CMOS 8-BIT MICROCONTROLLER

LC872H00 SERIES USER'S MANUAL

REV : 1.00

ON Semiconductor®

http://onsemi.com

ON Semiconductor

Digital Solution Division

Microcontroller & Flash Business Unit

Contents

Contents

Contents

Contents

Contents

Contents

1. Overview

1.1 Overview

The SANYO LC872H00 series comprises 8-bit microcomputers that, centered around a CPU running at a minimum bus cycle time of 83.3 ns, integrate on a single chip a number of hardware features such as 8K-byte flash ROM (onboard programmable), 256-byte RAM, two sophisticated 16-bit timer/counter devices (may be divided into 8-bit timers), two 8-bit timers with a prescaler, a base timer serving as a time-of-day clock, a synchronous SIO (with automatic block transmission/reception capabilities), an asynchronous/synchronous SIO interface, an UART (full duplex), 12-bit multifrequency PWM×2 channels, 12-bit, 9-channel AD converter with 12/8-bit resolution selector, a high-speed clock counter, a system clock frequency divider, an internal reset circuit, and 20-source 10-vector interrupt feature.

1.2 Features

● **Flash ROM**
- Programmable onboard with a wide range of power source (2.2 to 5.5V).
- Block-erasable in 128-byte units
- Writable in 2-byte units
- 8192 × 8 bits

● **RAM**
- 256 × 9 bits

● **Minimum bus cycle time**
- 83.3 ns(12MHz, VDD＝2.7V to 5.5V)
- 100 ns (10MHz, VDD＝2.2V to 5.5V)
- 250 ns (4MHz, VDD＝1.8V to 5.5V)

  Note: The bus cycle time here refers to the ROM read speed.

● **Minimum instruction cycle time (Tcyc)**
- 250ns (12MHz, VDD＝2.7V to 5.5V)
- 300ns (10MHz, VDD＝2.2V to 5.5V)
- 750ns (4MHz, VDD＝1.8V to 5.5V)

● **Ports**
- ● Normal withstand voltage I/O ports

Ports whose I/O direction can be designated in 1 bit units:	16 (P1n, P20, P21, P30, P31, P70 to P73)
Ports whose I/O direction can be designated in 4 bit units:	8 (P0n)

- ● Dedicated oscillator/input ports: 2 (CF1/XT1, CF2/XT2)
- ● Reset pins   1 ($\overline{\text{RES}}$)
- ● Power pins: 3 (VSS1, VSS2, VDD1)

● **Timers**
- ● Timer 0: 16-bit timer/counter with capture registers

Mode 0:	8-bit timer with an 8-bit programmable prescaler (with 8-bit capture registers) × 2 channels
Mode 1:	8-bit timer with an 8-bit programmable prescaler (with 8-bit capture registers) + 8-bit counter (with 8-bit capture registers)
Mode 2:	16-bit timer with an 8-bit programmable prescaler (with 16-bit capture registers)
Mode 3:	16-bit counter (with 16-bit capture registers)

- Timer 1: 16-bit timer/counter that supports PWM/toggle outputs

  Mode 0:   8-bit timer with an 8-bit prescaler (with toggle outputs) + 8-bit timer/counter (with toggle outputs)

  Mode 1:   8-bit PWM with an 8-bit prescaler × 2 channels

  Mode 2:   16-bit timer/counter with an 8-bit prescaler (with toggle outputs) (toggle outputs also from the lower-order 8 bits)

  Mode 3:   16-bit timer with an 8-bit prescaler (with toggle outputs) (The lower-order 8 bits can be used as PWM.)

- Timer 6: 8-bit timer with a 6-bit prescaler (with toggle output)

- Timer 7: 8-bit timer with a 6-bit prescaler (with toggle output)

- Base timer

  1) The clock is selectable from the subclock (32.768kHz crystal oscillation), system clock, and timer 0 prescaler output.

  2) Interrupts programmable in 5 different time schemes

  3) The base timer is unavailable when the CF oscillator circuit is selected.

● High-speed clock counter

  1) Can count clocks with a maximum clock rate of 20 MHz (at a main clock of 10 MHz).

  2) Can generate output real time.

● SIO

- SIO0: 8-bit synchronous serial interface

  1) LSB first/MSB first mode selectable

  2) Built-in 8-bit baudrate generator (maximum transfer clock cycle = 4/3 Tcyc)

- SIO1: 8-bit asynchronous/synchronous serial interface

  Mode 0:   Synchronous 8-bit serial I/O (2-or 3-wire configuration, 2 to 512 Tcyc transfer clocks)

  Mode 1:   Asynchronous serial I/O(half-duplex, 8 data bits, 1 stop bit, 8 to 2048Tcyc baudrates)

  Mode 2:   Bus mode 1 (start bit, 8 data bits, 2 to 512 Tcyc transfer clocks)

  Mode 3:   Bus mode 2 (start detect, 8 data bits, stop detect)

● UART1

- Full duplex

- 7/8/9 bit data bits selectable

- 1 stop bit (2 bits in continuous data transmission)

- Built-in baudrate generator

● AD converter: 12 bits × 9 channels

- 12/8 bit AD converter resolution selectable.

● PWM: Multifrequency 12-bit PWM × 2 channels

● Remote control receiver circuit (multiplexed with the P73/INT3/T0 IN pin)

- Noise rejection function (noise filter time constant selectable from 1Tcyc/32Tcyc/128Tcyc.)

● Clock output function

  1) Can generate clock outputs with a frequency of $\frac{1}{1}$, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$, or $\frac{1}{64}$ of the source clock selected as the system clock.

  2) Can generate the source clock for the subclock.

● **Watchdog timer**

    1) External RC watchdog timer

    2) Interrupt and reset signals selectable

● **Interrupts**

  ● 20 sources, 10 vector addresses

    1) Provides three levels (low (L), high (H), and highest (X)) of multiplex interrupt control. Any interrupt requests of the level equal to or lower than the current interrupt are not accepted.

    2) When interrupt requests to two or more vector addresses occur at the same time, the interrupt of the highest level takes precedence over the other interrupts. For interrupts of the same level, the interrupt into the smallest vector address takes precedence.

No.	Vector	Level	Interrupt Source
1	00003H	X or L	INT0
2	0000BH	X or L	INT1
3	00013H	H or L	INT2/T0L/INT4
4	0001BH	H or L	INT3/ INT5/base timer
5	00023H	H or L	T0H
6	0002BH	H or L	T1L/T1H
7	00033H	H or L	SIO0/UART1 receive
8	0003BH	H or L	SIO1/UART1 transmit
9	00043H	H or L	ADC/T6/T7/PWM4, PWM5
10	0004BH	H or L	Port 0

  • Priority levels X > H > L

  • Of interrupts of the same level, the one with the smallest vector address takes precedence.

● **Subroutine stack levels: 128 levels maximum (The stack is allocated in RAM.)**

● **High-speed multiplication/division instructions**

    • 16 bits × 8 bits   (5 Tcyc execution time)

    • 24 bits × 16 bits   (12 Tcyc execution time)

    • 16 bits ÷ 8 bits   (8 Tcyc execution time)

    • 24 bits ÷ 16 bits   (12 Tcyc execution time)

● **Oscillation circuits**

  ● Internal oscillation circuits

    1) Low-speed RC oscillation circuit: For system clock (100 k Hz)

    2) Medium-speed RC   oscillation circuit: For system clock (1 MHz)

    3) Multifrequency RC oscillation circuit: For system clock (8 MHz)

  ● External oscillation circuits

    1) High-speed CF oscillation circuit: For system clock, with internal Rf

    2) Low-speed crystal oscillation circuit: For low-speed system clock, with internal Rf

      (1) The CF and crystal oscillation circuits share the same pins. The active circuit is selected under program control.

      (2) Both the CF and crystal oscillator circuits stop operation on a system reset. When the reset is released, only the CF oscillation circuit resumes operation.

● **System clock divider function**

    • Can run on low current.

    • The minimum instruction cycle selectable from 300ns, 600ns, 1.2 μ s, 2.4 μ s, 4.8 μ s, 9.6 μ s, 19.2 μ s, 38.4 μ s, 76.8 μ s (at a main clock rate of 10 MHz).

● **Internal reset circuit**
  ● Power-on reset (POR) function

    1) POR reset is generated only at power-on time.

    2) The POR release level can be selected from 8 levels (1.67V, 1.97V, 2.07V, 2.37V, 2.57V, 2.87V, 3.86V, and 4.35V) through option configuration.

  ● Low-voltage detection reset (LVD) function

    1) LVD and POR functions are combined to generate resets when power is turned on and when power voltage falls below a certain level.

    2) The use/disuse of the LVD function and the low voltage threshold level (7 levels: 1.91V, 2.01V, 2.31V, 2.51V, 2.81V, 3.79V, 4.28V).

● **Standby function**
  ● HALT mode: Halts instruction execution while allowing the peripheral circuits to continue operation

    1) Oscillation is not stopped automatically.

    2) There are three ways of resetting the HALT mode.
       (1) Setting the reset pin to the low level
       (2) System resetting by watchdog timer or low-voltage detection
       (3) Occurrence of an interrupt

  ● HOLD mode: Suspends instruction execution and the operation of the peripheral circuits.

    1) The CF, low-/medium-speed RC, and crystal oscillators automatically stop operation.

    2) There are four ways of resetting the HOLD mode.
       (1) Setting the reset pin to the lower level.
       (2) System resetting by watchdog timer or low-voltage detection
       (3) Having an interrupt source established at either INT0, INT1, INT2, INT4 or INT5
          * INT0 and INT1 HOLD mode reset is available only when level detection is set.
       (4) Having an interrupt source established at port 0.

  ● X'tal HOLD mode: Suspends instruction execution and the operation of the peripheral circuits except the base timer (when X'tal oscillator is selected).

    1) The CF and low-/medium-speed RC oscillators automatically stop operation.

    2) The state of crystal oscillation established when the X'tal HOLD mode is entered is retained.

    3) There are five ways of resetting the X'tal HOLD mode.
       (1) Setting the reset pin to the low level.
       (2) System resetting by watchdog timer or low-voltage detection.
       (3) Having an interrupt source established at either INT0, INT1, INT2, INT4 or INT5
          * INT0 and INT1 HOLD mode reset is available only when level detection is set.
       (4) Having an interrupt source established at port 0.
       (5) Having an interrupt source established in the base timer circuit.
    Note: Available only when X'tal oscillation is selected.

● **On-chip Debugger Function**
  • Supports software debugging with the IC mounted on the target board.
  • For a small pin package, two channels of Onchip Debugger ports ((DBGP0(P0), DBGP1(P1)) are equipped.

● **Data security function (flash versions only)**
  • Protects the program data stored in flash memory from unauthorized read or copy.
  Note: This data security function does not necessarily provide absolute data security.

● **Package form**
  • QFP36 (7 × 7) (lead-free type)

● **Development tools**
  • On-chip debugger: TCB87 Type B + LC87F2H08A

● **Programming board**

Package	Programming board
QFP36	W87F24Q

1.3 Pinout

Top pins (left to right, 27–19):
- 27 P03/AN3
- 26 P02/AN2
- 25 P01/AN1
- 24 P00/AN0
- 23 VSS2
- 22 N.C.
- 21 P31/PWM5/INT5/T1IN
- 20 P30/PWM4/INT5/T1IN
- 19 P21/URX/INT4/T1IN

Left pins (28–36):
- P04/AN4 — 28
- P05/AN5/CKO/DBGP00 — 29
- P06/AN6/T6O/DBGP01 — 30
- P07/T7O/DBGP02 — 31
- N.C. — 32
- N.C. — 33
- P70/INT0/T0LCP/AN8 — 34
- P71/INT1/T0HCP/AN9 — 35
- P72/INT2/T0IN — 36

TOP VIEW

Right pins (18–10):
- 18 — P20/UTX/INT4/T1IN
- 17 — P17/T1PWMH/BUZ
- 16 — P16/T1PWML
- 15 — N.C.
- 14 — N.C.
- 13 — P15/SCK1/DBGP10
- 12 — P14/SI1/SB1/DBGP11
- 11 — P13/SO1/DBGP12
- 10 — P12/SCK0

Bottom pins (left to right, 1–9):
- 1 P73/INT3/T0IN
- 2 RES#
- 3 I.C.
- 4 VSS1
- 5 CF1/XT1
- 6 CF2/XT2
- 7 VDD1
- 8 P10/SO0
- 9 P11/SI0/SB0

SANYO: QFP36 (lead-free type)

Note   I.C. and N.C. pins must be held open (disconnected).

1.4   System Block Diagram
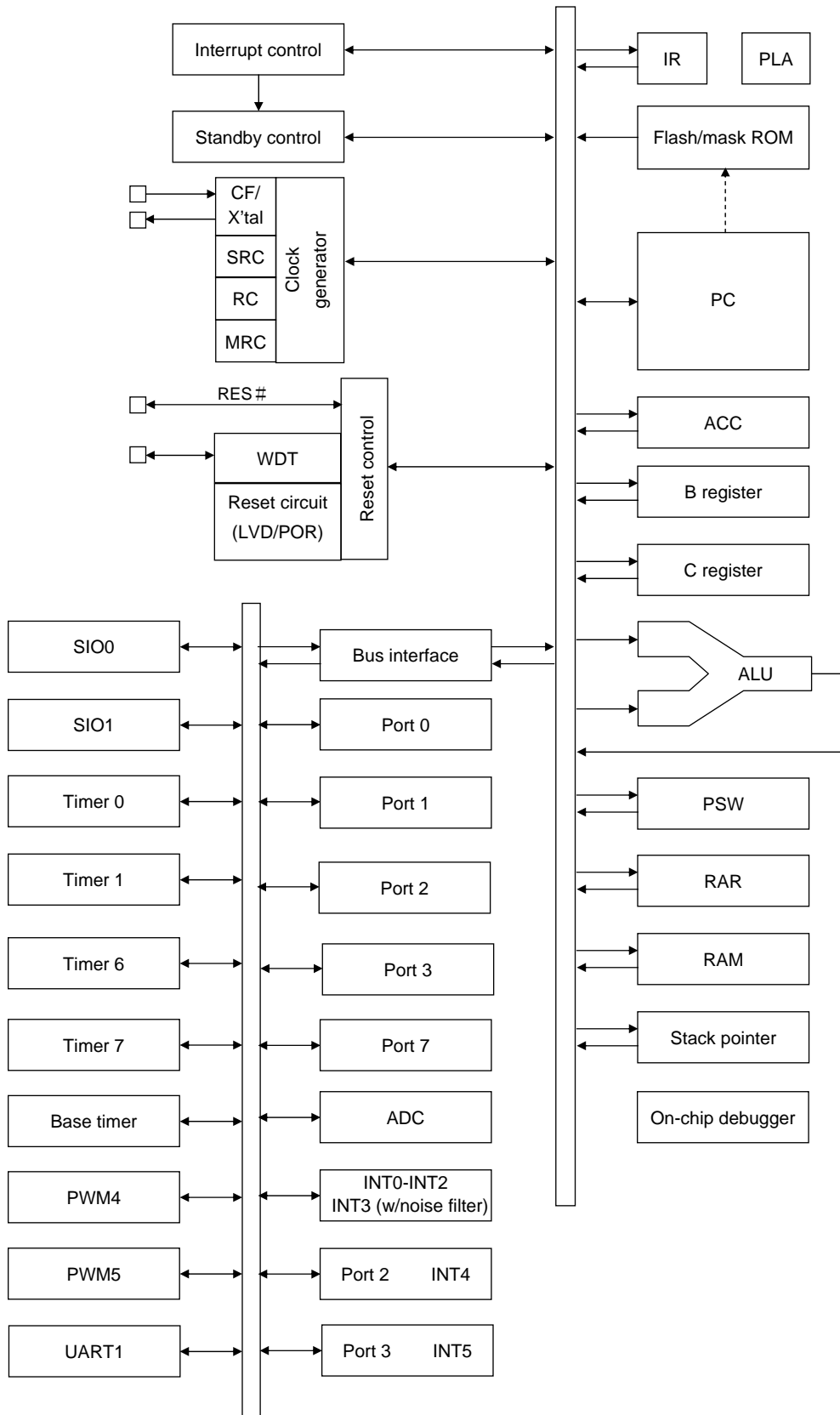
1.5　Pin Functions

Name	I/O	Description	Option
VSS1, VSS2	−	– Power supply	No
VDD1	−	+ Power supply	No
Port 0 P00 to P07	I/O	• 8-bit I/O port • I/O specifiable in 4-bit units • Pull-up resistors can be turned on and off in 4-bit units • HOLD release input • Port 0 interrupt input • Pin functions 　　　P05: System clock output 　　　P06: Timer 6 toggle output 　　　P07: Timer 7 toggle output 　　　P00 (AN0) to P06 (AN6): AD converter input 　　　P05 (DBGP00) to P07 (DBGP02): On-chip debugger-0 pins	Yes
Port 1 P10 to P17	I/O	• 8-bit I/O port • I/O specifiable in 1-bit units • Pull-up resistors can be turned on and off in 1-bit units • Pin functions 　　　P10: SIO0 data output 　　　P11: SIO0 data input/bus I/O 　　　P12: SIO0 clock I/O 　　　P13: SIO1 data output 　　　P14: SIO1 data input/bus I/O 　　　P15: SIO1 clock I/O 　　　P16: Timer 1 PWML output 　　　P17: Timer 1 PWMH output/beeper output 　　　P15(DBGP10) to P13(DBGP12): On-chip debugger -1 pins	Yes
Port 2 P20, P21	I/O	• 2-bit I/O port • I/O specifiable in 1-bit units • Pull-up resistors can be turned on and off in 1-bit units • Pin functions 　　　P20: UART transmit 　　　P21: UART receive 　　　P20, P21: INT4 input/HOLD release input/timer 1 event input 　　　　　/timer 0L capture input/timer 0H capture input Interrupt acknowledge type (table below)	Yes
Port 3 P30, P31	I/O	• 2-bit I/O port • I/O specifiable in 1-bit units • Pull-up resistors can be turned on and off in 1-bit units • Pin functions 　　P30: PWM4 output 　　P31: PWM5 output 　　P30, P31:INT5 input/HOLD release input/timer 1 event input 　　　　/timer 0L capture input/timer 0H capture input Interrupt acknowledge type (table below)	Yes

Port 2 interrupt acknowledge type:

	Rising	Falling	Rising & Falling	H level	L level
INT4	○	○	○	×	×

Port 3 interrupt acknowledge type:

	Rising	Falling	Rising & Falling	H level	L level
INT5	○	○	○	×	×

Name	I/O	Description	Option

Port 7
 	
P70 to P73
	I/O	• 4-bit I/O port
		• I/O specifiable in 1-bit units
		• Pull-up resistors can be turned on and off in 1-bit units
		• Pin functions
		    P70(AN8), P71(AN9): AD converter input
		    P70:INT0 input/HOLD release input/timer 0L capture input
		      /watchdog timer output
		    P71:INT1 input/HOLD release input/timer 0H capture input
		    P72:INT2 input/HOLD release input/timer 0 event input
		      /timer 0L capture input
		    P73:INT3 input (input with noise filtering)/timer 0 event input
		      /timer 0H capture input
		    Interrupt acknowledge type
		
		<table><tr><td></td><td>Rising</td><td>Falling</td><td>Rising & Falling</td><td>H level</td><td>L level</td></tr><tr><td>INT0</td><td>○</td><td>○</td><td>×</td><td>○</td><td>○</td></tr><tr><td>INT1</td><td>○</td><td>○</td><td>×</td><td>○</td><td>○</td></tr><tr><td>INT2</td><td>○</td><td>○</td><td>○</td><td>×</td><td>×</td></tr><tr><td>INT3</td><td>○</td><td>○</td><td>○</td><td>×</td><td>×</td></tr></table>
			No

$\overline{\text{RES}}$	I/O	External reset input/internal reset output	No
CF1/XT1	I	• Ceramic oscillator/32.768 kHz crystal oscillator input • Pin functions General-purpose input Must be configured as a general-purpose port and connected to VSS1 if not to be used.	No
CF2/XT2	I/O	• Ceramic resonator /32.768 kHz crystal resonator output • Pin functions General-purpose input Must be configured as a general-purpose port and connected to VSS1 if not to be used.	No

1.6 Onchip Debugger Pin Connection Requirements

For the treatment of the onchip debugger pins, refer to the separately available documents entitled "RD87 Onchip Debugger Installation Manual" and "LC872000 Series Onchip Debugger Pin Connection Requirements "

1.7 Recommended Unused Pin Connections

Port Name	Recommended Unused Pin Connections	
	Board	Software
P00～P07	Open	Output low
P10～P17	Open	Output low
P20～P21	Open	Output low
P30～P31	Open	Output low
P70～P73	Open	Output low
CF1/XT1	Pulled low with a 100kΩ resistor or less	General-purpose input port
CF2/XT2	Pulled low with a 100kΩ resistor or less	General-purpose input port

1.8 Port Output Types

The table below lists the types of port outputs and the presence/absence of a pull-up resistor. Data can be read into any input port even if it is in the output mode.

Port Name	Option Selected in Units of	Option Type	Output Type	Pull-up Resistor
P00 to P07	1 bit	1	CMOS	Programmable (Note 1)
		2	N-channel open drain	No
P10 to P17	1 bit	1	CMOS	Programmable
P20, P21		2	N-channel open drain	Programmable
P30, P31				
P70	―	No	N-channel open drain	Programmable
P71 to P73	―	No	CMOS	Programmable

Note 1: The control of the presence or absence of the programmable pull-up resistors for port 0 and the switching between low- and high-impedance pull-up connection is exercised in nibble (4- bit) units (P00 to 03 or P04 to 07).

1.9   User Option Table

Option name	Option to be applied on	Flash-ROM version	Option selected in units of	Option selection
Port output type	P00 to P07	○	1 bit	CMOS
				Nch-open drain
	P10 to P17	○	1 bit	CMOS
				Nch-open drain
	P20 to P21	○	1 bit	CMOS
				Nch-open drain
	P30 to P31	○	1 bit	CMOS
				Nch-open drain
Program start address	-	○	-	00000h
				01E00h
Low-voltage detection reset function	Detect function	○	-	Enable:Use
				Disable:Not Used
	Detect level	○	-	7-level
Power-on reset function	Power-On reset level	○	-	8-level

*1   Be sure to electrically short-circuit between the VSS1 and VSS2 pins.

2. Internal Configuration

2.1 Memory Space

This series of microcontrollers have the following three types of memory space:

1)   Program memory space: 256K bytes (128K bytes × 2 banks)

2)   Internal data memory space: 64K bytes (0000H to FDFFH out of 0000H to FFFFH is shared
                                     with the stack area.)

3)   External data memory space: 16M bytes
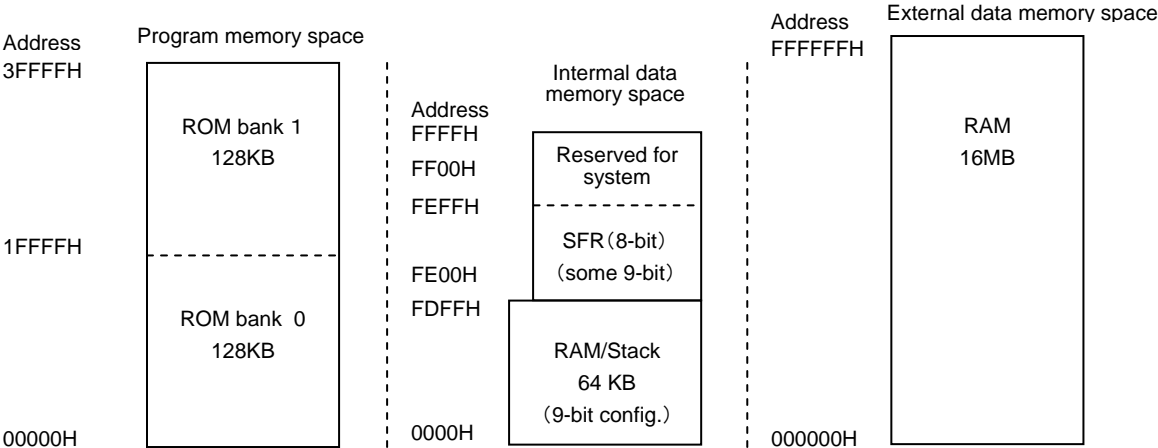
Note: SFR is the area in which special registers such as the accumulator are allocated (see Appendixes A-I).

Fig. 2.1.1 Types of Memory Space

2.2 Program Counter (PC)

The program counter (PC) is made up of 17 bits and a bank flag BNK. The value of BNK determines the bank. The lower-order 17 bits of the PC allows linear access to the 128K ROM space in the current bank.

Normally, the PC advances automatically in the current bank on each execution of an instruction. Bank switching is accomplished by executing a Return instruction after pushing necessary addresses onto the stack. When executing a branch or subroutine instruction, when accepting an interrupt, or when a reset is generated, the value corresponding to each operation is loaded into the PC.

Table 2.2.1 lists the values that are loaded into the PC when the respective operations are performed.

Table 2.2.1   Values Loaded in the PC

Operation		PC value	BNK value
Inter-rupt	Reset	00000H	0
	INT0	00003H	0
	INT1	0000BH	0
	INT2/T0L/INT4	00013H	0
	INT3 /INT5/Base timer	0001BH	0
	T0H	00023H	0
	T1L/T1H	0002BH	0
	SIO0/UART1 receive	00033H	0
	SIO1/UART1 transmit	0003BH	0
	ADC/T6/T7/PWM4, 5	00043H	0
	Port 0	0004BH	0
Unconditional branch instructions	JUMP a17	PC=a17	Unchanged
	BR r12	PC=PC+2+r12[-2048 to +2047]	Unchanged
Conditional branch instructions	BE, BNE, DBNZ, DBZ, BZ, BNZ, BZW, BNZW, BP, BN, BPC	PC=PC+nb+r8[-128 to +127] nb: Number of instruction bytes	Unchanged
Call instructions	CALL a17	PC=a17	Unchanged
	RCALL r12	PC=PC+2+r12[-2048 to +2047]	Unchanged
	RCALLA	PC=PC+1+Areg[0 to +255]	Unchanged
Return instructions	RET, RETI	PC16 to 08=(SP) PC07 to 00=(SP-1) (SP) denotes the contents of RAM address designated by the value of the stack pointer SP.	BNK is set to bit 8 of (SP-1).
Standard instructions	NOP, MOV, ADD, …	PC=PC+nb nb: Number of instruction bytes	Unchanged

2.3   Program Memory (ROM)

This series of microcontrollers have a program memory space of 256K bytes but the size of the ROM that is actually incorporated in the microcontroller varies with the CPU type of the microcontroller. The ROM table lookup instruction (LDC) can be used to refer all ROM data within the bank. Of the ROM space, the 256 bytes in ROM bank 0 (1FF00H-1FFFFH for this series of microcontrollers) are reserved as the option area. Consequently, this area is not available as a program area.

2.4   Internal Data Memory (RAM)

The LC870000 series microcontrollers have an internal data memory space of 64K bytes but the size of the RAM that is actually incorporated in the microcontroller varies with a model in the series of the microcontroller. 9 bits are used to access addresses 0000H to FDFFH of the 128K ROM space and 8 or 9 bits are used to access addresses FE00H to FFFFH. The 9th bit of RAM is implemented by bit 1 of the PSW and can be read and written.

The 128 bytes of RAM from 0000H to 007FH are paired to form 64 2-byte indirect address registers. The bit length of these indirect registers is normally 16 bits (8 bits × 2). When they are used by the ROM table lookup instruction (LDC), however, their bit length is set to 17 bits (9 higher-order bits + 8 lower-order bits).

As shown in Figure 2.4.1, the usable instructions vary depending on the address of RAM.

The efficiency improvement of use ROM and execution speed can be attempted by using these instructions properly.

FFFFH
FF00H
FEFFH

FE00H
FDFFH

2000H
1FFFH

0200H
01FFH

0100H
00FFH

0000H

Reserved for system

SFR space
* 8-bit

RAM/
Stack space
9-bit

*Note: Some registers are 9-bit

Bit instruction direct (long)

Bit instruction direct (short)

Non-bit instruction direct (long)/indirect,16-bit operation instruction direct/indirect

Non-bit instruction direct (short)
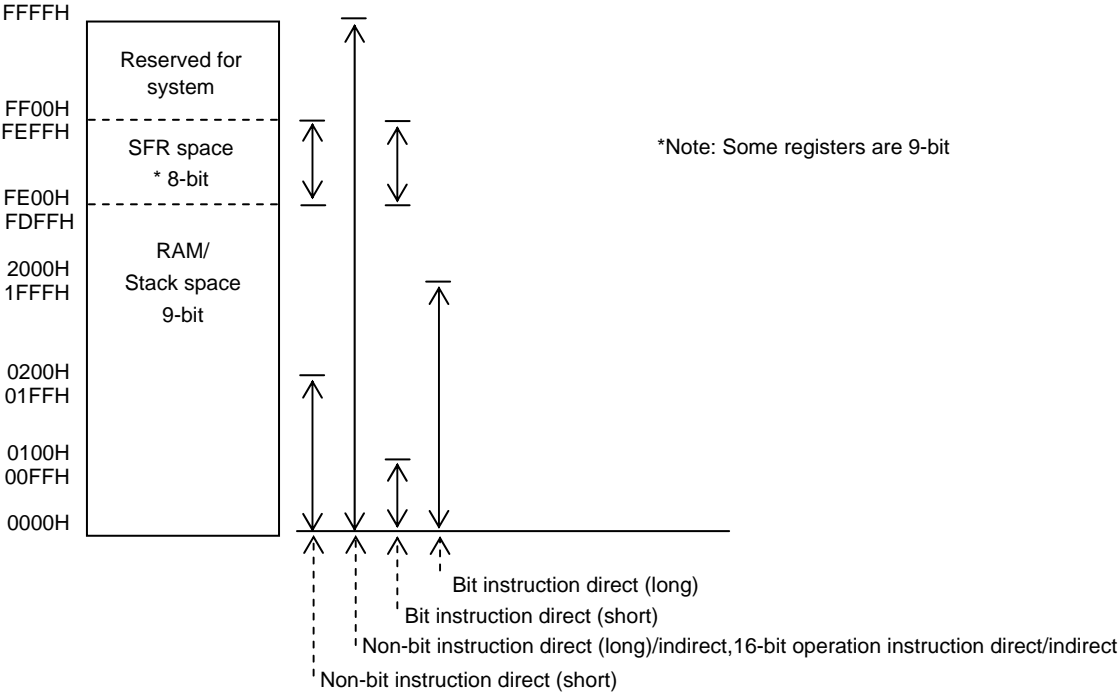
Fig. 2.4.1 RAM Addressing Map

When the value of the PC is stored in RAM during the execution of a subroutine call instruction or interrupt, assuming that SP represents the current value of the stack pointer, the value of BNK and the lower-order 8 bits of the (17-bit) PC are stored in RAM address SP + 1 and the higher-order 9 bits in SP + 2, after which SP is set to SP + 2.

2.5   Accumulator/A Register (ACC/A)

The accumulator (ACC), also called the A register, is an 8-bit register that is used for data computation, transfer, and I/O processing. It is allocated to address FE00H in the internal data memory space and initialized to 00H on a reset.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE00	0000 0000	R/W	AREG	AREG7	AREG6	AREG5	AREG4	AREG3	AREG2	AREG1	AREG0

2.6   B Register (B)

The B register is combined with the ACC to form a 16-bit arithmetic register during the execution of a 16-bit arithmetic instruction. During a multiplication or division instruction, the B register is used with the ACC and C register to store the results of computation. In addition, during an external memory access instruction (LDX or STX), the B register designates the higher-order 8 bits of the 24-bit address.

The B register is allocated to address FE01H of the internal data memory space and initialized to 00H on a reset.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE01	0000 0000	R/W	BREG	BREG7	BREG6	BREG5	BREG4	BREG3	BREG2	BREG1	BREG0

2.7   C Register (C)

The C register is used with the ACC and B register to store the results of computation during the execution of a multiplication or division instruction. In addition, during a C register offset indirect instruction, the C register stores the offset data (-128 to +127) to the contents of an indirect register.

The C register is allocated to address FE02H of the internal data memory space and initialized to 00H on a reset.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE02	0000 0000	R/W	CREG	CREG7	CREG6	CREG5	CREG4	CREG3	CREG2	CREG1	CREG0

2.8   Program Status Word (PSW)

The program status word (PSW) is made up of flags that indicate the status of computation results, a flag to access the 9th bit of RAM, and a flag to designate the bank during the LDCW instruction. The PSW is allocated to address FE06H of the internal data memory space and initialized to 00H on a reset.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE06	0000 0000	R/W	PSW	CY	AC	PSWB5	PSWB4	LDCBNK	OV	P1	PARITY

CY (bit 7): Carry flag

CY is set (to 1) when a carry occurs as the result of a computation and cleared (to 0) when no carry occurs. There are the following types of carries:

1)   Carry resulting from an addition

2)   Borrow resulting from a subtraction

3)   Borrow resulting from a comparison

4)   Carry resulting from a rotation

There are some instructions that do not affect this flag at all.

AC (bit 6): Auxiliary carry flag

AC is set (to 1) when a carry or borrow occurs in bit 3 (bit 3 of the higher-order byte during a 16-bit computation) as the result of an addition or subtraction and cleared (to 0) otherwise.

There are some instructions that do not affect this flag at all.

PSWB5, PSWB4 (bits 5 and 4): User bits

These bits can be read and written through instructions. They can be used by the user freely.

LDCBNK (bit 3): Bank flag for the table lookup instruction (LDCW)

This bit designates the ROM bank to be specified when reading the program ROM with a table lookup instruction.

(0: ROM-ADR = 0 to 1FFFF, 1: ROM-ADR = 20000 to 3FFFF)

OV (bit 2): Overflow flag

OV is set (to 1) when an overflow occurs as the result of an arithmetic operation and cleared (to 0) otherwise. An overflow occurs in the following cases:

1)   When MSB is used as the sign bit and when the result of negative number + negative number or negative number – positive number is a positive

2)   When MSB is used as the sign bit and when the result of positive number + positive number or positive number – negative number is a negative number

3)   When the higher-order 8 bits of a 16 bits × 8 bits multiplication is nonzero

4)   When the higher-order 16 bits of a 24 bits × 16 bits multiplication is nonzero

5)   When the divisor of a division is 0

There are some instructions that do not affect this flag at all.

P1 (bit 1): RAM bit 8 data flag

P1 is used to manipulate bit 8 of 9-bit internal data RAM (0000H to FDFFH). Its behavior varies depending on the instruction executed. See Table 2-4-2 for details.

PARITY (bit 0): Parity flag

This bit shows the parity of the accumulator (A register). The parity flag is set (to 1) when there are an odd number of 1s in the A register. It is cleared (to 0) when there are an even number of 1s in the A register.

2.9   Stack Pointer (SP)

The LC870000 series microcontrollers can use RAM addresses 0000H to FDFFH as a stack area. The size of RAM, however, varies depending on the model of the microcontroller. The SP is 16 bits long and made up of two registers: SPL (at address FE0A) and SPH (at address FE0B). It is initialized to 0000H on a reset.

The SP is incremented by 1 before data is saved in stack memory and decremented by 1 after the data is restored from stack memory.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE0A	0000 0000	R/W	SPL	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
FE0B	0000 0000	R/W	SPH	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8

The value of the SP changes as follows:

1)   When the PUSH instruction is executed: SP = SP + 1, RAM (SP) = DATA

2)   When the CALL instruction is executed: SP = SP + 1, RAM (SP) = ROMBANK + ADL

SP = SP + 1, RAM (SP) = ADH

3)   When the POP instruction is executed: DATA = RAM (SP), SP = SP - 1

4)   When the RET instruction is executed: ADH = RAM (SP), SP = SP - 1

ROM BANK + ADL = RAM(SP), SP = SP - 1

2.10 Indirect Addressing Registers

The LC870000 series microcontrollers are provided with three addressing schemes ([Rn], [Rn + C], [off]) that use the contents of indirect registers (indirect addressing modes). (See Section 2.11 for the addressing modes.) Used for these addressing modes are 64 2-byte indirect registers (R0 to R63) allocated to RAM addresses 0 to 7EH. The indirect registers can also be used as general-purpose registers (e.g., for saving 2-byte data). Naturally, these addresses can be used as ordinary RAM (on a 1 byte (9 bits) basis) if they are not used as indirect registers. R0 to R63 are "system reserved words" to the assembler and need not be defined by the user.
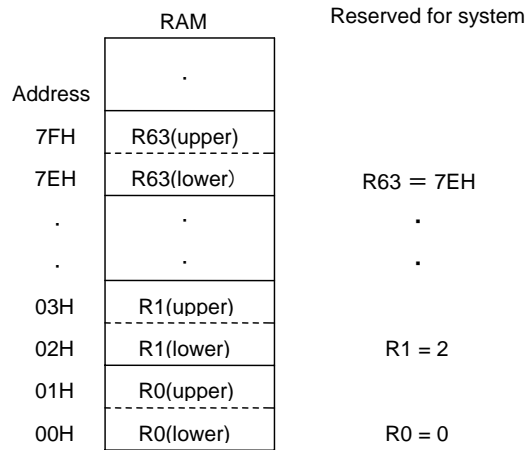
Fig. 2.10.1 Allocation of Indirect Registers

2.11 Addressing Modes

The LC870000 series microcontrollers support the following seven addressing modes:

1) Immediate (immediate data refers to data whose value has been established at program preparation (assembly) time.)
2) Indirect register (Rn) indirect ($0 \leqq n \leqq 63$)
3) Indirect register (Rn) + C register indirect ($0 \leqq n \leqq 63$)
4) Indirect register (R0) + Offset value indirect
5) Direct
6) ROM table look-up
7) External data memory access

The rest of this section describes these addressing modes.

2.11.1 Immediate Addressing (#)

The immediate addressing mode allows 8-bite (1-byte) or 16-bit (1-word) immediate data to be handled. Examples are given below.

Examples:

	LD	#12H;	Loads the accumulator with byte data (12H).
L1:	LDW	#1234H;	Loads the BA register pair with word data (1234H).
	PUSH	#34H;	Loads the stack with byte data (34H).
	ADD	#56H;	Adds byte data (56H) to the accumulator.
	BE	#78H, L1;	Compares byte data (78H) with the accumulator for a branch.

2.11.2   Indirect Register Indirect Addressing ([Rn])

In the indirect register indirect addressing mode, it is possible to select one of the indirect registers (R0 to R63) and use its contents to designate an address in RAM or SFR. When the selected register contains, for example, "FE02H," it designates the C register.

Example: When R3 contains "123H" (RAM address 6: 23H, RAM address 7: 01H)

	LD	[R3];	Transfers the contents of RAM address 123H to the accumulator.
L1:	STW	[R3];	Transfers the contents of BA register pair to RAM address 123H.
	PUSH	[R3];	Saves the contents of RAM address123H in the stack.
	SUB	[R3];	Subtracts the contents of RAM address 123H from the accumulator.
	DBZ	[R3], L1;	Decrements the contents of RAM address 123H by 1 and causes a branch if zero.

2.11.3   Indirect Register + C Register Indirect Addressing ([Rn, C])

In the indirect register + C register indirect addressing mode, the result of adding the contents of one of the indirect registers (R0 to R63) to the contents of the C register (-128 to +127 with MSB being the sign bit) designates an address in RAM or SFR. For example, if the selected indirect register contains "FE02H" and the C register contains "FFH (-1)," the address "B register (FE02H + (-1) = FE01H" is designated.

Examples: When R3 contains "123H" and the C register contains "02H"

	LD	[R3, C];	Transfers the contents of RAM address 125H to the accumulator.
L1:	STW	[R3, C];	Transfers the contents of the BA register pair to RAM address 125H.
	PUSH	[R3, C];	Saves the contents of 125H in the stack.
	SUB	[R3, C];	Subtracts the contents of RAM address 125H from the accumulator.
	DBZ	[R3, C], L1;	Decrements the contents of RAM address 125H by 1 and causes a branch if zero.

<Notes on this addressing mode >

The internal data memory space is divided into three closed functional areas as explained in Section 2.1, namely, 1) system reserved area (FF00 to FFFF), 2) SFR area (FE00 to FEFF), and 3) RAM/stack area (0000 to FDFF). Consequently, it is disallowed to point to a different area using the value of the C register from the basic area designated by the contents of Rn. For example, if the instruction "LD [R5,C]" is executed when R5 contains "0FDFFH" and the C register contains "1," since the basic area is 3) RAM/stack area (0000 to FDFF), the intended address "0FDFFH+1 = 0FE00H" lies outside the basic area and "0FFH" is consequently placed in the ACC. If the instruction "LD [R5,C]" is executed when R5 contains "0FEFFH" and the C register contains "2," since the basic area is 2) SFR area (FE00 to FEFF), the intended address "0FEFFH+2 = 0FF01H" lies outside the basic area. In this case, since SFR is confined in an 8-bit address space, the part of the address data addressing outside the 8-bit address space is ignored and the contents of 0FE01H (B register) are placed in the ACC as the result of the computation "0FF01H&0FFH+0FE00H = 0FE01."

2.11.4   Indirect Register (R0) + Offset Value indirect Addressing ([off])

In this addressing mode, the results of adding the 7-bit signed offset data off (-64 to + 63) to the contents of the indirect register R0 designate an address in RAM or SFR. If R0 contains "FE02H" and off has a value of "7EH(-2)," for example, the A register (FE02H + (-2) = FE00H) is designated.

Examples: When R0 contains "123H" (RAM address 0: 23H, RAM address 1: 01H)

	LD	[10H];	Transfers the contents of RAM address 133H to the accumulator.
L1:	STW	[10H];	Transfers the contents of the BA register pair to RAM address 133H.
	PUSH	[10H];	Saves the contents of RAM address 133H in the stack.
	SUB	[10H];	Subtracts the contents of RAM address 133H from the accumulator.
	DBZ	[10H], L1;	Decrements the contents of RAM address 133H by 1 and causes a branch if zero.

<Notes on this addressing mode>

The internal data memory space is divided into three closed functional areas as explained in Section 2.1, namely, 1) system reserved area (FF00 to FFFF), 2) SFR area (FE00 to FEFF), and 3) RAM/stack area (0000 to FDFF). Consequently, it is disallowed to point to a different area using an offset value from the basic area designated by the contents of R0. For example, if the instruction "LD [1]" is executed when R0 contains "0FDFFH," since the basic area is 3) RAM/stack area (0000 to FDFF), the intended address "0FDFFH+1 = 0FE00H" lies outside the basic area and "0FFH" is placed in the ACC as the results of LD. If the instruction "LD [2]" is executed when R0 contains "0FEFFH," since the basic area is 2) SFR (FE00 to FEFF), the intended address "0FEFFH+2 = 0FF01H" lies outside the basic area. In this case, since SFR is confined in an 8-bit address space, the part of the address data addressing outside the 8-bit address space is ignored and the contents of "0FE01H (B register) are placed in the ACC as the result of computation "0FF01H&0FFH+0FE00H = 0FE01."

2.11.5   Direct Addressing (dst)

The direct addressing mode allows a RAM or SFR address to be specified directly in an operand. In this addressing mode, the assembler automatically generates optimum instruction code from the address specified in the operand (the number of instruction bytes varies according to the address specified in the operand). Long (middle) range instructions (identified by an "L (M)" at the end of the mnemonic) are available to make the byte count of instructions constant (align instructions with the longest one).

Examples:

	LD	123H;	Transfers the contents of RAM address 123H to the accumulator (2-byte instruction).
	LDL	123H;	Transfers the contents of RAM address 123H to the accumulator (3-byte instruction).
L1:	STW	123H;	Transfers the contents of the BA register pair to RAM address 123H.
	PUSH	123H;	Saves the contents of RAM address 123H in the stack.
	SUB	123H;	Subtracts the contents of RAM address 123H from the accumulator.
	DBZ	123H, L1;	Decrements the contents of RAM address 123H by 1 and causes a branch if zero.

2.11.6 ROM Table Look-up Addressing

The LC870000 series microcontrollers can read 2-byte data into the BA register pair at once using the LDCW instruction. Three addressing modes [Rn], [Rn, C], and [off] are available for this purpose. (In this case only, Rn are configured as 17-bit registers (128K-byte space)).

For models with banked ROM, it is possible to reference the ROM data in the ROM bank (128K bytes) identified by the LDCBNK flag (bit 3) in the PSW. Consequently, when looking into the ROM table on a series model with banked ROM, execute the LDCW instruction after switching the bank using the SET1 or CLR1 instruction so that the LDCBNK flag designates the ROM bank where the ROM table resides.

Examples:

```
TBL: DB      34H
     DB      12H
     DW      5678H
     •       •
     •       •
     LDW     #TBL;           Loads the BA register pair with the TBL address.
     CHGP3   (TBL >> 17) & 1;   Loads LDCBNK in PSW with bit 17 of the TBL address. (Note 1)
     CHGP1   (TBL >> 16) & 1;   Loads P1 in PSW with bit 16 of the TBL address.
     STW     R0;             Load indirect register R0 with the TBL address (bits 16 to 0).
     LDCW    [1];            Reads the ROM table (B=78H, ACC=12H).
     MOV     #1, C;          Loads the C register with "01H."
     LDCW    [R0, C];        Reads the ROM table (B=78H, ACC=12H).
     INC     C;              Increments the C register by 1.
     LDCW    [R0, C]:        Reads the ROM table (B=56H, ACC=78H).
```

Note 1: LDCBNK (bit 3) of PSW need to be set up only for models with banked ROM.

2.11.7 External Data Memory Addressing

The LC870000 series microcontrollers can access external data memory spaces of up to 16M bytes (24 bits) using the LDX and STX instructions. To designate a 24-bit space, specify the contents of the B register (8 bits) as the highest-order byte of the address and the contents (16 bits) of (Rn), (Rn) + (C), or (R0) + off (either one) as the lower-order bytes of the address.

Examples:

```
     LDW     #3456H;    Sets up the lower-order 16 bits.
     STW     R5;        Loads the indirect register R5 with the lower-order 16 bits of the address.
     MOV     #12H, B;   Sets up the higher-order 8 bits of the address.
     LDX     [1];       Transfers the contents of external data memory (address 123456H) to the
                        accumulator.
```

2.12 Wait Sequence

2.12.1 Wait Sequence Occurrence

This series of microcontrollers performs wait sequences that suspend the execution of instructions in the following case:

1) When continuous data transmission is performed over the SIO0 with SIOCTR (SCON0, bit 4) set, a wait request is generated ahead of each transfer of 8-bit data, in which case a 1-cycle wait sequence (RAM data transfer) is performed.

2.12.2 What is a Wait Sequence?

1) When a wait request occurs out of a factor explained in Subsection 2.12.1, the CPU suspends the execution of the instruction for one cycle, during which transfers the required data. This is called a wait sequence.

2) The peripheral circuits such as timers and PWM continue processing during the wait sequence.

3) A wait sequence extends over no more than two cycles.

4) The microprocessor performs no wait sequence when it is in the HALT or HOLD mode.

5) Note that one cycle of discrepancy is introduced between the progresses of the program counter and time once a wait sequence occurs.

Table 2.4.2   Chart of State Transitions of Bit 8 (RAM / SFR) and P1

Instruction	BIT8 (RAM/SFR)	P1 (PSW BIT 1)	Remarks
LD#/LDW#	—	—	
LD	—	P1←REG8	
LDW	—	P1←REGH8	
ST	REG8←P1	—	
STW	REGL8, REGH8←P1	—	
MOV	REG8←P1	—	
PUSH#	RAM8←P1	—	
PUSH	RAM8←REG8	P1←REG8	
PUSHW	RAMH8←REGH8, RAML8←REGL8	P1←REGH8	
PUSH_P	RAM8←Pl	—	
PUSH_BA	RAMH8←P1, RAML8←P1	—	
POP	REG8←RAM8	P1←RAM8	P1←bit1 when PSW is popped
POPW	REGH8←RAMH8, REGL8←RAML8	Pl←RAMH8	P1←bit1 when higher-order address of PSW is popped
POP_P	—	P1←RAMl (bit l)	BIT8 ignored
POP_BA	—	P1←RAMH8	
XCH	REG8C↔P1	Same as left.	
XCHW	REGH8←P1, REGL8←Pl, P1←REGH8	Same as left.	
INC	INC 9 bits	P1←REG8 after computation	INC 9 bits
INCW	INC 17 bits, REGL8←lower byte of CY	P1←REGH8 after computation	INC 17 bits
DEC	DEC 9 bits	P1←REG8 after computation	DEC 9 bits
DECW	DEC 17 bits, REGL8← lower byte of CY inverted	P1←REGH8 after computation	DEC 17 bits
DBNZ	DEC 9 bits	P1←REG8	DEC 9 bits, check lower-order 8 bits
DBZ	DEC 9 bits	P1←REG8	DEC 9 bits, check lower-order 8 bits
SET1	—	—	
NOT1	—	—	
CLR1	—	—	
BPC	—	—	
BP	—	—	
BN	—	—	
MUL24 /DIV24	RAM8←"1"	—	Bit 8 of RAM address for storing results is set to 1.
FUNC	—	—	

Note: *A "1" is read if the processing target is an 8-bit register (no bit 8).*

Legends:

REG8:	*Bit 8 of a RAM or SFR location*
REGH8/REGL8:	*Bit 8 of the higher-order byte of a RAM location or SFR/bit 8 of the lower-order byte*
RAM8:	*Bit 8 of a RAM location*
RAMH8/RAML8:	*Bit 8 of the higher-order byte of a RAM location/bit 8 of the lower-order byte*

3. Peripheral System Configuration

This chapter describes the Internal functional blocks (peripheral system) of the LC872H00 series microcontrollers except the CPU core, RAM, and ROM. Port block diagrams are provided in Appendix A-II for reference.

3.1 Port 0

3.1.1 Overview

Port 0 is an 8-bit I/O port equipped with programmable pull-up resistors. It is made up of a data latch, a data direction register, and a control circuit. Control of the input/output signal direction and the pull-up resistors is accomplished through the data direction register in 4-bit units.

This port can also serve as a pin for external interrupts and can reset the HOLD mode. As a user option, either CMOS output with a programmable pull-up resistor or N-channel open drain output can be selected as the output type on a bit basis.

<Notes on the flash ROM version>

Port P05 is temporarily set low when the microcontroller is reset. During the reset sequence, do not apply a clock or any medium voltage level signal (including Hi-Z) to port P07.

For the treatment of the onchip debugger pins, refer to the separately available documents entitled "RD87 Onchip Debugger Installation Manual" and "LC872000 Series Onchip Debugger Pin Connection Requirements."

3.1.2 Functions

1) Input/output port (8 bits: P00-P07)
   - The port output data is controlled by port 0 data latch (P0: FE40) on a bit basis.
   - I/O control of P00 to P03 is accomplished by P0LDDR (P0DDR: FE41, bit 0).
   - I/O control of P04 to P07 is accomplished by P0HDDR (P0DDR: FE41, bit 1).
   - Port bits selected as CMOS outputs as user options are provided with programmable pull-up resistors.
   - The programmable pull-up resistors may be of either low impedance or high impedance type (user selectable).
   - The programmable pull-up resistors for P00 to P03 are controlled by the P0LPU (P0DDR: FE41, bit 2). Their type (either low impedance or high impedance) is selected by P0LPUS (P0DDR: FE41, bit 6).
   - The programmable pull-up resistors for P04 to P07 are controlled by P0HPU (P0DDR: FE41, bit 3). Their type (either low impedance or high impedance) is selected by P0HPUS (P0DDR: FE41, bit 7).

2) Interrupt pin function

P0FLG (P0DDR: FE41, bit 5) is set when an input port is specified and 0 level data is input to one of port bits whose corresponding bit in the port 0 data latch (P0: FE40) is set to 1.

In this case, if P0IE (P0DDR: FE41, bit 4) is 1, the HOLD mode is reset and an interrupt request to vector address 004BH is generated.

