEO status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATTEO status flag in SDMMC_EISTR is enabled.

- **DATCRC: Data CRC Error Status Enable**

0 (MASKED): The DATCRC status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATCRC status flag in SDMMC_EISTR is enabled.

- **DATEND: Data End Bit Error Status Enable**

0 (MASKED): The DATEND status flag in SDMMC_EISTR is masked.

1 (ENABLED): The DATEND status flag in SDMMC_EISTR is enabled.

- **CURLIM: Current Limit Error Status Enable**

0 (MASKED): The CURLIM status flag in SDMMC_EISTR is masked.

1 (ENABLED): The CURLIM status flag in SDMMC_EISTR is enabled.

- **ACMD: Auto CMD Error Status Enable**

0 (MASKED): The ACMD status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ACMD status flag in SDMMC_EISTR is enabled.

- **ADMA: ADMA Error Status Enable**

0 (MASKED): The ADMA status flag in SDMMC_EISTR is masked.

1 (ENABLED): The ADMA status flag in SDMMC_EISTR is enabled.

- **BOOTAE: Boot Acknowledge Error Status Enable**

0 (MASKED): The BOOTAE status flag in SDMMC_EISTR is masked.

1 (ENABLED): The BOOTAE status flag in SDMMC_EISTR is enabled.

48.13.27 SDMMC Normal Interrupt Signal Enable Register (SD_SDIO)

Name: SDMMC_NISIER (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	CINT
7	6	5	4	3	2	1	0
CREM	CINS	BRDRDY	BWRRDY	DMAINT	BLKGE	TRFC	CMDC

- **CMDC: Command Complete Signal Enable**

0 (MASKED): No interrupt is generated when the CMDC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CMDC status rises in SDMMC_NISTR.

- **TRFC: Transfer Complete Signal Enable**

0 (MASKED): No interrupt is generated when the TRFC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the TRFC status rises in SDMMC_NISTR.

- **BLKGE: Block Gap Event Signal Enable**

0 (MASKED): No interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

- **DMAINT: DMA Interrupt Signal Enable**

0 (MASKED): No interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

- **BWRRDY: Buffer Write Ready Signal Enable**

0 (MASKED): No interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

- **BRDRDY: Buffer Read Ready Signal Enable**

0 (MASKED): No interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

- **CINS: Card Insertion Signal Enable**

0 (MASKED): No interrupt is generated when the CINS status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CINS status rises in SDMMC_NISTR.

- **CREM: Card Removal Signal Enable**

0 (MASKED): No interrupt is generated when the CREM status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CREM status rises in SDMMC_NISTR.

- **CINT: Card Interrupt Signal Enable**

0 (MASKED): No interrupt is generated when the CINT status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CINT status rises in SDMMC_NISTR.

48.13.28 SDMMC Normal Interrupt Signal Enable Register (eMMC)

Name: SDMMC_NISIER (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
–	BOOTAR	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	BRDRDY	BWRRDY	DMAINT	BLKGE	TRFC	CMDC

- **CMDC: Command Complete Signal Enable**

0 (MASKED): No interrupt is generated when the CMDC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the CMDC status rises in SDMMC_NISTR.

- **TRFC: Transfer Complete Signal Enable**

0 (MASKED): No interrupt is generated when the TRFC status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the TRFC status rises in SDMMC_NISTR.

- **BLKGE: Block Gap Event Signal Enable**

0 (MASKED): No interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BLKGE status rises in SDMMC_NISTR.

- **DMAINT: DMA Interrupt Signal Enable**

0 (MASKED): No interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the DMAINT status rises in SDMMC_NISTR.

- **BWRRDY: Buffer Write Ready Signal Enable**

0 (MASKED): No interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BWRRDY status rises in SDMMC_NISTR.

- **BRDRDY: Buffer Read Ready Signal Enable**

0 (MASKED): No interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BRDRDY status rises in SDMMC_NISTR.

- **BOOTAR: Boot Acknowledge Received Signal Enable**

0 (MASKED): No interrupt is generated when the BOOTAR status rises in SDMMC_NISTR.

1 (ENABLED): An interrupt is generated when the BOOTAR status rises in SDMMC_NISTR.

48.13.29 SDMMC Error Interrupt Signal Enable Register (SD_SDIO)

Name: SDMMC_EISIER (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Command Timeout Error Signal Enable**

0 (MASKED): No interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

- **CMDCRC: Command CRC Error Signal Enable**

0 (MASKED): No interrupt is generated when the CDMCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDCRC status rises in SDMMC_EISTR.

- **CMDEND: Command End Bit Error Signal Enable**

0 (MASKED): No interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

- **CMDIDX: Command Index Error Signal Enable**

0 (MASKED): No interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

- **DATTEO: Data Timeout Error Signal Enable**

0 (MASKED): No interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

- **DATCRC: Data CRC Error Signal Enable**

0 (MASKED): No interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

- **DATEND: Data End Bit Error Signal Enable**

0 (MASKED): No interrupt is generated when the DATEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATEND status rises in SDMMC_EISTR.

- **CURLIM: Current Limit Error Signal Enable**

0 (MASKED): No interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

- **ACMD: Auto CMD Error Signal Enable**

0 (MASKED): No interrupt is generated when the ACMD status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ACMD status rises in SDMMC_EISTR.

- **ADMA: ADMA Error Signal Enable**

0 (MASKED): No interrupt is generated when the ADMA status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ADMA status rises in SDMMC_EISTR.

48.13.30 SDMMC Error Interrupt Signal Enable Register (eMMC)

Name: SDMMC_EISIER (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	BOOTAE	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Command Timeout Error Signal Enable**

0 (MASKED): No interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDTEO status rises in SDMMC_EISTR.

- **CMDCRC: Command CRC Error Signal Enable**

0 (MASKED): No interrupt is generated when the CDMCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDCRC status rises in SDMMC_EISTR.

- **CMDEND: Command End Bit Error Signal Enable**

0 (MASKED): No interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDEND status rises in SDMMC_EISTR.

- **CMDIDX: Command Index Error Signal Enable**

0 (MASKED): No interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CMDIDX status rises in SDMMC_EISTR.

- **DATTEO: Data Timeout Error Signal Enable**

0 (MASKED): No interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATTEO status rises in SDMMC_EISTR.

- **DATCRC: Data CRC Error Signal Enable**

0 (MASKED): No interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATCRC status rises in SDMMC_EISTR.

- **DATEND: Data End Bit Error Signal Enable**

0 (MASKED): No interrupt is generated when the DATEND status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the DATEND status rises in SDMMC_EISTR.

- **CURLIM: Current Limit Error Signal Enable**

0 (MASKED): No interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the CURLIM status rises in SDMMC_EISTR.

- **ACMD: Auto CMD Error Signal Enable**

0 (MASKED): No interrupt is generated when the ACMD status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ACMD status rises in SDMMC_EISTR.

- **ADMA: ADMA Error Signal Enable**

0 (MASKED): No interrupt is generated when the ADMA status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the ADMA status rises in SDMMC_EISTR.

- **BOOTAE: Boot Acknowledge Error Signal Enable**

0 (MASKED): No interrupt is generated when the BOOTAE status rises in SDMMC_EISTR.

1 (ENABLED): An interrupt is generated when the BOOTAE status rises in SDMMC_EISTR.

48.13.31 SDMMC Auto CMD Error Status Register

Name: SDMMC_ACESR

Access: Read/Write

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
CMDNI	–	–	ACMDIDX	ACMDEND	ACMDCRC	ACMDTEO	ACMD12NE

- **ACMD12NE: Auto CMD12 Not Executed**

If a memory multiple block data transfer is not started due to a command error, this bit is not set to 1 because it is not necessary to issue Auto CMD12. Setting this bit to 1 means the SDMMC cannot issue Auto CMD12 to stop a memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits (SDMMC_ACESR[4:1]) are meaningless.

This bit is set to 0 when an Auto CMD error is generated by Auto CMD23.

0: No error

1: Error

- **ACMDTEO: Auto CMD Timeout Error**

This bit is set to 1 if no response is returned within 64 SDCLK cycles from the end bit of the command. If this bit is set to 1, the other error status bits (SDMMC_ACESR[4:2]) are meaningless.

Table 48-7. Relations between ACMDCRC and ACMDTEO

ACMDCRC	ACMDTEO	Types of error
0	0	No error
0	1	Response Timeout error
1	0	Response CRC error
1	1	CMD line conflict

- **ACMDCRC: Auto CMD CRC Error**

This bit is set to 1 when detecting a CRC error in the command response (refer to Table 48-7 for more details).

- **ACMDEND: Auto CMD End Bit Error**

This bit is set to 1 when detecting that the end bit of the command response is 0.

0: No error

1: Error

- **ACMDIDX: Auto CMD Index Error**

This bit is set to 1 when the Command Index error occurs in response to a command.

0: No error

1: Error

- **CMDNI: Command Not Issued by Auto CMD12 Error**

Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 error (SDMMC_ACESR[4:1]). This bit is set to 0 when Auto CMD Error is generated by Auto CMD23.

0: No error

1: Error

48.13.32 SDMMC Host Control 2 Register (SD_SDIO)

Name: SDMMC_HC2R (SD_SDIO)

Access: Read/Write

15	14	13	12	11	10	9	8
PVALEN	ASINTEN	–	–	–	–	–	–

7	6	5	4	3	2	1	0
SLCKSEL	EXTUN	DRVSEL		VS18EN	UHSMS		

- **UHSMS: UHS Mode Select**

This field is used to select one of the UHS-I modes and is effective when 1.8V Signal Enable (VS18EN) is set to 1.

If Preset Value Enable is set to 1, the SDMMC sets SDCLK Frequency Select (SDCLKFSEL), Clock Generator Select (CLKGSEL) in SDMMC_CCR and Driver Strength Select (DRVSEL) according to SDMMC_PVR. In this case, one of the preset value registers is selected by this field. The user needs to reset SD Clock Enable (SDCLKEN) before changing this field to avoid generating a clock glitch. After setting this field, the user sets SDCLKEN to 1 again.

Value	Name	Description
0	SDR12	UHS SDR12 Mode
1	SDR25	UHS SDR25 Mode
2	SDR50	UHS SDR50 Mode
3	SDR104	UHS SDR104 Mode
4	DDR50	UHS DDR50 Mode

Note: This field is effective only if SDMMC_MC1R.DDR is set to 0.

- **VS18EN: 1.8V Signaling Enable**

This bit controls the voltage regulator for the I/O cell. 3.3V is supplied to the card regardless of the signaling voltage.

Setting this bit from 0 to 1 starts changing the signal voltage from 3.3V to 1.8V. The 1.8V regulator output must be stable within 5 ms.

Clearing this bit from 1 to 0 starts changing the signal voltage from 1.8V to 3.3V. The 3.3V regulator output must be stable within 5 ms.

The user can set this bit to 1 when the SDMMC supports 1.8V signaling (one of the support bits is set to 1: SDR50SUP, SDR104SUP or DDR50SUP in SDMMC_CA1R) and the card or device supports UHS-I (S18A = 1. Refer to “Bus Switch Voltage Switch Sequence in the “Physical Layer Simplified Specification V3.01”).

0: 3.3V signaling

1: 1.8V signaling

- **DRVSEL: Driver Strength Select**

The SDMMC output driver in 1.8V signaling is selected by this bit. In 3.3V signaling, this field is not effective. This field can be set according to the Driver Type A, C and D support bits in SDMMC_CA1R.

This field depends on the setting of Preset Value Enable (PVALEN):

– PVALEN = 0: This field is set by the user.

– PVALEN = 1: This field is automatically set by a value specified in one of the SDMMC_PVRx.

Value	Name	Description
0	TYPEB	Driver Type B is selected (Default)
1	TYPEA	Driver Type A is selected
2	TYPEC	Driver Type C is selected
3	TYPED	Driver Type D is selected

- **EXTUN: Execute Tuning**

This bit is set to 1 to start the tuning procedure and is automatically cleared when the tuning procedure is completed. The result of tuning is indicated to Sampling Clock Select (SCLKSEL). The tuning procedure is aborted by writing 0. Refer to Figure 2.29 in the "SD Host Controller Simplified Specification V3.00".

0: Not tuned or tuning completed

1: Execute tuning

- **SLCKSEL: Sampling Clock Select**

The SDMMC uses this bit to select the sampling clock to receive CMD and DAT.

This bit is set by the tuning procedure and is valid after completion of tuning (when EXTUN is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning has failed.

Writing 1 to this bit is meaningless and ignored. A tuning circuit is reset by writing to 0. This bit can be cleared by setting EXTUN to 1. Once the tuning circuit is reset, it takes time to complete the tuning sequence. Therefore, the user should keep this bit to 1 to perform a retuning sequence to complete a retuning sequence in a short time. Changing this bit is not allowed while the SDMMC is receiving a response or a read data block. Refer to Figure 2.29 in the "SD Host Controller Simplified Specification V3.00".

0: The fixed clock is used to sample data.

1: The tuned clock is used to sample data.

- **ASINTEN: Asynchronous Interrupt Enable**

This bit can be set to 1 if a card support asynchronous interrupts and Asynchronous Interrupt Support (ASINTSUP) is set to 1 in SDMMC_CA0R. Asynchronous interrupt is effective when DAT[1] interrupt is used in 4-bit SD mode. If this bit is set to 1, the user can stop the SDCLK during the asynchronous interrupt period to save power. During this period, the SDMMC continues to deliver the Card Interrupt to the host when it is asserted by the card.

0: Disabled

1: Enabled

- **PVALEN: Preset Value Enable**

As the operating SDCLK frequency and I/O driver strength depend on the system implementation, it is difficult to determine these parameters in the standard host driver. When Preset Value Enable (PVALEN) is set to 1, automatic SDCLK frequency generation and driver strength selection are performed without considering system-specific conditions. This bit enables the functions defined in SDMMC_PVR.

if this bit is set to 0, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the user.

if this bit is set to 1, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the SDMMC as specified in SDMMC_PVR.

0: SDCLK and Driver strength are controlled by the user.

1: Automatic selection by Preset Value is enabled.

48.13.33 SDMMC Host Control 2 Register (eMMC)

Name: SDMMC_HC2R (eMMC)

Access: Read/Write

15	14	13	12	11	10	9	8
PVALEN	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
SLCKSEL	EXTUN	DRVSEL		HS200EN			

- **HS200EN: HS200 Mode Enable**

This field is used to select the e.MMC HS200 mode. When HS200EN is set to $B_{(hexa)}$, the HS200 mode is enabled. Any other value except 0 is forbidden when interfacing an e.MMC device.

If Preset Value Enable is set to 1, SDMMC sets SDCLK Frequency Select (SDCLKFSEL), Clock Generator Select (CLKGSEL) in SDMMC_CCR and Driver Strength Select (DRVSEL) according to SDMMC_PVR. In this case, one of the preset value registers is selected by this field. The user needs to reset SD Clock Enable (SDCLKEN) before changing this field to avoid generating a clock glitch. After setting this field, the user sets SDCLKEN to 1 again.

Note: This field is effective only if DDR in SDMMC_MC1R is set to 0.

- **DRVSEL: Driver Strength Select**

The SDMMC output driver in 1.8V signaling is selected by this bit. In 3.3V signaling, this field is not effective. This field can be set according to the Driver Type A, C and D support bits in SDMMC_CA1R.

This field depends on setting of Preset Value Enable (PVALEN):

– PVALEN = 0: This field is set by the user.

– PVALEN = 1: This field is automatically set by a value specified in one of the SDMMC_PVRx.

Value	Name	Description
0	TYPEB	Driver Type B is selected (Default)
1	TYPEA	Driver Type A is selected
2	TYPEC	Driver Type C is selected
3	TYPED	Driver Type D is selected

- **EXTUN: Execute Tuning**

This bit is set to 1 to start the tuning procedure and is automatically cleared when the tuning procedure is completed. The result of tuning is indicated to Sampling Clock Select (SCLKSEL). The tuning procedure is aborted by writing 0. Refer to Figure 2.29 in the "SD Host Controller Simplified Specification V3.00".

0: Not tuned or tuning completed

1: Execute tuning

- **SLCKSEL: Sampling Clock Select**

The SDMMC uses this bit to select the sampling clock to receive CMD and DAT.

This bit is set by the tuning procedure and is valid after completion of tuning (when EXTUN is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning has failed.

Writing 1 to this bit is meaningless and ignored. A tuning circuit is reset by writing to 0. This bit can be cleared by setting EXTUN to 1. Once the tuning circuit is reset, it takes time to complete a tuning sequence. Therefore, the user should keep this bit to 1 to perform a retuning sequence to complete a retuning sequence in a short time. Changing this bit is not

allowed while the SDMMC is receiving a response or a read data block. Refer to Figure 2.29 in the “SD Host Controller Simplified Specification V3.00”.

0: The fixed clock is used to sample data.

1: The tuned clock is used to sample data.

- **PVALEN: Preset Value Enable**

As the operating SDCLK frequency and I/O driver strength depend on the system implementation, it is difficult to determine these parameters in the standard host driver. When Preset Value Enable (PVALEN) is set to 1, automatic SDCLK frequency generation and driver strength selection are performed without considering system-specific conditions. This bit enables the functions defined in SDMMC_PVR.

If this bit is set to 0, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the user.

If this bit is set to 1, SDCLKFSEL, CLKGSEL in SDMMC_CCR and DRVSEL in SDMMC_HC2R are set by the SDMMC as specified in SDMMC_PVR.

0: SDCLK and Driver strength are controlled by the user.

1: Automatic selection by Preset Value is enabled.

48.13.34 SDMMC Capabilities 0 Register

Name: SDMMC_CA0R

Access: Read/Write

31	30	29	28	27	26	25	24
SLTYPE		ASINTSUP	SB64SUP	–	V18VSUP	V30VSUP	V33VSUP
23	22	21	20	19	18	17	16
SRSUP	SDMASUP	HSSUP	–	ADMA2SUP	ED8SUP	MAXBLKL	
15	14	13	12	11	10	9	8
BASECLKF							
7	6	5	4	3	2	1	0
TEOCLKU	–	TEOCLKF					

Note: The Capabilities 0 Register is not supposed to be written by the user. However, the user can modify preset values only if Capabilities Write Enable (CAPWREN) is set to 1 in SDMMC_CACR.

- **TEOCLKF: Timeout Clock Frequency**

This bit shows the timeout clock frequency (TEOCLK) used to detect Data Timeout Error.

If this field is set to 0, the user must get the information via another method.

The Timeout Clock Unit (TEOCLKU) defines the unit of this field's value.

– TEOCLKU = 0: $f_{TEOCLK} = TEOCLKF_{KHz}$

– TEOCLKU = 1: $f_{TEOCLK} = TEOCLKF_{MHz}$

- **TEOCLKU: Timeout Clock Unit**

This bit shows the unit of the base clock frequency used to detect Data Timeout Error.

0: KHz

1: MHz

- **BASECLKF: Base Clock Frequency**

This value indicates the frequency of the base clock (BASECLK). The user uses this value to calculate the clock divider value (refer to SDCLK Frequency Select (SDCLKFSEL) in SDMMC_CCR).

If this field is set to 0, the user must get the information via another method.

$f_{BASECLK} = BASECLKF_{MHz}$

- **MAXBLKL: Max Block Length**

This field indicates the maximum block size that the user can read and write to the buffer in the SDMMC. Three sizes can be defined, as shown below. It is noted that the transfer block length is always 512 bytes for SD Memory Cards regardless of this field.

Value	Name	Description
0	512	512 bytes
1	1024	1024 bytes
2	2048	2048 bytes
3	NONE	Reserved

- **ED8SUP: 8-Bit Support for Embedded Device**

This bit indicates whether the SDMMC is capable of using the 8-bit Bus Width mode.

0: 8-bit bus width not supported

1: 8-bit bus width supported

- **ADMA2SUP: ADMA2 Support**

This bit indicates whether the SDMMC is capable of using ADMA2.

0: ADMA2 not supported

1: ADMA2 supported

- **HSSUP: High Speed Support**

This bit indicates whether the SDMMC and the system support High Speed mode and they can supply SDCLK frequency from 25 MHz to 50 MHz.

0: High Speed not supported

1: High Speed supported

- **SDMASUP: SDMA Support**

This bit indicates whether the SDMMC is capable of using SDMA to transfer data between system memory and the SDMMC directly.

0: SDMA not supported

1: SDMA supported

- **SRSUP: Suspend/Resume Support**

This bit indicates whether the SDMMC supports the Suspend/Resume functionality. If this bit is set to 0, the user does not issue either Suspend or Resume commands because the Suspend and Resume mechanism (refer to “Suspend and Resume Mechanism” in the “SD Host Controller Simplified Specification V3.00”) is not supported.

0: Suspend/Resume not supported

1: Suspend/Resume supported

- **V33VSUP: Voltage Support 3.3V**

0: 3.3V Voltage supply not supported

1: 3.3V Voltage supply supported

- **V30VSUP: Voltage Support 3.0V**

0: 3.0V Voltage supply not supported

1: 3.0V Voltage supply supported

- **V18VSUP: Voltage Support 1.8V**

0: 1.8V Voltage supply not supported

1: 1.8V Voltage supply supported

- **SB64SUP: 64-Bit System Bus Support**

Reading this bit to 1 means that the SDMMC supports the 64-bit Address Descriptor mode and is connected to the 64-bit address system bus.

0: 64-bit address bus not supported

1: 64-bit address bus supported

- **ASINTSUP: Asynchronous Interrupt Support**

Refer to section “Asynchronous Interrupt” in the “SDIO Simplified Specification V3.00”.

0: Asynchronous interrupt not supported

1: Asynchronous interrupt supported

- **SLTYPE: Slot Type**

This field indicates usage of a slot by a specific system. An SDMMC control register set is defined per slot.

Embedded Slot for One Device means that only one non-removable device is connected to a bus slot.

The Standard Host Driver controls a removable card (SLTYPE = 0) or one embedded device (SLTYPE = 1) connected to an SD bus slot.

Value	Description
0	Removable Card Slot
1	Embedded Slot for One Device
2	Reserved
3	Reserved

48.13.35 SDMMC Capabilities 1 Register

Name: SDMMC_CA1R

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CLKMULT							
15	14	13	12	11	10	9	8
RTMOD		TSDR50	–	TCNTRT			
7	6	5	4	3	2	1	0
–	DRVDSUP	DRVCSUP	DRVASUP	–	DDR50SUP	SDR104SUP	SDR50SUP

Note: The Capabilities 1 Register is not supposed to be written by the user. However, the user can modify preset values only if Capabilities Write Enable (CAPWREN) is set to 1 in SDMMC_CACR.

- **SDR50SUP: SDR50 Support**

0: SDR50 mode is not supported.

1: SDR50 mode is supported.

- **SDR104SUP: SDR104 Support**

0: SDR104 mode is not supported.

1: SDR104 mode is supported.

- **DDR50SUP: DDR50 Support**

0: DDR50 mode is not supported.

1: DDR50 mode is supported.

- **DRVASUP: Driver Type A Support**

0: Driver type A is not supported.

1: Driver type A is supported.

- **DRVCSUP: Driver Type C Support**

0: Driver type C is not supported.

1: Driver type C is supported.

- **DRVDSUP: Driver Type D Support**

0: Driver type D is not supported.

1: Driver type D is supported.

- **TCNTRT: Timer Count For Retuning**

This field indicates an initial value of the Retuning Timer for Retuning Mode (RTMOD) 1 to 3. Reading this field at 0 means that the Retuning Timer is disabled. The Retuning Timer initial value ranges from 0 to 1024 seconds.

$t_{TIMER} = 2^{(TCNTRT - 1)}$Seconds

- **TSDR50: Use Tuning for SDR50**

If this bit is set to 1, the SDMMC requires tuning to operate SDR50 (tuning is always required to operate SDR104).

0: SDR50 does not require tuning.

1: SDR50 requires tuning.

- **RTMOD: Retuning Modes**

This field selects the retuning method and limits the maximum data length.

There are two retuning timings: Retuning Request (RTREQ) controlled by the SDMMC, and expiration of a Retuning timer controlled by the user. By receiving either timing, the user executes the retuning procedure just before a next command issue.

The maximum data length per read/write command is restricted so that retuning procedures can be inserted during data transfers.

Retuning Mode 1:

The SDMMC does not have any internal logic to detect when retuning needs to be performed. In this case, the user should maintain all retuning timings by using the Retuning Timer. To enable inserting the retuning procedure during data transfers, the data length per Read/Write command must be limited to 4 Mbytes.

Retuning Mode 2:

The SDMMC has the capability to indicate the retuning timing by Retuning Request (RTREQ) during data transfers. Then the data length per Read/Write command must be limited to 4 Mbytes.

During non-data transfer, retuning timing is determined by either Retuning Request or Retuning Timer. If Retuning Request is used, Retuning Timer should be disabled.

Retuning Mode 3:

The SDMMC has the capability to take care of the retuning during data transfer (Auto Retuning). Retuning Request is not generated during data transfers and there is no limitation to data length per Read/Write command.

During non-data transfer, retuning timing is determined either by Retuning Request or Retuning Timer. If Retuning Request is used, Retuning Timer should be disabled.

Value	Name	Description	Data Length
0	MODE1	Timer	4 Mbytes (Max)
1	MODE2	Timer and Retuning Request	4 Mbytes (Max)
2	MODE3	Auto Retuning (for transfer) Timer and Retuning Request	Any
3	–	Reserved	–

- **CLKMULT: Clock Multiplier**

This field indicates the multiplier factor between the Base Clock (BASECLK) used for the Divided Clock Mode and the Multiplied Clock (MULTCLK) used for the Programmable Clock mode (refer to SDMMC_CCR).

Reading this field to 0 means that the Programmable Clock mode is not supported.

$$f_{MULTCLK} = f_{BASECLK} \times (CLKMULT + 1)$$

48.13.36 SDMMC Maximum Current Capabilities Register

Name: SDMMC_MCCAR

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
MAXCUR18V							
15	14	13	12	11	10	9	8
MAXCUR30V							
7	6	5	4	3	2	1	0
MAXCUR33V							

- **MAXCUR33V: Maximum Current for 3.3V**

This field indicates the maximum current capability for 3.3V voltage. This value is meaningful only if V33VSUP is set to 1 in SDMMC_CA0R. Reading MAXCUR33V at 0 means that the user must get information via another method.

$$Imax_{mA} = 4 \times MAXCURR33V$$

- **MAXCUR30V: Maximum Current for 3.0V**

This field indicates the maximum current capability for 3.0V voltage. This value is meaningful only if V30VSUP is set to 1 in SDMMC_CA0R. Reading MAXCUR30V at 0 means that the user must get information via another method.

$$Imax_{mA} = 4 \times MAXCURR30V$$

- **MAXCUR18V: Maximum Current for 1.8V**

This field indicates the maximum current capability for 1.8V voltage. This value is meaningful only if V18VSUP is set to 1 in SDMMC_CA0R. Reading MAXCUR18V at 0 means that the user must get information via another method.

$$Imax_{mA} = 4 \times MAXCURR18V$$

48.13.37 SDMMC Force Event Register for Auto CMD Error Status

Name: SDMMC_FERACES

Access: Write-only

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
CMDNI	–	–	ACMDIDX	ACMDEND	ACMDCRC	ACMDTEO	ACMD12NE

- **ACMD12NE: Force Event for Auto CMD12 Not Executed**

For testing purposes, the user can write this bit to 1 to rise the ACMD12NE status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

- **ACMDTEO: Force Event for Auto CMD Timeout Error**

For testing purposes, the user can write this bit to 1 to rise the ACMDTEO status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

- **ACMDCRC: Force Event for Auto CMD CRC Error**

For testing purposes, the user can write this bit to 1 to rise the ACMDCRC status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

- **ACMDEND: Force Event for Auto CMD End Bit Error**

For testing purposes, the user can write this bit to 1 to rise the ACMDEND status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

- **ACMDIDX: Force Event for Auto CMD Index Error**

For testing purposes, the user can write this bit to 1 to rise the ACMDIDX status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

- **CMDNI: Force Event for Command Not Issued by Auto CMD12 Error**

For testing purposes, the user can write this bit to 1 to rise the CMDNI status flag in SDMMC_ACESR.

Writing this bit to 0 has no effect.

48.13.38 SDMMC Force Event Register for Error Interrupt Status

Name: SDMMC_FEREIS

Access: Write-only

15	14	13	12	11	10	9	8
–	–	–	BOOTAE	–	–	ADMA	ACMD
7	6	5	4	3	2	1	0
CURLIM	DATEND	DATCRC	DATTEO	CMDIDX	CMDEND	CMDCRC	CMDTEO

- **CMDTEO: Force Event for Command Timeout Error**

For testing purposes, the user can write this bit to 1 to rise the CMDTEO status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **CMDCRC: Force Event for Command CRC Error**

For testing purposes, the user can write this bit to 1 to rise the CMDCRC status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **CMDEND: Force Event for Command End Bit Error**

For testing purposes, the user can write this bit to 1 to rise the CDMEND status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **CMDIDX: Force Event for Command Index Error**

For testing purposes, the user can write this bit to 1 to rise the CMDIDX status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **DATTEO: Force Event for Data Timeout error**

For testing purposes, the user can write this bit to 1 to rise the DATTEO status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **DATCRC: Force Event for Data CRC error**

For testing purposes, the user can write this bit to 1 to rise the DATCRC status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **DATEND: Force Event for Data End Bit Error**

For testing purposes, the user can write this bit to 1 to rise the DATEND status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **CURLIM: Force Event for Current Limit Error**

For testing purposes, the user can write this bit to 1 to rise the CURLIM status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **ACMD: Force Event for Auto CMD Error**

For testing purposes, the user can write this bit to 1 to rise the ACMD status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **ADMA: Force Event for ADMA Error**

For testing purposes, the user can write this bit to 1 to rise the ADMA status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

- **BOOTAE: Force Event for Boot Acknowledge Error**

For testing purposes, the user can write this bit to 1 to rise the BOOTAE status flag in SDMMC_EISTR.

Writing this bit to 0 has no effect.

48.13.39 SDMMC ADMA Error Status Register

Name: SDMMC_AESR

Access: Read-only

7	6	5	4	3	2	1	0
–	–	–	–	–	LMIS	ERRST	

- **ERRST: ADMA Error State**

This field indicates the state of ADMA when an error has occurred during an ADMA data transfer. This field never indicates 2 because ADMA never stops in this state.

Value	ADMA Error State when Error Occurred	Content of SDMMC_ASARx Registers
0	ST_STOP (Stop DMA)	Points to the descriptor following the error descriptor
1	ST_FDS (Fetch Descriptor)	Points to the error descriptor
2	–	(Not used)
3	ST_TRF (Transfer Data)	Points to the descriptor following the error descriptor

- **LMIS: ADMA Length Mismatch Error**

This error occurs in the following two cases:

– While Block Count Enable (BCEN) is being set, the total data length specified by the Descriptor table is different from that specified by the Block Count (BLKCNT) and Transfer Block Size (BLKSIZE).

– The total data length cannot be divided by the Transfer Block Size (BLKSIZE).

0: No error

1: Error

48.13.40 SDMMC ADMA System Address Register

Name: SDMMC_ASAR

Access: Read/Write

31	30	29	28	27	26	25	24
ADMASA							
23	22	21	20	19	18	17	16
ADMASA							
15	14	13	12	11	10	9	8
ADMASA							
7	6	5	4	3	2	1	0
ADMASA							

- **ADMASA: ADMA System Address**

This field holds the byte address of the executing command of the descriptor table. The 32-bit address descriptor uses SDMMC_ASAR. At the start of ADMA, the user must set the start address of the descriptor table. The ADMA increments this register address, which points to the next Descriptor line to be fetched.

When the ADMA Error (ADMA) status flag rises, this field holds a valid descriptor address depending on the ADMA Error State (ERRST). The user must program Descriptor Table on 32-bit boundary and set 32-bit boundary address to this register. ADMA2 ignores the lower 2 bits of this register and assumes it to be 0.

48.13.41 SDMMC Preset Value Register

Name: SDMMC_PVRx [x=0..7]

Access: Read/Write

15	14	13	12	11	10	9	8
DRVSEL		–	–	–	CLKGSEL	SDCLKFSEL	

7	6	5	4	3	2	1	0
SDCLKFSEL							

One of the Preset Value Registers is effective based on the selected bus speed mode. Table 48-8 defines the conditions to select one of the SDMMC_PVRs.

Table 48-8. Preset Value Register Select Condition

Selected Bus Speed Mode	VS18EN (SDMMC_HC2R)	HSEN (SDMMC_HC1R)	UHSMS (SDMMC_HC2R)
Default Speed	0	0	don't care
High Speed	0	Response Timeout Error	don't care
SDR12	1	don't care	0
SDR25	1	don't care	1
SDR50	1	don't care	2
SDR104/HS200	1	don't care	3
DDR50	1	don't care	4
Reserved	1	don't care	Other values

Table 48-9 shows the effective Preset Value Register according to the Selected Bus Speed mode.

Table 48-9. Preset Value Registers

SDMMC_PVRx	Selected Bus Speed Mode	Signal Voltage
SDMMC_PVR0	Initialization	3.3V or 1.8V
SDMMC_PVR1	Default Speed	3.3V
SDMMC_PVR2	High Speed	3.3V
SDMMC_PVR3	SDR12	1.8V
SDMMC_PVR4	SDR25	1.8V
SDMMC_PVR5	SDR50	1.8V
SDMMC_PVR6	SDR104/HS200	1.8V
SDMMC_PVR7	DDR50	1.8V

When Preset Value Enable (PVALEN) in SDMMC_HC2R is set to 1, SDCLK Frequency Select (SDLCKFSEL) and Clock Generator Select (CLKGSEL) in SDMMC_CCR, and Driver Strength Select (DRVSEL) in SDMMC_HC2R are automatically set based on the Selected Bus Speed mode. This means that the user does not need to set these fields when preset is enabled. A Preset Value Register for Initialization (SDMMC_PVR0) is not selected by Bus Speed mode. Before starting the initialization sequence, the user needs to set a clock preset value to SDCLKFSEL in SDMMC_CCR. PVALEN can be set to 1 after the initialization is completed.

Note: Preset Values in SDMMC_PVRx registers are not supposed to be written by the user. However, the user can modify preset values only if Capabilities Write Enable (CAPWREN) is set to 1 in SDMMC_CACR.

- **SDCLKFSEL: SDCLK Frequency Select**

Refer to SDCLKFSEL in SDMMC_CCR.

- **CLKGSEL: Clock Generator Select**

Refer to CLKGSEL in SDMMC_CCR.

- **DRVSEL: Driver Strength Select**

Refer to DRVSEL in SDMMC_HC2R.

48.13.42 SDMMC Slot Interrupt Status Register

Name: SDMMC_SISR

Access: Read-only

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	INTSSL	

- **INTSSL: Interrupt Signal for Each Slot**

These status bits indicate the logical OR of Interrupt Signals and Wakeup Signal for each SDMMC instance in the product (INTSSL[x] corresponds to SDMMCx instance in the product).

48.13.43 SDMMC Host Controller Version Register

Name: SDMMC_HCVR

Access: Read-only

15	14	13	12	11	10	9	8
VVER							

7	6	5	4	3	2	1	0
SVER							

- **SVER: Specification Version Number**

This status indicates the SD Host Controller Specification Version.

Value	Description
0	SD Host Specification Version 1.00
1	SD Host Specification Version 2.00, including the feature of the ADMA and Test Register
2	SD Host Specification Version 3.00

- **VVER: Vendor Version Number**

Reserved. Value subject to change. No functionality associated. This is the Atmel internal version of the macrocell.

48.13.44 SDMMC Additional Present State Register

Name: SDMMC_APSR

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	HDATLL			

- **HDATLL: DAT[7:4] High Line Level**

This status is used to check the DAT[7:4] line level to recover from errors, and for debugging.

48.13.45 SDMMC e.MMC Control 1 Register

Name: SDMMC_MC1R

Access: Read/Write

7	6	5	4	3	2	1	0
FCD	RSTN	BOOTA	OPD	DDR	–	CMDTYP	

- **CMDTYP: e.MMC Command Type**

Value	Name	Description
0	NORMAL	The command is not an e.MMC specific command.
1	WAITIRQ	This bit must be set to 1 when the e.MMC is in Interrupt mode (CMD40). Refer to “Interrupt Mode” in the “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51”.
2	STREAM	This bit must be set to 1 in the case of Stream Read(CMD11) or Stream Write (CMD20). Only effective for e.MMC up to revision 4.41.
3	BOOT	Starts a Boot Operation mode at the next write to SDMMC_CR. Boot data are read directly from e.MMC device.

- **DDR: e.MMC HSDDR Mode**

This bit selects the High Speed DDR mode.

0: High Speed DDR is not selected.

1: High Speed DDR is selected.

Note: The clock divider (DIV) in SDMMC_CCR must be set to a value different from 0 when HSEN is 1.

- **OPD: e.MMC Open Drain Mode**

This bit sets the command line in open drain.

0: The command line is in push-pull.

1: The command line is in open drain.

- **BOOTA: e.MMC Boot Acknowledge Enable**

This bit must be set according to the value of BOOT_ACK in the Extended CSD Register (refer to “Embedded MultiMedia Card (e.MMC) Electrical Standard 4.51”).

When this bit is set to 1, the SDMMC waits for boot acknowledge pattern from the e.MMC before receiving boot data.

If the boot acknowledge pattern is wrong, the BOOTAE status flag rises in SDMMC_EISTR if BOOTAE is set in SDMMC_EISTER. An interrupt is generated if BOOTAE is set in SDMMC_EISIER.

If the no boot acknowledge pattern is received, the DATTEO status flag rises in SDMMC_EISTR if DATTEO is set in SDMMC_EISTER. An interrupt is generated if DATTEO is set in SDMMC_EISIER.

- **RSTN: e.MMC Reset Signal**

This bit controls the e.MMC reset signal.

0: Reset signal is inactive.

1: Reset signal is active.

- **FCD: e.MMC Force Card Detect**

When using e.MMC, the user can set this bit to 1 to bypass the card detection procedure using the SDMMC_CD signal.

0 (DISABLED): e.MMC Forced Card Detect is disabled. The SDMMC_CD signal is used and debounce timing is applied.

1 (ENABLED): e.MMC Forced Card Detect is enabled.

48.13.46 SDMMC e.MMC Control 2 Register

Name: SDMMC_MC2R

Access: Write-only

7	6	5	4	3	2	1	0
–	–	–	–	–	–	ABOOT	SRESP

- **SRESP: e.MMC Abort Wait IRQ**

This bit is used to exit from the Interrupt mode. When this bit is written to 1, the SDMMC sends the CMD40 response automatically. This brings the e.MMC from Interrupt mode to the standard Data Transfer mode. Writing this bit to 0 is ignored.

Note: This bit is only effective when CMD_TYP in SDMMC_MC1R is set to WAITIRQ.

- **ABOOT: e.MMC Abort Boot**

This bit is used to exit from Boot mode. Writing this bit to 1 exits the Boot Operation mode. Writing 0 is ignored.

48.13.47 SDMMC AHB Control Register

Name: SDMMC_ACR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	BMAX	

- **BMAX: AHB Maximum Burst**

This field selects the maximum burst size in case of DMA transfer.

Value	Name	Description
0	INCR16	The maximum burst size is INCR16.
1	INCR8	The maximum burst size is INCR8.
2	INCR4	The maximum burst size is INCR4.
3	SINGLE	Only SINGLE transfers are performed.

48.13.48 SDMMC Clock Control 2 Register

Name: SDMMC_CC2R

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	FSDCLKD

- **FSDCLKD: Force SDCLK Disabled**

The user can choose to maintain the SDCLK during 8 SDCLK cycles after the end bit of the last data block in case of a read transaction, or after the end bit of the CRC status in case of a write transaction.

0: The SDCLK is forced and it cannot be stopped immediately after the transaction.

1: The SDCLK is not forced and it can be stopped immediately after the transaction.

48.13.49 SDMMC Retuning Control 1 Register

Name: SDMMC_RTC1R

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	TMREN

- **TMREN: Retuning Timer Enable**

Enable the retuning timer.

0 (DISABLED): The retuning timer is disabled.

1 (ENABLED): The retuning timer is enabled.

48.13.50 SDMMC Retuning Control 2 Register

Name: SDMMC_RTC2R

Access: Write-only

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	RLD

- **RLD: Retuning Timer Reload**

This bit is only efficient if the Retuning timer is enabled (TMREN is set to 1 in SDMMC_RTC1R). Once the Timer Counter Value (TCVAL) is set to a non-zero value in SDMMC_RTCVR, setting this bit to 1 starts the timer count. The retuning timer count restarts each time this bit is written to 1.

Writing this bit to 0 has no effect.

48.13.51 SDMMC Retuning Counter Value Register

Name: SDMMC_RTCVR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	TCVAL			

- **TCVAL: Retuning Timer Counter Value**

The TCVAL value corresponds to the time, in seconds, before expiration of the retuning timer. This value must range between 1 and 11. Any other value results in the retuning timer to be disabled.

48.13.52 SDMMC Retuning Interrupt Status Enable Register

Name: SDMMC_RTISTER

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	TEVT

- **TEVT: Retuning Timer Event**

0 (MASKED): The TEVT status flag in SDMMC_RTISTR is masked.

1 (ENABLED): The TEVT status flag in SDMMC_RTISTR is enabled.

48.13.53 SDMMC Retuning Interrupt Signal Enable Register

Name: SDMMC_RTISIER

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	TEVT

- **TEVT: Retuning Timer Event**

0 (MASKED): No interrupt is generated when the TEVT status rises in SDMMC_RTISTR.

1 (ENABLED): An interrupt is generated when the TEVT status rises in SDMMC_RTISTR.

48.13.54 SDMMC Retuning Interrupt Status Register

Name: SDMMC_RTISTR

Access: Read/Write

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	TEVT

- **TEVT: Retuning Timer Event**

This bit is set to 1 when the retuning timer count is elapsed if TEVT is set to 1 in SDMMC_RTISTER. An interrupt is generated if TEVT is set to 1 in SDMMC_RTISIER.

Writing this bit to 1 clears this bit.

0: No retuning timer event.

1: Retuning timer event.

48.13.55 SDMMC Retuning Status Slots Register

Name: SDMMC_RTSSR

Access: Read-only

7	6	5	4	3	2	1	0
–	–	–	–	–	–	TEVTSLOT	

- **TEVTSLOT: Retuning Timer Event Slots**

These status bits indicate the TEVT status for each SDMMC instance in the product (TEVTSLOT[x] corresponds to SDMMCx instance in the product).

48.13.56 SDMMC Tuning Control Register

Name: SDMMC_TUNCR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SMPLPT

- **SMPLPT: Sampling Point**

This bit selects the position of the sampling point into the data window for SDR104 and HS200 modes.

0: Sampling point is set at 50% of the data window.

1: Sampling point is set at 75% of the data window.

48.13.57 SDMMC Capabilities Control Register

Name: SDMMC_CACR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
KEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	CAPWREN

- **CAPWREN: Capabilities Write Enable**

This bit can only be written if KEY correspond to 46h.

0: Capabilities registers (SDMMC_CA0R and SDMMC_CA1R) cannot be written.

1: Capabilities registers (SDMMC_CA0R and SDMMC_CA1R) can be written.

- **KEY: Key**

Value	Name	Description
46h	KEY	Writing any other value in this field aborts the write operation of the CAPWREN bit. Always reads as 0.

48.13.58 SDMMC Calibration Control Register

Name: SDMMC_CALCR

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	CALP			
23	22	21	20	19	18	17	16
–	–	–	–	CALN			
15	14	13	12	11	10	9	8
CNTVAL							
7	6	5	4	3	2	1	0
–	–	TUNDIS	ALWYSON	–	–	–	EN

- **EN: PADs Calibration Enable**

This bit is automatically cleared once the calibration is done.

0: SDMMC I/O calibration disabled.

1: SDMMC I/O calibration enabled.

- **ALWYSON: Calibration Analog Always ON**

0: Calibration analog is shut down after each calibration.

1: Calibration analog remains powered after calibration.

- **TUNDIS: Calibration During Tuning Disabled**

0: Calibration is launched before each tuning.

1: Calibration is not launched at tuning.

- **CNTVAL: Calibration Counter Value**

Defines the number of HCLOCK cycles (divided by 4) required to cover the I/O calibration cell startup time.

$$CNTVAL_{Minimum} = \frac{t_{STARTUP}}{4 \times t_{HCLOCK}}$$

$t_{STARTUP}$ = 2 µs

- **CALN: Calibration N Status**

Calibration code for the n-channel transistors to match the required output impedance.

- **CALP: Calibration P Status**

Calibration code for the p-channel transistors to match the required output impedance.

49. Image Sensor Controller (ISC)

49.1 Description

The Image Sensor Controller (ISC) system manages incoming data from a parallel sensor. It supports a single active interface. The parallel interface protocol can use a free-running clock or a gated clock strategy. It supports the ITU-R BT 656/1120 422 protocol with a data width of 8 bits or 10 bits and raw Bayer format. The internal image processor includes adjustable white balance, color filter array interpolation, color correction, gamma correction, 12 bits to 10 bits compression, programmable color space conversion, horizontal and vertical chrominance subsampling module. The module also integrates a triple channel direct memory access controller master interface.

49.2 Embedded Characteristics

- Parallel 12-bit Interface for Raw Bayer, YCbCr, Monochrome and JPEG Compressed Sensor Interface
- BT.601/656/1120 Video Interface Supported
- Progressive Systems and Segmented Frame Systems
- Raw Bayer, YCbCr, Luminance (Black and White) Pixel Format Supported
- Resolution up to 2592 x 1944
- Input Pixel Clock up to 96 MHz
- Output Master Clock Generation
- Cropping
- Adjustable White Balance
- Raw Bayer Color Filter Array Interpolation
- Color Correction
- Gamma Correction
- Color Space Conversion
- Contrast and Brightness Control
- 4:4:4 to 4:2:2 Subsampler
- 4:2:2 to 4:2:0 Subsampler
- Rounding, Limiting and Packing unit
- Histogram Generation
- System Interface: Direct Memory Access Interface with Packed, Semi Planar and Planar output format
- Output Memory Format: 16 bpp RGB, 32 bpp RGB, 16 bpp, YCbCr 444, YCbCr 422, YCbCr 420, up to 12-bit raw Bayer

49.3 Block Diagram and Use Cases

49.3.1 Image Sensor Controller Functional Diagrams

Figure 49-1. Image Sensor Controller Block Diagram

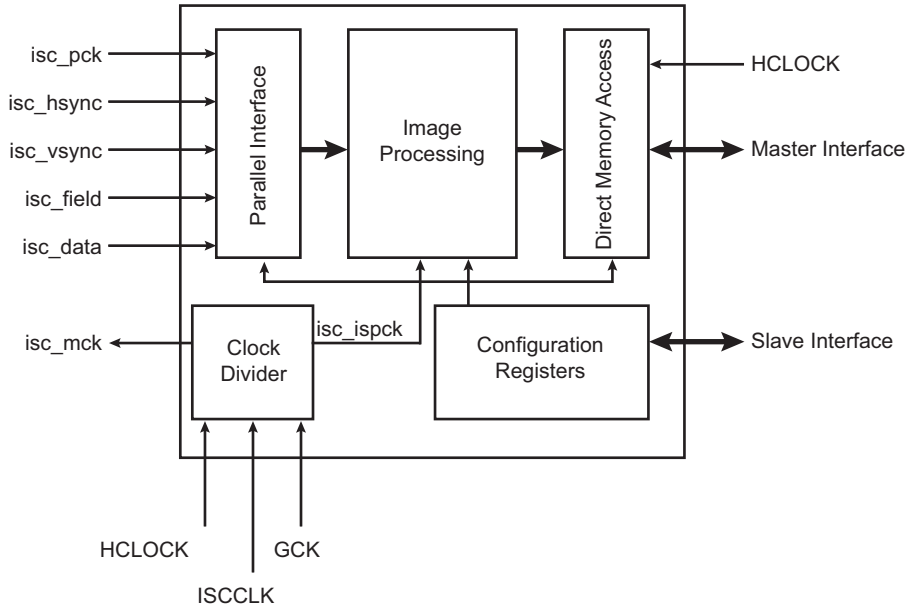
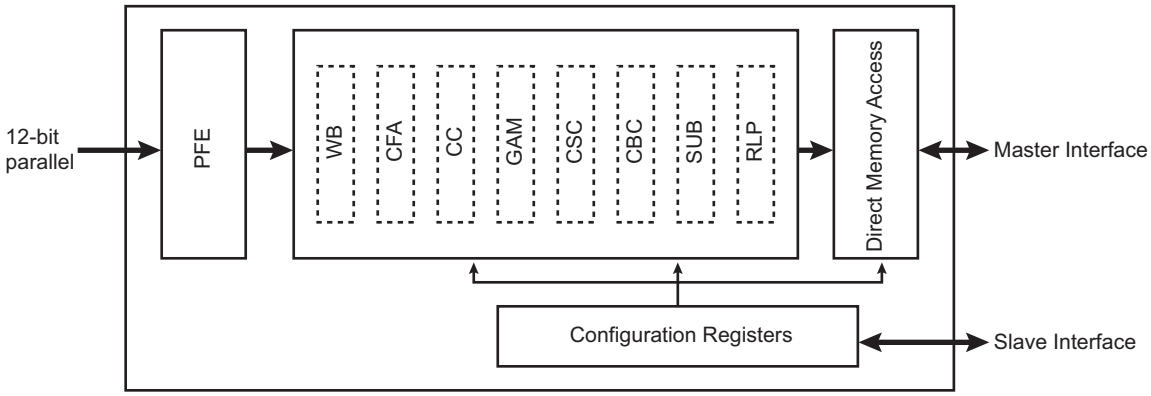


Figure 49-2. Image Sensor Controller Raw Bayer Signal Processor

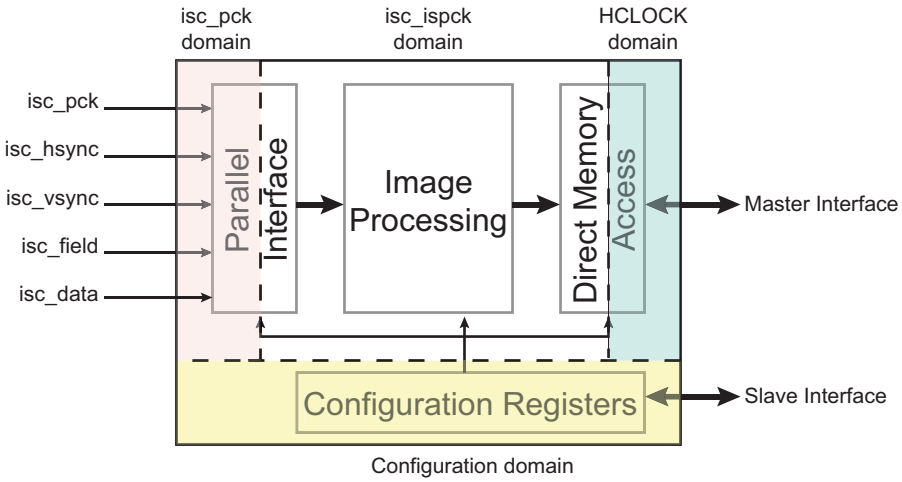


The ISC video pipeline integrates the following submodules:

- PFE: Parallel Front End to sample the camera sensor input stream
- WB: Programmable white balance in the Bayer domain
- CFA: Color filter array interpolation module
- CC: Programmable color correction
- GAM: Gamma correction
- CSC: Programmable color space conversion
- CBC: Contrast and Brightness control
- SUB: This module performs YCbCr444 to YCbCr420 chrominance subsampling.
- RLP: This module performs rounding, range limiting and packing of the incoming data.

49.3.2 Image Sensor Controller Clock Domain Diagram

Figure 49-3. Clock Domain Hierarchy



49.3.3 Image Sensor Controller Typical Use Cases

Figure 49-4. Raw Bayer Sensor

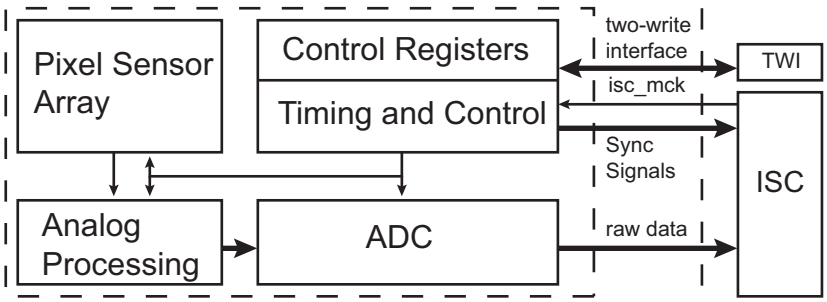


Figure 49-5. Raw Bayer Sensor with Embedded Image Processor

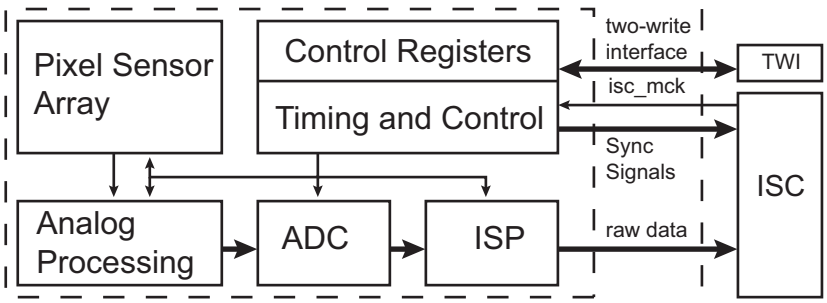


Figure 49-6. BT656 Video Interface Sensor

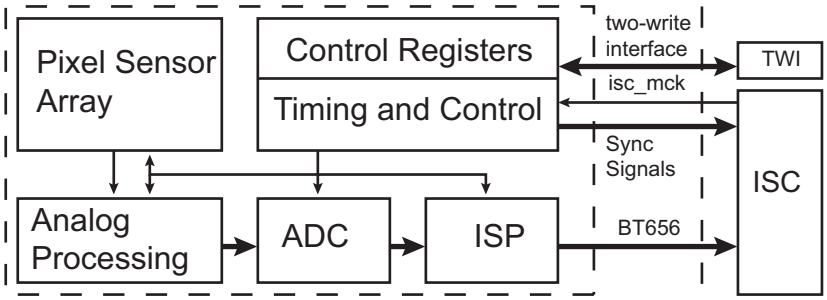


Figure 49-7. Sensor with JPEG Output

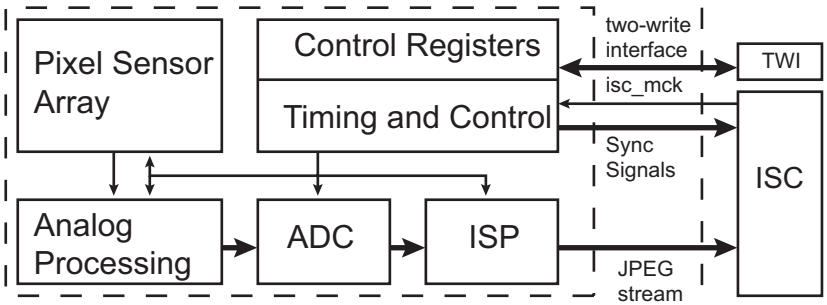
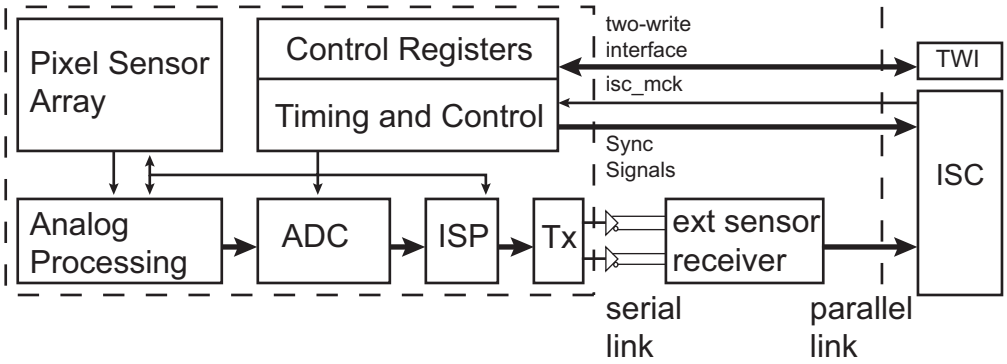


Figure 49-8. Serial CMOS Sensor with External Parallel Bridge



49.4 Product Dependencies

49.4.1 I/O Lines

The pins used for interfacing the ISC are multiplexed with the PIO lines. The programmer must first program the PIO controller to assign the ISC pins to their peripheral function. If I/O lines of the ISC are not used by the application, they can be used for other purposes by the PIO controller.

Table 49-1. I/O Lines

Instance	Signal	I/O Line	Peripheral
ISC	ISC_D0	PB26	F
ISC	ISC_D0	PC9	C
ISC	ISC_D0	PD7	E
ISC	ISC_D1	PB27	F

Table 49-1. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
ISC	ISC_D1	PC10	C
ISC	ISC_D1	PD8	E
ISC	ISC_D2	PB28	F
ISC	ISC_D2	PC11	C
ISC	ISC_D2	PD9	E
ISC	ISC_D3	PB29	F
ISC	ISC_D3	PC12	C
ISC	ISC_D3	PD10	E
ISC	ISC_D4	PB30	F
ISC	ISC_D4	PC13	C
ISC	ISC_D4	PD11	E
ISC	ISC_D4	PD12	F
ISC	ISC_D5	PB31	F
ISC	ISC_D5	PC14	C
ISC	ISC_D5	PD12	E
ISC	ISC_D5	PD13	F
ISC	ISC_D6	PC0	F
ISC	ISC_D6	PC15	C
ISC	ISC_D6	PD13	E
ISC	ISC_D6	PD14	F
ISC	ISC_D7	PC1	F
ISC	ISC_D7	PC16	C
ISC	ISC_D7	PD14	E
ISC	ISC_D7	PD15	F
ISC	ISC_D8	PC2	F
ISC	ISC_D8	PC17	C
ISC	ISC_D8	PD6	E
ISC	ISC_D8	PD16	F
ISC	ISC_D9	PC3	F
ISC	ISC_D9	PC18	C
ISC	ISC_D9	PD5	E
ISC	ISC_D9	PD17	F
ISC	ISC_D10	PB24	F
ISC	ISC_D10	PC19	C
ISC	ISC_D10	PD4	E
ISC	ISC_D10	PD18	F
ISC	ISC_D11	PB25	F

Table 49-1. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
ISC	ISC_D11	PC20	C
ISC	ISC_D11	PD3	E
ISC	ISC_D11	PD19	F
ISC	ISC_FIELD	PC8	F
ISC	ISC_FIELD	PC25	C
ISC	ISC_FIELD	PD18	E
ISC	ISC_FIELD	PD23	F
ISC	ISC_HSYNC	PC6	F
ISC	ISC_HSYNC	PC23	C
ISC	ISC_HSYNC	PD17	E
ISC	ISC_HSYNC	PD22	F
ISC	ISC_MCK	PC7	F
ISC	ISC_MCK	PC24	C
ISC	ISC_MCK	PD2	E
ISC	ISC_MCK	PD11	F
ISC	ISC_PCK	PC4	F
ISC	ISC_PCK	PC21	C
ISC	ISC_PCK	PD15	E
ISC	ISC_PCK	PD20	F
ISC	ISC_VSYNC	PC5	F
ISC	ISC_VSYNC	PC22	C
ISC	ISC_VSYNC	PD16	E
ISC	ISC_VSYNC	PD21	F

49.4.2 Power Management

The peripheral clock is not continuously provided to the ISC. The programmer must first enable the ISC clock in the Power Management Controller (PMC) before using the ISC.

49.4.3 Interrupt Sources

The ISC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the ISC interrupt requires the Interrupt Controller to be programmed first.

Table 49-2. Peripheral IDs

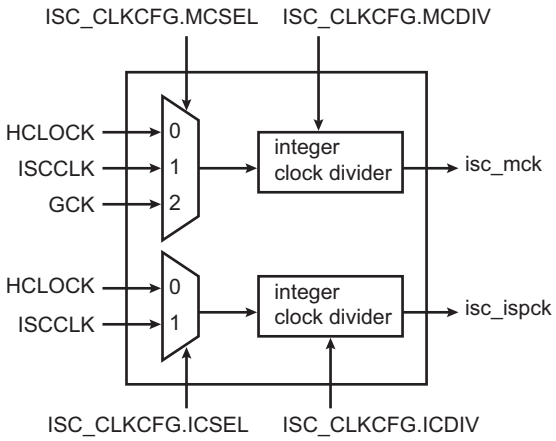
Instance	ID
ISC	46

49.5 Functional Description

49.5.1 ISC Clock Management

The ISC module provides the isc_mck output clock to the image sensor. The isc_mck clock has three selectable clock sources (ISC_CLKCFG.MCSEL field) and one programmable clock divider (ISC_CLKCFG.MCDIV). The clock is enabled using the ISC_CLKEN.MCEN field. The isc_mck is driven by the ISC and is the external reference clock of the CMOS sensor.

Figure 49-9. Clock Divider Block Diagram



The ISC digital pipeline requires internally a functional clock named isc_ispck. This clock is also fully programmable. This isc_ispck is enabled using the ISC_CLKEN.ICEN field. This clock is mandatory for ISC operation. The ISC module is designed to accept input signals that are asynchronous to the isc_ispck. Synchronization is done internally as long as the following relationship holds:

isc_pck frequency is less than or equal to isc_ispck, and isc_ispck is greater than or equal to HCLOCK.

49.5.1.1 Software Requirement

A software write operation to the ISC_CLKEN or ISC_CLKDIS register requires double clock domain synchronization and is not permitted when the ISC_CLKSR.SIP is asserted.

49.5.2 Parallel Interface Timing Description

The parallel interface protocol supports two operating modes.

Figure 49-10. Free-Running Pixel Clock

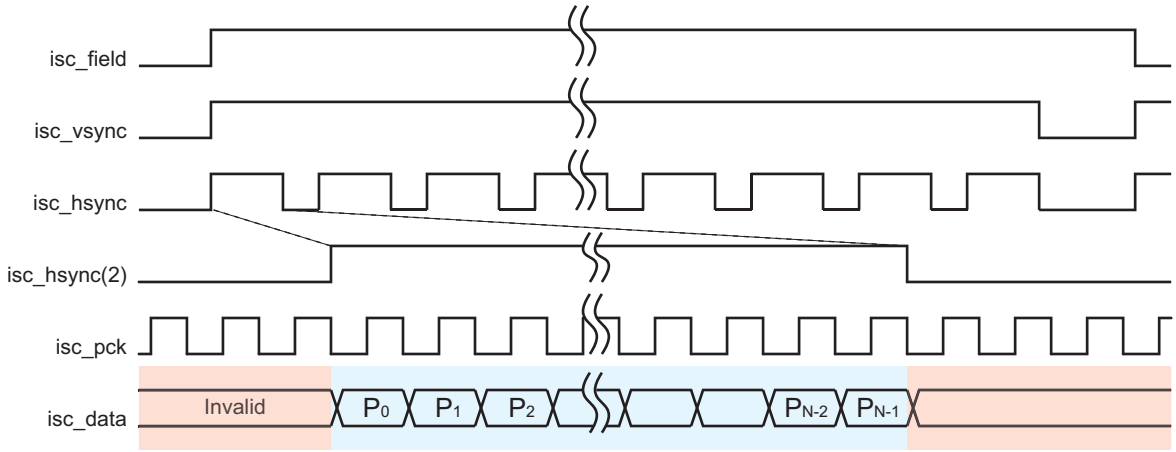
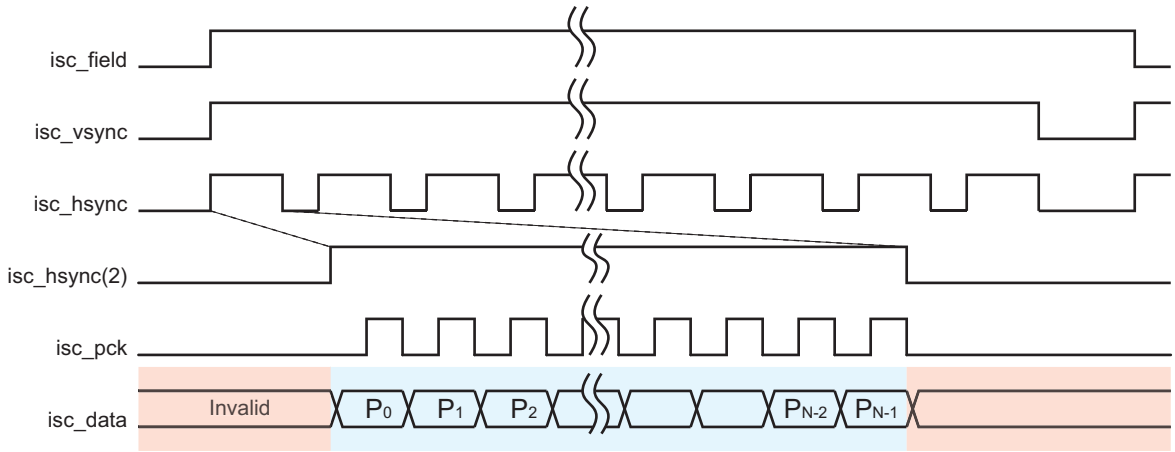


Figure 49-11. Gated Pixel Clock



49.5.3 BT.601/656/1120 Embedded Timing Synchronization Operation

The ISC module supports embedded synchronization decoding. When the ISC_PFE_CFG0.CCIR656 field is set, the decoder is activated and signals isc_vsync isc_hsync are not used to decode the valid pixels. If the CCIR10_8N is set, the bitstream is 10 bits wide, otherwise it is only 8 bits wide. When the ISC_PFE_CFG0.CCIR_CRC is set, the decoder automatically corrects the error.

Figure 49-12. Field/Segment Timing Relationship for Interlaced and Segmented Frame Systems

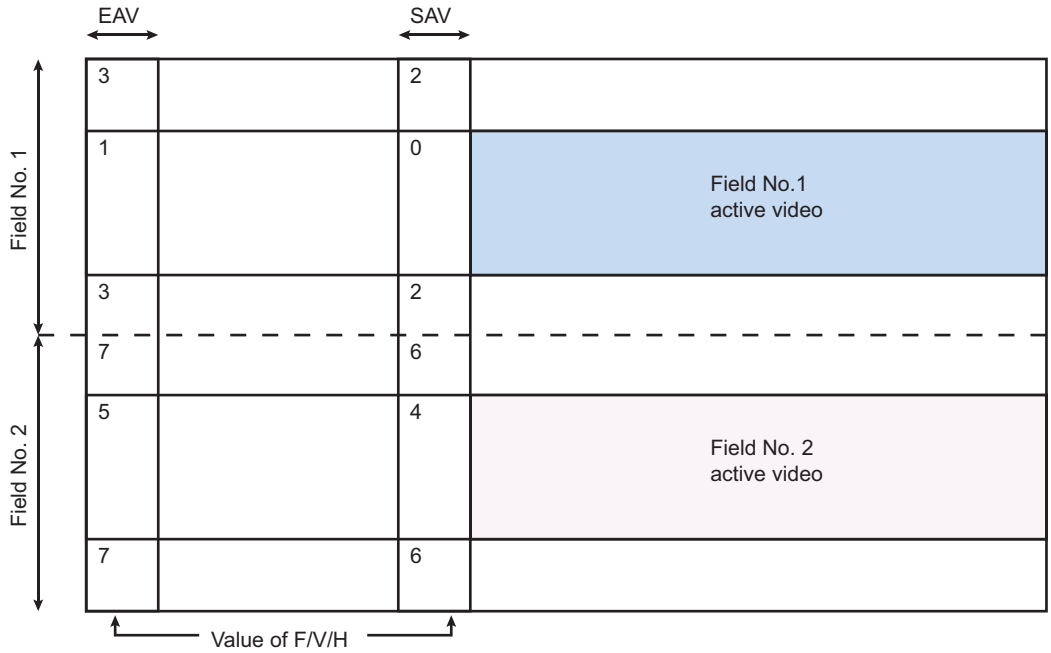
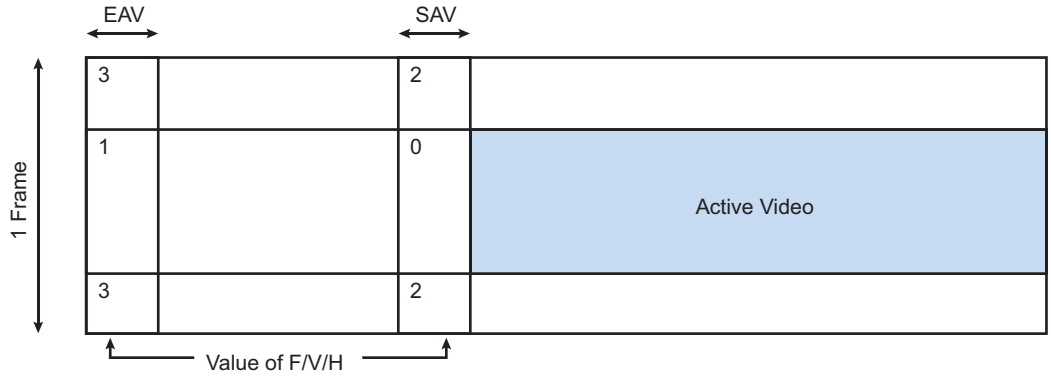


Figure 49-13. Frame Timing Relationship for Progressive Systems



49.5.4 Parallel Interface External Sensor Connections

49.5.4.1 YCbCr, 10-bit CCIR656 with Embedded Synchronization

This mode is activated when fields ISC_PFE_CFG0.CCIR656 and ISC_PFE_CFG0.CCIR10_8N are both set.

Interface Bit	First Word	Second Word	Third Word	Fourth Word
isc_data[11](MSB)	1	0	0	1
isc_data[10]	1	0	0	F
isc_data[9]	1	0	0	V
isc_data[8]	1	0	0	H
isc_data[7]	1	0	0	P3
isc_data[6]	1	0	0	P2
isc_data[5]	1	0	0	P1

Interface Bit	First Word	Second Word	Third Word	Fourth Word
isc_data[4]	1	0	0	P0
isc_data[3]	1	0	0	0
isc_data[2]	1	0	0	0
isc_data[1]	not used	not used	not used	not used
isc_data[0]	not used	not used	not used	not used

49.5.4.2 YCbCr, 8-bit CCIR656 with Embedded Synchronization

This mode is activated when field ISC_PFE_CFG0.CCIR656 is set and field ISC_PFE_CFG0.CCIR10_8N is cleared.

Interface Bit	First Word	Second Word	Third Word	Fourth Word
isc_data[11](MSB)	1	0	0	1
isc_data[10]	1	0	0	F
isc_data[9]	1	0	0	V
isc_data[8]	1	0	0	H
isc_data[7]	1	0	0	P3
isc_data[6]	1	0	0	P2
isc_data[5]	1	0	0	P1
isc_data[4]	1	0	0	P0
isc_data[3]	not used	not used	not used	not used
isc_data[2]	not used	not used	not used	not used
isc_data[1]	not used	not used	not used	not used
isc_data[0]	not used	not used	not used	not used

49.5.4.3 RAW Bayer Parallel Interface

The table below shows how to connect the data bus of a RAW Bayer sensor.

Interface	Bayer 12-bit	Bayer 11-bit	Bayer 10-bit	Bayer 9-bit	Bayer 8-bit
isc_data[11](MSB)	DOUT[11]	DOUT[10]	DOUT[9]	DOUT[8]	DOUT[7]
isc_data[10]	DOUT[10]	DOUT[9]	DOUT[8]	DOUT[7]	DOUT[6]
isc_data[9]	DOUT[9]	DOUT[8]	DOUT[7]	DOUT[6]	DOUT[5]
isc_data[8]	DOUT[8]	DOUT[7]	DOUT[6]	DOUT[5]	DOUT[4]
isc_data[7]	DOUT[7]	DOUT[6]	DOUT[5]	DOUT[4]	DOUT[3]
isc_data[6]	DOUT[6]	DOUT[5]	DOUT[4]	DOUT[3]	DOUT[2]
isc_data[5]	DOUT[5]	DOUT[4]	DOUT[3]	DOUT[2]	DOUT[1]
isc_data[4]	DOUT[4]	DOUT[3]	DOUT[2]	DOUT[1]	DOUT[0]
isc_data[3]	DOUT[3]	DOUT[2]	DOUT[1]	DOUT[0]	Not Used

Interface	Bayer 12-bit	Bayer 11-bit	Bayer 10-bit	Bayer 9-bit	Bayer 8-bit
isc_data[2]	DOUT[2]	DOUT[1]	DOUT[0]	Not Used	Not Used
isc_data[1]	DOUT[1]	DOUT[0]	Not Used	Not Used	Not Used
isc_data[0]	DOUT[0]	Not Used	Not Used	Not Used	Not Used

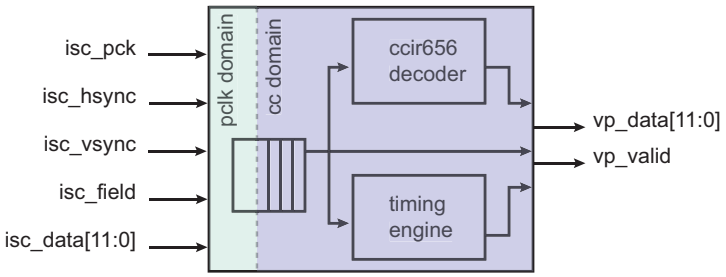
49.5.4.4 Monochrome Parallel Interface

The table below shows how to connect the data bus of a Monochrome sensor.

Interface	Mono 12-bit	Mono 11-bit	Mono 10-bit	Mono 9-bit	Mono 8-bit
isc_data[11](MSB)	DOUT[11]	DOUT[10]	DOUT[9]	DOUT[8]	DOUT[7]
isc_data[10]	DOUT[10]	DOUT[9]	DOUT[8]	DOUT[7]	DOUT[6]
isc_data[9]	DOUT[9]	DOUT[8]	DOUT[7]	DOUT[6]	DOUT[5]
isc_data[8]	DOUT[8]	DOUT[7]	DOUT[6]	DOUT[5]	DOUT[4]
isc_data[7]	DOUT[7]	DOUT[6]	DOUT[5]	DOUT[4]	DOUT[3]
isc_data[6]	DOUT[6]	DOUT[5]	DOUT[4]	DOUT[3]	DOUT[2]
isc_data[5]	DOUT[5]	DOUT[4]	DOUT[3]	DOUT[2]	DOUT[1]
isc_data[4]	DOUT[4]	DOUT[3]	DOUT[2]	DOUT[1]	DOUT[0]
isc_data[3]	DOUT[3]	DOUT[2]	DOUT[1]	DOUT[0]	Not Used
isc_data[2]	DOUT[2]	DOUT[1]	DOUT[0]	Not Used	Not Used
isc_data[1]	DOUT[1]	DOUT[0]	Not Used	Not Used	Not Used
isc_data[0]	DOUT[0]	Not Used	Not Used	Not Used	Not Used

49.5.5 Parallel Front End (PFE) Module

Figure 49-14. PFE Block Diagram



The Parallel Front End module performs data resampling across clock domain boundary. It includes a CCIR656 decoder used to convert a standard ITU-R BT.656 stream to 24-bit digital video. It also generates pixels, syncs flags and valid signals to the main video pipeline. It ouputs field, video and synchronization signals. The PFE can optionally crop and limit the incoming pixel stream to a predefined horizontal and vertical value. By default, the PFE only relies on the cmos sensor horizontal and vertical reference to sample the incoming pixel stream. A pixel is sampled if, and only if, the vertical and horizontal synchronizations are valid and a pixel clock edge is detected.

ISC_PFE_CFG0.BPS shows the number of bits per sample. The PFE module outputs a 12-bit data on the vp_data[11:0] bus, and asserts the vp_valid signal when the data can be sampled.

PFE VP_DATA Mapping	Raw Bayer 12-bit	Raw Bayer 10-bit	YUV422 8-bit	YUV422 10-bit	Mono 12-bit
VP_DATA[11]	RGGB[11]	RGGB[9]	YC422[7]	YC422[9]	Y[11]
VP_DATA[10]	RGGB[10]	RGGB[8]	YC422[6]	YC422[8]	Y[10]
VP_DATA[9]	RGGB[9]	RGGB[7]	YC422[5]	YC422[7]	Y[9]
VP_DATA[8]	RGGB[8]	RGGB[6]	YC422[4]	YC422[6]	Y[8]
VP_DATA[7]	RGGB[7]	RGGB[5]	YC422[3]	YC422[5]	Y[7]
VP_DATA[6]	RGGB[6]	RGGB[4]	YC422[2]	YC422[4]	Y[6]
VP_DATA[5]	RGGB[5]	RGGB[3]	YC422[1]	YC422[3]	Y[5]
VP_DATA[4]	RGGB[4]	RGGB[2]	YC422[0]	YC422[2]	Y[4]
VP_DATA[3]	RGGB[3]	RGGB[1]	YC422[7] or 0	YC422[1]	Y[3]
VP_DATA[2]	RGGB[2]	RGGB[0]	YC422[6] or 0	YC422[0]	Y[2]
VP_DATA[1]	RGGB[1]	RGGB[9] or 0	YC422[5] or 0	YC422[9] or 0	Y[1]
VP_DATA[0]	RGGB[0]	RGGB[8] or 0	YC422[4] or 0	YC422[8] or 0	Y[0]

Note: When ISC_PFE_CFG0.REP is set, missing VP_DATA LSBs are replaced with replicated LSBs of the incoming stream, otherwise they are forced to zero.

The PFE module also includes logic to synchronize capture request with the incoming pixel stream. Two operating modes are available: Single Shot and Continuous Acquisition. When the ISC_PFE_CFG0.CONT field is cleared, the ISC transfers a single image to memory,

Figure 49-15. Single Shot Mode

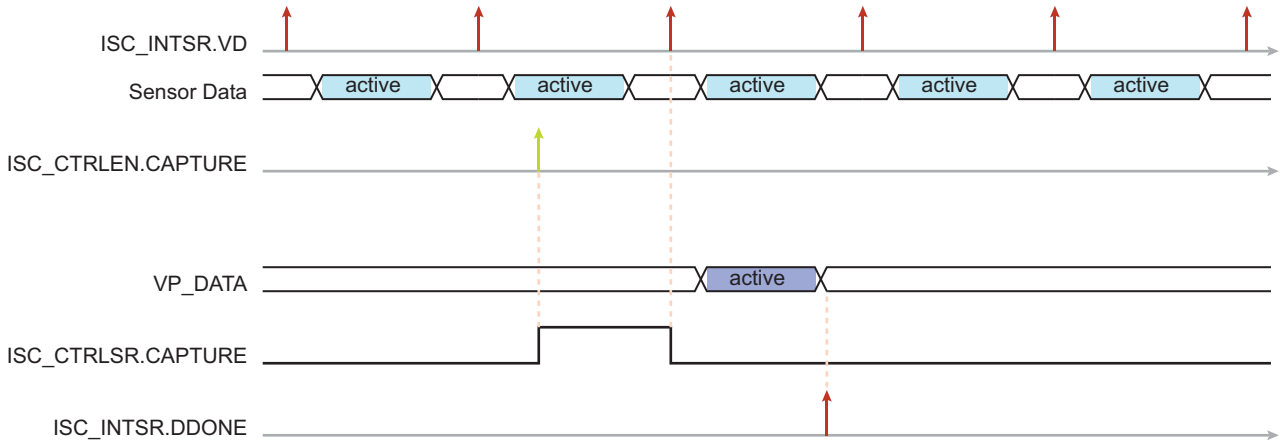
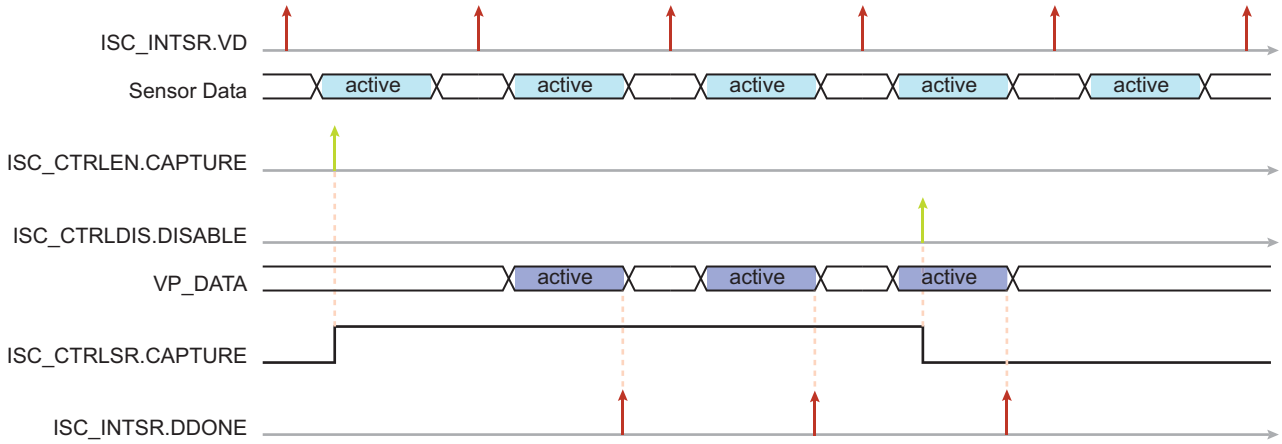
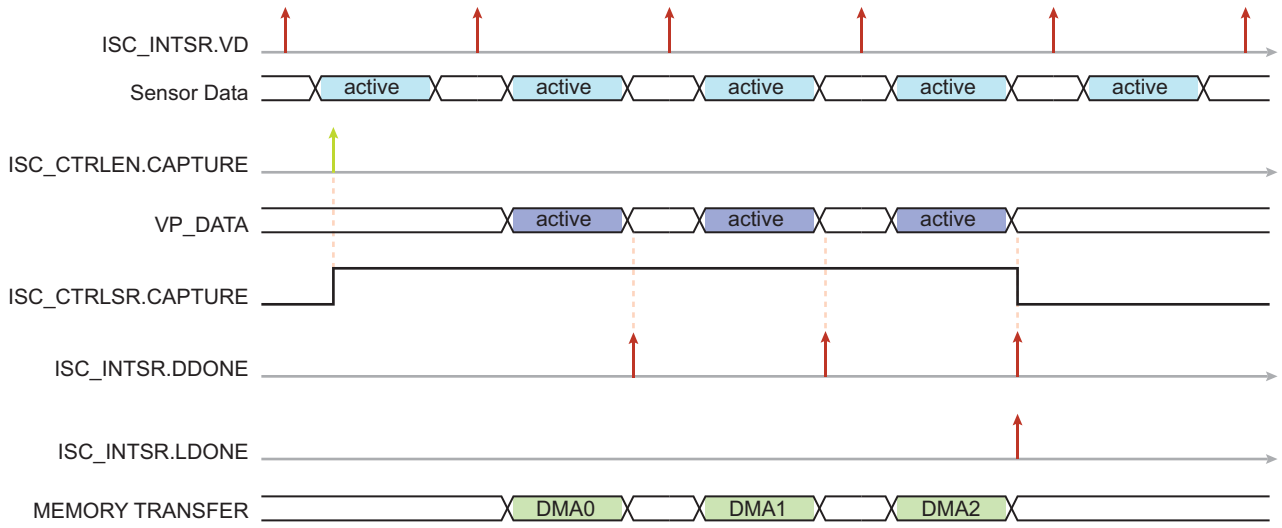


Figure 49-16. Continuous Acquisition Mode



When Continuous Acquisition mode is activated (ISC_PFE_CFG0.CONT is set), the data transfer terminates when either a DMA end of list is reached, a software disable is performed or a software reset is activated. The ISC_INTSR.DDONE is set at the end of the DMA data transfer.

Figure 49-17. Continuous Acquisition, DMA Terminated



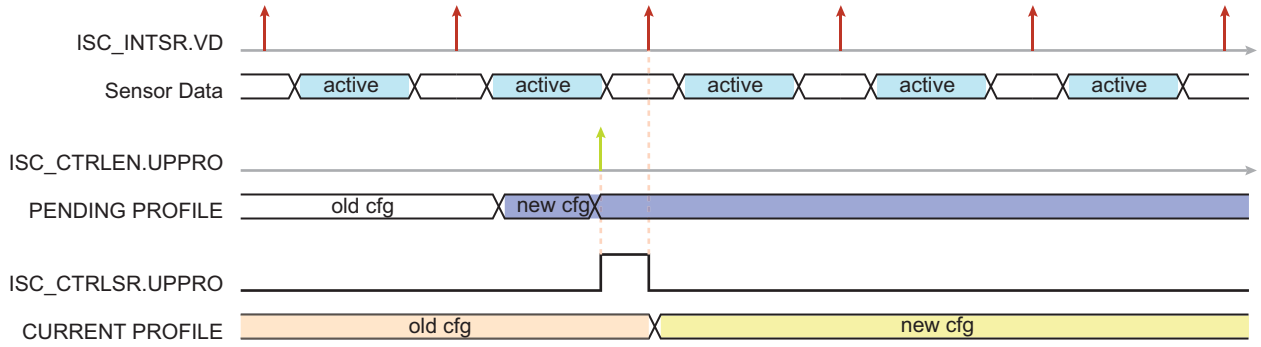
The linked list DMA transfer is terminated when an item of the list is programmed with the field ISC_DCTRL.NDE cleared or when the ISC_DNDA.NDA field is equal to zero. This configuration also clears the ISC_CTRLSR.CAPTURE field and sets the ISC_INTSR.LDONE interrupt flag.

The linked list DMA transfer starts if the field ISC_DCTRL.NDE is set and if the field ISC_DNDA.NDA is different from zero.

49.5.5.1 Update the ISC Profile

Each ISC register is double-buffered to simplify the software configuration and the synchronization with the associated frame buffer. When the configuration of the ISC is modified, the ISC_CTRLEN.UPPRO field must be set to transfer the configuration from the input buffer to the ISC video pipeline.

Figure 49-18. Update Profile Timing Diagram



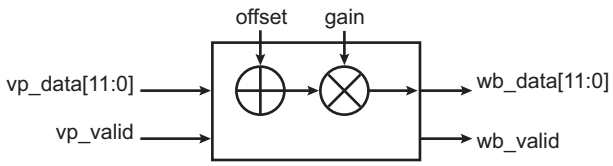
49.5.5.2 Software Requirements

Writing to the ISC_CTRLEN or ISC_CTRLDIS register requires a double domain synchronization, so it is forbidden to write these registers when the ISC_CTRLSR.SIP bit is asserted.

49.5.6 White Balance (WB) Module

The White Balance (WB) module captures the vp_data[11:0] bus from the PFE module when the vp_valid signal is asserted, and it generates a wb_data[11:0] data along with its validity signal wb_valid. When ISC_WB_CTRL.ENABLE is set, each Bayer color component (R, Gr, B, Gb) can be manually adjusted using an offset and a gain. The Bayer pattern is adjustable using the ISC_WB_CFG.BAYCFG field.

Figure 49-19. WB Block Diagram



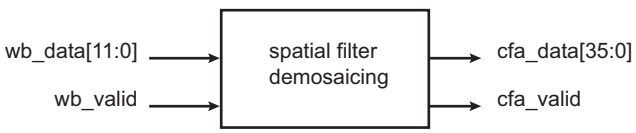
There are four {gain, offset} sets for each Bayer component. The output value is clipped.

ISC_WB_CTRL.ENABLE	WB_DATA Slice	Value
0	wb_data[11:0]	vp_data[11:0]
1	wb_data[11:0]	clipped((vp_data[11:0]+offset)*gain)

49.5.7 Color Filter Array (CFA) Interpolation Module

In a single-sensor system, each cell on the sensor has a specific color filter and microlens positioned above it. The raw data obtained from the sensor do not have the full R/G/B information at each cell position. Color interpolation is required to retrieve the missing components. The CFA module samples the wb_data[11:0] 12-bit bus when wb_valid is asserted and generates a 36-bit width data bus cfa_data[35:0] with the validity bit cfa_valid.

Figure 49-20. CFA Block Diagram



ISC_CFA_CTRL.ENABLE	CFA_DATA Slice	Value
0	cfa_data[35:24]	wb_data[11:0]
	cfa_data[23:12]	wb_data[11:0]
	cfa_data[11:0]	wb_data[11:0]
1	cfa_data[35:24]	R = spatial_filter_R(wb_data[11:0])
	cfa_data[23:12]	G = spatial_filter_G(wb_data[11:0])
	cfa_data[11:0]	B = spatial_filter_B(wb_data[11:0])

The filter kernel size is 5, and requires two additional lines to initialize the filter. When ISC_CFA_CFG.EITPOL is set, the missing information is interpolated from the nearest neighbor. If ISC_CFG_CFG.EITPOL is cleared, only valid pixels are used to initialize the filter kernel, but the output number of lines is less than the input number of lines. In that case, four lines are consumed for filling the kernel.

49.5.7.1 Frame Size Requirement when Edge Interpolation is Off, ISC_CFA_CFG.EITPOL Cleared

- Minimum number of rows (in): 5
- Minimum number of columns (in): 5
- Number or rows after CFA: Number of rows (in) - 4
- Number of columns after CFA: Number of columns (in) - 4

49.5.7.2 Frame Size Requirement when Edge Interpolation is On, ISC_CFA_CFG.EITPOL Set

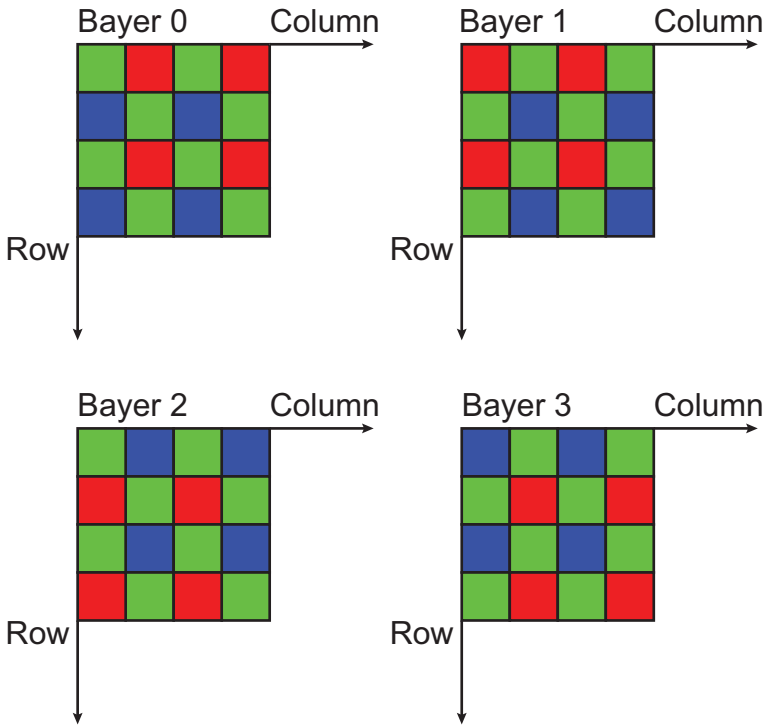
- Minimum number of rows (in): 3
- Minimum number of columns (in): 3
- Number of rows after CFA: Number of rows (in)
- Number of columns after CFA: Number of columns (in)

49.5.7.3 Bayer Mode and Edge Interpolation Description

When Edge Interpolation mode (ISC_CFA_CFG.EITPOL) is activated, dummy lines are generated using rows and columns replication.

The CFA module supports four sensor alignments using the ISC_CFA_CFG.BAYCFG field. Refer to Figure 49-21.

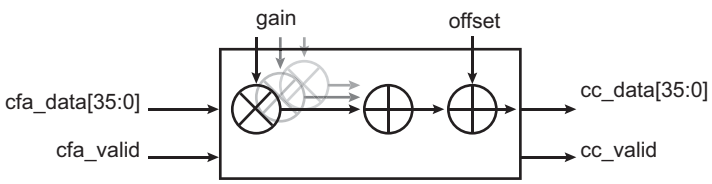
Figure 49-21. Supported Color Filter Array Patterns



49.5.8 Color Correction (CC) Module

RGB color correction is used to compensate for cross color bleeding in the filter used with the image sensor. The module samples the cfa_data[35:0] 36-bit bus when cfa_valid is asserted and generate a cc_data[35:0] 36-bit wide bus and a cc_valid signal.

Figure 49-22. CC Block Diagram



There are three {gain, offset} sets for color component R, G, B.

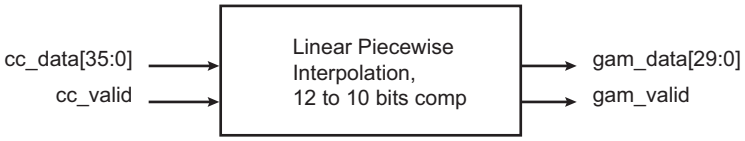
ISC_CC_CTRL.ENABLE	CC_DATA Slice	Value
0	cc_data[35:24]	cfa_data[35:24]
	cc_data[23:12]	cfa_data[23:12]
	cc_data[11:0]	cfa_data[11:0]
1	cc_data[35:24]	R=clipped(sum(cfa_data_x * gain_Rx) + offset_R)
	cc_data[23:12]	G=clipped(sum(cfa_data_x * gain_Gx) + offset_G)
	cc_data[11:0]	B=clipped(sum(cfa_data_x * gain_Bx) + offset_B)

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} RRGAIN & RGGAIN & RBGAIN \\ GRGAIN & GGGAIN & GBGAIN \\ BRGAIN & BGGAIN & BBGAIN \end{bmatrix} \times \begin{bmatrix} \text{cfa_data[35:24]} \\ \text{cfa_data[23:12]} \\ \text{cfa_data[11:0]} \end{bmatrix} + \begin{bmatrix} ROFST \\ GOFST \\ BOFST \end{bmatrix}$$

49.5.9 Gamma Curve (GAM) Module

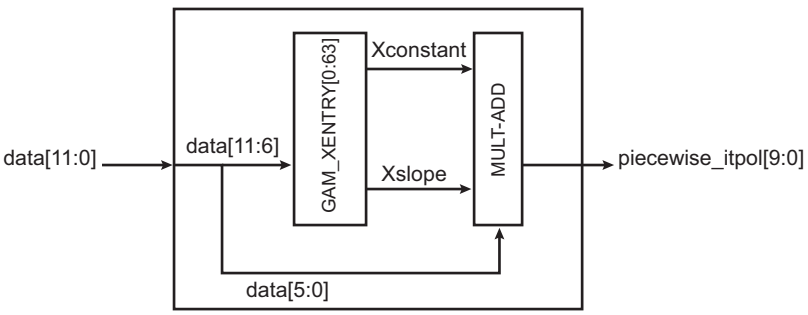
The GAM module samples the cc_data[35:0] bus when cc_valid is asserted, and generates gam_data[29:0] 30-bit width data along with the validity signal gam_valid. Imaging devices have non-linear characteristics, but the transfer function is approximated by a power function. The intensity of each of the linear RGB components is transformed to a non-linear signal through the use of the gamma correction submodule. The power function is linearly interpolated using 64 breakpoints. This also performs a 12-bit to 10-bit compression. The polynomial for the linear interpolation between breakpoints is ***i*** and ***i*** +1. Consequently, for each breakpoint, two values are required: constant and slope. The table values are programmable through the user interface when the gamma correction module is disabled (ISC_GAM_CTRL.ENABLE is cleared). ISC_GAM_RENTRY is used for Red gamma correction. ISC_GAM_GENTRY is used for Green gamma correction. ISC_GAM_BENTRY is used for Blue gamma correction. Each table entry is composed of a 10-bit (signed) slope and a 10-bit constant.

Figure 49-23. GAM Block Diagram



ISC_GAM_CTRL.ENABLE	ISC_GAM_CTRL.XLUT	GAM_DATA Slice	Value
0	0	gam_data[29:0]	cc_data[29:0]
1	0	gam_data[29:20]	cc_data[35:26]
		gam_data[19:10]	cc_data[23:14]
		gam_data[9:0]	cc_data[11:2]
1	1	gam_data[29:20]	R=piecewise_itpol(cc_data_r[35:24])
		gam_data[19:10]	G=piecewise_itpol(cc_data_r[23:12])
		gam_data[9:0]	B=piecewise_itpol(cc_data_r[11:0])

Figure 49-24. Piecewise Linear Interpolation Block Diagram



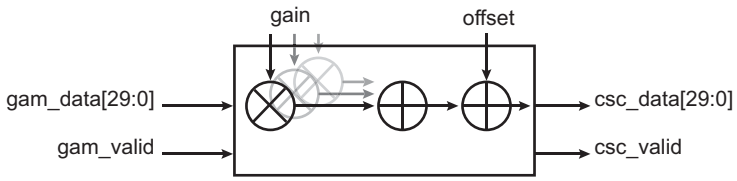
The interpolation consists of three tables that store the function values GAM_XENTRY[0:63] where X stands for R, G and B. The input of the table has six bits. It outputs a slope and a constant. The slope is later multiplied by the

data lsb (6-bit) and added to a constant. The final value is the gamma-corrected value of the input. This module performs a 12-to-10 compression.

49.5.10 Color Space Conversion (CSC) Module

By converting an image from RGB to YCbCr color space, it is possible to separate Y, Cb and Cr information. The CSC samples the gam_data[29:0] 30-bit data bus, extracts YCbCr information from the sampled data, and then generates the color-converted data csc_data[29:0] and the validity signal csc_valid.

Figure 49-25. CSC Block Diagram



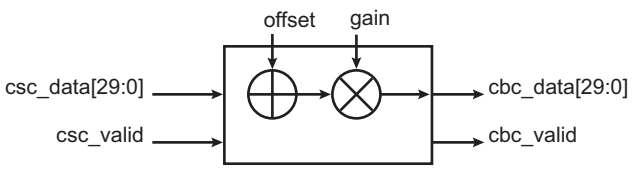
ISC_CSC_CTRL.ENABLE	CSC_DATA Slice	Value
0	csc_data[29:0]	gam_data[29:0]
1	csc_data[29:20]	Y = clipped(sum(gam_data_x * gain_Yx) + offset_y << 2)
	csc_data[19:10]	Cb = clipped(sum(gam_data_x * gain_Cbx)+offset_cb << 2)
	csc_data[9:0]	Cr = clipped(sum(gam_data_x * gain_Crx)+offset_cr << 2)

$$\begin{bmatrix} Y \\ CB \\ CR \end{bmatrix} = \begin{bmatrix} YR & YG & YB \\ CBR & CBG & CBB \\ CRR & CRG & CRB \end{bmatrix} \times \begin{bmatrix} \text{gam_data[29:20]} \\ \text{gam_data[19:10]} \\ \text{gam_data[9:0]} \end{bmatrix} + \begin{bmatrix} YOFST \\ CBOFST \\ CROFST \end{bmatrix}$$

49.5.11 Contrast and Brightness

Luminance is adjusted through the use of Brightness Offset and Contrast Gain. Chrominance is left unchanged. The CBC samples the csc_data[29:0] 30-bit bus when csc_valid is asserted and generates cbc_data[29:0] with the validity signal cbc_valid.

Figure 49-26. CBC Block Diagram

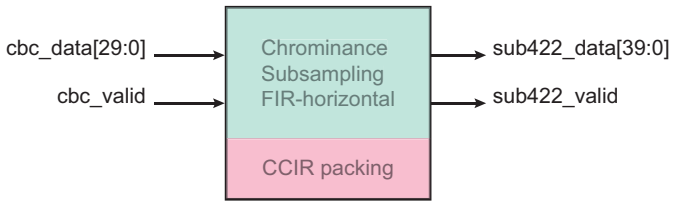


ISC_CBC_CTRL.ENABLE	ISC_CBC_CFG.CCIR	CBC_DATA Slice	Value
0	0	cbc_data[29:0]	csc_data[29:0]
1	0	cbc_data[29:20]	Y = clipped((csc_data[29:20]+offset)*gain)
		cbc_data[19:10]	Cb = csc_data[19:10]
		cbc_data[9:0]	Cr = csc_data[9:0]
1	1	cbc_data[29:10]	0
		cbc_data[9:0]	ccir656 stream with luminance correction

49.5.12 4:4:4 To 4:2:2 Chrominance Horizontal Subsampler (SUB422) Module

The color space conversion output stream is a full-bandwidth YCbCr 4:4:4 signal. The chrominance subsampling divides the horizontal chrominance sampling rate by two. A horizontal low pass filter is applied to avoid aliasing effect. The SUB422 module samples 444 full scale YCbCr cbc_data[29:0] 30-bit data, performs horizontal subsampling and generates the sub422_data[39:0] 40-bit data bus with its validity signal sub422_valid.

Figure 49-27. SUB422 Block Diagram



ISC_SUB422_CTRL.ENABLE	ISC_SUB422_CFG.CCIR	SUB422_DATA Slice	Value
0	0	sub422_data[29:0]	cbc_data[29:0]
1	0	sub422_data[39:30]	Y1= cbc_data1[29:20]
		sub422_data[29:20]	Y0 = cbc_data0[29:20]
		sub422_data[19:10]	Cb = filter_hor(cbc_data[19:10])
		sub422_data[9:0]	Cr = filter_hor(cbc_data[9:0])
1	1	sub422_data[39:30]	Y1 = cbc_data[9:0]
		sub422_data[29:20]	Y0 = cbc_data[9:0]
		sub422_data[19:10]	Cb = cbc_data[9:0]
		sub422_data[9:0]	Cr = cbc_data[9:0]

The filter_hor function included in the sub422 module is the chrominance horizontal filter.

sub422 data slice	YCbCr mapping
sub422_data[39:30]	Y1 (sample *n*)
sub422_data[29:20]	Y0 (sample *n-1*)
sub422_data[19:10]	Cb (from filter)
sub422_data[9:0]	Cr (from filter)

The filter chrominance position is selectable through the use of the ISC_SUB422_CFG.FILTER field.

Figure 49-28. Cosited Filter Configuration

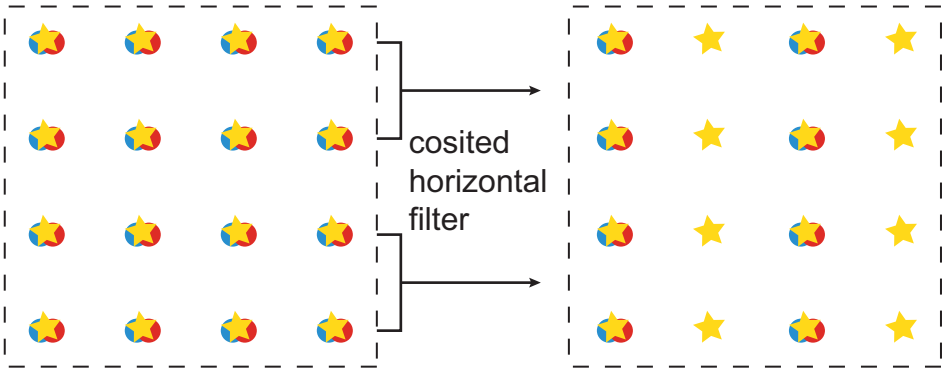
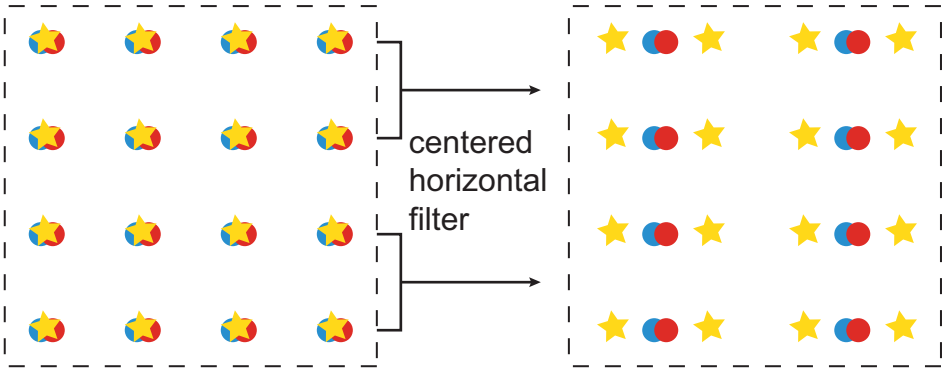


Figure 49-29. Centered Filter Configuration



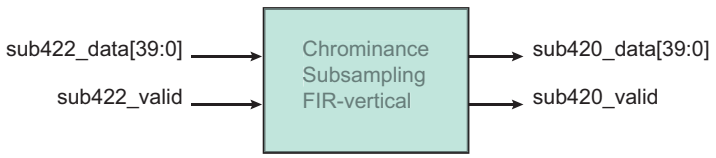
The SUB422 module performs luminance and chrominance packing. When the line length is odd, the missing luminance is a copy of the last but one luminance. It also means that the final dma stream written to memory is equal to the original horizontal size plus one when the line length is odd.

SUB422_DATA Slice	Line Length Even	Line Length Odd
sub422_data[39:30]	Y(n)	Y(n-1)
sub422_data[29:20]	Y(n-1)	Y(n-1)
sub422_data[19:10]	Cb (filtered)	Cb (filtered)
sub422_data[9:0]	Cr (filtered)	Cr (filtered)

49.5.13 4:2:2 To 4:2:0 Chrominance Vertical Subsampler (SUB420) Module

The chrominance subsampling divides the vertical chrominance sampling rate by two. A vertical low pass filter is applied to avoid aliasing effect. Two different filters are used when the source frame is interlaced, and the filter configuration depends on the field value (the field is propagated in the video pipeline).

Figure 49-30. SUB420 Block Diagram



The SUB420 module samples the sub422_data[39:0] 40-bit data when sub422_valid is asserted, then it performs a vertical subsampling and generates a valid sub420_data[39:0] 40-bit word and the corresponding sub420_valid signal.

ISC_SUB420_CTRL.ENABLE	SUB420_DATA Slice	Value
0	sub420_data[39:0]	sub422_data[39:0]
1	sub420_data[39:30]	Y1 = sub422_data[39:30]
	sub420_data[29:20]	Y0 = sub422_data[29:20]
	sub420_data[19:10]	Cb = filter_ver(sub422[19:10])
	sub420_data[9:0]	Cr = filter_ver(sub422[9:0])

The vertical filter is a two-tap filter; for progressive content the coefficient i {1, 1}. When an interlaced field is downsampled, the coefficients are different between the top and the bottom fields.

Figure 49-31. Vertical Chrominance Filter for Progressive Content (Cosited Chrominance Example)

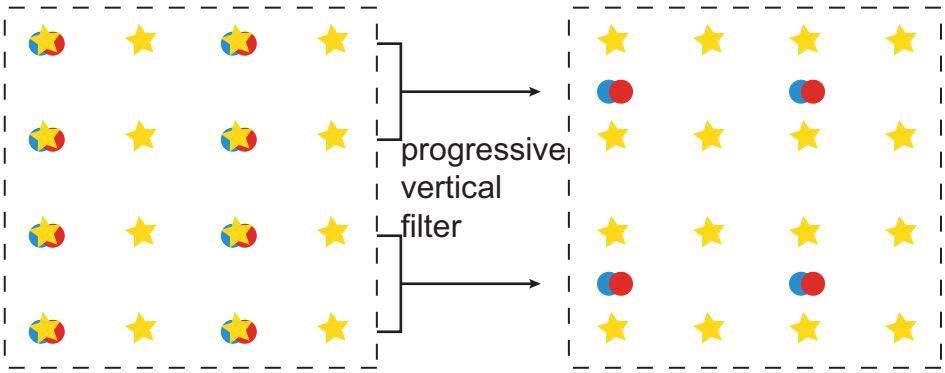


Figure 49-32. Field-dependent Chrominance Filter for Interlaced Content (Cosited Chrominance Example)

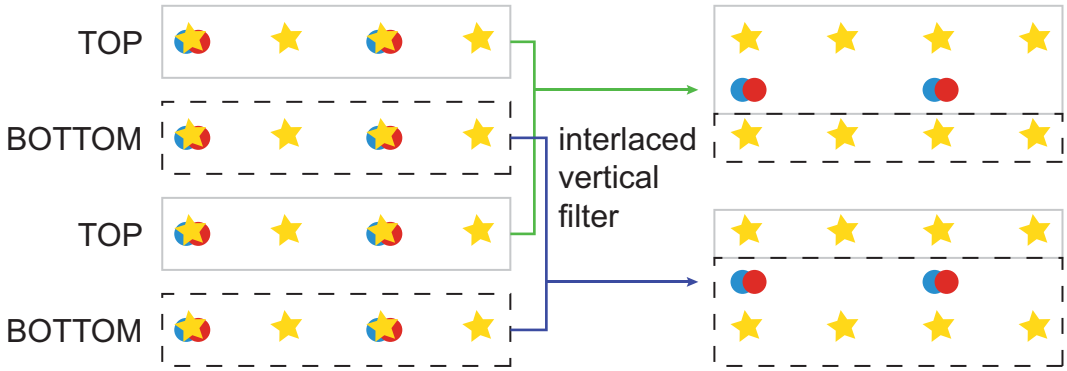


Table 49-3. Filter Configuration

ISC_SUB420_CTRL.FILTER	Field	Filter Configuration
0	progressive	{1, 1}
1	0 (TOP)	{3, 1}
	1 (BOTTOM)	{1, 3}

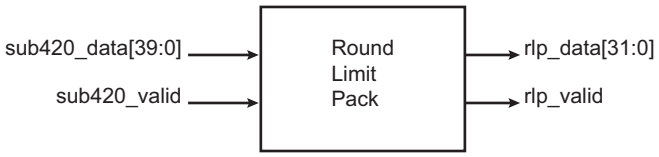
Table 49-4. Output Line Length Configuration

SUB420 Input Number of Rows	SUB420 Luminance Rows	SUB420 Chrominance Rows
M rows, M odd	M rows	(M+1)/2 rows
M rows, M even	M rows	M/2 rows

49.5.14 Rounding, Limiting and Packing (RLP) Module

This module is used to round, limit and pack in the incoming pixel stream before the DMA master module. The RLP samples the sub420_data[39:0] 40-bit data bus and generates rlp_data[31:0] 32-bit data words with the associated validity signal rlp_valid.

Figure 49-33. RLP Block Diagram



ISC_RLP_CFG	RLP_DATA Slice	Value
DAT8	rlp_data[31:8]	0
	rlp_data[7:0]	sub420_data[11:4]
DAT9	rlp_data[31:9]	0
	rlp_data[8:0]	sub420_data[11:3]
DAT10	rlp_data[31:10]	0
	rlp_data[9:0]	sub420_data[11:2]
DAT11	rlp_data[31:11]	0
	rlp_data[10:0]	sub420_data[11:1]
DAT12	rlp_data[31:12]	0
	rlp_data[11:0]	sub420_data[11:0]
DATY8	rlp_data[31:8]	0
	rlp_data[7:0]	Y = rounded(sub420_data[29:22])
DATY10	rlp_data[31:8]	0
	rlp_data[7:0]	Y = sub420_data[29:20])
ARGB444	rlp_data[31:16]	0
	rlp_data[15:12]	A = alpha[7:4]
	rlp_data[11:8]	R = sub420_data[29:26]
	rlp_data[7:4]	G = sub420_data[19:16]
	rlp_data[3:0]	B = sub420_data[9:6]

ISC_RLP_CFG	RLP_DATA Slice	Value
ARGB555	rlp_data[31:16]	0
	rlp_data[15]	A = alpha[7]
	rlp_data[14:10]	R = sub420_data[29:25]
	rlp_data[9:5]	G = sub420_data[19:15]
	rlp_data[4:0]	B = sub420_data[9:5]
RGB565	rlp_data[31:16]	0
	rlp_data[15:11]	R = sub420_data[29:25]
	rlp_data[10:5]	G = sub420_data[19:14]
	rlp_data[4:0]	B = sub420_data[9:5]
RGB32	rlp_data[31:24]	A = alpha[7:0]
	rlp_data[23:16]	R = sub420_data[29:22]
	rlp_data[15:8]	G = sub420_data[19:12]
	rlp_data[7:0]	B = sub420_data[9:2]
YCbCr422, YCbCr420	rlp_data[31:24]	Y1 = round(sub420_data[39:32])
	rlp_data[23:16]	Y0 = round(sub420_data[29:22])
	rlp_data[15:8]	Cb = round(sub420_data[19:12])
	rlp_data[7:0]	Cr = round(sub420_data[9:2])
YCbCr422, YCbCr420	rlp_data[31:24]	Y1 = round_limit(sub420_data[39:32])
	rlp_data[23:16]	Y0 = round_limit(sub420_data[29,22])
	rlp_data[15:8]	Cb = round_limit(sub420_data[19,12])
	rlp_data[7:0]	Cr = round_limit(sub420_data[9:2])
Undefined	rlp_data[31:0]	sub420_data[31:0]

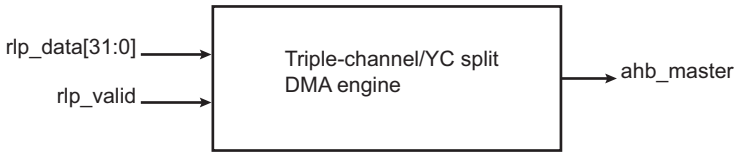
The round_limit function provides a simple method to round and limit the range of both Luminance and Chrominance signals.

ISC_RLP_CFG	8-bit Full Range	8-bit Limited Range
Y	0–255	16–235
Cb	0–255	16–240
Cr	0–255	16–240

49.5.15 DMA Interface

The descriptor-based DMA interface supports multiple buffers. A DMA stride value shows the offset between two consecutive lines (in bytes). If the stride is set to zero, the frame buffer is contiguous. When the ISC_DCTRL.WB field is set (Write Back), the DMA interface performs a single write operation to the ISC_DCTRL register, and sets ISC_DCTRL[7] to one and ISC_DCTRL[6] to the value of the frame field when interlaced content is being used. That means that interlaced fields are tagged with their relevant field values. The Write Back operation is always performed when the whole frame has been transferred to memory.

Figure 49-34. DMA Engine Block Diagram



ISC_DCFG.IMODE	DMA Engine Input Data
PACKED8	rlp_data[7:0]
PACKED16	rlp_data[15:0]
PACKED32	rlp_data[31:0]
YC422SP	rlp_data[31:0]
YC422P	rlp_data[31:0]
YC420SP	rlp_data[31:0]
YC420P	rlp_data[31:0]

When a bus error is detected, an interrupt flag is set. If the error occurs on a write operation, ISC_INTSR.WERR is asserted. If the error occurs on a read operation, ISC_INTSR.RERR is asserted. The ISC_INTSR.WERRID field gives details on the first error channel identifier.

49.5.15.1 Descriptor Memory Address Mapping

ISC_DCFG.IMODE	ISC_DAD0.AD0	ISC_DAD1.AD1	ISC_DAD2.AD2
PACKED8, PACKED16, PACKED32	data address	not used	not used
YC422SP	Y address	CbCr address	not used
YC422P	Y address	Cb address	Cr address
YC420SP	Y address	CbCr address	not used
YC420P	Y address	Cb address	Cr address

49.5.15.2 Descriptor Memory Mapping

Three descriptor views are available. Descriptor view 0 is used when the pixel or data stream is packed. Descriptor view 1 is used for YCbCr semi-planar pixel stream. Descriptor view 2 is used for YCbCr planar pixel stream.

Table 49-5. ISC_DCTRL.DVIEW = 0

Address	Register
ISC_DNDA+0x00	ISC_DCTRL
ISC_DNDA+0x04	ISC_DNDA
ISC_DNDA+0x08	ISC_DAD0
ISC_DNDA+0x0C	ISC_DST0

Table 49-6. ISC_DCTRL.DVIEW = 1

Address	Register
ISC_DNDA+0x00	ISC_DCTRL
ISC_DNDA+0x04	ISC_DNDA
ISC_DNDA+0x08	ISC_DAD0
ISC_DNDA+0x0C	ISC_DST0
ISC_DNDA+0x10	ISC_DAD1
ISC_DNDA+0x14	ISC_DST1

Table 49-7. ISC_DCTRL.DVIEW = 2

Address	Register
ISC_DNDA+0x00	ISC_DCTRL
ISC_DNDA+0x04	ISC_DNDA
ISC_DNDA+0x08	ISC_DAD0
ISC_DNDA+0x0C	ISC_DST0
ISC_DNDA+0x10	ISC_DAD1
ISC_DNDA+0x14	ISC_DST1
ISC_DNDA+0x18	ISC_DAD2
ISC_DNDA+0x1C	ISC_DST2

49.5.15.3 Example: Memory Mapping for 16-bit Packed, DMA Interface IMODE = 1 at ISC_DAD0.AD0 Location

Table 49-8. DAT8 Packing (ISC_RLP_CFG.MODE)

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW12	–	–	–	–	–	–	–	–	rlp_data1[7:0]								–	–	–	–	–	–	–	–	rlp_data0[7:0]							

Table 49-9. DAT9 Packing (ISC_RLP_CFG.MODE)

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW12	–	–	–	–	–	–	–	rlp_data1[8:0]									–	–	–	–	–	–	–	rlp_data0[8:0]								

Table 49-10. DAT10 Packing (ISC_RLP_CFG.MODE)

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW12	–	–	–	–	–	–	rlp_data1[9:0]										–	–	–	–	–	–	rlp_data0[9:0]									

Table 49-11. DAT11 Packing (ISC_RLP_CFG.MODE)

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW12	–	–	–	–	–	rlp_data1[10:0]											–	–	–	–	–	isc_data0[10:0]										

Table 49-12. DAT12 Packing (ISC_RLP_CFG.MODE)

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RAW12	–	–	–	–	rlp_data1[11:0]												–	–	–	–	rlp_data0[11:0]											

49.5.15.4 Example: Memory Mapping for 12-bit YC420SP, DMA Interface IMODE = 5

Table 49-13. Y Channel Located at ISC_DAD0.AD0 Memory Address

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y 8-bit	rlp_data1[31:24]								rlp_data1[23:16]								rlp_data0[31:24]								rlp_data0[23:16]							

Table 49-14. CbCr Channel Located at ISC_DAD1.AD1 Memory Address

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CC 16-bit	rlp_data1[15:0]																rlp_data0[15:0]															

49.5.15.5 Example: Memory Mapping for 12-bit YC420P, DMA Interface IMODE = 6

Table 49-15. Y Channel Located at ISC_DAD0.AD0 Memory Address

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Y 8-bit	rlp_data1[31:24]								rlp_data1[23:16]								rlp_data0[31:24]								rlp_data0[23:16]							

Table 49-16. Cb Channel Located at ISC_DAD1.AD1 Memory Address

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Cb 8-bit	rlp_data3[15:8]								rlp_data2[15:8]								rlp_data1[15:8]								rlp_data0[15:8]							

Table 49-17. Cr Channel Located at ISC_DAD2.AD2

Mem addr	0x3								0x2								0x1								0x0							
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Cr 8-bit	rlp_data3[7:0]								rlp_data2[7:0]								rlp_data1[7:0]								rlp_data0[7:0]							

49.5.16 Histogram Module

For each possible pixel value, the histogram counts the number of times the value was encountered in the current image. RGGB Bayer, RAW data or luminance histogram are available. There are 512 entries in the histogram entries, and each histogram bin can count up to 2^20 data. As the table entries are limited, each bin is actually a range, i.e., least significant bits are ignored. A write to ISC_CTRLEN.HISREQ initiates a new histogram. The counting operation ends when ISC_INTSR.HISDONE is set. At that time, a software or hardware dma transfer copies the table from the interface to the internal or external memory. To clear the table content (for a new operation), use the ISC_CTRLEN.HISCLR field. An automatic clear (reset after read) is available when bit ISC_HIS_CFG.RAR is set. In that case, as soon as the data is read from the table, the table entry is cleared.

Figure 49-35. Histogram Block Diagram

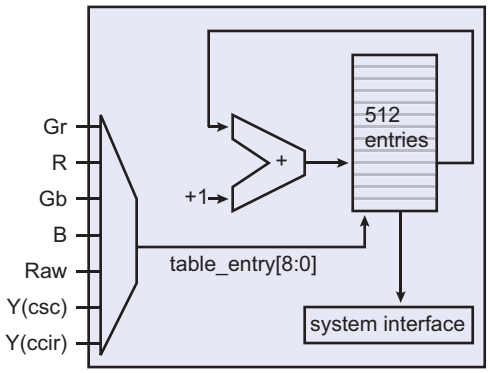
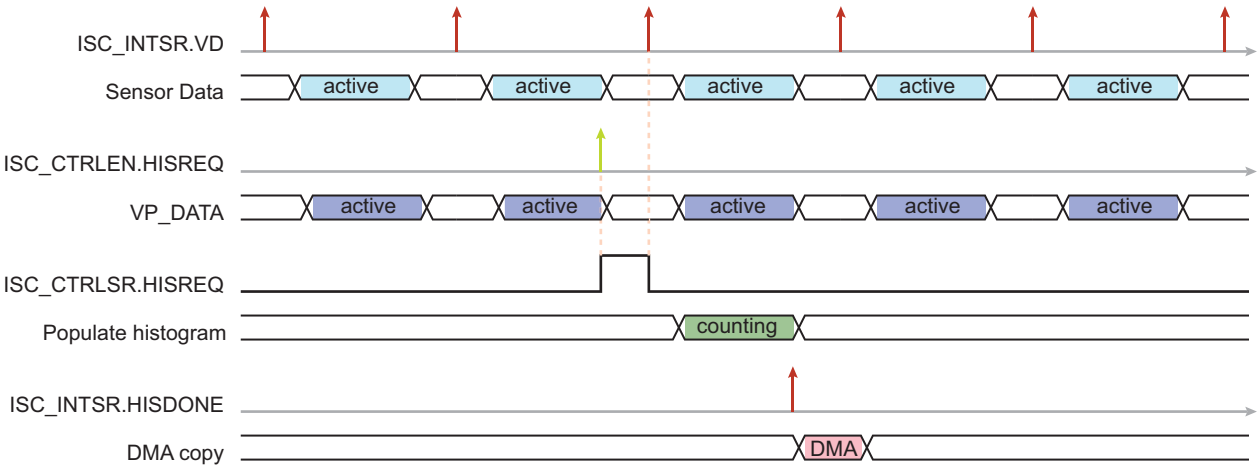


Figure 49-36. Histogram Request timing diagram



49.6 Image Sensor Controller (ISC) User Interface

Table 49-18. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Enable Register	ISC_CTRLEN	Write-only	–
0x04	Control Disable Register	ISC_CTRLDIS	Write-only	–
0x08	Control Status Register	ISC_CTRLSR	Read-only	0x00000000
0x0C	Parallel Front End Configuration 0 Register	ISC_PFE_CFG0	Read/Write	0x00000000
0x10	Parallel Front End Configuration 1 Register	ISC_PFE_CFG1	Read/Write	0x00000000
0x14	Parallel Front End Configuration 2 Register	ISC_PFE_CFG2	Read/Write	0x00000000
0x18	Clock Enable Register	ISC_CLKEN	Write-only	–
0x1C	Clock Disable Register	ISC_CLKDIS	Write-only	–
0x20	Clock Status Register	ISC_CLKSR	Read-only	0x00000000
0x24	Clock Configuration Register	ISC_CLKCFG	Read/Write	0x00000000
0x28	Interrupt Enable Register	ISC_INTEN	Write-only	–
0x2C	Interrupt Disable Register	ISC_INTDIS	Write-only	–
0x30	Interrupt Mask Register	ISC_INTMASK	Read-only	0x00000000
0x34	Interrupt Status Register	ISC_INTSR	Read-only	0x00000000
0x38–0x3C	Reserved	–	–	0x00000000
0x40–0x54	Reserved	–	–	0x00000000
0x58	White Balance Control Register	ISC_WB_CTRL	Read/Write	0x00000000
0x5C	White Balance Configuration Register	ISC_WB_CFG	Read/Write	0x00000000
0x60	White Balance Offset for R, GR Register	ISC_WB_O_RGR	Read/Write	0x00000000
0x64	White Balance Offset for B, GB Register	ISC_WB_O_BGB	Read/Write	0x00000000
0x68	White Balance Gain for R, GR Register	ISC_WB_G_RGR	Read/Write	0x00000000
0x6C	White Balance Gain for B, GB Register	ISC_WB_G_BGB	Read/Write	0x00000000
0x70	Color Filter Array Control Register	ISC_CFA_CTRL	Read/Write	0x00000000
0x74	Color Filter Array Configuration Register	ISC_CFA_CFG	Read/Write	0x00000000
0x78	Color Correction Control Register	ISC_CC_CTRL	Read/Write	0x00000000
0x7C	Color Correction RR RG Register	ISC_CC_RR_RG	Read/Write	0x00000000
0x80	Color Correction RB OR Register	ISC_CC_RB_OR	Read/Write	0x00000000
0x84	Color Correction GR GG Register	ISC_CC_GR_GG	Read/Write	0x00000000
0x88	Color Correction GB OG Register	ISC_CC_GB_OG	Read/Write	0x00000000
0x8C	Color Correction BR BG Register	ISC_CC_BR_BG	Read/Write	0x00000000
0x90	Color Correction BB OB Register	ISC_CC_BB_OB	Read/Write	0x00000000
0x94	Gamma Correction Control Register	ISC_GAM_CTRL	Read/Write	0x00000000
0x98	Gamma Correction Blue Entry 0	ISC_GAM_BENTRY0	Read/Write	0x00000000
...	...	...	...	...
0x194	Gamma Correction Blue Entry 63	ISC_GAM_BENTRY63	Read/Write	0x00000000

Table 49-18. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x198	Gamma Correction Green Entry 0	ISC_GAM_GENTRY0	Read/Write	0x00000000
...	...	...	...	...
0x294	Gamma Correction Green Entry 63	ISC_GAM_GENTRY63	Read/Write	0x00000000
0x298	Gamma Correction Red Entry 0	ISC_GAM_RENTRY0	Read/Write	0x00000000
...	...	...	...	...
0x394	Gamma Correction Red Entry 63	ISC_GAM_RENTRY63	Read/Write	0x00000000
0x398	Color Space Conversion Control Register	ISC_CSC_CTRL	Read/Write	0x00000000
0x39C	Color Space Conversion YR, YG Register	ISC_CSC_YR_YG	Read/Write	0x00000000
0x3A0	Color Space Conversion YB, OY Register	ISC_CSC_YB_OY	Read/Write	0x00000000
0x3A4	Color Space Conversion CBR CBG Register	ISC_CSC_CBR_CBG	Read/Write	0x00000000
0x3A8	Color Space Conversion CBB OCB Register	ISC_CSC_CBB_OCB	Read/Write	0x00000000
0x3AC	Color Space Conversion CRR CRG Register	ISC_CSC_CRR_CRG	Read/Write	0x00000000
0x3B0	Color Space Conversion CRB OCR Register	ISC_CSC_CRB_OCR	Read/Write	0x00000000
0x3B4	Contrast and Brightness Control Register	ISC_CBC_CTRL	Read/Write	0x00000000
0x3B8	Contrast and Brightness Configuration Register	ISC_CBC_CFG	Read/Write	0x00000000
0x3BC	Contrast and Brightness, Brightness Register	ISC_CBC_BRIGHT	Read/Write	0x00000000
0x3C0	Contrast and Brightness, Contrast Register	ISC_CBC_CONTRAST	Read/Write	0x00000000
0x3C4	Subsampling 4:4:4 to 4:2:2 Control Register	ISC_SUB422_CTRL	Read/Write	0x00000000
0x3C8	Subsampling 4:4:4 to 4:2:2 Configuration Register	ISC_SUB422_CFG	Read/Write	0x00000000
0x3CC	Subsampling 4:2:2 to 4:2:0 Control Register	ISC_SUB420_CTRL	Read/Write	0x00000000
0x3D0	Rounding, Limiting and Packing Configuration Register	ISC_RLP_CFG	Read/Write	0x00000000
0x3D4	Histogram Control Register	ISC_HIS_CTRL	Read/Write	0x00000000
0x3D8	Histogram Configuration Register	ISC_HIS_CFG	Read/Write	0x00000000
0x3DC	Reserved	–	–	–
0x3E0	DMA Configuration Register	ISC_DCFG	Read/Write	0x00000000
0x3E4	DMA Control Register	ISC_DCTRL	Read/Write	0x00000000
0x3E8	DMA Descriptor Address Register	ISC_DNDA	Read/Write	0x00000000
0x3EC	DMA Address 0 Register	ISC_DAD0	Read/Write	0x00000000
0x3F0	DMA Stride 0 Register	ISC_DST0	Read/Write	0x00000000
0x3F4	DMA Address 1 Register	ISC_DAD1	Read/Write	0x00000000
0x3F8	DMA Stride 1 Register	ISC_DST1	Read/Write	0x00000000
0x3FC	DMA Address 2 Register	ISC_DAD2	Read/Write	0x00000000
0x400	DMA Stride 2 Register	ISC_DST2	Read/Write	0x00000000
0x404–0x40C	Reserved	–	–	

Table 49-18. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x410	Histogram Entry 0	ISC_HIS_ENTRY0	Read-only	0x00000000
...	...	...	...	...
0xBFC	Histogram Entry 511	ISC_HIS_ENTRY511	Read-only	0x00000000

49.6.1 ISC Control Enable Register 0

Name: ISC_CTRLEN

Address: 0xF0008000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	HISCLR	HISREQ	UPPRO	CAPTURE

- **CAPTURE: Capture Input Stream Command**

0: Writing a zero to this bit has no effect.

1: Write one to start a single shot capture or a multiple frame.

- **UPPRO: Update Profile**

0: Writing a zero to this bit has no effect.

1: Write one to update the color profile.

- **HISREQ: Histogram Request**

0: Writing a zero to this bit has no effect.

1: Write one to update the histogram table.

- **HISCLR: Histogram Clear**

0: Writing a zero to this bit has no effect.

1: Write one to clear the histogram table.

49.6.2 ISC Control Disable Register 0

Name: ISC_CTRLDIS

Address: 0xF0008004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SWRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DISABLE

- **DISABLE: Capture Disable**

0: Writing a zero to this bit has no effect.

1: Write one to end the capture at the next Vertical Synchronization Detection.

- **SWRST: Software Reset**

0: Writing a zero to this bit has not effect.

1: Write one to perform a software reset of the interface.

49.6.3 ISC Control Status Register 0

Name: ISC_CTRLSR

Address: 0xF0008008

Access: Read-only

31	30	29	28	27	26	25	24
SIP	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	FIELD	–	HISREQ	UPPRO	CAPTURE

- **CAPTURE: Capture pending**

0: Capture mode is disabled.

1: Capture is pending.

- **UPPRO: Profile Update Pending**

0: There is no profile update pending request.

1: Indicates that the profile update request is still pending.

- **HISREQ: Histogram Request Pending**

0: There is no histogram pending request.

1: Indicates that the histogram request is still pending.

- **FIELD: Field Status (only relevant when the video stream is interlaced)**

0: The current field/segment is a top field

1: The current field/segment is a bottom field.

- **SIP: Synchronization In Progress**

0: The double domain synchronization is terminated.

1: The double domain synchronization is in progress.

49.6.4 ISC Parallel Front End Configuration 0 Register

Name: ISC_PFE_CFG0

Address: 0xF000800C

Access: Read/Write

31	30	29	28	27	26	25	24
REP	BPS			CCIR_REP	–	–	–

23	22	21	20	19	18	17	16
SKIPCNT							

15	14	13	12	11	10	9	8
–	–	ROWEN	COLEN	CCIR10_8N	CCIR_CRC	CCIR656	GATED

7	6	5	4	3	2	1	0
CONT	MODE			FPOL	PPOL	VPOL	HPOL

- **HPOL: Horizontal Synchronization Polarity**

0: HSYNC signal is active high, i.e. valid pixels are sampled when HSYNC is asserted.

1: HSYNC signal is active low, i.e. valid pixels are sampled when HSYNC is deasserted.

- **VPOL: Vertical Synchronization Polarity**

0: VSYNC signal is active high, i.e. valid pixels are sampled when VSYNC is asserted.

1: VSYNC signal is active low, i.e. valid pixels are sampled when VSYNC is deasserted.

- **PPOL: Pixel Clock Polarity**

0: The pixel stream is sampled on the rising edge of the pixel clock.

1: The pixel stream is sampled on the falling edge of the pixel clock.

- **FPOL: Field Polarity**

0: Top field is sampled when F value is 0; Bottom field is sampled when F value is 1

1: Top field is sampled when F value is 1; Bottom field is sampled when F value is 0

- **MODE: Parallel Front End Mode**

Value	Name	Description
0	PROGRESSIVE	Video source is progressive.
1	DF_TOP	Video source is interlaced, two fields are captured starting with top field.
2	DF_BOTTOM	Video source is interlaced, two fields are captured starting with bottom field.
3	DF_IMMEDIATE	Video source is interlaced, two fields are captured immediately.
4	SF_TOP	Video source is interlaced, one field is captured starting with the top field.
5	SF_BOTTOM	Video source is interlaced, one field is captured starting with the bottom field.
6	SF_IMMEDIATE	Video source is interlaced, one field is captured starting immediately.

- **CONT: Continuous Acquisition**

0: Single Shot mode

1: Video mode

- **GATED: Gated input clock**

0: The external pixel clock is free running.

1: The external pixel clock is gated.

- **CCIR656: CCIR656 input mode**

0: HSYNC and VSYNC signals are used to synchronize the input stream.

1: Embedded synchronization is used.

- **CCIR_CRC: CCIR656 CRC Decoder**

0: Embedded CRC is discarded.

1: Embedded CRC is decoded.

- **CCIR10_8N: CCIR 10 bits or 8 bits**

0: 8-bit mode

1: 10-bit mode

- **COLEN: Column Cropping Enable**

0: Column Cropping is disabled.

1: Column Cropping is enabled.

- **ROWEN: Row Cropping Enable**

0: Row Cropping is disabled

1: Row Cropping is enabled.

- **SKIPCNT: Frame Skipping Counter**

- **CCIR_REP: CCIR Replication**

0: Unused bits are stuck at 0.

1: Unused bits are copied from MSB.

- **BPS: Bits Per Sample**

Value	Name	Description
0	TWELVE	12-bit input
1	ELEVEN	11-bit input
2	TEN	10-bit input
3	NINE	9-bit input
4	EIGHT	8-bit input

- **REP: Up Multiply with Replication**

0: Unused bits are stuck at 0.

1: Unused bits are copied from MSB.

49.6.5 ISC Parallel Front End Configuration 1 Register

Name: ISC_PFE_CFG1

Address: 0xF0008010

Access: Read/Write

31	30	29	28	27	26	25	24
COLMAX							

23	22	21	20	19	18	17	16
COLMAX							

15	14	13	12	11	10	9	8
COLMIN							

7	6	5	4	3	2	1	0
COLMIN							

- **COLMIN: Column Minimum Limit**

Horizontal starting position of the cropping area

- **COLMAX: Column Maximum Limit**

Horizontal ending position of the cropping area

49.6.6 ISC Parallel Front End Configuration 2 Register

Name: ISC_PFE_CFG2

Address: 0xF0008014

Access: Read/Write

31	30	29	28	27	26	25	24
ROWMAX							

23	22	21	20	19	18	17	16
ROWMAX							

15	14	13	12	11	10	9	8
ROWMIN							

7	6	5	4	3	2	1	0
ROWMIN							

- **ROWMIN: Row Minimum Limit**

Vertical starting position of the cropping area

- **ROWMAX: Row Maximum Limit**

Vertical ending position of the cropping area

49.6.7 ISC Clock Enable Register

Name: ISC_CLKEN

Address: 0xF0008018

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	MCEN	ICEN

- **ICEN: ISP Clock Enable**

0: No effect.

1: Enables the ISP clock.

- **MCEN: Master Clock Enable**

0: No effect.

1: Enables the master clock.

49.6.8 ISC Clock Disable Register

Name: ISC_CLKDIS

Address: 0xF000801C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	MCSWRST	ICSWRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	MCDIS	ICDIS

- **ICDIS: ISP Clock Disable**

0: No effect.

1: Disables the ISP clock.

- **MCDIS: Master Clock Disable**

0: No effect.

1: Disables the master clock.

- **ICSWRST: ISP Clock Software Reset**

0: No effect.

1: Software reset the ISP clock.

- **MCSWRST: Master Clock Software Reset**

0: No effect.

1: Software reset the master clock.

49.6.9 ISC Clock Status Register

Name: ISC_CLKSR

Address: 0xF0008020

Access: Read-only

31	30	29	28	27	26	25	24
SIP	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	MCSR	ICSR

- **ICSR: ISP Clock Status Register**

0: The ISP clock is disabled.

1: The ISP clock is enabled.

- **MCSR: Master Clock Status Register**

0: The master clock is disabled.

1: The master clock is enabled.

- **SIP: Synchronization In Progress**

0: The double domain synchronization operation is over.

1: The double domain synchronization operation is in progress.

49.6.10 ISC Clock Configuration Register

Name: ISC_CLKCFG

Address: 0xF0008024

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	MCSEL	

23	22	21	20	19	18	17	16
MCDIV							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	ICSEL

7	6	5	4	3	2	1	0
ICDIV							

- **ICDIV: ISP Clock Divider**

$$f_{cc} = \frac{f_{ccref}}{ICDIV + 1}$$

- **ICSEL: ISP Clock Selection**

0: HCLOCK is selected.

1: ISCCLK is selected.

- **MCDIV: Master Clock Divider**

$$f_{mc} = \frac{f_{mcref}}{MCDIV + 1}$$

- **MCSEL: Master Clock Reference Clock Selection**

0: HCLOCK is selected.

1: ISCCLK is selected.

2: GCK is selected.

49.6.11 ISC Interrupt Enable Register

Name: ISC_INTEN

Address: 0xF0008028

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	CCIRERR	HDTO	VDTO	DAOV	VFPOV
23	22	21	20	19	18	17	16
–	–	–	RERR	–	–	–	WERR
15	14	13	12	11	10	9	8
–	–	HISCLR	HISDONE	–	–	LDONE	DDONE
7	6	5	4	3	2	1	0
–	–	DIS	SWRST	–	–	HD	VD

- **VD: Vertical Synchronization Detection Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **HD: Horizontal Synchronization Detection Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **SWRST: Software Reset Completed Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **DIS: Disable Completed Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **DDONE: DMA Done Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **LDONE: DMA List Done Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **HISDONE: Histogram Completed Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **HISCLR: Histogram Clear Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **WERR: Write Channel Error Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **RERR: Read Channel Error Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **VFPOV: Vertical Front Porch Overflow Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **DAOV: Data Overflow Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **VDTO: Vertical Synchronization Timeout Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **HDTO: Horizontal Synchronization Timeout Interrupt Enable**

0: No effect

1: The interrupt is enabled.

- **CCIRERR: CCIR Decoder Error Interrupt Enable**

0: No effect

1: The interrupt is enabled.

49.6.12 ISC Interrupt Disable Register

Name: ISC_INTDIS

Address: 0xF000802C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	CCIRERR	HDTO	VDTO	DAOV	VFPOV
23	22	21	20	19	18	17	16
–	–	–	RERR	–	–	–	WERR
15	14	13	12	11	10	9	8
–	–	HISCLR	HISDONE	–	–	LDONE	DDONE
7	6	5	4	3	2	1	0
–	–	DIS	SWRST	–	–	HD	VD

- **VD: Vertical Synchronization Detection Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **HD: Horizontal Synchronization Detection Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **SWRST: Software Reset Completed Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **DIS: Disable Completed Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **DDONE: DMA Done Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **LDONE: DMA List Done Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **HISDONE: Histogram Completed Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **HISCLR: Histogram Clear Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **WERR: Write Channel Error Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **RERR: Read Channel Error Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **VFPOV: Vertical Front Porch Overflow Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **DAOV: Data Overflow Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **VDTO: Vertical Synchronization Timeout Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **HDTO: Horizontal Synchronization Timeout Interrupt Disable**

0: No effect

1: The interrupt is disabled.

- **CCIRERR: CCIR Decoder Error Interrupt Disable**

0: No effect

1: The interrupt is disabled.

49.6.13 ISC Interrupt Mask Register

Name: ISC_INTMASK

Address: 0xF0008030

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	CCIRERR	HDTO	VDTO	DAOV	VFPOV
23	22	21	20	19	18	17	16
–	–	–	RERR	–	–	–	WERR
15	14	13	12	11	10	9	8
–	–	HISCLR	HISDONE	–	–	LDONE	DDONE
7	6	5	4	3	2	1	0
–	–	DIS	SWRST	–	–	HD	VD

- **VD: Vertical Synchronization Detection Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **HD: Horizontal Synchronization Detection Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **SWRST: Software Reset Completed Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **DIS: Disable Completed Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **DDONE: DMA Done Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **LDONE: DMA List Done Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **HISDONE: Histogram Completed Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **HISCLR: Histogram Clear Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **WERR: Write Channel Error Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **RERR: Read Channel Error Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **VFPOV: Vertical Front Porch Overflow Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **DAOV: Data Overflow Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **VDTO: Vertical Synchronization Timeout Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **HDTO: Horizontal Synchronization Timeout Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

- **CCIRERR: CCIR Decoder Error Interrupt Mask**

0: The interrupt is disabled.

1: The interrupt is enabled.

49.6.14 ISC Interrupt Status Register

Name: ISC_INTSR

Address: 0xF0008034

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	CCIRERR	HDTO	VDTO	DAOV	VFPOV

23	22	21	20	19	18	17	16
–	–	–	RERR	–	WERRID		WERR

15	14	13	12	11	10	9	8
–	–	HISCLR	HISDONE	–	–	LDONE	DDONE

7	6	5	4	3	2	1	0
–	–	DIS	SWRST	–	–	HD	VD

- **VD: Vertical Synchronization Detected Interrupt**

0: No vertical synchronization detection since the last read of the Interrupt Status register

1: A vertical synchronization has been detected.

- **HD: Horizontal Synchronization Detected Interrupt**

0: No horizontal synchronization detection since the last read of the Interrupt Status register

1: A horizontal synchronization has been detected.

- **SWRST: Software Reset Completed Interrupt**

0: No software reset completion since the last read of the Interrupt Status register

1: The software reset has completed.

- **DIS: Disable Completed Interrupt**

0: The disable has not occurred since the last read of the Interrupt Status register.

1: The disable has completed.

- **DDONE: DMA Done Interrupt**

0: No DMA Transfer Done Interrupt has occurred since the last read of the Interrupt Status register.

1: The DMA Transfer Done Interrupt has occurred.

- **LDONE: DMA List Done Interrupt**

0: No DMA List Done Interrupt has occurred since the last read of the Interrupt Status register.

1: The DMA List Done Interrupt has occurred.

- **HISDONE: Histogram Completed Interrupt**

0: No Histogram Completed Interrupt has been raised since the last read of the Interrupt Status register.

1: The Histogram Completed Interrupt has occurred.

- **HISCLR: Histogram Clear Interrupt**

0: No Histogram Clear Interrupt has been raised since the last read of the Interrupt Status register.

1: The Histogram Clear Interrupt has occurred.

- **WERR: Write Channel Error Interrupt**

0: No write channel error since the last read of the Interrupt Status register

1: A write channel error occurred.

- **WERRID: Write Channel Error Identifier**

Value	Name	Description
0	CH0	An error occurred for Channel 0 (RAW/RGB/Y)
1	CH1	An error occurred for Channel 1 (CbCr/Cb)
2	CH2	An error occurred for Channel 2 (Cr)
3	WB	Write back channel error

- **RERR: Read Channel Error Interrupt**

0: No read channel error since the last read of the Interrupt Status register

1: A read channel error occurred when the ISC read the descriptor.

- **VFPOV: Vertical Front Porch Overflow Interrupt**

0: No vertical front porch error occurred since the last read of the Interrupt Status register.

1: The vertical synchronization has been detected but the DMA channel is still busy.

- **DAOV: Data Overflow Interrupt**

0: No data overflow error occurred since the last reset of the Interrupt Status register.

1: A data overflow occurred.

- **VDTO: Vertical Synchronization Timeout Interrupt**

0: A vertical synchronization is detected.

1: No vertical synchronization is detected.

- **HDTO: Horizontal Synchronization Timeout Interrupt**

0: A horizontal synchronization is detected.

1: No horizontal synchronization is detected.

- **CCIRERR: CCIR Decoder Error Interrupt**

0: No CCIR CRC error detected since the last read of the Interrupt Status register

1: A CCIR CRC error has been detected.

49.6.15 ISC White Balance Control Register

Name: ISC_WB_CTRL

Address: 0xF0008058

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: White Balance Enable**

0: The white balance is disabled.

1: The white balance is enabled.

49.6.16 ISC White Balance Configuration Register

Name: ISC_WB_CFG

Address: 0xF000805C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	BAYCFG	

- **BAYCFG: White Balance Bayer Configuration (Pixel Color Pattern)**

Value	Name	Description
0	GRGR	Starting Row configuration is G R G R (Red Row)
1	RGRG	Starting Row configuration is R G R G (Red Row)
2	GBGB	Starting Row configuration is G B G B (Blue Row)
3	BGBG	Starting Row configuration is B G B G (Blue Row)

49.6.17 ISC White Balance Offset for R, GR Register

Name: ISC_WB_O_RGR

Address: 0xF0008060

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	GROFST				

23	22	21	20	19	18	17	16
GROFST							

15	14	13	12	11	10	9	8
–	–	–	ROFST				

7	6	5	4	3	2	1	0
ROFST							

- **ROFST: Offset Red Component (signed 13 bits 1:12:0)**
- **GROFST: Offset Green Component for Red Row (signed 13 bits 1:12:0)**

49.6.18 ISC White Balance Offset for B and GB Register

Name: ISC_WB_O_BGB

Address: 0xF0008064

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	GBOFST				

23	22	21	20	19	18	17	16
GBOFST							

15	14	13	12	11	10	9	8
–	–	–	BOFST				

7	6	5	4	3	2	1	0
BOFST							

- **BOFST: Offset Blue Component (signed 13 bits, 1:12:0)**
- **GBOFST: Offset Green Component for Blue Row (signed 13 bits, 1:12:0)**

49.6.19 ISC White Balance Gain for R, GR Register

Name: ISC_WB_G_RGR

Address: 0xF0008068

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	GRGAIN				

23	22	21	20	19	18	17	16
GRGAIN							

15	14	13	12	11	10	9	8
–	–	–	RGAIN				

7	6	5	4	3	2	1	0
RGAIN							

- **RGAIN: Red Component Gain (unsigned 13 bits, 0:4:9)**
- **GRGAIN: Green Component (Red row) Gain (unsigned 13 bits, 0:4:9)**

49.6.20 ISC White Balance Gain for B, GB Register

Name: ISC_WB_G_BGB

Address: 0xF000806C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	GBGAIN				

23	22	21	20	19	18	17	16
GBGAIN							

15	14	13	12	11	10	9	8
–	–	–	BGAIN				

7	6	5	4	3	2	1	0
BGAIN							

- **BGAIN: Blue Component Gain (unsigned 13 bits, 0:4:9)**
- **GBGAIN: Green Component (Blue row) Gain (unsigned 13 bits, 0:4:9)**

49.6.21 ISC Color Filter Array Control Register

Name: ISC_CFA_CTRL

Address: 0xF0008070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: Color Filter Array Interpolation Enable**

0: Color Filter Array Interpolation is disabled.

1: Color Filter Array Interpolation is enabled.

49.6.22 ISC Color Filter Array Configuration Register

Name: ISC_CFA_CFG

Address: 0xF0008074

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	EITPOL	–	–	BAYCFG	

- **BAYCFG: Color Filter Array Pattern**

Value	Name	Description
0	GRGR	Starting row configuration is G R G R (red row)
1	RGRG	Starting row configuration is R G R G (red row
2	GBGB	Starting row configuration is G B G B (blue row)
3	BGBG	Starting row configuration is B G B G (blue row)

- **EITPOL: Edge Interpolation**

0: Edges are not interpolated.

1: Edge interpolation is performed.

49.6.23 ISC Color Correction Control Register

Name: ISC_CC_CTRL

Address: 0xF0008078

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: Color Correction Enable**

0: Color correction is disabled.

1: Color correction is enabled.

49.6.24 ISC Color Correction RR RG Register

Name: ISC_CC_RR_RG

Address: 0xF000807C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	RGGAIN			

23	22	21	20	19	18	17	16
RGGAIN							

15	14	13	12	11	10	9	8
–	–	–	–	RRGAIN			

7	6	5	4	3	2	1	0
RRGAIN							

- **RRGAIN: Red Gain for Red Component (signed 12 bits, 1:3:8)**
- **RGGAIN: Green Gain for Red Component (signed 12 bits, 1:3:8)**

49.6.25 ISC Color Correction RB OR Register

Name: ISC_CC_RB_OR

Address: 0xF0008080

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	ROFST				

23	22	21	20	19	18	17	16
ROFST							

15	14	13	12	11	10	9	8
–	–	–	–	RBGAIN			

7	6	5	4	3	2	1	0
RBGAIN							

- **RBGAIN: Blue Gain for Red Component (signed 12 bits, 1:3:8)**
- **ROFST: Red Component Offset (signed 13 bits, 1:12:0)**

49.6.26 ISC Color Correction GR GG Register

Name: ISC_CC_GR_GG

Address: 0xF0008084

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	GGGAIN			

23	22	21	20	19	18	17	16
GGGAIN							

15	14	13	12	11	10	9	8
–	–	–	–	GRGAIN			

7	6	5	4	3	2	1	0
GRGAIN							

- **GRGAIN: Red Gain for Green Component (signed 12 bits, 1:3:8)**
- **GGGAIN: Green Gain for Green Component (signed 12 bits, 1:3:8)**

49.6.27 ISC Color Correction GB OG Register

Name: ISC_CC_GB_OG

Address: 0xF0008088

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	ROFST				

23	22	21	20	19	18	17	16
ROFST							

15	14	13	12	11	10	9	8
–	–	–	–	GBGAIN			

7	6	5	4	3	2	1	0
GBGAIN							

- **GBGAIN: Blue Gain for Green Component (signed 12 bits, 1:3:8)**
- **ROFST: Green Component Offset (signed 13 bits, 1:12:0)**

49.6.28 ISC Color Correction BR BG Register

Name: ISC_CC_BR_BG

Address: 0xF000808C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	BGGAIN			

23	22	21	20	19	18	17	16
BGGAIN							

15	14	13	12	11	10	9	8
–	–	–	–	BRGAIN			

7	6	5	4	3	2	1	0
BRGAIN							

- **BRGAIN: Red Gain for Blue Component (signed 12 bits, 1:3:8)**
- **BGGAIN: Green Gain for Blue Component (signed 12 bits, 1:3:8)**

49.6.29 ISC Color Correction BB OB Register

Name: ISC_CC_BB_OB

Address: 0xF0008090

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	BOFST				

23	22	21	20	19	18	17	16
BOFST							

15	14	13	12	11	10	9	8
–	–	–	–	BBGAIN			

7	6	5	4	3	2	1	0
BBGAIN							

- **BBGAIN: Blue Gain for Blue Component (signed 12 bits, 1:3:8)**
- **BOFST: Blue Component Offset (signed 13 bits, 1:12:0)**

49.6.30 ISC Gamma Correction Control Register

Name: ISC_GAM_CTRL

Address: 0xF0008094

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–			RENABLE	GENABLE	BENABLE	ENABLE

- **ENABLE: Gamma Correction Enable**

0: Gamma correction is disabled.

1: Gamma correction is enabled.

- **BENABLE: Gamma Correction Enable for B Channel**

0: 12 bits to 10 bits compression is performed skipping two bits.

1: Piecewise interpolation is used to perform 12 bits to 10 bits compression for the blue channel.

- **GENABLE: Gamma Correction Enable for G Channel**

0: 12 bits to 10 bits compression is performed skipping two bits.

1: Piecewise interpolation is used to perform 12 bits to 10 bits compression for the green channel.

- **RENABLE: Gamma Correction Enable for R Channel**

0: 12 bits to 10 bits compression is performed skipping two bits.

1: Piecewise interpolation is used to perform 12 bits to 10 bits compression for the red channel.

49.6.31 ISC Gamma Correction Blue Entry Register

Name: ISC_GAM_BENTRYx[x=0...63]

Address: 0xF0008098

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	BCONSTANT	

23	22	21	20	19	18	17	16
BCONSTANT							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	BSLOPE	

7	6	5	4	3	2	1	0
BSLOPE							

- **BSLOPE: Blue Color Slope for Piecewise Interpolation (signed 10 bits 1:3:6)**
- **BCONSTANT: Blue Color Constant for Piecewise Interpolation (unsigned 10 bits 0:10:0)**

49.6.32 ISC Gamma Correction Green Entry Register

Name: ISC_GAM_GENTRYx[x=0..63]

Address: 0xF0008198

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	GCONSTANT	

23	22	21	20	19	18	17	16
GCONSTANT							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	GSLOPE	

7	6	5	4	3	2	1	0
GSLOPE							

- **GSLOPE: Green Color Slope for Piecewise Interpolation (signed 10 bits 1:3:6)**
- **GCONSTANT: Green Color Constant for Piecewise Interpolation (unsigned 10 bits 0:10:0)**

49.6.33 ISC Gamma Correction Red Entry Register

Name: ISC_GAM_RENTRYx[x=0..63]

Address: 0xF0008298

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	RCONSTANT	

23	22	21	20	19	18	17	16
RCONSTANT							

15	14	13	12	11	10	9	8
–	–	–	–	–	–	RSLOPE	

7	6	5	4	3	2	1	0
RSLOPE							

- **RSLOPE: Red Color Slope for Piecewise Interpolation (signed 10 bits 1:3:6)**
- **RCONSTANT: Red Color Constant for Piecewise Interpolation (unsigned 10 bits 0:10:0)**

49.6.34 Color Space Conversion Control Register

Name: ISC_CSC_CTRL

Address: 0xF0008398

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: RGB to YCbCr Color Space Conversion Enable**

0: Color space conversion is disabled.

1: Color space conversion is enabled.

49.6.35 Color Space Conversion YR YG Register

Name: ISC_CSC_YR_YG

Address: 0xF000839C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	YGGAIN			

23	22	21	20	19	18	17	16
YGGAIN							

15	14	13	12	11	10	9	8
–	–	–	–	YRGAIN			

7	6	5	4	3	2	1	0
YRGAIN							

- **YRGAIN: Reg Gain for Luminance (signed 12 bits 1:3:8)**
- **YGGAIN: Green Gain for Luminance (signed 12 bits 1:3:8)**

49.6.36 Color Space Conversion YB OY Register

Name: ISC_CSC_YB_OY

Address: 0xF00083A0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	YOFST		

23	22	21	20	19	18	17	16
YOFST							

15	14	13	12	11	10	9	8
–	–	–	–	YBGAIN			

7	6	5	4	3	2	1	0
YBGAIN							

- **YBGAIN: Blue Gain for Luminance Component (12 bits signed 1:3:8)**
- **YOFST: Luminance Offset (11 bits signed 1:10:0)**

49.6.37 Color Space Conversion CBR CBG Register

Name: ISC_CSC_CBR_CBG

Address: 0xF00083A4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	CBGGAIN			

23	22	21	20	19	18	17	16
CBGGAIN							

15	14	13	12	11	10	9	8
–	–	–	–	CBRGAIN			

7	6	5	4	3	2	1	0
CBRGAIN							

- **CBRGAIN: Red Gain for Blue Chrominance (signed 12 bits, 1:3:8)**
- **CBGGAIN: Green Gain for Blue Chrominance (signed 12 bits 1:3:8)**

49.6.38 Color Space Conversion CBB OCB Register

Name: ISC_CSC_CBB_OCB

Address: 0xF00083A8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	CBOFST		

23	22	21	20	19	18	17	16
CBOFST							

15	14	13	12	11	10	9	8
–	–	–	–	CBBGAIN			

7	6	5	4	3	2	1	0
CBBGAIN							

- **CBBGAIN: Blue Gain for Blue Chrominance (signed 12 bits 1:3:8)**
- **CBOFST: Blue Chrominance Offset (signed 11 bits 1:10:0)**

49.6.39 Color Space Conversion CRR CRG Register

Name: ISC_CSC_CRR_CRG

Address: 0xF00083AC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	CRGGAIN			

23	22	21	20	19	18	17	16
CRGGAIN							

15	14	13	12	11	10	9	8
–	–	–	–	CRRGAIN			

7	6	5	4	3	2	1	0
CRRGAIN							

- **CRRGAIN: Red Gain for Red Chrominance (signed 12 bits 1:3:8)**
- **CRGGAIN: Green Gain for Red Chrominance (signed 12 bits 1:3:8)**

49.6.40 Color Space Conversion CRB OCR Register

Name: ISC_CSC_CRB_OCR

Address: 0xF00083B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	CROFST		

23	22	21	20	19	18	17	16
CROFST							

15	14	13	12	11	10	9	8
–	–	–	–	CRBGAIN			

7	6	5	4	3	2	1	0
CRBGAIN							

- **CRBGAIN: Blue Gain for Red Chrominance (signed 12 bits 1:3:8)**
- **CROFST: Red Chrominance Offset (signed 11 bits 1:10:0)**

49.6.41 Contrast And Brightness Control Register

Name: ISC_CBC_CTRL

Address: 0xF00083B4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: Contrast and Brightness Control Enable**

0: Contrast and brightness control is disabled.

1: Contrast and brightness control is enabled.

49.6.42 Contrast And Brightness Configuration Register

Name: ISC_CBC_CFG

Address: 0xF00083B8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	CCIRMODE		CCIR

- **CCIR: CCIR656 Stream Enable**

0: Raw mode

1: CCIR656 stream

- **CCIRMODE: CCIR656 Byte Ordering**

Value	Name	Description
0	CBY	Byte ordering Cb0, Y0, Cr0, Y1
1	CRY	Byte ordering Cr0, Y0, Cb0, Y1
2	YCB	Byte ordering Y0, Cb0, Y1, Cr0
3	YCR	Byte ordering Y0, Cr0, Y1, Cb0

49.6.43 Contrast And Brightness, Brightness Register

Name: ISC_CBC_BRIGHT

Address: 0xF00083BC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	BRIGHT		

7	6	5	4	3	2	1	0
BRIGHT							

- **BRIGHT: Brightness Control (signed 11 bits 1:10:0)**

49.6.44 Contrast And Brightness, Contrast Register

Name: ISC_CBC_CONTRAST

Address: 0xF00083C0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CONTRAST			
7	6	5	4	3	2	1	0
CONTRAST							

- **CONTRAST: Contrast (signed 12 bits 1:3:8)**

49.6.45 Subsampling 4:4:4 to 4:2:2 Control Register

Name: ISC_SUB422_CTRL

Address: 0xF00083C4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: 4:4:4 to 4:2:2 Chrominance Horizontal Subsampling Filter Enable**

0: Subsampler is disabled.

1: Subsampler is enabled.

49.6.46 Subsampling 4:4:4 to 4:2:2 Configuration Register

Name: ISC_SUB422_CFG

Address: 0xF00083C8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	FILTER		–	CCIRMODE		CCIR

- **CCIR: CCIR656 Input Stream**

0: Raw mode

1: CCIR mode

- **CCIRMODE: CCIR656 Byte Ordering**

Value	Name	Description
0	CBY	Byte ordering Cb0, Y0, Cr0, Y1
1	CRY	Byte ordering Cr0, Y0, Cb0, Y1
2	YCB	Byte ordering Y0, Cb0, Y1, Cr0
3	YCR	Byte ordering Y0, Cr0, Y1, Cb0

- **FILTER: Low Pass Filter Selection**

Value	Name	Description
0	FILT0CO	Cosited, {1}
1	FILT1CE	Centered {1, 1}
2	FILT2CO	Cosited {1,2,1}
3	FILT3CE	Centered {1, 3, 3, 1}

49.6.47 Subsampling 4:2:2 to 4:2:0 Control Register

Name: ISC_SUB420_CTRL

Address: 0xF00083CC

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	FILTER	–	–	–	ENABLE

- **ENABLE: 4:2:2 to 4:2:0 Vertical Subsampling Filter Enable (Center Aligned)**

0: Subsampler disabled

1: Subsampler enabled

- **FILTER: Interlaced or Progressive Chrominance Filter**

0: Progressive filter {0.5, 0.5}

1: Field-dependent filter, top field filter is {0.75, 0.25}, bottom field filter is {0.25, 0.75}

49.6.48 Rounding, Limiting and Packing Configuration Register

Name: ISC_RLP_CFG

Address: 0xF00083D0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
ALPHA							

7	6	5	4	3	2	1	0
–	–	–	–	MODE			

- **MODE: Rounding, Limiting and Packing Mode**

Value	Name	Description
0	DAT8	8-bit data
1	DAT9	9-bit data
2	DAT10	10-bit data
3	DAT11	11-bit data
4	DAT12	12-bit data
5	DATY8	8-bit luminance only
6	DATY10	10-bit luminance only
7	ARGB444	12-bit RGB+4-bit Alpha (MSB)
8	ARGB555	15-bit RGB+1-bit Alpha (MSB)
9	RGB565	16-bit RGB
10	ARGB32	24-bits RGB mode+8-bit Alpha
11	YYCC	YCbCr mode (full range, [0–255])
12	YYCC_LIMITED	YCbCr mode (limited range)

- **ALPHA: Alpha Value for Alpha-enabled RGB Mode**

49.6.49 Histogram Control Register

Name: ISC_HIS_CTRL

Address: 0xF00083D4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: Histogram Sub Module Enable**

0: Histogram disabled.

1: Histogram enabled.

49.6.50 Histogram Configuration Register

Name: ISC_HIS_CFG

Address: 0xF00083D8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	RAR
7	6	5	4	3	2	1	0
–	–	BAYSEL		–	MODE		

- **MODE: Histogram Operating Mode**

Value	Name	Description
0	Gr	Gr sampling
1	R	R sampling
2	Gb	Gb sampling
3	B	B sampling
4	Y	Luminance-only mode
5	RAW	Raw sampling
6	YCCIR656	Luminance only with CCIR656 10-bit or 8-bit mode

- **BAYSEL: Bayer Color Component Selection**

Value	Name	Description
0	GRGR	Starting row configuration is G R G R (red row)
1	RGRG	Starting row configuration is R G R G (red row)
2	GBGB	Starting row configuration is G B G B (blue row
3	BGBG	Starting row configuration is B G B G (blue row)

- **RAR: Histogram Reset After Read**

0: Reset after read mode is disabled

1: Reset after read mode is enabled

49.6.51 DMA Configuration Register

Name: ISC_DCFG

Address: 0xF00083E0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	CMBSIZE	

7	6	5	4	3	2	1	0
–	–	YMBSIZE		–	IMODE		

- **IMODE: DMA Input Mode Selection**

Value	Name	Description
0	PACKED8	8 bits, single channel packed
1	PACKED16	16 bits, single channel packed
2	PACKED32	32 bits, single channel packed
3	YC422SP	32 bits, dual channel
4	YC422P	32 bits, triple channel
5	YC420SP	32 bits, dual channel
6	YC420P	32 bits, triple channel

- **YMBSIZE: DMA Memory Burst Size Y channel**

Value	Name	Description
0	SINGLE	DMA single access
1	BEATS4	4-beat burst access
2	BEATS8	8-beat burst access
3	BEATS16	16-beat burst access

- **CMBSIZE: DMA Memory Burst Size C channel**

Value	Name	Description
0	SINGLE	DMA single access
1	BEATS4	4-beat burst access
2	BEATS8	8-beat burst access
3	BEATS16	16-beat burst access

49.6.52 DMA Control Register

Name: ISC_DCTRL

Address: 0xF00083E4

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
DONE	FIELD	WB	IE	–	DVIEW		DE

- **DE: Descriptor Enable**

0: Descriptor disabled

1: Descriptor enabled

- **DVIEW: Descriptor View**

Value	Name	Description
0	PACKED	Address {0} Stride {0} are updated
1	SEMIPLANAR	Address {0,1} Stride {0,1} are updated
2	PLANAR	Address {0,1,2} Stride {0,1,2} are updated

- **IE: Interrupt Enable**

0: DMA Done interrupt is generated.

1: DMA Done interrupt is not set.

- **WB: Write Back Operation Enable**

0: Write Back operation is skipped.

1: Write Back operation is performed.

- **FIELD: Value of Captured Frame Field Signal**(1)(2)

0: Field value is 0.

1: Field value is 1.

- **DONE: Descriptor Processing Status**(2)

0: Descriptor not processed yet

1: Descriptor processed

Notes: 1. Only relevant for interlaced content.

2. Appears only in descriptor located in memory. Can be used only if WB (Write Back) is set.

49.6.53 DMA Descriptor Address Register

Name: ISC_DNDA

Address: 0xF00083E8

Access: Read/Write

31	30	29	28	27	26	25	24
NDA							

23	22	21	20	19	18	17	16
NDA							

15	14	13	12	11	10	9	8
NDA							

7	6	5	4	3	2	1	0
NDA						–	–

- **NDA: Next Descriptor Address Register**

49.6.54 DMA Address 0 Register

Name: ISC_DAD0

Address: 0xF00083EC

Access: Read/Write

31	30	29	28	27	26	25	24
AD0							

23	22	21	20	19	18	17	16
AD0							

15	14	13	12	11	10	9	8
AD0							

7	6	5	4	3	2	1	0
AD0							

- **AD0: Channel 0 Address**

49.6.55 DMA Stride 0 Register

Name: ISC_DST0

Address: 0xF00083F0

Access: Read/Write

31	30	29	28	27	26	25	24
ST0							

23	22	21	20	19	18	17	16
ST0							

15	14	13	12	11	10	9	8
ST0							

7	6	5	4	3	2	1	0
ST0							

- **ST0: Channel 0 Stride**

49.6.56 DMA Address 1 Register

Name: ISC_DAD1

Address: 0xF00083F4

Access: Read/Write

31	30	29	28	27	26	25	24
AD1							

23	22	21	20	19	18	17	16
AD1							

15	14	13	12	11	10	9	8
AD1							

7	6	5	4	3	2	1	0
AD1							

- **AD1: Channel 1 Address**

49.6.57 DMA Stride 1 Register

Name: ISC_DST1

Address: 0xF00083F8

Access: Read/Write

31	30	29	28	27	26	25	24
ST1							

23	22	21	20	19	18	17	16
ST1							

15	14	13	12	11	10	9	8
ST1							

7	6	5	4	3	2	1	0
ST1							

- **ST1: Channel 1 Stride**

49.6.58 DMA Address 2 Register

Name: ISC_DAD2

Address: 0xF00083FC

Access: Read/Write

31	30	29	28	27	26	25	24
AD2							

23	22	21	20	19	18	17	16
AD2							

15	14	13	12	11	10	9	8
AD2							

7	6	5	4	3	2	1	0
AD2							

- **AD2: Channel 2 Address**

49.6.59 DMA Stride 2 Register

Name: ISC_DST2

Address: 0xF0008400

Access: Read/Write

31	30	29	28	27	26	25	24
ST2							

23	22	21	20	19	18	17	16
ST2							

15	14	13	12	11	10	9	8
ST2							

7	6	5	4	3	2	1	0
ST2							

- **ST2: Channel 2 Stride**

49.6.60 Histogram Entry

Name: ISC_HIS_ENTRYx [x=0..511]

Address: 0xF0008410

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	COUNT			

15	14	13	12	11	10	9	8
COUNT							

7	6	5	4	3	2	1	0
COUNT							

- **COUNT: Entry Counter**

50. Controller Area Network (MCAN)

50.1 Description

The Controller Area Network (MCAN) performs communication according to ISO11898-1 (Bosch CAN specification 2.0 part A,B) and to Bosch CAN FD specification V1.0. Additional transceiver hardware is required for connection to the physical layer.