3) Shared pin function

Pin P05 also serves as the system clock output pin, pin P06 as the timer 6 toggle output, pin P07 as the timer 7 toggle output, and P00 to P06 as the analog input channel pins AN0 to AN6.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE40	0000 0000	R/W	P0	P07	P06	P05	P04	P03	P02	P01	P00
FE41	0000 0000	R/W	P0DDR	P0HPUS	P0LPUS	P0FLG	P0IE	P0HPU	P0LPU	P0HDDR	P0LDDR
FE42	00HH 0000	R/W	P0FCR	T7OE	T6OE	-	-	CLKOEN	CKODV2	CKODV 1	CKODV0

3.1.3   Related Registers

3.1.3.1   Port 0 data latch (P0)
1) The port 0 data latch is an 8-bit register for controlling port 0 output data and port 0 interrupts.
2) When this register is read with an instruction, data at pins P00 to P07 is read in. If P0 (FE40) is manipulated with an instruction NOT1, CLR1, SET1, DBZ, DBNZ, INC, or DEC, the contents of the register are referenced instead of the data at port pins.
3) Port 0 data can always be read regardless of the I/O state of the port.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE40	0000 0000	R/W	P0	P07	P06	P05	P04	P03	P02	P01	P00

3.1.3.2   Port 0 data direction register (P0DDR)
1) The port 0 data direction register is a 8-bit register that controls the I/O direction of port 0 data in 4 bit units, the pull-up resistors in 4 bit units, and port 0 interrupts.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE41	0000 0000	R/W	P0DDR	P0HPUS	P0LPUS	P0FLG	P0IE	P0HPU	P0LPU	P0HDDR	P0LDDR

P0HPUS (bit 7): P0-P04 high/low impedance pull-up resistor select

A 1 in this bit selects high impedance pull-up resistors for pins P07 to P04 and a 0 selects low impedance pull-up resistors.

P0LPUS (bit 6): P03-P00 high/low impedance pull-up resistor select

A 1 in this bit selects high impedance pull-up resistors for pins P03 to P00 and a 0 selects low impedance pull-up resistors.

P0FLG (bit 5): P0 interrupt source flag

This flag is set when a low level is applied to a port 0 pin that is set up for input and the corresponding P0 (FE40) bit is set.

A HOLD mode reset signal and an interrupt request to vector address 004BH are generated when both this bit and the interrupt request enable bit (P0IE) are set to 1.

This bit must be cleared with an instruction as it is not cleared automatically.

P0IE (bit 4): P0 interrupt request enable

Setting this bit and P0FLG to 1 generates a HOLD mode reset signal and an interrupt request to vector address 004BH

P0HPU (bit 3): P07-P04 pull-up resistor control

> When this bit is set to 1 and P0HDDR to 0, pull-up resistors are connected to port bits P07to P04 that are selected as CMOS output.

P0LPU (bit 2): P03-P00 pull-up resistor control

> When this bit is set to 1 and P0LDDR to 0, pull-up resistors are connected to port bits P03 to P00 that are selected as CMOS output.

P0HDDR (bit 1): P07-P04 I/O control

> A 1 in this bit places P07 to P04 into the output mode in which case the contents of the corresponding port 0 data latch (P0) are output.

> When this bit is set to 0, P07 to P04 are placed into the input mode and P0FLG is set when a low level is detected at a port whose corresponding port 0 data latch (P0) bit is set to 1.

P0LDDR (bit 0): P03 - P00 I/O control

> A 1 in this bit places P03 to P00 into the output mode in which case the contents of the corresponding port 0 data latch (P0) are output.

> When this bit is set to 0, P03 to P00 are placed into the input mode and P0FLG is set when a low level is detected at a port whose corresponding port 0 data latch (P0) bit is set to 1.

P07-P04 pull-up resistor selection settings

P0HPUS	P0HPU	Port for which P0HDDR =0 and CMOS option is specified
X	0	Pull-up resistor OFF
X	0	Pull-up resistor OFF
0	1	Low impedance pull-up resistor ON
1	1	High impedance pull-up resistor ON

P03-P00 pull-up resistor selection settings

P0LPUS	P0LPU	Port for which P0LDDR=0 and CMOS option is specified
X	0	Pull-up resistor OFF
X	0	Pull-up resistor OFF
0	1	Low impedance pull-up resistor ON
1	1	High impedance pull-up resistor ON

3.1.3.3   Port 0 Function Control Register (P0FCR)

1)   This 6-bit register controls Port 0's shared output pins.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE42	00HH 0000	R/W	P0FCR	T7OE	T6OE	-	-	CLKOEN	CLKODV2	CLKODV1	CLKODV0

T7OE (bit 7):

> Controls the output data of pin P07. This bit is disabled when P07 is in the input mode.

> When P07 is in the output mode:

> 0: Carries the value of the port data latch.

> 1: Carries the OR of the waveform that toggles at the interval determined by timer 7 and the value of the port data latch.

T6OE (bit 6):

> Controls the output data of pin P06. This bit is disabled when P06 is in the input mode.

> When P06 is in the output mode:

> 0: Carries the value of the port data latch.

> 1: Carries the OR of the waveform that toggles at the interval determined by timer 6 and the value of the port data latch.

CLKOEN (bit 3):

Controls the output data of pin P05. This bit is disabled when P05 is in the input mode.

When P05 is in the output mode:

0: Carries the value of the port data latch.

1: Carries the OR of the system clock output and the value of the port data latch.

CLKODV2 (bit 2):

CLKODV1 (bit 1):

CLKODV0 (bit 0):

Define the frequency of the system clock to be placed at P05.

000: Frequency of source oscillator selected as system clock

001: 1/2 of frequency of source oscillator selected as system clock

010: 1/4 of frequency of source oscillator selected as system clock

011: 1/8 of frequency of source oscillator selected as system clock

100: 1/16 of frequency of source oscillator selected as system clock

101: 1/32 of frequency of source oscillator selected as system clock

110: 1/64 of frequency of source oscillator selected as system clock

111: Frequency of source oscillator selected as subclock

<Notes on the use of the clock output feature>

Take notes 1) to 3) given below when using the clock output feature. Anomalies may be observed in the waveform of the port clock output if these notes are violated.

1) Do not change the frequency of the clock output divider setting when CLKOEN (bit 3) is set to 1.
    → Do not change the settings of CLKODV2 to CLKODV0 (bits 2-0).

2) Do not change the system clock selection when CLKOEN (bit 3) is set to 1.
    → Do not change the settings of CLKB5 and CLKB4 (bits 5 and 4) of the OCR register.

3) CLKOEN will not go to 0 immediately even when the user executes an instruction that loads the P0FCR register with such data that sets the state of CLKOEN from 1 to 0. CLKOEN is set to 0 at the end of the clock that is being output (on detection of a falling edge of the clock). Accordingly, when changing the clock divider setting or changing the system clock selection after setting CLKOEN to 0 with an instruction, be sure to read the CLKOEN value in advance and make sure that it is 0.

3.1.4   Options

Two user options are available.

1) CMOS output (with a programmable pull-up resistor)
2) N-channel open drain output

3.1.5   HALT and HOLD Mode Operation

When in the HALT or HOLD mode, port 0 retains the state that is established when the HALT or HOLD mode is entered.

3.2 Port 1

3.2.1 Overview

Port 1 is an 8-bit I/O port equipped with programmable pull-up resistors. It is made up of a data latch, a data direction register, a function control register, and a control circuit. Control of the input/output signal direction is accomplished by the data direction register on a bit basis. Port 1 can also be used as a serial interface I/O port or PWM output port by manipulating its function control register.

As a user option, either CMOS output with a programmable pull-up resistor or N-channel open drain output can be selected as the output type on a bit basis.

<Notes on the flash ROM version>

Port P15 is temporarily set low when the microcontroller is reset. During the reset sequence, do not apply a clock or any medium voltage level signal (including Hi-Z) to port P13.

For the treatment of the onchip debugger pins, refer to the separately available documents entitled "RD87 Onchip Debugger Installation Manual" and "LC872000 Series Onchip Debugger Pin Connection Requirements."

3.2.2 Functions

1) I/O port (8 bits: P10 to P17)

   • The port output data is controlled by the port 1 data latch (P1: FE44) and the I/O direction is controlled by the port 1 data direction register (P1DDR: FE45).

   • Each port bit is provided with a programmable pull-up resistor.

2) Shared pin functions

   P17 is also used as the timer 1 PWMH/base timer BUZ output, P16 as the timer 1 PWML output, P15 to P13 as SIO1 I/O, and P12 to P10 as SIO0 I/O.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE44	0000 0000	R/W	P1	P17	P16	P15	P14	P13	P12	P11	P10
FE45	0000 0000	R/W	P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR
FE46	0000 0000	R/W	P1FCR	P17FCR	P16FCR	P15FCR	P14FCR	P13FCR	P12FCR	P11FCR	P10FCR
FE47	0000 H0H0	R/W	P1TST	FIX0	FIX0	MRCSFT	FIX0	-	DSNKOT	-	FIX0

Bits 7, 6, 4, and 0 of P1TST (FE47) are reserved for testing. They must always be set to 0.

Bit 2 of P1TST (FE47) is used to control the realtime output of the high-speed clock counter. It is explained in the chapter on high-speed clock counters.

Bit 5 of P1TST (FE47) is used to control the multifrequency RC oscillator. It is explained in the chapter on system clock generators.

3.2.3   Related Registers

3.2.3.1   Port 1 data latch (P1)
1) The port 1 data latch is an 8-bit register for controlling port 1 output data and pull-up resistors.
2) When this register is read with an instruction, data at pins P10 to P17 is read in. If P1 (FE44) is manipulated with an instruction NOT1, CLR1, SET1, DBZ, DBNZ, INC, or DEC, the contents of the register are referenced instead of the data at port pins.
3) Port 1 data can always be read regardless of the I/O state of the port.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE44	0000 0000	R/W	P1	P17	P16	P15	P14	P13	P12	P11	P10

3.2.3.2   Port 1 data direction register (P1DDR)
1) The port 1 data direction register is an 8-bit register that controls the I/O direction of port 1 data on a bit basis. Port P1n are placed in the output mode when bit P1nDDR is set to 1 and in the input mode when bit P1nDDR is set to 0.
2) When bit P1nDDR is set to 0, and the bit P1n of the port 1 data latch is set to 1, port P1n becomes an input with a pull-up resistor.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE45	0000 0000	R/W	P1DDR	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR

Register Data		Port P1n State		Internal Pull-up Resistor
P1n	P1nDDR	Input	Output	
0	0	Enabled	Open	OFF
1	0	Enabled	Internal pull-up resistor	ON
0	1	Enabled	Low	OFF
1	1	Enabled	High/open (CMOS/N-channel open drain)	OFF

3.2.3.3   Port 1 function control register (P1FCR)

1)   The port 1 function control register is an 8-bit register that controls the shared functions of port 1.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE46	0000 0000	R/W	P1FCR	P17FCR	P16FCR	P15FCR	P14FCR	P13FCR	P12FCR	P11FCR	P10FCR

n	P1nFCR	P1n	P1n Pin Data in Output Mode (P1nDDR = 1)
7	0	–	Value of port data latch (P17)
	1	0	Timer 1PWMH or BUZ data for base timer
	1	1	Timer 1PWMH or inverted BUZ data for base timer
6	0	–	Value of port data latch (P16)
	1	0	Timer 1 PWML data
	1	1	Inverted data of timer 1 PWML data
5	0	–	Value of port data latch (P15)
	1	0	SIO1 clock output data
	1	1	High output
4	0	–	Value of port data latch (P14)
	1	0	SIO1 output data
	1	1	High output
3	0	–	Value of port data latch (P13)
	1	0	SIO1 output data
	1	1	High output
2	0	–	Value of port data latch (P12)
	1	0	SIO0 clock output data
	1	1	High output
1	0	–	Value of port data latch (P11)
	1	0	SIO0 output data
	1	1	High output
0	0	–	Value of port data latch (P10)
	1	0	SIO0 output data
	1	1	High output

The high data output at a pin that is selected as an N-channel open drain output (user option) is represented by an open circuit.

P17FCR (bit 7): P17 function control (timer 1 PWMH or base timer BUZ output control)

This bit controls the output data at pin P17.

When P17 is placed in the output mode (P17DDR = 1) and P17FCR is set to 1, Timer 1 PWMH output or the EOR of the port data latch and the BUZ output data from the base timer is placed at pin 17.

*: Switching between the BUZ outputs from timer 1 PWMH and base timer is controlled by BUZSEL (ISL: FE5F, bit 3).

P16FCR (bit 6): P16 function control (timer 1 PWML output control)

This bit controls the output data at pin P16.

When P16 is placed in the output mode (P16DDR = 1) and P16FCR is set to 1, the EOR of timer 1 PWML output data and the port data latch is placed at pin 16.

P15FCR (bit 5): P15 function control (SIO1 clock output control)

This bit controls the output data at pin P15.

When P15 is placed in the output mode (P15DDR = 1) and P15FCR is set to 1, the OR of the SIO1 clock output data and the port data latch is placed at pin 15.

P14FCR (bit 4): P14 function control (SIO1 data output control)

This bit controls the output data at pin P14.

When P14 is placed in the output mode (P14DDR = 1) and P14FCR is set to 1, the OR of the SIO1 output data and the port data latch is placed at pin P14.

When the SIO1 is active, SIO1 input data is read from P14 regardless of the I/O state of P14.

P13FCR (bit 3): P13 function control (SIO1 data output control)

This bit controls the output data at pin P13.

When P13 is placed in the output mode (P13DDR = 1) and P13FCR is set to 1, the OR of the SIO1 output data and the port data latch is placed at pin P13.

P12FCR (bit 2): P12 function control (SIO0 clock output control)

This bit controls the output data at pin P12.

When P12 is placed in the output mode (P12DDR = 1) and P12FCR is set to 1, the OR of the SIO0 clock output data and the port data latch is placed at pin P12.

P11FCR (bit 1): P11 function control (SIO0 data output control)

This bit controls the output data at pin P11.

When P11 is placed in the output mode (P11DDR = 1) and P11FCR is set to 1, the OR of the SIO0 output data and the port data latch is placed at pin P11.

When the SIO0 is active, SIO0 input data is read from P11 regardless of the I/O state of P11.

P10FCR (bit 0): P10 function control (SIO0 data output control)

This bit controls the output data at pin P10.

When P10 is placed in the output mode (P10DDR = 1) and P10FCR is set to 1, the OR of the SIO0 output data and the port data latch is placed at pin P10.

3.2.4   Options

Two user options are available.

1)   CMOS output   (with a programmable pull-up resistor)
2)   N-channel open drain output   (with a programmable pull-up resistor)

3.2.5   HALT and HOLD Mode Operation

When in the HALT or HOLD mode, port 1 retains the state that is established when the HALT or HOLD mode is entered.

3.3   Port 2

3.3.1   Overview

Port 2 is an 8-bit I/O port equipped with programmable pull-up resistors. It is made up of a data latch, a data direction register, and a control circuit. Control of the input/output signal direction is accomplished by the data direction register on a bit basis.

Port 2 can also serve as an input port for external interrupts. It can also be used as an input port for the timer 1 count clock input, timer 0 capture signal input, and HOLD mode reset signal input.

As a user option, either CMOS output with a programmable pull-up resistor or N-channel open drain output with a programmable pull-up resistor can be selected as the output type on a bit basis.

3.3.2   Functions

1)   Input/output port (2 bits: P20 and P21)

   • The port 2 data latch (P2: FE48) is used to control port output data and the port 2 data direction register (P2DDR: FE49) to control the I/O direction of port data.

   • Each port bit is provided with a programmable pull-up resistor.

2)   Interrupt input pin function

   The port (INT4) selected out of P20 and P21 is provided with a pin interrupt function. This function senses a low edge, a high edge, or both edges and sets the interrupt flag. This port can also serve as timer 1 counter clock input or timer 0 capture signal input.

3)   Hold mode reset function

   • When the interrupt flag and interrupt enable flag are set by INT4, a HOLD mode reset signal is generated, resetting the HOLD mode. The CPU then enters the HALT mode (main oscillation by CR). When the interrupt is accepted, the CPU switches from the HALT mode to normal operating mode.

   • When a signal change that will set the INT4 interrupt flag is input in the HOLD mode, that interrupt flag is set. In this case, the HOLD mode is reset if the corresponding interrupt enable flag is set.

   The interrupt flag, however, cannot be set by a rising edge occurring when INT4 data that is established when the HOLD mode is entered is in the high state or by a falling edge occurring when INT4 data that is established when the HOLD mode is entered is in the low state. Consequently, to reset the HOLD mode with INT4, it is recommended that INT4 be used in the double edge interrupt mode.

4)   Shared pin function

   • Pins P20 and P21 of port 2 are also used by the UART1 input/output function. These pins are explained in the pertinent chapters.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE48	HHHH HH00	R/W	P2	-	-	-	-	-	-	P21	P20
FE49	HHHH HH00	R/W	P2DDR	-	-	-	-	-	-	P21DDR	P20DDR
FE4A	0000 0000	R/W	I45CR	INT5HEG	INT5LEG	INT5IF	INT5IE	INT4HEG	INT4LEG	INT4IF	INT4IE
FE4B	0000 0000	R/W	I45SL	I5SL3	I5SL2	I5SL1	I5SL0	I4SL3	I4SL2	I4SL1	I4SL0

3.3.3 Related Registers

3.3.3.1 Port 2 data latch (P2)

1) The port 2 data latch is a 2-bit register for controlling port 2 output data and pull-up resistors.

2) When this register is read with an instruction, data at pins P20 and P21 is read in. If P2 (FE48) is manipulated with an instruction NOT1, CLR1, SET1, DBZ, DBNZ, INC, or DEC, the contents of the register are referenced instead of the data at port pins.

3) Port 2 data can always be read regardless of the I/O state of the port.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE48	HHHH HH00	R/W	P2	-	-	-	-	-	-	P21	P20

3.3.3.2 Port 2 data direction register (P2DDR)

1) The port 2 data direction register is a 2-bit register that controls the I/O direction of port 2 data on a bit basis. Port P2n are placed in the output mode when bit P2nDDR is set to 1 and in the input mode when bit P2nDDR is set to 0.

2) When bit P2nDDR is set to 0 and the bit P2n of the port 2 data latch is set to 1, port P2n becomes an input with a pull-up resistor

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE49	HHHH HH00	R/W	P2DDR	-	-	-	-	-	-	P21DDR	P20DDR

Register Data		Port P2n State		Internal Pull-up Resistor
P2n	P2nDDR	Input	Output	
0	0	Enabled	Open	OFF
1	0	Enabled	Internal pull-up resistor	ON
0	1	Enabled	Low	OFF
1	1	Enabled	High/open (CMOS/N-channel open drain)	OFF

3.3.3.3 External interrupt 4/5 control register (I45CR)

1) This register is an 8-bit register for controlling external interrupts 4 and 5.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE4A	0000 0000	R/W	I45CR	INT5HEG	INT5LEG	INT5IF	INT5IE	INT4HEG	INT4LEG	INT4IF	INT4IE

INT5HEG (bit 7): Controls the detection of an INT5 rising edge.

INT5LEG (bit 6): Controls the detection of an INT5 falling edge.

INT5HEG	INT5LEG	INT5 Interrupt Conditions (Pin Data)
0	0	No edge detection
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Both edges detection

INT5IF (bit 5): INT5 interrupt source flag

This bit is set when the conditions specified by INT5HEG and INT5LEG are satisfied.

When this bit and the INT5 interrupt request enable bit (INT5IE) are set to1, a HOLD mode reset signal and an interrupt request to vector address 001BH are generated.

The interrupt flag, however, cannot be set by a rising edge occurring when INT5 data which is established when the HOLD mode is entered is in the high state or by a falling edge occurring when INT5 data which is established when the HOLD mode is entered is in the low state. Consequently, to reset the HOLD mode with INT5, it is recommended that INT5 be used in the double edge interrupt mode.

This bit must be cleared with an instruction as it is not cleared automatically.

INT5IE (bit 4): INT5 interrupt request enable

When this bit and INT5IF are set to 1, a HOLD mode reset signal and an interrupt request to vector address 001BH are generated.

INT4HEG (bit 3): INT4 rising edge detection control

INT4LEG (bit 2): INT4 falling edge detection control

INT4HEG	INT4LEG	INT4 Interrupt Conditions (Pin Data)
0	0	No edge detection
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Both edges detection

INT4IF (bit 1): INT4 interrupt source flag

This bit is set when the conditions specified by INT4HEG and INT4LEG are satisfied.

When this bit and the INT4 interrupt request enable bit (INT4IE) are set to 1, a HOLD mode reset signal and an interrupt request to vector address 0013H are generated.

The interrupt flag, however, cannot be set by a rising edge occurring when INT4 data which is established when the HOLD mode is entered is in the high state or by a falling edge occurring when INT4 data which is established when the HOLD mode is entered is in the low state. Consequently, to reset the HOLD mode with INT4, it is recommended that INT4 be used in the double edge interrupt mode.

This bit must be cleared with an instruction as it is not cleared automatically.

INT4IE (bit 0): INT4 interrupt request enable

When this bit and INT4IF are set to 1, a HOLD mode reset signal and an interrupt request to vector address 0013H are generated.

3.3.3.4   External interrupt 4/5 pin select register (I45SL)

1) This register is an 8-bit register used to select pins for the external interrupts 4 and 5.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE4B	0000 0000	R/W	I45SL	I5SL3	I5SL2	I5SL1	I5SL0	I4SL3	I4SL2	I4SL1	I4SL0

Ports

I5SL3 (bit 7): INT5 pin select

I5SL2 (bit 6): INT5 pin select

I5SL3	I5SL2	Pin Assigned to INT5
0	0	Port pin P30
0	1	Port pin P31
1	0	Inhibited
1	1	Inhibited

I5SL1 (bit 5): INT5 pin function select

I5SL0 (bit 4): INT5 pin function select

When the data change specified in the external interrupt 4/5 control register (I45CR) is given to the pin that is assigned to INT5, timer 1 count clock input and timer 0 capture signal are generated.

I5SL1	I5SL0	Function Other Than INT5 Interrupt
0	0	None
0	1	Timer 1 count clock input
1	0	Timer 0L capture signal input
1	1	Timer 0H capture signal input

I4SL3 (bit 3): INT4 pin select

I4SL2 (bit 2): INT4 pin select

I4SL3	I4SL2	Pin Assigned to INT4
0	0	Port pin P20
0	1	Port pin P21
1	0	Inhibited
1	1	Inhibited

I4SL1 (bit 1): INT4 pin function select

I4SL0 (bit 0): INT4 pin function select

When the data change specified in the external interrupt 4/5 control register (I45CR) is given to the pin that is assigned to INT4, timer 1 count clock input and timer 0 capture signal are generated.

I4SL1	I4SL0	Function other than INT4 Interrupt
0	0	None
0	1	Timer 1 count clock input
1	0	Timer 0L capture signal input
1	1	Timer 0H capture signal input

Notes:

1) *When timer 0L capture signal input or timer 0H capture signal input is specified for INT4 or INT5 together with in port 7, the signal from port 7 is ignored.*

2) *When INT4 and INT5 are specified in duplicate for timer 1 count clock input, timer 0L capture signal input, or timer 0H capture signal input, both interrupts are accepted. If both INT4 and INT5 events occur at the same time, however, only one event is recognized.*

3) *When at least one of INT4 and INT5 is specified as timer 1 count clock input, timer 1L functions as an event counter. If neither INT4 nor INT5 are specified for timer 1 count clock input, the timer 1L counter counts on every 2Tcyc.*

3.3.4   Options

Two user options are available.

1)   CMOS output   (with a programmable pull-up resistor)
2)   N-channel open drain output   (with a programmable pull-up resistor)

3.3.5   HALT and Hold Mode Operation

When in the HALT or HOLD mode, port 2 retains the state that is established when the HALT or HOLD mode is entered.

3.4   Port 3

3.4.1   Overview

Port 3 is a 2-bit I/O port equipped with programmable pull-up resistors. It is made up of a data latch, a data direction register, and a control circuit. Control of the input/output signal direction is accomplished by the data direction register on a bit basis.

Port 3 can also serve as an input port for external interrupts. It can also be used as an input port for the timer 1 count clock input, timer 0 capture signal input, and HOLD mode reset signal input.

As a user option, either CMOS output with a programmable pull-up resistor or N-channel open drain output with a programmable pull-up resistor can be selected as the output type on a bit basis.

3.4.2   Functions

1)   Input/output port (2 bits: P30 and P31)
- The port 3 data latch (P3: FE4C) is used to control the port output data and the port 3 data direction register (P3DDR: FE4D) to control the I/O direction of port data.
- Each port bit is provided with a programmable pull-up resistor.

2)   Interrupt input pin function

The port (INT5) selected out of P30 and P31 is provided with a pin interrupt function. This function senses a low edge, a high edge, or both edges and sets the interrupt flag. This port can also serve as timer 1 counter clock input or timer 0 capture signal input.

See the subsection on Port 2 for a description of the port operation for INT5 interrupts.

3)   Hold mode reset function
- When the interrupt flag and interrupt enable flag are set by INT5, a HOLD mode reset signal is generated, resetting the HOLD mode. The CPU then enters the HALT mode (main oscillation by CR). When the interrupt is accepted, the CPU switches from the HALT mode to normal operating mode.
- When a signal change that will set the INT5 interrupt flag is input in the HOLD mode, that interrupt flag is set. In this case, the HOLD mode is reset if the corresponding interrupt enable flag is set.

  The interrupt flag, however, cannot be set by a rising edge occurring when INT5 data that is established when the HOLD mode is entered is in the high state or by a falling edge occurring when INT5 data that is established when the HOLD mode is entered is in the low state. Consequently, to reset the HOLD mode with INT5, it is recommended that INT5 be used in the double edge interrupt mode.

4)   Shared pin function
- Pins P30 and P31 of port 3 are also used by the PWM4/PWM5 output function. These pins are explained in the pertinent chapters.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE4C	HHHH HH00	R/W	P3	-	-	-	-	-	-	P31	P30
FE4D	HHHH HH00	R/W	P3DDR	-	-	-	-	-	-	P31DDR	P30DDR

3.4.3 Related Registers

3.4.3.1 Port 3 data latch (P3)

1) This data latch is a 2-bit register for controlling the port 3 output data and its pull-up resistors.

2) When this register is read with an instruction, the data at pins P30 and P31 is read in. If P3 (FE4C) is manipulated using the NOT1, CLR1, SET1, DBZ, DBNZ, INC or DEC instruction, the contents of the register is referenced instead of the data at the pins.

3) Data can always be read from port 3 regardless of its I/O state.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE4C	HHHH HH00	R/W	P3	-	-	-	-	-	-	P31	P30

3.4.3.2 Port 3 data direction register (P3DDR)

1) The port 3 data direction register is a 2-bit register for controlling the I/O direction of port 3 data on a bit basis. Port P3n is placed in the output mode when bit P3nDDR is set to 1 and in the input mode when bit P3nDDR is set to 0.

2) Port P3n is designated as an input with a pull-up resistor when bit P3nDDR is set to 0 and bit P3n of port 3 data latch is set to 1.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE4D	HHHH HH00	R/W	P3DDR	-	-	-	-	-	-	P31DDR	P30DDR

Register Data		Port P3n State		Internal Pull-up Resistor
P3n	**P3nDDR**	**Input**	**Output**	
0	0	Enabled	Open	OFF
1	0	Enabled	Internal pull-up resistor	ON
0	1	Enabled	Low	OFF
1	1	Enabled	High/open (CMOS/N-channel open drain)	OFF

3.4.4 Options

Two user options are available.

1) CMOS output   (with a programmable pull-up resistor)
2) N-channel open drain output   (with a programmable pull-up resistor)

3.4.5 HALT and HOLD Mode Operation

When in the HALT or HOLD mode, port 3 retains the state that is established when the HALT or HOLD mode is entered.

3.5 Port 7

3.5.1 Overview

Port 7 is a 4-bit I/O port equipped with programmable pull-up resistors. It is made up of a data control latch and a control circuit. The input/output direction of port data can be controlled on a bit basis.

Port 7 can be used as an input port for external interrupts. It can also be used as an input port for the timer 0 count clock input, capture signal input, and HOLD mode reset signal input.

There is no user option for this port.

3.5.2 Functions

1) Input/output port (4 bits: P70 to P73)

   • The lower-order 4 bits of the port 7 control register (P7: FE5C) are used to control the port output data and the higher-order 4 bits to control the I/O direction of port data.

   • P70 is of the N-channel open drain output type and P71 to P73 are of CMOS output type.

   • Each port bit is provided with a programmable pull-up resistor.

2) Interrupt input pin function

   • P70 and P71 are assigned to INT0 and INT1, respectively, and used to detect a low or high level, or a low or high edge and set the interrupt flag.

   • P72 and P73 are assigned to INT2 and INT3, respectively, and used to detect a low or high edge, or both edges and set the interrupt flag.

3) Timer 0 count input function

   A count signal is sent to timer 0 each time a signal change such that the interrupt flag is set is supplied to the port selected from P72 and P73.

4) Timer 0L capture input function

   A timer 0L capture signal is generated each time a signal change such that the interrupt flag is set is supplied to the port selected from P70 and P72.

   When a selected level of signal is input to P70 that is specified for level-triggered interrupts, a timer 0L capture signal is generated at 1 cycle interval.

5) Timer 0H capture input function

   A timer 0H capture signal is generated each time a signal change such that the interrupt flag is set is supplied to the port selected from P71 and P73.

   When a selected level of signal is input to P71 that is specified for level-triggered interrupts, a timer 0H capture signal is generated at 1 cycle interval.

6) HOLD mode reset function

- When the interrupt flag and interrupt enable flag are set by INT0, INT1, or INT2, a HOLD mode reset signal is generated, resetting the HOLD mode. The CPU then enters the HALT mode (main oscillation by CR). When the interrupt is accepted, the CPU switches from the HALT mode to normal operating mode.

- When a signal change such that the interrupt flag is set is input to P70 or P71 in the HOLD mode, the interrupt flag is set. In this case, the HOLD mode is reset if the corresponding interrupt enable flag is set.

- When a signal change such that the interrupt flag is set is input to P72 in the HOLD mode, the interrupt flag is set. In this case, the HOLD mode is reset if the corresponding interrupt enable flag is set. The interrupt flag, however, cannot be set by a rising edge occurring when P72 data which is established when the HOLD mode is entered, is in the high state or by a falling edge occurring when P72 data which is established when the HOLD mode is entered, is in the low state. Consequently, to reset the HOLD mode with P72, it is recommended that P72 be used in the double edge interrupt mode.

7) Shared pin function

- The pins P70 is also used as the AN8 analog input channel pin and pin P71 as the AN9 analog input channel pin.

	Input	Output	Interrupt Input Signal Detection	Timer 0 Count Input	Capture Input	Hold Mode Reset
P70	With programmable pull-up resistor	N-channel open drain	L level, H level, L edge, H edge	-	Timer 0L	Enabled (Note)
P71		CMOS		-	Timer 0H	Enabled (Note)
P72			L edge, H edge, both edges	Available	Timer 0L	Enabled
P73				Available	Timer 0H	-

Note: P70 and P71 HOLD mode reset is available only when level detection is set.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5C	0000 0000	R/W	P7	P73DDR	P72DDR	P71DDR	P70DDR	P73DT	P72DT	P71DT	P70DT
FE5D	0000 0000	R/W	I01CR	INT1LH	INT1LV	INT1IF	INT1IE	INT0LH	INT0LV	INT0IF	INT0IE
FE5E	0000 0000	R/W	I23CR	INT3HEG	INT3LEG	INT3IF	INT3IE	INT2HEG	INT2LEG	INT2IF	INT2IE
FE5F	0000 0000	R/W	ISL	ST0HCP	ST0LCP	BTIMC1	BTIMC0	BUZON	NFSEL	NFON	ST0IN

3.5.3 Related Registers

3.5.3.1 Port 7 control register (P7)

1) The port 7 control register is an 8-bit register for controlling the I/O of port 7 data and pull-up resistors.

2) When this register is read with an instruction, data at pins P70 to P73 is read into bits 0 to 3. Bits 4 to 7 are loaded with bits 4 to 7 of register P7. If P7 (FE5C) is manipulated with an instruction NOT1, CLR1, SET1, DBZ, DBNZ, INC, or DEC, the contents of the register are referenced as bits 0 to 3 instead of the data at port pins.

3) Port 7 data can always be read regardless of the I/O state of the port

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5C	0000 0000	R/W	P7	P73DDR	P72DDR	P71DDR	P70DDR	P73DT	P72DT	P71DT	P70DT

Register Data		Port P7n State		Internal Pull-up Resistor
P7n	P7nDDR	Input	Output	
0	0	Enabled	Open	OFF
1	0	Enabled	Internal pull-up resistor	ON
0	1	Enabled	CMOS-Low	OFF
1	1	Enabled	CMOS-High (P70 is open)	ON

Ports

P73DDR (bit 7): P73 I/O control

A 1 or 0 in this bit controls the output (CMOS) or input of pin P73.

P72DDR (bit 6): P72 I/O control

A 1 or 0 in this bit controls the output (CMOS) or input of pin P72.

P71DDR (bit 5): P71 I/O control

A 1 or 0 in this bit controls the output (CMOS) or input of pin P71.

P70DDR (bit 4): P70 I/O control

A 1 or 0 in this bit controls the output (N-channel open drain) or input of pin P70.

P73DT (bit 3): P73 data

The value of this bit is output from pin P73 when P73DDR is set to 1.

A 1 or 0 in this bit turns on and off the internal pull-up resistor for pin P73.

P72DT (bit 2): P72 data

The value of this bit is output from pin P72 when P72DDR is set to 1.

A 1 or 0 in this bit turns on and off the internal pull-up resistor for pin P72.

P71DT (bit 1): P71 data

The value of this bit is output from pin P71 when P71DDR is set to 1.

A 1 or 0 in this bit turns on and off the internal pull-up resistor for pin P71.

P70DT (bit 0): P70 data

The value of this bit is output from pin P70 when P70DDR is set to 1. Since this bit is of N-channel open drain output type, however, it is placed in the high-impedance state when P70DT is set to 1.

A 1 or 0 in this bit turns on and off the internal pull-up resistor for pin P70.

3.5.3.2 External interrupt 0/1 control register (I01CR)

1) This register is an 8-bit register for controlling external interrupts 0 and 1.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5D	0000 0000	R/W	I01CR	INT1LH	INT1LV	INT1IF	INT1IE	INT0LH	INT0LV	INT0IF	INT0IE

INT1LH (bit 7): INT1 detection polarity select

INT1LV (bit 6): INT1 detection level/edge select

INT1LH	INT1LV	INT1 Interrupt Conditions (P71 Pin Data)
0	0	Falling edge detection
0	1	Low level detection
1	0	Rising edge detection
1	1	High level detection

INT1IF (bit 5): INT1 interrupt source flag

This bit is set when the conditions specified by INT1LH and INT1LV are satisfied. When this bit and the INT1 interrupt request enable bit (INT1IE) are set to 1, a HOLD mode reset signal and an interrupt request to vector address 000BH are generated.

This bit must be cleared with an instruction as it is not cleared automatically.

INT1IE (bit 4): INT1 interrupt request enable

When this bit and INT1IF are set to 1, a HOLD mode reset signal and an interrupt request to vector address 000BH are generated.

INT0LH (bit 3): INT0 detection polarity select

INT0LV (bit 2): INT0 detection level/edge select

INT0LH	INT0LV	INT0 Interrupt Conditions (P70 Pin Data)
0	0	Falling edge detection
0	1	Low level detection
1	0	Rising edge detection
1	1	High level detection

INT0IF (bit 1): INT0 interrupt source flag

This bit is set when the conditions specified by INT0LH and INT0LV are satisfied. When this bit and the INT0 interrupt request enable bit (INT0IE) are set to 1, a HOLD mode reset signal and an interrupt request to vector address 0003H are generated.

This bit must be cleared with an instruction as it is not cleared automatically.

INT0IE (bit 0): INT0 interrupt request enable

When this bit and INT0IF are set to 1, a HOLD mode reset signal and an interrupt request to vector address 0003H are generated.

3.5.3.3   External interrupt 2/3 control register (I23CR)

1)   This register is an 8 bit register for controlling external interrupts 2 and 3.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5E	0000 0000	R/W	I23CR	INT3HEG	INT3LEG	INT3IF	INT3IE	INT2HEG	INT2LEG	INT2IF	INT2IE

INT3HEG (bit 7): INT3 rising edge detection control

INT3LEG (bit 6): INT3 falling edge detection control

INT3HEG	INT3LEG	INT3 Interrupt Conditions (P73 Pin Data)
0	0	No edge detection
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Both edges detection

INT3IF (bit 5): INT3 interrupt source flag

This bit is set when the conditions specified by INT3HEG and INT3LEG are satisfied. When this bit and the INT3 interrupt request enable bit (INT3IE) are set to 1, an interrupt request to vector address 001BH is generated.

This bit must be cleared with an instruction as it is not cleared automatically.

Ports

INT3IE (bit 4): INT3 interrupt request enable

When this bit and INT3IF are set to 1, an interrupt request to vector address 001BH is generated.

INT2HEG (bit 3): INT2 rising edge detection control

INT2LEG (bit 2): INT2 falling edge detection control

INT2HEG	INT2LEG	INT2 Interrupt Conditions (P72 Pin Data)
0	0	No edge detection
0	1	Falling edge detection
1	0	Rising edge detection
1	1	Both edges detection

INT2IF (bit 1): INT2 interrupt source flag

This bit is set when the conditions specified by INT2HEG and INT2LEG are satisfied.

When this bit and the INT2 interrupt request enable bit (INT2IE) are set to 1, a HOLD mode reset signal and an interrupt request to vector address 0013H are generated.

The interrupt flag, however, cannot be set by a rising edge occurring when P72 data which is established when the HOLD mode is entered is in the high state or by a falling edge occurring when P72 data which is established when the HOLD mode is entered is in the low state. Consequently, to reset the HOLD mode with P72, it is recommended that P72 be used in the double edge interrupt mode.

This bit must be cleared with an instruction as it is not cleared automatically.

INT2IE (bit 0): INT2 interrupt request enable

When this bit and INT2IF are set to 1, a HOLD mode reset signal and an interrupt request to vector address 0013H are generated.

3.5.3.4   Input signal select register (ISL)

1)   This register is an 8-bit register controlling the timer 0 input, noise filter time constant, buzzer output, and base timer clock.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5F	0000 0000	R/W	ISL	ST0HCP	ST0LCP	BTIMC1	BTIMC0	BUZON	NFSEL	NFON	ST0IN

ST0HCP (bit 7): Timer 0H capture signal input port select

This bit selects the timer 0H capture signal input port.

When set to 1, a timer 0H capture signal is generated when an input that satisfies the INT1 interrupt detection conditions is supplied to P71. If the INT1 interrupt detection mode is set to "level detection," capture signals are generated at an interval of 1 Tcyc as long as the detection level is present at P71.

When this bit is set to 0, a timer 0H capture signal is generated when an input that satisfies the INT3 interrupt detection conditions is supplied to P73.

ST0LCP (bit 6): Timer 0L capture signal input port select

This bit selects the timer 0L capture signal input port.

When set to 1, a timer 0L capture signal is generated when an input that satisfies the INT0 interrupt detection conditions is supplied to P70. If the INT0 interrupt detection mode is set to "level detection," capture signals are generated at an interval of 1 Tcyc as long as the detection level is present at P70.

When this bit is set to 0, a timer 0L capture signal is generated when an input that satisfies the INT2 interrupt detection conditions is supplied to P72.

BTIMC1 (bit 5): Base timer clock select

BTIMC0 (bit 4): Base timer clock select

BTIMC1	BTIMC0	Base Timer Input Clock
0	0	Subclock
0	1	Cycle clock
1	0	Subclock
1	1	Timer/counter 0 prescaler output

BUZON (bit 3): Buzzer output select

This bit enables the buzzer output (fBST/16).

When set to 1, a signal that is obtained by dividing the base timer clock by 16 is sent to port P17 as buzzer output.

When this bit is set to 0, the buzzer output is fixed at the high level.

NFSEL (bit 2): Noise filter time constant select

NFON (bit 1): Noise filter time constant select

NFSEL	NFON	Noise Filter Time Constant
0	0	1 Tcyc
0	1	128 Tcyc
1	0	1 Tcyc
1	1	32 Tcyc

T0IN (bit 0): Timer 0 counter clock input port select

This bit selects the timer 0 counter clock signal input port.

When set to 1, a timer 0 count clock is generated when an input that satisfies the INT3 interrupt detection conditions is supplied to P73.

When this bit is set to 0, a timer 0 count clock is generated when an input that satisfies the INT2 interrupt detection conditions is supplied to P72.

Note: When timer 0L capture signal input or timer 0H capture signal input is specified for INT4 or INT5 together with in port 7, the signal from port 7 is ignored.

3.5.4   Options

There is no user option for port 7.

3.5.5   HALT and HOLD Mode Operation

The pull-up resistor to P70 is turned off.

P71 to P73 retain their state that is established when the HALT or HOLD mode is entered.

3.6　Timer / Counter 0 (T0)

3.6.1　Overview

The timer/counter 0 (T0) incorporated in this series of microcontrollers is a 16-bit timer/counter that provides the following four functions:

1) Mode 0:　Two-channel 8-bit programmable timer with a programmable prescaler (equipped with an 8-bit capture register)

2) Mode 1:　8-bit programmable timer with a programmable prescaler (equipped with an 8-bit capture register) + 8-bit programmable counter (equipped with an 8-bit capture register)

3) Mode 2:　16-bit programmable timer with a programmable prescaler (equipped with a 16-bit capture register)

4) Mode 3:　16-bit programmable counter (equipped with a 16-bit capture register)

3.6.2　Functions

1) Mode 0:　Two-channel 8-bit programmable timer with a programmable prescaler (equipped with an 8-bit capture register)

- Two independent 8-bit programmable timers (T0L and T0H) run on the clock (with a period of 1 to 256 Tcyc) from an 8-bit programmable prescaler.

- The contents of T0L are captured into the capture register T0CAL on external input detection signals from P70/INT0/T0LCP, P72/INT2/T0IN, P20, P21, P30, and P31 timer 0L capture input pins.

- The contents of T0H are captured into the capture register T0CAH on external input detection signals from P71/INT1/T0HCP, P73/INT3/T0IN, P20, P21, P30, and P31 timer 0H capture input pins.

$$\text{T0L period} = (\text{T0LR} + 1) \times (\text{T0PRR} + 1) \times \text{Tcyc}$$
$$\text{T0H period} = (\text{T0HR} + 1) \times (\text{T0PRR} + 1) \times \text{Tcyc}$$
$$\text{Tcyc} = \text{Period of cycle clock}$$

2) Mode 1:　8-bit programmable timer with a programmable prescaler (equipped with an 8-bit capture register) + 8-bit programmable counter (equipped with an 8-bit capture register)

- T0L serves as an 8-bit programmable counter that counts the number of external input detection signals from pins P72/INT2/T0IN and P73/INT3/T0IN.

- T0H serves as an 8-bit programmable timer that runs on the clock (with a period of 1 to 256 Tcyc) from an 8-bit programmable prescaler.

- The contents of T0L are captured into the capture register T0CAL on external input detection signals from P70/INT0/T0LCP, P72/INT2/T0IN, P20, P21, P30, and P31 timer 0L capture input pins.

- The contents of T0H are captured into the capture register T0CAH on external input detection signals from P71/INT1/T0HCP, P73/INT3/T0IN, P20, P21, P30, and P31 timer 0H capture input pins.

$$\text{T0L period} = (\text{T0LR} + 1)$$
$$\text{T0H period} = (\text{T0HR} + 1) \times (\text{T0PRR} + 1) \times \text{Tcyc}$$

3) Mode 2: 16-bit programmable timer with a programmable prescaler (equipped with a 16-bit capture register)

- In this mode, timer/counter 0 serves as a 16-bit programmable timer that runs on the clock (with a period of 1 to 256 Tcyc) from an 8-bit programmable prescaler.

- The contents of T0L and T0H are captured into the capture registers T0CAL and T0CAH at the same time on external input detection signals from P71/INT1/T0HCP, P73/INT3/T0IN, P20, P21, P30, and P31 timer 0H capture input pins.

$$\text{T0 period} = ([\text{T0HR, T0HL}] + 1) \times (\text{T0PRR} + 1) \times \text{Tcyc}$$

16 bits

4) Mode 3: 16-bit programmable counter (equipped with a 16-bit capture register)

- In this mode, timer/counter 0 serves as a 16-bit programmable counter that counts the number of external input detection signals from pins P72/INT2/T0IN and P73/INT3/T0IN.

- The contents of T0L and T0H are captured into the capture registers T0CAL and T0CAH at the same time on external input detection signals from P71/INT1/T0HCP, P73/INT3/T0IN, P20, P21, P30, and P31 timer 0H capture input pins.

$$\text{T0 period} = [\text{T0HR,T0HL}] + 1$$

16 bits

5) Interrupt generation

T0L or T0H interrupt requests are generated at the counter interval for timer/counter T0L or T0H if the interrupt request enable bit is set.

6) To control timer/counter 0 (T0), it is necessary to manipulate the following special function registers.

- T0CNT, T0PRR, T0L, T0H, T0LR, T0HR,
- P7, TSL, ISL, I01CR, I23CR
- P2, P2DDR, P3, P3DDR, I45CR, I45SL

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE10	0000 0000	R/W	T0CNT	T0HRUN	T0LRUN	T0LONG	T0LEXT	T0HCMP	T0HIE	T0LCMP	T0LIE
FE11	0000 0000	R/W	T0PRR	T0PRR7	T0PRR6	T0PRR5	T0PRR4	T0PRR3	T0PRR2	T0PRR1	T0PRR0
FE12	0000 0000	R	T0L	T0L7	T0L6	T0L5	T0L4	T0L3	T0L2	T0L1	T0L0
FE13	0000 0000	R	T0H	T0H7	T0H6	T0H5	T0H4	T0H3	T0H2	T0H1	T0H0
FE14	0000 0000	R/W	T0LR	T0LR7	T0LR6	T0LR5	T0LR4	T0LR3	T0LR2	T0LR1	T0LR0
FE15	0000 0000	R/W	T0HR	T0HR7	T0HR6	T0HR5	T0HR4	T0HR3	T0HR2	T0HR1	T0HR0
FE16	XXXX XXXX	R	T0CAL	T0CAL7	T0CAL6	T0CAL5	T0CAL4	T0CAL3	T0CAL2	T0CAL1	T0CAL0
FE17	XXXX XXXX	R	T0CAH	T0CAH7	T0CAH6	T0CAH5	T0CAH4	T0CAH3	T0CAH2	T0CAH1	T0CAH0

3.6.3 Circuit Configuration

3.6.3.1 Timer/counter 0 control register (T0CNT) (8-bit register)

1) This register controls the operation and interrupts of T0L and T0H.

3.6.3.2   Programmable prescaler match register (T0PRR)   (8-bit register)

1)   This register stores the match data for the programmable prescaler.

3.6.3.3   Programmable prescaler   (8-bit counter)

1)   Start/stop: This register runs in modes other than the HOLD mode.

2)   Count clock: Cycle clock (period = 1 Tcyc).

3)   Match signal: A match signal is generated when the count value matches the value of register T0PRR (period: 1 to 256 Tcyc)

4)   Reset: The counter starts counting from 0 when a match signal occurs or when data is written into T0PRR.

3.6.3.4   Timer/counter 0 low byte (T0L)   (8-bit counter)

1)   Start/stop: This counter is started and stopped by the 0/1 value of T0LRUN (timer 0 control register, bit 6).

2)   Count clock: Either prescaler's match signal or external signal must be selected through the 0/1 value of T0LEXT (timer 0 control register, bit 4).

3)   Match signal: A match signal is generated when the count value matches the value of the match buffer register (16 bits of data need to match in the 16-bit mode).

4)   Reset: This counter is reset when it stops operation or a match signal is generated.

3.6.3.5   Timer/counter 0 high byte (T0H)   (8-bit counter)

1)   Start/stop: This counter is started and stopped by the 0/1 value of T0HRUN (timer 0 control register, bit 7).

2)   Count clock: Either prescaler's match signal or T0L match signal must be selected through the 0/1 value of T0LONG (timer 0 control register, bit 5).

3)   Match signal: A match signal is generated when the count value matches the value of the match buffer register (16 bits of data need to match in the 16-bit mode).

4)   Reset: This counter is reset when it stops operation or a match signal is generated.

3.6.3.6   Timer/counter 0 match data register low byte (T0LR)   (8-bit register with a match buffer register)

1)   This register is used to store the match data for T0L. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with the lower-order byte of timer/counter 0 (16 bits of data need to match in the 16-bit mode).

2)   The match buffer register is updated as follows:

The match register matches T0LR when it is inactive (T0LRUN = 0). When the match register is running (T0LRUN = 1), it is loaded with the contents of T0LR when a match signal is generated.

3.6.3.7   Timer/counter 0 match data register high byte (T0HR)   (8-bit register with a match buffer register)

1)   This register is used to store the match data for T0H. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with the higher-order byte of timer/counter 0 (16 bits of data need to match in the 16-bit mode).

2)   The match buffer register is updated as follows:

The match register matches T0HR when it is inactive (T0HRUN = 0).

When the match register is running (T0HRUN = 1), it is loaded with the contents of T0HR when a match signal is generated.

3.6.3.8   Timer/counter 0 capture register low byte (T0CAL)   (8-bit register)

1)   Capture clock:

External input detection signals from pins P70/INT0/T0LCP and P72/INT2/T0IN, P20, P21, P30, and P31 when T0LONG (timer 0 control register, bit 5) is set to "0."

External input detection signals from pins P71/INT1/T0HCP and P73/INT3/T0IN, P20, P21, P30, and P31 when T0LONG (timer 0 control register, bit 5) is set to "1."

2)   Capture data: Contents of timer/counter 0 low byte (T0L).