All functions concerning the handling of messages are implemented by the Rx Handler and the Tx Handler. The Rx Handler manages message acceptance filtering, the transfer of received messages from the CAN core to the Message RAM, as well as providing receive message status information. The Tx Handler is responsible for the transfer of transmit messages from the Message RAM to the CAN core, as well as providing transmit status information.

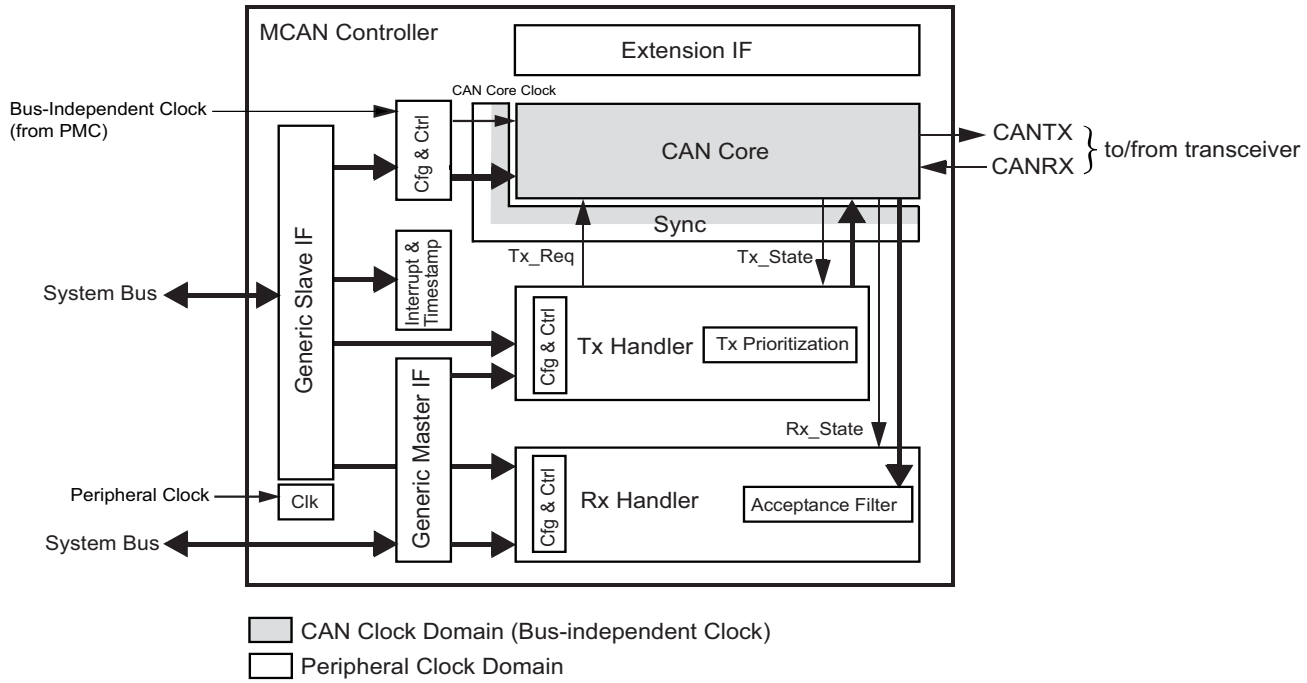
Acceptance filtering is implemented by a combination of up to 128 filter elements, where each element can be configured as a range, as a bit mask, or as a dedicated ID filter.

50.2 Embedded Characteristics

- Compliant with CAN Protocol Version 2.0 Part A, B and ISO 11898-1
- CAN FD with up to 64 Data Bytes Supported
- CAN Error Logging
- AUTOSAR Optimized
- SAE J1939 Optimized
- Improved Acceptance Filtering
- Two Configurable Receive FIFOs
- Separate Signalling on Reception of High Priority Messages
- Up to 64 Dedicated Receive Buffers
- Up to 32 Dedicated Transmit Buffers
- Configurable Transmit FIFO
- Configurable Transmit Queue
- Configurable Transmit Event FIFO
- Direct Message RAM Access for Processor
- Multiple MCANs May Share the Same Message RAM
- Programmable Loop-back Test Mode
- Maskable Module Interrupts
- Support for Asynchronous CAN and System Bus Clocks
- Power-down Support
- Debug on CAN Support

50.3 Block Diagram

Figure 50-1. MCAN Block Diagram



Note: Refer to section “Power Management Controller (PMC)” for details about the bus-independent clock (GCLK).

50.4 Product Dependencies

50.4.1 I/O Lines

The pins used to interface to the compliant external devices can be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the CAN pins to their peripheral functions.

Table 50-1. I/O Lines

Instance	Signal	I/O Line	Peripheral
MCAN0	CANRX0	PC2	C
MCAN0	CANRX0	PC11	E
MCAN0	CANTX0	PC1	C
MCAN0	CANTX0	PC10	E
MCAN1	CANRX1	PC27	D
MCAN1	CANTX1	PC26	D

50.4.2 Power Management

The MCAN can be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the MCAN clock.

In order to achieve a stable function of the MCAN, the system bus clock must always be faster than or equal to the CAN clock.

It is recommended to use the CAN clock at frequencies of 20, 40 or 80 MHz. To achieve these frequencies, PMC GCLK must select the UPLLCK (480 MHz) as source clock and divide by 24,12, or 6. GCLK allows the system bus and processor clock to be modified without affecting the bit rate communication.

50.4.3 Interrupt Sources

The two MCAN interrupt lines (MCAN_INT0, MCAN_INT1) are connected on internal sources of the Interrupt Controller.

Using the MCAN interrupts requires the Interrupt Controller to be programmed first.

Interrupt sources can be routed either to MCAN_INT0 or to MCAN_INT1. By default, all interrupt sources are routed to interrupt line MCAN_INT0/1. By programming MCAN_ILE.EINT0 and MCAN_ILE.EINT1, the interrupt sources can be enabled or disabled separately.

Table 50-2. Peripheral IDs

Instance	ID
MCAN0	56
MCAN1	57

50.4.4 Address Configuration

The LSBs [bits 15:2] for each section of the CAN Message RAM are configured in the respective buffer configuration registers.

The MSBs [bits 31:16] of the CAN Message RAM for CAN0 and CAN1 are configured in 0x00200000.

50.5 Functional Description

50.5.1 Operating Modes

50.5.1.1 Software Initialization

Software initialization is started by setting bit MCAN_CCCR.INIT, either by software or by a hardware reset, when an uncorrected bit error was detected in the Message RAM, or by going Bus_Off. While MCAN_CCCR.INIT is set, message transfer from and to the CAN bus is stopped and the status of the CAN bus output CANTX is recessive (HIGH). The counters of the Error Management Logic EML are unchanged. Setting MCAN_CCCR.INIT does not change any configuration register. Resetting MCAN_CCCR.INIT finishes the software initialization. Afterwards the Bit Stream Processor BSP synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (≡ Bus_Idle) before it can take part in bus activities and start the message transfer.

Access to the MCAN configuration registers is only enabled when both bits MCAN_CCCR.INIT and MCAN_CCCR.CCE are set (protected write).

MCAN_CCCR.CCE can only be configured when MCAN_CCCR.INIT = ‘1’. MCAN_CCCR.CCE is automatically cleared when MCAN_CCCR.INIT = ‘0’.

The following registers are cleared when MCAN_CCCR.CCE = ‘1’:

- High Priority Message Status (MCAN_HPMS)
- Receive FIFO 0 Status (MCAN_RXF0S)
- Receive FIFO 1 Status (MCAN_RXF1S)
- Transmit FIFO/Queue Status (MCAN_TXFQS)
- Transmit Buffer Request Pending (MCAN_TXBRP)
- Transmit Buffer Transmission Occurred (MCAN_TXBTO)
- Transmit Buffer Cancellation Finished (MCAN_TXBCF)
- Transmit Event FIFO Status (MCAN_TXEFS)

The Timeout Counter value MCAN_TOCV.TOC is loaded with the value configured by MCAN_TOCC.TOP when MCAN_CCCR.CCE = ‘1’.

In addition, the state machines of the Tx Handler and Rx Handler are held in idle state while MCAN_CCCR.CCE = ‘1’.

The following registers are only writeable while MCAN_CCCR.CCE = ‘0’

- Transmit Buffer Add Request (MCAN_TXBAR)
- Transmit Buffer Cancellation Request (MCAN_TXBCR)

MCAN_CCCR.TEST and MCAN_CCCR.MON can only be set when MCAN_CCCR.INIT = ‘1’ and MCAN_CCCR.CCE = ‘1’. Both bits may be cleared at any time. MCAN_CCCR.DAR can only be configured when MCAN_CCCR.INIT = ‘1’ and MCAN_CCCR.CCE = ‘1’.

50.5.1.2 Normal Operation

Once the MCAN is initialized and MCAN_CCCR.INIT is cleared, the MCAN synchronizes itself to the CAN bus and is ready for communication.

After passing the acceptance filtering, received messages including Message ID and DLC are stored into a dedicated Rx Buffer or into Rx FIFO 0 or Rx FIFO 1.

For messages to be transmitted, dedicated Tx Buffers and/or a Tx FIFO or a Tx Queue can be initialized or updated. Automated transmission on reception of remote frames is not implemented.

50.5.1.3 CAN FD Operation

There are two variants in the CAN FD frame format, first the CAN FD frame without bit rate switching where the data field of a CAN frame may be longer than 8 bytes. The second variant is the CAN FD frame where control field, data field, and CRC field of a CAN frame are transmitted with a higher bit rate than the beginning and the end of the frame.

The previously reserved bit in CAN frames with 11-bit identifiers and the first previously reserved bit in CAN frames with 29-bit identifiers will now be decoded as FDF bit. FDF = recessive signifies a CAN FD frame, FDF = dominant signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF, res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. The coding of res = recessive is reserved for future expansion of the protocol. In case the MCAN receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting bit MCAN_PSR.PXE. When Protocol Exception Handling is enabled (MCAN_CCCR.PXHD = 0), this causes the operation state to change from Receiver (MCAN_PSR.ACT = 2) to Integrating (MCAN_PSR.ACT = 00) at the next sample point. In case Protocol Exception Handling is disabled (MCAN_CCCR.PXHD = 1), the MCAN will treat a recessive res bit as an form error and will respond with an error frame.

CAN FD operation is enabled by programming CCCR.FDOE. In case CCCR.FDOE = ‘1’, transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via bit FDF in the respective Tx Buffer element. With CCCR.FDOE = ‘0’, received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if bit FDF of a Tx Buffer element is set. CCCR.FDOE and CCCR.BRSE can only be changed while CCCR.INIT and CCCR.CCE are both set.

With MCAN_CCCR.FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 0, only bit FDF of a Tx Buffer element is evaluated. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significant higher than in the CAN FD arbitration phase. In this case disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting according to ISO11898-1 until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wakeup messages in CAN Partial Networking have to be transmitted in Classic CAN format.
- End-of-line programming in case not all nodes are CAN FD-capable. Non-CAN FD nodes are held in Silent mode until programming has completed. Then all nodes revert to Classic CAN communication.

In the CAN FD format, the coding of the DLC differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN, the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to Table 50-3 below.

Table 50-3. Coding of DLC in CAN FD

DLC	9	10	11	12	13	14	15
Number of Data Bytes	12	16	20	24	32	48	64

In CAN FD frames, the bit timing will be switched inside the frame, after the BRS (Bit Rate Switch) bit, if this bit is recessive. Before the BRS bit, in the CAN FD arbitration phase, the nominal CAN bit timing is used as defined by the Nominal Bit Timing and Prescaler register (MCAN_NBTP). In the following CAN FD data phase, the fast CAN bit timing is used as defined by the Data Bit Timing and Prescaler register (MCAN_DBTP). The bit timing reverts back from the fast timing at the CRC delimiter or when an error is detected, whichever occurs first.

The maximum configurable bit rate in the CAN FD data phase depends on the CAN core clock frequency. Example: with a CAN clock frequency of 20 MHz and the shortest configurable bit time of 4 t_q, the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the bit ESI (Error Status Indicator) is determined by the transmitter's error state at the start of the transmission. If the transmitter is error passive, ESI is transmitted recessive, else it is transmitted dominant.

50.5.1.4 Transmitter Delay Compensation

During the data phase of a CAN FD transmission only one node is transmitting, all others are receivers. The length of the bus line has no impact. When transmitting via pin CANTX the protocol controller receives the transmitted data from its local CAN transceiver via pin CANRX. The received data is delayed by the transmitter delay. In case this delay is greater than NTSEG1 (time segment before sample point), a bit error is detected. In order to enable a data phase bit time that is even shorter than the transmitter delay, the delay compensation is introduced. Without transmitter delay compensation, the bit rate in the data phase of a CAN FD frame is limited by the transmitter delay.

Description

The MCAN protocol unit has implemented a delay compensation mechanism to compensate the transmitter delay, thereby enabling transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver.

To check for bit errors during the data phase, the delayed transmit data is compared against the received data at the secondary sample point. If a bit error is detected, the transmitter will react to this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The transmitter delay compensation enables configurations where the data bit time is shorter than the transmitter delay, it is described in detail in the new ISO11898-1. It is enabled by setting bit MCAN_DBTP.TDC.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the MCAN's transmit output CANTX through the transceiver to the receive input CANRX plus the transmitter delay compensation offset as configured by MCAN_TDCR.TDCO. The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (e.g. half of the bit time in the data phase). The position of the secondary sample point is rounded down to the next integer number of CAN core clock periods.

MCAN_PSR.TDCV shows the actual transmitter delay compensation value. MCAN_PSR.TDCV is cleared when MCAN_CCCR.INIT is set and is updated at each transmission of an FD frame while MCAN_DBTP.TDC is set.

The following boundary conditions have to be considered for the transmitter delay compensation implemented in the MCAN:

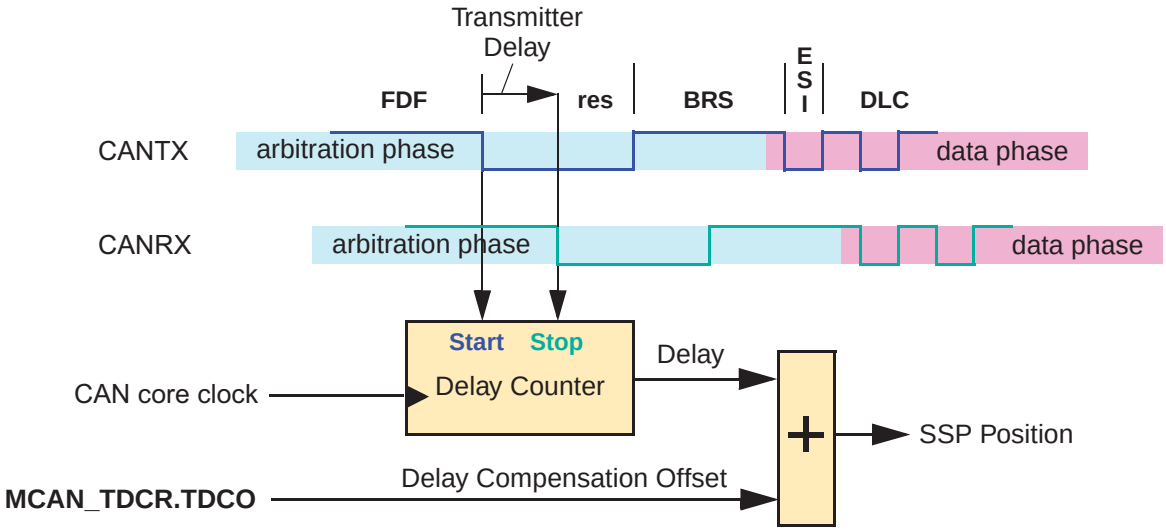
- The sum of the measured delay from CANTX to CANRX and the configured transceiver delay compensation offset MCAN_TDCR.TDCO has to be less than 6 bit times in the data phase.
- The sum of the measured delay from CANTX to CANRX and the configured transceiver delay compensation offset MCAN_TDCR.TDCO has to be less or equal 127 CAN core clock periods. In case this sum exceeds 127 CAN core clock periods, the maximum value of 127 CAN core clock periods is used for transceiver delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

Transmitter Delay Measurement

If transmitter delay compensation is enabled by programming MCAN_DBTP.TDC = '1', the measurement is started within each transmitted CAN FD frame at the falling edge of bit FDF to bit res. The measurement is stopped when this edge is seen at the receive input CANRX of the transmitter.

The resolution of this measurement is one mtq.

Figure 50-2. Transmitter Delay Measurement



To avoid that a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in a to early SSP position, the use of a transmitter delay compensation filter window can be enabled by programming MCAN_TDCR.TDCF.

This defines a minimum value for the SSP position. Dominant edges on CANRX, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least MCAN_TDCR.TDCF AND CANRX is low.

50.5.1.5 Restricted Operation Mode

In Restricted Operation mode, the node is able to receive data and remote frames and to give acknowledge to valid frames, but it does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, it does not send dominant bits, instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The error counters are not incremented. The processor can set the MCAN into Restricted Operation mode by setting bit MCAN_CCCR.ASM. The bit can only be set by the processor when both MCAN_CCCR.CCE and MCAN_CCCR.INIT are set to '1'. The bit can be reset by the processor at any time.

Restricted Operation mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

The Restricted Operation mode can be used in applications that adapt themselves to different CAN bit rates. In this case the application tests different bit rates and leaves the Restricted Operation mode after it has received a valid frame.

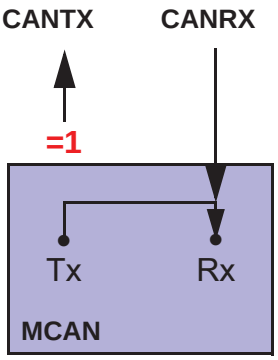
Note: The Restricted Operation Mode must not be combined with the Loop Back mode (internal or external).

50.5.1.6 Bus Monitoring Mode

The MCAN is set in Bus Monitoring mode by setting MCAN_CCCR.MON. In Bus Monitoring mode (see ISO11898-1, 10.12 Bus monitoring), the MCAN is able to receive valid data frames and valid remote frames, but cannot start a transmission. In this mode, it sends only recessive bits on the CAN bus. If the MCAN is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MCAN monitors this dominant bit, although the CAN bus may remain in recessive state. In Bus Monitoring mode, the Tx Buffer Request Pending register (MCAN_TXBRP) is held in reset state.

The Bus Monitoring mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. Figure 50-4 shows the connection of signals CANTX and CANRX to the MCAN in Bus Monitoring mode.

Figure 50-3. Pin Control in Bus Monitoring Mode



Bus Monitoring Mode

50.5.1.7 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898-1, 6.3.3 Recovery Management), the MCAN provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled. To support time-triggered communication as described in ISO 11898-1, chapter 9.2, the automatic retransmission may be disabled via MCAN_CCCR.DAR.

Frame Transmission in DAR Mode

In DAR mode, all transmissions are automatically cancelled after they start on the CAN bus. A Tx Buffer's Tx Request Pending bit TXBRP.TRPx is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

- Successful transmission:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set
  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx not set
- Successful transmission in spite of cancellation:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx set
  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set
- Arbitration lost or frame transmission disturbed:
  Corresponding Tx Buffer Transmission Occurred bit MCAN_TXBTO.TOx not set
  Corresponding Tx Buffer Cancellation Finished bit MCAN_TXBCF.CFx set

In case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = “10” (transmission in spite of cancellation).

50.5.1.8 Power-down (Sleep Mode)

The MCAN can be set into Power-down mode via bit MCAN_CCCR.CSR.

When all pending transmission requests have completed, the MCAN waits until bus idle state is detected. Then the MCAN sets MCAN_CCCR.INIT to prevent any further CAN transfers. Now the MCAN acknowledges that it is ready for power down by setting to one the bit MCAN_CCCR.CSA. In this state, before the clocks are switched off, further register accesses can be made. A write access to MCAN_CCCR.INIT will have no effect. Now the bus clock (peripheral clock) and the CAN core clock may be switched off.

To leave Power-down mode, the application has to turn on the MCAN clocks before clearing CC Control Register flag MCAN_CCCR.CSR. The MCAN will acknowledge this by clearing MCAN_CCCR.CSA. The application can then restart CAN communication by clearing the bit CCCR.INIT.

50.5.1.9 Test Modes

To enable write access to the MCAN Test register (MCAN_TEST) (see Section 50.6.5), bit MCAN_CCCR.TEST must be set. This allows the configuration of the test modes and test functions.

Four output functions are available for the CAN transmit pin CANTX by programming MCAN_TEST.TX. Additionally to its default function – the serial data output – it can drive the CAN Sample Point signal to monitor the MCAN's bit timing and it can drive constant dominant or recessive values. The actual value at pin CANRX can be read from MCAN_TEST.RX. Both functions can be used to check the CAN bus' physical layer.

Due to the synchronization mechanism between CAN clock and system bus clock domain, there may be a delay of several system bus clock periods between writing to MCAN_TEST.TX until the new configuration is visible at output pin CANTX. This applies also when reading input pin CANRX via MCAN_TEST.RX.

Note: Test modes should be used for production tests or self-test only. The software control for pin CANTX interferes with all CAN protocol functions. It is not recommended to use test modes for application.

External Loop Back Mode

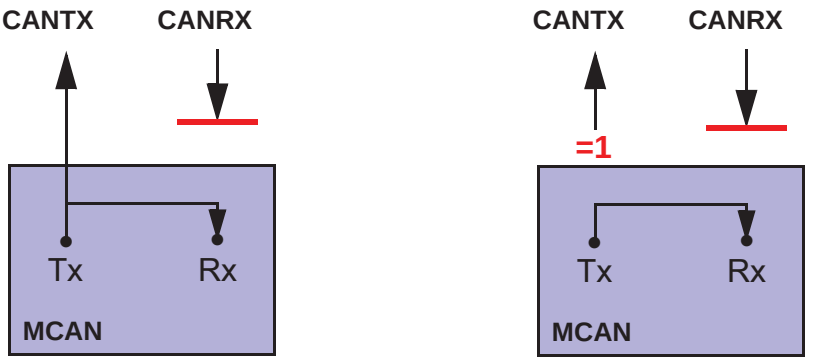
The MCAN can be set in External Loop Back mode by setting the bit MCAN_TEST.LBCK. In Loop Back mode, the MCAN treats its own transmitted messages as received messages and stores them (if they pass acceptance filtering) into an Rx Buffer or an Rx FIFO. Figure 50-4 shows the connection of signals CANTX and CANRX to the MCAN in External Loop Back mode.

This mode is provided for hardware self-test. To be independent from external stimulation, the MCAN ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in Loop Back mode. In this mode, the MCAN performs an internal feedback from its Tx output to its Rx input. The actual value of the CANRX input pin is disregarded by the MCAN. The transmitted messages can be monitored at the CANTX pin.

Internal Loop Back Mode

Internal Loop Back mode is entered by setting bits MCAN_TEST.LBCK and MCAN_CCCR.MON. This mode can be used for a "Hot Selftest", meaning the MCAN can be tested without affecting a running CAN system connected to the pins CANTX and CANRX. In this mode, pin CANRX is disconnected from the MCAN, and pin CANTX is held recessive. Figure 50-4 shows the connection of CANTX and CANRX to the MCAN when Internal Loop Back mode is enabled.

Figure 50-4. Pin Control in Loop Back Modes



External Loop Back Mode Internal Loop Back Mode

50.5.2 Timestamp Generation

For timestamp generation the MCAN supplies a 16-bit wrap-around counter. A prescaler TSCC.TCP can be configured to clock the counter in multiples of CAN bit times (1...16). The counter is readable via MCAN_TSCV.TSC. A write access to the Timestamp Counter Value register (MCAN_TSCV) resets the counter to zero. When the timestamp counter wraps around, interrupt flag MCAN_IR.TSW is set.

On start of frame reception / transmission the counter value is captured and stored into the timestamp section of an Rx Buffer / Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element.

By programming bit MCAN_TSCC.TSS an external 16-bit timestamp can be used.

50.5.3 Timeout Counter

To signal timeout conditions for Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO, the MCAN supplies a 16-bit Timeout Counter. It operates as down-counter and uses the same prescaler controlled by TSCC.TCP as the Timestamp Counter. The Timeout Counter is configured via the Timeout Counter Configuration register (MCAN_TOCC). The actual counter value can be read from MCAN_TOCV.TOC. The Timeout Counter can only be started while MCAN_CCCR.INIT = '0'. It is stopped when MCAN_CCCR.INIT = '1', e.g. when the MCAN enters Bus_Off state.

The operating mode is selected by MCAN_TOCC.TOS. When operating in Continuous mode, the counter starts when MCAN_CCCR.INIT is reset. A write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting.

When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored. Writing to MCAN_TOCV has no effect.

When the counter reaches zero, interrupt flag MCAN_IR.TOO is set. In Continuous mode, the counter is immediately restarted at MCAN_TOCC.TOP.

Note: The clock signal for the Timeout Counter is derived from the CAN Core's sample point signal. Therefore the point in time where the Timeout Counter is decremented may vary due to the synchronization / re-synchronization mechanism of the CAN Core. If the bit rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

50.5.4 Rx Handling

The Rx Handler controls the acceptance filtering, the transfer of received messages to the Rx Buffers or to one of the two Rx FIFOs, as well as the Rx FIFO's Put and Get Indices.

50.5.4.1 Acceptance Filtering

The MCAN offers the possibility to configure two sets of acceptance filters, one for standard identifiers and one for extended identifiers. These filters can be assigned to an Rx Buffer or to Rx FIFO 0,1. For acceptance filtering each list of filters is executed from element #0 until the first matching element. Acceptance filtering stops at the first matching element. The following filter elements are not evaluated for this message.

The main features are:

- Each filter element can be configured as
  - range filter (from - to)
  - filter for one or two dedicated IDs
  - classic bit mask filter
- Each filter element is configurable for acceptance or rejection filtering
- Each filter element can be enabled / disabled individually
- Filters are checked sequentially, execution stops with the first matching filter element

Related configuration registers are:

- Global Filter Configuration (MCAN_GFC)
- Standard ID Filter Configuration (MCAN_SIDFC)
- Extended ID Filter Configuration (MCAN_XIDFC)
- Extended ID and Mask (MCAN_XIDAM)

Depending on the configuration of the filter element (SFEC/EFEC) a match triggers one of the following actions:

- Store received frame in FIFO 0 or FIFO 1
- Store received frame in Rx Buffer
- Store received frame in Rx Buffer and generate pulse at filter event pin
- Reject received frame
- Set High Priority Message interrupt flag (MCAN_IR.HPM)
- Set High Priority Message interrupt flag (MCAN_IR.HPM) and store received frame in FIFO 0 or FIFO 1

Acceptance filtering is started after the complete identifier has been received. After acceptance filtering has completed, and if a matching Rx Buffer or Rx FIFO has been found, the Message Handler starts writing the received message data in portions of 32 bit to the matching Rx Buffer or Rx FIFO. If the CAN protocol controller has detected an error condition (e.g. CRC error), this message is discarded with the following impact on the effected Rx Buffer or Rx FIFO:

- Rx Buffer
  New Data flag of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC.
- Rx FIFO
  Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data. For error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC. In case the matching Rx FIFO is operated in Overwrite mode, the boundary conditions described in “Rx FIFO Overwrite Mode” have to be considered.

Note: When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

Range Filter

The filter matches for all received frames with Message IDs in the range defined by SF1ID/SF2ID resp. EF1ID/EF2ID.

There are two possibilities when range filtering is used together with extended frames:

EFT = “00”: The Message ID of received frames is ANDed with MCAN_XIDAM before the range filter is applied.

EFT = “11”: MCAN_XIDAM is not used for range filtering.

Filter for Specific IDs

A filter element can be configured to filter for one or two specific Message IDs. To filter for one specific Message ID, the filter element has to be configured with SF1ID = SF2ID resp. EF1ID = EF2ID.

Classic Bit Mask Filter

Classic bit mask filtering is intended to filter groups of Message IDs by masking single bits of a received Message ID. With classic bit mask filtering SF1ID/EF1ID is used as Message ID filter, while SF2ID/EF2ID is used as filter mask.

A zero bit at the filter mask will mask out the corresponding bit position of the configured ID filter, e.g. the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are one are relevant for acceptance filtering.

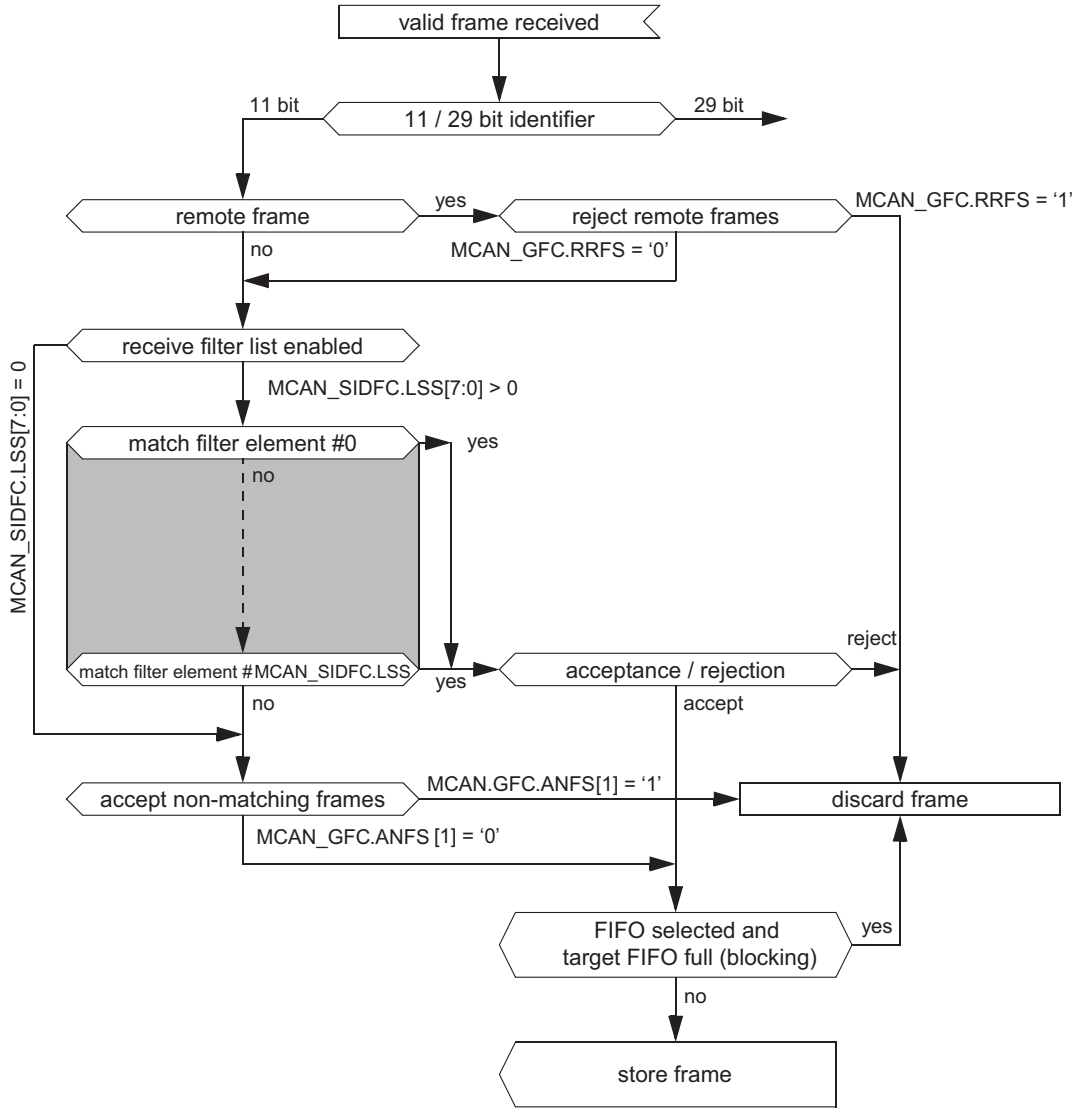
In case all mask bits are one, a match occurs only when the received Message ID and the Message ID filter are identical. If all mask bits are zero, all Message IDs match.

Standard Message ID Filtering

Figure 50-5 below shows the flow for standard Message ID (11-bit Identifier) filtering. The Standard Message ID Filter element is described in Section 50.5.7.5.

Controlled by MCAN_GFC and MCAN_SIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

Figure 50-5. Standard Message ID Filter Path



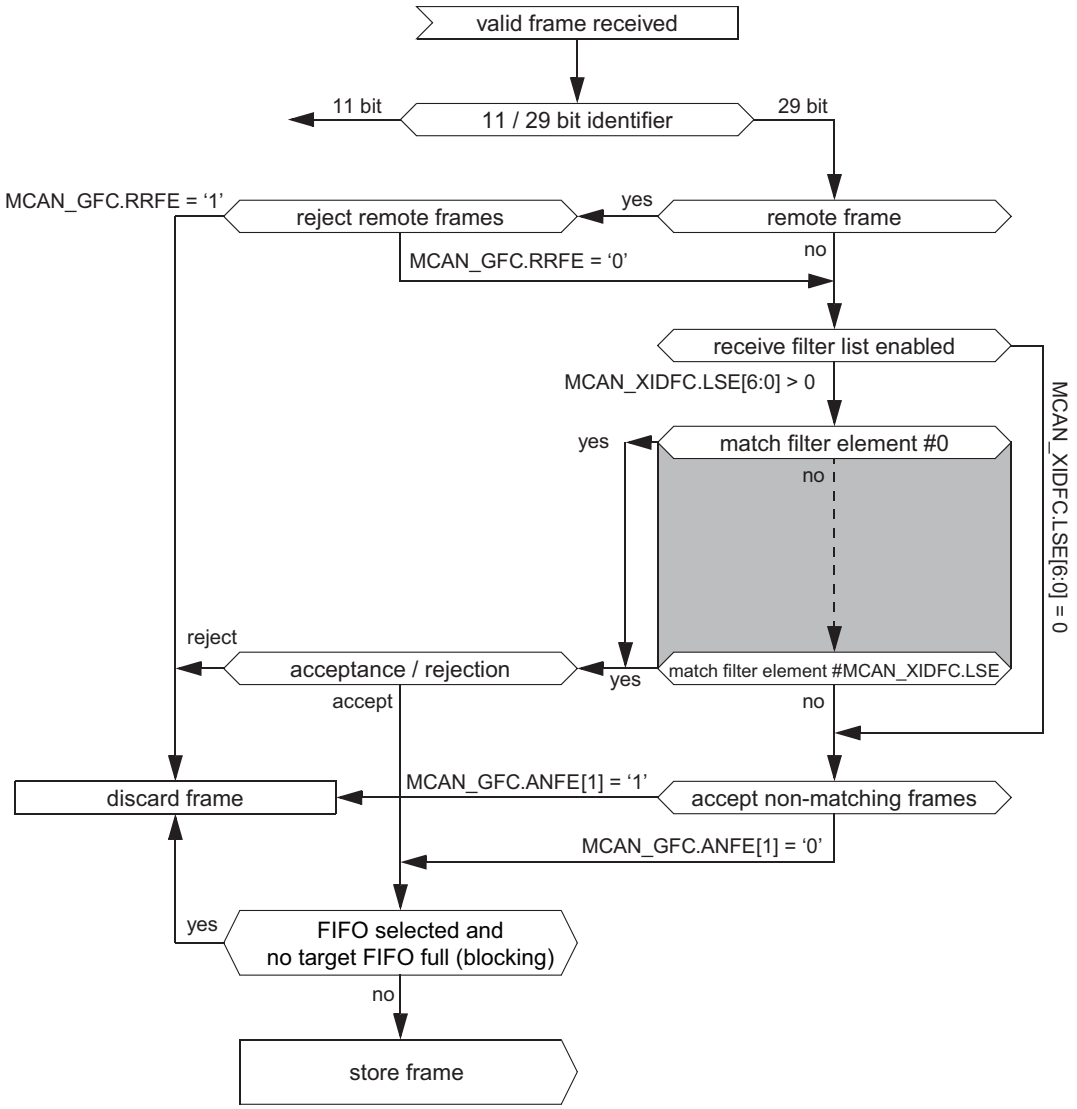
Extended Message ID Filtering

Figure 50-6 below shows the flow for extended Message ID (29-bit Identifier) filtering. The Extended Message ID Filter element is described in Section 50.5.7.6.

Controlled by MCAN_GFC and MCAN_XIDFC Message ID, Remote Transmission Request bit (RTR), and the Identifier Extension bit (IDE) of received frames are compared against the list of configured filter elements.

MCAN_XIDAM is ANDed with the received identifier before the filter list is executed.

Figure 50-6. Extended Message ID Filter Path



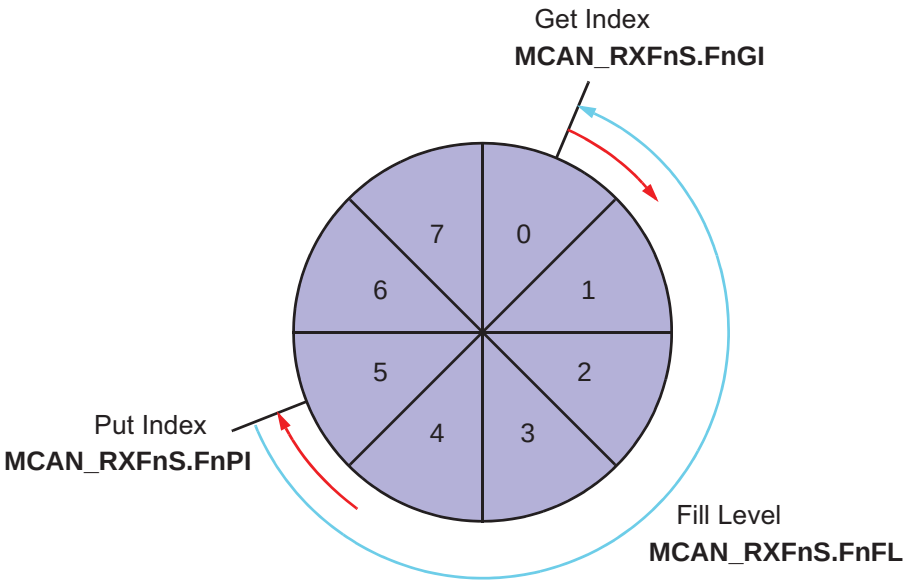
50.5.4.2 Rx FIFOs

Rx FIFO 0 and Rx FIFO 1 can be configured to hold up to 64 elements each. Configuration of the two Rx FIFOs is done via the Rx FIFO 0 Configuration register (MCAN_RXF0C) and the Rx FIFO 1 Configuration register (MCAN_RXF1C).

Received messages that passed acceptance filtering are transferred to the Rx FIFO as configured by the matching filter element. For a description of the filter mechanisms available for Rx FIFO 0 and Rx FIFO 1, see Section 50.5.4.1. The Rx FIFO element is described in Section 50.5.7.2.

To avoid an Rx FIFO overflow, the Rx FIFO watermark can be used. When the Rx FIFO fill level reaches the Rx FIFO watermark configured by MCAN_RXFnC.FnWM, interrupt flag MCAN_IR.RFnW is set. When the Rx FIFO Put Index reaches the Rx FIFO Get Index, an Rx FIFO Full condition is signalled by MCAN_RXFnS.FnF. In addition, the interrupt flag MCAN_IR.RFnF is set.

Figure 50-7. Rx FIFO Status



When reading from an Rx FIFO, Rx FIFO Get Index MCAN_RXFnS.FnGI × FIFO Element Size has to be added to the corresponding Rx FIFO start address MCAN_RXFnC.FnSA.

Table 50-4. Rx Buffer / FIFO Element Size

MCAN_RXESC.RBDS[2:0] MCAN_RXESC.FnDS[2:0]	Data Field [bytes]	FIFO Element Size [RAM words]
0	8	4
1	12	5
2	16	6
3	20	7
4	24	8
5	32	10
6	48	14
7	64	18

Rx FIFO Blocking Mode

The Rx FIFO Blocking mode is configured by MCAN_RXFnC.FnOM = ‘0’. This is the default operating mode for the Rx FIFOs.

When an Rx FIFO full condition is reached (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by MCAN_RXFnS.FnF = ‘1’. In addition, the interrupt flag MCAN_IR.RFnF is set.

In case a message is received while the corresponding Rx FIFO is full, this message is discarded and the message lost condition is signalled by MCAN_RXFnS.RFnL = ‘1’. In addition, the interrupt flag MCAN_IR.RFnL is set.

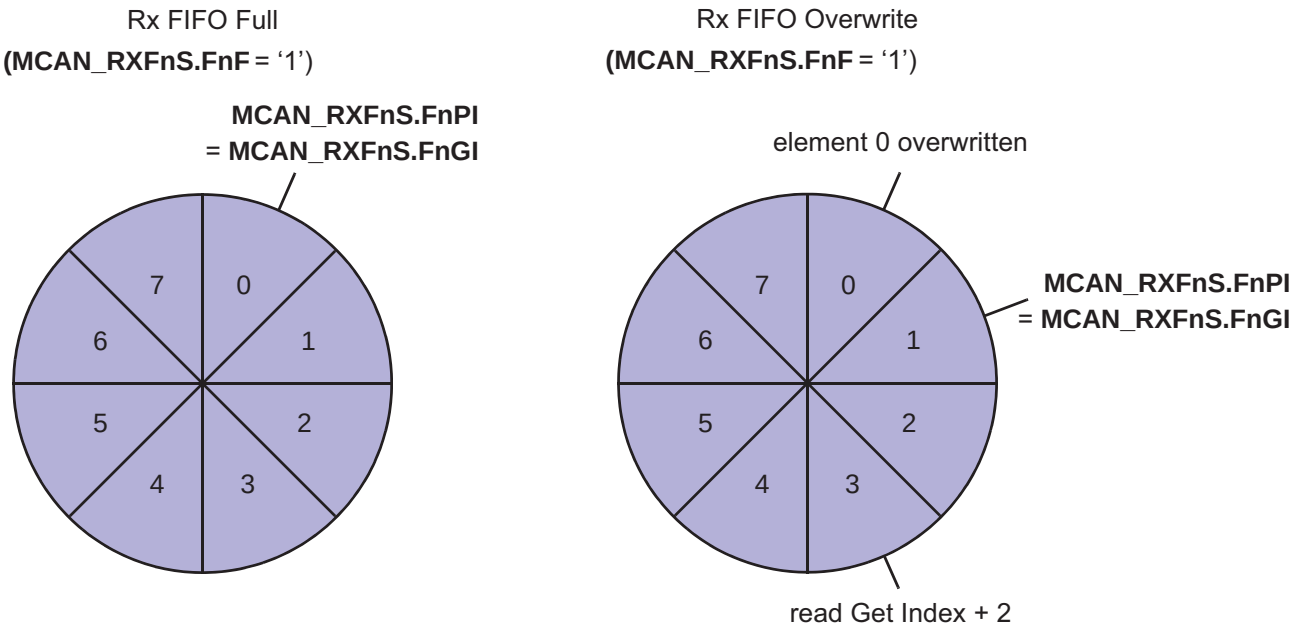
Rx FIFO Overwrite Mode

The Rx FIFO Overwrite mode is configured by MCAN_RXFnC.FnOM = ‘1’.

When an Rx FIFO full condition (MCAN_RXFnS.FnPI = MCAN_RXFnS.FnGI) is signalled by MCAN_RXFnS.FnF = ‘1’, the next message accepted for the FIFO will overwrite the oldest FIFO message. Put and get index are both incremented by one.

When an Rx FIFO is operated in Overwrite mode and an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (put index) while the processor is reading from the Message RAM (get index). In this case inconsistent data may be read from the respective Rx FIFO element. Adding an offset to the get index when reading from the Rx FIFO avoids this problem. The offset depends on how fast the processor accesses the Rx FIFO. Figure 50-8 shows an offset of two with respect to the get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

Figure 50-8. Rx FIFO Overflow Handling



After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index MCAN_RXFnA.FnA. This increments the get index to that element number. In case the put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (MCAN_RXFnS.FnF = ‘0’).

50.5.4.3 Dedicated Rx Buffers

The MCAN supports up to 64 dedicated Rx Buffers. The start address of the dedicated Rx Buffer section is configured via MCAN_RXBC.RBSA.

For each Rx Buffer, a Standard or Extended Message ID Filter Element with SFEC / EFEC = 7 and SFID2 / EFID2[10:9] = 0 has to be configured (see Section 50.5.7.5 and Section 50.5.7.6).

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element. The format is the same as for an Rx FIFO element. In addition, the flag MCAN_IR.DRX (Message stored in dedicated Rx Buffer) in MCAN_IR is set.

Table 50-5. Example Filter Configuration for Rx Buffers

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	0	0
1	ID message 2	0	1
2	ID message 3	0	2

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in the New Data 1 register (MCAN_NDAT1) and New Data 2 register (MCAN_NDAT2) is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the processor by writing a '1' to the respective bit position.

While an Rx Buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

Rx Buffer Handling

- Reset interrupt flag IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

50.5.4.4 Debug on CAN Support

Debug messages are stored into Rx Buffers. For debug handling three consecutive Rx buffers (e.g. #61, #62, #63) have to be used for storage of debug messages A, B, and C. The format is the same as for an Rx Buffer or an Rx FIFO element (see Section 50.5.7.2 "Rx Buffer and FIFO Element").

Advantage: Fixed start address for the DMA transfers (relative to MCAN_RXBC.RBSA), no additional configuration required.

For filtering of debug messages Standard / Extended Filter Elements with SFEC / EFEC = '111' have to be set up. Messages matching these filter elements are stored into the Rx Buffers addressed by SFID2 / EFID2[5:0].

After message C has been stored, the DMA request output m_can_dma_req is activated and the three messages can be read from the Message RAM under DMA control. The RAM words holding the debug messages will not be changed by the MCAN while m_can_dma_req is activated. The behavior is similar to that of an Rx Buffer with its New Data flag set.

After the DMA has completed, the MCAN is prepared to receive the next set of debug messages.

Filtering for Debug Messages

Filtering for debug messages is done by configuring one Standard / Extended Message ID Filter Element for each of the three debug messages. To enable a filter element to filter for debug messages SFEC / EFEC has to be

programmed to “111”. In this case fields SFID1 / SFID2 and EFID1 / EFID2 have a different meaning (see Section 50.5.7.5 and Section 50.5.7.6). While SFID2 / EFID2[10:9] controls the debug message handling state machine, SFID2 / EFID2[5:0] controls the location for storage of a received debug message.

When a debug message is stored, neither the respective New Data flag nor MCAN_IR.DRX are set. The reception of debug messages can be monitored via RXF1S.DMS.

Table 50-6. Example Filter Configuration for Debug Messages

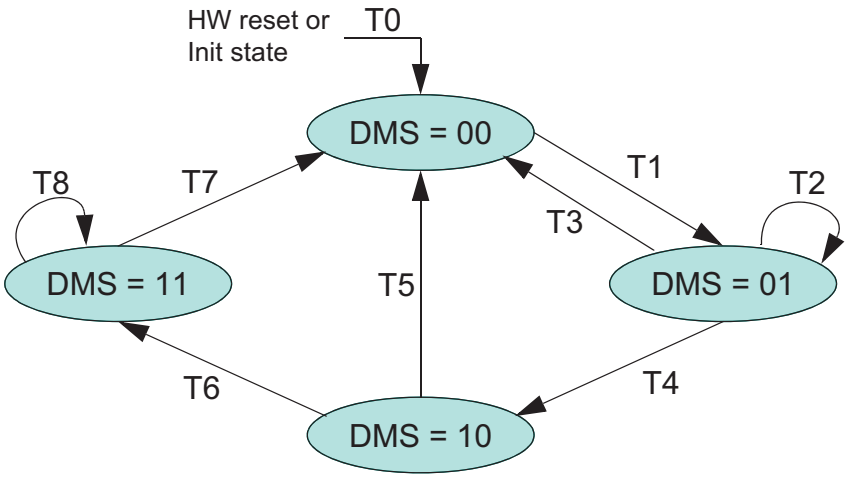
Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID debug message A	1	11 1101
1	ID debug message B	2	11 1110
2	ID debug message C	3	11 1111

Debug Message Handling

The debug message handling state machine ensures that debug messages are stored to three consecutive Rx Buffers in the correct order. If some messages are missing, the process is restarted. The DMA request is activated only when all three debug messages A, B, C have been received in the correct order.

The status of the debug message handling state machine is signalled via MCAN_RXF1S.DMS.

Figure 50-9. Debug Message Handling State Machine



T0: reset m_can_dma_req output, enable reception of debug messages A, B, and C

T1: reception of debug message A

T2: reception of debug message A

T3: reception of debug message C

T4: reception of debug message B

T5: reception of debug messages A, B

T6: reception of debug message C

T7: DMA transfer completed

T8: reception of debug message A,B,C (message rejected)

50.5.5 Tx Handling

The Tx Handler handles transmission requests for the dedicated Tx Buffers, the Tx FIFO, and the Tx Queue. It controls the transfer of transmit messages to the CAN Core, the Put and Get Indices, and the Tx Event FIFO. Up to 32 Tx Buffers can be set up for message transmission. The CAN mode for transmission (Classic CAN or CAN FD) can be configured separately for each Tx Buffer element. The Tx Buffer element is described in Section 50.5.7.3. Table 50-7 describes the possible configurations for frame transmission.

Table 50-7. Possible Configurations for Frame Transmission

MCAN_CCCR		Tx Buffer Element		
BRSE	**FDOE**	**FDF**	**BRS**	**Frame Transmission**
ignored	0	ignored	ignored	Classic CAN
0	1	0	ignored	Classic CAN
0	1	1	ignored	FD without bit rate switching
1	1	0	ignored	Classic CAN
1	1	1	0	FD without bit rate switching
1	1	1	1	FD with bit rate switching

Note: AUTOSAR requires at least three Tx Queue Buffers and support of transmit cancellation.

The Tx Handler starts a Tx scan to check for the highest priority pending Tx request (Tx Buffer with lowest Message ID) when MCAN_TXBRP is updated, or when a transmission has been started.

50.5.5.1 Transmit Pause

The transmit pause feature is intended for use in CAN systems where the CAN message identifiers are (permanently) specified to specific values and cannot easily be changed. These message identifiers may have a higher CAN arbitration priority than other defined messages, while in a specific application their relative arbitration priority should be inverse. This may lead to a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed because that other messages have a lower CAN arbitration priority.

If e.g. CAN ECU-1 has the transmit pause feature enabled and is requested by its application software to transmit four messages, it will, after the first successful message transmission, wait for two CAN bit times of bus idle before it is allowed to start the next requested message. If there are other ECUs with pending messages, those messages are started in the idle time, they would not need to arbitrate with the next message of ECU-1. After having received a message, ECU-1 is allowed to start its next transmission as soon as the received message releases the CAN bus.

The transmit pause feature is controlled by bit MCAN_CCCR.TXP. If the bit is set, the MCAN will, each time it has successfully transmitted a message, pause for two CAN bit times before starting the next transmission. This enables other CAN nodes in the network to transmit messages even if their messages have lower prior identifiers. Default is transmit pause disabled (MCAN_CCCR.TXP = '0').

This feature looses up burst transmissions coming from a single node and it protects against "babbling idiot" scenarios where the application program erroneously requests too many transmissions.

50.5.5.2 Dedicated Tx Buffers

Dedicated Tx Buffers are intended for message transmission under complete control of the processor. Each dedicated Tx Buffer is configured with a specific Message ID. In case that multiple Tx Buffers are configured with the same Message ID, the Tx Buffer with the lowest buffer number is transmitted first.

If the data section has been updated, a transmission is requested by an Add Request via MCAN_TXBAR.ARn. The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

A dedicated Tx Buffer allocates Element Size 32-bit words in the Message RAM (see Table 50-8). Therefore the start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index (0…31) × Element Size to the Tx Buffer Start Address TXBC.TBSA.

Table 50-8. Tx Buffer / FIFO / Queue Element Size

TXESC.TBDS[2:0]	Data Field [bytes]	Element Size [RAM words]
0	8	4
1	12	5
2	16	6
3	20	7
4	24	8
5	32	10
6	48	14
7	64	18

50.5.5.3 Tx FIFO

Tx FIFO operation is configured by programming MCAN_TXBC.TFQM to ‘0’. Messages stored in the Tx FIFO are transmitted starting with the message referenced by the Get Index MCAN_TXFQS.TFGI. After each transmission the Get Index is incremented cyclically until the Tx FIFO is empty. The Tx FIFO enables transmission of messages with the same Message ID from different Tx Buffers in the order these messages have been written to the Tx FIFO. The MCAN calculates the Tx FIFO Free Level MCAN_TXFQS.TFFL as difference between Get and Put Index. It indicates the number of available (free) Tx FIFO elements.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request increments the Put Index to the next free Tx FIFO element. When the Put Index reaches the Get Index, Tx FIFO Full (MCAN_TXFQS.TFQF = ‘1’) is signalled. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

When a single message is added to the Tx FIFO, the transmission is requested by writing a ‘1’ to the TXBAR bit related to the Tx Buffer referenced by the Tx FIFO’s Put Index.

When multiple (n) messages are added to the Tx FIFO, they are written to n consecutive Tx Buffers starting with the Put Index. The transmissions are then requested via MCAN_TXBAR. The Put Index is then cyclically incremented by n. The number of requested Tx buffers should not exceed the number of free Tx Buffers as indicated by the Tx FIFO Free Level.

When a transmission request for the Tx Buffer referenced by the Get Index is cancelled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level is recalculated. When transmission cancellation is applied to any other Tx Buffer, the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates Element Size 32-bit words in the Message RAM (see Table 50-8). Therefore the start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

50.5.5.4 Tx Queue

Tx Queue operation is configured by programming MCAN_TXBC.TFQM to ‘1’. Messages stored in the Tx Queue are transmitted starting with the message with the lowest Message ID (highest priority). In case that multiple Queue Buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New messages have to be written to the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQPI. An Add Request cyclically increments the Put Index to the next free Tx Buffer. In case that the Tx Queue is full (MCAN_TXFQS.TFQF = ‘1’), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been cancelled.

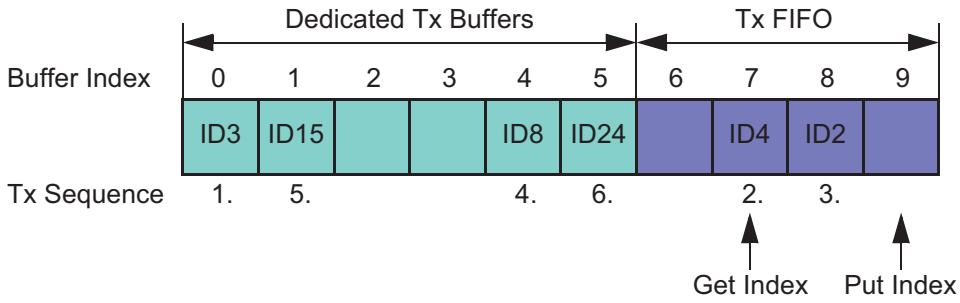
The application may use register MCAN_TXBRP instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates Element Size 32-bit words in the Message RAM (see Table 50-8). Therefore the start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQPI (0…31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA.

50.5.5.5 Mixed Dedicated Tx Buffers / Tx FIFO

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx FIFO. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Buffers assigned to the Tx FIFO is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

Figure 50-10. Example of Mixed Configuration Dedicated Tx Buffers / Tx FIFO



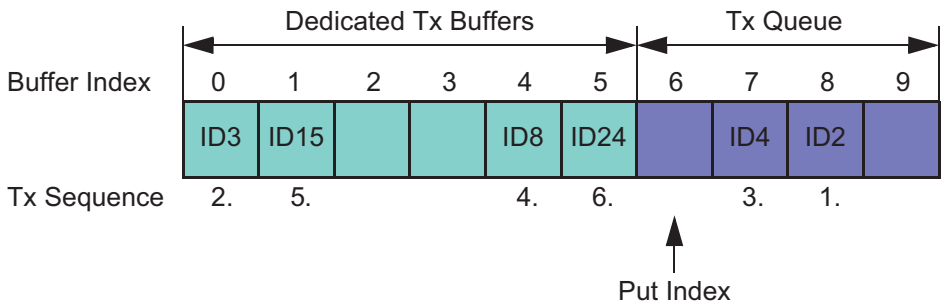
Tx prioritization:

- Scan dedicated Tx Buffers and oldest pending Tx FIFO Buffer (referenced by MCAN_TXFS.TFGI)
- Buffer with lowest Message ID gets highest priority and is transmitted next

50.5.5.6 Mixed Dedicated Tx Buffers / Tx Queue

In this case the Tx Buffers section in the Message RAM is subdivided into a set of dedicated Tx Buffers and a Tx Queue. The number of dedicated Tx Buffers is configured by MCAN_TXBC.NDTB. The number of Tx Queue Buffers is configured by MCAN_TXBC.TFQS. In case MCAN_TXBC.TFQS is programmed to zero, only dedicated Tx Buffers are used.

Figure 50-11. Example of Mixed Configuration Dedicated Tx Buffers / Tx Queue



Tx prioritization:

- Scan all Tx Buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

50.5.5.7 Transmit Cancellation

The MCAN supports transmit cancellation. This feature is especially intended for gateway applications and AUTOSAR-based applications. To cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer, the processor has to write a '1' to the corresponding bit position (=number of Tx Buffer) of register MCAN_TXBCR. Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signalled by setting the corresponding bit of register MCAN_TXBCF to '1'.

In case a transmit cancellation is requested while a transmission from a Tx Buffer is already ongoing, the corresponding TXBRP bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding MCAN_TXBTO and MCAN_TXBCF bits are set. If the transmission was not successful, it is not repeated and only the corresponding MCAN_TXBCF bit is set.

Note: In case a pending transmission is cancelled immediately before this transmission could have been started, there follows a short time window where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message which may have a lower priority than the second message in this node.

50.5.5.8 Tx Event Handling

To support Tx event handling the MCAN has implemented a Tx Event FIFO. After the MCAN has transmitted a message on the CAN bus, Message ID and timestamp are stored in a Tx Event FIFO element. To link a Tx event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

The Tx Event FIFO can be configured to a maximum of 32 elements. The Tx Event FIFO element is described in Section 50.5.4.4.

When a Tx Event FIFO full condition is signalled by IR.TEFF, no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented. In case a Tx event occurs while the Tx Event FIFO is full, this event is discarded and interrupt flag MCAN_IR.TEFL is set.

To avoid a Tx Event FIFO overflow, the Tx Event FIFO watermark can be used. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by MCAN_TXEFC.EFWM, interrupt flag MCAN_IR.TEFW is set.

When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index MCAN_TXEFS.EFGI has to be added to the Tx Event FIFO start address MCAN_TXEFC.EFSA.

50.5.6 FIFO Acknowledge Handling

The Get Indices of Rx FIFO 0, Rx FIFO 1, and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index (see Section 50.6.29, Section 50.6.33, and Section 50.6.47). Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level. There are two use cases:

When only a single element has been read from the FIFO (the one being pointed to by the Get Index), this Get Index value is written to the FIFO Acknowledge Index.

When a sequence of elements has been read from the FIFO, it is sufficient to write the FIFO Acknowledge Index only once at the end of that read sequence (value: Index of the last element read), to update the FIFO's Get Index.

Due to the fact that the processor has free access to the MCAN's Message RAM, special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This might be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also alters the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

Note: The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The MCAN does not check for erroneous values.

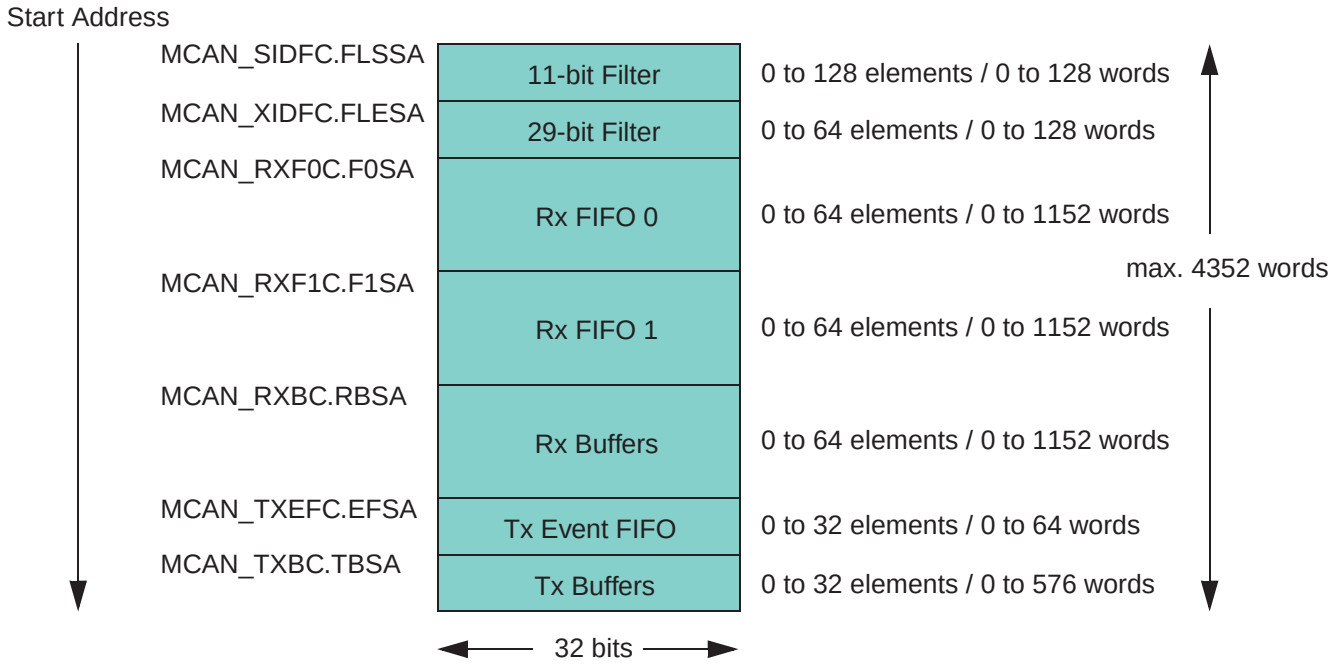
50.5.7 Message RAM

50.5.7.1 Message RAM Configuration

The Message RAM has a width of 32 bits. The MCAN module can be configured to allocate up to 4352 words in the Message RAM. It is not necessary to configure each of the sections listed in Figure 50-12, nor is there any restriction with respect to the sequence of the sections.

When operated in CAN FD mode, the required Message RAM size depends on the element size configured for Rx FIFO0, Rx FIFO1, Rx Buffers, and Tx Buffers via MCAN_RXESC.F0DS, MCAN_RXESC.F1DS, MCAN_RXESC.RBDS, and MCAN_TXESC.TBDS.

Figure 50-12. Message RAM Configuration



When the MCAN addresses the Message RAM, it addresses 32-bit words, not single bytes. The configurable start addresses are 32-bit word addresses; i.e., only bits 15 to 2 are evaluated, the two least significant bits are ignored.

Note: The MCAN does not check for erroneous configuration of the Message RAM. The configuration of the start addresses of the different sections and the number of elements of each section must be checked carefully to avoid falsification or loss of data.

50.5.7.2 Rx Buffer and FIFO Element

Up to 64 Rx Buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The structure of a Rx Buffer / FIFO element is shown in Table 50-9 below. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register MCAN_RXESC.

Table 50-9. Rx Buffer and FIFO Element

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R0	ESI	XTD	RTR	ID[28:0]																												
R1	ANMF	FIDX[6:0]							–								FDF	BRS	DLC[3:0]				RXTS[15:0]									
R2	DB3[7:0]								DB2[7:0]								DB1[7:0]								DB0[7:0]							
R3	DB7[7:0]								DB6[7:0]								DB5[7:0]								DB4[7:0]							
...	...								...								...								...							
Rn	DBm[7:0]								DBm-1[7:0]								DBm-2[7:0]								DBm-3[7:0]							

- **R0 Bit 31 ESI: Error State Indicator**

0: Transmitting node is error active.

1: Transmitting node is error passive.

- **R0 Bit 30 XTD: Extended Identifier**

Signals to the processor whether the received frame has a standard or extended identifier.

0: 11-bit standard identifier.

1: 29-bit extended identifier.

- **R0 Bit 29 RTR: Remote Transmission Request**

Signals to the processor whether the received frame is a data frame or a remote frame.

0: Received frame is a data frame.

1: Received frame is a remote frame.

Note: There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1), bit RTR reflects the state of the reserved bit r1.

- **R0 Bits 28:0 ID[28:0]: Identifier**

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

- **R1 Bit 31 ANMF: Accepted Non-matching Frame**

Acceptance of non-matching frames may be enabled via MCAN_GFC.ANFS and MCAN_GFC.ANFE.

0: Received frame matching filter index FIDX.

1: Received frame did not match any Rx filter element.

- **R1 Bits 30:24 FIDX[6:0]: Filter Index**

0-127: Index of matching Rx acceptance filter element (invalid if ANMF = ‘1’).
Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

- **R1 Bit 21 FDF: FD Format**

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

- **R1 Bit 20 BRS: Bit Rate Switch**

0: Frame received without bit rate switching.

1: Frame received with bit rate switching.

Note: Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

- **R1 Bits 19:16 DLC[3:0]: Data Length Code**

0-8: CAN + CAN FD: received frame has 0-8 data bytes.

9-15: CAN: received frame has 8 data bytes.

9-15: CAN FD: received frame has 12/16/20/24/32/48/64 data bytes.

- **R1 Bits 15:0 RXTS[15:0]: Rx Timestamp**

Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

- **R2 Bits 31:24 DB3[7:0]: Data Byte 3**
- **R2 Bits 23:16 DB2[7:0]: Data Byte 2**
- **R2 Bits 15:8 DB1[7:0]: Data Byte 1**
- **R2 Bits 7:0 DB0[7:0]: Data Byte 0**
- **R3 Bits 31:24 DB7[7:0]: Data Byte 7**
- **R3 Bits 23:16 DB6[7:0]: Data Byte 6**
- **R3 Bits 15:8 DB5[7:0]: Data Byte 5**
- **R3 Bits 7:0 DB4[7:0]: Data Byte 4**

... ...

- **Rn Bits 31:24 DBm[7:0]: Data Byte m**
- **Rn Bits 23:16 DBm-1[7:0]: Data Byte m-1**
- **Rn Bits 15:8 DBm-2[7:0]: Data Byte m-2**
- **Rn Bits 7:0 DBm-3[7:0]: Data Byte m-3**

Note: Depending on the configuration of the element size (MCAN_RXESC), between two and sixteen 32-bit words (Rn = 3 ..17) are used for storage of a CAN message's data field.

50.5.7.3 Tx Buffer Element

The Tx Buffers section can be configured to hold dedicated Tx Buffers as well as a Tx FIFO / Tx Queue. In case that the Tx Buffers section is shared by dedicated Tx buffers and a Tx FIFO / Tx Queue, the dedicated Tx Buffers start at the beginning of the Tx Buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler distinguishes between dedicated Tx Buffers and Tx FIFO / Tx Queue by evaluating the Tx Buffer configuration TXBC.TFQS and TXBC.NDTB. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field via register TXESC.

Table 50-10. Tx Buffer Element

	31	30	29	28:24	23	22	21	20	19:16	15:8	7:0
T0	ESI	XTD	RTR	ID[28:0]							
T1	MM[7:0]				EFC	reserved	FDF	BRS	DLC[3:0]	reserved	
T2	DB3[7:0]				DB2[7:0]					DB1[7:0]	DB0[7:0]
T3	DB7[7:0]				DB6[7:0]					DB5[7:0]	DB4[7:0]
...	...				...					...	...
Tn	DBm[7:0]				DBm-1[7:0]					DBm-2[7:0]	DBm-3[7:0]

- **T0 Bit 30 ESI: Error State Indicator**

T0 Bit 31 ESI: Error State Indicator

0: ESI bit in CAN FD format depends only on error passive flag

1: ESI bit in CAN FD format transmitted recessive

Note: The ESI bit of the transmit buffer is or'ed with the error passive flag to decide the value of the ESI bit in the transmitted FD frame. As required by the CAN FD protocol specification, an error active node may optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive. This feature can be used in gateway applications when a message from an error passive node is routed to another CAN network.

- **T0 Bit 30 XTD: Extended Identifier**

0: 11-bit standard identifier.

1: 29-bit extended identifier.

- **T0 Bit 29 RTR: Remote Transmission Request**

0: Transmit data frame.

1: Transmit remote frame.

Note: When RTR = 1, the MCAN transmits a remote frame according to ISO11898-1, even if MCAN_CCCR.FDOE enables the transmission in CAN FD format.

- **T0 Bits 28:0 ID[28:0]: Identifier**

Standard or extended identifier depending on bit XTD. A standard identifier has to be written to ID[28:18].

- **T1 Bits 31:24 MM[7:0]: Message Marker**

Written by processor during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status.

- **T1 Bit 23 EFC: Event FIFO Control**

0: Do not store Tx events.

1: Store Tx events.

- **T1 Bit 21 FDF: FD Format**

0: Frame transmitted in Classic CAN format

1: Frame transmitted in CAN FD format

- **T1 Bit 20 BRS: Bit Rate Switching**

0: CAN FD frames transmitted without bit rate switching

1: CAN FD frames transmitted with bit rate switching

Note: Bits ESI, FDF, and BRS are only evaluated when CAN FD operation is enabled (MCAN_CCCR.FDOE = 1). Bit BRS is only evaluated when in addition MCAN_CCCR.BRSE = 1.

- **T1 Bits 19:16 DLC[3:0]: Data Length Code**

0-8: CAN + CAN FD: transmit frame has 0-8 data bytes.

9-15: CAN: transmit frame has 8 data bytes.

9-15: CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes.

- **T2 Bits 31:24 DB3[7:0]: Data Byte 3**
- **T2 Bits 23:16 DB2[7:0]: Data Byte 2**
- **T2 Bits 15:8 DB1[7:0]: Data Byte 1**
- **T2 Bits 7:0 DB0[7:0]: Data Byte 0**
- **T3 Bits 31:24 DB7[7:0]: Data Byte 7**
- **T3 Bits 23:16 DB6[7:0]: Data Byte 6**
- **T3 Bits 15:8 DB5[7:0]: Data Byte 5**
- **T3 Bits 7:0 DB4[7:0]: Data Byte 4**

… … …

- **Tn Bits 31:24 DBm[7:0]: Data Byte m**
- **Tn Bits 23:16 DBm-1[7:0]: Data Byte m-1**
- **Tn Bits 15:8 DBm-2[7:0]: Data Byte m-2**
- **Tn Bits 7:0 DBm-3[7:0]: Data Byte m-3**

Note: Depending on the configuration of the element size (MCAN_TXESC), between two and sixteen 32-bit words (Tn = 3 ..17) are used for storage of a CAN message's data field.

50.5.7.4 Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the processor gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from register TXEFS.

Table 50-11. Tx Event FIFO Element

	31							24	23							16	15							8	7							0
E0	ESI	XTD	RTR	ID[28:0]																												
E1	MM[7:0]								ET [1:0]		FDF	BRS	DLC[3:0]				TXTS[15:0]															

- **E0 Bit 31 ESI: Error State Indicator**

0: Transmitting node is error active.

1: Transmitting node is error passive.

- **E0 Bit 30 XTD: Extended Identifier**

0: 11-bit standard identifier.

1: 29-bit extended identifier.

- **E0 Bit 29 RTR: Remote Transmission Request**

0: Data frame transmitted.

1: Remote frame transmitted.

- **E0 Bits 28:0 ID[28:0]: Identifier**

Standard or extended identifier depending on bit XTD. A standard identifier is stored into ID[28:18].

- **E1 Bits 31:24 MM[7:0]: Message Marker**

Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status.

- **E1 Bit 23:22 ET[1:0]: Event Type**

Value	Description
0	Reserved
1	Tx event
2	Transmission in spite of cancellation (always set for transmissions in DAR mode)
3	Reserved

- **E1 Bit 21 FDF: FD Format**

0: Standard frame format.

1: CAN FD frame format (new DLC-coding and CRC).

- **E1 Bit 20 BRS: Bit Rate Switch**

0: Frame transmitted without bit rate switching.

1: Frame transmitted with bit rate switching.

- **E1 Bits 19:16 DLC[3:0]: Data Length Code**

0-8: CAN + CAN FD: frame with 0-8 data bytes transmitted.

9-15: CAN: frame with 8 data bytes transmitted.

9-15: CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted

- **E1 Bits 15:0 TXTS[15:0]: Tx Timestamp**

Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.

50.5.7.5 Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address MCAN_SIDFC.FLSSA plus the index of the filter element (0…127).

Table 50-12. Standard Message ID Filter Element

	31	24	23	16	15	8	7	0
S0	SFT[1:0]	SFEC [2:0]	SFID1[10:0]		–		SFID2[10:0]	

- **Bits 31:30 SFT[1:0]: Standard Filter Type**

Value	Description
0	Range filter from SF1ID to SF2ID (SF2ID ≥ SF1ID)
1	Dual ID filter for SF1ID or SF2ID
2	Classic filter: SF1ID = filter, SF2ID = mask
3	Reserved

- **Bit 29:27 SFEC[2:0]: Standard Filter Element Configuration**

All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = “100”, “101”, or “110” a match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case register HPMS is updated with the status of the priority match.

Value	Description
0	Disable filter element
1	Store in Rx FIFO 0 if filter matches
2	Store in Rx FIFO 1 if filter matches
3	Reject ID if filter matches
4	Set priority if filter matches
5	Set priority and store in FIFO 0 if filter matches
6	Set priority and store in FIFO 1 if filter matches
7	Store into Rx Buffer or as debug message, configuration of SFT[1:0] ignored

- **Bits 26:16 SFID1[10:0]: Standard Filter ID 1**

First ID of standard ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.

- **Bits 10:0 SFID2[10:0]: Standard Filter ID 2**

This field has a different meaning depending on the configuration of SFEC:

- SFEC = “001”...“110”–Second ID of standard ID filter element
- SFEC = “111”–Filter for Rx Buffers or for debug messages

SFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

Value	Description
0	Store message in a Rx buffer
1	Debug Message A
2	Debug Message B
3	Debug Message C

SFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

50.5.7.6 Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address MCAN_XIDFC.FLESA plus two times the index of the filter element (0…63).

Table 50-13. Extended Message ID Filter Element

	31	24	23	16	15	8	7	0
F0	EFEC [2:0]		EFID1[28:0]					
F1	EFT[1:0]	–	EFID2[28:0]					

- **F0 Bit 31:29 EFEC[2:0]: Extended Filter Element Configuration**

All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = “100”, “101”, or “110”, a match sets the interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, register MCAN_HPMS is updated with the status of the priority match.

Value	Description
0	Disable filter element
1	Store in Rx FIFO 0 if filter matches
2	Store in Rx FIFO 1 if filter matches
3	Reject ID if filter matches
4	Set priority if filter matches
5	Set priority and store in FIFO 0 if filter matches
6	Set priority and store in FIFO 1 if filter matches
7	Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored

- **F0 Bits 28:0 EFID1[28:0]: Extended Filter ID 1**

First ID of extended ID filter element.

When filtering for Rx Buffers or for debug messages this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only MCAN_XIDAM masking mechanism (see Extended Message ID Filtering) is used.

- **F1 Bits 31:30 EFT[1:0]: Extended Filter Type**

Value	Description
0	Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID)
1	Dual ID filter for EF1ID or EF2ID
2	Classic filter: EF1ID = filter, EF2ID = mask
3	Range filter from EF1ID to EF2ID (EF2ID ≥ EF1ID), MCAN_XIDAM mask not applied

- **F1 Bits 28:0 EFID2[28:0]: Extended Filter ID 2**

This field has a different meaning depending on the configuration of EFEC:

- EFEC = “001”...“110”–Second ID of extended ID filter element
- EFEC = “111”–Filter for Rx Buffers or for debug messages

EFID2[10:9] decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence.

Value	Description
0	Store message in a Rx buffer
1	Debug Message A
2	Debug Message B
3	Debug Message C

EFID2[5:0] defines the index of the dedicated Rx Buffer element to which a matching message is stored.

50.5.8 Hardware Reset Description

After hardware reset, the registers of the MCAN hold the reset values listed in Table 50-14. Additionally the Bus_Off state is reset and the output CANTX is set to recessive (HIGH). The value 0x0001 (MCAN_CCCR.INIT = ‘1’) in the CC Control register enables software initialization. The MCAN does not influence the CAN bus until the processor resets MCAN_CCCR.INIT to ‘0’.

50.5.9 Access to Reserved Register Addresses

In case the application software accesses one of the reserved addresses in the MCAN register map (read or write access), interrupt flag MCAN_IR.ARA is set and, if enabled, the selected interrupt line is risen.

50.6 Controller Area Network (MCAN) User Interface

Table 50-14. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Core Release Register	MCAN_CREL	Read-only	0xrrrddddd[1]
0x04	Endian Register	MCAN_ENDN	Read-only	0x87654321
0x08	Customer Register	MCAN_CUST	Read/Write	0
0x0C	Data Bit Timing and Prescaler Register	MCAN_DBTP	Read/Write	0x00000A33
0x10	Test Register	MCAN_TEST	Read/Write	0x000000x0[2]
0x14	RAM Watchdog Register	MCAN_RWD	Read/Write	0x00000000
0x18	CC Control Register	MCAN_CCCR	Read/Write	0x00000001
0x1C	Nominal Bit Timing and Prescaler Register	MCAN_NBTP	Read/Write	0x06000A03
0x20	Timestamp Counter Configuration Register	MCAN_TSCC	Read/Write	0x00000000
0x24	Timestamp Counter Value Register	MCAN_TSCV	Read/Write	0x00000000
0x28	Timeout Counter Configuration Register	MCAN_TOCC	Read/Write	0xFFFF0000
0x2C	Timeout Counter Value Register	MCAN_TOCV	Read/Write	0x0000FFFF
0x30–0x3C	Reserved	–	–	–
0x40	Error Counter Register	MCAN_ECR	Read-only	0x00000000
0x44	Protocol Status Register	MCAN_PSR	Read-only	0x00000707
0x48	Transmit Delay Compensation Register	MCAN_TDCR	Read/Write	0x00000000
0x4C	Reserved	–	–	–
0x50	Interrupt Register	MCAN_IR	Read/Write	0x00000000
0x54	Interrupt Enable Register	MCAN_IE	Read/Write	0x00000000
0x58	Interrupt Line Select Register	MCAN_ILS	Read/Write	0x00000000
0x5C	Interrupt Line Enable Register	MCAN_ILE	Read/Write	0x00000000
0x60–0x7C	Reserved	–	–	–
0x80	Global Filter Configuration Register	MCAN_GFC	Read/Write	0x00000000
0x84	Standard ID Filter Configuration Register	MCAN_SIDFC	Read/Write	0x00000000
0x88	Extended ID Filter Configuration Register	MCAN_XIDFC	Read/Write	0x00000000
0x8C	Reserved	–	–	–
0x90	Extended ID AND Mask Register	MCAN_XIDAM	Read/Write	0x1FFFFFFF
0x94	High Priority Message Status Register	MCAN_HPMS	Read-only	0x00000000
0x98	New Data 1 Register	MCAN_NDAT1	Read/Write	0x00000000
0x9C	New Data 2 Register	MCAN_NDAT2	Read/Write	0x00000000
0xA0	Receive FIFO 0 Configuration Register	MCAN_RXF0C	Read/Write	0x00000000
0xA4	Receive FIFO 0 Status Register	MCAN_RXF0S	Read-only	0x00000000
0xA8	Receive FIFO 0 Acknowledge Register	MCAN_RXF0A	Read/Write	0x00000000
0xAC	Receive Rx Buffer Configuration Register	MCAN_RXBC	Read/Write	0x00000000
0xB0	Receive FIFO 1 Configuration Register	MCAN_RXF1C	Read/Write	0x00000000

Table 50-14. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0xB4	Receive FIFO 1 Status Register	MCAN_RXF1S	Read-only	0x00000000
0xB8	Receive FIFO 1 Acknowledge Register	MCAN_RXF1A	Read/Write	0x00000000
0xBC	Receive Buffer / FIFO Element Size Configuration Register	MCAN_RXESC	Read/Write	0x00000000
0xC0	Transmit Buffer Configuration Register	MCAN_TXBC	Read/Write	0x00000000
0xC4	Transmit FIFO/Queue Status Register	MCAN_TXFQS	Read-only	0x00000000
0xC8	Transmit Buffer Element Size Configuration Register	MCAN_TXESC	Read/Write	0x00000000
0xCC	Transmit Buffer Request Pending Register	MCAN_TXBRP	Read-only	0x00000000
0xD0	Transmit Buffer Add Request Register	MCAN_TXBAR	Read/Write	0x00000000
0xD4	Transmit Buffer Cancellation Request Register	MCAN_TXBCR	Read/Write	0x00000000
0xD8	Transmit Buffer Transmission Occurred Register	MCAN_TXBTO	Read-only	0x00000000
0xDC	Transmit Buffer Cancellation Finished Register	MCAN_TXBCF	Read-only	0x00000000
0xE0	Transmit Buffer Transmission Interrupt Enable Register	MCAN_TXBTIE	Read/Write	0x00000000
0xE4	Transmit Buffer Cancellation Finished Interrupt Enable Register	MCAN_TXBCIE	Read/Write	0x00000000
0xE8–0xEC	Reserved	–	–	–
0xF0	Transmit Event FIFO Configuration Register	MCAN_TXEFC	Read/Write	0x00000000
0xF4	Transmit Event FIFO Status Register	MCAN_TXEFS	Read-only	0x00000000
0xF8	Transmit Event FIFO Acknowledge Register	MCAN_TXEFA	Read/Write	0x00000000
0xFC	Reserved	–	–	–

Notes:
1. Due to clock domain crossing, there is a delay between when a register bit or field is written and when the related status register bits are updated.
2. The reset value for bit 7, MCAN_TEST.RX, is undefined.

50.6.1 MCAN Core Release Register

Name: MCAN_CREL

Address: 0xF8054000 (0), 0xFC050000 (1)

Access: Read-only

31	30	29	28	27	26	25	24
REL				STEP			
23	22	21	20	19	18	17	16
SUBSTEP				YEAR			
15	14	13	12	11	10	9	8
MON							
7	6	5	4	3	2	1	0
DAY							

- **DAY: Timestamp Day**

Two digits, BCD-coded. This field is set by generic parameter on MCAN synthesis.

- **MON: Timestamp Month**

Two digits, BCD-coded. This field is set by generic parameter on MCAN synthesis.

- **YEAR: Timestamp Year**

One digit, BCD-coded. This field is set by generic parameter on MCAN synthesis.

- **SUBSTEP: Sub-step of Core Release**

One digit, BCD-coded.

- **STEP: Step of Core Release**

One digit, BCD-coded.

- **REL: Core Release**

One digit, BCD-coded.

50.6.2 MCAN Endian Register

Name: MCAN_ENDN

Address: 0xF8054004 (0), 0xFC050004 (1)

Access: Read-only

31	30	29	28	27	26	25	24
ETV							
23	22	21	20	19	18	17	16
ETV							
15	14	13	12	11	10	9	8
ETV							
7	6	5	4	3	2	1	0
ETV							

- **ETV: Endianness Test Value**

The endianness test value is 0x87654321.

50.6.3 MCAN Customer Register

Name: MCAN_CUST

Address: 0xF8054008 (0), 0xFC050008 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
CSV							
23	22	21	20	19	18	17	16
CSV							
15	14	13	12	11	10	9	8
CSV							
7	6	5	4	3	2	1	0
CSV							

- **CSV: Customer-specific Value**

Customer-specific value.

50.6.4 MCAN Data Bit Timing and Prescaler Register

Name: MCAN_DBTP

Address: 0xF805400C (0), 0xFC05000C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
TDC	–	–	DBRP				
15	14	13	12	11	10	9	8
–	–	–	DTSEG1				
7	6	5	4	3	2	1	0
DTSEG2				–	DSJW		

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

The CAN bit time may be programmed in the range of 4 to 25 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 CAN core clock periods. t_q = (DBRP + 1) CAN core clock periods.

DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] t_q
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

- **DSJW: Data (Re) Synchronization Jump Width**

The duration of a synchronization jump is t_q x (DSJW + 1).

- **DTSEG2: Data Time Segment After Sample Point**

The duration of time segment is t_q x (DTSEG2 + 1).

- **DTSEG1: Data Time Segment Before Sample Point**

0: Forbidden.

1 to 31: The duration of time segment is t_q x (DTSEG1 + 1).

- **DBRP: Data Bit Rate Prescaler**

The value by which the peripheral clock is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

- **TDC: Transceiver Delay Compensation**

0 (DISABLED): Transceiver Delay Compensation disabled.

1 (ENABLED): Transceiver Delay Compensation enabled.

Notes:
1. With a CAN core clock frequency of 8 MHz, the reset value of 0x00000A33 configures the MCAN for a fast bit rate of 500 kbit/s.
2. The bit rate configured for the CAN FD data phase via MCAN_DBTP must be higher than or equal to the bit rate configured for the arbitration phase via MCAN_NBTP.

50.6.5 MCAN Test Register

Name: MCAN_TEST

Address: 0xF8054010 (0), 0xFC050010 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
RX	TX		LBCK	–	–	–	–

Write access to the Test Register has to be enabled by setting bit MCAN_CCCR.TEST to ‘1’.

All MCAN Test Register functions are set to their reset values when bit MCAN_CCCR.TEST is cleared.

Loop Back mode and software control of pin CANTX are hardware test modes. Programming of TX ≠ 0 disturbs the message transfer on the CAN bus.

- **LBCK: Loop Back Mode (read/write)**

0 (DISABLED): Reset value. Loop Back mode is disabled.

1 (ENABLED): Loop Back mode is enabled (see Section 50.5.1.9).

- **TX: Control of Transmit Pin (read/write)**

Value	Name	Description
0	RESET	Reset value, CANTX controlled by the CAN Core, updated at the end of the CAN bit time.
1	SAMPLE_POINT_MONITORING	Sample Point can be monitored at pin CANTX.
2	DOMINANT	Dominant (‘0’) level at pin CANTX.
3	RECESSIVE	Recessive (‘1’) at pin CANTX.

- **RX: Receive Pin (read-only)**

Monitors the actual value of pin CANRX.

0: The CAN bus is dominant (CANRX = ‘0’).

1: The CAN bus is recessive (CANRX = ‘1’).

50.6.6 MCAN RAM Watchdog Register

Name: MCAN_RWD

Address: 0xF8054014 (0), 0xFC050014 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
WDV							
7	6	5	4	3	2	1	0
WDC							

The RAM Watchdog monitors the Message RAM response time. A Message RAM access via the MCAN's Generic Master Interface starts the Message RAM Watchdog Counter with the value configured by MCAN_RWD.WDC. The counter is reloaded with MCAN_RWD.WDC when the Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag MCAN_IR.WDI is set. The RAM Watchdog Counter is clocked by the system bus clock (peripheral clock).

- **WDC: Watchdog Configuration (read/write)**

Start value of the Message RAM Watchdog Counter. The counter is disabled when WDC is cleared.

- **WDV: Watchdog Value (read-only)**

Watchdog Counter Value for the current message located in RAM.

50.6.7 MCAN CC Control Register

Name: MCAN_CCCR

Address: 0xF8054018 (0), 0xFC050018 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	TXP	EFBI	PXHD	–	–	BRSE	FDOE
7	6	5	4	3	2	1	0
TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT

- **INIT: Initialization (read/write)**

0 (DISABLED): Normal operation.

1 (ENABLED): Initialization is started.

Note: Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to ensure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value.

- **CCE: Configuration Change Enable (read/write, write protection)**

0 (PROTECTED): The processor has no write access to the protected configuration registers.

1 (CONFIGURABLE): The processor has write access to the protected configuration registers (while MCAN_CCCR.INIT = '1').

- **ASM: Restricted Operation Mode (read/write, write protection against '1')**

For a description of the Restricted Operation mode see Section 50.5.1.5.

0 (NORMAL): Normal CAN operation.

1 (RESTRICTED): Restricted Operation mode active.

- **CSA: Clock Stop Acknowledge (read-only)**

0: No clock stop acknowledged.

1: MCAN may be set in power down by stopping the peripheral clock and the CAN core clock.

- **CSR: Clock Stop Request (read/write)**

0 (NO_CLOCK_STOP): No clock stop is requested.

1 (CLOCK_STOP): Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle.

- **MON: Bus Monitoring Mode (read/write, write protection against '1')**

0 (DISABLED): Bus Monitoring mode is disabled.

1 (ENABLED): Bus Monitoring mode is enabled.

- **DAR: Disable Automatic Retransmission (read/write, write protection)**

0 (AUTO_RETX): Automatic retransmission of messages not transmitted successfully enabled.

1 (NO_AUTO_RETX): Automatic retransmission disabled.

- **TEST: Test Mode Enable (read/write, write protection against ‘1’)**

0 (DISABLED): Normal operation, MCAN_TEST register holds reset values.

1 (ENABLED): Test mode, write access to MCAN_TEST register enabled.

- **FDOE: CAN FD Operation Enable (read/write, write protection)**

0 (DISABLED): FD operation disabled.

1 (ENABLED): FD operation enabled.

- **BRSE: Bit Rate Switching Enable (read/write, write protection)**

0 (DISABLED): Bit rate switching for transmissions disabled.

1 (ENABLED): Bit rate switching for transmissions enabled.

- **PXHD: Protocol Exception Event Handling (read/write, write protection)**

0: Protocol exception handling enabled.

1: Protocol exception handling disabled.

- **EFBI: Edge Filtering during Bus Integration (read/write, write protection)**

0: Edge filtering is disabled.

1: Edge filtering is enabled. Two consecutive dominant tq required to detect an edge for hard synchronization.

- **TXP: Transmit Pause (read/write, write protection)**

If this bit is set, the MCAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame (see Section 50.5.5).

0: Transmit pause disabled.

1: Transmit pause enabled.

50.6.8 MCAN Nominal Bit Timing and Prescaler Register

Name: MCAN_NBTP

Address: 0xF805401C (0), 0xFC05001C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
NSJW							NBRP
23	22	21	20	19	18	17	16
NBRP							
15	14	13	12	11	10	9	8
NTSEG1							
7	6	5	4	3	2	1	0
–	NTSEG2						

This register can only be written if the bits CCE and INIT are set in MCAN_CCCR.

The CAN bit time may be programmed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 CAN core clock periods. $t_q = t_{core\ clock} \times (NBRP + 1)$.

NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.

Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] t_q
or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

- **NTSEG2: Nominal Time Segment After Sample Point**

0 to 127: The duration of time segment is $t_q \times (NTSEG2 + 1)$.

- **NTSEG1: Nominal Time Segment Before Sample Point**

0: Forbidden.

1 to 255: The duration of time segment is $t_q \times (NTSEG1 + 1)$.

- **NBRP: Nominal Bit Rate Prescaler**

0 to 511: The value by which the oscillator frequency is divided for generating the CAN time quanta. The CAN time is built up from a multiple of this quanta. CAN time quantum (tq) = $t_{core\ clock} \times (NBRP + 1)$

Note: With a CAN core clock frequency of 8 MHz, the reset value of 0x06000A03 configures the MCAN for a bit rate of 500 kbit/s.

- **NSJW: Nominal (Re) Synchronization Jump Width**

0 to 127: The duration of a synchronization jump is $t_q \times (NSJW + 1)$.

50.6.9 MCAN Timestamp Counter Configuration Register

Name: MCAN_TSCC

Address: 0xF8054020 (0), 0xFC050020 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	TCP			
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	TSS	

For a description of the Timestamp Counter see Section 50.5.2.

- **TSS: Timestamp Select**

Value	Name	Description
0	ALWAYS_0	Timestamp counter value always 0x0000
1	TCP_INC	Timestamp counter value incremented according to TCP
2	EXT_TIMESTAMP	External timestamp counter value used
3	ALWAYS_0	Timestamp counter value always 0x0000

- **TCP: Timestamp Counter Prescaler**

Configures the timestamp and timeout counters time unit in multiples of CAN bit times [1…16]. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used.

Note: With CAN FD, an external counter is required for timestamp generation (TSS = 2).

50.6.10 MCAN Timestamp Counter Value Register

Name: MCAN_TSCV

Address: 0xF8054024 (0), 0xFC050024 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TSC							
7	6	5	4	3	2	1	0
TSC							

- **TSC: Timestamp Counter (cleared on write)**

The internal/external Timestamp Counter value is captured on start of frame (both Receive and Transmit). When MCAN_TSCC.TSS = 1, the Timestamp Counter is incremented in multiples of CAN bit times [1…16] depending on the configuration of MCAN_TSCC.TCP. A wrap around sets interrupt flag MCAN_IR.TSW. Write access resets the counter to zero.

When MCAN_TSCC.TSS = 2, TSC reflects the external Timestamp Counter value. Thus a write access has no impact.

Note: A “wrap around” is a change of the Timestamp Counter value from non-zero to zero not caused by write access to MCAN_TSCV.

50.6.11 MCAN Timeout Counter Configuration Register

Name: MCAN_TOCC

Address: 0xF8054028 (0), 0xFC050028 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
TOP							
23	22	21	20	19	18	17	16
TOP							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	TOS		ETOC

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

For a description of the Timeout Counter, see Section 50.5.3.

- **ETOC: Enable Timeout Counter**

0 (NO_TIMEOUT): Timeout Counter disabled.

1 (TOS_CONTROLLED): Timeout Counter enabled.

For use of timeout function with CAN FD, see Section 50.5.3.

- **TOS: Timeout Select**

When operating in Continuous mode, a write to MCAN_TOCV presets the counter to the value configured by MCAN_TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by MCAN_TOCC.TOP. Down-counting is started when the first FIFO element is stored.

Value	Name	Description
0	CONTINUOUS	Continuous operation
1	TX_EV_TIMEOUT	Timeout controlled by Tx Event FIFO
2	RX0_EV_TIMEOUT	Timeout controlled by Receive FIFO 0
3	RX1_EV_TIMEOUT	Timeout controlled by Receive FIFO 1

- **TOP: Timeout Period**

Start value of the Timeout Counter (down-counter). Configures the Timeout Period.

50.6.12 MCAN Timeout Counter Value Register

Name: MCAN_TOCV

Address: 0xF805402C (0), 0xFC05002C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TOC							
7	6	5	4	3	2	1	0
TOC							

- **TOC: Timeout Counter (cleared on write)**

The Timeout Counter is decremented in multiples of CAN bit times [1…16] depending on the configuration of MCAN_TSCC.TCP. When decremented to zero, interrupt flag MCAN_IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via MCAN_TOCC.TOS.

50.6.13 MCAN Error Counter Register

Name: MCAN_ECR

Address: 0xF8054040 (0), 0xFC050040 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CEL							
15	14	13	12	11	10	9	8
RP	REC						
7	6	5	4	3	2	1	0
TEC							

- **TEC: Transmit Error Counter**

Actual state of the Transmit Error Counter, values between 0 and 255.

- **REC: Receive Error Counter**

Actual state of the Receive Error Counter, values between 0 and 127.

- **RP: Receive Error Passive**

0: The Receive Error Counter is below the error passive level of 128.

1: The Receive Error Counter has reached the error passive level of 128.

- **CEL: CAN Error Logging (cleared on read)**

The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF; the next increment of TEC or REC sets interrupt flag IR.ELO.

Note: When MCAN_CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented.

50.6.14 MCAN Protocol Status Register

Name: MCAN_PSR

Address: 0xF8054044 (0), 0xFC050044 (1)

Access: Read-only/

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	TDCV						
15	14	13	12	11	10	9	8
–	PXE	RFDF	RBRS	RESI	DLEC		
7	6	5	4	3	2	1	0
BO	EW	EP	ACT		LEC		

- **LEC: Last Error Code (set to 111 on read)**

The LEC indicates the type of the last error to occur on the CAN bus. This field is cleared when a message has been transferred (reception or transmission) without error.

Value	Name	Description
0	NO_ERROR	No error occurred since LEC has been reset by successful reception or transmission.
1	STUFF_ERROR	More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed.
2	FORM_ERROR	A fixed format part of a received frame has the wrong format.
3	ACK_ERROR	The message transmitted by the MCAN was not acknowledged by another node.
4	BIT1_ERROR	During transmission of a message (with the exception of the arbitration field), the device tried to send a recessive level (bit of logical value ‘1’), but the monitored bus value was dominant.
5	BIT0_ERROR	During transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device tried to send a dominant level (data or identifier bit logical value ‘0’), but the monitored bus value was recessive. During Bus_Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the processor to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed).
6	CRC_ERROR	The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match the CRC calculated from the received data.
7	NO_CHANGE	Any read access to the Protocol Status Register reinitializes the LEC to ‘7’. When the LEC shows value ‘7’, no CAN bus event was detected since the last processor read access to the Protocol Status Register.

- **ACT: Activity**

Monitors the CAN communication state of the CAN module.

Value	Name	Description
0	SYNCHRONIZING	Node is synchronizing on CAN communication
1	IDLE	Node is neither receiver nor transmitter
2	RECEIVER	Node is operating as receiver
3	TRANSMITTER	Node is operating as transmitter

- **EP: Error Passive**

0: The MCAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected.

1: The MCAN is in the Error_Passive state.

- **EW: Warning Status**

0: Both error counters are below the Error_Warning limit of 96.

1: At least one of error counter has reached the Error_Warning limit of 96.

- **BO: Bus_Off Status**

0: The MCAN is not Bus_Off.

1: The MCAN is in Bus_Off state.

- **DLEC: Data Phase Last Error Code (set to 111 on read)**

Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error.

- **RESI: ESI Flag of Last Received CAN FD Message (cleared on read)**

This bit is set together with RFDF, independently from acceptance filtering.

0: Last received CAN FD message did not have its ESI flag set.

1: Last received CAN FD message had its ESI flag set.

- **RBRS: BRS Flag of Last Received CAN FD Message (cleared on read)**

This bit is set together with RFDF, independently from acceptance filtering.

0: Last received CAN FD message did not have its BRS flag set.

1: Last received CAN FD message had its BRS flag set.

- **RFDF: Received a CAN FD Message (cleared on read)**

This bit is set independently from acceptance filtering.

0: Since this bit was reset by the CPU, no CAN FD message has been received

1: Message in CAN FD format with FDF flag set has been received

- **PXE: Protocol Exception Event (cleared on read)**

0: No protocol exception event occurred since last read access

1: Protocol exception event occurred

- **TDCV: Transceiver Delay Compensation Value**

0 to 127: Position of the secondary sample point, in CAN core clock periods, defined by the sum of the measured delay from CANTX to CANRX and MCAN_TDCR.TDCO.

50.6.15 MCAN Transmitter Delay Compensation Register

Name: MCAN_TDCR

Address: 0xF8054048 (0), 0xFC050048 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	TDCO						
7	6	5	4	3	2	1	0
–	TDCF						

- **TDCF: Transmitter Delay Compensation Filter**

0 to 127: defines the minimum value for the SSP position, in CAN core clock periods. Dominant edges on CANRX that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO.

- **TDCO: Transmitter Delay Compensation Offset**

0 to 127: Offset value, in CAN core clock periods, defining the distance between the measured delay from CANTX to CANRX and the secondary sample point.

50.6.16 MCAN Interrupt Register

Name: MCAN_IR

Address: 0xF8054050 (0), 0xFC050050 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ARA	PED	PEA	WDI	BO	EW
23	22	21	20	19	18	17	16
EP	ELO	–	–	DRX	TOO	MRAF	TSW
15	14	13	12	11	10	9	8
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM
7	6	5	4	3	2	1	0
RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register. The configuration of IE controls whether an interrupt is generated. The configuration of ILS controls on which interrupt line an interrupt is signalled.

- **RF0N: Receive FIFO 0 New Message**

0: No new message written to Receive FIFO 0.

1: New message written to Receive FIFO 0.

- **RF0W: Receive FIFO 0 Watermark Reached**

0: Receive FIFO 0 fill level below watermark.

1: Receive FIFO 0 fill level reached watermark.

- **RF0F: Receive FIFO 0 Full**

0: Receive FIFO 0 not full.

1: Receive FIFO 0 full.

- **RF0L: Receive FIFO 0 Message Lost**

0: No Receive FIFO 0 message lost.

1: Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero.

- **RF1N: Receive FIFO 1 New Message**

0: No new message written to Receive FIFO 1.

1: New message written to Receive FIFO 1.

- **RF1W: Receive FIFO 1 Watermark Reached**

0: Receive FIFO 1 fill level below watermark.

1: Receive FIFO 1 fill level reached watermark.

- **RF1F: Receive FIFO 1 Full**

0: Receive FIFO 1 not full.

1: Receive FIFO 1 full.

- **RF1L: Receive FIFO 1 Message Lost**

0: No Receive FIFO 1 message lost.

1: Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero.

- **HPM: High Priority Message**

0: No high priority message received.

1: High priority message received.

- **TC: Transmission Completed**

0: No transmission completed.

1: Transmission completed.

- **TCF: Transmission Cancellation Finished**

0: No transmission cancellation finished.

1: Transmission cancellation finished.

- **TFE: Tx FIFO Empty**

0: Tx FIFO non-empty.

1: Tx FIFO empty.

- **TEFN: Tx Event FIFO New Entry**

0: Tx Event FIFO unchanged.

1: Tx Handler wrote Tx Event FIFO element.

- **TEFW: Tx Event FIFO Watermark Reached**

0: Tx Event FIFO fill level below watermark.

1: Tx Event FIFO fill level reached watermark.

- **TEFF: Tx Event FIFO Full**

0: Tx Event FIFO not full.

1: Tx Event FIFO full.

- **TEFL: Tx Event FIFO Element Lost**

0: No Tx Event FIFO element lost.

1: Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.

- **TSW: Timestamp Wraparound**

0: No timestamp counter wrap-around.

1: Timestamp counter wrapped around.

- **MRAF: Message RAM Access Failure**

The flag is set, when the Rx Handler

- has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message.
- was not able to write a message to the Message RAM. In this case message storage is aborted.

In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Receive Buffer is not set, a partly stored message is overwritten when the next message is stored to this location.

The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the MCAN is switched into Restricted Operation mode (see Section 50.5.1.5). To leave Restricted Operation mode, the processor has to reset MCAN_CCCR.ASM.

0: No Message RAM access failure occurred.

1: Message RAM access failure occurred.

- **TOO: Timeout Occurred**

0: No timeout.

1: Timeout reached.

- **DRX: Message stored to Dedicated Receive Buffer**

The flag is set whenever a received message has been stored into a dedicated Receive Buffer.

0: No Receive Buffer updated.

1: At least one received message stored into a Receive Buffer.

- **ELO: Error Logging Overflow**

0: CAN Error Logging Counter did not overflow.

1: Overflow of CAN Error Logging Counter occurred.

- **EP: Error Passive**

0: Error_Passive status unchanged.

1: Error_Passive status changed.

- **EW: Warning Status**

0: Error_Warning status unchanged.

1: Error_Warning status changed.

- **BO: Bus_Off Status**

0: Bus_Off status unchanged.

1: Bus_Off status changed.

- **WDI: Watchdog Interrupt**

0: No Message RAM Watchdog event occurred.

1: Message RAM Watchdog event due to missing READY.

- **PEA: Protocol Error in Arbitration Phase**

0: No protocol error in arbitration phase

1: Protocol error in arbitration phase detected (MCAN_PSR.LEC differs from 0 or 7)

- **PED: Protocol Error in Data Phase**

0: No protocol error in data phase

1: Protocol error in data phase detected (MCAN_PSR.DLEC differs from 0 or 7)

- **ARA: Access to Reserved Address**

0: No access to reserved address occurred

1: Access to reserved address occurred

50.6.17 MCAN Interrupt Enable Register

Name: MCAN_IE

Address: 0xF8054054 (0), 0xFC050054 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ARAE	PEDE	PEAE	WDIE	BOE	EWE
23	22	21	20	19	18	17	16
EPE	ELOE	–	–	DRXE	TOOE	MRAFE	TSWE
15	14	13	12	11	10	9	8
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME
7	6	5	4	3	2	1	0
RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE

The following configuration values are valid for all listed bit names of this register:

0: Disables the corresponding interrupt.

1: Enables the corresponding interrupt.

- **RF0NE: Receive FIFO 0 New Message Interrupt Enable**
- **RF0WE: Receive FIFO 0 Watermark Reached Interrupt Enable**
- **RF0FE: Receive FIFO 0 Full Interrupt Enable**
- **RF0LE: Receive FIFO 0 Message Lost Interrupt Enable**
- **RF1NE: Receive FIFO 1 New Message Interrupt Enable**
- **RF1WE: Receive FIFO 1 Watermark Reached Interrupt Enable**
- **RF1FE: Receive FIFO 1 Full Interrupt Enable**
- **RF1LE: Receive FIFO 1 Message Lost Interrupt Enable**
- **HPME: High Priority Message Interrupt Enable**
- **TCE: Transmission Completed Interrupt Enable**
- **TCFE: Transmission Cancellation Finished Interrupt Enable**
- **TFEE: Tx FIFO Empty Interrupt Enable**
- **TEFNE: Tx Event FIFO New Entry Interrupt Enable**
- **TEFWE: Tx Event FIFO Watermark Reached Interrupt Enable**
- **TEFFE: Tx Event FIFO Full Interrupt Enable**
- **TEFLE: Tx Event FIFO Event Lost Interrupt Enable**
- **TSWE: Timestamp Wraparound Interrupt Enable**

- **MRAFE: Message RAM Access Failure Interrupt Enable**
- **TOOE: Timeout Occurred Interrupt Enable**
- **DRXE: Message stored to Dedicated Receive Buffer Interrupt Enable**
- **ELOE: Error Logging Overflow Interrupt Enable**
- **EPE: Error Passive Interrupt Enable**
- **EWE: Warning Status Interrupt Enable**
- **BOE: Bus_Off Status Interrupt Enable**
- **WDIE: Watchdog Interrupt Enable**
- **PEAE: Protocol Error in Arbitration Phase Enable**
- **PEDE: Protocol Error in Data Phase Enable**
- **ARAE: Access to Reserved Address Enable**

50.6.18 MCAN Interrupt Line Select Register

Name: MCAN_ILS

Address: 0xF8054058 (0), 0xFC050058 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ARAL	PEDL	PEAL	WDIL	BOL	EWL
23	22	21	20	19	18	17	16
EPL	ELOL	–	–	DRXL	TOOL	MRAFL	TSWL
15	14	13	12	11	10	9	8
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML
7	6	5	4	3	2	1	0
RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines.

0: Interrupt assigned to interrupt line MCAN_INT0.

1: Interrupt assigned to interrupt line MCAN_INT1.

- **RF0NL: Receive FIFO 0 New Message Interrupt Line**
- **RF0WL: Receive FIFO 0 Watermark Reached Interrupt Line**
- **RF0FL: Receive FIFO 0 Full Interrupt Line**
- **RF0LL: Receive FIFO 0 Message Lost Interrupt Line**
- **RF1NL: Receive FIFO 1 New Message Interrupt Line**
- **RF1WL: Receive FIFO 1 Watermark Reached Interrupt Line**
- **RF1FL: Receive FIFO 1 Full Interrupt Line**
- **RF1LL: Receive FIFO 1 Message Lost Interrupt Line**
- **HPML: High Priority Message Interrupt Line**
- **TCL: Transmission Completed Interrupt Line**
- **TCFL: Transmission Cancellation Finished Interrupt Line**
- **TFEL: Tx FIFO Empty Interrupt Line**
- **TEFNL: Tx Event FIFO New Entry Interrupt Line**
- **TEFWL: Tx Event FIFO Watermark Reached Interrupt Line**
- **TEFFL: Tx Event FIFO Full Interrupt Line**
- **TEFLL: Tx Event FIFO Event Lost Interrupt Line**
- **TSWL: Timestamp Wraparound Interrupt Line**

- **MRAFL: Message RAM Access Failure Interrupt Line**
- **TOOL: Timeout Occurred Interrupt Line**
- **DRXL: Message stored to Dedicated Receive Buffer Interrupt Line**
- **ELOL: Error Logging Overflow Interrupt Line**
- **EPL: Error Passive Interrupt Line**
- **EWL: Warning Status Interrupt Line**
- **BOL: Bus_Off Status Interrupt Line**
- **WDIL: Watchdog Interrupt Line**
- **PEAL: Protocol Error in Arbitration Phase Line**
- **PEDL: Protocol Error in Data Phase Line**
- **ARAL: Access to Reserved Address Line**

50.6.19 MCAN Interrupt Line Enable

Name: MCAN_ILE

Address: 0xF805405C (0), 0xFC05005C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	EINT1	EINT0

Each of the two interrupt lines to the processor can be enabled / disabled separately by programming bits EINT0 and EINT1.

- **EINT0: Enable Interrupt Line 0**

0: Interrupt line MCAN_INT0 disabled.

1: Interrupt line MCAN_INT0 enabled.

- **EINT1: Enable Interrupt Line 1**

0: Interrupt line MCAN_INT1 disabled.

1: Interrupt line MCAN_INT1 enabled.

50.6.20 MCAN Global Filter Configuration

Name: MCAN_GFC

Address: 0xF8054080 (0), 0xFC050080 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	ANFS		ANFE		RRFS	RRFE

Global settings for Message ID filtering. The Global Filter Configuration controls the filter path for standard and extended messages as described in Figure 50-5 and Figure 50-6.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **RRFE: Reject Remote Frames Extended**

0 (FILTER): Filter remote frames with 29-bit extended IDs.

1 (REJECT): Reject all remote frames with 29-bit extended IDs.

- **RRFS: Reject Remote Frames Standard**

0 (FILTER): Filter remote frames with 11-bit standard IDs.

1 (REJECT): Reject all remote frames with 11-bit standard IDs.

- **ANFE: Accept Non-matching Frames Extended**

Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated.

Value	Name	Description
0	RX_FIFO_0	Accept in Rx FIFO 0
1	RX_FIFO_1	Accept in Rx FIFO 1
2-3	REJECTED	Message rejected

- **ANFS: Accept Non-matching Frames Standard**

Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated.

Value	Name	Description
0	RX_FIFO_0	Accept in Rx FIFO 0
1	RX_FIFO_1	Accept in Rx FIFO 1
2-3	REJECTED	Message rejected

50.6.21 MCAN Standard ID Filter Configuration

Name: MCAN_SIDFC

Address: 0xF8054084 (0), 0xFC050084 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
LSS							
15	14	13	12	11	10	9	8
FLSSA							
7	6	5	4	3	2	1	0
FLSSA						–	–

Settings for 11-bit standard Message ID filtering. The Standard ID Filter Configuration controls the filter path for standard messages as described in Figure 50-5.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **FLSSA: Filter List Standard Start Address**

Start address of standard Message ID filter list (32-bit word address, see Figure 50-12).

Write FLSSA with the bits [15:2] of the 32-bit address.

- **LSS: List Size Standard**

0: No standard Message ID filter.

1-128: Number of standard Message ID filter elements.

>128: Values greater than 128 are interpreted as 128.

50.6.22 MCAN Extended ID Filter Configuration

Name: MCAN_XIDFC

Address: 0xF8054088 (0), 0xFC050088 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	LSE						
15	14	13	12	11	10	9	8
FLESA							
7	6	5	4	3	2	1	0
FLESA						–	–

Settings for 29-bit extended Message ID filtering. The Extended ID Filter Configuration controls the filter path for standard messages as described in Figure 50-6.

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **FLESA: Filter List Extended Start Address**

Start address of extended Message ID filter list (32-bit word address, see Figure 50-12).

Write FLESA with the bits [15:2] of the 32-bit address.

- **LSE: List Size Extended**

0: No extended Message ID filter.

1-64: Number of extended Message ID filter elements.

>64: Values greater than 64 are interpreted as 64.

50.6.23 MCAN Extended ID AND Mask

Name: MCAN_XIDAM

Address: 0xF8054090 (0), 0xFC050090 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	EIDM				
23	22	21	20	19	18	17	16
EIDM							
15	14	13	12	11	10	9	8
EIDM							
7	6	5	4	3	2	1	0
EIDM							

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **EIDM: Extended ID Mask**

For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active.

50.6.24 MCAN High Priority Message Status

Name: MCAN_HPMS

Address: 0xF8054094 (0), 0xFC050094 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
FLST	FIDX						
7	6	5	4	3	2	1	0
MSI		BIDX					

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

- **BIDX: Buffer Index**

Index of Receive FIFO element to which the message was stored. Only valid when MSI[1] = ‘1’.

- **MSI: Message Storage Indicator**

Value	Name	Description
0	NO_FIFO_SEL	No FIFO selected.
1	LOST	FIFO message lost.
2	FIFO_0	Message stored in FIFO 0.
3	FIFO_1	Message stored in FIFO 1.

- **FIDX: Filter Index**

Index of matching filter element. Range is 0 to MCAN_SIDFC.LSS - 1 resp. MCAN_XIDFC.LSE - 1.

- **FLST: Filter List**

Indicates the filter list of the matching filter element.

0: Standard filter list

1: Extended filter list

50.6.25 MCAN New Data 1

Name: MCAN_NDAT1

Address: 0xF8054098 (0), 0xFC050098 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24
23	22	21	20	19	18	17	16
ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
15	14	13	12	11	10	9	8
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8
7	6	5	4	3	2	1	0
ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0

- **NDx: New Data**

The register holds the New Data flags of Receive Buffers 0 to 31. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

0: Receive Buffer not updated

1: Receive Buffer updated from new message

50.6.26 MCAN New Data 2

Name: MCAN_NDAT2

Address: 0xF805409C (0), 0xFC05009C (1)

Access: Read/Write

31	30	29	28	27	26	25	24
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56
23	22	21	20	19	18	17	16
ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
15	14	13	12	11	10	9	8
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40
7	6	5	4	3	2	1	0
ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32

- **NDx: New Data**

The register holds the New Data flags of Receive Buffers 32 to 63. The flags are set when the respective Receive Buffer has been updated from a received frame. The flags remain set until the processor clears them. A flag is cleared by writing a ‘1’ to the corresponding bit position. Writing a ‘0’ has no effect. A hard reset will clear the register.

0: Receive Buffer not updated.

1: Receive Buffer updated from new message.

50.6.27 MCAN Receive FIFO 0 Configuration

Name: MCAN_RXF0C

Address: 0xF80540A0 (0), 0xFC0500A0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
F0OM	F0WM						
23	22	21	20	19	18	17	16
–	F0S						
15	14	13	12	11	10	9	8
F0SA							
7	6	5	4	3	2	1	0
F0SA						–	–

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **F0SA: Receive FIFO 0 Start Address**

Start address of Receive FIFO 0 in Message RAM (32-bit word address, see Figure 50-12).

Write F0SA with the bits [15:2] of the 32-bit address.

- **F0S: Receive FIFO 0 Size**

0: No Receive FIFO 0

1-64: Number of Receive FIFO 0 elements.

>64: Values greater than 64 are interpreted as 64.

The Receive FIFO 0 elements are indexed from 0 to F0S-1.

- **F0WM: Receive FIFO 0 Watermark**

0: Watermark interrupt disabled.

1-64: Level for Receive FIFO 0 watermark interrupt (MCAN_IR.RF0W).

>64: Watermark interrupt disabled.

- **F0OM: FIFO 0 Operation Mode**

FIFO 0 can be operated in Blocking or in Overwrite mode (see Section 50.5.4.2).

0: FIFO 0 Blocking mode.

1: FIFO 0 Overwrite mode.

50.6.28 MCAN Receive FIFO 0 Status

Name: MCAN_RXF0S

Address: 0xF80540A4 (0), 0xFC0500A4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	RF0L	F0F
23	22	21	20	19	18	17	16
–	–	F0PI					
15	14	13	12	11	10	9	8
–	–	F0GI					
7	6	5	4	3	2	1	0
–	F0FL						

- **F0FL: Receive FIFO 0 Fill Level**

Number of elements stored in Receive FIFO 0, range 0 to 64.

- **F0GI: Receive FIFO 0 Get Index**

Receive FIFO 0 read index pointer, range 0 to 63.

- **F0PI: Receive FIFO 0 Put Index**

Receive FIFO 0 write index pointer, range 0 to 63.

- **F0F: Receive FIFO 0 Full**

0: Receive FIFO 0 not full.

1: Receive FIFO 0 full.

- **RF0L: Receive FIFO 0 Message Lost**

This bit is a copy of interrupt flag MCAN_IR.RF0L. When MCAN_IR.RF0L is reset, this bit is also reset.

0: No Receive FIFO 0 message lost

1: Receive FIFO 0 message lost, also set after write attempt to Receive FIFO 0 of size zero

Note: Overwriting the oldest message when MCAN_RXF0C.F0OM = ‘1’ will not set this flag.

50.6.29 MCAN Receive FIFO 0 Acknowledge

Name: MCAN_RXF0A

Address: 0xF80540A8 (0), 0xFC0500A8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	F0AI					

- **F0AI: Receive FIFO 0 Acknowledge Index**

After the processor has read a message or a sequence of messages from Receive FIFO 0 it has to write the buffer index of the last element read from Receive FIFO 0 to F0AI. This will set the Receive FIFO 0 Get Index MCAN_RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level MCAN_RXF0S.F0FL.

50.6.30 MCAN Receive Buffer Configuration

Name: MCAN_RXBC

Address: 0xF80540AC (0), 0xFC0500AC (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RBSA							
7	6	5	4	3	2	1	0
RBSA						–	–

- **RBSA: Receive Buffer Start Address**

Configures the start address of the Receive Buffers section in the Message RAM (32-bit word address, see Figure 50-12). Also used to reference debug messages A,B,C.

Write RBSA with the bits [15:2] of the 32-bit address.

50.6.31 MCAN Receive FIFO 1 Configuration

Name: MCAN_RXF1C

Address: 0xF80540B0 (0), 0xFC0500B0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
F1OM	F1WM						
23	22	21	20	19	18	17	16
–	F1S						
15	14	13	12	11	10	9	8
F1SA							
7	6	5	4	3	2	1	0
F1SA						–	–

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **F1SA: Receive FIFO 1 Start Address**

Start address of Receive FIFO 1 in Message RAM (32-bit word address, see Figure 50-12).

Write F1SA with the bits [15:2] of the 32-bit address.

- **F1S: Receive FIFO 1 Size**

0: No Receive FIFO 1

1-64: Number of elements in Receive FIFO 1.

>64: Values greater than 64 are interpreted as 64.

The elements in Receive FIFO 1 are indexed from 0 to F1S - 1.

- **F1WM: Receive FIFO 1 Watermark**

0: Watermark interrupt disabled

1-64: Level for Receive FIFO 1 watermark interrupt (MCAN_IR.RF1W).

>64: Watermark interrupt disabled.

- **F1OM: FIFO 1 Operation Mode**

FIFO 1 can be operated in Blocking or in Overwrite mode (see Section 50.5.4.2).

0: FIFO 1 Blocking mode.

1: FIFO 1 Overwrite mode.

50.6.32 MCAN Receive FIFO 1 Status

Name: MCAN_RXF1S

Address: 0xF80540B4 (0), 0xFC0500B4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
DMS		–	–	–	–	RF1L	F1F
23	22	21	20	19	18	17	16
–	–	F1PI					
15	14	13	12	11	10	9	8
–	–	F1GI					
7	6	5	4	3	2	1	0
–	F1FL						

- **F1FL: Receive FIFO 1 Fill Level**

Number of elements stored in Receive FIFO 1, range 0 to 64.

- **F1GI: Receive FIFO 1 Get Index**

Receive FIFO 1 read index pointer, range 0 to 63.

- **F1PI: Receive FIFO 1 Put Index**

Receive FIFO 1 write index pointer, range 0 to 63.

- **F1F: Receive FIFO 1 Full**

0: Receive FIFO 1 not full.

1: Receive FIFO 1 full.

- **RF1L: Receive FIFO 1 Message Lost**

This bit is a copy of interrupt flag IR.RF1L. When IR.RF1L is reset, this bit is also reset.

0: No Receive FIFO 1 message lost.

1: Receive FIFO 1 message lost, also set after write attempt to Receive FIFO 1 of size zero.

Note: Overwriting the oldest message when MCAN_RXF1C.F1OM = ‘1’ will not set this flag.

- **DMS: Debug Message Status**

Value	Name	Description
0	IDLE	Idle state, wait for reception of debug messages, DMA request is cleared.
1	MSG_A	Debug message A received.
2	MSG_AB	Debug messages A, B received.
3	MSG_ABC	Debug messages A, B, C received, DMA request is set.

50.6.33 MCAN Receive FIFO 1 Acknowledge

Name: MCAN_RXF1A

Address: 0xF80540B8 (0), 0xFC0500B8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	F1AI					

- **F1AI: Receive FIFO 1 Acknowledge Index**

After the processor has read a message or a sequence of messages from Receive FIFO 1 it has to write the buffer index of the last element read from Receive FIFO 1 to F1AI. This will set the Receive FIFO 1 Get Index MCAN_RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level MCAN_RXF1S.F1FL.

50.6.34 MCAN Receive Buffer / FIFO Element Size Configuration

Name: MCAN_RXESC

Address: 0xF80540BC (0), 0xFC0500BC (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	RBDS		
7	6	5	4	3	2	1	0
–	F1DS			–	F0DS		

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Receive Buffer / Receive FIFO element. Data field sizes >8 bytes are intended for CAN FD operation only.

- **F0DS: Receive FIFO 0 Data Field Size**

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48-byte data field
7	64_BYTE	64-byte data field

- **F1DS: Receive FIFO 1 Data Field Size**

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48-byte data field
7	64_BYTE	64-byte data field

- **RBDS: Receive Buffer Data Field Size**

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48-byte data field
7	64_BYTE	64-byte data field

Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Receive Buffer or Receive FIFO, only the number of bytes as configured by MCAN_RXESC are stored to the Receive Buffer resp. Receive FIFO element. The rest of the frame's data field is ignored.

50.6.35 MCAN Tx Buffer Configuration

Name: MCAN_TXBC

Address: 0xF80540C0 (0), 0xFC0500C0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	TFQM	TFQS					
23	22	21	20	19	18	17	16
–	–	NDTB					
15	14	13	12	11	10	9	8
TBSA							
7	6	5	4	3	2	1	0
TBSA						–	–

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **TBSA: Tx Buffers Start Address**

Start address of Tx Buffers section in Message RAM (32-bit word address, see Figure 50-12).

Write TBSA with the bits [15:2] of the 32-bit address.

- **NDTB: Number of Dedicated Transmit Buffers**

0: No dedicated Tx Buffers.

1-32: Number of dedicated Tx Buffers.

>32: Values greater than 32 are interpreted as 32.

- **TFQS: Transmit FIFO/Queue Size**

0: No Tx FIFO/Queue.

1-32: Number of Tx Buffers used for Tx FIFO/Queue.

>32: Values greater than 32 are interpreted as 32.

- **TFQM: Tx FIFO/Queue Mode**

0: Tx FIFO operation.

1: Tx Queue operation.

Note: The sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers.

50.6.36 MCAN Tx FIFO/Queue Status

Name: MCAN_TXFQS

Address: 0xF80540C4 (0), 0xFC0500C4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	TFQF	TFQPI				
15	14	13	12	11	10	9	8
–	–	–	TFGI				
7	6	5	4	3	2	1	0
–	–	TFFL					

The Tx FIFO/Queue status is related to the pending Tx requests listed in register MCAN_TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (MCAN_TXBRP not yet updated).

- **TFFL: Tx FIFO Free Level**

Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

- **TFGI: Tx FIFO Get Index**

Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (MCAN_TXBC.TFQM = ‘1’).

- **TFQPI: Tx FIFO/Queue Put Index**

Tx FIFO/Queue write index pointer, range 0 to 31.

- **TFQF: Tx FIFO/Queue Full**

0: Tx FIFO/Queue not full.

1: Tx FIFO/Queue full.

Note: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers.
Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO.

50.6.37 MCAN Tx Buffer Element Size Configuration

Name: MCAN_TXESC

Address: 0xF80540C8 (0), 0xFC0500C8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	TBDS		

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

- **TBDS: Tx Buffer Data Field Size**

Value	Name	Description
0	8_BYTE	8-byte data field
1	12_BYTE	12-byte data field
2	16_BYTE	16-byte data field
3	20_BYTE	20-byte data field
4	24_BYTE	24-byte data field
5	32_BYTE	32-byte data field
6	48_BYTE	48- byte data field
7	64_BYTE	64-byte data field

Note: In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size MCAN_TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as “0xCC” (padding bytes).

50.6.38 MCAN Transmit Buffer Request Pending

Name: MCAN_TXBRP

Address: 0xF80540CC (0), 0xFC0500CC (1)

Access: Read-only

31	30	29	28	27	26	25	24
TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24
23	22	21	20	19	18	17	16
TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
15	14	13	12	11	10	9	8
TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8
7	6	5	4	3	2	1	0
TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0

- **TRPx: Transmission Request Pending for Buffer x**

Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register MCAN_TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register MCAN_TXBCR.

TXBRP bits are set only for those Tx Buffers configured via MCAN_TXBC. After a MCAN_TXBRP bit has been set, a Tx scan (see Section 50.5.5) is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID).

A cancellation request resets the corresponding transmission request pending bit of register MCAN_TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset.

After a cancellation has been requested, a finished cancellation is signalled via MCAN_TXBCF.

- after successful transmission together with the corresponding MCAN_TXBTO bit.
- when the transmission has not yet been started at the point of cancellation.
- when the transmission has been aborted due to lost arbitration.
- when an error occurred during frame transmission.

In DAR mode, all transmissions are automatically cancelled if they are not successful. The corresponding MCAN_TXBCF bit is set for all unsuccessful transmissions.

0: No transmission request pending

1: Transmission request pending

Note: MCAN_TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding MCAN_TXBRP bit is reset.

50.6.39 MCAN Transmit Buffer Add Request

Name: MCAN_TXBAR

Address: 0xF80540D0 (0), 0xFC0500D0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24
23	22	21	20	19	18	17	16
AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
15	14	13	12	11	10	9	8
AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8
7	6	5	4	3	2	1	0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0

- **ARx: Add Request for Transmit Buffer x**

Each Transmit Buffer has its own Add Request bit. Writing a ‘1’ will set the corresponding Add Request bit; writing a ‘0’ has no impact. This enables the processor to set transmission requests for multiple Transmit Buffers with one write to MCAN_TXBAR. MCAN_TXBAR bits are set only for those Transmit Buffers configured via TXBC. When no Transmit scan is running, the bits are reset immediately, else the bits remain set until the Transmit scan process has completed.

0: No transmission request added.

1: Transmission requested added.

Note: If an add request is applied for a Transmit Buffer with pending transmission request (corresponding MCAN_TXBRP bit already set), this Add Request is ignored.

50.6.40 MCAN Transmit Buffer Cancellation Request

Name: MCAN_TXBCR

Address: 0xF80540D4 (0), 0xFC0500D4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24
23	22	21	20	19	18	17	16
CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
15	14	13	12	11	10	9	8
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8
7	6	5	4	3	2	1	0
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0

- **CRx: Cancellation Request for Transmit Buffer x**

Each Transmit Buffer has its own Cancellation Request bit. Writing a ‘1’ will set the corresponding Cancellation Request bit; writing a ‘0’ has no impact. This enables the processor to set cancellation requests for multiple Transmit Buffers with one write to MCAN_TXBCR. MCAN_TXBCR bits are set only for those Transmit Buffers configured via TXBC. The bits remain set until the corresponding bit of MCAN_TXBRP is reset.

0: No cancellation pending.

1: Cancellation pending.

50.6.41 MCAN Transmit Buffer Transmission Occurred

Name: MCAN_TXBTO

Address: 0xF80540D8 (0), 0xFC0500D8 (1)

Access: Read-only

31	30	29	28	27	26	25	24
TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24
23	22	21	20	19	18	17	16
TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
15	14	13	12	11	10	9	8
TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8
7	6	5	4	3	2	1	0
TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0

- **TOx: Transmission Occurred for Buffer x**

Each Transmit Buffer has its own Transmission Occurred bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit of register MCAN_TXBAR.

0: No transmission occurred.

1: Transmission occurred.

50.6.42 MCAN Transmit Buffer Cancellation Finished

Name: MCAN_TXBCF

Address: 0xF80540DC (0), 0xFC0500DC (1)

Access: Read-only

31	30	29	28	27	26	25	24
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24
23	22	21	20	19	18	17	16
CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
15	14	13	12	11	10	9	8
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8
7	6	5	4	3	2	1	0
CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0

- **CFx: Cancellation Finished for Transmit Buffer x**

Each Transmit Buffer has its own Cancellation Finished bit. The bits are set when the corresponding MCAN_TXBRP bit is cleared after a cancellation was requested via MCAN_TXBCR. In case the corresponding MCAN_TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a ‘1’ to the corresponding bit of register MCAN_TXBAR.

0: No transmit buffer cancellation.

1: Transmit buffer cancellation finished.

50.6.43 MCAN Transmit Buffer Transmission Interrupt Enable

Name: MCAN_TXBTIE

Address: 0xF80540E0 (0), 0xFC0500E0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24
23	22	21	20	19	18	17	16
TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
15	14	13	12	11	10	9	8
TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8
7	6	5	4	3	2	1	0
TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0

- **TIEx: Transmission Interrupt Enable for Buffer x**

Each Transmit Buffer has its own Transmission Interrupt Enable bit.

0: Transmission interrupt disabled

1: Transmission interrupt enable

50.6.44 MCAN Transmit Buffer Cancellation Finished Interrupt Enable

Name: MCAN_TXBCIE

Address: 0xF80540E4 (0), 0xFC0500E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24
23	22	21	20	19	18	17	16
CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
15	14	13	12	11	10	9	8
CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8
7	6	5	4	3	2	1	0
CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0

- **CFIEx: Cancellation Finished Interrupt Enable for Transmit Buffer x**

Each Transmit Buffer has its own Cancellation Finished Interrupt Enable bit.

0: Cancellation finished interrupt disabled.

1: Cancellation finished interrupt enabled.

50.6.45 MCAN Transmit Event FIFO Configuration

Name: MCAN_TXEFC

Address: 0xF80540F0 (0), 0xFC0500F0 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	EFWM					
23	22	21	20	19	18	17	16
–	–	EFS					
15	14	13	12	11	10	9	8
EFSA							
7	6	5	4	3	2	1	0
EFSA						–	–

This register can only be written if the bits CCE and INIT are set in MCAN CC Control Register.

- **EFSA: Event FIFO Start Address**

Start address of Tx Event FIFO in Message RAM (32-bit word address, see Figure 50-12).

Write EFSA with the bits [15:2] of the 32-bit address.

- **EFS: Event FIFO Size**

0: Tx Event FIFO disabled.

1-32: Number of Tx Event FIFO elements.

>32: Values greater than 32 are interpreted as 32.

The Tx Event FIFO elements are indexed from 0 to EFS - 1.

- **EFWM: Event FIFO Watermark**

0: Watermark interrupt disabled.

1-32: Level for Tx Event FIFO watermark interrupt (MCAN_IR.TEFW).

>32: Watermark interrupt disabled.

50.6.46 MCAN Tx Event FIFO Status

Name: MCAN_TXEFS

Address: 0xF80540F4 (0), 0xFC0500F4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	TEFL	EFF
23	22	21	20	19	18	17	16
–	–	–	EFPI				
15	14	13	12	11	10	9	8
–	–	–	EFGI				
7	6	5	4	3	2	1	0
–	–	EFFL					

- **EFFL: Event FIFO Fill Level**

Number of elements stored in Tx Event FIFO, range 0 to 32.

- **EFGI: Event FIFO Get Index**

Tx Event FIFO read index pointer, range 0 to 31.

- **EFPI: Event FIFO Put Index**

Tx Event FIFO write index pointer, range 0 to 31.

- **EFF: Event FIFO Full**

0: Tx Event FIFO not full

1: Tx Event FIFO full

- **TEFL: Tx Event FIFO Element Lost**

This bit is a copy of interrupt flag MCAN_IR.TEFL. When MCAN_IR.TEFL is reset, this bit is also reset.

0: No Tx Event FIFO element lost

1: Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero.

50.6.47 MCAN Tx Event FIFO Acknowledge

Name: MCAN_TXEFA

Address: 0xF80540F8 (0), 0xFC0500F8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	EFAI				

- **EFAI: Event FIFO Acknowledge Index**

After the processor has read an element or a sequence of elements from the Tx Event FIFO, it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index MCAN_TXEFS.EFGI to EFAI + 1 and update the FIFO 0 Fill Level MCAN_TXEFS.EFFL.

51. Timer Counter (TC)

51.1 Description

A Timer Counter (TC) module includes three identical TC channels. The number of implemented TC modules is device-specific.

Each TC channel can be independently programmed to perform a wide range of functions including frequency measurement, event counting, interval measurement, pulse generation, delay timing and pulse width modulation.

Each channel has three external clock inputs, five internal clock inputs and two multipurpose input/output signals which can be configured by the user. Each channel drives an internal interrupt signal which can be programmed to generate processor interrupts.

The TC embeds a quadrature decoder (QDEC) connected in front of the timers and driven by TIOA0, TIOB0 and TIOB1 inputs. When enabled, the QDEC performs the input lines filtering, decoding of quadrature signals and connects to the timers/counters in order to read the position and speed of the motor through the user interface.

The TC block has two global registers which act upon all TC channels:

- Block Control Register (TC_BCR)—allows channels to be started simultaneously with the same instruction
- Block Mode Register (TC_BMR)—defines the external clock inputs for each channel, allowing them to be chained

51.2 Embedded Characteristics

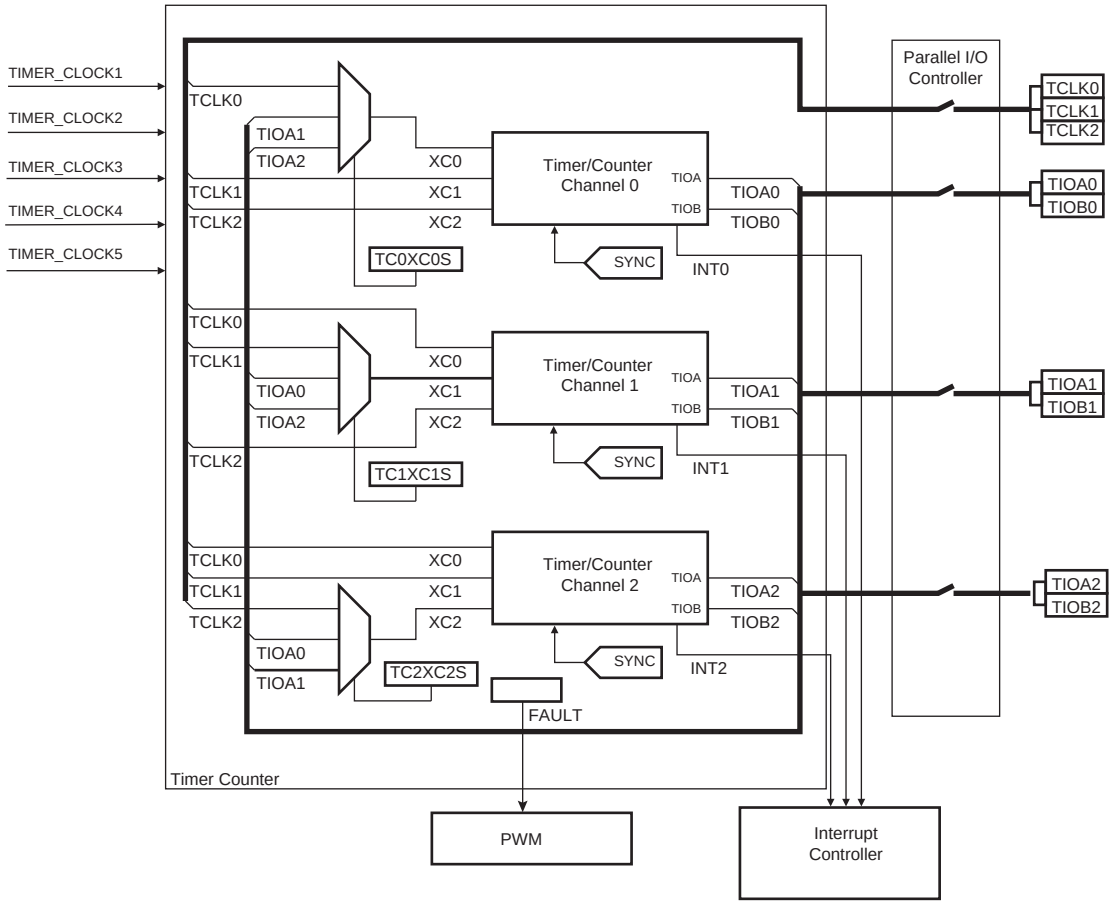
- Total number of TC channels implemented on this device: 6
- TC channel size: 32-bit
- Wide range of functions including:
  - Frequency measurement
  - Event counting
  - Interval measurement
  - Pulse generation
  - Delay timing
  - Pulse Width Modulation
  - Up/down capabilities
  - Quadrature decoder
  - 2-bit Gray up/down count for stepper motor
- Each channel is user-configurable and contains:
  - Three external clock inputs
  - Five Internal clock inputs
  - Two multipurpose input/output signals acting as trigger event
  - Trigger/capture events can be directly synchronized by PWM signals
- Internal interrupt signal
- Read of the Capture registers by the DMAC
- Compare event fault generation for PWM
- Register Write Protection

51.3 Block Diagram

Table 51-1. Timer Counter Clock Assignment

Name	Definition
TIMER_CLOCK1	GCLK [35], GCLK [36]
TIMER_CLOCK2	System bus clock divided by 8
TIMER_CLOCK3	System bus clock divided by 32
TIMER_CLOCK4	System bus clock divided by 128
TIMER_CLOCK5	slow_clock

Figure 51-1. Timer Counter Block Diagram



Note: The QDEC connections are detailed in Figure 51-17.

Table 51-2. Channel Signal Description

Signal Name	Description
XC0, XC1, XC2	External Clock Inputs
TIOAx	Capture Mode: Timer Counter Input Waveform Mode: Timer Counter Output
TIOBx	Capture Mode: Timer Counter Input Waveform Mode: Timer Counter Input/Output

Table 51-2. Channel Signal Description (Continued)

Signal Name	Description
INT	Interrupt Signal Output (internal signal)
SYNC	Synchronization Input Signal (from configuration register)

51.4 Pin List

Table 51-3. Pin List

Pin Name	Description	Type
TCLK0–TCLK2	External Clock Input	Input
TIOA0–TIOA2	I/O Line A	I/O
TIOB0–TIOB2	I/O Line B	I/O

51.5 Product Dependencies

51.5.1 I/O Lines

The pins used for interfacing the compliant external devices may be multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the TC pins to their peripheral functions.

Table 51-4. I/O Lines

Instance	Signal	I/O Line	Peripheral
TC0	TCLK0	PA21	D
TC0	TCLK1	PA29	A
TC0	TCLK1	PC5	C
TC0	TCLK1	PD13	A
TC0	TCLK2	PB5	A
TC0	TCLK2	PB24	D
TC0	TCLK2	PD22	A
TC0	TIOA0	PA19	D
TC0	TIOA1	PA27	A
TC0	TIOA1	PC3	C
TC0	TIOA1	PD11	A
TC0	TIOA2	PB6	A
TC0	TIOA2	PB22	D
TC0	TIOA2	PD20	A
TC0	TIOB0	PA20	D
TC0	TIOB1	PA28	A
TC0	TIOB1	PC4	C
TC0	TIOB1	PD12	A
TC0	TIOB2	PB7	A
TC0	TIOB2	PB23	D

Table 51-4. I/O Lines (Continued)

Instance	Signal	I/O Line	Peripheral
TC0	TIOB2	PD21	A
TC1	TCLK3	PB8	A
TC1	TCLK3	PB21	D
TC1	TCLK3	PD31	D
TC1	TCLK4	PA11	D
TC1	TCLK4	PC11	D
TC1	TCLK5	PA8	D
TC1	TCLK5	PB30	D
TC1	TIOA3	PB9	A
TC1	TIOA3	PB19	D
TC1	TIOA3	PD29	D
TC1	TIOA4	PA9	D
TC1	TIOA4	PC9	D
TC1	TIOA5	PA6	D
TC1	TIOA5	PB28	D
TC1	TIOB3	PB10	A
TC1	TIOB3	PB20	D
TC1	TIOB3	PD30	D
TC1	TIOB4	PA10	D
TC1	TIOB4	PC10	D
TC1	TIOB5	PA7	D
TC1	TIOB5	PB29	D

51.5.2 Power Management

The TC is clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the Timer Counter clock.

51.5.3 Interrupt Sources

The TC has an interrupt line connected to the interrupt controller. Handling the TC interrupt requires programming the interrupt controller before configuring the TC.

Table 51-5. Peripheral IDs

Instance	ID
TC0	35
TC1	36

51.5.4 Synchronization Inputs from PWM

The TC has trigger/capture inputs internally connected to the PWM. Refer to Section 51.6.14 “Synchronization with PWM” and to the implementation of the Pulse Width Modulation (PWM) in this product.

51.5.5 Fault Output

The TC has the FAULT output internally connected to the fault input of PWM. Refer to Section 51.6.18 “Fault Mode” and to the implementation of the Pulse Width Modulation (PWM) in this product.

51.6 Functional Description

51.6.1 Description

All channels of the Timer Counter are independent and identical in operation except when the QDEC is enabled. The registers for channel programming are listed in Table 51-6 “Register Mapping”.

51.6.2 32-bit Counter

Each 32-bit channel is organized around a 32-bit counter. The value of the counter is incremented at each positive edge of the selected clock. When the counter has reached the value 2^{32}-1 and passes to zero, an overflow occurs and the COVFS bit in the TC Status Register (TC_SR) is set.

The current value of the counter is accessible in real time by reading the TC Counter Value Register (TC_CV). The counter can be reset by a trigger. In this case, the counter value passes to zero on the next valid edge of the selected clock.

51.6.3 Clock Selection

At block level, input clock signals of each channel can be connected either to the external inputs TCLKx, or to the internal I/O signals TIOAx for chaining(1) by programming the TC Block Mode Register (TC_BMR). See Figure 51-2.

Each channel can independently select an internal or external clock source for its counter(2):

- External clock signals: XC0, XC1 or XC2
- Internal clock signals: GCLK [35], GCLK [36], System bus clock divided by 8, System bus clock divided by 32, System bus clock divided by 128, slow_clock

This selection is made by the TCCLKS bits in the TC Channel Mode Register (TC_CMR).

The selected clock can be inverted with the CLKI bit in the TC_CMR. This allows counting on the opposite edges of the clock.

The burst function allows the clock to be validated when an external signal is high. The BURST parameter in the TC_CMR defines this signal (none, XC0, XC1, XC2). See Figure 51-3.

Notes:
1. In Waveform mode, to chain two timers, it is mandatory to initialize some parameters:
   - Configure TIOx outputs to 1 or 0 by writing the required value to TC_CMR.ASWTRG.
   - Bit TC_BCR.SYNC must be written to 1 to start the channels at the same time.
2. In all cases, if an external clock or asynchronous internal clock GCLK is used, the duration of each of its levels must be longer than the peripheral clock period, so the clock frequency will be at least 2.5 times lower than the peripheral clock.

Figure 51-2. Clock Chaining Selection

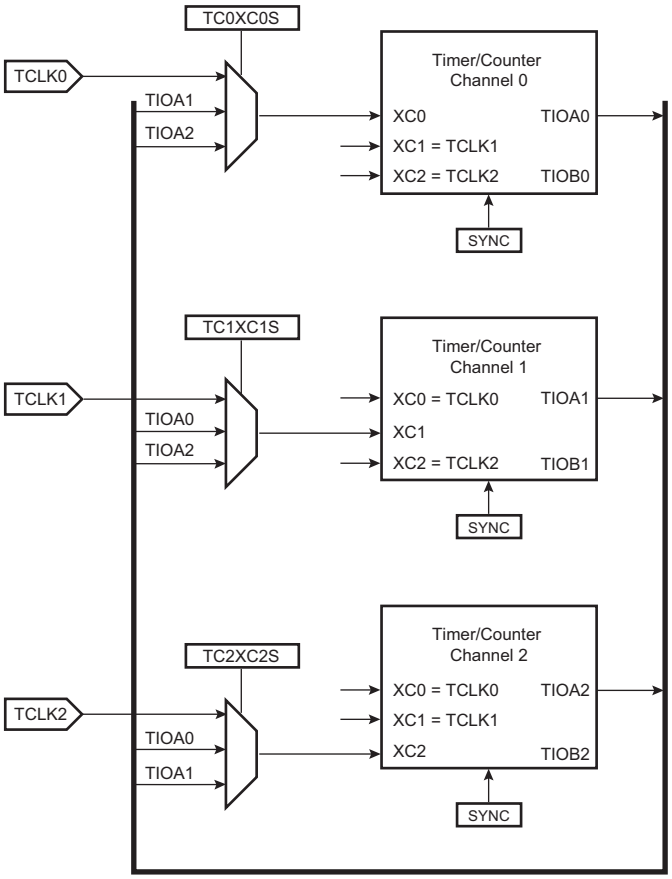
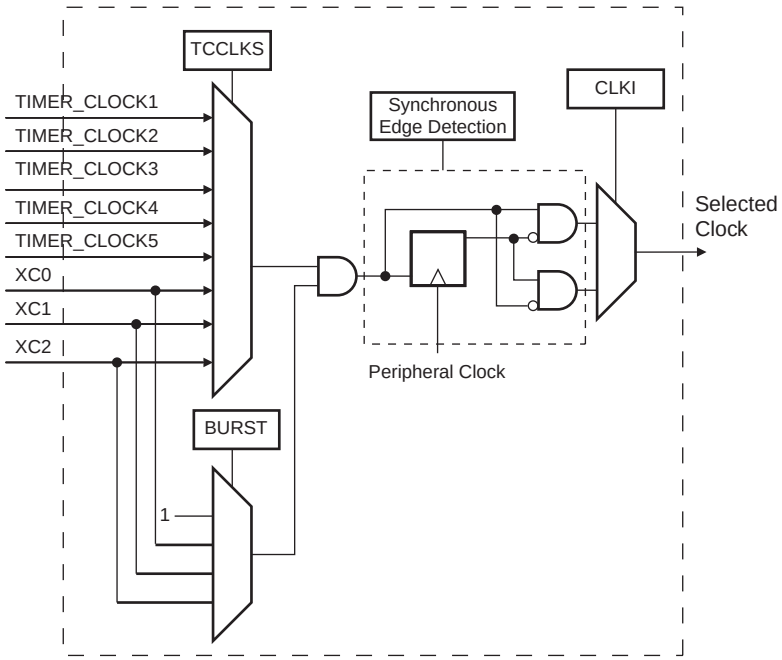


Figure 51-3. Clock Selection

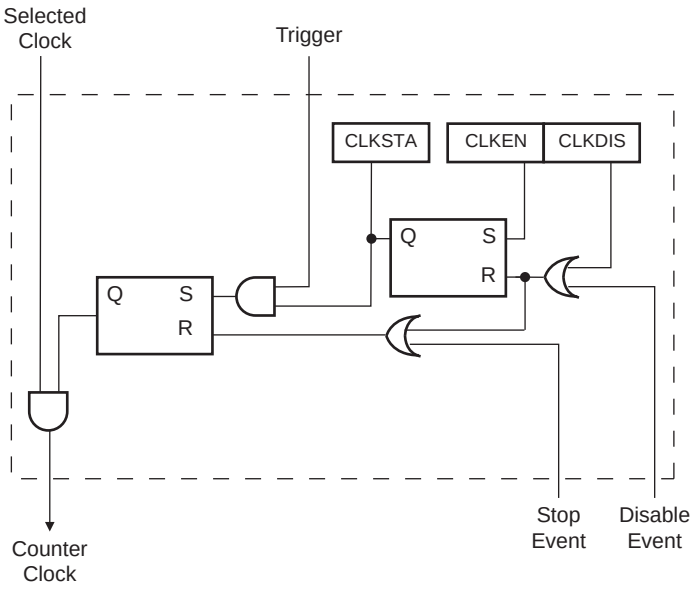


51.6.4 Clock Control

The clock of each counter can be controlled in two different ways: it can be enabled/disabled and started/stopped. See Figure 51-4.

- The clock can be enabled or disabled by the user with the CLKEN and the CLKDIS commands in the TC Channel Control Register (TC_CCR). In Capture mode it can be disabled by an RB load event if LDBDIS is set to 1 in the TC_CMR. In Waveform mode, it can be disabled by an RC Compare event if CPCDIS is set to 1 in TC_CMR. When disabled, the start or the stop actions have no effect: only a CLKEN command in the TC_CCR can re-enable the clock. When the clock is enabled, the CLKSTA bit is set in the TC_SR.
- The clock can also be started or stopped: a trigger (software, synchro, external or compare) always starts the clock. The clock can be stopped by an RB load event in Capture mode (LDBSTOP = 1 in TC_CMR) or an RC compare event in Waveform mode (CPCSTOP = 1 in TC_CMR). The start and the stop commands are effective only if the clock is enabled.

Figure 51-4. Clock Control



51.6.5 Operating Modes

Each channel can operate independently in two different modes:

- Capture mode provides measurement on signals.
- Waveform mode provides wave generation.

The TC operating mode is programmed with the WAVE bit in the TC_CMR.

In Capture mode, TIOAx and TIOBx are configured as inputs.

In Waveform mode, TIOAx is always configured to be an output and TIOBx is an output if it is not selected to be the external trigger.

51.6.6 Trigger

A trigger resets the counter and starts the counter clock. Three types of triggers are common to both modes, and a fourth external trigger is available to each mode.

Regardless of the trigger used, it will be taken into account at the following active edge of the selected clock. This means that the counter value can be read differently from zero just after a trigger, especially when a low frequency signal is selected as the clock.

The following triggers are common to both modes:

- Software Trigger: Each channel has a software trigger, available by setting SWTRG in TC_CCR.
- SYNC: Each channel has a synchronization signal SYNC. When asserted, this signal has the same effect as a software trigger. The SYNC signals of all channels are asserted simultaneously by writing TC_BCR (Block Control) with SYNC set.
- Compare RC Trigger: RC is implemented in each channel and can provide a trigger when the counter value matches the RC value if CPCTRG is set in the TC_CMR.

The channel can also be configured to have an external trigger. In Capture mode, the external trigger signal can be selected between TIOAx and TIOBx. In Waveform mode, an external event can be programmed on one of the following signals: TIOBx, XC0, XC1 or XC2. This external event can then be programmed to perform a trigger by setting bit ENETRG in the TC_CMR.

If an external trigger is used, the duration of the pulses must be longer than the peripheral clock period in order to be detected.

51.6.7 Capture Mode

Capture mode is entered by clearing the WAVE bit in the TC_CMR.

Capture mode allows the TC channel to perform measurements such as pulse timing, frequency, period, duty cycle and phase on TIOAx and TIOBx signals which are considered as inputs.

Figure 51-6 shows the configuration of the TC channel when programmed in Capture mode.

51.6.8 Capture Registers A and B

Registers A and B (RA and RB) are used as capture registers. They can be loaded with the counter value when a programmable event occurs on the signal TIOAx.

The LDRA field in the TC_CMR defines the TIOAx selected edge for the loading of register A, and the LDRB field defines the TIOAx selected edge for the loading of Register B.

The subsampling ratio defined by the SBSMPLR field in TC_CMR is applied to these selected edges, so that the loading of Register A and Register B occurs once every 1, 2, 4, 8 or 16 selected edges.

RA is loaded only if it has not been loaded since the last trigger or if RB has been loaded since the last loading of RA.

RB is loaded only if RA has been loaded since the last trigger or the last loading of RB.

Loading RA or RB before the read of the last value loaded sets the Overrun Error Flag (LOVRS bit) in the TC_SR. In this case, the old value is overwritten.

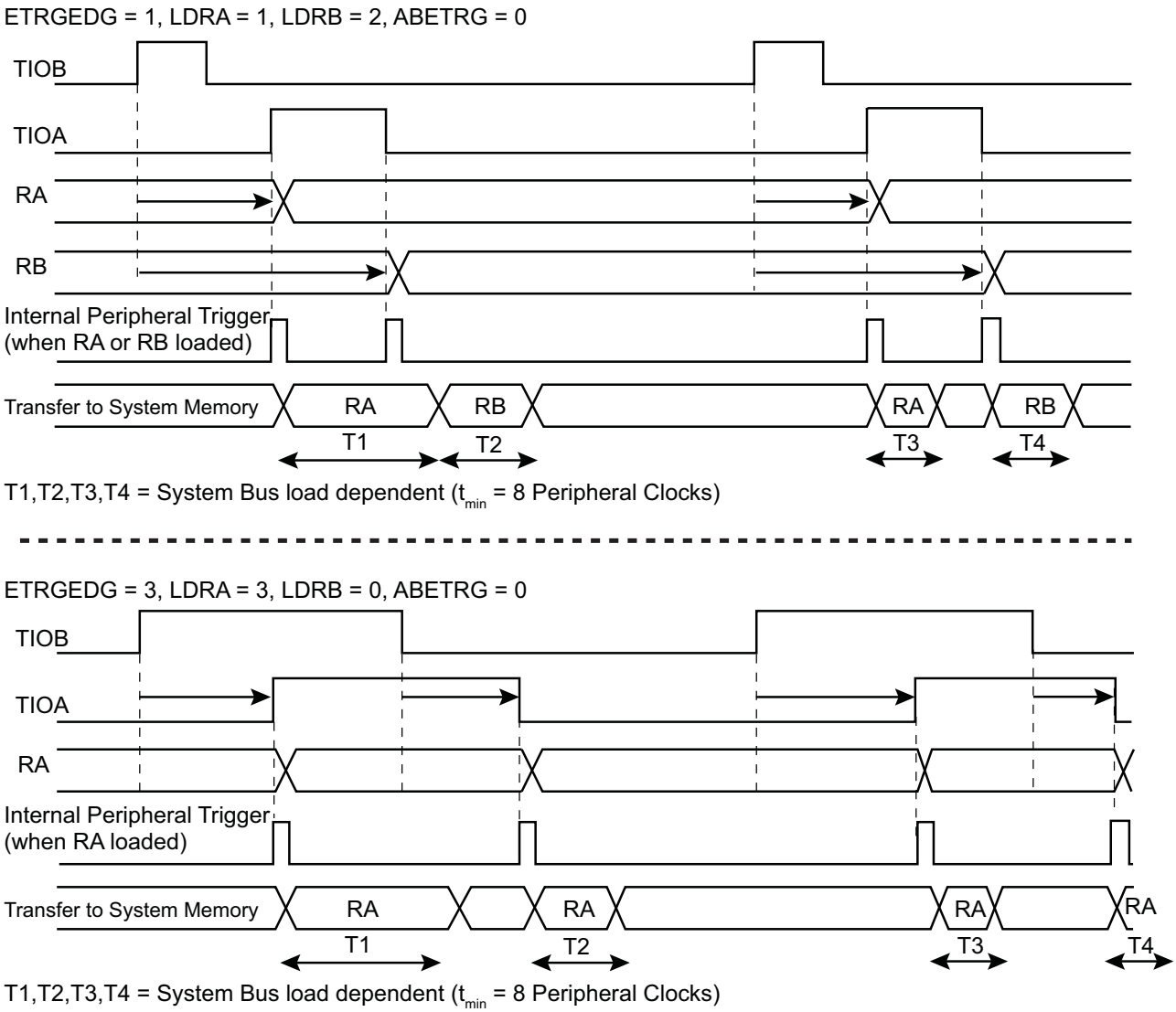
When DMA is used, the RAB register address must be configured as source address of the transfer. The RAB register provides the next unread value from Register A and Register B. It may be read by the DMA after a request has been triggered upon loading Register A or Register B.

51.6.9 Transfer with DMAC in Capture Mode

The DMAC can perform access from the TC to system memory in Capture mode only.

Figure 51-5 illustrates how TC_RA and TC_RB can be loaded in the system memory without CPU intervention.

Figure 51-5. Example of Transfer with DMAC in Capture Mode

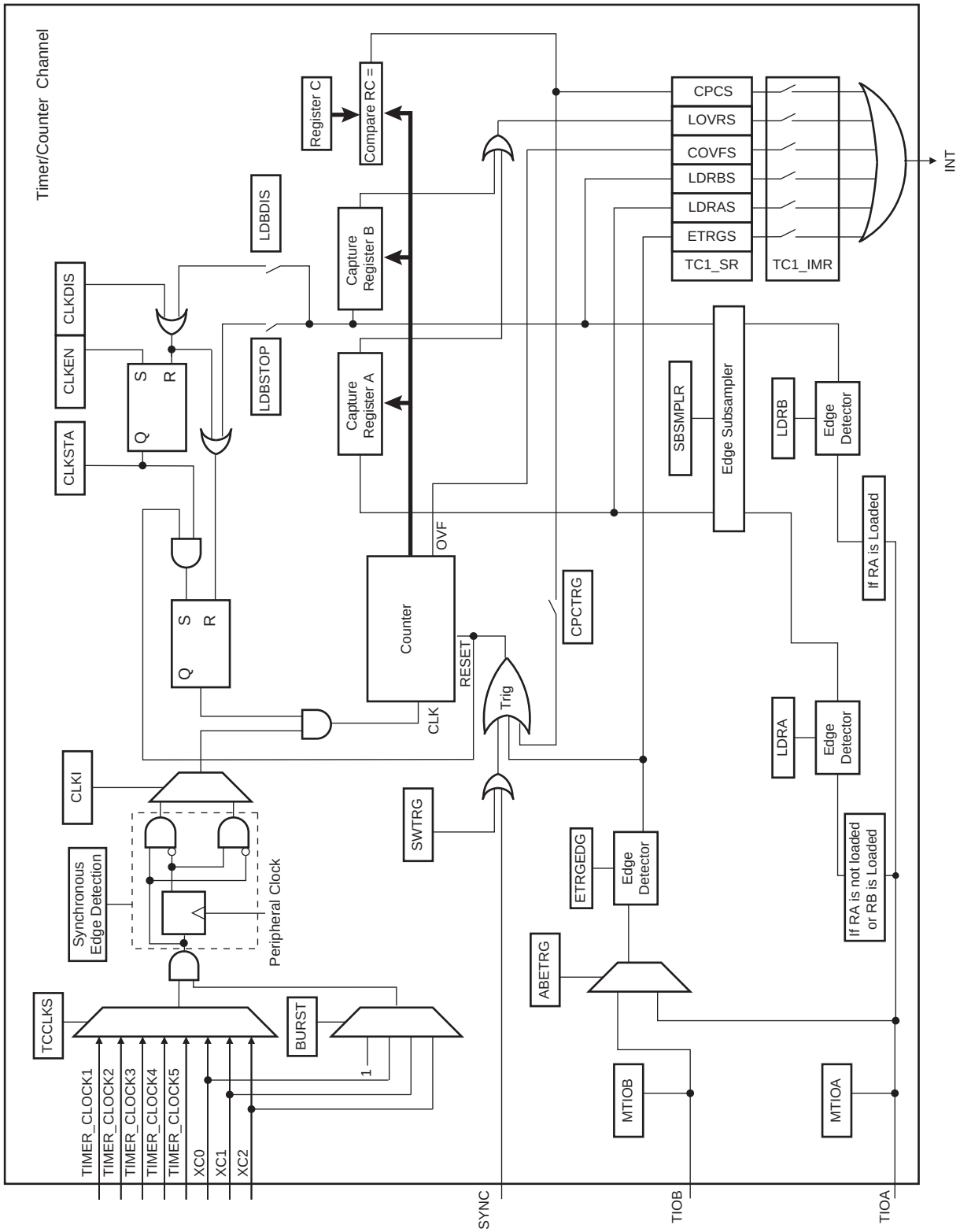


51.6.10 Trigger Conditions

In addition to the SYNC signal, the software trigger and the RC compare trigger, an external trigger can be defined.

The ABETRG bit in the TC_CMR selects TIOAx or TIOBx input signal as an external trigger or the trigger signal from the output comparator of the PWM module. The External Trigger Edge Selection parameter (ETRGEDG field in TC_CMR) defines the edge (rising, falling, or both) detected to generate an external trigger. If ETRGEDG = 0 (none), the external trigger is disabled.

Figure 51-6. Capture Mode



51.6.11 Waveform Mode

Waveform mode is entered by setting the TC_CMRx.WAVE bit.

In Waveform mode, the TC channel generates one or two PWM signals with the same frequency and independently programmable duty cycles, or generates different types of one-shot or repetitive pulses.

In this mode, TIOAx is configured as an output and TIOBx is defined as an output if it is not used as an external event (EEVT parameter in TC_CMR).

Figure 51-7 shows the configuration of the TC channel when programmed in Waveform operating mode.

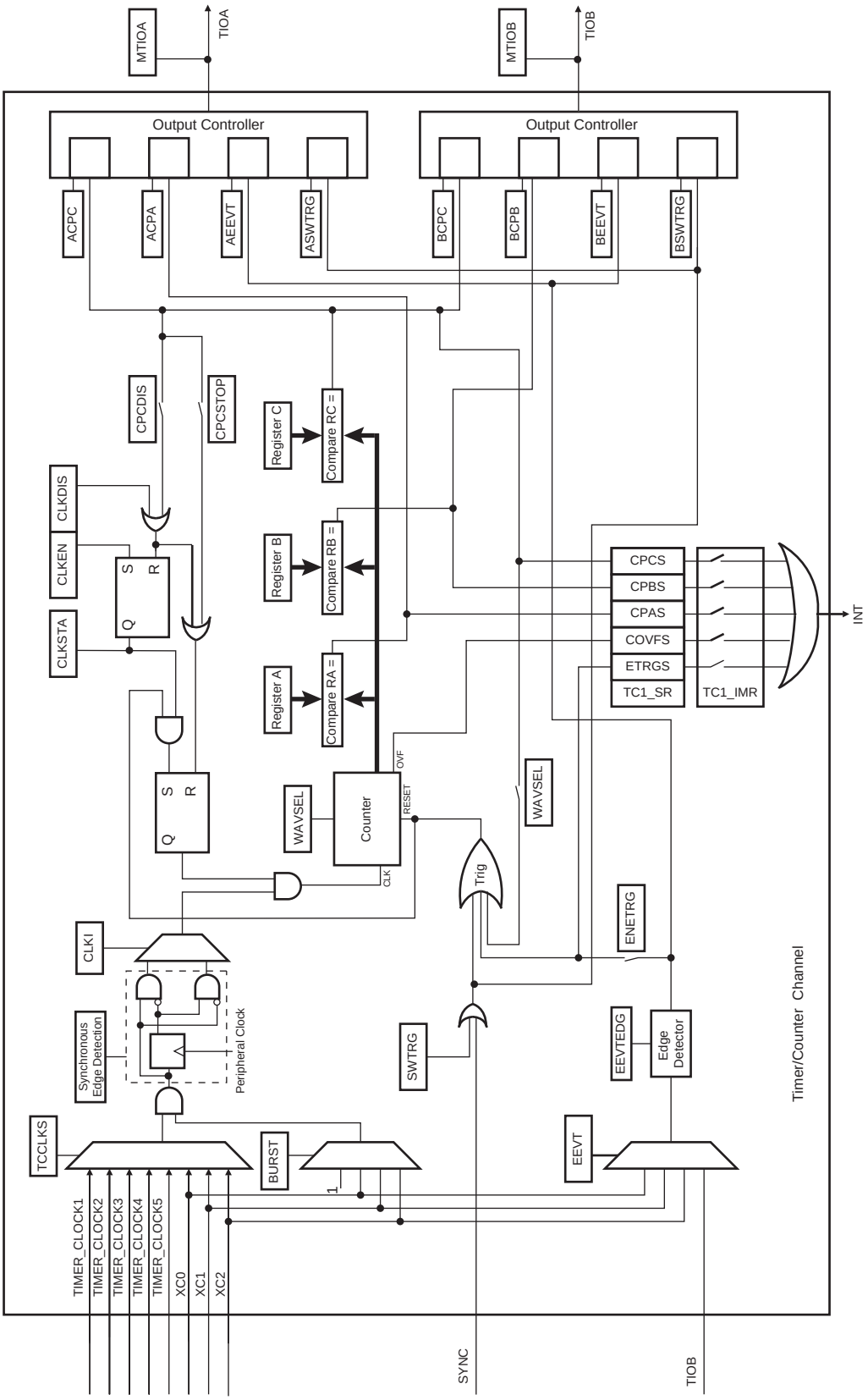
51.6.12 Waveform Selection

Depending on the WAVSEL parameter in TC_CMR, the behavior of TC_CV varies.

With any selection, TC_RA, TC_RB and TC_RC can all be used as compare registers.

RA Compare is used to control the TIOAx output, RB Compare is used to control the TIOBx output (if correctly configured) and RC Compare is used to control TIOAx and/or TIOBx outputs.

Figure 51-7. Waveform Mode



51.6.12.1 WAVSEL = 00

When WAVSEL = 00, the value of TC_CV is incremented from 0 to 2^{32}-1. Once 2^{32}-1 has been reached, the value of TC_CV is reset. Incrementation of TC_CV starts again and the cycle continues. See Figure 51-8.

An external event trigger or a software trigger can reset the value of TC_CV. It is important to note that the trigger may occur at any time. See Figure 51-9.

RC Compare cannot be programmed to generate a trigger in this configuration. At the same time, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

Figure 51-8. WAVSEL = 00 without Trigger

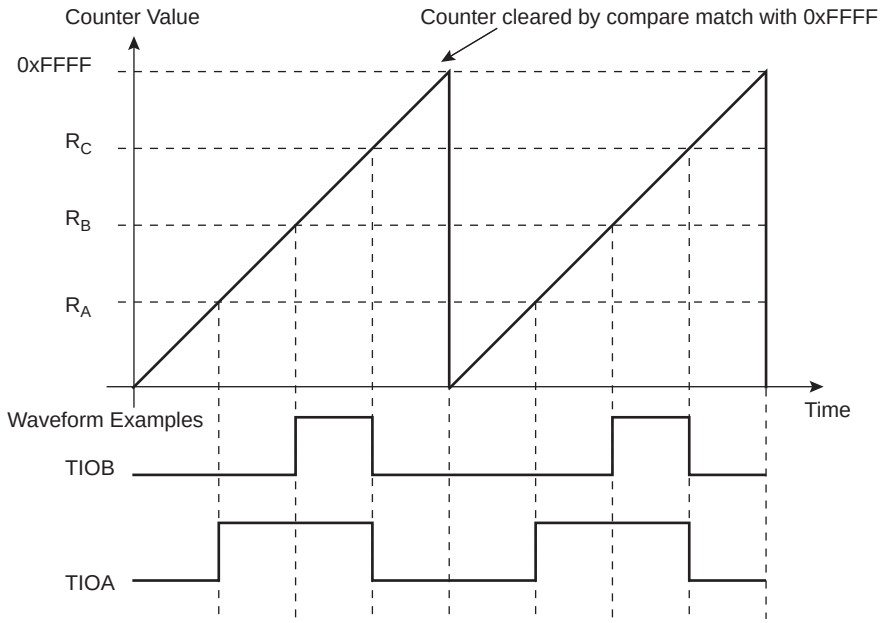
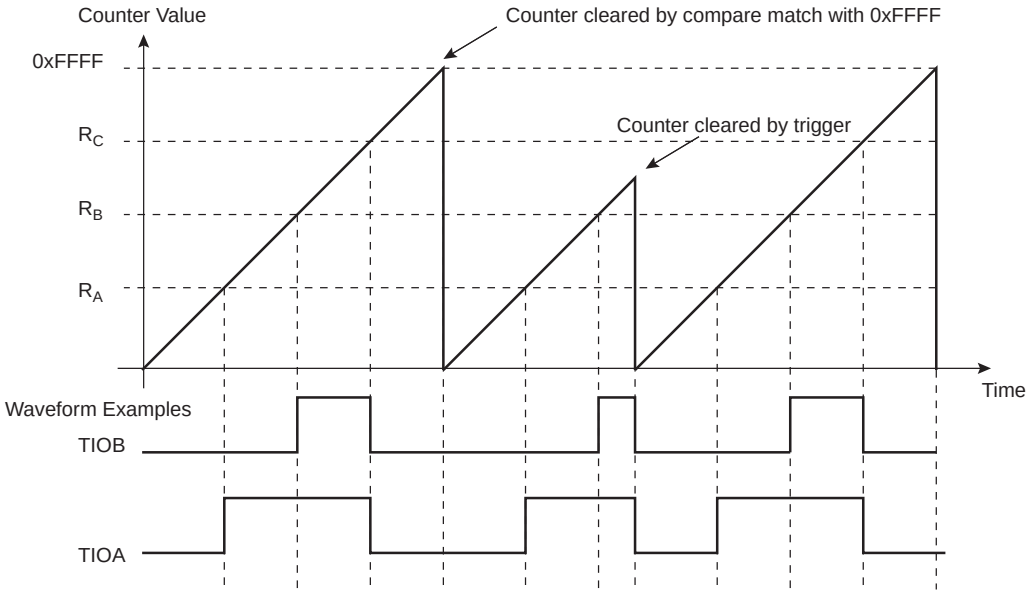


Figure 51-9. WAVSEL = 00 with Trigger



51.6.12.2 WAVSEL = 10

When WAVSEL = 10, the value of TC_CV is incremented from 0 to the value of RC, then automatically reset on a RC Compare. Once the value of TC_CV has been reset, it is then incremented and so on. See Figure 51-10.

It is important to note that TC_CV can be reset at any time by an external event or a software trigger if both are programmed correctly. See Figure 51-11.

In addition, RC Compare can stop the counter clock (CPCSTOP = 1 in TC_CMR) and/or disable the counter clock (CPCDIS = 1 in TC_CMR).

Figure 51-10. WAVSEL = 10 without Trigger

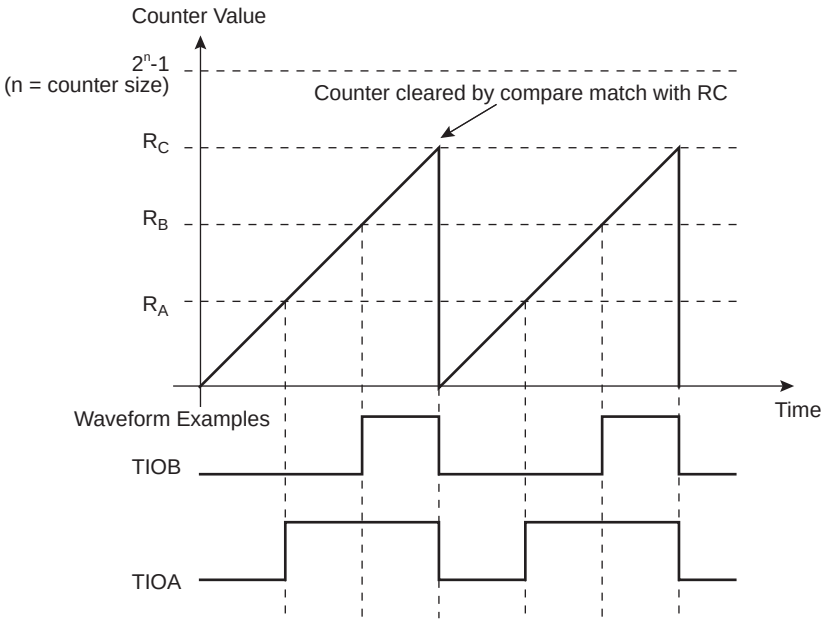
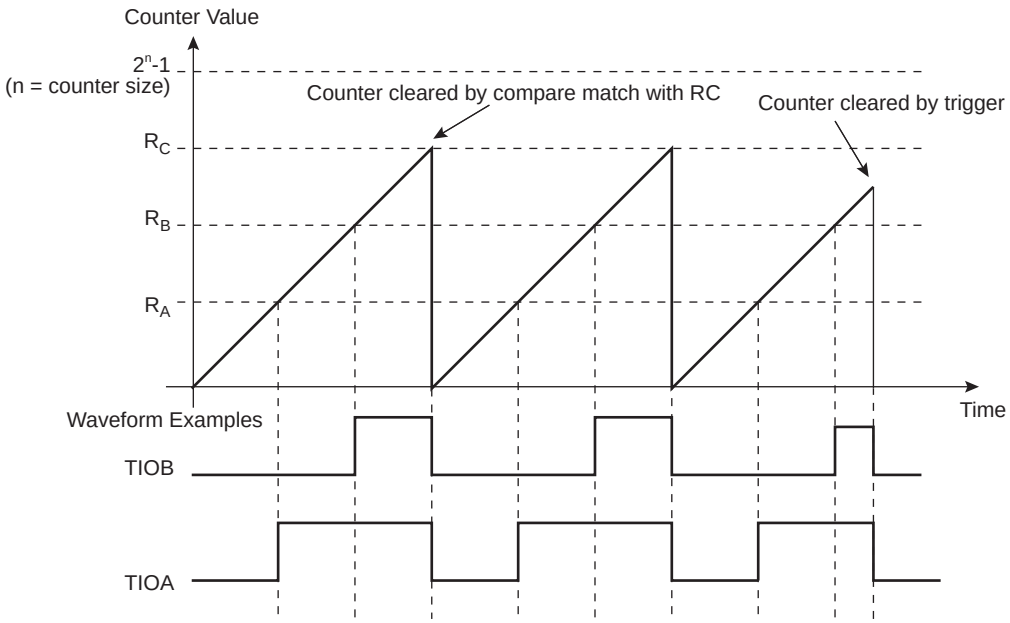


Figure 51-11. WAVSEL = 10 with Trigger



51.6.12.3 WAVSEL = 01

When WAVSEL = 01, the value of TC_CV is incremented from 0 to 2^{32}-1 . Once 2^{32}-1 is reached, the value of TC_CV is decremented to 0, then reincremented to 2^{32}-1 and so on. See Figure 51-12.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments. See Figure 51-13.

RC Compare cannot be programmed to generate a trigger in this configuration.

At the same time, RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

Figure 51-12. WAVSEL = 01 without Trigger

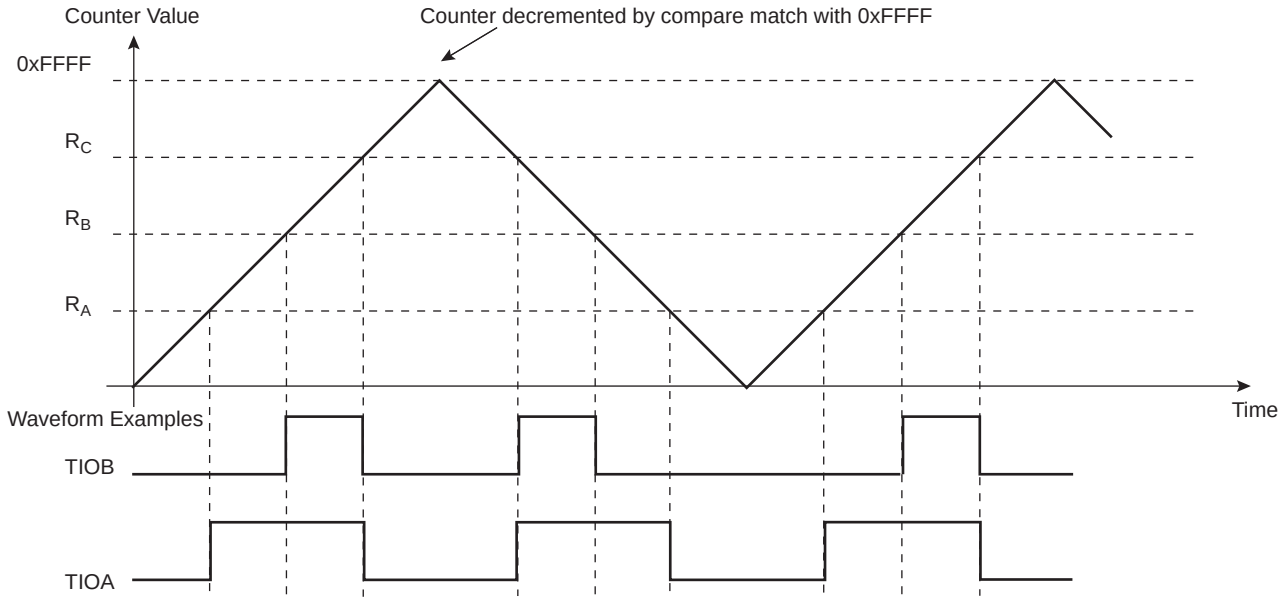
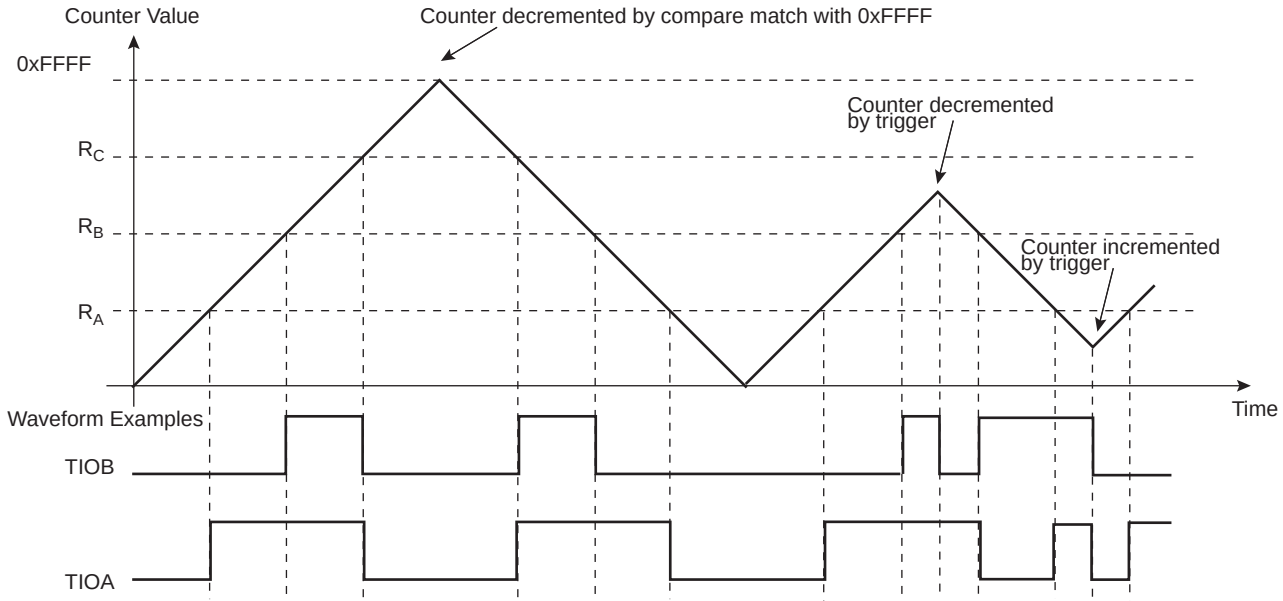


Figure 51-13. WAVSEL = 01 with Trigger



51.6.12.4 WAVSEL = 11

When WAVSEL = 11, the value of TC_CV is incremented from 0 to RC. Once RC is reached, the value of TC_CV is decremented to 0, then reincremented to RC and so on. See Figure 51-14.

A trigger such as an external event or a software trigger can modify TC_CV at any time. If a trigger occurs while TC_CV is incrementing, TC_CV then decrements. If a trigger is received while TC_CV is decrementing, TC_CV then increments. See Figure 51-15.

RC Compare can stop the counter clock (CPCSTOP = 1) and/or disable the counter clock (CPCDIS = 1).

Figure 51-14. WAVSEL = 11 without Trigger

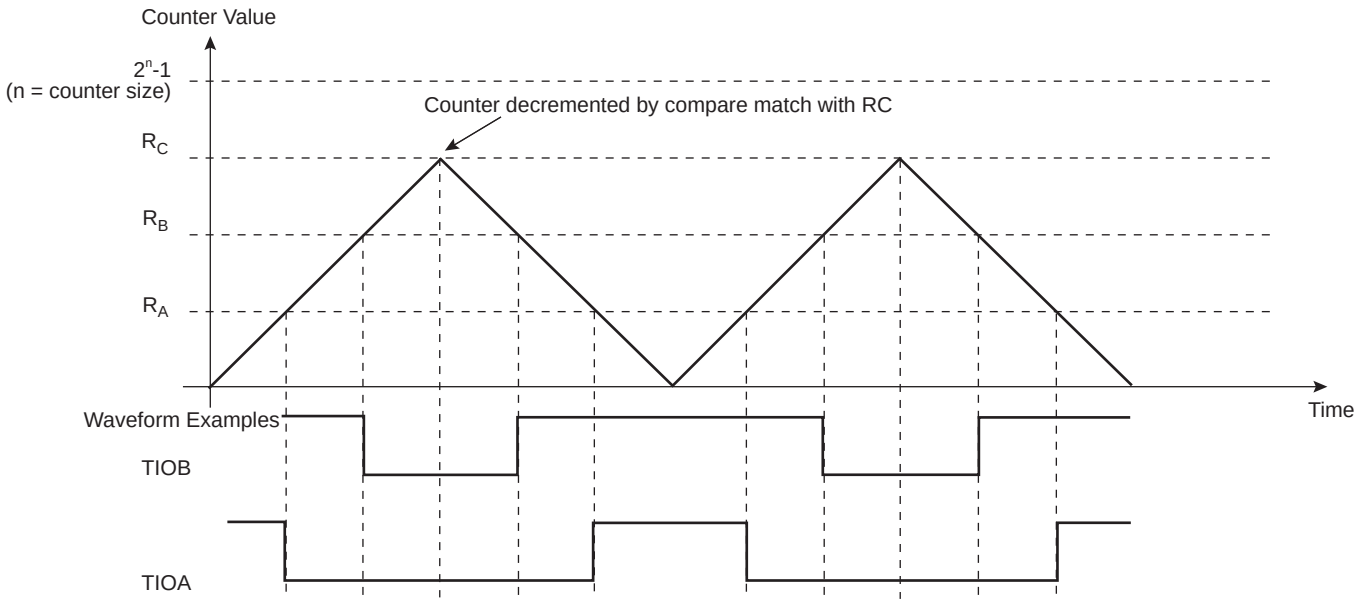
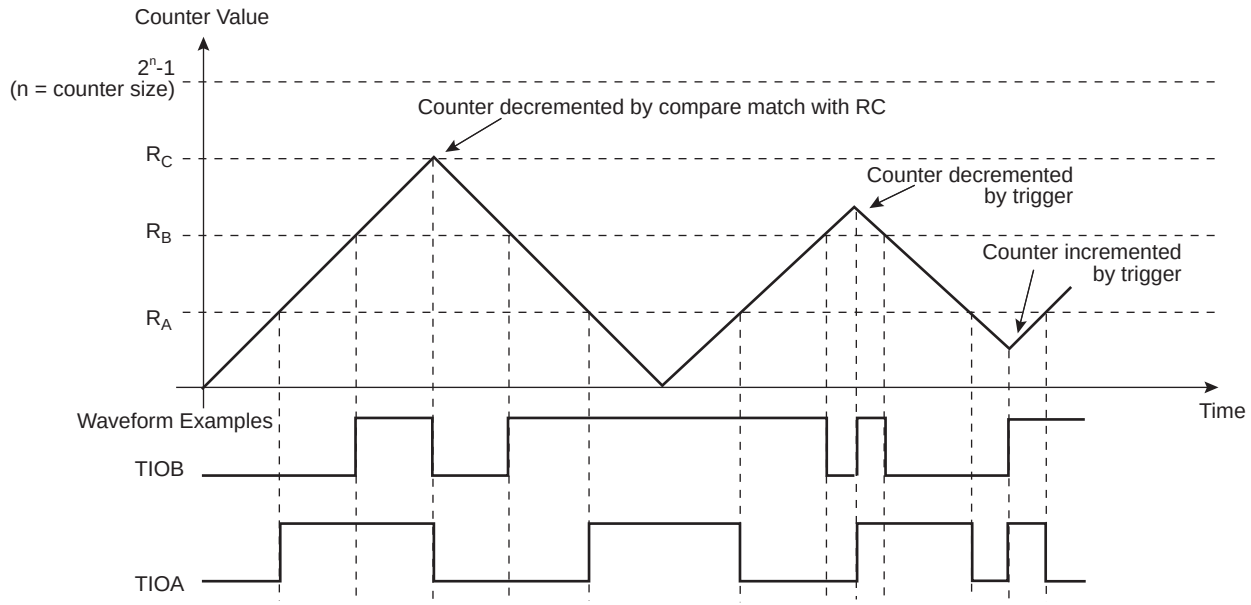


Figure 51-15. WAVSEL = 11 with Trigger



51.6.13 External Event/Trigger Conditions

An external event can be programmed to be detected on one of the clock sources (XC0, XC1, XC2) or TIOBx. The external event selected can then be used as a trigger.

The EEVT parameter in TC_CMR selects the external trigger. The EEVTEDG parameter defines the trigger edge for each of the possible external triggers (rising, falling or both). If EEVTEDG is cleared (none), no external event is defined.

If TIOBx is defined as an external event signal (EEVT = 0), TIOBx is no longer used as an output and the compare register B is not used to generate waveforms and subsequently no IRQs. In this case the TC channel can only generate a waveform on TIOAx.

When an external event is defined, it can be used as a trigger by setting bit ENETRG in the TC_CMR.

As in Capture mode, the SYNC signal and the software trigger are also available as triggers. RC Compare can also be used as a trigger depending on the parameter WAVSEL.

51.6.14 Synchronization with PWM

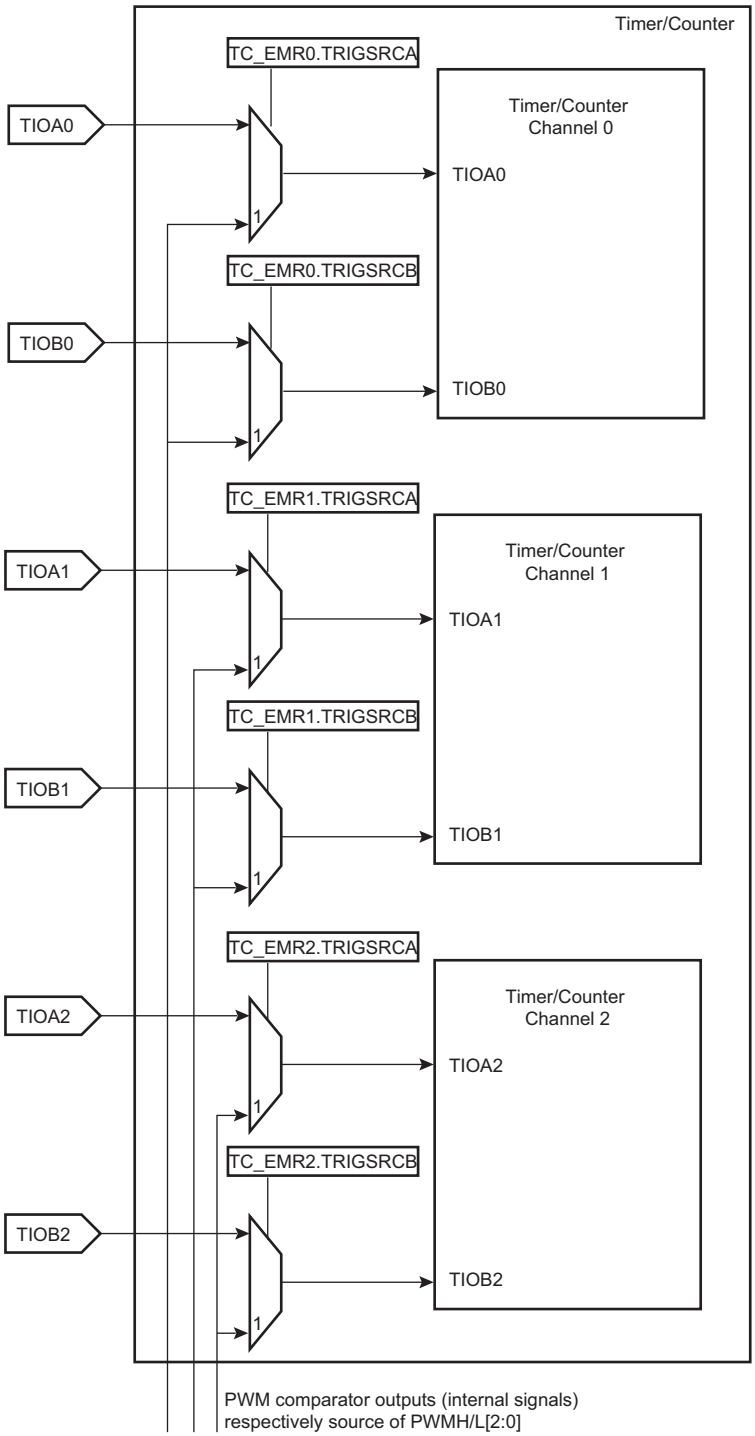
The inputs TIOAx/TIOBx can be bypassed, and thus channel trigger/capture events can be directly driven by the independent PWM module.

PWM comparator outputs (internal signals without dead-time insertion - OCx), respectively source of the PWMH/L[2:0] outputs, are routed to the internal TC inputs. These specific TC inputs are multiplexed with TIOA/B input signal to drive the internal trigger/capture events.

The selection can be programmed in the Extended Mode Register (TC_EMR) fields TRIGSRCA and TRIGSRCB (see Section 51.7.14 “TC Extended Mode Register”).

Each channel of the TC module can be synchronized by a different PWM channel as described in Figure 51-16.

Figure 51-16. Synchronization with PWM



51.6.15 Output Controller

The output controller defines the output level changes on TIOAx and TIOBx following an event. TIOBx Control is used only if TIOBx is defined as output (not as an external event).

The following events control TIOAx and TIOBx:

- Software Trigger
- External Event
- RC Compare

RA Compare controls TIOAx, and RB Compare controls TIOBx. Each of these events can be programmed to set, clear or toggle the output as defined in the corresponding parameter in TC_CMR.

51.6.16 Quadrature Decoder

51.6.16.1 Description

The quadrature decoder (QDEC) is driven by TIOA0, TIOB0, TIOB1 input pins and drives the timer/counter of channel 0 and 1. Channel 2 can be used as a time base in case of speed measurement requirements (refer to Figure 51-17).

When writing a 0 to bit QDEN of the TC_BMR, the QDEC is bypassed and the IO pins are directly routed to the timer counter function.

TIOA0 and TIOB0 are to be driven by the two dedicated quadrature signals from a rotary sensor mounted on the shaft of the off-chip motor.

A third signal from the rotary sensor can be processed through pin TIOB1 and is typically dedicated to be driven by an index signal if it is provided by the sensor. This signal is not required to decode the quadrature signals PHA, PHB.

Field TCCLKS of TC_CMRx must be configured to select XC0 input (i.e., 0x101). Field TC0XC0S has no effect as soon as the QDEC is enabled.

Either speed or position/revolution can be measured. Position channel 0 accumulates the edges of PHA, PHB input signals giving a high accuracy on motor position whereas channel 1 accumulates the index pulses of the sensor, therefore the number of rotations. Concatenation of both values provides a high level of precision on motion system position.

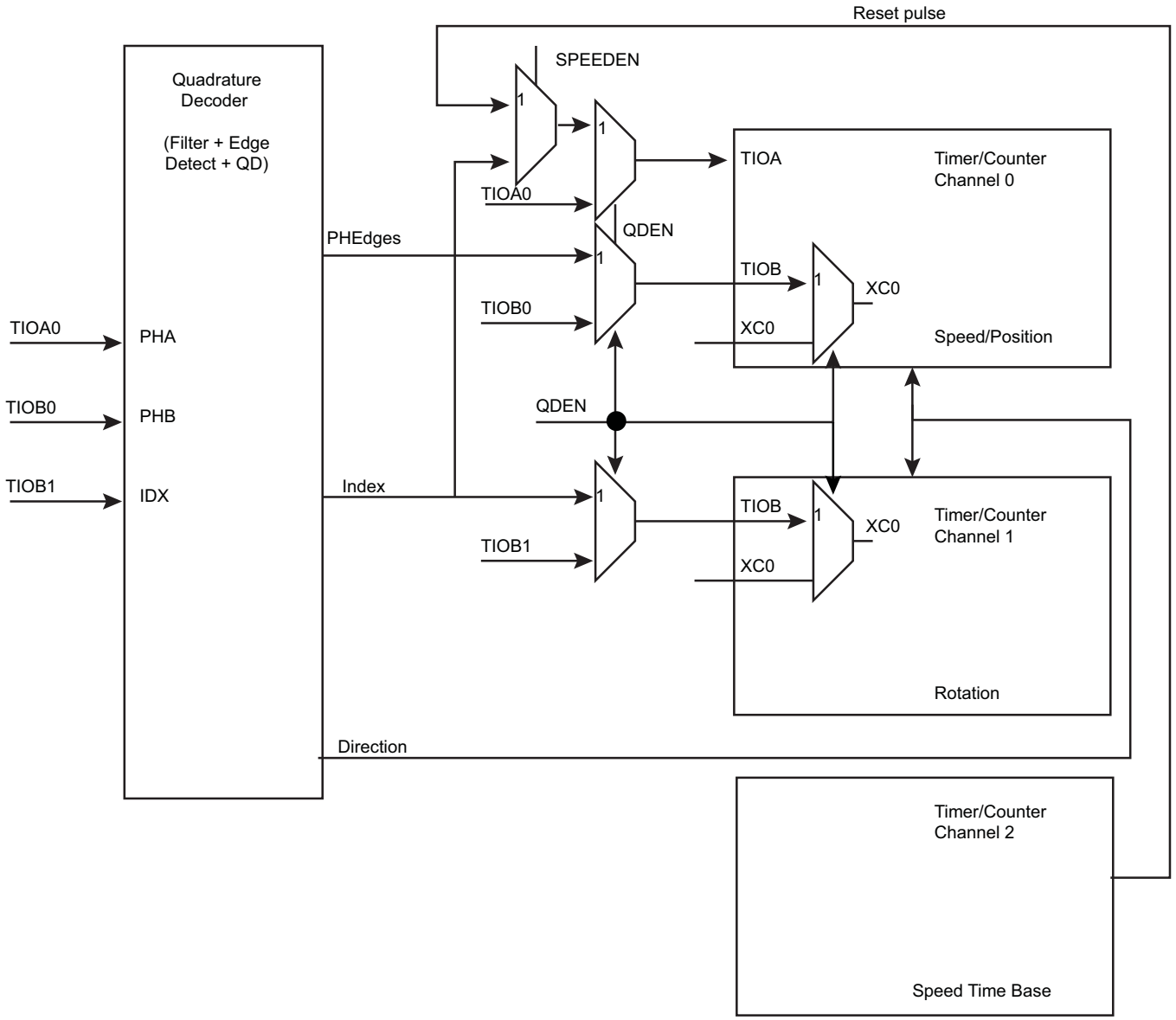
In Speed mode, position cannot be measured but revolution can be measured.

Inputs from the rotary sensor can be filtered prior to down-stream processing. Accommodation of input polarity, phase definition and other factors are configurable.

Interruptions can be generated on different events.

A compare function (using TC_RC) is available on channel 0 (speed/position) or channel 1 (rotation) and can generate an interrupt by means of the CPCS flag in the TC_SRx.

Figure 51-17. Predefined Connection of the Quadrature Decoder with Timer Counters



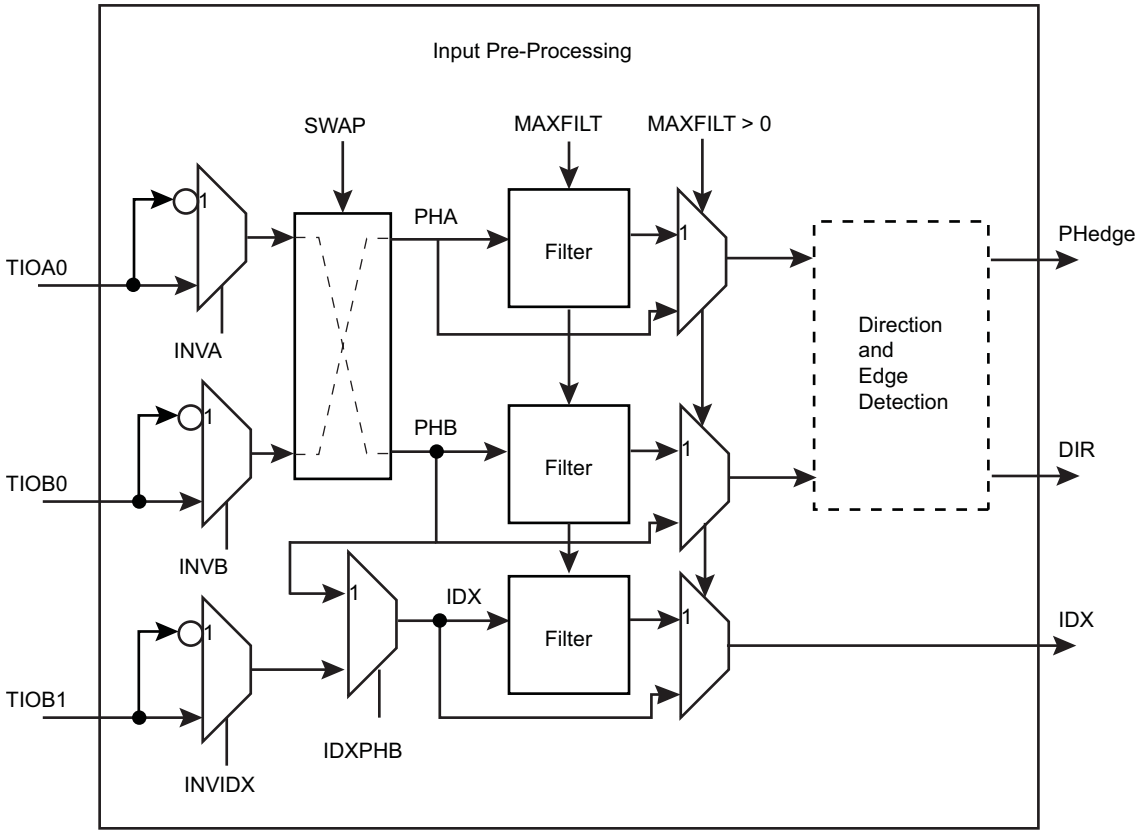
51.6.16.2 Input Preprocessing

Input preprocessing consists of capabilities to take into account rotary sensor factors such as polarities and phase definition followed by configurable digital filtering.

Each input can be negated and swapping PHA, PHB is also configurable.

The MAXFILT field in the TC_BMR is used to configure a minimum duration for which the pulse is stated as valid. When the filter is active, pulses with a duration lower than MAXFILT +1 $\times$ $t_{peripheral\ clock}$ ns are not passed to down-stream logic.

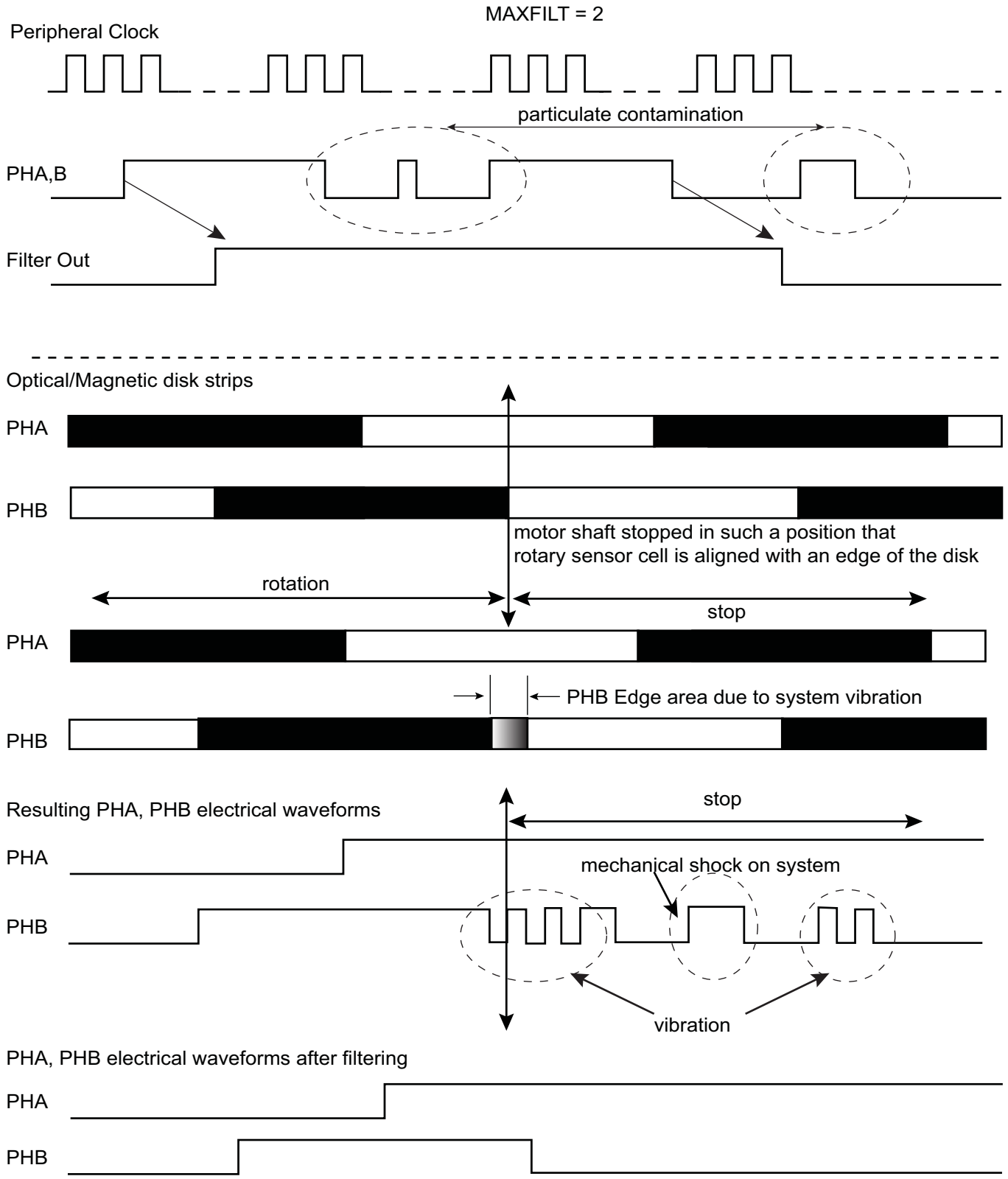
Figure 51-18. Input Stage



Input filtering can efficiently remove spurious pulses that might be generated by the presence of particulate contamination on the optical or magnetic disk of the rotary sensor.

Spurious pulses can also occur in environments with high levels of electro-magnetic interference. Or, simply if vibration occurs even when rotation is fully stopped and the shaft of the motor is in such a position that the beginning of one of the reflective or magnetic bars on the rotary sensor disk is aligned with the light or magnetic (Hall) receiver cell of the rotary sensor. Any vibration can make the PHA, PHB signals toggle for a short duration.

Figure 51-19. Filtering Examples



51.6.16.3 Direction Status and Change Detection

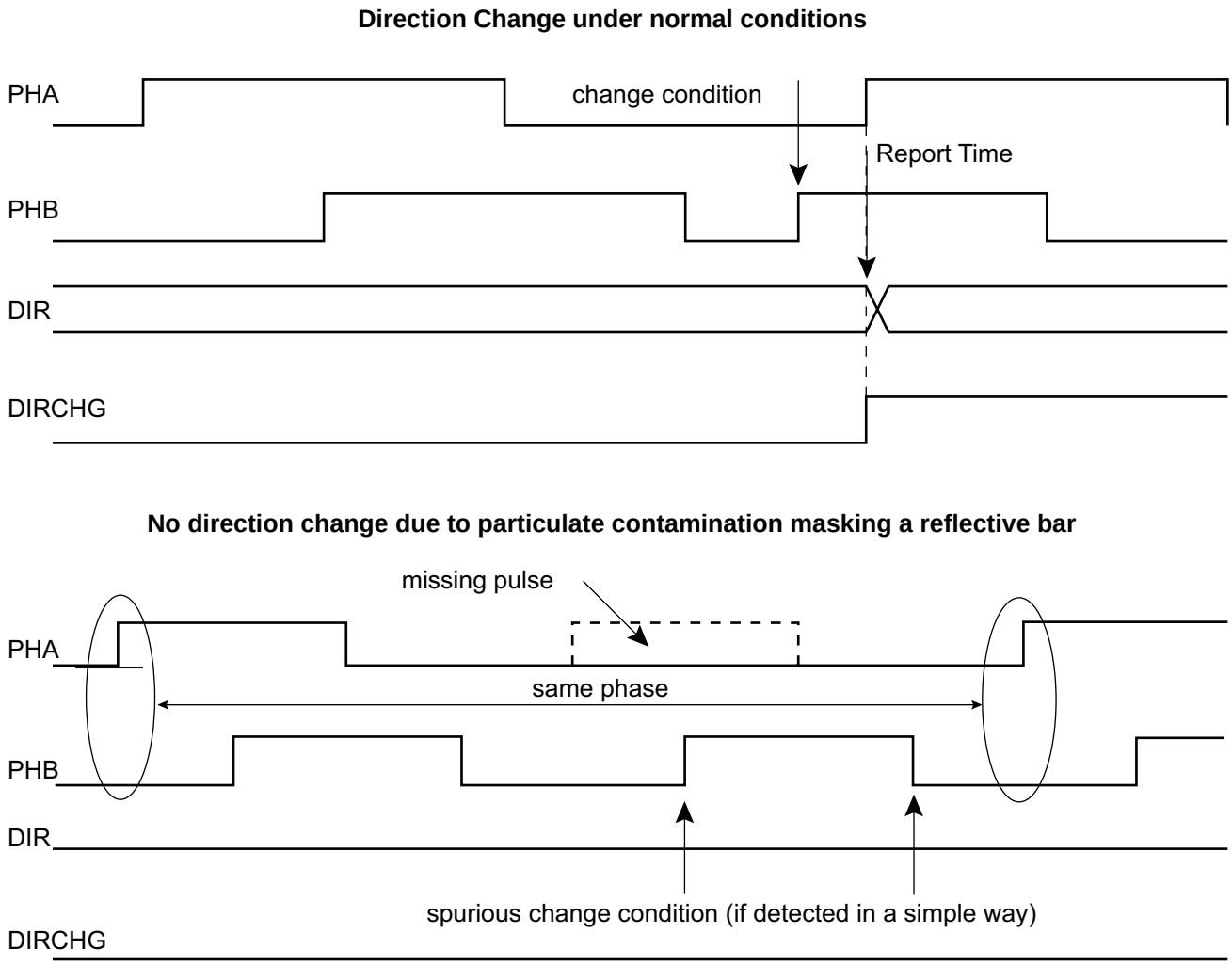
After filtering, the quadrature signals are analyzed to extract the rotation direction and edges of the two quadrature signals detected in order to be counted by timer/counter logic downstream.

The direction status can be directly read at anytime in the TC_QISR. The polarity of the direction flag status depends on the configuration written in TC_BMR. INVA, INVB, INVIDX, SWAP modify the polarity of DIR flag.

Any change in rotation direction is reported in the TC_QISR and can generate an interrupt.

The direction change condition is reported as soon as two consecutive edges on a phase signal have sampled the same value on the other phase signal and there is an edge on the other signal. The two consecutive edges of one phase signal sampling the same value on other phase signal is not sufficient to declare a direction change, for the reason that particulate contamination may mask one or more reflective bars on the optical or magnetic disk of the sensor. Refer to Figure 51-20 for waveforms.

Figure 51-20. Rotation Change Detection

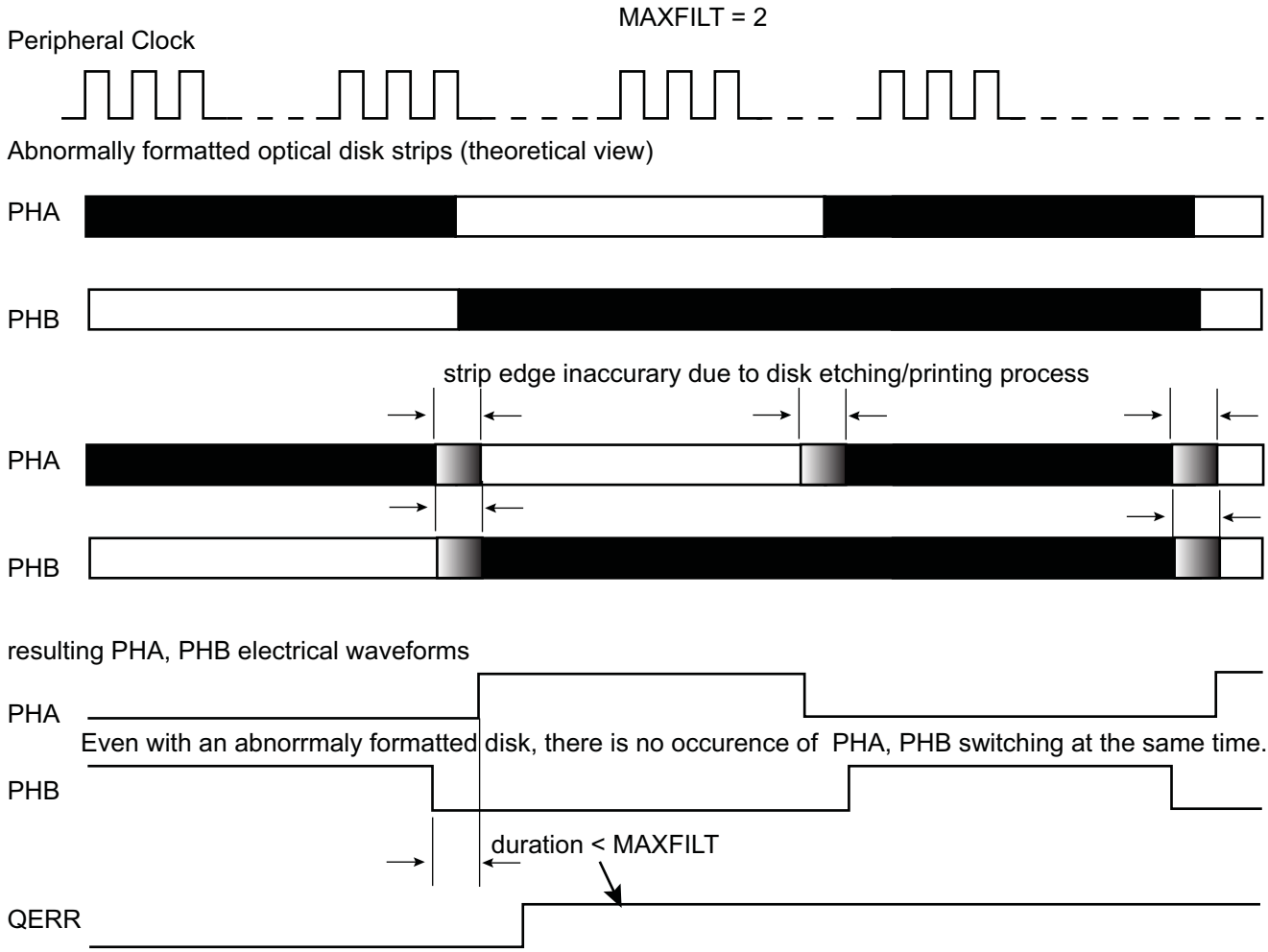


The direction change detection is disabled when QDTRANS is set in the TC_BMR. In this case, the DIR flag report must not be used.

A quadrature error is also reported by the QDEC via the QERR flag in the TC_QISR. This error is reported if the time difference between two edges on PHA, PHB is lower than a predefined value. This predefined value is

configurable and corresponds to (MAXFILT + 1) × $t_{peripheral\ clock}$ ns. After being filtered there is no reason to have two edges closer than (MAXFILT + 1) × $t_{peripheral\ clock}$ ns under normal mode of operation.

Figure 51-21. Quadrature Error Detection



MAXFILT must be tuned according to several factors such as the peripheral clock frequency, type of rotary sensor and rotation speed to be achieved.

51.6.16.4 Position and Rotation Measurement

When the POSEN bit is set in the TC_BMR, the motor axis position is processed on channel 0 (by means of the PHA, PHB edge detections) and the number of motor revolutions are recorded on channel 1 if the IDX signal is provided on the TIOB1 input. If there is no IDX signal available, it is possible to clear the internal counter for each revolution if the number of counts per revolution is configured in TC_RC0.RC and the TC_CMR.CPCTRG bit is written to 1. The position measurement can be read in the TC_CV0 register and the rotation measurement can be read in the TC_CV1 register.

Channel 0 and 1 must be configured in Capture mode (TC_CMR0.WAVE = 0). ‘Rising edge’ must be selected as the External Trigger Edge (TC_CMR.ETRGEDG = 0x01) and ‘TIOAx’ must be selected as the External Trigger (TC_CMR.ABETRG = 0x1).

In parallel, the number of edges are accumulated on timer/counter channel 0 and can be read on the TC_CV0 register.

Therefore, the accurate position can be read on both TC_CV registers and concatenated to form a 32-bit word.

The timer/counter channel 0 is cleared for each increment of IDX count value.

Depending on the quadrature signals, the direction is decoded and allows to count up or down in timer/counter channels 0 and 1. The direction status is reported on TC_QISR.

51.6.16.5 Speed Measurement

When SPEEDEN is set in the TC_BMR, the speed measure is enabled on channel 0.

A time base must be defined on channel 2 by writing the TC_RC2 period register. Channel 2 must be configured in Waveform mode (WAVE bit set) in TC_CMR2. The WAVSEL field must be defined with 0x10 to clear the counter by comparison and matching with TC_RC value. Field ACPC must be defined at 0x11 to toggle TIOAx output.

This time base is automatically fed back to TIOAx of channel 0 when QDEN and SPEEDEN are set.

Channel 0 must be configured in Capture mode (WAVE = 0 in TC_CMR0). The ABETRG bit of TC_CMR0 must be configured at 1 to select TIOAx as a trigger for this channel.

EDGTRG must be set to 0x01, to clear the counter on a rising edge of the TIOAx signal and field LDRA must be set accordingly to 0x01, to load TC_RA0 at the same time as the counter is cleared (LDRB must be set to 0x01). As a consequence, at the end of each time base period the differentiation required for the speed calculation is performed.

The process must be started by configuring bits CLKEN and SWTRG in the TC_CCR.

The speed can be read on field RA in TC_RA0.

Channel 1 can still be used to count the number of revolutions of the motor.

51.6.16.6 Detecting a Missing Index Pulse

To detect a missing index pulse due contamination, dust, etc., the TC_SR0.CPCS flag can be used. It is also possible to assert the interrupt line if the TC_SR0.CPCS flag is enabled as a source of the interrupt by writing a ‘1’ to TC_IER0.CPCS.

The TC_RC0.RC field must be written with the nominal number of counts per revolution provided by the rotary encoder, plus a margin to eliminate potential noise (e.g., if nominal count per revolution is 1024, then TC_RC0.RC = 1028).

If the index pulse is missing, the timer value is not cleared and the nominal value is exceeded, then the comparator on the RC triggers an event, TC_SR0.CPCS = 1, and the interrupt line is asserted if TC_IER0.CPCS = 1.

51.6.16.7 Missing Pulse Detection and Auto-correction

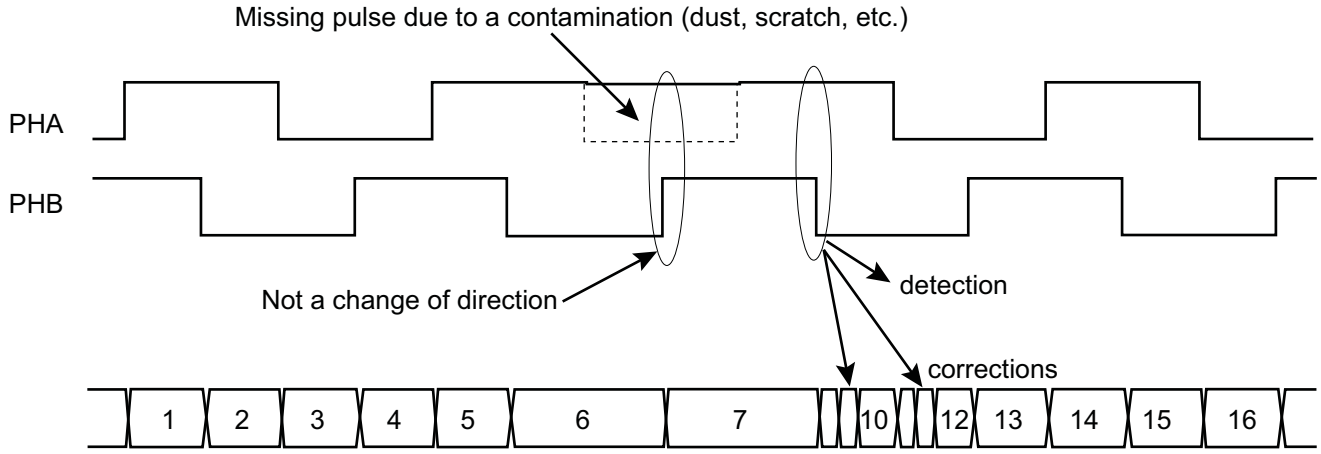
The QDEC is equipped with a circuitry which detects and corrects some errors that may result from contamination on optical disks or other materials producing the quadrature phase signals.

The detection and autocorrection only works if the Count mode is configured for both phases (EDGPHA = 1 in TC_BMR) and is enabled (AUTOC = 1 in TC_BMR).

If a pulse is missing on a phase signal, it is automatically detected and the pulse count reported in the CV field of the TC_CV0/1 is automatically corrected.

There is no detection if both phase signals are affected at the same location on the device providing the quadrature signals because the detection requires a valid phase signal to detect the contamination on the other phase signal.

Figure 51-22. Detection and Auto-correction of Missing Pulses



If a quadrature device is undamaged, the number of pulses counted for a predefined period of time must be the same with or without detection and auto-correction feature.

Therefore, if the measurement results differ, a contamination exists on the device producing the quadrature signals.

This does not substitute the measurements of the number of pulses between two index pulses (if available) but provides a complementary method to detect damaged quadrature devices.

When the device providing quadrature signals is severely damaged, potentially leading to a number of consecutive missing pulses greater than 1, the downstream processing may be affected. It is possible to define the maximum admissible number of consecutive missing pulses before issuing a Missing Pulse Error flag (MPE in TC_QISR). The threshold triggering a MPE flag report can be configured in field MAXCMP of the TC_BMR. If the field MAXCMP is cleared, MPE never rises. The flag MAXCMP can trigger an interrupt while the QDEC is operating, thus providing a real time report of a potential problem on the quadrature device.

51.6.17 2-bit Gray Up/Down Counter for Stepper Motor

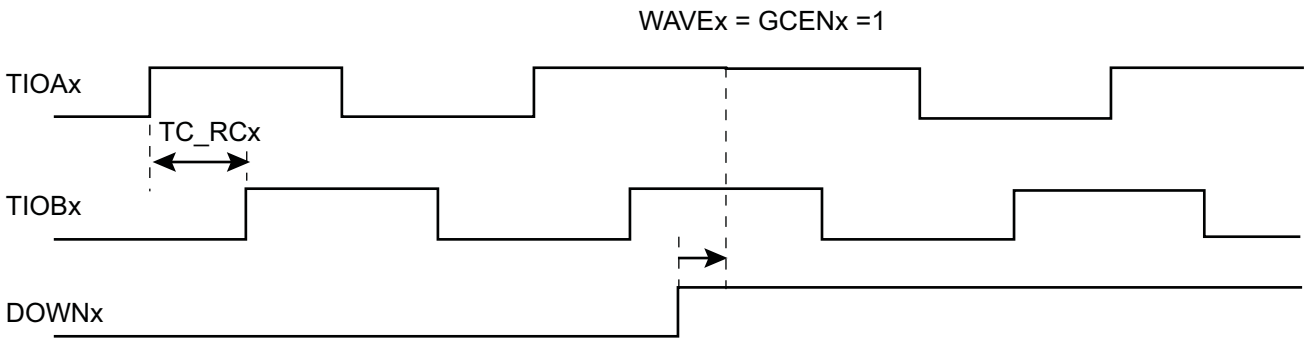
Each channel can be independently configured to generate a 2-bit Gray count waveform on corresponding TIOAx, TIOBx outputs by means of the GCEN bit in TC_SMMRx.

Up or Down count can be defined by writing bit DOWN in TC_SMMRx.

It is mandatory to configure the channel in Waveform mode in the TC_CMR.

The period of the counters can be programmed in TC_RCx.

Figure 51-23. 2-bit Gray Up/Down Counter



51.6.18 Fault Mode

At any time, the TC_RCx registers can be used to perform a comparison on the respective current channel counter value (TC_CVx) with the value of TC_RCx register.

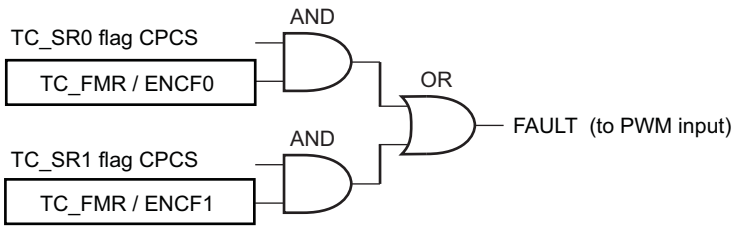
The CPCSx flags can be set accordingly and an interrupt can be generated.

This interrupt is processed but requires an unpredictable amount of time to be achieve the required action.

It is possible to trigger the FAULT output of the TIMER1 with CPCS from TC_SR0 and/or CPCS from TC_SR1. Each source can be independently enabled/disabled in the TC_FMR.

This can be useful to detect an overflow on speed and/or position when QDEC is processed and to act immediately by using the FAULT output.

Figure 51-24. Fault Output Generation



51.6.19 Register Write Protection

To prevent any single software error from corrupting TC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the TC Write Protection Mode Register (TC_WPMR).

The Timer Counter clock of the first channel must be enabled to access TC_WPMR.

The following registers can be write-protected:

- TC Block Mode Register
- TC Channel Mode Register: Capture Mode
- TC Channel Mode Register: Waveform Mode
- TC Fault Mode Register
- TC Stepper Motor Mode Register
- TC Register A
- TC Register B
- TC Register C
- TC Extended Mode Register

51.7 Timer Counter (TC) User Interface

Table 51-6. Register Mapping

Offset[1]	Register	Name	Access	Reset
0x00 + channel * 0x40 + 0x00	Channel Control Register	TC_CCR	Write-only	–
0x00 + channel * 0x40 + 0x04	Channel Mode Register	TC_CMR	Read/Write	0
0x00 + channel * 0x40 + 0x08	Stepper Motor Mode Register	TC_SMMR	Read/Write	0
0x00 + channel * 0x40 + 0x0C	Register AB	TC_RAB	Read-only	0
0x00 + channel * 0x40 + 0x10	Counter Value	TC_CV	Read-only	0
0x00 + channel * 0x40 + 0x14	Register A	TC_RA	Read/Write[2]	0
0x00 + channel * 0x40 + 0x18	Register B	TC_RB	Read/Write[2]	0
0x00 + channel * 0x40 + 0x1C	Register C	TC_RC	Read/Write	0
0x00 + channel * 0x40 + 0x20	Status Register	TC_SR	Read-only	0
0x00 + channel * 0x40 + 0x24	Interrupt Enable Register	TC_IER	Write-only	–
0x00 + channel * 0x40 + 0x28	Interrupt Disable Register	TC_IDR	Write-only	–
0x00 + channel * 0x40 + 0x2C	Interrupt Mask Register	TC_IMR	Read-only	0
0x00 + channel * 0x40 + 0x30	Extended Mode Register	TC_EMR	Read/Write	0
0xC0	Block Control Register	TC_BCR	Write-only	–
0xC4	Block Mode Register	TC_BMR	Read/Write	0
0xC8	QDEC Interrupt Enable Register	TC_QIER	Write-only	–
0xCC	QDEC Interrupt Disable Register	TC_QIDR	Write-only	–
0xD0	QDEC Interrupt Mask Register	TC_QIMR	Read-only	0
0xD4	QDEC Interrupt Status Register	TC_QISR	Read-only	0
0xD8	Fault Mode Register	TC_FMR	Read/Write	0
0xE4	Write Protection Mode Register	TC_WPMR	Read/Write	0
0xE8–0xFC	Reserved	–	–	–

Notes: 1. Channel index ranges from 0 to 2.
2. Read-only if TC_CMRx.WAVE = 0

51.7.1 TC Channel Control Register

Name: TC_CCRx [x=0..2]

Address: 0xF800C000 (0)[0], 0xF800C040 (0)[1], 0xF800C080 (0)[2], 0xF8010000 (1)[0], 0xF8010040 (1)[1], 0xF8010080 (1)[2]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	SWTRG	CLKDIS	CLKEN

- **CLKEN: Counter Clock Enable Command**

0: No effect.

1: Enables the clock if CLKDIS is not 1.

- **CLKDIS: Counter Clock Disable Command**

0: No effect.

1: Disables the clock.

- **SWTRG: Software Trigger Command**

0: No effect.

1: A software trigger is performed: the counter is reset and the clock is started.

51.7.2 TC Channel Mode Register: Capture Mode

Name: TC_CMRx [x=0..2] (CAPTURE_MODE)

Address: 0xF800C004 (0)[0], 0xF800C044 (0)[1], 0xF800C084 (0)[2], 0xF8010004 (1)[0], 0xF8010044 (1)[1], 0xF8010084 (1)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	SBSMPLR			LDRB		LDRA	
15	14	13	12	11	10	9	8
WAVE	CPCTRG	–	–	–	ABETRG	ETRGEDG	
7	6	5	4	3	2	1	0
LDBDIS	LDBSTOP	BURST		CLKI	TCCLKS		

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **TCCLKS: Clock Selection**

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal GCLK [35], GCLK [36] clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal System bus clock divided by 8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal System bus clock divided by 32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal System bus clock divided by 128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal slow_clock clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

To operate at maximum peripheral clock frequency, refer to Section 51.7.14 “TC Extended Mode Register”.

- **CLKI: Clock Invert**

0: Counter is incremented on rising edge of the clock.

1: Counter is incremented on falling edge of the clock.

- **BURST: Burst Signal Selection**

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	XC0	XC0 is ANDed with the selected clock.
2	XC1	XC1 is ANDed with the selected clock.
3	XC2	XC2 is ANDed with the selected clock.

- **LDBSTOP: Counter Clock Stopped with RB Loading**

0: Counter clock is not stopped when RB loading occurs.

1: Counter clock is stopped when RB loading occurs.

- **LDBDIS: Counter Clock Disable with RB Loading**

0: Counter clock is not disabled when RB loading occurs.

1: Counter clock is disabled when RB loading occurs.

- **ETRGEDG: External Trigger Edge Selection**

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	RISING	Rising edge
2	FALLING	Falling edge
3	EDGE	Each edge

- **ABETRG: TIOAx or TIOBx External Trigger Selection**

0: TIOBx is used as an external trigger.

1: TIOAx is used as an external trigger.

- **CPCTRG: RC Compare Trigger Enable**

0: RC Compare has no effect on the counter and its clock.

1: RC Compare resets the counter and starts the counter clock.

- **WAVE: Waveform Mode**

0: Capture mode is enabled.

1: Capture mode is disabled (Waveform mode is enabled).

- **LDRA: RA Loading Edge Selection**

Value	Name	Description
0	NONE	None
1	RISING	Rising edge of TIOAx
2	FALLING	Falling edge of TIOAx
3	EDGE	Each edge of TIOAx

- **LDRB: RB Loading Edge Selection**

Value	Name	Description
0	NONE	None
1	RISING	Rising edge of TIOAx
2	FALLING	Falling edge of TIOAx
3	EDGE	Each edge of TIOAx

- **SBSMPLR: Loading Edge Subsampling Ratio**

Value	Name	Description
0	ONE	Load a Capture Register each selected edge
1	HALF	Load a Capture Register every 2 selected edges
2	FOURTH	Load a Capture Register every 4 selected edges
3	EIGHTH	Load a Capture Register every 8 selected edges
4	SIXTEENTH	Load a Capture Register every 16 selected edges

51.7.3 TC Channel Mode Register: Waveform Mode

Name: TC_CMRx [x=0..2] (WAVEFORM_MODE)

Address: 0xF800C004 (0)[0], 0xF800C044 (0)[1], 0xF800C084 (0)[2], 0xF8010004 (1)[0], 0xF8010044 (1)[1], 0xF8010084 (1)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
BSWTRG		BEEVT		BCPC		BCPB	
23	22	21	20	19	18	17	16
ASWTRG		AEEVT		ACPC		ACPA	
15	14	13	12	11	10	9	8
WAVE	WAVSEL		ENETRG	EEVT		EEVTEDG	
7	6	5	4	3	2	1	0
CPCDIS	CPCSTOP	BURST		CLKI	TCCLKS		

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **TCCLKS: Clock Selection**

Value	Name	Description
0	TIMER_CLOCK1	Clock selected: internal GCLK [35], GCLK [36] clock signal (from PMC)
1	TIMER_CLOCK2	Clock selected: internal System bus clock divided by 8 clock signal (from PMC)
2	TIMER_CLOCK3	Clock selected: internal System bus clock divided by 32 clock signal (from PMC)
3	TIMER_CLOCK4	Clock selected: internal System bus clock divided by 128 clock signal (from PMC)
4	TIMER_CLOCK5	Clock selected: internal slow_clock clock signal (from PMC)
5	XC0	Clock selected: XC0
6	XC1	Clock selected: XC1
7	XC2	Clock selected: XC2

To operate at maximum peripheral clock frequency, refer to Section 51.7.14 “TC Extended Mode Register”.

- **CLKI: Clock Invert**

0: Counter is incremented on rising edge of the clock.

1: Counter is incremented on falling edge of the clock.

- **BURST: Burst Signal Selection**

Value	Name	Description
0	NONE	The clock is not gated by an external signal.
1	XC0	XC0 is ANDed with the selected clock.
2	XC1	XC1 is ANDed with the selected clock.
3	XC2	XC2 is ANDed with the selected clock.

- **CPCSTOP: Counter Clock Stopped with RC Compare**

0: Counter clock is not stopped when counter reaches RC.

1: Counter clock is stopped when counter reaches RC.

- **CPCDIS: Counter Clock Disable with RC Compare**

0: Counter clock is not disabled when counter reaches RC.

1: Counter clock is disabled when counter reaches RC.

- **EEVTEDG: External Event Edge Selection**

Value	Name	Description
0	NONE	None
1	RISING	Rising edge
2	FALLING	Falling edge
3	EDGE	Each edge

- **EEVT: External Event Selection**

Signal selected as external event.

Value	Name	Description	TIOB Direction
0	TIOB	TIOB(1)	Input
1	XC0	XC0	Output
2	XC1	XC1	Output
3	XC2	XC2	Output

Note: 1. If TIOB is chosen as the external event signal, it is configured as an input and no longer generates waveforms and subsequently no IRQs.

- **ENETRG: External Event Trigger Enable**

0: The external event has no effect on the counter and its clock.

1: The external event resets the counter and starts the counter clock.

Note: Whatever the value programmed in ENETRG, the selected external event only controls the TIOAx output and TIOBx if not used as input (trigger event input or other input used).

- **WAVSEL: Waveform Selection**

Value	Name	Description
0	UP	UP mode without automatic trigger on RC Compare
1	UPDOWN	UPDOWN mode without automatic trigger on RC Compare
2	UP_RC	UP mode with automatic trigger on RC Compare
3	UPDOWN_RC	UPDOWN mode with automatic trigger on RC Compare

- **WAVE: Waveform Mode**

0: Waveform mode is disabled (Capture mode is enabled).

1: Waveform mode is enabled.

- **ACPA: RA Compare Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **ACPC: RC Compare Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **AEEVT: External Event Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **ASWTRG: Software Trigger Effect on TIOAx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BCPB: RB Compare Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BCPC: RC Compare Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BEEVT: External Event Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

- **BSWTRG: Software Trigger Effect on TIOBx**

Value	Name	Description
0	NONE	None
1	SET	Set
2	CLEAR	Clear
3	TOGGLE	Toggle

51.7.4 TC Stepper Motor Mode Register

Name: TC_SMMRx [x=0..2]

Address: 0xF800C008 (0)[0], 0xF800C048 (0)[1], 0xF800C088 (0)[2], 0xF8010008 (1)[0], 0xF8010048 (1)[1], 0xF8010088 (1)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	DOWN	GCEN

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **GCEN: Gray Count Enable**

0: TIOAx [x=0..2] and TIOBx [x=0..2] are driven by internal counter of channel x.

1: TIOAx [x=0..2] and TIOBx [x=0..2] are driven by a 2-bit Gray counter.

- **DOWN: Down Count**

0: Up counter.

1: Down counter.

51.7.5 TC Register AB

Name: TC_RABx [x=0..2]

Address: 0xF800C00C (0)[0], 0xF800C04C (0)[1], 0xF800C08C (0)[2], 0xF801000C (1)[0], 0xF801004C (1)[1], 0xF801008C (1)[2]

Access: Read-only

31	30	29	28	27	26	25	24
RAB							

23	22	21	20	19	18	17	16
RAB							

15	14	13	12	11	10	9	8
RAB							

7	6	5	4	3	2	1	0
RAB							

- **RAB: Register A or Register B**

RAB contains the next unread capture Register A or Register B value in real time. It is usually read by the DMA after a request due to a valid load edge on TIOAx.

When DMA is used, the RAB register address must be configured as source address of the transfer.

51.7.6 TC Counter Value Register

Name: TC_CVx [x=0..2]

Address: 0xF800C010 (0)[0], 0xF800C050 (0)[1], 0xF800C090 (0)[2], 0xF8010010 (1)[0], 0xF8010050 (1)[1], 0xF8010090 (1)[2]

Access: Read-only

31	30	29	28	27	26	25	24
CV							

23	22	21	20	19	18	17	16
CV							

15	14	13	12	11	10	9	8
CV							

7	6	5	4	3	2	1	0
CV							

- **CV: Counter Value**

CV contains the counter value in real time.

51.7.7 TC Register A

Name: TC_RAx [x=0..2]

Address: 0xF800C014 (0)[0], 0xF800C054 (0)[1], 0xF800C094 (0)[2], 0xF8010014 (1)[0], 0xF8010054 (1)[1], 0xF8010094 (1)[2]

Access: Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1

31	30	29	28	27	26	25	24
RA							

23	22	21	20	19	18	17	16
RA							

15	14	13	12	11	10	9	8
RA							

7	6	5	4	3	2	1	0
RA							

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **RA: Register A**

RA contains the Register A value in real time.

51.7.8 TC Register B

Name: TC_RBx [x=0..2]

Address: 0xF800C018 (0)[0], 0xF800C058 (0)[1], 0xF800C098 (0)[2], 0xF8010018 (1)[0], 0xF8010058 (1)[1], 0xF8010098 (1)[2]

Access: Read-only if TC_CMRx.WAVE = 0, Read/Write if TC_CMRx.WAVE = 1

31	30	29	28	27	26	25	24
RB							
23	22	21	20	19	18	17	16
RB							
15	14	13	12	11	10	9	8
RB							
7	6	5	4	3	2	1	0
RB							

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **RB: Register B**

RB contains the Register B value in real time.

51.7.9 TC Register C

Name: TC_RCx [x=0..2]

Address: 0xF800C01C (0)[0], 0xF800C05C (0)[1], 0xF800C09C (0)[2], 0xF801001C (1)[0], 0xF801005C (1)[1], 0xF801009C (1)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
RC							

23	22	21	20	19	18	17	16
RC							

15	14	13	12	11	10	9	8
RC							

7	6	5	4	3	2	1	0
RC							

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **RC: Register C**

RC contains the Register C value in real time.

51.7.10 TC Status Register

Name: TC_SRx [x=0..2]

Address: 0xF800C020 (0)[0], 0xF800C060 (0)[1], 0xF800C0A0 (0)[2], 0xF8010020 (1)[0], 0xF8010060 (1)[1], 0xF80100A0 (1)[2]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	MTIOB	MTIOA	CLKSTA
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow Status (cleared on read)**

0: No counter overflow has occurred since the last read of the Status Register.

1: A counter overflow has occurred since the last read of the Status Register.

- **LOVRS: Load Overrun Status (cleared on read)**

0: Load overrun has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.

1: RA or RB have been loaded at least twice without any read of the corresponding register since the last read of the Status Register, if TC_CMRx.WAVE = 0.

- **CPAS: RA Compare Status (cleared on read)**

0: RA Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0.

1: RA Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1.

- **CPBS: RB Compare Status (cleared on read)**

0: RB Compare has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 0.

1: RB Compare has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 1.

- **CPCS: RC Compare Status (cleared on read)**

0: RC Compare has not occurred since the last read of the Status Register.

1: RC Compare has occurred since the last read of the Status Register.

- **LDRAS: RA Loading Status (cleared on read)**

0: RA Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.

1: RA Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0.

- **LDRBS: RB Loading Status (cleared on read)**

0: RB Load has not occurred since the last read of the Status Register or TC_CMRx.WAVE = 1.

1: RB Load has occurred since the last read of the Status Register, if TC_CMRx.WAVE = 0.

- **ETRGS: External Trigger Status (cleared on read)**

0: External trigger has not occurred since the last read of the Status Register.

1: External trigger has occurred since the last read of the Status Register.

- **CLKSTA: Clock Enabling Status**

0: Clock is disabled.

1: Clock is enabled.

- **MTIOA: TIOAx Mirror**

0: TIOAx is low. If TC_CMRx.WAVE = 0, this means that TIOAx pin is low. If TC_CMRx.WAVE = 1, this means that TIOAx is driven low.

1: TIOAx is high. If TC_CMRx.WAVE = 0, this means that TIOAx pin is high. If TC_CMRx.WAVE = 1, this means that TIOAx is driven high.

- **MTIOB: TIOBx Mirror**

0: TIOBx is low. If TC_CMRx.WAVE = 0, this means that TIOBx pin is low. If TC_CMRx.WAVE = 1, this means that TIOBx is driven low.

1: TIOBx is high. If TC_CMRx.WAVE = 0, this means that TIOBx pin is high. If TC_CMRx.WAVE = 1, this means that TIOBx is driven high.

51.7.11 TC Interrupt Enable Register

Name: TC_IERx [x=0..2]

Address: 0xF800C024 (0)[0], 0xF800C064 (0)[1], 0xF800C0A4 (0)[2], 0xF8010024 (1)[0], 0xF8010064 (1)[1], 0xF80100A4 (1)[2]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: No effect.

1: Enables the Counter Overflow Interrupt.

- **LOVRS: Load Overrun**

0: No effect.

1: Enables the Load Overrun Interrupt.

- **CPAS: RA Compare**

0: No effect.

1: Enables the RA Compare Interrupt.

- **CPBS: RB Compare**

0: No effect.

1: Enables the RB Compare Interrupt.

- **CPCS: RC Compare**

0: No effect.

1: Enables the RC Compare Interrupt.

- **LDRAS: RA Loading**

0: No effect.

1: Enables the RA Load Interrupt.

- **LDRBS: RB Loading**

0: No effect.

1: Enables the RB Load Interrupt.

- **ETRGS: External Trigger**

0: No effect.

1: Enables the External Trigger Interrupt.

51.7.12 TC Interrupt Disable Register

Name: TC_IDRx [x=0..2]

Address: 0xF800C028 (0)[0], 0xF800C068 (0)[1], 0xF800C0A8 (0)[2], 0xF8010028 (1)[0], 0xF8010068 (1)[1], 0xF80100A8 (1)[2]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: No effect.

1: Disables the Counter Overflow Interrupt.

- **LOVRS: Load Overrun**

0: No effect.

1: Disables the Load Overrun Interrupt (if TC_CMRx.WAVE = 0).

- **CPAS: RA Compare**

0: No effect.

1: Disables the RA Compare Interrupt (if TC_CMRx.WAVE = 1).

- **CPBS: RB Compare**

0: No effect.

1: Disables the RB Compare Interrupt (if TC_CMRx.WAVE = 1).

- **CPCS: RC Compare**

0: No effect.

1: Disables the RC Compare Interrupt.

- **LDRAS: RA Loading**

0: No effect.

1: Disables the RA Load Interrupt (if TC_CMRx.WAVE = 0).

- **LDRBS: RB Loading**

0: No effect.

1: Disables the RB Load Interrupt (if TC_CMRx.WAVE = 0).

- **ETRGS: External Trigger**

0: No effect.

1: Disables the External Trigger Interrupt.

51.7.13 TC Interrupt Mask Register

Name: TC_IMRx [x=0..2]

Address: 0xF800C02C (0)[0], 0xF800C06C (0)[1], 0xF800C0AC (0)[2], 0xF801002C (1)[0], 0xF801006C (1)[1], 0xF80100AC (1)[2]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
ETRGS	LDRBS	LDRAS	CPCS	CPBS	CPAS	LOVRS	COVFS

- **COVFS: Counter Overflow**

0: The Counter Overflow Interrupt is disabled.

1: The Counter Overflow Interrupt is enabled.

- **LOVRS: Load Overrun**

0: The Load Overrun Interrupt is disabled.

1: The Load Overrun Interrupt is enabled.

- **CPAS: RA Compare**

0: The RA Compare Interrupt is disabled.

1: The RA Compare Interrupt is enabled.

- **CPBS: RB Compare**

0: The RB Compare Interrupt is disabled.

1: The RB Compare Interrupt is enabled.

- **CPCS: RC Compare**

0: The RC Compare Interrupt is disabled.

1: The RC Compare Interrupt is enabled.

- **LDRAS: RA Loading**

0: The Load RA Interrupt is disabled.

1: The Load RA Interrupt is enabled.

- **LDRBS: RB Loading**

0: The Load RB Interrupt is disabled.

1: The Load RB Interrupt is enabled.

- **ETRGS: External Trigger**

0: The External Trigger Interrupt is disabled.

1: The External Trigger Interrupt is enabled.

51.7.14 TC Extended Mode Register

Name: TC_EMRx [x=0..2]

Address: 0xF800C030 (0)[0], 0xF800C070 (0)[1], 0xF800C0B0 (0)[2], 0xF8010030 (1)[0], 0xF8010070 (1)[1], 0xF80100B0 (1)[2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	NODIVCLK
7	6	5	4	3	2	1	0
–	–	TRIGSRCB		–	–	TRIGSRCA	

- **TRIGSRCA: Trigger Source for Input A**

Value	Name	Description
0	EXTERNAL_TIOAx	The trigger/capture input A is driven by external pin TIOAx
1	PWMx	The trigger/capture input A is driven internally by PWMx

- **TRIGSRCB: Trigger Source for Input B**

Value	Name	Description
0	EXTERNAL_TIOBx	The trigger/capture input B is driven by external pin TIOBx
1	PWMx	For TC0 to TC10: The trigger/capture input B is driven internally by the comparator output (see Figure 51-16) of the PWMx. For TC11: The trigger/capture input B is driven internally by the GTSUCOMP signal of the Ethernet MAC (GMAC).

- **NODIVCLK: No Divided Clock**

0: The selected clock is defined by field TCCLKS in TC_CMRx.

1: The selected clock is peripheral clock and TCCLKS field (TC_CMRx) has no effect.

51.7.15 TC Block Control Register

Name: TC_BCR

Address: 0xF800C0C0 (0), 0xF80100C0 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	SYNC

- **SYNC: Synchro Command**

0: No effect.

1: Asserts the SYNC signal which generates a software trigger simultaneously for each of the channels.

51.7.16 TC Block Mode Register

Name: TC_BMR

Address: 0xF800C0C4 (0), 0xF80100C4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	MAXCMP				MAXFILT	

23	22	21	20	19	18	17	16
MAXFILT				–	AUTOC	IDXPHB	SWAP

15	14	13	12	11	10	9	8
INVIDX	INVB	INVA	EDGPHA	QDTRANS	SPEEDEN	POSEN	QDEN

7	6	5	4	3	2	1	0
–	–	TC2XC2S		TC1XC1S		TC0XC0S	

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **TC0XC0S: External Clock Signal 0 Selection**

Value	Name	Description
0	TCLK0	Signal connected to XC0: TCLK0
1	–	Reserved
2	TIOA1	Signal connected to XC0: TIOA1
3	TIOA2	Signal connected to XC0: TIOA2

- **TC1XC1S: External Clock Signal 1 Selection**

Value	Name	Description
0	TCLK1	Signal connected to XC1: TCLK1
1	–	Reserved
2	TIOA0	Signal connected to XC1: TIOA0
3	TIOA2	Signal connected to XC1: TIOA2

- **TC2XC2S: External Clock Signal 2 Selection**

Value	Name	Description
0	TCLK2	Signal connected to XC2: TCLK2
1	–	Reserved
2	TIOA0	Signal connected to XC2: TIOA0
3	TIOA1	Signal connected to XC2: TIOA1

- **QDEN: Quadrature Decoder Enabled**

0: Disabled.

1: Enables the QDEC (filter, edge detection and quadrature decoding).

Quadrature decoding (direction change) can be disabled using QDTRANS bit.

One of the POSEN or SPEEDEN bits must be also enabled.

- **POSEN: Position Enabled**

0: Disable position.

1: Enables the position measure on channel 0 and 1.

- **SPEEDEN: Speed Enabled**

0: Disabled.

1: Enables the speed measure on channel 0, the time base being provided by channel 2.

- **QDTRANS: Quadrature Decoding Transparent**

0: Full quadrature decoding logic is active (direction change detected).

1: Quadrature decoding logic is inactive (direction change inactive) but input filtering and edge detection are performed.

- **EDGPHA: Edge on PHA Count Mode**

0: Edges are detected on PHA only.

1: Edges are detected on both PHA and PHB.

- **INVA: Inverted PHA**

0: PHA (TIOA0) is directly driving the QDEC.

1: PHA is inverted before driving the QDEC.

- **INVB: Inverted PHB**

0: PHB (TIOB0) is directly driving the QDEC.

1: PHB is inverted before driving the QDEC.

- **INVIDX: Inverted Index**

0: IDX (TIOA1) is directly driving the QDEC.

1: IDX is inverted before driving the QDEC.

- **SWAP: Swap PHA and PHB**

0: No swap between PHA and PHB.

1: Swap PHA and PHB internally, prior to driving the QDEC.

- **IDXPHB: Index Pin is PHB Pin**

0: IDX pin of the rotary sensor must drive TIOA1.

1: IDX pin of the rotary sensor must drive TIOB0.

- **AUTOC: Auto-Correction of missing pulses**

0 (DISABLED): The detection and auto-correction function is disabled.

1 (ENABLED): The detection and auto-correction function is enabled.

- **MAXFILT: Maximum Filter**

1–63: Defines the filtering capabilities.

Pulses with a period shorter than MAXFILT+1 peripheral clock cycles are discarded.

- **MAXCMP: Maximum Consecutive Missing Pulses**

0: The flag MPE in TC_QISR never rises.

1–15: Defines the number of consecutive missing pulses before a flag report.

51.7.17 TC QDEC Interrupt Enable Register

Name: TC_QIER

Address: 0xF800C0C8 (0), 0xF80100C8 (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: No effect.

1: Enables the interrupt when a rising edge occurs on IDX input.

- **DIRCHG: Direction Change**

0: No effect.

1: Enables the interrupt when a change on rotation direction is detected.

- **QERR: Quadrature Error**

0: No effect.

1: Enables the interrupt when a quadrature error occurs on PHA, PHB.

51.7.18 TC QDEC Interrupt Disable Register

Name: TC_QIDR

Address: 0xF800C0CC (0), 0xF80100CC (1)

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: No effect.

1: Disables the interrupt when a rising edge occurs on IDX input.

- **DIRCHG: Direction Change**

0: No effect.

1: Disables the interrupt when a change on rotation direction is detected.

- **QERR: Quadrature Error**

0: No effect.

1: Disables the interrupt when a quadrature error occurs on PHA, PHB.

51.7.19 TC QDEC Interrupt Mask Register

Name: TC_QIMR

Address: 0xF800C0D0 (0), 0xF80100D0 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	QERR	DIRCHG	IDX

- **IDX: Index**

0: The interrupt on IDX input is disabled.

1: The interrupt on IDX input is enabled.

- **DIRCHG: Direction Change**

0: The interrupt on rotation direction change is disabled.

1: The interrupt on rotation direction change is enabled.

- **QERR: Quadrature Error**

0: The interrupt on quadrature error is disabled.

1: The interrupt on quadrature error is enabled.

51.7.20 TC QDEC Interrupt Status Register

Name: TC_QISR

Address: 0xF800C0D4 (0), 0xF80100D4 (1)

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	DIR
7	6	5	4	3	2	1	0
–	–	–	–	MPE	QERR	DIRCHG	IDX

- **IDX: Index**

0: No Index input change since the last read of TC_QISR.

1: The IDX input has changed since the last read of TC_QISR.

- **DIRCHG: Direction Change**

0: No change on rotation direction since the last read of TC_QISR.

1: The rotation direction changed since the last read of TC_QISR.

- **QERR: Quadrature Error**

0: No quadrature error since the last read of TC_QISR.

1: A quadrature error occurred since the last read of TC_QISR.

- **MPE: Consecutive Missing Pulse Error**

0: The number of consecutive missing pulses has not reached the maximum value specified in MAXMP since the last read of TC_QISR.

1: An occurrence of MAXCMP consecutive missing pulses has been detected since the last read of TC_QISR.

- **DIR: Direction**

Returns an image of the actual rotation direction.

51.7.21 TC Fault Mode Register

Name: TC_FMR

Address: 0xF800C0D8 (0), 0xF80100D8 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	ENCF1	ENCF0

This register can only be written if the WPEN bit is cleared in the TC Write Protection Mode Register.

- **ENCF0: Enable Compare Fault Channel 0**

0: Disables the FAULT output source (CPCS flag) from channel 0.

1: Enables the FAULT output source (CPCS flag) from channel 0.

- **ENCF1: Enable Compare Fault Channel 1**

0: Disables the FAULT output source (CPCS flag) from channel 1.

1: Enables the FAULT output source (CPCS flag) from channel 1.

51.7.22 TC Write Protection Mode Register

Name: TC_WPMR

Address: 0xF800C0E4 (0), 0xF80100E4 (1)

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x54494D (“TIM” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x54494D (“TIM” in ASCII).

The Timer Counter clock of the first channel must be enabled to access this register.

See Section 51.6.19 “Register Write Protection” for a list of registers that can be write-protected and Timer Counter clock conditions.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x54494D	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

52. Pulse Density Modulation Interface Controller (PDMIC)

52.1 Description

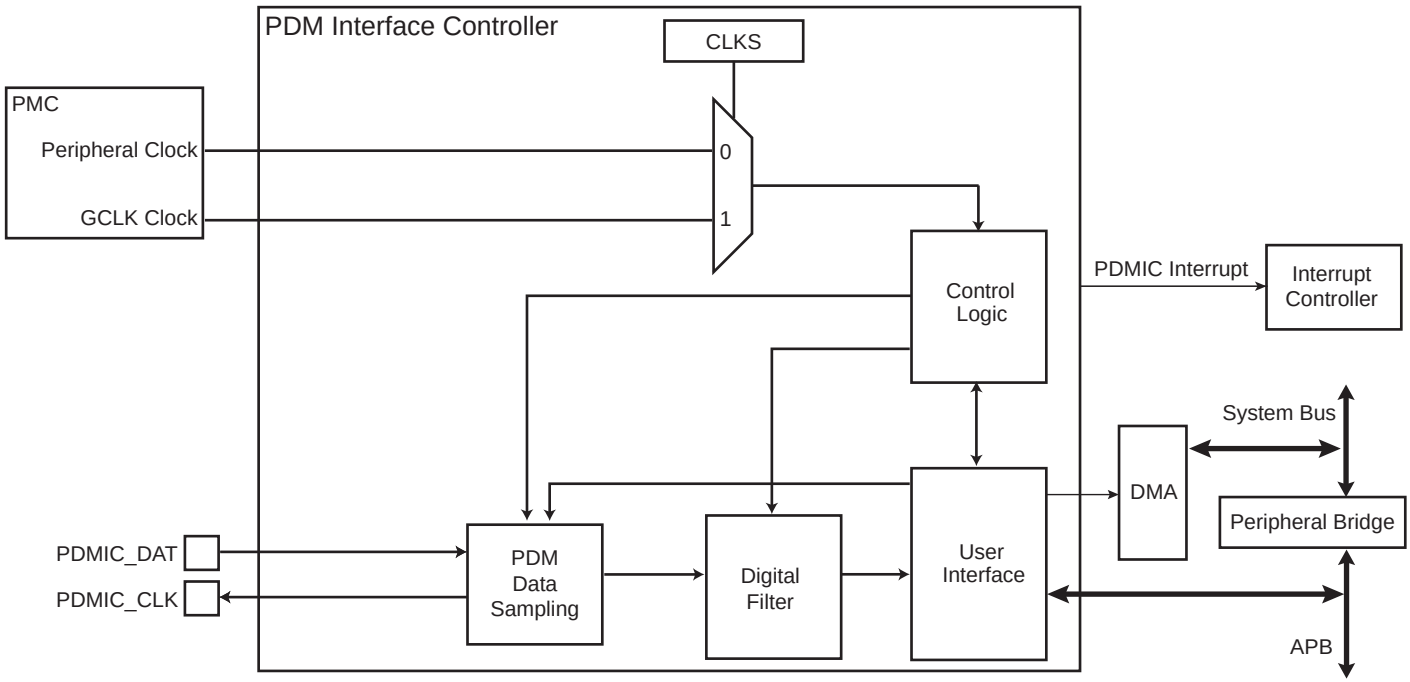
The Pulse Density Modulation Interface Controller (PDMIC) is a PDM interface controller and decoder that support mono PDM format. It integrates a clock generator driving the PDM microphone and embeds filters which decimate the incoming bitstream to obtain most common audio rates.

52.2 Embedded Characteristics

- 16-bit Resolution
- DMA Controller Support
- Up to 4 Conversions Stored
- PDM Clock Source can be Independent from Core Clock
- Register Write Protection

52.3 Block Diagram

Figure 52-1. PDMIC Block Diagram



52.4 Signal Description

Table 52-1. PDMIC Pin Description

Pin Name	Description	Type
PDMIC_CLK	Pulse Density Modulation Bitstream Sampling Clock	Output
PDMIC_DAT	Pulse Density Modulation Data	Input

52.5 Product Dependencies

52.5.1 I/O Lines

The pins used for interfacing the PDMIC are multiplexed with PIO lines. The programmer must first program the PIO controllers to assign the peripheral functions to PDMIC pins.

Table 52-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
PDMIC	PDMIC_CLK	PB12	D
PDMIC	PDMIC_CLK	PB27	D
PDMIC	PDMIC_DAT	PB11	D
PDMIC	PDMIC_DAT	PB26	D

52.5.2 Power Management

The PDMIC is not continuously clocked. The user must first enable the PDMIC peripheral clock and the PDMIC Generic Clock in the Power Management Controller (PMC) before using the controller.

52.5.3 Interrupt Sources

The PDMIC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the PDMIC interrupt requires the Interrupt Controller to be programmed first.

Table 52-3. Peripheral IDs

Instance	ID
PDMIC	48

52.6 Functional Description

52.6.1 PDM Interface

52.6.1.1 Description

The PDM clock (PDMIC_CLK) is used to sample the PDM bitstream.

The PDMIC_CLK frequency range is between peripheral clock/2 and peripheral clock/256 or between GCLK clock/2 and GCLK clock/256, depending on the selected clock source.

The GCLK clock frequency must always be at least three times lower than the peripheral clock frequency.

The field PRESCAL in the Mode Register (PDMIC_MR) must be programmed in order to provide a PDMIC_CLK frequency compliant with the microphone parameters.

52.6.1.2 Startup Sequence

To start processing the bitstream coming from the PDM interface, follow the steps below:

1. Clear all bits in the Control Register (PDMIC_CR) or compute a soft reset using the SWRST bit of PDMIC_CR.
2. Configure the PRESCAL field in PDMIC_MR according to the microphone specifications.
3. Enable the PDM mode and start the conversions using the ENPDM bit in PDMIC_CR.

52.6.2 Digital Signal Processing (Digital Filter)

52.6.2.1 Description

The PDMIC includes a DSP section containing a decimation filter, a droop compensation filter, a sixth-order low pass filter, a first-order high pass filter and an offset and gain compensation stage. A block diagram of the DSP section is represented in Figure 52-2. DSP Block Diagram.

Data processed by the filtering section are two's complement signals defined on 24 bits.

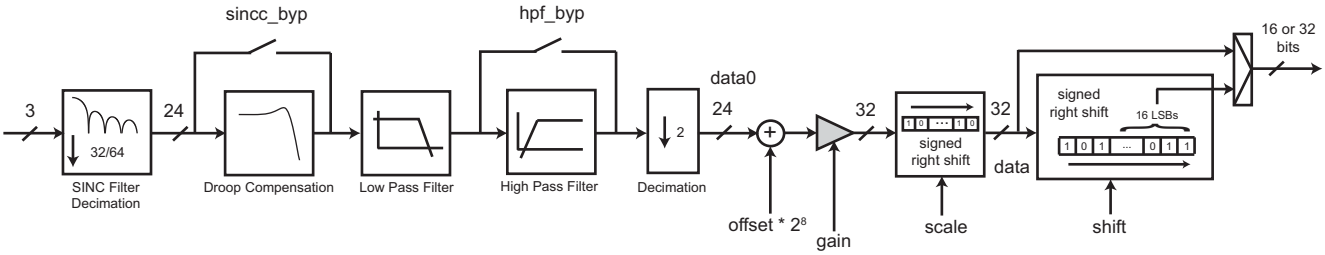
The filtering of the decimation stage is performed by a fourth-order sinc-based filter whose zeros are placed in order to minimize aliasing effects of the decimation. The decimation ratio of this filter is either 32 or 64. The droop induced by this filter can be compensated by the droop compensation stage.

The sixth-order low pass filter is used to decimate the sinc filter output by a ratio of 2.

An optional first-order high pass filter is implemented in order to eliminate the DC component of the incoming signal.

The overall decimation ratio of this DSP section is either 64 or 128. This fits an audio sampling rate of 48 kHz with a PDM microphone sampling frequency of either 3.072 or 6.144 MHz. The frequency response of the filters optimizes the gain flatness between 0 and 20 kHz (when the droop compensation filter is implemented and the high pass filter is bypassed) and highly reduces the aliasing effects of the decimation.

Figure 52-2. DSP Block Diagram



52.6.2.2 Decimation Filter

The sigma-delta architecture of the PDM microphone implies a filtering and a decimation of the bitstream at the output of the microphone bitstream. The decimation filter decimates the bitstream by either 32 or 64. To perform this operation, a fourth-order sinc filter with an Over-Sampling Ratio (OSR) of 32 or 64 is implemented with the following transfer function:

$$H(z) = \frac{1}{OSR^4}\left(\sum_{i=0}^{OSR-1} z^{-i}\right)^4$$

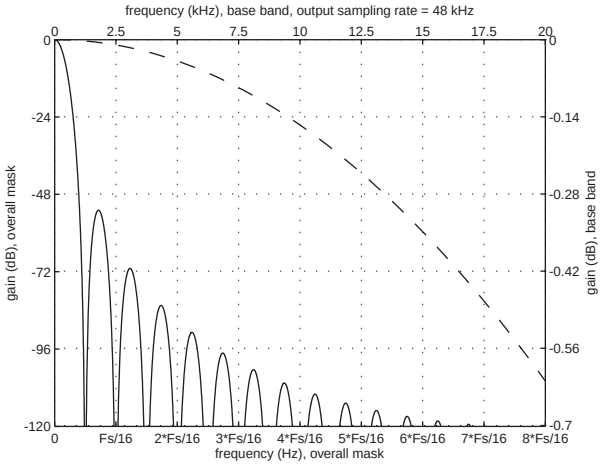
The DC gain of this filter is unity and does not depend on its OSR. However, as it generates a fourth-order zero at *Fs/OSR* frequency multiples (Fs being the sampling frequency of the microphone), the frequency response of the decimation filter depends on the OSR parameter. See Section 52.6.2.3 "Droop Compensation" for frequency plots.

Its non-flat frequency response can be compensated over the 0 to 20 kHz band by using the droop compensation filter when the decimated frequency is set to 48 kHz. See Section 52.6.2.3 "Droop Compensation".

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

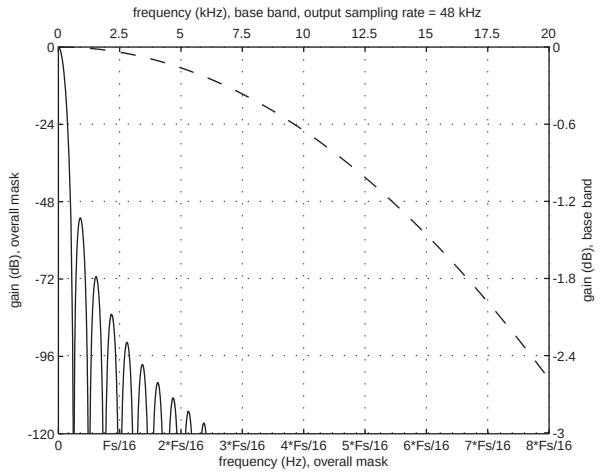
In Figure 52-3 and Figure 52-4, Fs is the sampling rate of the PDM microphone.

Figure 52-3. Spectral mask of an OSR = 32, Fs = 6.144 MHz, Fourth-Order Sinc Filter: Overall Response (continuous line) and 0 to 20 kHz Bandwidth Response (dashed line)



The zeros of this filter are located at multiples of Fs/32

Figure 52-4. Spectral Mask of an OSR = 64, Fs = 3.072 MHz, Fourth-order Sinc Filter: Overall Response (continuous line) and 0 to 20 kHz Bandwidth Response (dashed line)



The zeros of this filter are located at multiples of Fs/64.

52.6.2.3 Droop Compensation

The droop effect introduced by the sinc filter can be compensated in the 0 to 20 kHz by the droop compensation filter (see Figure 52-5). This is a second-order IIR filter which is applied on the signal output by the sinc. The default coefficients of the droop compensation filter are computed to optimize the droop of the sinc filter with the decimated frequency equal to 48 kHz.

This filter compensates the droop of the sinc filter regardless of the OSR value.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

This filter can be bypassed by setting the SINBYP bit in the PDMIC DSP Configuration Register 0 (PDMIC_DSPR0).

Figure 52-5. Droop Compensation Filter Overall Frequency Response

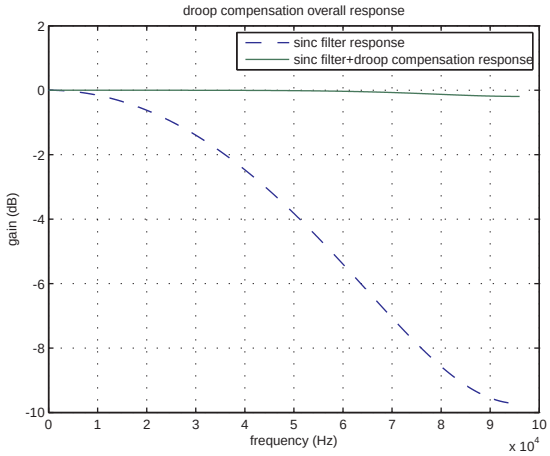
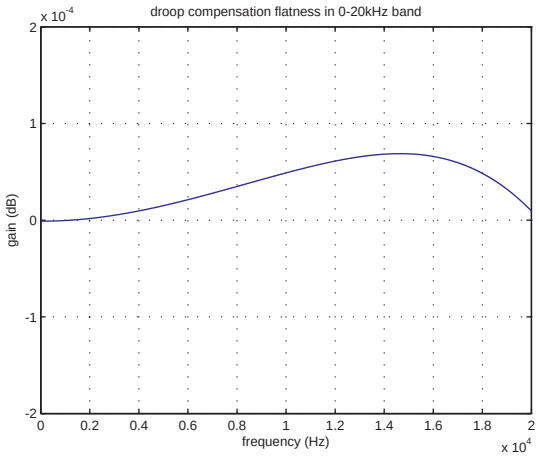


Figure 52-6. Droop Compensation Filter 0 to 20 kHz Band Flatness



52.6.2.4 Low Pass Filter

The PDMIC includes a sixth-order IIR filter that performs a low pass transfer function and decimates by 2 the output of the sinc filter. The coefficients are computed for a decimated sampling rate of 48 kHz and optimize the 0 to 20 kHz band flatness while rejecting the aliasing of the PDM microphone by at least 60 dB in the 28 to 48 kHz band.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

Figure 52-7 and Figure 52-8 are drawn for an output sampling frequency of 48 kHz.

Figure 52-7. Low Pass Filter Spectral Mask

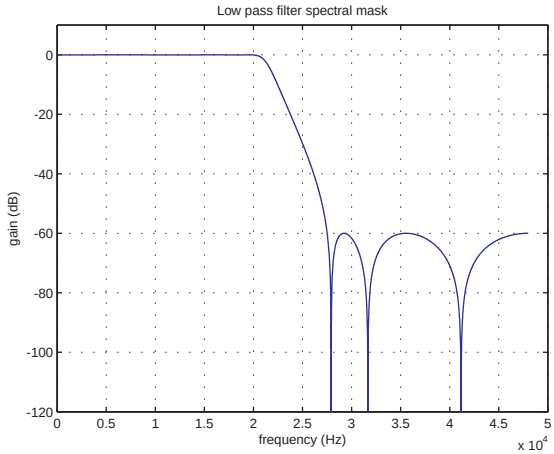
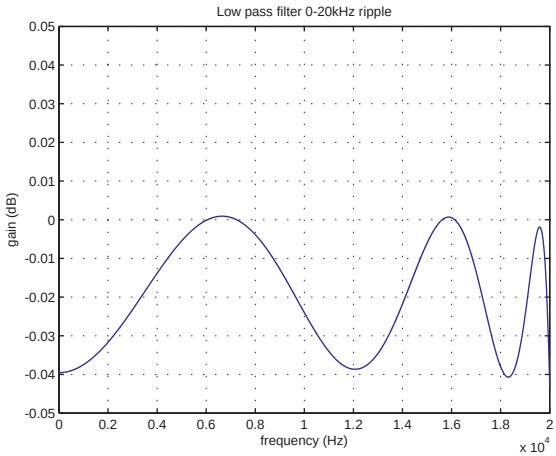


Figure 52-8. Low Pass Filter Ripple in the 0 to 20 kHz Band



52.6.2.5 High Pass Filter

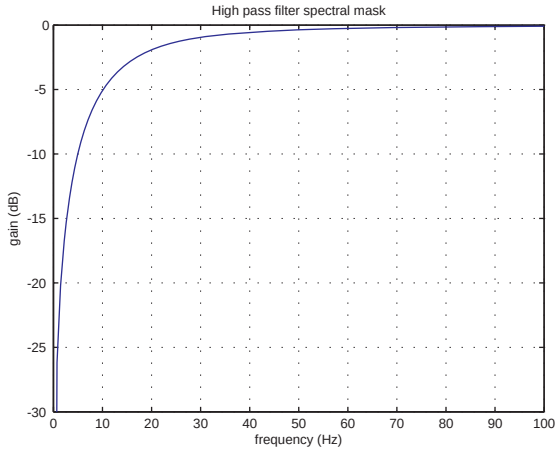
The PDMIC includes an optional first-order IIR filter performing a high pass transfer function after the low pass filter and before the decimation. The coefficients are computed for a decimated sampling rate of 48 kHz to obtain a -3dB cutoff frequency at 15 Hz.

If the decimated sampling rate is modified, the frequency response of this filter is scaled proportionally to the new frequency.

This filter can be bypassed by setting the HPFBYP bit in PDMIC_DSPR0 (see Section 52.7.8 “PDMIC DSP Configuration Register 0”).

Figure 52-9 is drawn for an output sampling frequency of 48 kHz.

Figure 52-9. High Pass Filter Spectral Mask in the 0 to 100 Hz Band



52.6.2.6 Gain and Offset Compensation

An offset, a gain, a scaling factor and a shift can be applied to a converted PDM microphone value using the following operation:

$$data = \frac{(data_0 + offset \times 2^8) \times dgain}{2^{scale + shift + 8}}$$

where:

- $data_0$ is a signed integer defined on 24 bits. It is the output of the filtering channel.
- *offset* is a signed integer defined on 16 bits (see PDMIC DSP Configuration Register 1). It is multiplied by 2^8 to have the same weight as $data_0$.
- *dgain* is an unsigned integer defined on 15 bits (see PDMIC DSP Configuration Register 1). Only the 32 MSBs of the multiplication operation are used for scaling and shifting operations. dgain defaults to 0 after reset, which forces CDR to 0. It must be programmed to a non-zero value to read non-zero data into the PDMIC_CDR register.
- *scale* is an unsigned integer defined on 4 bits (see PDMIC DSP Configuration Register 0). It shifts the multiplication operation result by *scale* bits to the right. Maximum allowed value is 15.
- *shift* is an unsigned integer defined on 4 bits (see PDMIC DSP Configuration Register 0). It shifts the multiplication operation result by *shift* bits to the right. Maximum allowed value is 15.

If the data transfer is configured in 32-bit mode (see PDMIC DSP Configuration Register 0), the 2^{shift} division is not performed and the 32-bit result of the remaining operation is sent.

If the data transfer is configured in 16-bit mode, the 2^{shift} division is performed. The result is then saturated to be within $\pm(2^{15}-1)$ and the 16 LSBs of this saturation operation are sent to the controller as the result of the PDM microphone conversion.

Default parameters are defined to output a 16-bit result whatever the data transfer configuration may be.

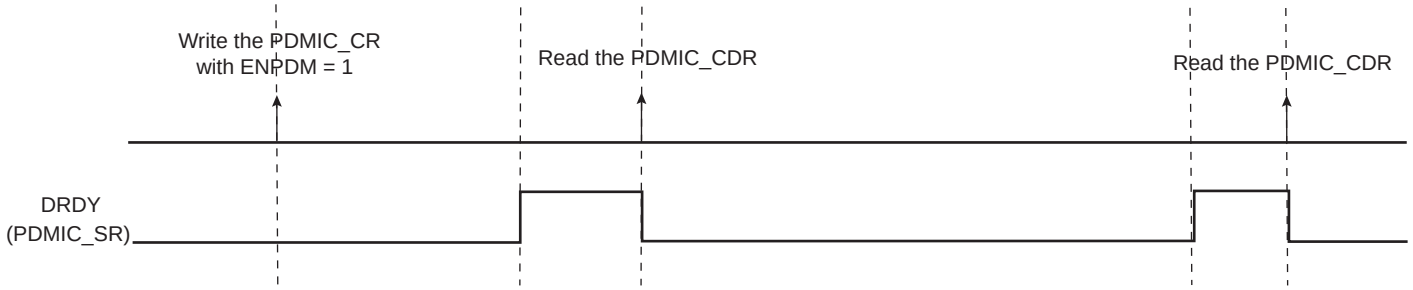
52.6.3 Conversion Results

When a conversion is completed, the resulting 16-bit digital value is stored in the PDMIC Converted Data Register (PDMIC_CDR).

The DRDY bit in the Interrupt Status Register (PDMIC_ISR) is set. In the case of a connected DMA Controller channel, DRDY rising triggers a data transfer request. In any case, DRDY can trigger an interrupt.

Reading PDMIC_CDR clears the DRDY flag.

Figure 52-10. DRDY Flag Behavior



If PDMIC_CDR is not read before further incoming data is converted, the Overrun Error (OVRE) flag is set in PDMIC_ISR. Likewise, new data converted when DRDY is high sets the OVRE bit (Overrun Error) in PDMIC_ISR. In case of overrun, the newly converted data is lost.

The OVRE flag is automatically cleared when PDMIC_ISR is read.

52.6.4 Register Write Protection

To prevent any single software error from corrupting PDMIC behavior, certain registers in the address space can be write-protected by setting the WPEN bit in the PDMIC Write Protection Mode Register (PDMIC_WPMR).

If a write access to a write-protected register is detected, the WPVS flag in the PDMIC Write Protection Status Register (PDMIC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS bit is automatically cleared after reading PDMIC_WPSR.

The following registers can be write-protected:

- PDMIC Mode Register
- PDMIC DSP Configuration Register 0
- PDMIC DSP Configuration Register 1

52.7 Pulse Density Modulation Interface Controller (PDMIC) User Interface

Table 52-4. Register Mapping

Offset[1]	Register	Name	Access	Reset
0x00	Control Register	PDMIC_CR	Read/Write	0x00000000
0x04	Mode Register	PDMIC_MR	Read/Write	0x00F00000
0x08–0x10	Reserved	–	–	–
0x14	Converted Data Register	PDMIC_CDR	Read-only	0x00000000
0x18	Interrupt Enable Register	PDMIC_IER	Write-only	–
0x1C	Interrupt Disable Register	PDMIC_IDR	Write-only	–
0x20	Interrupt Mask Register	PDMIC_IMR	Read-only	0x00000000
0x24	Interrupt Status Register	PDMIC_ISR	Read-only	0x00000000
0x28–0x54	Reserved	–	–	–
0x58	DSP Configuration Register 0	PDMIC_DSPR0	Read/Write	0x00000000
0x5C	DSP Configuration Register 1	PDMIC_DSPR1	Read/Write	0x00000001
0x60–0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	PDMIC_WPMR	Read/Write	0x00000000
0xE8	Write Protection Status Register	PDMIC_WPSR	Read-only	0x00000000
0xEC–0xFC	Reserved	–	–	–

Notes: 1. If an offset is not listed in the table, it must be considered as “reserved”.

52.7.1 PDMIC Control Register

Name: PDMIC_CR

Address: 0xF8018000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	ENPDM	–	–	–	SWRST

- **SWRST: Software Reset**

0: No effect.

1: Resets the PDMIC, simulating a hardware reset.

Warning: The read value of this bit is always 0.

- **ENPDM: Enable PDM**

0: Disables the PDM and stops the conversions.

1: Enables the PDM and starts the conversions.

52.7.2 PDMIC Mode Register

Name: PDMIC_MR

Address: 0xF8018004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	PRESCAL						
7	6	5	4	3	2	1	0
–	–	–	CLKS	–	–	–	–

This register can only be written if the WPEN bit is cleared in the PDMIC Write Protection Mode Register.

- **CLKS: Clock Source Selection**

0: Peripheral clock selected

1: GCLK clock selected (This clock source can be independent of the processor clock.)

- **PRESCAL: Prescaler Rate Selection**

PRESCAL determines the frequency of the PDM bitstream sampling clock (PDMIC_CLK):

$$PRESCAL = \frac{SELCK}{2 \times f_{PDMIC_CLK}} - 1$$

where SELCK is either $f_{peripheral\ clock}$ or $f_{GCLK\ clock}$ depending on the value of bit CLKS ($f_{peripheral\ clock}$ or $f_{GCLK\ clock}$ is the clock frequency in Hz).

52.7.3 PDMIC Converted Data Register

Name: PDMIC_CDR

Address: 0xF8018014

Access: Read-only

31	30	29	28	27	26	25	24
DATA							
23	22	21	20	19	18	17	16
DATA							
15	14	13	12	11	10	9	8
DATA							
7	6	5	4	3	2	1	0
DATA							

- **DATA: Data Converted**

The filtered output data is placed into this register at the end of a conversion and remains until it is read.

52.7.4 PDMIC Interrupt Enable Register

Name: PDMIC_IER

Address: 0xF8018018

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	OVRE	DRDY
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **DRDY: Data Ready Interrupt Enable**
- **OVRE: Overrun Error Interrupt Enable**

52.7.5 PDMIC Interrupt Disable Register

Name: PDMIC_IDR

Address: 0xF801801C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	OVRE	DRDY
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **DRDY: Data Ready Interrupt Disable**
- **OVRE: General Overrun Error Interrupt Disable**

52.7.6 PDMIC Interrupt Mask Register

Name: PDMIC_IMR

Address: 0xF8018020

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	OVRE	DRDY
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

- **DRDY: Data Ready Interrupt Mask**
- **OVRE: General Overrun Error Interrupt Mask**

52.7.7 PDMIC Interrupt Status Register

Name: PDMIC_ISR

Address: 0xF8018024

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	OVRE	DRDY
23	22	21	20	19	18	17	16
FIFOCNT							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

- **FIFOCNT: FIFO Count**

Number of conversions available in the FIFO (not a source of interrupt).

- **DRDY: Data Ready (cleared by reading PDMIC_CDR)**

0: No data has been converted since the last read of PDMIC_CDR.

1: At least one data has been converted and is available in PDMIC_CDR.

- **OVRE: Overrun Error (cleared on read)**

0: No overrun error has occurred since the last read of PDMIC_ISR.

1: At least one overrun error has occurred since the last read of PDMIC_ISR.

52.7.8 PDMIC DSP Configuration Register 0

Name: PDMIC_DSPR0

Address: 0xF8018058

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
SHIFT				SCALE			
7	6	5	4	3	2	1	0
–	OSR			SIZE	SINBYP	HPFBYP	–

This register can only be written if the WPEN bit is cleared in the PDMIC Write Protection Mode Register.

- **HPFBYP: High-Pass Filter Bypass**

0: High-pass filter enabled.

1: Bypasses the high-pass filter.

- **SINBYP: SINCC Filter Bypass**

0: Droop compensation filter enabled.

1: Bypasses the droop compensation filter.

- **SIZE: Data Size**

0: Converted data size is 16 bits.

1: Converted data size is 32 bits.

- **OSR: Global Oversampling Ratio**

Value	Name	Description
0	128	Global Oversampling ratio is 128 (SINC filter oversampling ratio is 64)
1	64	Global Oversampling ratio is 64 (SINC filter oversampling ratio is 32)

Note: Values not listed are reserved.

- **SCALE: Data Scale**

Shifts the multiplication operation result by SCALE bits to the right.

- **SHIFT: Data Shift**

Shifts the scaled result by SHIFT bits to the right.

52.7.9 PDMIC DSP Configuration Register 1

Name: PDMIC_DSPR1

Address: 0xF801805C

Access: Read/Write

31	30	29	28	27	26	25	24
OFFSET							
23	22	21	20	19	18	17	16
OFFSET							
15	14	13	12	11	10	9	8
–	DGAIN						
7	6	5	4	3	2	1	0
DGAIN							

This register can only be written if the WPEN bit is cleared in the PDMIC Write Protection Mode Register.

- **DGAIN: Gain Correction**

Gain correction to apply to the final result.

- **OFFSET: Offset Correction**

Offset correction to apply to the final result.

DGAIN and OFFSET values can be determined using the formula in Section 52.6.2.6 “Gain and Offset Compensation”.

52.7.10 PDMIC Write Protection Mode Register

Name: PDMIC_WPMR

Address: 0xF80180E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY corresponds to 0x414443 (“ADC” in ASCII).

1: Enables the write protection if WPKEY corresponds to 0x414443 (“ADC” in ASCII).

See Section 52.6.4 “Register Write Protection” for the list of registers that can be write-protected.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x414443	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0.

52.7.11 PDMIC Write Protection Status Register

Name: PDMIC_WPSR

Address: 0xF80180E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
WPVSRC							
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of PDMIC_WPSR.

1: A write protection violation has occurred since the last read of PDMIC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

53. Pulse Width Modulation Controller (PWM)

53.1 Description

The Pulse Width Modulation Controller (PWM) generates output pulses on 4 channels independently according to parameters defined per channel. Each channel controls two complementary square output waveforms. Characteristics of the output waveforms such as period, duty-cycle, polarity and dead-times (also called dead-bands or non-overlapping times) are configured through the user interface. Each channel selects and uses one of the clocks provided by the clock generator. The clock generator provides several clocks resulting from the division of the PWM peripheral clock. External triggers can be managed to allow output pulses to be modified in real time.

All accesses to the PWM are made through registers mapped on the peripheral bus. All channels integrate a double buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum, the duty-cycle or the dead-times.

Channels can be linked together as synchronous channels to be able to update their duty-cycle or dead-times at the same time.

The update of duty-cycles of synchronous channels can be performed by the DMA Controller channel which offers buffer transfer without processor Intervention.

The PWM includes a spread-spectrum counter to allow a constantly varying period (only for Channel 0). This counter may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

The PWM provides 1 independent comparison units capable of comparing a programmed value to the counter of the synchronous channels (counter of channel 0). These comparisons are intended to generate software interrupts, to trigger pulses on the 2 independent event lines (in order to synchronize ADC conversions with a lot of flexibility independently of the PWM outputs) and to trigger DMA Controller transfer requests.

PWM outputs can be overridden synchronously or asynchronously to their channel counter.

The PWM provides a fault protection mechanism with 6 fault inputs**,** capable to detect a fault condition and to override the PWM outputs asynchronously (outputs forced to ‘0’, ‘1’ or Hi-Z).

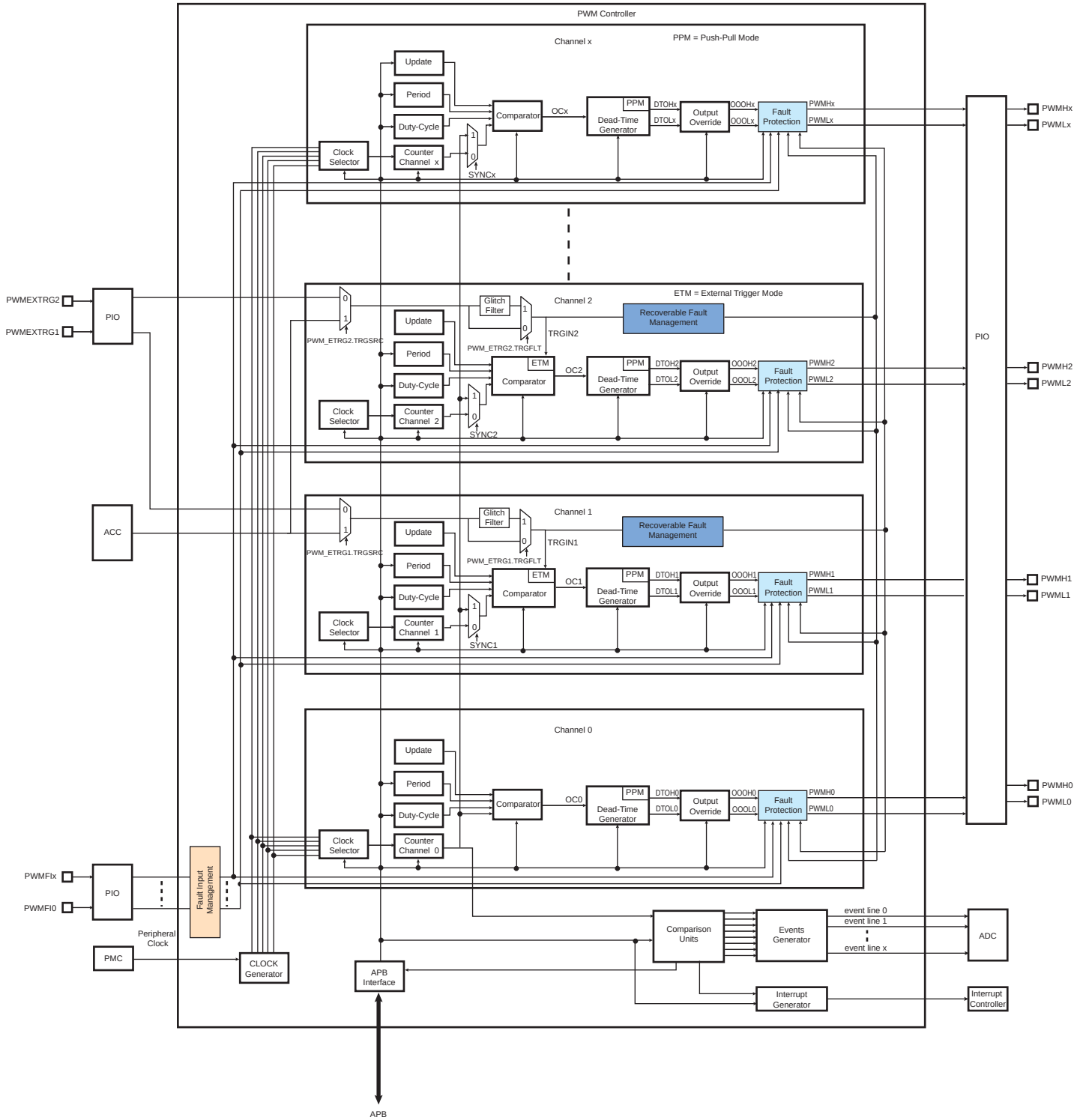
For safety usage, some configuration registers are write-protected.

53.2 Embedded Characteristics

- 4 Channels
- Common Clock Generator Providing Thirteen Different Clocks
  - A Modulo n Counter Providing Eleven Clocks
  - Two Independent Linear Dividers Working on Modulo n Counter Outputs
- Independent Channels
  - Independent 16-bit Counter for Each Channel
  - Independent Complementary Outputs with 16-bit Dead-Time Generator (Also Called Dead-Band or Non-Overlapping Time) for Each Channel
  - Independent Push-Pull Mode for Each Channel
  - Independent Enable Disable Command for Each Channel
  - Independent Clock Selection for Each Channel
  - Independent Period, Duty-Cycle and Dead-Time for Each Channel
  - Independent Double Buffering of Period, Duty-Cycle and Dead-Times for Each Channel
  - Independent Programmable Selection of The Output Waveform Polarity for Each Channel, with Double Buffering
  - Independent Programmable Center- or Left-aligned Output Waveform for Each Channel
  - Independent Output Override for Each Channel
  - Independent Interrupt for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration
  - Independent Update Time Selection of Double Buffering Registers (Polarity, Duty Cycle) for Each Channel, at Each Period for Left-Aligned or Center-Aligned Configuration
- External Trigger Input Management (e.g., for DC/DC or Lighting Control)
  - External PWM Reset Mode
  - External PWM Start Mode
  - Cycle-By-Cycle Duty Cycle Mode
  - Leading-Edge Blanking
- Two 2-bit Gray Up/Down Channels for Stepper Motor Control
- Spread Spectrum Counter to Allow a Constantly Varying Duty Cycle (only for Channel 0)
- Synchronous Channel Mode
  - Synchronous Channels Share the Same Counter
  - Mode to Update the Synchronous Channels Registers after a Programmable Number of Periods
  - Synchronous Channels Supports Connection of one DMA Controller Channel Which Offers Buffer Transfer Without Processor Intervention To Update Duty-Cycle Registers
- 2 Independent Events Lines Intended to Synchronize ADC Conversions
  - Programmable delay for Events Lines to delay ADC measurements
- 1 Comparison Units Intended to Generate Interrupts, Pulses on Event Lines and DMA Controller Transfer Requests
- 6 Programmable Fault Inputs Providing an Asynchronous Protection of PWM Outputs
  - 2 User Driven through PIO Inputs
  - PMC Driven when Crystal Oscillator Clock Fails
  - ADC Controller Driven through Configurable Comparison Function
  - Timer/Counter Driven through Configurable Comparison Function
- Register Write Protection

53.3 Block Diagram

Figure 53-1. Pulse Width Modulation Controller Block Diagram



Note: For a more detailed illustration of the fault protection circuitry, refer to Figure 53-16 “Fault Protection”.

53.4 I/O Lines Description

Each channel outputs two complementary external I/O lines.

Table 53-1. I/O Line Description

Name	Description	Type
PWMHx	PWM Waveform Output High for channel x	Output
PWMLx	PWM Waveform Output Low for channel x	Output
PWMFIx	PWM Fault Input x	Input
PWMEXTRGy	PWM Trigger Input y	Input

53.5 Product Dependencies

53.5.1 I/O Lines

The pins used for interfacing the PWM are multiplexed with PIO lines. The programmer must first program the PIO controller to assign the desired PWM pins to their peripheral function. If I/O lines of the PWM are not used by the application, they can be used for other purposes by the PIO controller.

All of the PWM outputs may or may not be enabled. If an application requires only four channels, then only four PIO lines are assigned to PWM outputs.

Table 53-2. I/O Lines

Instance	Signal	I/O Line	Peripheral
PWM	PWMEXTRG1	PB3	D
PWM	PWMEXTRG2	PB10	C
PWM	PWMFI0	PB2	D
PWM	PWMFI1	PB9	C
PWM	PWMH0	PA30	D
PWM	PWMH1	PB0	D
PWM	PWMH2	PB5	C
PWM	PWMH3	PB7	C
PWM	PWML0	PA31	D
PWM	PWML1	PB1	D
PWM	PWML2	PB6	C
PWM	PWML3	PB8	C

53.5.2 Power Management

The PWM is not continuously clocked. The programmer must first enable the PWM clock in the Power Management Controller (PMC) before using the PWM. However, if the application does not require PWM operations, the PWM clock can be stopped when not needed and be restarted later. In this case, the PWM will resume its operations where it left off.

53.5.3 Interrupt Sources

The PWM interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the PWM interrupt requires the Interrupt Controller to be programmed first.

Table 53-3. Peripheral IDs

Instance	ID
PWM	38

53.5.4 Fault Inputs

The PWM has the fault inputs connected to the different modules. Refer to the implementation of these modules within the product for detailed information about the fault generation procedure. The PWM receives faults from:

- PIO inputs
- the PMC
- the ADC controller
- Timer/Counters

Table 53-4. Fault Inputs

Fault Generator	External PWM Fault Input Number	Polarity Level(1)	Fault Input ID
PB2	PWMFI0	User-defined	0
PB9	PWMFI1	User-defined	1
PMC	–	To be configured to 1	2
ADC	–	To be configured to 1	3
Timer0	–	To be configured to 1	4
Timer1	–	To be configured to 1	5

Note: 1. FPOL field in PWMC_FMR.

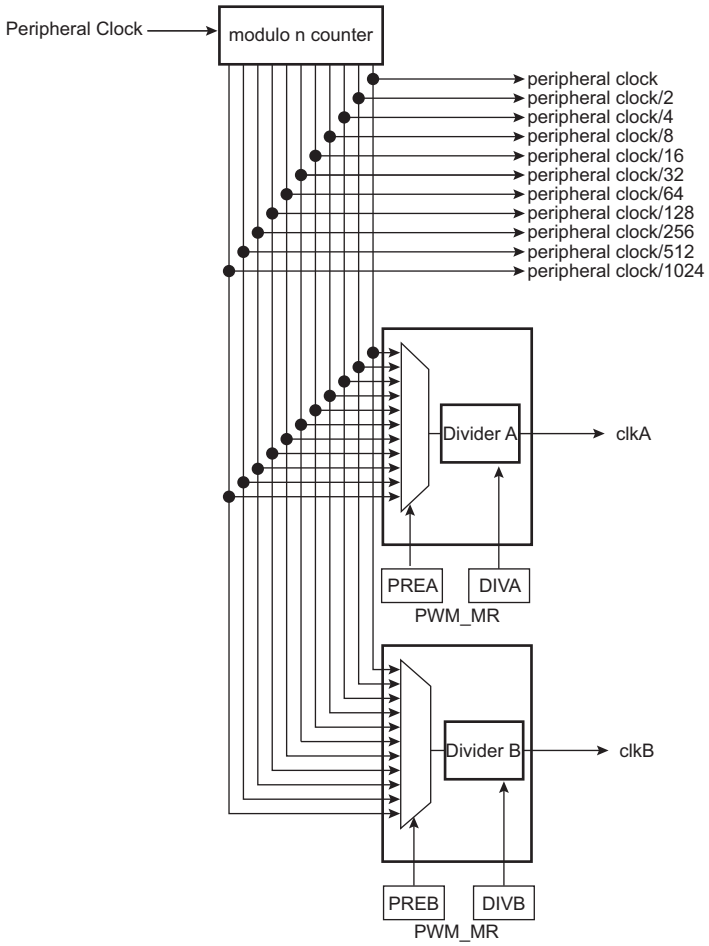
53.6 Functional Description

The PWM controller is primarily composed of a clock generator module and 4 channels.

- Clocked by the peripheral clock, the clock generator module provides 13 clocks.
- Each channel can independently choose one of the clock generator outputs.
- Each channel generates an output waveform with attributes that can be defined independently for each channel through the user interface registers.

53.6.1 PWM Clock Generator

Figure 53-2. Functional View of the Clock Generator Block Diagram



The PWM peripheral clock is divided in the clock generator module to provide different clocks available for all channels. Each channel can independently select one of the divided clocks.

The clock generator is divided into different blocks:

- a modulo n counter which provides 11 clocks: $f_{peripheral\ clock}$, $f_{peripheral\ clock}/2$, $f_{peripheral\ clock}/4$, $f_{peripheral\ clock}/8$, $f_{peripheral\ clock}/16$, $f_{peripheral\ clock}/32$, $f_{peripheral\ clock}/64$, $f_{peripheral\ clock}/128$, $f_{peripheral\ clock}/256$, $f_{peripheral\ clock}/512$, $f_{peripheral\ clock}/1024$
- two linear dividers (1, 1/2, 1/3, ... 1/255) that provide two separate clocks: clkA and clkB

Each linear divider can independently divide one of the clocks of the modulo n counter. The selection of the clock to be divided is made according to the PREA (PREB) field of the PWM Clock register (PWM_CLK). The resulting clock clkA (clkB) is the clock selected divided by DIVA (DIVB) field value.

After a reset of the PWM controller, DIVA (DIVB) and PREA (PREB) are set to ‘0’. This implies that after reset clkA (clkB) are turned off.

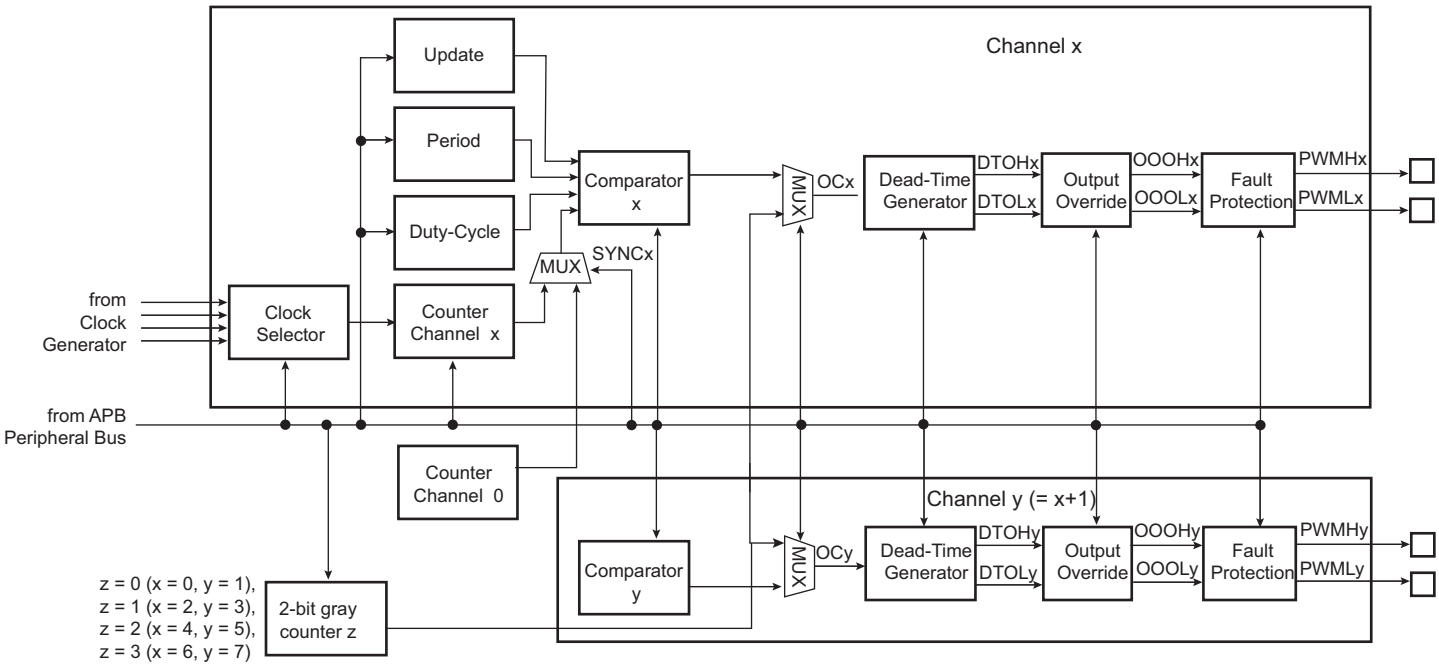
At reset, all clocks provided by the modulo n counter are turned off except the peripheral clock. This situation is also true when the PWM peripheral clock is turned off through the Power Management Controller.

CAUTION: Before using the PWM controller, the programmer must first enable the peripheral clock in the Power Management Controller (PMC).

53.6.2 PWM Channel

53.6.2.1 Channel Block Diagram

Figure 53-3. Functional View of the Channel Block Diagram



Each of the 4 channels is composed of six blocks:

- A clock selector which selects one of the clocks provided by the clock generator (described in Section 53.6.1 “PWM Clock Generator”).
- A counter clocked by the output of the clock selector. This counter is incremented or decremented according to the channel configuration and comparators matches. The size of the counter is 16 bits.
- A comparator used to compute the OCx output waveform according to the counter value and the configuration. The counter value can be the one of the channel counter or the one of the channel 0 counter according to SYNCx bit in the PWM Sync Channels Mode Register (PWM_SCM).
- A 2-bit configurable gray counter enables the stepper motor driver. One gray counter drives 2 channels.
- A dead-time generator providing two complementary outputs (DTOHx/DTOLx) which allows to drive external power control switches safely.
- An output override block that can force the two complementary outputs to a programmed value (OOOHx/OOOLx).
- An asynchronous fault protection mechanism that has the highest priority to override the two complementary outputs (PWMHx/PWMLx) in case of fault detection (outputs forced to ‘0’, ‘1’ or Hi-Z).

53.6.2.2 Comparator

The comparator continuously compares its counter value with the channel period defined by CPRD in the PWM Channel Period Register (PWM_CPRDx) and the duty-cycle defined by CDTY in the PWM Channel Duty Cycle Register (PWM_CDTYx) to generate an output signal OCx accordingly.

The different properties of the waveform of the output OCx are:

- the ***clock selection***. The channel counter is clocked by one of the clocks provided by the clock generator described in the previous section. This channel parameter is defined in the CPRE field of the PWM Channel Mode Register (PWM_CMRx). This field is reset at '0'.
- the ***waveform period***. This channel parameter is defined in the CPRD field of the PWM_CPRDx register.
  If the waveform is left-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value "X" (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(X \times CPRD)}{f_{\text{peripheral clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value "X" (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(X \times CRPD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times CRPD \times DIVB)}{f_{\text{peripheral clock}}}$$

  If the waveform is center-aligned, then the output waveform period depends on the counter source clock and can be calculated:
  By using the PWM peripheral clock divided by a given prescaler value "X" (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

  $$\frac{(2 \times X \times CPRD)}{f_{\text{peripheral clock}}}$$

  By using the PWM peripheral clock divided by a given prescaler value "X" (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

  $$\frac{(2 \times X \times CPRD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times CPRD \times DIVB)}{f_{\text{peripheral clock}}}$$

- the ***waveform duty-cycle***. This channel parameter is defined in the CDTY field of the PWM_CDTYx register.
  If the waveform is left-aligned, then:

  $$\text{duty cycle} = \frac{(period - 1/\text{fchannel_x_clock} \times CDTY)}{period}$$

  If the waveform is center-aligned, then:

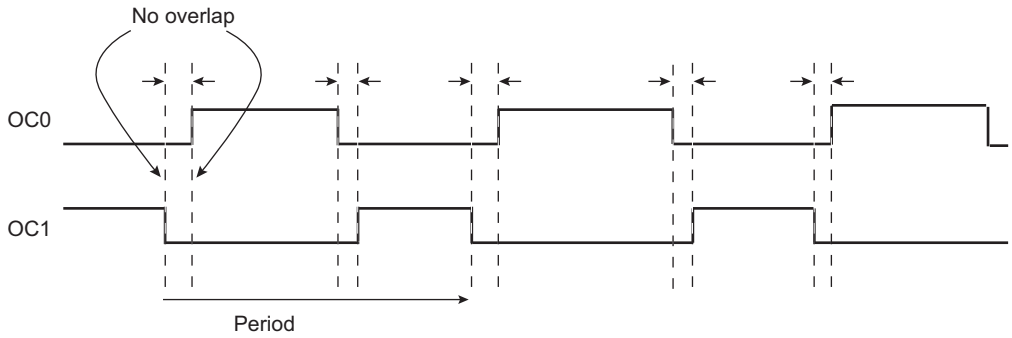
  $$\text{duty cycle} = \frac{((period/2) - 1/\text{fchannel_x_clock} \times CDTY))}{(period/2)}$$

- the ***waveform polarity.*** At the beginning of the period, the signal can be at high or low level. This property is defined in the CPOL bit of PWM_CMRx. By default, the signal starts by a low level. The DPOLI bit in PWM_CMRx defines the PWM polarity when the channel is disabled (CHIDx = 0 in PWM_SR). For more details, see Figure 53-5.

- DPOLI = 0: PWM polarity when the channel is disabled is the same as the one defined for the beginning of the PWM period.
- DPOLI = 1: PWM polarity when the channel is disabled is inverted compared to the one defined for the beginning of the PWM period.

- the ***waveform alignment***. The output waveform can be left- or center-aligned. Center-aligned waveforms can be used to generate non-overlapped waveforms. This property is defined in the CALG bit of PWM_CMRx. The default mode is left-aligned.

Figure 53-4. Non-Overlapped Center-Aligned Waveforms



Note: See Figure 53-5 for a detailed description of center-aligned waveforms.

When center-aligned, the channel counter increases up to CPRD and decreases down to 0. This ends the period.

When left-aligned, the channel counter increases up to CPRD and is reset. This ends the period.

Thus, for the same CPRD value, the period for a center-aligned channel is twice the period for a left-aligned channel.

Waveforms are fixed at 0 when:

- CDTY = CPRD and CPOL = 0 (Note that if TRGMODE = MODE3, the PWM waveform switches to 1 at the external trigger event (see Section 53.6.5.3 “Cycle-By-Cycle Duty Mode”)).
- CDTY = 0 and CPOL = 1

Waveforms are fixed at 1 (once the channel is enabled) when:

- CDTY = 0 and CPOL = 0
- CDTY = CPRD and CPOL = 1 (Note that if TRGMODE = MODE3, the PWM waveform switches to 0 at the external trigger event (see Section 53.6.5.3 “Cycle-By-Cycle Duty Mode”)).

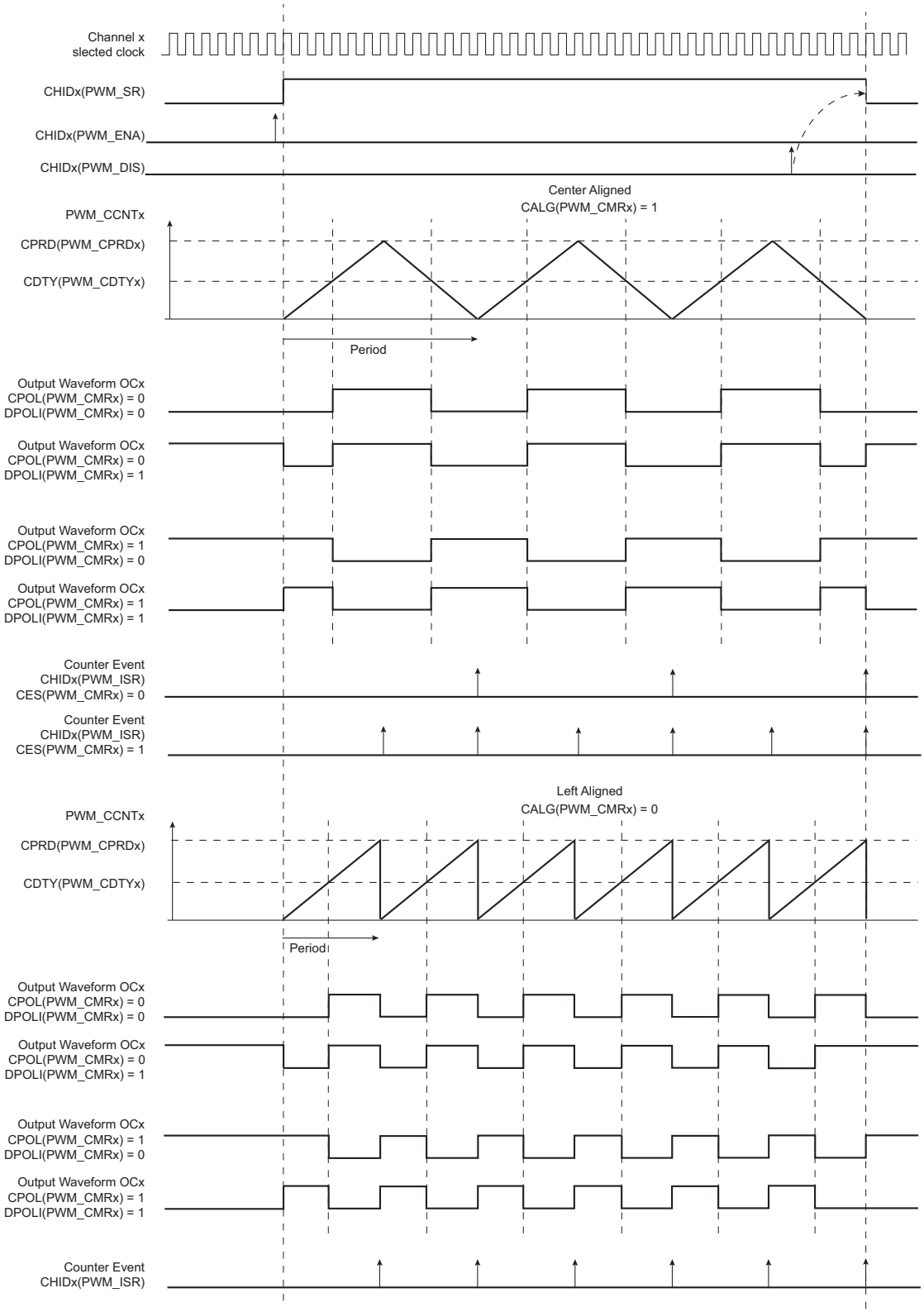
The waveform polarity must be set before enabling the channel. This immediately affects the channel output level.

Modifying CPOL in PWM Channel Mode Register while the channel is enabled can lead to an unexpected behavior of the device being driven by PWM.

In addition to generating the output signals OCx, the comparator generates interrupts depending on the counter value. When the output waveform is left-aligned, the interrupt occurs at the end of the counter period. When the output waveform is center-aligned, the bit CES of PWM_CMRx defines when the channel counter interrupt occurs. If CES is set to ‘0’, the interrupt occurs at the end of the counter period. If CES is set to ‘1’, the interrupt occurs at the end of the counter period and at half of the counter period.

Figure 53-5 illustrates the counter interrupts depending on the configuration.

Figure 53-5. Waveform Properties



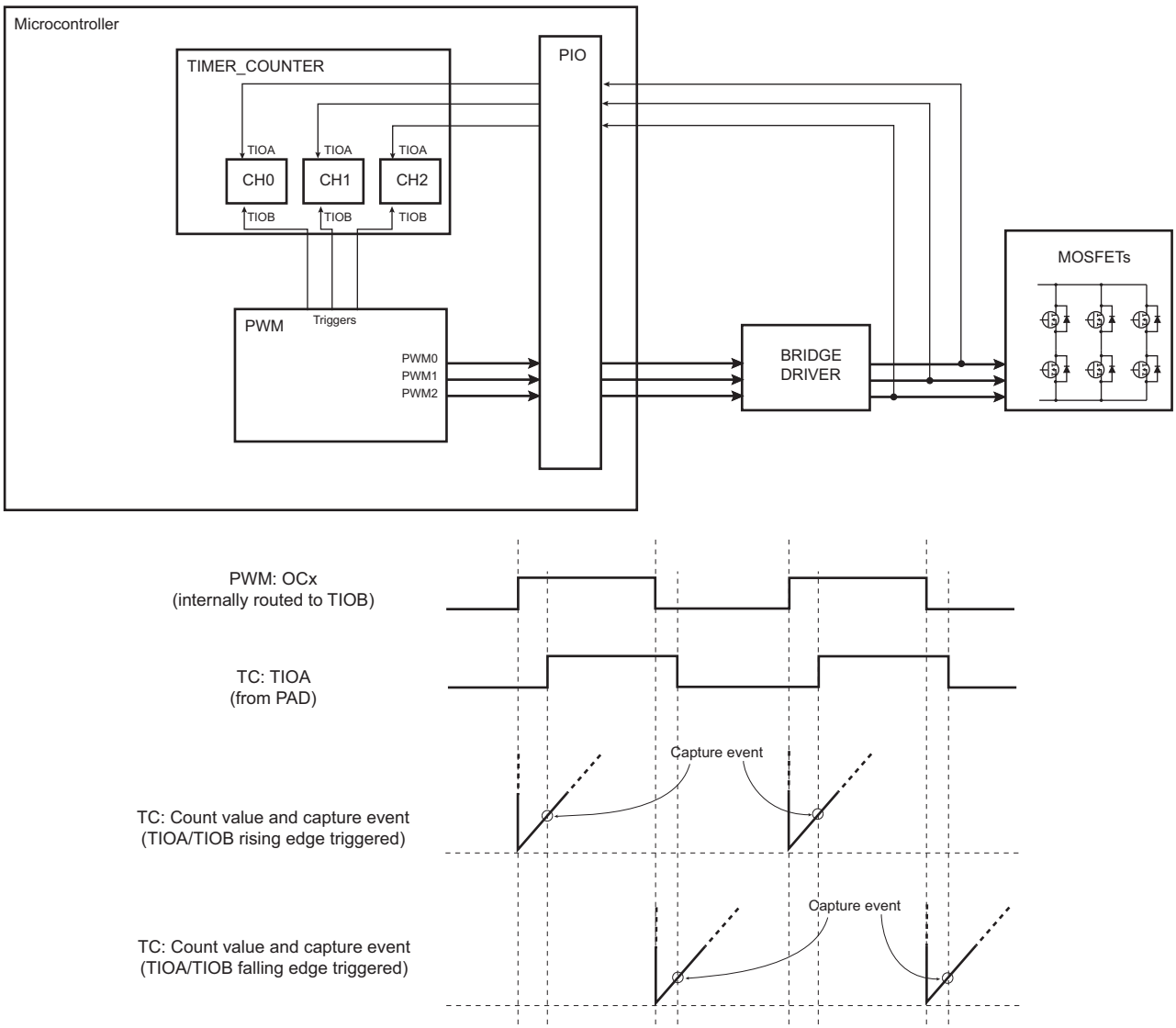
53.6.2.3 Trigger Selection for Timer Counter

The PWM controller can be used as a trigger source for the Timer Counter (TC) to achieve the two application examples described below.

Delay Measurement

To measure the delay between the channel x comparator output (OCx) and the feedback from the bridge driver of the MOSFETs (see Figure 53-6), the bit TCTS in the PWM Channel Mode Register must be at 0. This defines the comparator output of the channel x as the TC trigger source. The TIOB trigger (TC internal input) is used to start the TC; the TIOA input (from PAD) is used to capture the delay.

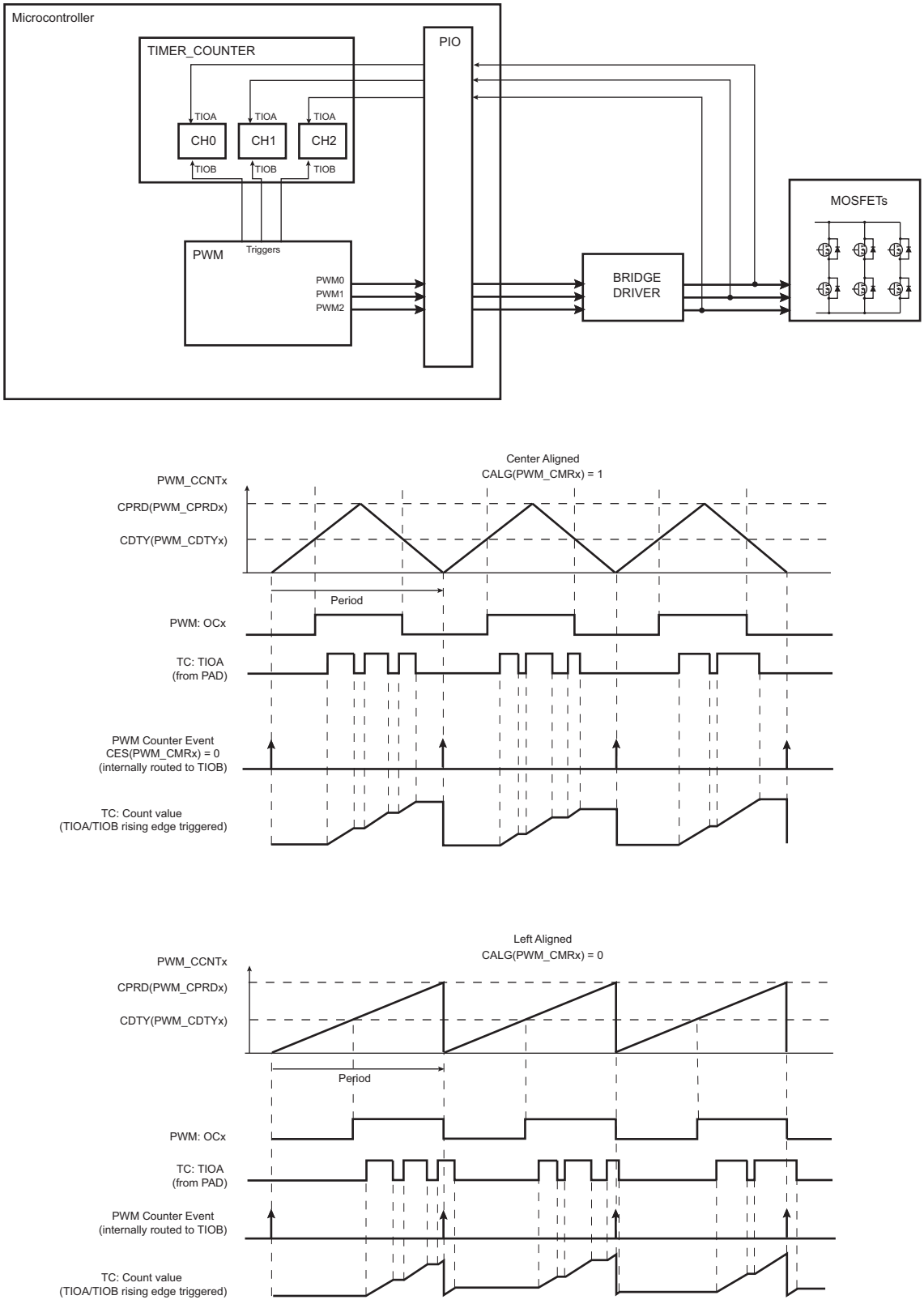
Figure 53-6. Triggering the TC: Delay Measurement



Cumulated ON Time Measurement

To measure the cumulated “ON” time of MOSFETs (see Figure 53-7), the bit TCTS of the PWM Channel Mode Register must be set to 1 to define the counter event (see Figure 53-5) as the Timer Counter trigger source.

Figure 53-7. Triggering the TC: Cumulated “ON” Time Measurement



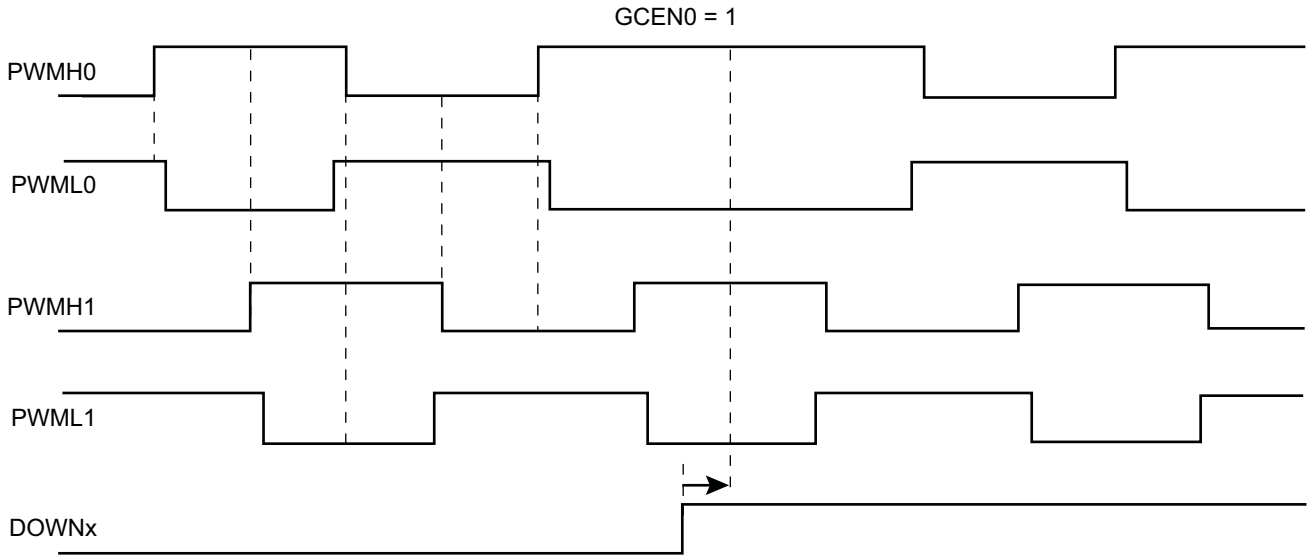
53.6.2.4 2-bit Gray Up/Down Counter for Stepper Motor

A pair of channels may provide a 2-bit gray count waveform on two outputs. Dead-time generator and other downstream logic can be configured on these channels.

Up or Down Count mode can be configured on-the-fly by means of PWM_SMMR configuration registers.

When GCEN0 is set to ‘1’, channels 0 and 1 outputs are driven with gray counter.

Figure 53-8. 2-bit Gray Up/Down Counter



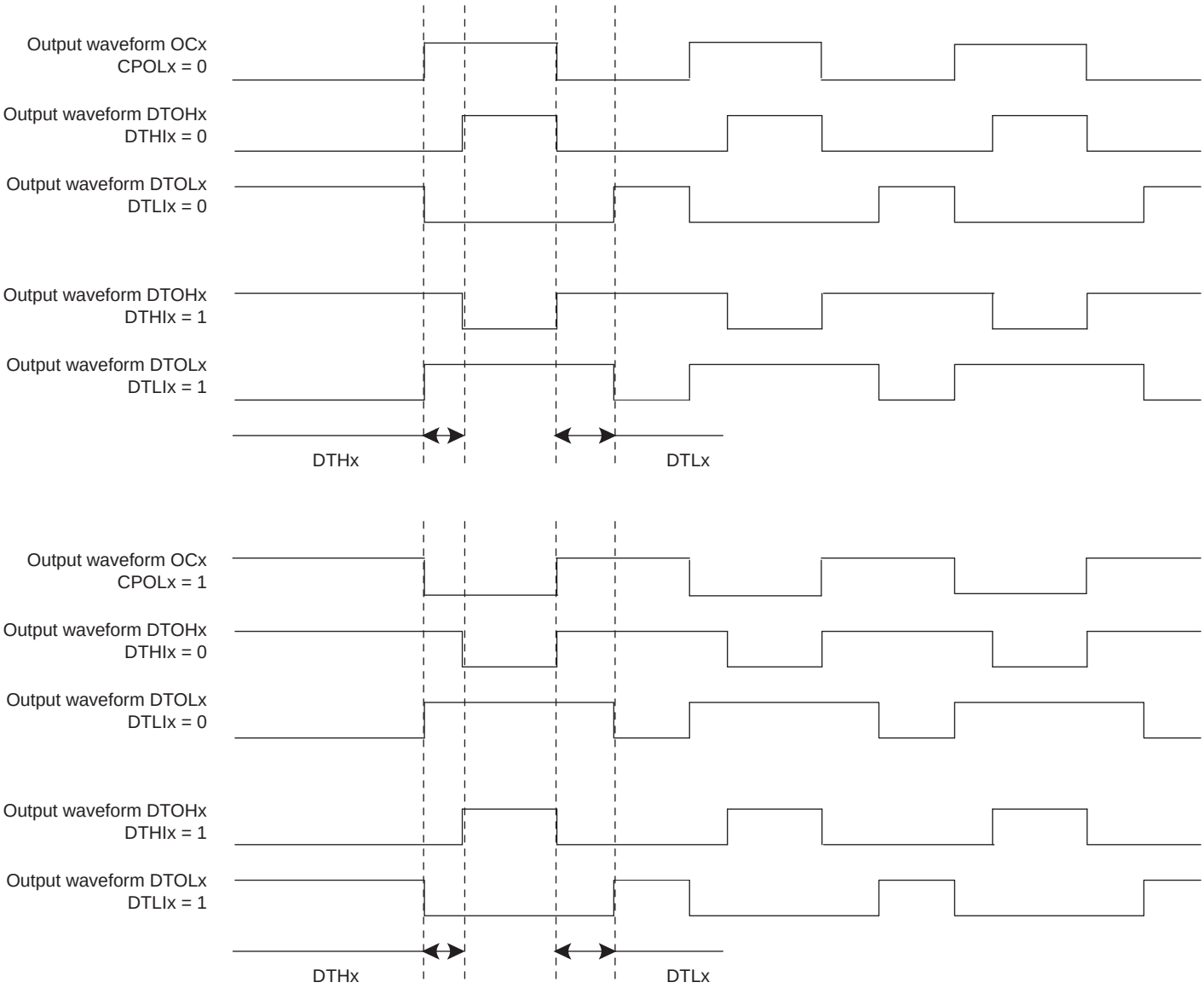
53.6.2.5 Dead-Time Generator

The dead-time generator uses the comparator output OCx to provide the two complementary outputs DTOHx and DTOLx, which allows the PWM macrocell to drive external power control switches safely. When the dead-time generator is enabled by setting the bit DTE to 1 or 0 in the PWM Channel Mode Register (PWM_CMRx), dead-times (also called dead-bands or non-overlapping times) are inserted between the edges of the two complementary outputs DTOHx and DTOLx. Note that enabling or disabling the dead-time generator is allowed only if the channel is disabled.

The dead-time is adjustable by the PWM Channel Dead Time Register (PWM_DTx). Each output of the dead-time generator can be adjusted separately by DTH and DTL. The dead-time values can be updated synchronously to the PWM period by using the PWM Channel Dead Time Update Register (PWM_DTUPDx).

The dead-time is based on a specific counter which uses the same selected clock that feeds the channel counter of the comparator. Depending on the edge and the configuration of the dead-time, DTOHx and DTOLx are delayed until the counter has reached the value defined by DTH or DTL. An inverted configuration bit (DTHI and DTLI bit in PWM_CMRx) is provided for each output to invert the dead-time outputs. The following figure shows the waveform of the dead-time generator.

Figure 53-9. Complementary Output Waveforms



PWM Push-Pull Mode

When a PWM channel is configured in Push-Pull mode, the dead-time generator output is managed alternately on each PWM cycle. The polarity of the PWM line during the idle state of the Push-Pull mode is defined by the DPOLI bit in the PWM Channel Mode Register (PWM_CMRx). The Push-Pull mode can be enabled separately on each channel by writing a one to bit PPM in the PWM Channel Mode Register.

Figure 53-10. PWM Push-Pull Mode

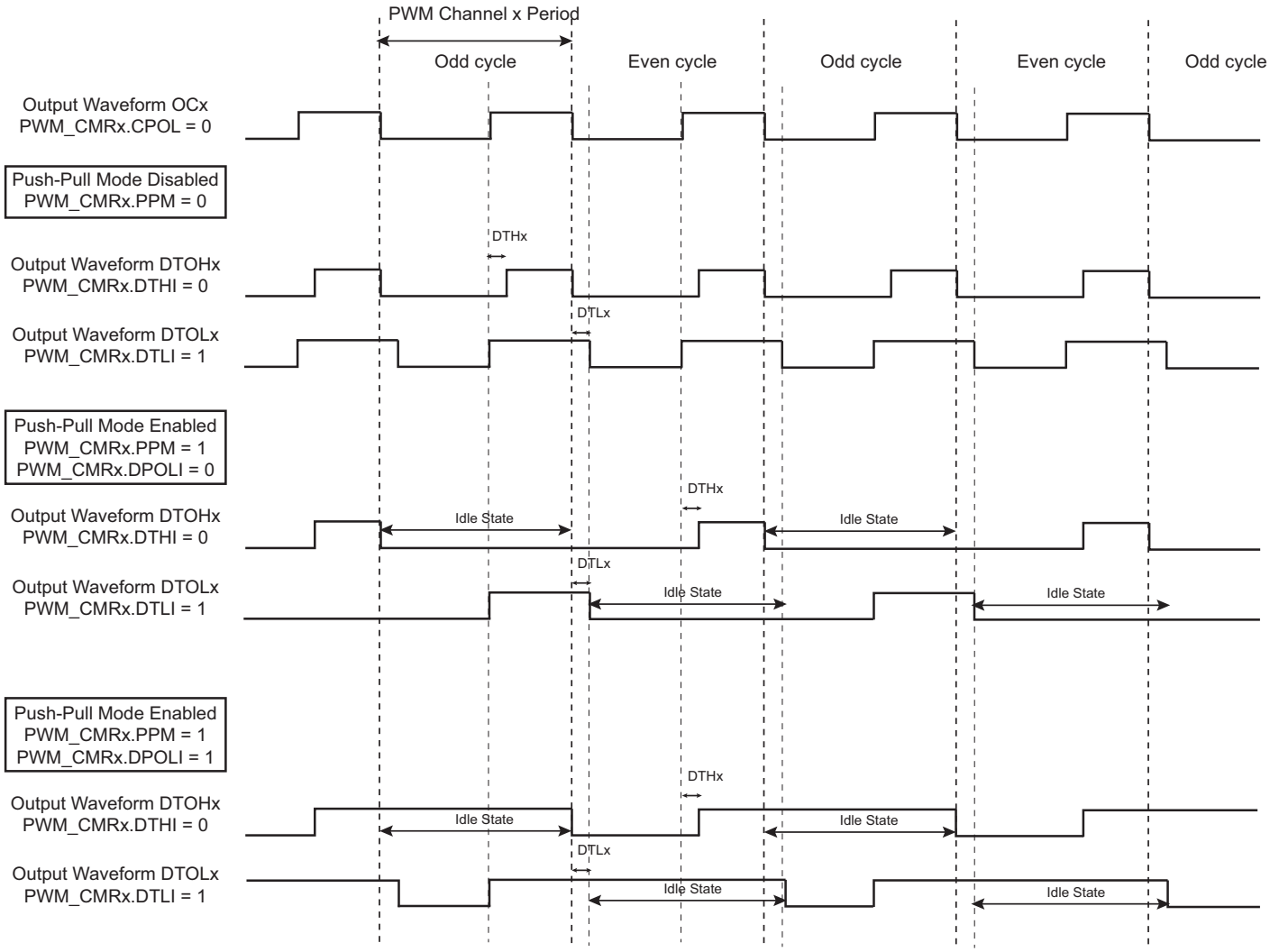


Figure 53-11. PWM Push-Pull Waveforms: Left-Aligned Mode

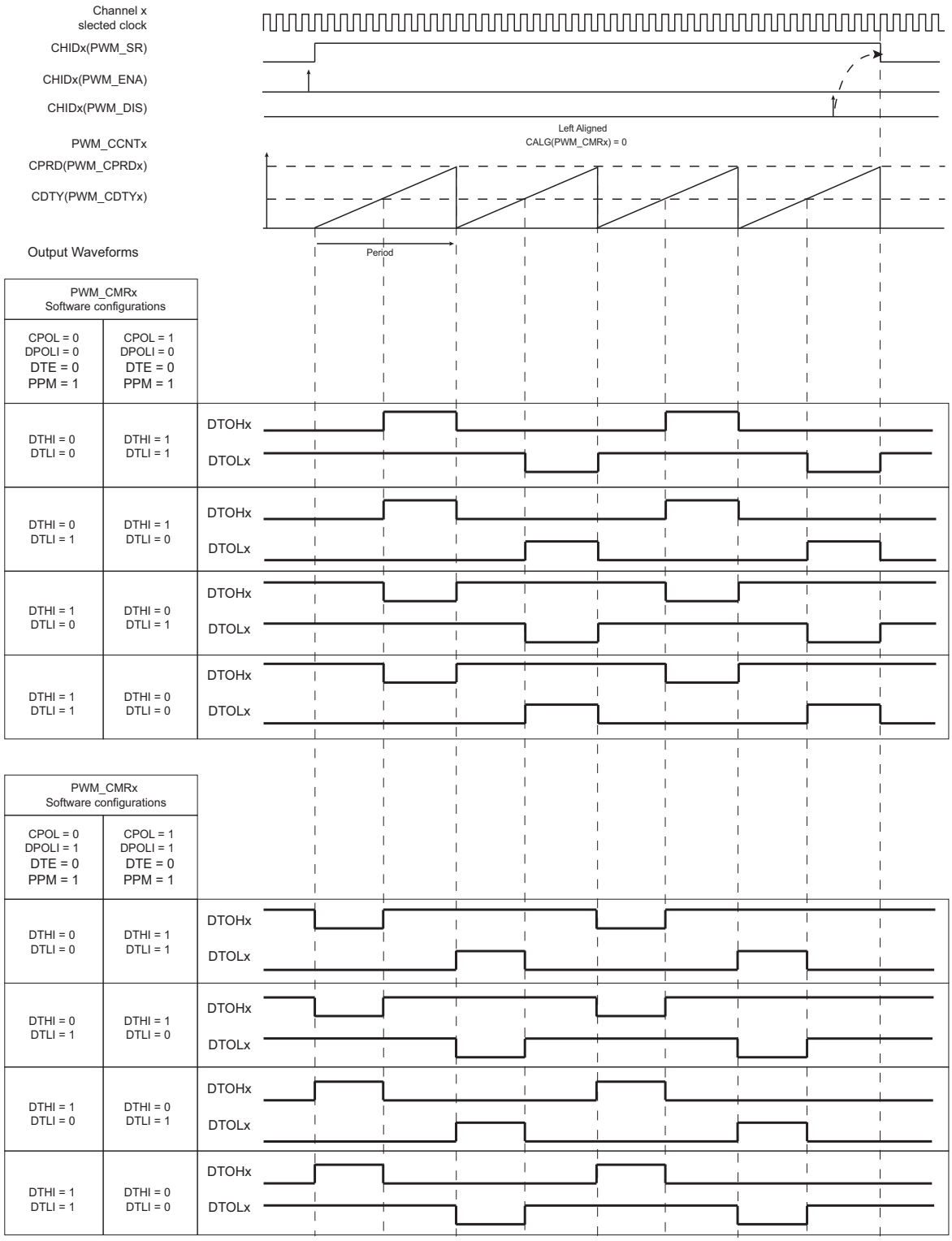
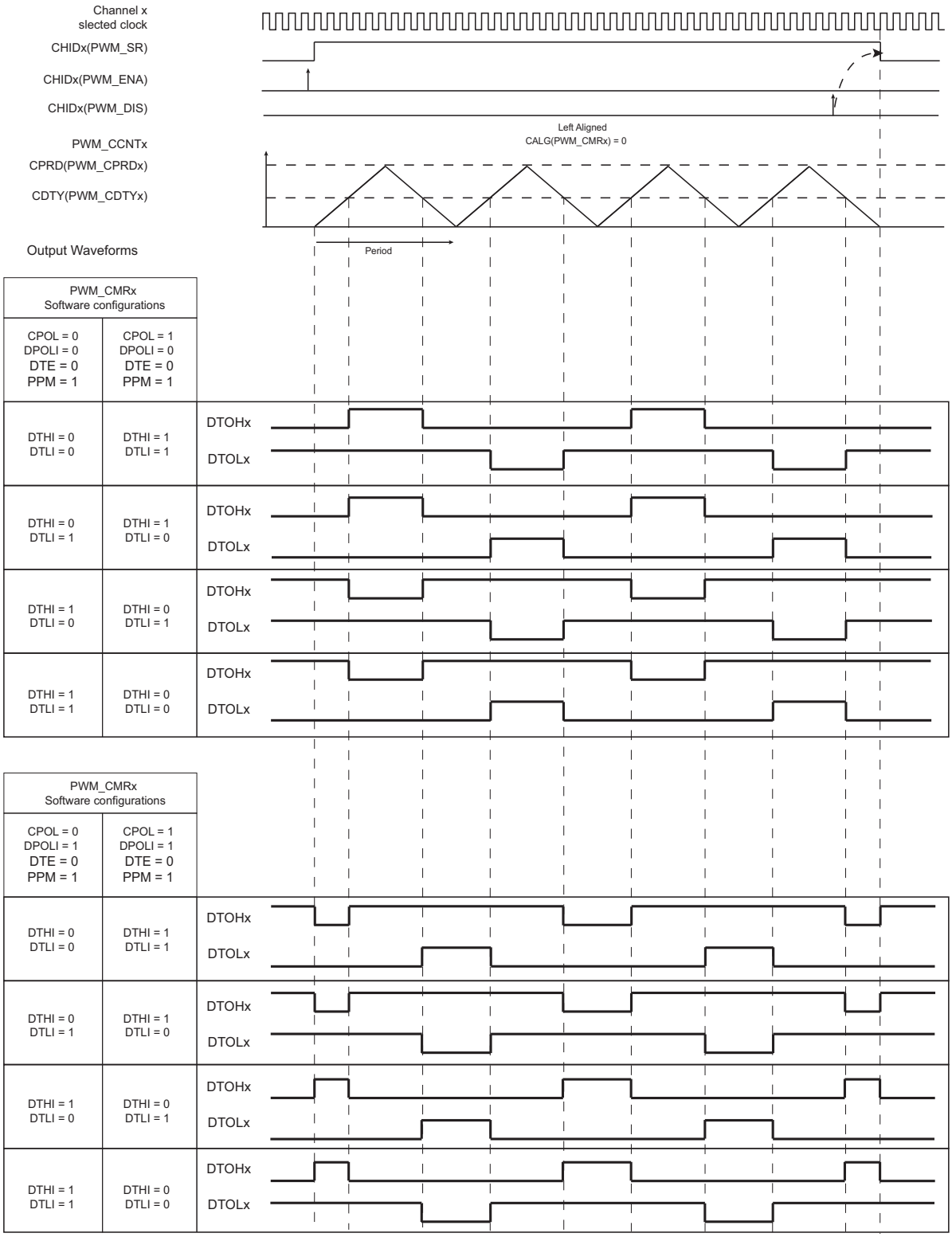


Figure 53-12. PWM Push-Pull Waveforms: Center-Aligned Mode



The PWM Push-Pull mode can be useful in transformer-based power converters, such as a half-bridge converter. The Push-Pull mode prevents the transformer core from being saturated by any direct current.