3.6.3.9   Timer/counter 0 capture register high byte (T0CAH) (8-bit register)

1)   Capture clock: External input detection signals from pins P71/INT1/T0HCP, P73/INT3/T0IN, P20, P21, P30, and P31

2)   Capture data: Contents of timer/counter 0 high byte (T0H)

Table 3.6.1 Timer 0 (T0H, T0L) Count Clocks

Mode	T0LONG	T0LEXT	T0H Count Clock	T0L Count Clock	[T0H, T0L] Count Clock
0	0	0	T0PRR match signal	T0PRR match signal	–
1	0	1	T0PRR match signal	External signal	–
2	1	0	–	–	T0PRR match signal
3	1	1	–	–	External signal
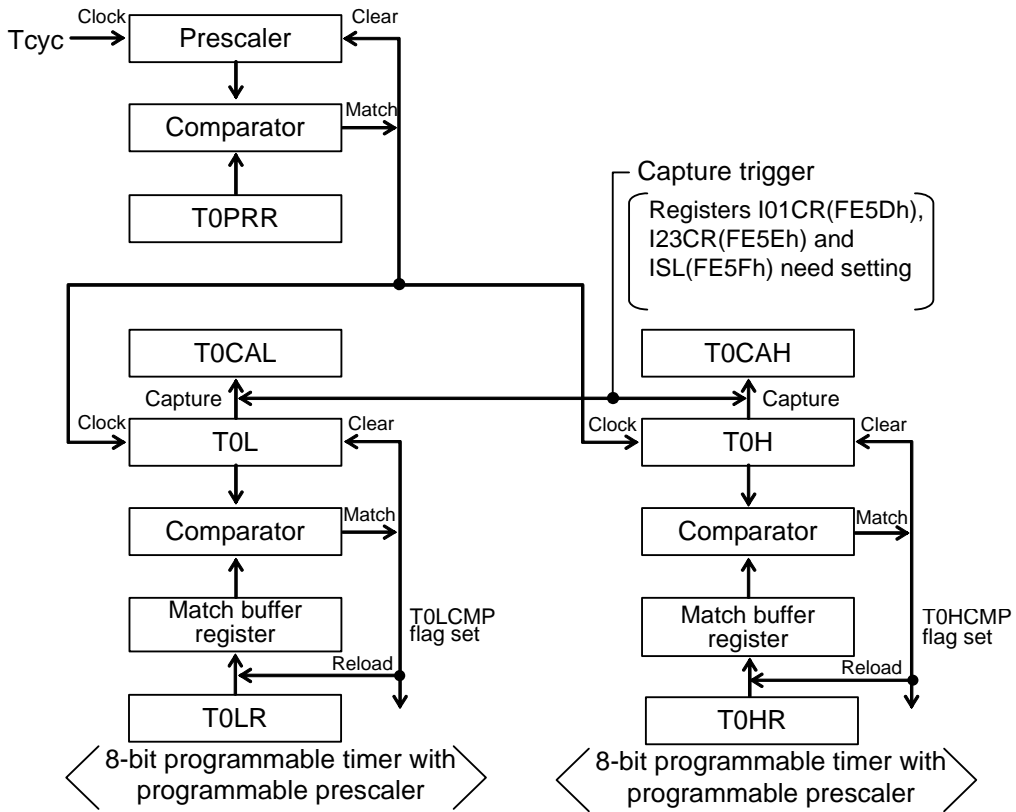
Figure 3.6.1   Mode 0 Block Diagram (T0LONG = 0 , T0LEXT = 0)
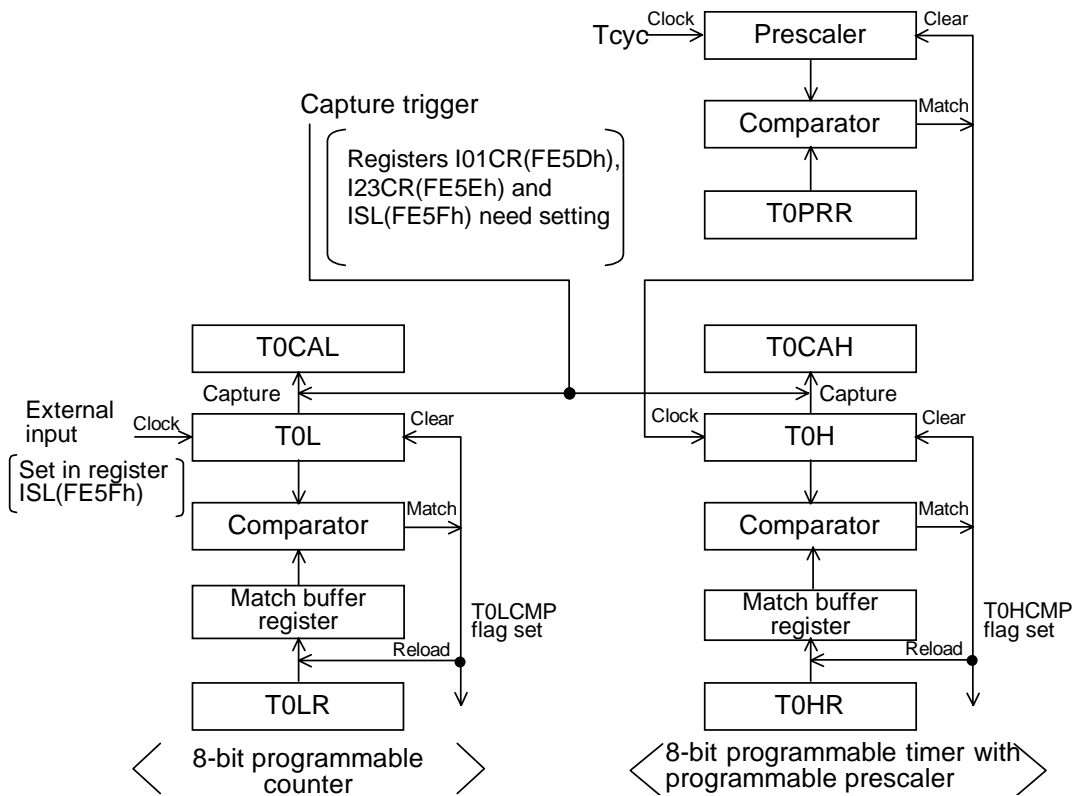
Figure 3.6.2   Mode 1 Block Diagram (T0LONG = 0, T0LEXT = 1)
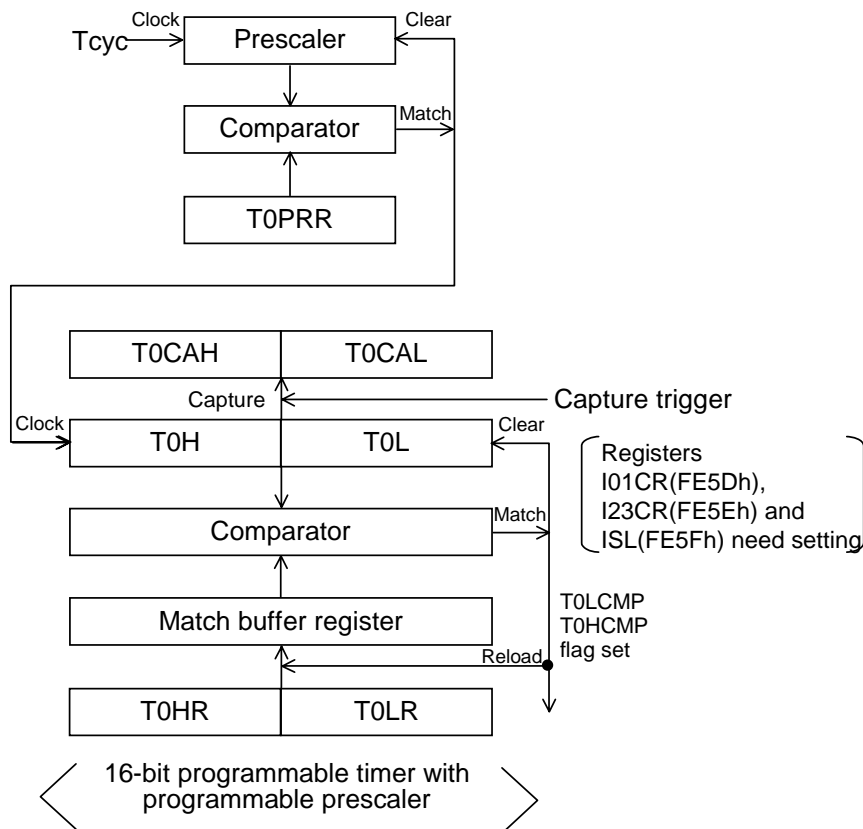
Figure 3.6.3   Mode 2 Block Diagram (T0LONG = 1, T0LEXT = 0)
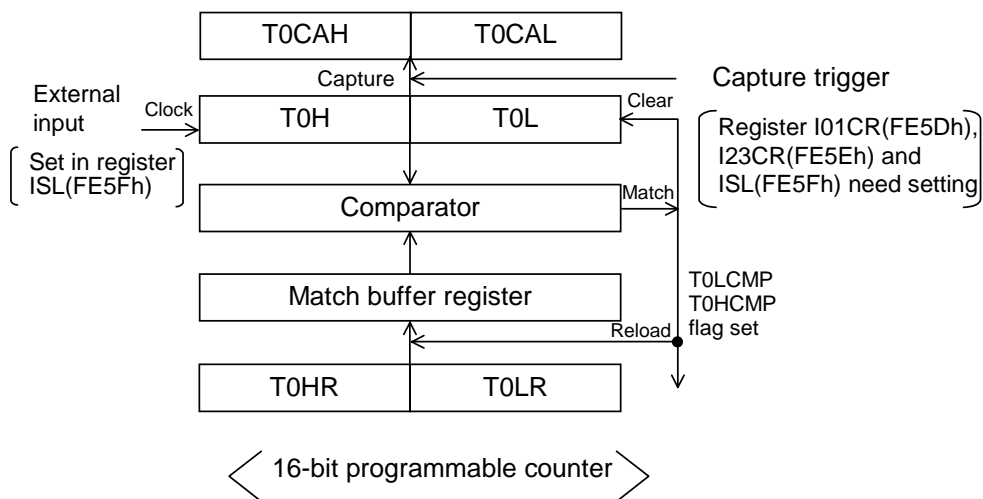
Figure 3.6.4   Mode 3 Block Diagram (T0LONG = 1, T0LEXT = 1)

3.6.4   Related Registers

3.6.4.1   Timer/counter 0 control register (T0CNT)

1)   This register is an 8-bit register that controls the operation and interrupts of T0L and T0H.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE10	0000 0000	R/W	T0CNT	T0HRUN	T0LRUN	T0LONG	T0LEXT	T0HCMP	T0HIE	T0LCMP	T0LIE

T0HRUN (bit 7): T0H count control

When this bit is set to 0, timer/counter 0 high byte (T0H) stops on a count value of 0. The match buffer register of T0H has the same value as T0HR.

When this bit is set to 1, timer/counter 0 high byte (T0H) performs the required counting operation. The match buffer register of T0H is loaded with the contents of T0HR when a match signal is generated.

T0LRUN (bit 6): T0L count control

When this bit is set to 0, timer/counter 0 low byte (T0L) stops on a count value of 0. The match buffer register of T0L has the same value as T0LR.

When this bit is set to 1, timer/counter 0 low byte (T0L) performs the required counting operation. The match buffer register of T0L is loaded with the contents of T0LR when a match signal is generated.

T0LONG (bit 5): Timer/counter 0 bit length select

When this bit is set to 0, timer/counter 0's higher- and lower-order bytes serve as independent 8-bit timers/counters.

When this bit is set to 1, timer/counter 0 functions as a 16-bit timer/counter. A match signal is generated when the count value of the 16-bit counter comprising T0H and T0L matches the contents of the match buffer register of T0H and T0L.

T0LEXT (bit 4): T0L input clock select

When this bit is set to 0, the count clock for T0L is the match signal for the prescaler.

When this bit is set to 1, the count clock for T0L is an external input signal.

T0HCMP (bit 3): T0H match flag

This bit is set when the value of T0H matches the value of the match buffer register for T0H while T0H is running (T0HRUN = 1) and a match signal is generated. Its state does not change if no match signal is generated. Consequently, this flag must be cleared with an instruction.

In the 16-bit mode (T0LONG = 1), a match needs to occur in all 16 bits of data for a match signal to occur.

T0HIE (bit 2): T0H interrupt request enable control

When this bit and T0HCMP are set to 1, an interrupt request to vector address 0023H is generated.

T0LCMP (bit 1): T0L match flag

This bit is set when the value of T0L matches the value of the match buffer register for T0L while T0L is running (T0LRUN = 1) and a match signal is generated. Its state does not change if no match signal is generated. Consequently, this flag must be cleared with an instruction.

In the 16-bit mode (T0LONG = 1), a match needs to occur in all 16 bits of data for a match signal to occur.

T0LIE (bit 0): T0L interrupt request enable control

When this bit and T0LCMP are set to 1, an interrupt request to vector address 0013H is generated.

Notes:

- *T0HCMP and T0LCMP must be cleared to 0 with an instruction.*
- *When the 16-bit mode is to be used, T0LRUN and T0HRUN must be set to the same value to control operation.*
- *T0LCMP and T0HCMP are set at the same time in the 16-bit mode.*

3.6.4.2 Timer 0 programmable prescaler match register (T0PRR)
1) Timer 0 programmable prescaler match register is an 8-bit register that is used to define the clock period (Tpr) of timer/counter 0.
2) The count value of the prescaler starts at 0 when T0PRR is loaded with data.
3) $TPr = (T0PRR + 1) \times Tcyc$ Tcyc = Period of cycle clock

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE11	0000 0000	R/W	T0PRR	T0PRR7	T0PRR6	T0PRR5	T0PRR4	T0PRR3	T0PRR2	T0PRR1	T0PRR0

3.6.4.3 Timer/counter 0 low byte (T0L)
1) This is a read-only 8-bite timer/counter. It counts the number of match signals from the prescaler or external signals.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE12	0000 0000	R	T0L	T0L7	T0L6	T0L5	T0L4	T0L3	T0L2	T0L1	T0L0

3.6.4.4 Timer/counter 0 high byte (T0H)
1) This is a read-only 8-bite timer/counter. It counts the number of match signals from the prescaler or overflows occurring in T0L.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE13	0000 0000	R	T0H	T0H7	T0H6	T0H5	T0H4	T0H3	T0H2	T0H1	T0H0

3.6.4.5 Timer/counter 0 match data register low byte (T0LR)
1) This register is used to store the match data for T0L. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with the lower-order byte of timer/counter 0 (16 bits of data need match in the 16-bit mode).
2) The match buffer register is updated as follows:

The match register matches T0LR when it is inactive (T0LRUN = 0).

When the match register is running (T0LRUN = 1), it is loaded with the contents of T0LR when a match signal is generated.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE14	0000 0000	R/W	T0LR	T0LR7	T0LR6	T0LR5	T0LR4	T0LR3	T0LR2	T0LR1	T0LR0

3.6.4.6 Timer/counter 0 match data register high byte (T0HR)

1) This register is used to store the match data for T0H. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with the higher-order byte of timer/counter 0 (16 bits of data need match in the 16-bit mode)

2) The match buffer register is updated as follows:

The match register matches T0HR when it is inactive (T0HRUN = 0).

When the match register is running (T0HRUN = 1), it is loaded with the contents of T0HR when a match signal is generated.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE15	0000 0000	R/W	T0HR	T0HR7	T0HR6	T0HR5	T0HR4	T0HR3	T0HR2	T0HR1	T0HR0

3.6.4.7 Timer/counter 0 capture register low byte (T0CAL)

1) This register is a read-only 8-bit register used to capture the contents of timer/counter 0 low byte (T0L) on an external input detection signal.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE16	XXXX XXXX	R	T0CAL	T0CAL7	T0CAL6	T0CAL5	T0CAL4	T0CAL3	T0CAL2	T0CAL1	T0CAL0

3.6.4.8 Timer/counter 0 capture register high byte (T0CAH) (8-bit register)

1) This register is a read-only 8-bit register used to capture the contents of timer/counter 0 high byte (T0H) on an external input detection signal.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE17	XXXX XXXX	R	T0CAH	T0CAH7	T0CAH6	T0CAH5	T0CAH4	T0CAH3	T0CAH2	T0CAH1	T0CAH0

3.7　High-speed Clock Counter

3.7.1　Overview

The high-speed clock counter is a 3-bit counter that is provided with a realtime output capability. It is coupled with timer/counter 0 to form a 11- or 19-bit high-speed counter. It can accept clocks with periods of as short as $\frac{1}{6}$ the cycle time. The high-speed clock counter is also equipped with a 4-bit capture register incorporating a carry bit.

3.7.2　Functions

1) 11-bit or 19-bit programmable high-speed counter
   * The 11-bit or 19-bit timer/counter, in conjunction with the timer/counter 0 low byte (T0L) and timer/counter 0 high byte (T0H), functions as a 11- or 19-bit programmable high-speed counter that counts up the external input signals from the P72/INT2/T0IN /NKIN pin. The coupled timer/counter 0 counts the number of overflows occurring in the 3-bit counter. In this case, timer 0 functions as a free-running counter.

2) Realtime Output
   * A realtime output is placed at pin P17. Realtime output is a function to change the state of output at a port into realtime when the count value of a counter reaches the required value. This change in output occurs asynchronously with any clock for the microcontroller.

3) Capture Operation
   * The value of the high-speed clock counter is captured into NKCOV and NKCAP2 to NKCAP0 in synchronization with the capture operation of T0L (timer 0 low byte). NKCOV is a carry into timer/counter 0. When this bit is set to 1, the capture value of timer/counter 0 must be corrected by +1. NKCAP2 to NKCAP0 carry the capture value of the high-speed clock counter.

4) Interrupt generation
   * The required timer/counter 0 flag is set when the high-speed clock counter and timer/counter 0 keep counting and their count value reaches "(timer 0's match register value+1) × 8+ NKCMP2 to NKCMP0." In this case, a T0L or T0H interrupt request is generated if the interrupt request enable bit is set.

5) To control the high-speed clock counter, it is necessary to manipulate the following special function registers:
   * NKREG, P1TST, T0CNT, T0L, T0H, T0LR, T0HR, ISL, I01CR, I23CR

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7D	0000 0000	R/W	NKREG	NKEN	NKCMP2	NKCMP1	NKCMP0	NKCOV	NKCAP2	NKCAP1	NKCAP0
FE47	0000 H0H0	R/W	P1TST	FIX0	FIX0	MRCSFT	FIX0	-	DSNKOT	-	FIX0
FE10	0000 0000	R/W	T0CNT	T0HRUN	T0LRUN	T0LONG	T0LEXT	T0HCMP	T0HIE	T0LCNP	T0LIE
FE12	0000 0000	R	T0L	T0L7	T0L6	T0L5	T0L4	T0L3	T0L2	T0L1	T0L0
FE13	0000 0000	R	T0H	T0H7	T0H6	T0H5	T0H4	T0H3	T0H2	T0H1	T0H0
FE14	0000 0000	R/W	T0LR	T0LR7	T0LR6	T0LR5	T0LR4	T0LR3	T0LR2	T0LR1	T0LR0
FE15	0000 0000	R/W	T0HR	T0HR7	T0HR6	T0HR5	T0HR4	T0HR3	T0HR2	T0HR1	T0HR0
FE16	XXXX XXXX	R	T0CAL	T0CAL7	T0CAL6	T0CAL5	T0CAL4	T0CAL3	T0CAL2	T0CAL1	T0CAL0
FE17	XXXX XXXX	R	T0CAH	T0CAH7	T0CAH6	T0CAH5	T0CAH4	T0CAH3	T0CAH2	T0CAH1	T0CAH0
FE5D	0000 0000	R/W	I01CR	INT1LH	INT1LV	INT1IF	INT1IE	INT0LH	INT0LV	INT0IF	INT0IE
FE5E	0000 0000	R/W	I23CR	INT3HEG	INT3LEG	INT3IF	INT3IE	INT2HEG	INT2LEG	INT2IF	INT2IE
FE5F	0000 0000	R/W	ISL	ST0HCP	ST0LCP	BTIMC1	BTIMC0	BUZON	NFSEL	NFON	ST0IN

3.7.3 Circuit Configuration

3.7.3.1 High-speed clock counter control register (NKREG) (8-bit register)

1) The high-speed clock counter control register controls the high-speed clock counter. It contains the start, count value setting, and counter value capture bits.

2) Start/stop: Controlled by the start/stop operation of timer/counter 0 low byte (T0L) when NKEN = 1.

3) Count clock: External input signals from the P72/INT2/T0IN/NKIN pin.

4) Realtime output: The realtime output port must be placed in the output mode.

When NKEN (BIT7) is set to 0, the realtime output port relinquishes its realtime output capability and synchronizes itself with the data in the port latch.

When the value that will result in NKEN = 1 is written into NKREG, the realtime output port restores its realtime output capability and holds the output data. In this state, the contents of the port latch must be replaced by the next realtime output value.

When the high-speed clock counter keeps counting and reaches the count value "(T0LR + 1) × 8 + value of NKCMP2 to NKCMP0," the realtime output turns to the required value. Subsequently, the realtime output port relinquishes the realtime output capability and synchronizes itself with the data in the port latch. To restore the realtime output capability, a value that will result in NKEN = 1 must be written into NKREG.

5) Capture clock: Generated in synchronization with the capture clock for T0L (timer 0 low byte).

3.7.3.2 P1TST register

1) The realtime output function is enabled when DSNKOT (P1TST register, bit 2) is set to 0.

2) The realtime output function is disabled when DSNKOT (P1TST register, bit 2) is set to 1. In this case, the realtime output pin functions as an ordinary port pin.

3.7.3.3 Timer/counter 0 operation

T0EXT (T0CNT, bit4) must be set to 1 when a high-speed clock counter is to be used.

When NKEN = 1 and T0LONG (T0CNT, bit5) = 0, timer 0H runs in the normal mode and timer 0L is coupled with the high-speed clock counter to form a 11-bit free-running counter. When NKEN = 1 and T0LONG (T0CNT, bit5) = 1, timer 0 is coupled with the NK counter to form a 19-bit free-running counter.

When a free-running counter reaches the count value "(timer 0 match register value + 1) × 8 + value of NKCMP2 to NKCMP0," a match detection signal occurs, generating the realtime output of the required value and setting the match flag of timer 0. No new match signal is detected until the next NKREG write operation is performed.

The match data for these free-running counters must always be greater than the current counter value. When updating the match data, the match register for timer 0 must be set up before loading the match register for NKREG (NKCMP2 to KNCMP0) with data. Even if the same value is loaded, it must be written into NKREG to start a search for a match.

3.7.4   Related Registers

1)   This register is an 8-bit register that controls the operation of the high-speed clock counter.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7D	0000 0000	RW	NKREG	NKEN	NKCMP2	NKCMP1	NKCMP0	NKCOV	NKCAP2	NKCAP1	NKCAP0

NKEN (bit 7): Counter control

When set to 0, the NK control circuit is inactive.

When set to 1, the NK control circuit is active. The timer 0 operation is switched to make up an asynchronous high-speed counter with timer 0 being the higher-order counter. Counting is started by setting this bit to 1 and starting timer 0 in the external clock mode.

NKCMP2-NKCMP0 (bits 6-4): Match register

Immediately when the counter reaches the value equivalent to "(timer 0's match register value+1) × 8 + value of NKCMP2 to NKCMP0 + 8," a match detected signal occurs, generating the realtime output of the required value and setting the timer 0's match flag. Subsequently, the realtime output port relinquishes the realtime output capability and changes its state in synchronization with the data in the port latch. The realtime output function and match detection function will not be resumed until the next NKREG write operation is performed.

NKCOV, NKCAP2-NKCAP0 (bits 3-0): Capture register

The NK counter value is captured into these bits in synchronization with the timer 0L capture operation.

NKCOV is a carry into timer 0. When this bit is set to 1, the capture value of timer 0 must be corrected by +1.

NKCAP2 to NKCAP0 carry the capture value of the NK counter. These bits are read only.

NK Counter

T0LONG = 0 (Timer 0: 8-bit mode)

```
┌──────────────────────┐              ┌──────────────────────┐
│  NK Comparison Value │              │  T0L Comparison Value│
└──────────┬───────────┘              └──────────┬───────────┘
           │                                     │
           ▼                                     ▼
┌──────────────────┐   ┌──────────┐   ┌──────────────────┐
│ NK Counter 3 Bits│──▶│ NK Carry │──▶│ T0L Counter 8 Bits│──▶ TOL Match Signal
└────────┬─────────┘   └────┬─────┘   └────────┬─────────┘
         │                  │                  │
         ▼                  ▼                  ▼
┌────────────────────────────────┐   ┌──────────────────┐
│          NK Capture            │◀──│    T0L Capture   │◀──●── Capture Signal
└────────────────────────────────┘   └──────────────────┘
```
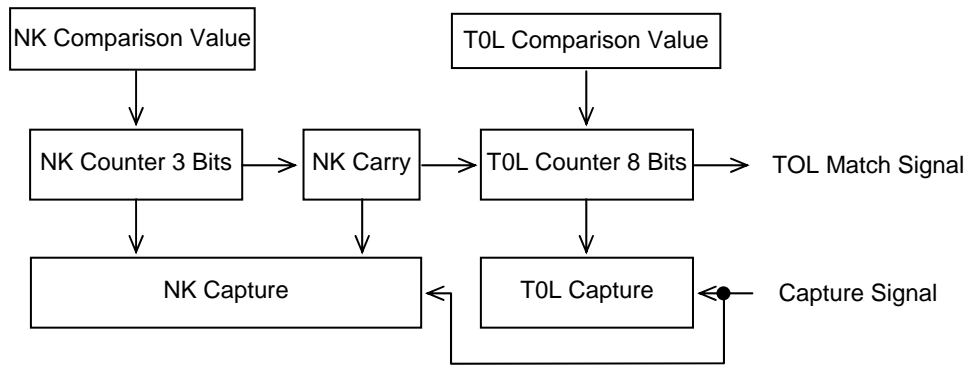
T0LONG = 1 (Timer 0: 16-bit mode)

```
┌──────────────────────┐        ┌──────────────────────┐   ┌──────────────────────┐
│  NK Comparison Value │        │  T0L Comparison Value│   │  T0H Comparison Value│
└──────────┬───────────┘        └──────────┬───────────┘   └──────────┬───────────┘
           │                               │                          │
           ▼                               ▼                          ▼
┌──────────────────┐  ┌──────────┐  ┌──────────────────┐  ┌──────────────────┐
│ NK Counter 3 Bits│─▶│ NK Carry │─▶│ T0L Counter 8 Bits│─▶│ T0H Counter 8 Bits│─▶ T0L Match Signal
└────────┬─────────┘  └────┬─────┘  └────────┬─────────┘  └────────┬─────────┘
         │                 │                 │                     │
         ▼                 ▼                 ▼                     ▼
┌────────────────────────────────┐  ┌──────────────┐  ┌──────────────┐
│          NK Capture            │◀─│  T0L Capture │◀─│  T0H Capture │◀──●── Capture Signal
└────────────────────────────────┘  └──────────────┘  └──────────────┘
```

3-34
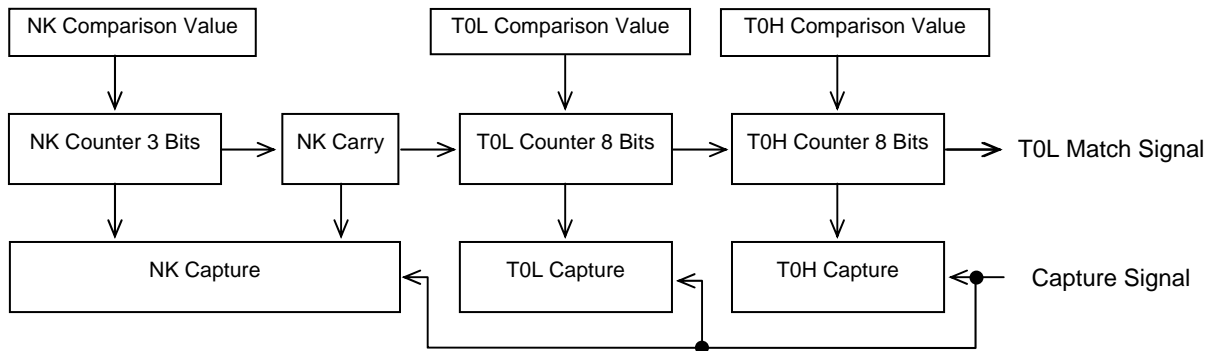
3.8   Timer/Counter 1 (T1)

3.8.1   Overview

The timer/counter 1 (T1) incorporated in this series of microcontrollers is a 16-bit timer/counter with a prescaler that provides the following four functions:

1)   Mode 0:   Two channels of 8-bit programmable timer with an 8-bit prescaler (with toggle output) + 8-bit programmable timer/counter (with toggle output)

2)   Mode 1:   Two channels of 8-bit PWM with an 8-bit prescaler

3)   Mode 2:   16-bit programmable timer/counter with an 8-bit prescaler (with toggle output) (the lower-order 8 bits may be used as a timer/counter with toggle output.)

4)   Mode 3:   16-bit programmable timer with an 8-bit prescaler (with toggle output) (the lower-order 8 bits may be used as a PWM.)

3.8.2   Functions

1)   Mode 0:   Two channels of 8-bit programmable timer with an 8-bit prescaler (with toggle output) + 8-bit programmable timer/counter (with toggle output)

- T1L functions as an 8-bit programmable timer/counter that counts the number of signals obtained by dividing the cycle clock by 2 or external events while T1H functions as an 8-bit programmable timer that counts the number of signals obtained by dividing the cycle clock by 2.

- Two independent 8-bit programmable timers (T1L and T1H) run on a clock that is obtained by dividing the cycle clock by 2.

- T1PWML and T1PWMH generate a signal that toggles at the interval of T1L and T1H period, respectively. (Note 1)

  T1L period = (T1LR+1) × (T1LPRC count) × 2Tcyc or (T1LR+1) × (T1LPRC count) events detected

  T1PWML period = T1L period × 2

  T1H period = (T1HR+1) × (T1HPRC count) × 2Tcyc

  T1PWMH period = T1H period × 2

2)   Mode 1:   Two channels of 8-bit PWM with an 8-bit prescaler

- Two independent 8-bit PWMs (T1PWML and T1PWMH) run on the cycle clock.

  T1PWML period = 256 × (T1LPRC count) × Tcyc

  T1PWML low period = (T1LR+1) × (T1LPRC count) × Tcyc

  T1PWMH period = 256 × (T1HPRC count) × Tcyc

  T1PWMH low period = (T1HR+1) × (T1HPRC count) × Tcyc

3)   Mode 2:   16-bit programmable timer/counter with an 8-bit prescaler (with toggle output) (the lower-order 8 bits may be used as a timer/counter with toggle output.)

- A 16-bit programmable timer/counter runs that counts the number of signals whose frequency is equal to that of the cycle clock divided by 2 or the number of external events. Since interrupts can occur from the lower-order 8-bit timer (T1L) at the interval of T1L period, the lower-order 8 bits of this 16-bit programmable timer/counter can be used as the reference timer.

- T1PWML and T1PWMH generate a signal that toggles at the interval of T1L and T1 periods, respectively. (Note 1)

  T1L period = (T1LR+1) × (T1LPRC count) × 2Tcyc
             or (T1LR+1) × (T1LPRC count) events detected

  T1PWML period = (T1L period) × 2

  T1 period = (T1HR+1) × (T1HPRC count) × T1L period
             or (T1HR+1) × (T1HPRC count) × (T1LR+1) × (T1LPRC count) events detected

  T1PWMH period = T1 period × 2

T1

4) Mode 3: 16-bit programmable timer with an 8-bit prescaler (with toggle output) (the lower-order 8 bits may be used as a PWM.)

- A 16-bit programmable timer runs on the cycle clock.
- The lower-order 8 bits run as a PWM (T1PWML) having a period of 256 Tcyc.
- T1PWMH generates a signal that toggles at the interval of T1 period. (Note 1)

T1PWML period = 256 × (T1LPRC count) × Tcyc
T1PWML low period = (T1LR+1) × (T1LPRC count) × Tcyc
T1 period = (T1HR+1) × (T1LPRC count) × T1PWML period
T1PWMH period = T1 period × 2

5) Interrupt generation

T1L or T1H interrupt request is generated at the counter period of the T1L or T1H timer if the interrupt request enable bit is set.

6) To control timer 1 (T1), it is necessary to manipulate the following special function registers:

- T1CNT, T1L, T1H, T1LR, T1HR, T1PRR
- P1, P1DDR, P1FCR
- P2, P2DDR, I45CR, I45SL

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE18	0000 0000	R/W	T1CNT	T1HRUN	T1LRUN	T1LONG	T1PWM	T1HCMP	T1HIE	T1LCMP	T1LIE
FE1A	0000 0000	R	T1L	T1L7	T1L6	T1L5	T1L4	T1L3	T1L2	T1L1	T1L0
FE1B	0000 0000	R	T1H	T1H7	T1H6	T1H5	T1H4	T1H3	T1H2	T1H1	T1H0
FE1C	0000 0000	R/W	T1LR	T1LR7	T1LR6	T1LR5	T1LR4	T1LR3	T1LR2	T1LR1	T1LR0
FE1D	0000 0000	R/W	T1HR	T1HR7	T1HR6	T1HR5	T1HR4	T1HR3	T1HR2	T1HR1	T1HR0
FE19	0000 0000	R/W	T1PRR	T1HPRE	T1HPRC2	T1HPRC1	T1HPRC0	T1LPRE	T1PRC2	T1LPRC1	T1LPRC0

Note 1: The output of the T1PWML is fixed at the high level if the T1L is stopped. If the T1L is running, the output of the T1PWML is fixed at the low level when T1LR = FFH. The output of T1PWMH is fixed at the high level if the T1H is stopped. If the T1H is running, the output of the T1PWMH is fixed at the low level when T1HR   =FFH.

3.8.3 Circuit Configuration

3.8.3.1 Timer 1 control register (T1CNT) (8-bit register)

1) The timer 1 control register controls the operation and interrupts of the T1L and T1H.

3.8.3.2 Timer 1 prescaler control register (T1PRR) (8-bit counter)

1) This register sets the clocks for T1L and T1H.

3.8.3.3 Timer 1 prescaler low byte (8-bit counter)

1) Start/stop: The start/stop of timer 1 prescaler low byte is controlled by the 0/1 value of T1LRUN (timer 1 control register, bit 6).

2) Count clock: Varies with the operating mode.

Mode	T1LONG	T1PWM	T1L Prescaler Count Clock
0	0	0	2 Tcyc/events (Note 1)
1	0	1	1 Tcyc (Note 2)
2	1	0	2 Tcyc/events (Note 1)
3	1	1	1 Tcyc (Note 2)

Note 1: T1L serves as an event counter when INT4 or INT5 is specified as the timer 1 count clock input in the external interrupt 4/5 pin select register (I45SL). It serves as a timer that runs using 2Tcyc as its count clock if neither INT4 nor INT5 are specified as the timer 1 count clock input.

Note 2: T1L will not run normally if INT4 or INT5 is specified as the timer 1 count clock input when T1PWM = 1. When T1PWM = 1, do not specify INT4 or INT5 as the timer 1 count clock input.

3) Prescaler count: Determined by the T1PRC value.

The count clock for T1L is generated at the intervals determined by the prescaler count.

T1LPRE	T1LPRC2	T1LPRC1	T1LPRC0	T1L Prescaler Count
0	–	–	–	1
1	0	0	0	2
1	0	0	1	4
1	0	1	0	8
1	0	1	1	16
1	1	0	0	32
1	1	0	1	64
1	1	1	0	128
1	1	1	1	256

4) Reset: When the timer 1 stops operation or a T1L reset signal is generated.

3.8.3.4   Timer 1 prescaler high byte   (8-bit counter)

1)   Start/stop: The start/stop of timer 1 prescaler high byte is controlled by the 0/1 value of T1HRUN
(timer 1 control register, bit 7).

2)   Count clock: Varies with the mode.

Mode	T1LONG	T1PWM	T1H Prescaler Count Clock
0	0	0	2 Tcyc
1	0	1	1 Tcyc
2	1	0	T1L match signal
3	1	1	$256 \times$ (T1LPRC count) $\times$ Tcyc

3)   Prescaler count: Determined by the T1PRC value.

The count clock for T1H is generated at the intervals determined by the prescaler count.

T1HPRE	T1HPRC2	T1HPRC1	T1HPRC0	T1H Prescaler Count
0	–	–	–	1
1	0	0	0	2
1	0	0	1	4
1	0	1	0	8
1	0	1	1	16
1	1	0	0	32
1	1	0	1	64
1	1	1	0	128
1	1	1	1	256

4)   Reset: When the timer 1 stops operation or a T1H reset signal is generated.

3.8.3.5   Timer 1 low byte (T1L)   (8-bit counter)

1)   Start/stop: The start/stop of the timer 1 low byte is controlled by the 0/1 value of T1LRUN (timer 1
control register, bit 6).

2)   Count clock: T1L prescaler output clock.

3)   Match signal: A match signal is generated when the count value matches the value of the match
buffer register.

4)   Reset: The timer 1 low byte is reset when it stops operation or a match signal occurs on the mode 0
or,2 condition.

3.8.3.6   Timer 1 high byte (T1H)   (8-bit counter)

1)   Start/stop: The start/stop of the timer 1 high byte is controlled by the 0/1 value of T1HRUN (timer 1
control register, bit 7).

2)   Count clock: T1H prescaler output clock

3)   Match signal: A match signal is generated when the count value matches the value of the match
buffer register.

4)   Reset: The timer 1 high byte is reset when it stops operation or a match signal occurs on the mode
0,2 or 3 condition.

3.8.3.7   Timer 1 match data register low byte (T1LR)   (8-bit register with a match buffer register)

1)   This register is used to store the match data for T1L. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with that of timer 1 low byte (T1L)

2)   The match buffer register is updated as follows:

T1LR and the match register has the same value when in inactive state (T1LRUN = 0).

If active (T1LRUN = 1), the match buffer register is loaded with the contents of T1LR when the value of T1L reaches 0.

3.8.3.8   Timer 1 match data register high byte (T1HR)   (8-bit register with a match buffer register

1)   This register is used to store the match data for T1H. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with that of timer 1 high byte (T1H).

2)   The match buffer register is updated as follows:

T1HR and the match register have the same value when in inactive state (T1HRUN = 0).

If active (T1HRUN = 1), the match buffer register is loaded with the contents of T1HR when the value of T1H reaches 0.

3.8.3.9   Timer 1 low byte output (T1PWML)

1)   The T1PWML output is fixed at the high level when T1L is inactive. If T1L is active, the T1PWML output is fixed at the low level when T1LR = FFH.

2)   Timer 1 low byte output is a toggle output whose state changes on a T1L match signal when T1PWM (timer 0 control register, bit 4) is set to 0.

3)   When T1PWM (timer 0 control register, bit 4) is set to 1, this PWM output is cleared on an T1L overflow and set on a T1L match signal.

3.8.3.10   Timer 1 high byte output (T1PWMH)

1)   The T1PWMH output is fixed at the high level when T1H is inactive. If T1H is active, the T1PWMH output is fixed at the low level when T1HR = FFH.

2)   The timer 1 high byte output is a toggle output whose state changes on a T1H match signal when T1PWM = 0 or T1LONG = 1.

3)   When T1PWM = 1 and T1LONG = 0, this PWM output is cleared on a T1H overflow and set on a T1H match signal.

Left diagram (Mode 0 - low):

2Tcyc
or
external events
Sets in
I45CR(FE4Ah)
I45SL(FE4Bh)
registers

Clock → T1L prescaler

Clock → T1L ← Clear

↓

Comparator — Match → T1PWML output (Invert)

↑

Match buffer register ← Reload

↑

T1LR → T1LCMP flag set

⟨ 8-bit programmable timer ⟩

Right diagram (Mode 0 - high):

2Tcyc → Clock → T1H prescaler

Clock → T1H ← Clear

↓

Comparator — Match → T1PWMH output (Invert)

↑

Match buffer register ← Reload

↑

T1HR → T1HCMP flag set

⟨ 8-bit programmable timer ⟩

Figure. 3.8.1   Mode 0 (T1LONG = 0, T1PWM = 0) Block Diagram

Left diagram (Mode 1 - low):

1Tcyc → Clock → T1L prescaler

Clock → T1L — Overflow → Reset

T1PWML output

↓

Comparator — Match

Set

↑

Match buffer register ← Reload

↑

T1LR → T1LCMP flag set

⟨ 8-bit PWM ⟩

Right diagram (Mode 1 - high):

1Tcyc → Clock → T1H prescaler

Clock → T1H — Overflow → Reset

T1PWMH output

↓

Comparator — Match

Set

↑

Match buffer register ← Reload

↑

T1HR → T1HCMP flag set

⟨ 8-bit PWM ⟩
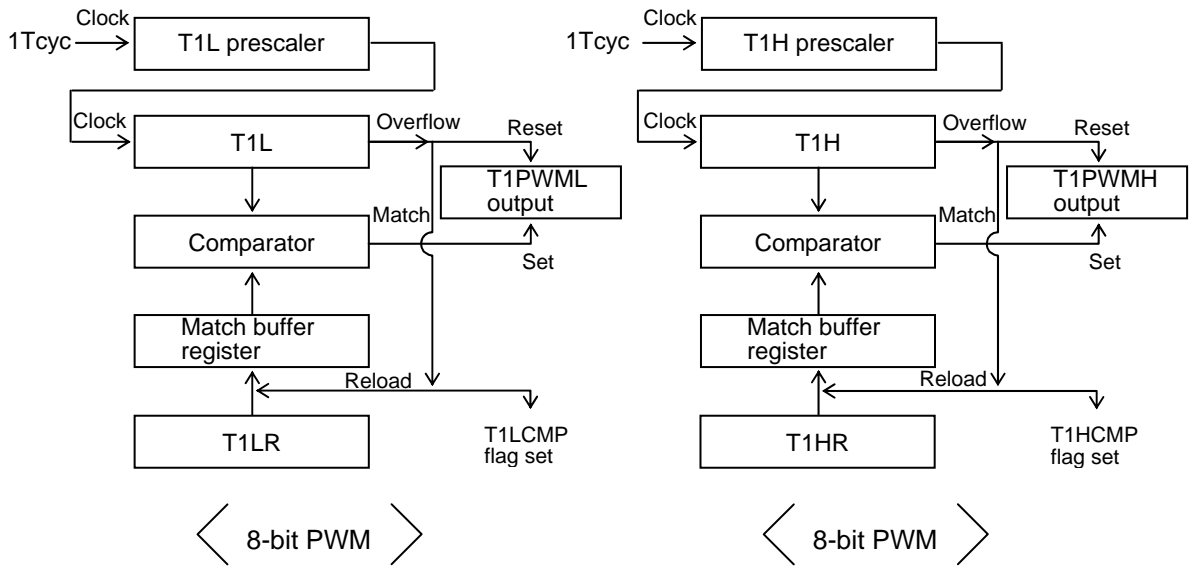
Figure. 3.8.2   Mode 1 (T1LONG = 0, T1PWM = 1) Block Diagram
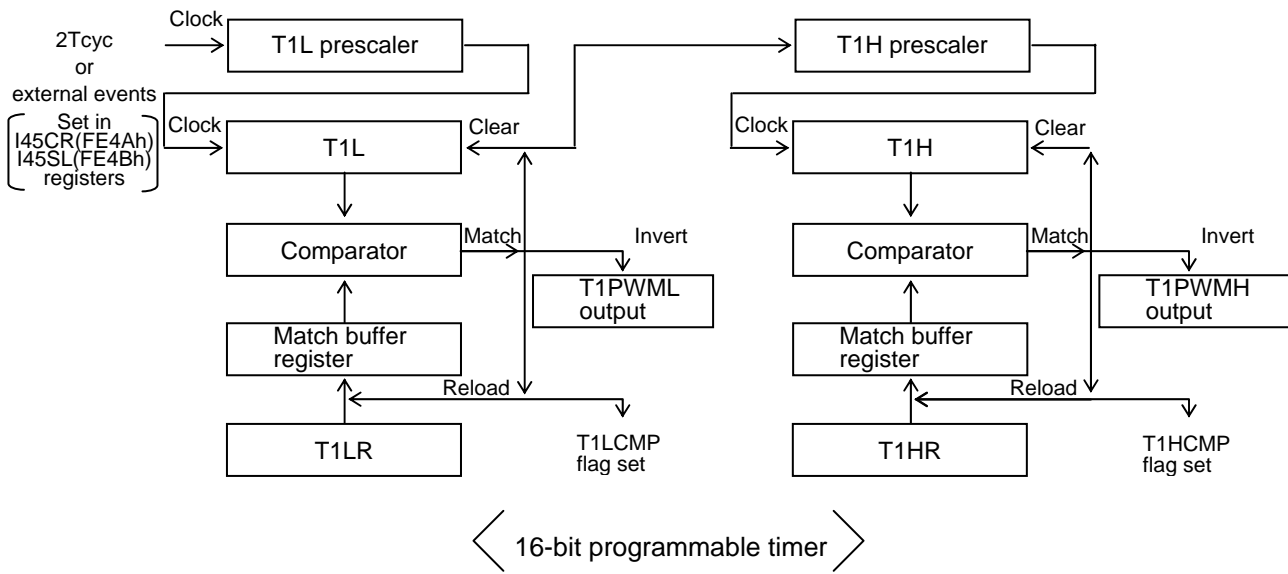
Fig. 3.8.3   Mode 2 (T1LONG = 1, T1PWM = 0) Block Diagram
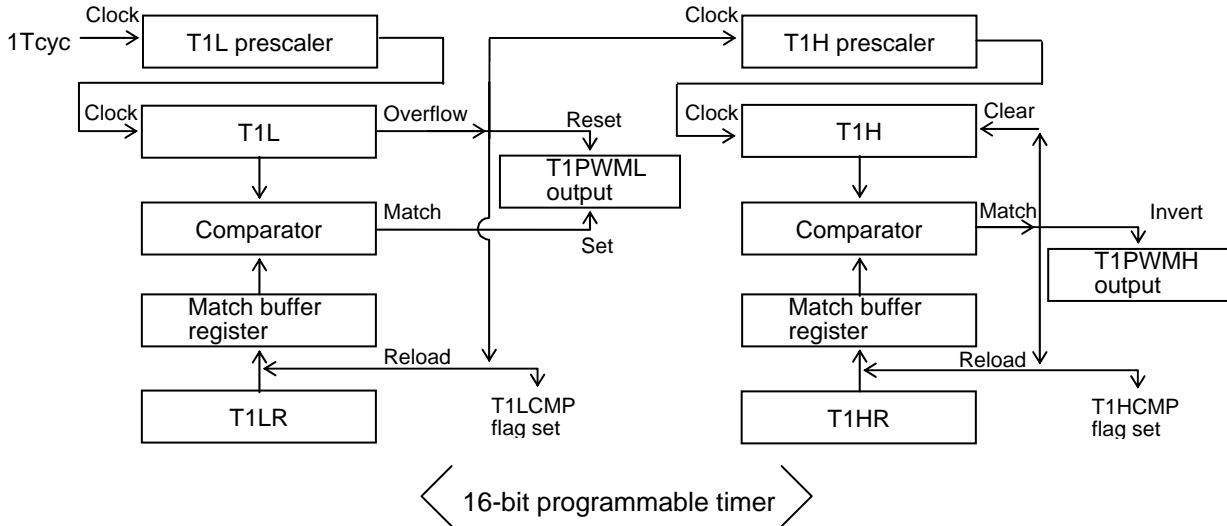
Fig. 3.8.4   Mode 3 (T1LONG = 1, T1PWM = 1) Block Diagram

3-41

3.8.4　Related Registers

3.8.4.1　Timer 1 control register (T1CNT)

1) Timer 1 control register is an 8-bit register that controls the operation and interrupts of T1L and T1H.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE18	0000 0000	R/W	T1CNT	T1HRUN	T1LRUN	T1LONG	T1PWM	T1HCMP	T1HIE	T1LCMP	T1LIE

T1HRUN (bit 7): T1H count control

When this bit is set to 0, timer 1 high byte (T1H) stops on a count value of 0. The match buffer register of T1H has the same value as T1HR.

When this bit is set to 1, timer 1 high byte (T1H) performs the required counting operation.

T1LRUN (bit 6): T1L count control

When this bit is set to 0, timer 1 low byte (T1L) stops on a count value of 0. The match buffer register of T1L has the same value as T1LR.

When this bit is set to 1, timer 1 low byte (T1L) performs the required counting operation.

T1LONG (bit 5): Timer 1 bit length select

When this bit is set to 0, timer 1's higher- and lower-order bytes serve as independent 8-bit timers.

When this bit is set to 1, timer 1 serves as a 16-bit timer since the timer 1 high byte (T1H) counts up at the interval of the timer 1 low byte (T1L).

Independent match signals are generated from T1H and T1L when their count value matches the contents of the corresponding match buffer register, regardless of the value of this bit.

T1PWM (bit 4): T1 output mode select

This bit and T1LONG (bit 5) determine the output mode of T1 (T1PWMH and T1PWML) as summarized in Table 3.8.1.

Table 3.8.1 Timer 1 Output (T1PWMH, T1PWML)

Mode	T1LONG	T1PWM	T1PWMH		T1PWML	
0	0	0	Toggle output	Period: $(T1HR+1) \times (T1HPRC\ count) \times 4 \times Tcyc$	Toggle output or	Period: $(T1LR+1) \times (T1LPRC\ count) \times 4 \times Tcyc$ Period: $2(T1LR+1) \times (T1LPRC\ count)$ events
1	0	1	PWM output	Period: $256 \times (T1HPRC\ count) \times Tcyc$	PWM output	Period: $256 \times (T1LPRC\ count) \times Tcyc$
2	1	0	Toggle output or	Period: $(T1HR+1) \times (T1HPRC\ count) \times (T1PWML\ period)$ Period: $2(T1HR+1) \times (T1HPRC\ count) \times (T1LR+1) \times (T1LPRC)$ events	Toggle output or	Period: $(T1LR+1) \times (T1LPRC\ count) \times 4 \times Tcyc$ Period: $2(T1LR+1) \times (T1LPRC\ count)$ events
3	1	1	Toggle output	Period: $(T1HR+1) \times (T1HPRC\ count) \times (T1PWML\ period) \times 2$	PWM output	Period: $256 \times (T1LPRC\ count) \times Tcyc$

T1HCMP (bit 3): T1H match flag

This flag is set if T1H reaches 0 when T1H is active (T1HRUN = 1).

This flag must be cleared with an instruction.

T1HIE (bit 2): T1H interrupt request enable control

An interrupt request is generated to vector address 002BH when this bit and T1HCMP are set to 1.

T1LCMP (bit 1): T1L match flag

This flag is set if T1L reaches 0 when T1L is active (T1LRUN = 1).

This flag must be cleared with an instruction.

T1LIE (bit 0): T1L interrupt request enable control

An interrupt request is generated to vector address 002BH when this bit and T1LCMP are set to 1.

Note:

• *T1HCMP and T1LCMP must be cleared to 0 with an instruction.*

3.8.4.2 Timer 1 prescaler control register (T1PRR)

1) This register sets up the count values for the timer 1 prescaler.
2) When the register value is changed while the timer is running, the change is reflected in the prescaler operation at the same timing when the match buffer register for the timer (T1L, T1H) is updated.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE19	0000 0000	R/W	T1PRR	T1HPRE	T1HPRC2	T1HPRC1	T1HPRC0	T1LPRE	T1LPRC2	T1LPRC1	T1LPRC0

T1HPRE (bit 7): Controls the timer 1 prescaler high byte.

T1HPRC2 (bit 6): Controls the timer 1 prescaler high byte.

T1HPRC1 (bit 5): Controls the timer 1 prescaler high byte.

T1HPRC0 (bit 4): Controls the timer 1 prescaler high byte.

T1HPRE	T1HPRC2	T1HPRC1	T1HPRC0	T1H Prescaler Count
0	–	–	–	1
1	0	0	0	2
1	0	0	1	4
1	0	1	0	8
1	0	1	1	16
1	1	0	0	32
1	1	0	1	64
1	1	1	0	128
1	1	1	1	256

T1LPRE (bit 3): Controls the timer 1 prescaler low byte.

T1LPRC2 (bit 2): Controls the timer 1 prescaler low byte.

T1LPRC1 (bit 1): Controls the timer 1 prescaler low byte.

T1LPRC0 (bit 0): Controls the timer 1 prescaler low byte.

T1LPRE	T1LPRC2	T1LPRC1	T1LPRC0	T1L Prescaler Count
0	–	–	–	1
1	0	0	0	2
1	0	0	1	4
1	0	1	0	8
1	0	1	1	16
1	1	0	0	32
1	1	0	1	64
1	1	1	0	128
1	1	1	1	256

3.8.4.3 Timer 1 low byte (T1L)
1) This is a read-only 8-bit timer. It counts up on every T1L prescaler output clock.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE1A	0000 0000	R	T1L	T1L7	T1L6	T1L5	T1L4	T1L3	T1L2	T1L1	T1L0

3.8.4.4 Timer 1 high byte (T1H)
1) This is a read-only 8-bit timer. It counts up on every T1H prescaler output clock.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE1B	0000 0000	R	T1H	T1H7	T1H6	T1H5	T1H4	T1H3	T1H2	T1H1	T1H0

3.8.4.5 Timer 1 match data register low byte (T1LR)
1) This register is used to store the match data for T1L. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with the value of timer 1 low byte.
2) Match buffer register is updated as follows:

T1LR and the match register has the same value when in inactive (T1LRUN = 0).

If active (T1LRUN = 1), the match buffer register is loaded with the contents of T1LR when the value of T1L reaches 0.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE1C	0000 0000	R/W	T1LR	T1LR7	T1LR6	T1LR5	T1LR4	T1LR3	T1LR2	T1LR1	T1LR0

3.8.4.6 Timer 1 match data register high byte (T1HR)
1) This register is used to store the match data for T1H. It has an 8-bit match buffer register. A match signal is generated when the value of this match buffer register coincides with the value of timer 1 high byte.
2) The match buffer register is updated as follows:

T1HR and the match register has the same value when in inactive (T1HRUN = 0).