Figure 53-13. Half-Bridge Converter Application: No Feedback Regulation

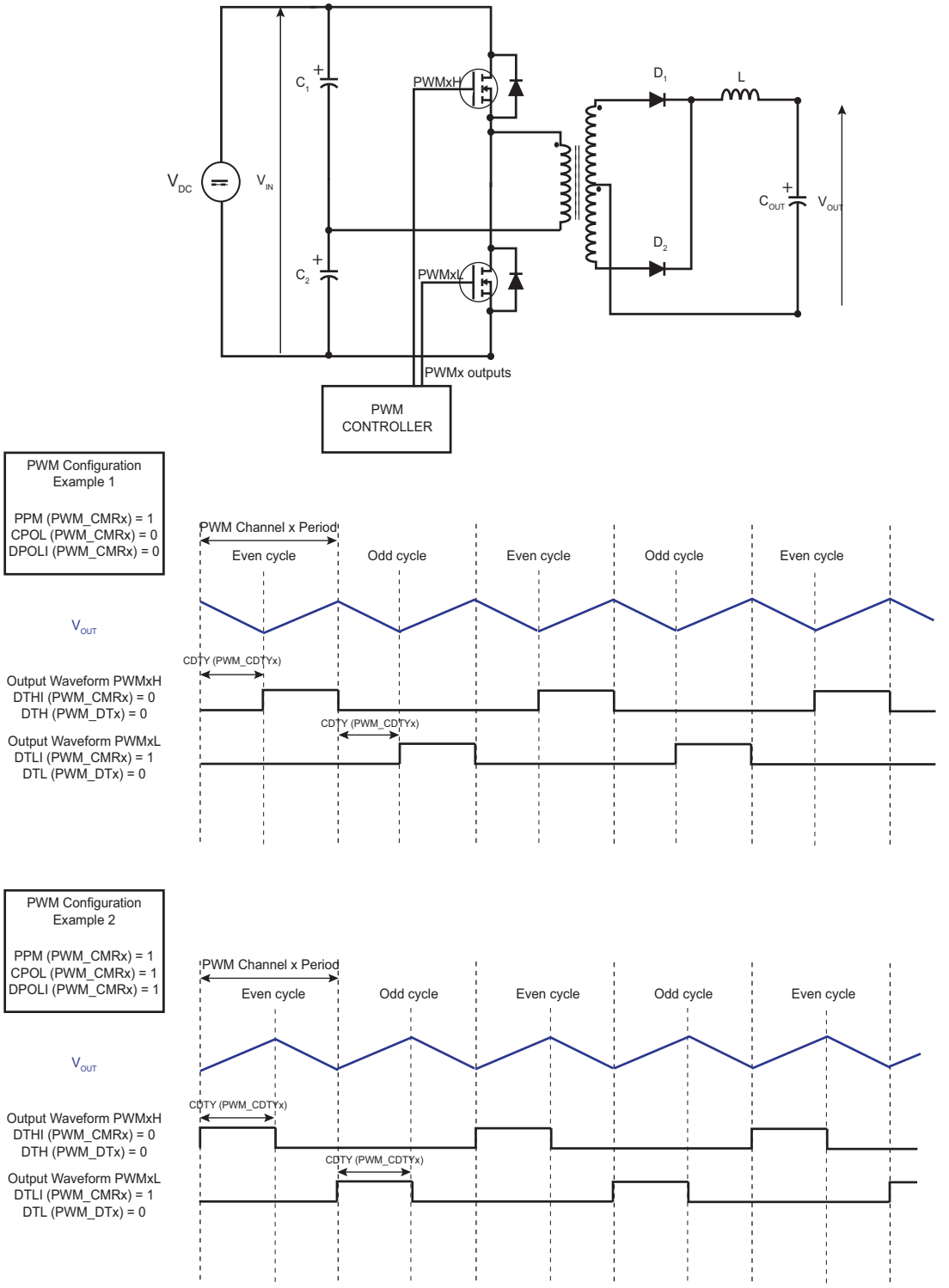
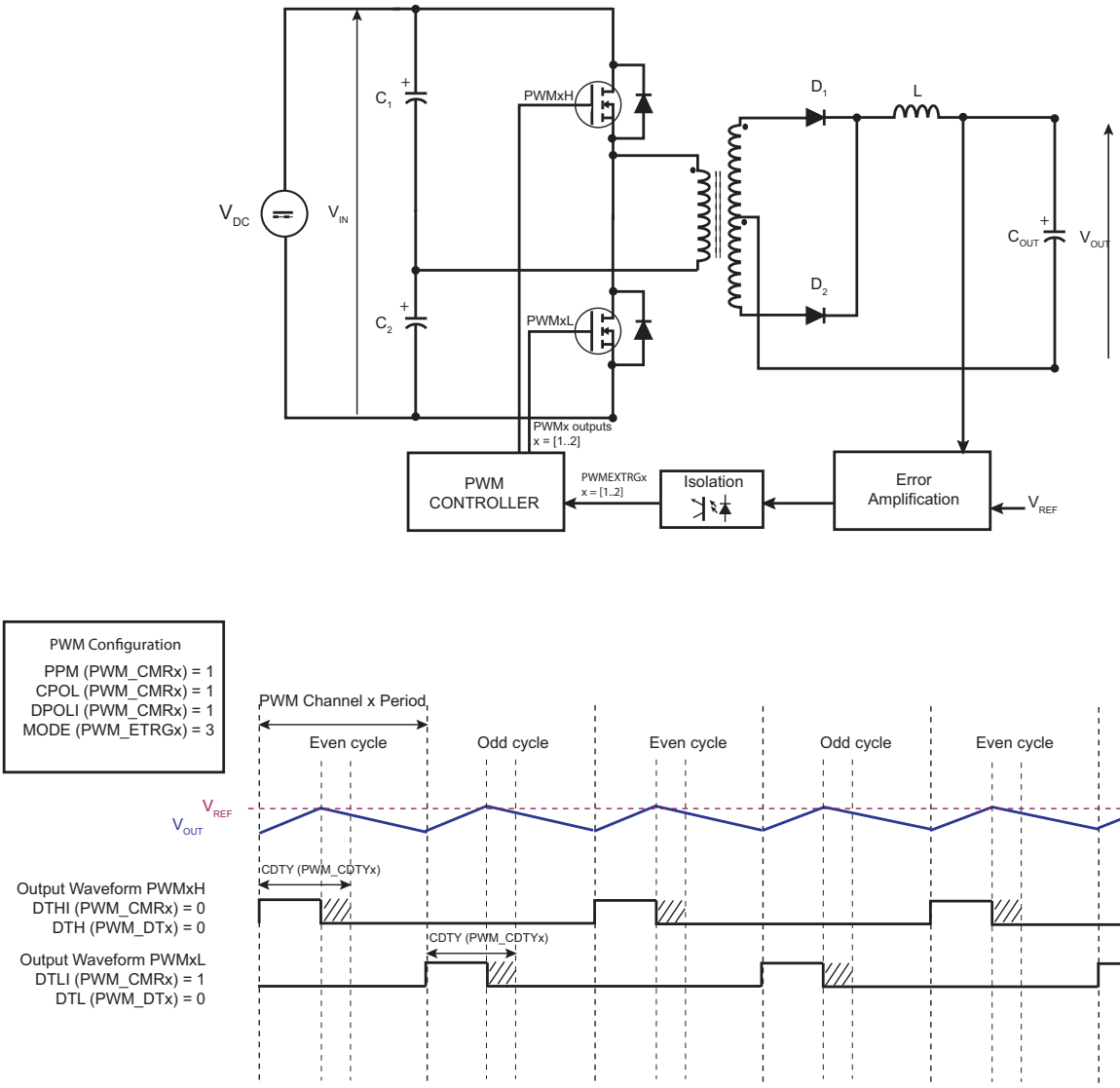


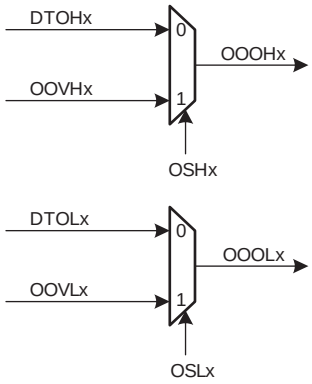
Figure 53-14. Half-Bridge Converter Application: Feedback Regulation



53.6.2.6 Output Override

The two complementary outputs DTOHx and DTOLx of the dead-time generator can be forced to a value defined by the software.

Figure 53-15. Override Output Selection



The fields OSHx and OSLx in the PWM Output Selection Register (PWM_OS) allow the outputs of the dead-time generator DTOHx and DTOLx to be overridden by the value defined in the fields OOVHx and OOVLx in the PWM Output Override Value Register (PWM_OOV).

The set registers PWM Output Selection Set Register (PWM_OSS) and PWM Output Selection Set Update Register (PWM_OSSUPD) enable the override of the outputs of a channel regardless of other channels. In the same way, the clear registers PWM Output Selection Clear Register (PWM_OSC) and PWM Output Selection Clear Update Register (PWM_OSCUPD) disable the override of the outputs of a channel regardless of other channels.

By using buffer registers PWM_OSSUPD and PWM_OSCUPD, the output selection of PWM outputs is done synchronously to the channel counter, at the beginning of the next PWM period.

By using registers PWM_OSS and PWM_OSC, the output selection of PWM outputs is done asynchronously to the channel counter, as soon as the register is written.

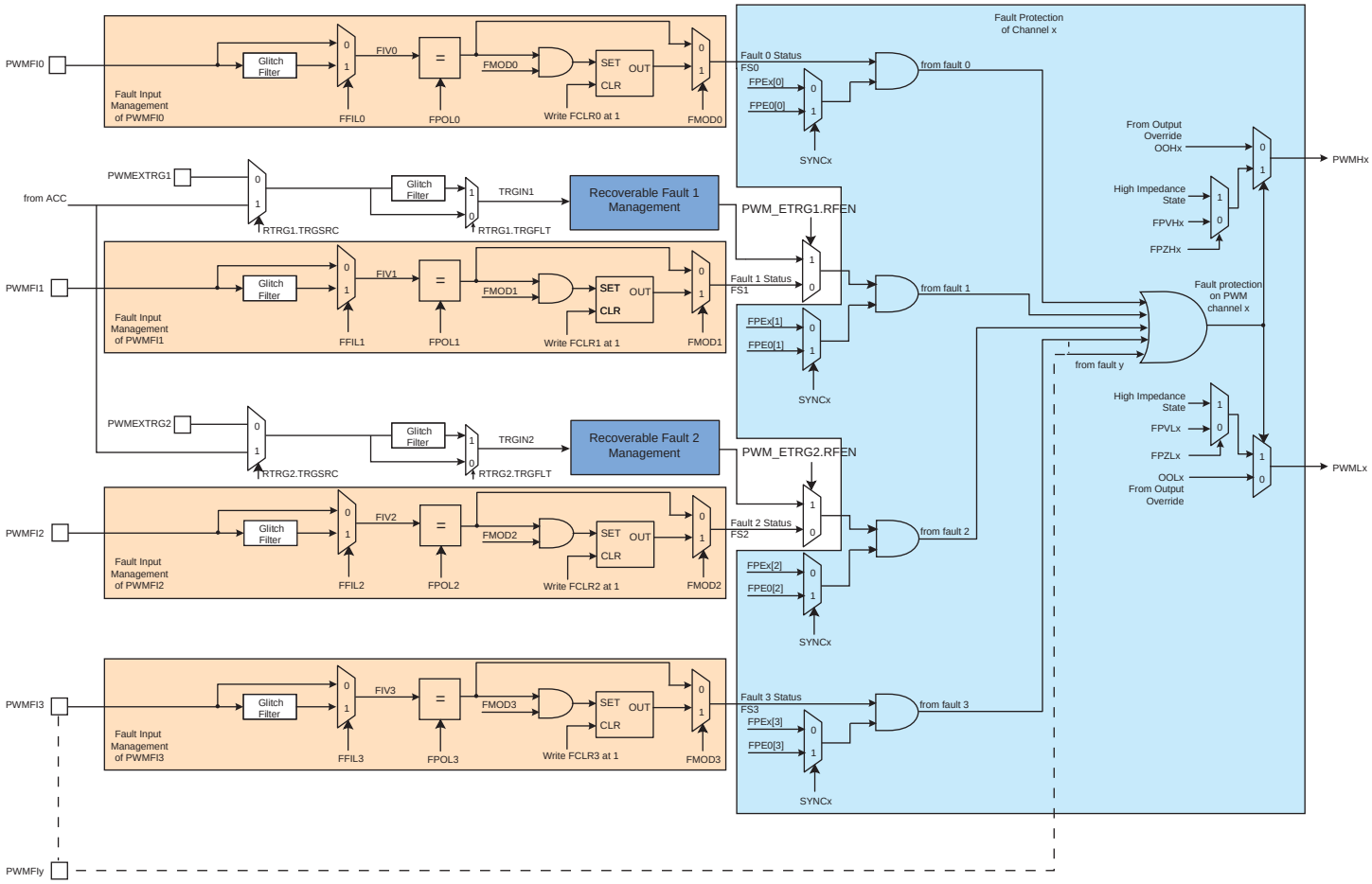
The value of the current output selection can be read in PWM_OS.

While overriding PWM outputs, the channel counters continue to run, only the PWM outputs are forced to user defined values.

53.6.2.7 Fault Protection

6 inputs provide fault protection which can force any of the PWM output pairs to a programmable value. This mechanism has priority over output overriding.

Figure 53-16. Fault Protection



The polarity level of the fault inputs is configured by the FPOL field in the PWM Fault Mode Register (PWM_FMR). For fault inputs coming from internal peripherals such as ADC or Timer Counter, the polarity level must be FPOL = 1. For fault inputs coming from external GPIO pins the polarity level depends on the user's implementation.

The configuration of the Fault Activation mode (FMOD field in PWMC_FMR) depends on the peripheral generating the fault. If the corresponding peripheral does not have “Fault Clear” management, then the FMOD configuration to use must be FMOD = 1, to avoid spurious fault detection. Refer to the corresponding peripheral documentation for details on handling fault generation.

Fault inputs may or may not be glitch-filtered depending on the FFIL field in PWM_FMR. When the filter is activated, glitches on fault inputs with a width inferior to the PWM peripheral clock period are rejected.

A fault becomes active as soon as its corresponding fault input has a transition to the programmed polarity level. If the corresponding bit FMOD is set to ‘0’ in PWM_FMR, the fault remains active as long as the fault input is at this polarity level. If the corresponding FMOD field is set to ‘1’, the fault remains active until the fault input is no longer at this polarity level and until it is cleared by writing the corresponding bit FCLR in the PWM Fault Clear Register (PWM_FCR). In the PWM Fault Status Register (PWM_FSR), the field FIV indicates the current level of the fault inputs and the field FIS indicates whether a fault is currently active.

Each fault can be taken into account or not by the fault protection mechanism in each channel. To be taken into account in the channel x, the fault y must be enabled by the bit FPEx[y] in the PWM Fault Protection Enable registers (PWM_FPE1). However, synchronous channels (see Section 53.6.2.9 “Synchronous Channels”) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[y]).

The fault protection on a channel is triggered when this channel is enabled and when any one of the faults that are enabled for this channel is active. It can be triggered even if the PWM peripheral clock is not running but only by a fault input that is not glitch-filtered.

When the fault protection is triggered on a channel, the fault protection mechanism resets the counter of this channel and forces the channel outputs to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV) and fields FPZHx/FPZLx in the PWM Fault Protection Value Register 2, as shown in Table 53-5. The output forcing is made asynchronously to the channel counter.

Table 53-5. Forcing Values of PWM Outputs by Fault Protection

FPZH/Lx	FPVH/Lx	Forcing Value of PWMH/Lx
0	0	0
0	1	1
1	–	High impedance state (Hi-Z)

CAUTION:

- To prevent any unexpected activation of the status flag FSy in PWM_FSR, the FMODy bit can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.
- To prevent any unexpected activation of the Fault Protection on the channel x, the bit FPEx[y] can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.

If a comparison unit is enabled (see Section 53.6.3 “PWM Comparison Units”) and if a fault is triggered in the channel 0, then the comparison cannot match.

As soon as the fault protection is triggered on a channel, an interrupt (different from the interrupt generated at the end of the PWM period) can be generated but only if it is enabled and not masked. The interrupt is reset by reading the interrupt status register, even if the fault which has caused the trigger of the fault protection is kept active.

Recoverable Fault

The PWM provides a Recoverable Fault mode on fault 1 and 2 (see Figure 53-16).

The recoverable fault signal is an internal signal generated as soon as an external trigger event occurs (see Section 53.6.5 “PWM External Trigger Mode”).

When the fault 1 or 2 is defined as a recoverable fault, the corresponding fault input pin is ignored and bits FFIL1/2, FMOD1/2 and FFIL1/2 are not taken into account.

The fault 1 is managed as a recoverable fault by the PWMEXTRG1 input trigger when PWM_ETRG1.RFEN = 1, PWM_ENA.CHID1 = 1, and PWM_ETRG1.TRGMODE ≠ 0.

The fault 2 is managed as a recoverable fault by the PWMEXTRG2 input trigger when PWM_ETRG2.RFEN = 1, PWM_ENA.CHID2 = 1, and PWM_ETRG2.TRGMODE ≠ 0.

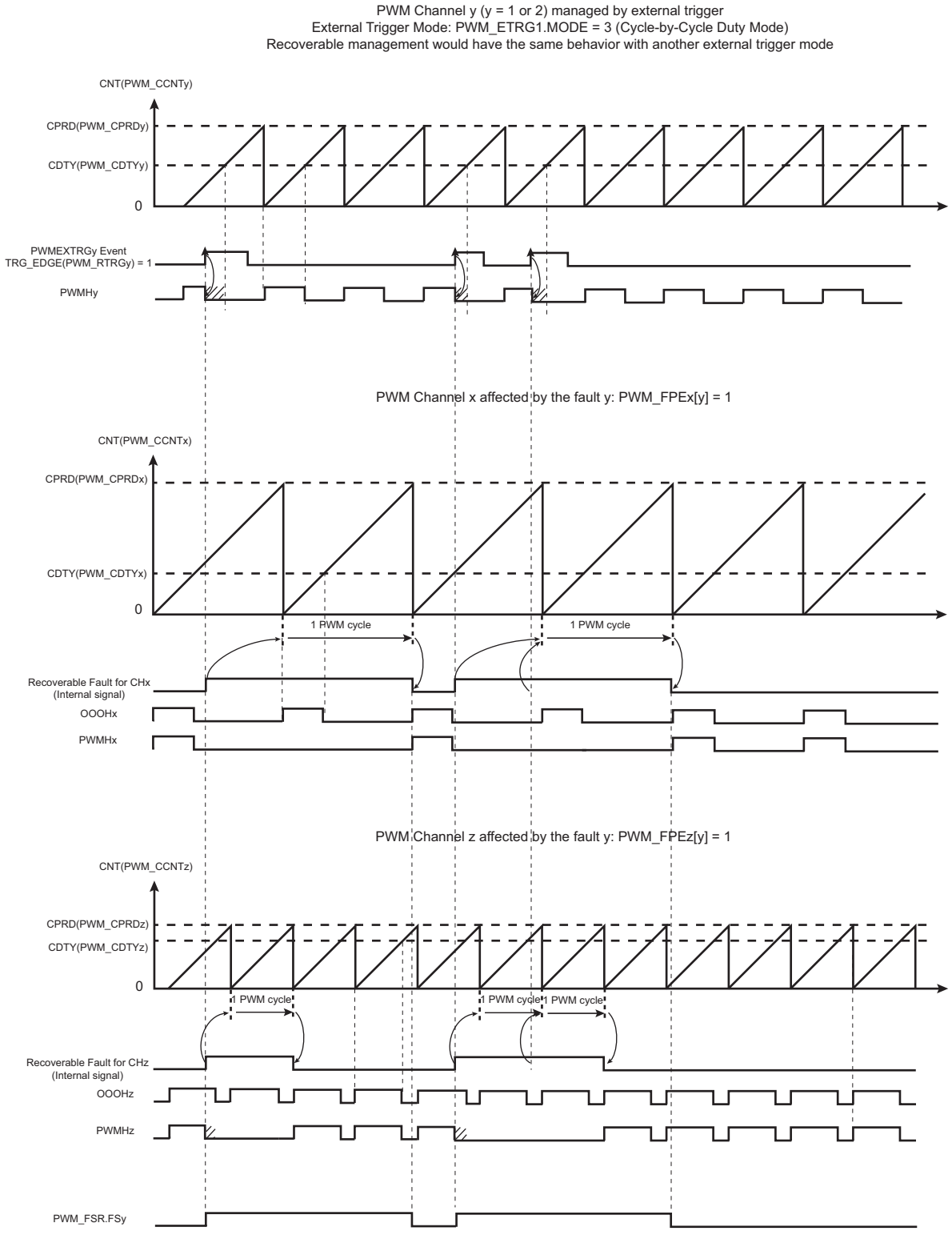
Recoverable fault 1 and 2 can be taken into account by all channels by enabling the bit FPEx[1/2] in the PWM Fault Protection Enable registers (PWM_FPEx). However the synchronous channels (see Section 53.6.2.9 “Synchronous Channels”) do not use their own fault enable bits, but those of the channel 0 (bits FPE0[1/2]).

When a recoverable fault is triggered (according to the PWM_ETRGx.TRGMODE setting), the PWM counter of the affected channels is not cleared (unlike in the classic fault protection mechanism) but the channel outputs are forced to the values defined by the fields FPVHx and FPVLx in the PWM Fault Protection Value Register 1 (PWM_FPV), as per Table 53-5. The output forcing is made asynchronously to the channel counter and lasts from

the recoverable fault occurrence to the end of the next PWM cycle (if the recoverable fault is no longer present) (see Figure 53-17).

The recoverable fault does not trigger an interrupt. The Fault Status FSy (with y = 1 or 2) is not reported in the PWM Fault Status Register when the fault y is a recoverable fault.

Figure 53-17. Recoverable Fault Management



53.6.2.8 Spread Spectrum Counter

The PWM macrocell includes a spread spectrum counter allowing the generation of a constantly varying duty cycle on the output PWM waveform (only for the channel 0). This feature may be useful to minimize electromagnetic interference or to reduce the acoustic noise of a PWM driven motor.

This is achieved by varying the effective period in a range defined by a spread spectrum value which is programmed by the field SPRD in the PWM Spread Spectrum Register (PWM_SSPR). The effective period of the output waveform is the value of the spread spectrum counter added to the programmed waveform period CPRD in the PWM Channel Period Register (PWM_CPRD0).

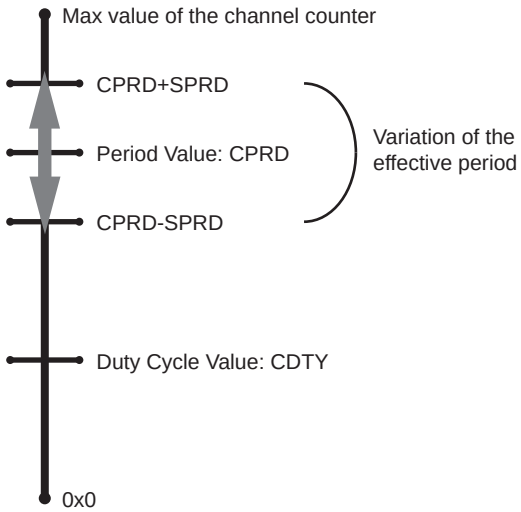
It will cause the effective period to vary from CPRD-SPRD to CPRD+SPRD. This leads to a constantly varying duty cycle on the PWM output waveform because the duty cycle value programmed is unchanged.

The value of the spread spectrum counter can change in two ways depending on the bit SPRDM in PWM_SSPR.

If SPRDM = 0, the Triangular mode is selected. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled or after reset and counts upwards at each period of the channel counter. When it reaches SPRD, it restarts to count from -SPRD again.

If SPRDM = 1, the Random mode is selected. A new random value is assigned to the spread spectrum counter at each period of the channel counter. This random value is between -SPRD and +SPRD and is uniformly distributed.

Figure 53-18. Spread Spectrum Counter



53.6.2.9 Synchronous Channels

Some channels can be linked together as synchronous channels. They have the same source clock, the same period, the same alignment and are started together. In this way, their counters are synchronized together.

The synchronous channels are defined by the SYNCx bits in the PWM Sync Channels Mode Register (PWM_SCM). Only one group of synchronous channels is allowed.

When a channel is defined as a synchronous channel, the channel 0 is also automatically defined as a synchronous channel. This is because the channel 0 counter configuration is used by all the synchronous channels.

If a channel x is defined as a synchronous channel, the fields/bits for the channel 0 are used instead of those of channel x:

- CPRE in PWM_CMR0 instead of CPRE in PWM_CMRx (same source clock)
- CPRD in PWM_CPRD0 instead of CPRD in PWM_CPRDx (same period)
- CALG in PWM_CMR0 instead of CALG in PWM_CMRx (same alignment)

Modifying the fields CPRE, CPRD and CALG of for channels with index greater than 0 has no effect on output waveforms.

Because counters of synchronous channels must start at the same time, they are all enabled together by enabling the channel 0 (by the CHID0 bit in PWM_ENA register). In the same way, they are all disabled together by disabling channel 0 (by the CHID0 bit in PWM_DIS register). However, a synchronous channel x different from channel 0 can be enabled or disabled independently from others (by the CHIDx bit in PWM_ENA and PWM_DIS registers).

Defining a channel as a synchronous channel while it is an asynchronous channel (by writing the bit SYNCx to '1' while it was at '0') is allowed only if the channel is disabled at this time (CHIDx = 0 in PWM_SR). In the same way, defining a channel as an asynchronous channel while it is a synchronous channel (by writing the SYNCx bit to '0' while it was '1') is allowed only if the channel is disabled at this time.

The UPDM field (Update Mode) in the PWM_SCM register selects one of the three methods to update the registers of the synchronous channels:

- Method 1 (UPDM = 0): The period value, the duty-cycle values and the dead-time values must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).The update is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM Sync Channels Update Control Register (PWM_SCUC) is set to '1' (see "Method 1: Manual write of duty-cycle values and manual trigger of the update").
- Method 2 (UPDM = 1): The period value, the duty-cycle values, the dead-time values and the update period value must be written by the processor in their respective update registers (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPD). The update of the period value and of the dead-time values is triggered at the next PWM period as soon as the bit UPDULOCK in the PWM_SCUC register is set to '1'. The update of the duty-cycle values and the update period value is triggered automatically after an update period defined by the field UPR in the PWM Sync Channels Update Period Register (PWM_SCUP) (see "Method 2: Manual write of duty-cycle values and automatic trigger of the update").
- Method 3 (UPDM = 2): Same as Method 2 apart from the fact that the duty-cycle values of ALL synchronous channels are written by the DMA Controller (see "Method 3: Automatic write of duty-cycle values and automatic trigger of the update"). The user can choose to synchronize the DMA Controller transfer request with a comparison match (see Section 53.6.3 "PWM Comparison Units"), by the fields PTRM and PTRCS in the PWM_SCM register. The DMA destination address must be configured to access only the PWM DMA Register (PWM_DMAR). The DMA buffer data structure must consist of sequentially repeated duty cycles. The number of duty cycles in each sequence corresponds to the number of synchronized channels. Duty cycles in each sequence must be ordered from the lowest to the highest channel index. The size of the duty cycle is 16 bits.

Table 53-6. Summary of the Update of Registers of Synchronous Channels

Register	UPDM = 0	UPDM = 1	UPDM = 2
Period Value (PWM_CPRDUPDx)	Write by the processor		
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to '1'		
Dead-Time Values (PWM_DTUPDx)	Write by the processor		
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to '1'		
Duty-Cycle Values (PWM_CDTYUPDx)	Write by the processor	Write by the processor	Write by the DMA Controller
	Update is triggered at the next PWM period as soon as the bit UPDULOCK is set to '1'	Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR	

Table 53-6. Summary of the Update of Registers of Synchronous Channels (Continued)

Register	UPDM = 0	UPDM = 1	UPDM = 2
Update Period Value (PWM_SCUPUPD)	Not applicable	Write by the processor	
	Not applicable	Update is triggered at the next PWM period as soon as the update period counter has reached the value UPR	

Method 1: Manual write of duty-cycle values and manual trigger of the update

In this mode, the update of the period value, the duty-cycle values and the dead-time values must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).

To trigger the update, the user must use the bit UPDULOCK in the PWM_SCUC register which allows to update synchronously (at the same PWM period) the synchronous channels:

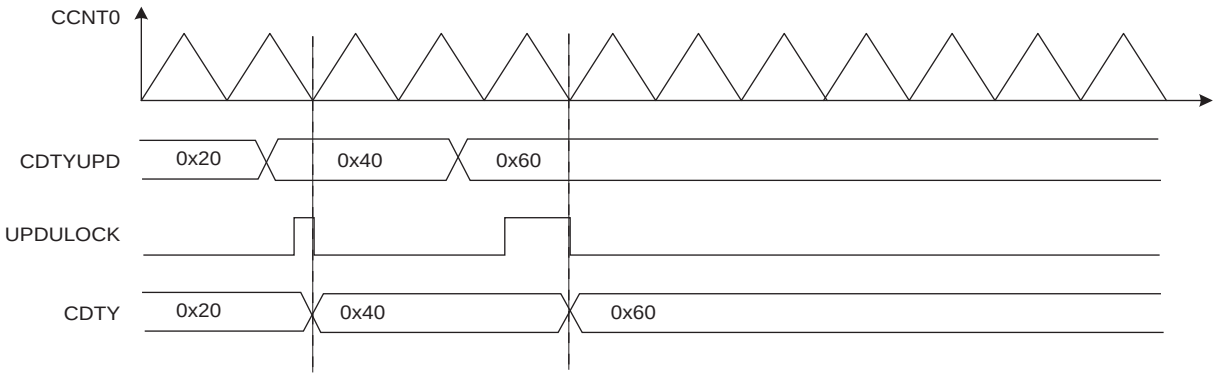
- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

Sequence for Method 1:

1. Select the manual write of duty-cycle values and the manual update by setting the UPDM field to ‘0’ in the PWM_SCM register.
2. Define the synchronous channels by the SYNCx bits in the PWM_SCM register.
3. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
4. If an update of the period value and/or the duty-cycle values and/or the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_CDTYUPDx and PWM_DTUPDx).
5. Set UPDULOCK to ‘1’ in PWM_SCUC.
6. The update of the registers will occur at the beginning of the next PWM period. When the UPDULOCK bit is reset, go to Step 4. for new values.

Figure 53-19. Method 1 (UPDM = 0)



Method 2: Manual write of duty-cycle values and automatic trigger of the update

In this mode, the update of the period value, the duty-cycle values, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_CDTYUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK in the PWM_SCUC register, which updates synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the update period by the UPR field in the PWM_SCUP register. The PWM controller waits UPR+1 period of synchronous channels before updating automatically the duty values and the update period value.

The status of the duty-cycle value write is reported in the PWM Interrupt Status Register 2 (PWM_ISR2) by the following flags:

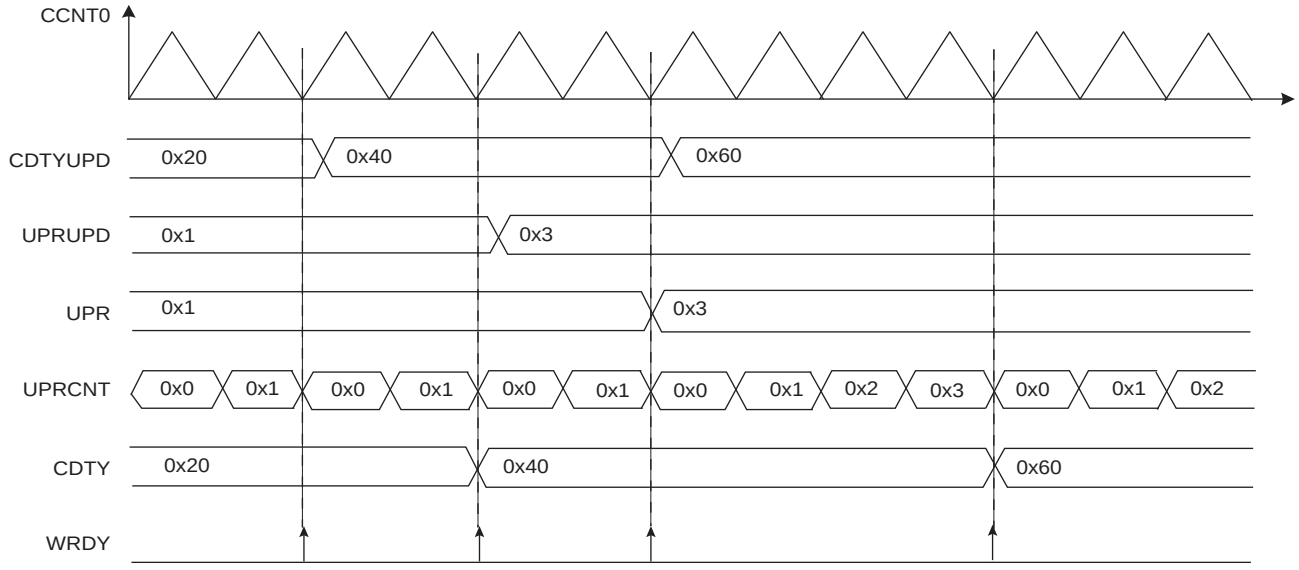
- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when the PWM_ISR2 register is read.

Depending on the interrupt mask in the PWM Interrupt Mask Register 2 (PWM_IMR2), an interrupt can be generated by these flags.

Sequence for Method 2:

1. Select the manual write of duty-cycle values and the automatic update by setting the field UPDM to ‘1’ in the PWM_SCM register
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
5. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 8.
6. Set UPDULOCK to ‘1’ in PWM_SCUC.
7. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 5. for new values.
8. If an update of the duty-cycle values and/or the update period is required, check first that write of new update values is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2.
9. Write registers that need to be updated (PWM_CDTYUPDx, PWM_SCUPUPD).
10. The update of these registers will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 8. for new values.

Figure 53-20. Method 2 (UPDM = 1)



Method 3: Automatic write of duty-cycle values and automatic trigger of the update

In this mode, the update of the duty cycle values is made automatically by the DMA Controller. The update of the period value, the dead-time values and the update period value must be done by writing in their respective update registers with the processor (respectively PWM_CPRDUPDx, PWM_DTUPDx and PWM_SCUPUPD).

To trigger the update of the period value and the dead-time values, the user must use the bit UPDULOCK which allows to update synchronously (at the same PWM period) the synchronous channels:

- If the bit UPDULOCK is set to ‘1’, the update is done at the next PWM period of the synchronous channels.
- If the UPDULOCK bit is not set to ‘1’, the update is locked and cannot be performed.

After writing the UPDULOCK bit to ‘1’, it is held at this value until the update occurs, then it is read 0.

The update of the duty-cycle values and the update period value is triggered automatically after an update period.

To configure the automatic update, the user must define a value for the Update Period by the field UPR in the PWM_SCUP register. The PWM controller waits UPR+1 periods of synchronous channels before updating automatically the duty values and the update period value.

Using the DMA Controller removes processor overhead by reducing its intervention during the transfer. This significantly reduces the number of clock cycles required for a data transfer, which improves microcontroller performance.

The DMA Controller must write the duty-cycle values in the synchronous channels index order. For example if the channels 0, 1 and 3 are synchronous channels, the DMA Controller must write the duty-cycle of the channel 0 first, then the duty-cycle of the channel 1, and finally the duty-cycle of the channel 3.

The status of the DMA Controller transfer is reported in PWM_ISR2 by the following flags:

- WRDY: this flag is set to ‘1’ when the PWM Controller is ready to receive new duty-cycle values and a new update period value. It is reset to ‘0’ when PWM_ISR2 is read. The user can choose to synchronize the WRDY flag and the DMA Controller transfer request with a comparison match (see Section 53.6.3 “PWM Comparison Units”), by the fields PTRM and PTRCS in the PWM_SCM register.
- UNRE: this flag is set to ‘1’ when the update period defined by the UPR field has elapsed while the whole data has not been written by the DMA Controller. It is reset to ‘0’ when PWM_ISR2 is read.

Depending on the interrupt mask in PWM_IMR2, an interrupt can be generated by these flags.

Sequence for Method 3:

1. Select the automatic write of duty-cycle values and automatic update by setting the field UPDM to 2 in the PWM_SCM register.
2. Define the synchronous channels by the bits SYNCx in the PWM_SCM register.
3. Define the update period by the field UPR in the PWM_SCUP register.
4. Define when the WRDY flag and the corresponding DMA Controller transfer request must be set in the update period by the PTRM bit and the PTRCS field in the PWM_SCM register (at the end of the update period or when a comparison matches).
5. Define the DMA Controller transfer settings for the duty-cycle values and enable it in the DMA Controller registers
6. Enable the synchronous channels by writing CHID0 in the PWM_ENA register.
7. If an update of the period value and/or of the dead-time values is required, write registers that need to be updated (PWM_CPRDUPDx, PWM_DTUPDx), else go to Step 10.
8. Set UPDULOCK to ‘1’ in PWM_SCUC.
9. The update of these registers will occur at the beginning of the next PWM period. At this moment the bit UPDULOCK is reset, go to Step 7. for new values.
10. If an update of the update period value is required, check first that write of a new update value is possible by polling the flag WRDY (or by waiting for the corresponding interrupt) in PWM_ISR2, else go to Step 13.
11. Write the register that needs to be updated (PWM_SCUPUPD).

12. The update of this register will occur at the next PWM period of the synchronous channels when the Update Period is elapsed. Go to Step 10. for new values.
13. Wait for the DMA status flag indicating that the buffer transfer is complete. If the transfer has ended, define a new DMA transfer for new duty-cycle values. Go to Step 5.

Figure 53-21. Method 3 (UPDM = 2 and PTRM = 0)

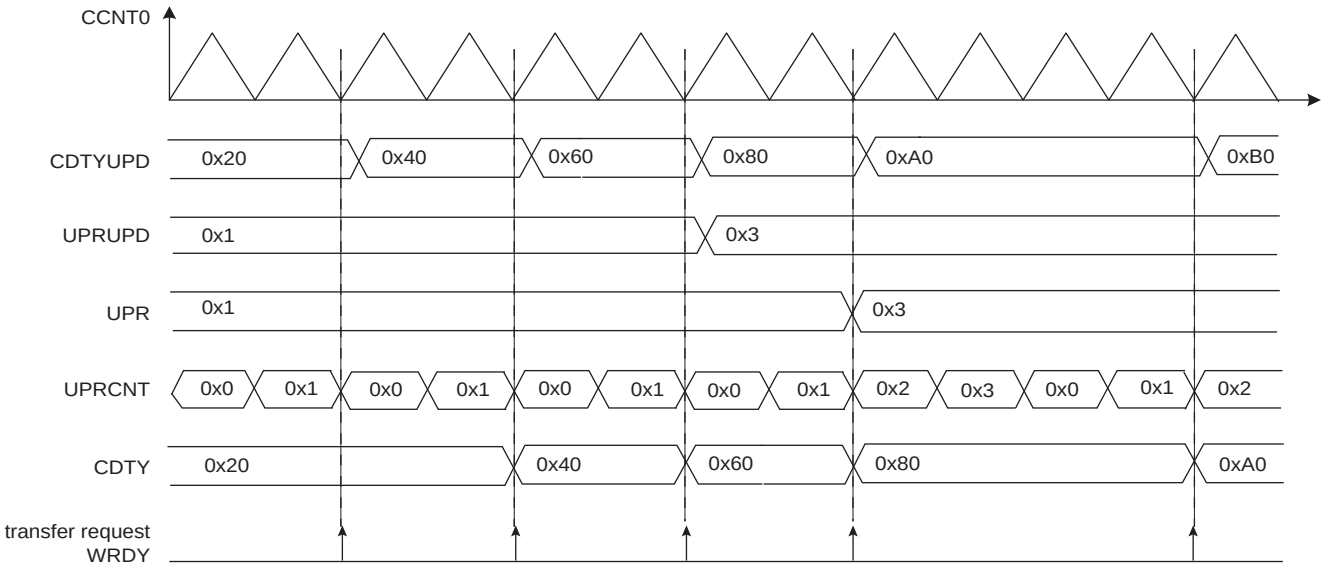
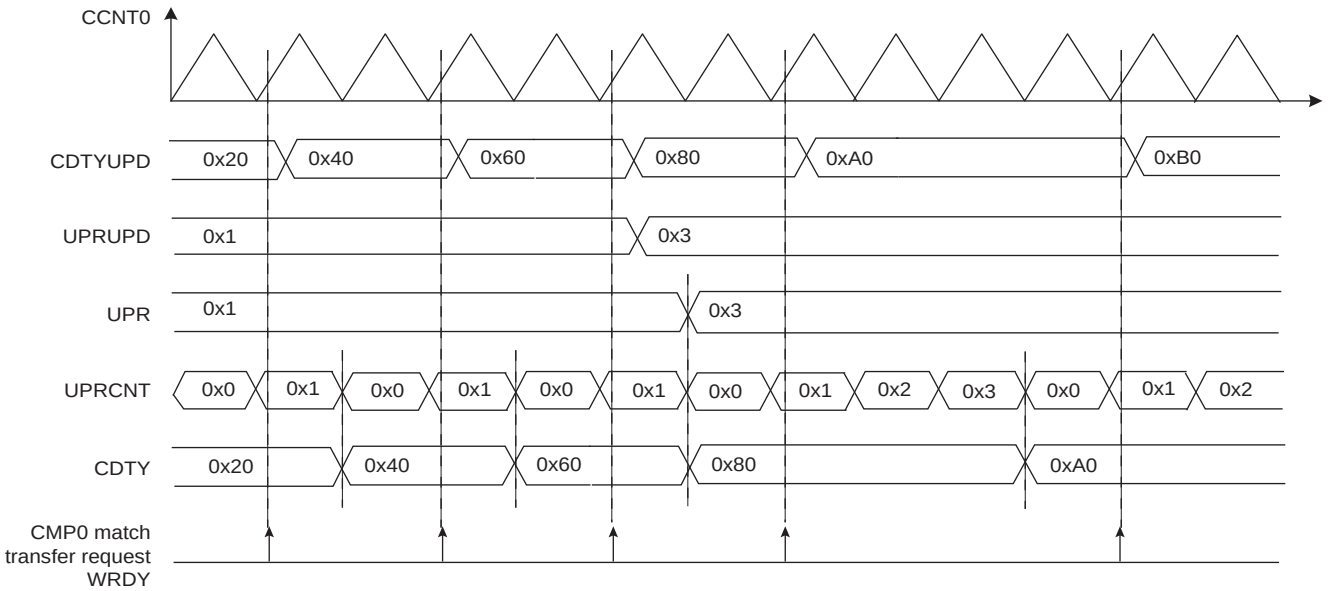


Figure 53-22. Method 3 (UPDM = 2 and PTRM = 1 and PTRCS = 0)



53.6.2.10 Update Time for Double-Buffering Registers

All channels integrate a double-buffering system in order to prevent an unexpected output waveform while modifying the period, the spread spectrum value, the polarity, the duty-cycle, the dead-times, the output override, and the synchronous channels update period.

This double-buffering system comprises the following update registers:

- PWM Sync Channels Update Period Update Register
- PWM Output Selection Set Update Register
- PWM Output Selection Clear Update Register
- PWM Spread Spectrum Update Register
- PWM Channel Duty Cycle Update Register
- PWM Channel Period Update Register
- PWM Channel Dead Time Update Register
- PWM Channel Mode Update Register

When one of these update registers is written to, the write is stored, but the values are updated only at the next PWM period border. In Left-aligned mode (CALG = 0), the update occurs when the channel counter reaches the period value CPRD. In Center-aligned mode, the update occurs when the channel counter value is decremented and reaches the 0 value.

In Center-aligned mode, it is possible to trigger the update of the polarity and the duty-cycle at the next half period border. This mode concerns the following update registers:

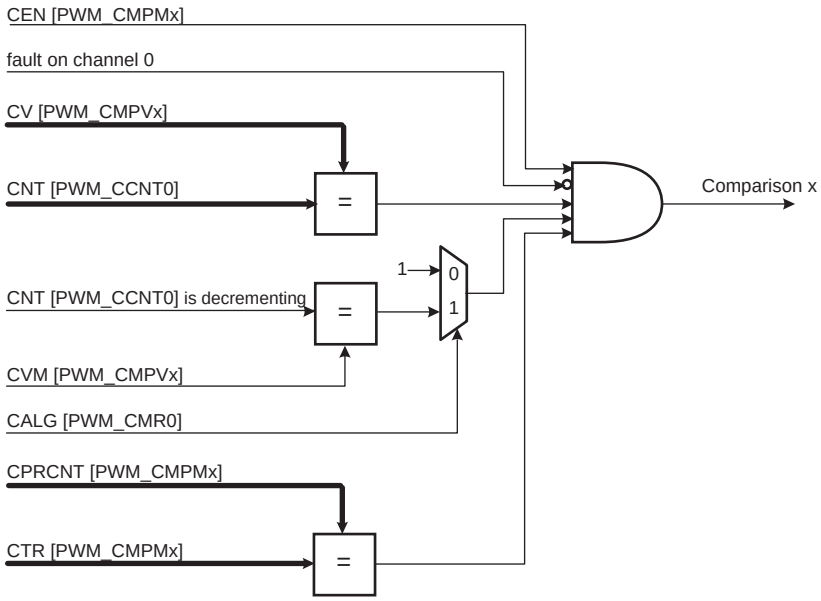
- PWM Channel Duty Cycle Update Register
- PWM Channel Mode Update Register

The update occurs at the first half period following the write of the update register (either when the channel counter value is incrementing and reaches the period value CPRD, or when the channel counter value is decrementing and reaches the 0 value). To activate this mode, the user must write a one to the bit UPDS in the PWM Channel Mode Register.

53.6.3 PWM Comparison Units

The PWM provides 1 independent comparison units able to compare a programmed value with the current value of the channel 0 counter (which is the channel counter of all synchronous channels, Section 53.6.2.9 “Synchronous Channels”). These comparisons are intended to generate pulses on the event lines (used to synchronize ADC, see Section 53.6.4 “PWM Event Lines”), to generate software interrupts and to trigger DMA Controller transfer requests for the synchronous channels (see “Method 3: Automatic write of duty-cycle values and automatic trigger of the update”).

Figure 53-23. Comparison Unit Block Diagram



The comparison x matches when it is enabled by the bit CEN in the PWM Comparison x Mode Register (PWM_CMPMx for the comparison x) and when the counter of the channel 0 reaches the comparison value defined by the field CV in PWM Comparison x Value Register (PWM_CMPVx for the comparison x). If the counter of the channel 0 is center-aligned (CALG = 1 in PWM Channel Mode Register), the bit CVM in PWM_CMPVx defines if the comparison is made when the counter is counting up or counting down (in Left-alignment mode CALG = 0, this bit is useless).

If a fault is active on the channel 0, the comparison is disabled and cannot match (see Section 53.6.2.7 “Fault Protection”).

The user can define the periodicity of the comparison x by the fields CTR and CPR in PWM_CMPMx. The comparison is performed periodically once every CPR+1 periods of the counter of the channel 0, when the value of the comparison period counter CPRCNT in PWM_CMPMx reaches the value defined by CTR. CPR is the maximum value of the comparison period counter CPRCNT. If CPR = CTR = 0, the comparison is performed at each period of the counter of the channel 0.

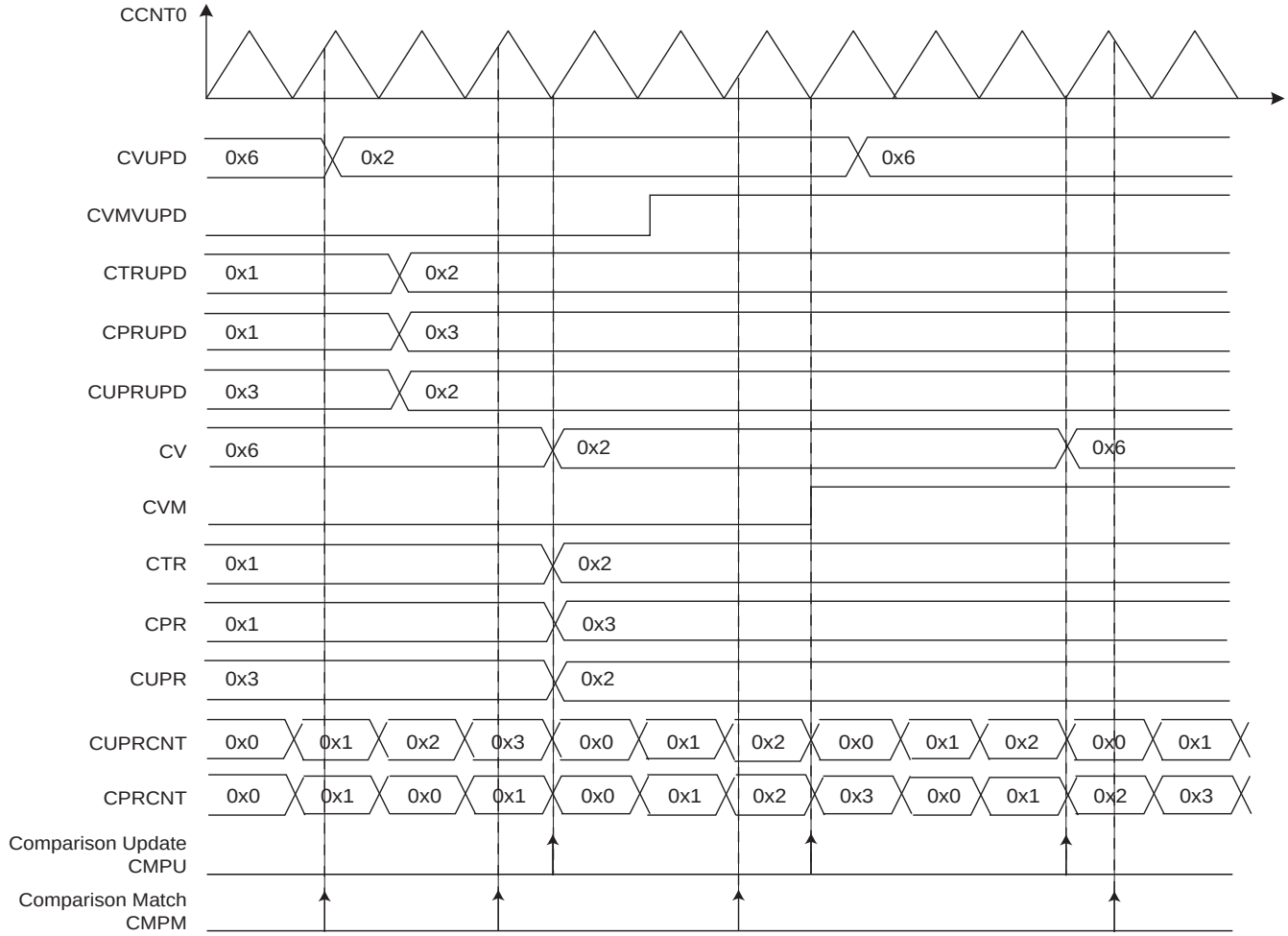
The comparison x configuration can be modified while the channel 0 is enabled by using the PWM Comparison x Mode Update Register (PWM_CMPMUPDx registers for the comparison x). In the same way, the comparison x value can be modified while the channel 0 is enabled by using the PWM Comparison x Value Update Register (PWM_CMPVUPDx registers for the comparison x).

The update of the comparison x configuration and the comparison x value is triggered periodically after the comparison x update period. It is defined by the field CUPR in PWM_CMPMx. The comparison unit has an update period counter independent from the period counter to trigger this update. When the value of the comparison update period counter CUPRCNT (in PWM_CMPMx) reaches the value defined by CUPR, the update is triggered. The comparison x update period CUPR itself can be updated while the channel 0 is enabled by using the PWM_CMPMUPDx register.

CAUTION: The write of PWM_CMPVUPDx must be followed by a write of PWM_CMPMUPDx.

The comparison match and the comparison update can be source of an interrupt, but only if it is enabled and not masked. These interrupts can be enabled by the PWM Interrupt Enable Register 2 and disabled by the PWM Interrupt Disable Register 2. The comparison match interrupt and the comparison update interrupt are reset by reading the PWM Interrupt Status Register 2.

Figure 53-24. Comparison Waveform



53.6.4 PWM Event Lines

The PWM provides 2 independent event lines intended to trigger actions in other peripherals (e.g., for the Analog-to-Digital Converter (ADC)).

A pulse (one cycle of the peripheral clock) is generated on an event line, when at least one of the selected comparisons is matching. The comparisons can be selected or unselected independently by the CSEL bits in the PWM Event Line x Register (PWM_ELMRx for the Event Line x).

An example of event generation is provided in Figure 53-26.

Figure 53-25. Event Line Block Diagram

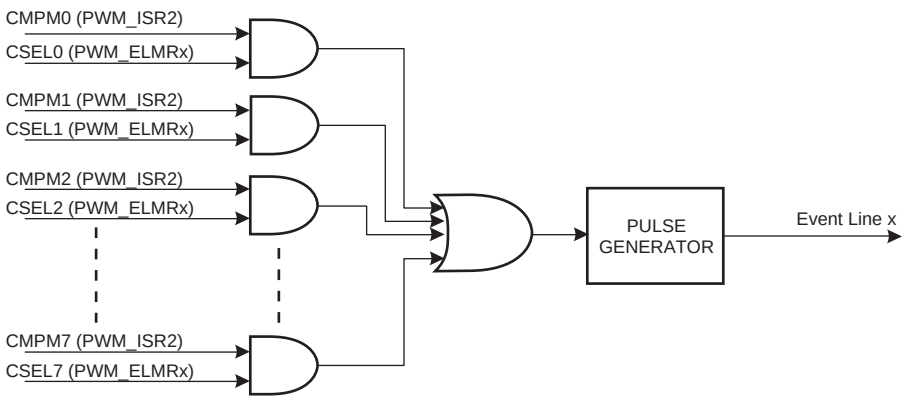
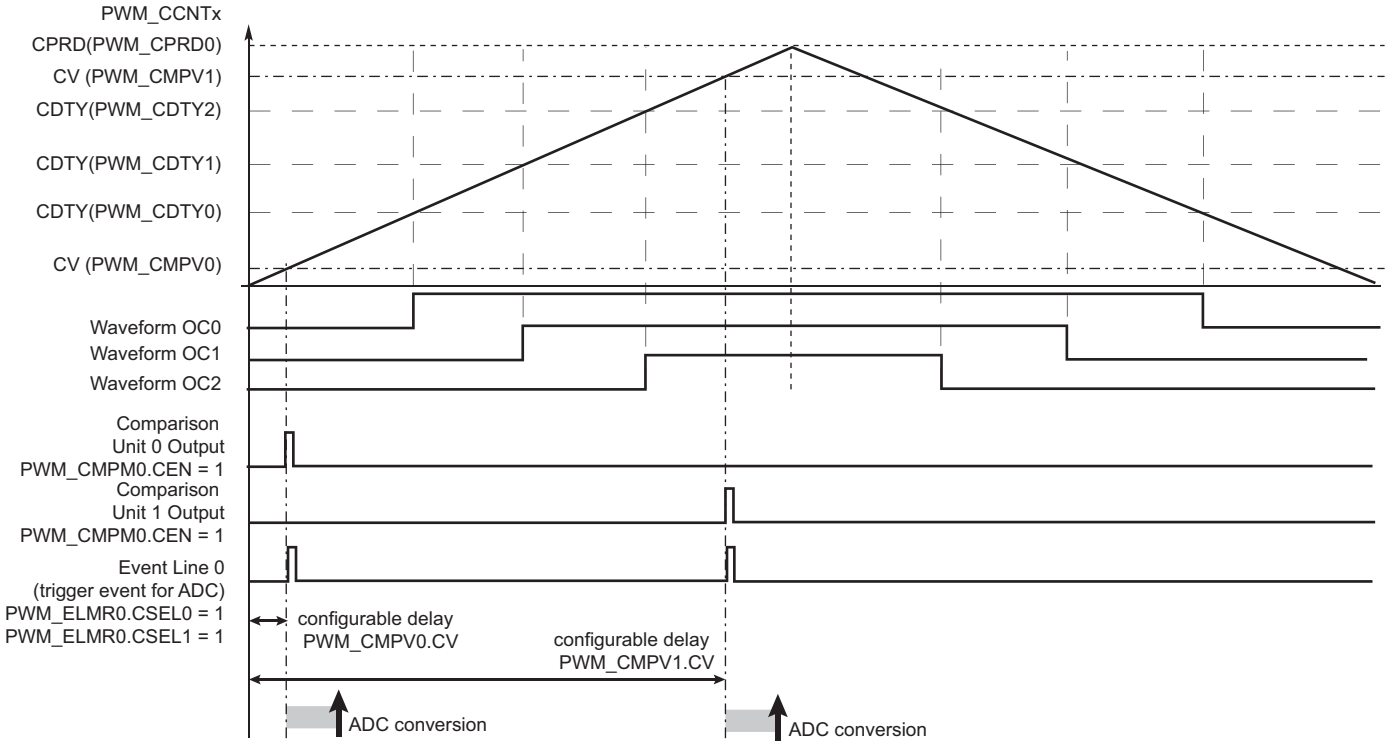


Figure 53-26. Event Line Generation Waveform (Example)



53.6.5 PWM External Trigger Mode

The PWM channels 1 and 2 can be configured to use an external trigger for generating specific PWM signals. The external trigger source can be selected through the bit TRGSRC of the PWM External Trigger Register (see Table 53-7).

Table 53-7. External Event Source Selection

Channel	Trigger Source Selection	Trigger Source
1	PWM_ETRG1.TRGSRC = 0	From PWMEXTRG1 input
	PWM_ETRG1.TRGSRC = 1	From Analog Comparator Controller
2	PWM_ETRG2.TRGSRC = 0	From PWMEXTRG2 input
	PWM_ETRG2.TRGSRC = 1	From Analog Comparator Controller

Each external trigger source can be filtered by writing a one to the TRGFILT bit in the corresponding PWM External Trigger Register (PWM_ETRGx).

Each time an external trigger event is detected, the corresponding PWM channel counter value is stored in the MAXCNT field of the PWM_ETRGx register if it is greater than the previously stored value. Reading the PWM_ETRGx register will clear the MAXCNT value.

Three different modes are available for channels 1 and 2 depending on the value of the TRGMODE field of the PWM_ETRGx register:

- TRGMODE = 1: External PWM Reset Mode (see Section 53.6.5.1 “External PWM Reset Mode”)
- TRGMODE = 2: External PWM Start Mode (see Section 53.6.5.2 “External PWM Start Mode”)
- TRGMODE = 3: Cycle-By-Cycle Duty Mode (see Section 53.6.5.3 “Cycle-By-Cycle Duty Mode”)

This feature is disabled when TRGMODE = 0.

This feature should only be enabled if the corresponding channel is left-aligned (CALG = 0 in PWM Channel Mode Register of channel 1 or 2) and not managed as a synchronous channel (SYNCx = 0 in PWM Sync Channels Mode Register where x = 1 or 2). Programming the channel to be center-aligned or synchronous while TRGMODE is not 0 could lead to unexpected behavior.

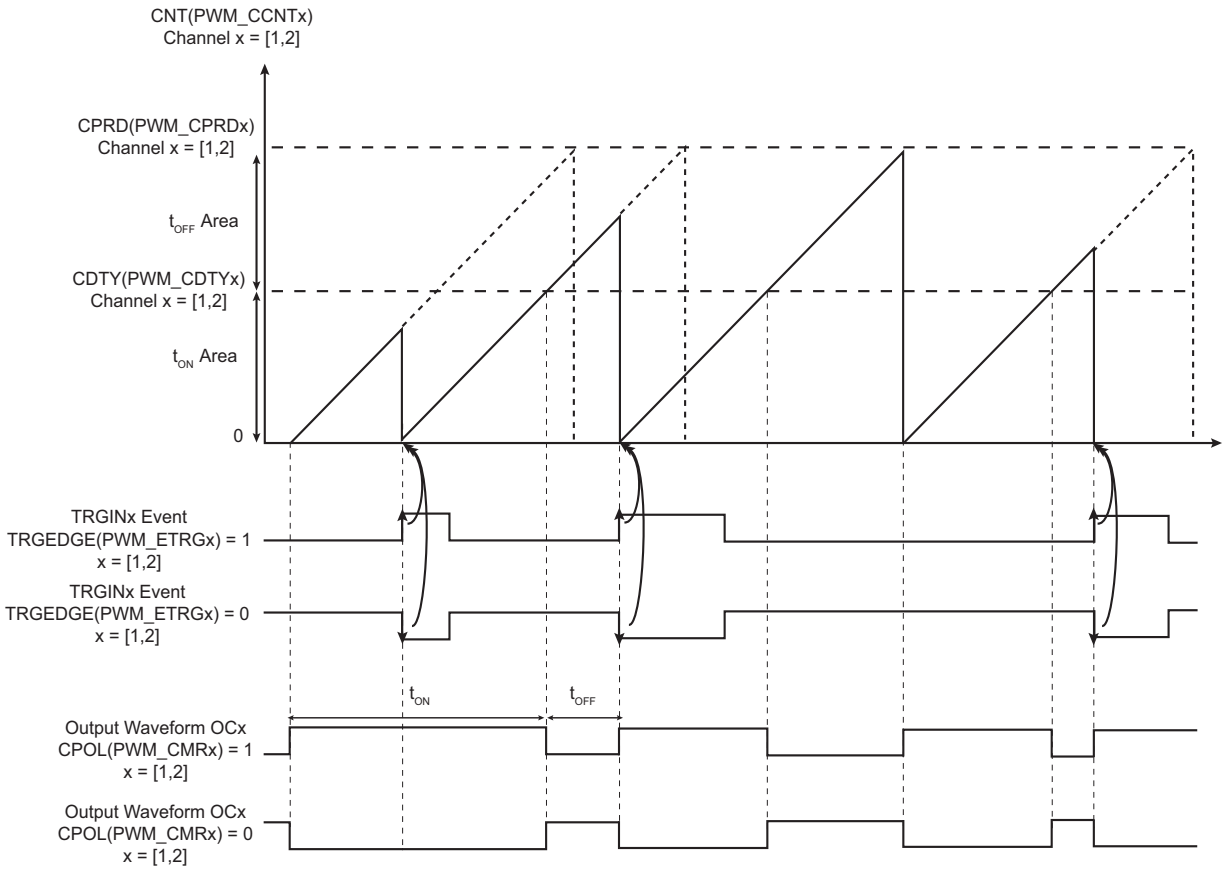
53.6.5.1 External PWM Reset Mode

External PWM Reset mode is selected by programming TRGMODE = 1 in the PWM_ETRGx register.

In this mode, when an edge is detected on the PWMEXTRGx input, the internal PWM counter is cleared and a new PWM cycle is restarted. The edge polarity can be selected by programming the TRGEDGE bit in the PWM_ETRGx register. If no trigger event is detected when the internal channel counter has reached the CPRD value in the PWM Channel Period Register, the internal counter is cleared and a new PWM cycle starts.

Note that this mode does not guarantee a constant t_{ON} or t_{OFF} time.

Figure 53-27. External PWM Reset Mode



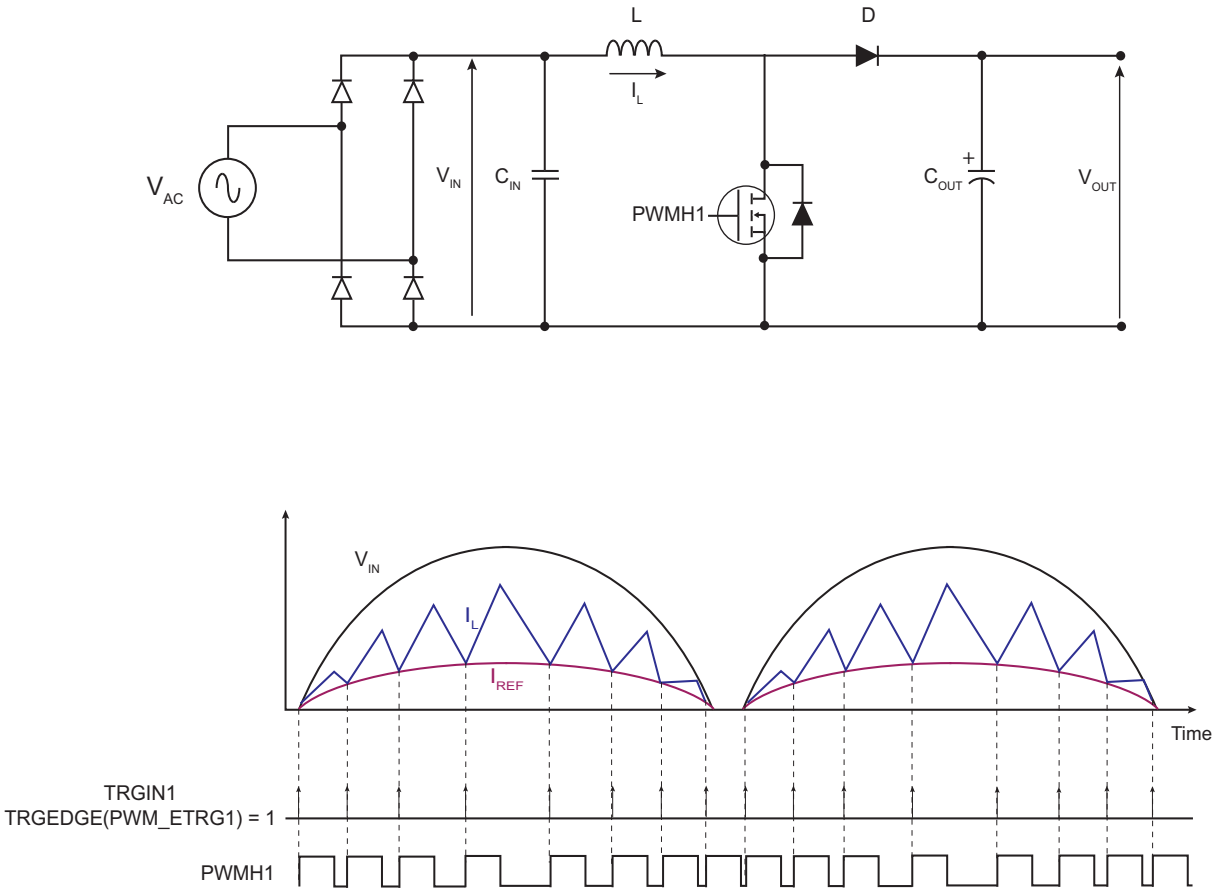
Application Example

The external PWM Reset mode can be used in power factor correction applications.

In the example below, the external trigger input is the PWMEXTRG1 (therefore the PWM channel used for regulation is the channel 1). The PWM channel 1 period (CPRD in the PWM Channel Period Register of the channel 1) must be programmed so that the TRGIN1 event always triggers before the PWM channel 1 period elapses.

In Figure 53-28, an external circuit (not shown) is required to sense the inductor current I_L. The internal PWM counter of the channel 1 is cleared when the inductor current falls below a specific threshold (I_{REF}). This starts a new PWM period and increases the inductor current.

Figure 53-28. External PWM Reset Mode: Power Factor Correction Application



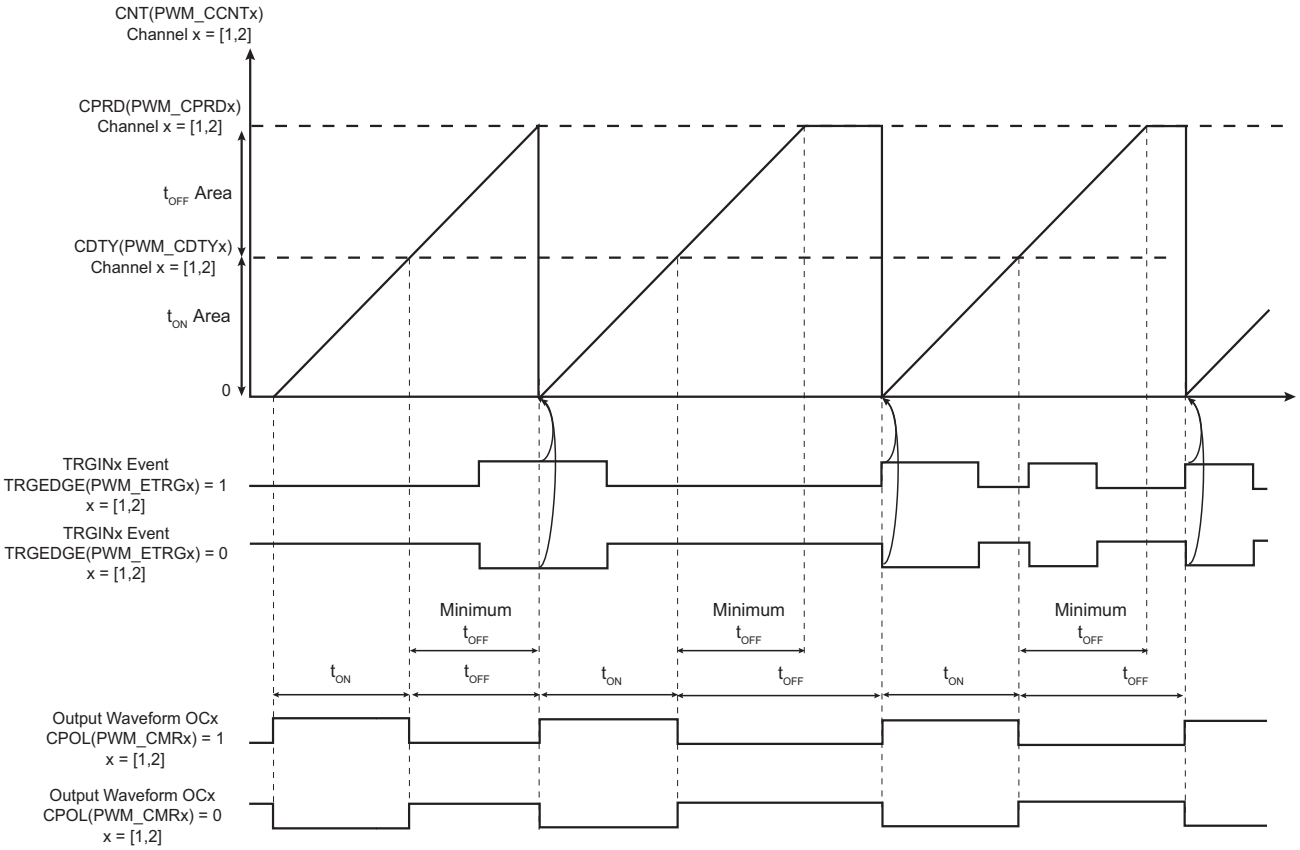
53.6.5.2 External PWM Start Mode

External PWM Start mode is selected by programming TRGMODE = 2 in the PWM_ETRGx register.

In this mode, the internal PWM counter can only be reset once it has reached the CPRD value in the PWM Channel Period Register and when the correct level is detected on the corresponding external trigger input. Both conditions have to be met to start a new PWM period. The active detection level is defined by the bit TRGEDGE of the PWM_ETRGx register.

Note that this mode guarantees a constant t_{ON} time and a minimum t_{OFF} time.

Figure 53-29. External PWM Start Mode



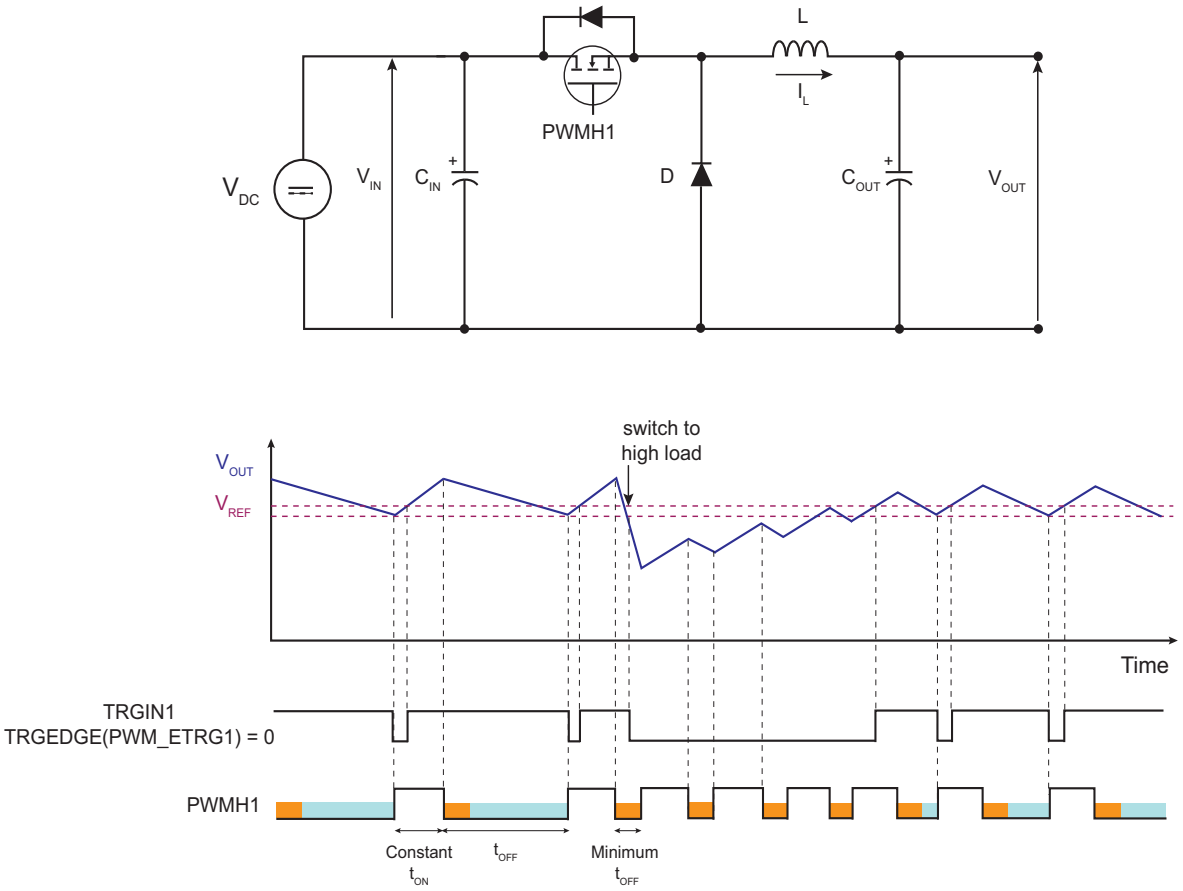
Application Example

The external PWM Start mode generates a modulated frequency PWM signal with a constant active level duration (t_{ON}) and a minimum inactive level duration (minimum t_{OFF}).

The t_{ON} time is defined by the CDTY value in the PWM Channel Duty Cycle Register. The minimum t_{OFF} time is defined by CDTY - CPRD (PWM Channel Period Register). This mode can be useful in Buck DC/DC Converter applications.

When the output voltage V_{OUT} is above a specific threshold (Vref), the PWM inactive level is maintained as long as V_{OUT} remains above this threshold. If V_{OUT} is below this specific threshold, this mode guarantees a minimum t_{OFF} time required for MOSFET driving (see Figure 53-30).

Figure 53-30. External PWM Start Mode: Buck DC/DC Converter



53.6.5.3 Cycle-By-Cycle Duty Mode

Description

Cycle-by-cycle duty mode is selected by programming TRGMODE = 3 in PWM_ETRGx.

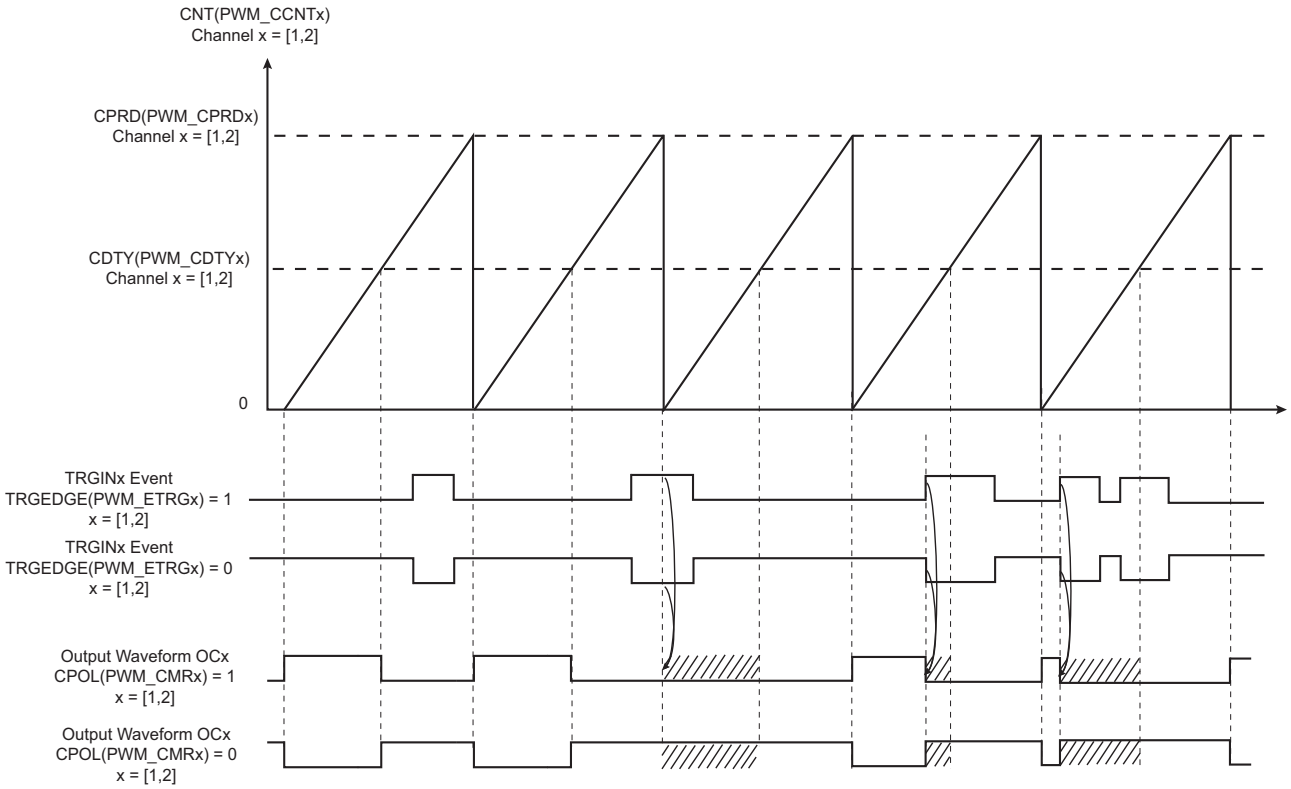
In this mode, the PWM frequency is constant and is defined by the CPRD value in the PWM Channel Period Register.

An external trigger event has no effect on the PWM output if it occurs while the internal PWM counter value is above the CDTY value of the PWM Channel Duty Cycle Register.

If the internal PWM counter value is below the value of CDTY of the PWM Channel Duty Cycle Register, an external trigger event makes the PWM output inactive.

The external trigger event can be detected on rising or falling edge according to the TRGEDGE bit in PWM_ETRGx.

Figure 53-31. Cycle-By-Cycle Duty Mode

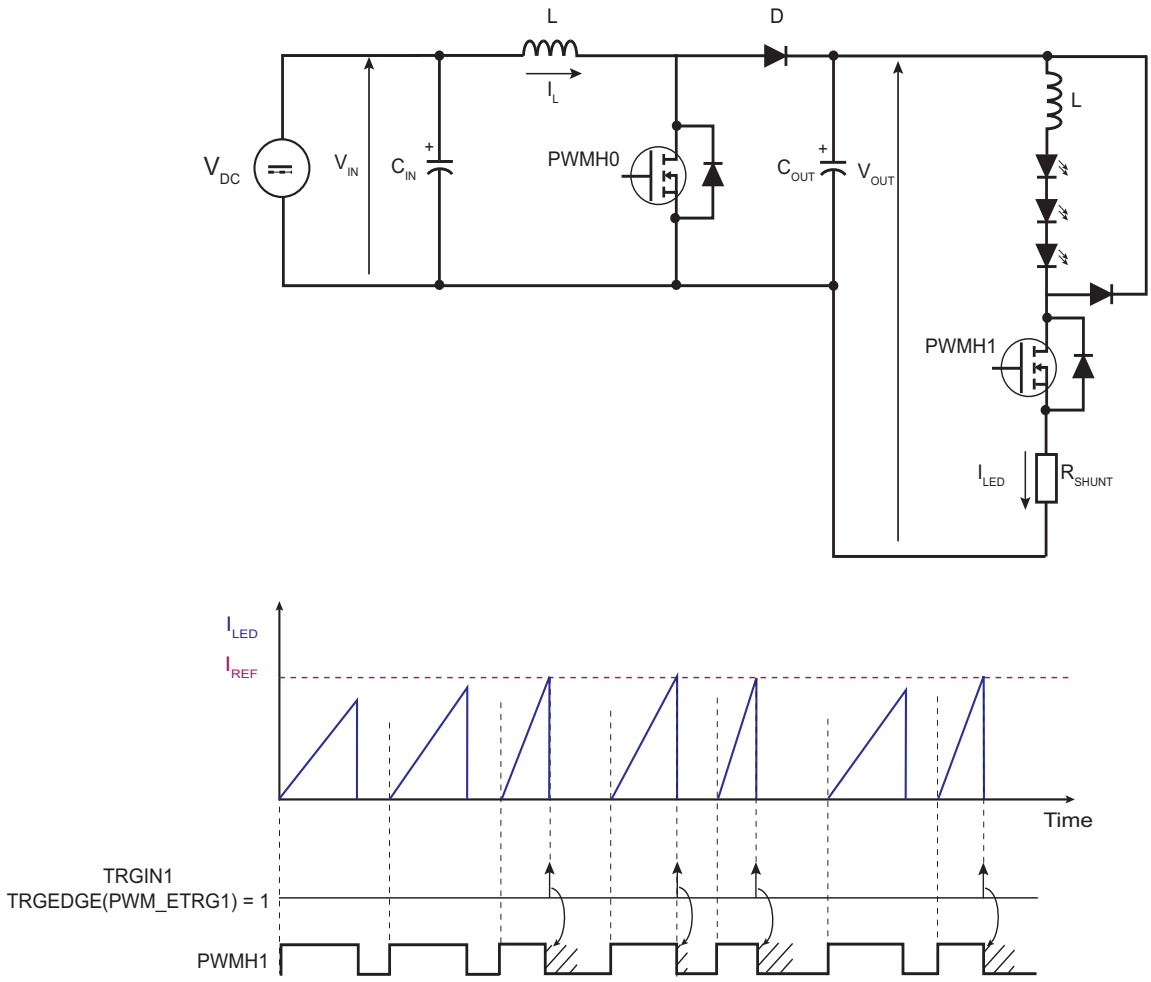


Application Example

Figure 53-32 illustrates an application example of the Cycle-by-cycle Duty mode.

In an LED string control circuit, Cycle-by-cycle Duty mode can be used to automatically limit the current in the LED string.

Figure 53-32. Cycle-By-Cycle Duty Mode: LED String Control



53.6.5.4 Leading-Edge Blanking (LEB)

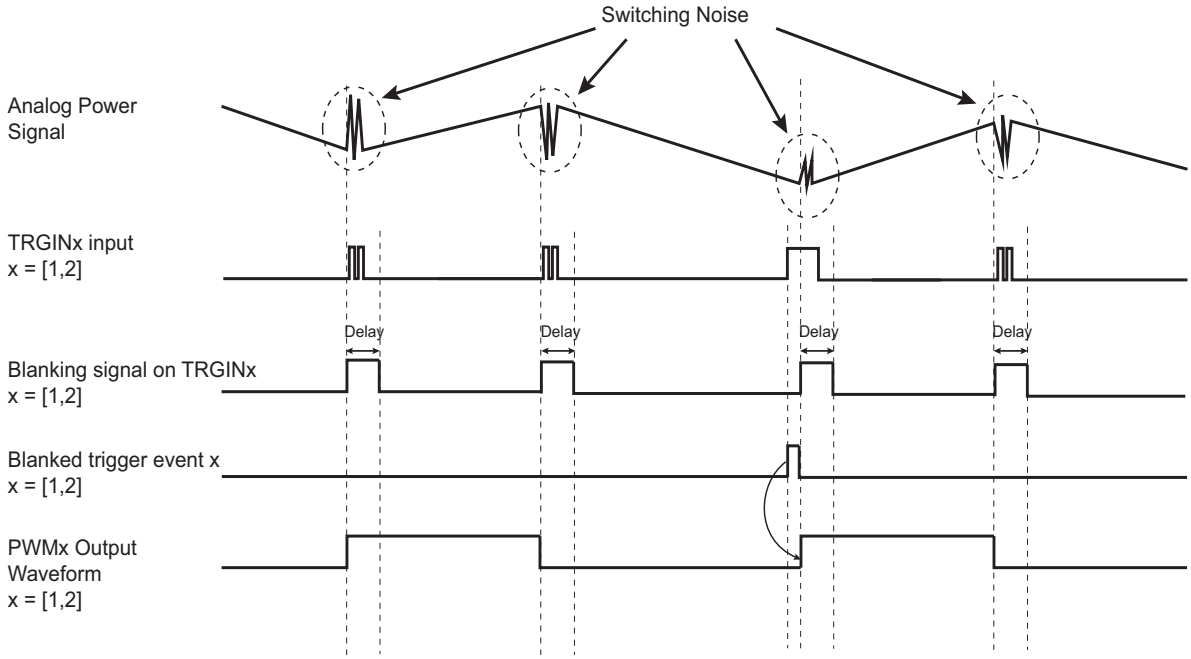
PWM channels 1 and 2 support leading-edge blanking. Leading-edge blanking masks the external trigger input when a transient occurs on the corresponding PWM output. It masks potential spurious external events due to power transistor switching.

The blanking delay on each external trigger input is configured by programming the LEBDELAYx in the PWM Leading-Edge Blanking Register.

The LEB can be enabled on both the rising and the falling edges for the PWMH and PWML outputs through the bits PWMLFEN, PWMLREN, PWMHFEN, PWMHREN.

Any event on the PWMEXTRGx input which occurs during the blanking time is ignored.

Figure 53-33. Leading-Edge Blanking



53.6.6 PWM Controller Operations

53.6.6.1 Initialization

Before enabling the channels, they must be configured by the software application as described below:

- Unlock User Interface by writing the WPCMD field in PWM_WPCR.
- Configuration of the clock generator (DIVA, PREA, DIVB, PREB in the PWM_CLK register if required).
- Selection of the clock for each channel (CPRE field in PWM_CMRx)
- Configuration of the waveform alignment for each channel (CALG field in PWM_CMRx)
- Selection of the counter event selection (if CALG = 1) for each channel (CES field in PWM_CMRx)
- Configuration of the output waveform polarity for each channel (CPOL bit in PWM_CMRx)
- Configuration of the period for each channel (CPRD in the PWM_CPRDx register). Writing in PWM_CPRDx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CPRDUPDx register to update PWM_CPRDx as explained below.
- Configuration of the duty-cycle for each channel (CDTY in the PWM_CDTYx register). Writing in PWM_CDTYx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_CDTYUPDx register to update PWM_CDTYx as explained below.
- Configuration of the dead-time generator for each channel (DTH and DTL in PWM_DTx) if enabled (DTE bit in PWM_CMRx). Writing in the PWM_DTx register is possible while the channel is disabled. After validation of the channel, the user must use PWM_DTUPDx register to update PWM_DTx
- Selection of the synchronous channels (SYNCx in the PWM_SCM register)
- Selection of the moment when the WRDY flag and the corresponding DMA Controller transfer request are set (PTRM and PTRCS in the PWM_SCM register)
- Configuration of the Update mode (UPDM in PWM_SCM register)
- Configuration of the update period (UPR in PWM_SCUP register) if needed
- Configuration of the comparisons (PWM_CMPVx and PWM_CMPMx)
- Configuration of the event lines (PWM_ELMRx)
- Configuration of the fault inputs polarity (FPOL in PWM_FMR)
- Configuration of the fault protection (FMOD and FFIL in PWM_FMR, PWM_FPV and PWM_FPE1)
- Enable of the interrupts (writing CHIDx and FCHIDx in PWM_IER1, and writing WRDY, UNRE, CMPMx and CMPUx in PWM_IER2)
- Enable of the PWM channels (writing CHIDx in the PWM_ENA register)

53.6.6.2 Source Clock Selection Criteria

The large number of source clocks can make selection difficult. The relationship between the value in the PWM Channel Period Register (PWM_CPRDx) and the PWM Channel Duty Cycle Register (PWM_CDTYx) helps the user select the appropriate clock. The event number written in the Period Register gives the PWM accuracy. The Duty-Cycle quantum cannot be lower than $1/CPRDx$ value. The higher the value of PWM_CPRDx, the greater the PWM accuracy.

For example, if the user sets 15 (in decimal) in PWM_CPRDx, the user is able to set a value from between 1 up to 14 in PWM_CDTYx. The resulting duty-cycle quantum cannot be lower than 1/15 of the PWM period.

53.6.6.3 Changing the Duty-Cycle, the Period and the Dead-Times

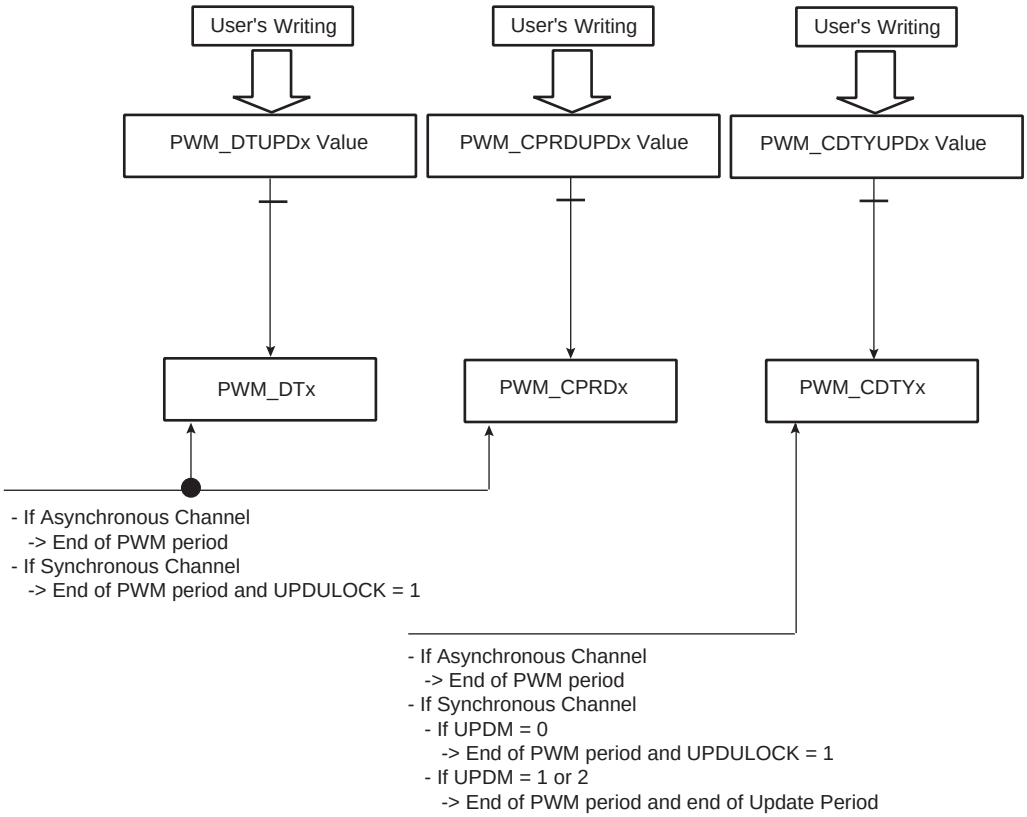
It is possible to modulate the output waveform duty-cycle, period and dead-times.

To prevent unexpected output waveform, the user must use the PWM Channel Duty Cycle Update Register (PWM_CDTYUPDx), the PWM Channel Period Update Register (PWM_CPRDUPDx) and the PWM Channel Dead Time Update Register (PWM_DTUPDx) to change waveform parameters while the channel is still enabled.

- If the channel is an asynchronous channel (SYNCx = 0 in PWM Sync Channels Mode Register (PWM_SCM)), these registers hold the new period, duty-cycle and dead-times values until the end of the current PWM period and update the values for the next period.
- If the channel is a synchronous channel and update method 0 is selected (SYNCx = 1 and UPDM = 0 in PWM_SCM register), these registers hold the new period, duty-cycle and dead-times values until the bit UPDULOCK is written at '1' (in PWM Sync Channels Update Control Register (PWM_SCUC)) and the end of the current PWM period, then update the values for the next period.
- If the channel is a synchronous channel and update method 1 or 2 is selected (SYNCx = 1 and UPDM = 1 or 2 in PWM_SCM register):
  - registers PWM_CPRDUPDx and PWM_DTUPDx hold the new period and dead-times values until the bit UPDULOCK is written at '1' (in PWM_SCUC) and the end of the current PWM period, then update the values for the next period.
  - register PWM_CDTYUPDx holds the new duty-cycle value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM Sync Channels Update Period Register (PWM_SCUP)) and the end of the current PWM period, then updates the value for the next period.

Note: If the update registers PWM_CDTYUPDx, PWM_CPRDUPDx and PWM_DTUPDx are written several times between two updates, only the last written value is taken into account.

Figure 53-34. Synchronized Period, Duty-Cycle and Dead-Time Update



53.6.6.4 Changing the Update Period of Synchronous Channels

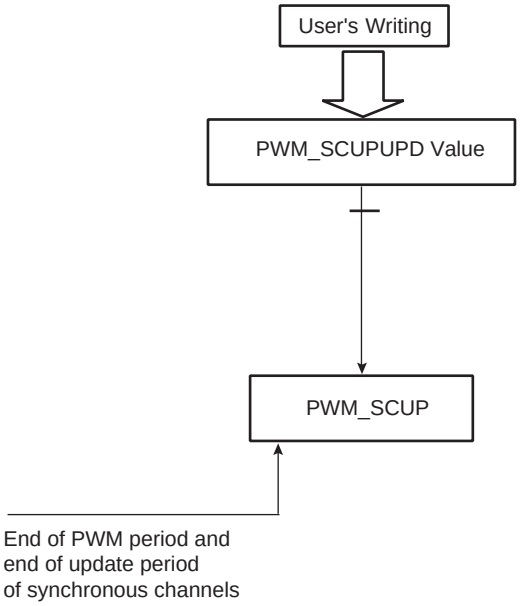
It is possible to change the update period of synchronous channels while they are enabled. See "Method 2: Manual write of duty-cycle values and automatic trigger of the update" and "Method 3: Automatic write of duty-cycle values and automatic trigger of the update" .

To prevent an unexpected update of the synchronous channels registers, the user must use the PWM Sync Channels Update Period Update Register (PWM_SCUPUPD) to change the update period of synchronous channels while they are still enabled. This register holds the new value until the end of the update period of synchronous channels (when UPRCNT is equal to UPR in PWM_SCUP) and the end of the current PWM period, then updates the value for the next period.

Note: If the update register PWM_SCUPUPD is written several times between two updates, only the last written value is taken into account.

Note: Changing the update period does make sense only if there is one or more synchronous channels and if the update method 1 or 2 is selected (UPDM = 1 or 2 in PWM Sync Channels Mode Register).

Figure 53-35. Synchronized Update of Update Period Value of Synchronous Channels



53.6.6.5 Changing the Comparison Value and the Comparison Configuration

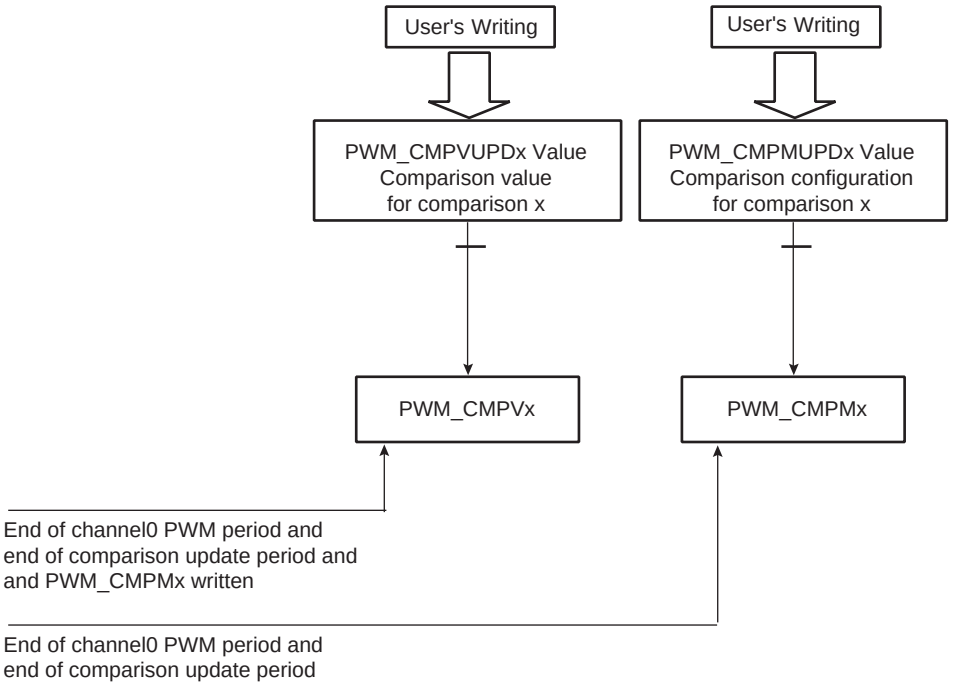
It is possible to change the comparison values and the comparison configurations while the channel 0 is enabled (see Section 53.6.3 "PWM Comparison Units").

To prevent unexpected comparison match, the user must use the PWM Comparison x Value Update Register (PWM_CMPVUPDx) and the PWM Comparison x Mode Update Register (PWM_CMPMUPDx) to change, respectively, the comparison values and the comparison configurations while the channel 0 is still enabled. These registers hold the new values until the end of the comparison update period (when CUPRCNT is equal to CUPR in PWM Comparison x Mode Register (PWM_CMPMx) and the end of the current PWM period, then update the values for the next period.

CAUTION: The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

Note: If the update registers PWM_CMPVUPDx and PWM_CMPMUPDx are written several times between two updates, only the last written value are taken into account.

Figure 53-36. Synchronized Update of Comparison Values and Configurations



53.6.6.6 Interrupt Sources

Depending on the interrupt mask in PWM_IMR1 and PWM_IMR2, an interrupt can be generated at the end of the corresponding channel period (CHIDx in the PWM Interrupt Status Register 1 (PWM_ISR1)), after a fault event (FCHIDx in PWM_ISR1), after a comparison match (CMPMx in PWM_ISR2), after a comparison update (CMPUx in PWM_ISR2) or according to the Transfer mode of the synchronous channels (WRDY and UNRE in PWM_ISR2).

If the interrupt is generated by the flags CHIDx or FCHIDx, the interrupt remains active until a read operation in PWM_ISR1 occurs.

If the interrupt is generated by the flags WRDY or UNRE or CMPMx or CMPUx, the interrupt remains active until a read operation in PWM_ISR2 occurs.

A channel interrupt is enabled by setting the corresponding bit in PWM_IER1 and PWM_IER2. A channel interrupt is disabled by setting the corresponding bit in PWM_IDR1 and PWM_IDR2.

53.6.7 Register Write Protection

To prevent any single software error that may corrupt PWM behavior, the registers listed below can be write-protected by writing the field WPCMD in the PWM Write Protection Control Register (PWM_WPCR). They are divided into six groups:

- Register group 0:
  - PWM Clock Register
- Register group 1:
  - PWM Disable Register
- Register group 2:
  - PWM Sync Channels Mode Register
  - PWM Channel Mode Register
  - PWM Stepper Motor Mode Register
  - PWM Fault Protection Value Register 2
  - PWM Leading-Edge Blanking Register
  - PWM Channel Mode Update Register
- Register group 3:
  - PWM Spread Spectrum Register
  - PWM Spread Spectrum Update Register
  - PWM Channel Period Register
  - PWM Channel Period Update Register
- Register group 4:
  - PWM Channel Dead Time Register
  - PWM Channel Dead Time Update Register
- Register group 5:
  - PWM Fault Mode Register
  - PWM Fault Protection Value Register 1

There are two types of write protection:

- SW write protection—can be enabled or disabled by software
- HW write protection—can be enabled by software but only disabled by a hardware reset of the PWM controller

Both types of write protection can be applied independently to a particular register group by means of the WPCMD and WPRGx fields in PWM_WPCR. If at least one type of write protection is active, the register group is write-protected. The value of field WPCMD defines the action to be performed:

- 0: Disables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 1: Enables SW write protection of the register groups of which the bit WPRGx is at ‘1’
- 2: Enables HW write protection of the register groups of which the bit WPRGx is at ‘1’

At any time, the user can determine whether SW or HW write protection is active in a particular register group by the fields WPSWS and WPHWS in the PWM Write Protection Status Register (PWM_WPSR).

If a write access to a write-protected register is detected, the WPVS flag in PWM_WPSR is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS and WPVSRC fields are automatically cleared after reading PWM_WPSR.

53.7 Pulse Width Modulation Controller (PWM) User Interface

Table 53-8. Register Mapping

Offset	Register	Name	Access	Reset
0x00	PWM Clock Register	PWM_CLK	Read/Write	0x0
0x04	PWM Enable Register	PWM_ENA	Write-only	–
0x08	PWM Disable Register	PWM_DIS	Write-only	–
0x0C	PWM Status Register	PWM_SR	Read-only	0x0
0x10	PWM Interrupt Enable Register 1	PWM_IER1	Write-only	–
0x14	PWM Interrupt Disable Register 1	PWM_IDR1	Write-only	–
0x18	PWM Interrupt Mask Register 1	PWM_IMR1	Read-only	0x0
0x1C	PWM Interrupt Status Register 1	PWM_ISR1	Read-only	0x0
0x20	PWM Sync Channels Mode Register	PWM_SCM	Read/Write	0x0
0x24	PWM DMA Register	PWM_DMAR	Write-only	–
0x28	PWM Sync Channels Update Control Register	PWM_SCUC	Read/Write	0x0
0x2C	PWM Sync Channels Update Period Register	PWM_SCUP	Read/Write	0x0
0x30	PWM Sync Channels Update Period Update Register	PWM_SCUPUPD	Write-only	–
0x34	PWM Interrupt Enable Register 2	PWM_IER2	Write-only	–
0x38	PWM Interrupt Disable Register 2	PWM_IDR2	Write-only	–
0x3C	PWM Interrupt Mask Register 2	PWM_IMR2	Read-only	0x0
0x40	PWM Interrupt Status Register 2	PWM_ISR2	Read-only	0x0
0x44	PWM Output Override Value Register	PWM_OOV	Read/Write	0x0
0x48	PWM Output Selection Register	PWM_OS	Read/Write	0x0
0x4C	PWM Output Selection Set Register	PWM_OSS	Write-only	–
0x50	PWM Output Selection Clear Register	PWM_OSC	Write-only	–
0x54	PWM Output Selection Set Update Register	PWM_OSSUPD	Write-only	–
0x58	PWM Output Selection Clear Update Register	PWM_OSCUPD	Write-only	–
0x5C	PWM Fault Mode Register	PWM_FMR	Read/Write	0x0
0x60	PWM Fault Status Register	PWM_FSR	Read-only	0x0
0x64	PWM Fault Clear Register	PWM_FCR	Write-only	–
0x68	PWM Fault Protection Value Register 1	PWM_FPV1	Read/Write	0x0
0x6C	PWM Fault Protection Enable Register	PWM_FPE	Read/Write	0x0
0x70–0x78	Reserved	–	–	–
0x7C	PWM Event Line 0 Mode Register	PWM_ELMR0	Read/Write	0x0
0x80	PWM Event Line 1 Mode Register	PWM_ELMR1	Read/Write	0x0
0x84–0x9C	Reserved	–	–	–
0xA0	PWM Spread Spectrum Register	PWM_SSPR	Read/Write	0x0
0xA4	PWM Spread Spectrum Update Register	PWM_SSPUP	Write-only	–
0xA8–0xAC	Reserved	–	–	–

Table 53-8. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0xB0	PWM Stepper Motor Mode Register	PWM_SMMR	Read/Write	0x0
0xC0	PWM Fault Protection Value 2 Register	PWM_FPV2	Read/Write	0x003F_003F
0xC4–0xE0	Reserved	–	–	–
0xE4	PWM Write Protection Control Register	PWM_WPCR	Write-only	–
0xE8	PWM Write Protection Status Register	PWM_WPSR	Read-only	0x0
0xEC–0xFC	Reserved	–	–	–
0x100–0x12C	Reserved	–	–	–
0x130	PWM Comparison 0 Value Register	PWM_CMPV0	Read/Write	0x0
0x134	PWM Comparison 0 Value Update Register	PWM_CMPVUPD0	Write-only	–
0x138	PWM Comparison 0 Mode Register	PWM_CMPM0	Read/Write	0x0
0x13C	PWM Comparison 0 Mode Update Register	PWM_CMPMUPD0	Write-only	–
0x140	PWM Comparison 1 Value Register	PWM_CMPV1	Read/Write	0x0
0x144	PWM Comparison 1 Value Update Register	PWM_CMPVUPD1	Write-only	–
0x148	PWM Comparison 1 Mode Register	PWM_CMPM1	Read/Write	0x0
0x14C	PWM Comparison 1 Mode Update Register	PWM_CMPMUPD1	Write-only	–
0x150	PWM Comparison 2 Value Register	PWM_CMPV2	Read/Write	0x0
0x154	PWM Comparison 2 Value Update Register	PWM_CMPVUPD2	Write-only	–
0x158	PWM Comparison 2 Mode Register	PWM_CMPM2	Read/Write	0x0
0x15C	PWM Comparison 2 Mode Update Register	PWM_CMPMUPD2	Write-only	–
0x160	PWM Comparison 3 Value Register	PWM_CMPV3	Read/Write	0x0
0x164	PWM Comparison 3 Value Update Register	PWM_CMPVUPD3	Write-only	–
0x168	PWM Comparison 3 Mode Register	PWM_CMPM3	Read/Write	0x0
0x16C	PWM Comparison 3 Mode Update Register	PWM_CMPMUPD3	Write-only	–
0x170	PWM Comparison 4 Value Register	PWM_CMPV4	Read/Write	0x0
0x174	PWM Comparison 4 Value Update Register	PWM_CMPVUPD4	Write-only	–
0x178	PWM Comparison 4 Mode Register	PWM_CMPM4	Read/Write	0x0
0x17C	PWM Comparison 4 Mode Update Register	PWM_CMPMUPD4	Write-only	–
0x180	PWM Comparison 5 Value Register	PWM_CMPV5	Read/Write	0x0
0x184	PWM Comparison 5 Value Update Register	PWM_CMPVUPD5	Write-only	–
0x188	PWM Comparison 5 Mode Register	PWM_CMPM5	Read/Write	0x0
0x18C	PWM Comparison 5 Mode Update Register	PWM_CMPMUPD5	Write-only	–
0x190	PWM Comparison 6 Value Register	PWM_CMPV6	Read/Write	0x0
0x194	PWM Comparison 6 Value Update Register	PWM_CMPVUPD6	Write-only	–
0x198	PWM Comparison 6 Mode Register	PWM_CMPM6	Read/Write	0x0
0x19C	PWM Comparison 6 Mode Update Register	PWM_CMPMUPD6	Write-only	–
0x1A0	PWM Comparison 7 Value Register	PWM_CMPV7	Read/Write	0x0
0x1A4	PWM Comparison 7 Value Update Register	PWM_CMPVUPD7	Write-only	–

Table 53-8. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x1A8	PWM Comparison 7 Mode Register	PWM_CMPM7	Read/Write	0x0
0x1AC	PWM Comparison 7 Mode Update Register	PWM_CMPMUPD7	Write-only	–
0x1B0–0x1FC	Reserved	–	–	–
0x200 + ch_num * 0x20 + 0x00	PWM Channel Mode Register(1)	PWM_CMR	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x04	PWM Channel Duty Cycle Register(1)	PWM_CDTY	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x08	PWM Channel Duty Cycle Update Register(1)	PWM_CDTYUPD	Write-only	–
0x200 + ch_num * 0x20 + 0x0C	PWM Channel Period Register(1)	PWM_CPRD	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x10	PWM Channel Period Update Register(1)	PWM_CPRDUPD	Write-only	–
0x200 + ch_num * 0x20 + 0x14	PWM Channel Counter Register(1)	PWM_CCNT	Read-only	0x0
0x200 + ch_num * 0x20 + 0x18	PWM Channel Dead Time Register(1)	PWM_DT	Read/Write	0x0
0x200 + ch_num * 0x20 + 0x1C	PWM Channel Dead Time Update Register(1)	PWM_DTUPD	Write-only	–
0x400 + ch_num * 0x20 + 0x00	PWM Channel Mode Update Register(1)	PWM_CMUPD	Write-only	–
0x400 + trg_num * 0x20 + 0x0C	PWM External Trigger Register(2)	PWM_ETRG	Read/Write	0x0
0x400 + trg_num * 0x20 + 0x10	PWM Leading-Edge Blanking Register(2)	PWM_LEBR	Read/Write	0x0

Notes: 1. Some registers are indexed with “ch_num” index ranging from 0 to 3.
2. Some registers are indexed with “trg_num” index ranging from 1 to 2.

53.7.1 PWM Clock Register

Name: PWM_CLK

Address: 0xF802C000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	PREB			
23	22	21	20	19	18	17	16
DIVB							
15	14	13	12	11	10	9	8
–	–	–	–	PREA			
7	6	5	4	3	2	1	0
DIVA							

This register can only be written if bits WPSWS0 and WPHWS0 are cleared in the PWM Write Protection Status Register.

- **DIVA: CLKA Divide Factor**

Value	Name	Description
0	CLKA_POFF	CLKA clock is turned off
1	PREA	CLKA clock is clock selected by PREA
2–255	PREA_DIV	CLKA clock is clock selected by PREA divided by DIVA factor

- **DIVB: CLKB Divide Factor**

Value	Name	Description
0	CLKB_POFF	CLKB clock is turned off
1	PREB	CLKB clock is clock selected by PREB
2–255	PREB_DIV	CLKB clock is clock selected by PREB divided by DIVB factor

- **PREA: CLKA Source Clock Selection**

Value	Name	Description
0	CLK	Peripheral clock
1	CLK_DIV2	Peripheral clock/2
2	CLK_DIV4	Peripheral clock/4
3	CLK_DIV8	Peripheral clock/8
4	CLK_DIV16	Peripheral clock/16
5	CLK_DIV32	Peripheral clock/32
6	CLK_DIV64	Peripheral clock/64
7	CLK_DIV128	Peripheral clock/128
8	CLK_DIV256	Peripheral clock/256
9	CLK_DIV512	Peripheral clock/512
10	CLK_DIV1024	Peripheral clock/1024
Other	–	Reserved

- **PREB: CLKB Source Clock Selection**

Value	Name	Description
0	CLK	Peripheral clock
1	CLK_DIV2	Peripheral clock/2
2	CLK_DIV4	Peripheral clock/4
3	CLK_DIV8	Peripheral clock/8
4	CLK_DIV16	Peripheral clock/16
5	CLK_DIV32	Peripheral clock/32
6	CLK_DIV64	Peripheral clock/64
7	CLK_DIV128	Peripheral clock/128
8	CLK_DIV256	Peripheral clock/256
9	CLK_DIV512	Peripheral clock/512
10	CLK_DIV1024	Peripheral clock/1024
Other	–	Reserved

53.7.2 PWM Enable Register

Name: PWM_ENA

Address: 0xF802C004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Channel ID**

0: No effect.

1: Enable PWM output for channel x.

53.7.3 PWM Disable Register

Name: PWM_DIS

Address: 0xF802C008

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

This register can only be written if bits WPSWS1 and WPHWS1 are cleared in the PWM Write Protection Status Register.

- **CHIDx: Channel ID**

0: No effect.

1: Disable PWM output for channel x.

53.7.4 PWM Status Register

Name: PWM_SR

Address: 0xF802C00C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Channel ID**

0: PWM output for channel x is disabled.

1: PWM output for channel x is enabled.

53.7.5 PWM Interrupt Enable Register 1

Name: PWM_IER1

Address: 0xF802C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x Interrupt Enable**
- **FCHIDx: Fault Protection Trigger on Channel x Interrupt Enable**

53.7.6 PWM Interrupt Disable Register 1

Name: PWM_IDR1

Address: 0xF802C014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x Interrupt Disable**
- **FCHIDx: Fault Protection Trigger on Channel x Interrupt Disable**

53.7.7 PWM Interrupt Mask Register 1

Name: PWM_IMR1

Address: 0xF802C018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x Interrupt Mask**
- **FCHIDx: Fault Protection Trigger on Channel x Interrupt Mask**

53.7.8 PWM Interrupt Status Register 1

Name: PWM_ISR1

Address: 0xF802C01C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	FCHID3	FCHID2	FCHID1	FCHID0

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	CHID3	CHID2	CHID1	CHID0

- **CHIDx: Counter Event on Channel x**

0: No new counter event has occurred since the last read of PWM_ISR1.

1: At least one counter event has occurred since the last read of PWM_ISR1.

- **FCHIDx: Fault Protection Trigger on Channel x**

0: No new trigger of the fault protection since the last read of PWM_ISR1.

1: At least one trigger of the fault protection since the last read of PWM_ISR1.

Note: Reading PWM_ISR1 automatically clears CHIDx and FCHIDx flags.

53.7.9 PWM Sync Channels Mode Register

Name: PWM_SCM

Address: 0xF802C020

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
PTRCS			PTRM	–	–	UPDM	
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	SYNC3	SYNC2	SYNC1	SYNC0

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

- **SYNCx: Synchronous Channel x**

0: Channel x is not a synchronous channel.

1: Channel x is a synchronous channel.

- **UPDM: Synchronous Channels Update Mode**

Value	Name	Description
0	MODE0	Manual write of double buffer registers and manual update of synchronous channels(1)
1	MODE1	Manual write of double buffer registers and automatic update of synchronous channels(2)
2	MODE2	Automatic write of duty-cycle update registers by the DMA Controller and automatic update of synchronous channels(2)

Notes: 1. The update occurs at the beginning of the next PWM period, when the UPDULOCK bit in PWM Sync Channels Update Control Register is set.

2. The update occurs when the Update Period is elapsed.

- **PTRM: DMA Controller Transfer Request Mode**

UPDM	PTRM	WRDY Flag and DMA Controller Transfer Request
0	x	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are never set to ‘1’.
1	x	The WRDY flag in PWM Interrupt Status Register 2 is set to ‘1’ as soon as the update period is elapsed, the DMA Controller transfer request is never set to ‘1’.
2	0	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the update period is elapsed.
	1	The WRDY flag in PWM Interrupt Status Register 2 and the DMA transfer request are set to ‘1’ as soon as the selected comparison matches.

- **PTRCS: DMA Controller Transfer Request Comparison Selection**

Selection of the comparison used to set the flag WRDY and the corresponding DMA Controller transfer request.

53.7.10 PWM DMA Register

Name: PWM_DMAR

Address: 0xF802C024

Access: Write- only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
DMADUTY							

15	14	13	12	11	10	9	8
DMADUTY							

7	6	5	4	3	2	1	0
DMADUTY							

Only the first 16 bits (channel counter size) are significant.

- **DMADUTY: Duty-Cycle Holding Register for DMA Access**

Each write access to PWM_DMAR sequentially updates the CDTY field of PWM_CDTYx with DMADUTY (only for channel configured as synchronous). See “Method 3: Automatic write of duty-cycle values and automatic trigger of the update” .

53.7.11 PWM Sync Channels Update Control Register

Name: PWM_SCUC

Address: 0xF802C028

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	UPDULOCK

- **UPDULOCK: Synchronous Channels Update Unlock**

0: No effect

1: If the UPDM field is set to ‘0’ in PWM Sync Channels Mode Register, writing the UPDULOCK bit to ‘1’ triggers the update of the period value, the duty-cycle and the dead-time values of synchronous channels at the beginning of the next PWM period. If the field UPDM is set to ‘1’ or ‘2’, writing the UPDULOCK bit to ‘1’ triggers only the update of the period value and of the dead-time values of synchronous channels.

This bit is automatically reset when the update is done.

53.7.12 PWM Sync Channels Update Period Register

Name: PWM_SCUP

Address: 0xF802C02C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
UPRCNT				UPR			

- **UPR: Update Period**

Defines the time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

- **UPRCNT: Update Period Counter**

Reports the value of the update period counter.

53.7.13 PWM Sync Channels Update Period Update Register

Name: PWM_SCUPUPD

Address: 0xF802C030

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	UPRUPD			

This register acts as a double buffer for the UPR value. This prevents an unexpected automatic trigger of the update of synchronous channels.

- **UPRUPD: Update Period Update**

Defines the wanted time between each update of the synchronous channels if automatic trigger of the update is activated (UPDM = 1 or UPDM = 2 in PWM Sync Channels Mode Register). This time is equal to UPR+1 periods of the synchronous channels.

53.7.14 PWM Interrupt Enable Register 2

Name: PWM_IER2

Address: 0xF802C034

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
7	6	5	4	3	2	1	0
–	–	–	–	UNRE	–	–	WRDY

- **WRDY: Write Ready for Synchronous Channels Update Interrupt Enable**
- **UNRE: Synchronous Channels Update Underrun Error Interrupt Enable**
- **CMPMx: Comparison x Match Interrupt Enable**
- **CMPUx: Comparison x Update Interrupt Enable**

53.7.15 PWM Interrupt Disable Register 2

Name: PWM_IDR2

Address: 0xF802C038

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
7	6	5	4	3	2	1	0
–	–	–	–	UNRE	–	–	WRDY

- **WRDY: Write Ready for Synchronous Channels Update Interrupt Disable**
- **UNRE: Synchronous Channels Update Underrun Error Interrupt Disable**
- **CMPMx: Comparison x Match Interrupt Disable**
- **CMPUx: Comparison x Update Interrupt Disable**

53.7.16 PWM Interrupt Mask Register 2

Name: PWM_IMR2

Address: 0xF802C03C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0

15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0

7	6	5	4	3	2	1	0
–	–	–	–	UNRE	–	–	WRDY

- **WRDY: Write Ready for Synchronous Channels Update Interrupt Mask**
- **UNRE: Synchronous Channels Update Underrun Error Interrupt Mask**
- **CMPMx: Comparison x Match Interrupt Mask**
- **CMPUx: Comparison x Update Interrupt Mask**

53.7.17 PWM Interrupt Status Register 2

Name: PWM_ISR2

Address: 0xF802C040

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CMPU7	CMPU6	CMPU5	CMPU4	CMPU3	CMPU2	CMPU1	CMPU0
15	14	13	12	11	10	9	8
CMPM7	CMPM6	CMPM5	CMPM4	CMPM3	CMPM2	CMPM1	CMPM0
7	6	5	4	3	2	1	0
–	–	–	–	UNRE	–	–	WRDY

- **WRDY: Write Ready for Synchronous Channels Update**

0: New duty-cycle and dead-time values for the synchronous channels cannot be written.

1: New duty-cycle and dead-time values for the synchronous channels can be written.

- **UNRE: Synchronous Channels Update Underrun Error**

0: No Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register.

1: At least one Synchronous Channels Update Underrun has occurred since the last read of the PWM_ISR2 register.

- **CMPMx: Comparison x Match**

0: The comparison x has not matched since the last read of the PWM_ISR2 register.

1: The comparison x has matched at least one time since the last read of the PWM_ISR2 register.

- **CMPUx: Comparison x Update**

0: The comparison x has not been updated since the last read of the PWM_ISR2 register.

1: The comparison x has been updated at least one time since the last read of the PWM_ISR2 register.

Note: Reading PWM_ISR2 automatically clears flags WRDY, UNRE and CMPSx.

53.7.18 PWM Output Override Value Register

Name: PWM_OOV

Address: 0xF802C044

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OOVL3	OOVL2	OOVL1	OOVL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OOVH3	OOVH2	OOVH1	OOVH0

- **OOVHx: Output Override Value for PWMH output of the channel x**

0: Override value is 0 for PWMH output of channel x.

1: Override value is 1 for PWMH output of channel x.

- **OOVLx: Output Override Value for PWML output of the channel x**

0: Override value is 0 for PWML output of channel x.

1: Override value is 1 for PWML output of channel x.

53.7.19 PWM Output Selection Register

Name: PWM_OS

Address: 0xF802C048

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSL3	OSL2	OSL1	OSL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSH3	OSH2	OSH1	OSH0

- **OSHx: Output Selection for PWMH output of the channel x**

0: Dead-time generator output DTOHx selected as PWMH output of channel x.

1: Output override value OOVHx selected as PWMH output of channel x.

- **OSLx: Output Selection for PWML output of the channel x**

0: Dead-time generator output DTOLx selected as PWML output of channel x.

1: Output override value OOVLx selected as PWML output of channel x.

53.7.20 PWM Output Selection Set Register

Name: PWM_OSS

Address: 0xF802C04C

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSSL3	OSSL2	OSSL1	OSSL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSSH3	OSSH2	OSSH1	OSSH0

- **OSSHx: Output Selection Set for PWMH output of the channel x**

0: No effect.

1: Output override value OOVHx selected as PWMH output of channel x.

- **OSSLx: Output Selection Set for PWML output of the channel x**

0: No effect.

1: Output override value OOVLx selected as PWML output of channel x.

53.7.21 PWM Output Selection Clear Register

Name: PWM_OSC

Address: 0xF802C050

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSCL3	OSCL2	OSCL1	OSCL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSCH3	OSCH2	OSCH1	OSCH0

- **OSCHx: Output Selection Clear for PWMH output of the channel x**

0: No effect.

1: Dead-time generator output DTOHx selected as PWMH output of channel x.

- **OSCLx: Output Selection Clear for PWML output of the channel x**

0: No effect.

1: Dead-time generator output DTOLx selected as PWML output of channel x.

53.7.22 PWM Output Selection Set Update Register

Name: PWM_OSSUPD

Address: 0xF802C054

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSSUPL3	OSSUPL2	OSSUPL1	OSSUPL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSSUPH3	OSSUPH2	OSSUPH1	OSSUPH0

- **OSSUPHx: Output Selection Set for PWMH output of the channel x**

0: No effect.

1: Output override value OOVHx selected as PWMH output of channel x at the beginning of the next channel x PWM period.

- **OSSUPLx: Output Selection Set for PWML output of the channel x**

0: No effect.

1: Output override value OOVLx selected as PWML output of channel x at the beginning of the next channel x PWM period.

53.7.23 PWM Output Selection Clear Update Register

Name: PWM_OSCUPD

Address: 0xF802C058

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	OSCUPL3	OSCUPL2	OSCUPL1	OSCUPL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	OSCUPH3	OSCUPH2	OSCUPH1	OSCUPH0

- **OSCUPHx: Output Selection Clear for PWMH output of the channel x**

0: No effect.

1: Dead-time generator output DTOHx selected as PWMH output of channel x at the beginning of the next channel x PWM period.

- **OSCUPLx: Output Selection Clear for PWML output of the channel x**

0: No effect.

1: Dead-time generator output DTOLx selected as PWML output of channel x at the beginning of the next channel x PWM period.

53.7.24 PWM Fault Mode Register

Name: PWM_FMR

Address: 0xF802C05C

Access: Read/Write

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
FFIL							

15	14	13	12	11	10	9	8
FMOD							

7	6	5	4	3	2	1	0
FPOL							

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Refer to Section 53.5.4 “Fault Inputs” for details on fault generation.

- **FPOL: Fault Polarity**

For each bit y of FPOL, where y is the fault input number:

0: The fault y becomes active when the fault input y is at 0.

1: The fault y becomes active when the fault input y is at 1.

- **FMOD: Fault Activation Mode**

For each bit y of FMOD, where y is the fault input number:

0: The fault y is active until the fault condition is removed at the peripheral(1) level.

1: The fault y stays active until the fault condition is removed at the peripheral(1) level AND until it is cleared in the PWM Fault Clear Register.

Note: 1. The peripheral generating the fault.

- **FFIL: Fault Filtering**

For each bit y of FFIL, where y is the fault input number:

0: The fault input y is not filtered.

1: The fault input y is filtered.

CAUTION: To prevent an unexpected activation of the status flag FSy in the PWM Fault Status Register, the bit FMODy can be set to ‘1’ only if the FPOLy bit has been previously configured to its final value.

53.7.25 PWM Fault Status Register

Name: PWM_FSR

Address: 0xF802C060

Access: Read-only

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
FS							

7	6	5	4	3	2	1	0
FIV							

Refer to Section 53.5.4 “Fault Inputs” for details on fault generation.

- **FIV: Fault Input Value**

For each bit y of FIV, where y is the fault input number:

0: The current sampled value of the fault input y is 0 (after filtering if enabled).

1: The current sampled value of the fault input y is 1 (after filtering if enabled).

- **FS: Fault Status**

For each bit y of FS, where y is the fault input number:

0: The fault y is not currently active.

1: The fault y is currently active.

53.7.26 PWM Fault Clear Register

Name: PWM_FCR

Address: 0xF802C064

Access: Write-only

31	30	29	28	27	26	25	24
–							

23	22	21	20	19	18	17	16
–							

15	14	13	12	11	10	9	8
–							

7	6	5	4	3	2	1	0
FCLR							

Refer to Section 53.5.4 “Fault Inputs” for details on fault generation.

- **FCLR: Fault Clear**

For each bit y of FCLR, where y is the fault input number:

0: No effect.

1: If bit y of FMOD field is set to ‘1’ and if the fault input y is not at the level defined by the bit y of FPOL field, the fault y is cleared and becomes inactive (FMOD and FPOL fields belong to PWM Fault Mode Register), else writing this bit to ‘1’ has no effect.

53.7.27 PWM Fault Protection Value Register 1

Name: PWM_FPV1

Address: 0xF802C068

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FPVL3	FPVL2	FPVL1	FPVL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	FPVH3	FPVH2	FPVH1	FPVH0

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Refer to Section 53.5.4 “Fault Inputs” for details on fault generation.

- **FPVHx: Fault Protection Value for PWMH output on channel x**

This bit is taken into account only if the bit FPZHx is set to ‘0’ in PWM Fault Protection Value Register 2.

0: PWMH output of channel x is forced to ‘0’ when fault occurs.

1: PWMH output of channel x is forced to ‘1’ when fault occurs.

- **FPVLx: Fault Protection Value for PWML output on channel x**

This bit is taken into account only if the bit FPZLx is set to ‘0’ in PWM Fault Protection Value Register 2.

0: PWML output of channel x is forced to ‘0’ when fault occurs.

1: PWML output of channel x is forced to ‘1’ when fault occurs.

53.7.28 PWM Fault Protection Enable Register

Name: PWM_FPE

Address: 0xF802C06C

Access: Read/Write

31	30	29	28	27	26	25	24
FPE3							
23	22	21	20	19	18	17	16
FPE2							
15	14	13	12	11	10	9	8
FPE1							
7	6	5	4	3	2	1	0
FPE0							

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

Only the first 6 bits (number of fault input pins) of fields FPE0, FPE1, FPE2 and FPE3 are significant.

Refer to Section 53.5.4 “Fault Inputs” for details on fault generation.

- **FPEx: Fault Protection Enable for channel x**

For each bit y of FPEx, where y is the fault input number:

0: Fault y is not used for the fault protection of channel x.

1: Fault y is used for the fault protection of channel x.

CAUTION: To prevent an unexpected activation of the fault protection, the bit y of FPEx field can be set to ‘1’ only if the corresponding FPOL field has been previously configured to its final value in PWM Fault Mode Register.

53.7.29 PWM Event Line x Register

Name: PWM_ELMRx

Address: 0xF802C07C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
CSEL7	CSEL6	CSEL5	CSEL4	CSEL3	CSEL2	CSEL1	CSEL0

- **CSELy: Comparison y Selection**

0: A pulse is not generated on the event line x when the comparison y matches.

1: A pulse is generated on the event line x when the comparison y match.

53.7.30 PWM Spread Spectrum Register

Name: PWM_SSPR

Address: 0xF802C0A0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	SPRDM

23	22	21	20	19	18	17	16
SPRD							

15	14	13	12	11	10	9	8
SPRD							

7	6	5	4	3	2	1	0
SPRD							

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

- **SPRD: Spread Spectrum Limit Value**

The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying PWM period for the output waveform.

- **SPRDM: Spread Spectrum Counter Mode**

0: Triangular mode. The spread spectrum counter starts to count from -SPRD when the channel 0 is enabled and counts upwards at each PWM period. When it reaches +SPRD, it restarts to count from -SPRD again.

1: Random mode. The spread spectrum counter is loaded with a new random value at each PWM period. This random value is uniformly distributed and is between -SPRD and +SPRD.

53.7.31 PWM Spread Spectrum Update Register

Name: PWM_SSPUP

Address: 0xF802C0A4

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
SPRDUP							

15	14	13	12	11	10	9	8
SPRDUP							

7	6	5	4	3	2	1	0
SPRDUP							

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the SPRD value. This prevents an unexpected waveform when modifying the spread spectrum limit value.

Only the first 16 bits (channel counter size) are significant.

- **SPRDUP: Spread Spectrum Limit Value Update**

The spread spectrum limit value defines the range for the spread spectrum counter. It is introduced in order to achieve constant varying period for the output waveform.

53.7.32 PWM Stepper Motor Mode Register

Name: PWM_SMMR

Address: 0xF802C0B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	–	–	DOWN1	DOWN0

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	–	–	–	–	–	GCEN1	GCEN0

- **GCENx: Gray Count Enable**

0: Disable gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1]

1: Enable gray count generation on PWML[2*x], PWMH[2*x], PWML[2*x +1], PWMH[2*x +1].

- **DOWNx: Down Count**

0: Up counter.

1: Down counter.

53.7.33 PWM Fault Protection Value Register 2

Name: PWM_FPV2

Address: 0xF802C0C0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	FPZL3	FPZL2	FPZL1	FPZL0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	FPZH3	FPZH2	FPZH1	FPZH0

This register can only be written if bits WPSWS5 and WPHWS5 are cleared in the PWM Write Protection Status Register.

- **FPZHx: Fault Protection to Hi-Z for PWMH output on channel x**

0: When fault occurs, PWMH output of channel x is forced to value defined by the bit FPVHx in PWM Fault Protection Value Register 1.

1: When fault occurs, PWMH output of channel x is forced to high-impedance state.

- **FPZLx: Fault Protection to Hi-Z for PWML output on channel x**

0: When fault occurs, PWML output of channel x is forced to value defined by the bit FPVLx in PWM Fault Protection Value Register 1.

1: When fault occurs, PWML output of channel x is forced to high-impedance state.

53.7.34 PWM Write Protection Control Register

Name: PWM_WPCR

Address: 0xF802C0E4

Access: Write-only

31	30	29	28	27	26	25	24
WPKEY							

23	22	21	20	19	18	17	16
WPKEY							

15	14	13	12	11	10	9	8
WPKEY							

7	6	5	4	3	2	1	0
WPRG5	WPRG4	WPRG3	WPRG2	WPRG1	WPRG0	WPCMD	

See Section 53.6.7 “Register Write Protection” for the list of registers that can be write-protected.

- **WPCMD: Write Protection Command**

This command is performed only if the WPKEY corresponds to 0x50574D (“PWM” in ASCII).

Value	Name	Description
0	DISABLE_SW_PROT	Disables the software write protection of the register groups of which the bit WPRGx is at ‘1’.
1	ENABLE_SW_PROT	Enables the software write protection of the register groups of which the bit WPRGx is at ‘1’.
2	ENABLE_HW_PROT	Enables the hardware write protection of the register groups of which the bit WPRGx is at ‘1’. Only a hardware reset of the PWM controller can disable the hardware write protection. Moreover, to meet security requirements, the PIO lines associated with the PWM can not be configured through the PIO interface.

- **WPRGx: Write Protection Register Group x**

0: The WPCMD command has no effect on the register group x.

1: The WPCMD command is applied to the register group x.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x50574D	PASSWD	Writing any other value in this field aborts the write operation of the WPCMD field. Always reads as 0

53.7.35 PWM Write Protection Status Register

Name: PWM_WPSR

Address: 0xF802C0E8

Access: Read-only

31	30	29	28	27	26	25	24
WPVSRC							

23	22	21	20	19	18	17	16
WPVSRC							

15	14	13	12	11	10	9	8
–	–	WPHWS5	WPHWS4	WPHWS3	WPHWS2	WPHWS1	WPHWS0

7	6	5	4	3	2	1	0
WPVS	–	WPSWS5	WPSWS4	WPSWS3	WPSWS2	WPSWS1	WPSWS0

- **WPSWSx: Write Protect SW Status**

0: The SW write protection x of the register group x is disabled.

1: The SW write protection x of the register group x is enabled.

- **WPHWSx: Write Protect HW Status**

0: The HW write protection x of the register group x is disabled.

1: The HW write protection x of the register group x is enabled.

- **WPVS: Write Protect Violation Status**

0: No write protection violation has occurred since the last read of PWM_WPSR.

1: At least one write protection violation has occurred since the last read of PWM_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protect Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

53.7.36 PWM Comparison x Value Register

Name: PWM_CMPVx

Address: 0xF802C130 [0], 0xF802C140 [1], 0xF802C150 [2], 0xF802C160 [3], 0xF802C170 [4], 0xF802C180 [5], 0xF802C190 [6], 0xF802C1A0 [7]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	CVM

23	22	21	20	19	18	17	16
CV							

15	14	13	12	11	10	9	8
CV							

7	6	5	4	3	2	1	0
CV							

Only the first 16 bits (channel counter size) of field CV are significant.

- **CV: Comparison x Value**

Define the comparison x value to be compared with the counter of the channel 0.

- **CVM: Comparison x Value Mode**

0: The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing.

1: The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing.

Note: This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register)

53.7.37 PWM Comparison x Value Update Register

Name: PWM_CMPVUPDx

Address: 0xF802C134 [0], 0xF802C144 [1], 0xF802C154 [2], 0xF802C164 [3], 0xF802C174 [4], 0xF802C184 [5], 0xF802C194 [6], 0xF802C1A4 [7]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	CVMUPD

23	22	21	20	19	18	17	16
CVUPD							

15	14	13	12	11	10	9	8
CVUPD							

7	6	5	4	3	2	1	0
CVUPD							

This register acts as a double buffer for the CV and CVM values. This prevents an unexpected comparison x match.

Only the first 16 bits (channel counter size) of field CVUPD are significant.

- **CVUPD: Comparison x Value Update**

Define the comparison x value to be compared with the counter of the channel 0.

- **CVMUPD: Comparison x Value Mode Update**

0: The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is incrementing.

1: The comparison x between the counter of the channel 0 and the comparison x value is performed when this counter is decrementing.

Note: This bit is not relevant if the counter of the channel 0 is left-aligned (CALG = 0 in PWM Channel Mode Register)

CAUTION: The write of the register PWM_CMPVUPDx must be followed by a write of the register PWM_CMPMUPDx.

53.7.38 PWM Comparison x Mode Register

Name: PWM_CMPMx

Address: 0xF802C138 [0], 0xF802C148 [1], 0xF802C158 [2], 0xF802C168 [3], 0xF802C178 [4], 0xF802C188 [5], 0xF802C198 [6], 0xF802C1A8 [7]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CUPRCNT				CUPR			

15	14	13	12	11	10	9	8
CPRCNT				CPR			

7	6	5	4	3	2	1	0
CTR				–	–	–	CEN

- **CEN: Comparison x Enable**

0: The comparison x is disabled and can not match.

1: The comparison x is enabled and can match.

- **CTR: Comparison x Trigger**

The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

- **CPR: Comparison x Period**

CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

- **CPRCNT: Comparison x Period Counter**

Reports the value of the comparison x period counter.

Note: The field CPRCNT is read-only

- **CUPR: Comparison x Update Period**

Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

- **CUPRCNT: Comparison x Update Period Counter**

Reports the value of the comparison x update period counter.

Note: The field CUPRCNT is read-only

53.7.39 PWM Comparison x Mode Update Register

Name: PWM_CMPMUPDx

Address: 0xF802C13C [0], 0xF802C14C [1], 0xF802C15C [2], 0xF802C16C [3], 0xF802C17C [4], 0xF802C18C [5], 0xF802C19C [6], 0xF802C1AC [7]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	CUPRUPD			
15	14	13	12	11	10	9	8
–	–	–	–	CPRUPD			
7	6	5	4	3	2	1	0
CTRUPD				–	–	–	CENUPD

This register acts as a double buffer for the CEN, CTR, CPR and CUPR values. This prevents an unexpected comparison x match.

- **CENUPD: Comparison x Enable Update**

0: The comparison x is disabled and can not match.

1: The comparison x is enabled and can match.

- **CTRUPD: Comparison x Trigger Update**

The comparison x is performed when the value of the comparison x period counter (CPRCNT) reaches the value defined by CTR.

- **CPRUPD: Comparison x Period Update**

CPR defines the maximum value of the comparison x period counter (CPRCNT). The comparison x value is performed periodically once every CPR+1 periods of the channel 0 counter.

- **CUPRUPD: Comparison x Update Period Update**

Defines the time between each update of the comparison x mode and the comparison x value. This time is equal to CUPR+1 periods of the channel 0 counter.

53.7.40 PWM Channel Mode Register

Name: PWM_CMRx [x=0..3]

Address: 0xF802C200 [0], 0xF802C220 [1], 0xF802C240 [2], 0xF802C260 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	PPM	DTLI	DTHI	DTE

15	14	13	12	11	10	9	8
–	–	TCTS	DPOLI	UPDS	CES	CPOL	CALG

7	6	5	4	3	2	1	0
–	–	–	–	CPRE			

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

- **CPRE: Channel Prescaler**

Value	Name	Description
0	MCK	Peripheral clock
1	MCK_DIV_2	Peripheral clock/2
2	MCK_DIV_4	Peripheral clock/4
3	MCK_DIV_8	Peripheral clock/8
4	MCK_DIV_16	Peripheral clock/16
5	MCK_DIV_32	Peripheral clock/32
6	MCK_DIV_64	Peripheral clock/64
7	MCK_DIV_128	Peripheral clock/128
8	MCK_DIV_256	Peripheral clock/256
9	MCK_DIV_512	Peripheral clock/512
10	MCK_DIV_1024	Peripheral clock/1024
11	CLKA	Clock A
12	CLKB	Clock B

- **CALG: Channel Alignment**

0: The period is left-aligned.

1: The period is center-aligned.

- **CPOL: Channel Polarity**

0: The OCx output waveform (output from the comparator) starts at a low level.

1: The OCx output waveform (output from the comparator) starts at a high level.

- **CES: Counter Event Selection**

The bit CES defines when the channel counter event occurs when the period is center-aligned (flag CHIDx in PWM Interrupt Status Register 1).

CALG = 0 (Left Alignment):

0/1: The channel counter event occurs at the end of the PWM period.

CALG = 1 (Center Alignment):

0: The channel counter event occurs at the end of the PWM period.

1: The channel counter event occurs at the end of the PWM period and at half the PWM period.

- **UPDS: Update Selection**

When the period is center aligned, the bit UPDS defines when the update of the duty cycle, the polarity value/mode occurs after writing the corresponding update registers.

CALG = 0 (Left Alignment):

0/1: The update always occurs at the end of the PWM period after writing the update register(s).

CALG = 1 (Center Alignment):

0: The update occurs at the next end of the PWM period after writing the update register(s).

1: The update occurs at the next end of the PWM half period after writing the update register(s).

- **DPOLI: Disabled Polarity Inverted**

0: When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is the same as the one defined by the CPOL bit.

1: When the PWM channel x is disabled (CHIDx(PWM_SR) = 0), the OCx output waveform is inverted compared to the one defined by the CPOL bit.

- **TCTS: Timer Counter Trigger Selection**

0: The comparator of the channel x (OCx) is used as the trigger source for the Timer Counter (TC).

1: The counter events of the channel x is used as the trigger source for the Timer Counter (TC).

- **DTE: Dead-Time Generator Enable**

0: The dead-time generator is disabled.

1: The dead-time generator is enabled.

- **DTHI: Dead-Time PWMHx Output Inverted**

0: The dead-time PWMHx output is not inverted.

1: The dead-time PWMHx output is inverted.

- **DTLI: Dead-Time PWMLx Output Inverted**

0: The dead-time PWMLx output is not inverted.

1: The dead-time PWMLx output is inverted.

- **PPM: Push-Pull Mode**

0: The Push-Pull mode is disabled for channel x.

1: The Push-Pull mode is enabled for channel x.

53.7.41 PWM Channel Duty Cycle Register

Name: PWM_CDTYx [x=0..3]

Address: 0xF802C204 [0], 0xF802C224 [1], 0xF802C244 [2], 0xF802C264 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CDTY							

15	14	13	12	11	10	9	8
CDTY							

7	6	5	4	3	2	1	0
CDTY							

Only the first 16 bits (channel counter size) are significant.

- **CDTY: Channel Duty-Cycle**

Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

53.7.42 PWM Channel Duty Cycle Update Register

Name: PWM_CDTYUPDx [x=0..3]

Address: 0xF802C208 [0], 0xF802C228 [1], 0xF802C248 [2], 0xF802C268 [3]

Access: Write-only.

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CDTYUPD							

15	14	13	12	11	10	9	8
CDTYUPD							

7	6	5	4	3	2	1	0
CDTYUPD							

This register acts as a double buffer for the CDTY value. This prevents an unexpected waveform when modifying the waveform duty-cycle.

Only the first 16 bits (channel counter size) are significant.

- **CDTYUPD: Channel Duty-Cycle Update**

Defines the waveform duty-cycle. This value must be defined between 0 and CPRD (PWM_CPRDx).

53.7.43 PWM Channel Period Register

Name: PWM_CPRDx [x=0..3]

Address: 0xF802C20C [0], 0xF802C22C [1], 0xF802C24C [2], 0xF802C26C [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CPRD							

15	14	13	12	11	10	9	8
CPRD							

7	6	5	4	3	2	1	0
CPRD							

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (channel counter size) are significant.

- **CPRD: Channel Period**

If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times CPRD)}{f_{\text{peripheral clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times CPRD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times CPRD \times DIVB)}{f_{\text{peripheral clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times CPRD)}{f_{\text{peripheral clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times CPRD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times CPRD \times DIVB)}{f_{\text{peripheral clock}}}$$

53.7.44 PWM Channel Period Update Register

Name: PWM_CPRDUPDx [x=0..3]

Address: 0xF802C210 [0], 0xF802C230 [1], 0xF802C250 [2], 0xF802C270 [3]

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CPRDUPD							

15	14	13	12	11	10	9	8
CPRDUPD							

7	6	5	4	3	2	1	0
CPRDUPD							

This register can only be written if bits WPSWS3 and WPHWS3 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPRD value. This prevents an unexpected waveform when modifying the waveform period.

Only the first 16 bits (channel counter size) are significant.

- **CPRDUPD: Channel Period Update**

If the waveform is left-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(X \times CPRDUPD)}{f_{\text{peripheral clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(X \times CRPDUPD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(X \times CRPDUPD \times DIVB)}{f_{\text{peripheral clock}}}$$

If the waveform is center-aligned, then the output waveform period depends on the channel counter source clock and can be calculated:

– By using the PWM peripheral clock divided by a given prescaler value “X” (where X = 2^{PREA} is 1, 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1024). The resulting period formula is:

$$\frac{(2 \times X \times CPRDUPD)}{f_{\text{peripheral clock}}}$$

– By using the PWM peripheral clock divided by a given prescaler value “X” (see above) and by either the DIVA or the DIVB divider. The formula becomes, respectively:

$$\frac{(2 \times X \times CPRDUPD \times DIVA)}{f_{\text{peripheral clock}}} \text{ or } \frac{(2 \times X \times CPRDUPD \times DIVB)}{f_{\text{peripheral clock}}}$$

53.7.45 PWM Channel Counter Register

Name: PWM_CCNTx [x=0..3]

Address: 0xF802C214 [0], 0xF802C234 [1], 0xF802C254 [2], 0xF802C274 [3]

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
CNT							

15	14	13	12	11	10	9	8
CNT							

7	6	5	4	3	2	1	0
CNT							

Only the first 16 bits (channel counter size) are significant.

- **CNT: Channel Counter Register**

Channel counter value. This register is reset when:

- the channel is enabled (writing CHIDx in the PWM_ENA register).
- the channel counter reaches CPRD value defined in the PWM_CPRDx register if the waveform is left-aligned.

53.7.46 PWM Channel Dead Time Register

Name: PWM_DTx [x=0..3]

Address: 0xF802C218 [0], 0xF802C238 [1], 0xF802C258 [2], 0xF802C278 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
DTL							

23	22	21	20	19	18	17	16
DTL							

15	14	13	12	11	10	9	8
DTH							

7	6	5	4	3	2	1	0
DTH							

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

Only the first 16 bits (dead-time counter size) of fields DTH and DTL are significant.

- **DTH: Dead-Time Value for PWMHx Output**

Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx).

- **DTL: Dead-Time Value for PWMLx Output**

Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx).

53.7.47 PWM Channel Dead Time Update Register

Name: PWM_DTUPDx [x=0..3]

Address: 0xF802C21C [0], 0xF802C23C [1], 0xF802C25C [2], 0xF802C27C [3]

Access: Write-only

31	30	29	28	27	26	25	24
DTLUPD							

23	22	21	20	19	18	17	16
DTLUPD							

15	14	13	12	11	10	9	8
DTHUPD							

7	6	5	4	3	2	1	0
DTHUPD							

This register can only be written if bits WPSWS4 and WPHWS4 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the DTH and DTL values. This prevents an unexpected waveform when modifying the dead-time values.

Only the first 16 bits (dead-time counter size) of fields DTHUPD and DTLUPD are significant.

- **DTHUPD: Dead-Time Value Update for PWMHx Output**

Defines the dead-time value for PWMHx output. This value must be defined between 0 and the value (CPRD – CDTY) (PWM_CPRDx and PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

- **DTLUPD: Dead-Time Value Update for PWMLx Output**

Defines the dead-time value for PWMLx output. This value must be defined between 0 and CDTY (PWM_CDTYx). This value is applied only at the beginning of the next channel x PWM period.

53.7.48 PWM Channel Mode Update Register

Name: PWM_CMUPDx [x=0..3]

Address: 0xF802C400 [0], 0xF802C420 [1], 0xF802C440 [2], 0xF802C460 [3]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	CPOLINVUP	–	–	–	CPOLUP	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

This register can only be written if bits WPSWS2 and WPHWS2 are cleared in the PWM Write Protection Status Register.

This register acts as a double buffer for the CPOL value. This prevents an unexpected waveform when modifying the polarity value.

- **CPOLUP: Channel Polarity Update**

The write of this bit is taken into account only if the bit CPOLINVUP is written at ‘0’ at the same time.

0: The OCx output waveform (output from the comparator) starts at a low level.

1: The OCx output waveform (output from the comparator) starts at a high level.

- **CPOLINVUP: Channel Polarity Inversion Update**

If this bit is written at ‘1’, the write of the bit CPOLUP is not taken into account.

0: No effect.

1: The OCx output waveform (output from the comparator) is inverted.

53.7.49 PWM External Trigger Register

Name: PWM_ETRGx [x=1..2]

Address: 0xF802C42C [1], 0xF802C44C [2]

Access: Read/Write

31	30	29	28	27	26	25	24
RFEN	TRGSRC	TRGFILT	TRGEDGE	–	–	TRGMODE	
23	22	21	20	19	18	17	16
MAXCNT							
15	14	13	12	11	10	9	8
MAXCNT							
7	6	5	4	3	2	1	0
MAXCNT							

- **MAXCNT: Maximum Counter value**

Maximum channel x counter value measured at the TRGINx event since the last read of the register.

At the TRGINx event, if the channel x counter value is greater than the stored MAXCNT value, then MAXCNT is updated by the channel x counter value.

- **TRGMODE: External Trigger Mode**

Value	Name	Description
0	OFF	External trigger is not enabled.
1	MODE1	External PWM Reset Mode
2	MODE2	External PWM Start Mode
3	MODE3	Cycle-by-cycle Duty Mode

- **TRGEDGE: Edge Selection**

Value	Name	Description
0	FALLING_ZERO	TRGMODE = 1: TRGINx event detection on falling edge. TRGMODE = 2, 3: TRGINx active level is 0
1	RISING_ONE	TRGMODE = 1: TRGINx event detection on rising edge. TRGMODE = 2, 3: TRGINx active level is 1

- **TRGFILT: Filtered input**

0: The external trigger input x is not filtered.

1: The external trigger input x is filtered.

- **RFEN: Recoverable Fault Enable**

0: The TRGINx signal does not generate a recoverable fault.

1: The TRGINx signal generate a recoverable fault in place of the fault x input.

- **TRGSRC: Trigger Source**

0: The TRGINx signal is driven by the PWMEXTRGx input.

1: The TRGINx signal is driven by the Analog Comparator Controller.

53.7.50 PWM Leading-Edge Blanking Register

Name: PWM_LEBRx [x=1..2]

Address: 0xF802C430 [1], 0xF802C450 [2]

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–

23	22	21	20	19	18	17	16
–	–	–	–	PWMHREN	PWMHFEN	PWMLREN	PWMLFEN

15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–

7	6	5	4	3	2	1	0
–	LEBDELAY						

- **LEBDELAY: Leading-Edge Blanking Delay for TRGINx**

Leading-edge blanking duration for external trigger x input. The delay is calculated according to the following formula:

$$LEBDELAY = (f_{peripheral\ clock} \times Delay) + 1$$

- **PWMLFEN: PWML Falling Edge Enable**

0: Leading-edge blanking is disabled on PWMLx output falling edge.

1: Leading-edge blanking is enabled on PWMLx output falling edge.

- **PWMLREN: PWML Rising Edge Enable**

0: Leading-edge blanking is disabled on PWMLx output rising edge.

1: Leading-edge blanking is enabled on PWMLx output rising edge.

- **PWMHFEN: PWMH Falling Edge Enable**

0: Leading-edge blanking is disabled on PWMHx output falling edge.

1: Leading-edge blanking is enabled on PWMHx output falling edge.

- **PWMHREN: PWMH Rising Edge Enable**

0: Leading-edge blanking is disabled on PWMHx output rising edge.

1: Leading-edge blanking is enabled on PWMHx output rising edge.

54. Secure Fuse Controller (SFC)

54.1 Description

The Secure Fuse Controller (SFC) interfaces the system with electrical fuses in a secure way.

The default value of a fuse is logic ‘0’ (not programmed). A programmed fuse is logic ‘1’.

An electrical fuse matrix is a type of non-volatile memory. Each fuse in the matrix can be programmed only one time. They are typically used to store calibration bits for analog cells such as oscillators, configuration settings, chip identifiers or cryptographic keys.

A specific number of fuse bits are programmed by Atmel during the production tests through the test interface. The remaining 544 fuse bits are programmed by the user and by software through the user interface.

The SFC automatically reads the fuse values on startup and stores them in 32-bit registers in order to make them accessible by the software. Only fuses set to level ‘1’ are programmed.

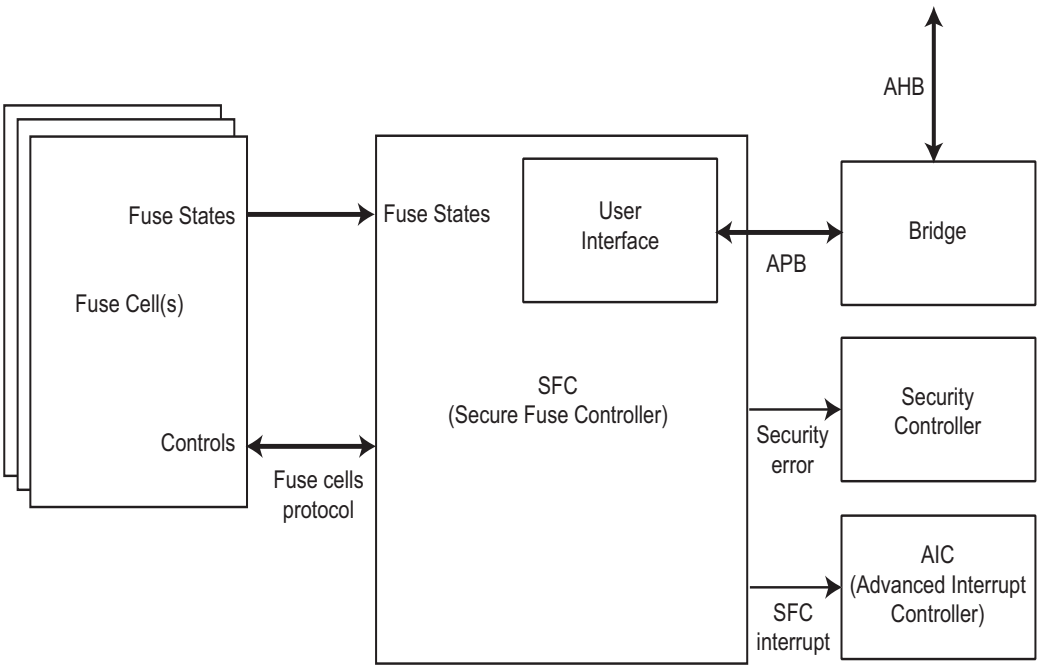
Several security mechanisms make irregular data recovery more complex to achieve.

54.2 Embedded Characteristics

- Fuse bits partitioned into two areas:
  - Atmel reserved area
  - 544-bit user area
- Program and read the fuse states by software
- Automatic check of programmed fuses
- Detection of irregular alteration of the fuse states in Atmel reserved area during startup and report
- Live detection of irregular alteration of all the fuse states and report
- Part of fuse states maskable for reading

54.3 Block Diagram

Figure 54-1. SFC Block Diagram



54.4 Functional Description

54.4.1 Accessing the SFC

Setting the write-once FUSE bit in the SFR_SECURE register disables access to the Secure Fuse Controller (SFC).

54.4.2 Fuse Partitioning

The fuses are split into a user area of 544 bits and an Atmel reserved area.

The Atmel reserved area is typically used to store calibration bits for analog cells such as oscillators, configuration settings, chip identifiers, etc. The user area fuses are programmed later on by the user.

54.4.3 Fuse Integrity Checking

The SFC automatically reads the fuses values at startup and stores them in 32-bit registers in order to make them accessible by software. At this time, the SFC checks the integrity of the fuse states in the Atmel reserved area.

If an inconsistency is detected, the CHECK error flag (named ACE for the Atmel reserved area) in the Status Register (SFC_SR) is set to ‘1’ and can trigger an interrupt. This flag is automatically cleared at ‘0’, when the Status Register (SFC_SR) is read.

54.4.4 Fuse Integrity Live Checking

The SFC automatically checks the integrity of all fuse states at every time after the startup is finished. This ensures that the fuses states cannot be changed without notice.

If an inconsistency is detected, the LCHECK error flag in the Status Register (SFC_SR) is set to ‘1’ and can trigger an interrupt. This flag is automatically cleared at ‘0’, when the Status Register (SFC_SR) is read.

54.4.5 Fuse Access

54.4.5.1 Fuse Reading

The fuse states are automatically latched at core startup and are available for reading in the Data Registers (SFC_DRx).

The fuse states of bits 0 to 31 are available in the Data Register 0 (SFC_DR0), the fuse states of bits 32 to 63 are available in the Data Register 1 (SFC_DR1) and so on.

When fuse programming is performed, the fuse states are automatically updated in the Data Registers (SFC_DRx).

54.4.5.2 Fuse Programming

All the fuses can be written by software.

The sequence of instructions to program fuses is the following:

1. Write the key code 0xFB in the Key Register (SFC_KR).
2. Write the word to program in the corresponding Data Register (SFC_DRx).
   For example, if fuses 0 to 31 must be programmed, Data Register 0 (SFC_DR0) must be written. If fuses 32 to 61 must be programmed, Data Register 1 (SFC_DR1) must be written. Only the data bits set to level ‘1’ are programmed.
3. Wait for flag PGMC to rise in the Status Register (SFC_SR) by polling or interrupt.
4. Check the value of flag PGMF: if it is set to 1, it means that the programming procedure failed. After programming, the fuses are read back in the corresponding SFC_DRx.

54.4.5.3 Fuse Masking

It is possible to mask a fuse array. Once the fuse masking is enabled, the data registers from SFC_DR20 to SFC_DR23 are read at a value of ‘0’, regardless of the fuse state (the registers that are masked depend on the SFC hardware customizing).

To activate fuse masking, the MSK bit of the SFC Mode Register (SFC_MR) must be written to level ‘1’. The MSK bit is set-only. Only a hardware reset can disable fuse masking.

The MSK bit has no effect on the programming of masked fuses.

54.4.6 Fuse Functions

The “Fuse Box Controller” section defines the fuse bits that can be used as general purpose bits when standard boot is used.

If secure boot is used, refer to the device “Secure Boot Strategy” application note included in the Secure Package.

54.5 Secure Fuse Controller (SFC) User Interface

Table 54-1. Register Mapping

Offset	Register	Name	Access	Reset
0x00	SFC Key Register	SFC_KR	Write-only	–
0x04	SFC Mode Register	SFC_MR	Read/Write	0x0
0x08–0x0C	Reserved	–	–	–
0x10	SFC Interrupt Enable Register	SFC_IER	Write-only	–
0x14	SFC Interrupt Disable Register	SFC_IDR	Write-only	–
0x18	SFC Interrupt Mask Register	SFC_IMR	Read-only	0x0
0x1C	SFC Status Register	SFC_SR	Read-only	0x0
0x20	SFC Data Register 0	SFC_DR0	Read/Write	0x0
0x24	SFC Data Register 1	SFC_DR1	Read/Write	0x0
...	...	...	...	...
0x7C	SFC Data Register 23	SFC_DR23	Read/Write	0x0
0x80–0xFC	Reserved	–	–	–

54.5.1 SFC Key Register

Name: SFC_KR

Address: 0xF804C000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
KEY							

- **KEY: Key Code**

This field must be written with the correct key code (0xFB) prior to any write in a Data Register (SFC_DRx) in order to enable the fuse programming. For each write of SFC_DRx, this field must be written immediately before.

54.5.2 SFC Mode Register

Name: SFC_MR

Address: 0xF804C004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	SASEL	–	–	–	MSK

- **MSK: Mask Data Registers**

0: No effect

1: The data registers from SFC_DR20 to SFC_DR23 are always read at 0x00000000.

Note: The MSK bit is set-only. Only a hardware reset can disable fuse masking.

- **SASEL: Sense Amplifier Selection**

0: Comparator type sense amplifier selected

1: Latch type sense amplifier selected

54.5.3 SFC Interrupt Enable Register

Name: SFC_IER

Address: 0xF804C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	ACE	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	LCHECK	–	–	PGMF	PGMC

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Enables the corresponding interrupt

- **PGMC: Programming Sequence Completed Interrupt Enable**
- **PGMF: Programming Sequence Failed Interrupt Enable**
- **LCHECK: Live Integrity Check Error Interrupt Enable**
- **ACE: Atmel Check Error Interrupt Enable**

54.5.4 SFC Interrupt Disable Register

Name: SFC_IDR

Address: 0xF804C014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	ACE	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	LCHECK	–	–	PGMF	PGMC

The following configuration values are valid for all listed bit names of this register:

0: No effect

1: Disables the corresponding interrupt

- **PGMC: Programming Sequence Completed Interrupt Disable**
- **PGMF: Programming Sequence Failed Interrupt Disable**
- **LCHECK: Live Integrity Check Error Interrupt Disable**
- **ACE: Atmel Check Error Interrupt Disable**

54.5.5 SFC Interrupt Mask Register

Name: SFC_IMR

Address: 0xF804C018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	ACE	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	LCHECK	–	–	PGMF	PGMC

The following configuration values are valid for all listed bit names of this register:

0: Corresponding interrupt is not enabled.

1: Corresponding interrupt is enabled.

- **PGMC: Programming Sequence Completed Interrupt Mask**
- **PGMF: Programming Sequence Failed Interrupt Mask**
- **LCHECK: Live Integrity Checking Error Interrupt Mask**
- **ACE: Atmel Check Error Interrupt Mask**

54.5.6 SFC Status Register

Name: SFC_SR

Address: 0xF804C01C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	ACE	APLE
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	LCHECK	–	–	PGMF	PGMC

- **PGMC: Programming Sequence Completed (cleared on read)**

0: No programming sequence completion since the last read of SFC_SR.

1: At least one programming sequence completion since the last read of SFC_SR.

- **PGMF: Programming Sequence Failed (cleared on read)**

0: No programming failure occurred during last programming sequence since the last read of SFC_SR.

1: A programming failure occurred since the last read of SFC_SR.

- **LCHECK: Live Integrity Checking Error (cleared on read)**

0: No live integrity check error since the last read of SFC_SR.

1: At least one live integrity check error since the last read of SFC_SR.

- **APLE: Atmel Programming Lock Error (cleared on read)**

0: No programming attempt has been made in the Atmel locked area since the last read of SFC_SR.

1: A programming attempt has been made in the Atmel locked area since the last read of SFC_SR.

- **ACE: Atmel Check Error (cleared on read)**

0: No check error in the Atmel reserved area since the last read of SFC_SR.

1: At least one check error in the Atmel reserved area since the last read of SFC_SR.

54.5.7 SFC Data Register x

Name: SFC_DRx [x=0..23]

Address: 0xF804C020

Access: Read/Write

31	30	29	28	27	26	25	24
DATA							
23	22	21	20	19	18	17	16
DATA							
15	14	13	12	11	10	9	8
DATA							
7	6	5	4	3	2	1	0
DATA							

- **DATA: Fuse Data**

READ: Reports the state of the corresponding fuses.

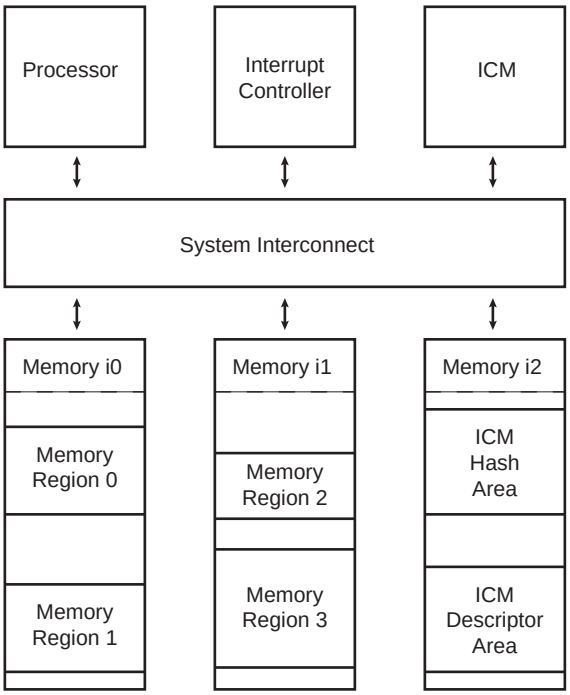
WRITE: The data to be programmed in the corresponding fuses. Only bits with a value of ‘1’ are programmed. Writing this register automatically triggers a programming sequence of the corresponding fuses. Note that a write to the Key Register (SFC_KR) with the correct key code must always precede any write to SFC_DRx.

55. Integrity Check Monitor (ICM)

55.1 Description

The Integrity Check Monitor (ICM) is a DMA controller that performs hash calculation over multiple memory regions through the use of transfer descriptors located in memory (ICM Descriptor Area). The Hash function is based on the Secure Hash Algorithm (SHA). The ICM controller integrates two modes of operation. The first one is used to hash a list of memory regions and save the digests to memory (ICM Hash Area). The second mode is an active monitoring of the memory. In that mode, the hash function is evaluated and compared to the digest located at a predefined memory address (ICM Hash Area). If a mismatch occurs, an interrupt is raised. See Figure 55-1 for an example of four-region monitoring. Hash and Descriptor areas are located in Memory instance i2, and the four regions are split in memory instances i0 and i1.

Figure 55-1. Four-region Monitoring Example



The ICM SHA engine is compliant with the American *FIPS (Federal Information Processing Standard) Publication 180-2* specification.

The following terms are concise definitions of the ICM concepts used throughout this document:

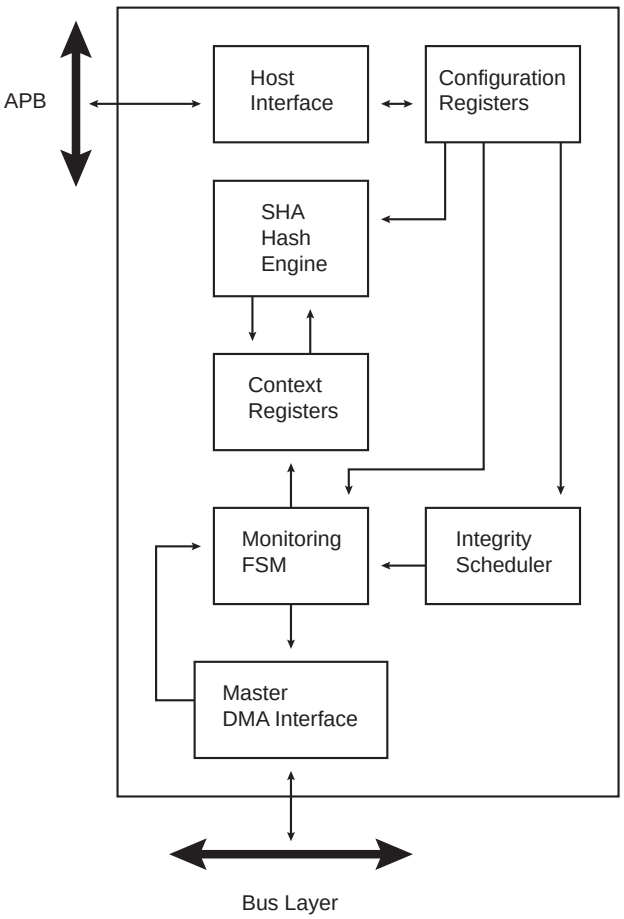
- Region—a partition of instruction or data memory space
- Region Descriptor—a data structure stored in memory, defining region attributes
- Region Attributes—region start address, region size, region SHA engine processing mode, Write Back or Compare function mode
- Context Registers—a set of ICM non-memory-mapped, internal registers which are automatically loaded, containing the attributes of the region being processed
- Main List—a list of region descriptors. Each element associates the start address of a region with a set of attributes.
- Secondary List—a linked list defined on a per region basis that describes the memory layout of the region (when the region is non-contiguous)
- Hash Area—predefined memory space where the region hash results (digest) are stored

55.2 Embedded Characteristics

- DMA AHB master interface
- Supports monitoring of up to 4 Non-Contiguous Memory Regions
- Supports block gathering through the use of linked list
- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256)
- Compliant with *FIPS Publication 180-2*
- Configurable Processing Period:
  - When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.
  - When SHA256 or SHA224 algorithm is processed, the runtime period is either 72 or 194 clock cycles.
- Programmable Bus burden

55.3 Block Diagram

Figure 55-2. Integrity Check Monitor Block Diagram



55.4 Product Dependencies

55.4.1 Power Management

The peripheral clock is not continuously provided to the ICM. The programmer must first enable the ICM clock in the Power Management Controller (PMC) before using the ICM.

55.4.2 Interrupt Sources

The ICM interface has an interrupt line connected to the Interrupt Controller.

Handling the ICM interrupt requires programming the interrupt controller before configuring the ICM.

Table 55-1. Peripheral IDs

Instance	ID
ICM	8

55.5 Functional Description

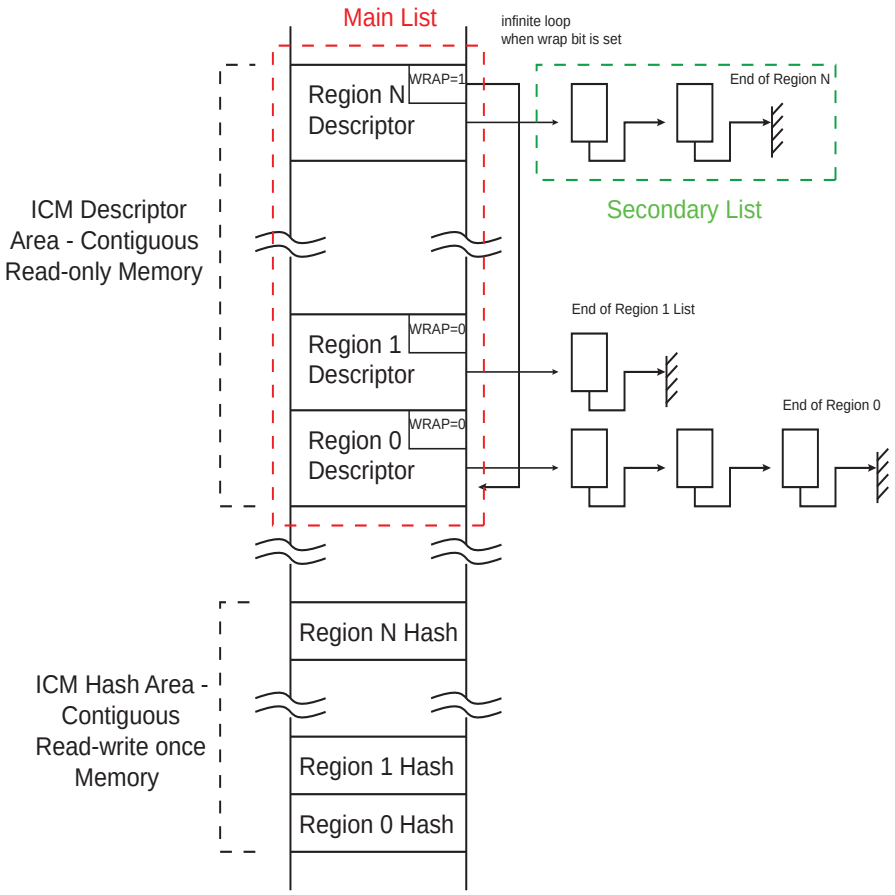
55.5.1 Overview

The Integrity Check Monitor (ICM) is a DMA controller that performs SHA-based memory hashing over memory regions. As shown in Figure 55-2, it integrates a DMA interface, a Monitoring Finite State Machine (FSM), an integrity scheduler, a set of context registers, a SHA engine, an interface for configuration and status registers.

The ICM integrates a Secure Hash Algorithm Engine (SHA). This engine requires a message padded according to FIPS180-2 specification when used as a SHA calculation unit only. Otherwise, if the ICM is used as integrated check for memory content, the padding is not mandatory. The SHA module produces an N-bit message digest each time a block is read and a processing period ends. N is 160 for SHA1, 224 for SHA224, 256 for SHA256.

When the ICM module is enabled, it sequentially retrieves a circular list of region descriptors from the memory (Main List described in Figure 55-3). Up to four regions may be monitored. Each region descriptor is composed of four words indicating the layout of the memory region (see Figure 55-4). It also contains the hashing engine configuration on a per region basis. As soon as the descriptor is loaded from the memory and context registers are updated with the data structure, the hashing operation starts. A programmable number of blocks (see TRSIZE field of the ICM_RCTRL structure member) is transferred from the memory to the SHA engine. When the desired number of blocks have been transferred, the digest is either moved to memory (Write Back function) or compared with a digest reference located in the system memory (Compare function). If a digest mismatch occurs, an interrupt is triggered if unmasked. The ICM module passes through the region descriptor list until the end of the list marked by an End of List bit set to one. To continuously monitor the list of regions, the WRAP bit must be set to one in the last data structure.

Figure 55-3. ICM Region Descriptor and Hash Areas



Each region descriptor supports gathering of data through the use of the Secondary List. Unlike the Main List, the Secondary List cannot modify the configuration attributes of the region. When the end of the Secondary List has been encountered, the ICM returns to the Main List. Memory integrity monitoring can be considered as a background service and the mandatory bandwidth shall be very limited. In order to limit the ICM memory bandwidth, use the BBC field of the ICM_CFG register to control ICM memory load.

Figure 55-4. Region Descriptor

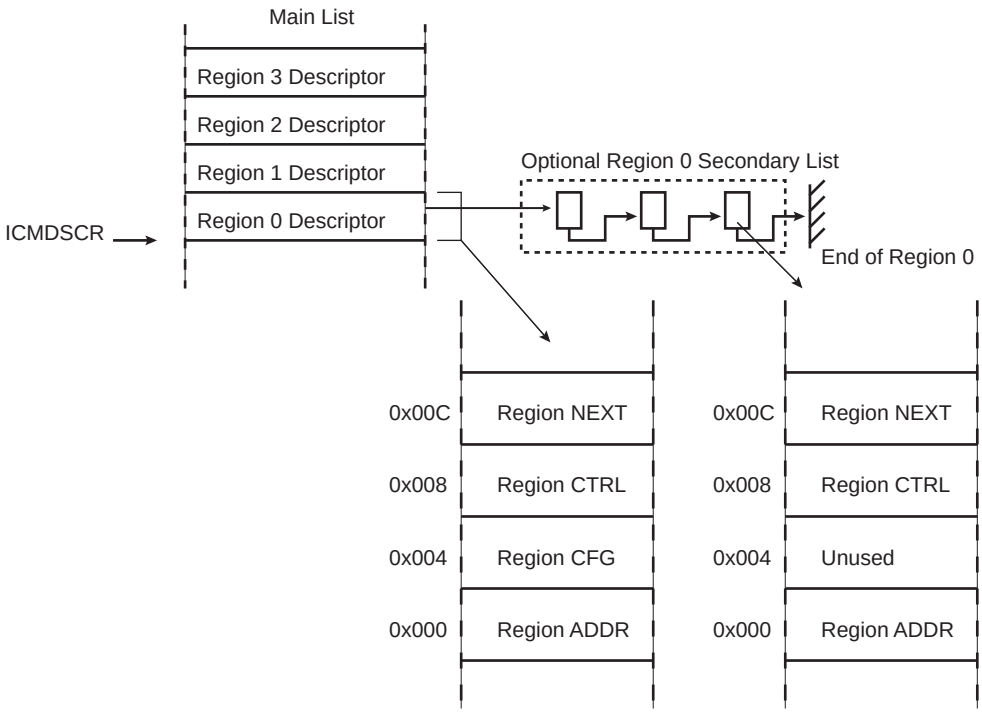
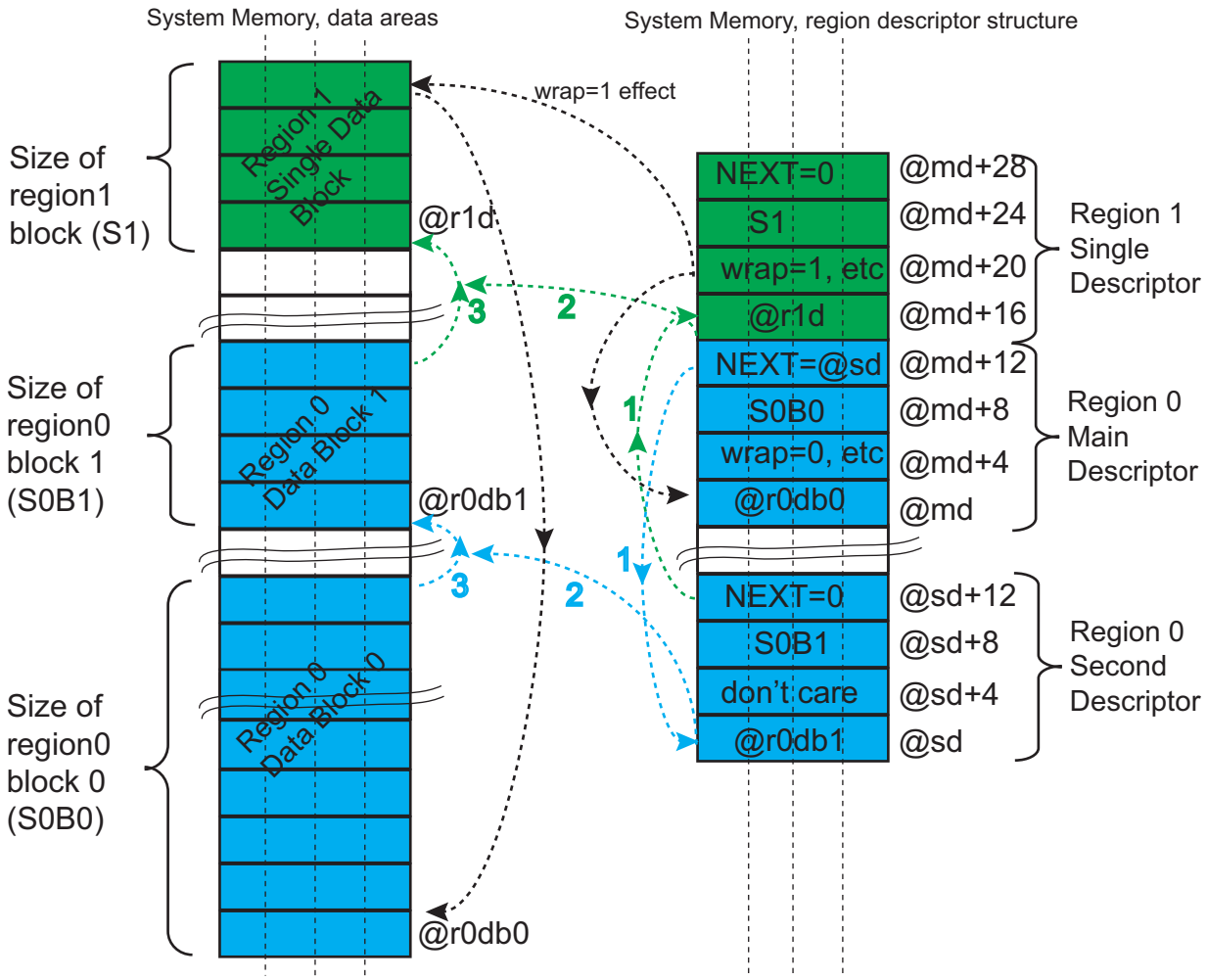


Figure 55-5 shows an example of the mandatory ICM settings to monitor three memory data blocks of the system memory (defined as two regions) with one region being not contiguous (two separate areas) and one contiguous memory area. For each said region, the SHA algorithm may be independently selected (different for each region). The wrap allows continuous monitoring.

Figure 55-5. Example: Monitoring of 3 Memory Data Blocks (Defined as 2 Regions)



55.5.2 ICM Region Descriptor Structure

The ICM Region Descriptor Area is a contiguous area of system memory that the controller and the processor can access. When the ICM controller is activated, the controller performs a descriptor fetch operation at *(ICM_DSCR) address. If the Main List contains more than one descriptor (i.e., more than one region is to be monitored), the fetch address is *(ICM_DSCR) + (RID<<4) where RID is the region identifier.

Table 55-2. Region Descriptor Structure (Main List)

Offset	Structure Member	Name
ICM_DSCR+0x000+RID*(0x10)	ICM Region Start Address	ICM_RADDR
ICM_DSCR+0x004+RID*(0x10)	ICM Region Configuration	ICM_RCFG
ICM_DSCR+0x008+RID*(0x10)	ICM Region Control	ICM_RCTRL
ICM_DSCR+0x00C+RID*(0x10)	ICM Region Next Address	ICM_RNEXT

55.5.2.1 ICM Region Start Address Structure Member

Name: ICM_RADDR

Address: ICM_DSCR+0x000+RID*(0x10)

Access: Read/Write

31	30	29	28	27	26	25	24
RADDR							

23	22	21	20	19	18	17	16
RADDR							

15	14	13	12	11	10	9	8
RADDR							

7	6	5	4	3	2	1	0
RADDR							

- **RADDR: Region Start Address**

This field indicates the first byte address of the region.

55.5.2.2 ICM Region Configuration Structure Member

Name: ICM_RCFG

Address: ICM_DSCR+0x004+RID*(0x10)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	ALGO			–	PROCDLY	SUIEN	ECIEN
7	6	5	4	3	2	1	0
WCIEN	BEIEN	DMIEN	RHIEN	–	EOM	WRAP	CDWBN

- **CDWBN: Compare Digest or Write Back Digest**

0: The digest is written to the Hash area.

1: The digest value is compared to the digest stored in the Hash area.

- **WRAP: Wrap Command**

0: The next region descriptor address loaded is the current region identifier descriptor address incremented by 0x10.

1: The next region descriptor address loaded is ICM_DSCR.

- **EOM: End Of Monitoring**

0: The current descriptor does not terminate the monitoring.

1: The current descriptor terminates the Main List. WRAP bit value has no effect.

- **RHIEN: Region Hash Completed Interrupt Disable (Default Enabled)**

0: The ICM_ISR RHC[*i*] flag is set when the field NEXT = 0 in a descriptor of the main or second list.

1: The ICM_ISR RHC[*i*] flag remains cleared even if the setting condition is met.

- **DMIEN: Digest Mismatch Interrupt Disable (Default Enabled)**

0: The ICM_ISR RBE[*i*] flag is set when the hash value just calculated from the processed region differs from expected hash value.

1: The ICM_ISR RBE[*i*] flag remains cleared even if the setting condition is met.

- **BEIEN: Bus Error Interrupt Disable (Default Enabled)**

0: The flag is set when an error is reported on the system bus by the bus MATRIX.

1: The flag remains cleared even if the setting condition is met.

- **WCIEN: Wrap Condition Interrupt Disable (Default Enabled)**

0: The ICM_ISR RWC[*i*] flag is set when the WRAP bit is set in a descriptor of the main list.

1: The ICM_ISR RWC[*i*] flag remains cleared even if the setting condition is met.

- **ECIEN: End Bit Condition Interrupt (Default Enabled)**

0: The ICM_ISR REC[*i*] flag is set when the descriptor having the EOM bit set is processed.

1: The ICM_ISR REC[*i*] flag remains cleared even if the setting condition is met.

- **SUIEN: Monitoring Status Updated Condition Interrupt (Default Enabled)**

0: The ICM_ISR RSU[*i*] flag is set when the corresponding descriptor is loaded from memory to ICM.

1: The ICM_ISR RSU[*i*] flag remains cleared even if the setting condition is met.

- **PROCDLY: Processing Delay**

Value	Name	Description
0	SHORTEST	SHA processing runtime is the shortest one
1	LONGEST	SHA processing runtime is the longest one

When SHA1 algorithm is processed, the runtime period is either 85 or 209 clock cycles.

When SHA256 or SHA224 algorithm is processed, the runtime period is either 72 or 194 clock cycles.

- **ALGO: SHA Algorithm**

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
4	SHA224	SHA224 algorithm processed

Values which are not listed in the table must be considered as “reserved”.

55.5.2.3 ICM Region Control Structure Member

Name: ICM_RCTRL

Address: ICM_DSCR+0x008+RID*(0x10)

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
TRSIZE							
7	6	5	4	3	2	1	0
TRSIZE							

- **TRSIZE: Transfer Size for the Current Chunk of Data**

55.5.2.4 ICM Region Next Address Structure Member

Name: ICM_RNEXT

Address: ICM_DSCR+0x00C+RID*(0x10)

Access: Read/Write

31	30	29	28	27	26	25	24
NEXT							
23	22	21	20	19	18	17	16
NEXT							
15	14	13	12	11	10	9	8
NEXT							
7	6	5	4	3	2	1	0
NEXT					–	–	–

- **NEXT: Region Transfer Descriptor Next Address**

When configured to 0, this field indicates that the current descriptor is the last descriptor of the Secondary List, otherwise it points at a new descriptor of the Secondary List.

55.5.3 ICM Hash Area

The ICM Hash Area is a contiguous area of system memory that the controller and the processor can access. The physical location is configured in the ICM hash area start address register. This address is a multiple of 128 bytes. If the CDWBN bit of the context register is cleared (i.e., Write Back activated), the ICM controller performs a digest write operation at the following starting location: *(ICM_HASH) + (RID<<5), where RID is the current region context identifier. If the CDWBN bit of the context register is set (i.e., Digest Comparison activated), the ICM controller performs a digest read operation at the same address.

55.5.3.1 Message Digest Example

Considering the following **512 bits message** (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000018”

The message is written to memory in a Little Endian (LE) system architecture.

Table 55-3. 512 bits Message Memory Mapping

Memory Address	Address Offset / Byte Lane							
	0x7 / 63:56	0x6 / 55:48	0x5 / 47:40	0x4 / 39:32	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	00	00	00	00	80	63	62	61
0x008–0x030	00	00	00	00	00	00	00	00
0x038	18	00	00	00	00	00	00	00

The digest is stored at the memory location pointed at by the ICM_HASH pointer with a Region Offset.

Table 55-4. LE Resulting SHA-160 Message Digest Memory Mapping

Memory Address	Address Offset / Byte Lane							
	0x7 / 63:56	0x6 / 55:48	0x5 / 47:40	0x4 / 39:32	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	6a	81	06	47	36	3e	99	a9
0x008	6c	c2	50	78	71	25	3e	ba
0x010	–	–	–	–	9d	d8	d0	9c

Table 55-5. Resulting SHA-256 Message Digest Memory Mapping

Memory Address	Address Offset / Byte Lane							
	0x7 / 63:56	0x6 / 55:48	0x5 / 47:40	0x4 / 39:32	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	ea	cf	01	8f	bf	16	78	ba
0x008	23	22	ae	5d	de	40	41	41
0x010	9c	7a	17	96	a3	61	03	b0
0x018	ad	15	00	f2	61	ff	10	b4

Considering the following **1024 bits message** (example given in FIPS 180-2):

“61626380000000000000000000000000000000000000000000000000000000000

Table 55-6. 1024 bits Message Memory Mapping

Memory Address	Address Offset / Byte Lane							
	0x7 / 63:56	0x6 / 55:48	0x5 / 47:40	0x4 / 39:32	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000	00	00	00	00	80	63	62	61
0x008–0x070	00	00	00	00	00	00	00	00
0x078	18	00	00	00	00	00	00	00

55.5.4 Using ICM as SHA Engine

The ICM can be configured to only calculate a SHA1, SHA224, SHA256 digest value.

55.5.4.1 Settings for Simple SHA Calculation

The start address of the system memory containing the data to hash must be configured in the transfer descriptor of the DMA embedded in the ICM.

The transfer descriptor is a system memory area integer multiple of 4 x 32-bit word and the start address of the descriptor must be configured in ICM_DSCR (the start address must be aligned on 64-bytes; six LSB must be cleared). If the data to hash is already padded according to SHA standards, only a single descriptor is required, and the EOM bit of ICM_RCFG must be written to 1. If the data to hash does not contain a padding area, it is possible to define the padding area in another system memory location, the ICM can be configured to automatically jump from a memory area to another one by configuring the descriptor register ICM_RNEXT with a value that differs from 0. Configuring the field NEXT of the ICM_RNEXT with the start address of the padding area forces the ICM to concatenate both areas, thus providing the SHA result from the start address of the hash area configured in ICM_HASH.

Whether the system memory is configured as a single or multiple data block area, the bits CDWBN and WRAP must be cleared in the region descriptor structure member ICM_RCFG. The bits WBDIS, EOMDIS, SLBDIS must be cleared in ICM_CFG.

The bits RHIEN or ECIEN must be written to 1 in the region descriptor structure member ICM_RCTRL. The flag RHC[*i*], *i* being the region index, is set (if RHIEN is set) when the hash result is available at address defined in ICM_HASH. The flag REC[*i*], *i* being the region index, is set (if ECIEN is set) when the hash result is available at the address defined in ICM_HASH.

An interrupt is generated if the bit RHC[*i*] is written to 1 in the ICM_IER (if RHC[*i*] is set in ICM_RCTRL of region *i*) or if the bit REC[*i*] is written to 1 in the ICM_IER (if REC[*i*] is set in ICM_RCTRL of region *i*).

55.5.4.2 Processing Period

The SHA engine processing period can be configured.

The short processing period allows to allocate bandwidth to the SHA module whereas the long processing period allocates more bandwidth on the system bus to other applications.

In SHA mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA256 and SHA224 modes, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

55.5.5 ICM Automatic Monitoring Mode

The ASCD bit of the ICM_CFG register is used to activate the ICM Automatic mode. When ICM_CFG.ASCD is set, the ICM performs the following actions:

- The ICM controller passes through the Main List once with CDWBN bit in the context register at 0 (i.e., Write Back activated) and EOM bit in context register at 0.
- When WRAP = 1 in ICM_RCFG, the ICM controller enters active monitoring with CDWBN bit in context register now set and EOM bit in context register cleared. Bits CDWBN and EOM in ICM_RCFG have no effect.

55.5.6 Programming the ICM for Multiple Regions

Table 55-7. Region Attributes

Transfer Type		Main List	ICM_RCFG			ICM_RNEXT	Comments
			CDWBN	WRAP	EOM	NEXT	
Single Region	Contiguous list of blocks Digest written to memory Monitoring disabled	1 item	0	0	1	0	The Main List contains only one descriptor. The Secondary List is empty for that descriptor. The digest is computed and saved to memory.
	Non-contiguous list of blocks Digest written to memory Monitoring disabled	1 item	0	0	1	Secondary List address of the current region identifier	The Main List contains only one descriptor. The Secondary List describes the layout of the non-contiguous region.
	Contiguous list of blocks Digest comparison enabled Monitoring enabled	1 item	1	1	0	0	When the hash computation is terminated, the digest is compared with the one saved in memory.
Multiple Regions	Contiguous list of blocks Digest written to memory Monitoring disabled	More than one item	0	0	1 for the last, 0 otherwise	0	ICM passes through the list once.
	Contiguous list of blocks Digest comparison is enabled Monitoring is enabled	More than one item	1	1 for the last, 0 otherwise	0	0	ICM performs active monitoring of the regions. If a mismatch occurs, an interrupt is raised.
	Non-contiguous list of blocks Digest is written to memory Monitoring is disabled	More than one item	0	0	1	Secondary List address	ICM performs hashing and saves digests to the Hash area.
	Non-contiguous list of blocks Digest comparison is enabled Monitoring is enabled	More than one item	1	1	0	Secondary List address	ICM performs data gathering on a per region basis.

55.5.7 Security Features

When an undefined register access occurs, the URAD bit in the Interrupt Status Register (ICM_ISR) is set if unmasked. Its source is then reported in the Undefined Access Status Register (ICM_UASR). Only the first undefined register access is available through the ICM_UASR.URAT field.

Several kinds of unspecified register accesses can occur:

- Unspecified structure member set to one detected when the descriptor is loaded
- Configuration register (ICM_CFG) modified during active monitoring
- Descriptor register (ICM_DSCR) modified during active monitoring
- Hash register (ICM_HASH) modified during active monitoring
- Write-only register read access

The URAD bit and the URAT field can only be reset by writing a 1 to the ICM_CTRL.SWRST bit.

55.6 Integrity Check Monitor (ICM) User Interface

Table 55-8. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Configuration Register	ICM_CFG	Read/Write	0x0
0x04	Control Register	ICM_CTRL	Write-only	–
0x08	Status Register	ICM_SR	Read-only	–
0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	ICM_IER	Write-only	–
0x14	Interrupt Disable Register	ICM_IDR	Write-only	–
0x18	Interrupt Mask Register	ICM_IMR	Read-only	0x0
0x1C	Interrupt Status Register	ICM_ISR	Read-only	0x0
0x20	Undefined Access Status Register	ICM_UASR	Read-only	0x0
0x24–0x2C	Reserved	–	–	–
0x30	Region Descriptor Area Start Address Register	ICM_DSCR	Read/Write	0x0
0x34	Region Hash Area Start Address Register	ICM_HASH	Read/Write	0x0
0x38	User Initial Hash Value 0 Register	ICM_UIHVAL0	Write-only	–
...	...	...	...	...
0x54	User Initial Hash Value 7	ICM_UIHVAL7	Write-only	–
0x78–0xE8	Reserved	–	–	–
0xEC–0xFC	Reserved	–	–	–

55.6.1 ICM Configuration Register

Name: ICM_CFG

Address: 0xF8040000

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
UALGO			UIHASH	–	–	DUALBUFF	ASCD
7	6	5	4	3	2	1	0
BBC				–	SLBDIS	EOMDIS	WBDIS

- **WBDIS: Write Back Disable**

0: Write Back Operations are permitted.

1: Write Back Operations are forbidden. Context register CDWBN bit is internally set to one and cannot be modified by a linked list element. The CDWBN bit of the ICM_RCFG structure member has no effect.

When ASCD bit of the ICM_CFG register is set, WBDIS bit value has no effect.

- **EOMDIS: End of Monitoring Disable**

0: End of Monitoring is permitted

1: End of Monitoring is forbidden. The EOM bit of the ICM_RCFG structure member has no effect.

- **SLBDIS: Secondary List Branching Disable**

0: Branching to the Secondary List is permitted.

1: Branching to the Secondary List is forbidden. The NEXT field of the ICM_RNEXT structure member has no effect and is always considered as zero.

- **BBC: Bus Burden Control**

This field is used to control the burden of the ICM system bus. The number of system clock cycles between the end of the current processing and the next block transfer is set to 2^{BBC}. Up to 32,768 cycles can be inserted.

- **ASCD: Automatic Switch To Compare Digest**

0: Automatic mode is disabled.

1: When this mode is enabled, the ICM controller automatically switches to active monitoring after the first Main List pass. Both CDWBN and WBDIS bits have no effect. A one must be written to the EOM bit in ICM_RCFG to terminate the monitoring.

- **DUALBUFF: Dual Input Buffer**

0: Dual Input Buffer mode is disabled.

1: Dual Input Buffer mode is enabled (better performances, higher bandwidth required on system bus).

- **UIHASH: User Initial Hash Value**

0: The secure hash standard provides the initial hash value.

1: The initial hash value is programmable. Field UALGO provides the SHA algorithm. The ALGO field of the ICM_RCFG structure member has no effect.

- **UALGO: User SHA Algorithm**

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
4	SHA224	SHA224 algorithm processed

55.6.2 ICM Control Register

Name: ICM_CTRL

Address: 0xF8040004

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RMEN				RMDIS			
7	6	5	4	3	2	1	0
REHASH				–	SWRST	DISABLE	ENABLE

- **ENABLE: ICM Enable**

0: No effect

1: When set to one, the ICM controller is activated.

- **DISABLE: ICM Disable Register**

0: No effect

1: The ICM controller is disabled. If a region is active, this region is terminated.

- **SWRST: Software Reset**

0: No effect

1: Resets the ICM controller.

- **REHASH: Recompute Internal Hash**

0: No effect

1: When REHASH[*i*] is set to one, Region *i* digest is re-computed. This bit is only available when region monitoring is disabled.

- **RMDIS: Region Monitoring Disable**

0: No effect

1: When bit RMDIS[*i*] is set to one, the monitoring of region with identifier *i* is disabled.

- **RMEN: Region Monitoring Enable**

0: No effect

1: When bit RMEN[*i*] is set to one, the monitoring of region with identifier *i* is activated.

Monitoring is activated by default.

55.6.3 ICM Status Register

Name: ICM_SR

Address: 0xF8040008

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
RMDIS				RAWRMDIS			
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: ICM Controller Enable Register**

0: ICM controller is disabled

1: ICM controller is activated

- **RAWRMDIS: Region Monitoring Disabled Raw Status**

0: Region *i* monitoring has been activated by writing a 1 in RMEN[*i*] of ICM_CTRL

1: Region *i* monitoring has been deactivated by writing a 1 in RMDIS[*i*] of ICM_CTRL

- **RMDIS: Region Monitoring Disabled Status**

0: Region *i* is being monitored (occurs after integrity check value has been calculated and written to Hash area)

1: Region *i* monitoring is not being monitored

55.6.4 ICM Interrupt Enable Register

Name: ICM_IER

Address: 0xF8040010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD
23	22	21	20	19	18	17	16
RSU				REC			
15	14	13	12	11	10	9	8
RWC				RBE			
7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed Interrupt Enable**

0: No effect

1: When RHC[*i*] is set to one, the Region *i* Hash Completed interrupt is enabled.

- **RDM: Region Digest Mismatch Interrupt Enable**

0: No effect

1: When RDM[*i*] is set to one, the Region *i* Digest Mismatch interrupt is enabled.

- **RBE: Region Bus Error Interrupt Enable**

0: No effect

1: When RBE[*i*] is set to one, the Region *i* Bus Error interrupt is enabled.

- **RWC: Region Wrap Condition detected Interrupt Enable**

0: No effect

1: When RWC[*i*] is set to one, the Region *i* Wrap Condition interrupt is enabled.

- **REC: Region End bit Condition Detected Interrupt Enable**

0: No effect

1: When REC[*i*] is set to one, the region *i* End bit Condition interrupt is enabled.

- **RSU: Region Status Updated Interrupt Disable**

0: No effect

1: When RSU[*i*] is set to one, the region *i* Status Updated interrupt is enabled.

- **URAD: Undefined Register Access Detection Interrupt Enable**

0: No effect

1: The Undefined Register Access interrupt is enabled.

55.6.5 ICM Interrupt Disable Register

Name: ICM_IDR

Address: 0xF8040014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD

23	22	21	20	19	18	17	16
RSU				REC			

15	14	13	12	11	10	9	8
RWC				RBE			

7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed Interrupt Disable**

0: No effect

1: When RHC[*i*] is set to one, the Region *i* Hash Completed interrupt is disabled.

- **RDM: Region Digest Mismatch Interrupt Disable**

0: No effect

1: When RDM[*i*] is set to one, the Region *i* Digest Mismatch interrupt is disabled.

- **RBE: Region Bus Error Interrupt Disable**

0: No effect

1: When RBE[*i*] is set to one, the Region *i* Bus Error interrupt is disabled.

- **RWC: Region Wrap Condition Detected Interrupt Disable**

0: No effect

1: When RWC[*i*] is set to one, the Region *i* Wrap Condition interrupt is disabled.

- **REC: Region End bit Condition detected Interrupt Disable**

0: No effect

1: When REC[*i*] is set to one, the region *i* End bit Condition interrupt is disabled.

- **RSU: Region Status Updated Interrupt Disable**

0: No effect

1: When RSU[*i*] is set to one, the region *i* Status Updated interrupt is disabled.

- **URAD: Undefined Register Access Detection Interrupt Disable**

0: No effect

1: Undefined Register Access Detection interrupt is disabled.

55.6.6 ICM Interrupt Mask Register

Name: ICM_IMR

Address: 0xF8040018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD

23	22	21	20	19	18	17	16
RSU				REC			

15	14	13	12	11	10	9	8
RWC				RBE			

7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed Interrupt Mask**

0: When RHC[*i*] is set to zero, the interrupt is disabled for region i.

1: When RHC[*i*] is set to one, the interrupt is enabled for region i.

- **RDM: Region Digest Mismatch Interrupt Mask**

0: When RDM[*i*] is set to zero, the interrupt is disabled for region i.

1: When RDM[*i*] is set to one, the interrupt is enabled for region i.

- **RBE: Region Bus Error Interrupt Mask**

0: When RBE[*i*] is set to zero, the interrupt is disabled for region i.

1: When RBE[*i*] is set to one, the interrupt is enabled for region i.

- **RWC: Region Wrap Condition Detected Interrupt Mask**

0: When RWC[*i*] is set to zero, the interrupt is disabled for region i.

1: When RWC[*i*] is set to one, the interrupt is enabled for region i.

- **REC: Region End bit Condition Detected Interrupt Mask**

0: When REC[*i*] is set to zero, the interrupt is disabled for region i.

1: When REC[*i*] is set to one, the interrupt is enabled for region i.

- **RSU: Region Status Updated Interrupt Mask**

0: When RSU[*i*] is set to zero, the interrupt is disabled for region i.

1: When RSU[*i*] is set to one, the interrupt is enabled for region i.

- **URAD: Undefined Register Access Detection Interrupt Mask**

0: Interrupt is disabled

1: Interrupt is enabled.

55.6.7 ICM Interrupt Status Register

Name: ICM_ISR

Address: 0xF804001C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	URAD

23	22	21	20	19	18	17	16
RSU				REC			

15	14	13	12	11	10	9	8
RWC				RBE			

7	6	5	4	3	2	1	0
RDM				RHC			

- **RHC: Region Hash Completed**

When RHC[*i*] is set, it indicates that the ICM has completed the region with identifier *i*.

- **RDM: Region Digest Mismatch**

When RDM[*i*] is set, it indicates that there is a digest comparison mismatch between the hash value of the region with identifier *i* and the reference value located in the Hash Area.

- **RBE: Region Bus Error**

When RBE[*i*] is set, it indicates that a bus error has been detected while hashing memory region *i*.

- **RWC: Region Wrap Condition Detected**

When RWC[*i*] is set, it indicates that a wrap condition has been detected.

- **REC: Region End bit Condition Detected**

When REC[*i*] is set, it indicates that an end bit condition has been detected.

- **RSU: Region Status Updated Detected**

When RSU[*i*] is set, it indicates that a region status updated condition has been detected.

- **URAD: Undefined Register Access Detection Status**

0: No undefined register access has been detected since the last SWRST.

1: At least one undefined register access has been detected since the last SWRST.

The URAD bit is only reset by the SWRST bit in the ICM_CTRL register.

The URAT field in the ICM_UASR indicates the unspecified access type.

55.6.8 ICM Undefined Access Status Register

Name: ICM_UASR

Address: 0xF8040020

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	URAT		

- **URAT: Undefined Register Access Trace**

Value	Name	Description
0	UNSPEC_STRUCT_MEMBER	Unspecified structure member set to one detected when the descriptor is loaded.
1	ICM_CFG_MODIFIED	ICM_CFG modified during active monitoring.
2	ICM_DSCR_MODIFIED	ICM_DSCR modified during active monitoring.
3	ICM_HASH_MODIFIED	ICM_HASH modified during active monitoring
4	READ_ACCESS	Write-only register read access

Only the first Undefined Register Access Trace is available through the URAT field.

The URAT field is only reset by the SWRST bit in the ICM_CTRL register.

55.6.9 ICM Descriptor Area Start Address Register

Name: ICM_DSCR

Address: 0xF8040030

Access: Read/Write

31	30	29	28	27	26	25	24
DASA							
23	22	21	20	19	18	17	16
DASA							
15	14	13	12	11	10	9	8
DASA							
7	6	5	4	3	2	1	0
DASA		–	–	–	–	–	–

- **DASA: Descriptor Area Start Address**

The start address is a multiple of the total size of the data structure (64 bytes).

55.6.10 ICM Hash Area Start Address Register

Name: ICM_HASH

Address: 0xF8040034

Access: Read/Write

31	30	29	28	27	26	25	24
HASA							
23	22	21	20	19	18	17	16
HASA							
15	14	13	12	11	10	9	8
HASA							
7	6	5	4	3	2	1	0
HASA	–	–	–	–	–	–	–

- **HASA: Hash Area Start Address**

This field points at the Hash memory location. The address must be a multiple of 128 bytes.

55.6.11 ICM User Initial Hash Value Register

Name: ICM_UIHVALx [x=0..7]

Address: 0xF8040038

Access: Write-only

31	30	29	28	27	26	25	24
VAL							
23	22	21	20	19	18	17	16
VAL							
15	14	13	12	11	10	9	8
VAL							
7	6	5	4	3	2	1	0
VAL							

- **VAL: Initial Hash Value**

When UIHASH bit of IMC_CFG register is set, the Initial Hash Value is user-programmable.

To meet the desired standard, use the following example values.

For ICM_UIHVAL0 field:

Example	Comment
0x67452301	SHA1 algorithm
0xC1059ED8	SHA224 algorithm
0x6A09E667	SHA256 algorithm

For ICM_UIHVAL1 field:

Example	Comment
0xEFCDAB89	SHA1 algorithm
0x367CD507	SHA224 algorithm
0xBB67AE85	SHA256 algorithm

For ICM_UIHVAL2 field:

Example	Comment
0x98BADCFE	SHA1 algorithm
0x3070DD17	SHA224 algorithm
0x3C6EF372	SHA256 algorithm

For ICM_UIHVAL3 field:

Example	Comment
0x10325476	SHA1 algorithm
0xF70E5939	SHA224 algorithm
0xA54FF53A	SHA256 algorithm

For ICM_UIHVAL4 field:

Example	Comment
0xC3D2E1F0	SHA1 algorithm
0xFFC00B31	SHA224 algorithm
0x510E527F	SHA256 algorithm

For ICM_UIHVAL5 field:

Example	Comment
0x68581511	SHA224 algorithm
0x9B05688C	SHA256 algorithm

For ICM_UIHVAL6 field:

Example	Comment
0x64F98FA7	SHA224 algorithm
0x1F83D9AB	SHA256 algorithm

For ICM_UIHVAL7 field:

Example	Comment
0xBEFA4FA4	SHA224 algorithm
0x5BE0CD19	SHA256 algorithm

Example of Initial Value for SHA-1 Algorithm

Register Address	Address Offset / Byte Lane			
	0x3 / 31:24	0x2 / 23:16	0x1 / 15:8	0x0 / 7:0
0x000 ICM_UIHVAL0	01	23	45	67
0x004 ICM_UIHVAL1	89	ab	cd	ef
0x008 ICM_UIHVAL2	fe	dc	ba	98
0x00C ICM_UIHVAL3	76	54	32	10
0x010 ICM_UIHVAL4	f0	e1	d2	c3

56. Advanced Encryption Standard Bridge (AESB)

56.1 Description

The Advanced Encryption Standard Bridge (AESB) is intended to provide on-the-fly off-chip memory encryption/decryption compliant with the American *FIPS (Federal Information Processing Standard) Publication 197* specification.

The AESB supports three confidentiality modes of operation for symmetrical key block cipher algorithms (ECB, CBC and CTR), as specified in the *NIST Special Publication 800-38A Recommendation*.

The 128-bit key is stored in four 32-bit registers (AESB_KEYWRx) which are all write-only.

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit registers (AESB_IDATARx and AESB_IVRx) which are all write-only.

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data will be ready to be read out on the four 32-bit output data registers (AESB_ODATARx).

56.2 Embedded Characteristics

- On-the-fly off-chip memory encryption/decryption
- Compliant with *FIPS Publication 197, Advanced Encryption Standard (AES)*
- 128-bit cryptographic key
- On-The-Fly encryption/decryption
- 12 clock cycles encryption/decryption processing time with a 128-bit cryptographic key
- Double input buffer optimizes runtime
- Support of the three standard modes of operation specified in the *NIST Special Publication 800-38A, Recommendation for Block Cipher Modes of Operation - Methods and Techniques:*
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Counter (CTR)
- Last Output Data mode allows optimized Message Authentication Code (MAC) generation

56.3 Product Dependencies

56.3.1 Power Management

The AESB may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the AESB clock.

56.3.2 Interrupt

The AESB interface has an interrupt line connected to the Interrupt Controller.

Handling the AESB interrupt requires programming the Interrupt Controller before configuring the AESB.

Table 56-1. Peripheral IDs

Instance	ID
AESB	10

56.4 Functional Description

The Advanced Encryption Standard Bridge (AESB) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in the AESB Mode Register (AESB_MR) allows selection between the encryption and the decryption processes.

The AESB is capable of using cryptographic keys of 128 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit key is defined in the Key Registers (AESB_KEYWRx).

The input to the encryption processes of the CBC mode includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in the Initialization Vector Registers (AESB_IVRx). The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. The Initialization Vector Registers are also used by the CTR mode to set the counter value.

56.4.1 Operating Modes

The AESB supports the following modes of operation:

- ECB—Electronic Code Book
- CBC—Cipher Block Chaining
- CTR—Counter

The data preprocessing, post-processing and data chaining for the operating modes are performed automatically. Refer to the *NIST Special Publication 800-38A Recommendation* for more complete information.

The modes are selected by the OPMOD field in AESB_MR.

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 megabyte of data. If the file to be processed is greater than 1 megabyte, this file must be split into fragments of 1 megabyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AESB_IDATARx registers, the AESB_IVRx registers must be cleared. For any fragment, after the transfer is completed and prior to transferring the next fragment, AESB_IVR0 must be programmed so that the fragment number (0 for the first fragment, 1 for the second one, and so on) is written in the 16 MSB of AESB_IVR0.

If the initial value of the counter is greater than 0 and the data buffer size to be processed is greater than 1 megabyte, the size of the first fragment to be processed must be 1 megabyte minus 16x(initial value) to prevent a rollover of the internal 1-bit counter.

56.4.2 Double Input Buffer

The input data register can be double-buffered to reduce the runtime of large files.

This mode allows writing a new message block when the previous message block is being processed.

The DUALBUFF bit in register AESB_MR must be set to 1 to access the double buffer.

56.4.3 Start Modes

The SMOD field in register AESB_MR allows selection of the Encryption (or Decryption) Start mode.

56.4.3.1 Manual Mode

The sequence is as follows:

1. Write AESB_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit key in the Key Registers (AESB_KEYWRx).

3. Write the initialization vector (or counter) in the Initialization Vector Registers (AESB_IVRx).

Note: The Initialization Vector Registers concern all modes except ECB.

4. Set the DATRDY (Data Ready) bit in the AESB Interrupt Enable Register (AESB_IER) depending on whether an interrupt is required, or not, at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized Input Data Registers (see Table 56-2).

Table 56-2. Authorized Input Data Registers

Operating Mode	Input Data Registers to Write
ECB	All
CBC	All
CTR	All

6. Set the START bit in the AESB Control Register (AESB_CR) to begin the encryption or decryption process.
7. When processing is complete, the DATRDY bit in the AESB Interrupt Status Register (AESB_ISR) raises. If an interrupt has been enabled by setting the DATRDY bit in AESB_IER, the interrupt line of the AESB is activated.
8. When the software reads one of the Output Data Registers (AESB_ODATARx), the AESB_ISR.DATRDY bit is automatically cleared.

56.4.3.2 Auto Mode

Auto mode is similar to Manual mode, except that in Auto mode, as soon as the correct number of Input Data registers is written, processing starts automatically without any action in the Control Register.

56.4.4 Last Output Data Mode

Last Output Data mode is used to generate cryptographic checksums on data (MAC) by means of a cipher block chaining encryption algorithm (the CBC-MAC algorithm for example).

After each end of encryption/decryption, the output data are available on the output data registers for Manual and Auto modes.

The Last Output Data (LOD) bit in AESB_MR allows retrieval of only the last data of several encryption/decryption processes.

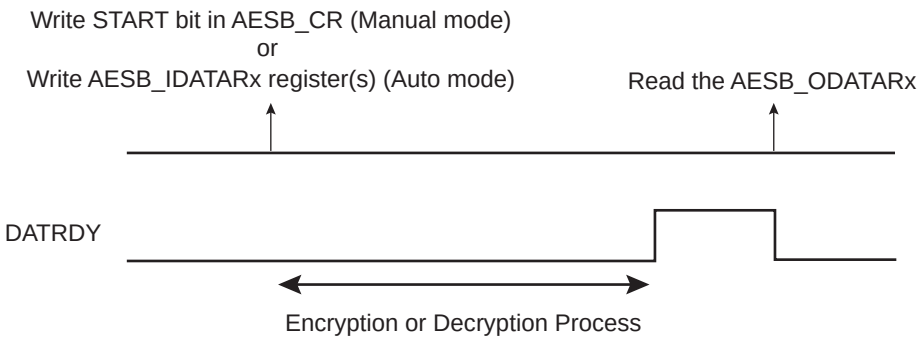
Those data are only available on the Output Data Registers (AESB_ODATARx).

56.4.5 Manual and Auto Modes

56.4.5.1 If AESB_MR.LOD = 0

The AESB_ISR.DATRDY bit is cleared when at least one of the Output Data Registers is read (see Figure 56-1).

Figure 56-1. Manual and Auto Modes with AESB_MR.LOD = 0

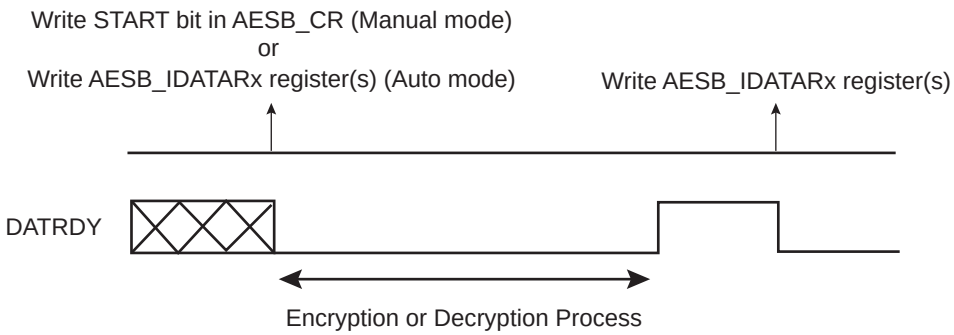


If the user does not want to read the output data registers between each encryption/decryption, the AESB_ISR.DATRDY bit will not be cleared. If the AESB_ISR.DATRDY bit is not cleared, the user cannot know the end of the following encryptions/decryptions.

56.4.5.2 If AESB_MR.LOD = 1

The AESB_ISR.DATRDY bit is cleared when at least one Input Data Register is written, so before the start of a new transfer (see Figure 56-2). No more Output Data Register reads are necessary between consecutive encryptions/decryptions.

Figure 56-2. Manual and Auto Modes with AESB_MR.LOD = 1



56.4.6 Automatic Bridge Mode

56.4.6.1 Description

The Automatic Bridge mode, when the AESB block is connected between the system bus and a DDR port, provides automatic encryption/decryption to/from a DDR port without any action on the part of the user. For Automatic Bridge mode, the OPMODE field must be configured to 0x4 in AESB_MR (see Section 56.6.2 “AESB Mode Register”). If bit AESB_MR.AAHB is set and field AESB_MR.OPMODE = 0x4, there is no compliance with the standard CTR mode of operation.

In case of write transfer, this mode automatically encrypts the data before writing it to the final slave destination. In case of read transfer, this mode automatically decrypts the data read from the target slave before putting it on the system bus.

Therefore, this mode does not work if the automatically encrypted data is moved at another address outside of the AESB IP scope. This means that for a given data, the encrypted value is not the same if written at different addresses.

56.4.6.2 Configuration

The Automatic Bridge mode can be enabled by setting bit AESB_MR.AAHB.

The IV (Initialization Vector) field of the AESB Initialization Vector Register x (AESB_IVRx) can be used to add a nonce in the encryption process in order to bring even more security (ignored if not filled). In this case, any value encrypted with a given nonce can only be decrypted with this nonce. If another nonce is set for the IV field, any value encrypted with the previous nonce cannot be decrypted anymore (see Section 56.6.10 “AESB Initialization Vector Register x”).

Dual buffer usage (write a 1 to bit AESB_MR.DUALBUFF) is recommended for improved performance.

56.5 Security Features

56.5.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in AESB_ISR raises. Its source is then reported in the Unspecified Register Access Type (URAT) field. Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data Register written during the data processing when SMOD = IDATAR0_START
- Output Data Register read during data processing
- Mode Register written during data processing
- Output Data Register read during subkeys generation
- Mode Register written during subkeys generation
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in AESB_CR.

56.6 Advanced Encryption Standard Bridge (AESB) User Interface

Table 56-3. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	AESB_CR	Write-only	–
0x04	Mode Register	AESB_MR	Read/Write	0x0
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	AESB_IER	Write-only	–
0x14	Interrupt Disable Register	AESB_IDR	Write-only	–
0x18	Interrupt Mask Register	AESB_IMR	Read-only	0x0
0x1C	Interrupt Status Register	AESB_ISR	Read-only	0x0
0x20	Key Word Register 0	AESB_KEYWR0	Write-only	–
0x24	Key Word Register 1	AESB_KEYWR1	Write-only	–
0x28	Key Word Register 2	AESB_KEYWR2	Write-only	–
0x2C	Key Word Register 3	AESB_KEYWR3	Write-only	–
0x30–0x3C	Reserved	–	–	–
0x40	Input Data Register 0	AESB_IDATAR0	Write-only	–
0x44	Input Data Register 1	AESB_IDATAR1	Write-only	–
0x48	Input Data Register 2	AESB_IDATAR2	Write-only	–
0x4C	Input Data Register 3	AESB_IDATAR3	Write-only	–
0x50	Output Data Register 0	AESB_ODATAR0	Read-only	0x0
0x54	Output Data Register 1	AESB_ODATAR1	Read-only	0x0
0x58	Output Data Register 2	AESB_ODATAR2	Read-only	0x0
0x5C	Output Data Register 3	AESB_ODATAR3	Read-only	0x0
0x60	Initialization Vector Register 0	AESB_IVR0	Write-only	–
0x64	Initialization Vector Register 1	AESB_IVR1	Write-only	–
0x68	Initialization Vector Register 2	AESB_IVR2	Write-only	–
0x6C	Initialization Vector Register 3	AESB_IVR3	Write-only	–
0x70–0xFC	Reserved	–	–	–

56.6.1 AESB Control Register

Name: AESB_CR

Address: 0xF001C000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SWRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	START

- **START: Start Processing**

0: No effect

1: Starts manual encryption/decryption process

- **SWRST: Software Reset**

0: No effect

1: Resets the AESB. A software triggered hardware reset of the AESB interface is performed.

56.6.2 AESB Mode Register

Name: AESB_MR

Address: 0xF001C004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CKEY				–	–	–	–
15	14	13	12	11	10	9	8
LOD	OPMOD			–	–	SMOD	
7	6	5	4	3	2	1	0
PROCDLY				DUALBUFF	AAHB	–	CIPHER

- **CIPHER: Processing Mode**

0: Decrypts data

1: Encrypts data

- **AAHB: Automatic Bridge Mode**

0: Automatic Bridge mode disabled

1: Automatic Bridge mode enabled

- **DUALBUFF: Dual Input Buffer**

Value	Name	Description
0x0	INACTIVE	AESB_IDATARx cannot be written during processing of previous block.
0x1	ACTIVE	AESB_IDATARx can be written during processing of previous block when SMOD = 0x2. It speeds up the overall runtime of large files.

- **PROCDLY: Processing Delay**

Processing Time = 12 × (PROCDLY + 1)

The Processing Time represents the number of clock cycles that the AESB needs in order to perform one encryption/decryption .

Note: The best performance is achieved with PROCDLY equal to 0.

- **SMOD: Start Mode**

Value	Name	Description
0x0	MANUAL_START	Manual mode
0x1	AUTO_START	Auto mode
0x2	IDATAR0_START	AESB_IDATAR0 access only Auto mode

Values which are not listed in the table must be considered as “reserved”.

- **OPMOD: Operating Mode**

Value	Name	Description
0x0	ECB	Electronic Code Book mode
0x1	CBC	Cipher Block Chaining mode
0x2	–	Reserved
0x3	–	Reserved
0x4	CTR	Counter mode (16-bit internal counter)

Values which are not listed in the table must be considered as “reserved”.

For CBC-MAC operating mode, configure OPMOD to 0x1 (CBC) and set LOD to 1.

Note: If the OPMODE field is set to 0x4 and AAHB = 1, there is no compliance with the standard CTR mode of operation.

- **LOD: Last Output Data Mode**

0: No effect.

After each end of encryption/decryption, the output data will be available either on the output data registers (Manual and Auto modes).

In Manual and Auto modes, the AESB_ISR.DATRDY bit is cleared when at least one of the Output Data registers is read.

1: The AESB_ISR.DATRDY bit is cleared when at least one of the Input Data Registers is written.

No more Output Data Register reads are necessary between consecutive encryptions/decryptions (see Section 56.4.4 “Last Output Data Mode”).

- **CKEY: Key**

Value	Name	Description
0xE	PASSWD	This field must be written with 0xE the first time that AES_MR is programmed. For subsequent programming of the AES_MR register, any value can be written, including that of 0xE.Always reads as 0.

56.6.3 AESB Interrupt Enable Register

Name: AESB_IER

Address: 0xF001C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**

56.6.4 AESB Interrupt Disable Register

Name: AESB_IDR

Address: 0xF001C014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Disable**
- **URAD: Unspecified Register Access Detection Interrupt Disable**

56.6.5 AESB Interrupt Mask Register

Name: AESB_IMR

Address: 0xF001C018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **DATRDY: Data Ready Interrupt Mask**
- **URAD: Unspecified Register Access Detection Interrupt Mask**

56.6.6 AESB Interrupt Status Register

Name: AESB_ISR

Address: 0xF001C01C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
URAT				–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: Output data not valid.

1: Encryption or decryption process is completed.

DATRDY is cleared when a Manual encryption/decryption occurs (START bit in AESB_CR) or when a software triggered hardware reset of the AESB interface is performed (SWRST bit in AESB_CR).

AESB_MR.LOD = 0: In Manual and Auto modes, the DATRDY bit can also be cleared when at least one of the Output Data Registers is read.

AESB_MR.LOD = 1: In Manual and Auto modes, the DATRDY bit can also be cleared when at least one of the Input Data Registers is written.

- **URAD: Unspecified Register Access Detection Status**

0: No unspecified register access has been detected since the last SWRST.

1: At least one unspecified register access has been detected since the last SWRST.

URAD bit is reset only by the SWRST bit in AESB_CR.

- **URAT: Unspecified Register Access**

Value	Name	Description
0x0	IDR_WR_PROCESSING	Input Data Register written during the data processing when SMOD = 0x2 mode
0x1	ODR_RD_PROCESSING	Output Data Register read during the data processing
0x2	MR_WR_PROCESSING	Mode Register written during the data processing
0x3	ODR_RD_SUBKGEN	Output Data Register read during the subkeys generation
0x4	MR_WR_SUBKGEN	Mode Register written during the subkeys generation
0x5	WOR_RD_ACCESS	Write-only register read access

Only the last Unspecified Register Access Type is available through the URAT field.

URAT field is reset only by the SWRST bit in AESB_CR.

56.6.7 AESB Key Word Register x

Name: AESB_KEYWRx

Address: 0xF001C020

Access: Write-only

31	30	29	28	27	26	25	24
KEYW							

23	22	21	20	19	18	17	16
KEYW							

15	14	13	12	11	10	9	8
KEYW							

7	6	5	4	3	2	1	0
KEYW							

- **KEYW: Key Word**

The four 32-bit Key Word registers set the 128-bit cryptographic key used for encryption/decryption.

AESB_KEYWR0 corresponds to the first word of the key, AESB_KEYWR3 to the last one.

These registers are write-only to prevent the key from being read by another application.

56.6.8 AESB Input Data Register x

Name: AESB_IDATARx

Address: 0xF001C040

Access: Write-only

31	30	29	28	27	26	25	24
IDATA							
23	22	21	20	19	18	17	16
IDATA							
15	14	13	12	11	10	9	8
IDATA							
7	6	5	4	3	2	1	0
IDATA							

- **IDATA: Input Data Word**

The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.

AESB_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, AESB_IDATAR3 to the last one.

These registers are write-only to prevent the input data from being read by another application.

56.6.9 AESB Output Data Register x

Name: AESB_ODATARx

Address: 0xF001C050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							

23	22	21	20	19	18	17	16
ODATA							

15	14	13	12	11	10	9	8
ODATA							

7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted.

AESB_ODATAR0 corresponds to the first word, AESB_ODATAR3 to the last one.

56.6.10 AESB Initialization Vector Register x

Name: AESB_IVRx

Address: 0xF001C060

Access: Write-only

31	30	29	28	27	26	25	24
IV							

23	22	21	20	19	18	17	16
IV							

15	14	13	12	11	10	9	8
IV							

7	6	5	4	3	2	1	0
IV							

- **IV: Initialization Vector**

The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.

AESB_IVR0 corresponds to the first word of the Initialization Vector, AESB_IVR3 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

For CBC mode, the IV input value corresponds to the initialization vector.

For CTR mode, the IV input value corresponds to the initial counter value.

Note: These registers are not used in ECB mode and must not be written. For Automatic Bridge dedicated mode, the IV input value corresponds to the initial nonce.

57. Advanced Encryption Standard (AES)

57.1 Description

The Advanced Encryption Standard (AES) is compliant with the American *FIPS (Federal Information Processing Standard) Publication 197* specification.

The AES supports all five confidentiality modes of operation for symmetrical key block cipher algorithms (ECB, CBC, OFB, CFB and CTR), as specified in the *NIST Special Publication 800-38A Recommendation,* as well as Galois/Counter Mode (GCM) as specified in the *NIST Special Publication 800-38D Recommendation*. It is compatible with all these modes via DMA Controller channels, minimizing processor intervention for large buffer transfers.

The 128-bit/192-bit/256-bit key is stored in four/six/eight 32-bit write-only AES Key Word registers (AES_KEYWR0–3).

The 128-bit input data and initialization vector (for some modes) are each stored in four 32-bit write-only AES Input Data registers (AES_IDATAR0–3) and AES Initialization Vector registers (AES_IVR0–3).

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data are ready to be read out on the four 32-bit AES Output Data registers (AES_ODATAR0–3) or through the DMA channels.

57.2 Embedded Characteristics

- Compliant with *FIPS Publication 197, Advanced Encryption Standard (AES)*
- 128-bit/192-bit/256-bit Cryptographic Key
- 12/14/16 Clock Cycles Encryption/Decryption Processing Time with a 128-bit/192-bit/256-bit Cryptographic Key
- Double Input Buffer Optimizes Runtime
- Automatic Padding supported for IPSEC and SSL standards
- IPSEC and SSL Protocol Layers Improved Performances (Tightly coupled with SHA)
- Support of the Modes of Operation Specified in the *NIST Special Publication 800-38A* and *NIST Special Publication 800-38D*:
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC) including CBC-MAC
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
  - Counter (CTR)
  - Galois/Counter Mode (GCM)
  - XEX-Based Tweaked-Codebook Mode (XTS)
- 8, 16, 32, 64 and 128-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allows Optimized Message Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

57.3 Product Dependencies

57.3.1 Power Management

The AES may be clocked through the Power Management Controller (PMC), so the programmer must first to configure the PMC to enable the AES clock.

57.3.2 Interrupt Sources

The AES interface has an interrupt line connected to the Interrupt Controller.

Handling the AES interrupt requires programming the Interrupt Controller before configuring the AES.

Table 57-1. Peripheral IDs

Instance	ID
AES	9

57.4 Functional Description

The Advanced Encryption Standard (AES) specifies a FIPS-approved cryptographic algorithm that can be used to protect electronic data. The AES algorithm is a symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information.

Encryption converts data to an unintelligible form called ciphertext. Decrypting the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in the AES Mode register (AES_MR) allows selection between the encryption and the decryption processes.

The AES is capable of using cryptographic keys of 128/192/256 bits to encrypt and decrypt data in blocks of 128 bits. This 128-bit/192-bit/256-bit key is defined in the AES_KEYWRx.

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 128-bit data block called the initialization vector (IV), which must be set in the AES_IVRx. The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message. The AES_IVRx are also used by the CTR mode to set the counter value.

57.4.1 AES Register Endianness

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The AES interface requires little-endian format words. However, in accordance with the protocol of the FIPS 197 specification, data is collected, processed and stored by the AES algorithm in big-endian form.

The following example illustrates how to configure the AES:

If the first 64 bits of a message (according to FIPS 197, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then the AES_IDATAR0 and AES_IDATAR1 registers must be written with the following pattern:

- AES_IDATAR0 = 0xcadefeca
- AES_IDATAR1 = 0x67452301

57.4.2 Operating Modes

The AES supports the following modes of operation:

- ECB: Electronic Code Book
- CBC: Cipher Block Chaining
- OFB: Output Feedback
- CFB: Cipher Feedback
  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)
  - CFB128 (CFB where the length of the data segment is 128 bits)
- CTR: Counter
- GCM: Galois/Counter Mode

The data preprocessing, data post-processing and data chaining for the concerned modes are performed automatically. Refer to the *NIST Special Publication 800-38A* and *NIST Special Publication 800-38D* for more complete information.

These modes are selected by setting the OPMOD field in the AES_MR.

In CFB mode, five data sizes are possible (8, 16, 32, 64 or 128 bits), configurable by means of the CFBS field in the AES_MR (Section 57.5.2 “AES Mode Register”).

In CTR mode, the size of the block counter embedded in the module is 16 bits. Therefore, there is a rollover after processing 1 megabyte of data. If the file to be processed is greater than 1 megabyte, this file must be split into fragments of 1 megabyte or less for the first fragment if the initial value of the counter is greater than 0. Prior to loading the first fragment into AES_IDATARx, AES_IVRx must be fully programmed with the initial counter value. For any fragment, after the transfer is completed and prior to transferring the next fragment, AES_IVRx must be programmed with the appropriate counter value.

57.4.3 Double Input Buffer

The AES_IDATARx can be double-buffered to reduce the runtime of large files.

This mode allows writing a new message block when the previous message block is being processed. This is only possible when DMA accesses are performed (SMOD = 0x2).

The DUALBUFF bit in the AES_MR must be set to ‘1’ to access the double buffer.

57.4.4 Start Modes

The SMOD field in the AES_MR allows selection of the encryption (or decryption) Start mode.

57.4.4.1 Manual Mode

The sequence of actions is as follows:

1. Write the AES_MR with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 128-bit/192-bit/256-bit key in the AES_KEYWRx.
3. Write the initialization vector (or counter) in the AES_IVRx.

Note: The AES_IVRx concerns all modes except ECB.

4. Set the bit DATRDY (Data Ready) in the AES Interrupt Enable register (AES_IER), depending on whether an interrupt is required or not at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized AES_IDATARx (see Table 57-2).
6. Set the START bit in the AES Control register (AES_CR) to begin the encryption or the decryption process.

7. When processing completes, the DATRDY flag in the AES Interrupt Status register (AES_ISR) is raised. If an interrupt has been enabled by setting the DATRDY bit in the AES_IER, the interrupt line of the AES is activated.
8. When software reads one of the AES_ODATARx, the DATRDY bit is automatically cleared.

Table 57-2. Authorized Input Data Registers

Operating Mode	Input Data Registers to Write
ECB	All
CBC	All
OFB	All
128-bit CFB	All
64-bit CFB	AES_IDATAR0 and AES_IDATAR1
32-bit CFB	AES_IDATAR0
16-bit CFB	AES_IDATAR0
8-bit CFB	AES_IDATAR0
CTR	All
GCM	All

Notes: 1. In 64-bit CFB mode, writing to AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.
2. In 32, 16, and 8-bit CFB modes, writing to AES_IDATAR1, AES_IDATAR2 and AES_IDATAR3 is not allowed and may lead to errors in processing.

57.4.4.2 Auto Mode

The Auto Mode is similar to the manual one, except that in this mode, as soon as the correct number of AES_IDATARx is written, processing is automatically started without any action in the AES_CR.

57.4.4.3 DMA Mode

The DMA Controller can be used in association with the AES to perform an encryption/decryption of a buffer without any action by software during processing.

The SMOD field in the AES_MR must be configured to 0x2 and the DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be configured with the address of AES_IDATAR0.

The DMA chunk size configuration depends on the AES mode of operation and is listed in Table 57-3 “DMA Data Transfer Type for the Different Operating Modes”.

When writing data to AES with a first DMA channel, data are first fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the AES with the second DMA channel, the source data is the data read from AES and data destination is the memory buffer. In this case, the source data size depends on the AES mode of operation and is listed in Table 57-3.

Table 57-3. DMA Data Transfer Type for the Different Operating Modes

Operating Mode	Chunk Size	Destination/Source Data Transfer Type
ECB	4	Word
CBC	4	Word
OFB	4	Word
CFB 128-bit	4	Word
CFB 64-bit	1	Word
CFB 32-bit	1	Word
CFB 16-bit	1	Half-word
CFB 8-bit	1	Byte
CTR	4	Word
GCM	4	Word

57.4.5 Last Output Data Mode

This mode is used to generate cryptographic checksums on data (MAC) by means of cipher block chaining encryption algorithm (CBC-MAC algorithm for example).

After each end of encryption/decryption, the output data are available either on the AES_ODATARx for Manual and Auto mode or at the address specified in the receive buffer pointer for DMA mode (see Table 57-4).

The Last Output Data (LOD) bit in the AES_MR allows retrieval of only the last data of several encryption/decryption processes.

Therefore, there is no need to define a read buffer in DMA mode.

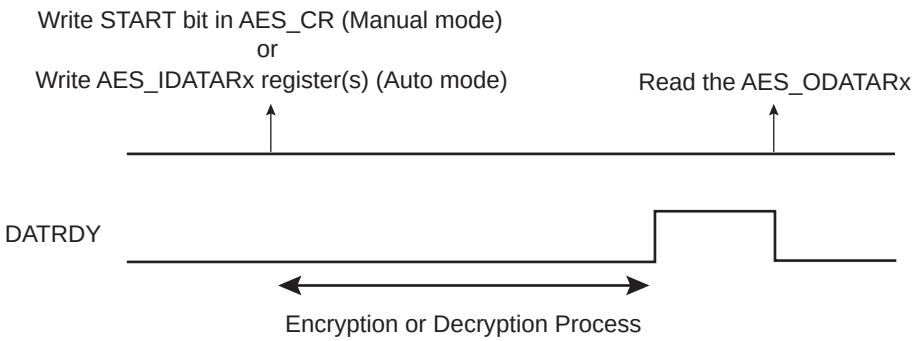
This data are only available on the AES_ODATARx.

57.4.5.1 Manual and Auto Modes

If AES_MR.LOD = 0

The DATRDY flag is cleared when at least one of the AES_ODATARx is read (see Figure 57-1).

Figure 57-1. Manual and Auto Modes with AES_MR.LOD = 0



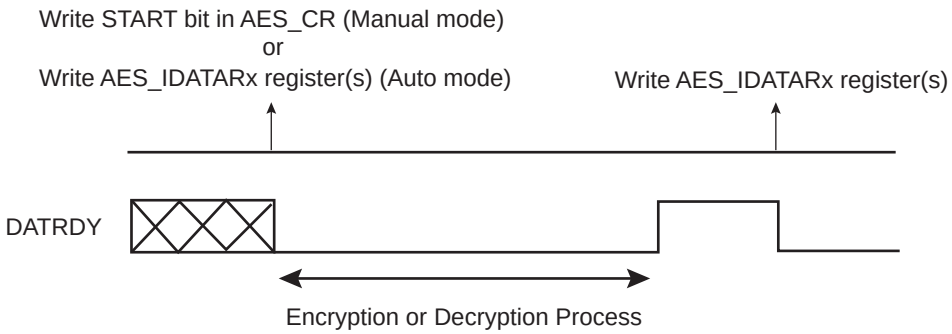
If the user does not want to read the AES_ODATARx between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user cannot know the end of the following encryptions/decryptions.

If AES_MR.LOD = 1

This mode is optimized to process AES CPC-MAC operating mode.

The DATRDY flag is cleared when at least one AES_IDATAR is written (see Figure 57-2). No more AES_ODATAR reads are necessary between consecutive encryptions/decryptions.

Figure 57-2. Manual and Auto Modes with AES_MR.LOD = 1



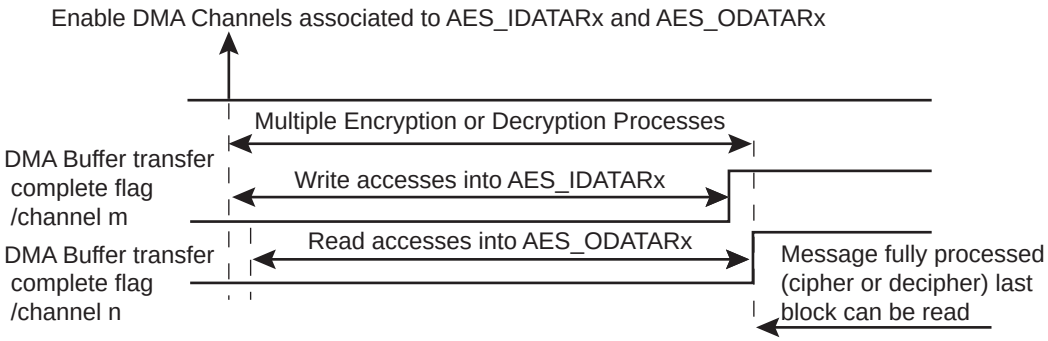
57.4.5.2 DMA Mode

If AES_MR.LOD = 0

This mode may be used for all AES operating modes except CBC-MAC where AES_MR.LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to AES_ODATARx (see Figure 57-3). Two DMA channels are required: one for writing message blocks to AES_IDATARx and one to obtain the result from AES_ODATARx.

Figure 57-3. DMA Transfer with AES_MR.LOD = 0



If AES_MR.LOD = 1

This mode is optimized to process AES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (see Figure 57-4).

In this case, no receive buffers are required.

The output data are only available on the AES_ODATARx.

Figure 57-4. DMA Transfer with AES_MR.LOD = 1

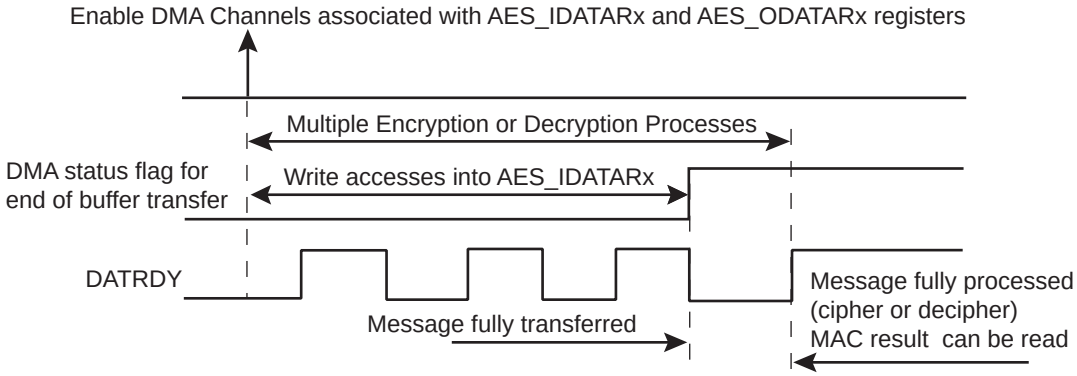


Table 57-4 summarizes the different cases.

Table 57-4. Last Output Data Mode Behavior versus Start Modes

Sequence	Manual and Auto Modes		DMA Transfer	
	AES_MR.LOD = 0	AES_MR.LOD = 1	AES_MR.LOD = 0	AES_MR.LOD = 1
DATRDY Flag Clearing Condition(1)	At least one AES_ODATAR must be read	At least one AES_IDATAR must be written	Not used	Managed by the DMA
End of Encryption/Decryption Notification	DATRDY	DATRDY	2 DMA Buffer transfer complete flags (channel m and channel n)	DMA buffer transfer complete flag, then AES DATRDY flag
Encrypted/Decrypted Data Result Location	In the AES_ODATARx	In the AES_ODATARx	At the address specified in the Channel Buffer Transfer Descriptor	In the AES_ODATARx

Note: 1. Depending on the mode, there are other ways of clearing the DATRDY flag. See Section 57.5.6 “AES Interrupt Status Register”.

Warning: In DMA mode, reading the AES_ODATARx before the last data transfer may lead to unpredictable results.

57.4.6 Galois/Counter Mode (GCM)

57.4.6.1 Description

GCM comprises the AES engine in CTR mode along with a universal hash function (GHASH engine) that is defined over a binary Galois field to produce a message authentication tag (the AES CTR engine and the GHASH engine are depicted in Figure 57-5).

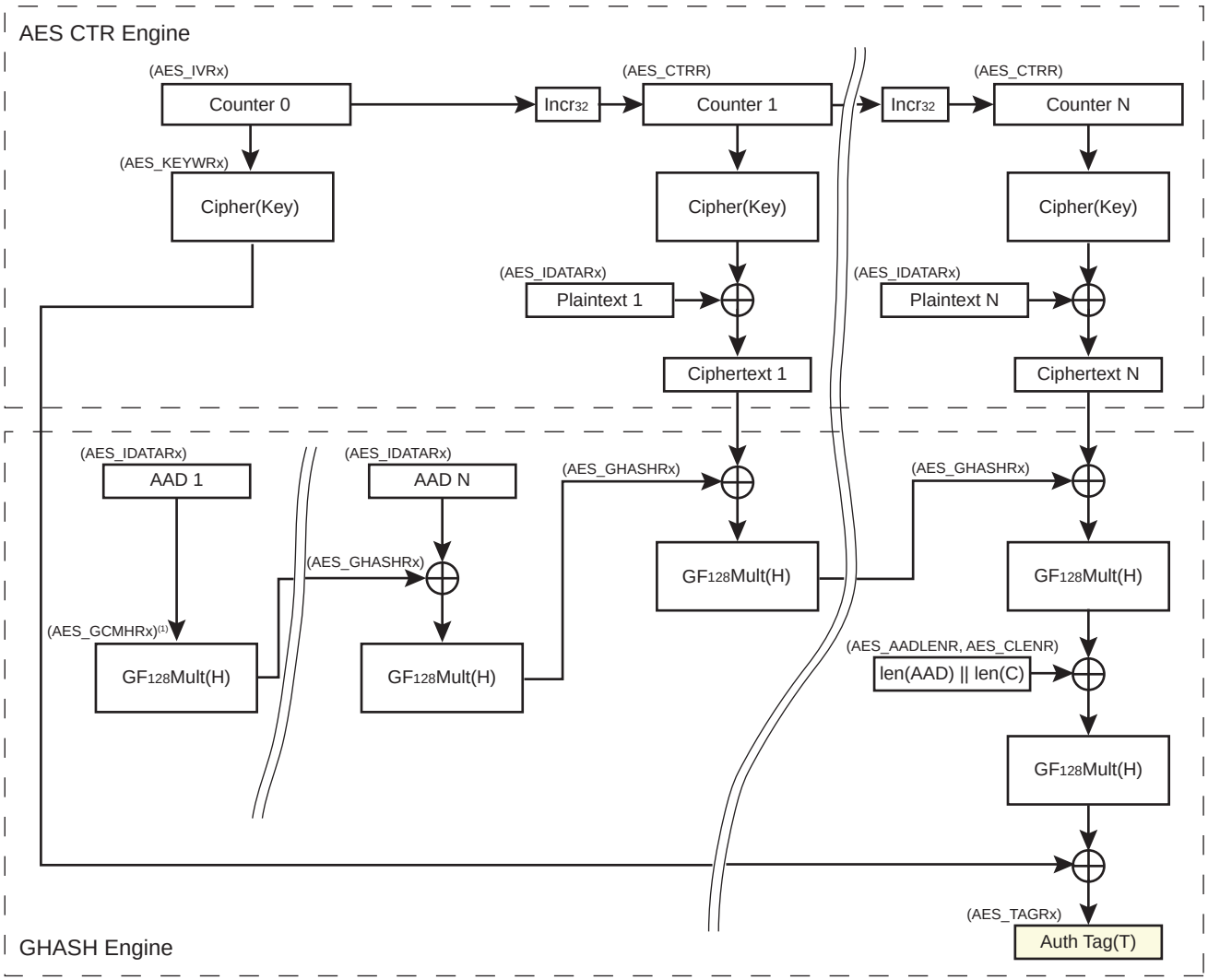
The GHASH engine processes data packets after the AES operation. GCM provides assurance of the confidentiality of data through the AES Counter mode of operation for encryption. Authenticity of the confidential data is assured through the GHASH engine. GCM can also provide assurance of data that is not encrypted. Refer to the *NIST Special Publication 800-38D* for more complete information.

GCM can be used with or without the DMA master. Messages may be processed as a single complete packet of data or they may be broken into multiple packets of data over time.

GCM processing is computed on 128-bit input data fields. There is no support for unaligned data. The AES key length can be whatever length is supported by the AES module.

The recommended programming procedure when using DMA is described in Section 57.4.6.3.

Figure 57-5. GCM Block Diagram



Note: 1. Optional

57.4.6.2 Key Writing and Automatic Hash Subkey Calculation

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM Hash Subkey *H* generation—The GCM hash subkey (*H*) is automatically generated. The GCM hash subkey generation must be complete before doing any other action. The DATRDY bit of the AES_ISR indicates when the subkey generation is complete (with interrupt if configured). The GCM hash subkey calculation is processed with the formula *H* = CIPHER(Key, <128 bits to zero>). The generated GCM *H* value is then available in the AES_GCMHRx. If the application software requires a specific hash subkey, the automatically generated H value can be overwritten in the AES_GCMHRx.
  The AES_GCMHRx can be written after the end of the hash subkey generation (see AES_ISR.DATRDY) and prior to starting the input data feed.
- AES_GHASHRx Clear—The AES_GHASHRx are automatically cleared. If a hash initial value is needed for the GHASH, it must be written to the AES_GHASHRx
  - after a write to AES_KEYWRx, if any
  - before starting the input data feed

57.4.6.3 GCM Processing

GCM processing comprises three phases:

1. Processing the Additional Authenticated Data (*AAD*), hash computation only.
2. Processing the Ciphertext (*C*), hash computation + ciphering/deciphering.
3. Generating the Tag using length of *AAD*, length of *C* and J_0 (see NIST documentation for details).

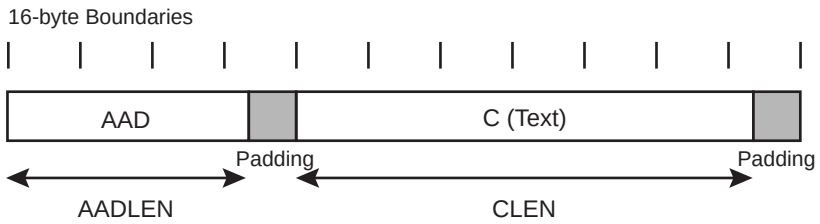
The Tag generation can be done either automatically, after the end of *AAD*/*C* processing if TAG_EN bit is set in the AES_MR or done manually, using the GHASH field in AES_GHASHRx (See subsections "Processing a Complete Message with Tag Generation" and "Manual GCM Tag Generation" below for details).

Processing a Complete Message with Tag Generation

Use this procedure only if J_0 four LSB bytes ≠ 0xFFFFFFFF.

NOTE: In the case where J_0 four LSB bytes = 0xFFFFFFFF or if the value is unknown, use the procedure described in "Processing a Complete Message without Tag Generation" followed by the procedure in "Manual GCM Tag Generation".

Figure 57-6. Full Message Alignment



To process a complete message with Tag generation, the sequence is as follows:

1. In AES_MR set OPMOD to GCM and GTAGEN to '1' (configuration as usual for the rest).
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. See Section 57.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Calculate the J_0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = \text{GHASH}_H(IV \,||\, 0^{s+64} \,||\, [\text{len}(IV)]_{64})$ if len(IV) ≠ 96. See "Processing a Message with only AAD (GHASHH)" for J_0 generation.
4. Set IV in AES_IVRx with inc32(J_0) (J_0 + 1 on 32 bits).
5. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR.
6. Fill the IDATA field of AES_IDATARx with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Wait for TAGRDY to be set (use interrupt if needed), then read the TAG field of AES_TAGRx to obtain the authentication tag of the message.

Processing a Complete Message without Tag Generation

Processing a message without generating the Tag can be used to customize the Tag generation, or to process a fragmented message. To manually generate the GCM Tag, refer to "Manual GCM Tag Generation".

To process a complete message without Tag generation, the sequence is as follows:

1. In AES_MR set OPMOD to GCM and GTAGEN to '0' (configuration as usual for the rest).
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in the AES_GCMHRx. See Section 57.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".

3. Calculate the J_0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = \text{GHASH}_H(IV \,||\, 0^{s+64} \,||\, [\text{len}(IV)]_{64})$ if len(IV) ≠ 96. See "Processing a Message with only AAD (GHASHH)" for J_0 generation example when len(IV) ≠ 96.
4. Set IV in AES_IVRx with inc32(J_0) (J_0 + 1 on 32 bits).
5. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR.
6. Fill the IDATA field of AES_IDATARx with the message to process according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if CLEN ≠ 0 (or wait for DATRDY), then read the GHASH field of AES_GHASHRx to obtain the hash value after the last processed data.

Processing a Fragmented Message without Tag Generation

If needed, a message can be processed by fragments, in such case automatic GCM Tag generation is not supported.

To process a message by fragments, the sequence is as follows:

- First fragment:

1. In AES_MR set OPMOD to GCM and GTAGEN to '0' (configuration as usual for the rest).
2. Set KEYW in AES_KEYWRx and wait for DATRDY bit of AES_ISR to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in the AES_GCMHRx. See Section 57.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Calculate the J_0 value as described in NIST documentation $J_0 = IV \,||\, 0^{31} \,||\, 1$ when len(IV) = 96 and $J_0 = \text{GHASH}_H(IV \,||\, 0^{s+64} \,||\, [\text{len}(IV)]_{64})$ if len(IV) ≠ 96. See "Processing a Message with only AAD (GHASHH)" for J_0 generation example when len(IV) ≠ 96.
4. Set IV in AES_IVRx with inc32(J_0) (J_0 + 1 on 32 bits).
5. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR according to the length of the first fragment, or set the fields with the full message length, both configurations work.
6. Fill the IDATA field of AES_IDATARx with the first fragment of the message to process (aligned on 16-byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in *AAD* phase), then read the GHASH field of AES_GHASHRx to obtain the value of the hash after the last processed data and finally read the CTR field of the AES_CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in *AAD* phase).

- Next fragment (or last fragment):

1. In AES_MR set OPMOD to GCM and GTAGEN to '0' (configuration as usual for the rest).
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in the AES_GCMHRx. See Section 57.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Set IV in AES_IVRx with:
   - If the first block of the fragment is a block of Additional Authenticated data, set IV in AES_IVRx with the J0 initial value
   - If the first block of the fragment is a block of Plaintext data, set IV in AES_IVRx with a value constructed as follows: 'LSB96(J0) || CTR' value, (96 bit LSB of J0 concatenated with saved CTR value from previous fragment).

4. Set AADLEN field in AES_AADLENR and CLEN field in AES_CLENR according to the length of the current fragment, or set the fields with the remaining message length, both configurations work.
5. Fill the GHASH field of AES_GHASHRx with the value stored after the previous fragment.
6. Fill the IDATA field of AES_IDATARx with the current fragment of the message to process (aligned on 16 byte boundary) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when the data have been processed (however, no output data are generated when processing *AAD*).
7. Make sure the last output data have been read if the fragment ends in C phase (or wait for DATRDY if the fragment ends in *AAD* phase), then read the GHASH field of AES_GHASHRx to obtain the value of the hash after the last processed data and finally read the CTR field of the AES_CTR to obtain the value of the CTR encryption counter (not needed when the fragment ends in *AAD* phase).

Note: Step 1 and 2 are required only if the value of the concerned registers has been modified.

Once the last fragment has been processed, the GHASH value will allow manual generation of the GCM tag. See "Manual GCM Tag Generation".

Manual GCM Tag Generation

This section describes the last steps of the GCM Tag generation.

The Manual GCM Tag Generation is used to complete the GCM Tag Generation when the message has been processed without Tag Generation.

Note: The Message Processing without Tag Generation must be finished before processing the Manual GCM Tag Generation.

To generate a GCM Tag manually, the sequence is as follows:

Processing $S = \text{GHASH}_H (AAD \,||\, 0^v \,||\, C \,||\, 0^u \,||\, [\text{len}(AAD)]_{64} \,||\, [\text{len}(C)]_{64})$:

1. In AES_MR set OPMOD to GCM and GTAGEN to '0' (configuration as usual for the rest).
2. Set KEYW in AES_KEYWRx and wait for DATRDY bit of AES_ISR to be set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in the AES_GCMHRx. See Section 57.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Set AADLEN field to 0x10 (16 bytes) in AES_AADLENR and CLEN field to '0' in AES_CLENR. This will allow running a single GHASH_H on a 16-byte input data (see Figure 57-7).
4. Fill the GHASH field of AES_GHASHRx with the state of the GHASH field stored at the end of the message processing.
5. Fill the IDATA field of AES_IDATARx according to the SMOD configuration used with '$\text{len}(AAD)_{64} \,||\, \text{len}(C)_{64}$' value as described in the NIST documentation and wait for DATRDY to be set; use interrupt if needed.
6. Read the GHASH field of AES_GHASHRx to obtain the current value of the hash.

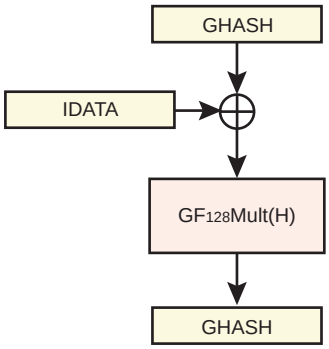
Processing $T = \text{GCTR}_K(J_0, S)$:

7. In AES_MR set OPMOD to CTR (configuration as usual for the rest).
8. Set the IV field in AES_IVRx with 'J_0' value.
9. Fill the IDATA field of AES_IDATARx with the GHASH value read at step 6 and wait for DATRDY to be set (use interrupt if needed).
10. Read the ODATA field of AES_ODATARx to obtain the GCM Tag value.

Note: Step 4 is optional if the GHASH field is to be filled with value '0' (0 length packet for instance).

Processing a Message with only AAD (GHASHH)

Figure 57-7. Single GHASHH Block Diagram (AADLEN ≤ 0x10 and CLEN = 0)



It is possible to process a message with only *AAD* setting the CLEN field to '0' in the AES_CLENR, this can be used for J_0 generation when len(*IV*) ≠ 96 for instance.

Example: Processing J_0 when len(*IV*) ≠ 96

To process J_0 = GHASH$_H$(*IV* || 0^{s+64} || [len(*IV*)]$_{64}$), the sequence is as follows:

1. In AES_MR set OPMOD to GCM and GTAGEN to '0' (configuration as usual for the rest).
2. Set KEYW in AES_KEYWRx and wait until DATRDY bit of AES_ISR is set (GCM hash subkey generation complete); use interrupt if needed. After the GCM hash subkey generation is complete the GCM hash subkey can be read or overwritten with specific value in the AES_GCMHRx. See Section 57.4.6.2 "Key Writing and Automatic Hash Subkey Calculation".
3. Set AADLEN field with 'len(*IV* || 0^{s+64} || [len(*IV*)]$_{64}$)' in AES_AADLENR and CLEN field to '0' in AES_CLENR. This will allow running a GHASHH only.
4. Fill the IDATA field of AES_IDATARx with the message to process (*IV* || 0^{s+64} || [len(*IV*)]$_{64}$) according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASHH step is over (use interrupt if needed).
5. Read the GHASH field of AES_GHASHRx to obtain the J_0 value.

Note: The GHASH value can be overwritten at any time by writing the GHASH field value of AES_GHASHRx, used to perform a GHASHH with an initial value for GHASH (write GHASH field between step 3 and step 4 in this case).

Processing a Single GF128 Multiplication

The AES can also be used to process a single multiplication in the Galois field on 128 bits (GF128) using a single GHASHH with custom *H* value (see Figure 57-7).

To run a GF128 multiplication (A x B), the sequence is as follows:

1. In AES_MR set OPMOD to GCM and GTAGEN to '0' (configuration as usual for the rest).
2. Set AADLEN field with 0x10 (16 bytes) in AES_AADLENR and CLEN field to '0' in AES_CLENR. This will allow running a single GHASHH.
3. Fill the H field of the AES_GCMHRx with B value.
4. Fill the IDATA field of AES_IDATARx with the A value according to the SMOD configuration used. If Manual Mode or Auto Mode is used, the DATRDY bit indicates when a GHASHH computation is over (use interrupt if needed).
5. Read the GHASH field of AES_GHASHRx to obtain the result.

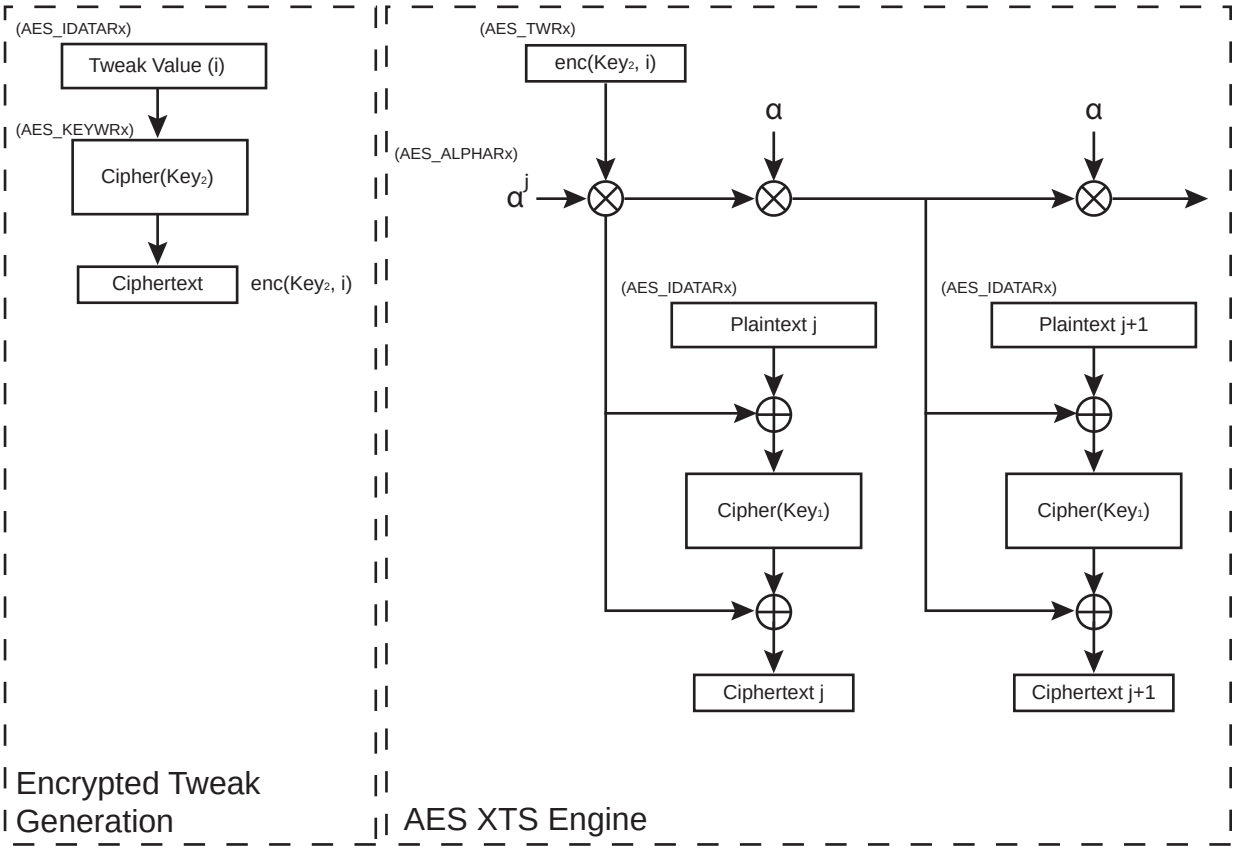
Note: The GHASH field of AES_GHASHRx can be initialized with a value C between step 3 and step 4 to run a ((A XOR C) x B) GF128 multiplication.

57.4.7 XEX-based Tweaked-codebook Mode (XTS)

XTS mode comprises the AES engine with XOR on inputs and outputs. After each encryption/decryption, the value used for the XOR is multiplied by the first GF(2^{128}) alpha primitive (0x2) and then used for the next encryption/decryption. The XTS mode uses two different keys and defines a Tweak Value (i) as additional input.

XTS processing is computed on 128-bit input data fields. There is no support for unaligned data (padding must be done manually if needed). The AES key length can be any length supported by the AES module.

Figure 57-8. XTS Block Diagram



57.4.7.1 XTS Processing Procedure

XTS processing comprises two phases:

1. Generate encrypted tweak with Key2 (this step is only required for the first processing, further consecutive processing does not require this step).
2. Process the data giving encrypted tweak and first alpha primitive for the first encryption/decryption.

Encrypted Tweak Generation

In the case of a new encryption/decryption, it is necessary to first encrypt the Tweak Value (i) with Key2. Here are the steps to follow to perform this step:

1. In AES_MR, set OPMODE to ECB, CIPHER bit to '1' and other fields to the required value.
2. Set KEYW in AES_KEYWRx with Key2.
3. Fill the IDATA field of AES_IDATARx with the Tweak value (i) according to the SMOD configuration used. If Manual mode or Auto mode is used, the DATRDY bit indicates when the data have been processed and can be read in AES_ODATARx.

Data Processing

To process data using XTS mode, follow the steps below:

1. In AES_MR, set OPMODE to XTS and other fields to the required value.
2. Set KEYW in AES_KEYWRx with Key1.
3. If the data to process is the first to be processed in the data unit, or if the data block to process is not consecutive to the previous processed data block in the same data unit, then it is required to fill the AES_TWRx register with the encrypted Tweak Value (see "Encrypted Tweak Generation" for details), and then to fill the AES_ALPHARx register with the alpha primitive corresponding to the block number in the data unit. Otherwise, ignore Step 3.
4. Fill the IDATA field of AES_IDATARx with the data to process according to the SMOD configuration used. If Manual mode or Auto mode is used, the DATRDY bit indicates when the data have been processed and can be read in AES ODATARx. Repeat Step 4 as long as consecutive data blocks are processed in the same data unit.

57.4.8 Automatic Padding Mode

When Automatic Padding mode is configured, the message is automatically padded after the last block is written. Depending on the size of the message, either a padding is performed after the last part of the message and padding blocks are added or only padding blocks are added.

Both IPSEC and SSL padding standards are supported.

The auto padding feature only supports CBC and CTR mode.

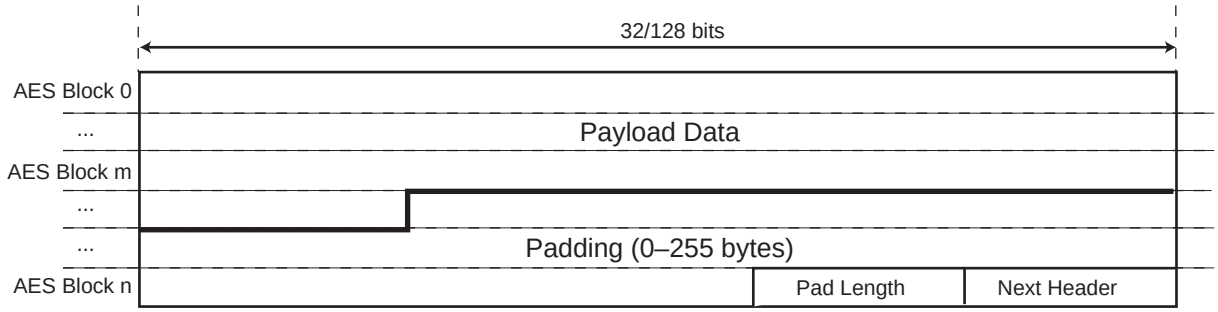
Note: When automatic padding is enabled and the field SMOD=2 in AES_MR, the bit DUALBUFF must be cleared in AES_MR.

57.4.8.1 IPSEC Padding

Automatic Padding is enabled by writing a '1' to the APEN bit in the AES Extended Mode register (AES_EMR).

IPSEC padding mode is selected by writing a '0' to the APM bit in the AES_EMR.

Figure 57-9. IPSEC Padding



The "Pad Length" in bytes can be configured in the PADLEN field and the "Next Header" value can be configured in the NHEAD field of the AES_EMR. The PADLEN field must be configured with the length of the padding section, not including the length of the "Pad Length" and "Next Header" sections.

The BCNT field in the AES Byte Counter register (AES_BCNT) defines the length, in bytes, of the message to process. It must be configured before writing the first data in AES_IDATARx and the remaining bytes to process can be read at anytime (BCNT value is decremented after each AES_IDATARx access).

AES_BCNT.BCNT and AES_EMR.PADLEN must be configured so that the sum of the length of the message (Payload Data) and of the length of the Padding, Pad Length (1 byte) and Next Header (1 byte) sections is a multiple of the AES block size (128 bits).

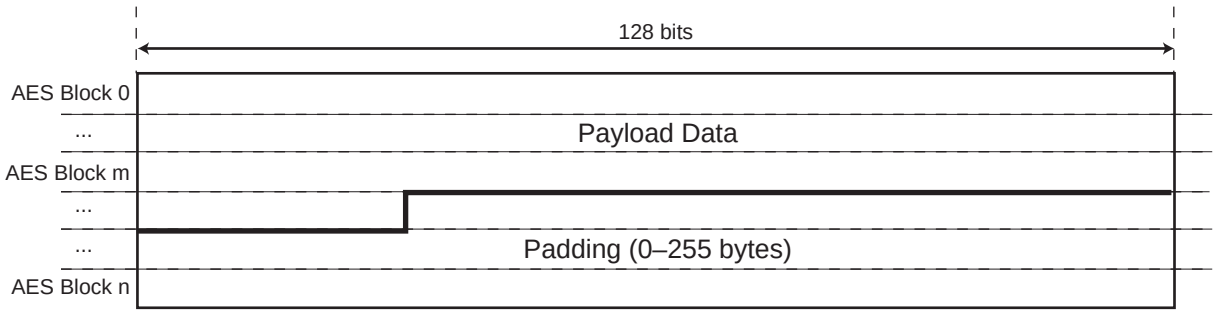
To process an IPSEC message using auto-padding, the sequence is as follows:

1. In AES_MR set OPMOD to either CBC or CTR mode.
2. In AES_EMR set APEN to ‘1’, APM to ‘0’, PADLEN to the desired padding length in byte and NHEAD to the desired Next Header field value.
3. In AES_BCNT set the BCNT field with the whole message length, without padding, in byte.
4. Set KEYW in AES_KEYWRx registers
5. Set IV in AES_IVRx registers if needed
6. Fill IDATA field of AES_IDATARx with the message to process according to the SMOD configuration used. On the last data block write only what is necessary (e.g., write only AES_IDATAR0 if last block size is ≤ 32 bits).
7. Wait for the DATRDY flag to be raised, meaning auto-padding completion and last block processing.

57.4.8.2 SSL Padding

Auto Padding is enabled by writing a ‘1’ to the APEN bit in the AES_EMR and SSL padding mode is selected by writing a ‘1’ to the APM bit in the AES_EMR.

Figure 57-10. SSL Padding



The padding length is configured in the field PADLEN of the AES_EMR.

The BCNT field in the AES Byte Counter register (AES_BCNT) defines the length, in bytes, of the message to process. It must be configured before writing the first data in AES_IDATARx and the remaining bytes to process can be read at anytime (BCNT value is decremented after each AES_IDATARx access).

AES_BCNT.BCNT and AES_EMR.PADLEN must be configured so that the length of the message plus the length of the padding section is a multiple of the AES block size (128 bits).

To process a complete SSL message, the sequence is as follows:

1. In AES_MR set OPMOD to either CBC or CTR mode.
2. In AES_EMR set APEN to ‘1’, APM to ‘1’, PADLEN to the desired padding length in bytes.
3. In AES_BCNT set the BCNT field with the whole message length, without padding, in bytes.
4. Set KEYW in AES_KEYWRx
5. Set IV in AES_IVRx if needed
6. Fill IDATA field of AES_IDATARx with the message to process according to the SMOD configuration used. On the last data block write only what is necessary (e.g., write only AES_IDATAR0 if last block size is ≤ 32 bits).
7. Wait for the DATRDY flag to be raised, meaning auto-padding completion and last block processing.

57.4.8.3 Flags

The EOPAD flag in the AES_ISR rises as soon as the automatic padding phase is over, meaning that all the extra padding blocks have been processed. Reading the AES_ISR clears this flag.

The PLENERR flag in the AES_ISR indicates an error in the frame configuration, meaning that the whole message length including padding does not respect the standard selected. The PLENERR flag rises at the end of the frame in case of wrong message length and is cleared reading the AES_ISR.

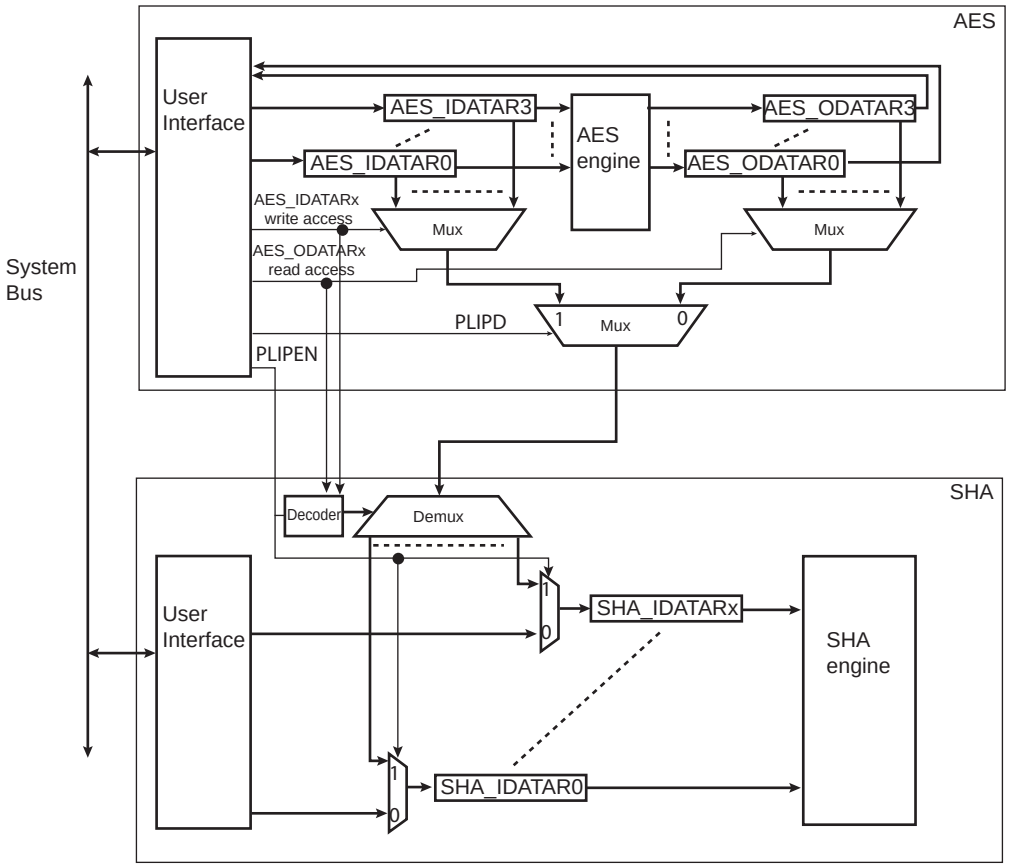
In IPSEC/SSL standard message length including padding must be a multiple of the AES block size when CBC mode is used and multiple of 32-bit if CTR mode is used.

57.4.9 Secure Protocol Layers Improved Performances

Secure protocol layers such as IPSec require encryption and authentication. For IPSec, the authentication is based on HMAC, thus SHA is required. To optimize performance, the AES embeds a mode of operation that enables the SHA module to process the input or output data of the AES module. If this mode is enabled, write access is required only into the AES_IDATARx registers, since SHA input data registers are automatically written by AES without software intervention. When the DMA is configured to transfer a buffer of data (input frame), only one transfer descriptor is required for both authentication and encryption/decryption processes and only one buffer is transferred through the system bus (reducing the load of the system bus).

Improved performance for secure protocol layers requires the bit PLIPEN to be set in AES_EMR.

Figure 57-11. Secure Protocol Layers Improved Performances Block Diagram



57.4.9.1 Cipher Mode

When bit PLIPD is cleared and bit PLIPEN=1, the message written into AES_IDATARx is first encrypted with the AES module and the encrypted message is authenticated with the SHA module. Therefore, when PLIPD is cleared, the AES_ODATARx are selected and sent to SHA_IDATARx as soon as AES_ODATARx are read. A read access in AES corresponds to a write access to the corresponding SHA_IDATARx. The number of SHA_IDATARx is greater than the number of AES_ODATARx, but the SHA module embeds the decoding logic to automatically dispatch the AES_ODATARx values into the corresponding SHA_IDATARx without software intervention.

57.4.9.2 Decipher Mode

When bit PLIPD is written to one and bit PLIPEN=1, the message written into AES_IDATARx is decrypted with the AES module and also sent to SHA for authentication. Therefore, when PLIPD=1, the AES_IDATARx are selected and sent to SHA_IDATARx as soon as AES_IDATARx are written. A write access in AES corresponds to a write access to the corresponding SHA_IDATARx. The number of SHA_IDATARx is greater than the number of AES_ODATARx, but the SHA module embeds the decoding logic to automatically dispatch the AES_IDATARx values into the corresponding SHA_IDATARx without software intervention.

57.4.9.3 Encapsulating Security Payload (ESP) IPSec Examples

The following examples describe how to configure AES and SHA to optimize processing an ESP IPSec frame for maximum performance.

The cipher (or decipher) of an ESP IPSec frame requires both encryption (or decryption) and authentication.

For cipher, the input frame located in the system memory must first be padded and the resulting buffer encrypted. The encrypted frame must be written back to the system memory and sent to the authentication module.

When the AES module is configured to improve the performance of the secure protocol layers (bit PLIPEN = 1), the data transfers are simplified, limiting the bandwidth requirements on the system bus.

Before configuring the DMA to start the transfer of the data buffer (input frame) to the AES, the following actions must be taken in registers:

- The BCNT field in AES_BCNT must be configured with the length of the message (Input Frame).
- The padding length of the AES must be configured in field PADLEN of AES_EMR. See Section 57.4.8 “Automatic Padding Mode” to configure Automatic padding mode.
- The next header value must be configured in field NHEAD of AES_EMR.
- Field SMOD in AES_MR and field SMOD in SHA_MR must be configured to 2.

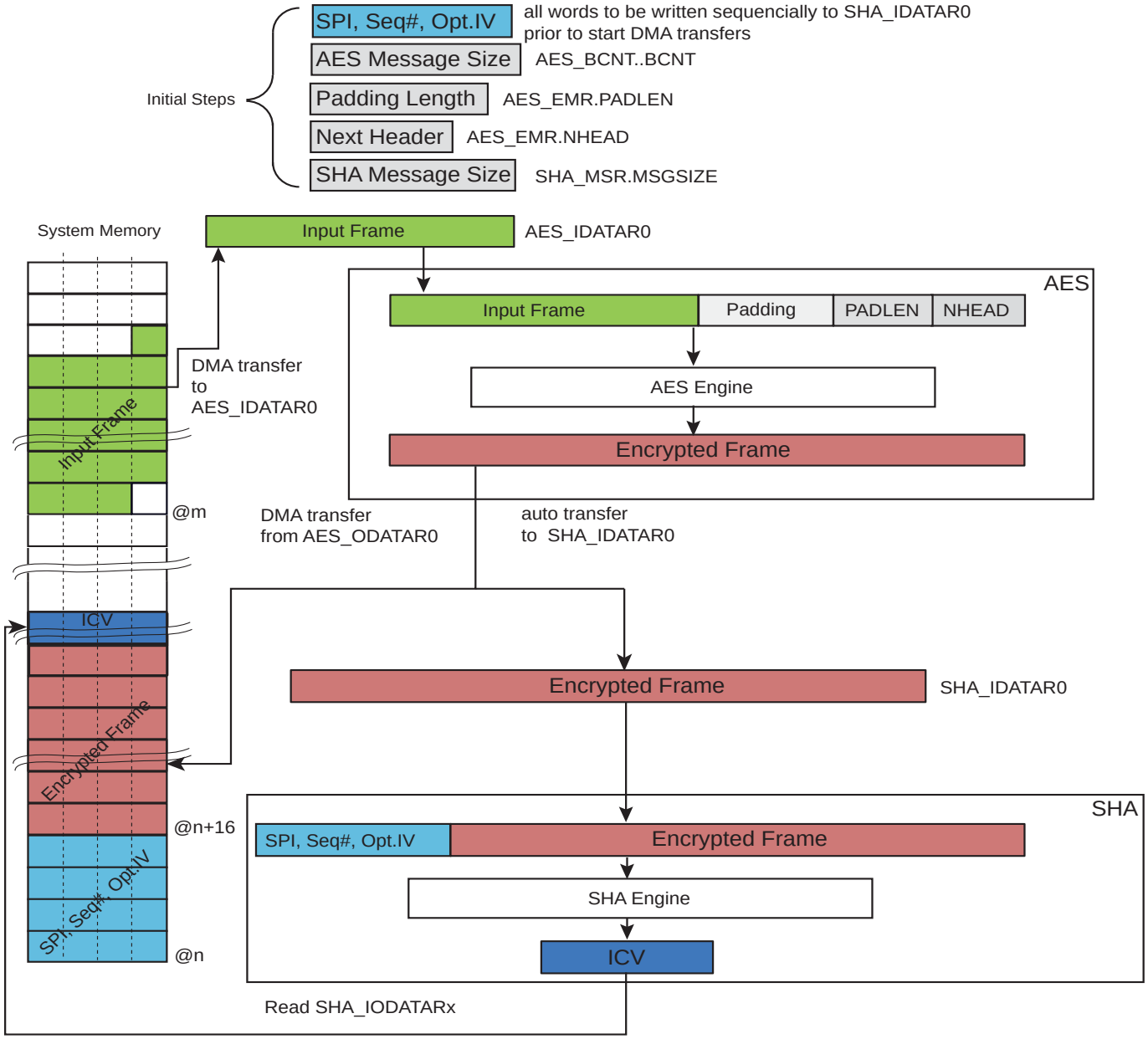
Note: When automatic padding is enabled and the field SMOD = 2 in AES_MR, the bit DUALBUFF must be cleared in AES_MR.

- The MSGSIZE field in SHA_MSR must be configured with the length of the authentication message including the optional extended sequence number (ESN) and header and trailer information required by the authentication algorithm used (HMAC, etc.). Refer to the section “Secure Hash Algorithm (SHA)” in this datasheet for more details on configurations for optimized processing of the header information.
- The Security Parameter Index (SPI, sequence number (SEQ#)) and the optional Initialization Vector (IV) must be configured sequentially in SHA_IDATAR0.
- A first DMA transfer descriptor must be configured to transfer the input frame from the system memory to the AES input data registers (AES_IDATARx), and a second DMA descriptor must be configured to transfer the encrypted frame from AES to the system memory.

Note: If the bit PLIPEN = 1 in AES_EMR, there is no need to define a transfer descriptor to load the encrypted frame into the SHA input data registers because the transfer is automatically performed while the second descriptor transfer is in progress.

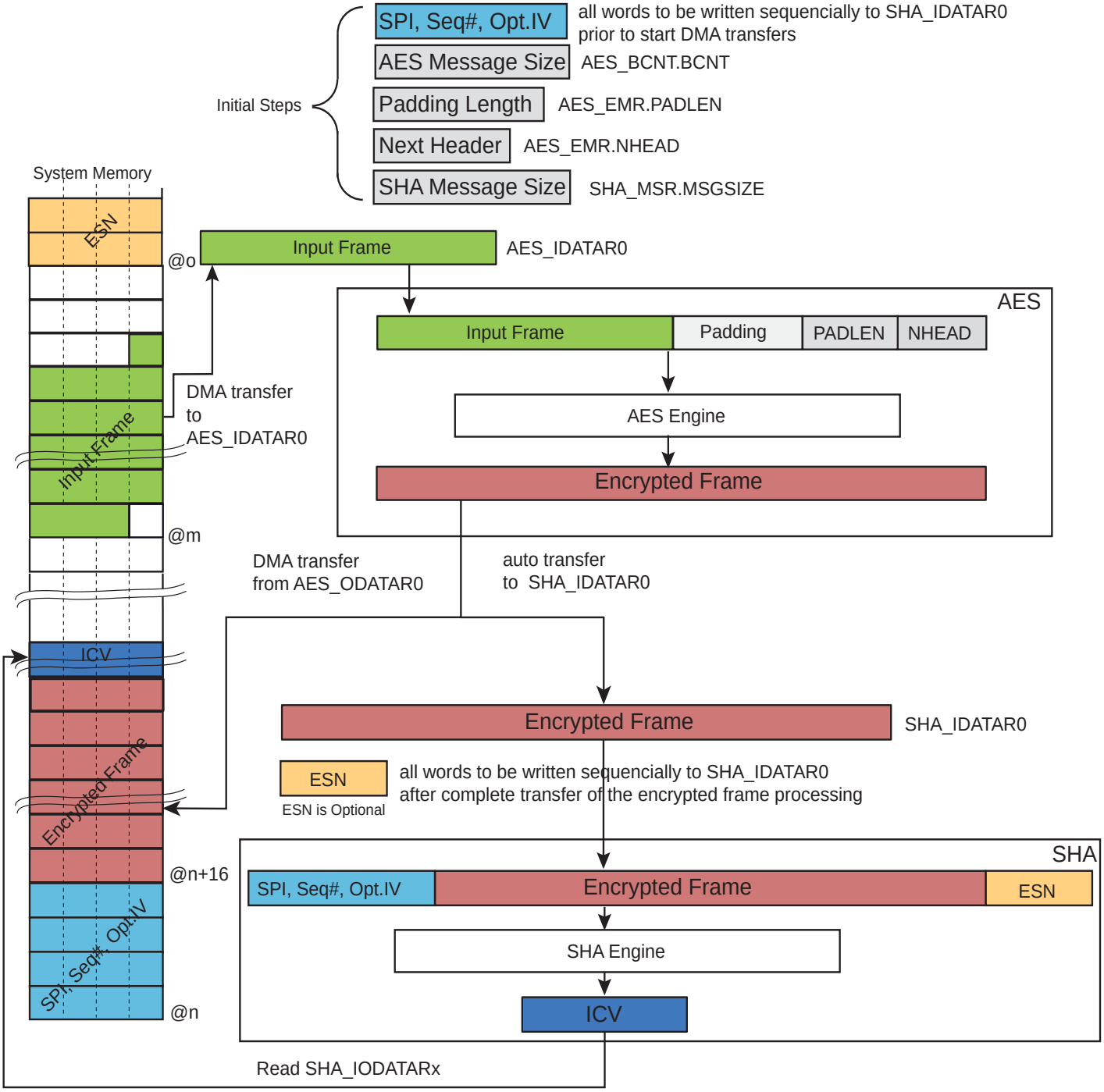
See Figure 57-12 and Figure 57-13.

Figure 57-12. Generation of an ESP IPSec Frame without ESN



If the optional extended sequence number is required for authentication, wait for the AES-to-system memory DMA buffer transfer to complete before configuring the ESN value. The ESN value must be configured in the SHA by writing sequentially each 32-bit word of the ESN into the SHA_IDATAR0 register. Wait for flag WRDY=1 in SHA_ISR before each write in the SHA_IDATAR0 register. Refer to Figure 57-13.

Figure 57-13. Generation of an ESP IPSec Frame with ESN



To decipher an ESP IPSec frame without the optional ESN trailer information, two DMA channels are required and the SHA must be configured in Automatic padding mode.

Note: AES automatic padding must be disabled when deciphering a frame.

- A first DMA transfer descriptor must be configured to load the received encrypted frame from the system memory to AES_IDATARx for decryption. The start address of the first transfer descriptor must be defined after the SPI, SEQ#, and optional IV (see Figure 57-14).
- A second DMA descriptor must be configured to transfer the decrypted frame from AES_ODATARx to the system memory.
- PLIPEN and PLIPD must be written to ‘1’ so that the data buffer is written in AES_IDATARx and in SHA_IDATARx.

The SHA has the capability to perform an automatic check with an expected integrity check value if this value is appended at the end of the frame buffer (SHA_MR.CHECK=2). Thus, if the first transfer descriptor includes the ICV for SHA, the first DMA transfer allows the decryption and authentication processes including the automatic check. The decrypted part resulting from ICV is not required for downstream processing and must be considered as dummy data.

The end of the decryption and authentication processes occur when flag CHECKF=1 in SHA_ISR. The authentication status is provided by field CHKST in SHA_ISR.

If the optional ESN trailer information is part of the ICV (see Figure 57-15), the ESN must be manually written into SHA_IDATAR0. The ESN value must be written after completion of the system memory-to-AES DMA buffer transfer. The ESN value must be configured in the SHA by writing sequentially each 32-bit word of the ESN into the SHA_IDATAR0 register. Wait for flag WRDY=1 in SHA_ISR before each write in the SHA_IDATAR0 register.

When the optional ESN trailer information is part of the ICV, it is not possible to include the ICV received in the input frame to the first transfer descriptor. Moreover, if the HMAC algorithm is used for authentication, no automatic check can be performed when optimizing the processing performances of the SHA module. For more details, see the section “Secure Hash Algorithm (SHA)” in this datasheet. The result of the HMAC read in the SHA_IODATARx must be manually compared with the ICV value of the input frame. The comparison must be performed after the end of the authentication process. The authentication process is completed when the DATRDY flag is set in SHA_ISR.

Figure 57-14. Decryption of an ESP IP Sec Frame without ESN

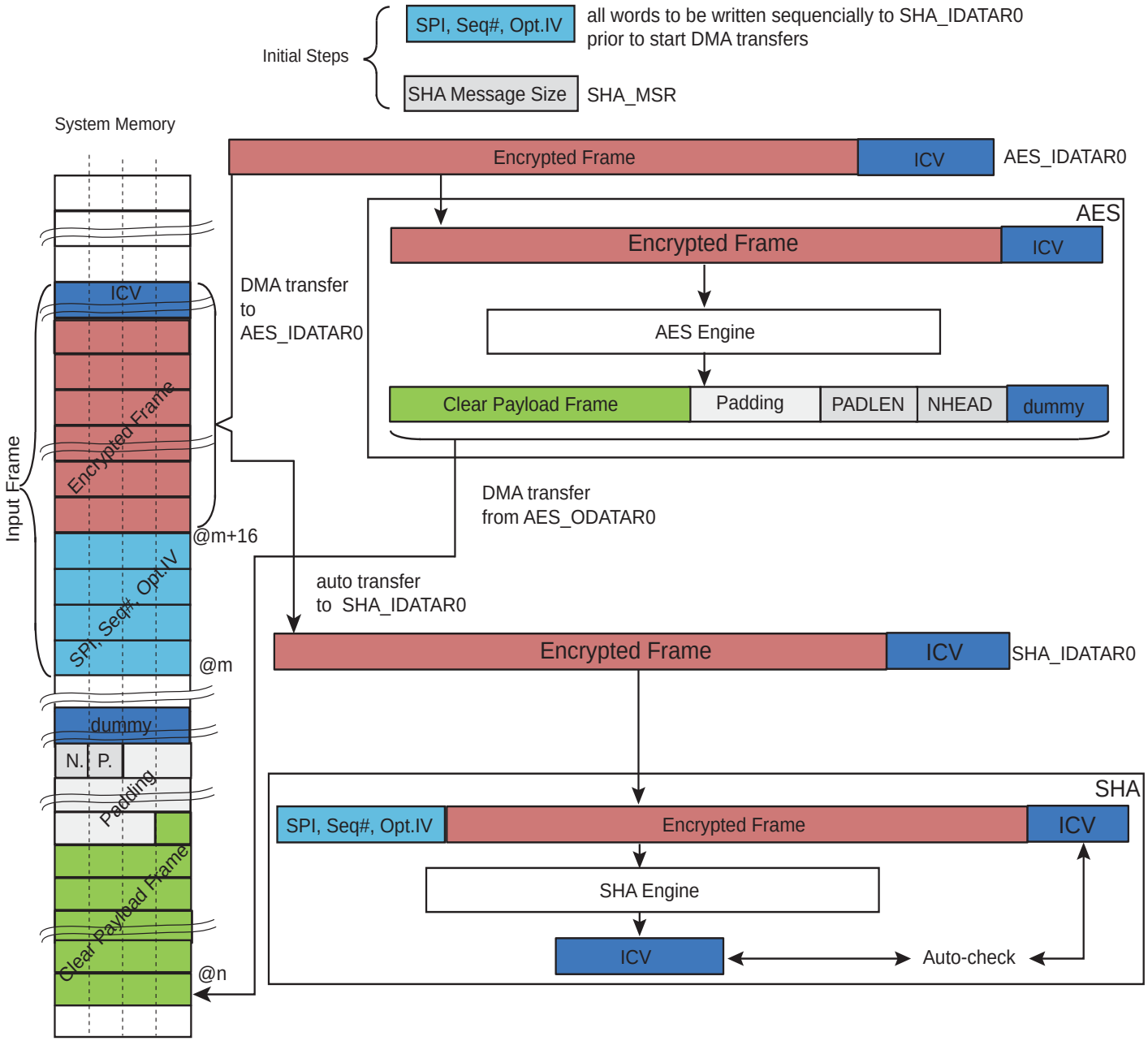
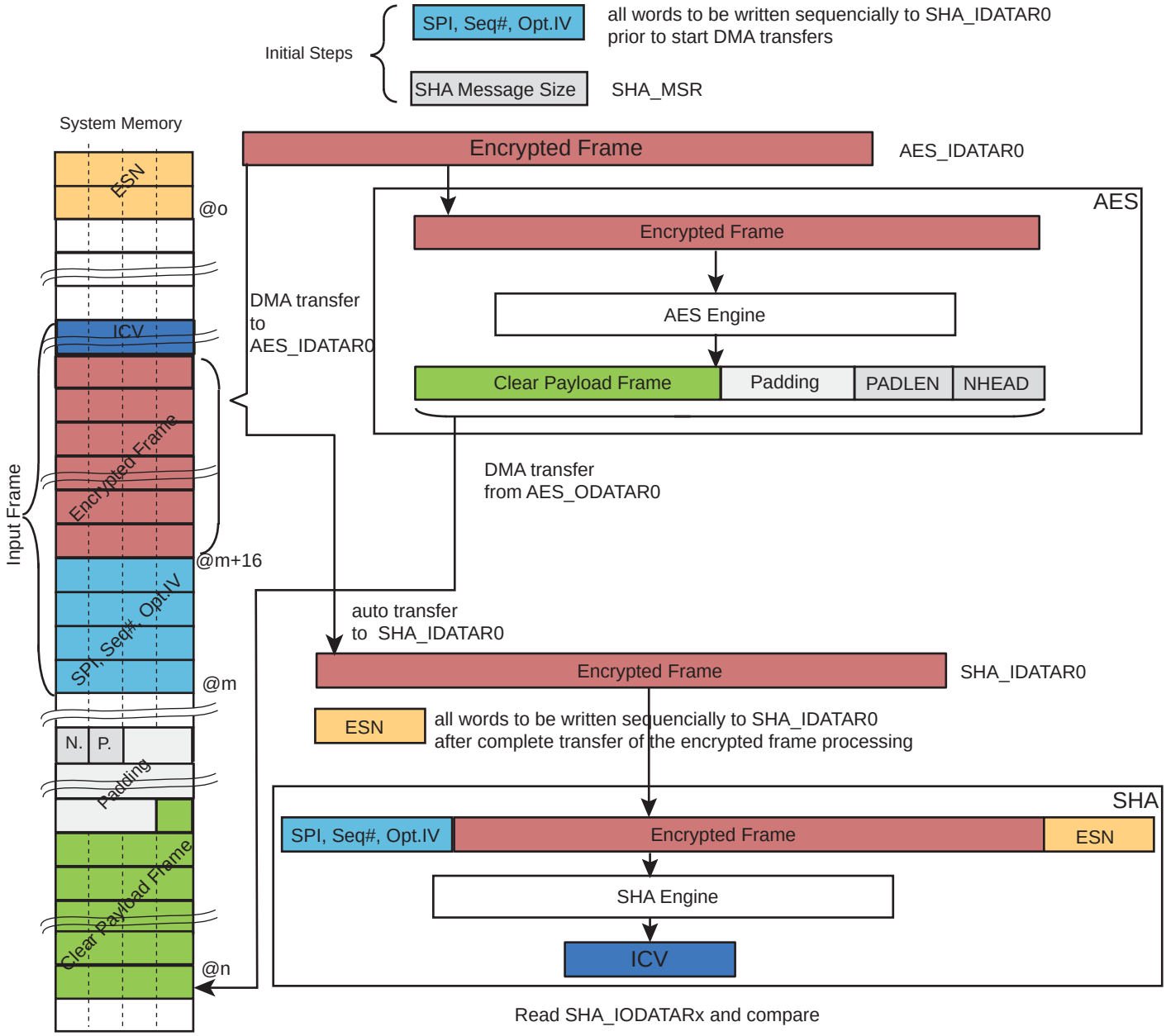


Figure 57-15. Decryption of an ESP IPSec Frame with ESN



57.4.10 Security Features

57.4.10.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD flag in the AES_ISR is raised. Its source is then reported in the Unspecified Register Access Type (URAT) field. Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data register written during the data processing when SMOD = IDATAR0_START
- Output Data register read during data processing
- Mode register written during data processing
- Output Data register read during subkeys generation
- Mode register written during subkeys generation
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in the AES_CR.

57.5 Advanced Encryption Standard (AES) User Interface

Table 57-5. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	AES_CR	Write-only	–
0x04	Mode Register	AES_MR	Read/Write	0x0
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	AES_IER	Write-only	–
0x14	Interrupt Disable Register	AES_IDR	Write-only	–
0x18	Interrupt Mask Register	AES_IMR	Read-only	0x0
0x1C	Interrupt Status Register	AES_ISR	Read-only	0x0
0x20	Key Word Register 0	AES_KEYWR0	Write-only	–
0x24	Key Word Register 1	AES_KEYWR1	Write-only	–
0x28	Key Word Register 2	AES_KEYWR2	Write-only	–
0x2C	Key Word Register 3	AES_KEYWR3	Write-only	–
0x30	Key Word Register 4	AES_KEYWR4	Write-only	–
0x34	Key Word Register 5	AES_KEYWR5	Write-only	–
0x38	Key Word Register 6	AES_KEYWR6	Write-only	–
0x3C	Key Word Register 7	AES_KEYWR7	Write-only	–
0x40	Input Data Register 0	AES_IDATAR0	Write-only	–
0x44	Input Data Register 1	AES_IDATAR1	Write-only	–
0x48	Input Data Register 2	AES_IDATAR2	Write-only	–
0x4C	Input Data Register 3	AES_IDATAR3	Write-only	–
0x50	Output Data Register 0	AES_ODATAR0	Read-only	0x0
0x54	Output Data Register 1	AES_ODATAR1	Read-only	0x0
0x58	Output Data Register 2	AES_ODATAR2	Read-only	0x0
0x5C	Output Data Register 3	AES_ODATAR3	Read-only	0x0
0x60	Initialization Vector Register 0	AES_IVR0	Write-only	–
0x64	Initialization Vector Register 1	AES_IVR1	Write-only	–
0x68	Initialization Vector Register 2	AES_IVR2	Write-only	–
0x6C	Initialization Vector Register 3	AES_IVR3	Write-only	–
0x70	Additional Authenticated Data Length Register	AES_AADLENR	Read/Write	–
0x74	Plaintext/Ciphertext Length Register	AES_CLENR	Read/Write	–
0x78	GCM Intermediate Hash Word Register 0	AES_GHASHR0	Read/Write	–
0x7C	GCM Intermediate Hash Word Register 1	AES_GHASHR1	Read/Write	–
0x80	GCM Intermediate Hash Word Register 2	AES_GHASHR2	Read/Write	–
0x84	GCM Intermediate Hash Word Register 3	AES_GHASHR3	Read/Write	–
0x88	GCM Authentication Tag Word Register 0	AES_TAGR0	Read-only	–
0x8C	GCM Authentication Tag Word Register 1	AES_TAGR1	Read-only	–

Table 57-5. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0x90	GCM Authentication Tag Word Register 2	AES_TAGR2	Read-only	_
0x94	GCM Authentication Tag Word Register 3	AES_TAGR3	Read-only	_
0x98	GCM Encryption Counter Value Register	AES_CTRR	Read-only	_
0x9C	GCM H Word Register 0	AES_GCMHR0	Read/Write	_
0xA0	GCM H Word Register 1	AES_GCMHR1	Read/Write	_
0xA4	GCM H Word Register 2	AES_GCMHR2	Read/Write	_
0xA8	GCM H Word Register 3	AES_GCMHR3	Read/Write	_
0xAC	Reserved	–	_	_
0xB0	Extended Mode Register	AES_EMR	Read/Write	0x0
0xB4	Byte Counter Register	AES_BCNT	Read/Write	0x0
0xC0	Tweak Word Register 0	AES_TWR0	Read/Write	0x0
0xC4	Tweak Word Register 1	AES_TWR1	Read/Write	0x0
0xC8	Tweak Word Register 2	AES_TWR2	Read/Write	0x0
0xCC	Tweak Word Register 3	AES_TWR3	Read/Write	0x0
0xD0	Alpha Word Register 0	AES_ALPHAR0	Write-only	_
0xD4	Alpha Word Register 1	AES_ALPHAR1	Write-only	_
0xD8	Alpha Word Register 2	AES_ALPHAR2	Write-only	_
0xDC	Alpha Word Register 3	AES_ALPHAR3	Write-only	_
0xE0	Reserved	–	_	_
0xE4–0xF8	Reserved	–	_	_
0xFC	Reserved	–	_	_

57.5.1 AES Control Register

Name: AES_CR

Address: 0xF002C000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SWRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	START

- **START: Start Processing**

0: No effect.

1: Starts manual encryption/decryption process.

- **SWRST: Software Reset**

0: No effect.

1: Resets the AES. A software-triggered hardware reset of the AES interface is performed.

57.5.2 AES Mode Register

Name: AES_MR

Address: 0xF002C004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CKEY				–	CFBS		
15	14	13	12	11	10	9	8
LOD	OPMOD			KEYSIZE		SMOD	
7	6	5	4	3	2	1	0
PROCDLY				DUALBUFF	–	GTAGEN	CIPHER

- **CIPHER: Processing Mode**

0: Decrypts data.

1: Encrypts data.

- **GTAGEN: GCM Automatic Tag Generation Enable**

0: Automatic GCM Tag generation disabled.

1: Automatic GCM Tag generation enabled.

- **DUALBUFF: Dual Input Buffer**

Value	Name	Description
0	INACTIVE	AES_IDATARx cannot be written during processing of previous block.
1	ACTIVE	AES_IDATARx can be written during processing of previous block when SMOD = 2. It speeds up the overall runtime of large files.

- **PROCDLY: Processing Delay**

Processing Time = N × (PROCDLY + 1)

where

N = 10 when KEYSIZE = 0

N = 12 when KEYSIZE = 1

N = 14 when KEYSIZE = 2

The processing time represents the number of clock cycles that the AES needs in order to perform one encryption/decryption.

Note: The best performance is achieved with PROCDLY equal to 0.

- **SMOD: Start Mode**

Value	Name	Description
0	MANUAL_START	Manual Mode
1	AUTO_START	Auto Mode
2	IDATAR0_START	AES_IDATAR0 access only Auto Mode (DMA)

Values which are not listed in the table must be considered as “reserved”.

If a DMA transfer is used, configure SMOD to 0x2. Refer to Section 57.4.4.3 “DMA Mode” for more details.

- **KEYSIZE: Key Size**

Value	Name	Description
0	AES128	AES Key Size is 128 bits
1	AES192	AES Key Size is 192 bits
2	AES256	AES Key Size is 256 bits

Values which are not listed in the table must be considered as “reserved”.

- **OPMOD: Operating Mode**

Value	Name	Description
0	ECB	ECB: Electronic Code Book mode
1	CBC	CBC: Cipher Block Chaining mode
2	OFB	OFB: Output Feedback mode
3	CFB	CFB: Cipher Feedback mode
4	CTR	CTR: Counter mode (16-bit internal counter)
5	GCM	GCM: Galois/Counter mode
6	XTS	XTS: XEX-based tweaked-codebook mode

Values which are not listed in the table must be considered as “reserved”.

For CBC-MAC operating mode, set OPMOD to CBC and LOD to 1.

- **LOD: Last Output Data Mode**

0: No effect.

After each end of encryption/decryption, the output data are available either on the output data registers (Manual and Auto modes) or at the address specified in the Channel Buffer Transfer Descriptor for DMA mode.

In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read.

1: The DATRDY flag is cleared when at least one of the Input Data Registers is written.

No more Output Data Register reads is necessary between consecutive encryptions/decryptions (see Section 57.4.5 “Last Output Data Mode”).

Warning: In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable results.

- **CFBS: Cipher Feedback Data Size**

Value	Name	Description
0	SIZE_128BIT	128-bit
1	SIZE_64BIT	64-bit
2	SIZE_32BIT	32-bit
3	SIZE_16BIT	16-bit
4	SIZE_8BIT	8-bit

Values which are not listed in the table must be considered as “reserved”.

- **CKEY: Key**

Value	Name	Description
0xE	PASSWD	This field must be written with 0xE the first time the AES_MR is programmed. For subsequent programming of the AES_MR, any value can be written, including that of 0xE. Always reads as 0.

57.5.3 AES Interrupt Enable Register

Name: AES_IER

Address: 0xF002C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	PLENERR	EOPAD	TAGRDY
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**
- **TAGRDY: GCM Tag Ready Interrupt Enable**
- **EOPAD: End of Padding Interrupt Enable**
- **PLENERR: Padding Length Error Interrupt Enable**

57.5.4 AES Interrupt Disable Register

Name: AES_IDR

Address: 0xF002C014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	PLENERR	EOPAD	TAGRDY
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Disable**
- **URAD: Unspecified Register Access Detection Interrupt Disable**
- **TAGRDY: GCM Tag Ready Interrupt Disable**
- **EOPAD: End of Padding Interrupt Disable**
- **PLENERR: Padding Length Error Interrupt Disable**

57.5.5 AES Interrupt Mask Register

Name: AES_IMR

Address: 0xF002C018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	PLENERR	EOPAD	TAGRDY
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **DATRDY: Data Ready Interrupt Mask**
- **URAD: Unspecified Register Access Detection Interrupt Mask**
- **TAGRDY: GCM Tag Ready Interrupt Mask**
- **EOPAD: End of Padding Interrupt Mask**
- **PLENERR: Padding Length Error Interrupt Mask**

57.5.6 AES Interrupt Status Register

Name: AES_ISR

Address: 0xF002C01C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	PLENERR	EOPAD	TAGRDY
15	14	13	12	11	10	9	8
URAT				–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready (cleared by setting bit START or bit SWRST in AES_CR or by reading AES_ODATARx)**

0: Output data not valid.

1: Encryption or decryption process is completed.

Note: If AES_MR.LOD = 1: In Manual and Auto mode, the DATRDY flag can also be cleared by writing at least one AES_IDATARx.

- **URAD: Unspecified Register Access Detection Status (cleared by writing SWRST in AES_CR)**

0: No unspecified register access has been detected since the last SWRST.

1: At least one unspecified register access has been detected since the last SWRST.

- **URAT: Unspecified Register Access (cleared by writing SWRST in AES_CR)**

Value	Name	Description
0	IDR_WR_PROCESSING	Input Data register written during the data processing when SMOD = 0x2 mode.
1	ODR_RD_PROCESSING	Output Data register read during the data processing.
2	MR_WR_PROCESSING	Mode register written during the data processing.
3	ODR_RD_SUBKGEN	Output Data register read during the subkeys generation.
4	MR_WR_SUBKGEN	Mode register written during the subkeys generation.
5	WOR_RD_ACCESS	Write-only register read access.

Only the last Unspecified Register Access Type is available through the URAT field.

- **TAGRDY: GCM Tag Ready**

0: GCM Tag is not valid.

1: GCM Tag generation is complete (cleared by reading GCM Tag, starting another processing or when writing a new key).

- **EOPAD: End of Padding**

0: Padding is not over.

1: Padding phase is over.

- **PLENERR: Padding Length Error**

0: No Padding Length Error occurred.

1: Padding Length Error detected.

57.5.7 AES Key Word Register x

Name: AES_KEYWRx [x=0..7]

Address: 0xF002C020

Access: Write-only

31	30	29	28	27	26	25	24
KEYW							

23	22	21	20	19	18	17	16
KEYW							

15	14	13	12	11	10	9	8
KEYW							

7	6	5	4	3	2	1	0
KEYW							

- **KEYW: Key Word**

The four/six/eight 32-bit Key Word registers set the 128-bit/192-bit/256-bit cryptographic key used for AES encryption/decryption.

AES_KEYWR0 corresponds to the first word of the key and respectively AES_KEYWR3/AES_KEYWR5/AES_KEYWR7 to the last one.

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

See Section 57.4.6.2 “Key Writing and Automatic Hash Subkey Calculation” for details.

These registers are write-only to prevent the key from being read by another application.

57.5.8 AES Input Data Register x

Name: AES_IDATARx [x=0..3]

Address: 0xF002C040

Access: Write-only

31	30	29	28	27	26	25	24
IDATA							
23	22	21	20	19	18	17	16
IDATA							
15	14	13	12	11	10	9	8
IDATA							
7	6	5	4	3	2	1	0
IDATA							

- **IDATA: Input Data Word**

The four 32-bit Input Data registers set the 128-bit data block used for encryption/decryption.

AES_IDATAR0 corresponds to the first word of the data to be encrypted/decrypted, and AES_IDATAR3 to the last one.

These registers are write-only to prevent the input data from being read by another application.

57.5.9 AES Output Data Register x

Name: AES_ODATARx [x=0..3]

Address: 0xF002C050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							
23	22	21	20	19	18	17	16
ODATA							
15	14	13	12	11	10	9	8
ODATA							
7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The four 32-bit Output Data registers contain the 128-bit data block that has been encrypted/decrypted.

AES_ODATAR0 corresponds to the first word, AES_ODATAR3 to the last one.

57.5.10 AES Initialization Vector Register x

Name: AES_IVRx [x=0..3]

Address: 0xF002C060

Access: Write-only

31	30	29	28	27	26	25	24
IV							
23	22	21	20	19	18	17	16
IV							
15	14	13	12	11	10	9	8
IV							
7	6	5	4	3	2	1	0
IV							

- **IV: Initialization Vector**

The four 32-bit Initialization Vector registers set the 128-bit Initialization Vector data block that is used by some modes of operation as an additional initial input.

AES_IVR0 corresponds to the first word of the Initialization Vector, AES_IVR3 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

For CBC, OFB and CFB modes, the IV input value corresponds to the initialization vector.

For CTR mode, the IV input value corresponds to the initial counter value.

Note: These registers are not used in ECB mode and must not be written.

57.5.11 AES Additional Authenticated Data Length Register

Name: AES_AADLENR

Address: 0xF002C070

Access: Read/Write

31	30	29	28	27	26	25	24
AADLEN							
23	22	21	20	19	18	17	16
AADLEN							
15	14	13	12	11	10	9	8
AADLEN							
7	6	5	4	3	2	1	0
AADLEN							

- **AADLEN: Additional Authenticated Data Length**

Length in bytes of the Additional Authenticated Data (*AAD*) that is to be processed.

Note: The maximum byte length of the *AAD* portion of a message is limited to the 32-bit counter length.

57.5.12 AES Plaintext/Ciphertext Length Register

Name: AES_CLENR

Address: 0xF002C074

Access: Read/Write

31	30	29	28	27	26	25	24
CLEN							
23	22	21	20	19	18	17	16
CLEN							
15	14	13	12	11	10	9	8
CLEN							
7	6	5	4	3	2	1	0
CLEN							

- **CLEN: Plaintext/Ciphertext Length**

Length in bytes of the plaintext/ciphertext (*C*) data that is to be processed.

Note: The maximum byte length of the C portion of a message is limited to the 32-bit counter length.

57.5.13 AES GCM Intermediate Hash Word Register x

Name: AES_GHASHRx [x=0..3]

Address: 0xF002C078

Access: Read/Write

31	30	29	28	27	26	25	24
GHASH							
23	22	21	20	19	18	17	16
GHASH							
15	14	13	12	11	10	9	8
GHASH							
7	6	5	4	3	2	1	0
GHASH							

- **GHASH: Intermediate GCM Hash Word x**

The four 32-bit Intermediate Hash Word registers expose the intermediate GHASH value. May be read to save the current GHASH value so processing can later be resumed, presumably on a later message fragment. Whenever a new key (AES_KEYWRx) is written to the hardware two automatic actions are processed:

- GCM hash subkey generation
- AES_GHASHRx Clear

See Section 57.4.6.2 “Key Writing and Automatic Hash Subkey Calculation” for details.

If an application software-specific hash initial value is needed for the GHASH, it must be written to the AES_GHASHRx:

- after a write to AES_KEYWRx, if any,
- before starting the input data feed.

57.5.14 AES GCM Authentication Tag Word Register x

Name: AES_TAGRx [x=0..3]

Address: 0xF002C088

Access: Read-only

31	30	29	28	27	26	25	24
TAG							

23	22	21	20	19	18	17	16
TAG							

15	14	13	12	11	10	9	8
TAG							

7	6	5	4	3	2	1	0
TAG							

- **TAG: GCM Authentication Tag x**

The four 32-bit Tag registers contain the final 128-bit GCM Authentication tag (*T*) when GCM processing is complete. TAG0 corresponds to the first word, TAG3 to the last word.

57.5.15 AES GCM Encryption Counter Value Register

Name: AES_CTRR

Address: 0xF002C098

Access: Read-only

31	30	29	28	27	26	25	24
CTR							
23	22	21	20	19	18	17	16
CTR							
15	14	13	12	11	10	9	8
CTR							
7	6	5	4	3	2	1	0
CTR							

- **CTR: GCM Encryption Counter**

Reports the current value of the 32-bit GCM counter.

57.5.16 AES GCM H Word Register x

Name: AES_GCMHRx [x=0..3]

Address: 0xF002C09C

Access: Read/Write

31	30	29	28	27	26	25	24
H							

23	22	21	20	19	18	17	16
H							

15	14	13	12	11	10	9	8
H							

7	6	5	4	3	2	1	0
H							

- **H: GCM H Word x**

The four 32-bit H Word registers contain the 128-bit GCM hash subkey *H* value.

Whenever a new key (AES_KEYWRx) is written to the hardware, two automatic actions are processed:

- GCM hash subkey *H* generation
- AES_GHASHRx Clear

If the application software requires a specific hash subkey, the automatically-generated H value can be overwritten in the AES_GCMHRx. See Section 57.4.6.2 “Key Writing and Automatic Hash Subkey Calculation” for details.

Generating a GCM hash subkey *H* by a write in the AES_GCMHRx enables to:

- select the GCM hash subkey *H* for GHASH operations,
- select one operand to process a single GF128 multiply.

57.5.17 AES Extended Mode Register

Name: AES_EMR

Address: 0xF002C0B0

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
NHEAD							
15	14	13	12	11	10	9	8
PADLEN							
7	6	5	4	3	2	1	0
–	–	PLIPD	PLIPEN	–	–	APM	APEN

- **APEN: Auto Padding Enable**

0: Auto Padding feature is disabled.

1: Auto Padding feature is enabled.

- **APM: Auto Padding Mode**

0: Auto Padding performed according to IPSEC standard.

1: Auto Padding performed according to SSL standard.

- **PLIPEN: Protocol Layer Improved Performance Enable**

0: Protocol layer improved performance is disabled.

1: Protocol layer improved performance is enabled.

- **PLIPD: Protocol Layer Improved Performance Decipher**

0: Protocol layer improved performance is in ciphering mode.

1: Protocol layer improved performance is in deciphering mode.

- **PADLEN: Auto Padding Length**

0–255: Padding Length in bytes

- **NHEAD: IPSEC Next Header**

0–255: IPSEC Next Header field

57.5.18 AES Byte Counter Register

Name: AES_BCNT

Address: 0xF002C0B4

Access: Read/Write

31	30	29	28	27	26	25	24
BCNT							
23	22	21	20	19	18	17	16
BCNT							
15	14	13	12	11	10	9	8
BCNT							
7	6	5	4	3	2	1	0
BCNT							

- **BCNT: Auto Padding Byte Counter**

Auto padding byte counter value. BCNT must be greater than 0.

57.5.19 AES Tweak Word Register x

Name: AES_TWRx [x=0..3]

Address: 0xF002C0C0

Access: Read/Write

31	30	29	28	27	26	25	24
TWEAK							

23	22	21	20	19	18	17	16
TWEAK							

15	14	13	12	11	10	9	8
TWEAK							

7	6	5	4	3	2	1	0
TWEAK							

- **TWEAK: Tweak Word x**

The four 32-bit Tweak Word registers contain the 128-bit Tweak value.

57.5.20 AES Alpha Word Register x

Name: AES_ALPHARx [x=0..3]

Address: 0xF002C0D0

Access: Write-only

31	30	29	28	27	26	25	24
ALPHA							
23	22	21	20	19	18	17	16
ALPHA							
15	14	13	12	11	10	9	8
ALPHA							
7	6	5	4	3	2	1	0
ALPHA							

- **ALPHA: Alpha Word x**

The four 32-bit Alpha Word registers contain the 128-bit primitive of GF(2^{128}) to use for the first processing.

58. Secure Hash Algorithm (SHA)

58.1 Description

The Secure Hash Algorithm (SHA) is compliant with the American *FIPS (Federal Information Processing Standard) Publication 180-2* specification.

The 512/1024-bit block of message is respectively stored in 16/32 x 32-bit registers, (SHA_IDATARx/SHA_IODATARx) which are write-only.

As soon as the input data is written, the hash processing may be started. The registers comprising the block of a padded message must be entered consecutively. Then the message digest is ready to be read out on the 5 up to 8/16 x 32-bit output data registers (SHA_IODATARx) or through the DMA channels.

58.2 Embedded Characteristics

- Supports Secure Hash Algorithm (SHA1, SHA224, SHA256, SHA384, SHA512)
- Supports Hash-based Message Authentication Code (HMAC) algorithm (HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512)
- Compliant with *FIPS Publication 180-2*
- Supports automatic padding of messages
- Supports up to 2 sets of initial hash values registers (HMAC acceleration or other)
- Supports automatic check of the hash (HMAC acceleration or other)
- Tightly coupled to AES for protocol layers improved performances
- Configurable Processing Period:
  - 85 Clock Cycles to obtain a fast SHA1 runtime, 88 clock cycles for SHA384, SHA512 or 209 Clock Cycles for Maximizing Bandwidth of Other Applications
  - 72 Clock Cycles to obtain a fast SHA224, SHA256 runtime or 194 Clock Cycles for Maximizing Bandwidth of Other Applications
- Connection to DMA Channel Capabilities Optimizes Data Transfers
- Double Input Buffer Optimizes Runtime

58.3 Product Dependencies

58.3.1 Power Management

The SHA may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the SHA clock.

58.3.2 Interrupt Sources

The SHA interface has an interrupt line connected to the Interrupt Controller.

Handling the SHA interrupt requires programming the interrupt controller before configuring the SHA.

Table 58-1. Peripheral IDs

Instance	ID
SHA	12

58.4 Functional Description

The Secure Hash Algorithm (SHA) module requires a padded message according to FIPS180-2 specification.This message can be provided with the padding to the SHA module, or the padding can be automatically computed by the SHA module if the size of the message is provided. The first block of the message must be indicated to the module by a specific command. The SHA module produces an N-bit message digest each time a block is written and processing period ends, where N is 160 for SHA1, 224 for SHA224,256 for SHA256, 384 for SHA384, 512 for SHA512. The SHA module is also capable of computing Hash-based Message Authentication Code (HMAC) algorithm.

58.4.1 SHA Algorithm

The SHA can process SHA1, SHA224, SHA256, SHA384, SHA512 by configuring the ALGO field in the SHA Mode register (SHA_MR).

58.4.2 HMAC Algorithm

The HMAC algorithm is as follows:

$HMAC_K(m) = h((K_0 \oplus opad) || h((K_0 \oplus ipad) || m))$

where:

- h = SHA function
- K_0 = the key K after any necessary preprocessing to form a block size key
- m = message to authenticate
- || = concatenation operator
- $\oplus$ = XOR operator
- ipad = predefined constant (0x3636...3636)
- opad = predefined constant (0x5C5C...5C5C)

The SHA provides a fully optimized processing of the HMAC algorithm by executing the following operations:

- starting the SHA algorithm from any user predefined hash value, thus ‘$h(K_0 \oplus ipad)$’ for first HMAC hash and ‘$h(K_0 \oplus opad)$’ for second HMAC hash
- performing automatic padding
- routing automatically the first hash result ‘$h((K_0 \oplus ipad) || m)$’ to the source of the second hash processing ‘$h((K_0 \oplus opad) || (\text{first hash result}))$’ including the concatenation of the first hash result to ‘$K_0 \oplus opad$’.

To perform the HMAC operation, the ALGO field value must be greater than 7, the automatic padding feature must be enabled (MSGSIZE and BYTCNT fields differ from 0) and the SHA internal initial hash value registers 0 and 1 must be configured, respectively, with the hash results of input blocks “$K_0 \oplus ipad$” and “$K_0 \oplus opad$” (refer to Section 58.4.5 “Internal Registers for Initial Hash Value or Expected Hash Result”).

The size of the message (‘m’) must be written in the MSGSIZE and BYTCNT fields.

The FIRST bit in the SHA Control register (SHA_CR) should be set before writing the first block of the message.

The SHA can process HMAC-SHA1, HMAC-SHA224, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512 by configuring the ALGO field in the SHA_MR.

58.4.3 Processing Period

The processing period can be configured.

The short processing period allocates bandwidth to the SHA module, whereas the long processing period allocates more bandwidth on the system bus to other applications. An example is DMA channels not associated with SHA.

In SHA1 mode, the shortest processing period is 85 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA384, SHA512 mode, the shortest processing period is 88 clock cycles + 2 clock cycles for start command synchronization. The longest period is 209 clock cycles + 2 clock cycles.

In SHA256 and SHA224 mode, the shortest processing period is 72 clock cycles + 2 clock cycles for start command synchronization. The longest period is 194 clock cycles + 2 clock cycles.

58.4.4 Double Input Buffer

The SHA Input Data registers (SHA_IDATARx) can be double-buffered to reduce the runtime of large files.

Double-buffering allows a new message block to be written while the previous message block is being processed. This is only possible when DMA accesses are performed (SMOD = 2).

The DUALBUFF bit in the SHA_MR must be set to have double input buffer access.

58.4.5 Internal Registers for Initial Hash Value or Expected Hash Result

The SHA module embeds two sets of internal registers (IR0, IR1) to store different data used by the SHA or HMAC algorithms (See Figure 58-1). These internal registers are accessed through SHA Input Data registers (SHA_IDATARx).

When the ALGO field selects SHA algorithms, IR0 can be configured with a user initial hash value. This initial hash value can be used to compute a custom hash algorithm with two sets of different initial constants, or to continue a hash computation by providing the intermediate hash value previously returned by the SHA module.

When the ALGO field selects SHA algorithms, IR1 can be configured with either a user initial hash value or an expected hash result. The expected hash result must be configured in the IR1 if the field CHECK = 1 (See Section 58.4.7 “Automatic Check”). If the field CHECK = 0 or 2, IR1 can be configured with a user initial hash value that differs from IR0 value.

When the ALGO field selects HMAC algorithms, IR0 must be configured with the hash result of $K_0 \oplus$ ipad and IR1 must be configured with the hash result of $K_0 \oplus$ opad. These precomputed first blocks speed up the HMAC computation by saving the time to compute the intermediate hash values of the first block which is constant while the secret key is constant (See Section 58.4.2 “HMAC Algorithm”).

Table 58-2. Configuration Values of Internal Registers

	SHA modes (ALGO < 8)			HMAC modes (ALGO > 7)
	CHECK = 0	CHECK = 1	CHECK = 2	
IR0	User Initial Hash	User Initial Hash	User Initial Hash	hash($K_0 \oplus$ ipad)
IR1	User Initial Hash	Expected Hash Result	User Initial Hash	hash($K_0 \oplus$ opad)

To calculate the initial HMAC values, follow this sequence:

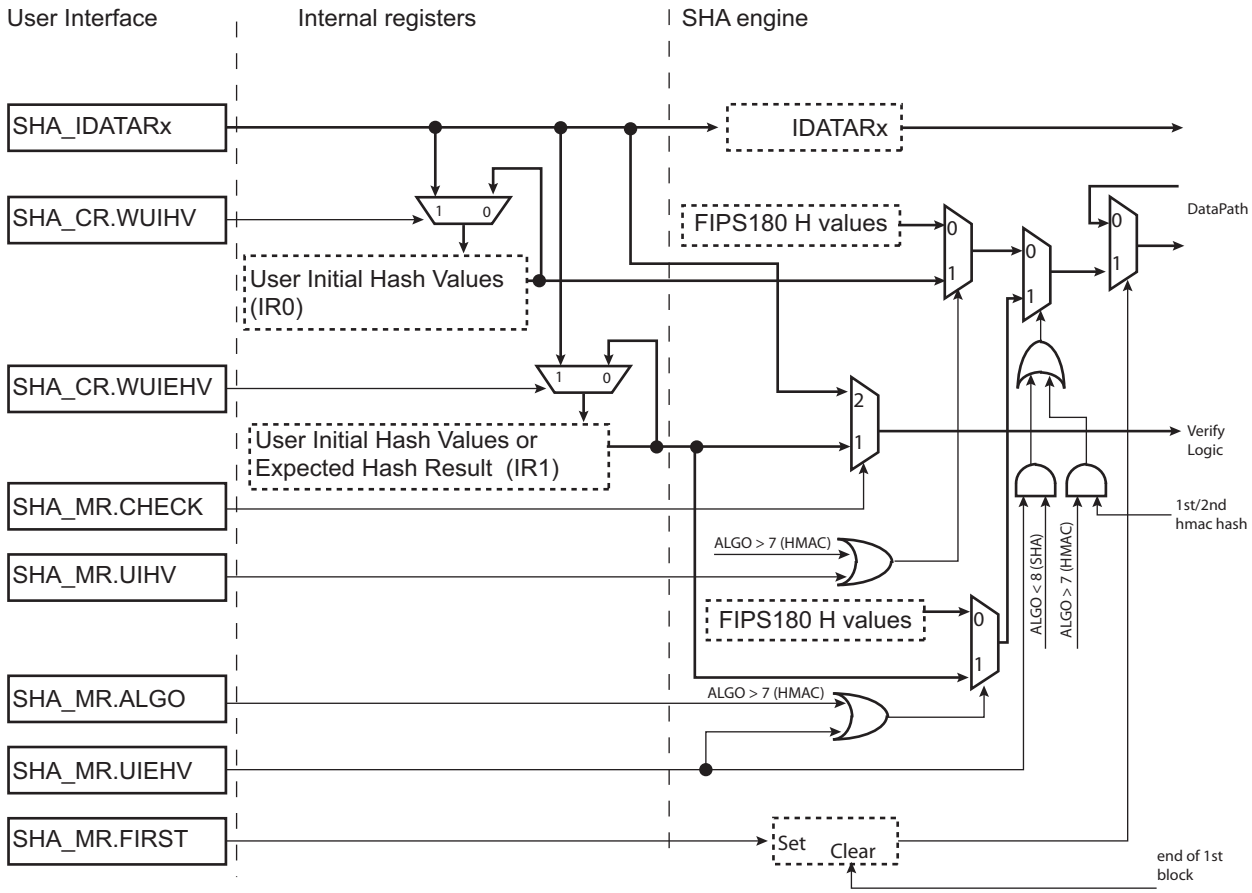
1. Calculate K_0.
2. Calculate $K_0 \oplus$ ipad and $K_0 \oplus$ opad.
3. Perform a hash of the result of $K_0 \oplus$ ipad and $K_0 \oplus$ opad (auto-padding must be disabled for that type of hash).
4. Write h($K_0 \oplus$ ipad) and h($K_0 \oplus$ opad) in IR0 and IR1 respectively.

To write IR0 or IR1, follow this sequence:

1. Set bit WUIHV (IR0) or WUIEHV (IR1) in SHA_CR
2. Write the data in SHA_IDATARx. The number of registers to write depends on the type of data (user initial hash values or expected hash result) and on the type of algorithm selected:
   - SHA_IDATAR0 to SHA_IDATAR4 for data used in algorithms based on SHA1
   - SHA_IDATAR0 to SHA_IDATAR7 for data used in algorithms based on SHA256
   - SHA_IDATAR0 to SHA_IDATAR15 for data used in algorithms based on SHA512
   - SHA_IDATAR0 to SHA_IDATAR6 for expected hash result of algorithms based on SHA224
   - SHA_IDATAR0 to SHA_IDATAR11 for expected hash result of algorithms based on SHA384
3. Clear bit WUIHV or WUIEHV in SHA_CR.

IR0 and IR1 are automatically selected for HMAC processing if the field ALGO selects HMAC algorithms. If SHA algorithms are selected, the internal registers are selected if the corresponding UIHV or UIEHV bits are set.

Figure 58-1. User Initial Hash Value and Expected Hash Internal Register Access



58.4.6 Automatic Padding

The SHA module features an automatic padding computation to speed up the execution of the algorithm.

The automatic padding function requires the following information:

- Complete message size in bytes to be written in the MSGSIZE field of the SHA Message Size register (SHA_MSR).
  The size of the message is written at the end of the last block, as required by the FIPS180-2 specification (the size is automatically converted into a bit-size).
- Number of remaining bytes (to write in the SHA_IDATARx) to be written in the BYTCNT field of the SHA Bytes Count register (SHA_BCR).
  Automatic padding occurs when the BYTCNT field reaches 0. At each write in the SHA Input registers, the BYTCNT field value is decreased by the number of bytes written.

The BYTCNT field value must be written with the same value as the MSGSIZE field value if the full message is processed. If the message is partially preprocessed and an initial hash value is used, BYTCNT must be written with the remaining bytes to hash while MSGSIZE holds the message size.

To disable the automatic padding feature, both the MSGSIZE and BYTCNT fields must be configured with 0.

58.4.7 Automatic Check

The SHA module features an automatic check of the hash result with the expected hash. A check failure can generate an interrupt if configured in the SHA Interrupt Enable register (SHA_IER).

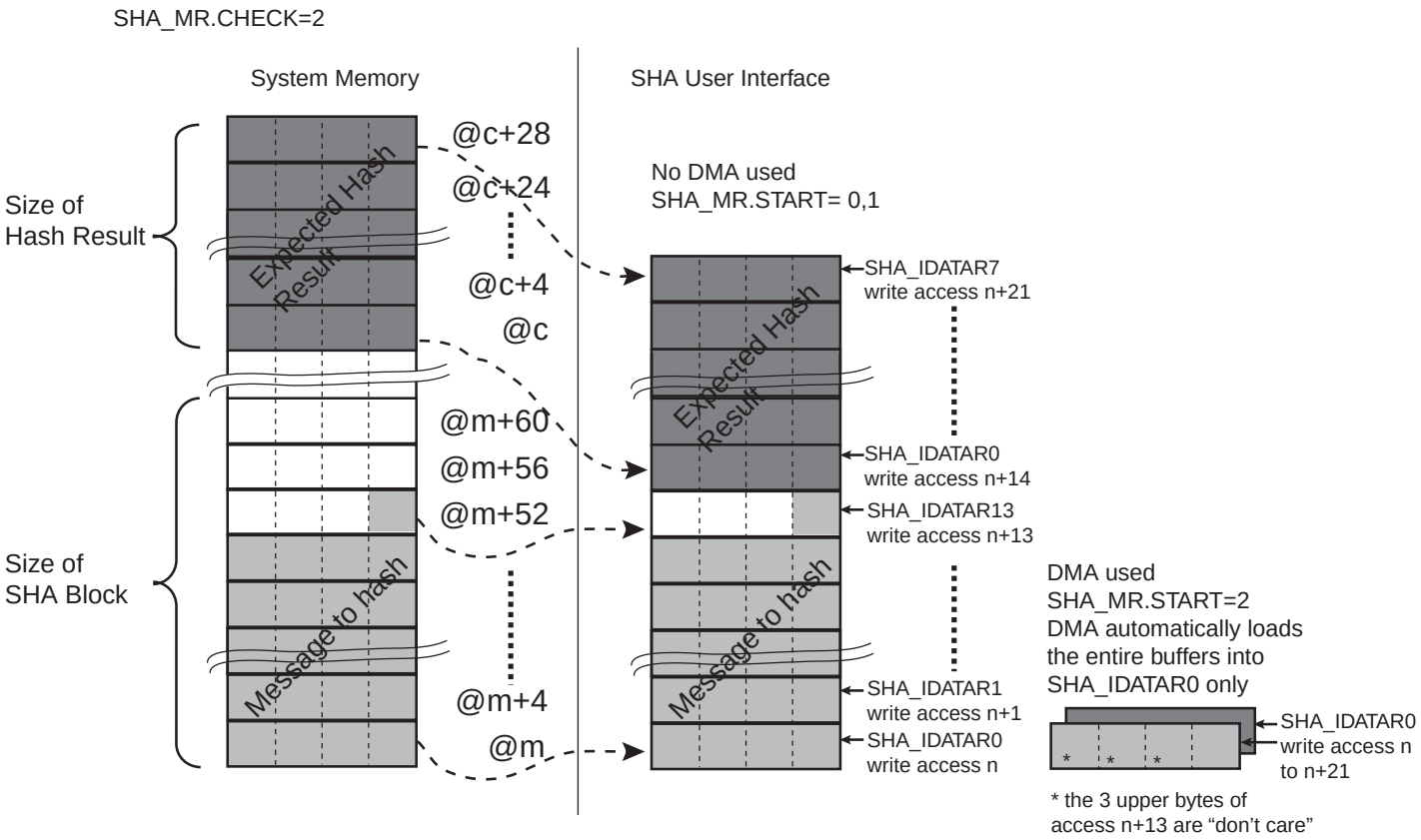
Automatic check requires the automatic padding feature to be enabled (MSGSIZE and BYTCNT fields must be greater than 0).

There are two methods to configure the expected hash result:

- if the field CHECK = 1, the expected hash result is read from the internal register (IR1). This method cannot be used when HMAC algorithms is selected because this register is already used to store user initial hash values for the second hash processing. IR1 cannot be read by software.
- If the field CHECK = 2, the expected hash result is written in the SHA_IDATARx after the message.

When CHECK = 2, the method can provide more flexibility of use if a message is stored in system memory together with its expected hash result. A DMA with linked list can be used to ease the transfer of the message and its expected hash result.

Figure 58-2. Message and Expected Hash Result Memory Mapping



The number of 32-bit words of the hash result to check with the expected hash can be selected with the CHKCNT field in the SHA_MR. The status of the check is available in the CHKST field in the SHA Interrupt Status Register (SHA_ISR).

An interrupt can be generated (if enabled) when the check is completed. The check occurs several clock cycles after the computation of the requested hash, so the interrupt and the CHECKF bit are set several clock cycles after the DATRDY flag of the SHA_ISR.

58.4.8 Protocol Layers Improved Performances

The SHA can be tightly coupled to the AES module to improve performances when processing protocol layers such as IPsec or OpenSSL.