If active (T1HRUN = 1), the match buffer register is loaded with the contents of T1HR when the value of T1H reaches 0.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE1D	0000 0000	R/W	T1HR	T1HR7	T1HR6	T1HR5	T1HR4	T1HR3	T1HR2	T1HR1	T1HR0

Mode 0, 2

Match buffer register value

T1L,T1H

Match signal

Interrupt flag set

T1PWML,T1PWMH

Mode 1

Counter value = FFH

Match

T1L、T1H

Match signal

Counter value = FFH
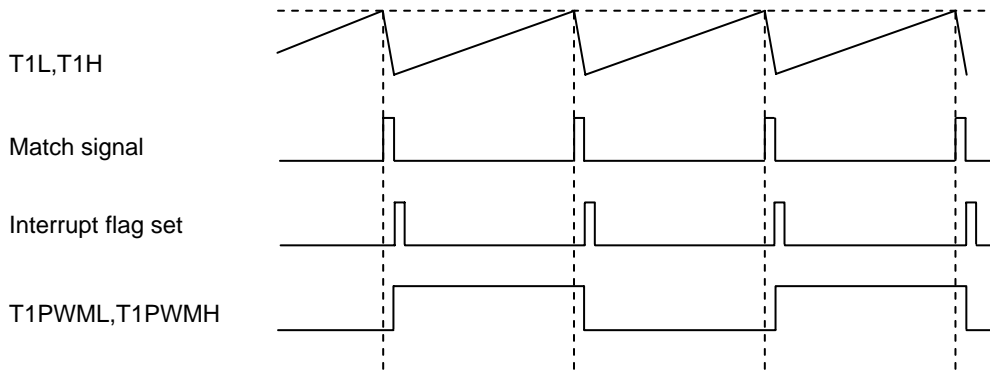
Interrupt flag set
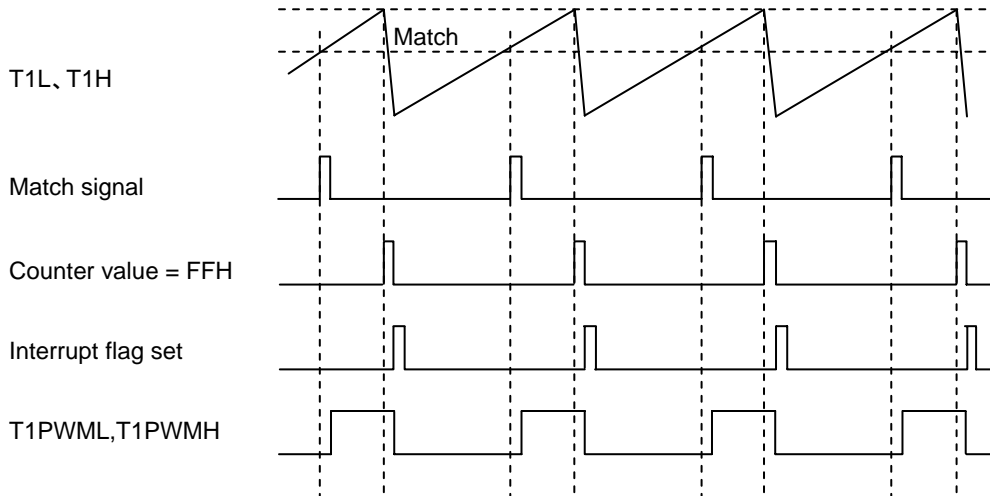
T1PWML,T1PWMH

Mode 3

Match buffer register value

T1H

Match signal

Interrupt flag set

T1PWMH
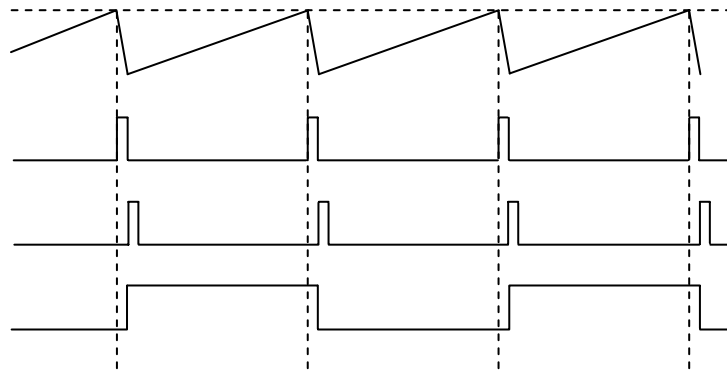
Counter value = FFH

T1L

Match

Match signal

Counter value = FFH

Interrupt flag set

T1PWML
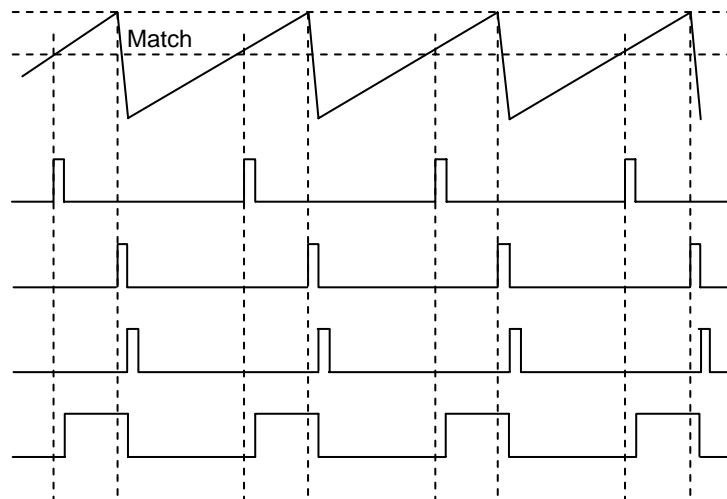
3.9   Timer 6 and Timer 7 (T6, T7)

3.9.1   Overview

The timer 6 (T6) and timer 7 (T7) incorporated in this series of microcontrollers are 8-bit timers with two independently controlled 6-bit prescalers.

3.9.2   Functions

1)   Timer 6 (T6)

Timer 6 is an 8-bit timer that runs on either 4Tcyc, 16Tcyc, or 64Tcyc clock. It can generate, at pin P06, toggle waveforms whose frequency is equal to the period of timer 6.

$$\text{T6 period} = (\text{T6R}+1) \times 4^n \text{Tcyc} \ (n=1, 2, 3)$$
$$\text{Tcyc} = \text{Period of cycle clock}$$

2)   Timer 7 (T7)

Timer 7 is an 8-bit timer that runs on either 4Tcyc, 16Tcyc, or 64Tcyc clock. It can generate, at pin P07, toggle waveforms whose frequency is equal to the period of timer 7.

$$\text{T7 period} = (\text{T7R}+1) \times 4^n \text{Tcyc} \ (n=1, 2, 3)$$
$$\text{Tcyc} = \text{Period of cycle clock}$$

3)   Interrupt generation

Interrupt requests to vector address 0043H are generated when the overflow flag is set at the interval of timer 6 or timer 7 period and the corresponding interrupt request enable bit is set.

4)   To control the timer 6 (T6) and timer 7 (T7), it is necessary to manipulate the following special function registers:

• T67CNT, T6R, T7R, P0FCR

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE78	0000 0000	R/W	T67CNT	T7C1	T7C0	T6C1	T6C0	T7OV	T7IE	T6OV	T6IE
FE7A	0000 0000	R/W	T6R	T6R7	T6R6	T6R5	T6R4	T6R3	T6R2	T6R1	T6R0
FE7B	0000 0000	R/W	T7R	T7R7	T7R6	T7R5	T7R4	T7R3	T7R2	T7R1	T7R0
FE42	00HH 0000	R/W	P0FCR	T7OE	T6OE	-	-	CLKOEN	CKODV2	CKODV1	CKODV0

3.9.3   Circuit Configuration

3.9.3.1   Timer 6/7 control register (T67CNT)   (8-bit register)
1)   The timer 6/7 control register controls the operation and interrupts of T6 and T7.

3.9.3.2   Timer 6 counter (T6CTR)   (8-bit counter)
1)   The timer 6 counter counts the number of clocks from the timer 6 prescaler (T6PR). The value of timer 6 counter (T6CTR) reaches 0 on the clock following the clock that brought about the value specified in the timer 6 period register (T6R), when the interrupt flag (T6OV) is set.

2)   When T6C0 and T6C1 (T67CNT: FE78, bit 4 and 5) are set to 0, the timer 6 counter stops at a count value of 0. In the other cases, the timer 6 counter continues operation.

3)   When data is written into T6R while timer 6 is running, both the timer 6's prescaler and counter aretemporarily cleared, then restart counting.

3.9.3.3   Timer 6 prescaler (T6PR)   (6-bit counter)

1)   This prescaler is used to define the clock period for the timer 6 determined by T6C0 and T6C1. (T67CNT: FE78, bits 4 and 5).

Table 3.9.1 Timer 6 Count Clocks

T6C1	T6C0	T6 Count Clock
0	0	Timer 6 prescaler and timer/counter are reset.
0	1	4 Tcyc
1	0	16 Tcyc
1	1	64 Tcyc

3.9.3.4   Timer 6 period setting register (T6R)   (8-bit register)

1)   This register defines the period of timer 6.

2)   When data is written into T6R while timer 6 is running, both the timer 6's prescaler and counter are temporarily cleared, then restart counting.

3.9.3.5   Timer 7 counter (T7CTR)   (8-bit counter)

1)   The timer 7 counter counts the number of clocks from the timer 7 prescaler (T7PR). The value of timer 7 counter (T7CTR) reaches 0 on the clock following the clock that brought about the value specified in the timer 7 period register (T7R), when the interrupt flag (T7OV) is set.

2)   When T7C0 and T7C1 (T67CNT: FE78 bits 6 and 7) are set to 0, the timer 7 counter stops at a count value of 0. In the other cases, the timer 7 counter continues operation.

3)   When data is written into T7R while timer 7 is running, both the timer 7's prescaler and counter are temporarily cleared, then restart counting.

3.9.3.6   Timer 7 prescaler (T7PR)   (6-bit counter)

1)   This prescaler is used to define the clock period for the timer 7 determined by T7C0 and T7C1 (T67CNT: FE78 bits 6 and 7).

Table 3.9.2 Timer 7 Count Clocks

T7C1	T7C0	T7 Count Clock
0	0	Timer 7 prescaler and timer/counter are reset.
0	1	4 Tcyc
1	0	16 Tcyc
1	1	64 Tcyc

3.9.3.7   Timer 7 period setting register (T7R)   (8-bit register)

1)   This register defines the period of timer 7.

2)   When data is written into T7R while timer 7 is running, both the timer 7's prescaler and counter are temporarily cleared, then restart counting.
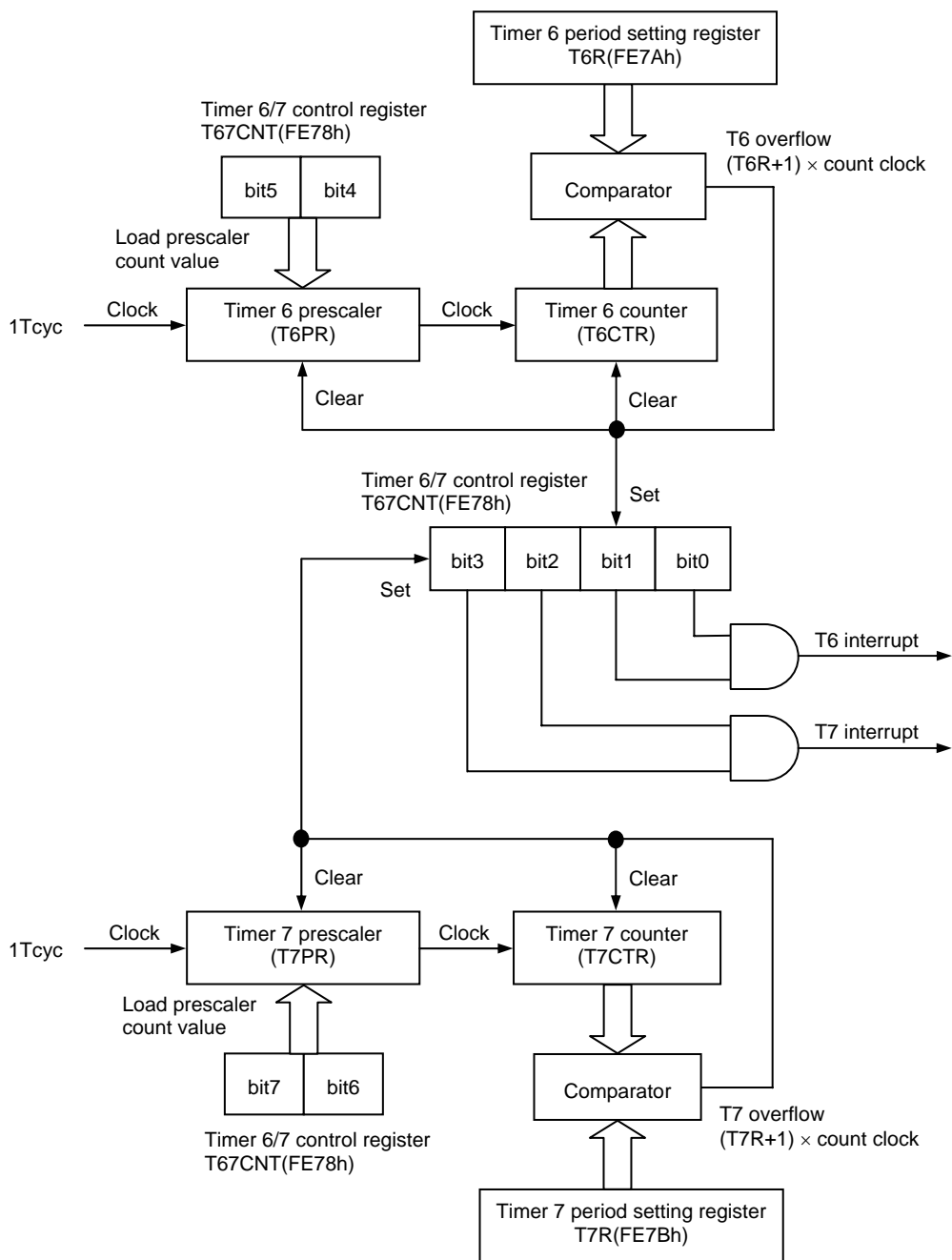
Figure 3.9.1   Timers 6/7 Block Diagram

3.9.4 Related Registers

3.9.4.1 Timer 6/7 control register (T67CNT)

1) The timer 6/7 control register is a 8-bit register that controls the operation and interrupts of T6 and T7.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE78	0000 0000	R/W	T67CNT	T7C1	T7C0	T6C1	T6C0	T7OV	T7IE	T6OV	T6IE

T7C1 (bit 7): T7 count clock control

T7C0 (bit 6): T7 count clock control

T7C1	T7C0	T7 Count Clock
0	0	Timer 7 prescaler and timer/counter are stopped in the reset state.
0	1	4 Tcyc
1	0	16 Tcyc
1	1	64 Tcyc

T6C1 (bit 5): T6 count clock control

T6C0 (bit 4): T6 count clock control

T6C1	T6C0	T6 Count Clock
0	0	Timer 6 prescaler and timer/counter are stopped in the reset state.
0	1	4 Tcyc
1	0	16 Tcyc
1	1	64 Tcyc

T7OV (bit 3): T7 overflow flag

This flag is set at the interval of timer 7's period when timer 7 is running.

This flag must be cleared with an instruction.

T7IE (bit 2): T7 interrupt request enable control

An interrupt to vector address 0043H is generated when this bit and T7OV are set to 1.

T6OV (bit 1): T6 overflow flag

This flag is set at the interval of timer 6's period when timer 6 is running.

This flag must be cleared with an instruction.

T6IE (bit 0): T6 interrupt request enable control

An interrupt to vector address 0043H is generated when this bit and T6OV are set to 1.

3.9.4.2 Timer 6 period setting register (T6R)

1) This register is an 8-bit register for defining the period of timer 6.

Timer 6 period = (T6R value + 1) × Timer 6 prescaler value (4, 16 or 64 Tcyc)

2) When data is written into T6R while timer 6 is running, both the timer 6's prescaler and counter are temporarily cleared, then restart counting.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7A	0000 0000	R/W	T6R	T6R7	T6R6	T6R5	T6R4	T6R3	T6R2	T6R1	T6R0

3.9.4.3 Timer 7 period setting register (T7R)

1) This register is an 8-bit register for defining the period of timer 7.

Timer 7 period = (T7R value + 1) × Timer 7 prescaler value (4, 16 or 64 Tcyc)

2) When data is written into T7R while timer 7 is running, both the timer 7's prescaler and counter are temporarily cleared, then restart counting.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7B	0000 0000	R/W	T7R	T7R7	T7R6	T7R5	T7R4	T7R3	T7R2	T7R1	T7R0

3.9.4.4 Port 0 function control register (P0FCR)

1) This register is a 6-bit register that controls the shared output function of port 0 pins. It controls the toggle outputs of timer 6 and timer 7.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE42	00HH 0000	R/W	P0FCR	T7OE	T6OE	-	-	CLKOEN	CKODV2	CKODV1	CKODV0

T7OE (bit 7):

This flag is used to control the timer 7 toggle output at pin P07.

This flag is disabled when pin P07 is set in the input mode.

When pin P07 is set in the output mode:

A 0 in this bit causes the value of port data latch to be presented at pin P07.

A 1 in this bit causes the OR of the value of the port data latch and the waveform which toggles at the interval equal to the timer 7 period at pin P07.

T6OE (bit 6):

This flag is used to control the timer 6 toggle output at pin P06.

This flag is disabled when pin P06 is set in the input mode.

When pin P06 is set in the output mode:

A 0 in this bit causes the value of port data latch to be presented at pin P06.

A 1 in this bit causes the OR of the value of the port data latch and the waveform which toggles at the interval equal to the timer 6 period at pin P06.

CLKOEN (bit 3):

CKODV2 (bit 2):

CKODV1 (bit 1):

CKODV0 (bit 0):

These 4 bits have nothing to do with the control functions on timers 6 and 7. See the description of port 0 for details on these bits.

3.10 Base Timer (BT)

3.10.1 Overview

The base timer (BT) incorporated in this series of microcontrollers is a 14-bit binary up-counter that provides the following five functions:

1) Clock timer
2) 14-bit binary up-counter
3) High-speed mode (when used as a 6-bit base timer)
4) Buzzer output
5) Hold mode reset

3.10.2 Functions

1) Clock timer

   The base timer can count clocks at 0.5 second intervals when a 32.768 kHz subclock is used as the count clock for the base timer. In this case, one of the three clocks, namely, cycle clock, timer/counter 0 prescaler output, and subclock must be loaded in the input signal select register (ISL) as the base timer count clock.

2) 14-bit binary up-counter

   A 14-bit binary up-counter can be constructed using an 8-bit binary up-counter and a 6-bit binary up-counter. These counters can be cleared under program control.

3) High speed mode (when used as a 6-bit base timer)

   When the base timer is used as a 6-bit timer, it can clock at intervals of approximately 2 ms if the 32.768 kHz subclock is used as the count clock. The bit length of the base timer can be specified using the base timer control register (BTCR).

4) Buzzer output function

   The base timer can generate 2kHz beeps when the 32.768 kHz subclock is used as the count clock. The buzzer output can be controlled using the input signal select register (ISL). The buzzer output can be transmitted via pin P17.

5) Interrupt generation

   An interrupt request to vector address 001BH is generated if an interrupt request is generated by the base timer when the interrupt request enable bit is set. The base timer can generate two types of interrupt requests: "base timer interrupt 0" and "base timer interrupt 1."

6) HOLD mode operation and HOLD mode resetting

   The base timer is enabled for operation in the HOLD mode when bit 2 of the power control register (PCON) is set. The HOLD mode can be reset by an interrupt from the base timer. This function allows the microcontroller to perform low-current intermittent operations.

7) To control the base timer, it is necessary to manipulate the following special function registers:
   - BTCR, ISL
   - P1, P1DDR, P1FCR

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7F	0000 0000	R/W	BTCR	BTFST	BTON	BTC11	BTC10	BTIF1	BTIE1	BTIF0	BTIE0
FE5F	0000 0000	R/W	ISL	ST0HCP	ST0LCP	BTIMC1	BTIMC0	BUZON	NFSEL	NFON	ST0IN

3.10.3   Circuit Configuration

3.10.3.1   8-bit binary up-counter

1)   This counter is an up-counter that receives, as its input, the signal selected by the input signal select register (ISL). It generates 2 kHz buzzer output and base timer interrupt 1 flag set signals. The overflow out of this counter serves as the clock to the 6-bit binary counter.

3.10.3.2   6-bit binary up counter

1)   This counter is a 6-bit up-counter that receives, as its input, the signal selected by the special function register (ISL) or the overflow signal from the 8-bit counter and generates set signals for base timer interrupts 0 and 1. The switching of the input clock is accomplished by the base timer control register (BTCR).

3.10.3.3   Base timer input clock source

1)   The clock input to the base timer (fBST) can be selected from "cycle clock," "timer 0 prescaler," and "subclock" via the input signal select register (ISL).
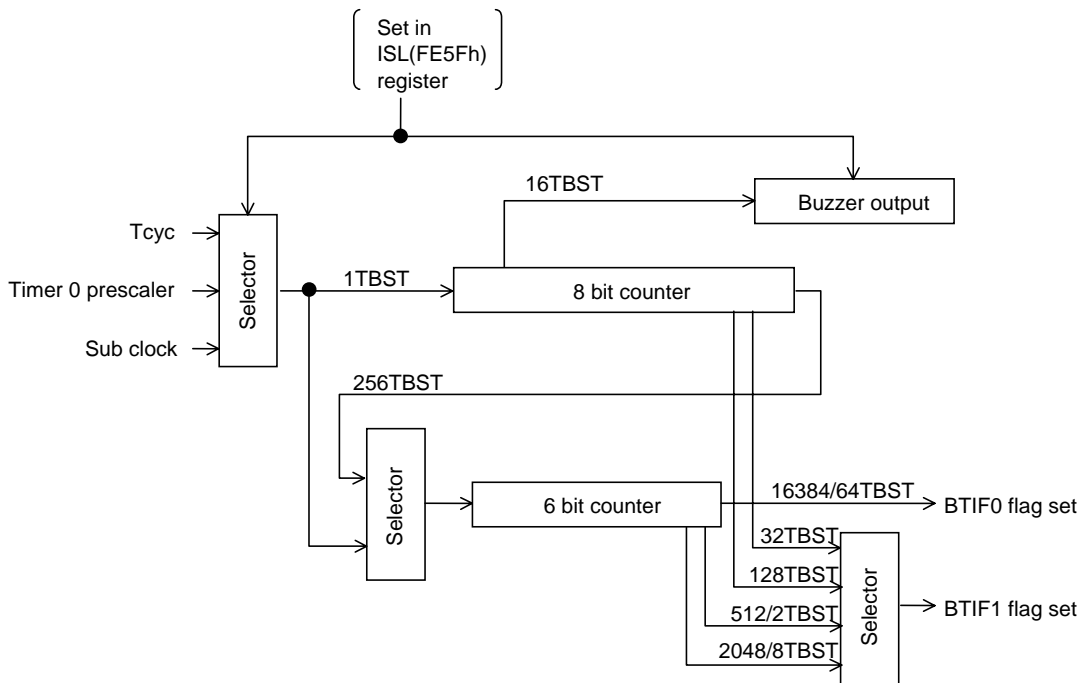
Figure 3.10.1   Base Timer Block Diagram

3.10.4   Related Registers

3.10.4.1   Base timer control register (BTCR)

1)   The base timer control register is an 8-bit register that controls the operation of the base timer.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7F	0000 0000	R/W	BTCR	BTFST	BTON	BTC11	BTC10	BTIF1	BTIE1	BTIF0	BTIE0

BTFST (bit 7): Base timer interrupt 0 period control

Used to select the interval at which base timer interrupt 0 is to occur. If this bit is set to 1, the base timer interrupt 0 flag is set when an overflow occurs in the 6-bit counter. The interval at which overflows occur is 64fBST.

If this bit is set to 0, the base timer interrupt 0 flag is set when an overflow occurs in the 14-bit counter. The interval at which overflows occur is 16384fBST.

This bit must be set to 1 when the high speed mode is to be used.

BTON (bit 6): Base timer operation control

When this bit is set to 0, the base timer stops when a count value of 0 is reached. When this bit is set to 1, the base timer continues operation.

BTC11 (bit 5): Base timer interrupt 1 period control

BTC10 (bit 4): Base timer interrupt 1 period control

BTFST	BTC11	BTC10	Base Timer Interrupt Cycle 0	Base Timer Interrupt Cycle 1
0	0	0	16384fBST	32fBST
1	0	0	64fBST	32fBST
0	0	1	16384fBST	128fBST
1	0	1	64fBST	128fBST
0	1	0	16384fBST	512fBST
0	1	1	16384fBST	2048fBST
1	1	0	64fBST	2fBST
1	1	1	64fBST	8fBST

fBST: The frequency of the input clock to the base timer that is selected through the input signal select register (ISL)..

BTIF1 (bit 3): Base timer interrupt 1 flag

This flag is set at the interval equal to the base timer interrupt 1 period that is defined by BTFST, BTC11, and BTC10.

This flag must be cleared with an instruction.

BTIE1 (bit 2): Base timer interrupt 1 request enable control

Setting this bit and BTIF1 to 1 generates "X'tal HOLD mode reset signal" and "interrupt request to vector address 001BH" conditions.

BTIF0 (bit 1): Base timer interrupt 0 flag

This flag is set at the interval equal to the base timer interrupt 0 period that is defined by BTFST, BTC11, and BTC10.

This flag must be cleared with an instruction.

BTIE0 (bit 0): Base timer interrupt 0 request enable control

Setting this bit and BTIF0 to 1 generates the "X'tal HOLD mode reset signal" and "interrupt request to vector address 001BH" conditions.

Notes:

- *Both of the system clock and base timer clock must not be selected as the subclock at the same time when BTFST = BTC10 = 1 (high speed mode).*

- *Note that BTIF1 is likely to be set to 1 when BTC11 and BTC10 are rewritten.*

- *If the hold mode is entered while running the base timer when the cycle clock or subclock is selected as the base timer clock source, the base timer is subject to the influence of unstable oscillations caused by the main clock and subclock when they are started following the resetting of the hold mode, resulting in an erroneous count from the base timer. When entering the hold mode, therefore, it is recommended that the base timer be stopped.*

- *This series of microcontrollers supports the "X'tal HOLD mode" that enables low-current intermittent operation. In this mode, only the base timer is allowed for operation.*

3.10.4.2 Input signal select register (ISL)

1) This register is an 8-bit register that controls the timer 0 input, noise filter time constant, buzzer output, and base timer clock.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5F	0000 0000	R/W	ISL	ST0HCP	ST0LCP	BTIMC1	BTIMC0	BUZON	NFSEL	NFON	ST0IN

ST0HCP (bit 7): Timer 0H capture signal input port select

ST0LCP (bit 6): Timer 0L capture signal input port select

These 2 bits have nothing to do with the control function on the base timer.

BTIMC1 (bit 5): Base timer clock select

BTIMC0 (bit 4): Base timer clock select

BTIMC1	BTIMC0	Base Timer Input Clock
0	0	Subclock
0	1	Cycle clock
1	0	Subclock
1	1	Timer/counter 0 prescaler output

BUZON (bit 3): Buzzer output/timer 1 PWMH output select

This bit enables the buzzer output ($\frac{fBST}{16}$), and selects data (Buzzer output/timer 1 PWMH) to be

sent to port P17.

When set to "1," timer 1 PWMH output becomes fixed-high, and a signal that is obtained by dividing the base timer clock by 16 is sent to port P17 as buzzer output.

When this bit is set to "0," the buzzer output becomes fixed-high, and timer 1 PWMH output is sent to port P17.

NFSEL (bit 2): Noise filter time constant select

NFON (bit 1): Noise filter time constant select

ST0IN (bit 0): Timer 0 counter clock input port select

These 3 bits have nothing to do with the control function on the base timer.

3.11 Serial Interface 0 (SIO0)

3.11.1 Overview

The serial interface SIO0 incorporated in this series of microcontrollers has the following function:

1) Synchronous 8-bit serial I/O (2- or 3-wire system, clock rates of $\frac{4}{3}$ to $\frac{512}{3}$ Tcyc)

3.11.2 Functions

1) Synchronous 8-bit serial I/O

   - Performs 2- or 3-wire synchronous serial communication. The clock may be an internal or external clock.
   - The clock rate of the internal clock is programmable within the range of $(n+1) \times \frac{2}{3}$ Tcyc (n = 1 to 255; Note: n = 0 is inhibited).

2) Interrupt generation

   An interrupt request is generated at the end of transmission when the interrupt request enable bit is set.

3) To control serial interface 0 (SIO0), it is necessary to manipulate the following special function registers.
   - SCON0, SBUF0, SBR0, SCTR0, SWCON0
   - P1, P1DDR, P1FCR

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE30	0000 0000	R/W	SCON0	FIX0	FIX0	SI0RUN	FIX0	SI0DIR	SI0OVR	SI0END	SI0IE
FE31	0000 0000	R/W	SBUF0	SBUF07	SBUF06	SBUF05	SBUF04	SBUF03	SBUF02	SBUF01	SBUF00
FE32	0000 0000	R/W	SBR0	SBRG07	SBRG06	SBRG05	SBRG04	SBRG03	SBRG02	SBRG01	SBRG00
FE33	0000 0000	R/W	SCTR0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0
FE37	0000 0000	R/W	SWCON0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0

3.11.3 Circuit Configuration

3.11.3.1 SIO0 control register (SCON0) (8-bit register)

1) The SIO0 control register controls the operation and interrupts of SIO0.

3.11.3.2 SIO0 data shift register (SBUF0) (8-bit register)

1) The SIO0 data shift register is an 8-bit shift register that performs data input and output operations at the same time.

3.11.3.3 SIO0 baudrate generator register (SBR0) (8-bit register)

1) This is an 8-bit register that defines the baudrate for SIO0 serial transmission.

2) It can generate clocks at intervals of $(n+1) \times \frac{2}{3}$ Tcyc (n = 1 to 255; Note: n = 0 is inhibited).

3.11.3.4 Continuous data bit register (SCTR0) (8-bit register)

1) This register is not available for this series of microcontrollers because they are provided with no continuous transfer capability.

3.11.3.5 Continuous data transfer control register (SWCON0) (8-bit register)

1) This register is not available for this series of microcontrollers because they are provided with no continuous transfer capability.
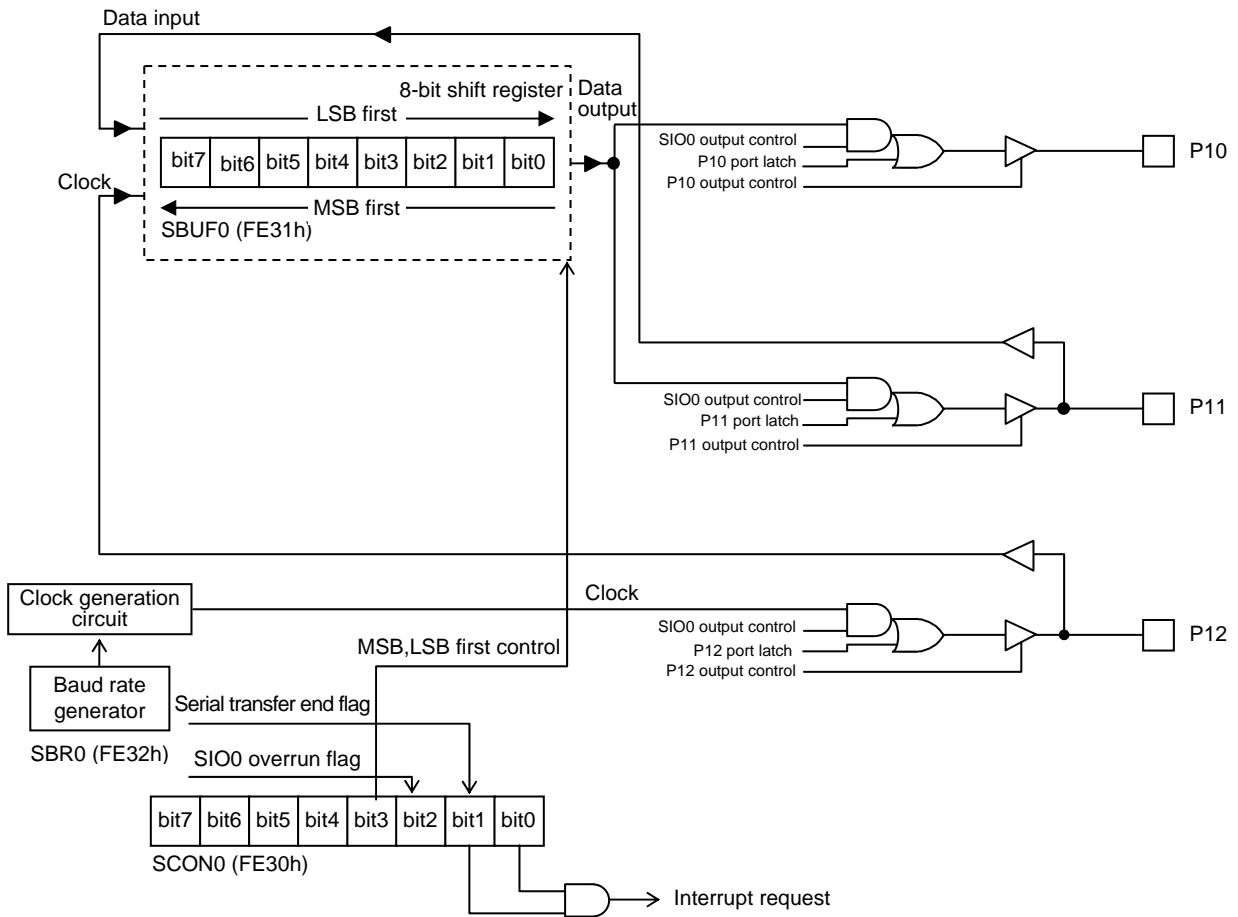
Figure 3.11.1   SIO0 Synchronous 8-bit Serial I/O Block Diagram

3.11.4 Related Registers

3.11.4.1 SIO0 control register (SCON0)

1) The SIO0 control register is an 8-bit register that controls the operation and interrupts of SIO0.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE30	0000 0000	R/W	SCON0	FIX0	FIX0	SI0RUN	SI0RUN	SI0DIR	SI0OVR	SI0END	SI0IE

SI0BNK (bit 7): Fixed bit

This bit must always be set to 0.

SI0WRT (bit 6): Fixed bit

This bit must always be set to 0.

SI0RUN (bit 5): SIO0 operation flag

1) A 1 in this bit indicates that SIO0 is running.

2) This bit must be set with an instruction.

3) This bit is automatically cleared at the end of serial transmission (on the rising edge of the last clock involved in the transfer).

SI0CTR (bit 4): Fixed bit

This bit must always be set to 0.

SI0DIR (bit 3): MSB/LSB first select

1) A 1 in this bit places SIO0 into the MSB first mode.

2) A 0 in this bit places SIO0 into the LSB first mode.

SI0OVR (bit 2): SIO0 overrun flag

1) This bit is set when a falling edge of the input clock is detected with SI0RUN=0.

2) Read this bit and judge if the communication is performed normally at the end of the communication.

3) This bit must be cleared with an instruction.

SI0END (bit 1): End of serial transmission flag

1) This bit is set at the end of serial transmission (on the rising edge of the last clock involved in the transfer).

2) This bit must be cleared with an instruction.

SI0IE (bit 0): SI00 interrupt request enable control

1) When this bit and SI0END are set to 1, an interrupt request to vector address 0033H is generated.

3.11.4.2 SIO0 data shift register (SBUF0)

1) SIO0 data shift register is an 8-bit shift register for serial transmission.

2) Data to be transmitted/received is written to and read from this shift register directly.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE31	0000 0000	R/W	SBUF0	SBUF07	SBUF06	SBUF05	SBUF04	SBUF03	SBUF02	SBUF01	SBUF00

3.11.4.3   Baudrate generator register (SBR0)

1)   The baudrate generator register is an 8-bit register that defines the baudrate of SIO0.

2)   The baudrate is computed as follows:

$$TSBR0 = (SBR0\ value + 1) \times \frac{2}{3}\ Tcyc$$

SBR0 can takes a value from 1 to 255 and the valid value range of TSBR0 is from $\frac{4}{3}$ to $\frac{512}{3}$ Tcyc.

* The SBR0 value of 00[H] is disallowed.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE32	0000 0000	R/W	SBR0	SBRG07	SBRG06	SBRG05	SBRG04	SBRG03	SBRG02	SBRG01	SBRG00

3.11.4.4   Fixed bit

This bit must always be set to 0.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE33	0000 0000	R/W	SCTR0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0

3.11.4.5   Fixed bit

This bit must always be set to 0.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE37	0000 0000	R/W	SWCON0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0

3.11.5   SIO0 Transmission Examples

3.11.5.1   Synchronous 8-bit mode

1)   Setting the clock
   - Set up SBR0 when using an internal clock.

2)   Setting the transmission mode
   - Set as follows:
     SI0DIR = ?, SI0IE = 1

3)   Setting up the ports

	P12
Internal clock	Output
External clock	Input

	P10	**P11**
Data transmission only	Output	–
Data reception only	–	Input
Data transmission/reception (3-wire)	Output	Input
Data transmission/reception (2-wire)	–	N-channel open drain output

3-59

4) Setting up output data
   - Write the output data into SBUF0 in the data transmission or data transmission/reception mode.

5) Starting operation
   - Set SI0RUN.

6) Reading data (after an interrupt)
   - Read SBUF0 (SBUF0 has been loaded with serial data from the data I/O port even in the transmission mode).
   - Clear SI0END.
   - Return to step 4) when repeating transmission/reception processing.

3.11.6   SIO0 HALT Mode Operation

3.11.6.1   Synchronous 8-bit mode

1) SIO0's synchronous 8-bit mode processing is enabled in the HALT mode.

2) The HALT mode can be reset by an interrupt that is generated during SIO0 synchronous 8-bit mode processing.

3.12   Serial Interface 1 (SIO1)

3.12.1   Overview

The serial interface SIO1 incorporated in this series of microcontrollers provides the following four functions:

1)   Mode 0: Synchronous 8-bit serial I/O (2- or 3-wire system, clock rates of 2 to 512 Tcyc)
2)   Mode 1: Asynchronous serial I/O (Half-duplex, 8 data bits, 1 stop bit, baud rates of 8 to 2048 Tcyc)
3)   Mode 2: Bus-master (start bit, 8 data bits, transfer clock of 2 to 512 Tcyc)
4)   Mode 3: Bus-slave (start detection, 8 data bits, stop detection)

3.12.2   Functions

1)   Mode 0: Synchronous 8-bit serial I/O

- Performs 2- or 3-wire synchronous serial communication. The clock may be an internal or external clock.
- The clock rate of the internal clock is programmable within the range of 2 to 512 Tcyc.

2)   Mode 1: Asynchronous serial (UART)

- Performs half-duplex, 8 data bits/1 stop bit asynchronous serial communication.
- The baudrate is programmable within the range of 8 to 2048 Tcyc.

3)   Mode 2: Bus-master

- SIO1 is used as a bus master controller.
- The start conditions are automatically generated but the stop conditions must be generated by manipulating ports.
- Clock synchronization is used. Since it is possible to verify the transfer-time bus data at the end of transfer, this mode can be combined with mode 3 to provide support for multi-master configurations.
- The period of the output clock is programmable within the range of 2 to 512 Tcyc.

4)   Mode 3: Bus-slave

- SIO1 is used as a slave device of the bus.
- Start/stop condition detection processing is performed but the detection of an address match condition and the generation of an acknowledge require program intervention.
- SIO1 can generate an interrupt after automatically placing the clock line at the low level on the falling edge of the eighth clock for recognition by a program.

5)   Interrupt generation

An interrupt request is generated at the end of communication if the interrupt request enable flag is set.

6)   To control serial interface 1 (SIO1), it is necessary to control the following special function registers.
- SCON1, SBUF1, SBR1
- P1, P1DDR, P1FCR

Address	Initial Value	R/W	Name	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE34	0000 0000	R/W	SCON1	-	SI1M1	SI1M0	SI1RUN	SI1REC	SI1DIR	SI1OVR	SI1END	SI1IE
FE35	00000 0000	R/W	SBUF1	SBUF18	SBUF17	SBUF16	SBUF15	SBUF14	SBUF13	SBUF12	SBUF11	SBUF10
FE36	0000 0000	R/W	SBR1	-	SBRG17	SBRG16	SBRG15	SBRG14	SBRG13	SBRG12	SBRG11	SBRG10

3.12.3 Circuit Configuration

3.12.3.1 SIO1 control register (SCON1)   (8-bit register)
1) The SIO1 control register controls the operation and interrupts of SIO1.

3.12.3.2 SIO1 shift register (SIOSF1)   (8-bit shift register)
1) This register is a shift register used to transfer and receive SIO1 data.
2) This register cannot be accessed with an instruction. It is accessed via SBUF1.

3.12.3.3 SIO1 data register (SBUF1)   (9-bit register)
1) The lower-order 8 bits of SBUF1 are transferred to SIOSF1 at the beginning of data transfer.
2) At the end of data transfer, the contents of SIOSF1 are placed in the lower-order 8 bits of SBUF1. In modes 1, 2, and 3, since the 9th input data is placed in bit 8 of SBUF1, it is possible to check for a stop bit.

3.12.3.4 SIO1 baudrate generator register (SBR1)   (8-bit reload counter)
1) This is a reload counter for generating internal clocks.
2) The generator can generate clocks of 2 to 512 Tcyc in modes 0 and 2 and clocks of 8 to 2048 Tcyc in mode 1.

Table 3.12.1   SIO1 Operations and Operating Modes

		Synchronous (Mode 0)		UART (Mode 1)		Bus Master (Mode 2)		Bus Slave (Mode 3)	
		Transmit SI1REC=0	Receive SI1REC=1	Transmit SI1REC=0	Receive SI1REC=1	Transmit SI1REC=0	Receive SI1REC=1	Transmit SI1REC=0	Receive SI1REC=1
Start bit		None	None	Output (Low)	Input (Low)	See 1 and 2 below	Not required	Not required	Note 2
Data output		8 (Shift data)	8 (All 1s)	8 (Shift data)	8 (All 1s)	8 (Shift data)	8 (All 1s)	8 (Shift data)	8 (All 1s)
Data input		8 (Input pin)	←	8 (Input pin)	←	8 (Input pin)	←	8 (Input pin)	←
Stop bit		None	←	Output (High)	Input (H/L)	Input (H/L)	Output (SBUF1 bit8)	Input (H/L)	Output (L)
Clock		8	←	9 (Internal)	←	9	←	Low output on falling edge of 8th clock	←
Operation start		SI1RUN ↑	←	1) SI1RUN ↑ 2) Start bit detected	Start bit detected	1)No start bit on falling edge of SI1END when SI1RUN=1 2) With start bit on rising edge of SI1RUN when SI1END=0	1) On left side	1) On right side	1)Clock released on falling edge of SI1END when SI1RUN=1 2) Start bit detected when SI1RUN=0 and SI1END=0
Period		2 to 512 Tcyc	←	8 to 2048 Tcyc	←	2 to 512Tcyc	←	2 to 512Tcyc	←
SI1RUN (bit 5)	Set	Instruction	←	1)Instruction 2) Start bit detected	Start bit detected	Instruction	Already set	Already set	Start bit detected
	Clear	End of processing	←	End of stop bit	←	1) Stop condition detected 2) When arbitration lost (Note 1)	←	1) Stop condition detected 2) Ack=1 detected	←
SI1END (bit 1)	Set	End of processing	←	End of stop bit	←	1) Rising edge of 9th clock 2) Stop condition detect	←	1) Falling edge of 8th clock 2) Stop condition detect	←
	Clear	Instruction	←	Instruction	←	Instruction	←	Instruction	←

(Continued on next page)

SIO1

Table 3.12.1   SIO1 Operations and Operating Modes (cont.)

		Synchronous (Mode 0)		UART (Mode 1)		Bus Master (Mode 2)		Bus Slave (Mode 3)	
		Transmit SI1REC=0	Receive SI1REC=1	Transmit SI1REC=0	Receive SI1REC=1	Transmit SI1REC=0	Receive SI1REC=1	Transmit SI1REC=0	Receive SI1REC=1
SI1OVR (bit 2)	Set	1) Falling edge of clock detected when SI1RUN=0	←	1) Falling edge of clock detected when SI1RUN=0 2) SI1END set conditions met when SI1END=1	←	1) Falling edge of clock detected when SI1RUN=0 2) SI1END set conditions met when SI1END=1	←	1) Falling edge of clock detected when SI1RUN=0 2) SI1END set conditions met when SI1END=1 3) Start bit detected	←
	Clear	Instruction	←	Instruction	←	Instruction	←	Instruction	←
Shifter data update		SBUF1→ Shifter at beginning of operation	←	SBUF1→ Shifter at beginning of operation	←	SBUF1→ Shifter at beginning of operation	←	SBUF1→ Shifter at beginning of operation	←
Shifter→ SBUF1 (bits 0 to 7)		Rising edge of 8th clock	←	When 8 bit data transferred	When 8-bit data received	Rising edge of 8th clock	←	Rising edge of 8th clock	←
Automatic update of SBUF1 bit 8		None	←	Input data read in on stop bit	←	Input data read in on rising edge of 9th clock	←	Input data read in on rising edge of 9th clock	←

Note 1: If internal data output state="H" and data port state= "L" conditions are detected at the rising edges of the first to 8th clocks, the microcontroller recognizes a bus contention loss and clears SI1RUN (and also stops the generation of the clock immediately).
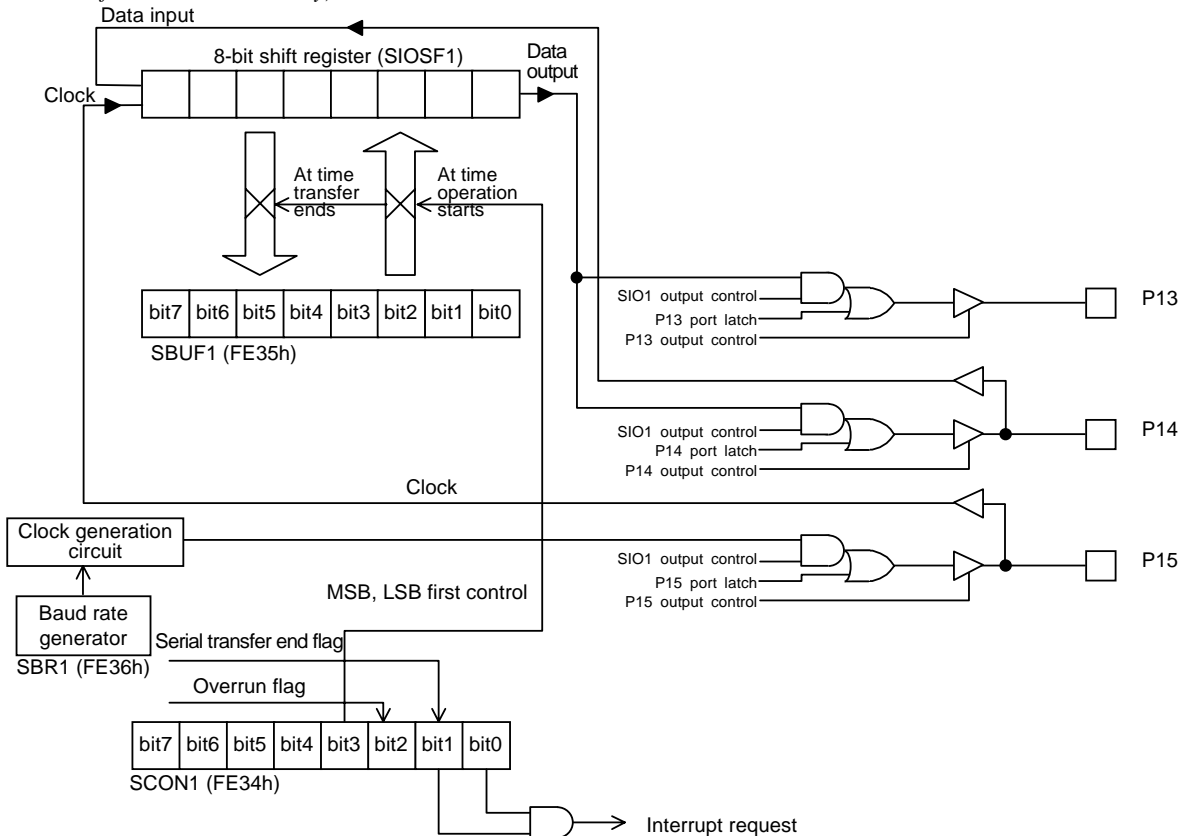
Figure 3.12.1   SIO1 Mode 0: Synchronous 8-bit Serial I/O Block Diagram (SI1M1=0, SI1M0=0)

3-64
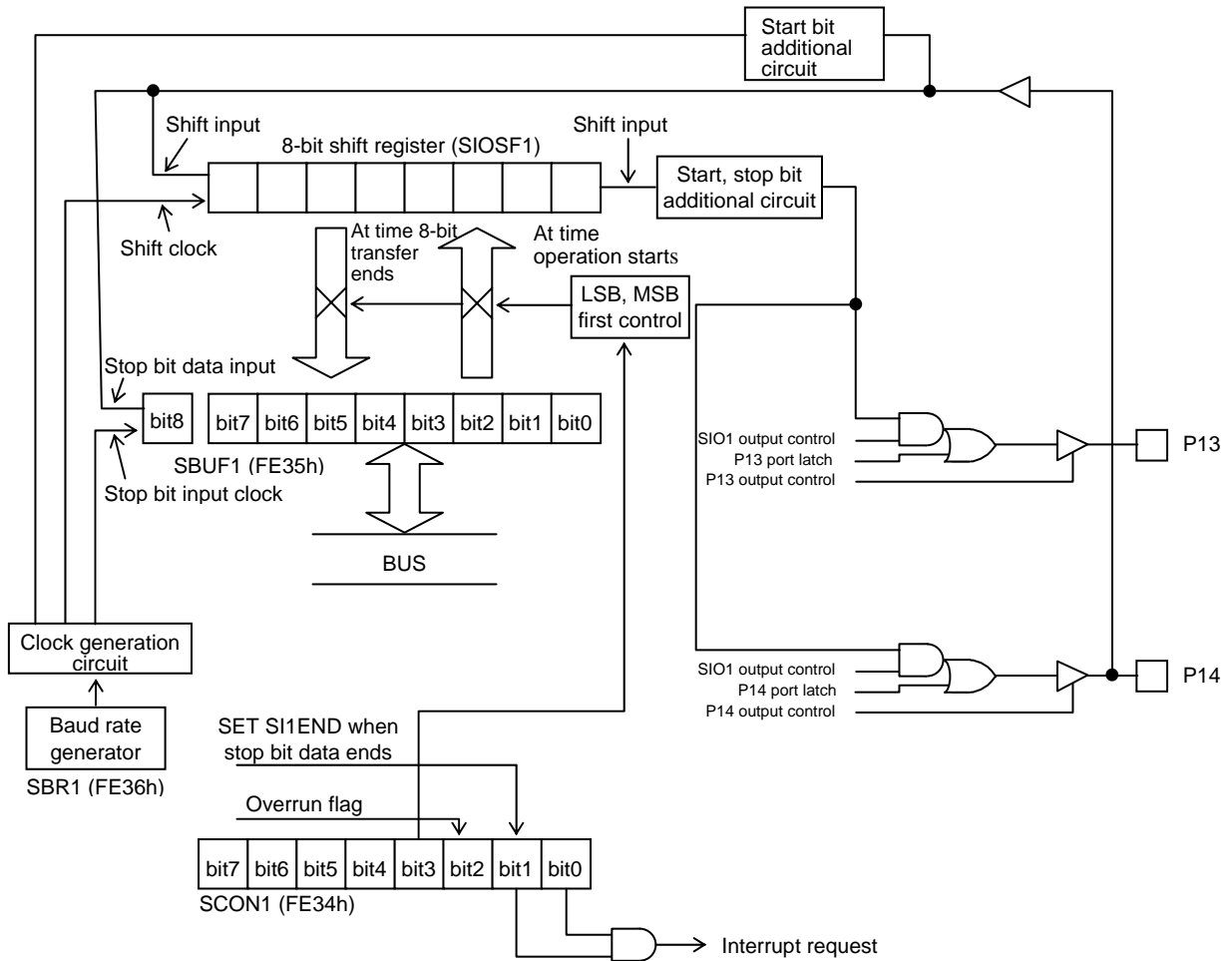
Figure 3.12.2   SIO1 Mode 1: Asynchronous Serial [UART] Block Diagram (SI1M1=0, SI1M0=1)

SIO1

3.12.4 SIO1 Transmission Examples

3.12.4.1 Synchronous serial transmission (mode 0)

1) Setting the clock
   - Set up SBR1 when using an internal clock.
2) Setting the transmission mode
   - Set as follows:
     SI1M0=0, SI1M1=0, SI1DIR, SI1IE=1
3) Setting up the ports and SI1REC (BIT4)

	P15
Internal clock	Output
External clock	Input

	P13	**P14**	**SI1REC**
Data transmission only	Output	–	0
Data reception only	–	Input	1
Data transmission/reception (3-wire)	Output	Input	0
Data transmission/reception (2-wire)	–	N-channel open drain output	0

4) Setting up output data
   - Write output data into SBUF1 in the data transmission mode (SI1REC=0).
5) Starting operation
   - Set SI1RUN.
6) Reading data (after an interrupt)
   - Read SBUF1 (SBUF1 has been loaded with serial data from the data I/O port even in the transmission mode).
   - Clear SI1END and exit interrupt processing.
   - Return to step 4) when repeating processing.