When the AES is configured to be tightly coupled to SHA (AES_MR), SHA must be always configured in Double Buffer mode (DUALBUFF = 1 in SHA_MR).

Refer to section AES for details.

58.4.9 Start Modes

The SMOD field in the SHA_MR is used to select the Hash Processing Start mode.

58.4.9.1 Manual Mode

In Manual mode, the sequence is as follows:

1. Set the bit DATRDY (Data Ready) in the SHA_IER, depending on whether an interrupt is required at the end of processing.
2. If the initial hash values differ from the FIPS standard, set the bits UIHV and UIEHV in the SHA_MR depending on the configure the initial values.
   If the initial hash values comply with the FIPS180-2 specification, clear the bits UIHV and UIEHV in the SHA_MR.
3. If automatic padding is required, configure the MSGSIZE field in the SHA_MSR with the number of bytes of the message, and configure the BYTCNT field in the SHA_BCR with the remaining number of bytes to write. The BYTCNT field must be written with a value different from MSGSIZE field value if the message is preprocessed and completed by using user initial hash values.
   If automatic padding is not required, configure the MSGSIZE field in the SHA_MSR and the BYTCNT field in the SHA_BCR to 0.
4. For the first block of a message, the FIRST command must be set by writing a 1 into the corresponding bit of the SHA Control Register (SHA_CR). For the other blocks, there is nothing to write.
5. Write the block to be processed in the SHA_IDATARx.
6. To begin processing, set the START bit in the SHA_CR.
7. When processing is completed, the bit DATRDY in the Interrupt Status register (SHA_ISR) raises. If an interrupt has been enabled by setting the bit DATRDY in SHA_IER, the interrupt line of the SHA is activated.
8. Repeat the write procedure for each block, start procedure and wait for the interrupt procedure up to the last block of the entire message. Each time the start procedure is complete, the DATRDY flag is cleared.
9. After the last block is processed (DATRDY flag is set, if an interrupt has been enabled by setting the bit DATRDY in SHA_IER, the interrupt line of the SHA is activated), read the message digest in the Output Data Registers. The DATRDY flag is automatically cleared when reading the SHA_IODATARx registers.

58.4.9.2 Auto Mode

In Auto mode, processing starts as soon as the correct number of SHA_IDATARx is written. No action in the SHA_CR is necessary.

58.4.9.3 DMA Mode

The DMA can be used in association with the SHA to perform the algorithm on a complete message without any action by the software during processing.

The SMOD field in SHA_MR must be configured to 2.

The DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be set to point to the SHA_IDATAR0.

The DMA chunk size must be set to transfer, for each trigger request, 16 words of 32 bits.

The FIRST bit of the SHA_CR must be set before starting the DMA when the first block is transferred.

The DMA generates an interrupt when the end of buffer transfer is completed but the SHA processing is still in progress. The end of SHA processing is indicated by the flag DATRDY in the SHA_SR.
If automatic padding is disabled, the end of SHA processing requires two interrupts to be verified. The DMA end of transfer interrupt must be verified first, then the SHA DATRDY interrupt must be enabled and verified (see Figure 58-3).

If automatic padding is enabled, the end of SHA processing requires only one interrupt to be verified (see Figure 58-4). The DMA end of transfer is not required, so the SHA DATRDY interrupt must be enabled prior to start the DMA and DATRDY interrupt is the only one to be verified.

Figure 58-3. interrupts Processing with DMA

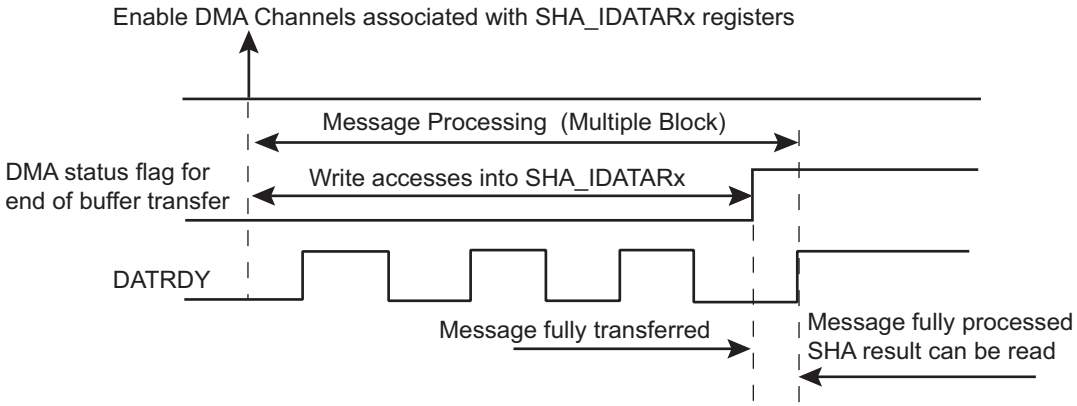
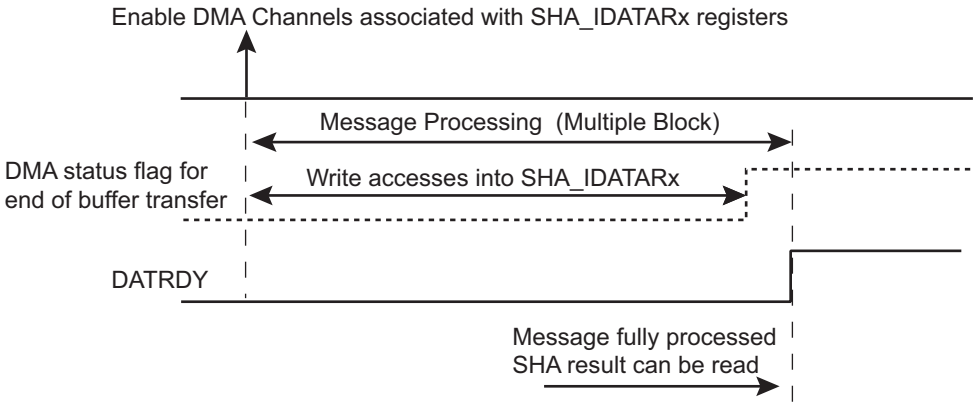


Figure 58-4. interrupts processing with DMA and automatic padding



58.4.9.4 SHA Register Endianism

In ARM processor-based products, the system bus and processors manipulate data in little-endian form. The SHA interface requires little-endian format words. However, in accordance with the protocol of FIPS 180-2 specification, data is collected, processed and stored by the SHA algorithm in big-endian form.

The following example illustrates how to configure the SHA:

If the first 64 bits of a message (according to FIPS 180-2, i.e., big-endian format) to be processed is 0xcafedeca_01234567, then the SHA_IDATAR0 and SHA_IDATAR1 registers must be written with the following pattern:

- SHA_IDATAR0 = 0xcadefeca
- SHA_IDATAR1 = 0x67452301

In a little-endian system, the message (according to FIPS 180-2) starting with pattern 0xcafedeca_01234567 is stored into memory as follows:

- 0xca stored at initial offset (for example 0x00),
- then 0xfe stored at initial offset + 1 (i.e., 0x01),
- 0xde stored at initial offset + 2 (i.e., 0x02),
- 0xca stored at initial offset + 3 (i.e., 0x03).

If the message is received through a serial-to-parallel communication channel, the first received character is 0xca and it is stored at the first memory location (initial offset). The second byte, 0xfe, is stored at initial offset + 1.

When reading on a 32-bit little-endian system bus, the first word read back from system memory is 0xcadefeca.

When the SHA_IODATARx registers are read, the hash result is organized in little-endian format, allowing system memory storage in the same format as the message.

Taking an example from the FIPS 180-2 specification Appendix B.1, the endianism conversion can be observed.

For this example, the 512-bit message is:

0x6162638000000000

58.5 Secure Hash Algorithm (SHA) User Interface

Table 58-3. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	SHA_CR	Write-only	–
0x04	Mode Register	SHA_MR	Read/Write	0x0000100
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	SHA_IER	Write-only	–
0x14	Interrupt Disable Register	SHA_IDR	Write-only	–
0x18	Interrupt Mask Register	SHA_IMR	Read-only	0x0
0x1C	Interrupt Status Register	SHA_ISR	Read-only	0x0
0x20	Message Size Register	SHA_MSR	Read/Write	0x0
0x24–0x2C	Reserved	–	–	–
0x30	Bytes Count Register	SHA_BCR	Read/Write	0x0
0x34–0x3C	Reserved	–	–	–
0x40	Input Data 0 Register	SHA_IDATAR0	Write-only	–
...	...	...	...	...
0x7C	Input Data 15 Register	SHA_IDATAR15	Write-only	–
0x80	Input/Output Data 0 Register	SHA_IODATAR0	Read/Write	0x0
...	...	...	...	...
0x9C	Input/Output Data 7 Register	SHA_IODATAR7	Read/Write	0x0
0xA0	Input/Output Data 8 Register	SHA_IODATAR8	Read/Write	0x0
...	...	...	...	...
0xBC	Input/Output Data 15 Register	SHA_IODATAR15	Read/Write	0x0
0xC0–0xFC	Reserved	–	–	–

58.5.1 SHA Control Register

Name: SHA_CR

Address: 0xF0028000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	WUIEHV	WUIHV	–	–	–	SWRST
7	6	5	4	3	2	1	0
–	–	–	FIRST	–	–	–	START

- **START: Start Processing**

0: No effect.

1: Starts manual hash algorithm process.

- **FIRST: First Block of a Message**

0: No effect.

1: Indicates that the next block to process is the first one of a message.

- **SWRST: Software Reset**

0: No effect.

1: Resets the SHA. A software-triggered hardware reset of the SHA interface is performed.

- **WUIHV: Write User Initial Hash Values**

0: SHA_IDATARx accesses are routed to the data registers.

1: SHA_IDATARx accesses are routed to the internal registers (IR0).

- **WUIEHV: Write User Initial or Expected Hash Values**

0: SHA_IDATARx accesses are routed to the data registers.

1: SHA_IDATARx accesses are routed to the internal registers (IR1).

58.5.2 SHA Mode Register

Name: SHA_MR

Address: 0xF0028004

Access: Read/Write

31	30	29	28	27	26	25	24
CHKCNT				–	–	CHECK	

23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	DUALBUFF

15	14	13	12	11	10	9	8
–	–	–	–	ALGO			

7	6	5	4	3	2	1	0
–	UIEHV	UIHV	PROCDLY	–	–	SMOD	

- **SMOD: Start Mode**

Value	Name	Description
0	MANUAL_START	Manual Mode
1	AUTO_START	Auto Mode
2	IDATAR0_START	SHA_IDATAR0 access only Auto Mode

Values not listed in the table must be considered as “reserved”.

If a DMA transfer is used, configure the SMOD value with 1 or 2. Refer to Section 58.4.9.3 “DMA Mode” for more details.

- **PROCDLY: Processing Delay**

Value	Name	Description
0	SHORTEST	SHA processing runtime is the shortest one
1	LONGEST	SHA processing runtime is the longest one (reduces the SHA bandwidth requirement, reduces the system bus overload)

When SHA1 algorithm is processed, runtime period is either 85 or 209 clock cycles.

When SHA256 or SHA224 algorithm is processed, runtime period is either 72 or 194 clock cycles.

When SHA384 or SHA512 algorithm is processed, runtime period is either 88 or 209 clock cycles.

- **UIHV: User Initial Hash Value Registers**

0: The SHA algorithm is started with the standard initial values as defined in the FIPS180-2 specification.

1: The SHA algorithm is started with the user initial hash values stored in the internal register 0 (IR0). If HMAC is configured, UIHV has no effect (i.e. IR0 is selected).

- **UIEHV: User Initial or Expected Hash Value Registers**

0: The SHA algorithm is started with the standard initial values as defined in the FIPS180-2 specification.

1: The SHA algorithm is started with the user initial hash values stored in the internal register 1 (IR1). If HMAC is configured, UIEHV has no effect (i.e. IR1 is always selected).

- **ALGO: SHA Algorithm**

Value	Name	Description
0	SHA1	SHA1 algorithm processed
1	SHA256	SHA256 algorithm processed
2	SHA384	SHA384 algorithm processed
3	SHA512	SHA512 algorithm processed
4	SHA224	SHA224 algorithm processed
8	HMAC_SHA1	HMAC algorithm with SHA1 Hash processed
9	HMAC_SHA256	HMAC algorithm with SHA256 Hash processed
10	HMAC_SHA384	HMAC algorithm with SHA384 Hash processed
11	HMAC_SHA512	HMAC algorithm with SHA512 Hash processed
12	HMAC_SHA224	HMAC algorithm with SHA224 Hash processed

Values not listed in the table must be considered as “reserved”.

- **DUALBUFF: Dual Input Buffer**

Value	Name	Description
0	INACTIVE	SHA_IDATARx and SHA_IODATARx cannot be written during processing of previous block.
1	ACTIVE	SHA_IDATARx and SHA_IODATARx can be written during processing of previous block when SMOD value = 2. It speeds up the overall runtime of large files.

- **CHECK: Hash Check**

Value	Name	Description
0	NO_CHECK	No check is performed
1	CHECK_EHV	Check is performed with expected hash stored in internal expected hash value registers.
2	CHECK_MESSAGE	Check is performed with expected hash provided after the message.

Values not listed in table must be considered as “reserved”.

- **CHKCNT: Check Counter**

Number of 32-bit words to check. The value 0 indicates that the number of words to compare will be based on the algorithm selected (5 words for SHA1, 7 words for SHA224, 8 words for SHA256, 12 words for SHA384, 16 words for SHA512).

58.5.3 SHA Interrupt Enable Register

Name: SHA_IER

Address: 0xF0028010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	CHECKF
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Enable**
- **URAD: Unspecified Register Access Detection Interrupt Enable**
- **CHECKF: Check Done Interrupt Enable**

58.5.4 SHA Interrupt Disable Register

Name: SHA_IDR

Address: 0xF0028014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	CHECKF
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **DATRDY: Data Ready Interrupt Disable**
- **URAD: Unspecified Register Access Detection Interrupt Disable**
- **CHECKF: Check Done Interrupt Disable**

58.5.5 SHA Interrupt Mask Register

Name: SHA_IMR

Address: 0xF0028018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	CHECKF
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **DATRDY: Data Ready Interrupt Mask**
- **URAD: Unspecified Register Access Detection Interrupt Mask**
- **CHECKF: Check Done Interrupt Mask**

58.5.6 SHA Interrupt Status Register

Name: SHA_ISR

Address: 0xF002801C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
CHKST				–	–	–	CHECKF
15	14	13	12	11	10	9	8
	URAT			–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	WRDY	–	–	–	DATRDY

- **DATRDY: Data Ready (cleared by writing a 1 to bit SWRST or START in SHA_CR, or by reading SHA_IODATARx)**

0: Output data is not valid.

1: 512/1024-bit block process is completed.

DATRDY is cleared when one of the following conditions is met:

- Bit START in SHA_CR is set.
- Bit SWRST in SHA_CR is set.
- The hash result is read.

- **WRDY: Input Data Register Write Ready**

0: SHA_IDATAR0 cannot be written

1: SHA_IDATAR0 can be written

- **URAD: Unspecified Register Access Detection Status (cleared by writing a 1 to SWRST bit in SHA_CR)**

0: No unspecified register access has been detected since the last SWRST.

1: At least one unspecified register access has been detected since the last SWRST.

- **URAT: Unspecified Register Access Type (cleared by writing a 1 to SWRST bit in SHA_CR)**

Value	Description
0	SHA_IDATAR0 to SHA_IDATAR15 written during the data processing in DMA mode (URAD = 1 and URAT = 0 can occur only if DUALBUFF is cleared in SHA_MR).
1	Output Data Register read during the data processing.
2	SHA_MR written during the data processing.
3	Write-only register read access.

Only the last Unspecified Register Access Type is available through the URAT field.

- **CHECKF: Check Done Status (cleared by writing START or SWRST bits in SHA_CR or by reading SHA_IODATARx)**

0: Hash check has not been computed.

1: Hash check has been computed, status is available in the CHKST bits.

- **CHKST: Check Status (cleared by writing START or SWRST bits in SHA_CR or by reading SHA_IODATARx)**

The value 5 indicates identical hash values (expected hash = hash result). Any other value indicates different hash values.

58.5.7 SHA Message Size Register

Name: SHA_MSR

Address: 0xF0028020

Access: Read/Write

31	30	29	28	27	26	25	24
MSGSIZE							

23	22	21	20	19	18	17	16
MSGSIZE							

15	14	13	12	11	10	9	8
MSGSIZE							

7	6	5	4	3	2	1	0
MSGSIZE							

- **MSGSIZE: Message Size**

The size in bytes of the message. When MSGSIZE differs from 0, the SHA appends the corresponding value converted in bits after the padding section, as described in the FIPS180-2 specification.

To disable automatic padding, MSGSIZE field must be written to 0.

58.5.8 SHA Bytes Count Register

Name: SHA_BCR

Address: 0xF0028030

Access: Read/Write

31	30	29	28	27	26	25	24
BYTCNT							

23	22	21	20	19	18	17	16
BYTCNT							

15	14	13	12	11	10	9	8
BYTCNT							

7	6	5	4	3	2	1	0
BYTCNT							

- **BYTCNT: Remaining Byte Count Before Auto Padding**

When the hash processing starts from the beginning of a message (without preprocessed hash part), BYTCNT must be written with the same value as the MSGSIZE. If a part of the message has been already hashed and the hash does not start from the beginning, BYTCNT must be configured with the number of bytes remaining to process before padding section.

When read, provides the size in bytes of message remaining to be written before the automatic padding starts.

BYTCNT field is automatically updated each time a write occurs in the SHA_IDATARx and SHA_IODATARx.

When BYTCNT reaches 0, the MSGSIZE is converted into bit count and appended at the end of the message after the padding as described in the FIPS180-2 specification.

To disable automatic padding, MSGSIZE and BYTCNT fields must be written to 0.

58.5.9 SHA Input Data x Register

Name: SHA_IDATARx [x=0..15]

Address: 0xF0028040

Access: Write-only

31	30	29	28	27	26	25	24
IDATA							
23	22	21	20	19	18	17	16
IDATA							
15	14	13	12	11	10	9	8
IDATA							
7	6	5	4	3	2	1	0
IDATA							

- **IDATA: Input Data**

The 32-bit Input Data registers allow to load the data block used for hash processing.

These registers are write-only to prevent the input data from being read by another application.

SHA_IDATAR0 corresponds to the first word of the block, SHA_IDATAR15 to the last word of the last block in case SHA algorithm is set to SHA1, SHA224, SHA256 or SHA_IODATA15R to the last word of the block if SHA algorithm is SHA384 or SHA512 (refer to Section 58.5.10 “SHA Input/Output Data Register x”).

58.5.10 SHA Input/Output Data Register x

Name: SHA_IODATARx [x=0..15]

Address: 0xF0028080

Access: Read/Write

31	30	29	28	27	26	25	24
IODATA							
23	22	21	20	19	18	17	16
IODATA							
15	14	13	12	11	10	9	8
IODATA							
7	6	5	4	3	2	1	0
IODATA							

- **IODATA: Input/Output Data**

These registers can be used to read the resulting message digest and to write the second part of the message block when the SHA algorithm is SHA-384 or SHA-512.

SHA_IODATA0R to SHA_IODATA15R can be written or read but reading these offsets does not return the content of corresponding parts (words) of the message block. Only results from SHA calculation can be read through these registers.

When SHA processing is in progress, these registers return 0x0000.

SHA_IODATAR0 corresponds to the first word of the message digest; SHA_IODATAR4 to the last one in SHA1 mode, SHA_ODATAR6 in SHA224, SHA_IODATAR7 in SHA256, SHA_IODATAR11 in SHA384 or SHA_IODATAR15 in SHA512.

When SHA224 is selected, the content of SHA_ODATAR7 must be ignored.

When SHA384 is selected, the content of SHA_IODATAR12 to SHA_IODATAR15 must be ignored.

59. Triple Data Encryption Standard (TDES)

59.1 Description

The Triple Data Encryption Standard (TDES) is compliant with the American *FIPS (Federal Information Processing Standard) Publication 46-3* specification.

The TDES supports the four different confidentiality modes of operation (ECB, CBC, OFB and CFB), specified in the *FIPS (Federal Information Processing Standard) Publication 81* and is compatible with the Peripheral Data Controller channels for all of these modes, minimizing processor intervention for large buffer transfers.

The 64-bit long keys and input data (and initialization vector for some modes) are each stored in two corresponding 32-bit write-only registers:

Key x Word Registers TDES_KEYxWR0 and TDES_KEYxWR1

Input Data Registers TDES_IDATAR0 and TDES_IDATAR1

Initialization Vector Registers TDES_IVR0 and TDES_IVR1

As soon as the initialization vector, the input data and the key are configured, the encryption/decryption process may be started. Then the encrypted/decrypted data is ready to be read out on the two 32-bit Output Data registers (TDES_ODATARx) or through the DMA channels.

59.2 Embedded Characteristics

- Supports Single Data Encryption Standard (DES) and Triple Data Encryption Algorithm (TDEA or TDES)
- Compliant with *FIPS Publication 46-3, Data Encryption Standard (DES)*
- 64-bit Cryptographic Key for TDES
- Two-key or Three-key Algorithms for TDES
- 18-clock Cycles Encryption/Decryption Processing Time for DES
- 50-clock Cycles Encryption/Decryption Processing Time for TDES
- Supports eXtended Tiny Encryption Algorithm (XTEA)
- 128-bit key for XTEA and Programmable Round Number up to 64
- Supports the Four Standard Modes of Operation specified in the *FIPS Publication 81, DES Modes of Operation*
  - Electronic Code Book (ECB)
  - Cipher Block Chaining (CBC)
  - Cipher Feedback (CFB)
  - Output Feedback (OFB)
- 8-, 16-, 32- and 64-bit Data Sizes Possible in CFB Mode
- Last Output Data Mode Allowing Optimized Message (Data) Authentication Code (MAC) Generation
- Connection to DMA Optimizes Data Transfers for all Operating Modes

59.3 Product Dependencies

59.3.1 Power Management

The TDES may be clocked through the Power Management Controller (PMC), so the programmer must first configure the PMC to enable the TDES clock.

59.3.2 Interrupt Sources

The TDES interface has an interrupt line connected to the interrupt controller. Handling the TDES interrupt requires programming the interrupt controller before configuring the TDES.

Table 59-1. Peripheral IDs

Instance	ID
TDES	11

59.4 Functional Description

The Data Encryption Standard (DES) and the Triple Data Encryption Algorithm (TDES) specify FIPS-approved cryptographic algorithms that can be used to protect electronic data. The TDES bit in the TDES Mode Register (TDES_MR) is used to select either the single DES or the Triple DES mode.

Encryption (enciphering) converts data to an unintelligible form called ciphertext. Decrypting (deciphering) the ciphertext converts the data back into its original form, called plaintext. The CIPHER bit in TDES_MR is used to choose between encryption and decryption.

A DES is capable of using cryptographic keys of 64 bits to encrypt and decrypt data in blocks of 64 bits. This 64-bit key is defined in the Key 1 Word Registers (TDES_KEY1WRx).

A TDES key consists of three DES keys, which is also referred to as a key bundle. These three 64-bit keys are defined, respectively, in the Key 1, 2 and 3 Word Registers (TDES_KEY1WRx, TDES_KEY2WRx and TDES_KEY3WRx). In Triple DES mode (TDESMOD = 1 in TDES_MR), the KEYMOD bit in TDES_MR is used to choose between a two- and a three-key algorithm, as summarized in Table 59-2.

Table 59-2. TDES Algorithms Summary

Algorithm	Mode	Data Processing Sequence Steps		
		First	Second	Third
Three-key	Encryption	Encryption with Key 1	Decryption with Key 2	Encryption with Key 3
	Decryption	Decryption with Key 3	Encryption with Key 2	Decryption with Key 1
Two-key	Encryption	Encryption with Key 1	Decryption with Key 2	Encryption with Key 1
	Decryption	Decryption with Key 1	Encryption with Key 2	Decryption with Key 1

The input to the encryption processes of the CBC, CFB, and OFB modes includes, in addition to the plaintext, a 64-bit data block called the initialization vector (IV), which must be set in the Initialization Vector Registers (TDES_IVRx). The initialization vector is used in an initial step in the encryption of a message and in the corresponding decryption of the message.

The XTEA algorithm can be used instead of DES/TDES by configuring the TDESMOD field in TDES_MR with the appropriate value 0x2. An XTEA key consists of a 128-bit key. They are defined in the Key 1 and 2 Word Registers (TDES_KEY1WRx, TDES_KEY2WRx).

The number of rounds of XTEA is defined in TDES_XTEA_RNDR and can be programmed up to 64 (1 round = 2 Feistel network rounds).

All the start and operating modes of the TDES algorithm can be applied to the XTEA algorithm.

59.4.1 Operating Modes

The TDES supports the following operating modes:

- ECB—Electronic Code Book
- CBC—Cipher Block Chaining

- OFB—Output Feedback
- CFB—Cipher Feedback
  - CFB8 (CFB where the length of the data segment is 8 bits)
  - CFB16 (CFB where the length of the data segment is 16 bits)
  - CFB32 (CFB where the length of the data segment is 32 bits)
  - CFB64 (CFB where the length of the data segment is 64 bits)

The data preprocessing, post-processing and data chaining for each mode are automatically performed. Refer to the *FIPS Publication 81* for more complete information.

These modes are selected by setting the OPMOD field in TDES_MR.

In CFB mode, four data sizes are possible (8, 16, 32 and 64 bits), configurable by means of the CFBS field in TDES_MR (see Section 59.5.2 “TDES Mode Register”).

59.4.2 Start Modes

The SMOD field in TDES_MR selects the Encryption (or Decryption) start mode.

59.4.2.1 Manual Mode

The sequence is as follows:

1. Write the TDES_MR register with all required fields, including but not limited to SMOD and OPMOD.
2. Write the 64-bit key(s) in the different Key Word Registers (TDES_KEYxWRx), depending on whether one, two or three keys are required.
3. Write the initialization vector (or counter) in the Initialization Vector Registers (TDES_IVRx).

Note: The Initialization Vector Registers concern all modes except ECB.

4. Set the bit DATRDY (Data Ready) in the TDES Interrupt Enable register (TDES_IER), depending on whether an interrupt is required or not at the end of processing.
5. Write the data to be encrypted/decrypted in the authorized Input Data Registers (see Table 59-3).

Note: In 32-, 16- and 8-bit CFB modes, writing to TDES_IDATAR1 is not allowed and may lead to processing errors.

6. Set the START bit in the TDES Control Register (TDES_CR) to begin the encryption or decryption process.
7. When the processing completes, the bit DATRDY in the TDES Interrupt Status Register (TDES_ISR) rises. If an interrupt has been enabled by setting the bit DATRDY in TDES_IER, the interrupt line of the TDES is activated.
8. When the software reads one of the Output Data Registers (TDES_ODATARx), the DATRDY bit is automatically cleared.

Table 59-3. Authorized Input Data Registers

Operating Mode	Input Data Registers to Write
ECB	All
CBC	All
OFB	All
CFB 64-bit	All
CFB 32-bit	TDES_IDATAR0
CFB 16-bit	TDES_IDATAR0
CFB 8-bit	TDES_IDATAR0

59.4.2.2 Auto Mode

The Auto Mode is similar to the Manual Mode, except that as soon as the correct number of Input Data registers is written, processing is automatically started without any action in TDES_CR.

59.4.2.3 DMA Mode

The DMA Controller can be used in association with the TDES to perform an encryption/decryption of a buffer without any action by the software during processing.

The SMOD field of TDES_MR must be set to 0x2 and the DMA must be configured with non-incremental addresses.

The start address of any transfer descriptor must be set in TDES_IDATAR0.

The DMA chunk size configuration depends on the TDES mode of operation and is listed in Table 59-4.

When writing data to TDES with the first DMA channel, data will be fetched from a memory buffer (source data). It is recommended to configure the size of source data to “words” even for CFB modes. On the contrary, the destination data size depends on the mode of operation. When reading data from the TDES with the second DMA channel, the source data is the data read from TDES and data destination is the memory buffer. In this case, source data size depends on the TDES mode of operation and is listed in Table 59-4.

Table 59-4. DMA Data Transfer Type for the Different Operating Modes

Operating Mode	Chunk Size	Destination/Source Data Transfer Type
ECB	1	Word
CBC	1	Word
OFB	1	Word
CFB 64-bit	1	Word
CFB 32-bit	1	Word
CFB 16-bit	1	Half-word
CFB 8-bit	1	Byte

59.4.3 Last Output Data Mode

This mode is used to generate cryptographic checksums on data (MAC) using a CBC-MAC or a CFB encryption algorithm (See *FIPS Publication 81 Appendix F*).

After each end of encryption/decryption, the output data is available either on the output data registers for Manual and Auto modes or at the address specified in the receive buffer pointer for DMA mode (See Table 59-5 “Last Output Data Mode Behavior versus Start Modes”).

The Last Output Data bit (LOD) in TDES_MR can be used to retrieve only the last data of several encryption/decryption processes.

This data is only available on the Output Data Registers (TDES_ODATARx).

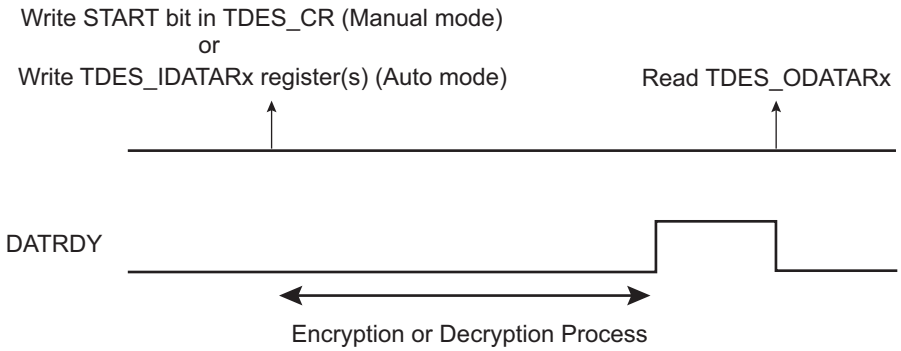
Therefore, there is no need to define a read buffer in DMA mode.

59.4.3.1 Manual and Auto Modes

TDES_MR.LOD = 0

The DATRDY flag is cleared when at least one of the Output Data Registers is read. See Figure 59-1.

Figure 59-1. Manual and Auto Modes with LOD = 0

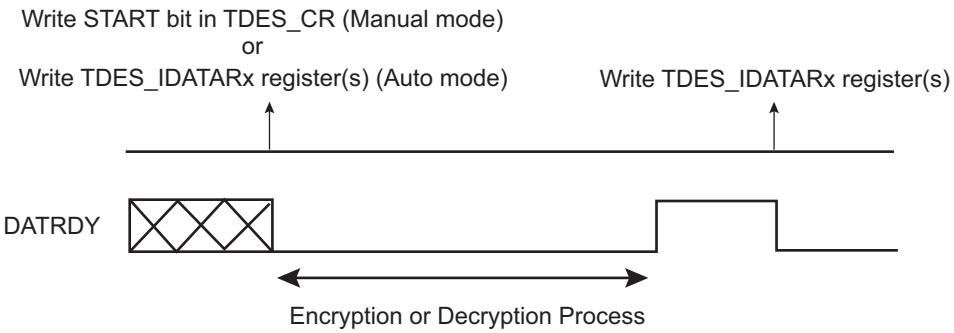


If the user does not want to read the output data registers between each encryption/decryption, the DATRDY flag will not be cleared. If the DATRDY flag is not cleared, the user will not be informed of the end of the encryptions/decryptions that follow.

TDES_MR.LOD = 1

The DATRDY flag is cleared when at least one Input Data Register is written, before the start of a new transfer. See Figure 59-2. No further Output Data Register reads are necessary between consecutive encryptions/decryptions.

Figure 59-2. Manual and Auto Modes with LOD = 1



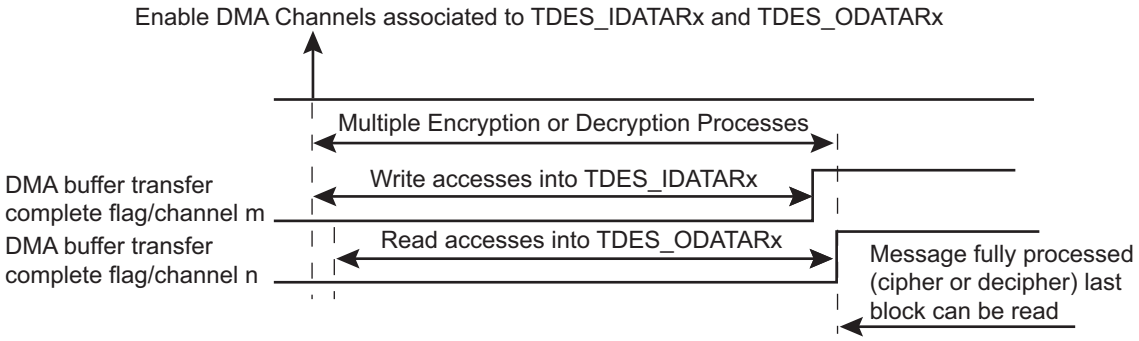
59.4.3.2 DMA Mode

TDES_MR.LOD = 0

This mode may be used for all TDES operating modes except CBC-MAC where LOD = 1 mode is recommended.

The end of the encryption/decryption is indicated by the end of DMA transfer associated to TDES_ODATARx (see Figure 59-3). Two DMA channels are required: one for writing message blocks to TDES_IDATARx and one to obtain the result from TDES_ODATARx.

Figure 59-3. DMA Transfer with LOD = 0



TDES_MR.LOD = 1

This mode is optimized to process the TDES CBC-MAC operating mode.

The user must first wait for the DMA buffer transfer complete flag, then for the flag DATRDY to rise to ensure that the encryption/decryption is completed (see Figure 59-4).

In this case, no receive buffers are required.

The output data is only available on TDES_ODATARx.

Figure 59-4. DMA Transfer with LOD = 1

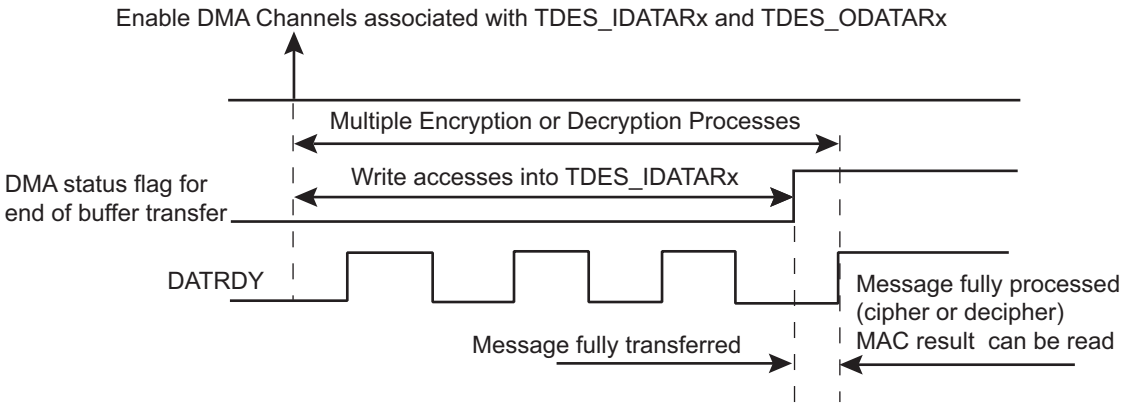


Table 59-5 summarizes the different cases.

Table 59-5. Last Output Data Mode Behavior versus Start Modes

Sequence	Manual and Auto Modes		DMA Transfer	
	LOD = 0	LOD = 1	LOD = 0	LOD = 1
DATRDY Flag Clearing Condition (1)	At least one Output Data Register must be read	At least one Input Data Register must be written	Not used	Managed by the DMA
End of Encryption/Decryption	DATRDY	DATRDY	2 DMA Buffer transfer complete flags (channel m and channel n)	DMA buffer transfer complete flag, then TDES DATRDY flag
Encrypted/Decrypted Data Result Location	In the Output Data Registers	In the Output Data Registers	Not available	In the Output Data Registers

Note: 1. Depending on the mode, there are other ways of clearing the DATRDY flag. See: Section 59.5.6 “TDES Interrupt Status Register”.

Warning: In DMA mode, reading to the Output Data registers before the last data transfer may lead to unpredictable results.

59.4.4 Security Features

59.4.4.1 Unspecified Register Access Detection

When an unspecified register access occurs, the URAD bit in TDES_ISR is set. Its source is then reported in the Unspecified Register Access Type field (URAT). Only the last unspecified register access is available through the URAT field.

Several kinds of unspecified register accesses can occur:

- Input Data Register written during the data processing in DMA mode
- Output Data Register read during the data processing
- Mode Register written during the data processing
- Write-only register read access

The URAD bit and the URAT field can only be reset by the SWRST bit in TDES_CR.

59.5 Triple Data Encryption Standard (TDES) User Interface

Table 59-6. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TDES_CR	Write-only	–
0x04	Mode Register	TDES_MR	Read/Write	0x2
0x08–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	TDES_IER	Write-only	–
0x14	Interrupt Disable Register	TDES_IDR	Write-only	–
0x18	Interrupt Mask Register	TDES_IMR	Read-only	0x0
0x1C	Interrupt Status Register	TDES_ISR	Read-only	0x0000001E
0x20	Key 1 Word Register 0	TDES_KEY1WR0	Write-only	–
0x24	Key 1 Word Register 1	TDES_KEY1WR1	Write-only	–
0x28	Key 2 Word Register 0	TDES_KEY2WR0	Write-only	–
0x2C	Key 2 Word Register 1	TDES_KEY2WR1	Write-only	–
0x30	Key 3 Word Register 0	TDES_KEY3WR0	Write-only	–
0x34	Key 3 Word Register 1	TDES_KEY3WR1	Write-only	–
0x38–0x3C	Reserved	–	–	–
0x40	Input Data Register 0	TDES_IDATAR0	Write-only	–
0x44	Input Data Register 1	TDES_IDATAR1	Write-only	–
0x48–0x4C	Reserved	–	–	–
0x50	Output Data Register 0	TDES_ODATAR0	Read-only	0x0
0x54	Output Data Register 1	TDES_ODATAR1	Read-only	0x0
0x58–0x5C	Reserved	–	–	–
0x60	Initialization Vector Register 0	TDES_IVR0	Write-only	–
0x64	Initialization Vector Register 1	TDES_IVR1	Write-only	–
0x68–0x6C	Reserved	–	–	–
0x70	XTEA Rounds Register	TDES_XTEA_RNDR	Read/Write	0x0
0x74–0xFC	Reserved	–	–	–

59.5.1 TDES Control Register

Name: TDES_CR

Address: 0xFC044000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	SWRST
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	START

- **START: Start Processing**

0: No effect

1: Starts Manual encryption/decryption process.

- **SWRST: Software Reset**

0: No effect

1: Resets the TDES. A software triggered hardware reset of the TDES interface is performed.

59.5.2 TDES Mode Register

Name: TDES_MR

Address: 0xFC044004

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	CFBS	
15	14	13	12	11	10	9	8
LOD	–	OPMOD		–	–	SMOD	
7	6	5	4	3	2	1	0
–	–	–	KEYMOD	–	TDESMOD		CIPHER

- **CIPHER: Processing Mode**

0 (DECRYPT): Decrypts data.

1 (ENCRYPT): Encrypts data.

- **TDESMOD: ALGORITHM Mode**

Value	Name	Description
0x0	SINGLE_DES	Single DES processing using TDES_KEY1WRx registers
0x1	TRIPLE_DES	Triple DES processing using TDES_KEY1WRx, TDES_KEY2WRx and TDES_KEY3WRx registers
0x2	XTEA	XTEA processing using TDES_KEY1WRx, TDES_KEY2WRx

Values which are not listed in the table must be considered as “reserved”.

- **KEYMOD: Key Mode**

0: Three-key algorithm is selected.

1: Two-key algorithm is selected. There is no need to write TDES_KEY3WRx registers.

- **SMOD: Start Mode**

Value	Name	Description
0x0	MANUAL_START	Manual Mode
0x1	AUTO_START	Auto Mode
0x2	IDATAR0_START	TDES_IDATAR0 accesses only Auto Mode

Values which are not listed in the table must be considered as “reserved”.

If a DMA transfer is used, 0x2 must be configured. Refer to Section 59.4.3.2 “DMA Mode” for more details.

- **OPMOD: Operating Mode**

Value	Name	Description
0x0	ECB	Electronic Code Book mode
0x1	CBC	Cipher Block Chaining mode
0x2	OFB	Output Feedback mode
0x3	CFB	Cipher Feedback mode

For CBC-MAC operating mode, please set OPMOD to CBC and LOD to 1.

- **LOD: Last Output Data Mode**

0: No effect.

After each end of encryption/decryption, the output data is available either on the output data registers (Manual and Auto modes) .

In Manual and Auto modes, the DATRDY flag is cleared when at least one of the Output Data registers is read.

1: The DATRDY flag is cleared when at least one of the Input Data Registers is written.

No more Output Data Register reads are necessary between consecutive encryptions/decryptions (see Section 59.4.3 “Last Output Data Mode”).

Warning: In DMA mode, reading to the Output Data registers before the last data encryption/decryption process may lead to unpredictable result.

- **CFBS: Cipher Feedback Data Size**

Value	Name	Description
0x0	SIZE_64BIT	64-bit
0x1	SIZE_32BIT	32-bit
0x2	SIZE_16BIT	16-bit
0x3	SIZE_8BIT	8-bit

59.5.3 TDES Interrupt Enable Register

Name: TDES_IER

Address: 0xFC044010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

- **URAD: Unspecified Register Access Detection Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

59.5.4 TDES Interrupt Disable Register

Name: TDES_IDR

Address: 0xFC044014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

- **URAD: Unspecified Register Access Detection Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

59.5.5 TDES Interrupt Mask Register

Name: TDES_IMR

Address: 0xFC044018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

- **URAD: Unspecified Register Access Detection Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

59.5.6 TDES Interrupt Status Register

Name: TDES_ISR

Address: 0xFC04401C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
		URAT		–	–	–	URAD
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready (cleared by setting bit START or bit SWRST in TDES_CR or by reading TDES_ODATARx)**

0: Output data is not valid.

1: Encryption or decryption process is completed.

Note: If TDES_MR.LOD = 1: In Manual and Auto modes, the DATRDY flag can also be cleared by writing at least one TDES_IDATARx.

- **URAD: Unspecified Register Access Detection Status (cleared by setting bit TDES_CR.SWRST)**

0: No unspecified register access has been detected since the last write of bit TDES_CR.SWRST.

1: At least one unspecified register access has been detected since the last write of bit TDES_CR.SWRST.

- **URAT: Unspecified Register Access (cleared by setting bit TDES_CR.SWRST)**

Value	Name	Description
0x0	IDR_WR_PROCESSING	Input Data Register written during data processing when SMOD = 0x2 mode.
0x1	ODR_RD_PROCESSING	Output Data Register read during data processing.
0x2	MR_WR_PROCESSING	Mode Register written during data processing.
0x3	WOR_RD_ACCESS	Write-only register read access.

Only the last Unspecified Register Access Type is available through the URAT field.

59.5.7 TDES Key 1 Word Register x

Name: TDES_KEY1WRx

Address: 0xFC044020

Access: Write-only

31	30	29	28	27	26	25	24
KEY1W							

23	22	21	20	19	18	17	16
KEY1W							

15	14	13	12	11	10	9	8
KEY1W							

7	6	5	4	3	2	1	0
KEY1W							

- **KEY1W: Key 1 Word**

The two 32-bit Key 1 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY1W0 refers to the first word of the key and KEY1W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

In XTEA mode, the key is defined on 128 bits. These registers contain the 64 LSB bits of the encryption/decryption key.

59.5.8 TDES Key 2 Word Register x

Name: TDES_KEY2WRx

Address: 0xFC044028

Access: Write-only

31	30	29	28	27	26	25	24
KEY2W							

23	22	21	20	19	18	17	16
KEY2W							

15	14	13	12	11	10	9	8
KEY2W							

7	6	5	4	3	2	1	0
KEY2W							

- **KEY2W: Key 2 Word**

The two 32-bit Key 2 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY2W0 refers to the first word of the key and KEY2W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

Note: KEY2WRx registers are not used in DES mode.

In XTEA mode, the key is defined on 128 bits. These registers contain the 64 MSB bits of the encryption/decryption key.

59.5.9 TDES Key 3 Word Register x

Name: TDES_KEY3WRx

Address: 0xFC044030

Access: Write-only

31	30	29	28	27	26	25	24
KEY3W							

23	22	21	20	19	18	17	16
KEY3W							

15	14	13	12	11	10	9	8
KEY3W							

7	6	5	4	3	2	1	0
KEY3W							

- **KEY3W: Key 3 Word**

The two 32-bit Key 3 Word registers are used to set the 64-bit cryptographic key used for encryption/decryption.

KEY3W0 refers to the first word of the key and KEY3W1 to the last one.

These registers are write-only to prevent the key from being read by another application.

Note: KEY3WRx registers are not used in DES mode, TDES with two-key algorithm selected and XTEA mode.

59.5.10 TDES Input Data Register x

Name: TDES_IDATARx

Address: 0xFC044040

Access: Write-only

31	30	29	28	27	26	25	24
IDATA							

23	22	21	20	19	18	17	16
IDATA							

15	14	13	12	11	10	9	8
IDATA							

7	6	5	4	3	2	1	0
IDATA							

- **IDATA: Input Data**

The two 32-bit Input Data registers are used to set the 64-bit data block used for encryption/decryption.

IDATA0 refers to the first word of the data to be encrypted/decrypted, and IDATA1 to the last one.

These registers are write-only to prevent the input data from being read by another application.

59.5.11 TDES Output Data Register x

Name: TDES_ODATARx

Address: 0xFC044050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							

23	22	21	20	19	18	17	16
ODATA							

15	14	13	12	11	10	9	8
ODATA							

7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The two 32-bit Output Data registers contain the 64-bit data block which has been encrypted/decrypted.

ODATA1 refers to the first word, ODATA2 to the last one.

59.5.12 TDES Initialization Vector Register x

Name: TDES_IVRx

Address: 0xFC044060

Access: Write-only

31	30	29	28	27	26	25	24
IV							
23	22	21	20	19	18	17	16
IV							
15	14	13	12	11	10	9	8
IV							
7	6	5	4	3	2	1	0
IV							

- **IV: Initialization Vector**

The two 32-bit Initialization Vector registers are used to set the 64-bit initialization vector data block, which is used by some modes of operation as an additional initial input.

IV1 refers to the first word of the Initialization Vector, IV2 to the last one.

These registers are write-only to prevent the Initialization Vector from being read by another application.

Note: These registers are not used for the ECB mode and must not be written.

59.5.13 TDES XTEA Rounds Register

Name: TDES_XTEA_RNDR

Address: 0xFC044070

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	XTEA_RNDS					

- **XTEA_RNDS: Number of Rounds**

This 6-bit field is used to define the number of complete rounds (1 complete round = 2 Feistel rounds) processed in XTEA algorithm.

The value of XTEA_RNDS has no effect if the TDESMOD field in TDES_MR is set to 0x0 or 0x1.

Note: 0x00 corresponds to 1 complete round, 0x01 corresponds to 2 complete rounds, etc.

60. True Random Number Generator (TRNG)

60.1 Description

The True Random Number Generator (TRNG) passes the American *NIST Special Publication 800-22* (*A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications*) and the *Diehard Suite of Tests*.

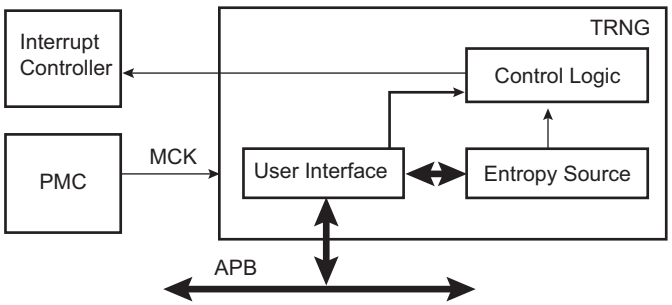
The TRNG may be used as an entropy source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3.

60.2 Embedded Characteristics

- Passes *NIST Special Publication 800-22* Test Suite
- Passes *Diehard Suite of Tests*
- May be used as Entropy Source for seeding an NIST approved DRNG (Deterministic RNG) as required by FIPS PUB 140-2 and 140-3
- Provides a 32-bit Random Number Every 84 Clock Cycles

60.3 Block Diagram

Figure 60-1. TRNG Block Diagram



60.4 Product Dependencies

60.4.1 Power Management

The TRNG interface may be clocked through the Power Management Controller (PMC), thus the programmer must first configure the PMC to enable the TRNG user interface clock. The user interface clock is independent from any clock that may be used in the entropy source logic circuitry. The source of entropy can be enabled before enabling the user interface clock.

60.4.2 Interrupt Sources

The TRNG interface has an interrupt line connected to the Interrupt Controller. In order to handle interrupts, the Interrupt Controller must be programmed before configuring the TRNG.

Table 60-1. Peripheral IDs

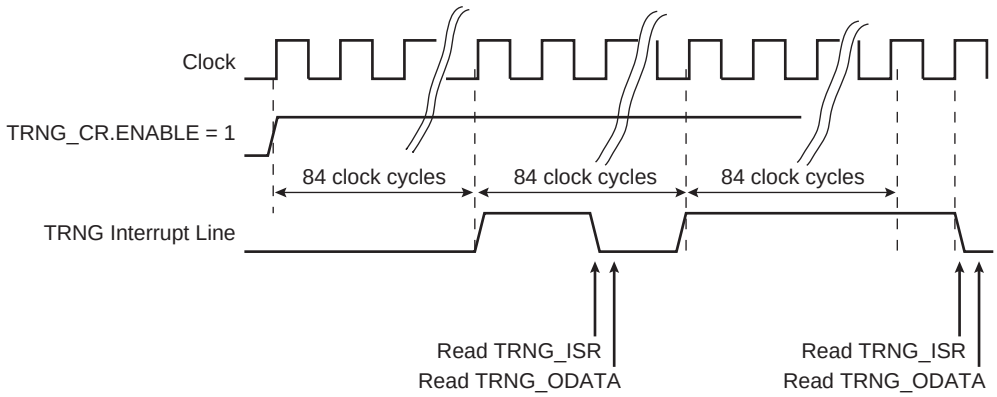
Instance	ID
TRNG	47

60.5 Functional Description

As soon as the TRNG is enabled in the control register (TRNG_CR), the generator provides one 32-bit value every 84 clock cycles. The TRNG interrupt line can be enabled in the TRNG_IER (respectively disabled in the TRNG_IDR). This interrupt is set when a new random value is available and is cleared when the status register (TRNG_ISR) is read. The flag DATRDY of the (TRNG_ISR) is set when the random data is ready to be read out on the 32-bit output data register (TRNG_ODATA).

The normal mode of operation checks that the status register flag equals 1 before reading the output data register when a 32-bit random value is required by the software application.

Figure 60-2. TRNG Data Generation Sequence



60.6 True Random Number Generator (TRNG) User Interface

Table 60-2. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	TRNG_CR	Write-only	–
0x04–0x0C	Reserved	–	–	–
0x10	Interrupt Enable Register	TRNG_IER	Write-only	–
0x14	Interrupt Disable Register	TRNG_IDR	Write-only	–
0x18	Interrupt Mask Register	TRNG_IMR	Read-only	0x0000_0000
0x1C	Interrupt Status Register	TRNG_ISR	Read-only	0x0000_0000
0x20–0x4C	Reserved	–	–	–
0x50	Output Data Register	TRNG_ODATA	Read-only	0x0000_0000
0x54–0xFC	Reserved	–	–	–

60.6.1 TRNG Control Register

Name: TRNG_CR

Address: 0xFC01C000

Access: Write-only

31	30	29	28	27	26	25	24
KEY							
23	22	21	20	19	18	17	16
KEY							
15	14	13	12	11	10	9	8
KEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	ENABLE

- **ENABLE: Enables the TRNG to Provide Random Values**

0: Disables the TRNG.

1: Enables the TRNG if 0x524E47 (“RNG” in ASCII) is written in KEY field at the same time.

- **KEY: Security Key**

Value	Name	Description
0x524E47	PASSWD	Writing any other value in this field aborts the write operation.

60.6.2 TRNG Interrupt Enable Register

Name: TRNG_IER

Address: 0xFC01C010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Enable**

0: No effect.

1: Enables the corresponding interrupt.

60.6.3 TRNG Interrupt Disable Register

Name: TRNG_IDR

Address: 0xFC01C014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Disable**

0: No effect.

1: Disables the corresponding interrupt.

60.6.4 TRNG Interrupt Mask Register

Name: TRNG_IMR

Address: 0xFC01C018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready Interrupt Mask**

0: The corresponding interrupt is not enabled.

1: The corresponding interrupt is enabled.

60.6.5 TRNG Interrupt Status Register

Name: TRNG_ISR

Address: 0xFC01C01C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
		–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	DATRDY

- **DATRDY: Data Ready**

0: Output data is not valid or TRNG is disabled.

1: New random value is completed.

DATRDY is cleared when this register is read.

60.6.6 TRNG Output Data Register

Name: TRNG_ODATA

Address: 0xFC01C050

Access: Read-only

31	30	29	28	27	26	25	24
ODATA							

23	22	21	20	19	18	17	16
ODATA							

15	14	13	12	11	10	9	8
ODATA							

7	6	5	4	3	2	1	0
ODATA							

- **ODATA: Output Data**

The 32-bit Output Data register contains the 32-bit random data.

61. Analog-to-Digital Converter (ADC)

61.1 Description

The ADC is based on a 12-bit Analog-to-Digital Converter (ADC) managed by an ADC Controller providing enhanced resolution up to 14 bits. Refer to Figure 61-1 “Analog-to-Digital Converter Block Diagram”. It also integrates a 12-to-1 analog multiplexer, making possible the analog-to-digital conversions of 12 analog lines. The conversions extend from 0V to the voltage carried on pin ADVREF.

Conversion results are reported in a common register for all channels, as well as in a channel-dedicated register.

The 13-bit and 14-bit resolution modes are obtained by averaging multiple samples to decrease quantization noise. For the 13-bit mode, 4 samples are used, which gives a real sample rate of 1/4 of the actual sample frequency. For the 14-bit mode, 16 samples are used, giving a real sample rate of 1/16 of the actual sample frequency. This arrangement allows conversion speed to be traded off against for better accuracy.

The software trigger, external trigger on rising edge of the ADTRG pin or internal triggers from Timer Counter output(s) are configurable.

The comparison circuitry allows automatic detection of values below a threshold, higher than a threshold, in a given range or outside the range, thresholds and ranges being fully configurable.

The ADC Controller internal fault output is directly connected to the PWM fault input. This input can be asserted by means of comparison circuitry to immediately put the PWM output in a safe state (pure combinational path).

The ADC also integrates a Sleep mode and a conversion sequencer and connects with a DMA channel. These features reduce both power consumption and processor intervention.

This ADC has a selectable single-ended or fully differential input.

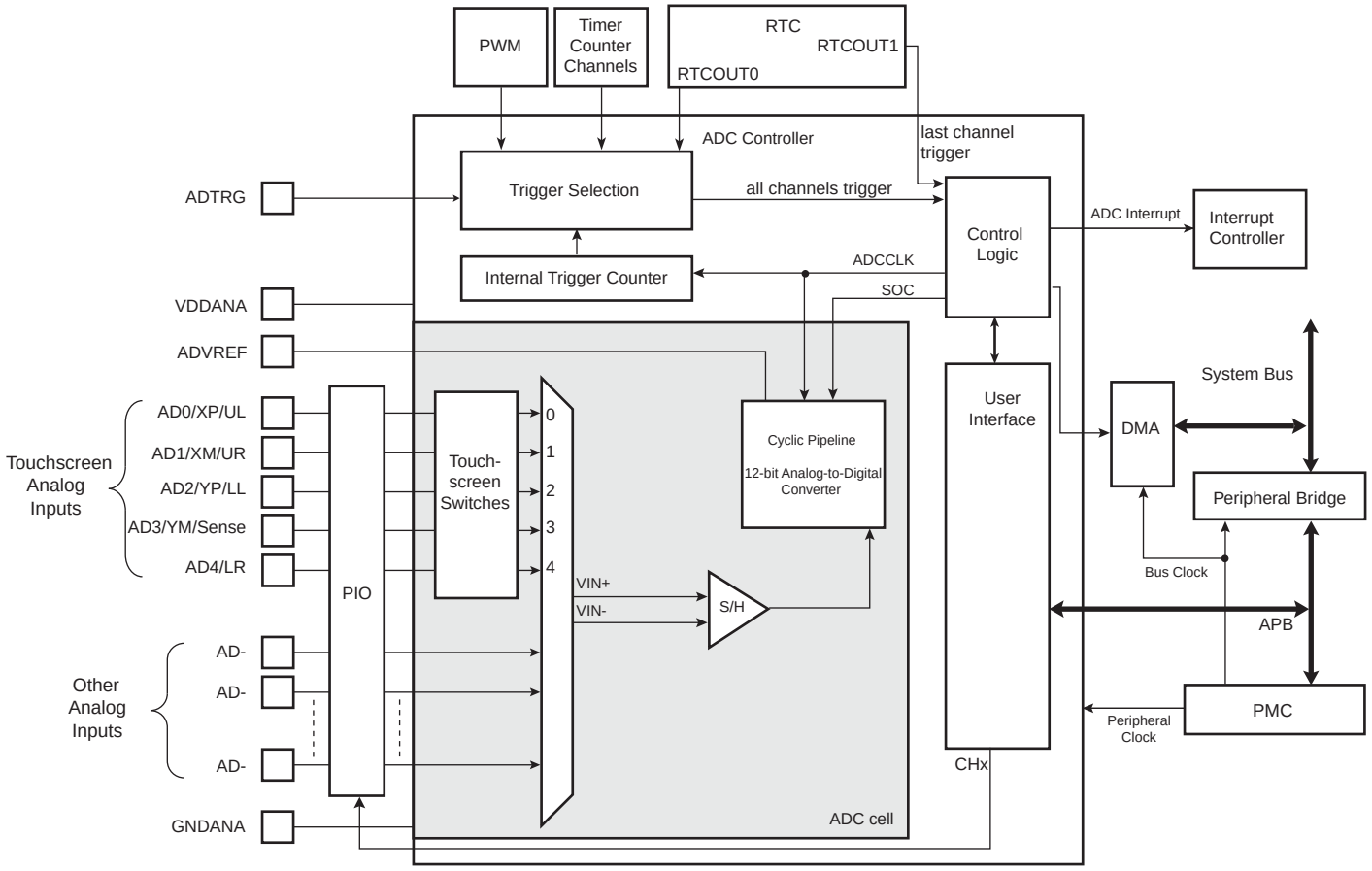
This ADC Controller includes a Resistive Touchscreen Controller. It supports 4-wire and 5-wire technologies.

61.2 Embedded Characteristics

- 12-bit Resolution with Enhanced Mode up to 14 bits
- 1 Msps Conversion Rate
- Digital Averaging Function providing Enhanced Resolution Mode up to 14 bits
- Wide Range of Power Supply Operation
- Selectable Single-Ended or Differential Input Voltage
- Digital correction of offset and gain errors
- Resistive 4-wire and 5-wire Touchscreen Controller
  - Position and Pressure Measurement for 4-wire Screens
  - Position Measurement for 5-wire Screens
  - Average of Up to 8 Measures for Noise Filtering
- Programmable Pen Detection Sensitivity
- Integrated Multiplexer Offering Up to 12 Independent Analog Inputs
- Individual Enable and Disable of Each Channel
- Hardware or Software Trigger from:
  - External Trigger Pin
  - Timer Counter Outputs (Corresponding TIOA Trigger)
  - ADC Internal Trigger Counter
  - Trigger on Pen Contact Detection
  - PWM Event Line
- Drive of PWM Fault Input
- DMA Support
- Two Sleep Modes (Automatic Wakeup on Trigger)
  - Lowest Power Consumption (Voltage Reference OFF Between Conversions)
  - Fast Wakeup Time Response on Trigger Event (Voltage Reference ON Between Conversions)
- Channel Sequence Customization
- Automatic Window Comparison of Converted Values
- Asynchronous Partial Wakeup (SleepWalking) on external trigger
- Register Write Protection

61.3 Block Diagram

Figure 61-1. Analog-to-Digital Converter Block Diagram



61.4 Signal Description

Table 61-1. ADC Pin Description

Pin Name	Description
VDDANA	Analog power supply
ADVREF	Reference voltage
AD0–AD11	Analog input channels
ADTRG	External trigger

61.5 Product Dependencies

61.5.1 Power Management

The ADC Controller is not continuously clocked. The programmer must first enable the ADC Controller peripheral clock in the Power Management Controller (PMC) before using the ADC Controller. However, if the application does not require ADC operations, the ADC Controller clock can be stopped when not needed and restarted when necessary. Configuring the ADC Controller does not require the ADC Controller clock to be enabled.

61.5.2 Interrupt Sources

The ADC interrupt line is connected on one of the internal sources of the Interrupt Controller. Using the ADC interrupt requires the interrupt controller to be programmed first.

Table 61-2. Peripheral IDs

Instance	ID
ADC	40

61.5.3 I/O Lines

The digital input ADTRG is multiplexed with digital functions on the I/O line and the selection of ADTRG is made using the PIO controller.

The analog inputs ADC_ADx are multiplexed with digital functions on the I/O lines. ADC_ADx inputs are selected as inputs of the ADCC when writing a one in the corresponding CHx bit of ADC_CHER and the digital functions are not selected.

Table 61-3. I/O Lines

Instance	Signal	I/O Line	Peripheral
ADC	ADTRG	PD31	A
ADC	AD0	PD19	X1
ADC	AD1	PD20	X1
ADC	AD2	PD21	X1
ADC	AD3	PD22	X1
ADC	AD4	PD23	X1
ADC	AD5	PD24	X1
ADC	AD6	PD25	X1
ADC	AD7	PD26	X1
ADC	AD8	PD27	X1
ADC	AD9	PD28	X1
ADC	AD10	PD29	X1
ADC	AD11	PD30	X1

61.5.4 Hardware Triggers

The ADC can use internal signals to start conversions. Refer to the ADC_MR.TRGSEL field description for exact wiring of internal triggers.

61.5.5 Fault Output

The ADC Controller has the FAULT output connected to the FAULT input of PWM. Refer to Section 61.6.18 “Fault Output” and to section “Pulse Width Modulation Controller (PWM)”.

61.6 Functional Description

61.6.1 Analog-to-Digital Conversion

Once the programmed startup time (ADC_MR.STARTUP) has elapsed, ADC conversions are sequenced by three operating times:

- Tracking time—the time for the ADC to charge its input sampling capacitor to the input voltage. When several channels are converted consecutively, the inherent tracking time is 6 ADC clock cycles. However, the tracking time can be increased using the TRACKTIM field in the Mode Register (ADC_MR).
- ADC inherent conversion time—the time for the ADC to convert the sampled analog voltage. This time is constant and is defined from start of conversion to end of conversion.
- Channel conversion period—the effective time between the end of the current channel conversion and the end of the next channel conversion.

Figure 61-2. Sequence of Consecutive ADC Conversions with TRACKTIM = 0

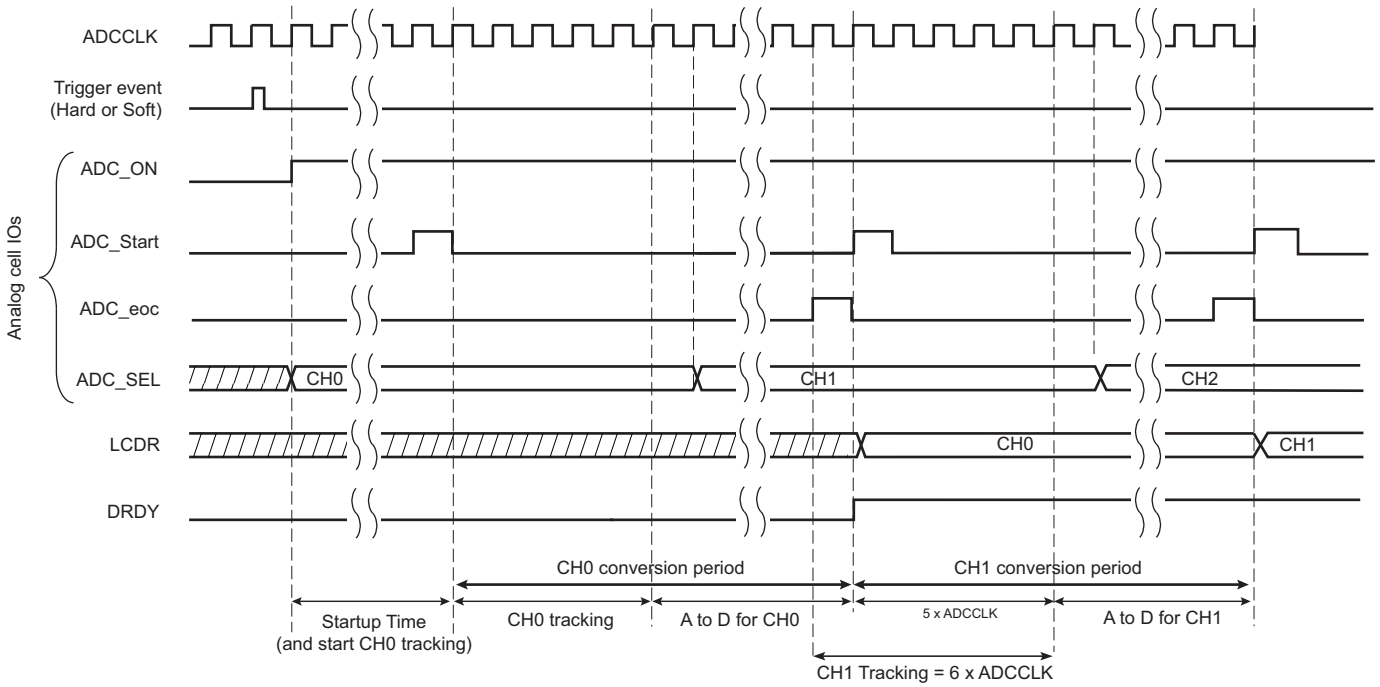
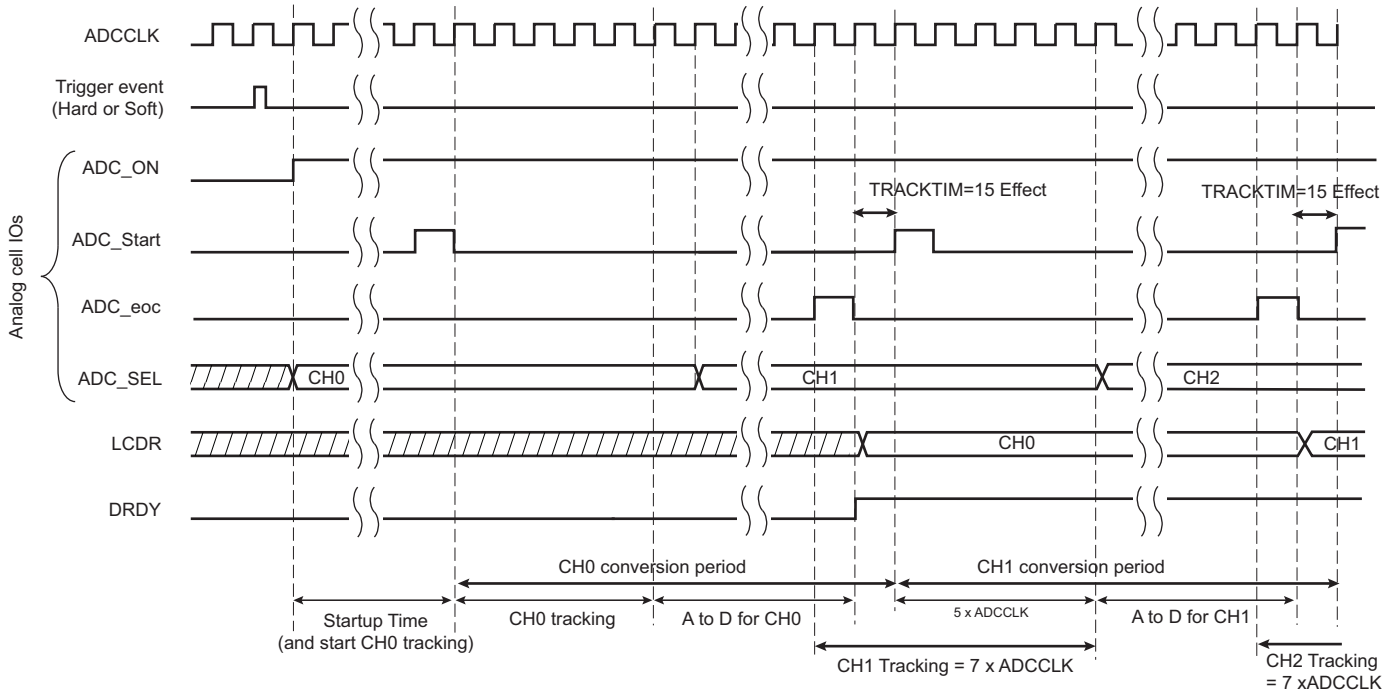


Figure 61-3. Sequence of Consecutive ADC Conversions with TRACKTIM = 15



61.6.2 ADC Clock

The ADC uses the ADC clock (ADCCLK) to perform conversions. The ADC clock frequency is selected in the PRESCAL field of ADC_MR.

To generate the ADC clock, the prescaler has two clock sources: the peripheral clock and the GCLK clock. This clock source is selected using the SRCCLK bit in the Extended Mode Register (ADC_EMR).

If GCLK is selected as a source clock, the ADC clock frequency is independent of the processor/bus clock. At reset, the peripheral clock is selected.

If the SRCCLK bit in ADC_EMR is cleared, the prescaler clock (presc_clk) is driven by peripheral_clock. If the SRCCLK bit in ADC_EMR is set, the prescaler clock is driven by GCLK. The ADC clock frequency is between fpresc_clk/2, if PRESCAL is 0, and fpresc_clk/512, if PRESCAL is set to 255 (0xFF).

PRESCAL must be programmed to provide the ADC clock frequency parameter given in section “Electrical Characteristics”.

61.6.3 ADC Reference Voltage

The voltage reference input of the ADC is the ADVREF pin. Refer to the “Electrical Characteristics” section for further details.

61.6.4 Conversion Resolution

The ADC has a native resolution of 12 bits.

The ADC controller provides enhanced resolution up to 14 bits by means of digital averaging.

If ADTRG is asynchronous to the ADC peripheral clock, the internal resynchronization introduces a jitter of 1 peripheral clock. This jitter may reduce the resolution of the converted signal.

The same applies when using the independent clock (ADC_MR.SRCCLK = 1), if the provided clock is asynchronous to ADC peripheral clock.

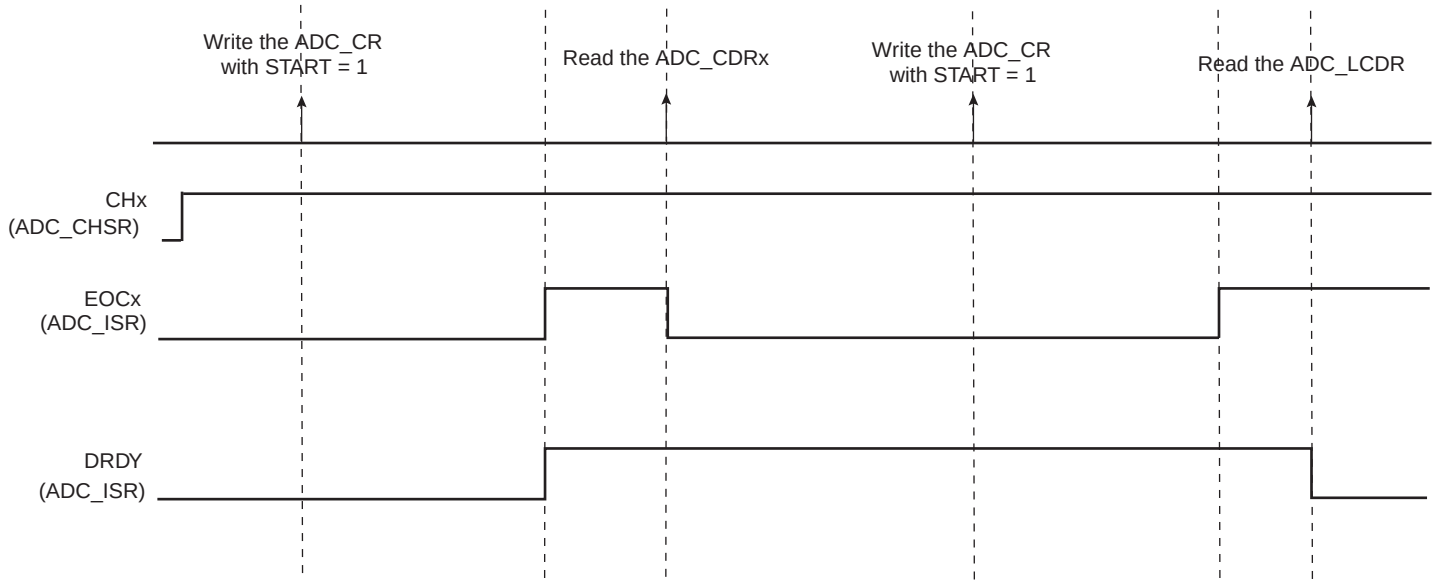
61.6.5 Conversion Results

When a conversion is completed, the resulting digital value is stored in the Channel Data register (ADC_CDRx) of the current channel and in the ADC Last Converted Data register (ADC_LCDR). By setting the TAG option in the Extended Mode Register (ADC_EMR), ADC_LCDR presents the channel number associated with the last converted data in the CHNB field.

When a conversion is completed, the channel EOC bit and the DRDY bit in the Interrupt Status register (ADC_ISR) are set. In the case of a connected DMA channel, DRDY rising triggers a data request. In any case, either EOC and DRDY can trigger an interrupt.

Reading one of the ADC_CDRx clears the corresponding EOC bit. Reading ADC_LCDR clears the DRDY bit.

Figure 61-4. EOCx and DRDY Flag Behavior

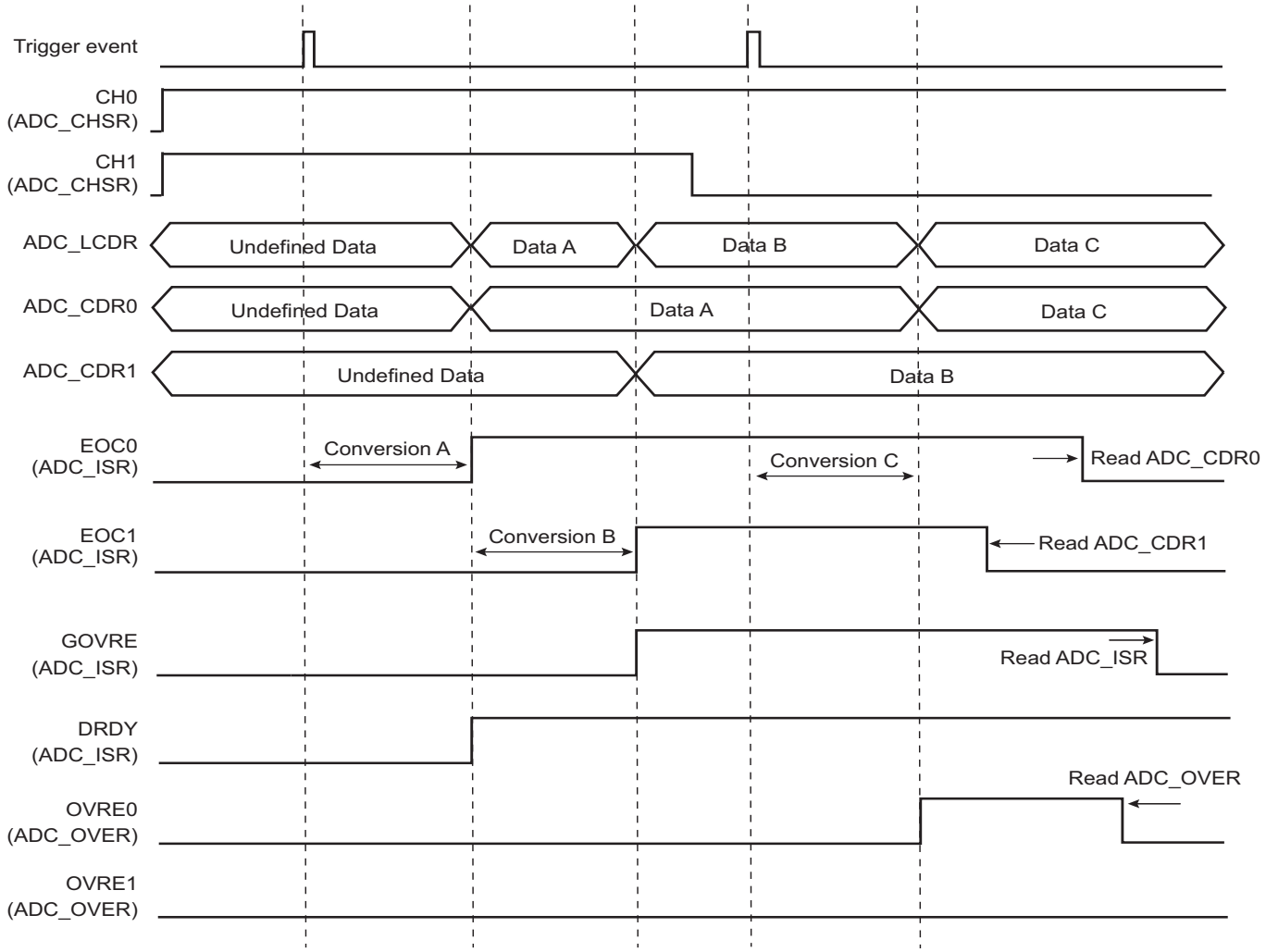


If ADC_CDR is not read before further incoming data is converted, the corresponding OVREx flag is set in the Overrun Status register (ADC_OVER).

If new data is converted when DRDY is high, the GOVRE bit is set in ADC_ISR.

The OVREx flag is automatically cleared when ADC_OVER is read, and the GOVRE flag is automatically cleared when ADC_ISR is read.

Figure 61-5. EOCx, OVREx and GOVREx Flag Behavior



Warning: If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable.

61.6.6 Conversion Results Format

The conversion results can be signed (2's complement) or unsigned depending on the value of the SIGNMODE field ("SIGNMODE: Sign Mode" in ADC_EMR).

If conversion results are signed and resolution is less than 16 bits, the sign is extended up to the bit 15 (e.g., 0xF43 for 12-bit resolution is read as 0xFF43, and 0x467 is read as 0x0467).

61.6.7 Conversion Triggers

Conversions of the active analog channels are started with a software or hardware trigger. The software trigger is provided by writing the Control register (ADC_CR) with the START bit at 1.

The list of external/internal events is provided in Section 61.7.2 "ADC Mode Register". The hardware trigger is selected using the TRGSEL field in ADC_MR. The selected hardware trigger is enabled if TRGMOD = 1, 2 or 3 in the ADC Trigger Register (ADC_ TRGR).

The ADC also provides a dual trigger mode (ADC_LCTMR.DUALTRIG = 1) in which the higher index channel can be sampled at a rhythm different from the other channels. The trigger of the last channel is generated by the RTC. Refer to section Section 61.6.12 “Last Channel Specific Measurement Trigger”.

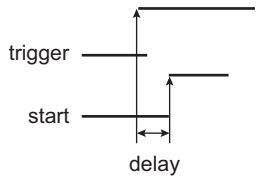
The TRGMOD field in the ADC Trigger register (ADC_TRGR) selects the hardware trigger from the following:

- any edge, either rising or falling or both, detected on the external trigger pin ADTRG
- the Pen Detect, depending on how the PENDET bit is set in the ADC Touchscreen Mode register (ADC_TSMR)
- a continuous trigger, meaning the ADC Controller restarts the next sequence as soon as it finishes the current one
- a periodic trigger, which is defined by programming the TRGPER field in ADC_TRGR

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence according to configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, and ADC_TSMR.

If a hardware trigger is selected, the start of a conversion is triggered after a delay starting at each rising edge of the selected signal. Due to asynchronous handling, the delay may vary in a range of two peripheral clock periods to one ADC clock period. This delay introduces sampling jitter in the A/D conversion process and may therefore degrade the conversion performance (e.g., SNR, THD).

Figure 61-6. Hardware Trigger Delay



If one of the TIOA outputs is selected, the corresponding Timer Counter channel must be programmed in Waveform mode.

Only one start command is necessary to initiate a conversion sequence on all the channels. The ADC hardware logic automatically performs the conversions on the active channels, then waits for a new request. The Channel Enable (ADC_CHER) and Channel Disable (ADC_CHDR) registers enable the analog channels to be enabled or disabled independently.

If the ADC is used with a DMA, only the transfers of converted data from enabled channels are performed and the resulting data buffers should be interpreted accordingly.

61.6.8 Sleep Mode and Conversion Sequencer

The ADC Sleep mode maximizes power saving by automatically deactivating the ADC when it is not being used for conversions. Sleep mode is selected by setting the SLEEP bit in ADC_MR.

Sleep mode is managed by a conversion sequencer, which automatically processes the conversions of all channels at lowest power consumption.

This mode can be used when the minimum period of time between two successive trigger events is greater than the startup period of the ADC. See section “Electrical Characteristics”.

When a start conversion request occurs, the ADC is automatically activated. As the analog cell requires a startup time, the logic waits during this time and starts the conversion on the enabled channels. When all conversions are complete, the ADC is deactivated until the next trigger. Events triggered during the sequence are ignored.

The conversion sequencer allows automatic processing with minimum processor intervention and optimized power consumption. Conversion sequences can be performed periodically using the internal timer (ADC_TRGR) or the

PWM event line. The periodic acquisition of several samples can be processed automatically without any intervention of the processor via the DMA.

The sequence can be customized by programming the Sequence Channel Registers ADC_SEQR1 and ADC_SEQR2 and setting the USEQ bit of the Mode Register (ADC_MR). The user can choose a specific order of channels and can program up to 12 conversions by sequence. The user is free to create a personal sequence by writing channel numbers in ADC_SEQR1 and ADC_SEQR2. Not only can channel numbers be written in any sequence, channel numbers can be repeated several times. When the bit USEQ in ADC_MR is set, the fields USCHx in ADC_SEQR1 and ADC_SEQR2 are used to define the sequence. Only enabled USCHx fields will be part of the sequence. Each USCHx field has a corresponding enable, CHx-1, in ADC_CHER.

If all ADC channels (i.e., 12) are used on an application board, there is no restriction of usage of the user sequence. However, if some ADC channels are not enabled for conversion but rather used as pure digital inputs, the respective indexes of these channels cannot be used in the user sequence fields (see ADC_SEQRx). For example, if channel 4 is disabled (ADC_CSR[4] = 0), ADC_SEQRx fields USCH1 up to USCH12 must not contain the value 4. Thus the length of the user sequence may be limited by this behavior.

As an example, if only four channels over 12 (CH0 up to CH3) are selected for ADC conversions, the user sequence length cannot exceed four channels. Each trigger event may launch up to four successive conversions of any combination of channels 0 up to 3 but no more (i.e., in this case the sequence CH0, CH0, CH1, CH1, CH1 is impossible).

A sequence that repeats the same channel several times requires more enabled channels than channels actually used for conversion. For example, the sequence CH0, CH0, CH1, CH1 requires four enabled channels (four free channels on application boards) whereas only CH0, CH1 are really converted.

Note: The reference voltage pins always remain connected in Normal mode as in Sleep mode.

61.6.9 Comparison Window

The ADC Controller features automatic comparison functions. It compares converted values to a low threshold, a high threshold or both, depending on the value of the CMPMODE field in ADC_EMR. The comparison can be done on all channels or only on the channel specified in the CMPSEL field of ADC_EMR. To compare all channels, the CMPALL bit of ADC_EMR must be set.

If set, the CMPTYPE bit of ADC_EMR can be used to discard all conversion results that do not match the comparison conditions. Once a conversion result matches the comparison conditions, all the subsequent conversion results are stored in ADC_LCDR (even if these results do not meet the comparison conditions). Setting the CMPRST bit in ADC_CR immediately stops the conversion result storage until the next comparison match.

If the CMPTYPE bit in ADC_EMR is cleared, all conversions are stored in ADC_LCDR. Only the conversions that match the comparison conditions trigger the COMPE flag in ADC_ISR.

Moreover, a filtering option can be set by writing the number of consecutive comparison matches needed to raise the flag. This number can be written and read in the CMPFILTER field of ADC_EMR. The filtering option is dedicated to reinforcing the detection of an analog signal overpassing a predefined threshold. The filter is cleared as soon as ADC_ISR is read, so this filtering function must be used with peripheral DMA controller and works only when using Interrupt mode (no polling).

The flag can be read on the COMPE bit of the Interrupt Status register (ADC_ISR) and can trigger an interrupt.

The high threshold and the low threshold can be read/write in the Compare Window register (ADC_CWR).

Depending on the sign of the conversion, chosen with the SIGNMODE field in the ADC Extended Mode Register, the high threshold and low threshold values must be signed or unsigned to maintain consistency during the comparison. If the conversion is signed, both thresholds must also be signed; if the conversion is unsigned, both thresholds must be unsigned. If comparison occurs on all channels, the SIGNMODE field must be set to ALL_UNSIGNED or ALL_SIGNED and the thresholds must be set accordingly.

61.6.10 Differential and Single-ended Input Modes

61.6.10.1 Input-output Transfer Functions

The ADC can be configured to operate in the following input voltage modes:

- Single-ended—ADC_COR.DIFFx = 0. This is the default mode after a reset.
- Differential—ADC_COR.DIFFx = 1 (see Figure 61-7). In Differential mode, the ADC requires differential input signals having a VDD/2 common mode voltage (refer to the “Electrical Characteristics” section).

The following equations give the unsigned ADC input-output transfer function in each mode[1]. With signed conversions (refer to field ADC_EMR.SIGNMODE), subtract 2047 from the ADC_LCDR.DATA value given below.

Single-ended mode:

$$\text{ADC_LCDR.LDATA} = \frac{ADx - GNDANA}{ADVREF - GNDANA} \times 2^{12}$$

Differential mode:

$$\text{ADC_LCDR.LDATA} = \left(1 + \frac{ADx - \text{ADx+1}}{ADVREF - GNDANA}\right) \times 2^{11}$$

Note: 1. Equations assume ADC_EMR.OSR = 1

If the ANACH bit is set in ADC_MR, the ADC can manage both differential channels and single-ended channels. If the ANACH bit is cleared, the parameters defined in ADC_COR are applied to all channels.

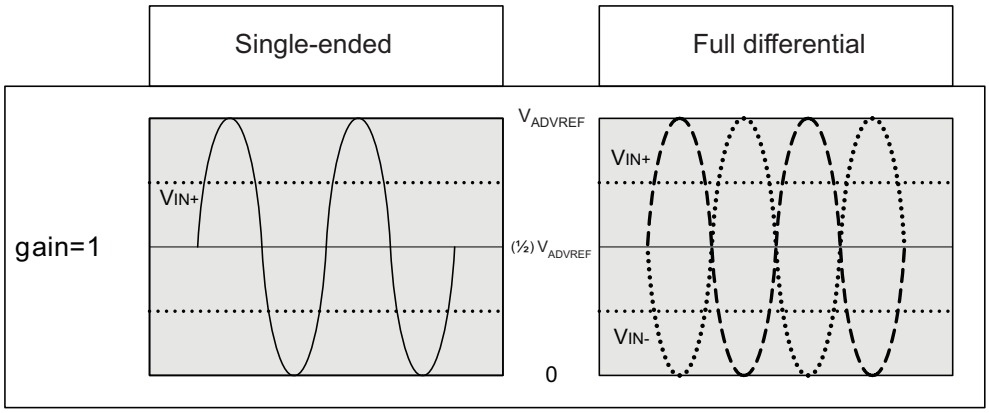
Table 61-4 gives the internal positive and negative ADC inputs assignment with respect to the programmed mode (ADC_COR.DIFFx).

For example, if Differential mode is required on channel 0, input pins AD0 and AD1 are used. In this case, only channel 0 must be enabled by writing a 1 to ADC_CHER.CH0.

Table 61-4. Input Pins and Channel Numbers

Input Pin	Channel Number	
	Single-ended Mode	Differential Mode
AD0	CH0	CH0
AD1	CH1	
AD2	CH2	CH2
AD3	CH3	
AD4	CH4	CH4
AD5	CH5	
AD6	CH6	CH6
AD7	CH7	
AD8	CH8	CH8
AD9	CH9	
AD10	CH10	CH10
AD11	CH11	

Figure 61-7. Analog Full Scale Ranges in Single-Ended/Differential Applications



61.6.11 ADC Timings

The ADC startup time is programmed through the STARTUP field in ADC_MR. See section “Electrical Characteristics”.

The ADC controller provides an inherent tracking time of six ADC clock cycles.

A minimal tracking time is necessary for the ADC to guarantee the best converted final value between two conversions. The tracking time can be adjusted to accommodate a range of source impedances. If more than six ADC clock cycles are required, the tracking time can be increased using the TRACKTIM field in ADC_MR.

Warning: No input buffer amplifier to isolate the source is included in the ADC. See section “Electrical Characteristics”.

61.6.12 Last Channel Specific Measurement Trigger

The last channel (higher index available) embeds a specific mode allowing a measurement trigger period which differs from other active channels. This allows efficient management of the conversions especially if the channel is driven by a device with a variation of a different frequency from other converted channels (for example, but not limited to, temperature sensor).

The last channel can be sampled in different ways through the ADC controller. The different methods of sampling depend on the configuration field TRGMOD in ADC_TRGR and bit CH11 in ADC_CHSR.

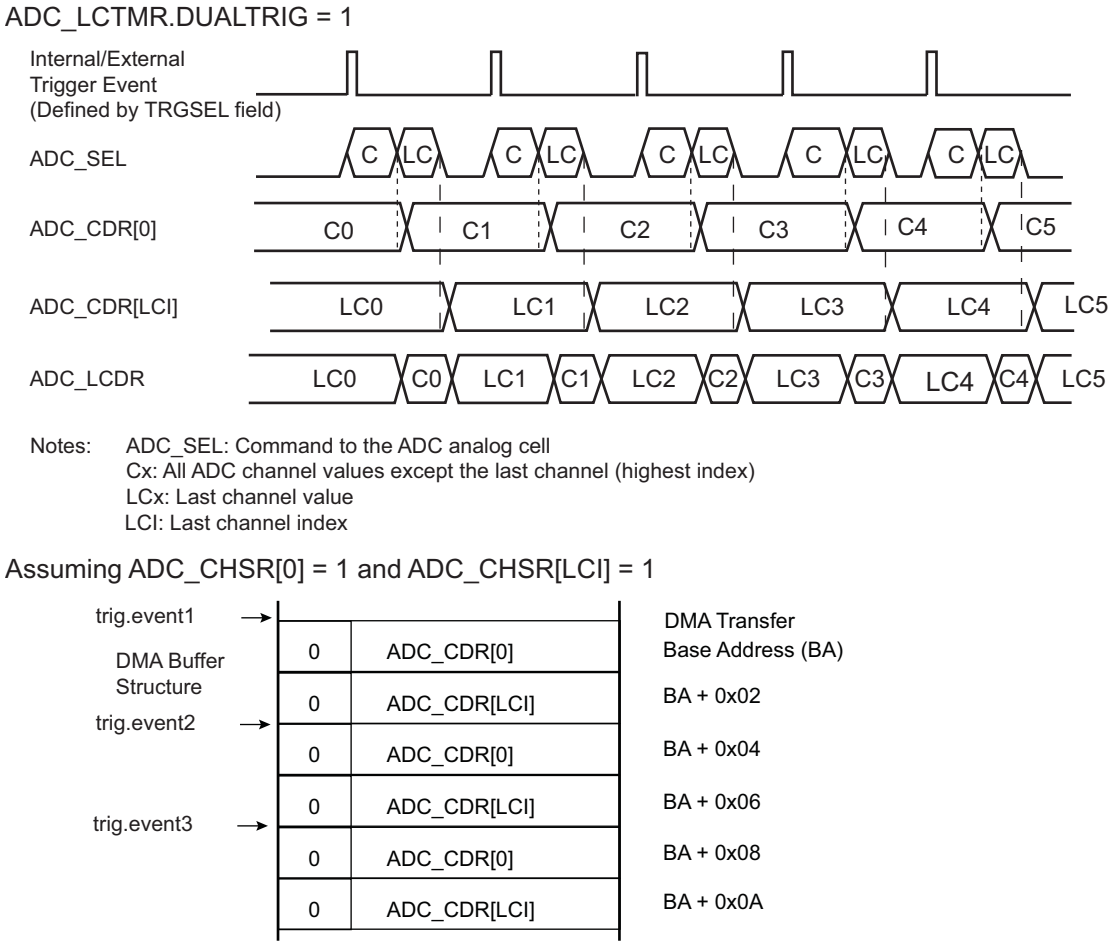
The last channel conversion can be triggered like the other channels by enabling CH11 of ADC_CHER.

The manual start can only be performed if field TRGMOD = 0. When the START bit in ADC_CR is set, the last channel conversion is scheduled together with the other enabled channels (if any). The result of the conversion is placed in ADC_CDR11 register and the associated flag EOC11 is set in ADC_ISR.

If the last channel is enabled in ADC_CHSR, DUALTRIG is cleared and field TRGMOD = 1, 2, 3, 5, the last channel is periodically converted together with the other enabled channels and the result is placed in the ADC_LCDR and ADC_CDR11 registers. Thus the last channel conversion result is part of the DMA Controller buffer (see Figure 61-8).

When the conversion result matches the conditions defined in the ADC_LCTMR and ADC_LCCWR, the LCCHG flag is set in ADC_ISR.

Figure 61-8. Same Trigger for All Channels (ADC_CHSR[LCI] = 1 and ADC_TRGR.TRGMOD = 1, 2, 3, 5)



If the last channel is driven by a device with a slower variation compared to other channels (temperature sensor for example), the channel can be enabled/disabled at any time. However, this may not be optimal for downstream processing.

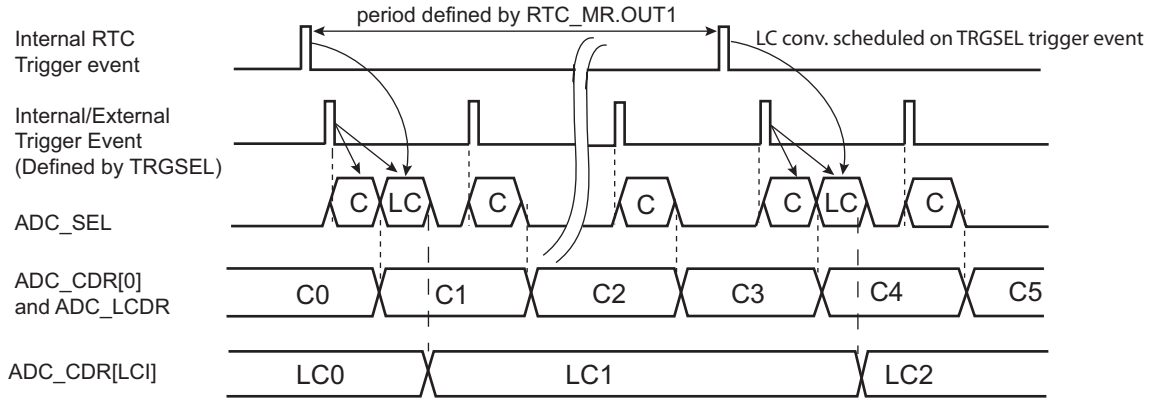
The ADC controller allows a different way of triggering the measurement when DUALTRIG is set in the Last Channel Trigger Mode Register (ADC_LCTMR) but CH11 is not set in ADC_CHSR.

Under these conditions, the last channel conversion is triggered with a period defined by the OUT1 field in the RTC_MR (Real-time Clock Mode Register) while other channels are still active. OUT1 configures an internal trigger generated by the RTC, totally independent of the internal/external triggers. The RTC event will be processed on the next internal/external trigger event as described in Figure 61-9. The internal/external trigger for other channels is selected through the TRGSEL field of ADC_MR.

When DUALTRIG = 1, the result of each conversion of channel 11 is only uploaded in the ADC_CDR11 register and not in ADC_LCDR (see Figure 61-9). Therefore there is no change in the structure of the peripheral DMA controller buffer due to the conversion of the last channel: only the enabled channels are kept in the buffer. The end of conversion of the last channel is reported by the EOC11 flag in ADC_ISR.

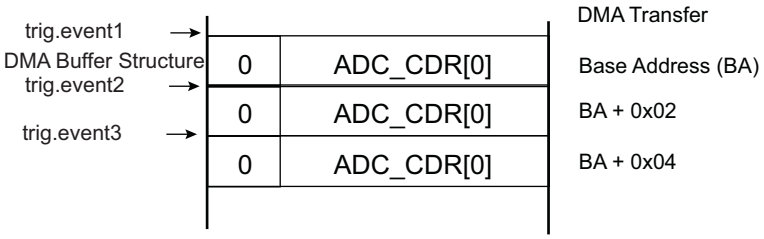
Figure 61-9. Independent Trigger Measurement for Last Channel (ADC_CHSR[LCI] = 0 and ADC_TRGR.TRGMOD = 1, 2, 3, 5)

ADC_LCTMR.DUALTRIG = 1



Notes: ADC_SEL: Command to the ADC analog cell
Cx: All ADC channel values except the last channel (highest index)
LCx: Last channel value
LCI: Last channel index

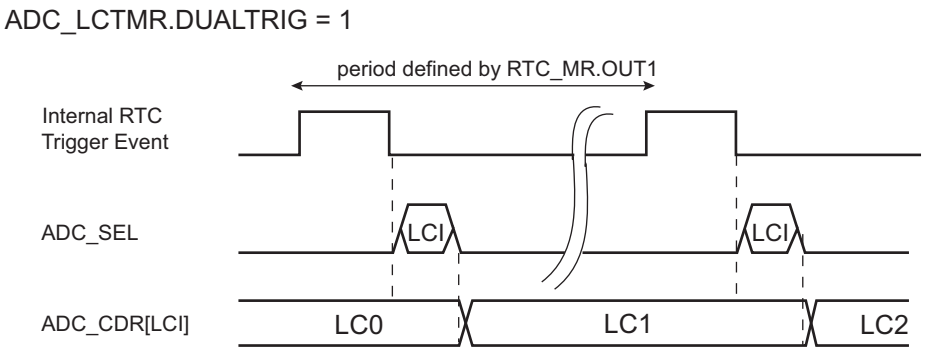
Assuming ADC_CHSR[0] = 1



If DUALTRIG = 1 and field ADC_TRGR.TRGMOD = 0 and none of the channels are enabled in ADC_CHSR (ADC_CHSR = 0), then only channel 11 is converted at a rate defined by the trigger event signal that can be configured in RTC_MR.OUT1 (see Figure 61-10).

This mode of operation, when combined with the Sleep mode operation of the ADC Controller, provides a low-power mode for last channel measure. This assumes there is no other ADC conversion to schedule at a high sampling rate or no other channel to convert.

Figure 61-10. Only Last Channel Measurement Triggered at Low Speed (ADC_CHSR[LCI] = 0 and ADC_TRGR.TRGMOD = 0)



Notes: ADC_SEL: Command to the ADC analog cell
LCx: Last channel value
LCI: Last channel index

61.6.13 Enhanced Resolution Mode and Digital Averaging Function

61.6.13.1 Enhanced Resolution Mode

The Enhanced Resolution mode is enabled if the OSR field is configured to 1 or 2 in ADC_EMR. The enhancement is based on a digital averaging function.

There is no averaging on the last index channel if the measure is triggered by an RTC event.

In this mode, the ADC Controller will trade off conversion speed against accuracy by averaging multiple samples, thus providing a digital low-pass filter function.

The selected oversampling ratio applies to all enabled channels when triggered by an RTC event.

$$\text{ADC_LCDR.LDATA} = \frac{1}{M} \times \sum_{k=0}^{k=N-1} ADC(k)$$

where N and M are given in the table below.

Table 61-5. Digital Averaging Function Configuration versus OSR Values

ADC_EMR.OSR Value	ADC_LCDR.LDATA Length	N Value	M Value	Full Scale Value	Maximum Value
0	12 bits	1	1	4095	4095
1	13 bits	4	2	8191	8190
2	14 bits	16	4	16383	16381

The average result is valid in ADC_CDRx (x corresponds to the index of the channel) only if the EOCn flag is set in ADC_ISR and if the OVREn flag is cleared in ADC_OVER. The average result for all channels is valid in ADC_LCDR only if DRDY is set and GOVRE is cleared in ADC_ISR.

Note that ADC_CDRs are not buffered. Therefore, when an averaging sequence is ongoing, the value in these registers changes after each averaging sample. However, overrun flags in ADC_OVER rise as soon as the first

sample of an averaging sequence is received. Thus the previous averaged value is not read, even if the new averaged value is not ready.

Consequently, when an overrun flag rises in ADC_OVER, it means that the previous unread data is lost but it does not mean that this data has been overwritten by the new averaged value as the averaging sequence concerning this channel can still be ongoing.

When an oversampling is performed, the maximum value that can be read on ADC_CDRx or ADC_LCDR is not the full-scale value, even if the maximum voltage is supplied on the analog input. Refer to Table 61-5 “Digital Averaging Function Configuration versus OSR Values”.

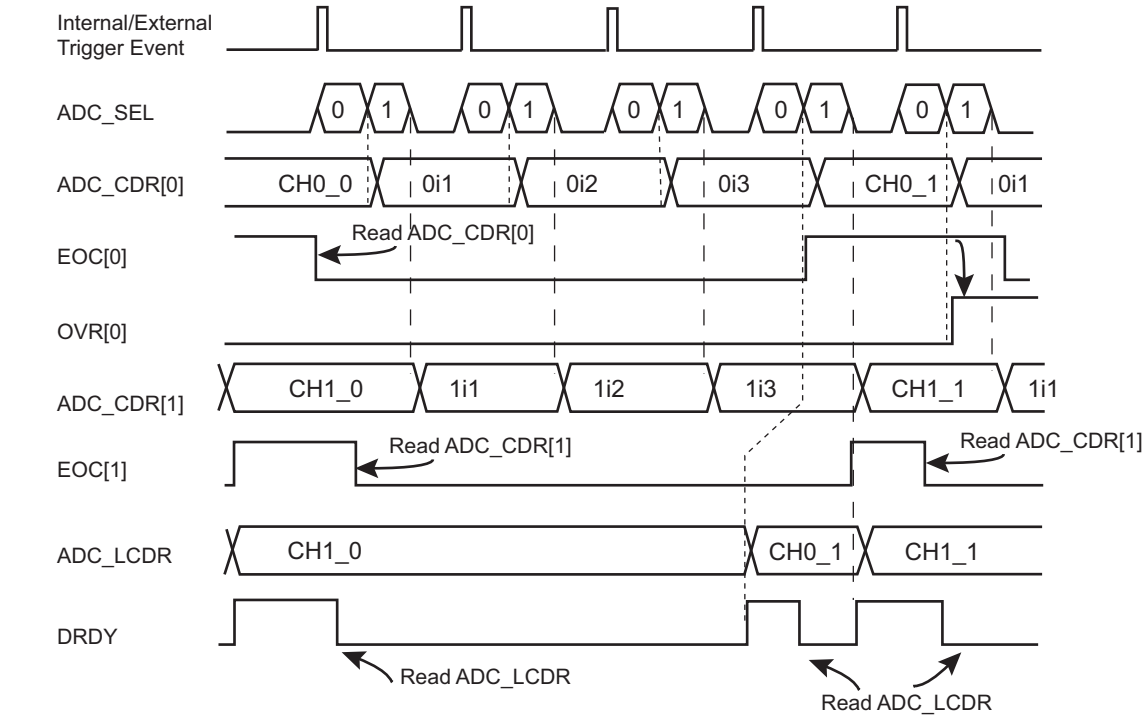
61.6.13.2 Averaging Function versus Trigger Events

The samples can be defined in different ways for the averaging function depending on the configuration of the ASTE bit in ADC_EMR and the USEQ bit in ADC_MR.

When USEQ = 0, there are two possible ways to generate the averaging through the trigger event. If ASTE = 0 in ADC_EMR, every trigger event generates one sample for each enabled channel as described in Figure 61-11. Therefore four trigger events are requested to obtain the result of averaging if OSR = 1.

Figure 61-11. Digital Averaging Function Waveforms Over Multiple Trigger Events

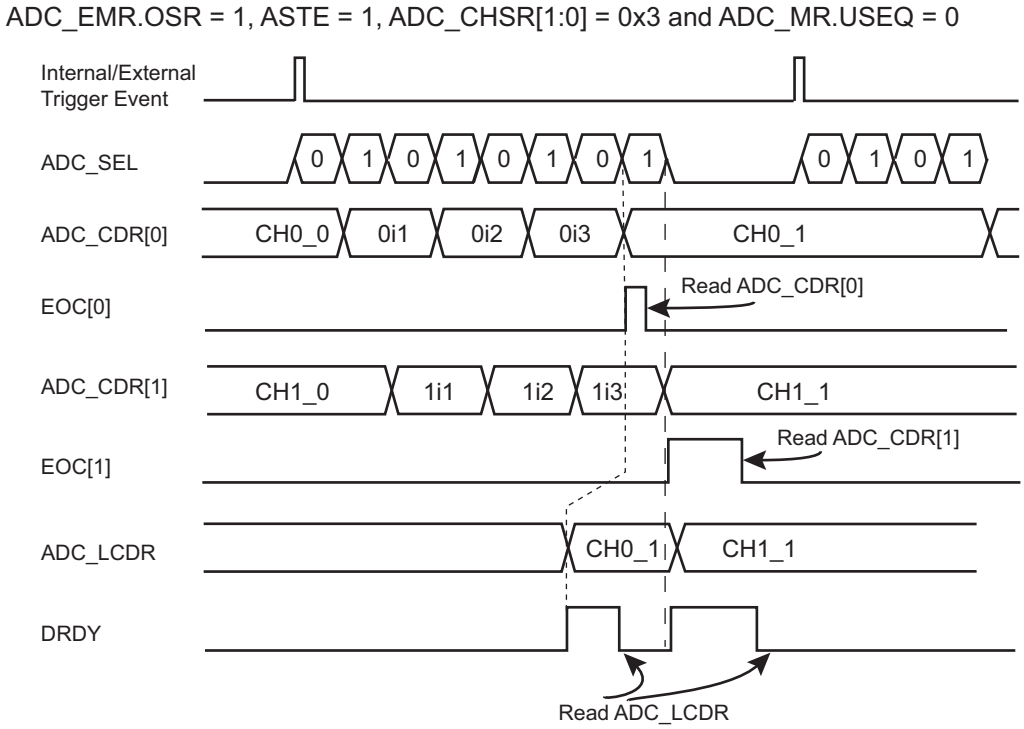
ADC_EMR.OSR = 1, ASTE = 0, ADC_CHSR[1:0] = 0x3 and ADC_MR.USEQ = 0



Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

If ASTE = 1 in ADC_EMR and USEQ = 0 in ADC_MR, the sequence to be converted, defined in ADC_CHSR, is automatically repeated n times (where n corresponds to the oversampling ratio defined in the OSR field in ADC_EMR). As a result, only one trigger is required to obtain the result of the averaging function as described in Figure 61-12.

Figure 61-12. Digital Averaging Function Waveforms on a Single Trigger Event



Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

When USEQ = 1, the user can define the channel sequence to be converted by configuring ADC_SEQRx and ADC_CHER so that channels are not interleaved during the averaging period. Under these conditions, a sample is defined for each end of conversion as described in Figure 61-13.

When USEQ = 1 and ASTE = 1, OSR can be only configured to 1. Up to three channels can be converted in this mode. The averaging result will be placed in the corresponding ADC_CDRx and in ADC_LCDR for each trigger event. The ADC real sample rate remains the maximum ADC sample rate divided by 4.

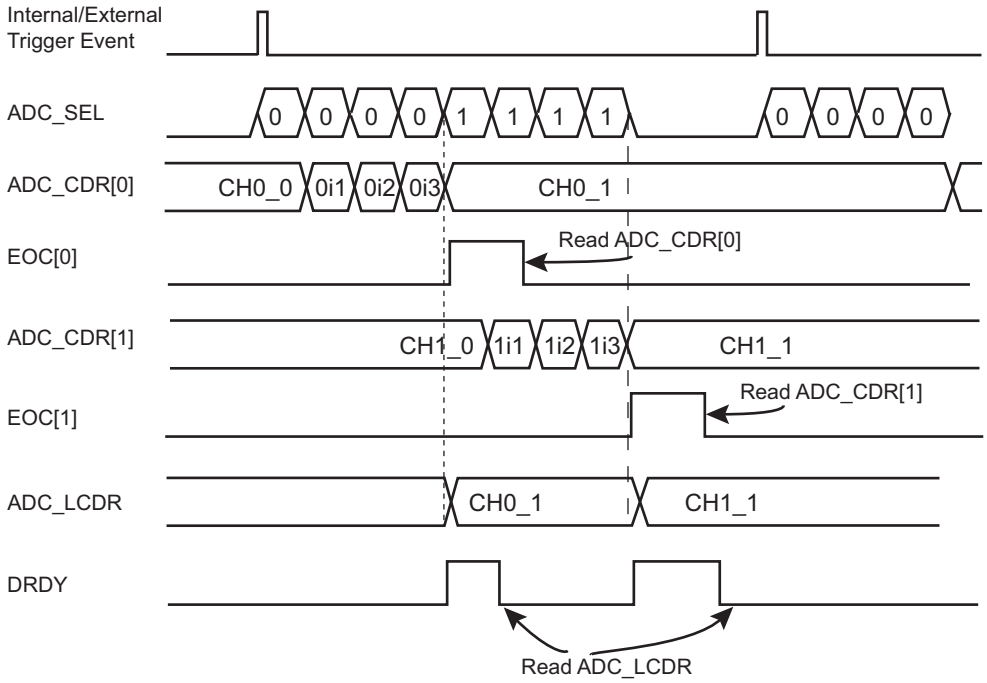
It is important that the user sequence follows a specific pattern. The user sequence must be programmed in such a way that it generates a stream of conversion, where a same channel is successively converted.

Table 61-6. Example Sequence Configurations (USEQ = 1, ASTE = 1, OSR = 1)

	Number of Channels Non-interleaved Averaging - Register Value		
Register	**1 (e.g., CH0)**	**2 (e.g., CH0, CH1)**	**3 (e.g., CH0, CH1, CH2)**
ADC_CHSR	0x0000_000F	0x0000_00FF	0x0000_0FFF
ADC_SEQR1	0x0000_0000	0x1111_0000	0x1111_0000
ADC_SEQR2	0x0000_0000	0x0000_0000	0x0000_2222

Figure 61-13. Digital Averaging Function Waveforms on a Single Trigger Event, Non-interleaved

ADC_EMR.OSR = 1, ASTE = 1, ADC_CHSR[7:0] = 0xFF and ADC_MR.USEQ = 1
ADC_SEQR1 = 0x1111_0000



Note: ADC_SEL: Command to the ADC analog cell
0i1, 0i2, 0i3, 1i1, 1i2, 1i3 are intermediate results and CH0_0, CH0_1, CH1_0 and CH1_1 are final results of average function.

61.6.14 Automatic Error Correction

The ADC features automatic error correction of conversion results. Offset and gain error corrections are available. The correction can be enabled for each channel and correction values (offset and gain) are the same for all channels.

To enable error correction, the corresponding ECORRx bit must be set in the Channel Error Correction Register (ADC_CECR). The offset and gain values used to compensate the results are the same for all correction-enabled channels and programmed in the Correction Values Register (ADC_CVR).

The error correction for channels used with the touchscreen is available in the ADC Touchscreen Correction Values Register (ADC_TSCVR).

The ADCMODE field in ADC_EMR is used to configure a running mode of the ADC Normal mode, Offset Error mode, or Gain Error mode (see Section 61.7.16 “ADC Extended Mode Register”).

After a reset, the running mode of the ADC is Normal mode. Offset Error mode and Gain Error mode are used to determine values of offset compensation and gain compensation, respectively, to apply to conversion results. Table 61-7 provides formulas to obtain the compensation values, with:

- OFFSETCORR—the Offset Correction value. OFFSETCORR is a signed value.
- GAINCORR—the Gain Correction value
- GCi—the intermediate Gain Compensation value
- Gs—the value 13
- ConvValue—the value converted by the ADC (as returned in ADC_LCDR or ADC_CDR)

- Resolution—the resolution used to process the conversion (either RESOLUTION, RESOLUTION+1 or RESOLUTION+2).

Table 61-7. ADC Running Modes

ADC_EMR.ADCMODE	Mode	Description
0	Normal	Normal mode of operation to perform conversions
1	Offset Error	For unsigned conversions: OFFSETCORR = ConvValue – $2^{(\text{Resolution} - 1)}$
		For signed conversions: OFFSETCORR = ConvValue
2	Gain Error	GCi = ConvValue
3		$GAINCORR = \frac{3584}{GCi - ConvValue} \times 2^{(Gs)}$

The final conversion result after error correction is obtained using the following formula:

$$\text{Corrected Data} = (\text{Converted Data} + OFFSETCORR) \times \frac{GAINCORR}{2^{(Gs)}}$$

61.6.15 Touchscreen

61.6.15.1 Touchscreen Mode

The TSMODE parameter of the ADC Touchscreen Mode register (ADC_TSMR) is used to enable/disable the touchscreen functionality, to select the type of screen (4-wire or 5-wire) and, in the case of a 4-wire screen and to activate (or not) the pressure measurement.

In 4-wire mode, channel 0, 1, 2 and 3 must not be used for classic ADC conversions. Likewise, in 5-wire mode, channel 0, 1, 2, 3, and 4 must not be used for classic ADC conversions.

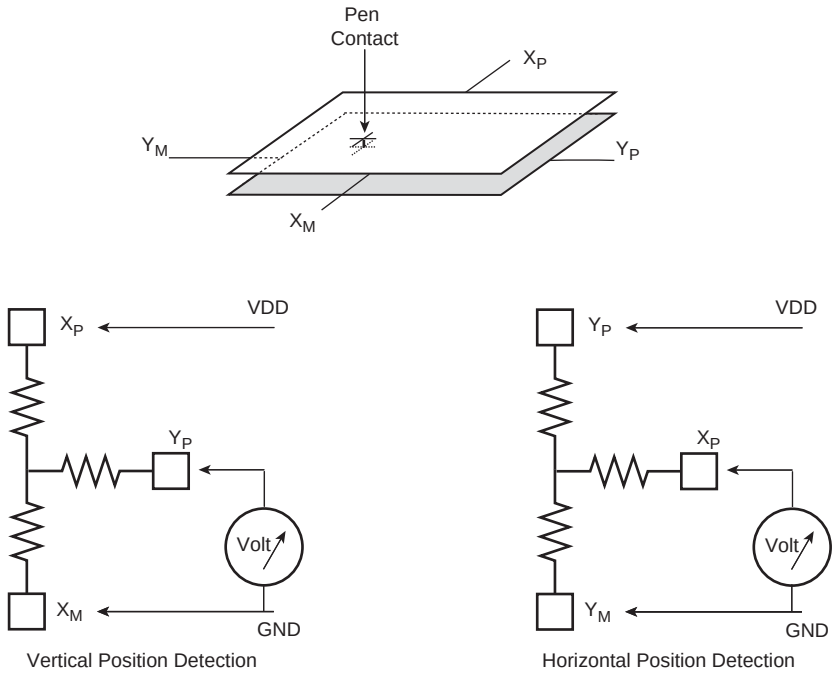
61.6.15.2 4-wire Resistive Touchscreen Principles

A resistive touchscreen is based on two resistive films, each one being fitted with a pair of electrodes, placed at the top and bottom on one film, and on the right and left on the other. In between, there is a layer acting as an insulator, but also enables contact when you press the screen. This is illustrated in Figure 61-14.

The ADC controller can perform the following tasks without external components:

- position measurement
- pressure measurement
- pen detection

Figure 61-14. Touchscreen Position Measurement



61.6.15.3 4-wire Position Measurement Method

As shown in Figure 61-14, to detect the position of a contact, a supply is first applied from top to bottom. Due to the linear resistance of the film, there is a voltage gradient from top to bottom. When a contact is performed on the screen, the voltage propagates at the point the two surfaces come into contact with the second film. If the input impedance on the right and left electrodes sense is high enough, the film does not affect this voltage, despite its resistive nature.

For the horizontal direction, the same method is used, but by applying supply from left to right. The range depends on the supply voltage and on the loss in the switches that connect to the top and bottom electrodes.

In an ideal world (linear, with no loss through switches), the horizontal position is equal to:

VY_M / VDD or VY_P / VDD.

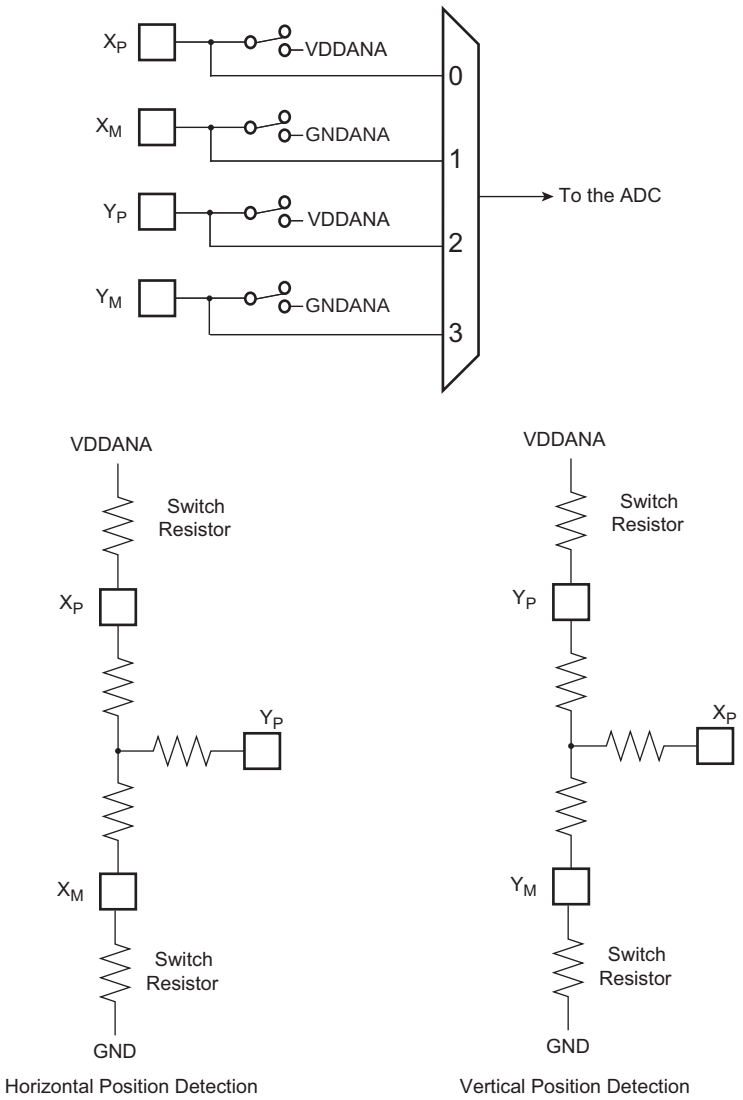
The implementation with on-chip power switches is shown in Figure 61-15. The voltage measurement at the output of the switch compensates for the switches loss.

It is possible to correct for switch loss by performing the operation:

$[VY_P - VX_M] / [VX_P - VX_M]$.

This requires additional measurements, as shown in Figure 61-15.

Figure 61-15. Touchscreen Switches Implementation



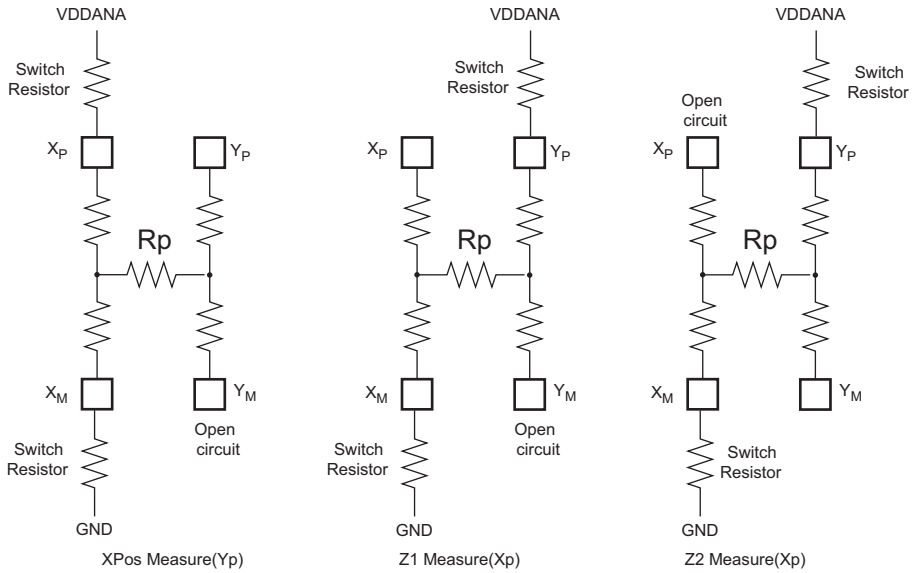
61.6.15.4 4-wire Pressure Measurement Method

The method to measure the pressure (Rp) applied to the touchscreen is based on the known resistance of the X-Panel resistance (Rxp).

Three conversions (Xpos,Z1,Z2) are necessary to determine the value of Rp (Zaxis resistance).

$Rp = Rxp \times (Xpos/1024) \times [(Z2/Z1)-1]$

Figure 61-16. Pressure Measurement



61.6.15.5 5-wire Resistive Touchscreen Principles

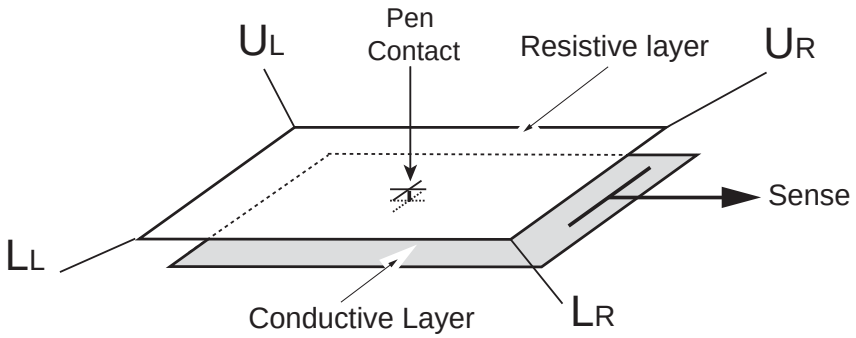
To make a 5-wire touchscreen, a resistive layer with a contact point at each corner and a conductive layer are used.

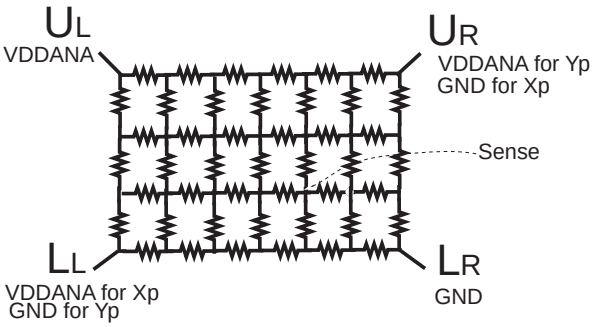
The 5-wire touchscreen differs from the 4-wire type mainly in that the voltage gradient is applied only to one layer, the resistive layer, while the other layer is the sense layer for both measurements.

The measurement of the X position is obtained by biasing the upper left corner and lower left corner to VDDANA and the upper right corner and lower right to ground.

To measure along the Y axis, bias the upper left corner and upper right corner to VDDANA and bias the lower left corner and lower right corner to ground.

Figure 61-17. 5-Wire Principle





61.6.15.6 5-wire Position Measurement Method

In an application only monitoring clicks, 100 points per second is typically needed. For handwriting or motion detection, the number of measurements to consider is approximately 200 points per second. This must take into account that multiple measurements are included (over sampling, filtering) to compute the correct point.

The 5-wire touchscreen panel works by applying a voltage at the corners of the resistive layer and measuring the vertical or horizontal resistive network with the sense input. The ADC converts the voltage measured at the point the panel is touched.

A measurement of the Y position of the pointing device is made by:

- Connecting Upper left (UL) and upper right (UR) corners to VDDANA
- Connecting Lower left (LL) and lower right (LR) corners to ground.

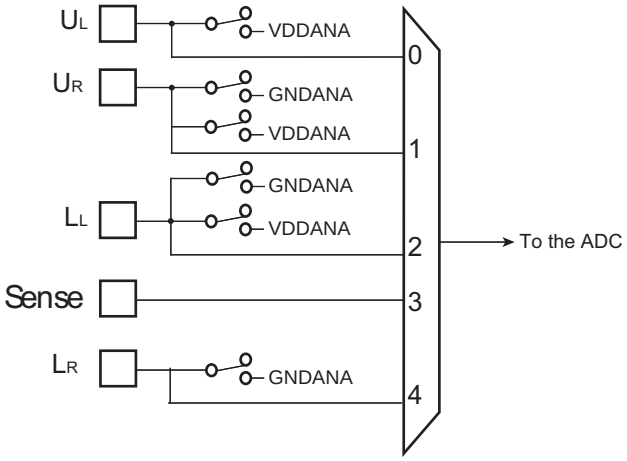
The voltage measured is determined by the voltage divider developed at the point of touch (Y position) and the SENSE input is converted by ADC.

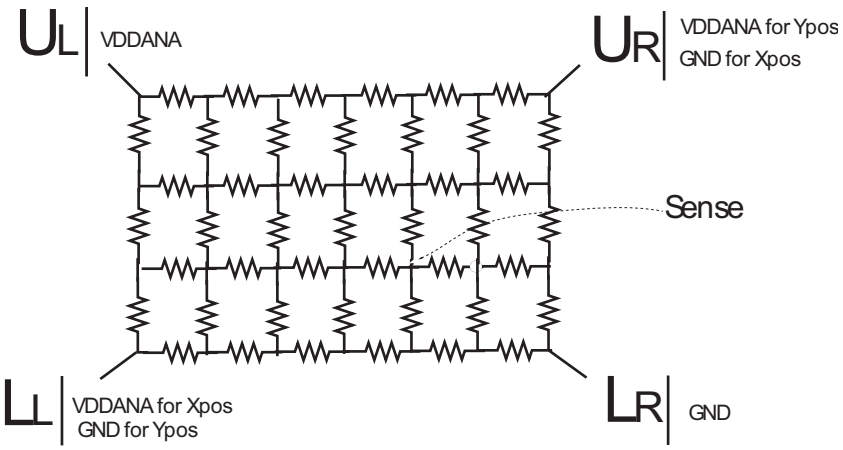
A measurement of the X position of the pointing device is made by:

- Connecting the upper left (UL) and lower left (LL) corners to ground
- Connecting the upper right and lower right corners to VDDANA.

The voltage measured is determined by the voltage divider developed at the point of touch (X position) and the SENSE input is converted by ADC.

Figure 61-18. Touchscreen Switches Implementation



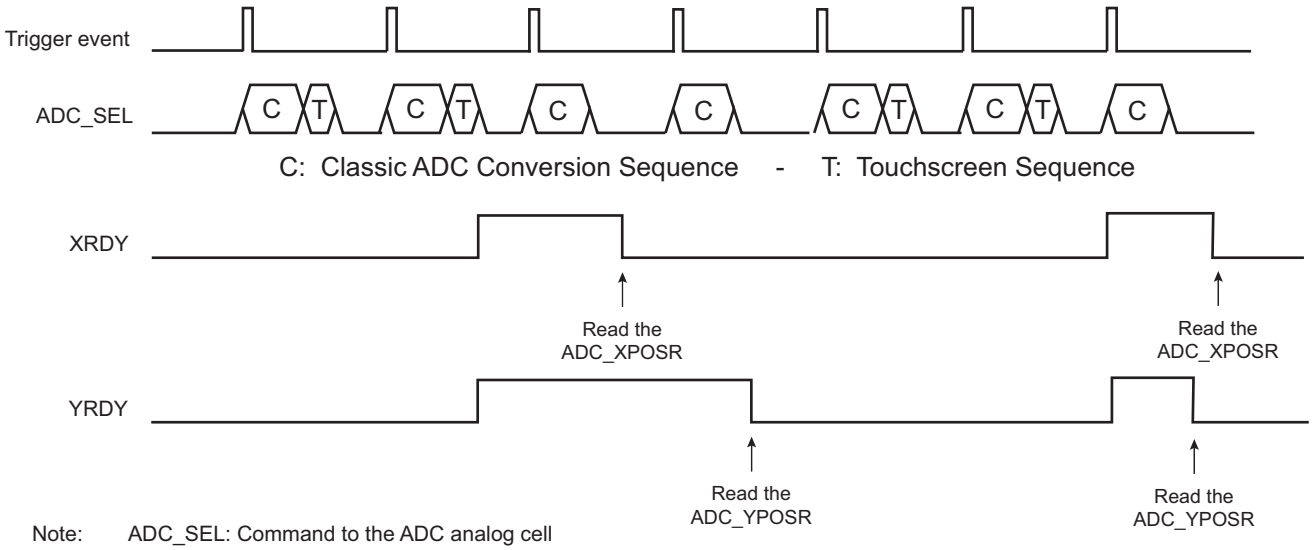


61.6.15.7 Sequence and Noise Filtering

The ADC Controller can manage ADC conversions and touchscreen measurement. On each trigger event the sequence of ADC conversions is performed as described in Section 61.6.8 “Sleep Mode and Conversion Sequencer”. The touchscreen measure frequency can be specified in number of trigger events by writing the TSFREQ parameter in ADC_TSMR. An internal counter counts triggers up to TSFREQ, and every time it rolls out, a touchscreen sequence is appended to the classic ADC conversion sequence (see Figure 61-19).

Additionally the user can average multiple touchscreen measures by writing the TSAV parameter in ADC_TSMR. This can be 1, 2, 4 or 8 measures performed on consecutive triggers as illustrated in Figure 61-19 below. Consequently, the TSFREQ parameter must be greater or equal to the TSAV parameter.

Figure 61-19. Insertion of Touchscreen Sequences (TSFREQ = 2; TSAV = 1)



61.6.15.8 Measured Values, Registers and Flags

As soon as the controller finishes the Touchscreen sequence, XRDY, YRDY and PRDY are set and can generate an interrupt. These flags can be read in the ADC Interrupt Status register (ADC_ISR). They are reset independently by reading in the ADC Touchscreen X Position register (ADC_XPOSR), the ADC Touchscreen Y Position register (ADC_YPOSR) and the ADC Touchscreen Pressure register (ADC_PRESSR).

ADC_XPOSR presents XPOS (VX - VXmin) on its LSB and XSCALE (VXMAX - VXmin) aligned on the 16th bit.

ADC_YPOSR presents YPOS (VY - VYmin) on its LSB and YSCALE (VYMAX - VYmin) aligned on the 16th bit.

To improve the quality of the measure, the user must calculate XPOS/XSCALE and YPOS/YSCALE.

VXMAX, VXmin, VYMAX, and VYmin are measured at the first startup of the controller. These values can change during use, so it can be necessary to refresh them. Refresh can be done by writing ‘1’ in the TSCALIB field of the control register (ADC_CR).

ADC_PRESSR presents Z1 on its LSB and Z2 aligned on the 16th bit. See Section 61.6.15.4 “4-wire Pressure Measurement Method”.

61.6.15.9 Pen Detect Method

When there is no contact, it is not necessary to perform a conversion. However, it is important to detect a contact by keeping the power consumption as low as possible.

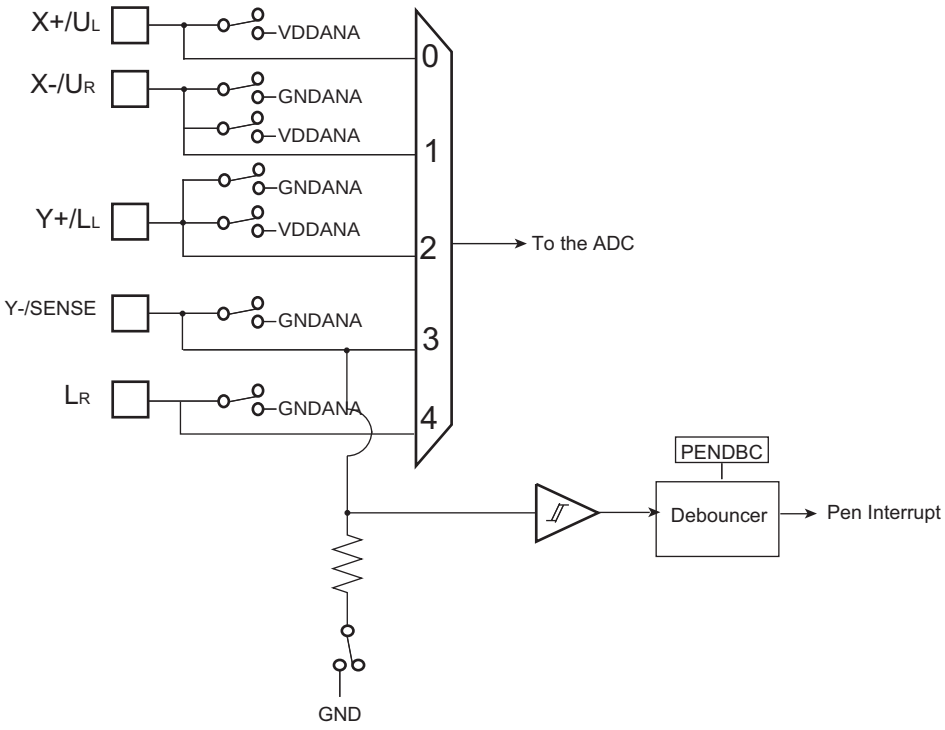
The implementation polarizes one panel by closing the switch on (X_P/U_L) and ties the horizontal panel by an embedded resistor connected to Y_M / Sense. This resistor is enabled by a fifth switch. Since there is no contact, no current is flowing and there is no related power consumption. As soon as a contact occurs, a current is flowing in the Touchscreen and a Schmitt trigger detects the voltage in the resistor.

The Touchscreen Interrupt configuration is entered by programming the PENDET bit in ADC_TSMR. If this bit is written at 1, the controller samples the pen contact state when it is not converting and waiting for a trigger.

To complete the circuit, a programmable debouncer is placed at the output of the Schmitt trigger. This debouncer is programmable up to 2^{15} ADC clock periods. The debouncer length can be selected by programming the field PENDBC in ADC_TSMR.

Due to the analog switch's structure, the debouncer circuitry is only active when no conversion (touchscreen or classic ADC channels) is in progress. Thus, if the time between the end of a conversion sequence and the arrival of the next trigger event is lower than the debouncing time configured on PENDBC, the debouncer will not detect any contact.

Figure 61-20. Touchscreen Pen Detect



The touchscreen pen detect can be used to generate an ADC interrupt to wake up the system. The pen detect generates two types of status, reported in ADC_ISR:

- the PEN bit is set as soon as a contact exceeds the debouncing time as defined by PENDBC and remains set until ADC_ISR is read.
- the NOPEN bit is set as soon as no current flows for a time over the debouncing time as defined by PENDBC and remains set until ADC_ISR is read.

Both bits are automatically cleared as soon as ADC_ISR is read, and can generate an interrupt by writing ADC_IER.

Moreover, the rising of either one of them clears the other, they cannot be set at the same time.

The PENS bit of ADC_ISR shows the current status of the pen contact.

61.6.16 Asynchronous and Partial Wakeup (SleepWalking)

This operating mode is a means of data pre-processing that qualifies an incoming event, thus allowing the ADC to decide whether or not to wake up the system. Asynchronous and partial wakeup is mainly used when the system is in Wait mode (see the PMC section for further details). It can also be enabled when the system is fully running.

Once the Asynchronous and partial wakeup mode is enabled, no access must be performed in the ADC before a wakeup is performed by the ADC.

When the Asynchronous and partial wakeup mode is enabled for the ADC (see the PMC section), the PMC decodes a clock request from the ADC. The clock request is generated as soon as a trigger event occurs. Only a trigger from RTC or ADTRG pin can be used in partial wakeup mode. The selection between RTC or ADTRG pin is performed through the ADC_MR.TRGSEL field.

If the system is in Wait mode (processor and peripheral clocks switched off), the PMC restarts the fast RC oscillator and provides the clock only to the ADC.

To perform a conversion at regular intervals with RTC trigger, the RTC must be configured with the following settings: RTC_MR.OUT0=7 and RTC_MR.THIGH=7. The period of the trigger can be defined in RTC_MR.TPERIOD.

To trigger a conversion using the ADTRG pin, the minimum high level duration of the ADTRG signal must be greater than 2 clock periods of the fast RC oscillator. The maximum duration of the high level must be limited to the amount of startup and conversion time.

As soon as the clock is provided by the PMC, the ADC processes the conversions and compares the converted values with LOWTHRES and HIGHTHRES field values in ADC_CWR.

The ADC instructs the PMC to disable the clock if the converted value does not meet the conditions defined by LOWTHRES and HIGHTHRES field values in ADC_CWR.

If the converted value meets the conditions, the ADC instructs the PMC to exit the full system from Wait mode.

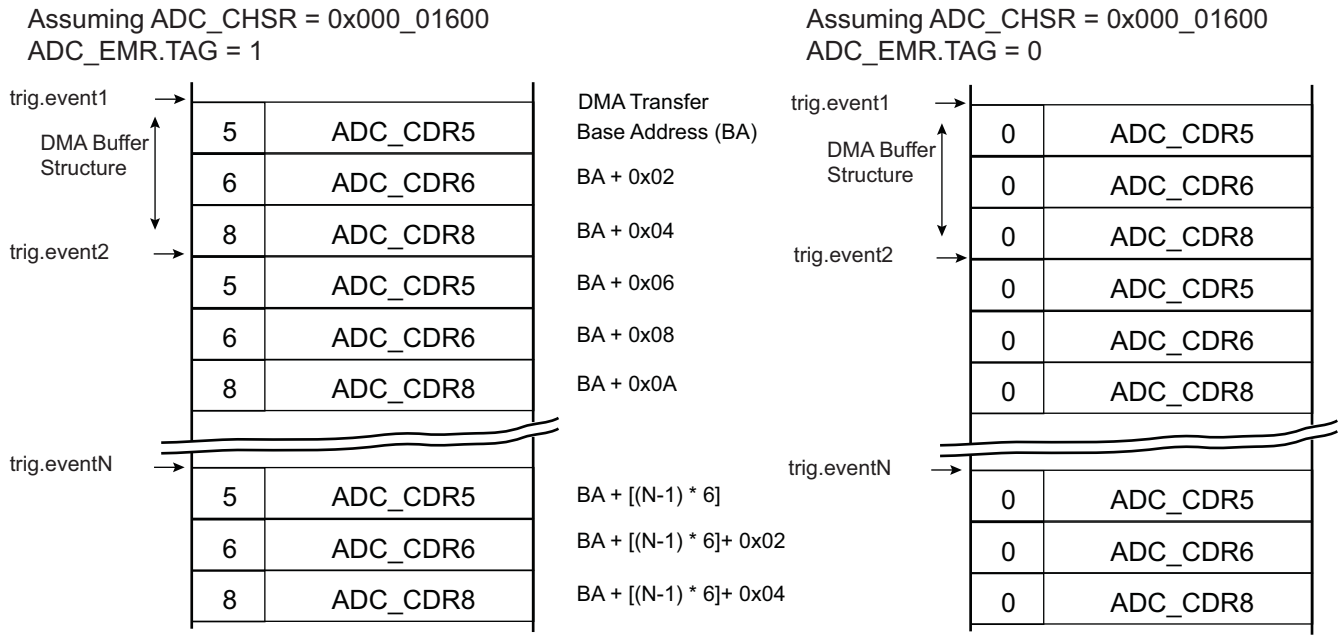
If the processor and peripherals are running, the ADC can be configured in Asynchronous and partial wakeup mode by enabling the PMC_SLPWK_ER (see the PMC section). When a trigger event occurs, the ADC requests the clock from the PMC and the comparison is performed. If there is a comparison match, the ADC continues to request the clock. If there is no match, the clock is switched off for the ADC only, until a new trigger event is detected.

It is recommended to write a ‘1’ to the SLEEP bit to reduce the power consumption of the analog part of the ADC when the system is waiting for a trigger event.

61.6.17 Buffer Structure

The DMA read channel is triggered each time a new data is stored in ADC_LCDR. The same structure of data is repeatedly stored in ADC_LCDR each time a trigger event occurs. Depending on user mode of operation (ADC_MR, ADC_CHSR, ADC_SEQR1, ADC_SEQR2, ADC_TSMR) the structure differs. Each data read to DMA buffer, carried on a half-word (16-bit), consists of last converted data right aligned and when TAG is set in ADC_EMR, the four most significant bits are carrying the channel number thus allowing an easier post-processing in the DMA buffer or better checking the DMA buffer integrity.

Figure 61-21. Buffer Structure



As soon as touchscreen conversions are required, the pen detection function may help the post-processing of the buffer. Refer to Section 61.6.17.4 “Pen Detection Status”.

61.6.17.1 Classic ADC Channels Only (Touchscreen Disabled)

When no touchscreen conversion is required (i.e., TSMODE = 0 in ADC_TSMR), the structure of data within the buffer is defined by ADC_MR, ADC_CHSR, ADC_SEQRx. See Figure 61-21.

If the user sequence is not used (i.e., USEQ is cleared in ADC_MR) then only the value of ADC_CHSR defines the data structure. For each trigger event, enabled channels will be consecutively stored in ADC_LCDR and automatically read to the buffer.

When the user sequence is configured (i.e., USEQ is set in ADC_MR) not only does ADC_CHSR modify the data structure of the buffer, but ADC_SEQRx registers may modify the data structure of the buffer as well.

61.6.17.2 Touchscreen Channels Only

When only touchscreen conversions are required (i.e., TSMODE ≠ 0 in ADC_TSMR and ADC_CHSR equals 0), the structure of data within the buffer is defined by ADC_TSMR.

When TSMODE = 1 or 3, each trigger event adds two half-words in the buffer (assuming TSAV = 0), first half-word being XPOS of ADC_XPOSR then YPOS of ADC_YPOSR. If TSAV/TSFREQ ≠ 0, the data structure remains unchanged. Not all trigger events add data to the buffer.

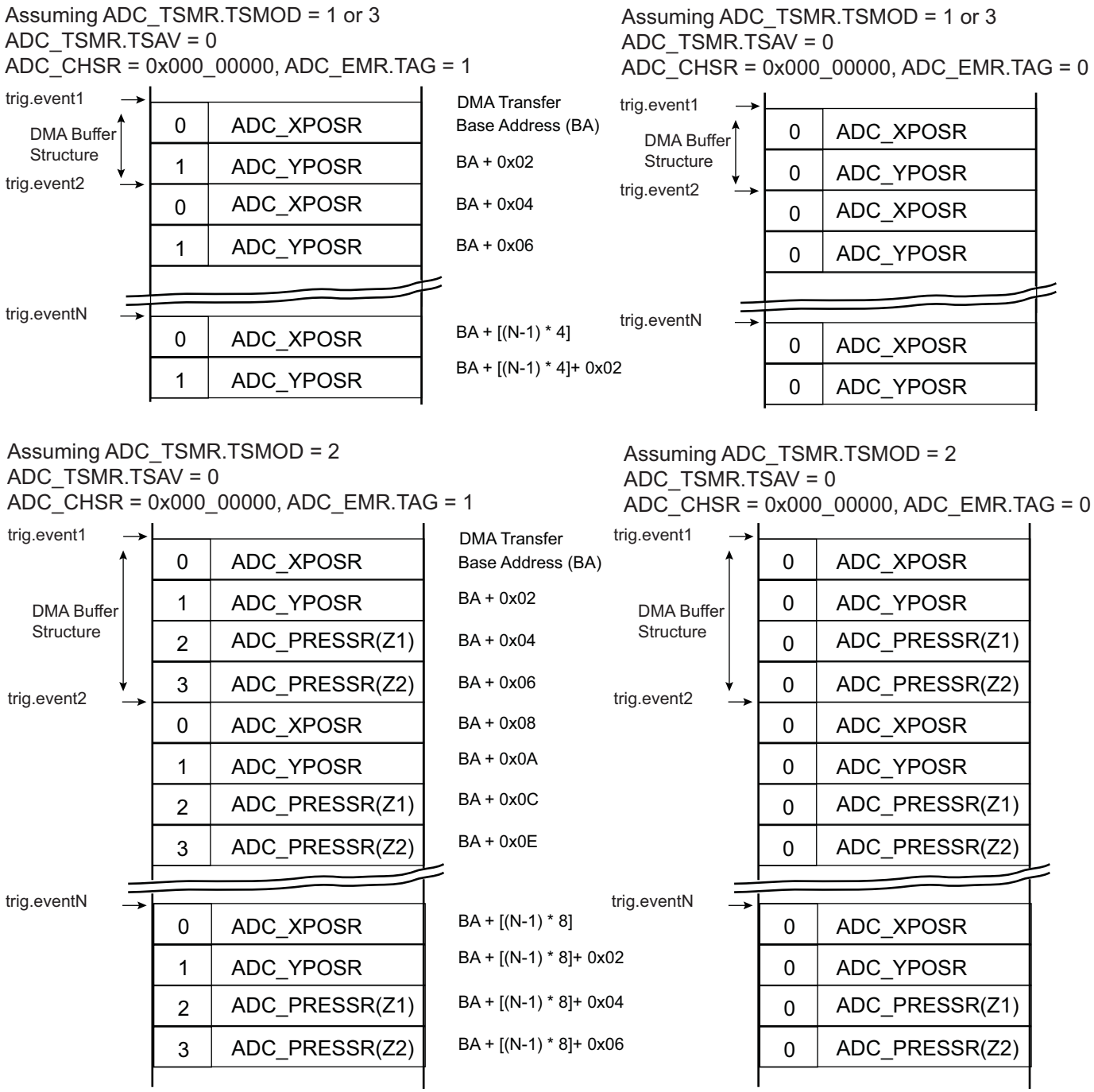
When TSMODE = 2, each trigger event adds four half-words to the buffer (assuming TSAV = 0), first half-word being XPOS of ADC_XPOSR followed by YPOS of ADC_YPOSR and finally Z1 followed by Z2, both located in ADC_PRESSR.

When TAG is set (ADC_EMR), the CHNB field (four most significant bits of ADC_LCDR) is cleared when XPOS is transmitted and set when YPOS is transmitted, allowing an easier post-processing of the buffer or a better checking of the buffer integrity. In case 4-wire with Pressure mode is selected, Z1 value is transmitted to the buffer along with tag set to 2 and Z2 is tagged with value 3.

XSCALE and YSCALE (calibration values) are not transmitted to the buffer because they are supposed to be constant and moreover only measured at the very first startup of the controller or upon user request.

There is no change in buffer structure whatever the value of PENDET bit configuration in ADC_TSMR but it is recommended to use the pen detection function for buffer post-processing (refer to Section 61.6.17.4 “Pen Detection Status”).

Figure 61-22. Buffer Structure When Only Touchscreen Channels are Enabled



61.6.17.3 Interleaved Channels

When both classic ADC channels (CH4/CH5 up to CH12 are set in ADC_CHSR) and touchscreen conversions are required (TSMODE ≠ 0 in ADC_TSMR) the structure of the buffer differs according to TSAV and TSFREQ values.

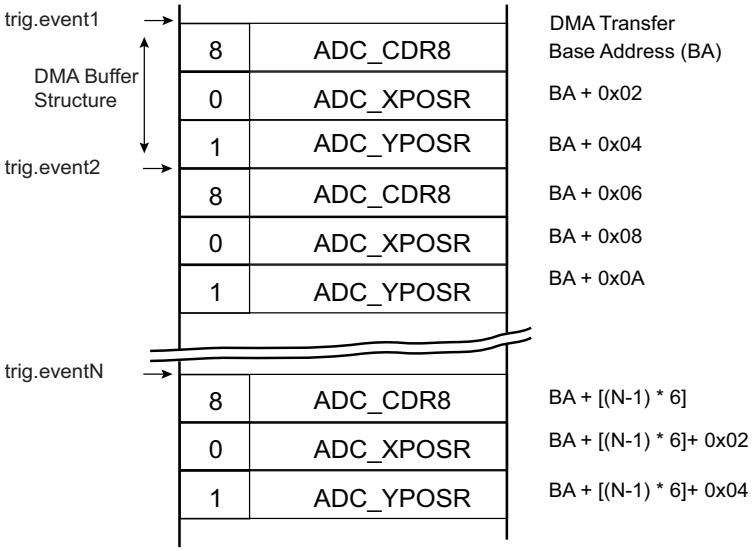
If TSFREQ ≠ 0, not all events generate touchscreen conversions, therefore the buffer structure is based on 2^{TSFREQ} trigger events. Given a TSFREQ value, the location of touchscreen conversion results depends on TSAV value.

When TSFREQ = 0, TSAV must equal 0.

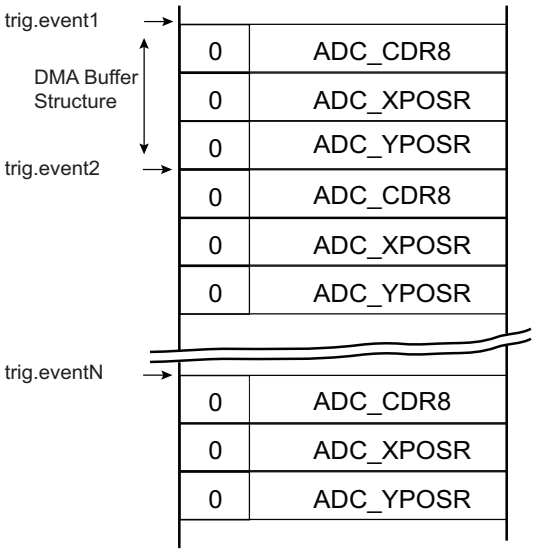
There is no change in buffer structure whatever the value of PENDET bit configuration in ADC_TSMR but it is recommended to use the pen detection function for buffer post-processing (refer to Section 61.6.17.4 “Pen Detection Status”).

Figure 61-23. Buffer Structure When Classic ADC and Touchscreen Channels are Interleaved

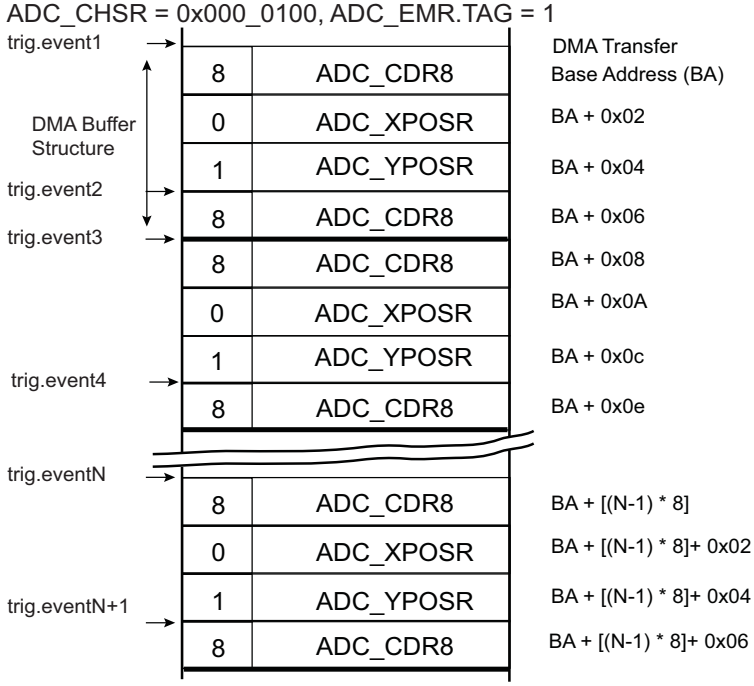
Assuming ADC_TSMR.TSMOD = 1
ADC_TSMR.TSAV = ADC_TSMR.TSFREQ = 0
ADC_CHSR = 0x000_0100, ADC_EMR.TAG = 1



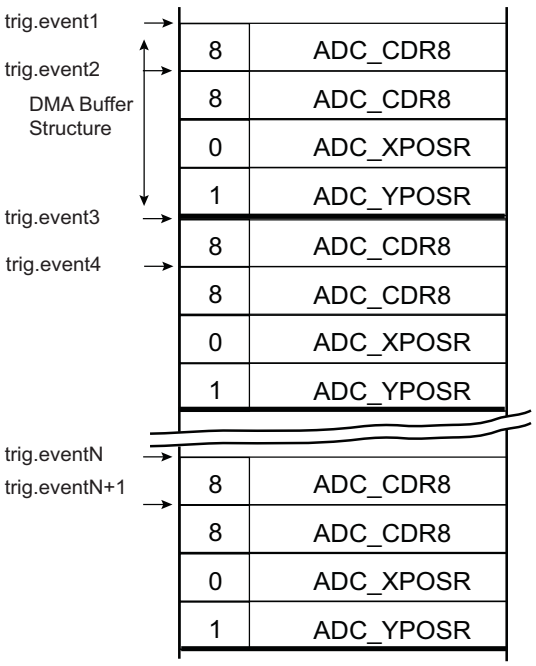
Assuming ADC_TSMR.TSMOD = 1
ADC_TSMR.TSAV = ADC_TSMR.TSFREQ = 0
ADC_CHSR = 0x000_0100, ADC_EMR.TAG = 0



Assuming ADC_TSMR.TSMOD = 1
ADC_TSMR.TSAV = 0, ADC_TSMR.TSFREQ = 1
ADC_CHSR = 0x000_0100, ADC_EMR.TAG = 1



Assuming ADC_TSMR.TSMOD = 1
ADC_TSMR.TSAV = 1, ADC_TSMR.TSFREQ = 1
ADC_CHSR = 0x000_0100, ADC_EMR.TAG = 1



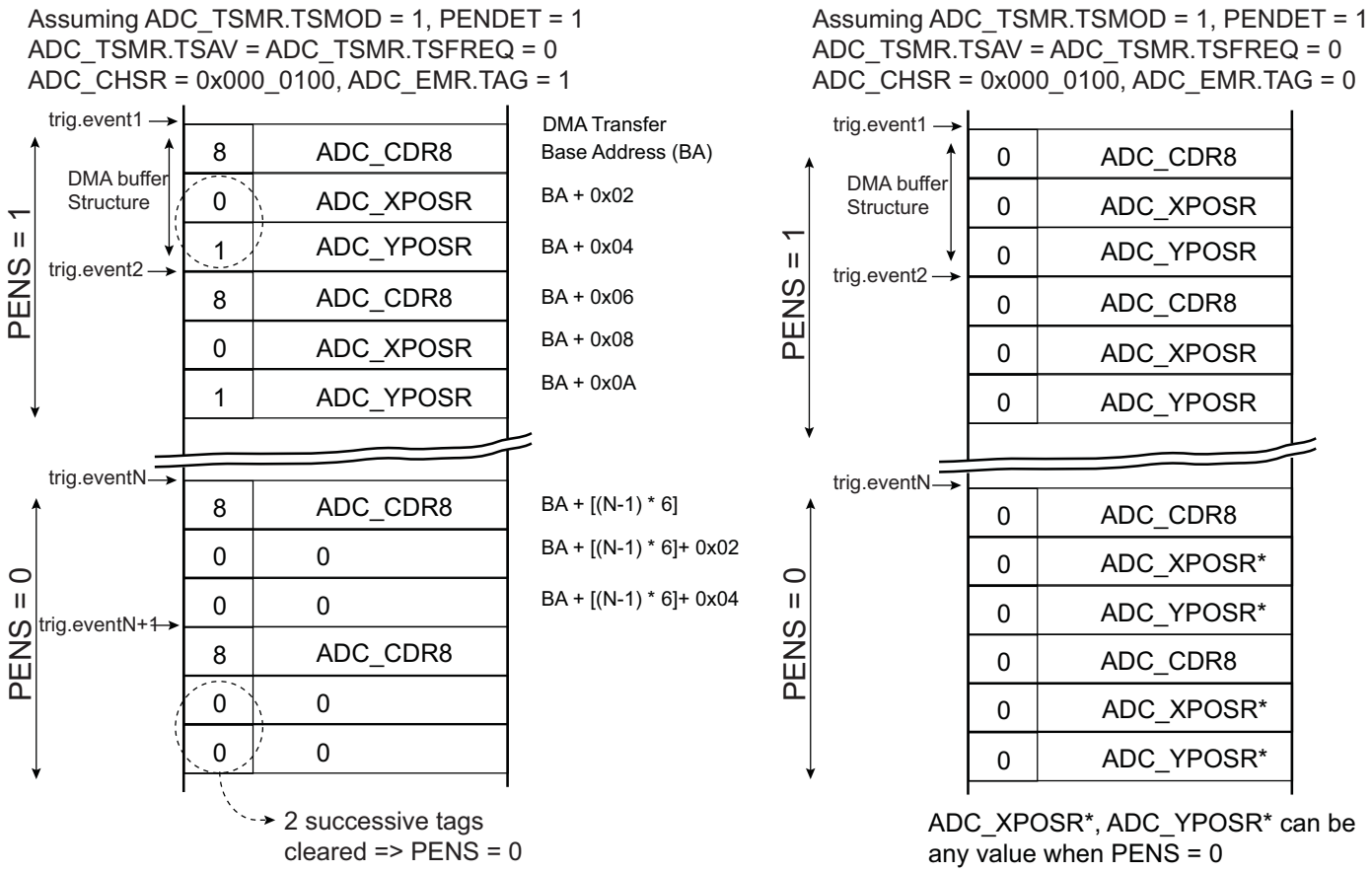
61.6.17.4 Pen Detection Status

If the pen detection measure is enabled (PENDET is set in ADC_TSMR), the XPOS, YPOS, Z1, Z2 values transmitted to the buffer through ADC_LCDR are cleared (including the CHNB field), if the PENS flag of ADC_ISR is 0. When the PENS flag is set, XPOS, YPOS, Z1, Z2 are normally transmitted.

Therefore, using pen detection together with tag function eases the post-processing of the buffer, especially to determine which touchscreen converted values correspond to a period of time when the pen was in contact with the screen.

When the pen detection is disabled or the tag function is disabled, XPOS, YPOS, Z1, Z2 are normally transmitted without tag and no relationship can be found with pen status, thus post-processing may not be easy.

Figure 61-24. Buffer Structure With and Without Pen Detection Enabled



61.6.18 Fault Output

The ADC Controller internal fault output is directly connected to PWM fault input. Fault output may be asserted depending on the configuration of ADC_EMR and ADC_CWR and converted values. When the compare occurs, the ADC fault output generates a pulse of one peripheral clock cycle to the PWM fault input. This fault line can be enabled or disabled within PWM. Should it be activated and asserted by the ADC Controller, the PWM outputs are immediately placed in a safe state (pure combinational path). Note that the ADC fault output connected to the PWM is not the COMPE bit. Thus the Fault mode (FMOD) within the PWM configuration must be FMOD = 1.

61.6.19 Register Write Protection

To prevent any single software error from corrupting ADC behavior, certain registers in the address space can be write-protected by setting the bit WPEN in the "ADC Write Protection Mode Register" (ADC_WPMR).

If a write access to the protected registers is detected, the WPVS flag in the "ADC Write Protection Status Register" (ADC_WPSR) is set and the field WPVSRC indicates the register in which the write access has been attempted.

The WPVS flag is automatically reset by reading ADC_WPSR.

The following registers are write-protected when WPEN is set in ADC_WPMR:

- ADC Mode Register
- ADC Channel Sequence 1 Register
- ADC Channel Sequence 2 Register
- ADC Channel Enable Register
- ADC Channel Disable Register
- ADC Last Channel Trigger Mode Register
- ADC Last Channel Compare Window Register
- ADC Extended Mode Register
- ADC Compare Window Register
- ADC Channel Offset Register
- ADC Analog Control Register
- ADC Touchscreen Mode Register
- ADC Trigger Register
- ADC Correction Values Register
- ADC Channel Error Correction Register
- ADC Touchscreen Correction Values Register

61.7 Analog-to-Digital (ADC) User Interface

Table 61-8. Register Mapping

Offset	Register	Name	Access	Reset
0x00	Control Register	ADC_CR	Write-only	–
0x04	Mode Register	ADC_MR	Read/Write	0x00000000
0x08	Channel Sequence Register 1	ADC_SEQR1	Read/Write	0x00000000
0x0C	Channel Sequence Register 2	ADC_SEQR2	Read/Write	0x00000000
0x10	Channel Enable Register	ADC_CHER	Write-only	–
0x14	Channel Disable Register	ADC_CHDR	Write-only	–
0x18	Channel Status Register	ADC_CHSR	Read-only	0x00000000
0x1C	Reserved	–	–	–
0x20	Last Converted Data Register	ADC_LCDR	Read-only	0x00000000
0x24	Interrupt Enable Register	ADC_IER	Write-only	–
0x28	Interrupt Disable Register	ADC_IDR	Write-only	–
0x2C	Interrupt Mask Register	ADC_IMR	Read-only	0x00000000
0x30	Interrupt Status Register	ADC_ISR	Read-only	0x00000000
0x34	Last Channel Trigger Mode Register	ADC_LCTMR	Read/Write	0x00000000
0x38	Last Channel Compare Window Register	ADC_LCCWR	Read/Write	0x00000000
0x3C	Overrun Status Register	ADC_OVER	Read-only	0x00000000
0x40	Extended Mode Register	ADC_EMR	Read/Write	0x00000000
0x44	Compare Window Register	ADC_CWR	Read/Write	0x00000000
0x48	Reserved	–	–	–
0x4C	Channel Offset Register	ADC_COR	Read/Write	0x00000000
0x50	Channel Data Register 0	ADC_CDR0	Read-only	0x00000000
0x54	Channel Data Register 1	ADC_CDR1	Read-only	0x00000000
...	...	...	...	...
0x7C	Channel Data Register 11	ADC_CDR11	Read-only	0x00000000
0x80–0x90	Reserved	–	–	–
0x94	Analog Control Register	ADC_ACR	Read/Write	0x00000101
0x98–0xAC	Reserved	–	–	–
0xB0	Touchscreen Mode Register	ADC_TSMR	Read/Write	0x00000000
0xB4	Touchscreen X Position Register	ADC_XPOSR	Read-only	0x00000000
0xB8	Touchscreen Y Position Register	ADC_YPOSR	Read-only	0x00000000
0xBC	Touchscreen Pressure Register	ADC_PRESSR	Read-only	0x00000000
0xC0	Trigger Register	ADC_TRGR	Read/Write	0x00000000
0xC4–0xD0	Reserved	–	–	–
0xD4	Correction Values Register	ADC_CVR	Read/Write	0x00000000
0xD8	Channel Error Correction Register	ADC_CECR	Read/Write	0x00000000

Table 61-8. Register Mapping (Continued)

Offset	Register	Name	Access	Reset
0xDC	Touchscreen Correction Values Register	ADC_TSCVR	Read/Write	0x00000000
0xE0	Reserved	–	–	–
0xE4	Write Protection Mode Register	ADC_WPMR	Read/Write	0x00000000
0xE8	Write Protection Status Register	ADC_WPSR	Read-only	0x00000000
0xEC–0xFC	Reserved	–	–	–

Note: Any offset not listed in the table must be considered as “reserved”.

61.7.1 ADC Control Register

Name: ADC_CR

Address: 0xFC030000

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	CMPRST	–	TSCALIB	START	SWRST

- **SWRST: Software Reset**

0: No effect.

1: Resets the ADC, simulating a hardware reset.

- **START: Start Conversion**

0: No effect.

1: Begins analog-to-digital conversion.

- **TSCALIB: Touchscreen Calibration**

0: No effect.

1: Programs screen calibration (VDD/GND measurement)

If conversion is in progress, the calibration sequence starts at the beginning of a new conversion sequence. If no conversion is in progress, the calibration sequence starts at the second conversion sequence located after the TSCALIB command (Sleep mode, waiting for a trigger event).

TSCALIB measurement sequence does not affect the Last Converted Data Register (ADC_LCDR).

- **CMPRST: Comparison Restart**

0: No effect.

1: Stops the conversion result storage until the next comparison match.

61.7.2 ADC Mode Register

Name: ADC_MR

Address: 0xFC030004

Access: Read/Write

31	30	29	28	27	26	25	24
USEQ	–	TRANSFER		TRACKTIM			
23	22	21	20	19	18	17	16
ANACH	–	–	–	STARTUP			
15	14	13	12	11	10	9	8
PRESCAL							
7	6	5	4	3	2	1	0
–	FWUP	SLEEP	–	TRGSEL			–

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **TRGSEL: Trigger Selection**

Value	Name	Description
0	ADC_TRIG0	ADTRG
1	ADC_TRIG1	TIOA0
2	ADC_TRIG2	TIOA1
3	ADC_TRIG3	TIOA2
4	ADC_TRIG4	PWM event line 0
5	ADC_TRIG5	PWM event line 1
6	ADC_TRIG6	TIOA3
7	ADC_TRIG7	RTCOUT0

Note: The trigger selection can be performed only if TRGMOD = 1, 2 or 3 in ADC Trigger Register.

- **SLEEP: Sleep Mode**

Value	Name	Description
0	NORMAL	Normal Mode: The ADC core and reference voltage circuitry are kept ON between conversions.
1	SLEEP	Sleep Mode: The wakeup time can be modified by programming the FWUP bit.

- **FWUP: Fast Wakeup**

Value	Name	Description
0	OFF	If SLEEP is 1, then both ADC core and reference voltage circuitry are OFF between conversions
1	ON	If SLEEP is 1, then Fast Wakeup Sleep mode: The voltage reference is ON between conversions and ADC core is OFF

- **PRESCAL: Prescaler Rate Selection**

  $PRESCAL = (f_{peripheral\ clock} / (2 \times f_{ADCCLK})) - 1.$

- **STARTUP: Startup Time**

Value	Name	Description
0	SUT0	0 periods of ADCCLK
1	SUT8	8 periods of ADCCLK
2	SUT16	16 periods of ADCCLK
3	SUT24	24 periods of ADCCLK
4	SUT64	64 periods of ADCCLK
5	SUT80	80 periods of ADCCLK
6	SUT96	96 periods of ADCCLK
7	SUT112	112 periods of ADCCLK
8	SUT512	512 periods of ADCCLK
9	SUT576	576 periods of ADCCLK
10	SUT640	640 periods of ADCCLK
11	SUT704	704 periods of ADCCLK
12	SUT768	768 periods of ADCCLK
13	SUT832	832 periods of ADCCLK
14	SUT896	896 periods of ADCCLK
15	SUT960	960 periods of ADCCLK

- **ANACH: Analog Change**

Value	Name	Description
0	NONE	No analog change on channel switching: DIFF0 is used for all channels.
1	ALLOWED	Allows different analog settings for each channel. See ADC Channel Offset Register.

- **TRACKTIM: Tracking Time**

Value	Name	Description
0	ADCCLK6	The tracking time is 6 ADC clock cycles.
1–14	–	The tracking time is 6 ADC clock cycles.
15	ADCCLK7	The tracking time is 7 ADC clock cycles.

- **TRANSFER: Transfer Time**

  The TRANSFER field must be set to 2 to guarantee the optimal transfer time.

- **USEQ: Use Sequence Enable**

Value	Name	Description
0	NUM_ORDER	Normal mode: The controller converts channels in a simple numeric order depending only on the channel index.
1	REG_ORDER	User Sequence mode: The sequence respects what is defined in ADC_SEQR1 and ADC_SEQR2 registers and can be used to convert the same channel several times.

61.7.3 ADC Channel Sequence 1 Register

Name: ADC_SEQR1

Address: 0xFC030008

Access: Read/Write

31	30	29	28	27	26	25	24
USCH8				USCH7			
23	22	21	20	19	18	17	16
USCH6				USCH5			
15	14	13	12	11	10	9	8
USCH4				USCH3			
7	6	5	4	3	2	1	0
USCH2				USCH1			

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **USCHx: User Sequence Number x**

The allowed range is 0 up to 11, thus only the sequencer from CH0 to CH11 can be used.

This register activates only if the USEQ field in ADC_MR field is set to ‘1’.

Any USCHx field is processed only if the CHx-1 it in ADC_CHSR reads logical ‘1’, else any value written in USCHx does not add the corresponding channel in the conversion sequence.

Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, according to user needs.

When configuring consecutive fields with the same value, the associated channel is sampled as many time as the number of consecutive values, this part of the conversion sequence being triggered by a unique event.

61.7.4 ADC Channel Sequence 2 Register

Name: ADC_SEQR2

Address: 0xFC03000C

Access: Read/Write

31	30	29	28	27	26	25	24
–				–			
23	22	21	20	19	18	17	16
–				–			
15	14	13	12	11	10	9	8
–				USCH11			
7	6	5	4	3	2	1	0
USCH10				USCH9			

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **USCHx: User Sequence Number x**

The sequence number x (USCHx) can be programmed by the Channel number CHy where y is the value written in this field. The allowed range is 0 up to 11. So it is only possible to use the sequencer from CH0 to CH11.

This register activates only if the USEQ field in ADC_MR is set to ‘1’.

Any USCHx field is processed only if the CHx-1 bit in ADC_CHSR reads logical ‘1’. Else, any value written in USCHx does not add the corresponding channel in the conversion sequence.

Configuring the same value in different fields leads to multiple samples of the same channel during the conversion sequence. This can be done consecutively, or not, according to user needs.

61.7.5 ADC Channel Enable Register

Name: ADC_CHER

Address: 0xFC030010

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CH11	CH10	CH9	CH8
7	6	5	4	3	2	1	0
CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **CHx: Channel x Enable**

0: No effect.

1: Enables the corresponding channel.

Note: If USEQ = 1 in ADC_MR, CHx corresponds to the enable of sequence number x+1 described in ADC_SEQR1 and ADC_SEQR2 (e.g. CH0 enables sequence number USCH1).

61.7.6 ADC Channel Disable Register

Name: ADC_CHDR

Address: 0xFC030014

Access: Write-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CH11	CH10	CH9	CH8
7	6	5	4	3	2	1	0
CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **CHx: Channel x Disable**

0: No effect.

1: Disables the corresponding channel.

Warning: If the corresponding channel is disabled during a conversion or if it is disabled and then reenabled during a conversion, its associated data and corresponding EOCx and GOVRE flags in ADC_ISR and OVREx flags in ADC_OVER are unpredictable.

61.7.7 ADC Channel Status Register

Name: ADC_CHSR

Address: 0xFC030018

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	CH11	CH10	CH9	CH8
7	6	5	4	3	2	1	0
CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0

- **CHx: Channel x Status**

0: The corresponding channel is disabled.

1: The corresponding channel is enabled.

61.7.8 ADC Last Converted Data Register

Name: ADC_LCDR

Address: 0xFC030020

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	CHNBOSR				
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
LDATA							
7	6	5	4	3	2	1	0
LDATA							

- **LDATA: Last Data Converted**

The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed.

If OSR = 0 and TAG = 1 in ADC_EMR, the 4 MSB of LDATA carry the channel number to obtain a packed system memory buffer made of 1 converted data stored in a halfword (16-bit) instead of 1 converted data in a 32-bit word, thus dividing by 2 the size of the memory buffer.

- **CHNBOSR: Channel Number in Oversampling Mode**

Indicates the last converted channel when the TAG bit is set in ADC_EMR and the OSR field is not equal to 0 in ADC_EMR0. If the TAG bit is not set, CHNBOSR = 0.

61.7.9 ADC Interrupt Enable Register

Name: ADC_IER

Address: 0xFC030024

Access: Write-only

31	30	29	28	27	26	25	24
–	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	LCCHG	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOC11	EOC10	EOC9	EOC8
7	6	5	4	3	2	1	0
EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Enables the corresponding interrupt.

- **EOCx: End of Conversion Interrupt Enable x**
- **LCCHG: Last Channel Change Interrupt Enable**
- **XRDY: Touchscreen Measure XPOS Ready Interrupt Enable**
- **YRDY: Touchscreen Measure YPOS Ready Interrupt Enable**
- **PRDY: Touchscreen Measure Pressure Ready Interrupt Enable**
- **DRDY: Data Ready Interrupt Enable**
- **GOVRE: General Overrun Error Interrupt Enable**
- **COMPE: Comparison Event Interrupt Enable**
- **PEN: Pen Contact Interrupt Enable**
- **NOPEN: No Pen Contact Interrupt Enable**

61.7.10 ADC Interrupt Disable Register

Name: ADC_IDR

Address: 0xFC030028

Access: Write-only

31	30	29	28	27	26	25	24
–	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	LCCHG	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOC11	EOC10	EOC9	EOC8
7	6	5	4	3	2	1	0
EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0

The following configuration values are valid for all listed bit names of this register:

0: No effect.

1: Disables the corresponding interrupt.

- **EOCx: End of Conversion Interrupt Disable x**
- **LCCHG: Last Channel Change Interrupt Disable**
- **XRDY: Touchscreen Measure XPOS Ready Interrupt Disable**
- **YRDY: Touchscreen Measure YPOS Ready Interrupt Disable**
- **PRDY: Touchscreen Measure Pressure Ready Interrupt Disable**
- **DRDY: Data Ready Interrupt Disable**
- **GOVRE: General Overrun Error Interrupt Disable**
- **COMPE: Comparison Event Interrupt Disable**
- **PEN: Pen Contact Interrupt Disable**
- **NOPEN: No Pen Contact Interrupt Disable**

61.7.11 ADC Interrupt Mask Register

Name: ADC_IMR

Address: 0xFC03002C

Access: Read-only

31	30	29	28	27	26	25	24
–	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	LCCHG	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOC11	EOC10	EOC9	EOC8
7	6	5	4	3	2	1	0
EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0

The following configuration values are valid for all listed bit names of this register:

0: The corresponding interrupt is disabled.

1: The corresponding interrupt is enabled.

- **EOCx: End of Conversion Interrupt Mask x**
- **LCCHG: Last Channel Change Interrupt Mask**
- **XRDY: Touchscreen Measure XPOS Ready Interrupt Mask**
- **YRDY: Touchscreen Measure YPOS Ready Interrupt Mask**
- **PRDY: Touchscreen Measure Pressure Ready Interrupt Mask**
- **DRDY: Data Ready Interrupt Mask**
- **GOVRE: General Overrun Error Interrupt Mask**
- **COMPE: Comparison Event Interrupt Mask**
- **PEN: Pen Contact Interrupt Mask**
- **NOPEN: No Pen Contact Interrupt Mask**

61.7.12 ADC Interrupt Status Register

Name: ADC_ISR

Address: 0xFC030030

Access: Read-only

31	30	29	28	27	26	25	24
PENS	NOPEN	PEN	–	–	COMPE	GOVRE	DRDY
23	22	21	20	19	18	17	16
–	PRDY	YRDY	XRDY	LCCHG	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	EOC11	EOC10	EOC9	EOC8
7	6	5	4	3	2	1	0
EOC7	EOC6	EOC5	EOC4	EOC3	EOC2	EOC1	EOC0

- **EOCx: End of Conversion x (automatically set / cleared)**

0: The corresponding analog channel is disabled, or the conversion is not finished. This flag is cleared when reading the corresponding ADC_CDRx registers.

1: The corresponding analog channel is enabled and conversion is complete.

- **LCCHG: Last Channel Change (cleared on read)**

0: There is no comparison match (defined in the Last Channel Compare Window register (ADC_LCCWR) since the last read of ADC_ISR.

1: The temperature value reported on ADC_CDR11 has changed since the last read of ADC_ISR, according to what is defined in the Last Channel Trigger Mode register (ADC_LCTMR) and Last Channel Compare Window register (ADC_LCCWR).

- **XRDY: Touchscreen XPOS Measure Ready (cleared on read)**

0: No measure has been performed since the last read of ADC_XPOSR.

1: At least one measure has been performed since the last read of ADC_ISR.

- **YRDY: Touchscreen YPOS Measure Ready (cleared on read)**

0: No measure has been performed since the last read of ADC_YPOSR.

1: At least one measure has been performed since the last read of ADC_ISR.

- **PRDY: Touchscreen Pressure Measure Ready (cleared on read)**

0: No measure has been performed since the last read of ADC_PRESSR.

1: At least one measure has been performed since the last read of ADC_ISR.

- **DRDY: Data Ready (automatically set / cleared)**

0: No data has been converted since the last read of ADC_LCDR.

1: At least one data has been converted and is available in ADC_LCDR.

- **GOVRE: General Overrun Error (cleared on read)**

0: No general overrun error occurred since the last read of ADC_ISR.

1: At least one general overrun error has occurred since the last read of ADC_ISR.

- **COMPE: Comparison Event (cleared on read)**

0: No comparison event since the last read of ADC_ISR.

1: At least one comparison event (defined in ADC_EMR and ADC_CWR) has occurred since the last read of ADC_ISR.

- **PEN: Pen contact (cleared on read)**

0: No pen contact since the last read of ADC_ISR.

1: At least one pen contact since the last read of ADC_ISR.

- **NOPEN: No Pen Contact (cleared on read)**

0: No loss of pen contact since the last read of ADC_ISR.

1: At least one loss of pen contact since the last read of ADC_ISR.

- **PENS: Pen Detect Status**

0: The pen does not press the screen.

1: The pen presses the screen.

Note: PENS is not a source of interruption.

61.7.13 ADC Last Channel Trigger Mode Register

Name: ADC_LCTMR

Address: 0xFC030034

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	CMPMOD		–	–	–	DUALTRIG

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **DUALTRIG: Dual Trigger ON**

0: All channels are triggered by event defined by TRGSEL in ADC_MR.

1: Last channel (higher index) trigger period is defined by OUT1 in RTC_MR.

- **CMPMOD: Last Channel Comparison Mode**

Value	Name	Description
0	LOW	Generates the LCCHG flag in ADC_ISR when the converted data is lower than the low threshold of the window.
1	HIGH	Generates the LCCHG flag in ADC_ISR when the converted data is higher than the high threshold of the window.
2	IN	Generates the LCCHG flag in ADC_ISR when the converted data is in the comparison window.
3	OUT	Generates the LCCHG flag in ADC_ISR when the converted data is out of the comparison window.

61.7.14 ADC Last Channel Compare Window Register

Name: ADC_LCCWR

Address: 0xFC030038

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	HIGHTHRES			
23	22	21	20	19	18	17	16
HIGHTHRES							
15	14	13	12	11	10	9	8
–	–	–	–	LOWTHRES			
7	6	5	4	3	2	1	0
LOWTHRES							

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **LOWTHRES: Low Threshold**

Low threshold associated to compare settings of ADC_LCTMR.

- **HIGHTHRES: High Threshold**

High threshold associated to compare settings of ADC_LCTMR.

61.7.15 ADC Overrun Status Register

Name: ADC_OVER

Address: 0xFC03003C

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	OVRE11	OVRE10	OVRE9	OVRE8
7	6	5	4	3	2	1	0
OVRE7	OVRE6	OVRE5	OVRE4	OVRE3	OVRE2	OVRE1	OVRE0

- **OVREx: Overrun Error x**

0: No overrun error on the corresponding channel since the last read of ADC_OVER.

1: An overrun error has occurred on the corresponding channel since the last read of ADC_OVER.

61.7.16 ADC Extended Mode Register

Name: ADC_EMR

Address: 0xFC030040

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	ADCMODE		–	SIGNMODE		TAG
23	22	21	20	19	18	17	16
–	–	SRCCLK	ASTE	–	–	OSR	
15	14	13	12	11	10	9	8
–	–	CMPFILTER		–	–	CMPALL	–
7	6	5	4	3	2	1	0
CMPSEL				–	CMPTYPE	CMPMODE	

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **CMPMODE: Comparison Mode**

Value	Name	Description
0	LOW	When the converted data is lower than the low threshold of the window, generates the COMPE flag in ADC_ISR or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode.
1	HIGH	When the converted data is higher than the high threshold of the window, generates the COMPE flag in ADC_ISR or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode.
2	IN	When the converted data is in the comparison window, generates the COMPE flag in ADC_ISR or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode.
3	OUT	When the converted data is out of the comparison window, generates the COMPE flag in ADC_ISR or, in Partial Wakeup mode, defines the conditions to exit system from Wait mode.

- **CMPTYPE: Comparison Type**

Value	Name	Description
0	FLAG_ONLY	Any conversion is performed and comparison function drives the COMPE flag.
1	START_CONDITION	Comparison conditions must be met to start the storage of all conversions until the CMPRST bit is set.

- **CMPSEL: Comparison Selected Channel**

If CMPALL = 0: CMPSEL indicates which channel has to be compared.

If CMPALL = 1: No effect.

- **CMPALL: Compare All Channels**

0: Only channel indicated in CMPSEL field is compared.

1: All channels are compared.

- **CMPFILTER: Compare Event Filtering**

Number of consecutive compare events necessary to raise the flag = CMPFILTER+1

When programmed to 0, the flag rises as soon as an event occurs.

Refer to Section 61.6.9 “Comparison Window” when using filtering option (CMPFILTER > 0).

- **OSR: Over Sampling Rate**

Value	Name	Description
0	NO_AVERAGE	No averaging. ADC sample rate is maximum.
1	OSR4	1-bit enhanced resolution by averaging. ADC sample rate divided by 4.
2	OSR16	2-bit enhanced resolution by averaging. ADC sample rate divided by 16.

- **ASTE: Averaging on Single Trigger Event**

Value	Name	Description
0	MULTI_TRIG_AVERAGE	The average requests several trigger events.
1	SINGLE_TRIG_AVERAGE	The average requests only one trigger event.

- **SRCCLK: External Clock Selection**

0 (PERIPH_CLK): The peripheral clock is the source for the ADC prescaler.

1 (GCLK): GCLK is the source clock for the ADC prescaler, thus the ADC clock can be independent of the core/peripheral clock.

- **TAG: Tag of ADC_LCDR**

0: Sets CHNB field to zero in ADC_LCDR.

1: Appends the channel number to the conversion result in ADC_LCDR.

- **SIGNMODE: Sign Mode**

Value	Name	Description
0	SE_UNSG_DF_SIGN	Single-Ended channels: Unsigned conversions. Differential channels: Signed conversions.
1	SE_SIGN_DF_UNSG	Single-Ended channels: Signed conversions. Differential channels: Unsigned conversions.
2	ALL_UNSIGNED	All channels: Unsigned conversions.
3	ALL_SIGNED	All channels: Signed conversions.

If conversion results are signed and resolution is below 16 bits, the sign is extended up to the bit 15 (for example, 0xF43 for 12-bit resolution will be read as 0xFF43 and 0x467 will be read as 0x0467). See Section 61.6.6 “Conversion Results Format”.

- **ADCMODE: ADC Running Mode**

Value	Name	Description
0	NORMAL	Normal mode of operation.
1	OFFSET_ERROR	Offset Error mode to measure the offset error. See Table 61-7.
2	GAIN_ERROR_HIGH	Gain Error mode to measure the gain error. See Table 61-7.
3	GAIN_ERROR_LOW	Gain Error mode to measure the gain error. See Table 61-7.

See Section 61.6.14 “Automatic Error Correction” for details on ADC running mode.

61.7.17 ADC Compare Window Register

Name: ADC_CWR

Address: 0xFC030044

Access: Read/Write

31	30	29	28	27	26	25	24
		HIGHTHRES					
23	22	21	20	19	18	17	16
HIGHTHRES							
15	14	13	12	11	10	9	8
		LOWTHRES					
7	6	5	4	3	2	1	0
LOWTHRES							

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **LOWTHRES: Low Threshold**

Low threshold associated to compare settings of ADC_EMR.

- **HIGHTHRES: High Threshold**

High threshold associated to compare settings of ADC_EMR.

61.7.18 ADC Channel Offset Register

Name: ADC_COR

Address: 0xFC03004C

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	DIFF11	DIFF10	DIFF9	DIFF8
23	22	21	20	19	18	17	16
DIFF7	DIFF6	DIFF5	DIFF4	DIFF3	DIFF2	DIFF1	DIFF0
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	–

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **DIFFx: Differential Inputs for Channel x**

0: Corresponding channel is set in Single-ended mode.

1: Corresponding channel is set in Differential mode.

61.7.19 ADC Channel Data Register

Name: ADC_CDRx [x=0..11]

Address: 0xFC030050

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
		DATA					
7	6	5	4	3	2	1	0
DATA							

- **DATA: Converted Data**

The analog-to-digital conversion data is placed into this register at the end of a conversion and remains until a new conversion is completed. ADC_CDRx is only loaded if the corresponding analog channel is enabled.

61.7.20 ADC Analog Control Register

Name: ADC_ACR

Address: 0xFC030094

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	–	–	IBCTL	
7	6	5	4	3	2	1	0
–	–	–	–	–	–	PENDETSENS	

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **PENDETSENS: Pen Detection Sensitivity**

Modifies the pen detection input pull-up resistor value. See section “Electrical Characteristics” for further details.

- **IBCTL: ADC Bias Current Control**

Adapts performance versus power consumption. See section “Electrical Characteristics” for further details.

61.7.21 ADC Touchscreen Mode Register

Name: ADC_TSMR

Address: 0xFC0300B0

Access: Read/Write

31	30	29	28	27	26	25	24
PENDBC				–	–	–	PENDET
23	22	21	20	19	18	17	16
–	NOTSDMA	–	–	TSSCTIM			
15	14	13	12	11	10	9	8
–	–	–	–	TSFREQ			
7	6	5	4	3	2	1	0
–	–	TSAV		–	–	TSMODE	

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **TSMODE: Touchscreen Mode**

Value	Name	Description
0	NONE	No Touchscreen
1	4_WIRE_NO_PM	4-wire Touchscreen without pressure measurement
2	4_WIRE	4-wire Touchscreen with pressure measurement
3	5_WIRE	5-wire Touchscreen

When TSMOD equals 01 or 10 (i.e., 4-wire mode), channels 0, 1, 2 and 3 must not be used for classic ADC conversions. When TSMOD equals 11 (i.e., 5-wire mode), channels 0, 1, 2, 3, and 4 must not be used.

- **TSAV: Touchscreen Average**

Value	Name	Description
0	NO_FILTER	No Filtering. Only one ADC conversion per measure
1	AVG2CONV	Averages 2 ADC conversions
2	AVG4CONV	Averages 4 ADC conversions
3	AVG8CONV	Averages 8 ADC conversions

- **TSFREQ: Touchscreen Frequency**

Defines the touchscreen frequency compared to the trigger frequency.

TSFREQ must be greater or equal to TSAV.

The touchscreen frequency is:

$$\text{Touchscreen Frequency} = \text{Trigger Frequency} / 2^{\text{TSFREQ}}$$

- **TSSCTIM: Touchscreen Switches Closure Time**

Defines closure time of analog switches necessary to establish the measurement conditions.

The closure time is:

$$\text{Switch Closure Time} = (\text{TSSCTIM} \times 4) \text{ ADCCLK periods.}$$

- **PENDET: Pen Contact Detection Enable**

0: Pen contact detection disabled.

1: Pen contact detection enabled.

When PENDET = 1, XPOS, YPOS, Z1, Z2 values of ADC_XPOSR, ADC_YPOSR, ADC_PRESSR are automatically cleared when PENS = 0 in ADC_ISR.

- **NOTSDMA: No TouchScreen DMA**

0: XPOS, YPOS, Z1, Z2 are transmitted in ADC_LCDR.

1: XPOS, YPOS, Z1, Z2 are never transmitted in ADC_LCDR, therefore the buffer does not contains touchscreen values.

- **PENDBC: Pen Detect Debouncing Period**

Debouncing period = 2^{PENDBC} ADCCLK periods.

61.7.22 ADC Touchscreen X Position Register

Name: ADC_XPOSR

Address: 0xFC0300B4

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	XSCALE			
23	22	21	20	19	18	17	16
XSCALE							
15	14	13	12	11	10	9	8
–	–	–	–	XPOS			
7	6	5	4	3	2	1	0
XPOS							

- **XPOS: X Position**

The position measured is stored here. If XPOS = 0 or XPOS = XSIZE, the pen is on the border.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), XPOS is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

- **XSCALE: Scale of XPOS**

Indicates the max value that XPOS can reach. This value should be close to 2^{12}.

61.7.23 ADC Touchscreen Y Position Register

Name: ADC_YPOSR

Address: 0xFC0300B8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	YSCALE			
23	22	21	20	19	18	17	16
YSCALE							
15	14	13	12	11	10	9	8
–	–	–	–	YPOS			
7	6	5	4	3	2	1	0
YPOS							

- **YPOS: Y Position**

The position measured is stored here. If YPOS = 0 or YPOS = YSIZE, the pen is on the border.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), YPOS is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

- **YSCALE: Scale of YPOS**

Indicates the max value that YPOS can reach. This value should be close to 2^{12}.

61.7.24 ADC Touchscreen Pressure Register

Name: ADC_PRESSR

Address: 0xFC0300BC

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	Z2			
23	22	21	20	19	18	17	16
Z2							
15	14	13	12	11	10	9	8
–	–	–	–	Z1			
7	6	5	4	3	2	1	0
Z1							

- **Z1: Data of Z1 Measurement**

Data Z1 necessary to calculate pen pressure.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), Z1 is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

- **Z2: Data of Z2 Measurement**

Data Z2 necessary to calculate pen pressure.

When pen detection is enabled (PENDET set to ‘1’ in ADC_TSMR), Z2 is tied to 0 while there is no detection of contact on the touchscreen (i.e., when PENS bit is cleared in ADC_ISR).

Note: These two values are unavailable if TSMODE is not set to 2 in ADC_TSMR.

61.7.25 ADC Trigger Register

Name: ADC_TRGR

Address: 0xFC0300C0

Access: Read/Write

31	30	29	28	27	26	25	24
TRGPER							
23	22	21	20	19	18	17	16
TRGPER							
15	14	13	12	11	10	9	8
–	–	–	–	–	–	–	–
7	6	5	4	3	2	1	0
–	–	–	–	–	TRGMOD		

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **TRGMOD: Trigger Mode**

Value	Name	Description
0	NO_TRIGGER	No trigger, only software trigger can start conversions
1	EXT_TRIG_RISE	External trigger rising edge
2	EXT_TRIG_FALL	External trigger falling edge
3	EXT_TRIG_ANY	External trigger any edge
4	PEN_TRIG	Pen Detect Trigger (shall be selected only if PENDET is set and TSAMOD = Touchscreen only mode)
5	PERIOD_TRIG	ADC internal periodic trigger (see field TRGPER)
6	CONTINUOUS	Continuous mode

- **TRGPER: Trigger Period**

Effective only if TRGMOD defines a periodic trigger.

Defines the periodic trigger period, with the following equation:

Trigger Period = (TRGPER + 1) / ADCCLK

The minimum time between two consecutive trigger events must be strictly greater than the duration time of the longest conversion sequence depending on the configuration of registers ADC_MR, ADC_CHSR, ADC_SEQRx, ADC_TSMR.

When TRGMOD is set to pen detect trigger (i.e., 100) and averaging is used (i.e., field TSAV ≠ 0 in ADC_TSMR) only one measure is performed. Thus, XRDY, YRDY, PRDY, DRDY will not rise on pen contact trigger. To achieve measurement, several triggers must be provided either by software or by setting the TRGMOD on continuous trigger (i.e., 110) until flags rise.

61.7.26 ADC Correction Values Register

Name: ADC_CVR

Address: 0xFC0300D4

Access: Read/Write

31	30	29	28	27	26	25	24
GAINCORR							
23	22	21	20	19	18	17	16
GAINCORR							
15	14	13	12	11	10	9	8
OFFSETCORR							
7	6	5	4	3	2	1	0
OFFSETCORR							

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **OFFSETCORR: Offset Correction**

Offset correction to apply on converted data. The offset is signed (2’s complement), only bits 0 to 11 are relevant (other bits are ignored and read as 0).

- **GAINCORR: Gain Correction**

Gain correction to apply on converted data. Only bits 0 to 13 are relevant (other bits are ignored and read as 0).

61.7.27 ADC Channel Error Correction Register

Name: ADC_CECR

Address: 0xFC0300D8

Access: Read/Write

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
–	–	–	–	–	–	–	–
15	14	13	12	11	10	9	8
–	–	–	–	ECORR11	ECORR10	ECORR9	ECORR8
7	6	5	4	3	2	1	0
ECORR7	ECORR6	ECORR5	ECORR4	ECORR3	ECORR2	ECORR1	ECORR0

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **ECORRx: Error Correction Enable for channel x**

0: Automatic error correction is disabled for channel x.

1: Automatic error correction is enabled for channel x.

61.7.28 ADC Touchscreen Correction Values Register

Name: ADC_TSCVR

Address: 0xFC0300DC

Access: Read/Write

31	30	29	28	27	26	25	24
TSGAINCORR							
23	22	21	20	19	18	17	16
TSGAINCORR							
15	14	13	12	11	10	9	8
TSOFFSETCORR							
7	6	5	4	3	2	1	0
TSOFFSETCORR							

This register can only be written if the WPEN bit is cleared in the ADC Write Protection Mode Register.

- **TSOFFSETCORR: Touchscreen Offset Correction**

Offset correction to apply on converted data for the touchscreen channels. The offset is signed (2's complement), only bits 0 to 11 are relevant (other bits are ignored and read as 0).

- **TSGAINCORR: Touchscreen Gain Correction**

Gain correction to apply on converted data for the touchscreen channels. Only bits 0 to 13 are relevant (other bits are ignored and read as 0).

61.7.29 ADC Write Protection Mode Register

Name: ADC_WPMR

Address: 0xFC0300E4

Access: Read/Write

31	30	29	28	27	26	25	24
WPKEY							
23	22	21	20	19	18	17	16
WPKEY							
15	14	13	12	11	10	9	8
WPKEY							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPEN

- **WPEN: Write Protection Enable**

0: Disables the write protection if WPKEY value corresponds to 0x414443 (“ADC” in ASCII).

1: Enables the write protection if WPKEY value corresponds to 0x414443 (“ADC” in ASCII).

See Section 61.6.19 “Register Write Protection” for the list of write-protected registers.

- **WPKEY: Write Protection Key**

Value	Name	Description
0x414443	PASSWD	Writing any other value in this field aborts the write operation of the WPEN bit. Always reads as 0

61.7.30 ADC Write Protection Status Register

Name: ADC_WPSR

Address: 0xFC0300E8

Access: Read-only

31	30	29	28	27	26	25	24
–	–	–	–	–	–	–	–
23	22	21	20	19	18	17	16
WPVSRC							
15	14	13	12	11	10	9	8
WPVSRC							
7	6	5	4	3	2	1	0
–	–	–	–	–	–	–	WPVS

- **WPVS: Write Protection Violation Status**

0: No write protection violation has occurred since the last read of ADC_WPSR.

1: A write protection violation has occurred since the last read of ADC_WPSR. If this violation is an unauthorized attempt to write a protected register, the associated violation is reported into field WPVSRC.

- **WPVSRC: Write Protection Violation Source**

When WPVS = 1, WPVSRC indicates the register address offset at which a write access has been attempted.

62. Electrical Characteristics

62.1 Absolute Maximum Ratings

Table 62-1. Absolute Maximum Ratings*

Storage Temperature.....-60°C to +150°C
Voltage on Input Pins with Respect to Ground..... -0.3V to +4.0V
Maximum Operating Voltage VDDCORE, VDDPLLA, VDDUTMIC and VDDHSIC.....1.5V
VDDIODDR.....2.0V
VDDBU.....4.0V
VDDIOPx, VDDUTMII, VDDISC, VDDSDMMC, VDDOSC, VDDANA, VDDAUDIOPLL.....4.0V
VDDFUSE.....3.0V
Total DC Output Current on all I/O lines.....350 mA

*NOTICE: Stresses beyond those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

62.2 DC Characteristics

The following characteristics are applicable to the operating temperature range T_A = -40°C to +105°C, unless otherwise specified.

Table 62-2. Recommended Thermal Operating Conditions

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
T_A	Operating Temperature	–	-40	–	+105	°C
T_J	Junction Temperature	–	-40	–	+125	°C
R_{thJA}	Junction-to-ambient thermal resistance	LFBGA289	–	34.8	–	°C/W
		TFBGA256	–	27.4	–	
		TFBGA196	–	44.6	–	
R_{thJC}	Junction-to-case thermal resistance	LFBGA289	–	10.9	–	°C/W
		TFBGA256	–	TBD	–	
		TFBGA196	–	11.2	–	
P_D	Power Dissipation	At T_A = 85°C, LFBGA289	–	–	1149	mW
		At T_A = 105°C, LFBGA289	–	–	575	mW
P_D	Power Dissipation	At T_A = 85°C, TFBGA256	–	–	1460	mW
		At T_A = 105°C, TFBGA256	–	–	730	mW
P_D	Power Dissipation	At T_A = 85°C, TFBGA196	–	–	897	mW
		At T_A = 105°C, TFBGA196	–	–	448	mW

Table 62-3. DC Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DDCORE}	DC Supply Core	–	1.1	1.2	1.32	V
	Allowable Voltage Ripple	rms value 10 kHz to 20 MHz	–	–	15	mV
	Slope	–	1.3	–	–	V/ms
V_{DDBU}	DC Supply I/Os, Backup	Must be established first	1.65		3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	30	mV
	Slope	–	2.4	–	–	V/ms
V_{DDANA}	DC Supply I/Os, Backup	The ADC is not functional below 2.0V	1.65		3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 20 MHz	–	–	20	mV
	Slope	–	2.4	–	–	V/ms
V_{DDIOP0}	DC Supply LCD I/Os		1.65	–	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 20 MHz	–	–	20	mV
V_{DDIOP1}	DC Supply Peripheral I/Os	Peripheral I/O Lines	1.65	–	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 20 MHz	–	–	20	mV
V_{DDIOP2}	DC Supply Peripheral I/Os	Peripheral I/O Lines	1.65	–	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 20 MHz	–	–	20	mV
V_{DDPLLA}	PLL A and Main Oscillator Supply	–	1.1	1.2	1.32	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
		rms value > 10 MHz	–	–	20	
$V_{DDUTMIC}$	DC Supply UDPHS and UHPHS UTMI+ Core	–	1.1	1.2	1.32	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
$V_{DDUTMII}$	DC Supply UDPHS and UHPHS UTMI+ Interface	–	3.0	3.3	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
V_{DDHSIC}	DC Supply HSIC Phy	–	1.1	1.2	1.3	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
$V_{DDAUDIOPLL}$	DC Supply AUDIO PLL	–	3.0	3.3	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
V_{DDFUSE}	DC Supply Fuse Box	For fuse programming only	2.25	2.5	2.75	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
$V_{DDSDMMC}$	DC Supply Peripheral I/Os	SDMMC I/Os Lines	1.65	–	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV

Table 63-3. DC Characteristics (Continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DDISC}	DC Supply Peripheral I/Os	ISC I/Os Lines	1.65	–	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	20	mV
V_{DDOSC}	DC Supply Oscillator	–	1.65	–	3.6	V
	Allowable Voltage Ripple	rms value 10 kHz to 10 MHz	–	–	15	mV
$V_{DDIODDR}$	DC Supply SDRAM I/Os	- LPDDR1-DDR2 Interface I/O lines - LPDDR2-LPDDR3 Interface I/O lines - DDR3L Interface I/O lines - DDR3 Interface I/O lines	1.7 1.14 1.283 1.425	1.8 1.2 1.35 1.5	1.95 1.30 1.45 1.575	V
V_{IL}	Low-level Input Voltage(2)	VDDIO in 3.3V range	-0.3	–	0.8	V
		VDDIO in 1.8V range	-0.3	–	0.3 x V_{DDIO}	
V_{IH}	High-level Input Voltage(2)	VDDIO in 3.3V range	2	–	V_{DDIO} +0.3	V
		VDDIO in 1.8V range	0.7 x V_{DDIO}	–	V_{DDIO} +0.3	
V_{OL}	Low-level Output Voltage	I_O MAX	–	–	0.41	V
V_{OH}	High-level Output Voltage	I_O MAX	V_{DDIO} - 0.4	–	–	V
V_{hys}	Schmitt Trigger Hysteresis	All PIO lines, VDDIOx in 3.3V range	0.34	–	–	V
		All PIO lines, VDDIOx in 1.8V range	0.2	–	–	
I_{OL}	I_{OIL}(or I_{SINK}) (VOL = 0.4V)	All GPIO_x, 1.8V: Low	-1	–	–	mA
		All GPIO_x, 1.8V: Medium	-10	–	–	
		All GPIO_x, 1.8V: High	-18	–	–	
I_{OH}	I_{OH} (or I_{SOURCE}) (VOH = VDDIO - 0.4V)	All GPIO_x, 1.8V: Low	–	–	1	mA
		All GPIO_x, 1.8V: Medium	–	–	10	
		All GPIO_x, 1.8V: High	–	–	18	
I_{OL}	I_{OIL}(or I_{SINK}) (VOL = 0.4V)	All GPIO_x, 3.3V: Low	-2	–	–	mA
		All GPIO_x, 3.3V: Medium	-20	–	–	
		All GPIO_x, 3.3V: High	-32	–	–	
I_{OH}	I_{OH} (or I_{SOURCE}) (VOH = VDDIO - 0.4V)	All GPIO_x, 3.3V: Low	–	–	2	mA
		All GPIO_x, 3.3V: Medium	–	–	20	
		All GPIO_x, 3.3V: High	–	–	32	
I_{IL}	Low-level Input Current	LCDPCK, ISI_MCK, GPIO, QSPI_SCK	0	–	0.1	µA
			10	–	70	
I_{IH}	High-level Input Current	LCDPCK, ISI_MCK, GPIO, QSPI_SCK	0	–	0.1	µA
			10	–	60	
I_{IL}	Low-level Input Current	GPIO_AD	0	–	0.1	µA
			5	–	10	
I_{IH}	High-level Input Current	GPIO_AD	0	–	0.1	µA
			5	–	10	

Table 63-3. DC Characteristics (Continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
R_{PULLUP}	Pull-up Resistor	GPIO_CLK, GPIO_IO, GPIO: 1.8V	80	143	310	kΩ
		GPIO_CLK, GPIO_IO, GPIO: 3.3V	40	66	130	
$R_{PULLDOWN}$	Pull-down Resistor	GPIO_CLK, GPIO_IO, GPIO: 1.8V	80	161	430	kΩ
		GPIO_CLK, GPIO_IO, GPIO: 3.3V	40	77	160	
R_{PULLUP}	Pull-up Resistor	GPIO_AD: 1.8V	280	380	480	kΩ
		GPIO_AD: 3.3V	280	380	480	
$R_{PULLDOWN}$	Pull-down Resistor	GPIO_AD: 1.8V	280	380	480	kΩ
		GPIO_AD: 3.3V	280	380	480	
R_{SERIAL}	Serial Resistor	GPIO	–	30	–	Ω
		GPIO_IO	–	13	–	Ω
		GPIO_CLK, GPIO_AD	–	0	–	Ω

Notes: 1. V_{DDIO} voltage must be equal to V_{DDIN} voltage.
2. Current injection may lead to performance degradation or functional failures.

Table 62-4. I/O Switching Frequency

GPIO Type	Drive	VDD			Unit
	C_{Load} = 30pF	1.8V	2.5V	3.3V	
GPIO_IO	Low	8	12	15	MHz
	Medium	60	80	90	
	High	80	110	110	
GPIO_CLK	Low	10	15	18	
	Medium	90	100	120	
	High	–	–	–	
GPIO_AD	Low	10	15	18	
	Medium	90	100	120	
	High	–	–	–	
GPIO	Low	7	10	12	
	Medium	40	50	60	
	High	50	60	70	

Table 62-5. QSPI I/O Switching Frequency

GPIO Type	Description	Name	Conditions	Min	Max	Unit
GPIO_QSPI	Maximum output frequency	f_{max}	Load = 30 pF	–	133	MHz
	Output duty cycle	–	Load = 30 pF	45	55	%

62.3 Power Consumption

This section provides information about the current consumption on different power supply rails of the device. It gives current consumption in:

- Active mode when running a CoreMark and in predefined use cases
- Low-power modes: Backup mode, Idle mode and Ultra Low-power mode
- By peripheral with representative activity

62.4 Active Mode

Active mode is the normal running mode with the ARM core clock running off a PLL. The power management controller is used to adapt the frequency and to disable the peripheral clocks.

62.4.1 Active Mode Power Consumption Versus Modes

The power consumption values are measured under the following operating conditions:

- Parts are from typical process
- V_{DDIOPx} = 3.3V
- $V_{DDSDMMC0}$ and $V_{DDSDMMC1}$ = 1.8V to 3.3V (high frequency)
- V_{DDCORE} = 1.2V +/-2%
- V_{DDBU} = 1.6V to 3.6V
- T_A = as specified in Table 62-6 and Table 62-7
- There is no consumption on the device's I/Os.
- All peripheral clocks are disabled.

Figure 62-1. Measurement Schematics

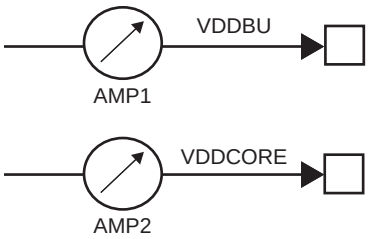


Table 62-6. Typical Peripheral Power Consumption by Peripheral in Active Mode

Peripheral	Conditions (T_A = 25°C)	Consumption on VDDCORE	Conditions	Consumption (typ)	Unit
GMAC	Peripheral Clock Enabled	–	See Note 1.	12 *MCK + 1840*DR (Data Rate in Mbits/s)	µA
XDMAC0		–	See Note 2.	17.6 * MCK + 4.92 * DR (MCK in MHz, DataRate in Mbytes/s)	
XDMAC1		–			
ICM		3.52	–	–	µA/MHz
AES		8.79			
AESB		7.49			
TDES		0.85			
SHA		3.88			
MPDDRC		45.21	See Note 3.	67 * MCK + 3.11 * DR + 1609 (MCK in MHz, DataRate in Mbytes/s)	µA
HSMC		20.12	–	–	µA/MHz
PIOA		11.39			
FLEXCOM0		-			
FLEX0_USART		5.21			
FLEX0_SPI		7.39			
FLEX0_TWI		3.88			
FLEXCOM1		See FLEXCOM0			
FLEXCOM2		See FLEXCOM0			
FLEXCOM3		See FLEXCOM0			
FLEXCOM4		See FLEXCOM0			
UART0		1.09			
UART1		0.97			
UART2		1.21			
UART3		0.85			
UART4		0.85			
TWIHS0		3.27			
TWIHS1		3.39			
SDMMC0		8.61			
SDMMC1		8.61			
SPI0		4.61			
SPI1		4.48			
TC0		3.03			
TC1		4			
PWM		7.03			
ADC		2.3			

Table 63-6. Typical Peripheral Power Consumption by Peripheral in Active Mode (Continued)

Peripheral	Conditions (T_A = 25°C)	Consumption on VDDCORE	Conditions	Consumption (typ)	Unit
UHPHS	Peripheral Clock Enabled	–	See Note 4.	12 *MCK + 4900*DR (MCK in MHz, Data Rate in Mbytes/s)	µA
UDPHS			See Note 5.	10 *MCK + 2060*DR (MCK in MHz, Data Rate in Mbytes/s)	
SSC0		1.58	–	–	µA/MHz
SSC1		1.58			
LCDC		–	See Note 6.	9.8 *MCK + 32 *Pix_CLK + 15.7*DR_Baselayer +30.1*DR_Overlayer (MCK & Pixelclock in MHz, DataRate in Mbytes/s);	µA
ISC			See Note 7.	10.9*MCK + 22.4*ISPCK + 12.38*DR_sensor (MCK&ISPCK in MHz, DataRate in Mbytes/s)	
TRNG		760	–	–	
PDMIC		8.24	–	–	µA/MHz
QSPI0		1.94			
QSPI1		1.94			
I2SC0		0.61			
I2SC1		0.61			
CAN0		9.7			
CAN1		7.27			

Notes:

1. In Linux OS, use the ‘iperf’ command to perform bidirectional data transfers. Measure GMAC consumption at different transfer speeds.
2. XDMAC is initialized and one channel performs a memory-to-memory transfer. During test, the data rate is adjusted by changing the DMA setting and the burst size.
3. DDR3 devices are initialized (fully functional). XDMAC performs a memory-to-memory transfer inside the DDR area. Total consumption of MPDDRC and XDMAC is measured. MPDDRC consumption is calculated by discounting XDMAC consumption.
4. In Linux OS, measure UHPHS consumption at different transfer speeds.
5. In Linux OS, build a mass storage using UDPHS. Measure UDPHS consumption at different transfer
6. The LCD timing engine and each display layer are switched on in sequence. The static image (using random data) is displayed under various resolutions. The 24-bpp RGB888 color space is set for all layers. Auxiliary functions such as rotation, scaling, color space conversion, color look-up table, and chroma upsampling are disabled.
7. ISC performs image sensor preview.

In order to maximize performances, each Peripheral Clock has been timed to H32MX clock frequency. The peripheral frequency can be reduced with the help of a divider in PMC_PCR.

Table 62-7. Power Consumption in Active Mode: AMP2

Conditions T_A = 25°C		Consumption			
		Dhrystone (mA)	DMIPS	CoreMark (mA)	CoreMark
PLL clock is 1000 MHz, ARM Core clock is 500 MHz, MCK is 166 MHz. – Caches L1 and L2 enabled – Code running off of internal SRAM – Code speed optimization – Run Dhrystone / CoreMark benchmark – Peripheral clock disabled	MRL-B	114.4	1.277	108.8	2.65
	MRL-A	237.2	1.277	233.7	2.65

62.5 Low-power Modes

The various low-power modes enable balancing device power consumption and wakeup time. The modes are described below, in the order of lowest to highest power consumption.

62.5.1 Backup Mode

The Backup mode allows to achieve the lowest power consumption in the system with limited functionality.

In this mode, only the backup area is powered, maintaining the RTC, the backup registers, the backup SRAM and the security module running. This mode is entered by shutting down all the power rails except the VDDBU (refer to Section 23. “Shutdown Controller (SHDWC)”). To exit Backup mode, the SHDN pin (connected to the enable of the external Power Management IC) must be driven high by an internal event (RTC) or by one of the external events listed below:

- WKUP0 to WKUP9 pins (level transition, configurable debouncing)
- Character received on a serial com receiver (RXLP)
- Analog comparison

The Backup Mode functionality has been extended with the possibility to keep the DDR memory in self-refresh state.

62.5.2 Backup Mode with DDR in Self-refresh

The Backup mode with DDR in self-refresh is used to keep the DDR contents when the system is powered off. This mode is achieved by maintaining the backup area and the VDDIODDR powered. The sequence below must be performed to enter the Backup Self-refresh mode:

- Software saves all the context information to resume (application-dependent).
- Put the DDR in Self-refresh mode and wait until the self-refresh status is OK (refer to Section 33. “Multiport DDR-SDRAM Controller (MPDDRC)”).
- Set SFRBU_DDRBUMCR.BUMEN (Section 19.3.3 “DDR BU Mode Control Register”)
- Enter the Backup mode as described above.

The method to exit this mode is the same as described for the Backup mode. Once the system is restarted, the software checks the state of the SFRBU_DDRBUMCR.BUMEN bit and reinitializes the DDR controller. The DDR memory exits Self-refresh mode when a memory access in the DDR memory space is performed.

62.5.3 Ultra Low-power (ULP) Mode

The purpose of the Ultra Low-power mode is to achieve the lowest power consumption with the system in Retention mode and able to resume on wake up events (any interrupt or hardware event). This mode is a combination of the Wait for Interrupt mode of the ARM core and the system clocks frequency reduced or shut-off.

To obtain the best results, care must be taken that the I/Os (pull-up/pull-down, etc.), USB transceivers, etc., are set to the appropriate state.

The Ultra Low-power mode features two submodes:

- ULP0 mode
- ULP1 mode

62.5.3.1 ULP0 Mode

The ULP0 mode maintains a very low frequency clock to wake up on any interrupt.

The selection of the clock depends on the current consumption target versus wake up time. The higher the frequency, the higher the power consumption.

The sequence to enter ULP0 mode is detailed below. The code used to enter this mode must be executed out of the internal SRAM.

1. Set the DDR to Self-Refresh mode.
2. Set the interrupts to wake up the system.
3. Disable all peripheral clocks.
4. Set the I/Os to an appropriate state and disable the USB transceivers (refer to section SFR).
5. Switch the system clock to Slow Clock.
6. Disable the PLLs, the main oscillator and the 12 MHz RC oscillator.
7. Enter the Wait for Interrupt mode and disable the PCK clock in the PMC_SCDR.

The wake up from ULP0 mode is triggered by any enabled interrupt. When resuming, the software reconfigures the system (oscillator, PLL, etc.) in the same state as before WFI.

62.5.3.2 ULP1 Mode

Unlike the ULP0 mode, all the clocks are off in the ULP1 mode, but the number of wake up sources is limited to the list below:

- WKUP0 pin (level transition, configurable debouncing)
- WKUP1 Secumod wakeup signal
- WKUP2 pin to WKUP9 pin (shared with PIOBU0 to PIOBU7)
- RTC alarm
- USB Resume from Suspend mode
- SDMMC card detect
- RXLP event
- ACC event
- Any SleepWalking event coming from TWI, FLEXCOMx, SPI, ADC

The sequence to enter the ULP1 mode is detailed below. The code used to enter this mode must be executed out of the internal SRAM.

1. Set the DDR to Self-Refresh mode.
2. Set the events to enable a system wakeup.
3. Disable all peripheral clocks.
4. Set the I/Os to an appropriate state and disable the USB transceivers (refer to the section “Special Function Register (SFR)”).

5. Switch the system clock to the 12 MHz RC oscillator.
6. Disable the PLLs and the main oscillator.
7. Enter the ULP1 mode by either:
    1. setting the WAITMODE bit in CKGR_MOR, or
    2. setting the LPM bit in PMC_FSMR and executing the processor WaitForEvent (WFE) instruction.
8. After setting the WAITMODE bit or using the WFE instruction, wait for the PMC_SR.MCKRDY bit to be set.

62.5.4 Idle Mode

The purpose of Idle mode is to optimize power consumption of the device versus response time. In this mode, only the core clock is stopped. The peripheral clocks, including the DDR controller clock, can be enabled. The current consumption in this mode is application-dependent and can be reduced by enabling Dynamic Clock Gating (L2CC_POWCR.DCKGATEN = 1).

This mode is entered via the Wait for Interrupt (WFI) instruction and PCK disabling.

The processor can be awakened from an interrupt. The system resumes where it was before entering WFI mode.

62.5.5 Low-power Mode Summary Table

The modes detailed above are the main low-power modes. Each part can be set to on or off separately and wakeup sources can be configured individually. Table 62-8 shows a summary of the low-power mode configurations.

Table 62-8. Low-power Mode Configuration Summary

	Low-power Mode				
	Backup		Ultra-low-power		
Submode:	–	Self-refresh	ULP0	ULP1	Idle
64 kHz RC Oscillator, 32 kHz Oscillator, RTC, Backup Memory and Registers, POR	ON				
12 MHz RC Oscillator	OFF				ON
VDDCORE Regulator	OFF		ON		
Core	OFF (Not powered)		Powered (Not clocked)		
Memory, Peripherals	OFF (Not powered)		Powered (512 Hz)	Powered (Not clocked)	Powered (Clocked)
Mode Entry	Shutdown Controller, FLEXCOM SleepWalking	DDR in Self-refresh, Shutdown Controller	DDR in Self-refresh Frequency reduced in PMC, WFI	DDR in Self-refresh, CKGR_MOR.WAITMODE = 1	DDR in Self-refresh, WFI
Potential Wakeup Sources	WKUP0 pin, any PIOBU configured as WKUP pin, RTC alarm, any level above comparator source or character received	Backup mode sources	Any interrupt	Wakeup pins, WOL	Any interrupt
Core at Wakeup	Reset		Clocked back at 512 Hz	Clocked back at 12 MHz	Clocked back at full speed
PIO State While in Low-power Mode	Reset		Previous state saved		
PIO State at Wakeup	Inputs with pull-ups	Unchanged			
Consumption[2]	I_{VDDBU}= 4.5 µA typ[3] at 25°C/3.0V	I_{VDDBU}= 4.5 µA typ[3] at 25°C/3.0V $I_{VDDIODDR}$ = 40 µA	0.21 mA at 25°C/1.1V 0.27 mA at 25°C/1.2V	0.17 mA at 25°C/1.1V 0.27 mA at 25°C/1.2V	28 mA at 25°C/1.2V[4]
Wakeup Time[1]	Startup time	Startup time	300 ms	15 µs	800 ns at 498 MHz

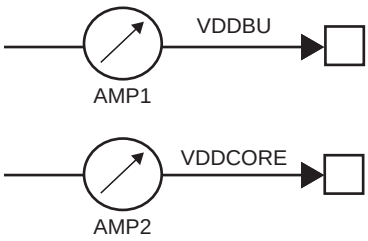
Notes:
1. When considering wakeup time, the time required to start the PLL is not taken into account. Once started, the device works with the main oscillator. The user has to add the PLL startup time if it is needed in the system. The wakeup time is defined as the time taken for wakeup until the first instruction is fetched.
2. The external loads on PIOs are not taken into account in the calculation.
3. Total current consumption.
4. Dynamic Clock Gating enabled (L2CC_POWCR.DCKGATEN = 1)

62.5.6 Low-power Consumption Versus Modes

The low-power consumption values are measured under the following operating conditions:

- Parts are from typical process
- V_{DDIOPx} = 3.3V
- $V_{DDSDMMC0}$ and $V_{DDSDMMC1}$ = 1.8V to 3.3V (high frequency)
- V_{DDCORE} = 1.2V +/-2%
- V_{DDBU} = 1.6V to 3.6V
- T_A = as specified in Table 62-9, Table 62-10, Table 62-11
- There is no consumption on the device's I/Os.
- All peripheral clocks are disabled.

Figure 62-2. Measurement Schematics



In order to maximize performances, each Peripheral Clock has been timed to H32MX clock frequency. The peripheral frequency can be reduced with the help of a divider in PMC_PCR.

Table 62-9. Typical Power Consumption in Idle Mode: AMP2

Conditions	Consumption				Unit
	T_A 25°C	T_A 70°C	T_A 85°C	T_A 105°C	
PLL clock is 1000 MHz, ARM Core clock is 500 MHz, MCK is 166 MHz. – Core clock is stopped – Peripheral clocks, including the DDR Controller clock, can be enabled – Mode is entered via Wait for Interrupt (WFI) instruction and PCK disabling – Measure IDDCORE + IDDBU – Peripheral clock disabled	28.16	29.63	30.82	33.44	mA

Table 62-10. VDDCORE Power Consumption in Ultra Low-power Mode: AMP2

Mode	Conditions	Consumption (mA)				Wakeup Time µs
		T_A 25°C	T_A 70°C	T_A 85°C	T_A 105°C	
ULP1 Fast Wakeup	ARM Core clock is disabled. MCK is 0.	0.27	1.44	2.38	4.63	15
ULP0 12 MHz	ARM Core clock is disabled. MCK is 12 MHz.	3.16	4.24	5.26	7.67	13
ULP0 750 kHz	ARM Core clock is disabled. MCK is 750 kHz.	1.55	2.62	3.68	6.06	205
ULP0 187 kHz	ARM Core clock is disabled. MCK is 187.5 kHz.	1.47	2.54	3.58	5.97	820
ULP0 32 kHz	ARM Core clock is disabled. MCK is 32 kHz.	0.28	1.52	2.59	5.03	4690
ULP0 512 Hz	ARM Core clock is disabled. MCK is 512 Hz.	0.275	1.51	2.58	5.025	300000

Table 62-11. Typical Power Consumption for Backup Mode

V_{DDBU} (V)	Conditions	Consumption (µA)				Unit
		T_A = 25°C	T_A = 70°C	T_A = 85°C	T_A = 105°C	
1.6	V_{DDBU} Only	4.2	12.1	19.3	36.8	µA
1.7		4.2	12.1	19.3	36.85	
1.8		4.25	12.1	19.35	36.85	
1.9		4.25	12.1	19.35	36.9	
2		4.3	12.1	19.4	36.95	
2.1		4.3	12.15	19.4	36.95	
2.2		4.3	12.2	19.45	37	
2.3		4.35	12.2	19.45	37	
2.4		4.35	12.2	19.5	37	
2.5		4.38	12.25	19.5	37.05	
2.6		4.4	12.25	19.55	37.1	
2.7		4.4	12.3	19.55	37.1	
2.8		4.4	12.3	19.6	37.15	
2.9		4.45	12.35	19.6	37.2	
3		4.45	12.35	19.65	37.3	
3.1		4.48	12.45	19.8	37.7	
3.2		4.55	12.8	20.25	38.3	
3.3		4.9	13.4	20.9	38.9	
3.4		5.5	14.1	21.6	39.7	
3.5		6.2	14.9	22.4	40.5	
3.6		7	15.7	23.3	41.4	

62.6 Clock Characteristics

62.6.1 Processor Clock Characteristics

Table 62-12. Processor Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
$1/(t_{CPPCK})$	Processor Clock Frequency	VDDCORE[1.1V, 1.32V] T = 85°C	250(1)	400	MHz
		VDDCORE[1.2V, 1.32V] T = 85°C	250(1)	500	

Note: 1. Limitation for DDR2 (125 MHz) usage only. There are no limitations to DDR3, DDR3L, LPDDR1, LPDDR2 and LPDDR3.

62.6.2 Master Clock Characteristics

The master clock is the maximum clock at which the system is able to run. It is given by the smallest value of the internal bus clock and EBI clock.

Table 62-13. Master Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
$1/(t_{CPMCK})$	Master Clock Frequency	VDDCORE[1.1V, 1.32V] T_A = 85°C	125(1)	133	MHz
		VDDCORE[1.2V, 1.32V], in DDR2 or LPDDR1 mode, VDDIODDR[1.8V, 1.95V], T_A = 85°C in LPDDR2 or LPDDR3 mode, VDDIODDR[1.2V, 1.30V], T_A = 85°C in DDR3 mode, VDDIODDR[1.5V, 1.575V], T_A = 85°C in DDR3L mode, VDDIODDR[1.35V, 1.45V], T_A = 85°C Security disabled	125(1)	166(2)	

Notes: 1. Limitation for DDR2 usage only. There are no limitations to DDR3, DDR3L, LPDDR1, LPDDR2 and LPDDR3.
2. The JEDEC standard specifies a maximum clock frequency of 125 MHz for DDR3 and DDR3L in DLL Off mode. However, check with memory suppliers for higher frequencies.

62.7 Oscillator Characteristics

62.7.1 Main Oscillator Characteristics

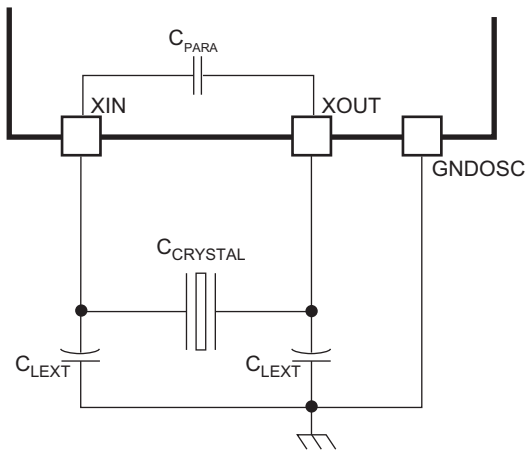
Table 62-14. 8 to 24 MHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	Operating Frequency	FREQ[2] = 00, 11 FREQ = 01 FREQ = 10	8 12 16	–	12 16 24	MHz
–	Duty Cycle	–	40	50	60	%
t_{START}	Startup Time	FREQ[2] = 00, 11 FREQ = 01 FREQ = 10	–	–	18.5 10.5 6	ms
I_{DDON}	Current Consumption (on VDDIO)	@ 12 MHz @ 24 MHz	–	1.2 1.7	3.5 4	mA
I_{DD_STDBY}	Standby Current	–	–	0.02	0.1	µA
C_{PARA}	Internal Parasitic Capacitance[1]	From XIN to XOUT	1.4	1.6	1.8	pF
P_{ON}	Drive level	FREQ = 00 FREQ = 01, 11 FREQ = 10	–	–	150 300 400	µW

Notes:
1. The external capacitors value can be determined by using the following formula:
   $C_{LEXT} = (2 \times C_{CRYSTAL}) - C_{BOARD} - (C_{PARA} \times 2)$
   where:
   C_{LEXT}: external capacitor value which must be soldered from XIN to GND and XOUT to GND
   $C_{CRYSTAL}$: crystal targeted load
   C_{BOARD}: external board parasitic capacitance (from XIN to GND or XOUT to GND)
   C_{PARA}: internal parasitic capacitance

2. The SFR_UTMICKTRIM.FREQ field defines the input frequency for the UTMI and the main oscillator. It is important to select the correct FREQ value because this has a direct influence on USB frequency.

Figure 62-3. Main Oscillator Schematics



62.7.1.1 Recommended Crystal Characteristics

The following characteristics are applicable to the operating temperature range T_A = -40°C to 85°C and to the worst case of power supply, unless otherwise specified.

Table 62-15. Recommended Crystal Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
ESR	Equivalent Series Resistance	FREQ = 00, 11	–	–	100	Ω
		FREQ = 10, 01	–	–	80	
CM	Motional Capacitance	FREQ = 00	5	–	9	fF
		FREQ = 01, 10, 11	1.3	–	3.2	
CS	Shunt Capacitance	FREQ = 00, 01, 10	–	–	3	pF
		FREQ = 11	–	–	1.3	
$C_{CRYSTAL}$	Allowed crystal capacitive load	From crystal specification FREQ = 00, 01, 11 FREQ = 10	 12.5 8	–	 18 12.5	pF

62.7.1.2 XIN Clock Characteristics

Table 62-16. XIN Clock Electrical Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$1/(t_{CPXIN})$	XIN Clock Frequency	–	–	–	50	MHz
t_{CPXIN}	XIN Clock Period	–	20	–	–	ns
t_{CHXIN}	XIN Clock High Half-period	–	0.4 x t_{CPXIN}	–	0.6 x t_{CPXIN}	ns
t_{CLXIN}	XIN Clock Low Half-period	–	0.4 x t_{CPXIN}	–	0.6 x t_{CPXIN}	ns
CIN	XIN Input Capacitance	–	–	–	25	pF
RIN	XIN Pulldown Resistor	–	–	–	500	kΩ
VIN	XIN Voltage	–	V_{DDOSC}	–	V_{DDOSC}	V

Note: These characteristics apply only when the Main Oscillator is in Bypass mode (i.e., when MOSCEN = 0 and OSCBYPASS = 1 in CKGR_MOR). See “PMC Clock Generator Main Oscillator Register” in PMC section.

62.7.2 12 MHz RC Oscillator Characteristics

Table 62-17. 12 MHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	RC Oscillator Frequency	@ 25°C	11.63	–	12.12	MHz
t_{START}	Startup Time	–	–	–	15	µs
Duty	Duty Cycle	–	45	50	55	%
I_{DDON}	Current Consumption	After startup time	–	160	350	µA

62.7.3 32.768 kHz Crystal Oscillator Characteristics

Table 62-18. 32.768 kHz Crystal Oscillator Characteristics

Symbol	Parameter	Conditions		Min	Typ	Max	Unit
f_{OSC}	Operating Frequency	Normal mode with crystal		–	32.768	–	kHz
t_{START}	Startup Time	C_m > 3fF		–	–	1200	ms
I_{DDON}	Current Consumption	ESR < 50k ohm	$C_{CRYSTAL32}$ = 12.5 pF	–	440	900	nA
			$C_{CRYSTAL32}$ = 6 pF		600	900	
		ESR < 100k ohm	$C_{CRYSTAL32}$ = 12.5 pF		800	1200	
			$C_{CRYSTAL32}$ = 6 pF		700	1200	
C_{PARA32}	Internal Parasitic Capacitance	Between XIN32 and XOUT32		1.4	1.6	1.8	pF

Figure 62-4. 32 kHz Oscillator Schematics

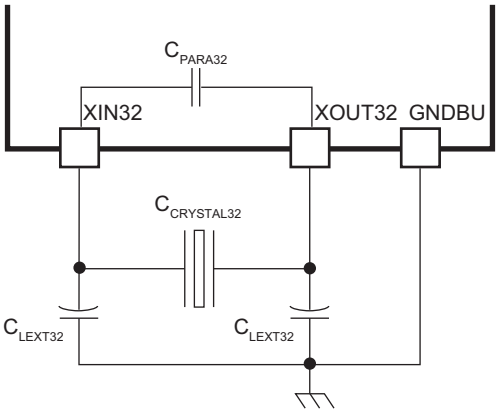


Table 62-19. Recommended 32.768 kHz Crystal Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
ESR	Equivalent Series Resistor	Crystal at 32.768 kHz	–	–	100	kΩ
–	Duty Cycle	–	40	50	60	%
C_m	Motional Capacitance	Crystal at 32.768 kHz	3	–	8	fF
C_{SHUNT}	Shunt Capacitance	Crystal at 32.768 kHz	0.6	–	2	pF
$C_{CRYSTAL32}$	Allowed Crystal Capacitance Load(1)	From crystal specification	6	–	12.5	pF
P_{ON}	Drive Level	–	–	–	0.2	µW

Note: 1. The external capacitors value can be determined by using the following formula:

$C_{LEXT32} = (2 \times C_{CRYSTAL32}) - C_{BOARD} - (C_{PARA32} \times 2)$

where:

C_{LEXT32}: external capacitor value which must be soldered from XIN32 to GND and XOUT32 to GND

$C_{CRYSTAL32}$: crystal targeted load.

C_{BOARD}: external board parasitic capacitance (from XIN to GND or XOUT to GND)

C_{PARA32}: internal parasitic capacitance

62.7.4 64 kHz RC Oscillator Characteristics

Table 62-20. 64 kHz RC Oscillator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{OSC}	RC Oscillator Frequency	–	40		90	kHz
t_{START}	Startup Time	–	11	17	30	µs
I_{DDON}	Current Consumption	After startup time	–	93	140	nA

62.8 PLL Characteristics

Table 62-21. PLLA Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{IN}	Input Frequency	–	8	–	24	MHz
f_{OUT}	Output Frequency	–	600	–	1200	MHz
I_{PLL}	Current Consumption	Active mode	–	–	14.5	mA
		Standby mode	–	–	2	µA
t_{START}	Startup Time	–	–	–	60	µs

Table 62-22. UTMI PLL Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{IN}	Input Frequency	–	12	–	24	MHz
f_{OUT}	Output Frequency	–		480		MHz
$I_{VDDUTMII}$	Current Consumption	In Active mode, on VDDUTMII, @480 MHz	–	6.3	7.0	mA
t_{START}	Startup Time	–	–	–	60	µs

Table 62-23. Audio PLL Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{IN}	Input Frequency	–	12	–	24	MHz
$f_{AUDIOCORECLK}$	AUDIOCORECLK frequency range	–	620	–	700	MHz
$f_{AUDIOPINCLK}$	AUDIOPINCLK frequency range	–	8	12.288	48	MHz
$f_{AUDIOPLLCLK}$	AUDIOPLLCLK frequency range	–	–	–	150	MHz
I_{PLL}	Current Consumption	On VDDAUDIOPLL	6	–	20	mA
t_{START}	Startup Time	From OFF to stable AUDIOCORECLK frequency	–	–	100	µs
t_{SET}	Settling Time(1)	When changing FRACR or NR in PMC_AUDIO_PLL0 or PMC_AUDIO_PLL1	–	–	100	µs

Note: 1. Loop filter is set as recommended in fields BIAS_FILTER and DCO_FILTER of PMC_AUDIO_PLL0.

62.9 USB HS Characteristics

62.9.1 Electrical Characteristics

The device conforms to all voltage, power, and timing characteristics and specifications set forth in the USB 2.0 Specification. Refer to the USB 2.0 Specification for more information.

62.9.2 Dynamic Power Consumption

Table 62-24. USB Transceiver Dynamic Power Consumption

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{BIAS}	Bias Generator Current Consumption	–	–	0.7	0.8	mA
$I_{VDDUTMII}$	HS Transceiver Current Consumption	HS transmission	–	47	60	mA
	HS Transceiver Current Consumption	HS reception	–	18	27	mA
	LS / FS Transceiver Current Consumption	FS transmission 0m cable(1)	–	4	6	mA
	LS / FS Transceiver Current Consumption	FS transmission 5m cable(1)	–	26	30	mA
	LS / FS Transceiver Current Consumption	FS reception(1)	–	3	4.5	mA
$I_{VDDUTMIC}$	Core	–	–	5.5	9	mA

Note: 1. Including 1 mA due to pull-up/pull-down current consumption.

62.10 ADC Characteristics

Electrical data are in accordance with an operating temperature range from -40°C to +85°C unless otherwise specified.

ADVREF is the positive reference of the ADC.

62.10.1 ADC Power Supply

62.10.1.1 Power Supply Characteristics

Table 62-25. Power Supply Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
I_{VDDIN}	Analog Current Consumption	Sleep mode(1) Fast Wakeup mode Normal mode, single sampling	-	2 0.4 2.2	4 0.6 3.0	µA mA mA
$I_{VDDCORE}$	Digital Current Consumption	Sleep mode(1) Normal mode	-	1 80	2 100	µA µA

Note: 1. In Sleep mode, the ADC core, the Sample and Hold and the internal reference operational amplifier are off.

62.10.2 External Reference Voltage

V_{ADVREF} is an external reference voltage applied on the pin ADVREF. The quality of the reference voltage V_{ADVREF} is critical to the performance of the ADC. A DC variation of the reference voltage V_{ADVREF} is converted to a gain error by the ADC. The noise generated by V_{ADVREF} is converted by the ADC to count noise.

Table 62-26. ADVREF Electrical Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{ADVREF}	Voltage Range	Full operational	2	–	VDDANA	V
	RMS Noise	Bandwidth 10 kHz to 1 MHz	–	–	100	µV
R_{ADVREF}	Input DC Impedance	ADC reference resistance bridge(1)	6	8	10	kΩ
I_{ADVREF}	Current	V_{ADVREF} = 3.3V	-	-	460	µA

Note: 1. When the ADC is off, the ADVREF impedance has a minimum of 1 MΩ.

62.10.3 ADC Timings

Table 62-27. ADC Timing Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
f_{ADC_Clock}	Clock Frequency	–	0.2	–	20	MHz
f_S	Sampling Frequency(1)	–	–	–	1	MHz
t_{START}	ADC Startup Time	Sleep mode to Normal mode Fast Wakeup mode to Normal mode	–	–	4 2	µs

Note: 1. $t_{ADC_}$Clock = $1/f_{ADC_Clock}$
ADC conversion time = 21 t_{ADC_Clock}
The Tracking time of the ADC has a minimal value of $t_{TRACKTIM}$ = 15 t_{ADC_Clock}.

62.10.4 ADC Transfer Function

The DATA code in ADC_CDR is up to 12-bit positive integer or two's complement (signed integer).

62.10.4.1 Differential Mode (12-bit mode)

A differential input voltage $V_{IN} = V_{INP} - V_{INN}$ can be applied between two selected differential pins, e.g. ADC0_AD0 and ADC0_AD1. The ideal code C_i is calculated by using the following formula and rounding the result to the nearest positive integer.

$$C_i = \frac{2047}{V_{ADVREF}} \times V_{IN}$$

For the other resolution defined by RES, the code C_i is extended to the corresponding resolution.

Table 62-28 is a computation example for the above formula, where V_{ADVREF} = 3V.

Table 62-28. Input Voltage Values in Differential Mode, Non-signed Output

Signed C_i	V_{IN}
-2048	-3
0	0
2047	3

62.10.4.2 Single-ended Mode (12-bit mode)

A single input voltage V_{IN} can be applied to selected pins, e.g., ADC0_AD0 or ADC0_AD1. The ideal code C_i is calculated using the following formula and rounding the result to the nearest positive integer.

The single-ended ideal code conversion formula is:

$$C_i = \frac{4095}{V_{ADVREF}} \times V_{IN}$$

For the other resolution defined by RES, the code C_i is extended to the corresponding resolution.

Table 62-29 is a computation example for the above formula, where V_{ADVREF} = 3V:

Table 62-29. Input Voltage Values in Single-ended Mode

Non-signed C_i	V_{IN}
0	0
2047	1.5
4095	3

62.10.4.3 Example of LSB Computation

The LSB is relative to the analog scale V_{ADVREF}.

The term LSB expresses the quantization step in volts, also used for one ADC code variation.

- Single-ended (SE) (ex: V_{ADVREF} = 3.0V)
  - Gain = 1, LSB = (3.0V / 4096) = 732 µV
- Differential (DIFF) (ex: V_{ADVREF} = 3.0V)
  - Gain = 0.5, LSB = (6.0V / 4096) = 1465 µV

The data include the ADC performances, as the PGA and ADC core cannot be separated. The temperature and voltage dependencies are given as separate parameters.

62.10.4.4 Gain and Offset Errors

For:

- a given gain error: E_G (%)
- a given ideal code (C_i)
- a given offset error: E_O (LSB of 12 bits)

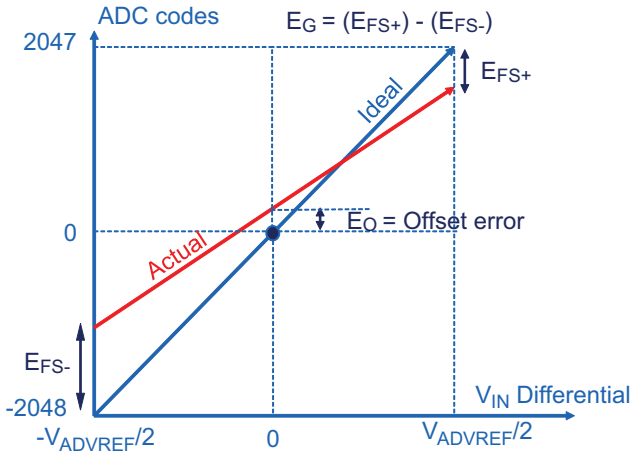
in 12-bit mode, the actual code (C_A) is calculated using the following formula

$$C_A = \left(1 + \frac{E_G}{100}\right) \times (C_i - 2047) + 2047 + E_O$$

Differential Mode

In Differential mode, the offset is defined when the differential input voltage is zero.

Figure 62-5. Gain and Offset Errors in Differential Mode



where:

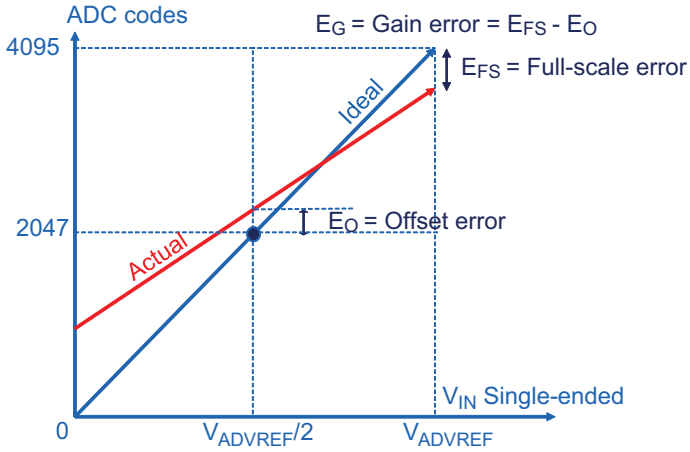
- Full-scale error $E_{FS} = (E_{FS+}) - (E_{FS-})$, unit is LSB code
- Offset error E_O is the offset error measured for V_{IN} = 0V
- Gain error $E_G = 100 \times E_{FS} / 4096$, unit in %

The error values in Table 62-31 include the sample-and-hold error as well as the PGA gain error.

Single-ended Mode

Figure 62-6 illustrates the ADC output code relative to an input voltage V_{IN} between 0V (Ground) and V_{ADVREF}. The ADC is configured in Single-ended mode by connecting internally the negative differential input to $V_{ADVREF} / 2$. As the ADC continues to work internally in Differential mode, the offset is measured at $V_{ADVREF} / 2$. The offset at V_{INP} = 0 can be computed using the transfer function and the corresponding E_G and E_O.

Figure 62-6. Gain and Offset Errors in Single-ended Mode



where:

- Full-scale error $E_{FS} = (E_{FS+}) - (E_{FS-})$, unit is LSB² code
- Offset error E_O is the offset error measured for V_{INP} = 0V
- Gain error $E_G = 100 \times E_{FS} / 2048$, unit in %

The error values in Table 62-31 include the DAC, the sample-and-hold error as well as the PGA gain error.

62.10.5 ADC Electrical Characteristics

Table 62-30. ADC INL and DNL, V_{ADVREF} = 3.3V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Differential Mode						
INL	Integral Non-Linearity	–	1	–	1	LSB
DNL	Differential Non-Linearity	–	1	–	1	LSB
Single-Ended Mode						
INL	Integral Non-Linearity	–	1.5	–	1.5	LSB
DNL	Differential Non-Linearity	–	1	–	1	LSB

Table 62-31. ADC Offset and Gain Error, V_{ADVREF} = 3.3V

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Differential Mode						
E_O	Differential Offset Error	–	-2.0	–	2.0	LSB
E_G	Differential Gain Error	–	-0.2	–	0.2	%
Single-Ended Mode						
E_O	Single-ended Offset Error	–	-2.0	–	2.0	LSB
E_G	Single-ended Gain Error	–	-0.2	–	0.2	%

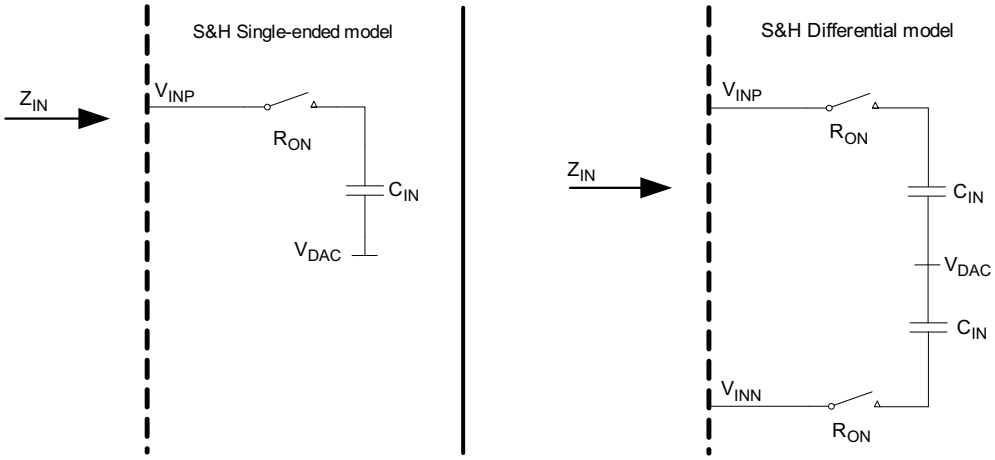
Table 62-32. ADC Analog Input Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{FS}	Analog Input Full Scale Range(1)	ADC_COR.DIFFx = 0	0	–	V_{ADVREF}	V
		ADC_COR.DIFFx = 1	$-V_{ADVREF}$	–	V_{ADVREF}	
V_{INCM}	Common Mode input range(2)	ADC_COR.DIFFx = 1	$0.4 \times V_{VDDANA}$	–	$0.6 \times V_{VDDANA}$	V
C_{IN}	ADC sampling capacitance		–	–	3	pF
C_{P_ADx}	Analog input parasitic capacitance(4)	ADx pin configured as analog input	–	–	10	
Z_{IN}	Common Mode Input impedance(3)	On ADx pin	$1 / (f_S \times C_{P_ADx})$	–	–	Ω

Notes:
1. $V_{FS} = V_{ADx}$ in Single-ended mode, $V_{FS} = (V_{ADx} - V_{ADx+1})$ in Differential mode
2. $V_{INCM} = (V_{ADx} + V_{ADx+1}) / 2$
3. See Figure 63-7. When converting one single channel, most of the input parasitic capacitance in not switched, therefore the common mode input impedance reduces to $Z_{IN} = 1 / (f_S \times C_{IN})$
4. Includes C_{IN}

62.10.6 ADC Channel Input Impedance

Figure 62-7. Input Channel Model



where:

- Z_{IN} is the input impedance in Single-ended or Differential mode
- C_{IN} = 2 pF ±20% depending on the gain value and mode (SE or DIFF); temperature dependency is negligible
- R_{ON} is typical 2 kΩ and 8 kΩ max (worst case process and high temperature)

The following formula is used to calculate input impedance:

$$Z_{IN} = \frac{1}{f_S \times C_{IN}}$$

where:

- f_S is the sampling frequency of the ADC channel
- Typ values are used to compute ADC input impedance Z_{IN}

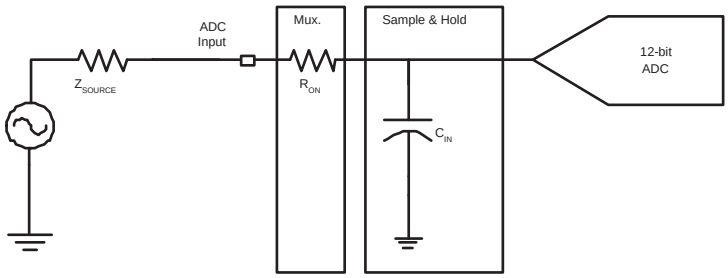
Table 62-33. Z_{IN} Input Impedance

f_S (MHz)	1	0.5	0.25	0.125	0.0625	0.03125	0.015625	0.007813
C_{IN} = 2 pF								
Z_{IN} (MΩ)	0.5	1	2	4	8	16	32	64

Track and Hold Time versus Source Output Impedance

Figure 62-8 shows a simplified acquisition path.

Figure 62-8. Simplified Acquisition Path



During the tracking phase, the ADC tracks the input signal during the tracking time shown below:

$$t_{TRACK} = n \times C_{IN} \times (R_{ON} + Z_{SOURCE}) / 1000$$

where

- Tracking time expressed in ns and Z_{SOURCE} expressed in Ω
- n = 8 for 12-bit accuracy
- R_{ON} = 2 kΩ

Table 62-34. Number of Tau:n

Resolution (bits)	12
RES	0
n	8

The ADC already includes a tracking time of 15 $t_{ADC\ Clock}$.

62.11 Analog Comparator Characteristics

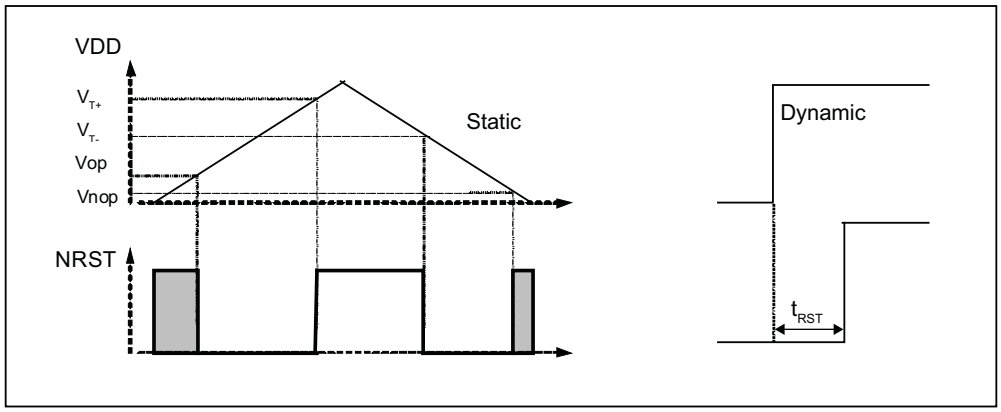
Table 62-35. Analog Comparator Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{DDBU}	Power Supply Voltage Range (VDDBU)	Analog Comparator is supplied by VDDIN	1.62	3.3	3.6	V
V_{COMPx}	Input Voltage Range	On COMPP or COMPN input	0	–	VDDBU	V
V_{hys}	Hysteresis	–	35	–	70	mV
TPD	Propagation Delay	100mV Overdrive	–	–	350	µs
f_{in}	COMPx Input Signal Frequency	Common mode and differential	–	–	1	kHz
I_{VDDBU}	Current Consumption (VDDBU)	OFF Mode (ACC_MR.ACEN = 0)	–		50	nA
		ON Mode (ACC_MR.ACEN = 1)	–	100	200	
t_{START}	Startup Time	–	–	–	300	µs

62.12 POR Characteristics

Figure 62-9 provides a general presentation of Power-On-Reset (POR) characteristics.

Figure 62-9. General Presentation of POR Behavior



When a very slow (versus t_{RST}) supply rising slope is applied on the POR VDD pin, the reset time becomes negligible and the reset signal is released when VDD raises higher than V_{T+}.

When a very fast (versus t_{RST}) supply rising slope is applied on the POR VDD pin, the voltage threshold becomes negligible and the reset signal is released after t_{RST}. It is the smallest possible reset time.

Table 62-36. VDDBU Power-On Reset Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{T+}	Threshold Voltage Rising	–	1.3	–	1.5	V
V_{T-}	Threshold Voltage Falling	–	1.22	–	1.4	V
V_{hys}	Hysteresis Voltage	–	50	–	160	mV
t_{RST}	Reset Timeout Period	–	890	–	5100	µs

Table 62-37. VDDCORE Power-On Reset Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{T+}	Threshold Voltage Rising	–	0.927	–	1.075	V
V_{T-}	Threshold Voltage Falling	–	0.848	–	1.025	V
V_{hys}	Hysteresis Voltage	–	38	–	109	mV
t_{RST}	Reset Timeout Period	–	150	–	650	µs

Table 62-38. VDDANA Power-On Reset Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V_{T+}	Threshold Voltage Rising	–	1.3	–	1.5	V
V_{T-}	Threshold Voltage Falling	–	1.22	–	1.4	V
V_{hys}	Hysteresis Voltage	–	50	–	160	mV
t_{RST}	Reset Timeout Period	–	130	–	650	µs

62.13 SMC Timings

62.13.1 Timing Conditions

SMC timings are given in max corners.

Timings assuming a capacitance load on data, control and address pads are given in Table 62-39.

Table 62-39. Capacitance Load

Supply	Corner	
	Max	Min
3.3V	50 pF	5 pF
1.8V	30 pF	5 pF

In the tables that follow, t_{CPMCK} is the MCK period.

62.13.2 SMC IOSET1 Timing Extraction

62.13.2.1 SMC IOSET1 Read Timings

Table 62-40. SMC IOSET1 Read Signals - NRD Controlled (READ_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (nrd hold = 0)						
SMC_1	Data Setup before NRD High	16.4	15	–	–	ns
SMC_2	Data Hold after NRD High	0	0	–	–	ns
HOLD SETTINGS (nrd hold ≠ 0)						
SMC_3	Data Setup before NRD High	14.4	13	–	–	ns
SMC_4	Data Hold after NRD High	0	0	–	–	ns
HOLD or NO HOLD SETTINGS (nrd hold ≠ 0, nrd hold =0)						
SMC_5	NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 Valid before NRD High	(nrd setup + nrd pulse) × t_{CPMCK}	(nrd setup + nrd pulse) × t_{CPMCK}	–	–	ns
SMC_6	NCS low before NRD High	(nrd setup + nrd pulse - ncs rd setup) × t_{CPMCK}	(nrd setup + nrd pulse - ncs rd setup) × t_{CPMCK}	–	–	ns
SMC_7	NRD Pulse Width	nrd pulse × t_{CPMCK}	nrd pulse × t_{CPMCK}	–	–	ns

Table 62-41. SMC IOSET1 Read Signals - NCS Controlled (READ_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (ncs rd hold = 0)						
SMC_8	Data Setup before NCS High	17.9	15.7	–	–	ns
SMC_9	Data Hold after NCS High	0	0	–	–	ns
HOLD SETTINGS (ncs rd hold ≠ 0)						
SMC_{10}	Data Setup before NCS High	15.9	13.7	–	–	ns
SMC_{11}	Data Hold after NCS High	0	0	–	–	ns
HOLD or NO HOLD SETTINGS (ncs rd hold ≠ 0, ncs rd hold = 0)						
SMC_{12}	NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS High	(ncs rd setup + ncs rd pulse) × t_{CPMCK}	(ncs rd setup + ncs rd pulse) × t_{CPMCK}	–	–	ns
SMC_{13}	NRD low before NCS High	(ncs rd setup + ncs rd pulse - nrd setup) × t_{CPMCK}	(ncs rd setup + ncs rd pulse - nrd setup) × t_{CPMCK}	–	–	ns
SMC_{14}	NCS Pulse Width	ncs rd pulse length × t_{CPMCK}	ncs rd pulse length × t_{CPMCK}	–	–	ns

62.13.2.2 SMC IOSET1 Write Timings

Table 62-42. SMC IOSET1 Write Signals - NWE Controlled (WRITE_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
HOLD or NO HOLD SETTINGS (nwe hold ≠ 0, nwe hold = 0)						
SMC_{15}	Data Out Valid before NWE High	nwe pulse × t_{CPMCK}	nwe pulse × t_{CPMCK}	–	–	ns
SMC_{16}	NWE Pulse Width	nwe pulse × t_{CPMCK}	nwe pulse × t_{CPMCK}	–	–	ns
SMC_{17}	NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NWE low	nwe setup × t_{CPMCK}	nwe pulse × t_{CPMCK}	–	–	ns
SMC_{18}	NCS low before NWE high	(nwe setup - ncs rd setup + nwe pulse) × t_{CPMCK}	(nwe setup - ncs rd setup + nwe pulse) × t_{CPMCK}	–	–	ns
HOLD SETTINGS (nwe hold ≠ 0)						
SMC_{19}	NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 change	nwe hold × t_{CPMCK}	nwe hold × t_{CPMCK}	–	–	ns
SMC_{20}	NWE High to NCS Inactive(1)	(nwe hold - ncs wr hold) × t_{CPMCK}	(nwe hold - ncs wr hold) × t_{CPMCK}	–	–	ns
NO HOLD SETTINGS (nwe hold = 0)						
SMC_{21}	NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25, NCS change(1)	2.3	1.3	–	–	ns

Note: 1. hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “ncs wr hold length” or “NWE hold length”.

Table 62-43. SMC IOSET1 Write NCS Controlled (WRITE_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
SMC_{22}	Data Out Valid before NCS High	ncs wr pulse × t_{CPMCK}	ncs wr pulse × t_{CPMCK}	–	–	ns
SMC_{23}	NCS Pulse Width	SMC_{14}	SMC_{14}	–	–	ns
SMC_{24}	NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS low	ncs wr setup × t_{CPMCK}	ncs wr setup × t_{CPMCK}	–	–	ns
SMC_{25}	NWE low before NCS high	(ncs wr setup - nwe setup + ncs pulse) × t_{CPMCK}	(ncs wr setup - nwe setup + ncs pulse) × t_{CPMCK}	–	–	ns
SMC_{26}	NCS High to Data Out, NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25, change	ncs wr hold × t_{CPMCK}	ncs wr hold × t_{CPMCK}	–	–	ns
SMC_{27}	NCS High to NWE Inactive	(ncs wr hold - nwe hold) × t_{CPMCK}	(ncs wr hold - nwe hold) × t_{CPMCK}	–	–	ns

62.13.3 SMC IOSET2 Timing Extraction

62.13.3.1 SMC IOSET2 Read Timings

Table 62-44. SMC IOSET2 Read Signals - NRD Controlled (READ_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (nrd hold = 0)						
SMC_1	Data Setup before NRD High	16.5	15	–	–	ns
SMC_2	Data Hold after NRD High	0	0	–	–	ns
HOLD SETTINGS (nrd hold ≠ 0)						
SMC_3	Data Setup before NRD High	14	12.6	–	–	ns
SMC_4	Data Hold after NRD High	0	0	–	–	ns
HOLD or NO HOLD SETTINGS (nrd hold ≠ 0, nrd hold =0)						
SMC_5	NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 Valid before NRD High	(nrd setup + nrd pulse) × t_{CPMCK}	(nrd setup + nrd pulse) × t_{CPMCK}	–	–	ns
SMC_6	NCS low before NRD High	(nrd setup + nrd pulse - ncs rd setup) × t_{CPMCK}	(nrd setup + nrd pulse - ncs rd setup) × t_{CPMCK}	–	–	ns
SMC_7	NRD Pulse Width	nrd pulse × t_{CPMCK}	nrd pulse × t_{CPMCK}	–	–	ns

Table 62-45. SMC IOSET2 Read Signals - NCS Controlled (READ_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
NO HOLD SETTINGS (ncs rd hold = 0)						
SMC_8	Data Setup before NCS High	18.1	15.7	–	–	ns
SMC_9	Data Hold after NCS High	0	0	–	–	ns
HOLD SETTINGS (ncs rd hold ≠ 0)						
SMC_{10}	Data Setup before NCS High	15.7	13.3	–	–	ns
SMC_{11}	Data Hold after NCS High	0	0	–	–	ns
HOLD or NO HOLD SETTINGS (ncs rd hold ≠ 0, ncs rd hold = 0)						
SMC_{12}	NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS High	(ncs rd setup + ncs rd pulse) × t_{CPMCK}	(ncs rd setup + ncs rd pulse) × t_{CPMCK}	–	–	ns
SMC_{13}	NRD low before NCS High	(ncs rd setup + ncs rd pulse - nrd setup) × t_{CPMCK}	(ncs rd setup + ncs rd pulse - nrd setup) × t_{CPMCK}	–	–	ns
SMC_{14}	NCS Pulse Width	ncs rd pulse length × t_{CPMCK}	ncs rd pulse length × t_{CPMCK}	–	–	ns

62.13.3.2 SMC IOSET2 Write Timings

Table 62-46. SMC IOSET2 Write Signals - NWE Controlled (WRITE_MODE = 1)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
HOLD or NO HOLD SETTINGS (nwe hold ≠ 0, nwe hold = 0)						
SMC_{15}	Data Out Valid before NWE High	nwe pulse × t_{CPMCK}	nwe pulse × t_{CPMCK}	–	–	ns
SMC_{16}	NWE Pulse Width	nwe pulse × t_{CPMCK}	nwe pulse × t_{CPMCK}	–	–	ns
SMC_{17}	NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NWE low	nwe setup × t_{CPMCK}	nwe pulse × t_{CPMCK}	–	–	ns
SMC_{18}	NCS low before NWE high	(nwe setup - ncs rd setup + nwe pulse) × t_{CPMCK}	(nwe setup - ncs rd setup + nwe pulse) × t_{CPMCK}	–	–	ns
HOLD SETTINGS (nwe hold ≠ 0)						
SMC_{19}	NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 change	nwe hold × t_{CPMCK}	nwe hold × t_{CPMCK}	–	–	ns
SMC_{20}	NWE High to NCS Inactive(1)	(nwe hold - ncs wr hold) × t_{CPMCK}	(nwe hold - ncs wr hold) × t_{CPMCK}	–	–	ns
NO HOLD SETTINGS (nwe hold = 0)						
SMC_{21}	NWE High to Data OUT, NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25, NCS change((1)	1.2	0.6	–	–	ns

Note: 1. hold length = total cycle duration - setup duration - pulse duration. “hold length” is for “ncs wr hold length” or “NWE hold length”.

Table 62-47. SMC IOSET2 Write NCS Controlled (WRITE_MODE = 0)

Symbol	Parameter	Min		Max		Unit
	Power supply	1.8V	3.3V	1.8V	3.3V	
SMC_{22}	Data Out Valid before NCS High	ncs wr pulse × t_{CPMCK}	ncs wr pulse × t_{CPMCK}	–	–	ns
SMC_{23}	NCS Pulse Width	SMC_{14}	SMC_{14}	–	–	ns
SMC_{24}	NBS0/A0 NBS1, NBS2/A1, NBS3, A2–A25 valid before NCS low	ncs wr setup × t_{CPMCK}	ncs wr setup × t_{CPMCK}	–	–	ns
SMC_{25}	NWE low before NCS high	(ncs wr setup - nwe setup + ncs pulse) × t_{CPMCK}	(ncs wr setup - nwe setup + ncs pulse) × t_{CPMCK}	–	–	ns
SMC_{26}	NCS High to Data Out, NBS0/A0, NBS1, NBS2/A1, NBS3, A2–A25, change	ncs wr hold × t_{CPMCK}	ncs wr hold × t_{CPMCK}	–	–	ns
SMC_{27}	NCS High to NWE Inactive	(ncs wr hold - nwe hold) × t_{CPMCK}	(ncs wr hold - nwe hold) × t_{CPMCK}	–	–	ns

Figure 62-10. SMC Timings - NCS Controlled Read and Write

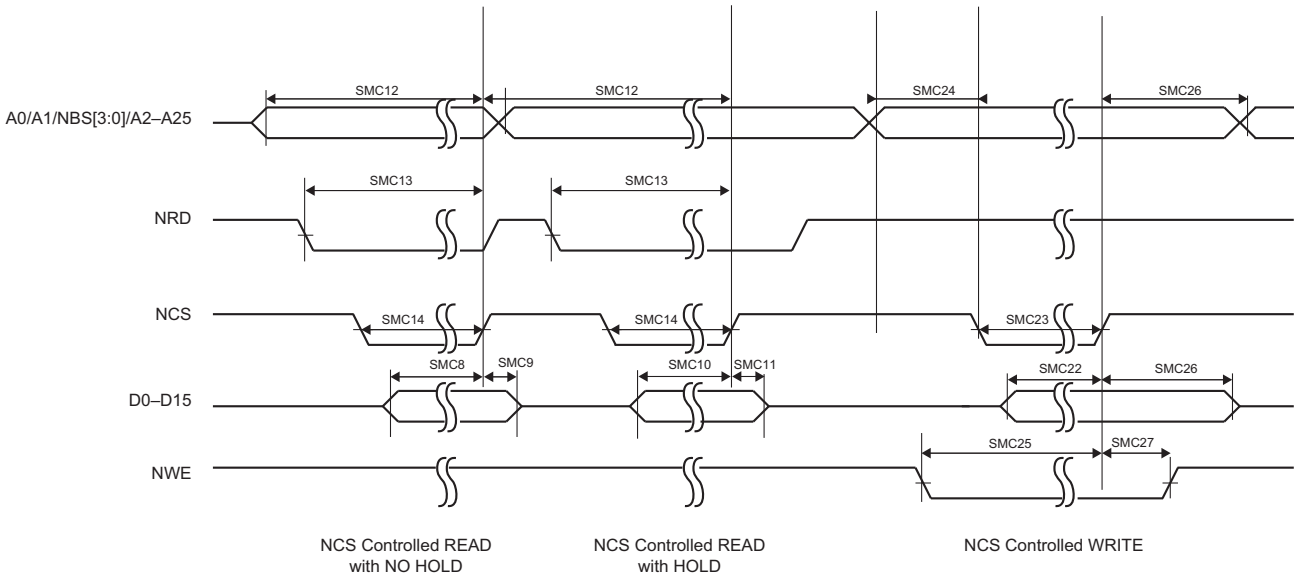
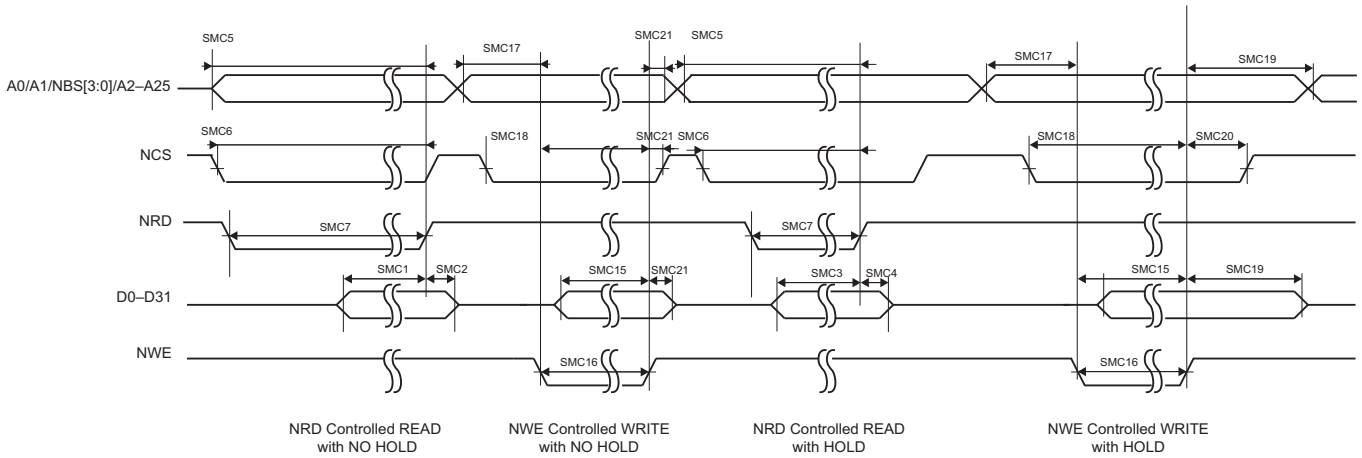


Figure 62-11. SMC Timings - NRD Controlled Read and NWE Controlled Write



62.14 SPI Timings

62.14.1 Maximum SPI Frequency

The following formulas give maximum SPI frequency in Master Read and Write modes and in Slave Read and Write modes.

Master Write Mode

The SPI is only sending data to a slave device such as an LCD, for example. The limit is given by SPI_2 (or SPI_5) timing.

Master Read Mode

$$f_{SPCK}Max = \frac{1}{SPI_0(or\ SPI_3) + t_{valid}}$$

t_{valid} is the slave time response to output data after deleting an SPCK edge. For a non-volatile memory with t_{valid} (or t_v) = 6 ns, f_{SPCK}max = 60 MHz at V_{DDIO} = 3.3V.

Slave Read Mode

In Slave mode, SPCK is the input clock for the SPI. The max SPCK frequency is given by setup and hold timings SPI_7/SPI_8 (or SPI_{10}/SPI_{11}). Since this gives a frequency well above the pad limit, the limit in Slave Read mode is given by SPCK pad.

Slave Write Mode

$$f_{SPCK}Max = \frac{1}{SPI_6(or\ SPI_9) + t_{setup}}$$

t_{setup} is the setup time from the master before sampling data (6 ns).

This gives f_{SPCK}Max = 60 MHz @ V_{DDIO} = 3.3V.

62.14.2 Timing Conditions

Timings assuming a capacitance load on MISO, SPCK and MOSI are given in Table 62-48.

Table 62-48. Capacitance Load for MISO, SPCK and MOSI (SPI0 and SPI1)

Supply	Corner	
	Max	Min
3.3V	40 pF	5 pF
1.8V	20 pF	5 pF

62.14.3 Timing Extraction

In Figure 62-13 “SPI Master Modes 1 and 2” and Figure 62-14 “SPI Master Modes 0 and 3” below, the MOSI line shifting edge is represented with a hold time = 0. However, it is important to note that for this device, the MISO line is sampled prior to the MOSI line shifting edge. As shown in Figure 62-12 “MISO Capture in Master Mode”, the device sampling point extends the propagation delay (t_p) for slave and routing delays to more than half the SPI clock period, whereas the common sampling point allows only less than half the SPI clock period.

As an example, an SPI Slave working in Mode 0 is safely driven if the SPI Master is configured in Mode 0.

Figure 62-12. MISO Capture in Master Mode

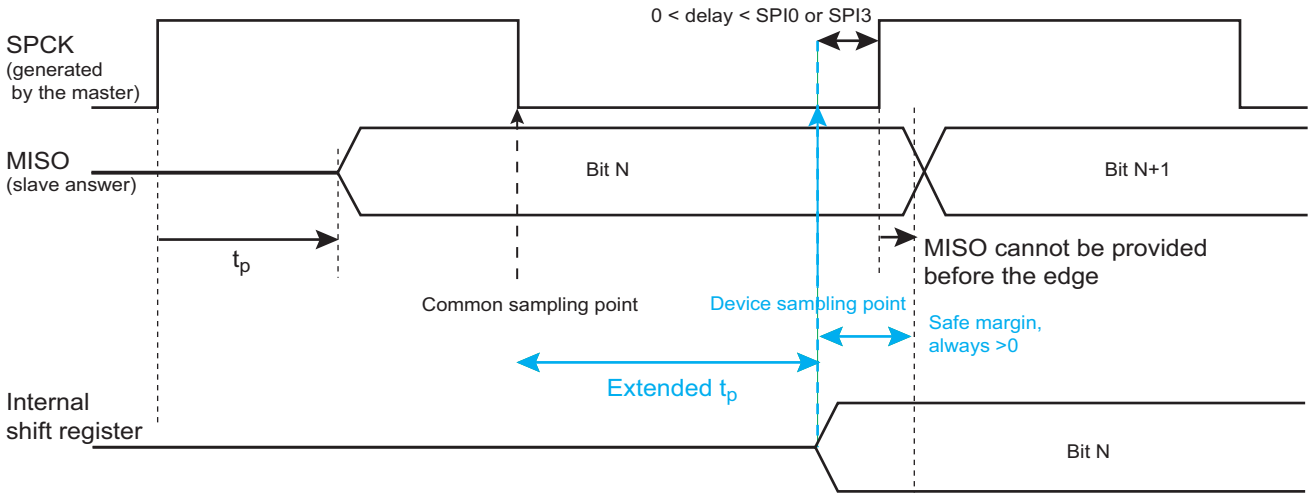


Figure 62-13. SPI Master Modes 1 and 2

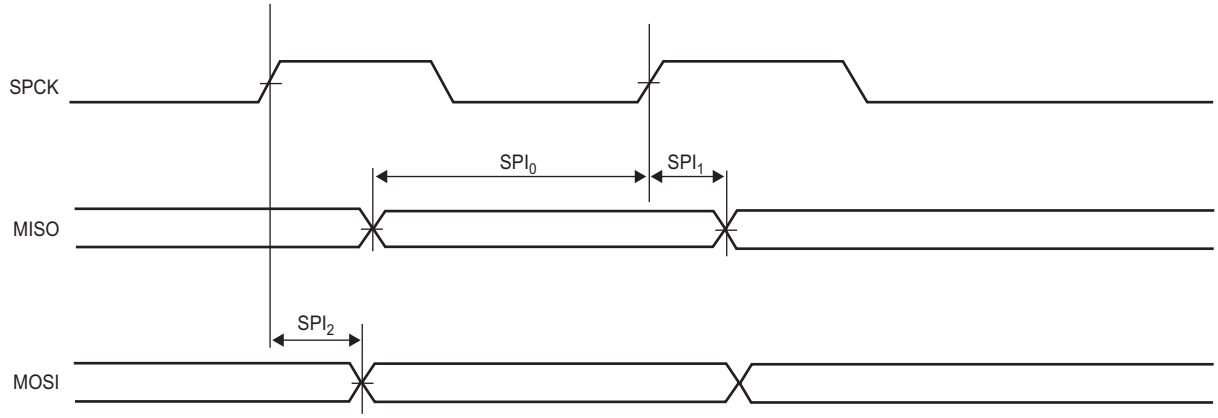


Figure 62-14. SPI Master Modes 0 and 3

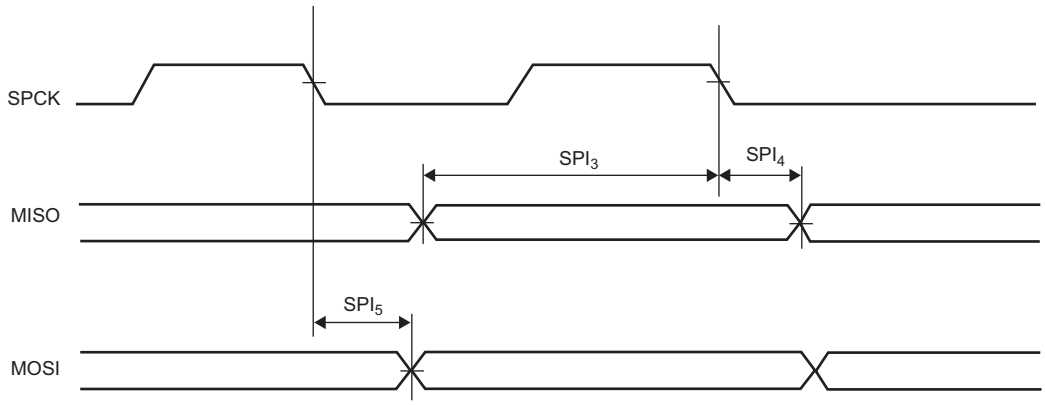


Figure 62-15. SPI Slave Modes 0 and 3

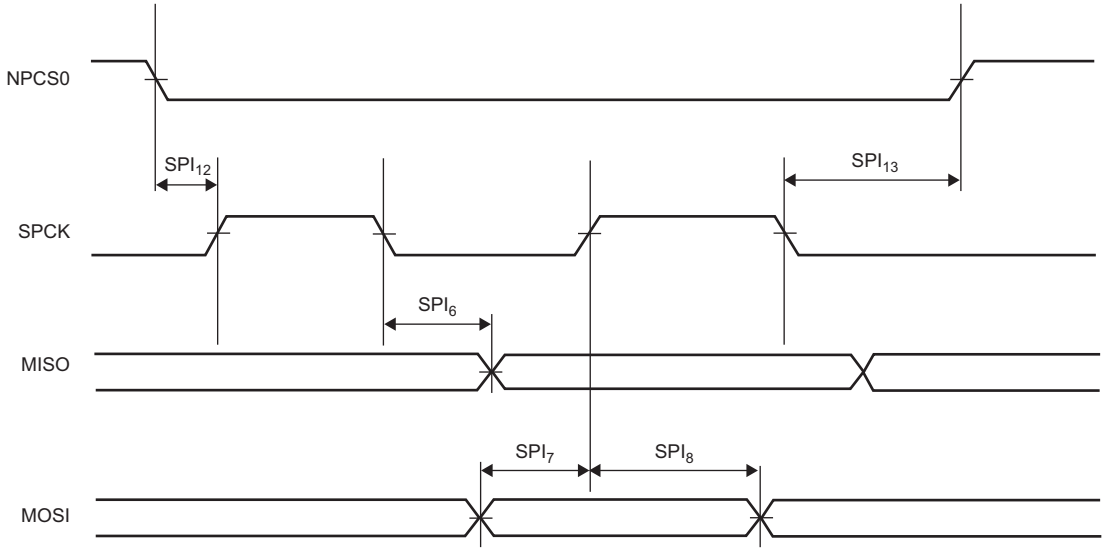


Figure 62-16. SPI Slave Modes 1 and 2

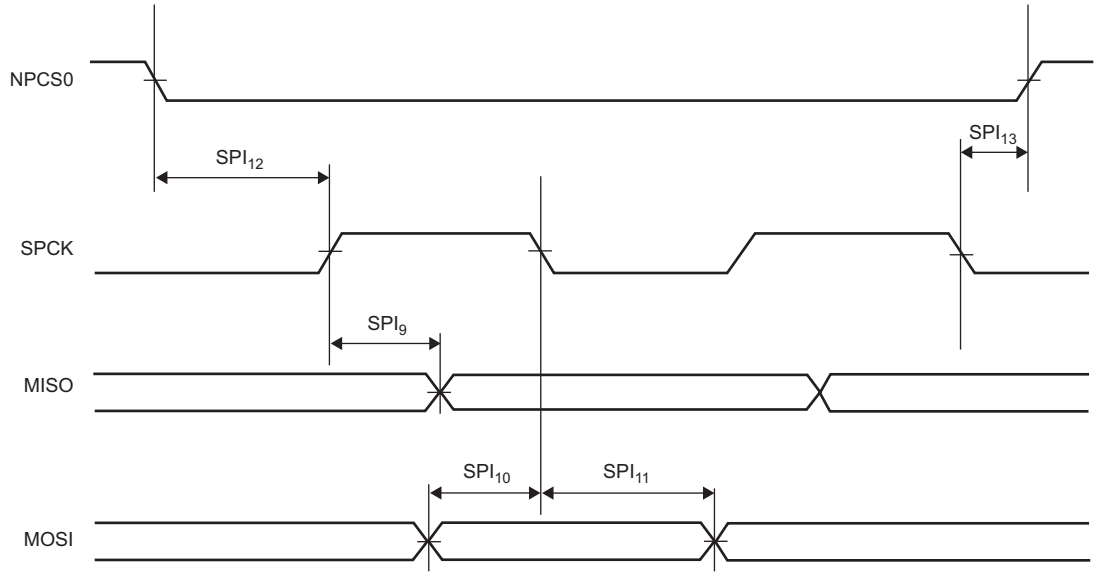


Figure 62-17. SPI Slave Mode - NPCS Timings

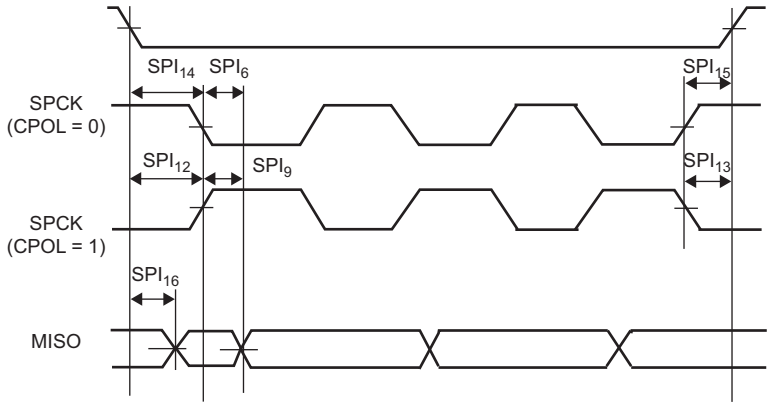


Table 62-49. SPI0 IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.3	–	12.4	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	1.9	0	2.4	ns
SPI_3	MISO Setup time before SPCK falls	13.8	–	12.6	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	1.2	0	2.3	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.5	12.6	8.4	10.9	ns
SPI_7	MOSI Setup time before SPCK rises	1.5	–	1.4	–	ns
SPI_8	MOSI Hold time after SPCK rises	1.7	–	1.5	–	ns
SPI_9	SPCK rising to MISO(1)	10	12	8	10.2	ns
SPI_{10}	MOSI Setup time before SPCK falls	1.5	–	1.4	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	1.7	–	1.5	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4.4	–	4.3	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1.5	–	1.3	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	3.9	–	3.9	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.8	–	0.5	–	ns
SPI_{16}	NPCS0 falling to MISO valid	13.3	–	11.7	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-18 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-50. SPI0 IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.5	–	13	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	2.5	0	3	ns
SPI_3	MISO Setup time before SPCK falls	14.9	–	13.6	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	2.7	0	3.4	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.3	12.	8.5	11.2	ns
SPI_7	MOSI Setup time before SPCK rises	2.3	–	2.2	–	ns
SPI_8	MOSI Hold time after SPCK rises	1	–	0.9	–	ns
SPI_9	SPCK rising to MISO(1)	9.9	12	8.1	10.5	ns
SPI_{10}	MOSI Setup time before SPCK falls	2.3	–	2.2	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	1	–	0.9	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	6.1	–	5.9	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.8	–	0.7	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	5.6	–	5.6	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.2	–	0.1	–	ns
SPI_{16}	NPCS0 falling to MISO valid	15.1	–	13.8	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-18 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-51. SPI1 IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.6	–	13.1	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	0.8	0	1.2	ns
SPI_3	MISO Setup time before SPCK falls	15	–	13.7	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	0.9	0	1.6	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.3	12.4	8.3	11.2	ns
SPI_7	MOSI Setup time before SPCK rises	3.5	–	3.4	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.8	–	0.7	–	ns
SPI_9	SPCK rising to MISO(1)	9.8	11.8	7.8	10.4	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.5	–	3.4	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.8	–	0.7	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4.9	–	4.8	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1.1	–	0.9	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	4.4	–	4.4	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.5	–	0.3	–	ns
SPI_{16}	NPCS0 falling to MISO valid	14.7	–	13.4	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-18 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-52. SPI1 IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	15.5	–	13.6	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	1.2	0	1.7	ns
SPI_3	MISO Setup time before SPCK falls	14.9	–	13.7	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	0.5	0	1.6	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.7	12.9	8.6	11.1	ns
SPI_7	MOSI Setup time before SPCK rises	3	–	2.9	–	ns
SPI_8	MOSI Hold time after SPCK rises	1	–	0.9	–	ns
SPI_9	SPCK rising to MISO(1)	10.3	12.4	8.2	10.5	ns
SPI_{10}	MOSI Setup time before SPCK falls	3	–	2.9	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	1	–	0.9	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4.4	–	4.3	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1	–	0.9	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	4	–	4	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.4	–	0.3	–	ns
SPI_{16}	NPCS0 falling to MISO valid	14.5	–	12.9	–	ns

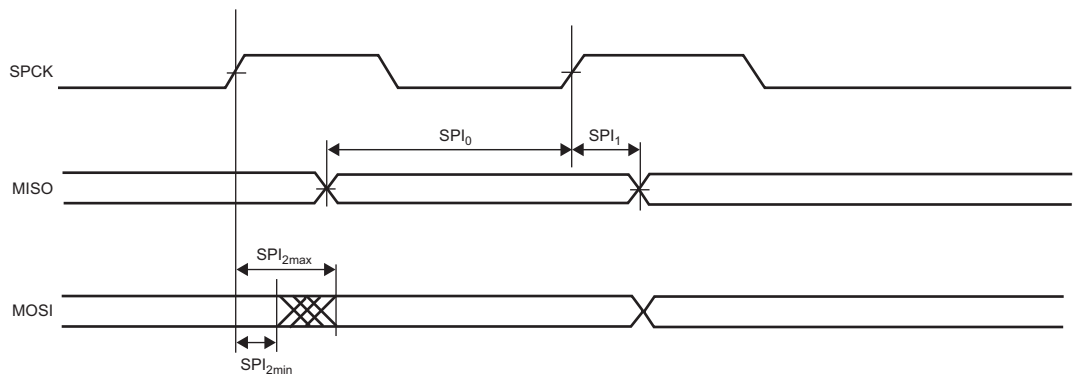
Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-18 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-53. SPI1 IOSET3 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	13.7	–	11.7	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	3.1	0	2.9	ns
SPI_3	MISO Setup time before SPCK falls	14.1	–	12.4	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	3.1	0	3.3	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	9.5	11.4	7.5	9.6	ns
SPI_7	MOSI Setup time before SPCK rises	4.5	–	4.4	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.7	–	0.5	–	ns
SPI_9	SPCK rising to MISO(1)	9.1	11	7.1	9.8	ns
SPI_{10}	MOSI Setup time before SPCK falls	4.5	–	4.4	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.7	–	0.5	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	5.1	–	4.9	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.9	–	0.8	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	4.7	–	4.6	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.3	–	0.2	–	ns
SPI_{16}	NPCS0 falling to MISO valid	13.9	–	12.2	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-18 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Figure 62-18. Minimum and Maximum Access Time for SPI Output Signal



62.15 FLEXCOM Timings

62.15.1 Timing Conditions

Timings assuming a capacitance load on MISO, SPCK and MOSI are given in Table 62-48.

Table 62-54. Capacitance Load for MISO, SPCK and MOSI (FLEXCOM 0, 1, 2, 3, 4)

Supply	Corner	
	Max	Min
3.3V	40 pF	5 pF
1.8V	20 pF	5 pF

62.15.2 Timing Extraction

Figure 62-19. FLEXCOM in SPI Master Modes 1 and 2

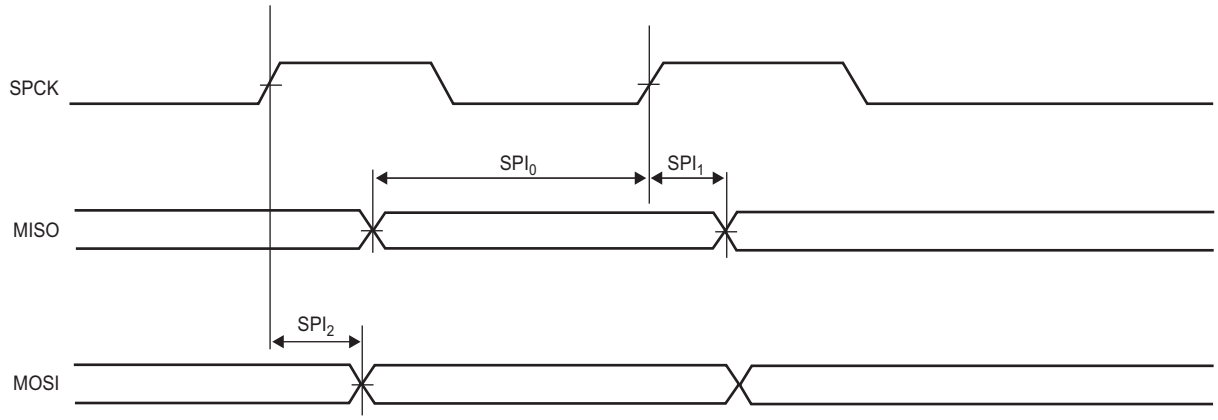


Figure 62-20. FLEXCOM in SPI Master Modes 0 and 3

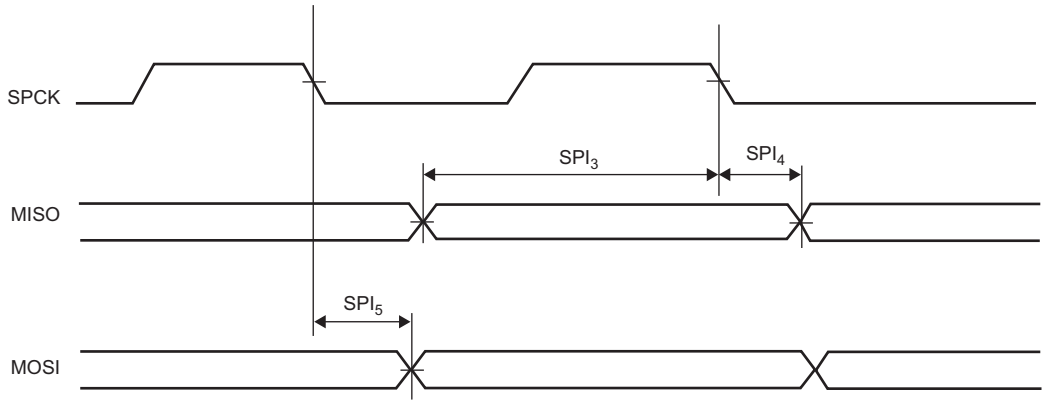


Figure 62-21. FLEXCOM in SPI Slave Modes 0 and 3

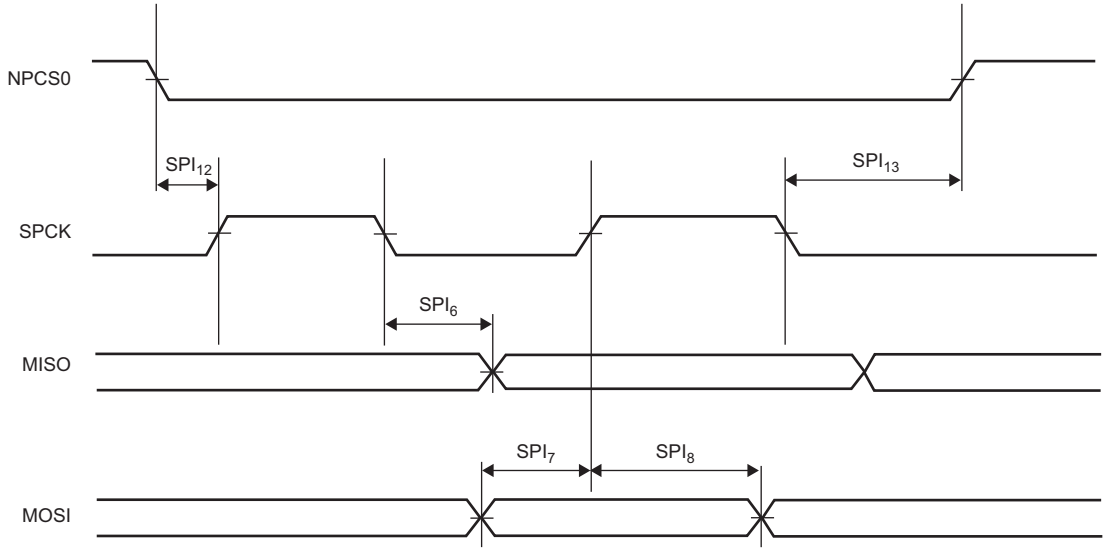


Figure 62-22. FLEXCOM in SPI Slave Modes 1 and 2

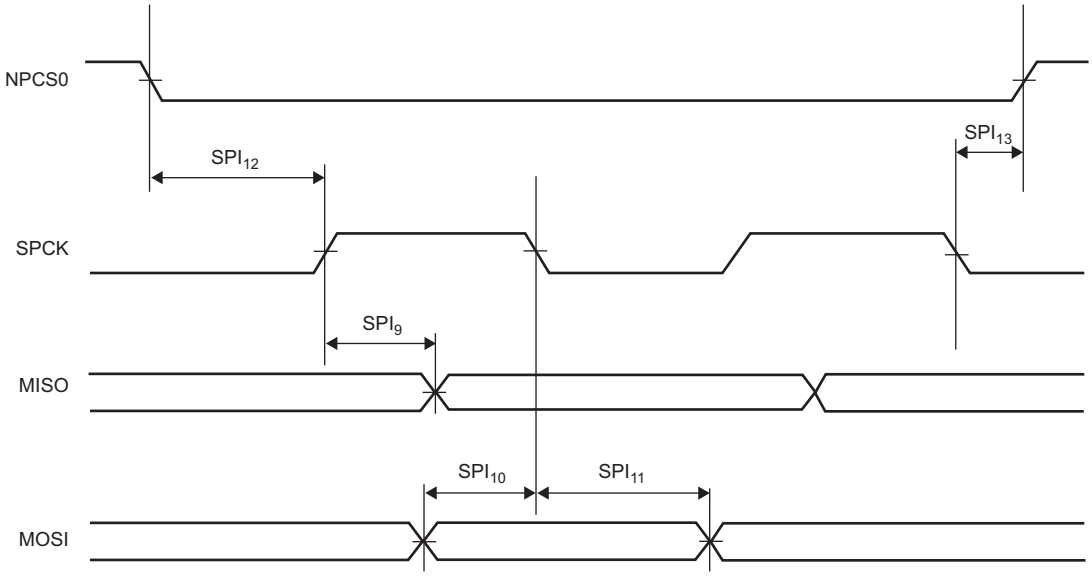


Figure 62-23. FLEXCOM in SPI Slave Mode - NPCS Timings

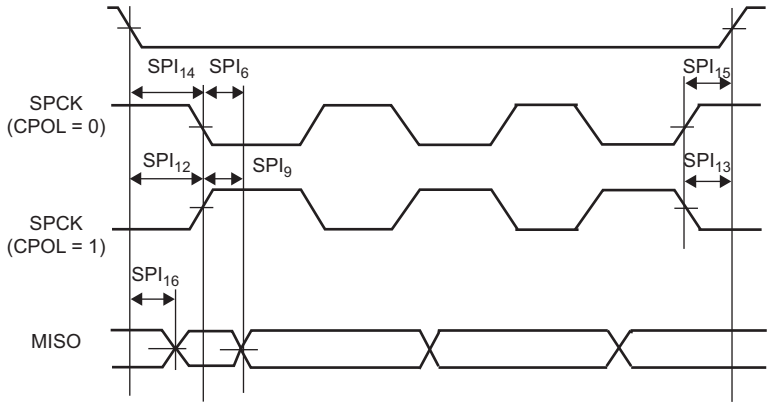


Table 62-55. FLEXCOM0 in SPI Mode IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Master Mode						
SPI_0	MISO Setup time before SPCK rises	14.3	–	12.8	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	3.9	0	4.2	ns
SPI_3	MISO Setup time before SPCK falls	14.7	–	13.4	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	3.9	0	4.6	ns
Slave Mode						
SPI_6	SPCK falling to MISO(1)	10.9	13.4	9.1	11.9	ns
SPI_7	MOSI Setup time before SPCK rises	5.3	–	5	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.4	–	0.3	–	ns
SPI_9	SPCK rising to MISO(1)	10.7	13.1	9	11.5	ns
SPI_{10}	MOSI Setup time before SPCK falls	5.3	–	5	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.4	–	0.3	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	5.6	–	5.4	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.7	–	0.6	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	5.4	–	5.2	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.2	–	0.1	–	ns
SPI_{16}	NPCS0 falling to MISO valid	17.5	–	16.2	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-56. FLEXCOM1 in SPI Mode IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	15.7	–	13.8	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	2.5	0	2.6	ns
SPI_3	MISO Setup time before SPCK falls	15.2	–	13.9	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	1.7	0	2.4	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	11.4	13.7	9.4	12.3	ns
SPI_7	MOSI Setup time before SPCK rises	2.4	–	2.1	–	ns
SPI_8	MOSI Hold time after SPCK rises	1	–	0.9	–	ns
SPI_9	SPCK rising to MISO(1)	11	13.1	9	11.6	ns
SPI_{10}	MOSI Setup time before SPCK falls	2.4	–	2.1	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	1	–	0.9	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	3.9	–	3.7	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1.1	–	1	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	3.4	–	3.3	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.6	–	0.4	–	ns
SPI_{16}	NPCS0 falling to MISO valid	16.8	–	15.5	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-57. FLEXCOM2 in SPI Mode IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	15.3	–	13.4	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	2.5	0	3	ns
SPI_3	MISO Setup time before SPCK falls	15.6	–	13.7	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	2.6	0	3.1	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	11.7	13.5	9.4	11.7	ns
SPI_7	MOSI Setup time before SPCK rises	2	–	1.6	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.5	–	0.5	–	ns
SPI_9	SPCK rising to MISO(1)	11.7	13.5	9.4	11.5	ns
SPI_{10}	MOSI Setup time before SPCK falls	2	–	1.6	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.5	–	0.5	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4	–	3.7	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.6	–	0.6	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	3.9	–	3.7	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.4	–	0.3	–	ns
SPI_{16}	NPCS0 falling to MISO valid	16.8	–	13.6	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-58. FLEXCOM2 in SPI Mode IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	13.3	–	11.2	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	4.4	0	4.1	ns
SPI_3	MISO Setup time before SPCK falls	4.3	–	12.5	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0.3	4.4	0.4	4.4	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.1	12.2	8.2	10.4	ns
SPI_7	MOSI Setup time before SPCK rises	3.3	–	3.1	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.8	–	0.7	–	ns
SPI_9	SPCK rising to MISO(1)	9.8	11.8	7.9	9.8	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.3	–	3.1	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.8	–	0.7	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	5.3	–	5.2	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.8	–	0.6	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	5	–	4.9	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.2	–	0.1	–	ns
SPI_{16}	NPCS0 falling to MISO valid	15.9	–	14.2	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-59. FLEXCOM3 in SPI Mode IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Master Mode						
SPI_0	MISO Setup time before SPCK rises	14.6	–	12.7	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	2.6	0	2.9	ns
SPI_3	MISO Setup time before SPCK falls	14.2	–	13	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	1.8	0	2.9	ns
Slave Mode						
SPI_6	SPCK falling to MISO(1)	11.6	14.1	9.5	12.7	ns
SPI_7	MOSI Setup time before SPCK rises	3.3	–	3.2	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.9	–	0.7	–	ns
SPI_9	SPCK rising to MISO(1)	11.2	13.6	9.1	12.2	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.3	–	3.2	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.9	–	0.7	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	2.8	–	2.6	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1.5	–	1.3	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	2.3	–	2.2	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	1	–	0.7	–	ns
SPI_{16}	NPCS0 falling to MISO valid	15.4	–	14.1	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-60. FLEXCOM3 in SPI Mode IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.1	–	12.6	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	3.8	0	4.1	ns
SPI_3	MISO Setup time before SPCK falls	14.9	–	13.7	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	3.9	0	4.5	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.6	13.1	8.6	11.8	ns
SPI_7	MOSI Setup time before SPCK rises	3.7	–	3.5	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.6	–	0.5	–	ns
SPI_9	SPCK rising to MISO(1)	10.2	12.6	8.2	11.1	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.7	–	3.5	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.6	–	0.5	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4.5	–	4.3	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1	–	0.9	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	4	–	3.9	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.4	–	0.3	–	ns
SPI_{16}	NPCS0 falling to MISO valid	16.9	–	15.6	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-61. FLEXCOM3 in SPI Mode IOSET3 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Master Mode						
SPI_0	MISO Setup time before SPCK rises	14.2	–	12.7	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	3.4	0	3.7	ns
SPI_3	MISO Setup time before SPCK falls	15.1	–	13.8	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)(1)	0	3.5	0	4.2	ns
Slave Mode						
SPI_6	SPCK falling to MISO(1)	11.4	14.1	9.4	12.8	ns
SPI_7	MOSI Setup time before SPCK rises	5.4	–	5.1	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.4	–	0.3	–	ns
SPI_9	SPCK rising to MISO(1)	11	13.6	9	12.2	ns
SPI_{10}	MOSI Setup time before SPCK falls	5.4	–	5.1	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.4	–	0.3	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4.3	–	4.2	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1.1	–	0.9	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	3.8	–	3.7	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.5	–	0.3	–	ns
SPI_{16}	NPCS0 falling to MISO valid	17.5	–	16.2	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-62. FLEXCOM4 in SPI Mode IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.7	–	13.1	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	2.7	0	3.2	ns
SPI_3	MISO Setup time before SPCK falls	15.2	–	13.8	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	2.9	0	3.6	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.9	13.2	8.9	11.9	ns
SPI_7	MOSI Setup time before SPCK rises	3.3	–	3.2	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.7	–	0.6	–	ns
SPI_9	SPCK rising to MISO(1)	10.5	12.7	8.5	11.3	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.3	–	3.2	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.7	–	0.6	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	5.7	–	5.5	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.6	–	0.5	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	5.2	–	5	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0	–	0	–	ns
SPI_{16}	NPCS0 falling to MISO valid	17.8	–	16.5	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-63. FLEXCOM4 in SPI Mode IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.7	–	11.5	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	2.7	0	4.7	ns
SPI_3	MISO Setup time before SPCK falls	15.2	–	12.8	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0	2.9	0.8	5.1	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	10.9	13.2	7.8	10.2	ns
SPI_7	MOSI Setup time before SPCK rises	3.3	–	2.2	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.7	–	0.8	–	ns
SPI_9	SPCK rising to MISO(1)	10.5	12.7	7.7	9.9	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.3	–	2.2	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.7	–	0.8	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	5.6	–	3.3	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	0.6	–	1.1	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	5.2	–	3.1	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0	–	0.8	–	ns
SPI_{16}	NPCS0 falling to MISO valid	17.8	–	12.9	–	ns

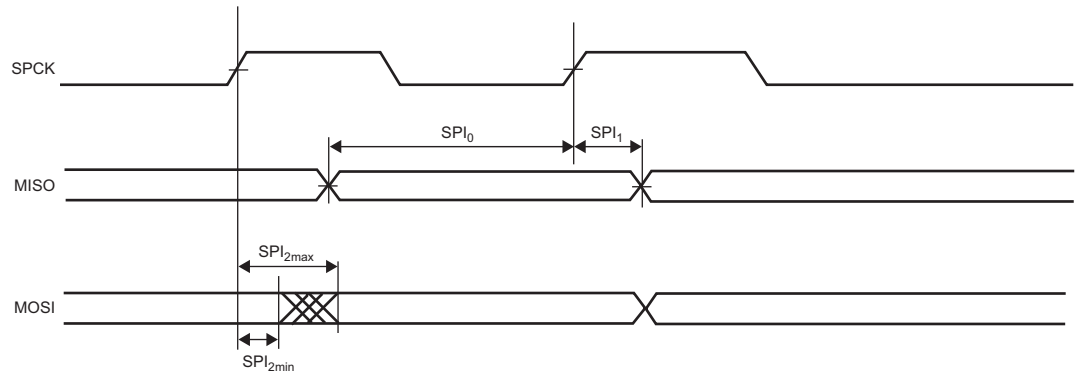
Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Table 62-64. FLEXCOM4 in SPI Mode IOSET3 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master Mode					
SPI_0	MISO Setup time before SPCK rises	14.2	–	12.2	–	ns
SPI_1	MISO Hold time after SPCK rises	0	–	0	–	ns
SPI_2	SPCK rising to MOSI(1)	0	3.6	0	3.6	ns
SPI_3	MISO Setup time before SPCK falls	15.1	–	13.3	–	ns
SPI_4	MISO Hold time after SPCK falls	0	–	0	–	ns
SPI_5	SPCK falling to MOSI(1)	0.1	3.7	0.1	4	ns
	Slave Mode					
SPI_6	SPCK falling to MISO(1)	9.9	12.4	8	10.5	ns
SPI_7	MOSI Setup time before SPCK rises	3.9	–	3.7	–	ns
SPI_8	MOSI Hold time after SPCK rises	0.8	–	0.7	–	ns
SPI_9	SPCK rising to MISO(1)	9.5	11.9	7.7	9.9	ns
SPI_{10}	MOSI Setup time before SPCK falls	3.9	–	3.7	–	ns
SPI_{11}	MOSI Hold time after SPCK falls	0.8	–	0.7	–	ns
SPI_{12}	NPCS0 setup to SPCK rising	4.8	–	4.6	–	ns
SPI_{13}	NPCS0 hold after SPCK falling	1	–	0.9	–	ns
SPI_{14}	NPCS0 setup to SPCK falling	4.4	–	4.3	–	ns
SPI_{15}	NPCS0 hold after SPCK rising	0.5	–	0.3	–	ns
SPI_{16}	NPCS0 falling to MISO valid	16	–	14.2	–	ns

Note: 1. For output signals, the minimum and maximum access times must be extracted. The minimum access time is the time between the SPCK rising or falling edge and the signal change. The maximum access time is the time between the SPCK rising or falling edge and the signal stabilization. Figure 62-24 illustrates the minimum and maximum accesses for SPI_2. The same applies for SPI_5, SPI_6 and SPI_9.

Figure 62-24. Minimum and Maximum Access Time for SPI Output Signal



62.16 QSPI Timings

62.16.1 Maximum QSPI Frequency

The following formulas give maximum QSPI frequency in Master read and write modes.

Master Write Mode

The QSPI sends data to a slave device only, e.g. an LCD. The limit is given by $QSPI_2$ (or $QSPI_5$) timing.

Master Read Mode

$$f_{SCK}\text{max} = \frac{1}{QSPI_0(\text{or } QSPI_3) + t_{VALID}}$$

t_{VALID} is the slave time response to output data after detecting a QSCK edge.

For a non-volatile memory with t_{VALID} (or t_v) = 12 ns, f_{SCK}max = 67 MHz at V_{DDIO} = 3.3V.

62.16.2 Timing Conditions

Timings assuming a capacitance load are given in Table 62-65.

Table 62-65. Capacitance Load (QSPI 0 and QSPI1)

Supply	Corner	
	Max	Min
3.3V	30 pF	5 pF
1.8V	20 pF	5 pF

62.16.3 Timing Extraction

Figure 62-25. QSPI Master Mode 0

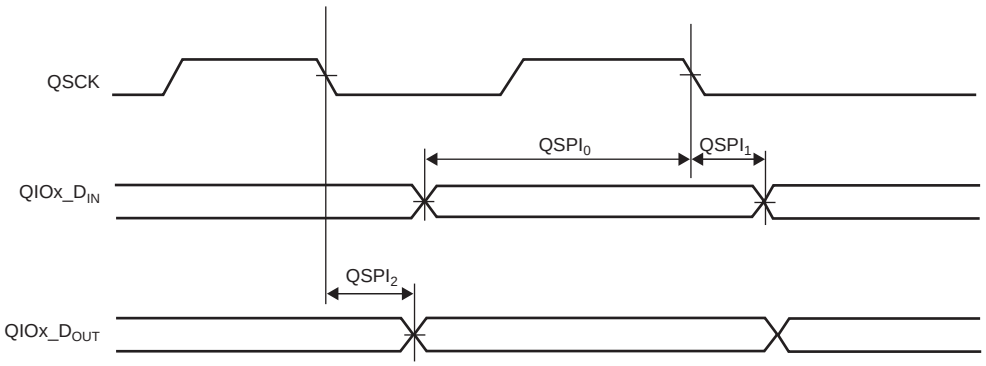


Figure 62-26. QSPI Master Mode 1

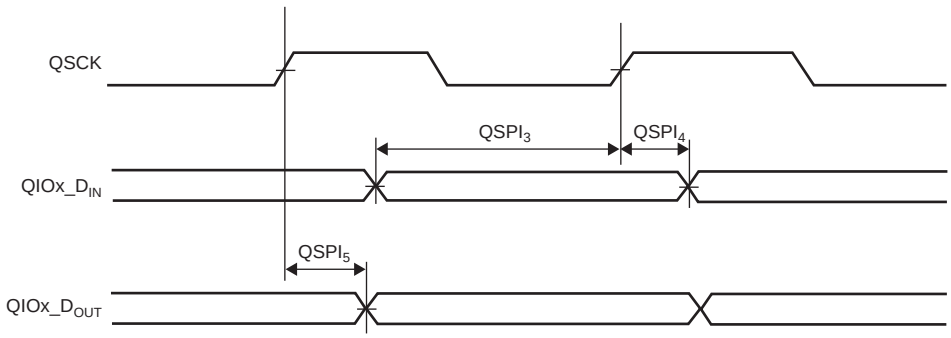


Figure 62-27. QSPI Master Mode 2

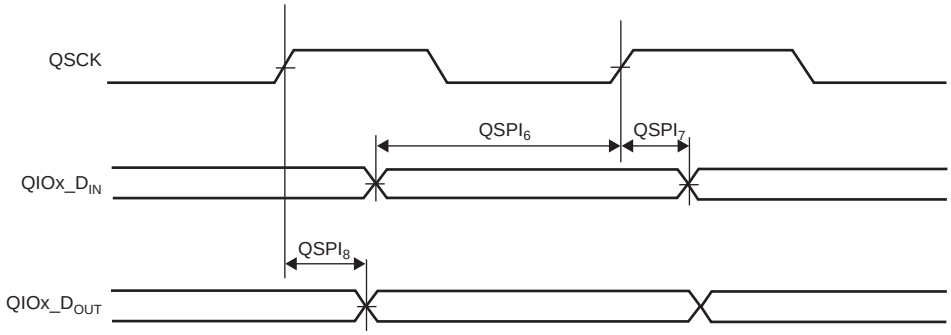


Figure 62-28. QSPI Master Mode 3

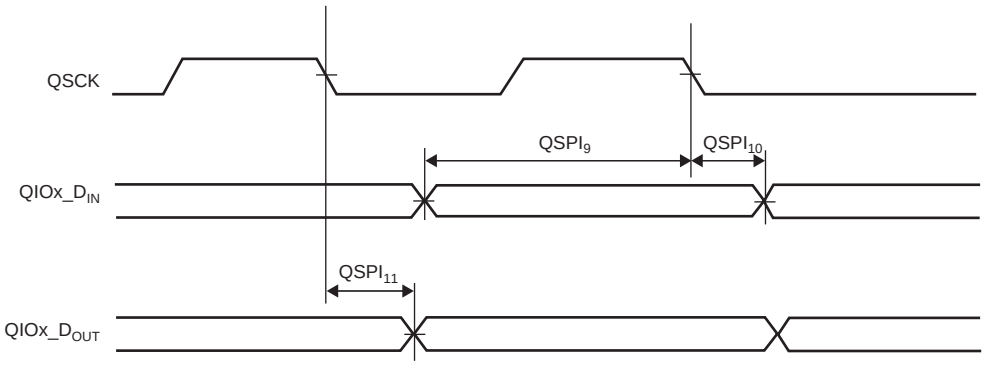


Table 62-66. QSPI0 IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Mode 0						
$QSPI_0$	QIOx Input setup time before SCK falls	1.5	–	1.4	–	ns
$QSPI_1$	QIOx Input hold time after SCK falls	0.4	–	0.5	–	ns
$QSPI_2$	SCK falling to QIOx valid	0	3.4	0	2.4	ns
Mode 1						
$QSPI_3$	QIOx Input setup time before SCK rises	11	–	8.7	–	ns
$QSPI_4$	QIOx Input hold time after SCK rises	0.1	–	0	–	ns
$QSPI_5$	SCK rising to QIOx valid	0	3	0	2.2	ns
Mode 2						
$QSPI_6$	QIOx Input setup time before SCK rises	1.7	–	1.4	–	ns
$QSPI_7$	QIOx Input hold time after SCK rises	0.1	–	0	–	ns
$QSPI_8$	SCK rising to QIOx valid	0	3.3	0	2.3	ns
Mode 3						
$QSPI_9$	QIOx Input setup time before SCK falls	11	–	8.9	–	ns
$QSPI_{10}$	QIOx Input hold time after SCK falls	0.4	–	0.4	–	ns
$QSPI_{11}$	SCK falling to QIOx valid	0	3.2	0	2.4	ns

Table 62-67. QSPI0 IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Mode 0						
$QSPI_0$	QIOx Input setup time before SCK falls	2.2	–	2.2	–	ns
$QSPI_1$	QIOx Input hold time after SCK falls	0.8	–	0.7	–	ns
$QSPI_2$	SCK falling to QIOx valid	0	1.8	0	2	ns
Mode 1						
$QSPI_3$	QIOx Input setup time before SCK rises	12.7	–	10.5	–	ns
$QSPI_4$	QIOx Input hold time after SCK rises	0.3	–	0.2	–	ns
$QSPI_5$	SCK rising to QIOx valid	0	2.2	0	1.9	ns
Mode 2						
$QSPI_6$	QIOx Input setup time before SCK rises	2.6	–	2.5	–	ns
$QSPI_7$	QIOx Input hold time after SCK rises	0.3	–	0.2	–	ns
$QSPI_8$	SCK rising to QIOx valid	0	2.4	0	2	ns
Mode 3						
$QSPI_9$	QIOx Input setup time before SCK falls	12	–	10.5	–	ns
$QSPI_{10}$	QIOx Input hold time after SCK falls	0.8	–	0.7	–	ns
$QSPI_{11}$	SCK falling to QIOx valid	0	1.5	0	1.9	ns

Table 62-68. QSPI0 IOSET3 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Mode 0						
$QSPI_0$	QIOx Input setup time before SCK falls	1.8	–	1.7	–	ns
$QSPI_1$	QIOx Input hold time after SCK falls	0.8	–	0.7	–	ns
$QSPI_2$	SCK falling to QIOx valid	0	1.8	0	2.1	ns
Mode 1						
$QSPI_3$	QIOx Input setup time before SCK rises	12.3	–	10.1	–	ns
$QSPI_4$	QIOx Input hold time after SCK rises	0.5	–	0.3	–	ns
$QSPI_5$	SCK rising to QIOx valid	0	2.2	0	2	ns
Mode 2						
$QSPI_6$	QIOx Input setup time before SCK rises	2	–	1.7	–	ns
$QSPI_7$	QIOx Input hold time after SCK rises	0.5	–	0.3	–	ns
$QSPI_8$	SCK rising to QIOx valid	0	2.5	0	2.2	ns
Mode 3						
$QSPI_9$	QIOx Input setup time before SCK falls	11.7	–	10.2	–	ns
$QSPI_{10}$	QIOx Input hold time after SCK falls	0.8	–	0.7	–	ns
$QSPI_{11}$	SCK falling to QIOx valid	0	1.5	1.2	2	ns

Table 62-69. QSPI1 IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Mode 0						
$QSPI_0$	QIOx Input setup time before SCK falls	1.1	–	0.9	–	ns
$QSPI_1$	QIOx Input hold time after SCK falls	1	–	0.7	–	ns
$QSPI_2$	SCK falling to QIOx valid	0	3.2	0	2.4	ns
Mode 1						
$QSPI_3$	QIOx Input setup time before SCK rises	12	–	9.7	–	ns
$QSPI_4$	QIOx Input hold time after SCK rises	0.8	–	0.5	–	ns
$QSPI_5$	SCK rising to QIOx valid	0	2.7	0	2.1	ns
Mode 2						
$QSPI_6$	QIOx Input setup time before SCK rises	1.1	–	0.8	–	ns
$QSPI_7$	QIOx Input hold time after SCK rises	0.8	–	0.5	–	ns
$QSPI_8$	SCK rising to QIOx valid	0	3	0	2.3	ns
Mode 3						
$QSPI_9$	QIOx Input setup time before SCK falls	12.2	–	10	–	ns
$QSPI_{10}$	QIOx Input hold time after SCK falls	1	–	0.7	–	ns
$QSPI_{11}$	SCK falling to QIOx valid	0	3	0	2.4	ns

Table 62-70. QSPI1 IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Mode 0						
$QSPI_0$	QIOx Input setup time before SCK falls	1.9	–	1.9	–	ns
$QSPI_1$	QIOx Input hold time after SCK falls	0.8	–	0.7	–	ns
$QSPI_2$	SCK falling to QIOx valid	0	1.5	0	1.9	ns
Mode 1						
$QSPI_3$	QIOx Input setup time before SCK rises	12.6	–	10.4	–	ns
$QSPI_4$	QIOx Input hold time after SCK rises	0.2	–	0.1	–	ns
$QSPI_5$	SCK rising to QIOx valid	0	1.9	0	1.8	ns
Mode 2						
$QSPI_6$	QIOx Input setup time before SCK rises	2.2	–	2.1	–	ns
$QSPI_7$	QIOx Input hold time after SCK rises	0.2	–	0.1	–	ns
$QSPI_8$	SCK rising to QIOx valid	0	2.2	0	2	ns
Mode 3						
$QSPI_9$	QIOx Input setup time before SCK falls	11.9	–	10.4	–	ns
$QSPI_{10}$	QIOx Input hold time after SCK falls	0.8	–	0.7	–	ns
$QSPI_{11}$	SCK falling to QIOx valid	0	1.3	0	1.9	ns

Table 62-71. QSPI1 IOSET3 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Mode 0						
$QSPI_0$	QIOx Input setup time before SCK falls	1.8	–	1.8	–	ns
$QSPI_1$	QIOx Input hold time after SCK falls	0.9	–	0.7	–	ns
$QSPI_2$	SCK falling to QIOx valid	0	1.3	0	1.8	ns
Mode 1						
$QSPI_3$	QIOx Input setup time before SCK rises	13.1	–	10.9	–	ns
$QSPI_4$	QIOx Input hold time after SCK rises	0.4	–	0.2	–	ns
$QSPI_5$	SCK rising to QIOx valid	0	1.7	0	1.7	ns
Mode 2						
$QSPI_6$	QIOx Input setup time before SCK rises	2.2	–	2.1	–	ns
$QSPI_7$	QIOx Input hold time after SCK rises	0.4	–	0.2	–	ns
$QSPI_8$	SCK rising to QIOx valid	0	2	0	1.8	ns
Mode 3						
$QSPI_9$	QIOx Input setup time before SCK falls	12.5	–	11	–	ns
$QSPI_{10}$	QIOx Input hold time after SCK falls	0.9	–	0.7	–	ns
$QSPI_{11}$	SCK falling to QIOx valid	0	1.1	0	1.7	ns

62.17 TWI Timings

Figure 62-29. Two-wire Serial Bus Timing

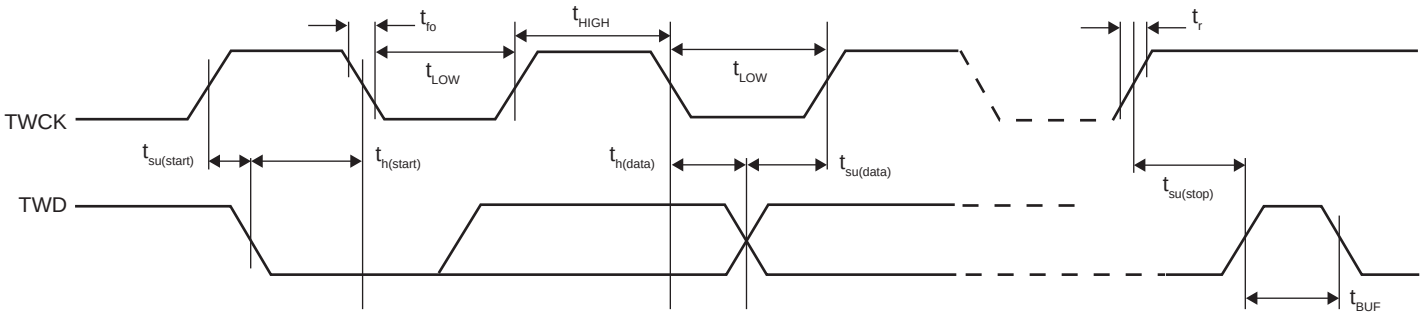


Table 62-72 describes the requirements for devices connected to the Two-wire Serial Bus.

Table 62-72. Two-wire Serial Bus Requirements

Symbol	Parameter	Conditions	Min	Max	Unit
V_{IL}	Input Low-voltage	–	-0.3	$0.3 \times V_{DDIO}$	V
V_{IH}	Input High-voltage	–	$0.7 \times V_{DDIO}$	V_{CC} + 0.3	V
V_{hys}	Hysteresis of Schmitt Trigger Inputs	–	0.150	–	V
V_{OL}	Output Low-voltage	3 mA sink current	–	0.4	V
t_r	Rise Time for both TWD and TWCK		$20 + 0.1C_b$ (2)	300	ns
t_{fo}	Output Fall Time from V_{IHmin} to V_{ILmax}	10 pF < C_b < 400 pF Figure 62-29	$20 + 0.1C_b$ (2)	250	ns
C_i (1)	Capacitance for each I/O Pin	–	–	10	pF
f_{TWCK}	TWCK Clock Frequency	–	0	400	kHz
Rp	Value of Pull-up Resistor	$f_{TWCK} \le 100$ kHz	$(V_{DDIO}$ - 0.4V) ÷ 3mA	1000ns ÷ C_b	Ω
		$f_{TWCK} > 100$ kHz	$(V_{DDIO}$ - 0.4V) ÷ 3mA	300ns ÷ C_b	Ω
t_{LOW}	Low Period of the TWCK Clock	$f_{TWCK} \le 100$ kHz	(3)	–	µs
		$f_{TWCK} > 100$ kHz	(3)	–	µs
t_{HIGH}	High Period of the TWCK Clock	$f_{TWCK} \le 100$ kHz	(4)	–	µs
		$f_{TWCK} > 100$ kHz	(4)	–	µs
$t_{h(start)}$	Hold Time (repeated) START condition	$f_{TWCK} \le 100$ kHz	t_{HIGH}	–	µs
		$f_{TWCK} > 100$ kHz	t_{HIGH}	–	µs
$t_{su(start)}$	Setup Time for a Repeated START condition	$f_{TWCK} \le 100$ kHz	t_{HIGH}	–	µs
		$f_{TWCK} > 100$ kHz	t_{HIGH}	–	µs
$t_{h(data)}$	Data Hold Time	$f_{TWCK} \le 100$ kHz	0	(HOLD + 3) × $t_{peripheral\ clock}$	µs
		$f_{TWCK} > 100$ kHz	0	(HOLD + 3) × $t_{peripheral\ clock}$	µs
$t_{su(data)}$	Data Setup Time	$f_{TWCK} \le 100$ kHz	t_{LOW} - (HOLD + 3) × $t_{peripheral\ clock}$	–	ns
		$f_{TWCK} > 100$ kHz	t_{LOW} - (HOLD + 3) × $t_{peripheral\ clock}$	–	ns

Table 62-72. Two-wire Serial Bus Requirements (Continued)

Symbol	Parameter	Conditions	Min	Max	Unit
$t_{su(stop)}$	Setup time for STOP condition	$f_{TWCK} \leq 100$ kHz	t_{HIGH}	–	µs
		$f_{TWCK} > 100$ kHz	t_{HIGH}	–	µs
t_{BUF}	Bus free time between a STOP and START condition	$f_{TWCK} \leq 100$ kHz	t_{LOW}	–	µs
		$f_{TWCK} > 100$ kHz	t_{LOW}	–	µs

Notes:
1. Required only for $f_{TWCK} > 100$ kHz
2. C_b = capacitance of one bus line in pF. Per I2C Standard, C_b Max = 400 pF
3. The TWCK low period is defined as follows: $t_{LOW} = ((CLDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$
4. The TWCK high period is defined as follows: $t_{HIGH} = ((CHDIV \times 2^{CKDIV}) + 4) \times t_{MCK}$

62.18 MPDDRC Timings

62.18.1 Board Design Constraints

As the SAMA5D2 series embeds impedance calibrated pads, there are no capacitive constraints on DDR signals. However, a board must be designed and equipped in order to respect propagation time and intrinsic delay in the SDRAM device. In all cases, line length to memory device must not exceed 5 cm.

62.18.2 DDR2-SDRAM

Note: For DDR2 memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured to 1.

Table 62-73. System Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.1V, 1.32V], T_A = 85°C	7.5	8.0	ns
		VDDCORE[1.2V, 1.32V], VDDIODDR[1.75V, 1.9V], T_A = 85°C	6.0	8.0	ns

62.18.3 LPDDR1-SDRAM

Note: For LPDDR1 memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured as follows:

SHIFT_SAMPLING = 0 for 0 < DDR_CLK < 94 MHz

SHIFT_SAMPLING = 1 for 94 MHz < DDR_CLK < 166 MHz

Table 62-74. System Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.1V, 1.32V], T_A = 85°C	7.5	–	ns
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.2V, 1.32V], VDDIODDR[1.75V, 1.9V], T_A = 85°C	6.4	–	ns

62.18.4 LPDDR2/LPDDR3-SDRAM

Note: For LPDDR2/LPDDR3 memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured as follows:

SHIFT_SAMPLING = 0 for 0 < DDR_CLK < 80 MHz

SHIFT_SAMPLING = 1 for 80 MHz < DDR_CLK < 166 MHz

Table 62-75. System Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.1V, 1.32V], T_A = 85°C	7.5	–	ns
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.1V, 1.32V], VDDIODDR[1.18V, 1.3V], T_A = 85°C	6.9	–	ns

62.18.5 DDR3/DDR3L-SDRAM

Note: For DDR3/DDR3L memory, the SHIFT_SAMPLING field value in the MPRDDRC_RD_DATA_PATH register must be configured to 2.

Table 62-76. System Clock Waveform Parameters

Symbol	Parameter	Conditions	Min	Max	Unit
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.1V, 1.32V], T_A = 85°C	7.5	–	ns
t_{DDRCK}	DDRCK Cycle Time	VDDCORE[1.1V, 1.32V], VDDIODDR[1.18V, 1.3V], T_A = 85°C	6.9	–	ns

62.19 SSC Timings

62.19.1 Timing Conditions

Timings assuming a capacitance load are given in Table 62-77.

Table 62-77. Capacitance Load (SSC0 and SSC1)

Supply	Corner	
	Max	Min
3.3V	30 pF	5 pF
1.8V	20 pF	5 pF

62.19.2 Timing Extraction

Figure 62-30. SSC Transmitter, TK and TF in Output

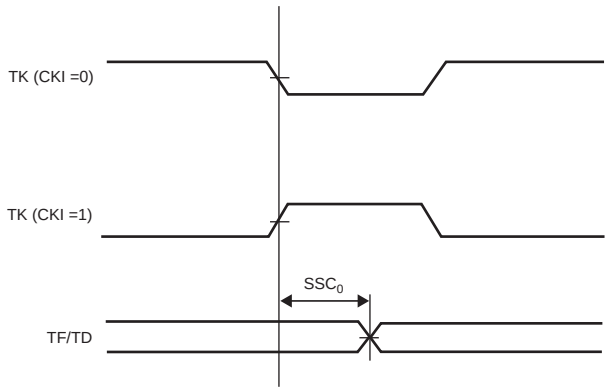


Figure 62-31. SSC Transmitter, TK in Input and TF in Output

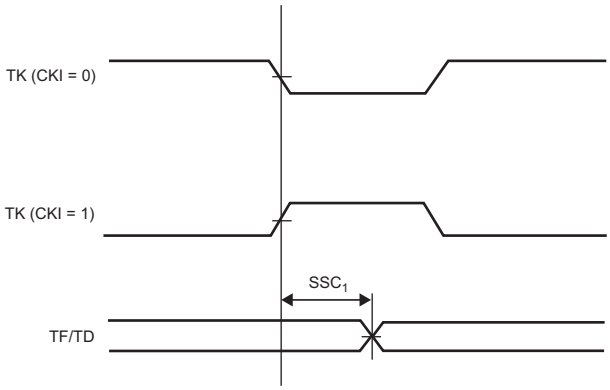


Figure 62-32. SSC Transmitter, TK in Output and TF in Input

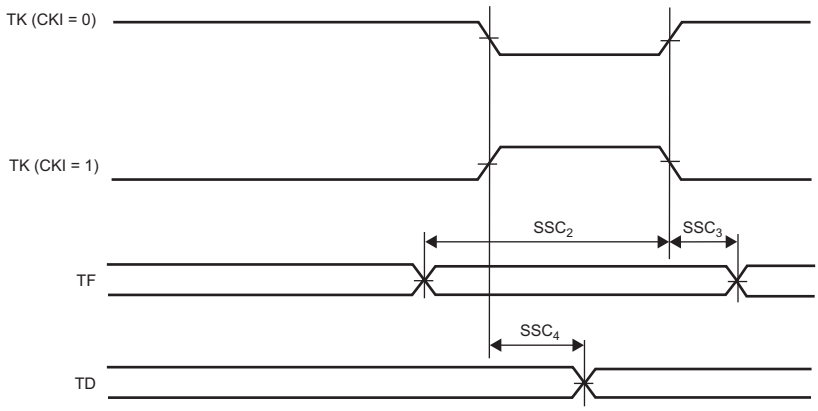


Figure 62-33. SSC Transmitter, TK and TF in Input

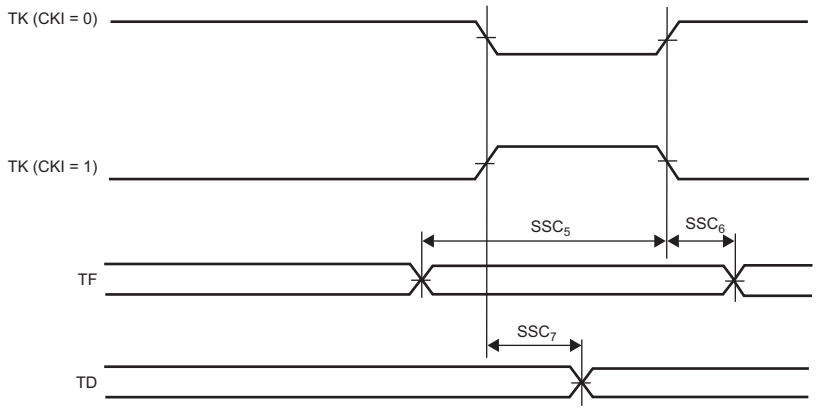


Figure 62-34. SSC Receiver RK and RF in Input

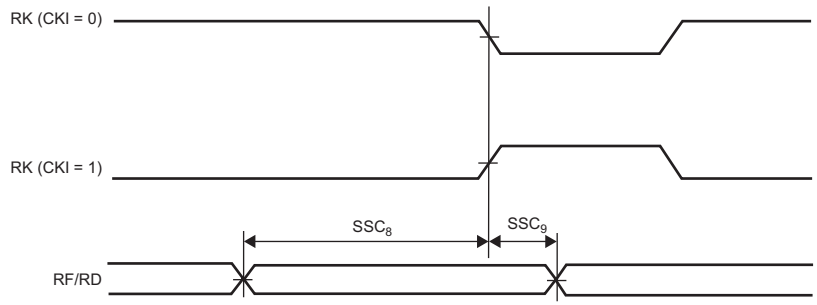


Figure 62-35. SSC Receiver, RK in Input and RF in Output

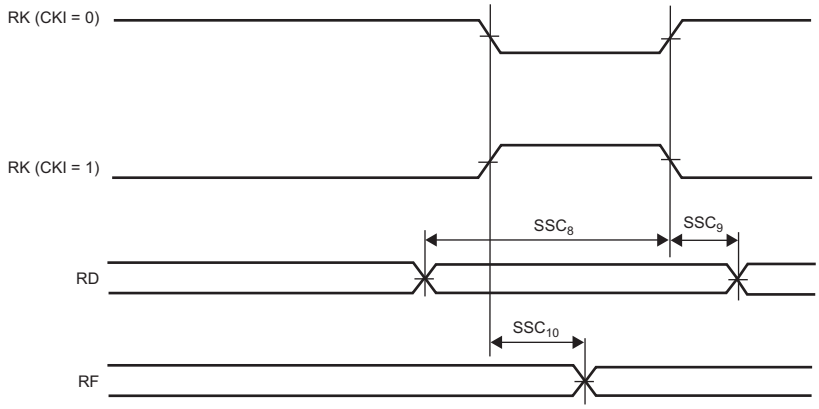


Figure 62-36. SSC Receiver, RK and RF in Output

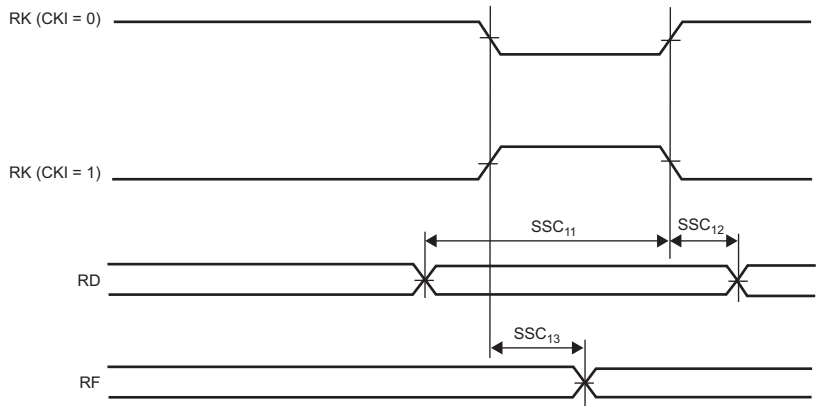


Figure 62-37. SSC Receiver, RK in Output and RF in Input

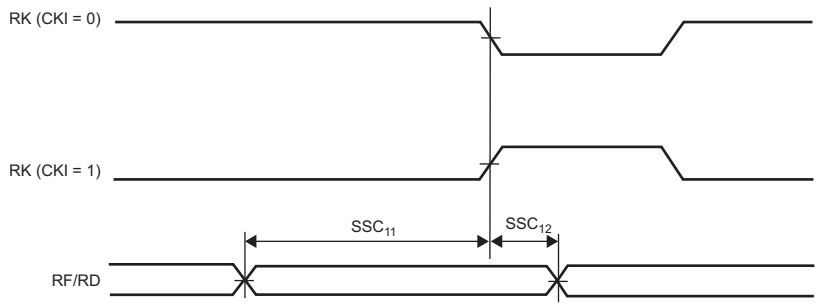


Table 62-78. SSC0 IOSET1 Timings

Symbol	Power supply / Parameter	Conditions	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Transmitter						
SSC_0	TK edge to TF/TD (TK output, TF output) (1)	–	0	3	0	3.3	ns
SSC_1	TK edge to TF/TD (TK input, TF output) (1)	–	3.7	13	3	11.3	ns
SSC_2	TF setup time before TK edge (TK output)	–	12.8	–	11.2	–	ns
SSC_3	TF hold time after TK edge (TK output)	–	0	–	0	–	ns
SSC_4	TK edge to TF/TD (TK output, TF input)(1)	–	0	3	0	3.3	ns
		STTDLY = 0 START = 4, 5 or 7	$2 \times t_{CPMCK}$	$3 + (2 \times t_{CPMCK})$	$2 \times t_{CPMCK}$	$3.3 + (2 \times t_{CPMCK})$	ns
SSC_5	TF setup time before TK edge (TK input)	–	0	–	0	–	
SSC_6	TF hold time after TK edge (TK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	
SSC_7	TK edge to TF/TD (TK input, TF input)(1)	–	3.7	13	3.2	11.3	
		STTDLY = 0 START = 4, 5 or 7	$3.7 + (3 \times t_{CPMCK})$	$13 + (3 \times t_{CPMCK})$	$3.2 + (3 \times t_{CPMCK})$	$11.3 + (3 \times t_{CPMCK})$	
	Receiver						
SSC_8	RF/RD setup time before RK edge (RK input)	–	0	–	0	–	ns
SSC_9	RF/RD hold time after RK edge (RK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{10}	RK edge to RF (RK input) (1)	–	3.5	12	2.9	10.4	ns
SSC_{11}	RF/RD setup time before RK edge (RK output)	–	$13.6 - t_{CPMCK}$	–	$12 - t_{CPMCK}$	–	ns
SSC_{12}	RF/RD hold time after RK edge (RK output)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{13}	RK edge to RF (RK output) (1)	–	0	3	0	3.3	ns

Note: 1. For output signals (TF, TD, RF), minimum and maximum access times are defined. The minimum access time is the time between the TK (or RK) edge and the signal change. The maximum access time is the time between the TK edge and the signal stabilization. Figure 62-38 illustrates the minimum and maximum accesses for SSC_0. The same applies for SSC_1, SSC_4, SSC_7, SSC_{10} and SSC_{13}.