3.12.4.2 Asynchronous serial transmission (Mode 1)

1) Setting the baudrate
   - Set up SBR1.
2) Setting the transmission mode
   - Set as follows:
     SI1M0=1, SI1M1=0, SI1DIR, SI1IE=1
3) Setting up the ports.

	P13	**P14**
Data transmission/reception (2-wire)	Output	Input
Data transmission/reception (1-wire)	–	N-channel open drain output

4) Starting transmission
   • Set SI1REC to 0 and write output data into SBUF1.
   • Set SI1RUN.

   Note: Use the SIO1 data I/O port(P14) when using the SIO1 transmission only in mode 1.

   In mode 1, transmission is automatically started when a falling edge of receive data is detected. While mode 1 is on, the falling edge of data is always sensed at the data I/O port (P14). Consequently, if the transmit port is assigned to the data output port (P13), it is likely that data transmissions are started unexpectedly according to the changes in the state of P14.

5) Starting receive operation
   • Set SI1REC to 1. (Once SI1REC is set to 1, do not attempt to write data to the SCON1 register until the SI1END flag is set.)
   • Detect the falling edge of receive data.

6) Reading data (after an interrupt)
   • Read SBUF1. (SBUF1 has been loaded with serial data read from the data I/O port even in the transmission mode. When SBUF1 is read in, the data about the position of the stop bit is read into bit 1 of the PSW.)
   • Clear SI1END and exit interrupt processing.
   • Return to step 4) when repeating processing.

   Note: Make sure that the following conditions are met when performing continuous mode reception processing with SIO1 in mode 1 (UART):
   • *The number of stop bits is set to 2 or greater.*
   • *Clearing of SI1END during interrupt processing terminates before the next start bit arrives.*

3.12.4.3 Bus-master mode (mode 2)
1) Setting the clock
   • Set up SBR1.
2) Setting the mode.
   • Set as follows:
     SI1M0 = 0, SI1M1 = 1, SI1DIR, SI1IE = 1, SI1REC = 0
3) Setting up the ports
   • Designate the clock and data ports as N-channel open drain output ports.
4) Starting communication (sending an address)
   • Load SBUF1 with address data.
   • Set SI1RUN (transfer a start bit + SBUF1 (8 bits) + stop bit (H)).
5) Checking for address data (after an interrupt)
   • Read SBUF1. (SBUF1 has been loaded with serial data from the data I/O port even in the transmission mode. When SBUF1 is read in, the data about the position of the stop bit is read into bit 1 of the PSW.)
   • Check for an acknowledge by reading bit 1 of the PSW.
   • Check that the data read from SBUF1 matches the sent data. A mismatch implies that the current transmission and another master operation overlap.
6) Sending data
   • Load SBUF1 with output data.
   • Clear SI1END and exit interrupt processing (transfer SBUF1 (8 bits) + stop bit (H)).

7) Checking sent data (after an interrupt)
   - Read SBUF1. (SBUF1 has been loaded with serial data from the data I/O port even in the transmission mode. When SBUF1 is read in, the data about the position of the stop bit is read into bit 1 of the PSW.)
   - Check for an acknowledge by reading bit 1 of the PSW.
   - Check that the data read from SBUF1 matches the sent data. A mismatch implies that the current transmission and another master operation overlap.
   - Return to step 6) when continuing data transmission.
   - Go to step 10) to terminate communication.

8) Receiving data
   - Set SI1REC to 1.
   - Clear SI1END and exit interrupt processing (receive (8 bits) + SBUF1 bit 8 (acknowledge) output).

9) Reading received data (after an interrupt)
   - Read SBUF1.
   - Return to step 8) to continue reception of data.
   - Go to * in step 10) to terminate processing. At this moment, SBUF1 bit 8 data has already been presented as acknowledge data and the clock for the master side has been released.

10) Terminating communication
   - Manipulate the clock output port (P15FCR = 0, P15DDRb = 1, P15 = 0) and set the clock output to 0.
   - Manipulate the data output port (P14FCR = 0, P14DDR = 1, P14 = 0) and set the data output to 0.
   - Restore the clock output port into the original state (P15FCR = 1, P15DDR = 1, P15 = 0) and release the clock output.
   * - Wait for all slaves to release the clock and the clock to be set to 1.
   - Allow for a data setup time, then manipulate the data output port (P14FCR = 0, P14DDR = 1, P14 = 1) and set the data output to 1. In this case, the SIO1 overrun flag (SI1OVR:FE34, bit 2) is set but this will exert no influence on the operation of SIO1.
   - Restore the data output port into the original state (set P14FCR to 1, then P14DDR to 1 and P14 to 0).
   - Clear SI1END and SI1OVR, then exit interrupt processing.
   - Return to step 4) to repeat processing.

3.12.4.4 Bus-slave mode (mode 3)

1) Setting the clock
   - Set up SBR1 (to set the acknowledge data setup time).

2) Setting the transmission mode
   - Set as follows:
     SI1M0 = 1, SI1M1 = 1, SI1DIR, SI1IE = 1, SI1REC = 0

3) Setting up ports
   - Designate the clock and data ports as N-channel open drain output ports.

4) Starting communication (waiting for an address)

*1 • Set SI1REC.

*2 • SI1RUN is automatically set on detection of a start bit.

• Perform receive processing (8 bits) and set the clock output to 0 on the falling edge of the 8th clock, which generates an interrupt.

5) Checking address data (after an interrupt)

• Detecting a start condition sets SI1OVR. Check SI1RUN=1 and SI1OVR=1 to determine if the address has been received.

(SI1OVR is not automatically cleared. Clear it by instruction.)

• Read SBUF1 and check the address.

• If no address match occurs, clear SI1RUN and SI1END and exit interrupt processing, then wait for a stop condition detection at * of step 8).

6) Receiving data

* • Clear SI1END and exit interrupt processing. (If a receive sequence has been performed, send an acknowledge and release the clock port after the lapse of (SBR1 value + 1) × Tcyc.)

• When a stop condition is detected, SI1RUN is automatically cleared and an interrupt is generated. Then, clear SI1END to exit interrupt processing and return to *2 in step 4).

• Perform a receive operation (8 bits), then set the clock output to 0 on the falling edge of the 8th clock, after which an interrupt occurs. The clock counter will be cleared if a start condition is detected in the middle of receive processing. In such a case, another 8 clocks are required to generate an interrupt.

• Read SBUF1 and store the read data.

Note: *Bit 8 of SBUF1 is not yet updated because the rising edge of 9th clock has not yet occurred.*

• Return to * in step 6) to continue receive processing.

7) Sending data

• Clear SI1REC.

• Load SBUF1 with output data.

• Clear SI1END and exit interrupt processing. (Send an acknowledge for the preceding reception operation and release the clock port after the lapse of (SBR1 value + 1) × Tcyc.)

*1 • Perform a send operation (8 bits) and set the clock output to 0 on the falling edge of the 8th clock, after which an interrupt occurs.

*2 • Go to *3 in step 7) if SI1RUN is set to 1.

• If SI1RUN is set to 0, implying an interrupt from *4 in step 7), clear SI1END and SI1OVR and return to *1 in step 4).

*3 • Read SBUF1 and check send data as required.

Note: *Bit 8 of SBUF1 is not yet updated because the rising edge of 9th clock has not yet occurred.*

• Load SBUF1 with the next output data.

• Clear SI1END and exit interrupt processing. (Release the clock port after the lapse of (SBR1 value + 1) × Tcyc).

• Return to *1 in step7) if an acknowledge from the master is present (L).

• If there is no acknowledge presented from the master (H), SIO1, recognizing the end of data transmission, automatically clears SI1RUN and release the data port. However, in a case that restart condition comes just after the event, SI1REC must be set to "1" before exiting the interrupt (SI1REC is for detecting a start condition and is not set automatically). It may disturb the transmission of address from the master if there is an unexpected restart just after slave's transmission (when SI1REC is not set by instruction).

*4 • When a stop condition is detected, an interrupt is generated and processing returns to *2 in step 7).

8) Terminating communication
   - Set SI1REC.
   - Return to * in step 6) to cause communication to automatically terminate.
   - To force communication to termination, clear SI1RUN and SI1END (release the clock port).
   * • An interrupt occurs when a stop condition is detected. Then, clear SI1END and SI1OVR and return to *2 in step 4).

3.12.5   Related Registers

3.12.5.1   SIO1 control register (SCON1)

1) The SIO1 control register is an 8-bit register that controls the operation and interrupts of SIO1.

Address	Initial Value	R/W	Name	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE34	0000 0000	R/W	SCON1	-	SI1M1	SI1M0	SI1RUN	SI1REC	SI1DIR	SI1OVR	SI1END	SI1IE

SI1M1 (bit 7): SIO1 mode control

SI1M0 (bit 6): SIO1 mode control

Table 3.12.2   SIO1 Operation Modes

Mode	SI1M1	SI1M0	Operating Mode
0	0	0	Synchronous 8-bit SIO
1	0	1	UART (1 stop bit, no parity)
2	1	0	Bus master mode
3	1	1	Bus slave mode

SI1RUN (bit 5): SIO1 operation flag
1) A 1 in this bit indicates that SIO1 is running.
2) See Table 3.12.1 for the conditions for setting and clearing this bit.

SI1REC (bit 4): SIO1 receive/send control
1) Setting this bit to 1 places SIO1 into the receive mode.
2) Setting this bit to 0 places SIO1 into the send mode.

SI1DIR (bit 3): MSB/LSB first select
1) Setting this bit to 1 places SIO1 into the MSB first mode.
2) Setting this bit to 0 places SIO1 into the LSB first mode.

SI1OVR (bit 2): SIO1 overrun flag
1) This bit is set when a falling edge of the input clock is detected with SI1RUN = 0.
2) In modes 1, 2, and 3, this bit is set if the conditions for setting SI1END are established when SI1END = 1.
3) In mode 3 this bit is set when the start condition is detected.
4) This bit must be cleared with an instruction.

SI1END (bit 1): End of serial transmission flag
1) This bit is set when serial transmission terminates (see Table 3.12.1).
2) This bit must be cleared with an instruction.

SI1IE (bit 0): SIO1 interrupt request enable control

When this bit and SI1END are set to 1, an interrupt request to vector address 003BH is generated.

3.12.5.2 Serial buffer 1 (SBUF1)

1) Serial buffer 1 is a 9-bit register used to store data to be handled during SIO1 serial transmission.

2) The lower-order 8 bits of SBUF1 are transferred to the data shift register for data transmission/reception at the beginning of transmission processing and the contents of the shift register are placed in the lower-order 8 bits of SBUF1 when 8-bit data is transferred.

3) In modes 1, 2, and 3, bit 8 of SBUF1 is loaded with the 9th data bit that is received (data about the position of the stop bit).

Address	Initial Value	R/W	Name	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE35	00000 0000	R/W	SBUF1	SBUF18	SBUF17	SBUF16	SBUF15	SBUF14	SBUF13	SBUF12	SBUF11	SBUF10

3.12.5.3 Baudrate generator register (SBR1)

1) The baudrate generator register is an 8-bit register that defines the baudrate of SIO1.

2) Loading this register with data causes the baudrate generating counter to be initialized immediately.

3) The baudrate varies from mode to mode (the baudrate generator is disabled in mode 3).

Modes 0 and 2: $TSBR1 = (SBR1\ value + 1) \times 2\ Tcyc$
(Value range = 2 to 512 Tcyc)

Mode 1: $TSBR1 = (SBR1\ value + 1) \times 8Tcyc$
(Value range = 8 to 2048Tcyc)

Address	Initial Value	R/W	Name	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE36	0000 0000	R/W	SBR1	-	SBRG17	SBRG16	SBRG15	SBRG14	SBRG13	SBRG12	SBRG11	SBRG10

3.13 Asynchronous Serial Interface 1 (UART1)

3.13.1 Overview

This series of microcontrollers incorporates an asynchronous serial interface 1 (UART1) that has the following characteristics and features:

1) Data length: 7, 8, and 9 bits (LSB first)

2) Stop bits: 1 bit (2 bits in continuous communication mode)

3) Parity bits: None

4) Clock rate: Programmable within the range of $(\frac{16}{3}$ to $\frac{2048}{3})$ Tcyc or $(\frac{64}{3}$ to $\frac{8192}{3})$ Tcyc

5) Full duplex communication

The independent transmitter and receiver blocks allow both transmit and receive operations to be performed at the same time. Both transmitter and receiver blocks adopt a double buffer configuration, so that data can be transmitted and received continuously.

3.13.2 Functions

1) Asynchronous serial (UART1)

- Performs full duplex asynchronous serial communication using a data length of 7, 8, or 9 bits with 1 stop bit.

- The clock rate of the UART1 is programmable within the range of $(\frac{16}{3}$ to $\frac{2048}{3})$ Tcyc or $(\frac{64}{3}$ to $\frac{8192}{3})$ Tcyc.

2) Continuous data transmission/reception

- Performs continuous transmission of serial data whose data length and clock rate are fixed (the data length and clock rate that are identified at the beginning of transmission are used).

- The number of stop bits used in the continuous transmission mode is 2 (see Figure 3.13.4).

- Performs continuous reception of serial data whose data length and clock rate vary on each receive operation.

- The clock rate of the UART1 is programmable within the range of $(\frac{16}{3}$ to $\frac{2048}{3})$ Tcyc or $(\frac{64}{3}$ to $\frac{8192}{3})$ Tcyc.

- The transmit data is read from the transmit data register (TBUF) and the received data is stored in the receive data register (RBUF).

3) Interrupt generation

Interrupt requests are generated at the beginning of each transmission and at the end of each reception if the interrupt request enable bit is set.

4) To control the asynchronous serial interface 1 (UART1), it is necessary to manipulate the following special function registers:

- UCON0, UCON1, UBR, TBUF, RBUF P2, P2DDR

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FED0	0000 0000	R/W	UCON0	UBRSEL	STRDET	RECRUN	STPERR	U0B3	RBIT8	RECEND	RECIE
FED1	0000 0000	R/W	UCON1	TRUN	8/9BIT	TDDR	TCMOS	8/7BIT	TBIT8	TEPTY	TRNSIE
FED2	0000 0000	R/W	UBR	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0
FED3	0000 0000	R/W	TBUF	T1BUF7	T1BUF6	T1BUF5	T1BUR4	T1BUF3	T1BUF2	T1BUF1	T1BUF0
FED4	0000 0000	R/W	RBUF	R1BUF7	R1BUF6	R1BUF5	R1BUR4	R1BUF3	R1BUF2	R1BUF1	R1BUF0

3.13.3   Circuit Configuration

3.13.3.1   UART1 control register 0 (UCON0)   (8-bit register)
1)   The UART1 control register 0 controls the receive operation and interrupts of the UART1.

3.13.3.2   UART1 control register 1 (UCON1)   (8-bit register)
1)   The UART1 control register 1 controls the transmit operation, data length, and interrupts of the UART1.

3.13.3.3   UART1 baudrate generator (UBR)   (8-bit reload counter)
1)   The UART1 baudrate generator is a reload counter for generating internal clocks.
2)   It can generate clocks at intervals of $(n+1) \times \frac{8}{3}$ Tcyc or $(n+1) \times \frac{32}{3}$ Tcyc (n = 1 to 255; Note: $n = 0$ is inhibited).

3.13.3.4   UART1 transmit data register (TBUF)   (8-bit register)
1)   The UART1 transmit data register is an 8-bit register for storing the data to be transmitted.

3.13.3.5   UART1 transmit shift register (TSFT)   (11-bit shift register)
1)   The UART1 transmit shift register is used to send transmit data via UART1.
2)   This register cannot be accessed directly with an instruction. It must be accessed through the transmit data register (TBUF).

3.13.3.6   UART1 receive data register (RBUF)   (8-bit register)
1)   The UART1 receive data register is an 8-bit register for storing received data.

3.13.3.7   UART1 receive shift register (RSFT)   (11-bit shift register)
1)   The UART1 receive shift register is used to receive serial data via UART1.
2)   This register cannot be accessed directly with an instruction. It must be accessed through the receive data register (RBUF).

UART1
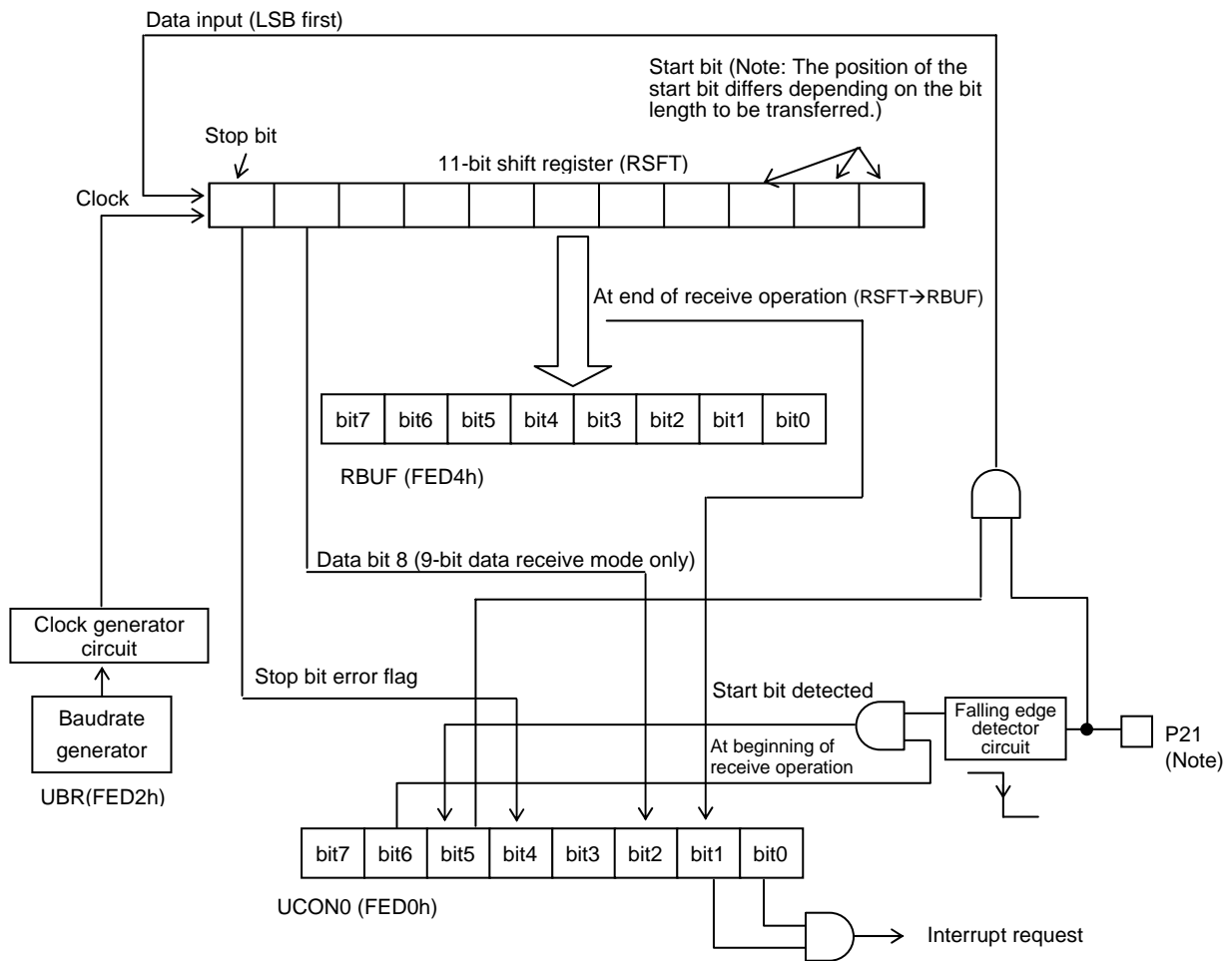
Note: Bit 1 of P2DDR (at FE49) must be set to 0 when the UART1 is to be used in the receive mode (the UART1 will not function normally if bit 1 is set to 1).

Figure 3.13.1   UART1 Block Diagram (Receive Mode)

Stop bit (Note: The position of
the stop bit differs depending on
the bit length to be transferred.)

Start bit

Clock

11-bit shift register (TSFT)

"H" | "H" bit8 | "H" bit 7 | | | | | | | "L"

Data output (LSB first)

End of transfer to shift register

At beginning of transmit
operation (TBUF→TSFT)

bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0

TBUF(FED3h)

Clock generator
circuit

Baudrate
generator

UBR (FED2h)

Data bit 8
(When transfer data
bit length is 9 bits).

UART1 output control

UART1 output format control

P20
(Note)

bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0

UCON1 (FED1h)

Bit 7 of UCON0 (FED0h)

Interrupt request

*Note: Bit 0 of P2DDR (at FE49) must be set to 0 when the UART1 is to be used in the transmit mode
(the UART1 will not function normally if bit 0 is set to 1).*

Figure 3.13.2   UART1 Block Diagram (Transmit Mode)

3.13.4   Related Registers

3.13.4.1   UART1 control register 0 (UCON0)

1) The UART1 control register 0 is an 8-bit register that controls the receive operation and interrupts of UART1.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FED0	0000 0000	R/W	UCON0	UBRSEL	STRDET	RECRUN	STPERR	U0B3	RBIT8	RECEND	RECIE

UBRSEL (bit 7): UART1 baudrate generator period control

1) When this bit is set to 1, the UART1 baudrate generator generates 'clocks at a frequency of $(n+1)\times(\frac{32}{3})$ Tcyc

2) When this bit is set to 0, the UART1 baudrate generator generates 'clocks at a frequency of $(n+1)\times(\frac{8}{3})$ Tcyc.

* n represents the value that is defined in the UBR (FED2) for the UART baudrate generator.

STRDET (bit 6): UART1 start bit detection control

1) When this bit is set to 1, the start bit detection (falling edge detection) function is enabled.

2) When this bit is set to 0, the start bit detection (falling edge detection) function is disabled.

  * This bit must be set to 1 to enable the start bit detection function when the UART1 is to be used in the continuous receive mode.

  * If this bit is set to 1 when the receive port (P21) is held at the low level, RECRUN is automatically set to start UART receive processing.

RECRUN (bit 5): UART1 start of receive operation flag

1) This bit is set and a receive operation starts if a falling edge of the signal at receive port (P21) is detected when the start bit detection function is enabled (STRDET = 1).

2) This bit is automatically cleared at the end of the receive operation (clearing this bit during a receive operation will abort the receive operation).

  * When a receive operation is forced to terminate prematurely, RECEND is set to 1 and the contents of the receive shift register are transferred to RBUF. STPERR is set to 1 if the state of the last data bit that is received on the forced termination is low.

STPERR (bit 4): UART1 stop bit error flag

1) This bit is set at the end of a receive operation if the state of the received stop bit (the last data bit received) is low.

2) This bit must be cleared with an instruction.

U0B3 (bit 3): General-purpose flag

1) This bit can be used as a general-purpose flag bit. Any attempt to manipulate this bit exerts no influence on the operation of this functional block.

RBIT8 (bit 2): UART1 receive data bit 8 storage bit

1) This bit position is loaded with bit 8 of the received data when the data length is set to 9 bits (UCON1: 8/9BIT=1, 8/7BIT=0). (If the receive operation is terminated prematurely, this bit position is loaded with the last received bit but one.)

2) This bit must be cleared with an instruction.

RECEND (bit 1): End of UART1 reception flag
1)   This bit is set at the end of a receive operation (When this bit is set, the received data is transferred from the receive shift register (RSFT) to the receive data register (RBUF).

2)   This bit must be cleared with an instruction.

*   In the continuous receive mode, the next receive operation is not carried out even when the UART1 detects such data as sets the start of receive operation flag (RECRUN) before this bit is set.

RECIE (bit 0): UART1 receive interrupt request enable control
1)   When this bit and RECEND are set to 1, an interrupt request to vector address 0033H is generated.

3.13.4.2 UART1 control register 1 (UCON1)
1)   The UART1 control register 1 is an 8-bit register that controls the transmission processing, data length, and interrupts of and for UART1.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FED1	0000 0000	R/W	UCON1	TRUN	8/9BIT	TDDR	TCMOS	8/7BIT	TBIT8	TEPTY	TRNSIE

TRUN (bit 7): UART1 transmission control
1)   When this bit is set to 1, the UART1 starts a transmit operation.

2)   This bit is automatically cleared at the end of the transmit operation. (If this bit is cleared in the middle of a transmit operation, the operation is aborted immediately.)

*   In the continuous transmission mode, this bit is cleared at the end of a transmit operation but is automatically set within the same cycle (Tcyc). Consequently, transmit operations occur with intervening 1-Tcyc waits.

*   In the continuous transmission mode, TRUN will not be set automatically if a bit-manipulation-instruction (NOT1, CLR1, or SET1) is executed on the UCON1 register in the same cycle in which TRUN is to be automatically cleared..

8/9BIT (bit6): UART1 transmit data length control
1)   This bit and 8/7BIT (bit 3) are used to control the length of data to be transferred through the UART.

8/9 BIT	8/7 BIT	Data Length (in bits)
1	0	9
0	0	8
0	1	7
1	1	Inhibited

*   The UART1 will not run normally if the data length is changed in the middle of a transmit operation. Be sure to manipulate this bit after confirming the completion of a transmit operation.

*   The same data length is used when both transmission and receive operations are to be performed at the same time.

TDDR (bit 5): UART1 transmit port output control
1)   When this bit is set to 1, the transmit data is placed at the transmit port (P20). No transmit data is generated if bit 0 of P2DDR (FE49) is set to 1.

2)   When this bit is set to 0, no transmit data is placed at the transmit port (P20).

*   The transmit port is placed in the "HIGH/open (CMOS/N-channel open drain)" mode if this bit is set to 1 when the UART1 has stopped a transmit operation (TRUN = 0).

*   This bit must always be set to 0 when the UART1 transmission function is not to be used.

TCMOS (bit 4): UART1 transmit port output type control

1) When this bit is set to 1, the output type of the transmit port (P20) is set to "CMOS."

2) When this bit is set to 0, the output type of the transmit port (P20) is set to "N-channel open drain."

8/7BIT (bit3): UART1 transmit data length control

1) This bit and 9/8BIT (bit 6) are used to control the length of data to be transferred through the UART.

TBIT8 (bit 2): UART1 transmit data bit 8 storage bit

1) This bit carries bit 8 of the transmit data when the data length is set to 9 bits (8/9BIT = 1, 8/7BIT=0).

TEPTY (bit 1): UART1 transmit shift register transfer flag

1) This bit is set when the data transfer from the transmit data register (TBUF) to the transmit shift register (TSFT) ends at the beginning of a transmit operation. (This bit is set in the cycle (Tcyc) following the one in which the transmit control bit (TRUN) is set to 1.)

2) This bit must be cleared with an instruction.

* When performing continuous mode transmission processing, make sure that this bit is set before each loading of the next transmit data into the transmit data register (TBUF). When this bit is subsequently cleared, the transmit control bit (TRUN) is automatically set at the end of the transmit operation.

TRNSIE (bit 0): UART1 transmit interrupt request enable/disable control

1) An interrupt request to vector address 003BH is generated when this bit and TEPTY are set to 1.

3.13.4.3   UART1 baudrate generator (UBR)

1) The UART1 baudrate generator is an 8-bit register that defines the baudrate of the UART1.

2) The counter for the baudrate generator is initialized when a UART1 serial transmit operation is stopped or terminated (UCON0: RECRUN=0, UCON1: TRUN=0).

* Do not change the baudrate in the middle of UART1 serial transmission processing. The UART1 will not function normally if the baudrate is changed during UART1 serial transmission processing. Always make sure to terminate the UART1 processing before changing the baudrate.

* The same baudrate is used when both transmit and receive operations are to be performed at the same time (this holds also true when the transmit and receive operations are to be performed in the continuous transmission mode).

* When (UCON0:UBRSEL=0)

$$\text{TUBR} = (\text{UBR value} + 1) \times \frac{8}{3} \text{ Tcyc (value range: } \frac{16}{3} \text{ to } \frac{2048}{3} \text{ Tcyc)}$$

* When (UCON0:UBRSEL=1)

$$\text{TUBR} = (\text{UBR value} + 1) \times \frac{32}{3} \text{ Tcyc (value range: } \frac{64}{3} \text{ to } \frac{8192}{3} \text{ Tcyc)}$$

* Setting UBR to 00[H] is inhibited.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FED2	0000 0000	R/W	UBR	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0

3.13.4.4   UART1 transmit data register (TBUF)

1) The UART1 transmit data register is an 8-bit register that stores the data to be transmitted through the UART1.

2) Data from the TBUF is transferred to the transmit shift register (TSFT) at the beginning of a transmit operation. (Load the next data after checking the transmit shift register transfer flag (UCON1:TEPTY).)

* Bit 8 of the transmit data must be loaded into the transmit data bit 8 storage bit (UCON1:TBIT8).

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FED3	0000 0000	R/W	TBUF	T1BUF7	T1BUF6	T1BUF5	T1BUF4	T1BUF3	T1BUF2	T1BUF1	T1BUF0

3.13.4.5   UART1 receive data register (RBUF)

1) The UART1 receive data register is an 8-bit register that stores the data that is received through the UART1.

2) The data from the receive shift register (RSFT) is transferred to this RBUF at the end of a receive operation.

* Bit 8 of the received data is placed in the receive data b it 8 storage bit (UCON0:RBIT8).

* Bit 7 of RBUF is loaded with 0 if the receive data length is set to 7.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FED4	0000 0000	R/W	RBUF	R1BUF7	R1BUF6	R1BUF5	R1BUF4	R1BUF3	R1BUF2	R1BUF1	R1BUF0

3.13.5   UART1 Continuous Communication Processing Examples

3.13.5.1   Continuous 8-bit data receive mode (first received data = 55H)
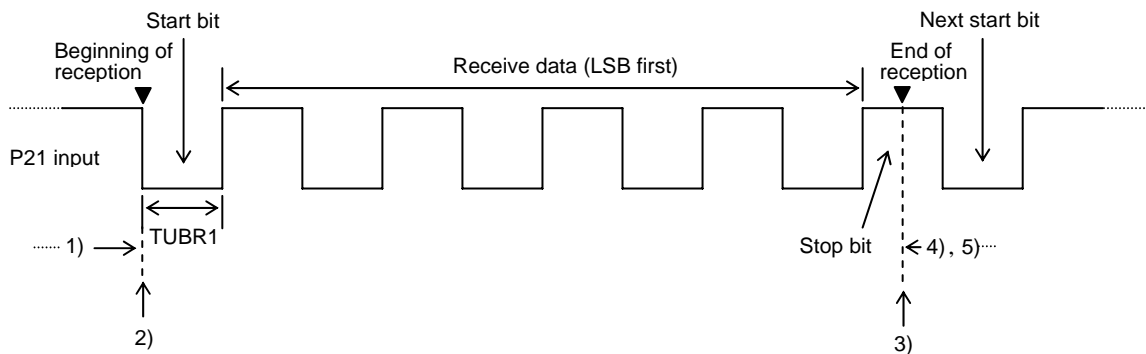
Figure 3.13.3   Example of Continuous 8-bit Data Reception Mode Processing

1) Setting the clock
   - Set the baudrate (UBR).

   Setting the data length mode
   - Clear UCON1:8/9BIT and 8/7BIT.

   Configuring the UART1 for receive processing and setting up the receive port and receive interrupts
   - Set up the receive control register (UCON0 = 41H).
   * Set P21DDR (P2DDR:BIT1) to 0 and P21 (P2:BIT1) to 0.

2) Starting a receive operation
   - UCON0:RECRUN is set when a falling edge of the signal at the receive port (P21) is detected.

3) End of receive operation
   - When the receive operation ends, UCON0:RECRUN is automatically cleared and UCON0:RECEND is set. The UART1 then waits for the start bit of the next received data.

4) Receive interrupt processing
   - Read the received data (RBUF).
   - Clear UCON0:RECEND and STPERR and exit the interrupt processing routine.
   * When changing the data length and baudrate for the next receive operation, do so before the start bit (falling edge of the signal) is detected at the receive port (P21).

5) Next receive data processing
   - Subsequently, repeat steps 2), 3), and 4) shown above.
   - To terminate continuous mode receive processing, clear UCON0:STRDET during a receive operation, and this receive operation will be the last receive operation that the UART1 executes.

3.13.5.2   Continuous 8-bit data transmit mode (first transmit data = 55H)
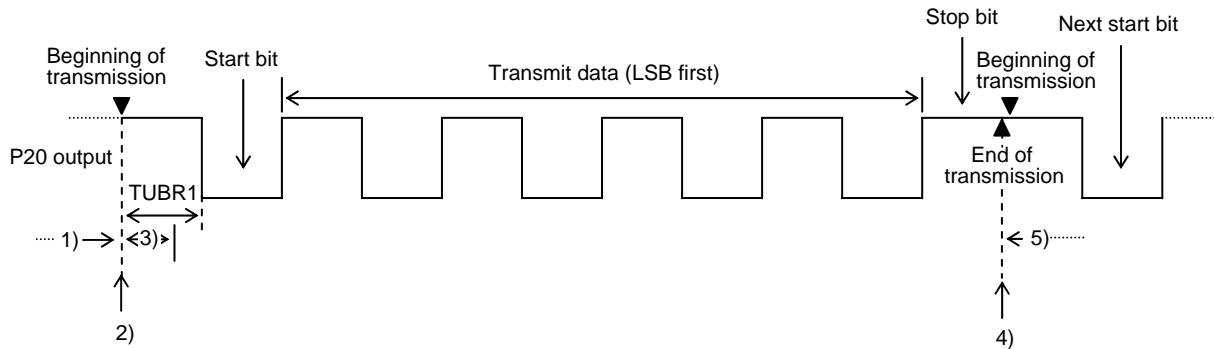
Figure 3.13.4   Example of Continuous 8-bit Data Transmit Mode Processing

1)   Setting the clock
    • Set the baudrate (UBR).
   Setting up transmit data
    • Load the transmit data (TBUF =55H).
   Setting the data length, transmit port, and interrupts
    • Set up the transmit control register (UCON1 = 31H).
    * Set P20DDR (P2DDR:BIT0) to 0 and P20 (P2:BIT0) to 0.

2)   Starting a transmit operation
    • Set UCON1:TRUN.

3)   Transmit interrupt processing
    • Load the next transmit data (TBUF = xxH).
    • Clear UCON1:TEPTY and exit the interrupt processing routine.

4)   End of transmit operation
    • When the transmit operation ends, UCON1:TRUN is automatically cleared and automatically set in the same cycle (Tcyc) (at the continuous data transmission mode only; this processing takes 1 Tcyc of time). The UART1 then starts the transmission of the next transmit data.

5)   Next transmit data processing
    • Subsequently, repeat steps 3) and 4) shown above.
    • To terminate continuous mode transmit processing, clear UCON1:TRNSIE while not clearing UCON1:TEPTY and exit the interrupt in the step 3) processing, and the transmit operation that is being performed at that time will be the last transmit operation that the UART1 executes.

3.13.5.3 Setting Up the UART1 communications ports

(1)   When using port 2 as the UART1 port

1)   Setting up the receive port (P21)

Register Data		Receive Port (P21) State	Internal Pull-up Resistor
P21	**P21DDR**		
0	0	Input	Off
1	0	Input	On

* The UART1 can receive no data normally if P21DDR is set to 1.

2)   Setting up the transmit port (P20)

Register Data				Transmit Port (P20) State	Internal Pull-up Resistor
P20	**P20DDR**	**TDDR**	**TCMOS**		
0	0	1	1	CMOS output	Off
0	0	1	0	N-channel open drain output	Off
1	0	1	0	N-channel open drain output	On

* The UART1 transmits no data if P20DDR is set to 1.

3.13.6   UART1 HALT Mode Operation

3.13.6.1 Receive mode

1)   UART1's receive mode processing is enabled in the HALT mode. (If UCON0:STRDET is set to 1 when the microcontroller enters the HALT mode, receive processing will be restarted if data such that UCON0:RECRUN is set at the end of a receive operation.)

2)   The HALT mode can be reset using the UART1 receive interrupt.

3.13.6.2 Transmit mode

1)   UART1's transmit mode processing is enabled in the HALT mode. (If the continuous transmission mode is specified when the microcontroller enters the HALT mode, the UART1 will restart transmission processing after terminating a transmit operation. Since UCON1:TEPTY cannot be cleared in this case, the UART1 stops processing after completing that transmit operation.)

2)   The HALT mode can be reset using the UART1 transmit interrupt.

3.14 PWM4 and PWM5

3.14.1 Overview

This series of microcontrollers incorporates two 12-bit PWMs, named PWM4 and PWM5. Each PWM is made up of a PWM generator circuit that generates multi frequency 8-bit fundamental PWM waves and a 4-bit additional pulse generator.

3.14.2 Functions

1) PWM4: Fundamental wave PWM mode (register PWM4L=0)
   - Fundamental wave period = $\frac{(16\,\text{to}\,256)}{3}$ Tcyc (programmable in ($\frac{16}{3}$)Tcyc increments, common to PWM5)
   - High-level pulse width = 0 to (Fundamental wave period – $\frac{1}{3}$)Tcyc (programmable in ($\frac{1}{3}$)Tcyc increments)

2) PWM4: Fundamental wave + Additional pulse PWM mode
   - Fundamental wave period = $\frac{(16\,\text{to}\,256)}{3}$ Tcyc (programmable in ($\frac{16}{3}$)Tcyc increments, common to PWM5)
   - Overall period = Fundamental wave period × 16
   - High-level pulse width = 0 to (Overall period – $\frac{1}{3}$)Tcyc (programmable in ($\frac{1}{3}$)Tcyc increments)

3) PWM5: Fundamental wave PWM mode (register PWM5L=0)
   - Fundamental wave period = $\frac{(16\,\text{to}\,256)}{3}$ Tcyc (programmable in ($\frac{16}{3}$)Tcyc increments, common to PWM4)
   - High-level pulse width = 0 to (Fundamental wave period – $\frac{1}{3}$)Tcyc (programmable in ($\frac{1}{3}$)Tcyc increments)

4) PWM5: Fundamental wave + Additional pulse PWM mode
   - Fundamental wave period = $\frac{(16\,\text{to}\,256)}{3}$ Tcyc (programmable in ($\frac{16}{3}$)Tcyc increments, common to PWM4)
   - Overall period = Fundamental wave period × 16
   - High-level pulse width = 0 to (Overall period – $\frac{1}{3}$)Tcyc (programmable in ($\frac{1}{3}$)Tcyc increments)

5) Interrupt generation
   - Interrupt requests are generated at the intervals equal to the overall PWM period if the interrupt request enable bit is set.

6) To control PWM4 and PWM5, it is necessary to manipulate the following special function registers:
   - ・ PWM4L, PWM4H, PWM5L, PWM5H, PWM4C
   - ・ P3, P3DDR

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE72	0000 HHHH	R/W	PWM4L	PWM4L3	PWM4L2	PWM4L1	PWM4L0	-	-	-	-
FE73	0000 0000	R/W	PWM4H	PWM4H7	PWM4H6	PWM4H5	PWM4H4	PWM4H3	PWM4H2	PWM4H1	PWM4H0
FE74	0000 HHHH	R/W	PWM5L	PWM5L3	PWM5L2	PWM5L1	PWM5L0	-	-	-	-
FE75	0000 0000	R/W	PWM5H	PWM5H7	PWM5H6	PWM5H5	PWM5H4	PWM5H3	PWM5H2	PWM5H1	PWM5H0
FE76	0000 0000	R/W	PWM4C	PWM4C7	PWM4C6	PWM4C5	PWM4C4	ENPWM5	ENPWM4	PWM4OV	PWM4IE

3.14.3 Circuit Configuration

3.14.3.1 PWM4/PWM5 control register (PWM4C)   (8-bit register)
1)   The PWM4/PWM5 control register controls the operation and interrupts of PWM4 and PWM5.

3.14.3.2 PWM4 compare register L (PWM4L)   (4-bit register)
1)   The PWM4 compare register L controls the additional pulses of PWM4.
2)   PWM4L is assigned bits 7 to 4 and all of its lower-order 4 bits are apparently set to 1 when it is read.

3.14.3.3 PWM4 compare register H (PWM4H)   (8-bit register)
1)   The PWM4 compare register H controls the fundamental pulse width of PWM4.
2)   When bits 7 to 4 of PWM4L are all fixed at 0, PWM4 can serve as period-programmable 8-bit PWM that is controlled by PWM4H.

3.14.3.4 PWM5 compare register L (PWM5L)   (4-bit register)
1)   The PWM5 compare register L controls the additional pulses of PWM5.
2)   PWM5L is assigned bits 7 to 4 and all of its lower-order 4 bits are apparently set to 1 when it is read.

3.14.3.5 PWM5 compare register H (PWM5H)   (8-bit register)
1)   The PWM5 compare register H controls the fundamental pulse width of PWM5.
2)   When bits 7 to 4 of PWM5L are all fixed at 0, PWM5 can serve as period-programmable 8-bit PWM that is controlled by PWM5H.
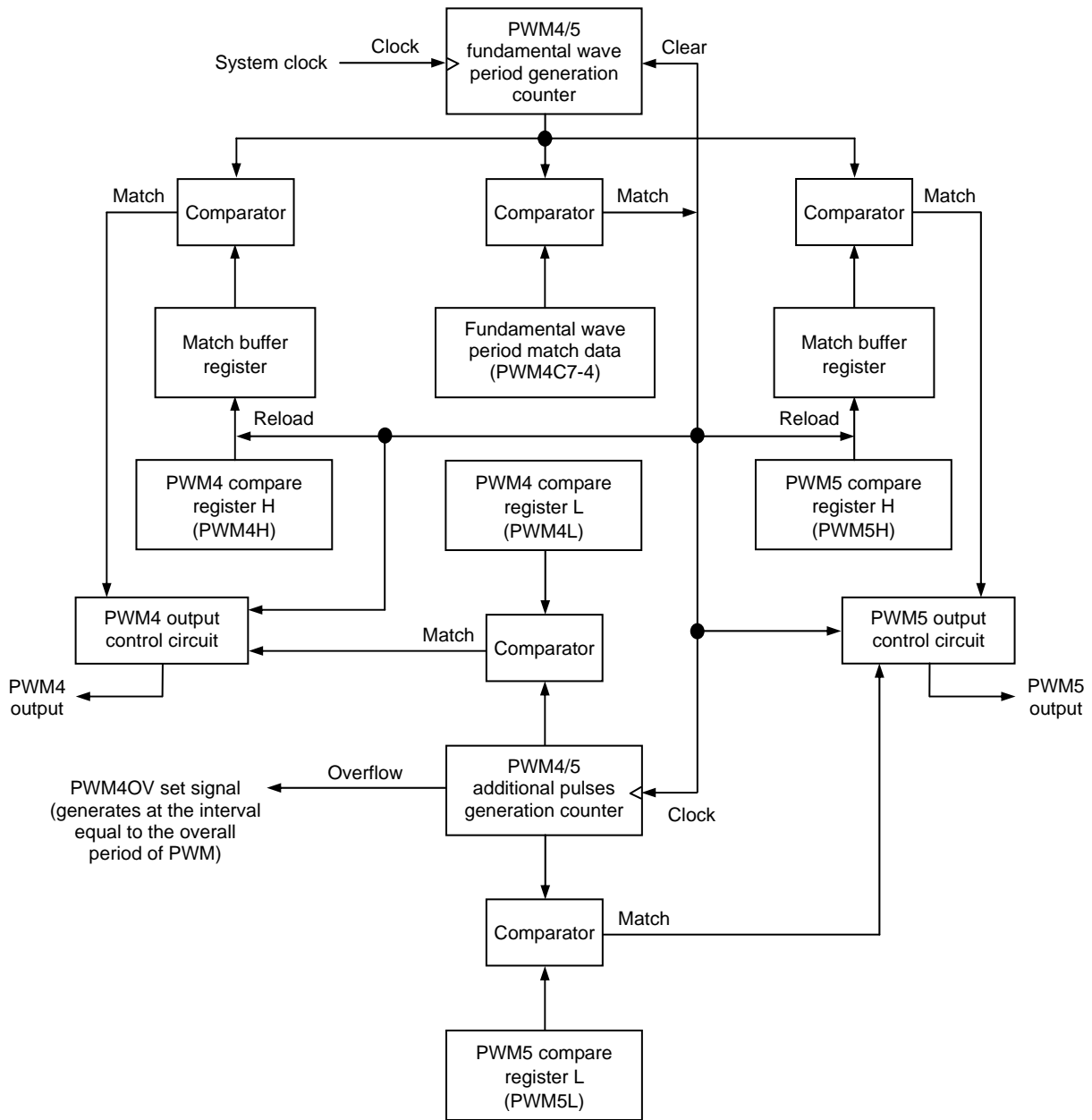
Figure 3.14.1   PWM4 and PWM5 Block Diagram

3.14.4   Related Registers

3.14.4.1   PWM4/PWM5 control register (PWM4C)   (8-bit register)

1)   The PWM4/PWM5 control register controls the operation and interrupts of PWM4 and PWM5.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE76	0000 0000	R/W	PWM4C	PWM4C7	PWM4C6	PWM4C5	PWM4C4	ENPWM5	ENPWM4	PWM4OV	PWM4IE

PWM4C7 to PWM4C4 (bits 7 to 4): PWM4/PWM5 period control

- Fundamental wave period   = (Value represented by (PWM4C7 to PWM4C4) + 1) $\times (\frac{16}{3})$Tcyc
- Overall period   = Fundamental wave period $\times$ 16

ENPWM5 (bit 3): PWM5 operation control

- When this bit is set to 1, the PWM5 is activated.
- When this bit is set to 0, the PWM5 is deactivated.

ENPWM4 (bit 2): PWM4 operation control

- When this bit is set to 1, the PWM4 is activated.
- When this bit is set to 0, the PWM4 is deactivated.

PWM4OV (bit 1): PWM4/PWM5 overflow flag

- This bit is set at the interval equal to the overall period of PWM.
- This flag must be cleared with an instruction.

PWM4IE (bit 0): PWM4/PWM5 interrupt request enable control

An interrupt to vector addresses 004BH is generated when this bit and PWM4OV are both set to 1.

3.14.4.2   PWM4 compare register L (PWM4L)   (4-bit register)

1)   The PWM4 compare register L controls the additional pulses of PWM4.

2)   PWM4L is assigned bits 7 to 4 and all of its lower-order 4 bits are apparently set to 1 when it is read.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE72	0000 HHHH	R/W	PWM4L	PWM4L3	PWM4L2	PWM4L1	PWM4L0	-	-	-	-

3.14.4.3   PWM4 compare register H (PWM4H)   (8-bit register)

1)   The PWM4 compare register H controls the fundamental pulse width of PWM4.

Fundamental pulse width = (Value represented by PWM4H7 to PWM4H0) $\times (\frac{1}{3})$Tcyc

2)   When bits 7 to 4 of PWM4L are all fixed at 0, PWM4 can serve as period-programmable 8-bit PWM that is controlled by PWM4H.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE73	0000 0000	R/W	PWM4H	PWM4H7	PWM4H6	PWM4H5	PWM4H4	PWM4H3	PWM4H2	PWM4H1	PWM4H0

3.14.4.4 PWM5 compare register L (PWM5L) (4-bit register)

1) The PWM5 compare register L controls the additional pulses of PWM5.

2) PWM5L is assigned bits 7 to 4 and all of its lower-order 4 bits are apparently set to 1 when it is read.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE74	0000 HHHH	R/W	PWM5L	PWM5L3	PWM5L2	PWM5L1	PWM5L0	-	-	-	-

3.14.4.5 PWM5 compare register H (PWM5H) (8-bit register)

1) The PWM5 compare register H controls the fundamental pulse width of PWM5.

Fundamental pulse width = (Value represented by PWM5H7 to PWM5H0) $\times (\frac{1}{3})$Tcyc

2) When bits 7 to 4 of PWM5L are all fixed at 0, PWM5 can serve as period-programmable 8-bit PWM that is controlled by PWM5H.

Address	Initial Value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE75	0000 0000	R/W	PWM5H	PWM5H7	PWM5H6	PWM5H5	PWM5H4	PWM5H3	PWM5H2	PWM5H1	PWM5H0
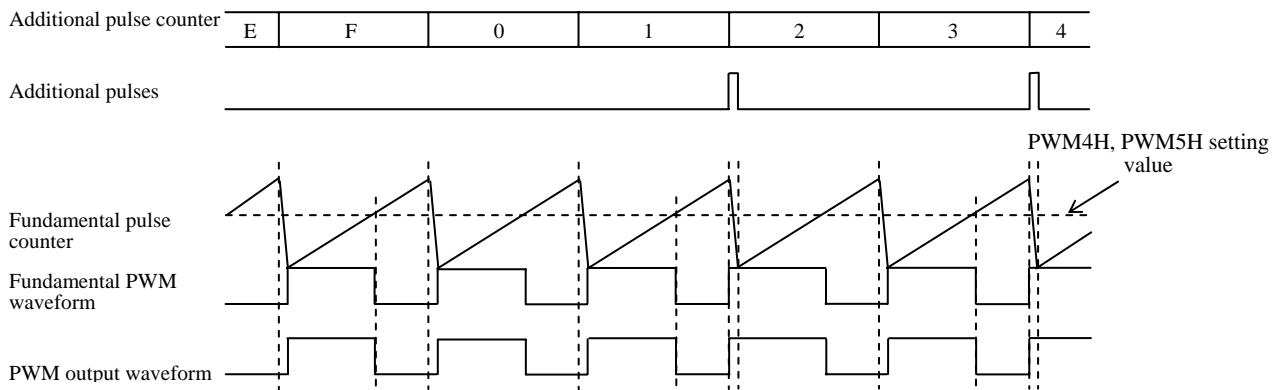
3.14.5 Setting Up the PWM4 and PWM5 Output Ports

1) The P30 settings and conditions for generating PWM4 outputs are summarized below.

Register Data			P30 State
P30	P30DDR	ENPWM4	
0	1	0	LOW
0	1	1	PWM4 output data
1	1	0	HIGH/Open (CMOS/N-channel open drain)
1	1	1	HIGH/Open (CMOS/N-channel open drain)

2) The P31 settings and conditions for generating PWM5 outputs are summarized below.

Register Data			P31 State
P31	P31DDR	ENPWM5	
0	1	0	LOW
0	1	1	PWM5 output data
1	1	0	HIGH/Open (CMOS/N-channel open drain)
1	1	1	HIGH/Open (CMOS/N-channel open drain)

PWM

● The 12-bit PWM has the following waveform structure:

- The overall period consists of 16 fundamental wave periods.
- A fundamental wave period is represented by an 8-bit PWM. (PWM compare register H)(PWMH)
- 4 bits are used to designate the fundamental wave period to which additional pulses are to be added.   (PWM compare register L)(PWML)

  12-bit register structure   →   (PWMH), (PWML) = XXXX XXXX, XXXX (12BIT)

● How pulses are added to the fundamental wave periods (Example 1)

- PWM compare register H (PWMH)   =   00   [H]
- PWM compare register L (PWML)   =   0 to F [H]
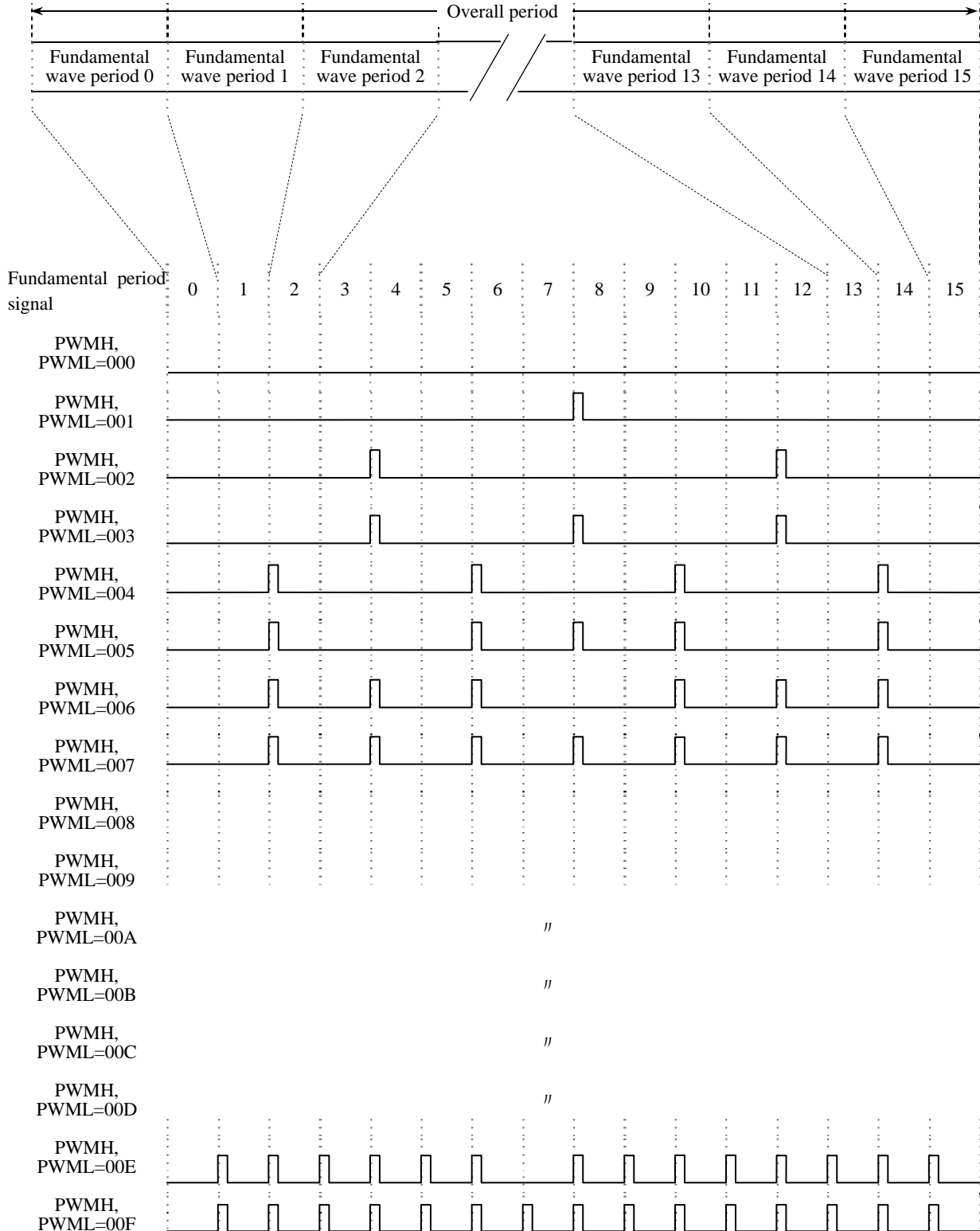
● How pulses are added to fundamental wave periods (Example 2)

  • PWM compare register H (PWMH) = 01 [H]
  • PWM compare register L (PWML) = 0 to F [H]
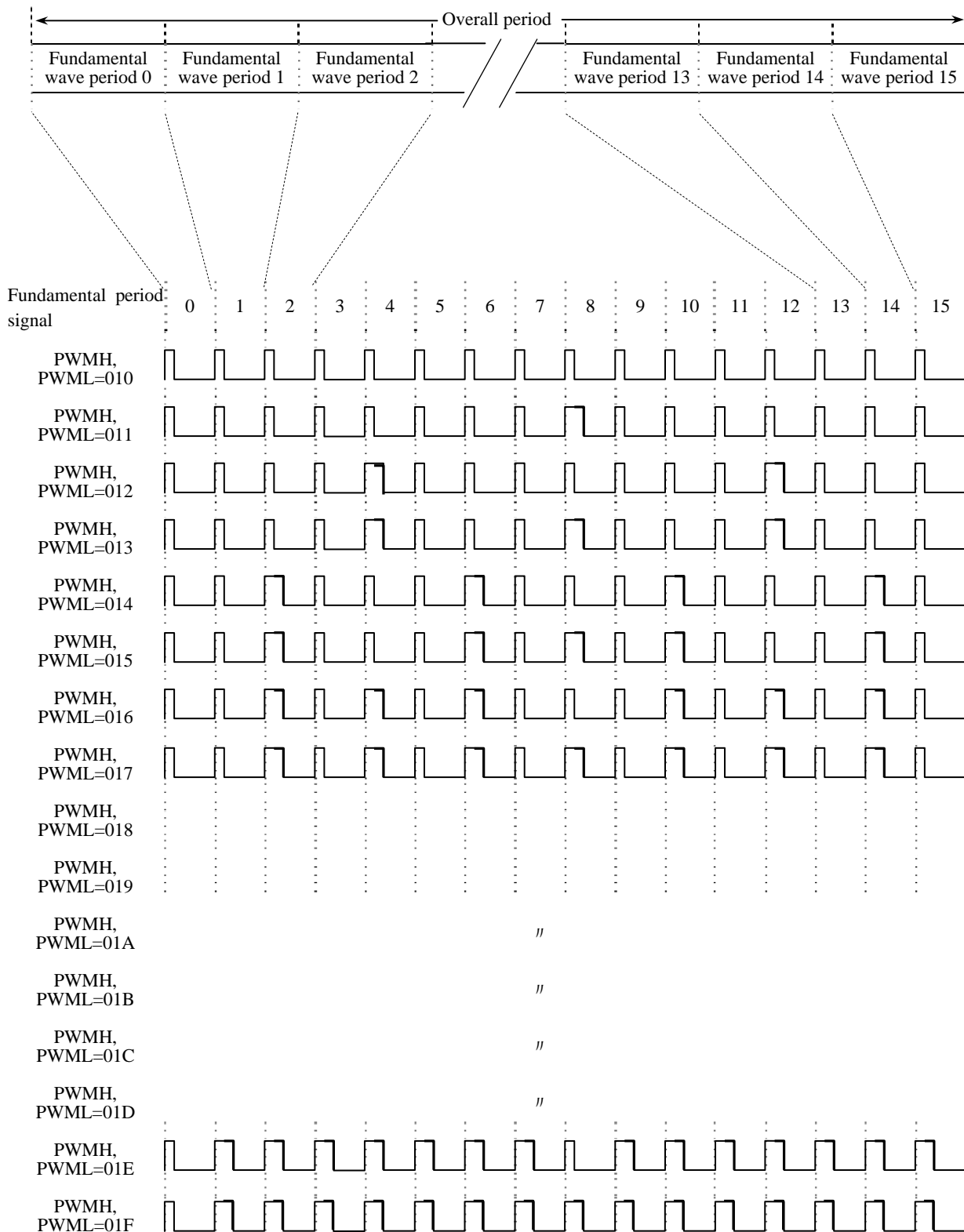
● The fundamental wave period is variable within the range of $\frac{(16\,to\,256)}{3}$ Tcyc.

  Fundamental wave period = (Value represented by PWM4C7 to PWM4C4 + 1) $\times \frac{16}{3}$ Tcyc

  ・ The overall period can be changed by changing the fundamental wave period.

  ・ The overall period is made up of 16 fundamental wave periods.

PWM

Examples:
- Wave comparison when the 12-bit PWM contains 237[H].

    12-bit register configuration → (PWMH), (PWML) = 237[H]

1. Pulse added system (this series)

Fundamental period signal
PWMH, PWML=237

| | | | | | | | | | | | | | | | |
|0|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|

(Overall period)
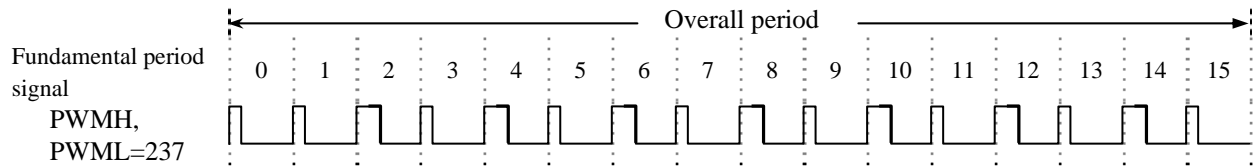
2. Ordinary system

    Since the ripple component of the integral output in this system is greater than that of the pulse added system as seen from the figure below, the pulse added system is considered better for motor-controlling uses.

PWMH, PWML=237

(Overall period)

Ripple
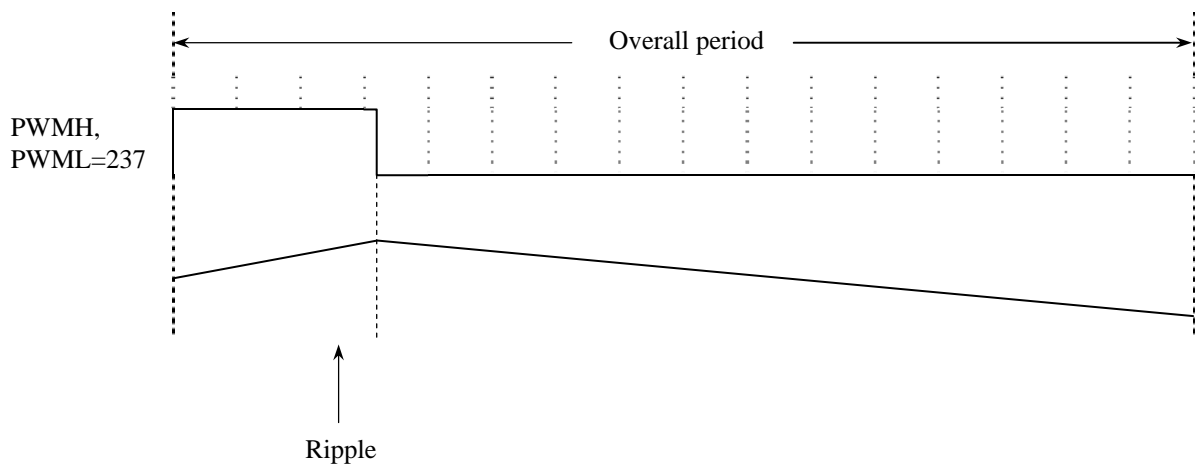
3.15 AD Converter (ADC12)

3.15.1 Overview

This series of microcontrollers incorporates a 12-bit resolution AD converter that has the features listed below. It allows the microcontroller to take in analog signals easily.

1) 12-bit resolution
2) Successive approximation
3) AD conversion mode select (resolution switching)
4) 9-channel analog input
5) Conversion time select
6) Automatic reference voltage generation control

3.15.2 Functions

1) Successive approximation
   - The ADC has a resolution of 12 bits.
   - Requires some conversion time.
   - The conversion results are placed in the AD conversion results registers (ADRLC, ADRHC).

2) AD conversion select (resolution switching)

   The AD converter supports two AD conversion modes: 12- and 8-bit conversion modes so that the appropriate conversion resolution can be selected according to the operating conditions of the application. The AD mode register (ADMRC) is used to select the AD conversion mode.

3) 9-channel analog input

   The signal to be converted is selected using the AD converter control register (ADCRC) out of 9 types of analog signals that are supplied from port 0 pins and pins P70 and P71.

4) Conversion time select

   The AD conversion time can be set to 1/1 to 1/128 (frequency division ratio). The AD mode register (ADMRC) and AD conversion results register low byte (ADRLC) are used to select the conversion time for appropriate AD conversion.

5) Automatic reference voltage generation control

   The ADC incorporates a reference voltage generator that automatically generates the reference voltage when the AD converter is started. Generation of the reference voltage stops automatically at the end of AD conversion, which dispenses with the deed to manually provide on/off control of the reference voltage. There is also no need to supply the reference voltage externally.

6) It is necessary to manipulate the following special control registers to control the AD converter:

• ADCRC, ADMRC, ADRLC, ADRHC

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE58	0000 0000	R/W	ADCRC	AD CHSEL3	AD CHSEL2	AD CHSEL1	AD CHSEL0	ADCR3	AD START	AD ENDF	ADIE
FE59	0000 0000	R/W	ADMRC	ADMD4	ADMD3	ADMD2	ADMD1	ADMD0	ADMR2	ADTM1	ADTM0
FE5A	0000 0000	R/W	ADRLC	DATAL3	DATAL2	DATAL1	DATAL0	ADRL3	ADRL2	ADRL1	ADTM2
FE5B	0000 0000	R/W	ADRHC	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

3.15.3 Circuit Configuration

3.15.3.1 AD conversion control circuit

1) The AD conversion control circuit runs in two modes: 12- and 8-bit AD conversion modes.

3.15.3.2 Comparator circuit

1) The comparator circuit consists of a comparator that compares the analog input with the reference voltage and a control circuit that controls the reference voltage generator circuit and the conversion results. The end of conversion bit (ADENDF) of the AD control register (ADCRC) is set when an analog input channel is selected and the AD conversion terminates in the conversion time designated by the conversion time control register. The conversion results are placed in the AD conversion results registers (ADRHC, ADRLC).

3.15.3.3 Multiplexer 1 (MPX1)

1) Multiplexer 1 is used to select the analog signal to be subject to AD conversion from 9 channels of analog signals.

3.15.3.4 Automatic reference voltage generator circuit

1) The reference voltage generator circuit consists of a network of ladder resistors and a multiplexer (MPX2) and generates the reference voltage that is supplied to the comparator circuit. Generation of the reference voltage is automatically started when an AD conversion starts and stopped when the conversion ends. The reference voltage output ranges from VDD to VSS.

3.15.4 Related Registers

3.15.4.1 AD control register (ADCRC)

1) The AD control register is an 8-bit register that controls the operation of the AD converter.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE58	0000 0000	R/W	ADCRC	AD CHSEL3	AD CHSEL2	AD CHSEL1	AD CHSEL0	ADCR3	AD START	AD ENDF	ADIE

ADCHSEL3 (bit 7):
ADCHSEL2 (bit 6):
ADCHSEL1 (bit 5):   }   **AD conversion input signal select**
ADCHSEL0 (bit 4):

These 4 bits are used to select the signal to be subject to AD conversion.

AD CHSEL3	AD CHSEL2	AD CHSEL1	AD CHSEL0	Signal Input Pin
0	0	0	0	P00/AN0
0	0	0	1	P01/AN1
0	0	1	0	P02/AN2
0	0	1	1	P03/AN3
0	1	0	0	P04/AN4
0	1	0	1	P05/AN5
0	1	1	0	P06/AN6
0	1	1	1	–
1	0	0	0	P70/AN8
1	0	0	1	P71/AN9

ADCRC3 (bit 3): Fixed bit

This bit must always be set to 0.

ADSTART (bit 2): AD converter operation control

This bit starts (1) and stops (0) AD conversion processing. AD conversion starts when this bit is set to 1. This bit is automatically reset when AD conversion terminates. The conversion time is defined using the ADTM2 bit of the AD conversion results register low byte (ADRLC) and bits ADTM1 and ADTM0 of the AD mode register (ADMRC).

AD conversion stops when this bit is set to 0. Correct conversion results cannot be obtained if this bit is cleared during AD conversion processing. This bit must never be cleared or the microcontroller must never be placed in the HALT or HOLD mode while AD conversion processing is in progress.

ADENDF (bit 1): End of AD conversion flag

This bit identifies the end of AD conversion. It is set when AD conversion is finished. Then, an interrupt request to vector address 0043H is generated if ADIE is set to 1. If ADENDF is set to 0, it indicates that no AD conversion is in progress.

This flag must be cleared with an instruction.

ADIE (bit 0): AD conversion interrupt request enable control

An interrupt request to vector address 0043H is generated when this bit and ADENDF are set to 1.

Notes:

- *Setting ADCHSEL3 to ADCHSEL0 to any value from '1010' to '1111' and '0111' is inhibited.*
- *Do not place the microcontroller in the HALT or HOLD mode with ADSTART set to 1. Make sure that ADSTART is set to 0 before putting the microcontroller in the HALT or HOLD mode.*

ADC12

3.15.4.2 AD mode register (ADMRC)

1)   The AD mode register is an 8-bit register for controlling the operation mode of the AD converter.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE59	0000 0000	R/W	ADMRC	ADMD4	ADMD3	ADMD2	ADMD1	ADMD0	ADMR2	ADTM1	ADTM0

ADMD4 (bit 7): Fixed bit

This bit must always be set to 0.

ADMD3 (bit 6): AD conversion mode control (resolution select)

This bit selects the AD converter's resolution between 12-bit AD conversion mode (0) and 8-bit AD conversion mode (1).

When this bit is set to 1, the AD converter serves as an 8-bit AD converter. The conversion results are placed only in the AD conversion results register high byte (ADRHC); the contents of the AD conversion results register low byte (ADRLC) remain unchanged.

When this bit is set to 0, the AD converter serves as a 12-bit AD converter. The conversion results are placed in the AD conversion results register high byte (ADRHC) and the higher-order 4 bits of the AD conversion results register low byte (ADRLC).

ADMD2 (bit 5): Fixed bit

This bit must always be set to 0.

ADMD1 (bit 4): Fixed bit

This bit must always be set to 0.

ADMD0 (bit 3): Fixed bit

This bit must always be set to 0.

ADMR2 (bit 2): Fixed bit

This bit must always be set to 0.

ADTM1 (bit 1):
ADTM0 (bit 0):   **AD conversion time control**

These bits and bit 0 (ADTM2) of the AD conversion results register low byte define the conversion time.

ADRLC Register	ADMRC Register		Frequency Division Ratio
ADTM2	ADTM1	ADTM0	
0	0	0	1/1
0	0	1	1/2
0	1	0	1/4
0	1	1	1/8
1	0	0	1/16
1	0	1	1/32
1	1	0	1/64
1	1	1	1/128

How to calculate the conversion time

- 12-bit AD conversion mode: Conversion time＝((52/(division ratio))+2) ×(1/3) ×Tcyc
- 8-bit AD conversion mode: Conversion time＝((32/(division ratio))+2) ×(1/3) ×Tcyc

Notes:

- *The conversion time is doubled in the following cases:*
  - *1) The AD conversion is carried out in the 12-bit AD conversion mode for the first time after a system reset.*
  - *2) The AD conversion is carried out for the first time after the AD conversion mode is switched from 8-bit to 12-bit AD conversion mode.*
- *The conversion time determined by the above formula is taken in the second and subsequent conversions or in the AD conversions that are carried out in the 8-bit AD conversion mode.*

3.15.4.3 AD conversion results register low byte (ADRLC)

1) The AD conversion results register low byte is used to hold the lower-order 4 bits of the results of an AD conversion carried out in the 12-bit AD conversion mode and to control the conversion time.

2) Since the data in this register is not established during an AD conversion, the conversion results must be read out only after the AD conversion is completed.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5A	0000 0000	R/W	ADRLC	DATAL3	DATAL2	DATAL1	DATAL0	ADRL3	ADRL2	ADRL1	ADTM2

DATAL3 (bit 7):
DATAL2 (bit 6):
DATAL1 (bit 5):
DATAL0 (bit 4):   **Lower-order 4 bits of AD conversion results**

ADRL3 (bit 3): Fixed bit

This bit must always be set to 0.

ADRL2 (bit 2): Fixed bit

This bit must always be set to 0.

ADRL1 (bit 1): Fixed bit

This bit must always be set to 0.

ADTM2 (bit 0): AD conversion time control

This bit and AD mode register bits ADTM1 and ADTM0 are used to control the conversion time. See the subsection on the AD mode register for the procedure to set the conversion time.

Note:

- *The conversion results data contains some errors (quantization error + combination error). Be sure to use only valid conversion results while referring to the latest "SANYO Semiconductor Data Sheet.*

3.15.4.4 AD conversion results register high byte (ADRHC)

1) The AD conversion results register high byte is used to hold the higher-order 8 bits of the results of an AD conversion that is carried out in the 12-bit AD conversion mode. The register stores the whole 8 bits of an AD conversion that is carried out in the 8-bit AD conversion mode.

2) Since the data in this register is not established during an AD conversion, the conversion results must be read out only after the AD conversion is completed.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5B	0000 0000	R/W	ADRHC	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0

3.15.5 AD Conversion Example

3.15.5.1 12-bit AD conversion mode

1) Setting up the 12-bit AD conversion mode

- Set the ADMD3 bit of the AD mode register (ADMRC) to 0.

2) Setting up the conversion time

- To set the conversion time to 1/32, set bit 0 (ADTM2) of the AD conversion results register low byte to 1, bit 1 (ADTM1) of the AD mode register to 0, and bit 0 (ADTM0) of the AD mode register to 1.

3) Setting up the input channel

- When using AD channel input AN5, set AD control register (ADCRC) bit 7 (ADCHSEL3) to 0, bit 6 (ADCHSEL2) to 1, bit 5 (ADCHSEL1) to 0, and bit 4 (ADCHSEL0) to 1.

4) Starting AD conversion

- Set bit 2 (ADSTART) of the AD mode register (ADCRC) to 1.

- The conversion time will be twice the normal conversion time immediately after a system reset and for the first AD conversion that is carried out after the AD conversion mode is switched from 8-bit to 12-bit conversion mode. In the second and subsequent AD conversions, the normal conversion time is taken.

5) Testing the end of AD conversion flag

- Monitor bit 1 (ADENDF) of the AD mode register (ADCRC) until it is set to 1.

- After verifying that bit 1 (ADENDF) is set to 1, clear it to zero.

6) Reading the AD conversion results

- Read the contents of the AD conversion results registers high byte (ADRHC) and low byte (ADRLC). The read conversion data contains some errors (quantization error + combination error). Be sure to use only valid conversion results while referring to the latest "SANYO Semiconductor Data Sheet."

- Pass the read data to the application software.

- Return to step 4) to repeat the conversion processing.

3.15.6 Hints on the Use of the ADC

1) The conversion time that the user can select varies depending on the frequency of the cycle clock. When preparing a program, refer to the latest edition of "SANYO Semiconductor Data Sheet" to select an appropriate conversion time.

2) Setting ADSTART to 0 while conversion is in progress will stop the conversion function.

3) Do not place the microcontroller in the HALT or HOLD mode while AD conversion processing is in progress. Make sure that ADSTART is set to 0 before putting the microcontroller in the HALT or HOLD mode.

4) ADSTART is automatically reset and the AD converter stops operation if a reset is triggered while AD conversion processing is in progress.

5) When conversion is finished, the end of AD conversion flag (ADENDF) is set and, at the same time, the AD conversion operation control bit (ADSTART) is reset. The end of conversion condition can be identified by monitoring ADENDF. An interrupt request to vector address 0043H is generated by setting ADIE.

6) The conversion time is doubled in the following cases:

   - The AD conversion is carried out in the 12-bit AD conversion mode for the first time after a system reset.

   - The AD conversion is carried out for the first time after the AD conversion mode is switched from 8-bit to 12-bit AD conversion mode.

   The conversion time determined by the formula given in the paragraph entitled "How to calculate the conversion time" is taken in the second and subsequent conversions or in the AD conversions that are carried out in the 8-bit AD conversion mode.

7) The conversion results data contains some errors (quantization error + combination error). Be sure to use only valid conversion results while referring to the latest "SANYO Semiconductor Data Sheet."

8) Make sure that only input voltages that fall within the specified range are supplied to pins P00/AN0 to P06/AN6 and pins P70/AN8 and P71/AN9. Application of a voltage greater than VDD or lower than VSS to an input pin may exert adverse influences on the converted value of the channel in question or other channels.

9) Take the following measures to prevent reduction in conversion accuracy due to noise interferences:

   - Add external bypass capacitors of several μF plus thousands pF near the VDD1 and VSS1 pins (as close as possible; 5 mm or less is desirable).

   - Add an appropriate external low-pass filter (RC), which is appropriated to reject noise interferences, or capacitors close to each analog input pin. To preclude adverse coupling influences, use a ground that is free of noise interferences (as a guideline, R = approx. 5kΩ or less, C = 1000 pF to 0.1 μF).

   - Do not lay analog signal lines close to, in parallel with, or in a crossed arrangement with digital pulse signal lines or signal lines in which large current changes can occur. Shield both ends of analog signal lines with noise-free ground shields.

   - Make sure that no digital pulses are applied to or generated out of pins adjacent to the analog input pin that is being subject to conversion.

   - Correct conversion results may not be obtained because of noise interferences if the state of port outputs is changing. To minimize the adverse influences of noise interferences, it is necessary to keep the line resistance across the power supply and the VDD pins of the microcontroller at minimum. This should be kept in mind when designing an application circuit.

   - Adjust the amplitudes of the voltage at the oscillator pin and the I/O voltages at the other pins so that they fall within the voltage range between VDD and VSS.

ADC12

10) To obtain valid conversion data, perform conversion operations on the input several times, discard the maximum and minimum values of the conversion results, and take an average of the remaining data.

4. Control Functions

4.1 Interrupt Function

4.1.1 Overview

This series of microcontrollers has the capabilities to control three levels of multiple interrupts, i.e., low level (L), high level (H), and highest level (X). The master interrupt enable and interrupt priority control registers are used and enable or disable interrupts and determine the priority of interrupts.

4.1.2 Functions

1) Interrupt processing

- Peripheral modules generate an interrupt request to the predetermined vector address when the interrupt request and interrupt request enable flags are set to 1.

- When the microcontroller receives an interrupt request from a peripheral module, it determines the priority and interrupt enable status of the interrupt. If the interrupt request is legitimate for processing, the microcontroller saves the value of PC in the stack and causes a branch to the predetermined vector address.

- The return from the interrupt routine is accomplished by the RETI instruction, which restores the old state of the PC and interrupt level.

2) Multilevel interrupt control

- The interrupt function supports three levels of interrupts, that is, the low level (L), high level (H), and highest level (X). The interrupt function will not accept any interrupt requests of the same level or lower than that of the interrupt that is currently being processed.

3) Interrupt priority

- When interrupt requests to two or more vector addresses occur at the same time, the interrupt request of the highest level takes precedence over the other interrupt requests. Among the interrupt requests of the same level, the one whose vector address is the smallest is prioritary.

4) Interrupt request enable control

- The master interrupt enable register can be used to control the enabling/disabling of H- and L-level interrupt requests.

- Interrupt requests of the X level cannot be disabled.

5) Interrupt disable period

- Interrupts are held disabled for a period of 2Tcyc after a write is made to the IE (FE08H) or IP (FE09H) register, or the HOLD mode is reset.

- No interrupt can occur during the interval between the execution of an instruction that loads the PCON (FE07H) register and the execution of the next instruction.

- No interrupt can occur during the interval between the execution of a RETI instruction and the execution of the next instruction.

Interrupt

6) Interrupt level control
    - Interrupt levels can be selected on a vector address basis.

Table of Interrupts

No.	Vector Address	Selectable Level	Interrupt Sources
1	00003H	X or L	INT0
2	0000BH	X or L	INT1
3	00013H	H or L	INT2/T0L/INT4
4	0001BH	H or L	INT3/INT5/base timer
5	00023H	H or L	T0H
6	0002BH	H or L	T1L/T1H
7	00033H	H or L	SIO0/UART1 receive
8	0003BH	H or L	SIO1/UART1 transmit
9	00043H	H or L	ADC/T6/T7/PWM4/PWM5
10	0004BH	H or L	Port 0

- Priority levels: $X > H > L$
- Of interrupts of the same level, the one with the smallest vector address takes precedence.

7) To enable interrupts and to specify their priority, it is necessary to manipulate the following special function registers:
    - IE, IP

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE08	0000 HH00	R/W	IE	IE7	XFLG	HFLG	LFLG	-	-	XCNT1	XCNT0
FE09	0000 0000	R/W	IP	IP4B	IP43	IP3B	IP33	IP2B	IP23	IP1B	IP13

4.1.3 Circuit Configuration

4.1.3.1 Master interrupt enable control register (IE) (6-bit register)

1) The master interrupt enable control registers enables and disables H- and L-level interrupts.
2) The interrupt level flag of the register can be read.
3) The register selects the level (L or X) of interrupts to vector addresses 00003H and 0000BH.

4.1.3.2 Interrupt priority control register (IP) (8-bit register)

1) The interrupt priority control register selects the level (H or L) of interrupts to vector addresses 00013H to 0004BH.

4.1.4   Related Registers

4.1.4.1   Master interrupt enable control register (IE)

1)   The master interrupt enable control register is a 6-bit register for controlling the interrupts. Bits 6 to 4 of this register are read only.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE08	0000 HH00	R/W	IE	IE7	XFLG	HFLG	LFLG	-	-	XCNT1	XCNT0

IE7 (bit 7): H-/L-level interrupt enables/disables control

- A 1 in this bit enables H- and L-level interrupt requests to be accepted.
- A 0 in this bit disables H- and L-level interrupt request to be accepted.
- X-level interrupt requests are always enabled regardless of the state of this bit.

XFLG (bit 6): X-level interrupt flag (R/O)

- This bit is set when an X-level interrupt is accepted and reset when execution returns from the processing of the X-level interrupt.
- This bit is read only. No instruction can rewrite the value of this bit directly.

HFLG (bit 5): H-level interrupt flag (R/O)

- This bit is set when an H-level interrupt is accepted and reset when execution returns from the processing of the H-level interrupt.
- This bit is read only. No instruction can rewrite the value of this bit directly.

LFLG (bit 4): L-level interrupt flag (R/O)

- This bit is set when an L-level interrupt is accepted and reset when execution returns from the processing of the L-level interrupt.
- This bit is read only. No instruction can rewrite the value of this bit directly.

(Bits 3, 2): These bits do not exist. They are always read as "1."

XCNT1 (bit 1): 0000BH Interrupt level control flag

- A 1 in this bit sets all interrupts to vector address 0000BH to the L-level.
- A 0 in this bit sets all interrupts to vector address 0000BH to the X-level.

XCNT0 (bit 0): 00003H Interrupt level control flag

- A 1 in this bit sets all interrupts to vector address 00003H to the L-level.
- A 0 in this bit sets all interrupts to vector address 00003H to the X-level.

4.1.4.2   Interrupt priority control register (IP)

1)   The interrupt priority control register is an 8-bit register that selects the interrupt level (H/L) of interrupts to vector addresses 00013H to 0004BH.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE09	0000 0000	R/W	IP	IP4B	IP43	IP3B	IP33	IP2B	IIP23	IP1B	IP13

	Interrupt Vector Address	IP Bit	Value	Interrupt Level
7	0004BH	IP4B	0	L
			1	H
6	00043H	IP43	0	L
			1	H
5	0003BH	IP3B	0	L
			1	H
4	00033H	IP33	0	L
			1	H
3	0002BH	IP2B	0	L
			1	H
2	00023H	IP23	0	L
			1	H
1	0001BH	IP1B	0	L
			1	H
0	00013H	IP13	0	L
			1	H

4.2   System Clock Generator Function

4.2.1   Overview

This series of microcontrollers incorporates five systems of oscillator circuits, i.e., the main clock oscillator, subclock oscillator, low- and medium-speed RC oscillators, and multifrequency RC oscillator as system clock generator circuits. The low- and medium-speed RC and multifrequency RC oscillator circuits have built-in resistors and capacitors, so that no external circuit is required.
The system clock can be selected from these five types of clock sources under program control.

4.2.2   Functions

1) System clock select
   - Allows the system clock to be selected under program control from five types of clocks generated by the main clock oscillator, subclock oscillator, low- and medium-speed RC oscillator, and multifrequency RC oscillator.

2) System clock frequency division
   - Divides frequency of the oscillator clock selected as the system clock and supplies the resultant clock to the system as the system clock.
   - The frequency divider circuit is made up of two stages:

     The first stage allows the selection of division ratios of $\frac{1}{1}$ and $\frac{1}{2}$.

     The second stage allows the selection of division ratios of $\frac{1}{1}$, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$, $\frac{1}{64}$, and $\frac{1}{128}$.

3) Oscillator circuit control
   - Allows the start/stop control of the five systems of oscillators to be executed independently through microcontroller instructions. The main clock and subclock oscillator circuits share pins (CF1/XT1 and CF2/XT2) and cannot be used at the same time.
   - The CF oscillator circuit may be either a low power dissipation type CF oscillation low amplifier or a CF oscillation normal amplifier.

4) Multiplexed input pin function
   - The CF oscillation/crystal oscillation pins (CF1/XT1 and CF2/XT2) can also be used as general-purpose input ports.

5) Oscillator circuit states and operating modes

Mode/Clock	Main Clock	Sub clock	Low-speed RC Oscillator	Medium-speed RC Oscillator	Multifrequency RC Oscillator	System Clock
Reset	Stopped	Stopped	Stopped	Running	Stopped	Medium-speed RC oscillator
Reset released	Running	Stopped	Stopped	Running	Stopped	Medium-speed RC oscillator
Normal mode	Programmable	Programmable	Programmable	Programmable	Programmable	Programmable
HALT	State established at entry time	State established at entry time	State established at entry time	State established at entry time	State established at entry time	State established at entry time
HOLD	Stopped	Stopped	Stopped	Stopped	Stopped	Stopped
Immediately after exit from HOLD mode	State established at entry time	State established at entry time	Running	Running	Stopped	Low- or medium-speed RC oscillator according to the state that has been defined on entry by bit 1 of SLWRC register
X'tal HOLD	Stopped	State established at entry time	Stopped	Stopped	Stopped	Stopped
Immediately after exit from X'tal HOLD mode	State established at entry time	State established at entry time	State established at entry time	State established at entry time	State established at entry time	State established at entry time

Note: See Section 4.4," Standby Function," for the procedures to enter and exit the microcontroller operating modes

System Clock

- **HOLD mode**
  - All oscillators stopped.
  - If SLWRC register, bit 1 is set to 1 Since SLWRC register, bit 0 is set, and OCR register, bits 4 and 5 are cleared, the low-speed RC oscillator is activated and designated as system clock when the HOLD mode is released.
  - If SLWRC register, bit 1 is set to 0 Since OCR register, bits 1, 4, and 5 are cleared, the medium-speed RC oscillator is activated and designated as system clock when the HOLD mode is released.
  - Since MRCR register bits 6 and 7 are cleared, the multifrequency RC oscillator is stopped when the HOLD mode is released.

- **Reset**
  - Main clock stops in reset state and starts oscillation when reset state is cleared.
  - Subclock stopped
  - Low--speed RC oscillator stopped
  - Medium-speed RC oscillator started
  - Multifrequency RC oscillator stopped

- **Normal operating mode**
  - Start/stop of oscillators programmable

- **HALT mode**
  - All oscillators retain the state established when HALT mode is entered.

- **X'tal HOLD**
  - Main clock, low- and medium-speed RC oscillators, and multifrequency RC oscillator are stopped. Subclock retains the state established when X'tal HOLD mode is entered.
  - Contents of OCR, SLWRC, and MRCR registers remain unchanged.
  - CPU enters this mode after selecting subclock, low- or medium-speed RC oscillator as system clock source.
  - When X'tal HOLD mode is exited, the oscillators return to the state established when the mode is entered.

6) To control the system clock, it is necessary to manipulate the following special function registers:
   - PCON, OCR, CLKDIV, MRCR, XT2PC, P1TST, CFLVM, SLWRC

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE07	HHHH H000	R/W	PCON	-	-	-	-	-	XTIDLE	PDN	IDLE
FE0C	HHHH H000	R/W	CLKDIV	-	-	-	-	-	CLKDV2	CLKDV1	CLKDV0
FE0D	00HX XXXX	R/W	MRCR	MRCSEL	MRCST	-	RCCTD4	RCCTD3	RCCTD2	RCCTD1	RCCTD0
FE0E	0000 XX00	R/W	OCR	CLKSGL	EXTOSC	CLKCB5	CLKCB4	XT2IN	XT1IN	RCSTOP	CFSTOP
FE43	0000 0000	R/W	XT2PC	XT2PCB7	XT2PCB6	XT2PCB5	XT2PCB4	XTCFIN	XT2PCB2	XT2PCB1	XT2PCB0
FE47	0000 H0H0	R/W	P1TST	FIX0	FIX0	MRCSFT	FIX0	-	DSNKOT	-	FIX0
FE57	HHH0 HH00	R/W	CFLVM	-	-	-	CFMON	-	-	FIX0	FIX0
FE7C	HHHH H000	R/W	SLWRC	-	-	-	-	-	CFLAMP	SLRCSEL	SLRCSTAT

4.2.3 Circuit Configuration

4.2.3.1 Main clock oscillator circuit

1) The main clock oscillator circuit gets ready for oscillation by connecting a ceramic oscillator and a capacitor to the CF1/XT1 and CF2/XT2 pins and controlling the OCR and XT2PC registers.

2) The data at the CF1/XT1 and CF2/XT2 pins can be read as bits 2 and 3 of the OCR register.

3) The general-purpose input configuration must be selected and the CF1/XT1 and CF2/XT2 pins must be connected to VSS1 when neither main nor subclock is to be used or they are not to be used as general-purpose input ports.

4.2.3.2 Subclock oscillator circuit

1) The subclock oscillator circuit gets ready for oscillation by connecting a crystal oscillator (32.768 kHz standard), a capacitor and a damping resistor to the CF1/XT1 and CF2/XT2 pins and controlling the OCR and XT2PC registers.

2) The data at the CF2/XT2 pin can be read as bit 3 of the OCR register. The data at the CF1/XT1 pin is not read as bit 2 of the OCR register.

4.2.3.3 Internal low-speed RC oscillator

1) The low-speed RC oscillator oscillates according to the internal resistor and capacitor (at 100 kHz standard).

2) The internal low-speed RC oscillator serves as the system clock that is to be used for low-power, low-speed operation.

4.2.3.4   Internal medium-speed RC oscillator (conventional RC oscillator)

1)   The medium-speed RC oscillator oscillates according to the internal resistor and capacitor (at 1 MHz standard).

2)   The clock from the medium-speed RC oscillator is designated as the system clock after the reset state is released. After the HOLD mode is exited, the clock from the medium- or low-speed RC oscillator that is selected on entry into the HOLD mode is designated as the system clock.

4.2.3.5   Multifrequency RC oscillator circuit

1)   The multifrequency RC oscillator circuit oscillates according to the internal resistor and capacitor.

2)   The circuit counts the clocks with a source oscillation frequency of 16 MHz with a 5-bit counter. The maximum allowable clock rate is 8 MHz.

3)   The circuit toggles out a clock each time the counter value matches the preset count value.

4)   The multifrequency RC oscillator circuit is suited to generate a main clock which does not require the precision in frequency that the external CF oscillator would provide.

4.2.3.6   Power control register (PCON)   (3-bit register)

1)   The power control register specifies the operating mode (normal/HALT/HOLD/X'tal HOLD).

4.2.3.7   Oscillation control register (OCR)   (8-bit register)

1)   The oscillation control register controls the start/stop operations of the oscillator circuits.

2)   This register selects the system clock.

3)   The register sets the division ratio of the oscillation clock to be used as the system clock to $\frac{1}{1}$ or $\frac{1}{2}$.

4)   The state of the CF1/XT1 and CF2/XT2 pins can be read as bits 2 and 3 of this register.

4.2.3.8   Low-speed RC oscillation control register (SLWRC)   (3 bits)

1)   The low-speed RC oscillation control register controls the start/stop operations of the low-/medium-speed RC oscillator circuit.

2)   The register switches between the low-speed RC oscillation clock and the medium-speed RC oscillation clock.

3)   This register also selects the amplifier size of the CF oscillator circuit. The CF oscillator low amplification is effective for reducing power dissipation under such conditions as low voltage, CF= 4 MHz, system frequency division ratio = 1/4 to 1/16.

4.2.3.9   CF1/XT1 and CF2/XT2 general-purpose port input control register (XT2PC)   (8-bit register)

1)   The CF1/XT1 and CF2/XT2 general-purpose port input control register controls the general-purpose input at the CF1/XT1 and CF2/XT2 pins.

4.2.3.10   Multifrequency RC oscillator control register (MRCR)   (8-bit register)

1)   The multifrequency oscillator control register controls the start/stop operations of the multifrequency RC oscillator circuit.

2)   The register selects either CF or multifrequency RC oscillator as the main clock source.

3)   The register also defines the frequency of the multifrequency RC oscillator clock.

4.2.3.11   System clock division control register (CLKDIV)   (3-bit register)

1)   The system clock division control register controls the operation of the system clock divider circuit. The division ratios of $\frac{1}{1}$, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$, $\frac{1}{64}$, and $\frac{1}{128}$ are allowed.

System Clock
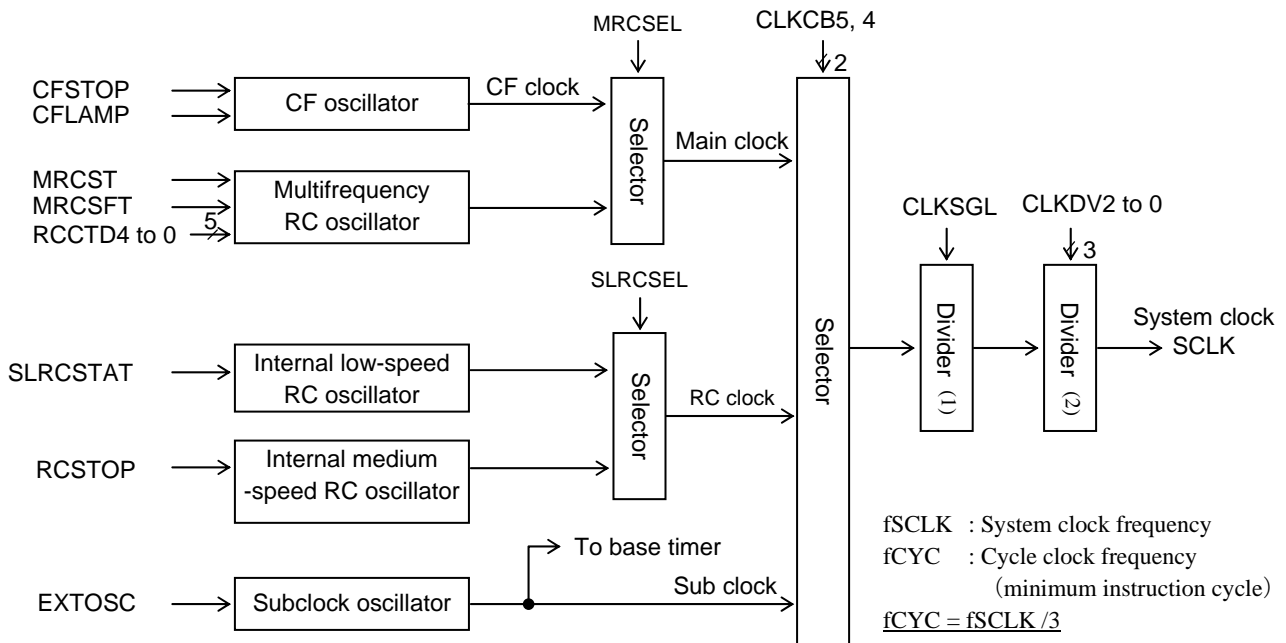
Fig. 4.2.1   System Clock Generator Block Diagram

4.2.4   Related Registers

4.2.4.1   Power Control Register (PCON)   (3-bit register)

1)   The power control register is a 3-bit register used to specify the operating mode (normal/HALT/HOLD/ X'tal HOLD).
   • See Section 4.4, Standby Function, for the procedures to enter and exit the microcontroller operating modes.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE07	HHHH H000	R/W	PCON	-	-	-	-	-	XTIDLE	PDN	IDLE

(bits 7to3): These bits do not exist. They are always read as 1.

XTIDLE (bit 2): X'tal HOLD mode setting flag

PDN (bit 1): HOLD mode setting flag

XTIDLE	PDN	Operating mode
-	0	Normal or HALT mode
0	1	HOLD mode
1	1	X'tal HOLD mode

1)   These bits must be set with an instruction.
   • When the microcontroller enters the HOLD mode, all oscillations (main clock, subclock, low-/medium-speed RC) are suspended and the related registers are placed in the states described below.

   If SLWRC register, bit 1 is se t to 1, SLWRC register, bit 0 is set and OCR register, bits 4 and 5 are cleared.

   If SLWRC register, bit 1 is set to 0, OCR register, bits 1, 4, and 5 are cleared.
   • When the microcontroller exits the HOLD mode, the low- or medium-speed RC oscillator starts operation and is designated as the system clock source. The main clock and subclock returns to the state that is established before the microcontroller entered the HOLD mode.

4-8

- When the microcontroller enters the X'tal HOLD mode, all oscillations except XT (i.e., main clock and low-/medium-speed RC) are suspended but the state of the OCR register remains unchanged.
- Since the X'tal HOLD mode is used usually for low-current clock counting, less current will be consumed if the system clock is switched to the subclock and low- and medium-speed RC oscillations are suspended before the X'tal HOLD mode is entered.

2) XTIDLE must be cleared with an instruction.

3) PDN is cleared when a HOLD mode resetting signal (INT0, INT1, INT2, INT4, INT5, or P0INT) or a reset occurs.

4) Bit 0 is automatically set when PDN is set.

IDLE (bit 0): HALT mode setting flag

1) Setting this bit places the microcontroller into the HALT mode.

2) This bit is automatically set whenever bit 1 is set.

3) This bit is cleared on acceptance of an interrupt request or on receipt of a reset signal.

4.2.4.2   Oscillation Control Register (OCR)   (8-bit register)

1) The oscillation control register is an 8-bit register that controls the operation of the oscillator circuits, selects the system clock, and reads data from the CF1/XT1 and CF2/XT2 pins. Except for read-only bits 3 and 2, all bits of this register can be read or written.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE0E	0000 XX00	R/W	OCR	CLKSGL	EXTOSC	CLKCB5	CLKCB4	XT2IN	XT1IN	RCSTOP	CFSTOP

CLKSGL (bit 7): Clock division ratio select

1) When this bit is set to 1, the clock selected by bits 4 and 5 is used as the system clock as is.

2) When this bit is set to 0, the clock having a clock rate of $\frac{1}{2}$ of the clock selected by bits 4 and 5 is used as the system clock.

EXTOSC (bit 6): CF1/XT1 and CF2/XT2 function control

1) When this bit is set to 1 and CFSTOP (bit 0) are set to 1, the CF1/XT1 and CF2/XT 2 pins serve as the pins for subclock oscillation and get ready for oscillation when a crystal oscillator (32.768 kHz standard), capacitors, and damping resistors are connected. When the OCR register is read in this case, bit 3 reads the data at the CF2/XT2 pin and bit 2 reads 0.

2) When this bit is set to 0, the CF1/XT1 and CF2/XT2 pins serve as the pins for main clock oscillation and get ready for oscillation when a ceramic oscillator, capacitors, feedback resistors, and damping resistors are connected. Start/stop control of the main clock oscillation is provided by CFSTOP (bit 0). When the OCR register is read in this case, bit 3 reads the data at the CF2/XT2 pin and bit 2 reads the data at the CF1/XT1 pin.

CLKCB5 (bit 5): System clock select

CLKCB4 (bit 4): System clock select

1) CLKCB5 and CLKCB4 are used to select the system clock.

2) CLKCB5 and CLKCB4 are cleared at reset time or when the HOLD mode is entered.

CLKCB5	CLKCB4	System clock
0	0	Internal low-/medium-speed RC oscillator
0	1	Main clock
1	0	Subclock
1	1	Main clock

XT2IN (bit 3): CF2/XT2 data (read-only)

XT1IN (bit 2): CF1/XT1 data (read-only)

1) Data that can be read via XT1IN varies as shown in the Table below according to the value of EXTOSC (bit 6).

RCSTOP (bit 1)): Internal medium-speed RC oscillator control

1) Setting this bit to 1 stops the oscillation of the internal medium-speed RC oscillator circuit.

2) Setting this bit to 0 starts the oscillation of the internal medium-speed RC oscillator circuit.

3) When a reset occurs, this bit is cleared and the internal RC oscillator circuit is enabled for oscillation.

4) When the microcontroller enters the HOLD mode, this bit is set as described below according to the state of bit 1 of the SLWRC register.

   If SLWRC register, bit 1 is set to 1, the state of this bit remains unchanged.

   If SLWRC register, bit 1 is set to 0, this bit is cleared and the oscillator starts oscillation and is designated as the system clock source when the microcontroller exits the HOLD mode.

CFSTOP (bit 0): Main clock oscillator control

1) Setting this bit to 1 stops the oscillation of the main clock.

2) Setting this bit to 0 starts the oscillation of the main clock oscillator circuit.

3) This bit is cleared to enable oscillation on a reset.

OCR register(FE0EH)		XT2PC register	CF1/XT1, CF2/XT2 state	OCR register (FE0EH)	
EXTOSC	CFSTOP	XTCFIN		XT2IN	XT1IN
X	0	0	Main clock oscillator active	CF2/XT2 pin data	CF1/XT1 pin data
0	1	0	Main clock oscillator stopped	Unpredictable	Unpredictable
1	1	X	Subclock oscillator active	CF2/XT2 pin data	Read 0
X	0	1	Inhibited	CF2/XT2 pin data	CF1/XT1 pin data
0	1	1	General-purpose input	CF2/XT2 pin data	CF1/XT1 pin data

Note: To use the CF1/XT1 and CF2/XT2 pins as general-purpose input port pins, set XTCFIN (XT2PC register (FE43H), bit 3) to 1, EXTOSC (OCR register (FE0EH), bit 6) to 0, and CFSTOP (OCR register (FE0EH), bit 0) to 1.

4.2.4.3 Low-speed RC oscillator control register (SLWRC) (3-bit register)

1) The low-speed RC oscillator control register (SLWRC) is a 3-bit register that controls the operation of the low-/medium-speed RC oscillator circuits and the amplifier size of the CF oscillator circuit.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7C	HHHH H000	R/W	SLWRC	-	-	-	-	-	CFLAMP	SLRCSEL	SLRCSTAT

(bits 7 to 3): These bits do not exist. They are always read as 1.

CFLAMP (bit 2): CF oscillator amplifier size select control

1) A 1 in this bit selects the low amplifier size for the CF oscillator circuit.

2) A 0 in this bit selects the normal amplifier size for the CF oscillator circuit.

*: See Subsection 4.2.5 as a predefined procedure is required to switch the selection.

SLRCSEL (bit 1): Internal low-/medium-speed RC oscillator clock select control

1) A 1 in this bit selects the clock for the internal low-speed RC oscillator.

2) A 0 in this bit selects the clock for the internal medium-speed RC oscillator.

SLRCSTAT (bit 0): Internal low-speed RC oscillator control

1) A 1 in this bit starts the internal low-speed RC oscillator circuit.

2) A 0 in this bit stops the internal low-speed RC oscillator circuit.

3) This bit is cleared at reset time.

4) This bit is set as described below according to the state of bit 1 of the SLWRC register when the microcontroller enters the HOLD mode.

    If SLRCSEL, bit 1 is set to1 , this bit is set and the oscillator starts oscillation and is designated as the system clock source when the microcontroller exits the HOLD mode.

    If SLRCSEL, bit 1 is set to 0, the state of this bit remains unchanged.

4.2.4.4   CF1/XT1, CF2/XT2 general-purpose port input control register (XT2PC)   (8-bit register)

1) The XT2 general-purpose input control register is an 8-bit register that controls the general-purpose input at the CF1/XT1 and CF2/XT2 pins.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE43	0000 0000	R/W	XT2PC	XT2PCB7	XT2PCB6	XT2PCB5	XT2PCB4	XTCFIN	XT2PCB2	XT2PCB1	XT2PCB0

XT2PCB 7 to XT2PCB4 (bits 7 to 4): General-purpose flags

These bits can be used as general-purpose flag bits. Any manipulations of these bits exert no influence on the operation of this function block.

XTCFIN (bit 3): CF1/XT1 and CF2/XT2 input control

1) This bit and EXTOSC (OCR register (FE0EH), bit 6) and CFSTOP (OCR register (FE0EH), bit 0) are used to select the function of the CF1/XT1 and CF2/XT2 pins between main clock, subclock, and general-purpose input port pins. (See 4.2.4.2, "Oscillation control register," for details.)