Table 62-79. SSC0 IOSET2 Timings

Symbol	Power supply / Parameter	Conditions	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
		Transmitter					
SSC_0	TK edge to TF/TD (TK output, TF output) (1)	–	0	3.4	0	3.7	ns
SSC_1	TK edge to TF/TD (TK input, TF output) (1)	–	3.5	12.3	2.8	10.5	ns
SSC_2	TF setup time before TK edge (TK output)	–	12	–	10.3	–	ns
SSC_3	TF hold time after TK edge (TK output)	–	0	–	0	–	ns
SSC_4	TK edge to TF/TD (TK output, TF input) (1)	–	0	3.4	0	3.5	ns
		STTDLY = 0 START = 4, 5 or 7	$2 \times t_{CPMCK}$	3.4 + $(2 \times t_{CPMCK})$	$2 \times t_{CPMCK}$	3.5 + $(2 \times t_{CPMCK})$	ns
SSC_5	TF setup time before TK edge (TK input)	–	0	–	0	–	
SSC_6	TF hold time after TK edge (TK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	
SSC_7	TK edge to TF/TD (TK input, TF input) (1)	–	3.6	12.3	3	10.4	
		STTDLY = 0 START = 4, 5 or 7	3.6 + $(3 \times t_{CPMCK})$	12.3 + $(3 \times t_{CPMCK})$	3 + $(3 \times t_{CPMCK})$	10.4 + $(3 \times t_{CPMCK})$	
		Receiver					
SSC_8	RF/RD setup time before RK edge (RK input)	–	0	–	0	–	ns
SSC_9	RF/RD hold time after RK edge (RK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{10}	RK edge to RF (RK input) (1)	–	3.3	11.5	2.7	9.8	ns
SSC_{11}	RF/RD setup time before RK edge (RK output)	–	13.8 - t_{CPMCK}	–	12.1 - t_{CPMCK}	–	ns
SSC_{12}	RF/RD hold time after RK edge (RK output)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{13}	RK edge to RF (RK output) (1)	–	0	2.8	0	3.1	ns

Note: 1. For output signals (TF, TD, RF), minimum and maximum access times are defined. The minimum access time is the time between the TK (or RK) edge and the signal change. The maximum access time is the time between the TK edge and the signal stabilization. Figure 62-38 illustrates the minimum and maximum accesses for SSC_0. The same applies for SSC_1, SSC_4, SSC_7, SSC_{10} and SSC_{13}.

Table 62-80. SSC1 IOSET1 Timings

Symbol	Power supply / Parameter	Conditions	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
Transmitter							
SSC_0	TK edge to TF/TD (TK output, TF output) (1)	–	0	2.6	0	2.7	ns
SSC_1	TK edge to TF/TD (TK input, TF output) (1)	–	3.6	12.7	3	10.9	ns
SSC_2	TF setup time before TK edge (TK output)	–	13.4	–	11.2	–	ns
SSC_3	TF hold time after TK edge (TK output)	–	0	–	0	–	ns
SSC_4	TK edge to TF/TD (TK output, TF input) (1)	–	0	2.1	0	2	ns
		STTDLY = 0 START = 4, 5 or 7	$2 \times t_{CPMCK}$	2.1 + $(2 \times t_{CPMCK})$	$2 \times t_{CPMCK}$	2 + $(2 \times t_{CPMCK})$	ns
SSC_5	TF setup time before TK edge (TK input)	–	0	–	0	–	
SSC_6	TF hold time after TK edge (TK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	
SSC_7	TK edge to TF/TD (TK input, TF input) (1)	–	3.6	12.2	3	10.2	
		STTDLY = 0 START = 4, 5 or 7	3.6 + $(3 \times t_{CPMCK})$	12.2 + $(3 \times t_{CPMCK})$	3 + $(3 \times t_{CPMCK})$	10.2 + $(3 \times t_{CPMCK})$	
Receiver							
SSC_8	RF/RD setup time before RK edge (RK input)	–	0	–	0	–	ns
SSC_9	RF/RD hold time after RK edge (RK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{10}	RK edge to RF (RK input) (1)	–	3.4	11.8	2.7	9.9	ns
SSC_{11}	RF/RD setup time before RK edge (RK output)	–	12.2 - t_{CPMCK}	–	10.3 - t_{CPMCK}	–	ns
SSC_{12}	RF/RD hold time after RK edge (RK output)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{13}	RK edge to RF (RK output) (1)	–	0	3.3	0	3.4	ns

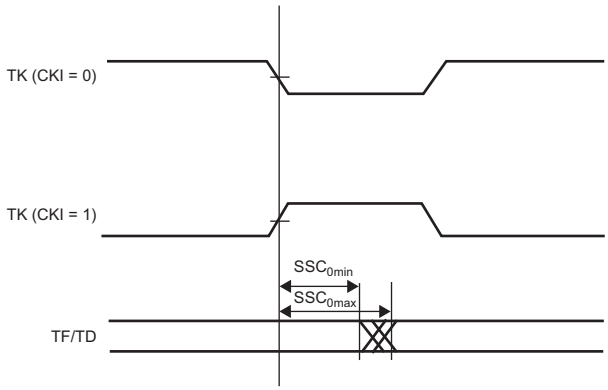
Note: 1. For output signals (TF, TD, RF), minimum and maximum access times are defined. The minimum access time is the time between the TK (or RK) edge and the signal change. The maximum access time is the time between the TK edge and the signal stabilization. Figure 62-38 illustrates the minimum and maximum accesses for SSC_0. The same applies for SSC_1, SSC_4, SSC_7, SSC_{10} and SSC_{13}.

Table 62-81. SSC1 IOSET2 Timings

Symbol	Power supply / Parameter	Conditions	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
		Transmitter					
SSC_0	TK edge to TF/TD (TK output, TF output) (1)	–	0	2.5	0	2.6	ns
SSC_1	TK edge to TF/TD (TK input, TF output) (1)	–	3.7	13	3.1	11.3	ns
SSC_2	TF setup time before TK edge (TK output)	–	14.3	–	12.2	–	ns
SSC_3	TF hold time after TK edge (TK output)	–	0	–	0	–	ns
SSC_4	TK edge to TF/TD (TK output, TF input) (1)	–	0	2.1	0	1.8	ns
		STTDLY = 0 START = 4, 5 or 7	$2 \times t_{CPMCK}$	2.1 + $(2 \times t_{CPMCK})$	$2 \times t_{CPMCK}$	1.8 + $(2 \times t_{CPMCK})$	ns
SSC_5	TF setup time before TK edge (TK input)	–	0	–	0	–	
SSC_6	TF hold time after TK edge (TK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	
SSC_7	TK edge to TF/TD (TK input, TF input) (1)	–	3.7	12.6	3.1	10.4	
		STTDLY = 0 START = 4, 5 or 7	3.7 + $(3 \times t_{CPMCK})$	12.6 + $(3 \times t_{CPMCK})$	3.1 + $(3 \times t_{CPMCK})$	10.4 + $(3 \times t_{CPMCK})$	
		Receiver					
SSC_8	RF/RD setup time before RK edge (RK input)	–	0	–	0	–	ns
SSC_9	RF/RD hold time after RK edge (RK input)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{10}	RK edge to RF (RK input) (1)	–	3.6	12.2	3	10.3	ns
SSC_{11}	RF/RD setup time before RK edge (RK output)	–	12.3 - t_{CPMCK}	–	10.5 - t_{CPMCK}	–	ns
SSC_{12}	RF/RD hold time after RK edge (RK output)	–	t_{CPMCK}	–	t_{CPMCK}	–	ns
SSC_{13}	RK edge to RF (RK output) (1)	–	0	2.9	0	3.1	ns

Note: 1. For output signals (TF, TD, RF), minimum and maximum access times are defined. The minimum access time is the time between the TK (or RK) edge and the signal change. The maximum access time is the time between the TK edge and the signal stabilization. Figure 62-38 illustrates the minimum and maximum accesses for SSC_0. The same applies for SSC_1, SSC_4, SSC_7, SSC_{10} and SSC_{13}.

Figure 62-38. Minimum and Maximum Access Time of Output Signals



62.20 PDMIC Timings

62.20.1 Timing Conditions

Timings assuming capacitance loads are given in Table 62-82.

Table 62-82. Capacitance Load

Supply	Corner	
	Max	Min
3.3V	30 pF	5 pF
1.8V	20 pF	5 pF

62.20.2 Timing Extraction

Figure 62-39. PDMIC Timing Diagram

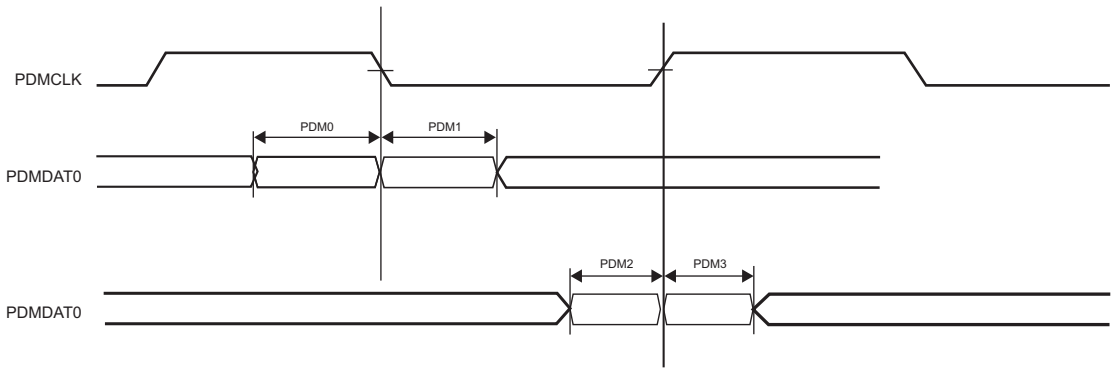


Table 62-83. PDMIC IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
$PDMIC_0$	DATA setup time right	3.5	–	3.5	–	ns
$PDMIC_1$	DATA hold time right	3.1	–	3.5	–	ns
$PDMIC_2$	DATA setup time left	3.5	–	3.5	–	ns
$PDMIC_3$	DATA hold time left	3.1	–	3.5	–	ns

Table 62-84. PDMIC IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
$PDMIC_0$	DATA setup time right	4.2	–	4.2	–	ns
$PDMIC_1$	DATA hold time right	2	–	2	–	ns
$PDMIC_2$	DATA setup time left	4.2	–	4.2	–	ns
$PDMIC_3$	DATA hold time left	2	–	2	–	ns

62.21 I2SC Timings

62.21.1 Timing Conditions

Timings assuming capacitance loads are given in Table 62-82.

Table 62-85. Capacitance Load (I2SC0 and I2SC1)

Supply	Corner	
	Max	Min
3.3V	30 pF	5 pF
1.8V	20 pF	5 pF

62.21.2 Timing Extraction

Table 62-86. I2SC0 IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master					
$I2SC_0$	SDI Input setup time before SCK rises	12.5	–	10.8	–	ns
$I2SC_1$	SDI Input hold time after SCK rises	0	–	0	–	ns
$I2SC_2$	SCK falling to SDO valid	0	3.9	0	4	ns
$I2SC_3$	SCK falling to WS valid	0	2.7	0	3.1	ns
	Slave					
$I2SC_4$	SDI Input setup time before SCK rises	1.1	–	1	–	ns
$I2SC_5$	SDI Input hold time after SCK rises	1.3	–	1.2	–	ns
$I2SC_6$	WS Input setup time before SCK rises	2	–	1.8	–	ns
$I2SC_7$	WS Input hold time after SCK rises	0.9	–	0.8	–	ns
$I2SC_8$	SCK falling to SDO valid	4.2	14	3.6	12	ns

Table 62-87. I2SC0 IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master					
$I2SC_0$	SDI Input setup time before SCK rises	11.7	–	9.7	–	ns
$I2SC_1$	SDI Input hold time after SCK rises	0	–	0	–	ns
$I2SC_2$	SCK falling to SDO valid	0	3	0	3	ns
$I2SC_3$	SCK falling to WS valid	0.1	4.7	0.2	4.8	ns
	Slave					
$I2SC_4$	SDI Input setup time before SCK rises	1.7	–	1.5	–	ns
$I2SC_5$	SDI Input hold time after SCK rises	0.6	–	0.4	–	ns
$I2SC_6$	WS Input setup time before SCK rises	3.6	–	3.4	–	ns
$I2SC_7$	WS Input hold time after SCK rises	0.7	–	0.6	–	ns
$I2SC_8$	SCK falling to SDO valid	3.7	12	3	10	ns

Table 62-88. I2SC1 IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master					
$I2SC_0$	SDI Input setup time before SCK rises	13.1	–	11.4	–	ns
$I2SC_1$	SDI Input hold time after SCK rises	0	–	0	–	ns
$I2SC_2$	SCK falling to SDO valid	0	3.5	0	3.6	ns
$I2SC_3$	SCK falling to WS valid	0	2.9	0	3	ns
	Slave					
$I2SC_4$	SDI Input setup time before SCK rises	1.4	–	1.3	–	ns
$I2SC_5$	SDI Input hold time after SCK rises	1	–	0.8	–	ns
$I2SC_6$	WS Input setup time before SCK rises	2.4	–	2.1	–	ns
$I2SC_7$	WS Input hold time after SCK rises	0.8	–	0.7	–	ns
$I2SC_8$	SCK falling to SDO valid	4.4	13.8	3.7	11.9	ns

Table 62-89. I2SC1 IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
	Master					
$I2SC_0$	SDI Input setup time before SCK rises	12.9	–	11.2	–	ns
$I2SC_1$	SDI Input hold time after SCK rises	0	–	0	–	ns
$I2SC_2$	SCK falling to SDO valid	0	3.6	0	3.7	ns
$I2SC_3$	SCK falling to WS valid	0	2.9	0	3	ns
	Slave					
$I2SC_4$	SDI Input setup time before SCK rises	1.1	–	1	–	ns
$I2SC_5$	SDI Input hold time after SCK rises	1.2	–	1	–	ns
$I2SC_6$	WS Input setup time before SCK rises	2.2	–	2	–	ns
$I2SC_7$	WS Input hold time after SCK rises	0.9	–	0.7	–	ns
$I2SC_8$	SCK falling to SDO valid	4.3	14	3.7	12	ns

62.22 ISC Timings

62.22.1 Timing Conditions

Timings assuming capacitance loads are given in Table 62-90.

Table 62-90. Capacitance Load

Supply	Corner	
	Max	Min
3.3V	30 pF	5 pF
1.8V	20 pF	5 pF

62.22.2 Timing Extraction

Figure 62-40. ISC Timing Diagram

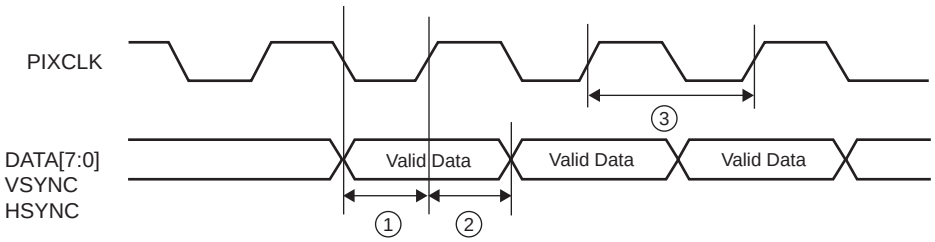


Table 62-91. ISC IOSET1 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
ISC_1	DATA/VSYNC/HSYNC setup time	3.9	–	3.6	–	ns
ISC_2	DATA/VSYNC/HSYNC hold time	0.7	–	0.6	–	ns
ISC_3	CONTROL VSYNC/HSYNC/FIELD setup time	4.4	–	4.2	–	ns
ISC_4	CONTROL VSYNC/HSYNC/FIELD hold time	0.4	–	0.3	–	ns
ISC_5	PIXCLK frequency	96	–	96	–	MHz

Table 62-92. ISC IOSET2 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
ISC_1	DATA/VSYNC/HSYNC setup time	4.3	–	4.2	–	ns
ISC_2	DATA/VSYNC/HSYNC hold time	0.5	–	0.3	–	ns
ISC_3	CONTROL VSYNC/HSYNC/FIELD setup time	4.6	–	4.4	–	ns
ISC_4	CONTROL VSYNC/HSYNC/FIELD hold time	0.2	–	0	–	ns
ISC_5	PIXCLK frequency	96	–	96	–	MHz

Table 62-93. ISC IOSET3 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
ISC_1	DATA/VSYNC/HSYNC setup time	4.6	–	4.2	–	ns
ISC_2	DATA/VSYNC/HSYNC hold time	0.5	–	0.4	–	ns
ISC_3	CONTROL VSYNC/HSYNC/FIELD setup time	4.3	–	4	–	ns
ISC_4	CONTROL VSYNC/HSYNC/FIELD hold time	1.5	–	0.4	–	ns
ISC_5	PIXCLK frequency	96	–	96	–	MHz

Table 62-94. ISC IOSET4 Timings

Symbol	Power Supply / Parameter	1.8V Min	1.8V Max	3.3V Min	3.3V Max	Unit
ISC_1	DATA/VSYNC/HSYNC setup time	4.3	–	4	–	ns
ISC_2	DATA/VSYNC/HSYNC hold time	0.5	–	0.4	–	ns
ISC_3	CONTROL VSYNC/HSYNC/FIELD setup time	4.2	–	4	–	ns
ISC_4	CONTROL VSYNC/HSYNC/FIELD hold time	0.5	–	0.3	–	ns
ISC_5	PIXCLK frequency	96	–	96	–	MHz

62.23 SDMMC Timings

The Secure Digital Multimedia Card (SDMMC) Controller supports the embedded MultiMedia Card (e.MMC) Specification V4.51, the SD Memory Card Specification V3.0, and the SDIO V3.0 specification. It is compliant with the SD Host Controller Standard V3.0 specification.

Features are different for the two instances of SDMMC:

SDMMC0: SD 3.0, eMMC 4.51, 8 bits

SDMMC1: SD 2.0, eMMC 4.41, 4 bits only

In SDR104 mode (SD 3.0), SDMMC0 is limited to 120 MHz (instead of 208 MHz). In HS200 mode (eMMC 4.51), SDMMC0 is limited to 120 MHz (instead of 200 MHz).

62.24 GMAC Timings

62.24.1 Timing Conditions

Timings assuming a capacitance load on data and clock are given in Table 62-95.

Table 62-95. Capacitance Load on Data, Clock Pads

Supply	Corner	
	Max	Min
3.3V	20 pF	0 pF

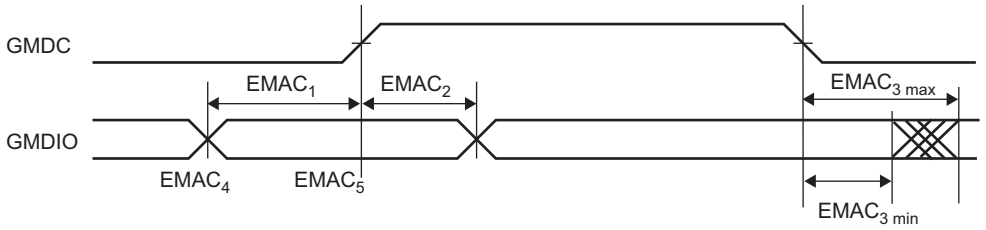
62.24.2 Timing Constraints

Table 62-96. Ethernet MAC Signals Relative to GMDC

Symbol	Parameter	Min	Max	Unit
$EMAC_1$	Setup for GMDIO from GMDC rising	10	–	ns
$EMAC_2$	Hold for GMDIO from GMDC rising	10	–	ns
$EMAC_3$	GMDIO toggling from GMDC rising(1)	0	300	ns

Note: 1. For Ethernet MAC output signals, minimum and maximum access time are defined. The minimum access time is the time between the GMDC rising edge and the signal change. The maximum access timing is the time between the GMDC rising edge and the signal stabilizes. Figure 62-41 illustrates minimum and maximum accesses for $EMAC_3$.

Figure 62-41. Minimum and Maximum Access Time of Ethernet MAC Output Signals

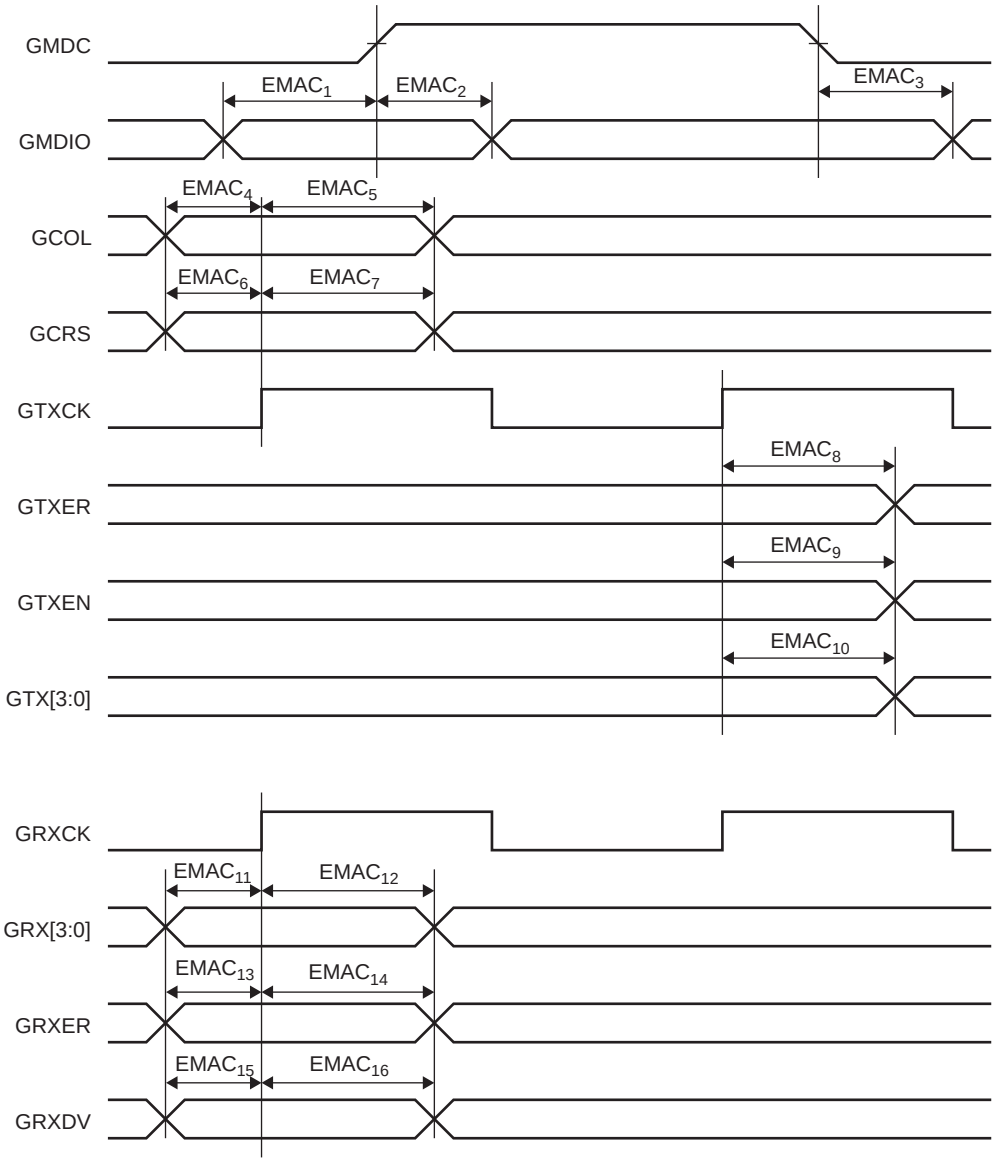


62.24.2.1 Ethernet MAC MII Mode

Table 62-97. Ethernet MAC MII Specific Signals

Symbol	Parameter	Min	Max	Unit
$EMAC_4$	Setup for GCOL from GTXCK rising	10	–	ns
$EMAC_5$	Hold for GCOL from GTXCK rising	10	–	ns
$EMAC_6$	Setup for GCRS from GTXCK rising	10	–	ns
$EMAC_7$	Hold for GCRS from GTXCK rising	10	–	ns
$EMAC_8$	GTXER toggling from GTXCK rising	10	25	ns
$EMAC_9$	GTXEN toggling from GTXCK rising	10	25	ns
$EMAC_{10}$	GTX toggling from GTXCK rising	10	25	ns
$EMAC_{11}$	Setup for GRX from GRXCK	10	–	ns
$EMAC_{12}$	Hold for GRX from GRXCK	10	–	ns
$EMAC_{13}$	Setup for GRXER from GRXCK	10	–	ns
$EMAC_{14}$	Hold for GRXER from GRXCK	10	–	ns
$EMAC_{15}$	Setup for GRXDV from GRXCK	10	–	ns
$EMAC_{16}$	Hold for GRXDV from GRXCK	10	–	ns

Figure 62-42. Ethernet MAC MII Mode

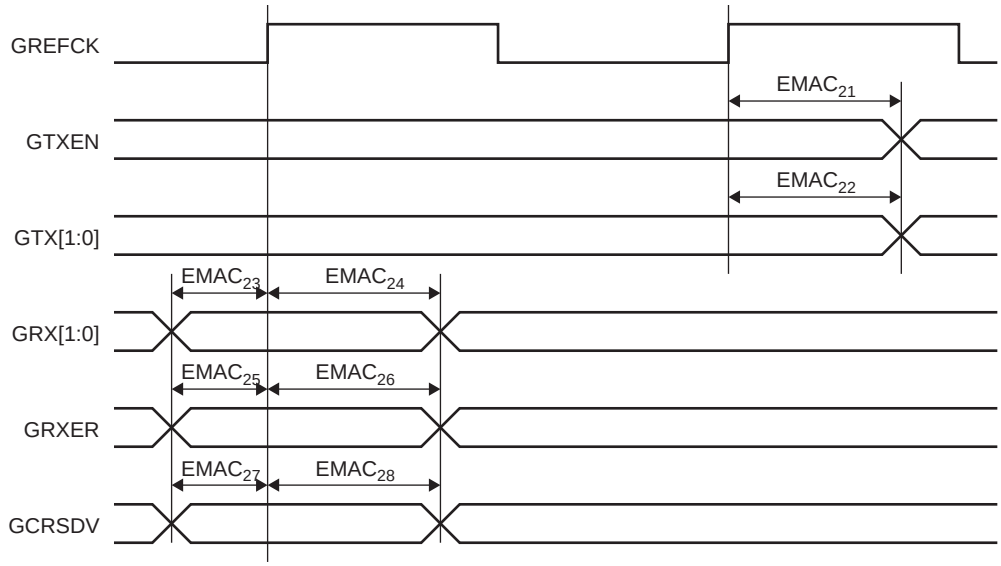


62.24.2.2 Ethernet MAC RMII Mode

Table 62-98. Ethernet MAC RMII Mode

Symbol	Parameter	Min	Max	Unit
$EMAC_{21}$	GTXEN toggling from GREFCK rising	2	16	ns
$EMAC_{22}$	GTX toggling from GREFCK rising	2	16	ns
$EMAC_{23}$	Setup for GRX from GREFCK rising	4	–	ns
$EMAC_{24}$	Hold for GRX from GREFCK rising	2	–	ns
$EMAC_{25}$	Setup for GRXER from GREFCK rising	4	–	ns
$EMAC_{26}$	Hold for GRXER from GREFCK rising	2	–	ns
$EMAC_{27}$	Setup for GCRSDV from GREFCK rising	4	–	ns
$EMAC_{28}$	Hold for GCRSDV from GREFCK rising	2	–	ns

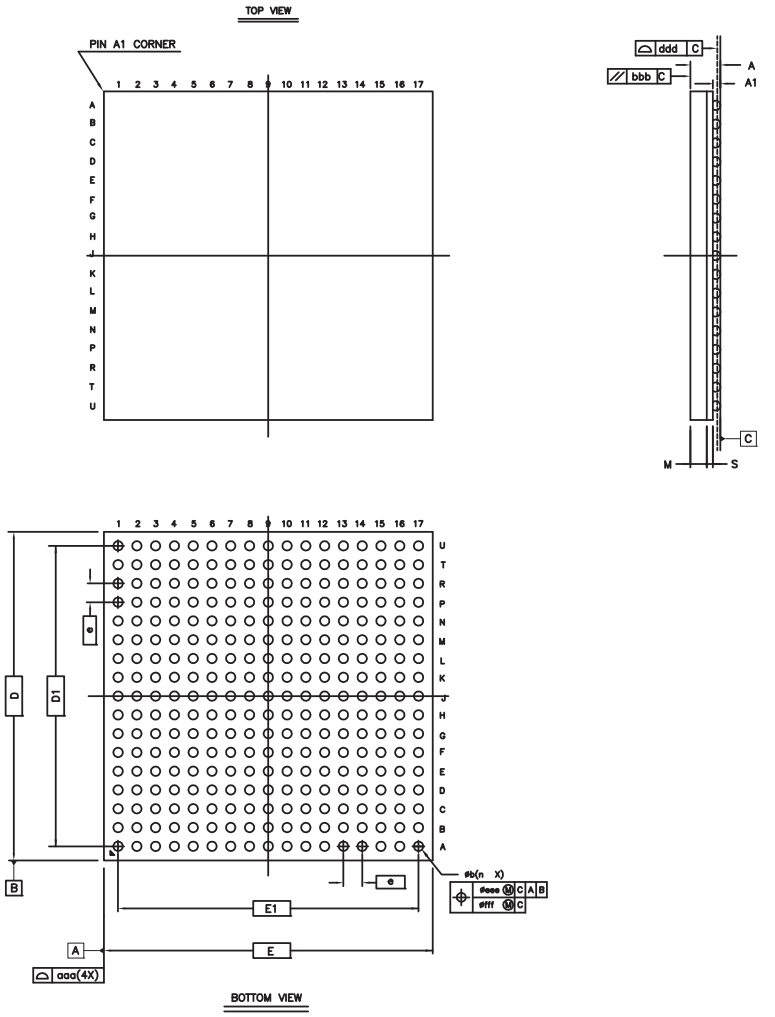
Figure 62-43. Ethernet MAC RMII Timings



63. Mechanical Characteristics

63.1 289-ball LFBGA Mechanical Characteristics

Figure 63-1. 289-ball LFBGA Package Drawing



		Symbol	Common Dimensions		
			MIN.	NOM.	MAX.
Package :			LFBGA		
Body Size:	X	E	14.000		
	Y	D	14.000		
Ball Pitch :		e	0.800		
Total Thickness :		A			1.400
Mold Thickness :		M	0.700 Ref.		
Substrate Thickness :		S	0.260 Ref.		
Ball Diameter :			0.400		
Stand Off :		A1	0.270	–	0.370
Ball Width :		b	0.380	–	0.480
Package Edge Tolerance :		aaa	0.150		
Mold Parallelism :		bbb	0.200		
Coplanarity:		ddd	0.120		
Ball Offset (Package) :		eee	0.150		
Ball Offset (Ball) :		fff	0.080		
Ball Count :		n	289		
Edge Ball Center to Center :	X	E1	12.800		
	Y	D1	12.800		

Table 63-1. 289-ball LFBGA Package Characteristics

Moisture Sensitivity Level	3

Table 63-2. Device and 289-ball LFBGA Package Weight

445	mg

Table 63-3. Package Reference

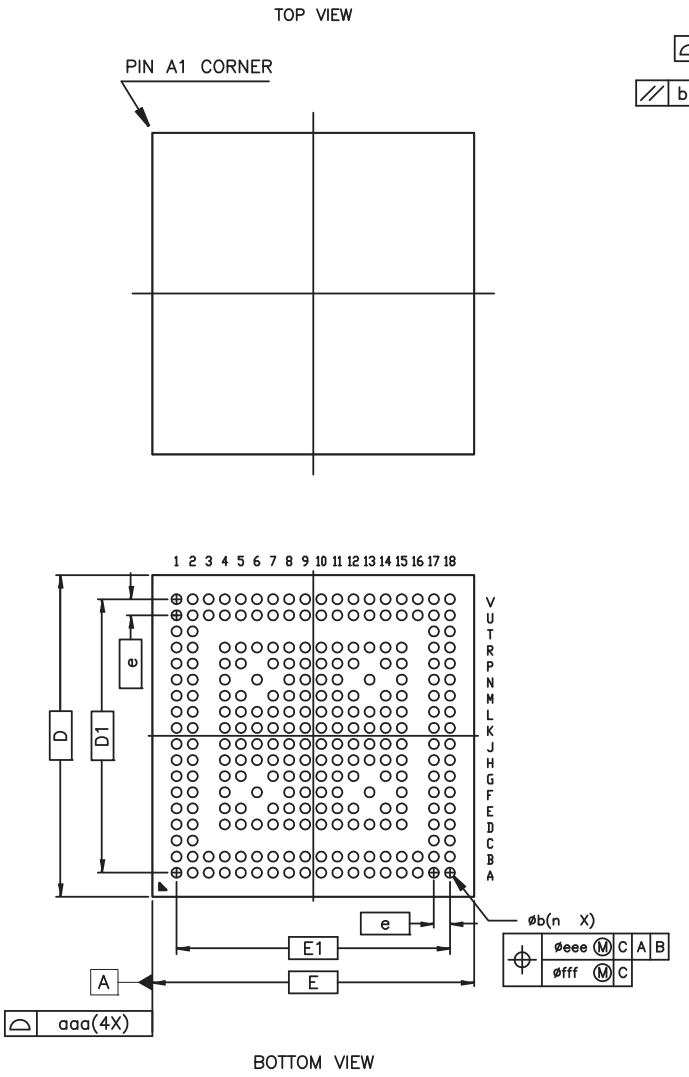
JEDEC Drawing Reference	NA
J-STD-609 Classification	e8

Table 63-4. 289-ball LFBGA Package Information

Ball Land	0.450 mm +/-0.05
Nominal Ball Diameter	0.4 mm
Solder Mask Opening	0.350 mm +/-0.05
Solder Mask Definition	SMD
Solder	OSP

63.2 256-ball TFBGA Mechanical Characteristics

Figure 63-2. 256-ball TFBGA Package



		Symbol	Common Dimensions		
			MIN.	NOM.	MAX.
Package :			TFBGA		
Body Size:	X	E		8.000	
	Y	D		8.000	
Ball Pitch :		e		0.400	
Total Thickness :		A			1.050
Mold Thickness :		M		0.530 Ref.	
Substrate Thickness :		S		0.260 Ref.	
Ball Diameter :				0.250	
Stand Off :		A1	0.110	–	0.210
Ball Width :		b	0.200	–	0.300
Package Edge Tolerance :		aaa		0.100	
Mold Parallelism :		bbb		0.100	
Coplanarity:		ddd		0.080	
Ball Offset (Package) :		eee		0.150	
Ball Offset (Ball) :		fff		0.080	
Ball Count :		n		256	
Edge Ball Center to Center :	X	E1		6.800	
	Y	D1		6.800	

Table 63-5. 256-ball TFBGA Package Characteristics

Moisture Sensitivity Level	3

Table 63-6. Device and 256-ball TFBGA Package Weight

110.3	mg

Table 63-7. Package Reference

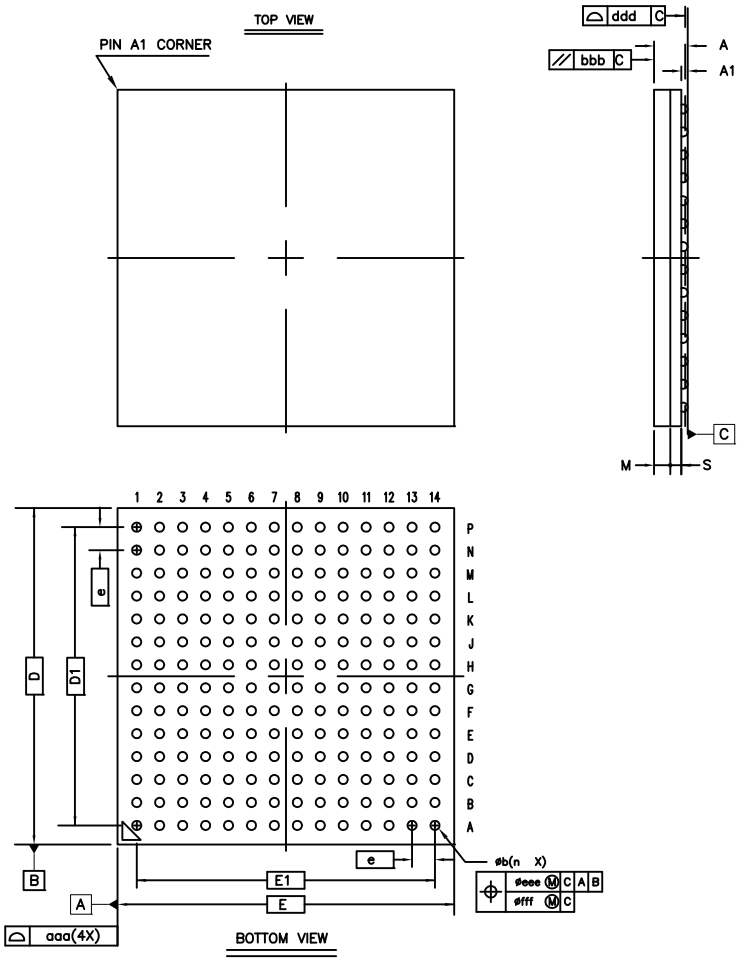
JEDEC Drawing Reference	NA
J-STD-609 Classification	e8

Table 63-8. 256-ball TFBGA Package Information

Ball Land	0.350 mm +/-0.05
Nominal Ball Diameter	0.25 mm
Solder Mask Opening	0.250 mm +/-0.05
Solder Mask Definition	SMD
Solder	OSP

63.3 196-ball TFBGA Mechanical Characteristics

Figure 63-3. Mechanical Overview of the 196-ball TFBGA Package



		Symbol	Common Dimensions		
			MIN.	NOM.	MAX.
Package :			TFBGA		
Body Size:	X	E	11.000		
	Y	D	11.000		
Ball Pitch :		e	0.750		
Total Thickness :		A			1.200
Mold Thickness :		M	0.530 Ref.		
Substrate Thickness :		S	0.360 Ref.		
Ball Diameter :			0.300		
Stand Off :		A1	0.160		0.260
Ball Width :		b	0.270		0.370
Package Edge Tolerance :		aaa	0.100		
Mold Parallelism :		bbb	0.100		
Coplanarity:		ddd	0.080		
Ball Offset (Package) :		eee	0.150		
Ball Offset (Ball) :		fff	0.080		
Ball Count :		n	196		
Edge Ball Center to Center :	X	E1	9.750		
	Y	D1	9.750		

Table 63-9. 196-ball TFBGA Package Characteristics

Moisture Sensitivity Level	3

Table 63-10. Device and 196-ball TFBGA Package Weight

234.2	mg

Table 63-11. Package Reference

JEDEC Drawing Reference	NA
J-STD-609 Classification	e8

Table 63-12. 196-ball TFBGA Package Information

Ball Land	0.350 mm +/-0.05
Nominal Ball Diameter	0.3 mm

Table 63-12. 196-ball TFBGA Package Information (Continued)

Solder Mask Opening	0.275 mm +/-0.30
Solder Mask Definition	SMD
Solder	OSP

64. Schematic Checklist

The schematic checklist provides the user with the requirements regarding the different pin connections that must be considered before starting any new board design as well as information on the minimum hardware resources required to quickly develop an application with the SAMA5D2. It does not consider PCB layout constraints.

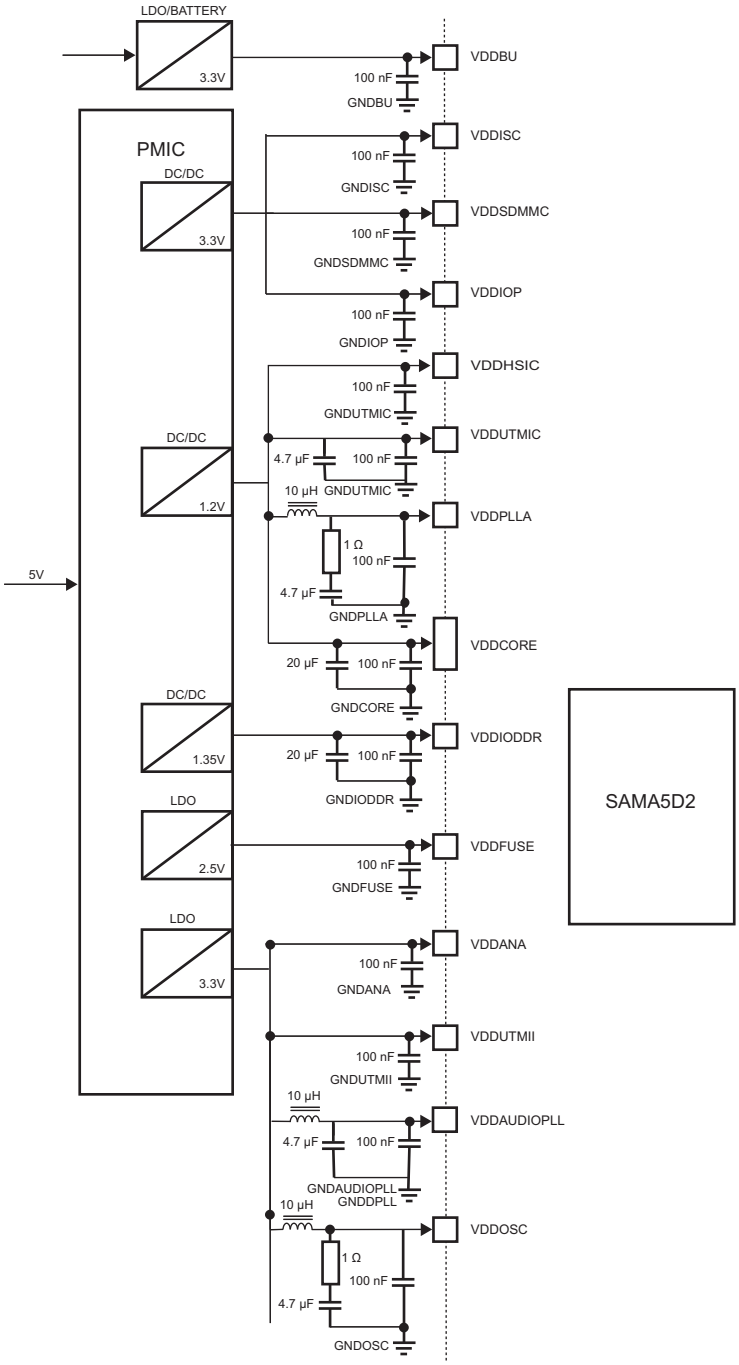
It also provides recommendations regarding low-power design constraints to minimize power consumption.

This information is not intended to be exhaustive. Its objective is to cover as many configurations of use as possible.

64.1 Power Supply

CAUTION: The board design must comply with the powerup and powerdown sequence guidelines provided in the datasheet to guarantee reliable operation of the device.

Figure 64-1. 1.2V, 1.35V/1.5V, 2V, 2.5V, 3.3V Power Supplies Schematics(1)



Note: 1. These values are given only as typical examples.

Table 64-1. Power Supply Connections

Signal Name	Recommended Pin Connection	Description
VDDCORE	1.08V to 1.32V Decoupling/Filtering capacitors (10 µF and 100 nF)[1][2]	Powers the core Supply ripple must not exceed 10 mVrms.
VDDPLLA	1.1V to 1.32V Decoupling capacitor (100 nF)[1][2]	Powers the PLLA cell. The VDDPLLA power supply pin draws small current, but it is noise-sensitive. Care must be taken in VDDPLLA power supply routing, decoupling and also on bypass capacitors. Supply ripple must not exceed 10 mVrms.
VDDIODDR	1.14V to 1.30V 1.283V to 1.45V 1.425V to 1.575V 1.7V to 1.95V Decoupling/Filtering capacitors (10 µF and 100 nF)[1][2]	Powers LPDDR2-LPDDR3 interface Powers DDR3L interface Powers DDR3 interface Powers LPDDR1-DDR2 interface Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
VDDISC	1.65V to 3.6V Decoupling capacitor (100 nF)[1][2]	Powers the ISC Interface I/O lines.
VDDIOP0,1,2	1.65V to 3.6V Decoupling capacitors (100 nF)[1][2]	Powers the peripherals I/O lines. Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
VDDBU	1.65V to 3.6V Decoupling capacitor (100 nF)[1][2]	Powers the Slow Clock oscillator, the internal 64 kHz RC and a part of the System Controller. Must be established first. Supply ripple must not exceed 30 mVrms.
VDDUTMIC	1.1V to 1.32V Decoupling capacitors (100 nF)[1][2]	DC Supply UTMI Phy (Core) and PLL UTMI Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
VDDUTMII	3.0V to 3.6V Decoupling capacitor (100 nF)[1][2]	DC Supply UTMI Phy (Interface) Decoupling/filtering capacitors must be added to improve startup stability and reduce source voltage drop.
VDDHSIC	1.1V to 1.3V Decoupling capacitors (100 nF)[1][2]	DC Supply HSIC Phy
VDDOSC	1.65V to 3.6V Decoupling/Filtering RLC circuit[1]	Powers the main oscillator cell. The VDDOSC power supply pin is noise-sensitive. Care must be taken in VDDOSC power supply routing, decoupling and also on bypass capacitors. Supply ripple must not exceed 30 mVrms.
VDDSDMMC	1.65V to 3.6V Decoupling capacitor (100 nF)[1][2]	Powers the SDMMC I/O lines.
VDDANA	1.65V to 3.6V Decoupling capacitor (100 nF)[1][2]	Powers the analog parts.
VDDFUSE	2.25V to 2.75V Decoupling capacitor (100 nF)[1][2]	Powers the fuse box for programming. VDDFUSE must not be left floating.

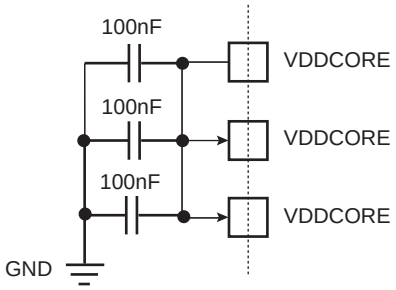
Table 64-1. Power Supply Connections (Continued)

Signal Name	Recommended Pin Connection	Description
VDDAUDIOPLL	3.0V to 3.6V Decoupling capacitor (100 nF)[1][2]	Powers the Audio PLL.
GNDCORE	Core Chip ground	GNDCORE pins are common to VDDCORE and VCCCORE pins. GNDCORE pins should be connected as shortly as possible to the system ground plane.
GNDPLLA	PLLA cell ground	GNDPLL pin is provided for VDDPLLA pins. GNDPLL pin should be connected as shortly as possible to the system ground plane.
GNDIODDR	DDR2/LPDDR1/LPDDR2/DDR3/LPDDR3 interface I/O lines ground	GNDIODDR pins should be connected as shortly as possible to the system ground plane.
GNDISC	VDDISC ground	GNDISC pins are common to VDDISC pins. GNDISC pins should be connected as shortly as possible to the system ground plane.
GNDIOP0,1,2	Peripherals and ISC I/O lines ground	GNDIOPx pins are common to VDDIOPx pins. GNDIOP pins should be connected as shortly as possible to the system ground plane.
GNDBU	Backup ground	GNDBU pin is provided for VDDBU pins. GNDBU pin should be connected as shortly as possible to the system ground plane.
GNDUTMIC	VDDUTMIC and VDDHSIC ground	GNDUTMIC pins are common to VDDUTMIC and VDDHSIC pins. GNDUTMIC pins should be connected as shortly as possible to the system ground plane.
GNDUTMII	UDPHS and UHPHS UTMI+ Core and Interface, and PLL UTMI ground	GNDUTMII pins are common to VDDUTMII and VDDUTMIC pins. GNDUTMII pins should be connected as shortly as possible to the system ground plane.
GNDOSC	Oscillator ground	GNDOSC pin is provided for VDDOSC pins. GNDOSC pin should be connected as shortly as possible to the system ground plane.
GNDSDMMC	SDMMC ground	SDMMC pins are common to VDDSDMMC pins. GNDSDMMC pins should be connected as shortly as possible to the system ground plane.
GNDANA	Analog ground	GNDANA pins are common to VDDANA pins. GNDANA pins should be connected as shortly as possible to the system ground plane.
GNDFUSE	Fuse box ground	GNDFUSE pins are common to VDDFUSE pins. GNDFUSE pins should be connected as shortly as possible to the system ground plane.

Table 64-1. Power Supply Connections (Continued)

Signal Name	Recommended Pin Connection	Description
GNDAUDIOPLL GNDDPLL	Audio PLL ground	GNDAUDIOPLL and GNDDPLL pins are common to VDDAUDIOPLL. GNDAUDIOPLL and GNDDPLL pins should be connected as shortly as possible to the system ground plane.

Note: 1. These values are given only as typical examples.
2. Decoupling capacitors must be connected as close as possible to the microprocessor and on each relevant pin.



For more information, see Table 62-37 “VDDCORE Power-On Reset Characteristics”.

64.2 Power-On Reset

The SAMA5D2 embeds several Power-On Resets (PORs) to ensure the power supply is established when the reset is released. These PORs are dedicated to VDDBU, VDDIOP and VDDCORE respectively.

64.3 Shutdown Considerations

When pin SHDN is asserted, NRST must be maintained to ‘L’ prior to remove the power supplies. After a delay of five SLCK periods, VDDPLL, then VDDCORE, then VDDIOP and VDDANA can be removed. At last, other power supplies can be removed.

VDDBU must never be removed when other supplies are present.

In case VDDBU is provided by a battery and the battery is discharged and reach the POR threshold (~1.4/1.6V), the SHDN pin MUST BE tied to 0 (reset state must be “0” level), in order to avoid restarting the external regulator if reset state is “1”.

64.4 Wakeup Considerations

When SHDN is rising, NRST is to be maintained to ‘L’ prior to establish the power supplies. Then VDDIOP and VDDANA are to be established, followed by VDDCORE and VDDPLL. At the end, other power supplies can be established. After a delay of five SLCK periods, the user can assert NRST to ‘H’ and make the system wake up. Asynchronous partial wakeup is also achievable using the RXLP or analog comparator (“RXLP” and “Analog comparator”).

64.5 Clock, Oscillator and PLL

Table 64-2. Clock, Oscillator and PLL Connections

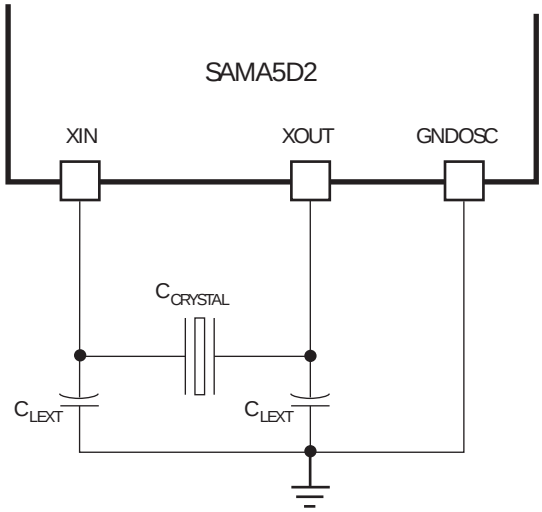
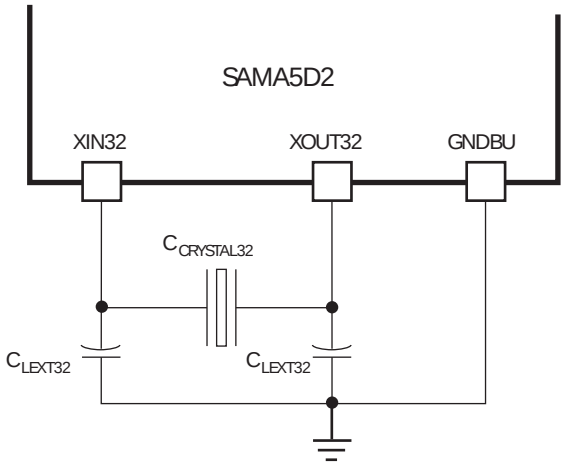
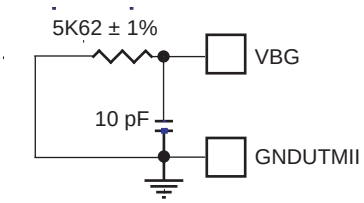
	Signal Name	Recommended Pin Connection	Description
	XIN XOUT 12 MHz Main Oscillator in Normal Mode	Crystals between 8 and 16 MHz USB High Speed (not Full Speed) Host and Device peripherals need a 12 MHz clock. Capacitors on XIN and XOUT (Crystal Load Capacitance dependent)	Crystal Load Capacitance to check ($C_{CRYSTAL}$)  Refer to Section 62. “Electrical Characteristics”.
	XIN XOUT 12 MHz Main Oscillator in Bypass Mode	XIN: external clock source XOUT: can be left unconnected USB High speed (not Full Speed) Host and Device peripherals need a 12 MHz clock.	Square wave signal, high level = VDDOSC External clock source up to 50 MHz Duty Cycle: 40 to 60% Refer to Section 62. “Electrical Characteristics”.
	XIN XOUT 12 MHz Main Oscillator Disabled	XIN: can be left unconnected XOUT: can be left unconnected USB High Speed (not Full Speed) Host and Device peripherals need a 12 MHz clock.	Typical nominal frequency 12 MHz (Internal 12 MHz RC Oscillator) Duty Cycle: 45 to 55% Refer to Section 62. “Electrical Characteristics”

Table 64-2. Clock, Oscillator and PLL Connections (Continued)

	Signal Name	Recommended Pin Connection	Description
	XIN32 XOUT32 Slow Clock Oscillator	32.768 kHz Crystal Capacitors on XIN32 and XOUT32 (Crystal Load Capacitance dependent)	Crystal load capacitance to check ($C_{CRYSTAL32}$)  Refer to Section 62. “Electrical Characteristics”.
	XIN32 XOUT32 Slow Clock Oscillator in Bypass Mode	XIN32: external clock source XOUT32: can be left unconnected	Square wave signal, high level = VDDBU External clock source up to 44 kHz Duty Cycle: 40 to 60% Refer to Section 62. “Electrical Characteristics”.
	XIN32 XOUT32 Slow Clock Oscillator Disabled	XIN32: can be left unconnected XOUT32: can be left unconnected	Typical nominal frequency 32 kHz (internal 64 kHz RC oscillator) Duty Cycle: 45 to 55% Refer to Section 62. “Electrical Characteristics”.
	VBG	0.9–1.1V(2)	Bias Voltage Reference for USB To reduce as much as possible the noise on VBG pin, check the layout considerations below: - VBG path as short as possible - Ground connection to GNDUTMII  VBG can be left unconnected if USB is not used. Refer to Section 4. “Signal Description”.

64.5.1 How to Define the Oscillator Load Capacitance

The load capacitance is the equivalent capacitor value that circuit must “show” to the crystal in order to oscillate at the target frequency. The load is set by two external capacitors placed on each side of the crystal to gnd.

The load of the crystal is the external capacitor value divided by two as it is represented in the figure below (in this case 2 * C_{LOAD} includes ALL the parasitics capacitors).

The C_{LOAD} value must be specified by the crystal manufacturer.

Parameters such as the parasitics of internal ASIC due to routing, IO pads, package and board must be calculated in order to reach the crystal load (refer to Section 62.7 “Oscillator Characteristics” to Section 62.7.3 “32.768 kHz Crystal Oscillator Characteristics”).

The external capacitor value can be determined by using the following formula:

$$C_{EXT} = (2 * C_{LOAD}) - C_{BOARD} - C_{PACKAGE} - C_{PAD} - C_{ROUTING} - (C_{PARA} * 2)$$

where:

- C_{EXT}: external capacitor value which must be soldered from aio33xin to gnd and from aio33xout to gnd
- C_{LOAD}: crystal-targeted load. Refer to C_{LOAD} parameter in the crystal electrical specification.
- C_{BOARD}: external calculated (or measured) value due to board parasitics
- $C_{PACKAGE}$: parasitics capacitance due to package and bonding
- C_{PAD}: parasitics capacitance of the I/O pad used for internal connection
- $C_{ROUTING}$: parasitics due to internal chip routing
- C_{PARA}: internal load parasitic due to internal structure. Refer to C_{PARA} parameter in Section 62.7 “Oscillator Characteristics”.

Table 64-3. Main Oscillator Load Capacitance

Oscillator	12 MHz < Frequency < 24 MHz	Frequency = 24 MHz
Main Oscillator	12.5 pF < C_{LOAD} < 17.5 pF	10 pF < C_{LOAD} < 12.5 pF

Table 64-4. 32.768 kHz Oscillator Load Capacitance

Oscillator	Frequency
32.768 kHz Oscillator	6 pF < C_{LOAD} < 12.5 pF

In our case, the minimum targeted load for the 32.768 kHz oscillator is 6 pF. The typical parasitic load of the oscillator is 0.5 pF. If routed + externals capacitances (package + board + external soldered + internal connection and internal pads parasitics) are about 1 pF, the external load must be 6 pF – 0.5 pF - 1 pF = 4.5 pF, which means that 9 pF is the target value (9 pF from aio33xin to gnd and 11 pF from aio33xout to gnd).

If a 12.5 pF load is targeted, the externals capacitances must be 12.5 pF - 0.5 pF - 1 pF = 11 pF, which is 22 pF (22 pF from aio33xin to gnd and 22 pF from aio33xout to gnd).

Example 1: Crystal C_{LOAD}= 6 pF
Cl = 6 pF then Cxin_gnd_total = 12 pF and Cxout_gnd_total = 12 pF
9 pF external capacitor needed

Example 2: Crystal C_{LOAD} = 12.5 pF
Cl = 12.5 pF then Cxin_gnd = 25 pF and Cxout_gnd = 25 pF
22 pF external capacitor needed

64.6 ICE and JTAG

Table 64-5. ICE and JTAG Connections(1)

	Signal Name	Recommended Pin Connection	Description
	TCK	Pull-up (100 kΩ)(2) If Debug mode is not required, this pin can be used as GPIO.	This pin is a Schmitt trigger input. Internal pull-up resistor to V_{DDIOP} (100 kΩ).
	TMS	Pull-up (100 kΩ)(2) If Debug mode is not required, this pin can be used as GPIO.	This pin is a Schmitt trigger input. Internal pull-up resistor to V_{DDIOP} (100 kΩ).
	TDI	Pull-up (100 kΩ)(2) If Debug mode is not required, this pin can be used as GPIO.	This pin is a Schmitt trigger input. Internal pull-up resistor to V_{DDIOP} (100 kΩ).
	TDO	Floating If Debug mode is not required, this pin can be used as GPIO.	Output driven at up to V_{DDIOP}.
	NTRST	Refer to the pin description section. If Debug mode is not required, this pin can be used as GPIO.	This pin is a Schmitt trigger input. Internal pull-up resistor to V_{DDIOP} (100 kΩ).
	JTAGSEL	In harsh environments(3), it is strongly recommended to tie this pin to GNDBU if not used or to add an external low-value resistor (such as 1 kΩ).	Internal pull-down resistor to GNDBU (15 kΩ). Must be tied to V_{DDBU} to enter JTAG Boundary Scan.

Notes:
1. It is recommended to establish accessibility to a JTAG connector for debug in any case.
2. These values are given only as typical examples.
3. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

64.7 Reset and Test

Table 64-6. Reset and Test Connections

	Signal Name	Recommended Pin Connection	Description
	NRST	Application-dependent Can be connected to a pushbutton for hardware reset.	NRST pin is a Schmitt trigger input. No internal pull-up resistor.
	TST	In harsh environments(1), it is strongly recommended to tie this pin to GNDBU to add an external low-value resistor (such as 10 kΩ).	This pin is a Schmitt trigger input. Internal pull-down resistor to GNDBU (15 kΩ).

Note: 1. In a well-shielded environment subject to low magnetic and electric field interference, the pin may be left unconnected. In noisy environments, a connection to ground is recommended.

64.8 Shutdown/Wakeup Logic

Table 64-7. Shutdown/Wakeup Logic Connections

	Signal Name	Recommended Pin Connection	Description
	SHDN	Application-dependent A typical application connects pin SHDN to the shutdown input of the DC/DC Converter providing the main power supplies.	This pin is a push-pull output. SHDN pin is driven low to GNDBU by the Shutdown Controller (SHDWC).
	WKUP	0 V to V_{DDBU}	This pin is an input only. WKUP behavior can be configured through the Shutdown Controller (SHDWC).

64.9 Parallel Input/Output (PIO)

Table 64-8. PIO Connections

	Signal Name	Recommended Pin Connection	Description
	PAx PBx PCx PDx	Application-dependent	All PIOs are pulled up inputs (100 kΩ) at reset except those which are multiplexed with the Address Bus signals that require to be enabled as peripherals: In Section 5. “Package and Pinout”, refer to the column ‘Reset State’ of the Pin Description table. Schmitt trigger on all inputs. To reduce power consumption if not used, the relevant PIO can be configured as an output, driven at ‘0’ with internal pull-up disabled.

64.10 Analog-to-Digital Converter (ADC)

Table 64-9. ADC Connections

	Signal Name	Recommended Pin Connection	Description
	ADVREF	3.3 V to VDDANA Decoupling/filtering capacitors Application-dependent	ADVREF is a pure analog input. To reduce power consumption if the ADC is not used, connect ADVREF to GNDANA.

64.11 External Bus Interface (EBI)

Table 64-10. EBI Connections

	Signal Name	Recommended Pin Connection	Description
	D0–D15	Application-dependent	Data Bus (D0 to D15) All data lines are pulled up inputs at reset.
	A0–A25	Application-dependent	Address Bus (A0 to A25) All address lines are pulled up inputs at reset.

Table 64-11 and Table 64-12 detail the connections to be applied between the EBI pins and the external devices for each memory controller.

Table 64-11. EBI Pins and External Static Devices Connections

Signals: EBI_	Pins of the Interfaced Device		
	8-bit Static Device	2 x 8-bit Static Devices	16-bit Static Device
Controller	**SMC (Static Memory Controller)**		
D0–D7	D0–D7	D0–D7	D0–D7
D8–D15	–	D8–D15	D8–D15
A0/NBS0	A0	–	NLB
A1	A1	A0	A0
A2–A22	A[2:22]	A[1:21]	A[1:21]
A23–A25	A[23:25]	A[22:24]	A[22:24]
NCS0	CS	CS	CS
NCS1	CS	CS	CS
NCS2	CS	CS	CS
NCS3/NANDCS	CS	CS	CS
NRD/NANDOE	OE	OE	OE
NWE/NWR0/NANDWE	WE	WE(1)	WE
NWR1/NBS1	–	WE(1)	NUB

Note: 1. NWR0 enables lower byte writes. NWR1 enables upper byte writes.

Table 64-12. EBI Pins and NAND Flash Device Connections

Signals: EBI_	Pins of the Interfaced Device	
	8-bit NAND Flash	16-bit NAND Flash
Controller	**NFC (NAND Flash Controller)**	
D0–D7	NFD0–NFD7	NFD0–NFD7
D8–D15	–	NFD8–NFD15
A21/NANDALE	ALE	ALE
A22/NANDCLE	CLE	CLE
NRD/NANDOE	RE	RE
NWE/NWR0/NANDWE	WE	WE

Table 64-12. EBI Pins and NAND Flash Device Connections (Continued)

Signals: EBI_	Pins of the Interfaced Device	
	8-bit NAND Flash	16-bit NAND Flash
Controller	**NFC (NAND Flash Controller)**	
NCS3/NANDCS	CE	CE
NANDRDY	R/$\overline{B}$	R/$\overline{B}$
A0/NBS0	–	–
A1–A20	–	–
A23–A25	–	–
NWR1/NBS1	–	–
NCS0	–	–
NCS1	–	–
NCS2	–	–
NWAIT	–	–

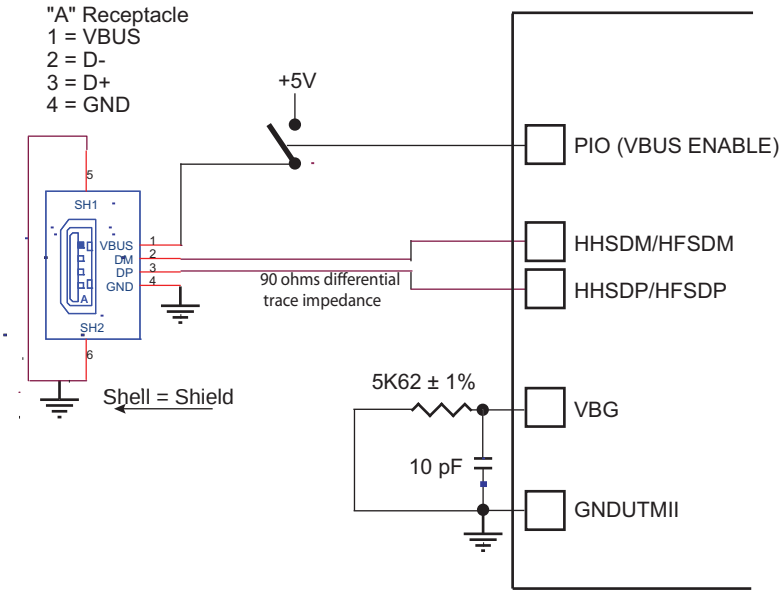
64.12 USB High-Speed Host Port (UHPHS) / USB High-Speed Device Port (UDPHS)

Table 64-13. UHPHS/UDPHS Connections

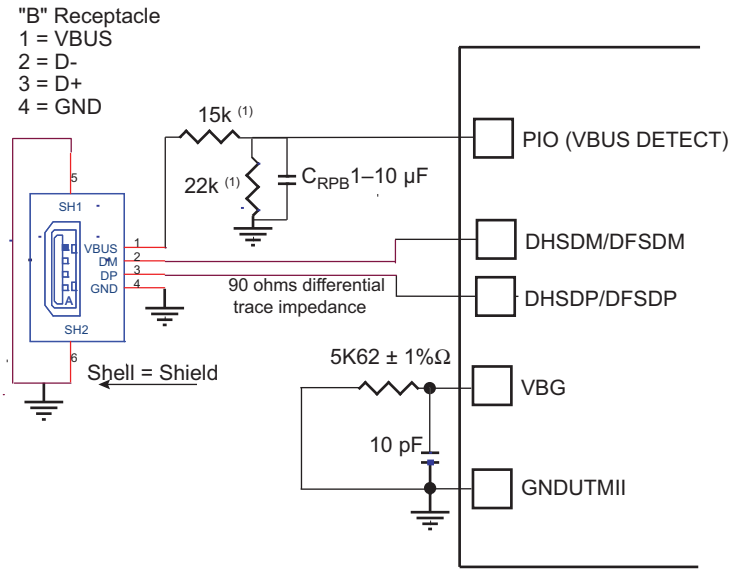
	Signal Name	Recommended Pin Connection	Description
UHPHS	HHSDPA/DHSDPB[1] HHSDMA/DHSDMB[1]	Application-dependent[2][3]	Pull-down output at reset
UDPHS	HHSDP/HHSDM	Application-dependent[2]	Pull-down output at reset
HSIC	HHSTROBE/HHDATA	Application-dependent[2]	Pull-down output at reset

Notes: 1. UDPHS shares Port A with UHPHS.

2. Example of a USB High Speed Host connection: refer to section Section 39. “USB Host High Speed Port (UHPHS)”.



3. Typical USB High Speed Device connection: refer to Section 38. “USB High Speed Device Port (UDPHS)”.



Note: 1. The values shown on the 22 kΩ and 15 kΩ resistors are only valid with 3.3-V supplied PIOs.

64.13 Boot Program Hardware Constraints

Refer to Section 15. “Standard Boot Strategies” for more details on the boot program.

64.14 Layout and Design Constraints

Note: All PIOs shared, multiplexed, connected to various components and connectors must be connected through serial resistors 22R and 39R for clock signals. The resistors must be populated as close as possible to the SAMA5D2.

64.14.1 General Considerations

This chapter provides routing guidelines for layout and design of a printed circuit board using high-speed interfaces, Serial, Ethernet and USB 2.0. The signal integrity rules for high-speed interfaces need to be considered. In fact, it is highly recommended that the board design be simulated to determine optimum layout for signal integrity and quality. Keep in mind that this document can only highlight the most important issues that should be considered when designing the SAMA5D2 board. The designer has to take into account the corresponding information (specification, design guidelines, etc.) contained in the documentation of all other devices that are to be implemented on board.

64.14.2 Considerations for High-Speed Differential Interfaces

The following is a list of suggestions for designing with high-speed differential signals.

This should help implementing these interfaces while providing maximum SAMA5D2 board performance.

- Use controlled impedance PCB traces that match the specified differential impedance.
- Keep the trace lengths of the differential signal pairs as short as possible.
- The differential signal pair traces should be trace-length matched and the maximum trace-length mismatch should not exceed the specified values. Match each differential pair per segment.
- Maintain parallelism and symmetry between differential signals with the trace spacing needed to achieve the specified differential impedance.
- Maintain maximum possible separation between the differential pairs and any high-speed clocks/periodic signals (CMOS/TTL) and any connector leaving the PCB (such as I/O connectors, control and signal headers, or power connectors).
- Route differential signals on the signal layer nearest to the ground plane using a minimum of vias and corners. This will reduce signal reflections and impedance changes. Use GND stitching vias when changing layers.
- It is best to put CMOS/TTL and differential signals on different layers which should be isolated by the power and ground planes.
- Avoid tight bends. When it becomes necessary to turn 90°, use two 45° turns or an arc instead of making a single 90° turn.
- Do not route traces under crystals, crystal oscillators, clock synthetizers, magnetic devices or ICs that use, and/or generate, clocks.
- Stubs on differential signals should be avoided due to the fact that stubs cause signal reflections and affect signal quality.
- Keep the length of high-speed clock and periodic signal traces that run parallel to high-speed signal lines at a minimum to avoid crosstalk. Based on EMI testing experience, the minimum suggested spacing to clock signals is 50 mil.
- Use a minimum of 20 mil spacing between the differential signal pairs and other signal traces for optimal signal quality. This helps to prevent crosstalk.
- Route all traces over continuous planes (VCC or GND) with no interruptions.
- Avoid crossing over anti-etch if at all possible. Crossing over anti-etch (split planes) increases inductance and radiation levels by forcing a greater loop area.

64.14.3 DDR Layout and Design Considerations

Refer to document “SAMA5D2 Layout Recommendations”, Atmel literature No. 44041.

64.14.4 e.MMC routing

Refer to the Micron Technical Note TN-FC-35: e•MMC PCB Design Guide. This document is intended as guide for PCB designers using Micron e•MMC devices and discusses the primary issues affecting design and layout.

64.14.5 USB Trace Routing Guidelines

- Maintain parallelism between USB differential signals with the trace spacing needed to achieve 90-ohm differential impedance. Deviations will normally occur due to package breakout and routing to connector pins. Just ensure the amount and lengths of the deviations are kept to the minimum possible.
- Use an impedance calculator to determine the trace width and spacing required for the specific board stack-up being used. Example: Layer Stacking, 7.5-mil traces with 7.5-mil spacing results in approximately 90-ohm differential trace impedance.
- Minimize the length of high-speed clock and periodic signal traces that run parallel to high-speed USB signal lines, to minimize crosstalk. Based on EMI testing experience, the minimum suggested spacing to clock signals is 50 mils.
- Based on simulation data, use 20-mil minimum spacing between high-speed USB signal pairs and other signal traces for optimal signal quality. This helps to prevent crosstalk.

Table 64-14. USB Trace Routing Guidelines

Parameter	Trace Routing
Signal length allowance for the SAMA5D2	14.0 inches Valid for a damping value of the PCB trace of 0.11 dB/inch @ 0.4 GHz (common value for FR-4 based material)
Differential impedance	90 ohms +/-15%
Single-ended impedance	45 ohms +/-10%
Trace width (W)	5 mils (microstrip routing)
Spacing between differential pairs (intra-pair)	6 mils (microstrip routing)
Spacing between pairs (inter-pair)	Min. 20 mils
Spacing between differential pairs and high-speed periodic signals	Min. 50 mils
Spacing between differential pairs and low-speed non-periodic signals	Min. 20 mils
Length matching between differential pairs (intra-pair)	150 mils
Reference plane	GND referenced preferred
Spacing from edge of plane	Min. 40 mils
Vias usage	Try to minimize the number of vias

64.14.6 QSPI Pull-up Resistors

The ROM code **removes** the internal pull-up resistors when it configures PIO controller to mux the QSPI controller I/O lines. Therefore the probing step may fail if the Quad I/O mode of the memory has not been enabled yet and if this memory does not embed internal pull-up resistor on #HOLD or #RESET pin.

This is why we recommend to add external pull-up resistors if needed on the four I/O data lines MOSI/IO0, MISO/IO1, #WP/IO2 and #HOLD/IO3.

Another solution is to update the Quad Enable non-volatile bit in the relevant register to reassign #WP and #HOLD/#RESET pins to functions IO2 and IO3.

65. Marking

All devices are marked with the Atmel logo and the ordering code.

Additional marking is as follows:

where

- “YY”: Manufactory year
- “WW”: Manufactory week
- “C”: Assembly country code
- “V”: Revision
- “XXXXXX”: Lot number

66. Ordering Information

Table 66-1. SAMA5D2 Ordering Information

Ordering Code	MRL	Package	Carrier Type	Operating Temperature Range
ATSAMA5D22A-CU	A	TFBGA196	Tray	-40°C to 85°C
ATSAMA5D22A-CUR	A		Tape and reel	
ATSAMA5D24A-CU	A	TFBGA256	Tray	
ATSAMA5D24A-CUR	A		Tape and reel	
ATSAMA5D27A-CU	A	LFBGA289	Tray	
ATSAMA5D27A-CUR	A		Tape and reel	
ATSAMA5D28A-CU[1]	A		Tray	
ATSAMA5D21B-CU	B	TFBGA196	Tray	-40°C to 85°C
ATSAMA5D21B-CUR	B		Tape and reel	
ATSAMA5D22B-CN	B		Tray	-40°C to 105°C
ATSAMA5D22B-CNR	B		Tape and reel	
ATSAMA5D22B-CU	B		Tray	-40°C to 85°C
ATSAMA5D22B-CUR	B		Tape and reel	
ATSAMA5D23B-CN	B		Tray	-40°C to 105°C
ATSAMA5D23B-CNR	B		Tape and reel	
ATSAMA5D23B-CU	B		Tray	-40°C to 85°C
ATSAMA5D23B-CUR	B		Tape and reel	
ATSAMA5D24B-CU	B	TFBGA256	Tray	-40°C to 85°C
ATSAMA5D24B-CUR	B		Tape and reel	
ATSAMA5D26B-CN	B	LFBGA289	Tray	-40°C to 105°C
ATSAMA5D26B-CNR	B		Tape and reel	
ATSAMA5D26B-CU	B		Tray	-40°C to 85°C
ATSAMA5D26B-CUR	B		Tape and reel	
ATSAMA5D27B-CN	B		Tray	-40°C to 105°C
ATSAMA5D27B-CNR	B		Tape and reel	
ATSAMA5D27B-CU	B		Tray	-40°C to 85°C
ATSAMA5D27B-CUR	B		Tape and reel	
ATSAMA5D28B-CN	B		Tray	-40°C to 105°C
ATSAMA5D28B-CNR	B		Tape and reel	
ATSAMA5D28B-CU	B		Tray	-40°C to 85°C
ATSAMA5D28B-CUR	B		Tape and reel	

Note: 1. To order ATSAMA5D28 and ATSAMA5D23, contact an Atmel Sales Representative.

67. Errata

Errata is described in the following sections:

Section 67.1 “Errata - SAMA5D2 MRL-B Parts”

Section 67.2 “Errata - SAMA5D2 MRL-A Parts”

67.1 Errata - SAMA5D2 MRL-B Parts

This section describes errata relevant to the devices listed in Table 67-1.

Table 67-1. SAMA5D2 MRL-B Parts

Device Name
ATSAMA5D21B
ATSAMA5D22B
ATSAMA5D23B
ATSAMA5D24B
ATSAMA5D26B
ATSAMA5D27B
ATSAMA5D28B

67.1.1 ROM Code: SDMMC0 and SDMMC1 boot issue

Issue: **The card detect pin is not correctly sampled in the ROM code, which leads to a nondeterministic boot ability on the SDMMC0/SDMMC1 interfaces (SDCard or eMMC).**

Workaround: Use another boot media (e.g., serial flash) for the first level boot, and either deactivate the boot on SDMMC0/1 in the Boot Configuration Word in the Fuse area or remove any bootable program stored in the eMMC or SDCard connected to the chip at startup.

67.1.2 SDMMC Software ‘Reset for All’ Command

Issue: **Software ‘Reset for All’ command is not guaranteed**

The software ‘Reset for All’ command is not guaranteed, and some registers of the host controller may not properly reset. The setting of the different registers must be checked before reinitializing the SD card.

Workaround: None

67.1.3 FLEXCOM SMBUS

Issue: **FLEXCOM SMBUS alert signalling is not functional**

The TWI function embedded in the FLEXCOM does not support SMBUS alert signal management.

Workaround: If this signal is mandatory in the application, the user can use one of the standalone TWIs (TWIHS0, TWIHS1) supporting the SMBUS alert signaling.

67.1.4 TWI/TWIHS Clear Command

Issue: **The TWI/TWIHS Clear command does not work**

Bus reset using the “CLEAR” bit of the TWI/TWIHS control register does not work correctly during a bus busy state..

Workaround: When the TWI master detects the SDA line stuck in low state the procedure to recover is:

1. Reconfigure the SDA/SCL lines as PIO.
2. Try to assert a Logic 1 on the SDA line (PIO output = 1).
3. Read the SDA line state. If the PIO state is a Logic 0, then generate a clock pulse on SCL (1-0-1 transition).
4. Read the SDA line state. If the SDA line = 0, go to Step 3; if SDA = 1, go to Step 5.
5. Generate a STOP condition.
6. Reconfigure SDA/SCL PIOs as peripheral.

67.1.5 SSC TD Output

Issue: **Unexpected delay on TD output**

When SSC is configured with the following conditions:

- RCMR.START = Start on falling edge/Start on Rising edge/Start on any edge,
- RFMR.FSOS = None (input),
- TCMR.START = Receive Start,

an unexpected delay of 2 or 3 system clock cycles is added to the TD output.

Workaround: None

67.1.6 I2SC First Sent Data

Issue: **I2SC first sent data corrupted**

Right after I2SC reset, the first data sent by I2SC controller on the I2SDO line is corrupted. The following data are not affected.

Workaround: None

67.1.7 Quad I/O Serial Peripheral Interface (QSPI)

Issue: **QSPI hangs with long DLYCS**

QSPI hangs if a command is written to any QSPI register during the DLYCS delay. There is no status bit to flag the end of the delay.

Workaround: The field DLYCS defines a minimum period for which Chip Select is deasserted, required by some memories. This delay is generally < 60 ns and comprises internal execution time, arbitration and latencies. Thus, DLYCS must be configured to be slightly higher than the value specified for the slave device. The software must wait for this same period of time plus an additional delay before a command can be written to the QSPI.

67.1.8 Master/Processor Clock Prescaler

Issue: Change of the field PMC_MCKR.PRES is not allowed if Master/Processor Clock Prescaler frequency is too high

PMC_MCKR.PRES cannot be changed if the clock applied to the Master/Processor Clock Prescaler (see “Master Clock Controller”, in Section 30. “Power Management Controller (PMC)”) is greater than 312 MHz (VDDCORE[1.1, 1.32]) and 394 MHz (VDDCORE[1.2, 1.32]).

Workaround:

1. Set PMC_MCKR.CSS to MAIN_CLK.
2. Set PMC_MCKR.PRES to the required value.
3. Change PMC_MCKR.CSS to the new clock source (PLLA_CLK, UPLLCK).

67.2 Errata - SAMA5D2 MRL-A Parts

This section describes errata relevant to the devices listed in Table 67-2.

Table 67-2. SAMA5D2 MRL-A Parts

Device Name
ATSAMA5D22A
ATSAMA5D24A
ATSAMA5D27A
ATSAMA5D28A

67.2.1 ROM Code: Main External Clock Frequency Support for SAM-BA Monitor

Issue: **ROM code v1.1 supports ONLY a 12 and 16 MHz external clock frequency to allow USB connection to be used for SAM-BA Monitor**

Workaround: None

67.2.2 ROM Code: Watchdog after SAM-BA Monitor Connection

Issue: **Watchdog reset occurs when reenabling the watchdog**

When no bootable program is found in an external memory, the Watchdog is disabled just before the ROM Code runs SAM-BA Monitor. The ROM code sets the Watchdog Timer Mode register (WDT_MR) to the value 0x00008000 and then clears the counter value. If a program loaded and executed using the SAM-BA Monitor Go command reenables the watchdog, a watchdog reset is immediately executed whatever the value of the watchdog counter.

Workaround: To avoid any unexpected watchdog reset when reenabling the watchdog, the following sequence has to be performed:

4. Write 0x00000000 in the WDT_MR.
5. Wait for three slow clock cycles.
6. Write the final value in the WDT_MR.

67.2.3 Unique Serial Number

Issue: **The serial number stored in the SFR registers (SFR_SN0 and SFR_SN1) is not correct**

The serial number (SFR_SN0, SFR_SN1) has only 16 bits set. This serial number cannot be used as a 64-bit unique ID.

Workaround: None

67.2.4 Fuse Masking

Issue: **The Partial Fuse Masking function does not work**

The fuse masking function described in Section 54. “Secure Fuse Controller (SFC)” does not work. If the ROM code is used in Secure mode, the overall fuses are masked by the ROM code even if some of them are not used.

Workaround: None

67.2.5 Fuse Writing

Issue: **The first two bits of each 32-bit block of the fuse matrix cannot be written**

The first two bits of each 32-bit block of the fuse matrix cannot be written, so that any word (32 bits) written needs to set to 0 the first two bits of each word (32 bits) of the fuse matrix.

Workaround: None

67.2.6 HSIC Startup

Issue: **At HSIC startup, the strobe default state is wrong**

The strobe line should be at logic state 0 when HSIC is powered ON, and disabled. Currently, powering up the product sets the strobe line at logic state 1 before the HSIC is enabled. In this case, a connected device tries to connect before the HSIC is enabled.

Workaround: Connect the device after the SAMA5D2 has been started.

67.2.7 ADC SleepWalking

Issue: **ADC SleepWalking is not functional**

Workaround: None

67.2.8 ADC Last Channel Low-speed Trigger

Issue: **The last channel can be triggered at low speed but cannot be programmed by the OUT1 field of the RTC. Only the 1-Hz sampling period is available.**

Workaround: None

67.2.9 ADC Trigger Events

Issue: **ADC trigger events RTCOUT0 and RTCOUT1 are not functional**

RTCOUT0 issue leads to ADC Sleepwalking not functional.

RTCOUT1 issue makes the last channel specific measurement trigger work at 1 Hz only.

Workaround: None

67.2.10 ACC Output

Issue: **ACC output connection issue**

The Analog Comparator (ACC) output is not connected to the PWM event line.

Workaround: None

67.2.11 SDMMC Software 'Reset For all' Command

Issue: **Software 'Reset For all' command is not guaranteed**

The software 'Reset For all' command is not guaranteed, and some registers of the host controller may not properly reset. The setting of the different registers must be checked before reinitializing the SD card.

Workaround: None

67.2.12 SDMMC Status Flag INTCLKS

Issue: **Status flag INTCLKS may not work correctly**

When the SDMMC internal clock is disabled (SDMMC_CCR. INTCLKEN = 0) and reenabled after a few cycles (SDMMC_CCR. INTCLKEN = 1), the status flag INTCLKS may get stuck at 0.

Workaround: A delay loop of 6 cycles minimum of the slowest clock (HCLOCK or BASECLK) must be inserted between SDMMC_CCR. INTCLKEN = 0 and SDMMC_CCR. INTCLKEN = 1.

67.2.13 PMC GCLK Fields

Issue: **GCLK fields are reprogrammed unexpectedly**

When configuring a peripheral featuring no GCLK, the GCLK fields (GCKEN, GCKCSS, GCKDIV) of FLEXCOM0 are reconfigured. No other parameter is modified.

Workaround: When accessing a peripheral featuring no GCLK, fill the GCLK fields with the GCLK configuration of FLEXCOM0.

67.2.14 PMC SleepWalking

Issue: **PMC SleepWalking is not functional**

In Ultra-Low Power mode (ULP1) using simultaneously partial wakeup (SleepWalking) and full wakeup (PIOBU used as wakeup pins or internal events RTC, etc.) may not resume from ULP1.

Workaround: None.

67.2.15 CLASSD Peripheral

Issue: **Unexpected offset and noise level in Differential Output mode**

When the CLASSD peripheral is set to Differential Output mode (PWMTYP = 1), a significant output offset and an increased level of noise are observed on the audio outputs. The offset is systematic and is equal to 1/16 of the digital full scale.

Workaround: To avoid the offset, add the opposite offset on the input signal of the CLASSD peripheral.

67.2.16 FLEXCOM SMBUS

Issue: **FLEXCOM SMBUS alert signalling is not functional**

The TWI function embedded in the FLEXCOM does not support SMBUS alert signal management.

Workaround: If this signal is mandatory in the application, the user can use one of the standalone TWIs (TWIHS0, TWIHS1) supporting the SMBUS alert signaling.

67.2.17 TWI/TWIHS Clear Command

Issue: **The TWI/TWIHS Clear command does not work**

Bus reset using the “CLEAR” bit of the TWI/TWIHS control register does not work correctly during a bus busy state..

Workaround: When the TWI master detects the SDA line stuck in low state the procedure to recover is:

1. Reconfigure the SDA/SCL lines as PIO.
2. Try to assert a Logic 1 on the SDA line (PIO output = 1).
3. Read the SDA line state. If the PIO state is a Logic 0, then generate a clock pulse on SCL (1-0-1 transition).
4. Read the SDA line state. If the SDA line = 0, go to Step 3; if SDA = 1, go to Step 5.
5. Generate a STOP condition.
6. Reconfigure SDA/SCL PIOs as peripheral.

67.2.18 MPDDRC t_{FAW}

Issue: **t_{FAW} timing violation**

DDR2/LPDDR2 memory devices with 8 banks have an additional requirement for t_{FAW}: no more than four Activate commands must be issued in any given t_{FAW} period.

Workaround: Increase the value of t_{RRD} to 3 to avoid the issue.

67.2.19 Audio PLL

Issue: **Audio PLL output frequency range**

The frequency range of the AUDIOCORECLK signal (AUDIOPLL output) provided in Table 62-23 “Audio PLL Characteristics” (f_{CORE} parameter) does not comply with the applicable specification.

Workaround: The AUDIOCORECLK signal can be operated from 720 MHz to 790 MHz if the following restricted operating conditions are met:

- Junction temperature (T_j) range: 0°C to +40°C
- VDDCORE/VDDPLL supply range: 1.20V to 1.32V
- Bits <29:28> in register PMC_AUDIO_PLL0 are set to $(01)_2$

67.2.20 SSC TD Output

Issue: **Unexpected delay on TD output**

When SSC is configured with the following conditions:

- RCMR.START = Start on falling edge/Start on Rising edge/Start on any edge,
- RFMR.FSOS = None (input),
- TCMR.START = Receive Start,

an unexpected delay of 2 or 3 system clock cycles is added to the TD output.

Workaround: None

67.2.21 I2SC First Sent Data

Issue: **I2SC first sent data corrupted**

Right after I2SC reset, the first data sent by I2SC controller on the I2SDO line is corrupted. The following data are not affected.

Workaround: None

67.2.22 Quad I/O Serial Peripheral Interface (QSPI)

Issue: QSPI hangs with long DLYCS

QSPI hangs if a command is written to any QSPI register during the DLYCS delay. There is no status bit to flag the end of the delay.

Workaround: The field DLYCS defines a minimum period for which Chip Select is de-asserted, required by some memories. This delay is generally < 60 ns and comprises internal execution time, arbitration and latencies. Thus, DLYCS must be configured to be slightly higher than the value specified for the slave device. The software must wait for this same period of time plus an additional delay before a command can be written to the QSPI.

67.2.23 Master/Processor Clock Prescaler

Issue: Change of the field PMC_MCKR.PRES is not allowed if Master/Processor Clock Prescaler frequency is too high

PMC_MCKR.PRES cannot be changed if the clock applied to the Master/Processor Clock Prescaler (see “Master Clock Controller”, in Section 30. “Power Management Controller (PMC)”) is greater than 312 MHz (VDDCORE[1.1, 1.32]) and 394 MHz (VDDCORE[1.2, 1.32]).

Workaround:

1. Set PMC_MCKR.CSS to MAIN_CLK.
2. Set PMC_MCKR.PRES to the required value.
3. Change PMC_MCKR.CSS to the new clock source (PLLA_CLK, UPLLCK).

68. Revision History

In the table that follows, the most recent version of the document appears first.

Table 68-1. SAMA5D2 Datasheet Rev. 11267E Revision History

Issue Date	Changes
25-Jul-16	Deleted Section 61. “Security Module”.

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History

Issue Date	Changes
12-May-16	Minor formatting and editorial changes throughout
	“Introduction” Updated listed DDR memories
	“Features” Frequency of digital fractional PLL for audio “11.289 MHz” corrected to “11.2896 MHz” “Two 64-bit, 16-channel DMA controllers” changed to “51 DMA Channels including two 16-channel 64-bit Central DMA Controllers”
	Section 1. “Description” Updated description of Low-power modes
	Section 2. “Configuration Summary” “Class D amplifier” changed to “stereo Class D amplifier” Updated text at end of section
	Section 3. “Block Diagram” Figure 3-1 “SAMA5D2 Series Block Diagram”: added ISC_MSK input; updated description of crystal oscillators; “PWMEXTRIG0-1” renumbered to “PWMEXTRG1–2” Added note “See Section 35. “DMA Controller (XDMAC)” for peripheral connections to DMA.”
	Section 4. “Signal Description” Table 4-1 “Signal Description List”: NRST signal function “Microcontroller Reset” changed to “Microprocessor Reset”; “PWMEXTRG0-1” renumbered to “PWMEXTRG1–2”; “Self-refresh mode” changed to “Backup Self-refresh mode” in DDR_CKE comments
	Section 5. “Package and Pinout” Separated content into Section 5.1 “Packages” and Section 5.2 “Pinouts” Table 5-2 “Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A)”: “ADVREFP” corrected to “ADVREF”; “PWMEXTRG0” and “PWMEXTRG1” renumbered to “PWMEXTRG1” and “PWMEXTRG2”; removed empty function cells for primary signals PA30, PA31, and PB0–PB7; removed “SEC, FILTER” from “Reset State” column header; added footnote on reset states Added Table 5-3 “Pin Description (SAMA5D23 pins different from those in SAMA5D21/SAMA5D22)” and Table 5-4 “Pin Description (SAMA5D28B pins different from those in SAMA5D28A)”
	Section 6. “Power Considerations” Table 6-1 “SAMA5D2 Power Supplies”: updated rows VDDUTMIC, VDDHSIC and VDDOSC Section 6.4.1 “VDDBU Power Architecture”: reworded second paragraph and deleted “typically less than 2 µA”
	Section 7. “Memories” Section 7.2.1 “External Bus Interface”: “The slew rates are determined by programming the SFR_EBICFG bit in SFR registers” changed to “The drive levels are configured with the DRIVEx field in the EBI Configuration Register (SFR_EBICFG)”

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 8. “Event System” Section 8-1 “Real-time Event Mapping List”: instance of “ADC_ADTRG” corrected to “ADTRG”
	Section 9. “System Controller” Section 9.1 “Power-On Reset”: “dedicated to VDDBU, VDDIOP and VDDCORE” changed to “dedicated to monitoring VDDBU, VDDIOP and VDDCORE”
	Section 10. “Peripherals” Table 10-1 “Peripheral identifiers”: in ‘Instance Name’ column, renamed CAN0 and CAN1 to MCAN0 and MCAN1 Section 10.4 “Peripheral Clock Types”: in SLOW_CLOCK description, “32768-Hz crystal oscillator or by the on-chip 32-kHz RC oscillator” changed to “32.768 kHz crystal oscillator or by the on-chip 64 kHz RC oscillator”
	Section 11. “Chip Identifier (CHIPID)” Updated Table 11-1 “SAMA5D2 Chip ID Registers”
	Section 13. “L2 Cache Controller (L2CC)” Table 13-2 “Register Mapping”: reset value 0x0000_0000 changed to 0x0000_0111 for L2CC_TRCR and L2CC_DRCR
	Section 14. “Debug and Test Features” Table 14-1 “Debug and Test Pin List”: NRST pin function “Microcontroller Reset” changed to “Microprocessor Reset”
	Section 15. “Standard Boot Strategies” “Boot Sequence Control Register (BSCR)” renamed to “Boot Sequence Controller Configuration Register” Section 15.1 “Description”: “This microcontroller can be configured” changed to “This microprocessor can be configured” Figure 15-10 “Galois Field Table Mapping”: modified Galois field table offsets Section 15.4.2 “Boot Sequence Controller Configuration Register”: added address Section 15.4.3 “Boot Configuration Word”: added reference to “Customer Fuse Matrix” Added Section 15.4.6.5 “QSPI Flash Boot” Table 15-3 “PIO Driven during Boot Program Execution”: NAND Flash PIO line PIOC17 changed to PIOB0
	Section 18. “Special Function Registers (SFR)” Table 18-1 “Register Mapping”: removed EBI Configuration Register / SFR_EBICFG (offset 0x40 now reserved) Section 18.3.1 “DDR Configuration Register”: added note Removed section “EBI Configuration Register”
	Section 21. “Watchdog Timer (WDT)” Section 21.4 “Functional Description”: in eighth paragraph, “To prevent a software deadlock that continuously triggers the watchdog, the reload of the watchdog must occur...” changed to “The reload of the watchdog must occur...”
	Section 25. “Real-time Clock (RTC)” Reworked Section 25.5.6 “Updating Time/Calendar” Reworked Figure 25-7 “Calibration Circuitry Waveforms” AD index ‘7’ replaced with generic ‘n’ in Section 25.5.8 “Waveform Generation” Updated Figure 25-8 “Waveform Generation for ADC Trigger Event” Section 25.6.2 “RTC Mode Register”: - updated descriptions of fields OUT0 and OUT1 - added fields TPERIOD and THIGH
	Section 27. “Low Power Asynchronous Receiver (RXLP)” Pin/signal name “LPRXD” changed to “RXD”

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 29. “Clock Generator” Section 29.2 “Embedded Characteristics”: AUDIOPLLCK changed to AUDIOPLLCLK Figure 29-1 “Clock Generator Block Diagram”: lines changed to arrows for OSCSEL to multiplexer, for MOSCSEL to multiplexer, and for PLLADIV2 to “PLLA and Divider” block Figure 29-5 “Divider and PLLA Block Diagram”: added PLLADIV2 divider Updated Section 29.8 “Audio PLL”
	Section 30. “Power Management Controller (PMC)” AUDIOPLLCK changed to AUDIOPLLCLK in Section 30.15 “Programmable Clock Controller” and Section 30.16 “Generic Clock Controller” Figure 30-1 “General Clock Block Diagram”: added PLLA block; repositioned PLLACK signal; at bottom of diagram “PCKx” changed to “PCKx (to pads)” Table 30-3 “Register Mapping”: PMC_AUDIO_PLL0 reset value ‘0x0000_0000’ changed to ‘0x0000_00D0’
	Section 30. “Power Management Controller (PMC)” (cont”d) Section 30.22.11 “PMC Master Clock Register”: updated CSS field description Section 30.22.13 “PMC Programmable Clock Register”: added addresses 0xF0014044 and 0xF0014048; updated CSS field description Section 30.22.39 “PMC Audio PLL Control Register 0”: added fields DCO_FILTER (bits 29:28), DCO_GAIN (bits 27:24) and PLLFLT (bits 7:4) Section 30.22.40 “PMC Audio PLL Control Register 1”: updated DIV field description (cont’don next page)

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 31. “Parallel Input/Output Controller (PIO)”
	Section 31.4.2 “External Interrupt Lines”: “are generally multiplexed” changed to “are multiplexed”
	Section 31.5 “Functional Description”: removed entire section “Peripheral Muxing Example”
	Table 31-4 “Register Mapping”:
	- added reset value for PIO_CFGR, PIO_ODSR, PIO_IMR, S_PIO_CFGR, S_PIO_ODSR and S_PIO_IMR
	- “PIO I/O Freeze Register” corrected to “PIO I/O Freeze Configuration Register”
	- defined offset range 0x400–0x4FC as reserved
	- reserved offset range 0x5E8–0x5F8 changed to 0x5E8–0x5FC
	- “Secure PIO I/O Freeze Register” corrected to “Secure PIO I/O Freeze Configuration Register”
	Removed duplicated or invalid addresses in Section 31.7.1 “PIO Mask Register”, Section 31.7.2 “PIO Configuration Register”, Section 31.7.3 “PIO Pin Data Status Register”, Section 31.7.4 “PIO Lock Status Register”, Section 31.7.5 “PIO Set Output Data Register”, and Section 31.7.6 “PIO Clear Output Data Register”
	Section 31.7.7 “PIO Output Data Status Register”: removed duplicated or invalid addresses; access “Read-only or Read/Write” corrected to “Read/Write”
	Removed duplicated or invalid addresses in Section 31.7.8 “PIO Interrupt Enable Register”, Section 31.7.9 “PIO Interrupt Disable Register”, Section 31.7.10 “PIO Interrupt Mask Register”, and Section 31.7.11 “PIO Interrupt Status Register”
	Section 31.7.12 “PIO I/O Freeze Configuration Register”: corrected title (was “PIO Freeze Configuration Register”); removed duplicated or invalid addresses; access “Read/Write” corrected to “Write-only”
	Removed duplicated or invalid addresses in Section 31.7.15 “Secure PIO Mask Register”, Section 31.7.16 “Secure PIO Configuration Register”, Section 31.7.17 “Secure PIO Pin Data Status Register”, Section 31.7.18 “Secure PIO Lock Status Register”, Section 31.7.19 “Secure PIO Set Output Data Register” and Section 31.7.20 “Secure PIO Clear Output Data Register”
	Section 31.7.21 “Secure PIO Output Data Status Register”: removed duplicated or invalid addresses; access “Read-only or Read/Write” corrected to “Read/Write”
	Removed duplicated or invalid addresses in Section 31.7.22 “Secure PIO Interrupt Enable Register”, Section 31.7.23 “Secure PIO Interrupt Disable Register”, Section 31.7.24 “Secure PIO Interrupt Mask Register”, and Section 31.7.25 “Secure PIO Interrupt Status Register”
	Section 31.7.29 “Secure PIO I/O Freeze Configuration Register”: corrected title (was “Secure PIO Freeze Configuration Register”); removed duplicated or invalid addresses; access “Read/Write” corrected to “Write-only”
	Section 31.7.30 “Secure PIO Slow Clock Divider Debouncing Register”: added sentence about register write protection
	Section 32. “External Memories”
	Table 32-1 “DDR/LPDDR I/O Lines Description”: updated DDR_VREF function description
	Section 33. “Multiport DDR-SDRAM Controller (MPDDRC)”
	Section 33.4.1 “Low-power DDR1-SDRAM Initialization”: in first paragraph, removed content about configuring register SFR_DDRCFG
	Section 33.6 “Software Interface/SDRAM Organization, Address Mapping”: modified description of Interleaved mode (“at each SDRAM end page” corrected to “at each DDRSDRAM end of page”)
	Harmonized register naming throughout Section 33.7 “AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface”
	Removed all MPDDRC DLL registers (offset range 0x100–0x158 now reserved)
	Section 33.7.3 “MPDDRC Configuration Register”: modified description of DECOD bit value ‘1’ (“at each SDRAM end page” corrected to “at each DDR-SDRAM end of page”)
	Section 33.7.12 “MPDDRC I/O Calibration Register”: updated RZQ values in RDIV field description

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 34. “Static Memory Controller (SMC)” Section 34.17.3 “NFC Initialization”: instances of “rbn” changed to “Ready/Busy” Section 34.20.3 “NFC Status Register”: bit RB_EDGE3 (bit 27) replaced by RB_EDGE0 (bit 24); updated RB_RISE and RB_FALL bit descriptions Bit RB_EDGE3 (bit 27) replaced by RB_EDGE0 (bit 24) in Section 34.20.4 “NFC Interrupt Enable Register”, Section 34.20.5 “NFC Interrupt Disable Register” and Section 34.20.6 “NFC Interrupt Mask Register” Deleted invalid addresses in Section 34.20.30 “PMECC Error Location SIGMA0 Register” and Section 34.20.31 “PMECC Error Location SIGMAx Register” Section 34.20.32 “PMECC Error Location x Register”: register index “x=0..23” corrected to “x=0..31” Section 34.20.36 “Timings Register”: removed RBNSEL field
	Section 35. “DMA Controller (XDMAC)” Added XDMAC_CCx.CSIZE configuration to Table 35-2 “DMA Channels Definition (XDMAC0)” and Table 35-3 “DMA Channels Definition (XDMAC1)” Table 35-5 “Register Mapping”: - XDMAC_GCFG access Read-only corrected to Read/Write - XDMAC_GWAC access Read-only corrected to Read/Write Section 35.9.2 “XDMAC Global Configuration Register”: access Read-only corrected to Read/Write Section 35.9.3 “XDMAC Global Weighted Arbiter Configuration Register”: access Read-only corrected to Read/Write
	Section 36. “LCD Controller (LCDC)” Updated ”Section 36.2 “Embedded Characteristics” Updated Section 36.6.1.1 “Pixel Clock Period Configuration”
	Section 37. “Ethernet MAC (GMAC)” Table 37-1 “GMAC Connections in Different Modes”: added table Note on GTXCK Updated Section 37.5.3 “Interrupt Sources” Section 37.7.1.2 “Receive Buffer List” and Section 37.7.1.3 “Transmit Buffer List”: added note at end of sections on queue pointer intilaization Section 37.8.107 “GMAC Transmit Buffer Queue Base Address Register Priority Queue x” and Section 37.8.108 “GMAC Receive Buffer Queue Base Address Register Priority Queue x”: changed sentence on register initialization
	Section 39. “USB Host High Speed Port (UHPHS)” Section 39.2 “Embedded Characteristics”: “X Hosts (A and B) High Speed (EHCI)” corrected to “2 Hosts (A and B) High Speed (EHCI)” Table 39-2 “Register Mapping”: inserted reserved offset 0x0C Section 39.7.19 “EHCI: REG06 - AHB Error Status”: instances of “INSNREG[8:4]” changed to “INSNREG06[8:4]”
	Section 40. “Audio Class D Amplifier (CLASSD)” Section 40.2 “Embedded Characteristics”: DSP clock frequency “11.289 MHz” corrected to “11.2896 MHz” Section 40.5.2 “Power Management”: field name “NOVRLAP” corrected to “NOVRVAL” Figure 40-21 “Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Single-ended Outputs”: changed title (was “Use Case 4B: Stereo Audio DAC With Passive Low Pass Filter and Differential Outputs”)
	Section 42. “Synchronous Serial Controller (SSC)” Figure 42-19 “Interrupt Block Diagram”: “RXSYNC” renamed to “RXSYN”; “TXSYNC” renamed to “TXSYN” Section 42.8.10 “Register Write Protection”: in first sentence, “AIC behavior” corrected to “SSC behavior”

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 43. “Two-wire Interface (TWIHS)”
	Section 43.6.3.10 “SMBus Mode”: Deleted “A dedicated bus line, SMBALERT, allows a slave to get a master attention” from listed exceptions
	Section 43.6.5.6 “SMBus Mode”: Deleted “A dedicated bus line, SMBALERT, allows a slave to get a master attention” from listed exceptions
	Deleted note about debugger read access in Section 43.7.6 “TWIHS Status Register”, Section 43.7.13 “TWIHS Receive Holding Register” and Section 43.7.25 “TWIHS Write Protection Status Register”
	Section 44. “Flexible Serial Communication Controller (FLEXCOM)”
	Section 44.7.1.2 “Fractional Baud Rate in Asynchronous Mode”: in first paragraph, deleted sentence “This feature is only available when using USART Normal mode.”
	Figure 44-8 “Preamble Patterns, Default Polarity Assumed”: instances of “8 bit width” changed to “8-bit”
	Figure 44-11 “Asynchronous Start Detection”: added missing arrowheads
	Section 44.7.3.11 “Receiver Timeout”: removed redundant paragraphs on STTTO and RETTO; reworded two bullets
	Section 44.7.4 “ISO7816 Mode”: at end of second paragraph, “value 0x5 for protocol T = 1” changed to “value 0x6 for protocol T = 1”
	Section 44.7.4.2 “Protocol T = 0”: reworded content under “Receive NACK Inhibit”
	Section 44.7.7 “USART Comparison Function on Received Character”: modified information about the CMPMODE bit
	Table 44-18 “Register Mapping”: added TWI SleepWalking Matching Register (FLEX_TWI_SWMR)
	Section 44.10.41 “USART Write Protection Mode Register”: rephrased WPEN bit description
	Corrected order of all sections from Section 44.10.66 “TWI Interrupt Enable Register” to Section 44.10.76 “TWI SleepWalking Matching Register”
	Section 44.10.76 “TWI SleepWalking Matching Register”: added addresses
	Section 46. “Serial Peripheral Interface (SPI)”
	Figure 46-1 “Block Diagram”: added GCLK output from PMC to SPI
	Modified transmission condition description in Section 46.7.3 “Master Mode Operations”
	Section 46.7.4 “SPI Slave Mode”: added sentence about NSS rising between characters
	Section 46.7.5 “SPI Comparison Function on Received Character”: in seventh paragraph, added “if SleepWalking mode is disabled” to sentence “The comparison trigger event is...”
	Updated Section 46.7.8 “Register Write Protection”
	Section 46.8.2 “SPI Mode Register”: added bits BRSRCCLK (Bit Rate Source Clock) and LSBHALF (LSB Timing Selection); updated description of field DLYBCS
	Section 46.8.12 “SPI Chip Select Register”: updated description of fields CSNAAT, SCBR, DLYBS and DLYBCT
	Section 47. “Quad Serial Peripheral Interface (QSPI)”
	Section 47.2 “Embedded Characteristics”: added bullet “Interface to Serial Flash Memories Operating in Single Data Rate or Double Data Rate Modes”
	Section 47. “Quad Serial Peripheral Interface (QSPI)” (cont’d)
	NSS renamed to QCS in Figure 47-2 “QSPI Transfer Format (QSPI_SCR.CPHA = 0, 8 bits per transfer)” and Figure 47-3 “QSPI Transfer Format (QSPI_SCR.CPHA = 1, 8 bits per transfer)”
	Section 47.7.2 “QSPI Mode Register”: added note “This field is forced to LASTXFER when SMM is written to ‘1’ to CSMODE field description; modified equation in description of fields DLYBCT and DLYCS
	Section 47.7.5 “QSPI Status Register”: updated descriptions of bits CSR and INSTRE
	Section 47.7.9 “QSPI Serial Clock Register”: modified equation in description of fields SCBR and DLYBS

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 48. “Secure Digital Multimedia Card Controller (SDMMC)” Added Section 48.3 “Embedded Features for SDMMC0 and SDMMC1” Figure 48-1 “Block Diagram”: added two notes Table 48-3 “Register Mapping”: updated SDMMC_APSR reset value (SDMMC0 different from SDMMC1) Section 48.13.18 “SDMMC Software Reset Register”: updated SWRSTCMD bit description Section 48.13.58 “SDMMC Calibration Control Register”: in CNTVAL field description, “$t_{STARTUP}$ = ...” corrected to “$t_{STARTUP}$ = 2 µs”
	Section 49. “Image Sensor Controller (ISC)” Added Section 49.4 “Product Dependencies” Table 49-18 “Register Mapping”: defined offset range 0x404–0x40C as reserved
	Section 50. “Controller Area Network (MCAN)” “GCLK3” changed to “GCLK” in Section 50.3 “Block Diagram” and Section 50.4.2 “Power Management” Added Table 50-2 “Peripheral IDs” Updated Section 50.5.1.3 “CAN FD Operation” Section 50.5.1.4 “Transmitter Delay Compensation”: modified title (was “Transceiver Delay Compensation”); revised content Section 50.5.1.5 “Restricted Operation Mode”: added ‘Note’ Section 50.5.3 “Timeout Counter”: “baud rate” changed to “bit rate” in ‘Note’ Section 50.5.4.1 “Acceptance Filtering”: “described in Section” corrected to “described in “Rx FIFO Overwrite Mode” Updated Figure 50-5 “Standard Message ID Filter Path”and Figure 50-6 “Extended Message ID Filter Path” Updated register names in Figure 50-7 “Rx FIFO Status” and Figure 50-8 “Rx FIFO Overflow Handling” Section 50.5.7.2 “Rx Buffer and FIFO Element”: “R1 Bit 21 FDF: Extended Data Length” renamed to “R1 Bit 21 FDF: FD Format” Section 50.5.7.4 “Tx Event FIFO Element”: “E1 Bit 21 FDF: Extended Data Length” renamed to “E1 Bit 21 FDF: FD Format” Table 50-14 “Register Mapping”: - deleted row “0x00–0x04 / Reserved” - “Fast Bit Timing and Prescaler Register” renamed to “Data Bit Timing and Prescaler Register” - “Bit Timing and Prescaler Register” renamed to “Nominal Bit Timing and Prescaler Register” Section 50.6.4 “MCAN Data Bit Timing and Prescaler Register”: changed name (was “MCAN Fast Bit Timing and Prescaler Register”); field FBRP replaced by field DBRP Section 50.6.7 “MCAN CC Control Register”: updated descriptions of fields FDOE, BRSE, PXHD and EFB; removed NISO bit Section 50.6.8 “MCAN Nominal Bit Timing and Prescaler Register”: “NBRP: Nominal Baud Rate Prescaler” changed to “NBRP: Nominal Bit Rate Prescaler” Section 50.6.9 “MCAN Timestamp Counter Configuration Register”: updated TSS field description Section 50.6.10 “MCAN Timestamp Counter Value Register”: updated TSC field description
	Section 50. “Controller Area Network (MCAN)” (cont’d) Section 50.6.20 “MCAN Global Filter Configuration”: added details on register description; updated ANFE and ANFS field descriptions. Added details on register description in Section 50.6.21 “MCAN Standard ID Filter Configuration” and Section 50.6.22 “MCAN Extended ID Filter Configuration” Section 50.6.24 “MCAN High Priority Message Status”: updated description of MSI field value ‘1’

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 51. “Timer Counter (TC)” Replaced TIOA, TIOB, TCLK with TIOAx, TIOBx, TCLKx Table 51-1 “Timer Counter Clock Assignment”: updated definitions Section 51.6.3 “Clock Selection”: updated bullet “Internal clock signals”, updated notes 1 and 2 Section 51.6.9 “Transfer with DMAC in Capture Mode”: updated title (added “in Capture Mode”) Updated Figure 51-5 “Example of Transfer with DMAC in Capture Mode” Section 51.6.16.4 “Position and Rotation Measurement”: updated text in first paragraph Added Section 51.6.16.6 “Detecting a Missing Index Pulse” Updated TCCLKS field description in Section 51.7.2 “TC Channel Mode Register: Capture Mode” and Section 51.7.3 “TC Channel Mode Register: Waveform Mode”
	Section 53. “Pulse Width Modulation Controller (PWM)” Throughout, “PWMTRG” and “EXTTRG” renamed to “PWMEXTRG” Table 53-2 “I/O Lines”: “PWMEXTRG0” and “PWMEXTRG1” renumbered to “PWMEXTRG1” and “PWMEXTRG2” Updated Section “Recoverable Fault” Updated Figure 53-1 “Pulse Width Modulation Controller Block Diagram” and added note below figure Updated Figure 53-16 “Fault Protection”
	Section 54. “Secure Fuse Controller (SFC)” Table 54-1 “Register Mapping”: removed reset value from SFC_IER and SFC_IDR (both registers are write-only)
	Section 55. “Integrity Check Monitor (ICM)” Table 55-8 “Register Mapping”: ICM_SR access “Write-only” corrected to “Read-only”
	Section 57. “Advanced Encryption Standard (AES)” Table 57-5 “Register Mapping”: AES_ALPHAR[0..3] access “Write” corrected to “Write-only” Section 57.5.20 “AES Alpha Word Register x”: access “Write” corrected to “Write-only”
	Section 59. “Triple Data Encryption Standard (TDES)” Section 59.4.1 “Operating Modes”: deleted sentence “The OFB and CFB modes of operation are only available if 2-key mode is selected (KEYMOD = 1 in TDES_MR).” Section 59.4.3 “Last Output Data Mode”: deleted sentence “No more Output Data Register reads are necessary between consecutive encryptions/decryptions (see Section 59.4.3 “Last Output Data Mode”).” Section 59.5.2 “TDES Mode Register”: in OPMOD field description, deleted sentence “The OFB and CFB modes of operation are only available if 2-key mode is selected (KEYMOD = 1).”
	Section 61. “Security Module” Updated Figure 61-2 “Security Module Internal Memory Map”
	Section 61. “Analog-to-Digital Converter (ADC)” Section 61.1 “Description”: - deleted sentence “A digital error correction circuit based on the multi-bit redundant signed digit (RSD) algorithm is implemented to reduce INL and DNL errors.” - deleted sentence “Finally, the user can configure ADC timings, such as startup time and tracking time.”

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 61. “Analog-to-Digital Converter (ADC)” (cont’d) Updated Section 61.2 “Embedded Characteristics” Updated Figure 61-1 “Analog-to-Digital Converter Block Diagram” Revised Section 61.5 “Product Dependencies” Revised Section 61.6.1 “Analog-to-Digital Conversion” Updated Section 61.6.3 “ADC Reference Voltage” and Section 61.6.4 “Conversion Resolution” Updated Section 61.6.7 “Conversion Triggers” Section 61.6.9 “Comparison Window”: in fourth paragraph, instance of “ADC_SR” corrected to “ADC_ISR” Section 61.6.10 “Differential and Single-ended Input Modes”: changed title (was “Differential Inputs”) and revised content Updated Section 61.6.11 “ADC Timings”, Section 61.6.12 “Last Channel Specific Measurement Trigger”, Section 61.6.13 “Enhanced Resolution Mode and Digital Averaging Function” and Section 61.6.14 “Automatic Error Correction” Instances of GND renamed to GNDANA in Figure 61-15 “Touchscreen Switches Implementation”, Figure 61-18 “Touchscreen Switches Implementation” and Figure 61-20 “Touchscreen Pen Detect” Updated Section 61.6.16 “Asynchronous and Partial Wakeup (SleepWalking)” Section 61.6.17.1 “Classic ADC Channels Only (Touchscreen Disabled)”: changed title (was “Classical ADC Channels Only”) Section 61.6.19 “Register Write Protection”: updated list of protectable registers Table 61-8 “Register Mapping”: - defined 0x48 as reserved - added row 0x4C / Channel Offset Register / ADC_COR - added offset 0x7C for ADC_CDR11 - defined offset range 0x80–0x90 as reserved - added row 0x94 / Analog Control Register / ADC_ACR - defined offset range 0xC4–0xD0 as reserved - added row 0xD4 / Correction Values Register / ADC_CVR - added row 0xD8 / Channel Error Correction Register / ADC_CECR - added row 0xDC / Touchscreen Correction Values Register / ADC_TSCVR - defined offset 0xE0 as reserved Section 61.7.2 “ADC Mode Register”: updated TRACKIM field description Added LCCHG (Last Channel Change) bit in Section 61.7.9 “ADC Interrupt Enable Register”, Section 61.7.10 “ADC Interrupt Disable Register”, Section 61.7.11 “ADC Interrupt Mask Register” and Section 61.7.12 “ADC Interrupt Status Register” Section 61.7.13 “ADC Last Channel Trigger Mode Register”: updated CMPMOD field description Section 61.7.16 “ADC Extended Mode Register”: updated CMPMODE field description; added descriptions for fields OSR ASTE Section 61.7.18 “ADC Channel Offset Register”: added address; removed bits OFF11:OFF0 from bitmap; modified DIFFx field description Section 61.7.20 “ADC Analog Control Register”: added address; added IBCTL field Section 61.7.25 “ADC Trigger Register”: added sentence about write protection Removed Section “Correction Select Register” Added sentence about write protection in Section 61.7.26 “ADC Correction Values Register” and Section 61.7.27 “ADC Channel Error Correction Register” Added Section 61.7.28 “ADC Touchscreen Correction Values Register”

Table 68-2. SAMA5D2 Datasheet Rev. 11267D Revision History (Continued)

Issue Date	Changes
12-May-16	Section 62. “Electrical Characteristics” “ADVREFP” corrected to “ADVREF” Section 62.2 “DC Characteristics”: in first sentence, “T_A = -40°C to +85°C” changed to “T_A = -40°C to +105°C” Added Table 62-2 “Recommended Thermal Operating Conditions” Updated Section 62.4 “Active Mode” Table 62-8 “Low-power Mode Configuration Summary”: updated values for ‘Consumption’ and ‘Wakeup Time’ Updated Section 62.5.6 “Low-power Consumption Versus Modes” Table 62-9 “Typical Power Consumption in Idle Mode: AMP2”: updated consumption values Table 62-10 “VDDCORE Power Consumption in Ultra Low-power Mode: AMP2”: updated consumption values; updated wakeup time for ULP1 Fast Wakeup mode Table 62-11 “Typical Power Consumption for Backup Mode”: updated consumption values Updated Table 62-12 “Processor Clock Waveform Parameters” and Table 62-13 “Master Clock Waveform Parameters” Updated Section 62.7.1 “Main Oscillator Characteristics” Table 62-17 “12 MHz RC Oscillator Characteristics”: updated startup time values Updated Section 62.7.3 “32.768 kHz Crystal Oscillator Characteristics” Updated Table 62-23 “Audio PLL Characteristics” Section 62.10 “ADC Characteristics”: deleted sentence “The VREFN pin must be connected to ground.” Table 62-25 “Power Supply Characteristics”: updated Analog Current Consumption value for Fast Wakeup mode Table 62-26 “ADVREF Electrical Characteristics”: “VREFP” corrected to “ADVREF”; updated Current value Section 62.10.4.1 “Differential Mode (12-bit mode)” and Section 62.10.4.2 “Single-ended Mode (12-bit mode)”: in equation, “V_{VREFP}” corrected to “V_{ADVREF}” Section 62.10.4.4 “Gain and Offset Errors”: “V_{VREFP}” corrected to “V_{ADVREF}” Table 62-27 “ADC Timing Characteristics”: updated footnote Added Table 62-32 “ADC Analog Input Characteristics” Table 62-37 “VDDCORE Power-On Reset Characteristics”: updated Hysteresis Voltage values Section 62.14.1 “Maximum SPI Frequency”: updated values in “Master Read Mode” and “Slave Write Mode” Revised Section 62.18 “MPDDRC Timings” Corrected CKI values in Figure 62-33 “SSC Transmitter, TK and TF in Input”, Figure 62-35 “SSC Receiver, RK in Input and RF in Output”, Figure 62-36 “SSC Receiver, RK and RF in Output” and Figure 62-38 “Minimum and Maximum Access Time of Output Signals”
	Section 64. “Schematic Checklist” Figure 64-1 “1.2V, 1.35V/1.5V, 2V, 2.5V, 3.3V Power Supplies Schematics[(1)]”: GNDHSIC changed to GNDUTMIC Table 64-1 “Power Supply Connections”: updated GNDUTMIC row; removed GNDHSIC row; in second footnote, “microcontroller” changed to “microprocessor” Table 64-2 “Clock, Oscillator and PLL Connections”: “(internal 32-kHz RC oscillator) changed to “(internal 64 kHz RC oscillator)” Section 64.5.1 “How to Define the Oscillator Load Capacitance”: instances of “32-KHz Oscillator” changed to “32.768 kHz Oscillator” Added Section 64.14.6 “QSPI Pull-up Resistors”
	Updated Section 66. “Ordering Information”
	Section 67. “Errata” Updated content (errata now collected in Section 67.1 “Errata - SAMA5D2 MRL-B Parts” and Section 67.2 “Errata - SAMA5D2 MRL-A Parts”)

Table 68-3. SAMA5D2 Datasheet Rev. 11267C Revision History

Issue Date	Changes
8-Jan-16	Changed datasheet status from ‘Preliminary’ to ‘Complete’.
	Added “Introduction” and transferred Description to Section 1.
	Section 2. “Configuration Summary” Added device compatibility information
	Section 4. “Signal Description” Table 4-1 “Signal Description List”: modified rows PIOBU 0-7 and DDR_RESETN
	Section 6. “Power Considerations” Added Section 6.4.1 “VDDBU Power Architecture” Updated Section 6.2 “Powerup Considerations” and Section 6.3 “Powerdown Considerations”
	Section 7. “Memories” Updated Section 7.1.2 “Internal ROM”
	Section 10. “Peripherals” Updated Table 10-1 “Peripheral identifiers” and Section 10.4 “Peripheral Clock Types”
	Section 16. “AXI Matrix (AXIMX)” Table 16-1 “Register Mapping”: removed 0x00000000 reset value from all rows
	Section 17. “Matrix (H64MX/H32MX)” Section 17.2 “Embedded Characteristics”: removed "Master number forwarding to slaves" characteristic Updated Table 17-1 “List of H64MX Masters”, Table 17-2 “List of H64MX Slaves”, Table 17-4 “List of H32MX Masters”, Table 17-5 “List of H32MX Slaves” Table 17-3 “Master to Slave Access on H64MX”: updated ‘SDMMC0-SDMMC1’ row Table 17-6 “Master to Slave Access on H32MX”: updated ‘Slave 5’ rows Section 17.12.2 “Security of APB Slaves”: added introduction and bulleted list introduced by “As a general rule” Added Section 17.12.3 “Security Types of AHB Master Peripherals” and Section 17.12.4 “Security Types of AHB Slave Peripherals” Section 17-9 “Peripheral Identifiers”: corrected some security type names Section 17.13 “AHB Matrix (MATRIX) User Interface”: added introduction and modified reset value of Updated Security Areas Split Slave x Registers in Table 17-10 “Register Mapping”
	Section 24. “Watchdog Timer (WDT)” Replaced “Idle mode” with “Sleep mode (Idle mode)” in Section 24.1 “Description” and with “Sleep mode” in Section 24.4 “Functional Description”
	Section 22. “Reset Controller (RSTC)” Renamed 'proc_nreset' to 'Processor Reset', 'periph_nreset' to 'Peripheral Reset', 'backup_neset' to 'Backup Reset', 'rstc_irq' to 'Reset Controller Interrupt', 'wd_fault' to 'Watchdog Fault', ‘user_reset’ to User Reset. Updated text and figures to show that Processor Reset and Peripheral Reset signals are merged.
	Section 23. “Shutdown Controller (SHDWC)” Updated Figure 23-1 “Shutdown Controller Block Diagram” and Table 23-1 “I/O Lines Description” Section 23.7.3 “Shutdown Status Register”: corrected register table (added WKUPIS9) Section 23.7.4 “Shutdown Wakeup Inputs Register”: corrected register table (added WKUPT9 and WKUPEN9)

Table 68-3. SAMA5D2 Datasheet Rev. 11267C Revision History (Continued)

Issue Date	Changes
8-Jan-16	Section 29. “Real-time Clock (RTC)” Removed RTC Milliseconds Register (RTC_MSR) and all related information in Section 29.1 “Description”, Section 29.2 “Embedded Characteristics”, Section 29.5 “Functional Description” and Section 29.6 “Real-time Clock (RTC) User Interface”. Table 29-1 “Register Mapping”: modified RTC_CALR reset value Section 29.6.1 “RTC Control Register”: updated CALEVSEL field description Updated Section 29.6.22 “RTC TimeStamp Source Register”
	Section 29. “Clock Generator” Section 29.2 “Embedded Characteristics”: replaced "400 to 1000 MHz programmable PLL" with "600 to 1200 MHz programmable PLL" and replaced “HCLOCK” with “HCLOCK_LS/HS” and “PCLOCK” with “PCLOCK_LS/HS” Section 29.4 “Slow Clock”: removed “This allows the slow clock to be valid in a short time (about 100 µs)” Section 29.8 “Audio PLL”: updated all equations and added “in the 700 MHz range” after “The PLL core operates at 700 MHz (AUDIOCORECLOCK)” Updated Figure 29-3. Main Clock Block Diagram and Figure 29-4. Main Clock Source Selection
	Section 30. “Power Management Controller (PMC)” Updated Section 30.6 “Matrix Clock Controller” Updated Section 30-1 “General Clock Block Diagram” Section 30.19 “Programming Sequence”, sub-section “Selecting Master Clock and Processor Clock”: updated sequence following "If a new value for CSS field corresponds to PLL Clock" Section 30.22.11 “PMC Master Clock Register”: updated H32MXDIV field description
	Section 33. “Multi-port DDR-SDRAM Controller (MPDDRC)” Section 33-2 “Single Write Access, Row Closed, DDR-SDRAM Devices” to Section 33-8 “SINGLE Write Access Followed by a Read Access, DDR2-SDRAM Devices”: replaced “D[15:0]” with “DATA” Updated Section 33.7.9 “MPDDRC Low-power DDR2 Low-power DDR3 Low-power Register” Section 33.7.10 “MPDDRC Low-power DDR2 Low-power DDR3 and DDR3 Calibration and MR4 Register”: updated MR4_READ field description
	Section 34. “Static Memory Controller (SMC)” Removed NFCCMD field and modified Section 34.17.2.1 “Building NFC Address Command Example” and Section 34.17.2.2 “NFC Address Command” accordingly Table 34-20 “Register Mapping”: corrected offset values of PMECC Error Location 31 Register and of subsequent reserved range; removed reset value from HSMC_CTRL (register is write-only)
	Section 42. “DMA Controller (XDMAC)” Section 42.5.4.1 “Single Block With Single Microblock Transfer”: added text on memory-to-memory transfer Section 42.8 “XDMAC Software Requirements”: added bullet on memory-to-memory transfer Table 42-5 “Register Mapping”: corrected access of XDMAC_GTYPE, XDMAC_GWAC, XDMAC_CIM Section 42.9.6 “XDMAC Global Interrupt Mask Register”: corrected access to Read-only Section 42.9.28 “XDMAC Channel x [x = 0..15] Configuration Register”: corrected INITD and PERID field descriptions
	Section 36. “LCD Controller (LCDC)” Modified width of fields in Section 36.7.2 “LCD Controller Configuration Register 1” and Section 36.7.3 “LCD Controller Configuration Register 2”
	Section 40. “Audio Class D Amplifier (CLASSD)” Replaced ‘audio clock’ with ‘generic clock’ and ‘ACLK’ with ‘GCLK’ throughout the section

Table 68-3. SAMA5D2 Datasheet Rev. 11267C Revision History (Continued)

Issue Date	Changes
8-Jan-16	Section 41. “Inter-IC Sound Controller (I2SC)”
	Section 41.6.3 “Master, Controller and Slave Modes”: removed text fragment: ‘in order to avoid unwanted glitches on the I2SWS and I2SCK pins.’
	Section 41.8.2 “Inter-IC Sound Controller Mode Register”: removed text fragment: ‘in order to avoid unexpected behavior on the I2SWS, I2SCK and I2SDO outputs.’ and added note (2) below IMCKDIV field description.
	Section 44. “Flexible Serial Communication Controller (FLEXCOM)”
	Restored all references to ISO7816 specification
	Updated Figure 44-3 “Fractional Baud Rate Generator”
	Added Figure 44-27 “RTS line software control when FLEX_US_MR.USART_MODE = 2”
	Section 44.10.6 “USART Mode Register”: updated USART_MODE field description (SPI_MASTER item)
	Section 44.10.44 “SPI Mode Register”: added LBHPC bit
	Section 55. “Universal Asynchronous Receiver Transmitter (UART)”
	Section 55.6.9 “UART Baud Rate Generator Register”: in CD field description, corrected equation after “If BRSRCCK = 1”
	Section 59. “Quad SPI Interface (QSPI)”
	Section 59.7.5 “QSPI Status Register”: updated RDRF, TDRE, TXEMPTY, and OVRES field descriptions
	Section 48. “Secure Digital Multimedia Card Controller (SDMMC)”
	Section 48.12.41 “SDMMC Preset Value Register”: updated CLKGSEL field description
	Section 49. “Image Sensor Controller (ISC)”
	Section 49.1 “Description”: removed "serial csi-2 based CMOS/CCD sensor" (not supported).
	Section 50. “Controller Area Network (MCAN)”
	Changed MCAN interrupt line names to MCAN_INT0 and MCAN_INT1 thoughout the section
	Section 50.6.7 “MCAN CC Control Register”: added bit NISO
	Section 51. “Timer Counter (TC)”
	Reformatted and renamed Table 51-2 “Channel Signal Description”
	Section 51.6.3 “Clock Selection”: updated notes (1) and (2)
	Section 52. “Pulse Density Modulation Interface Controller (PDMIC)”
	Replaced all instances of “PCK” with “GCLK”
	Section 52.2 “Embedded Characteristics”: removed ‘Multiplexed PDM Input Support’ characteristic
	Updated Section 52.5.2 “Power Management” and Section 52.6.2.1 “Description”
	Section 52.6.2.6 “Gain and Offset Compensation”: updated *dgain* bullet
	Section 52.7.3 “PDMIC Converted Data Register”: updated DATA field description
	Section 52.7.8 “PDMIC DSP Configuration Register 0”: updated OSR field description
	Section 61. “Security Module”
	Section 61.5.5 “SECUMOD Status Clear Register”: removed MCKM field description
	Section 61.5.18 “SECUMOD Wake Up Register”: removed TPML field description
	Section 77. “Analog-to-Digital Converter (ADC)”
	Section 77.7.2 “ADC Mode Register”: updated TRACKTIM and TRANSFER field descriptions.

Table 68-3. SAMA5D2 Datasheet Rev. 11267C Revision History (Continued)

Issue Date	Changes
8-Jan-16	Section 62. “Electrical Characteristics” Updated tables from Table 62-3 “DC Characteristics” to Table 62-35 “Analog Comparator Characteristics” Updated Figure 62-3 “Main Oscillator Schematics” Corrected Gain Error formula under Figure 62-6 “Gain and Offset Errors in Single-ended Mode” Removed Figure 63-4 “Single-ended Mode ADC” and Figure 63-5 “Differential Mode ADC” Updated wake-up pin numbers in Section 62.5.1 “Backup Mode” and Section 62.5.3.2 “ULP1 Mode” Updated Section 62.5.4 “Idle Mode” and Section 62.23 “SDMMC Timings” Section 62.14.3 “Timing Extraction”: added introduction and Figure 62-12 “MISO Capture in Master Mode”
	Section 64. “Schematic Checklist” Removed Table 65-12. “EBI Pins and NAND Flash Device Connections” and Table 65-13. “DDR2 I/O Lines Usage vs Operating Modes”
	Reorganized Section 66. “Ordering Information”
	Updated Section 67. “Errata”

Table 68-4. SAMA5D2 Datasheet Rev. 11267B Revision History

Issue Date	Changes
13-Nov-15	"Features" Updated Security features
	Section 3. "Block Diagram" Updated Figure 3-1 "SAMA5D2 Series Block Diagram".
	Section 5. "Package and Pinout" Updated Table 5-2 "Pin Description (SAMA5D21, SAMA5D22, SAMA5D24, SAMA5D26, SAMA5D27, SAMA5D28A)" Removed Section 4.2 "Input/Output Description" and Section 4-3 "SAMA5D2 I/O Type Description"
	Section 6. "Power Considerations" Updated Table 6-1 "SAMA5D2 Power Supplies" Updated Figure 6-1 "Recommended Powerup Sequence", Figure 6-2 "Recommended Powerdown Sequence", Figure 6-3 "Recommended Backup Mode Entry", Figure 6-4 "Recommended Power Supply Sequencing at Wakeup"
	Section 8. "Event System" Updated Table 8-1 "Real-time Event Mapping List"
	Section 15. "Standard Boot Strategies" Replaced all instances of "GPBR" with "BUREG".
	Section 20. "Special Function Registers (SFR)" Updated Section 20.3.15 "I2S Register"
	Section 20. "Advanced Interrupt Controller (AIC)" Removed Sections "Interrupt Vectoring" and "Fast Interrupt Vectoring" Updated Section 20.8.3.3 "Interrupt Handlers"and Section 20.8.4.3 "Fast Interrupt Handlers"
	Section 29. "Power Management Controller (PMC)" Replaced "generated clock" with "generic clock", and "GCK" with "GCLK" Updated Section 29.22.8 "PMC Clock Generator Main Oscillator Register"
	Section 37. "Parallel Input/Output Controller (PIO)" Removed all references to programmable I/O delay
	Added Section 32. "External Memories"
	Section 33. "Multi-port DDR-SDRAM Controller (MPDDRC)" Section 33.4.3 "Low-power DDR2-SDRAM Initialization": added Step 14., Step 15. and Step 21. Section 33.4.5 "Low-power DDR3-SDRAM Initialization": added Step 14., Step 15. and Step 21. Section 33.7.8 "MPDDRC Memory Device Register": updated DBW field description; corrected location of fields RL3 and WL
	Section 37. "Ethernet MAC (GMAC)" Updated Section 37.1 "Description" Section 37.5.2 "Power Management": deleted reference to PMC_PCER Section 37.5.3 "Interrupt Sources": deleted reference to 'Advanced Interrupt Controller'. Replaced by 'Interrupt Controller'. Section 37.6.14 "IEEE 1588 Support": deleted reference to GMAC_TSSx. Removed reference to 'output pins' in 2nd paragraph. Section 37.6.15 "Time Stamp Unit": added information on GTSUCOMP signal in last paragraph

Table 68-4. SAMA5D2 Datasheet Rev. 11267B Revision History (Continued)

Issue Date	Changes
13-Nov-15	Section 39. “Audio Class D Amplifier (CLASSD)” Updated Figure 39-1. CLASSD Block Diagram
	Section 41. “Inter-IC Sound Controller (I2SC)” Replaced all instances of “PCKx” with “GCLK” Removed all references to Time Division Multiplexed (TDM) format (not supported) Section 41.1 “Description”: replaced “The I2SC can use either a single DMA Controller channel for both audio channels or one DMA Controller channel per audio channel.” with “The I2SC uses a single DMA Controller channel for both audio channels.”, and updated Section 41.2 “Embedded Characteristics” and Section 41.6.8 “DMA Controller Operation” accordingly Section 41.8.2 “Inter-IC Sound Controller Mode Register”: removed fields RXDMA and TXDMA
	Section 44. “Flexible Serial Communication Controller (FLEXCOM)” Added SPI mode in UART/USART Replaced all instances of ‘PCK’ with ‘GCLK’ Replaced all instances of ‘DMAC/PDC’ with ‘DMAC’ Removed SleepWalking characteristic from UART/USART mode Removed all references to ISO7816 specification Section 44.10.6 “USART Mode Register”updated USCLKS field description Section 44.10.44 “SPI Mode Register”: updated BRSRCCLK and DLYBCS field descriptions Section 44.10.54 “SPI Chip Select Register”: updated CSNAAT, SCBR, DLYBS and DLYBCT field descriptions Section 44.10.64 “TWI Clock Waveform Generator Register”: updated BRSRCCLK and CKSRC field descriptions Updated Figure 44-1 “FLEXCOM Block Diagram” and Figure 44-67 “Master Mode Block Diagram”
	Section 42. “Two-wire Interface (TWIHS)” Replaced all instances of “PMC_PCK” with “GCLK”
	Section 55. “Universal Asynchronous Receiver Transmitter (UART)” Replaced "Processor-Independent Source Clock" with "Processor-Independent Generic Source Clock" and "PCK" with "GCLK"
	Section 48. “Secure Digital Multimedia Card Controller (SDMMC)” Updated revision of supported e.MMC specification (from V4.41 to V4.51)
	Section 51. “Pulse Density Modulation Interface Controller (PDMIC)” Removed all references to PDC Removed Section 1.6.4 “Buffer Structure”
	Section 54. “Secure Fuse Controller (SFC)” Removed all references to lock fuse (not supported) Section 54.4.5.3 “Fuse Masking”: corrected data register names Section 54.5.2 “SFC Mode Register”: updated MSK field description Table 54-1 “Register Mapping”: modified SFC_IER and SFC_IDR access type from “Read/Write” to “Write-only”
	Section 57. “Advanced Encryption Standard (AES)” Updated Figure 57-12 “Generation of an ESP IPSec Frame without ESN” and Figure 57-13 “Generation of an ESP IPSec Frame with ESN”
	Added Section 61. “Security Module”

Table 68-4. SAMA5D2 Datasheet Rev. 11267B Revision History (Continued)

Issue Date	Changes
	Section 61. “Analog-to-Digital Converter (ADC)”
	Updated enhanced resolution value from 12 bits to 14 bits
	Renamed “Hold time” to “Transfer time”
	Replaced all instances of “PMC PCK” with “GCLK”
	Added Section 61.6.6 “Conversion Results Format”, Section 61.7.13 “ADC Last Channel Trigger Mode Register”, Section 61.7.14 “ADC Last Channel Compare Window Register”
	Section 61.2 “Embedded Characteristics”: corrected conversion rate
	Section 61.6.9 “Comparison Window”: added paragraph about bit SIGNMODE
	Section 61.6.14 “Automatic Error Correction”: replaced “GAIN_ERROR_SIZE-1” with appropriate value; replaced “Gs-1” with “Gs” in formulas
	Section 61.6.14 “Automatic Error Correction”, Section 62.7.27 “Correction Values Register”: replaced “GAIN_ERROR_SIZE-1” and “OFFSET_ERROR_SIZE-1” with appropriate values
	Section 61.7.2 “ADC Mode Register”: updated TRGSEL and TRACKTIM field descriptions
	Updated Section 61.7.8 “ADC Last Converted Data Register”
	Section 61.7.16 “ADC Extended Mode Register”: added bit SIGNMODE
	Updated Section 61.7.18 “ADC Channel Offset Register”
	Updated Section 61-1 “Analog-to-Digital Converter Block Diagram” and Section 61-7 “Analog Full Scale Ranges in Single-Ended/Differential Applications”
	Updated Table 62-5 “Oversampling Digital Output Range Values”
13-Nov-15	Section 62. “Electrical Characteristics”
	Added:
	- Section 62.11 “Analog Comparator Characteristics”
	- Section 62.14.1 “Maximum SPI Frequency”
	- Section 62.16.1 “Maximum QSPI Frequency”
	- Table 62-4 “I/O Switching Frequency”
	- Table 62-5 “QSPI I/O Switching Frequency”
	- Table 62-22 “UTMI PLL Characteristics”
	- Table 62-23 “Audio PLL Characteristics”
	Updated:
	- Table 62-1 “Absolute Maximum Ratings*”
	- Table 62-3 “DC Characteristics”
	- Table 63-8 “Typical Peripheral Power Consumption by Peripheral in Active Mode” to Table 62-11 “Typical Power Consumption for Backup Mode”
	- Table 62-14 “8 to 24 MHz Crystal Oscillator Characteristics”
	- Table 62-17 “12 MHz RC Oscillator Characteristics” to Table 62-22 “UTMI PLL Characteristics”
	- Table 62-36 “VDDBU Power-On Reset Characteristics” to Table 62-38 “VDDANA Power-On Reset Characteristics”
	Reworked Section 62.9 “USB HS Characteristics”
	Section 64. “Schematic Checklist”
	Updated:
	- Section 64.14.3 “DDR Layout and Design Considerations”
	- Figure 64-1 “1.2V, 1.35V/1.5V, 2V, 2.5V, 3.3V Power Supplies Schematics(1)”
	- Table 64-1 “Power Supply Connections”

Table 68-5. SAMA5D2 Datasheet Rev. 11267A Revision History

Issue Date	Changes
10-Sep-15	Preliminary Datasheet - First issue
25-Feb-15	Advance Information Datasheet.

Table of Contents

Introduction . . . 1

Features . . . 1

1. Description . . . 4

2. Configuration Summary . . . 5

3. Block Diagram . . . 6

4. Signal Description . . . 7

5. Package and Pinout . . . 13

5.1 Packages . . . 13

5.2 Pinouts . . . 13

6. Power Considerations . . . 37

6.1 Power Supplies . . . 37

6.2 Powerup Considerations . . . 37

6.3 Powerdown Considerations . . . 39

6.4 Power Supply Sequencing at Backup Mode Entry and Exit . . . 39

7. Memories . . . 42

7.1 Embedded Memories . . . 43

7.2 External Memory . . . 43

8. Event System . . . 47

8.1 Real-time Event List . . . 47

8.2 Real-time Event Mapping . . . 48

9. System Controller . . . 49

9.1 Power-On Reset . . . 51

10. Peripherals . . . 52

10.1 Peripheral Mapping . . . 52

10.2 Peripheral Identifiers . . . 52

10.3 Peripheral Signal Multiplexing on I/O Lines . . . 54

10.4 Peripheral Clock Types . . . 55

11. Chip Identifier (CHIPID) . . . 56

11.1 Description . . . 56

11.2 Embedded Characteristics . . . 56

11.3 Chip Identifier (CHIPID) User Interface . . . 57

12. ARM Cortex-A5 . . . 62

12.1 Description . . . 62

12.2 Embedded Characteristics . . . 63

12.3 Block Diagram . . . 63

12.4 Programmer Model . . . 64

12.5 Memory Management Unit . . . 71

13. L2 Cache Controller (L2CC) . . . 76

13.1 Description . . . 76
13.2 Embedded Characteristics . . . 76
13.3 Product Dependencies . . . 76
13.4 Functional Description . . . 76
13.5 L2 Cache Controller (L2CC) User Interface . . . 78

14. Debug and Test Features . . . 112
14.1 Description . . . 112
14.2 Embedded Characteristics . . . 112
14.3 Debug and Test Block Diagrams . . . 113
14.4 Application Examples . . . 115
14.5 Debug and Test Pin Description . . . 116
14.6 Functional Description . . . 117
14.7 Boundary JTAG ID Register . . . 119
14.8 Cortex-A5 DP Identification Code Register IDCODE . . . 120

15. Standard Boot Strategies . . . 122
15.1 Description . . . 122
15.2 Flow Diagram . . . 122
15.3 Chip Setup . . . 123
15.4 Boot configuration . . . 123
15.5 SAM-BA Monitor . . . 146
15.6 Fuse Box Controller . . . 150

16. AXI Matrix (AXIMX) . . . 151
16.1 Description . . . 151
16.2 Embedded Characteristics . . . 151
16.3 Operation . . . 151
16.4 AXI Matrix (AXIMX) User Interface . . . 152

17. Matrix (H64MX/H32MX) . . . 154
17.1 Description . . . 154
17.2 Embedded Characteristics . . . 154
17.3 MATRIX0 (H64MX) . . . 155
17.4 MATRIX1 (H32MX) . . . 157
17.5 Memory Mapping . . . 158
17.6 Special Bus Granting Mechanism . . . 158
17.7 No Default Master . . . 159
17.8 Last Access Master . . . 159
17.9 Fixed Default Master . . . 159
17.10 Arbitration . . . 159
17.11 Register Write Protection . . . 162
17.12 TrustZone Extension to AHB and APB . . . 162
17.13 AHB Matrix (MATRIX) User Interface . . . 176

18. Special Function Registers (SFR) . . . 195
18.1 Description . . . 195
18.2 Embedded Characteristics . . . 195
18.3 Special Function Registers (SFR) User Interface . . . 196

19. Special Function Registers Backup (SFRBU) . . . 212
19.1 Description . . . 212

19.2 Embedded Characteristics . . . 212
19.3 Special Function Registers Backup (SFRBU) User Interface . . . 213

20. Advanced Interrupt Controller (AIC) . . . 218
20.1 Description . . . 218
20.2 Embedded Characteristics . . . 218
20.3 Block Diagram . . . 219
20.4 Application Block Diagram . . . 219
20.5 AIC Detailed Block Diagram . . . 220
20.6 I/O Line Description . . . 220
20.7 Product Dependencies . . . 221
20.8 Functional Description . . . 222
20.9 Advanced Interrupt Controller (AIC) User Interface . . . 231

21. Watchdog Timer (WDT) . . . 253
21.1 Description . . . 253
21.2 Embedded Characteristics . . . 253
21.3 Block Diagram . . . 253
21.4 Functional Description . . . 254
21.5 Watchdog Timer (WDT) User Interface . . . 256

22. Reset Controller (RSTC) . . . 261
22.1 Description . . . 261
22.2 Embedded Characteristics . . . 261
22.3 Block Diagram . . . 261
22.4 Functional Description . . . 262
22.5 Reset Controller (RSTC) User Interface . . . 268

23. Shutdown Controller (SHDWC) . . . 272
23.1 Description . . . 272
23.2 Embedded Characteristics . . . 272
23.3 Block Diagram . . . 272
23.4 I/O Lines Description . . . 273
23.5 Product Dependencies . . . 273
23.6 Functional Description . . . 273
23.7 Shutdown Controller (SHDWC) User Interface . . . 275

24. Periodic Interval Timer (PIT) . . . 280
24.1 Description . . . 280
24.2 Embedded Characteristics . . . 280
24.3 Block Diagram . . . 280
24.4 Functional Description . . . 281
24.5 Periodic Interval Timer (PIT) User Interface . . . 282

25. Real-time Clock (RTC) . . . 287
25.1 Description . . . 287
25.2 Embedded Characteristics . . . 287
25.3 Block Diagram . . . 287
25.4 Product Dependencies . . . 288
25.5 Functional Description . . . 288
25.6 Real-time Clock (RTC) User Interface . . . 300

26. Slow Clock Controller (SCKC) . . . 328
26.1 Description . . . 328
26.2 Embedded Characteristics . . . 328
26.3 Block Diagram . . . 328
26.4 Functional Description . . . 328
26.5 Slow Clock Controller (SCKC) User Interface . . . 330

27. Low Power Asynchronous Receiver (RXLP) . . . 332
27.1 Description . . . 332
27.2 Embedded Characteristics . . . 332
27.3 Block Diagram . . . 332
27.4 Product Dependencies . . . 332
27.5 Functional Description . . . 333
27.6 Low Power Asynchronous Receiver (RXLP) User Interface . . . 336

28. Analog Comparator Controller (ACC) . . . 343
28.1 Description . . . 343
28.2 Embedded Characteristics . . . 343
28.3 Block Diagram . . . 343
28.4 Signal Description . . . 343
28.5 Product Dependencies . . . 343
28.6 Functional Description . . . 344
28.7 Analog Comparator Controller (ACC) User Interface . . . 345

29. Clock Generator . . . 350
29.1 Description . . . 350
29.2 Embedded Characteristics . . . 350
29.3 Block Diagram . . . 351
29.4 Slow Clock . . . 352
29.5 Main Clock . . . 353
29.6 Divider and PLLA Block . . . 356
29.7 UTMI Phase Lock Loop Programming . . . 357
29.8 Audio PLL . . . 357

30. Power Management Controller (PMC) . . . 359
30.1 Description . . . 359
30.2 Embedded Characteristics . . . 359
30.3 Block Diagram . . . 360
30.4 Master Clock Controller . . . 361
30.5 Processor Clock Controller . . . 361
30.6 Matrix Clock Controller . . . 362
30.7 LCDC Clock Controller . . . 362
30.8 ISC Clock Controller . . . 362
30.9 USB Device and Host Clocks . . . 362
30.10 DDR2/LPDDR/LPDDR2 Clock . . . 363
30.11 Fast Startup from Ultra Low-power (ULP) Mode 0 . . . 364
30.12 Fast Startup from Ultra Low-power (ULP) Mode 1 . . . 365
30.13 Peripheral Clock Controller . . . 367
30.14 Asynchronous Partial Wakeup (SleepWalking) . . . 367
30.15 Programmable Clock Controller . . . 369
30.16 Generic Clock Controller . . . 369
30.17 Main Clock Failure Detector . . . 369

30.18 32.768 kHz Crystal Oscillator Frequency Monitor . . . 370
30.19 Programming Sequence . . . 371
30.20 Clock Switching Details . . . 373
30.21 Register Write Protection . . . 377
30.22 Power Management Controller (PMC) User Interface . . . 378

31. Parallel Input/Output Controller (PIO) . . . 424
31.1 Description . . . 424
31.2 Embedded Characteristics . . . 424
31.3 Block Diagram . . . 425
31.4 Product Dependencies . . . 426
31.5 Functional Description . . . 426
31.6 I/O Lines Programming Example . . . 438
31.7 Parallel Input/Output Controller (PIO) User Interface . . . 440

32. External Memories . . . 478
32.1 Multiport DDR-SDRAM Controller (MPDDRC) . . . 479
32.2 External Bus Interface (EBI) . . . 487

33. Multiport DDR-SDRAM Controller (MPDDRC) . . . 490
33.1 Description . . . 490
33.2 Embedded Characteristics . . . 491
33.3 MPDDRC Module Diagram . . . 492
33.4 Product Dependencies, Initialization Sequence . . . 493
33.5 Functional Description . . . 502
33.6 Software Interface/SDRAM Organization, Address Mapping . . . 518
33.7 AHB Multiport DDR-SDRAM Controller (MPDDRC) User Interface . . . 525

34. Static Memory Controller (SMC) . . . 571
34.1 Description . . . 571
34.2 Embedded Characteristics . . . 572
34.3 Block Diagram . . . 573
34.4 I/O Lines Description . . . 573
34.5 Multiplexed Signals . . . 574
34.6 Application Example . . . 574
34.7 Product Dependencies . . . 575
34.8 External Memory Mapping . . . 575
34.9 Connection to External Devices . . . 576
34.10 Standard Read and Write Protocols . . . 578
34.11 Scrambling/Unscrambling Function . . . 584
34.12 Automatic Wait States . . . 585
34.13 Data Float Wait States . . . 588
34.14 External Wait . . . 592
34.15 Slow Clock Mode . . . 598
34.16 Register Write Protection . . . 600
34.17 NFC Operations . . . 600
34.18 PMECC Controller Functional Description . . . 612
34.19 Software Implementation . . . 618
34.20 Static Memory Controller (SMC) User Interface . . . 623

35. DMA Controller (XDMAC) . . . 676
35.1 Description . . . 676

35.2 Embedded Characteristics . . . 676
35.3 Block Diagram . . . 677
35.4 DMA Controller Peripheral Connections . . . 678
35.5 Functional Description . . . 682
35.6 Linked List Descriptor Operation . . . 686
35.7 XDMAC Maintenance Software Operations . . . 688
35.8 XDMAC Software Requirements . . . 689
35.9 Extensible DMA Controller (XDMAC) User Interface . . . 690

36. LCD Controller (LCDC) . . . 728
36.1 Description . . . 728
36.2 Embedded Characteristics . . . 728
36.3 Block Diagram . . . 729
36.4 I/O Lines Description . . . 729
36.5 Product Dependencies . . . 730
36.6 Functional Description . . . 732
36.7 LCD Controller (LCDC) User Interface . . . 764

37. Ethernet MAC (GMAC) . . . 943
37.1 Description . . . 943
37.2 Embedded Characteristics . . . 943
37.3 Block Diagram . . . 944
37.4 Signal Interfaces . . . 944
37.5 Product Dependencies . . . 945
37.6 Functional Description . . . 947
37.7 Programming Interface . . . 974
37.8 Ethernet MAC (GMAC) User Interface . . . 978

38. USB High Speed Device Port (UDPHS) . . . 1112
38.1 Description . . . 1112
38.2 Embedded Characteristics . . . 1112
38.3 Block Diagram . . . 1113
38.4 Typical Connection . . . 1114
38.5 Product Dependencies . . . 1114
38.6 Functional Description . . . 1115
38.7 USB High Speed Device Port (UDPHS) User Interface . . . 1138

39. USB Host High Speed Port (UHPHS) . . . 1191
39.1 Description . . . 1191
39.2 Embedded Characteristics . . . 1191
39.3 Block Diagram . . . 1192
39.4 Typical Connection . . . 1193
39.5 Product Dependencies . . . 1193
39.6 Functional Description . . . 1195
39.7 USB Host High Speed Port (UHPHS) User Interface . . . 1196

40. Audio Class D Amplifier (CLASSD) . . . 1229
40.1 Description . . . 1229
40.2 Embedded Characteristics . . . 1229
40.3 Block Diagram . . . 1230
40.4 Pin Name List . . . 1230
40.5 Product Dependencies . . . 1231

40.6 Functional Description . . . 1231
40.7 Audio Class D Amplifier (CLASSD) User Interface . . . 1245

41. Inter-IC Sound Controller (I2SC) . . . 1258
41.1 Description . . . 1258
41.2 Embedded Characteristics . . . 1258
41.3 Block Diagram . . . 1259
41.4 I/O Lines Description . . . 1259
41.5 Product Dependencies . . . 1260
41.6 Functional Description . . . 1261
41.7 I2SC Application Examples . . . 1266
41.8 Inter-IC Sound Controller (I2SC) User Interface . . . 1268

42. Synchronous Serial Controller (SSC) . . . 1281
42.1 Description . . . 1281
42.2 Embedded Characteristics . . . 1281
42.3 Block Diagram . . . 1282
42.4 Application Block Diagram . . . 1282
42.5 SSC Application Examples . . . 1283
42.6 Pin Name List . . . 1284
42.7 Product Dependencies . . . 1285
42.8 Functional Description . . . 1287
42.9 Synchronous Serial Controller (SSC) User Interface . . . 1298

43. Two-wire Interface (TWIHS) . . . 1325
43.1 Description . . . 1325
43.2 Embedded Characteristics . . . 1326
43.3 List of Abbreviations . . . 1326
43.4 Block Diagram . . . 1326
43.5 Product Dependencies . . . 1327
43.6 Functional Description . . . 1328
43.7 Two-wire Interface High Speed (TWIHS) User Interface . . . 1380

44. Flexible Serial Communication Controller (FLEXCOM) . . . 1413
44.1 Description . . . 1413
44.2 Embedded Characteristics . . . 1414
44.3 Block Diagram . . . 1416
44.4 I/O Lines Description . . . 1417
44.5 Product Dependencies . . . 1417
44.6 Register Accesses . . . 1419
44.7 USART Functional Description . . . 1419
44.8 SPI Functional Description . . . 1470
44.9 TWI Functional Description . . . 1490
44.10 Flexible Serial Communication Unit (FLEXCOM) User Interface . . . 1541

45. Universal Asynchronous Receiver Transmitter (UART) . . . 1661
45.1 Description . . . 1661
45.2 Embedded Characteristics . . . 1661
45.3 Block Diagram . . . 1661
45.4 Product Dependencies . . . 1662
45.5 Functional Description . . . 1663
45.6 Universal Asynchronous Receiver Transmitter (UART) User Interface . . . 1674

46. Serial Peripheral Interface (SPI) . . . 1689
46.1 Description . . . 1689
46.2 Embedded Characteristics . . . 1689
46.3 Block Diagram . . . 1690
46.4 Application Block Diagram . . . 1690
46.5 Signal Description . . . 1691
46.6 Product Dependencies . . . 1691
46.7 Functional Description . . . 1693
46.8 Serial Peripheral Interface (SPI) User Interface . . . 1714

47. Quad Serial Peripheral Interface (QSPI) . . . 1738
47.1 Description . . . 1738
47.2 Embedded Characteristics . . . 1738
47.3 Block Diagram . . . 1739
47.4 Signal Description . . . 1739
47.5 Product Dependencies . . . 1740
47.6 Functional Description . . . 1741
47.7 Quad Serial Peripheral Interface (QSPI) User Interface . . . 1761

48. Secure Digital Multimedia Card Controller (SDMMC) . . . 1781
48.1 Description . . . 1781
48.2 Embedded Characteristics . . . 1781
48.3 Embedded Features for SDMMC0 and SDMMC1 . . . 1782
48.4 Reference Documents . . . 1782
48.5 Block Diagram . . . 1783
48.6 Application Block Diagram . . . 1784
48.7 Pin Name List . . . 1784
48.8 Product Dependencies . . . 1785
48.9 SD/SDIO Operating Mode . . . 1785
48.10 e.MMC Operating Mode . . . 1785
48.11 SDR104 / HS200 Tuning . . . 1787
48.12 I/O Calibration . . . 1790
48.13 Secure Digital MultiMedia Card Controller (SDMMC) User Interface . . . 1791

49. Image Sensor Controller (ISC) . . . 1882
49.1 Description . . . 1882
49.2 Embedded Characteristics . . . 1882
49.3 Block Diagram and Use Cases . . . 1883
49.4 Product Dependencies . . . 1885
49.5 Functional Description . . . 1888
49.6 Image Sensor Controller (ISC) User Interface . . . 1909

50. Controller Area Network (MCAN) . . . 1977
50.1 Description . . . 1977
50.2 Embedded Characteristics . . . 1977
50.3 Block Diagram . . . 1978
50.4 Product Dependencies . . . 1978
50.5 Functional Description . . . 1980
50.6 Controller Area Network (MCAN) User Interface . . . 2008

51. Timer Counter (TC) . . . 2065
51.1 Description . . . 2065

51.2 Embedded Characteristics . . . 2065
51.3 Block Diagram . . . 2066
51.4 Pin List . . . 2067
51.5 Product Dependencies . . . 2067
51.6 Functional Description . . . 2069
51.7 Timer Counter (TC) User Interface . . . 2092

52. Pulse Density Modulation Interface Controller (PDMIC) . . . 2126
52.1 Description . . . 2126
52.2 Embedded Characteristics . . . 2126
52.3 Block Diagram . . . 2126
52.4 Signal Description . . . 2126
52.5 Product Dependencies . . . 2127
52.6 Functional Description . . . 2127
52.7 Pulse Density Modulation Interface Controller (PDMIC) User Interface . . . 2134

53. Pulse Width Modulation Controller (PWM) . . . 2146
53.1 Description . . . 2146
53.2 Embedded Characteristics . . . 2147
53.3 Block Diagram . . . 2148
53.4 I/O Lines Description . . . 2149
53.5 Product Dependencies . . . 2149
53.6 Functional Description . . . 2151
53.7 Pulse Width Modulation Controller (PWM) User Interface . . . 2191

54. Secure Fuse Controller (SFC) . . . 2247
54.1 Description . . . 2247
54.2 Embedded Characteristics . . . 2247
54.3 Block Diagram . . . 2248
54.4 Functional Description . . . 2248
54.5 Secure Fuse Controller (SFC) User Interface . . . 2250

55. Integrity Check Monitor (ICM) . . . 2258
55.1 Description . . . 2258
55.2 Embedded Characteristics . . . 2259
55.3 Block Diagram . . . 2259
55.4 Product Dependencies . . . 2260
55.5 Functional Description . . . 2260
55.6 Integrity Check Monitor (ICM) User Interface . . . 2273

56. Advanced Encryption Standard Bridge (AESB) . . . 2287
56.1 Description . . . 2287
56.2 Embedded Characteristics . . . 2287
56.3 Product Dependencies . . . 2287
56.4 Functional Description . . . 2288
56.5 Security Features . . . 2291
56.6 Advanced Encryption Standard Bridge (AESB) User Interface . . . 2292

57. Advanced Encryption Standard (AES) . . . 2304
57.1 Description . . . 2304
57.2 Embedded Characteristics . . . 2304
57.3 Product Dependencies . . . 2305

57.4 Functional Description . . . 2305
57.5 Advanced Encryption Standard (AES) User Interface . . . 2327

58. Secure Hash Algorithm (SHA) . . . 2351
58.1 Description . . . 2351
58.2 Embedded Characteristics . . . 2351
58.3 Product Dependencies . . . 2351
58.4 Functional Description . . . 2352
58.5 Secure Hash Algorithm (SHA) User Interface . . . 2360

59. Triple Data Encryption Standard (TDES) . . . 2373
59.1 Description . . . 2373
59.2 Embedded Characteristics . . . 2373
59.3 Product Dependencies . . . 2373
59.4 Functional Description . . . 2374
59.5 Triple Data Encryption Standard (TDES) User Interface . . . 2380

60. True Random Number Generator (TRNG) . . . 2395
60.1 Description . . . 2395
60.2 Embedded Characteristics . . . 2395
60.3 Block Diagram . . . 2395
60.4 Product Dependencies . . . 2395
60.5 Functional Description . . . 2396
60.6 True Random Number Generator (TRNG) User Interface . . . 2397

61. Analog-to-Digital Converter (ADC) . . . 2404
61.1 Description . . . 2404
61.2 Embedded Characteristics . . . 2405
61.3 Block Diagram . . . 2406
61.4 Signal Description . . . 2406
61.5 Product Dependencies . . . 2406
61.6 Functional Description . . . 2408
61.7 Analog-to-Digital (ADC) User Interface . . . 2436

62. Electrical Characteristics . . . 2472
62.1 Absolute Maximum Ratings . . . 2472
62.2 DC Characteristics . . . 2472
62.3 Power Consumption . . . 2476
62.4 Active Mode . . . 2476
62.5 Low-power Modes . . . 2479
62.6 Clock Characteristics . . . 2485
62.7 Oscillator Characteristics . . . 2486
62.8 PLL Characteristics . . . 2489
62.9 USB HS Characteristics . . . 2490
62.10 ADC Characteristics . . . 2491
62.11 Analog Comparator Characteristics . . . 2497
62.12 POR Characteristics . . . 2498
62.13 SMC Timings . . . 2499
62.14 SPI Timings . . . 2504
62.15 FLEXCOM Timings . . . 2512
62.16 QSPI Timings . . . 2524
62.17 TWI Timings . . . 2529

62.18 MPDDRC Timings . . . 2531
62.19 SSC Timings . . . 2532
62.20 PDMIC Timings . . . 2540
62.21 I2SC Timings . . . 2541
62.22 ISC Timings . . . 2543
62.23 SDMMC Timings . . . 2544
62.24 GMAC Timings . . . 2545

63. Mechanical Characteristics . . . 2549
63.1 289-ball LFBGA Mechanical Characteristics . . . 2549
63.2 256-ball TFBGA Mechanical Characteristics . . . 2551
63.3 196-ball TFBGA Mechanical Characteristics . . . 2553

64. Schematic Checklist . . . 2555
64.1 Power Supply . . . 2556
64.2 Power-On Reset . . . 2559
64.3 Shutdown Considerations . . . 2559
64.4 Wakeup Considerations . . . 2559
64.5 Clock, Oscillator and PLL . . . 2560
64.6 ICE and JTAG . . . 2563
64.7 Reset and Test . . . 2564
64.8 Shutdown/Wakeup Logic . . . 2564
64.9 Parallel Input/Output (PIO) . . . 2564
64.10 Analog-to-Digital Converter (ADC) . . . 2564
64.11 External Bus Interface (EBI) . . . 2565
64.12 USB High-Speed Host Port (UHPHS) / USB High-Speed Device Port (UDPHS) . . . 2567
64.13 Boot Program Hardware Constraints . . . 2568
64.14 Layout and Design Constraints . . . 2568

65. Marking . . . 2571

66. Ordering Information . . . 2572

67. Errata . . . 2573
67.1 Errata - SAMA5D2 MRL-B Parts . . . 2573
67.2 Errata - SAMA5D2 MRL-A Parts . . . 2576

68. Revision History . . . 2582

Table of Contents . . . i

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**

© 2016 Atmel Corporation. / Rev.: Atmel-11267E-ATARM-SAMA5D2-Datasheet_25-Jul-16.

Atmel®, Atmel logo and combinations thereof, Enabling Unlimited Possibilities®, and others are registered trademarks or trademarks of Atmel Corporation in U.S. and other countries. ARM®, ARM Connected® logo, and others are the registered trademarks or trademarks of ARM Ltd. Windows® is a registered trademark of Microsoft Corporation in U.S. and/or other countries. Other terms and product names may be trademarks of others.

DISCLAIMER: The information in this document is provided in connection with Atmel products. No license, express or implied, by estoppel or otherwise, to any intellectual property right is granted by this document or in connection with the sale of Atmel products. EXCEPT AS SET FORTH IN THE ATMEL TERMS AND CONDITIONS OF SALES LOCATED ON THE ATMEL WEBSITE, ATMEL ASSUMES NO LIABILITY WHATSOEVER AND DISCLAIMS ANY EXPRESS, IMPLIED OR STATUTORY WARRANTY RELATING TO ITS PRODUCTS INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL, PUNITIVE, SPECIAL OR INCIDENTAL DAMAGES (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS AND PROFITS, BUSINESS INTERRUPTION, OR LOSS OF INFORMATION) ARISING OUT OF THE USE OR INABILITY TO USE THIS DOCUMENT, EVEN IF ATMEL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. Atmel makes no representations or warranties with respect to the accuracy or completeness of the contents of this document and reserves the right to make changes to specifications and products descriptions at any time without notice. Atmel does not make any commitment to update the information contained herein. Unless specifically provided otherwise, Atmel products are not suitable for, and shall not be used in, automotive applications. Atmel products are not intended, authorized, or warranted for use as components in applications intended to support or sustain life.

SAFETY-CRITICAL, MILITARY, AND AUTOMOTIVE APPLICATIONS DISCLAIMER: Atmel products are not designed for and will not be used in connection with any applications where the failure of such products would reasonably be expected to result in significant personal injury or death (“Safety-Critical Applications”) without an Atmel officer's specific written consent. Safety-Critical Applications include, without limitation, life support devices and systems, equipment or systems for the operation of nuclear facilities and weapons systems. Atmel products are not designed nor intended for use in military or aerospace applications or environments unless specifically designated by Atmel as military-grade. Atmel products are not designed nor intended for use in automotive applications unless specifically designated by Atmel as automotive-grade.

Mouser Electronics

Authorized Distributor

Click to View Pricing, Inventory, Delivery & Lifecycle Information:

Atmel:

ATSAMA5D27A-CU ATSAMA5D28A-CU ATSAMA5D2-XULT ATSAMA5D21A-CU ATSAMA5D23A-CNR ATSAMA5D26A-CUR ATSAMA5D28A-CUR ATSAMA5D26A-CNR ATSAMA5D28A-CN ATSAMA5D22A-CNR ATSAMA5D27A-CNR ATSAMA5D26A-CN ATSAMA5D23A-CN ATSAMA5D26BCU ATSAMA5D24BCUR ATSAMA5D26BCNR ATSAMA5D26BCUR ATSAMA5D27BCNR ATSAMA5D24BCU ATSAMA5D27BCN