XT2PCB2 to XT2PCB0 (bits 2 to 0): General-purpose flags

These bits can be used as general-purpose flag bits. Any manipulations of these bits exert no influence on the operation of this function block.

4.2.4.5   Multifrequency RC oscillator control register (MRCR)   (7-bit register)

1) The multifrequency RC oscillator control register is an 7-bit register that controls the operation of the multifrequency RC oscillator circuit and selects the main clock.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE0D	00HX XXXX	R/W	MRCR	MRCSEL	MRCST	-	RCCTD4	RCCTD3	RCCTD2	RCCTD1	RCCTD0

System Clock

MRCSEL (bit 7): Multifrequency RC oscillator clock select

1) When this bit is set to 1, the clock output from the multifrequency RC oscillator is selected as the main clock. The multifrequency RC oscillator clock will be the system clock if the main clock is selected as the system clock in the above-mentioned OCR register.

2) When this bit is set to 0, the multifrequency RC oscillator clock is not selected as the main clock; CF is designated as the main clock.

3) This bit is cleared when the microcontroller enters the HOLD mode.

MRCST (bit 6): Multifrequency RC oscillation start control

1) A 1 in this bit starts the multifrequency RC oscillator circuit.

2) A 0 in this bit stops the multifrequency RC oscillator circuit.

3) This bit is cleared when the microcontroller enters the HOLD mode.

RCCTD4 (bit 4):

RCCTD3 (bit 3):

RCCTD2 (bit 2):   **Multifrequency RC oscillator frequency select**

RCCTD1 (bit 1):

RCCTD0 (bit 0):

1) These bits set up the source oscillator clock counter.

2) The frequency of the clock generated by the multifrequency RC oscillator is:

Source oscillation frequency $/ ((\text{RCCTD value} + 1) \times 2)$

3) The initial value of RCCTD is unpredictable.

Note 1: The system clock may set to an excessively high rate depending on the count value configured. This may cause malfunctions if it exceeds the operating clock range.

Note 2: Data may not be set up properly if the subclock or internal low-/medium-speed RC oscillator clock is selected as the system clock and RCCTD is rewritten with MRCSEL (bit 7) set to "H." Be sure to set MRCSEL (bit 7) to "L" when rewriting RCCTD while selecting the subclock or internal low-/medium-speed RC oscillator clock as the system clock.

Note 3: When switching the system clock, allow an <u>oscillation stabilization time of 100 μs or longer</u> after the multifrequency RC oscillator circuit switches from the "oscillation stopped" to "oscillation enabled" state.

Note 4: The multifrequency RC oscillator circuit may be of 6- or 5-bit counter type which is dependent on the type of the microcontroller. You need to pay attention to this fact and to the correct set of development tools when using this function. This series adopts a 5-bit counter.

Note 5: Depending on the type of SANYO microcontrollers, RCCTD is initialized to a frequency that is close to that of the internal (low-speed) RC oscillator or the value of RCCTD is unpredictable. The value of RCCTD of this series microcontrollers is unpredictable.

4.2.4.6 P1TST register (P1TST) (6-bit register)

1) This register controls the frequency shifting of the source oscillation clock from the multifrequency RC oscillator circuit.

2) This register is used to avoid adverse influences of radio interferences caused by the source oscillation clock.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE47	0000 H0H0	R/W	P1TST	FIX0	FIX0	MRCSFT	FIX0	-	DSNKOT	-	FIX0

FIX0 (bits 7, 6, 4, and 0）: These bits are used for testing only. Must always be set to 0.

MRCSFT (bit 5): Controls the frequency shift of the source oscillation clock from the multifrequency RC oscillator circuit.

1) A 0 in this bit initializes the source oscillation clock frequency.

2) A 1 in this bit shifts the frequency of the source oscillation clock slightly to a lower frequency.

DSNKOT (bit 2): Realtime output control

1) This bit is used to control the realtime output of the high-speed clock counter. It exerts no influence on the generation of the system clock.

(bits 3 and1）: These bits do not exist. They are always read as 1.

4.2.4.7 System clock divider control register (CLKDIV) (3-bit register)

1) The system clock divider control register controls the frequency division processing of the system clock.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE0C	HHHH H000	R/W	CLKDIV	-	-	-	-	-	CLKDV2	CLKDV1	CLKDV0

(bits 7 to 3): These bits do not exist. 1 is always read when these bits are read.

CLKDV2 (bit 2):

CLKDV1 (bit 1): ⎫ **Define the division ratio of the system clock.**

CLKDV0 (bit 0):

CLKDV2	CLKDV1	CLKDV0	Division Ratio
0	0	0	$\frac{1}{1}$
0	0	1	$\frac{1}{2}$
0	1	0	$\frac{1}{4}$
0	1	1	$\frac{1}{8}$
1	0	0	$\frac{1}{16}$
1	0	1	$\frac{1}{32}$
1	1	0	$\frac{1}{64}$
1	1	1	$\frac{1}{128}$

4.2.5 Example of Switching the CF Oscillator Amplifier Size

1) System clock state
   - Put the system clock into a state other than the CF oscillation (main).

2) Switch the CF oscillation amplifier size to "low."
   - Set CFLAMP (bit 2) of the low-speed RC oscillator control register to 1.

3) Wait for the CF oscillation stabilization time.
   - Wait for the CF oscillation stabilization time that is specified in the Semiconductor Bulletin brochure.

4)   Check for normal CF oscillation (this step is highly recommended especially when using a low-voltage configuration).
•   Using the CF oscillation monitoring feature, make sure that the system clock is oscillating.

5)   Switch the system clock source.
•   Set oscillator control register, CLKCB4 (bit 4) to 1 and CLKCB5 (bit 5) to 0 to switch the system clock source to "CF oscillator (main)."

Note 1: Do not switch the amplifier size of the CF oscillator when the system clock is set to "CF oscillator (main)." Switching the amplifier size in this case may cause unstable oscillation, resulting in a system malfunction.

Note 2: The operating voltage range differs for the CF oscillator low and normal size amplifiers. Refer to the pertinent Semiconductor Data Sheet document before using the low size CF oscillator amplifier.

4.3 CF Oscillation (Main Clock) Monitoring Function

4.3.1 Overview

The CF oscillation monitor function checks the CF oscillator circuit for normal oscillation when the microcontroller switches the system clock source to CF oscillation for the main clock. This precludes system deadlock and other system malfunctions from being incurred by any abnormalities that occur in the CF oscillator circuit.

4.3.2 Functions

1) Main clock oscillation counter
   - Is a 9-bit binary counter used to monitor the operating state of the CF oscillator circuit.
2) CF oscillation monitor register
   - Used to start and stop CF oscillation monitoring and to check the operating state of the oscillator circuit.
3) To control the CF oscillation monitoring function, it is necessary to manipulate the following special function register:
   - CFLVM

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE57	HHH0 HH00	R/W	CFLVM	-	-	-	CFMON	-	-	FIX0	FIX0

4.3.3 Circuit Configuration

The CF oscillation monitor circuit consists of a 9-bit binary counter for monitoring CF oscillation and the CF oscillation monitor register (CFLVM) that controls the binary counter. When the monitor bit of the CF oscillation monitor register is set, the 9-bit counter starts counting the number of CF oscillation clocks. As CF oscillation continues normally, an overflow eventually occurs in the counter, which resets the monitor bit, indicating that oscillation is continuing normally.

4.3.4 Related Register

4.3.4.1 CF oscillation monitor register(CFLVM)　(3-bit register)

1) The CF oscillation monitor register is a 3-bit register that is used to control CF oscillation monitoring operation.

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE57	HHH0 HH00	R/W	CFLVM	-	-	-	CFMON	-	-	FIX0	FIX0

(bits 7 to 5, 3, 2): These bits do not exist. They are read as 1s.

CF Oscillation Monitor

CFMON (bit 4): CF oscillation monitoring control

Setting this bit to 1 starts monitoring CF oscillation. This bit is eventually reset to 0 if the CF oscillation continues normally. This bit must be reset to 0 to stop monitoring CF oscillation. The period during which CF oscillation clocks are to be counted is calculated as follows:

CF monitoring count time= Source oscillation period $\times$ 512

FIX0 (bit 1): Fixed bit

Must always be set to 0.

FIX0 (bit 0): Fixed bit

Must always be set to 0.

4.3.5   CF Oscillation Monitoring Example

1)   At power-on time, system reset time, or exit from the HOLD mode
   • Switch the system clock to internal RC oscillation and start monitoring.

2)   Oscillation start time of the CF oscillator circuit
   • Wait for several to several scores of milliseconds until the CF oscillator circuit for the main clock starts oscillation stably.

3)   Configuring for the initiation of CF oscillation monitoring and polling
   • Set the CFMON bit (bit 4) of the CF oscillation monitor register (CFLVM) to 1.
   • Poll the CFMON bit (bit 4); it will be reset to 0 in source oscillation period $\times$ 512 if oscillation is continuing normally.
   • It is recommended that step 3) be repeated several times even when normal oscillation is once confirmed. If the confirmation of normal oscillation fails, the application in the set unit should recognize this condition as an oscillation error and take error recovery actions including error handling processing and continuation of step 3).

4)   Switching the system clock source to CF oscillation for the main clock.

* Proceed with the next processing by the application.

4.4   Standby Function

4.4.1   Overview

This series of microcontrollers supports three standby modes, called the HALT, HOLD, and X'tal HOLD modes, that are used to reduce current consumption at power-failure time or in program standby mode. In a standby mode, the execution of all instructions is suspended.

4.4.2   Functions

1)   HALT mode

- The microcontroller suspend the execution of instructions but its peripheral circuits continue processing.
- The HALT mode is entered by setting bit 0 of the PCON register to 1.
- Bit 0 of the PCON register is cleared and the microcontroller returns to the normal operating mode when a reset occurs or an interrupt request is accepted.

2)   HOLD mode

- All oscillations are suspended. The microcontroller suspend the execution of instructions and its peripheral circuits stop processing.
- The HOLD mode is entered by setting bit 1 of the PCON register to 1 when bit 2 is set to 0. In this case, bit 0 of the PCON register (HALT mode flag) is automatically set.
- When a reset occurs or a HOLD mode resetting signal (INT0, INT1, INT2, INT4, INT5, orP0INT) occurs, bit 1 of the PCON register is cleared and the microcontroller switches into the HALT mode.

3)   X'tal HOLD mode

- All oscillations except the subclock oscillation are suspended. The microcontroller suspend the execution of instructions and all the peripheral circuits except the base timer stop processing.
- The X'tal HOLD mode is entered by setting bit 1 of the PCON register to 1 when bit 2 is set to 1. In this case, bit 0 of the PCON register (HALT mode flag) is automatically set.
- When a reset occurs or a HOLD mode resetting signal (base timer interrupt, remote controller receiver interrupt, INT0, INT1, INT2, INT4, INT5, or P0INT) occurs, bit 1 of the PCON register is cleared and the microcontroller switches into the HALT mode.

Note: Do not allow the microcontroller to enter into the HALT, HOLD, or X'tal HOLD mode while AD conversion is in progress. Make sure that ADSTART is set to 0 before placing the microcontroller into one of the above-mentioned standby modes.

4.4.3 Related Registers

4.4.3.1 Power Control Register (PCON) (3-bit register)

1) The power control register is a 3-bit register that specifies the operating mode (normal/HALT/HOLD/ X'tal HOLD).

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE07	HHHH H000	R/W	PCON	-	-	-	-	-	XTIDLE	PDN	IDLE

(bits 7 to 3): These bits do not exist. They are always read as "1."

XTIDLE (bit 2): X'tal HOLD mode setting flag

PDN (bit 1): HOLD mode setting flag

XTIDLE	PDN	Operating mode
−	0	Normal or HALT mode
0	1	HOLD mode
1	1	X'tal HOLD mode

1) These bits must be set with an instruction.

- When the microcontroller enters the HOLD mode, all oscillations (main clock, subclock, and low-/medium-speed RC) are suspended and the related registers are set as described below.

  If SLWRC register, bit 1 is set to 1, SLWRC register, bit 0 is set and OCR register, bits 4 and 5 are cleared.

  If SLWRC register, bit 1 is set to 0, OCR register bits 1, 4, and 5 are cleared.

- When the microcontroller exits the HOLD mode, low- or medium-speed RC oscillator resumes oscillation and is designated as the system clock source according to the state of SLWRC and OCR registers. The main clock and the subclock oscillator return to their states that are established before the HOLD mode is entered.

- When the microcontroller enters the X'tal HOLD mode, all oscillations except XT (i.e., main clock and low-/medium-speed RC) are suspended but the state of the OCR register remains unchanged.

- Since the X'tal HOLD mode is used usually for low-current clock counting or infrared remote controller wait control, less current will be consumed if the system clock is switched to the subclock and RC oscillation is suspended before the X'tal HOLD mode is entered.

2) XTIDLE must be cleared with an instruction.

3) PDN is cleared when a HOLD mode resetting signal (INT0, INT1, INT2, INT4, INT5, or P0INT) or a reset occurs.

4) Bit 0 is automatically set when PDN is set.

IDLE (bit 0): HALT mode setting flag

1) Setting this bit places the microcontroller into the HALT mode.

2) When bit 1 is set, this bit is also set.

3) This bit is cleared on acceptance of an interrupt request or on receipt of a reset signal.

Table 4.4.1   Standby Mode Operations

Item/mode	Reset State	HALT Mode	HOLD Mode	X'tal HOLD Mode
Entry conditions	• RES applied • Reset from watchdog timer	PCON register Bit 1=0 Bit 0=1	PCON register Bit 2=0 Bit 1=1	PCON register Bit 2=1 Bit 1=1
Data changed on entry	Initialized as shown in separate table.	WDT bits 2 to 0 are cleared if WDT register (FE0F), bit 4 is set.	• WDT bits 2 to 0 are cleared if WDT register (FE0F), bit 4 is set. • PCON, bit 0 turns to 1. • If SLWRC register (FE7C), bit 1 is reset OCR register (FE0E), bits 5, 4, and 1 are cleared • If SLWRC register (FE7C), bit 1 is set SLWRC register (FE7C), bit 0 is set and OCR register (FE0E), bits 5 and 4 are cleared.	• WDT bits 2 to 0 are cleared if WDT register (FE0F), bit 4 is set. • PCON, bit 0 turns to 1.
Main clock oscillation	Stopped	State established at entry time	Stopped	Stopped
Internal low-speed RC oscillation	Stopped	State established at entry time	Stopped	Stopped
Internal medium-speed RC oscillation	Running	State established at entry time	Stopped	Stopped
Subclock oscillation	Stopped	State established at entry time	Stopped	State established at entry time
Multifrequency RC oscillation	Stopped	State established at entry time	Stopped	Stopped
CPU	Initialized	Stopped	Stopped	Stopped
I/O pin state	See Table 4.4.2.	←	←	←
RAM	• RES: Unpredictable • When watchdog timer reset: Data preserved	Data preserved	Data preserved	Data preserved
Base timer and remote controller receiver circuit	Stopped	State established at entry time	Stopped	State established at entry time
Peripheral modules except base timer and remote controller receiver circuit	Stopped	State established at entry time (Note 2)	Stopped	Stopped
Exit conditions	Entry conditions canceled.	• Interrupt request accepted. • Reset/entry conditions established	• Interrupt request from INT0 to INT2, INT4, INT5 or P0INT • Reset/entry conditions established	• Interrupt request from INT0 to INT2, INT4, INT5, P0INT, base timer, or remote controller receiver circuit • Reset/entry conditions established
Returned mode	Normal mode	Normal mode (Note1)	HALT (Note1)	HALT (Note1)
Data changed on exit	None	PCON register, bit 0＝0	PCON register, bit 1＝0	PCON register, bit 1＝0

Note 1: The microcontroller switches into the reset state if it exits the current mode on the establishment of reset/entry conditions.

Note 2: Some serial transmission functions are disabled.

Standby

Table 4.4.2   Pin States and Operating Modes (LC872H00 series)

Pin Name	Reset Time	Normal Mode	HALT Mode	HOLD Mode	On Exit from HOLD
$\overline{\text{RES}}$	• Input	←	←	←	←
CF1 /XT1	• Inverter input for CF oscillation • Oscillation not started • Feedback resistor present between CF1 and CF2.	• CF oscillation inverter input/general-purpose input selected by bit 3 of register XT2PC (FE43H). • Oscillation enabled or disabled by register OCR (FE0EH). • Feedback resistor between CF1 and CF2 controlled by a program.	←	• CF oscillation inverter input/general-purpose input in setting established at entry time. • Feedback resistor between CF1 and CF2 in state established on entry to HOLD mode	• State established on entry into HOLD mode
CF2 /XT2	• Inverter output for CF oscillation • Oscillation not started • Feedback resistor present between CF1 and CF2. • VDD level present regardless of CF1 state.	• CF oscillation inverter output/general-purpose input selected by bit 3 of register XT2PC (FE43H). • Oscillation enabled or disabled by register OCR (FE0EH). • Feedback resistor between CF1 and CF2 controlled by program.	←	• CF oscillation inverter output/general-purpose input in setting established on entry into HOLD mode. • Feedback resistor between CF1 and CF2 in state established on entry to HOLD mode	• State established on entry into HOLD mode
P00-P07	• Input mode • Pull-up resistor off	• Input/output/pull-up resistor controlled by a program	←	←	• Same as in normal mode.
P10-P17	• Input mode • Pull-up resistor off	• Input/output/pull-up resistor controlled by a program	←	←	←
P20-P21	• Input mode • Pull-up resistor off	• Input/output/pull-up resistor controlled by a program	←	←	←
P30-P31	• Input mode • Pull-up resistor off	• Input/output/pull-up resistor controlled by a program	←	←	←
P70	• Input mode • Pull-up resistor off	• Input/output/pull-up resistor controlled by a program • Watchdog timer Nch output transistor controlled by a program (1920 to 2048 Tcyc required to go off since on-time is automatically extended).	• Input mode • Pull-up resistor off • Watchdog timer Nch output transistor is off (automatic on-time extension function is reset).	←	• Same as in normal mode.
P71-P73	• Input mode • Pull-up resistor off	• Input/output/pull-up resistor controlled by a program	←	←	←

- All modes

- Reset state entry conditions
- Low level applied to RES pin.
- Reset signal generated by watchdog timer

- Reset
  - Main clock stopped in reset state, started after reset state is released.
  - Subclock stopped
  - Low-speed RC oscillator stopped
  - Medium-speed RC oscillator started
  - Multifrequency RC oscillator stopped.
  - All registers initialized

- Reset state cancellation conditions
- Lapse of predetermined time after resetting conditions are reset.

- HOLD mode entry conditions
- PCON register (FE07H), bit 2 set to 0 and bit 1 to 1

- X'tal HOLD mode entry conditions
- PCON register (FE07H), bit 2 set to 1 and bit 1 to 1

- HOLD mode
  - All oscillators stopped
  - If SLWRC register, bit 1 is set to 1:
    Since SLWRC register, bit 0 is set and OCR register, bits 4 and 5 are cleared, low-speed RC oscillator starts oscillation as system clock when CPU exits HOLD mode
  - If SLWRC register, bit 1 is set to 0:
    Since OCR register, bits 1, 4, and 5 are cleared, medium-speed RC oscillator starts oscillation as system clock when CPU exits HOLD mode.
  - Since MRCR register, bits 6 and 7 are cleared, the multifrequency RC oscillator is held stopped even after the this mode is exited
  - CPU and peripheral modules are stopped.

- Normal operating mode
  - Start/stop of oscillators programmable
  - CPU and peripheral modules run normally.

- X'tal HOLD
  - Main clock, low-/medium-speed RC, and multifrequency RC oscillators stopped. Subclock retains the state established when X'tal HOLD mode is entered.
  - Contents of OCR, SLWRC, and MRCR registers remain unchanged
  - CPU enters this mode after selecting subclock or low- or medium-speed RC oscillator clock as system clock.
  - CPU and all peripheral modules except base timer and remote controller receiver circuit stop operation and base timer and remote controller receiver circuit retains the state established when X'tal HOLD mode is entered.
  - When X'tal HOLD mode is exited, the oscillators return to the state established when the mode is entered

- HALT mode
  - All oscillators retain the state established when the HALT mode is entered.
  - CPU stopped. Peripheral modules keep running.

- HOLD mode resetting conditions
- INT0 or INT1 level interrupt request generated
- Request for INT2, INT4, INT5 or port 0 interrupt generated
- Resetting conditions established (Note 1)

- X'tal HOLD mode resetting conditions
- Base timer or remote controller receiver circuit interrupt request generated.
- INT0 or INT1 level interrupt request generated.
- Request for INT2, INT4, INT5, or port 0 interrupt generated
- Resetting condition established (Note 1)

- HALT mode entry conditions
- PCON register (FE07H), bit 1 set to 0 and bit 0 to 1

- HALT mode resetting conditions
- Interrupt request accepted (Note 2)
- Resetting conditions established (Note 1)

Note 1: The CPU enters the reset state when the resetting conditions are established.
Note 2: The CPU cannot return from the HALT mode since no interrupt request can be accepted unless its interrupt level is higher than the interrupt level that placed the CPU into the HALT, HOLD, or X'tal HOLD mode.

Interrupt level at which the CPU entered HALT, HOLD, or X'tal HOLD mode	Interrupt request level that can reset HALT mode
No interrupt request present	X, H and L levels
L level	X and H levels
H level	X level
X level	None (unable to reset with interrupt)

Fig. 4.4.1   Standby Mode State Transition Diagram

4.5   Reset Function

4.5.1   Overview

The reset function initializes the microcontroller when it is powered on or while it is running.

4.5.2   Functions

This series of microcontrollers provides the following three types of resetting function:

1)   External reset via the $\overline{RES}$ pin

The microcontroller is reset without fail by applying and holding a low level to the $\overline{RES}$ pin for 200 μs or longer. Note, however, that a low level of a small duration (less than 200 μs) is likely to trigger a reset.

The $\overline{RES}$ pin can serve as a power-on reset pin when it is provided with an external time constant element.

2)   Internal reset

The internal reset function is available in two types: the power-on reset (POR) that triggers a reset when power is turned on and the low-voltage detection reset (LVD) that triggers a reset when the power voltage falls below a certain level. Options are available to set the power-on reset resetting level, to enable and disable the low-voltage detection reset function, and its threshold level.

3)   Runaway detection/reset function using a watchdog timer

The watchdog timer of this series of microcontrollers can be used to detect and reset runaway conditions by connecting a resistor and a capacitor to its external interrupt pin (P70/INT0/T0LCP) and making an appropriate time constant element.

An example of a resetting circuit is shown in Figure 4.5.1. The external circuit connected to the reset pin shows an example that the internal reset function is disabled and an external power-on reset circuit is configured.
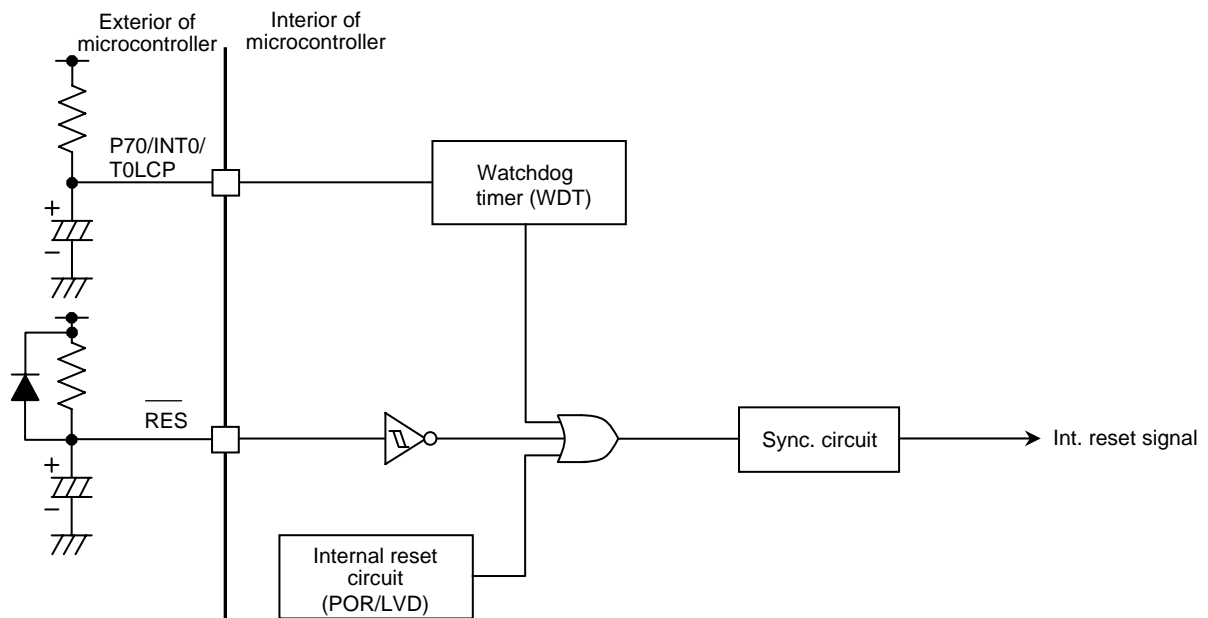
Figure 4.5.1   Sample Reset Circuit Block Diagram

4.5.3   Reset State

When a reset is generated by the $\overline{\text{RES}}$ pin, internal reset circuit, or watchdog timer, the hardware functional blocks of the microcontroller are initialized by a reset signal that is in synchronization with the system clock.

Since the system clock is switched to the internal medium speed RC oscillator when a reset occurs, hardware initialization is also carried out immediately even at power-on time. The system clock must be switched to the main clock when the main clock gets stabilized. The program counter is initialized to 0000H on a reset. See Appendix (AI), 87 Register Map, for the initial values of the special function registers (SFR).

<Notes and precautions>

- *The stack pointer is initialized to 0000H.*

- *Data RAM is never initialized by a reset. Consequently, the contents of RAM are unpredictable at power-on time.*

- *When using the internal reset function, it is necessary to implement and connect an external circuit to the reset pin according the user's operating environment. Be sure to review and observe the operating specifications, circuit configuration, precautions, and considerations discussed in section 4.7, "Internal Reset Function."*

4.6   Watchdog Timer Function

4.6.1   Overview

This series of microcontrollers incorporates a watchdog timer that, with an external RC circuit, detects program runaway conditions.

The watchdog timer charges the external RC circuit that is connected to the P70/INT0/T0LCP pin and, when the level at the pin reaches the high level, triggers a reset or interrupt, regarding that a program runaway occurred.

4.6.2   Functions

1)   Detection of a runaway condition

A program that discharges the RC circuit periodically needs to be prepared. If such a program hangs, it will not execute instructions that discharge the RC circuit. This causes the P potential at the P70/INT0/T0LCP pin to the high level, setting the runaway detect flag.

2)   Actions to be taken following the detection of a runaway condition

The microcontroller can take one of the following actions when the watchdog timer detects a runaway condition:

- Reset (program reexecution)
- External interrupt INT0 (program continuation)

    The priority of the external interrupt INT0 can be changed using the master interrupt enable control register (IE).

4.6.3   Circuit Configuration

The watchdog timer is made up of a high-threshold buffer, a pulse stretcher circuit, and a watchdog timer control register. Its configuration diagram is shown in Figure 4.6.1.

- High threshold buffer

The high-threshold buffer detects the charging voltage of the external capacitor.

- Pulse stretcher circuit

The pulse stretcher circuit discharges the external capacitor for longer than the specified time to ensure reliable discharging. The stretching time is from 1,920 to 2,048 Tcyc.

- Watchdog timer control register (WDT)

    The watchdog timer control register controls the operation of the watchdog timer.
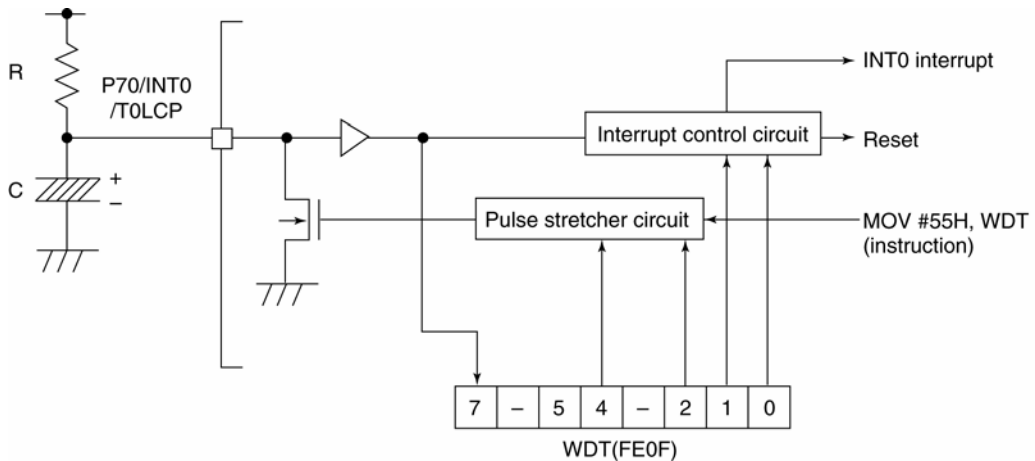
Fig. 4.6.1   Watchdog Timer Circuit

4.6.4   Related Registers

1)   Watchdog timer control register (WDT)

Address	Initial value	R/W	Name	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE0F	0H00 H000	R/W	WDT	WDTFLG	-	WDTB5	WDTHLT	-	WDTCLR	WDTRST	WDTRUN

Bit Name	Function
WDTFLG (bit 7)	Runaway detection flag
	0: No runaway
	1: runaway
WDTB5 (bit 5)	General-purpose flag
	Can be used as a general-purpose flag.
WDTHLT (bit 4)	HALT/HOLD mode function control
	0: Enables the watchdog timer.
	1: Disables the watchdog timer.
WDTCLR (bit 2)	Watchdog timer clear control
	0: Disables the watchdog timer for clearing.
	1: Enables the watchdog timer for clearing.
WDTRST (bit 1)	Runaway-time reset control
	0: Suppresses resetting on a runaway condition.
	1: Triggers a reset on a runway condition.
WDTRUN (bit 0)	Watchdog timer operation control
	0: Maintains watchdog timer operating state.
	1: Starts watchdog timer operation.

WDTFLG (bit 7): Runaway detection flag

This bit is set when a runaway condition is detected by the watchdog timer. The application can identify the occurrence of a runaway condition by monitoring this bit (provided that WDTRST is set to 1).

This bit is not reset automatically. It must be reset with an instruction.

Watchdog timer

WDTB5 (bit 5): General-purpose flag

This bit can be used as a general-purpose flag. Manipulating this bit exert no influence on the operation of the functional block.

WDTHLT (bit 4): HALT/HOLD mode function control

This bit enables (0) or disables (1) the watchdog timer when the microcontroller is in the HALT or HOLD state. When this bit is set to "1," WDT2 to WDT0 are reset and the watchdog timer is stopped in the HALT or HOLD state. When this bit is set to "0," WDT2 to WDT0 remain unchanged and the watchdog timer continues operation even when the microcontroller enters the HALT or HOLD state.

WDTCLR (bit 2): Watchdog timer clear control

This bit enables (1) or disables (0) the discharge of capacitance from the external capacitor. Setting the bit to "1" drives the pin P70/INT0/T0LCP N-channel transistors, discharging the external capacitors and clearing the watchdog timer. The pulse stretcher also functions during this process. Setting the bit to "0" disables operation of the N-channel transistors and the clearing of the watchdog timer.

WDTRST (bit 1): Runaway-time reset control

This bit enables (1) or disables (0) the watchdog timer from triggering a reset when it detects a program hangup. When this bit set to "1," a reset is generated and execution restarts at program address 0000H when a program hangup is detected. When the bit is set to "0," no reset occurs when a program hangup is detected. Instead, an external interrupt INT0 is generated and a call is made to vector address 0003H.

WDTRUN (bit 0): Watchdog timer operation control

This bit starts (1) or maintains the state of (0) the watchdog timer. A "1" in this bit starts the watchdog timer function and a "0" exerts no influence on the operation of the watchdog timer. This means, that once the watchdog timer is started, a program will not be able to stop the watchdog timer (stopped by a reset).

Caution!

If WDTRST is set to 1, a reset is triggered when INT0 is set to 1 even if the watchdog timer is inactive. The N-channel transistor at pin P70/INT0/T0LCP is turned on if the watchdog timer is stopped (WDTRUN = 0) by setting the watchdog timer clear control bit (WDTCLR) to "1." Keep this in mind when programming if the watchdog timer function is not to be used. More current than usual may be consumed depending on the program or application circuit.

---

2) Master interrupt enable control register (IE)
   See subsubsection 4.1.4.1, "Master interrupt Enable Control Register," for details.

3) Port 7 control register (P7)
   See subsubsection 3.5.3.1, "Port 7 Control Register," for details.

4.6.5 Using the Watchdog Timer

Code a program so that instructions for clearing the watchdog timer periodically are executed. Select the resistance R and the capacitance C such that the time constant of the external RC circuit is greater than the time interval required to clear the watchdog timer.

1) Initializing the watchdog timer

All bits of the watchdog timer control register (WDT) are reset when a reset occurs. If the P70/INT0/T0LCP pin has been charged up to the high level, discharge it down to the low level before starting the watchdog timer. The built-in N-channel transistor is used for discharging. Since it has an on resistance, a discharging time equal to the time constant of the external capacitance is required.

Set bits 0 and 4 of the P7 mode register P7 (FE5C) to 0, 0 or 1, 1 to make the P7 port output open.

Starting discharge

Load WDT with "04H" to turn on the N-channel transistor at the P70/INT0/T0LCP pin to start discharging the capacitor.

Checking the low level

Checking for data at the P70/INT0/T0LCP pin

Read the data at the P70/INT0/T0LCP pin with a LD or similar instruction. A "0" indicates that the P70/INT0/T0LCP pin is at the low level.

2) Starting the watchdog timer

(1) Set bit 2(WDTCLR) and bit 0 (WDTRUN) to "1."

(2) Also set bit 1 (WDTRST) to "1" when a reset is to be triggered when a runaway condition is detected.

(3) To suspend the operation of the watchdog timer in the HOLD or HALT mode, set bit 4 (WDTHLT) at the same time.

The watchdog timer starts functioning when bit 0 (WDTRUN) is set to "1." Once the watchdog timer starts operation, WDT is disabled for write; it is allowed only to clear the watchdog timer and WDT read. Consequently, the watchdog timer can never be stopped with an instruction. The function of the watchdog timer is stopped only when a reset occurs or when the microcontroller enters the HALT or HOLD mode with WDTHLT being set. In this case, bits WDT2 to WDT0 are reset.

3) Clearing the watchdog timer

When the watchdog timer starts operation, the external RC circuit connected to the P70/INT0/T0LCP pin is charged. When voltage at this pin reaches the high level, a reset or interrupt is generated as specified in the watchdog timer control register (WDT). To run the program in the normal mode, it is necessary to periodically discharge the RC circuit before the voltage at the P70/INT0/T0LCP pin reaches the high level (clearing the watchdog timer). Execute the following instruction to clear the watchdog timer while it is running:

MOV #55H,WDT

This instruction turns on the N-channel transistor at the P70/INT0/T0LCP pin. Owing to the pulse stretcher function (keeps the transistor on after the MOV instruction is executed), the capacitor keeps discharging for a period from a minimum of 1,920 cycle times to a maximum of 2,048 cycle times.

4) Detecting a runaway condition

Unless the above-mentioned instruction is executed, the RC circuit keeps charging because the watchdog timer is not cleared. As charging proceeds and the voltage at the P70/INT0/T0LCP pin reaches the high level, the watchdog timer considers that a program hangup has occurred and triggers a reset or interrupt. In this case, the runaway detection flag WDTFLG is set.

If WDTRST is found to be "1" in this case, a reset occurs and execution restarts at address 0000H. If WDTRST is "0," an external interrupt (INT0) is generated and control is transferred to vector address 0003H.

● **Hints on Use**

1) To realize ultra-low-power operation using the HOLD mode, it is necessary not to use the watchdog timer at all or to disable the watchdog timer from running in the HOLD mode by setting WDTHLT to "1."

Be sure to set WDTCLR to "0" when the watchdog timer is not to be used.

2) The P70/INT0/T0LCP pin has two input levels. The threshold level of the input pins of the watchdog timer circuit is higher than that of the port inputs and the interrupt detection level.

Refer to the latest "Semiconductor News" for the input levels.
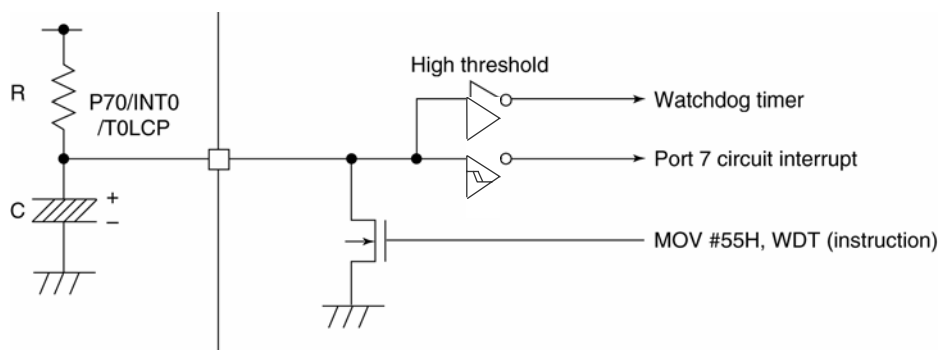
Fig. 4.6.2   P70/INT0/T0LCP Pin (without an Optional Pull-up Resistor)

3) The external resistor to be connected to the watchdog timer can be omitted by setting bits 4 and 0 of the P7 register (FE5C) to 0, 1 and connecting a <u>pull-up resistor</u> to the P70/INT0/T0LCP pin (see Figure 4.6.3).

The resistance of the pull-up resistor to be adopted in this case varies according to the power source voltage VDD. Calculate the time constant of the watchdog timer while referring to the latest "Semiconductor News."
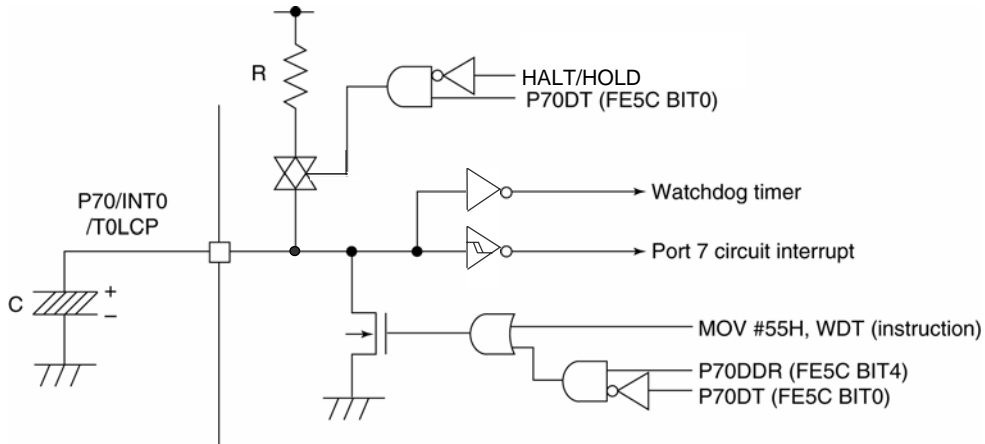
Fig. 4.6.3   Sample Application Circuit with a Pull-up Resistor

4.7 Internal Reset Function

4.7.1 Overview

This series of microcontroller incorporates internal reset functions called the power-on reset (POR) and low voltage detection reset (LVD). The use of these functions will contribute to the reduction in the number of externally required reset circuit components (reset IC, etc.).

4.7.2 Functions

1) Power-on reset (POR) function

POR is a hardware feature that generates a reset to the microcontroller at power-on time. This function allows the user to select the POR release level by option only when the disuse of the low voltage detection reset function is selected. It is necessary to use the undermentioned low voltage detection reset function together with this function, or configure an external reset circuit if there are possibilities that chatter can occur or a momentary power loss occur at power-on time.

2) Low voltage detection reset (LVD) function

This function, when used together with the POR function, can generate a reset when power is turned on and when the power level lowers. As a user option the use or disuse and the detection level of this function can be specified.

4.7.3 Circuit Configuration

The internal reset circuit consists of POR, LVD, pulse stretcher circuit, capacitor C_{RES} discharging transistor, external capacitor C_{RES} +pull-up resistor R_{RES} or pull-up resistor R_{RES} alone. The circuit diagram of the internal reset circuit is given in Subsection 4.7.1.

• Pulse stretcher circuit

The pulse stretcher circuit stretches the POR and LVD reset signals. It is used to stretch the internal reset period and discharge the external capacitor C_{RES} connected to the RESET pin. The stretching time is from 30μs to 100μs.

• Capacitor CRES discharging transistor

This is an N-channel transistor used to discharge the external capacitor C_{RES} connected to the RESET pin. If the capacitor C_{RES} is not to be connected to the RESET pin, it is possible to monitor the internal reset signal by connecting only the external pull-up resistor R_{RES}.

• Option selector circuit

The option selector circuit is used to configure the LVD options. This circuit selects the enable or disable of LVD and its detection level. See Subsection 4.7.4.

• External capacitor CRES +Pull-up resistor RRES

After the reset signal from the internal reset circuit is released, the reset period is further stretched according to the external CR time constant. This enables the microcontroller to avoid the repetitive entries and releases of the reset state from occurring when the power-on chatter occurs. The circuit configuration shown in Figure 4.7.1, in which the capacitor C_{RES} and pull-up resistor R_{RES} are externally connected, is recommended when both POR and LVD functions are to be used. The recommended constant values are: C_{RES} =0.022 μ F and R_{RES} =510 k Ω. The external pull-up resistor R_{RES} must always be installed even when the set's specifications inhibit the installation of the external capacitor C_{RES}.
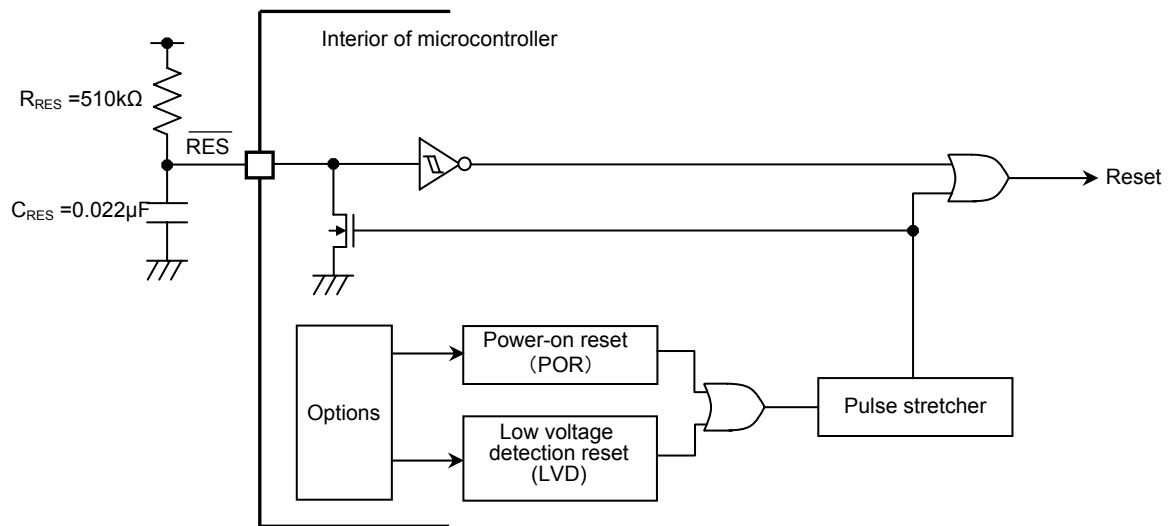
Figure 4.7.1 Internal Reset Circuit Configuration

4.7.4 Options

The POR and LVD options are available for the reset circuit.

1) LVD Reset Function Options			
"Enable": Use		"Disable": Not Used	
2) LVD Reset Level Option		3) POR Release Level Option	
Typical Value of Selected Option	Min. Operating VDD Value (*)	Typical Value of Selected Option	Min. Operating VDD Value (*)
—	—	"1.67V"	1.8V to
"1.91V"	2.1V to	"1.97V"	2.1V to
"2.01V"	2.2V to	"2.07V"	2.2V to
"2.31V"	2.5V to	"2.37V"	2.5V to
"2.51V"	2.7V to	"2.57V"	2.7V to
"2.81V"	3.0V to	"2.87V"	3.0V to
"3.79V"	4.0V to	"3.86V"	4.0V to
"4.28V"	4.5V to	"4.35V"	4.5V to

*: The minimum operating VDD value specifies the approximate lower limit value of the VDD value beyond which the selected POR release level or LVD reset level can be effected without generating a reset.

1) LVD reset function option

   When the LVD reset function is enabled, a reset is generated at the voltage that is selected by the LVD reset level option.

   Note 1: In this configuration, an operating current of several μA always flows in all modes.

   No LVD reset is generated when "Disable" is selected.

   Note 2: In this configuration, no operating current will flow in all modes.

   * See the sample operating waveforms of the reset circuit shown in Subsection 4.7.5 for details.

2) LVD reset level option

   The LVD reset level can be selected from 7 level values only when the LVD reset function is enabled. Select the appropriate detection level according to the user's operating conditions.

3) POR release level option

   The POR release level can be selected out of 8 levels only when the LVD reset function is disabled. When not using the internal reset circuit, set the POR release level to the lowest level (1.67V) that will not affect the minimum guaranteed operating voltage.

   Note 3: No operating current flows when the POR reset state is released.

   Note 4: See the notes in paragraph 2) of Subsection 4.7.6 when selecting a POR release level that is lower than the minimum guaranteed operating voltage (1.67V).

Internal reset

● **Selection example 1**

Selecting the optimum LVD reset level to keep the microcontroller running without resetting it until VDD falls below 2.7V according to the set's requirements

Set the LVD reset function option to "Enable" and select "2.51V" as the LVD reset level.


```
                    ↑
                    │
             Set operating range
                    │
                    ↓    VDD=2.7V
...................................... LVD release voltage (LVDET+LVHYS)
─·─·─·─·─·─·─·─·─·─·─·─·─·─·─· LVD reset voltage (LVDET=Typ. 2.51V)
```
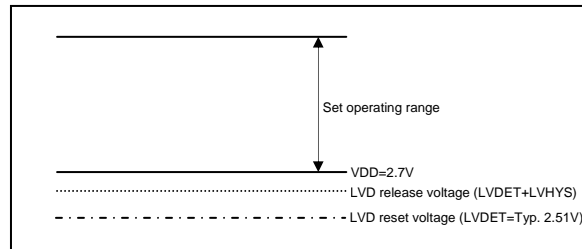
● **Selection example 2**

Selecting the optimum LVD reset level that meets the guaranteed operating conditions of VDD = 2.7V/tCYC = 250 ns

Set the LVD reset function option to "Enable" and select "2.81V" as the LVD reset level option.


```
                    ↑
                    │
        Microcontroller guaranteed operating range
        (VDD=2.7V to 5.5V/tCYC=250ns)
                    │
...................................... LVD release voltage (LVDET+LVHYS)
─·─·─·─·─·─·─·─·─·─·─·─·─·─·─· LVD reset voltage (LVDET=Typ. 2.81V)
                    ↓    Operation guarantee voltage
                         (VDD=2.7V / tCYC=250ns)
```
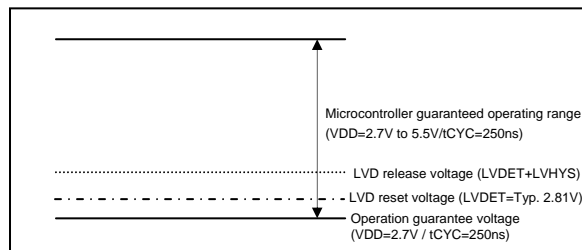
● **Selection example 3**

Disabling the internal reset circuit and using an external reset IC that can detect and react at 3.0V (see also paragraph 1) of Subsection 4.7.7)

Set the LVD reset function option to "Disable" and select "1.67V" as the POR release level option.


```
                    ↑
                    │
             Set operating range
                    │
                    ↓    VDD=3.1V
...................................... External 3.0V detection circuit



─·─·─·─·─·─·─·─·─·─·─·─·─·─·─· POR release voltage
                              (PORRL=Typ. 1.67V)
```
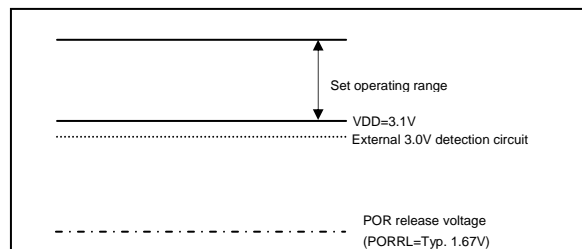
Note 5: The operation guarantee values (voltage/operating frequency) shown in the examples vary with the microcontroller type. Be sure to refer to the latest SANYO Semiconductor Datasheet.

4.7.5 Sample Operating Waveforms of the Internal Reset Circuit

1) Waveform observed when only POR is used (LVD not used)

(RESET pin: Pull-up resistor R_{RES} only)
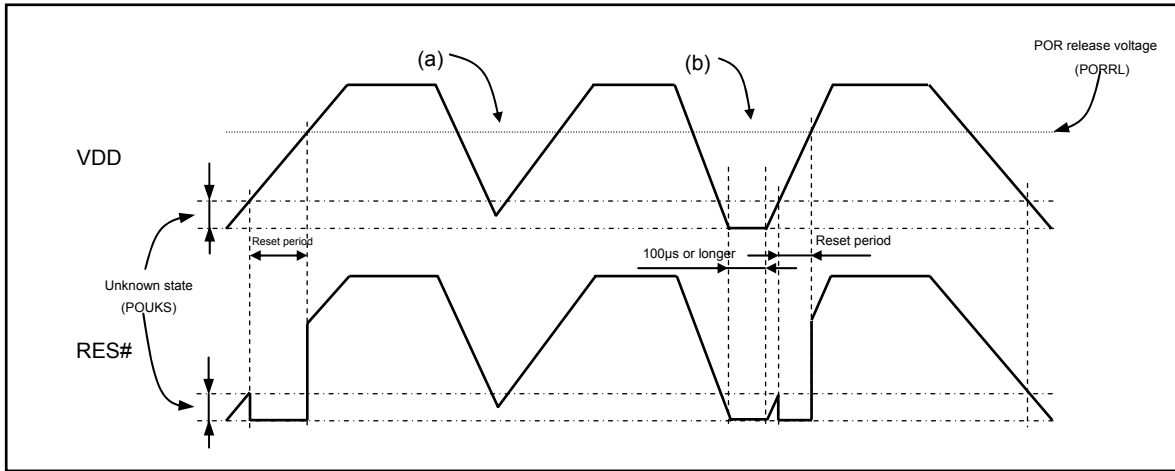
- There exists an unknown-state (LVUKS), before the POR transistor starts functioning normally.

- The POR function generates a reset only when power is turned on starting at the VSS level. The reset release voltage in this case may have some range. Refer to the latest SANYO Semiconductor Datasheet for details.

- No stable reset will be generated if power is turned on again when the power level does not go down to the VSS level as shown in (a). If such a case is anticipated, use the LVD function together with the POR function as explained in 2) or implement an external reset circuit.

- A reset is generated only when the power level goes down to the VSS level as shown in (b) and power is turned on again after this condition continues for $100\,\mu$ s or longer.

2) Waveform observed when both POR and LVD functions are used

(RESET pin: Pull-up resistor R_{RES} only)
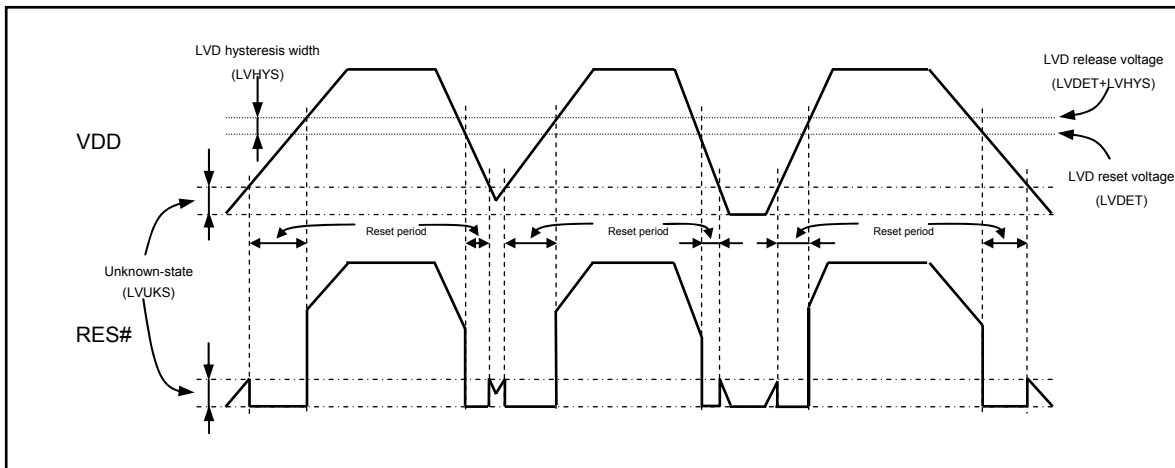
- There also exists an unknown-state (LVUKS), before the POR transistor starts functioning normally when both POR and LVD functions are used.

- Resets are generated both when power is turned on and when the power level lowers. The reset release voltage and entry voltage in this case may have some range. Refer to the latest SANYO Semiconductor Datasheet for details.

- A hysteresis width (LVHYS) is provided to prevent the repetitions of reset release and entry cycles near the detection level.

4.7.6   Notes on the Use of the Internal Reset Circuit

1) When generating resets only with the POR function

When generating resets using only the POR function, do not short the RESET pin directly to VDD as when using it with the LVD function. Be sure to use an external capacitor C_{RES} of an appropriate capacitance and a pull-up resistor R_{RES} or the pull-up resistor R_{RES} alone. Test the circuit intensively under the anticipated power supply conditions to verify that resets are reliably generated.
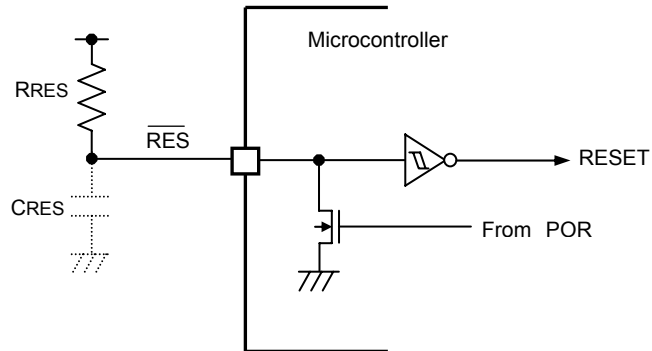
Figure 4.7.2   Reset Circuit Configuration Using only the internal POR Function

2) When selecting a release voltage level of 1.67V only with the internal POR function

When selecting an internal POR release level of 1.67V, connect the external capacitor C_{RES} and pull-up resistor R_{RES} of the values that match the power supply's rise time to the R_{ESET} pin and make necessary adjustments so that the reset state is released after the release voltage exceeds the minimum guaranteed operating voltage. Or, set and hold the voltage level of the RESET pin at the low level until the release voltage exceeds the minimum guaranteed operating voltage.
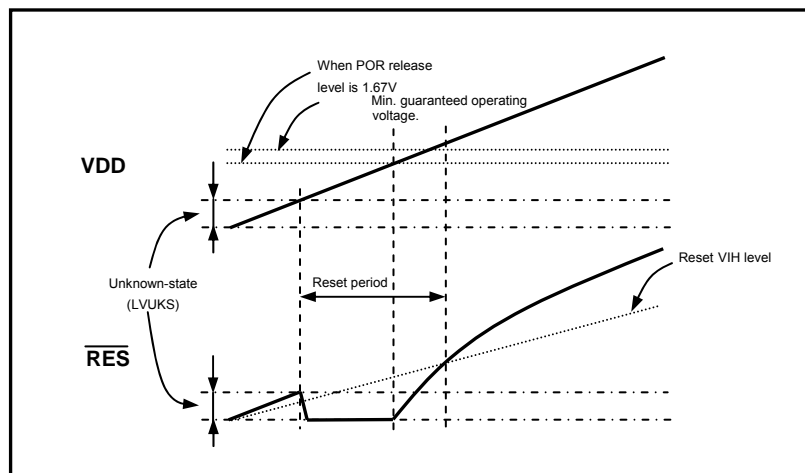
Figure 4.7.3   Sample Release Level Waveform in Internal POR Only Configuration

3) When temporary power interruption or voltage fluctuations shorter than several hundreds μs are anticipated

The response time measured from the time the LVD senses a power voltage drop at the option-selected level till it generates a reset signal is defined as the minimum low-voltage detection width TLVDW shown in Figure 4.7.4 (see SANYO Semiconductor Datasheet). If temporary power interruption or power voltage fluctuations shorter than this minimum low-voltage detection width are anticipated, be sure to take preventive measures shown in Figure 4.7.5 or other necessary measures.
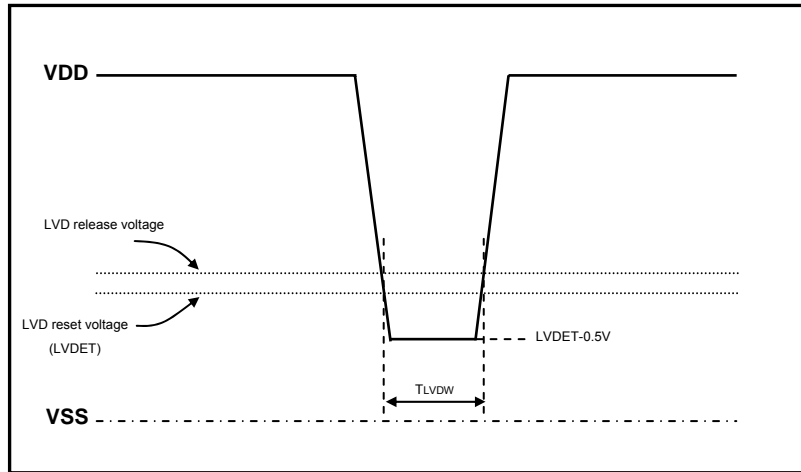
Figure 4.7.4   Example of Power Interruption or Voltage Fluctuation   Waveform
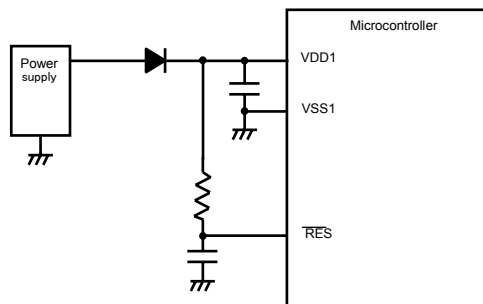
Figure 4.7.5   Example of Power Interruption/Voltage Fluctuation Countermeasures

4.7.7   Notes to be Taken When Not Using the Internal Reset Circuit

1) When configuring an external reset IC without using the internal reset circuit

The POR function is activated and the capacitor C_{RES} discharging N-channel transistor connected to the RESET pin turns on when power is turned on even if the internal reset circuit is not used. For this reason, when connecting an external reset IC, adopt the reset IC of a type whose detection level is not lower than the minimum guaranteed operating voltage level and select the lowest POR release level (1.67V) that does not affect the minimum guaranteed operating voltage. The figures given below show sample reset circuit configurations that use reset ICs of Nch open drain and CMOS types, respectively.
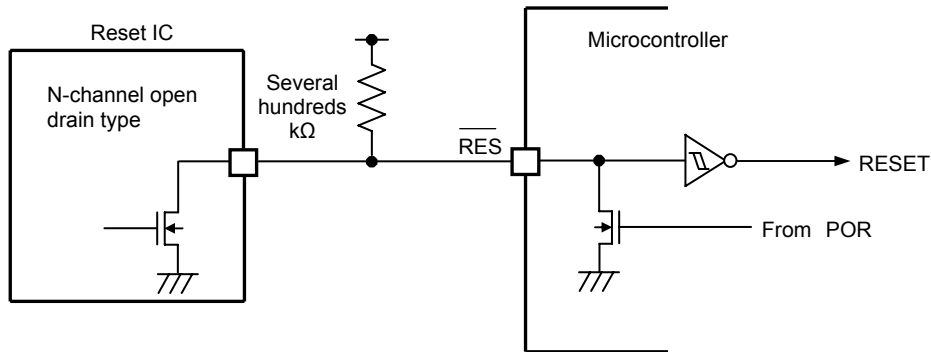
Figure 4.7.6   Sample Reset Circuit Configuration Using an N-channel Open Drain Type Reset IC
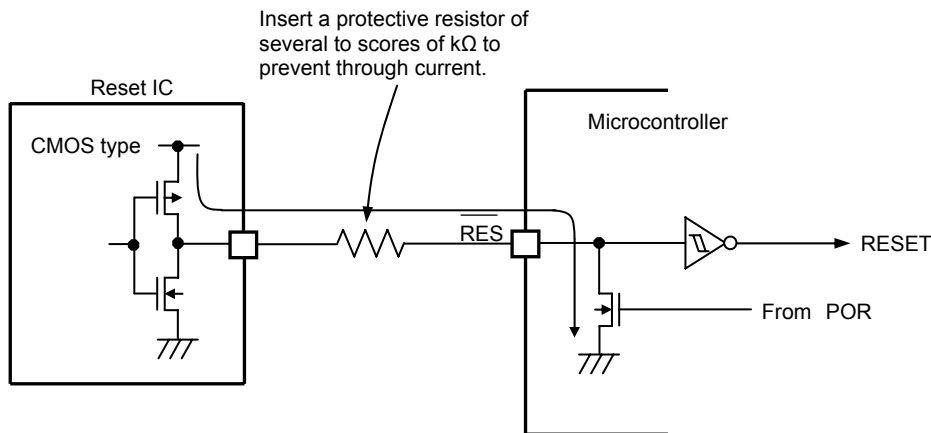
Figure 4.7.7   Sample Reset Circuit Configuration Using a CMOS Type Reset IC

2)   When configuring the external POR circuit without using the internal reset circuit

The internal POR is active at power-on time even if the internal reset circuit is not used as in the case 1) in Subsection 4.7.7. When configuring an external POR circuit with a C_{RES} value of 0.1 μ F or larger to obtain a longer reset period than with the internal POR, however, <u>be sure to connect an external diode D_{RES} as shown in Figure 4.7.8.</u>
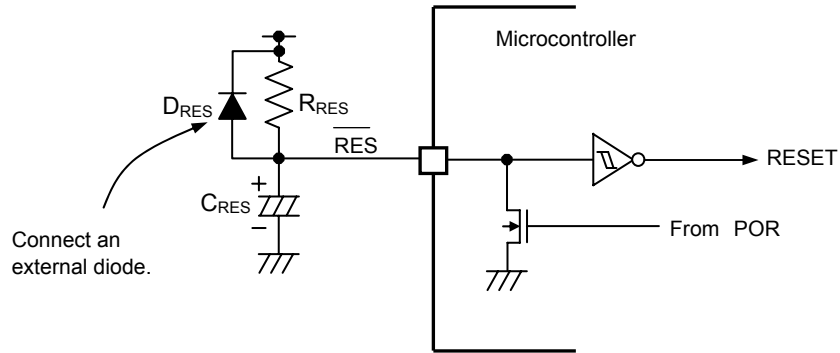
Figure 4.7.8   Sample External POR Circuit Configuration

<u>Internal reset</u>

5. Instructions

5.1 Glossary of Mnemonics

This chapter provides a description of a set of instructions available for the LC872H00 series microcontrollers. The symbols used in the individual instruction descriptions are explained below.

#:   Identifies an operand that is referred to via immediate addressing.

[ ]:   Identifies an operand that is referred to via indirect addressing.

Rn: Denotes a word (16-bit) register designating an indirect address. A maximum of 64 registers (128 bytes) starting at address 0 of RAM ($0 \leqq n \leqq 63$). A register is 16 bits long (17 bits for the LDCW instruction) whose lower-order byte is located in RAM at address $n*2$ and whose higher-order byte is located at address $n*2+1$.

dst: Identifies an operand that is referred to via direct addressing or the address itself (16 bits).

bit: Denotes a 3-bit data ($0 \leqq bit \leqq 7$) indicating a bit position

i: Denotes an 8-bit immediate data.

off: Denotes an offset data (7 bits with a sign) referenced via indirect addressing.

r8(12):   Denotes an 8-bit (12-bit) address data relative to the PC (an address apace of -128 to +127 or -2,048 to +2,047 bytes).

a17:   Denotes a PC absolute 17-bit address data. (an address space of 128K bytes).

wi: Denotes a 16-bit immediate data.

w(adr):   Denotes a word register (or RAM). Address adr represents the lower-order byte and adr + 1 represents the higher-order byte.

++( ):   Identifies a preincrement (the CPU increments the operand by 1 before executing the instruction).

( )++:   Identifies a postincrement (the CPU increments the operand by 1 after executing the instruction).

——( ):   Identifies a predecrement (the CPU decrements the operand by 1 before executing the instruction).

( )——:   Identifies a postdecrement (the CPU decrements the operand by 1 after executing the instruction).

A: Denotes the A register (accumulator, 8 bits long). Can be used with the B register as a register pair (16 bits long).

B: Denotes the B register (8 bits long), Can be used with the A register as a register pair (16 bits long).

C: Denotes the general-purpose C register (8 bits long). Used for [Rn + C(-128 to +127)] addressing.

PC: Denotes the program counter.

SP: Denotes the stack pointer.

CY: Denotes the carry flag.

AC: Denotes the auxiliary carry flag.

OV: Denotes the overflow flag.

P1:   Denotes bit 1 of the PSW (Program Status Word).

REG8:   Denotes bit 8 (9th bit) of the register to be accessed.

REGH8:   Denotes bit 8 (18th bit) of the higher-order byte of the word register to be accessed.

REGL8:   Denotes bit 8 (18th bit) of the lower-order byte of the word register to be accessed.

RAM8:   Denotes bit 8 (9th bit) of the RAM location to be accessed.

RAMH8:   Denotes bit 8 (18th bit) of the higher-order byte of the word in RAM to be accessed.

RAML8:   Denotes bit 8 (18th bit) of the lower-order byte of the word in RAM to be accessed.

ROMw((Rn)): Denotes word data in ROM. The ROM address designated by the Rn register contains the lower-order byte of the word and the contents of the Rn register plus 1 contains the higher-order byte. The Rn register that is regarded as containing a word only when accessing ROM is 17 bits long and can access a ROM space of 128K bytes (Valid only in LDCW).

ext(adr): Denotes the contents (8 bits) of external data memory designated by adr (24-bit addressing). Valid only in LDX and STX.

&: Denotes a logical AND operation.

|: Denotes a logical OR operation.

||: Denotes a logical EOR (exclusive OR) operation.

bit&&dst_H: Denotes 8-bit data generated by combining data items.(= bit $*$ 20H | dst_H where $0 \leq$ bit≤ 7, $0 \leq$ dst_H$\leq$ 1FH). Valid in bit manipulation/evaluation instructions.

5.2 Instruction Descriptions

ADD #i (ADD immediate data to accumulator)

Instruction code	[1 0 1 0 0 0 0 1][i7i6i5i4i3i2i1i0] A1H
Byte count	2
Cycle count	1
Function	(A) ← (A) + i, (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Adds immediate data (i) to the contents of the accumulator (A) and places the result in the accumulator (A).

[Example]

		A	**CY**	**AC**	**OV**
MOV	#55H, A	55H	--	--	--
ADD	#13H	68H	0	0	0
ADD	#0AH	72H	0	1	0
ADD	#0FH	81H	0	1	1
ADD	#80H	01H	1	0	1

ADD [Rn] (ADD indirect byte to accumulator)

Instruction code	[1 0 1 0 0 0 1 0][0 n5n4n3n2n1n0 0] A2H
Byte count	2
Cycle count	2
Function	(A) ← (A) + ((Rn)), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Adds the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H		RAM 0002H	RAM 0003H	A	RAM 0055H	CY	AC	OV
LDW	#0055H	--	--	55H	--	--	--	--
STW	R1	55H	00H	55H	--	--	--	--
ST	[R1]	55H	00H	55H	55H	--	--	--
ADD	[R1]	55H	00H	AAH	55H	0	0	1
ADD	[R1]	55H	00H	FFH	55H	0	0	0
ADD	[R1]	55H	00H	54H	55H	1	1	0
ADD	[R1]	55H	00H	A9H	55H	0	0	1

AD [Rn,C] (ADD indirect byte(with C register) to accumulator)

Instruction code	[1 0 1 0 0 0 1 0][0 n5n4n3n2n1n0 1] A2H
Byte count	2
Cycle count	2
Function	(A) ← (A) + ((Rn) + (C)), (PC) ←(PC) + 2
Affected flags	CY, AC, OV

[Description]

Adds the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leqq C$ reg.$\leqq 127$)

[Example]

		C	RAM 0002H	RAM 0003H	A	RAM 0059H	CY	AC	OV
R1 points to addresses 02H and 03H									
LDW	#0055H	--	--	--	--	--	--	--	--
STW	R1	--	55H	00H	55H	--	--	--	--
MOV	#04H, C	04H	55H	00H	55H	--	--	--	--
ST	[R1, C]	04H	55H	00H	55H	55H	--	--	--
ADD	[R1, C]	04H	55H	00H	AAH	55H	0	0	1
ADD	[R1, C]	04H	55H	00H	FFH	55H	0	0	0
ADD	[R1, C]	04H	55H	00H	54H	55H	1	1	0
ADD	[R1, C]	04H	55H	00H	A9H	55H	0	0	1

ADD [off] (ADD indirect byte (with displacement) to accumulator)

Instruction code	[1 0 1 0 0 0 1 0][1 off6off5off4off3off2off1off0] A2H
Byte count	2
Cycle count	2
Function	(A) ← (A) + ((R0) + off), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Adds the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect register R0 and off and places the result in the accumulator (A).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to address 00H and 01H

		RAM 0000H	RAM 0001H	A	RAM 0059H	CY	AC	OV
LDW	#0055H	--	--	55H	--	--	--	--
STW	R0	55H	00H	55H	--	--	--	--
ST	[04H]	55H	00H	55H	55H	--	--	--
ADD	[04H]	55H	00H	AAH	55H	0	0	1
ADD	[04H]	55H	00H	FFH	55H	0	0	0
ADD	[04H]	55H	00H	54H	55H	1	1	0
ADD	[04H]	55H	00H	A9H	55H	0	0	1

ADD dst (ADD direct byte to accumulator)

Instruction code	0000H≦dst＜0100H	: [A4H][dst_L(7-0)][i(7-0)]	(2bytes)	A3H-A6H
	0100H≦dst＜0200H	: [A5H][dst_L(7-0)][i(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [A3H][dst_L(7-0)][i(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [A6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [A6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2 ┊ 3			
Cycle count	1 ┊ 2			
Function	(A) ← (A) + (dst), (PC) ← (PC) + (2 or 3)			
Affected flags	CY, AC, OV			

[Description]

Adds the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0034H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	34H	55H	55H	--	--	--
ADD	34H	AAH	55H	0	0	1
ADD	34H	FFH	55H	0	0	0
ADD	34H	54H	55H	1	1	0
ADD	34H	A9H	55H	0	0	1

ADDL dst (ADD Long range direct byte to accumulator)

Instruction code	0000H≦dst＜ffffH : [A6H][dst_L(7-0)][dst_H(7-0)] (3bytes) A6H
Byte count	3
Cycle count	2
Function	(A) ← (A) + (dst),　(PC) ← (PC) + 3
Affected flags	CY, AC, OV

[Description]
Adds the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

[Example]

		A	RAM 0200H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	200H	55H	55H	--	--	--
ADDL	200H	AAH	55H	0	0	1
ADDL	200H	FFH	55H	0	0	0
ADDL	200H	54H	55H	1	1	0
ADDL	200H	A9H	55H	0	0	1

ADDC #i (ADD immediate data and Carry to accumulator)

Instruction code	[1 0 1 1 0 0 0 1][i7i6i5i4i3i2i1i0] B1H
Byte count	2
Cycle count	1
Function	(A) ← (A) + (CY) + i, (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Adds the contents of the accumulator (A), the value of the carry flag (CY), and immediate data (8) and places the result in the accumulator (A).

[Example]

		A	**CY**	**AC**	**OV**
MOV	#55H, A	55H	--	--	--
ADD	#13H	68H	0	0	0
ADDC	#0AH	72H	0	1	0
ADDC	#0FH	81H	0	1	1
ADDC	#80H	01H	1	0	1
ADDC	#0FH	11H	0	1	0

ADDC [Rn] (ADD indirect byte and Carry to accumulator)

Instruction code	[1 0 1 1 0 0 1 0][0 n5n4n3n2n1n0 0] B2H
Byte count	2
Cycle count	2
Function	(A) ← (A) + (CY) + ((Rn)), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Adds the contents of the accumulator (A), the value of the carry flag (CY), and the contents of the data memory (RAM) or special function register (SFR) designated by the indirect register Rn and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

		RAM 0002H	RAM 0003H	A	RAM 0055H	CY	AC	OV
R1 points to addresses 02H and 03H								
LDW	#0055H	--	--	55H	--	--	--	--
STW	R1	55H	00H	55H	--	--	--	--
ST	[R1]	55H	00H	55H	55H	--	--	--
ADD	[R1]	55H	00H	AAH	55H	0	0	1
ADDC	[R1]	55H	00H	FFH	55H	0	0	0
ADDC	[R1]	55H	00H	54H	55H	1	1	0
ADDC	[R1]	55H	00H	AAH	55H	0	0	1

ADDC [Rn, C] (ADD indirect byte (with C register) and Carry to accumulator)

Instruction code	[1 0 1 1 0 0 1 0][0 n5n4n3n2n1n0 1] B2H
Byte count	2
Cycle count	2
Function	(A) ← (A) + (CY) + ((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Adds the contents of the accumulator (A), the value of the carry flag (CY), and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect register Rn and the C register and places the result in the accumulator (A).

The valid value range of n is $0 \leq n \leq 63$.

C register: 8-bit signed indirect displacement data $(-128 \leq$ C reg.$\leq 127)$

[Example]

R1 points to addresses 02H and 03H

	C	RAM 0002H	RAM 0003H	A	RAM 0059H	CY	AC	OV
LDW #0055H	--	--	--	--	--	--	--	--
STW R1	--	55H	00H	55H	--	--	--	--
MOV #04H, C	04H	55H	00H	55H	--	--	--	--
ST [R1, C]	04H	55H	00H	55H	55H	--	--	--
ADD [R1, C]	04H	55H	00H	AAH	55H	0	0	1
ADDC [R1, C]	04H	55H	00H	FFH	55H	0	0	0
ADDC [R1, C]	04H	55H	00H	54H	55H	1	1	0
ADDC [R1, C]	04H	55H	00H	AAH	55H	0	0	1

ADDC [off] (ADD indirect byte(with off) and Carry to accumulator)

Instruction code	[1 0 1 1 0 0 1 0][1 off6off5off4off3off2off1off0] B2H
Byte count	2
Cycle count	2
Function	(A) ← (A) + (CY) + ((R0) + off), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Adds the contents of the accumulator (A), the value of the carry flag (CY), and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect register R0 and off and places the result in the accumulator (A).

off: 7-bit signed indirect address offset data $(-64 \leqq off \leqq 63)$

[Example]

R0 points to address
00H and 01H

		RAM 0000H	RAM 0001H	A	RAM 0059H	CY	AC	OV
LDW	#0055H	--	--	55H	--	--	--	--
STW	R0	55H	00H	55H	--	--	--	--
ST	[04H]	55H	00H	55H	55H	--	--	--
ADD	[04H]	55H	00H	AAH	55H	0	0	1
ADDC	[04H]	55H	00H	FFH	55H	0	0	0
ADDC	[04H]	55H	00H	54H	55H	1	1	0
ADDC	[04H]	55H	00H	AAH	55H	0	0	1

ADDC dst (ADD direct byte and Carry to accumulator)

Instruction code	0000H≦dst＜0100H	: [B4H][dst_L(7-0)]	(2bytes)	B3H-B6H
	0100H≦dst＜0200H	: [B5H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [B3H][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [B6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [B6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2	3		
Cycle count	1	2		
Function	(A) ← (A) + (CY) + (dst), (PC) ← (PC) + (2 or 3)			
Affected flags	CY, AC, OV			

[Description]

Adds the contents of the accumulator (A), the value of the carry flag (CY), and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0034H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	34H	55H	55H	--	--	--
ADD	34H	AAH	55H	0	0	1
ADDC	34H	FFH	55H	0	0	0
ADDC	34H	54H	55H	1	1	0
ADDC	34H	AAH	55H	0	0	1

ADDCL dst (ADD Long range direct byte and Carry to accumulator)

Instruction code	0000H≦dst＜ffffH : [B6H][dst_L(7-0)][dst_H(7-0)] (3bytes) B6H
Byte count	3
Cycle count	2
Function	(A) ← (A) + (CY) + (dst), (PC) ← (PC) + 3
Affected flags	CY, AC, OV

[Description]

Adds the contents of the accumulator (A), the value of the carry flag (CY), and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

[Example]

		A	RAM 0200H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	200H	55H	55H	--	--	--
ADDL	200H	AAH	55H	0	0	1
ADDCL	200H	FFH	55H	0	0	0
ADDCL	200H	54H	55H	1	1	0
ADDCL	200H	AAH	55H	0	0	1

AND #i (AND immediate data to accumulator)

Instruction code	[1 1 1 1 0 0 0 1][i7i6i5i4i3i2i1i0]	F1H
Byte count	2	
Cycle count	1	
Function	(A) ← (A) & i, (PC) ← (PC) + 2	
Affected flags		

[Description]

Takes the AND of the contents of the accumulator (A) and immediate data and places the result in the accumulator (A).

[Example]

		A
MOV	#0FFH, A	FFH
AND	#0FAH	FAH
AND	#0AFH	AAH
AND	#0FH	0AH
AND	#0F0H	00H

AND [Rn] (AND indirect byte to accumulator)

Instruction code	[1 1 1 1 0 0 1 0][0 n5n4n3n2n1n0 0]	F2H
Byte count	2	
Cycle count	2	
Function	(A) ← (A) & ((Rn)), (PC) ← (PC) + 2	
Affected flags		

[Description]

Takes the AND of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

		RAM 0002H	RAM 0003H	A	RAM 0055H
LDW	#0055H	--	--	55H	--
STW	R1	55H	00H	55H	--
MOV	#0FCH, [R1]	55H	00H	55H	FCH
AND	[R1]	55H	00H	54H	FCH
MOV	#0F0H, [R1]	55H	00H	54H	F0H
AND	[R1]	55H	00H	50H	F0H
MOV	#0FH, [R1]	55H	00H	50H	0FH
AND	[R1]	55H	00H	00H	0FH

AND [Rn,C] (AND indirect byte(with C register) to accumulator)

Instruction code	[1 1 1 1 0 0 1 0][0 n5n4n3n2n1n0 1]	F2H
Byte count	2	
Cycle count	2	
Function	(A) ← (A) & ((Rn) + (C)), (PC) ← (PC) + 2	
Affected flags		

[Description]
Takes the AND of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register and places the result in the accumulator (A).
The valid value range of n is 0≦n≦63.
C register: 8-bit signed indirect displacement data (-128≦C reg.≦127)

[Example]

R1 points to addresses 02H and 03H

	C	RAM 0002H	RAM 0003H	A	RAM 0059H
LDW #0055H	--	--	--	55H	--
STW R1	--	55H	00H	55H	--
MOV #04H, C	04H	55H	00H	55H	--
MOV #0FFH, A	04H	55H	00H	FFH	--
MOV #0F0H, [R1, C]	04H	55H	00H	FFH	F0H
AND [R1, C]	04H	55H	00H	F0H	F0H
MOV #0FH, [R1, C]	04H	55H	00H	F0H	0FH
AND [R1, C]	04H	55H	00H	00H	0FH

AND [off] (AND indirect byte(with displacement) to accumulator)

Instruction code	[1 1 1 1 0 0 1 0][1 off6off5off4off3off2off1off0]	F2H
Byte count	2	
Cycle count	2	
Function	(A) ← (A) & ((R0) + off), (PC) ← (PC) + 2	
Affected flags		

[Description]

Takes the AND of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect register R0 and off and places the result in the accumulator (A).

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to address 00H and 01H

		RAM 0000H	RAM 0001H	A	RAM 0059H
LDW	#0055H	--	--	55H	--
STW	R0	55H	00H	55H	--
MOV	#0FFH, A	55H	00H	FFH	--
MOV	#0F0H, [04H]	55H	00H	FFH	F0H
AND	[04H]	55H	00H	F0H	F0H
MOV	#0FH, [04H]	55H	00H	F0H	0FH
AND	[04H]	55H	00H	00H	0FH

5-18

AND dst (AND direct byte to accumulator)

Instruction code	0000H≦dst＜0100H	: [F4H][dst_L(7-0)]	(2bytes)	F3H-F6H
	0100H≦dst＜0200H	: [F5H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [F3H][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [F6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [F6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2 ┊ 3			
Cycle count	1 ┊ 2			
Function	(A) ← (A) & (dst), (PC) ← (PC) + (2 or 3)			
Affected flags				

[Description]

Takes the AND of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0034H
MOV	#0FFH, A	FFH	--
MOV	#0FAH, 34H	FFH	FAH
AND	34H	FAH	FAH
MOV	#0AFH, 34H	FAH	AFH
AND	34H	AAH	AFH
MOV	#0F0H, 34H	AAH	F0H
AND	34H	A0H	F0H
MOV	#0FH, 34H	A0H	0FH
AND	34H	00H	0FH

ANDL dst (AND Long range direct byte to accumulator)

Instruction code	0000H≦dst≦ffffH : [F6H][dst_L(7-0)][dst_H(7-0)] (3bytes) F6H
Byte count	3
Cycle count	2
Function	(A) ← (A) & (dst), (PC) ← (PC) + (2 or 3)
Affected flags	

[Description]

Takes the AND of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

[Example]

		A	RAM 0200H
MOV	#0FFH, A	FFH	--
MOV	#0FAH, 200H	FFH	FAH
ANDL	200H	FAH	FAH
MOV	#0AFH, 200H	FAH	AFH
ANDL	200H	AAH	AFH
MOV	#0F0H, 200H	AAH	F0H
ANDL	200H	A0H	F0H
MOV	#0FH, 200H	A0H	0FH
ANDL	200H	00H	0FH

BE #i, r8 (Compare immediate data to accumulator and Branch near relative address if Equal)

Instruction code	[0 0 0 0 0 0 0 1][i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0] 01H
Byte count	3
Cycle count	2
Function	if(A) = i (PC) ← (PC) + 3 + r8 else (PC) ← (PC) + 3
Affected flags	CY

[Description]

Compares immediate data (i) with the contents of the accumulator (A) and, if they match, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the operands do not match, the CPU adds 3 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value in the accumulator (A) is smaller than the immediate data (i) and reset if the value in the accumulator (A) is greater than or equal to the immediate data (i).

   (A)<i : CY = 1
   (A)≧i : CY = 0

[Example]

		PC	Instruction Code	A	CY
MOV	#02H, A	01F01H	430002H	02H	-
BE	#01H, LA	01F04H	010107H	02H	0
BE	#03H, LA	01F07H	010304H	02H	1
BE	#02H, LA	01F0AH	010201H	02H	0
NOP		01F0DH	00H		
LA: NOP		01F0EH	00H	02H	0

BE [Rn], #i, r8
(Compare immediate data to indirect byte and Branch near relative address if Equal)

Instruction code	[0 0 0 0 0 0 1 0][0 n5n4n3n2n1n0 0][i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0] 02H
Byte count	4
Cycle count	3
Function	if((Rn)) = i (8-bit compare) (PC) ← (PC) + 4 + r8 else (PC) ← (PC) + 4
Affected flags	CY

[Description]

Compares immediate data (i) with the contents of the data memory (RAM) or special function register (SFR) designated by the indirect register Rn and, if they match, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the operands do not match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value designated by the indirect register Rn is smaller than the immediate data (i) and reset if the value is greater than or equal to the immediate data (i).

   ((Rn))<i : CY = 1
   ((Rn))≧i : CY = 0

The valid value range of n is $0 \leq n \leq 63$.

[Example]

R1 points to addresses 02H and 03H

			PC	Instruction Code	CY	RAM 0002H	RAM 0003H	RAM 0055H
	LDW	#0055H	01EFAH	475500H	-	--	--	--
	STW	R1	01EFDH	970200H	-	55H	00H	--
	MOV	#02H, [R1]	01F00H	420202H	-	55H	00H	02H
	BE	[R1],#01H,LA	01F03H	02020109H	0	55H	00H	02H
	BE	[R1],#03H,LA	01F07H	02020305H	1	55H	00H	02H
	BE	[R1],#02H,LA	01F0BH	02020201H	0	55H	00H	02H
	NOP		01F0FH	00H		55H	00H	02H
LA:	NOP		01F10H	00H	0	55H	00H	02H

BE [Rn, C], #i, r8
(Compare immediate data to indirect(with C reg.) byte and Branch near relative address if Equal)

Instruction code	[0 0 0 0 0 0 1 0][0 n5n4n3n2n1n0 0][i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0]	02H
Byte count	4	
Cycle count	3	
Function	if((Rn) + C) = i (8-bit compare)　　(PC) ← (PC) + 4 + r 8	
	else(PC) ← (PC) + 4	
Affected flags	CY	

[Description]

Compares immediate data (i) with the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect register Rn and the C register and, if they match, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the operands do not match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value designated by the result of arithmetic operation between the indirect register Rn and the C register is smaller than the immediate data (i) and reset if the value is greater than or equal to the immediate data (i).

　$((Rn)+C) < i$ ： CY = 1
　$((Rn)+C) \geqq i$ ： CY = 0

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data $(-128 \leqq C\ reg. \leqq 127)$

[Example]

		PC	Instruction Code	CY	RAM 0002H	RAM 0003H	RAM 0059H	C
R1 points to addresses 02H and 03H								
MOV	#04H, C	01EF7H	430204H	-	--	--	--	04H
LDW	#0055H	01EFAH	475500H	-	--	--	--	04H
STW	R1	01EFDH	970200H	-	55H	00H	--	04H
MOV	#02H, [R1, C]	01F00H	420302H	-	55H	00H	02H	04H
BE	[R1, C],#01H,LA	01F03H	02030109H	0	55H	00H	02H	04H
BE	[R1, C],#03H,LA	01F07H	02030305H	1	55H	00H	02H	04H
BE	[R1, C],#02H,LA	01F0BH	02030201H	0	55H	00H	02H	04H
NOP		01F0FH	00H					
LA: NOP		01F10H	00H	0	55H	00H	02H	04H

BE [off], #i, r8
(Compare immediate data to indirect(with displacement) byte and Branch near relative address if Equal)

Instruction code	[0 0 0 0 0 0 1 0][1 off6off5off4off3off2off1off0] [i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0]	02H
Byte count	4	
Cycle count	3	
Function	if((R0) + off) = i (8-bit compare)　(PC) ← (PC) + 4 + r8 else(PC) ← (PC) + 4	
Affected flags	CY	

[Description]
Compares immediate data (i) with the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect register R0 and off and, if they match, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the operands do not match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value designated by the result of arithmetic operation between the indirect register R0 and off is smaller than the immediate data (i) and reset if the value is greater than or equal to the immediate data (i).

　　((R0) + off)＜i　　:　CY = 1
　　((R0) + off)≧i　　:　CY = 0

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to address 00H and 01H

			PC	Instruction Code	CY	RAM 0000H	RAM 0001H	RAM 0059H
	LDW	#0055H	01EFAH	475500H	-	--	--	--
	STW	R0	01EFDH	970000H	-	55H	00H	--
	MOV	#02H, [04H]	01F00H	428402H	-	55H	00H	02H
	BE	[04H],#01H,LA	01F03H	02840109H	0	55H	00H	02H
	BE	[04H],#03H,LA	01F07H	02840305H	1	55H	00H	02H
	BE	[04H],#02H,LA	01F0BH	02840201H	0	55H	00H	02H
	NOP		01F0FH	00H				
LA:	NOP		01F10H	00H	0	55H	00H	02H

BE dst, r8 (Compare direct byte data to accumulator and Branch near relative address if Equal)

Instruction code	0000H≦dst＜0100H	: [04H][dst_L(7-0)][r8]	(3bytes)	03H-06H
	0100H≦dst＜0200H	: [05H][dst_L(7-0)][r8]	(3bytes)	
	fe00H≦dst＜ff00H	: [03H][dst_L(7-0)][r8]	(3bytes)	
	0200H≦dst＜fe00H	: [06H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
	ff00H≦dst≦ffffH	: [06H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
Byte count	3　　　4			
Cycle count	2　　　3			
Function	if(A) = dst (8-bit compare)　　(PC) ← (PC) + (3 or 4) + r8			
	else(PC) ← (PC) + (3 or 4)			
Affected flags	CY			

[Description]

Compares the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if they match, adds the relative address (r) + 3 or 4 to the program counter (PC) and places the result in the program counter (PC). If the operands do not match, the CPU adds 3 or 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value in the accumulator (A) is smaller than the contents of the data memory (RAM) designated by dst and reset if the value is greater than or equal to the contents of the data memory (RAM).

　　(A)＜dst　:　CY = 1
　　(A)≧dst　:　CY = 0

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

			PC	Instruction Code	A	CY	RAM 0055H
	MOV	#02H, 55H	01F00H	445502H	--	-	02H
	MOV	#01H, A	01F03H	430001H	01H	-	02H
	BE	55H, LA	01F06H	04550DH	01H	1	02H
	MOV	#03H, A	01F09H	430003H	03H	1	02H
	BE	55H, LA	01F0CH	045507H	03H	0	02H
	MOV	#02H, A	01F0FH	430002H	02H	0	02H
	BE	55H, LA	01F12H	045501H	02H	0	02H
	NOP		01F15H	00H			
LA:	NOP		01F16H	00H	02H	0	02H

BEL dst, r8
(Compare Long range direct byte data to accumulator and Branch near relative address if Equal)

Instruction code	0000H≦dst≦ffffH : [06H][dst_L(7-0)][dst_H(7-0)][r8] (4bytes) 06H
Byte count	4
Cycle count	3
Function	if(A) = dst (8-bit compare) (PC) ← (PC) + 4 + r8
	else(PC) ← (PC) + 4
Affected flags	CY

[Description]

Compares the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if they match, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the operands do not match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value in the accumulator (A) is smaller than the contents of the data memory (RAM) designated by dst and reset if the value is greater than or equal to the contents of the data memory (RAM).

(A)＜dst ： CY = 1

(A)≧dst ： CY = 0

[Example]

			PC	Instruction Code	A	CY	RAM 0200H
	MOV	#02H, 200H	01EFFH	46000202H	--	-	02H
	MOV	#01H, A	01F03H	430001H	01H	-	02H
	BEL	200H, LA	01F06H	0600020FH	01H	1	02H
	MOV	#03H, A	01F0AH	430003H	03H	1	02H
	BEL	200H, LA	01F0DH	06000208H	03H	0	02H
	MOV	#02H, A	01F11H	430002H	02H	0	02H
	BEL	200H, LA	01F14H	06000201H	02H	0	02H
	NOP		01F18H	00H			
LA:	NOP		01F19H	00H	02H	0	02H

BN dst, bit, r8 (Branch near relative address if direct bit is Negative)

Instruction code		28H - 2FH, 38H - 3FH, C7H	
	0H≦dst＜100H	: [bit][28H][dst_L(7-0)][r8(7-0)]	(3bytes)
	fe00H≦dst＜ff00H	: [bit][38H][dst_L(7-0)][r8(7-0)]	(3bytes)
	100H≦dst＜2000H	: [C7H][dst_L(7-0)][bit&&dst_H][r8(7-0)]	(4bytes)
Byte count	3 ┊ 4		
Cycle count	2 ┊ 3		
Function	if(dst).bit = 0 (PC) ← (PC) + (3 or 4) + r8		
	else(PC) ← (PC) + (3 or 4)		
Affected flags			

[Description]
Increments the contents of the program counter (PC) by 3 or 4, adds data r7-r0 to the PC, and places the result in the PC if the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) pointed to by dst is reset. If the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) pointed to by dst is set, the CPU increments the PC by 3 or 4.
The byte count and cycle count vary according to the address value of the operand (dst).

[Example 1]

			PC	Instruction Code	RAM 0010H
	MOV	#0FEH, 010H	00F1AH	4410FEH	FEH
	BN	010H, 0, LA	00F1DH	28103FH	FEH
LA:	INC	010H	00F5FH	8C10H	FFH
	NOP		00F61H	00H	FFH

The CPU causes a branch to the label LA because bit 0 of address 0010H is held at "0" when the BN instruction is executed.

[Example 2]

			PC	Instruction Code	A
	MOV	#001H,A	00F1AH	430001H	01H
	BN	A, 0, LA	00F1DH	38003FH	01H
	DEC	A	00F20H	9B00H	00H
LA:	INC	A			

Execution proceeds to the next instruction because bit 0 of A is held at "1" when the BN instruction is executed.

BNM dst, bit, r8
(Branch near relative address if Middle range direct bit is Negative)

Instruction code	0H≦dst＜2000H　　　 : [C7H][dst_L(7-0)][bit&&dst_H][r8(7-0)]　(4bytes)　 C7H
Byte count	4
Cycle count	3
Function	if(dst).bit = 0　(PC) ← (PC) + 4 + r8
	else(PC) ← (PC) + 4
Affected flags	

[Description]

Increments the contents of the program counter (PC) by 4, adds data r7-r0 to the PC, and places the result in the PC if the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 is reset. If the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 is set, the CPU increments the PC by 4.

[Example 1]

			PC	Instruction Code	RAM 0010H
	MOV	#0FEH, 010H	00F19H	4410FEH	FEH
	BNM	010H, 0, LA	00F1CH	C710003FH	FEH
LA:	INC	010H	00F5FH	8C10H	FFH
	NOP		00F61H	00H	FFH

The CPU causes a branch to the label LA because bit 0 of address 0010H is held at "0" when the BNM instruction is executed.

[Example 2]

			PC	Instruction Code	RAM 0200H
	MOVL	#001H,00200H	00F18H	46000201H	01H
	BNM	00200H, 0, LA	00F1CH	C700023FH	01H
	DECL	00200H	00F20H	9E0002H	00H
LA:	INCL	00200H			

Execution proceeds to the next instruction because bit 0 of address 0200H is held at "1" when the BNM instruction is executed.

BNE #i, r8
(Compare immediate data to accumulator and Branch near relative address if Not Equal)

Instruction code	[0 0 0 1 0 0 0 1][i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0]	11H
Byte count	3	
Cycle count	2	
Function	if(A)≠i (8-bit compare)　(PC) ← (PC) + 3 + r8　else(PC) ← (PC) + 3	
Affected flags	CY	

[Description]

Adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC) if the comparison between immediate data (i) and the contents of the accumulator (A) is a mismatch. If the result of comparison is a match, the CPU adds 3 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value in the accumulator (A) is smaller than the immediate data (i) and reset if the value in the accumulator (A) is greater than or equal to the immediate data (i).

$\quad$ (A)＜i　:　CY = 1
$\quad$ (A)≧i　:　CY = 0

[Example]

			PC	Instruction Code	A	CY
	MOV	#02H, A	01F01H	430002H	02H	-
	BNE	#02H, LA	01F04H	110204H	02H	0
	BNE	#03H, LA	01F07H	110301H	02H	1
	NOP		01F0AH	00H		
LA:	NOP		01F0BH	00H	02H	1

BNE [Rn], #i, r8
(Compare immediate data to indirect byte and Branch near relative address if Not Equal)

Instruction code	[0 0 0 1 0 0 1 0][0 n5n4n3n2n1n0 0][i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0] 12H
Byte count	4
Cycle count	3
Function	if((Rn))≠i (8-bit compare) (PC) ← (PC) + 4 + r8 else(PC) ← (PC) + 4
Affected flags	CY

[Description]

Adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC) if the comparison between immediate data (i) and the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn is a mismatch. If the result of comparison is a match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value designated by the indirect address register Rn is smaller than the immediate data (i) and reset if the value designated by the indirect address register Rn is greater than or equal to the immediate data (i).

   ((Rn))<i : CY = 1
   ((Rn))≧i : CY = 0

The valid value range of n is 0≦n≦63.

[Example]

R1 points to addresses 02H and 03H

		PC	Instruction Code	CY	RAM 0002H	RAM 0003H	RAM 0055H
LDW	#0055H	01EFAH	475500H	-	--	--	--
STW	R1	01EFDH	970200H	-	55H	00H	--
MOV	#02H, [R1]	01F00H	420202H	-	55H	00H	02H
BNE	[R1],#02H,LA	01F03H	12020205H	0	55H	00H	02H
BNE	[R1],#03H,LA	01F07H	12020301H	1	55H	00H	02H
NOP		01F0BH	00H		55H	00H	02H
LA: NOP		01F0CH	00H	1	55H	00H	02H

BNE [Rn, C], #i, r8
(Compare immediate data to indirect(with C reg.) byte and Branch near relative address if Not Equal)

Instruction code	[0 0 0 1 0 0 1 0][0 n5n4n3n2n1n0 0][i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0]	12H
Byte count	4	
Cycle count	3	
Function	if((Rn) + C)≠i (8-bit compare) (PC) ← (PC) + 4 + r8	
	else(PC) ← (PC) + 4	
Affected flags	CY	

[Description]

Adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC) if the comparison between immediate data (i) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register is a mismatch. If the result of comparison is a match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value designated by the result of arithmetic operation between the indirect address register Rn and the C register is smaller than the immediate data (i) and reset if the value designated by the result of arithmetic operation between the indirect address register Rn and the C register is greater than or equal to the immediate data (i).

    ((Rn)+C)＜i   :   CY = 1
    ((Rn)+C)≧i   :   CY = 0

The valid value range of n is 0≦n≦63.

C register: 8-bit signed indirect displacement data (-128≦C reg.≦127)

[Example]

R1 points to addresses 02H and 03H

		PC	Instruction Code	CY	RAM 0002H	RAM 0003H	RAM 0059H	C
MOV	#04H, C	01EF7H	430204H	-	--	--	--	04H
LDW	#0055H	01EFAH	475500H	-	--	--	--	04H
STW	R1	01EFDH	970200H	-	55H	00H	--	04H
MOV	#02H, [R1, C]	01F00H	420302H	-	55H	00H	02H	04H
BNE	[R1, C],#02H,LA	01F03H	02030109H	0	55H	00H	02H	04H
BNE	[R1, C],#03H,LA	01F07H	02030305H	1	55H	00H	02H	04H
NOP		01F0FH	00H					
LA: NOP		01F10H	00H	0	55H	00H	02H	04H

BNE [off], #i, r8
(Compare immediate data to indirect(with displacement) byte and Branch near relative address if Not Equal)

Instruction code	[1 0 1 0 0 0 1 0][1 off6off5off4off3off2off1off0] 12H
	[i7i6i5i4i3i2i1i0][r7r6r5r4r3r2r1r0]
Byte count	4
Cycle count	3
Function	if((Rn) + off)≠i (8-bit compare) (PC) ← (PC) + 4 + r8
	else(PC) ← (PC) + 4
Affected flags	CY

[Description]
Adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC) if the comparison between immediate data (i) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off is a mismatch. If the result of comparison is a match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value designated by the result of arithmetic operation between the indirect address register R0 and off is smaller than the immediate data (i) and reset if the value designated by the result of arithmetic operation between the indirect address register R0 and off is greater than or equal to the immediate data (i).

   ((R0)+off)<i   :   CY = 1

   ((R0)+off)≧i   :   CY = 0

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]
R1 points to addresses 00H and 01H

			PC	Instruction Code	CY	RAM 0000H	RAM 0001H	RAM 0059H
	LDW	#0055H	01EFAH	475500H	-	--	--	--
	STW	R0	01EFDH	970000H	-	55H	00H	--
	MOV	#02H, [04H]	01F00H	428402H	-	55H	00H	02H
	BNE	[04H],#02H,LA	01F03H	12840205H	0	55H	00H	02H
	BNE	[04H],#03H,LA	01F07H	12840301H	1	55H	00H	02H
	NOP		01F0BH	00H				
LA:	NOP		01F0CH	00H	1	55H	00H	02H

BNE dst, r8
(Compare direct byte data to accumulator and Branch near relative address if Not Equal)

Instruction code	0000H≦dst＜0100H	: [14H][dst_L(7-0)][r8]	(3bytes)	13H-16H
	0100H≦dst＜0200H	: [15H][dst_L(7-0)][r8]	(3bytes)	
	fe00H≦dst＜ff00H	: [13H][dst_L(7-0)][r8]	(3bytes)	
	0200H≦dst＜fe00H	: [16H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
	ff00H≦dst≦ffffH	: [16H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
Byte count	3 : 4			
Cycle count	2 : 3			
Function	if(A)≠dst (8-bit compare)　　(PC) ← (PC) + (3 or 4) + r8 else(PC) ← (PC) + (3 or 4)			
Affected flags	CY			

[Description]
Compares the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if they do not match, adds the relative address (r) + 3 or 4 to the program counter (PC) and places the result in the program counter (PC). If the operands match, the CPU adds 3 or 4 to the program counter (PC) and executes the next instruction.
The carry flag (CY) is set if the value in the accumulator (A) is smaller than the contents of the data memory (RAM) designated by dst and reset if the value is greater than or equal to the contents of the data memory (RAM).
  (A)＜dst　:　CY = 1
  (A)≧dst　:　CY = 0
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

			PC	Instruction Code	A	CY	RAM 0055H
	MOV	#02H, 55H	01F00H	445502H	--	-	02H
	MOV	#02H, A	01F03H	430002H	02H	-	02H
	BNE	55H, LA	01F06H	145507H	02H	0	02H
	MOV	#01H, A	01F09H	430001H	01H	0	02H
	BNE	55H, LA	01F0CH	145501H	01H	1	02H
	NOP		01F0FH	00H			
LA:	NOP		01F10H	00H	01H	1	02H

BNEL dst, r8
(Compare Long range direct byte data to accumulator and Branch near relative address if Not Equal)

Instruction code	0000H≦dst≦ffffH : [16H][dst_L(7-0)][dst_H(7-0)][r8] (4bytes) 16H
Byte count	4
Cycle count	3
Function	if(A)≠dst (8-bit compare) (PC) ← (PC) + 4 + r8
	else(PC) ← (PC) + 4
Affected flags	CY

[Description]

Compares the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if they do not match, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the operands match, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The carry flag (CY) is set if the value in the accumulator (A) is smaller than the contents of the data memory (RAM) designated by dst and reset if the value is greater than or equal to the contents of the data memory (RAM).

  (A)<dst : CY = 1
  (A)≧dst : CY = 0

[Example]

			PC	Instruction Code	A	CY	RAM 0200H
	MOV	#02H, 200H	01EFFH	46000202H	--	-	02H
	MOV	#02H, A	01F03H	430002H	02H	-	02H
	BNEL	200H, LA	01F06H	16000208H	02H	0	02H
	MOV	#01H, A	01F0AH	430001H	01H	0	02H
	BNEL	200H, LA	01F0DH	16000201H	01H	1	02H
	NOP		01F11H	00H			
LA:	NOP		01F12H	00H	01H	1	02H

BP dst, bit, r8 (Branch near relative address if direct bit is Positive)

Instruction code				08H - 0FH, 18H - 1FH, E7H
	0H≦dst＜100H	: [bit][08H][dst_L(7-0)][r8(7-0)]		(3bytes)
	fe00H≦dst＜ff00H	: [bit][18H][dst_L(7-0)][r8(7-0)]		(3bytes)
	100H≦dst＜2000H	: [E7H][dst_L(7-0)][bit&&dst_H][r8(7-0)]		(4bytes)
Byte count	3	4		
Cycle count	2	3		
Function	if(dst).bit = 1　(PC) ← (PC) + (3 or 4) + r8			
	else(PC) ← (PC) + (3 or 4)			
Affected flags				

[Description]

Increments the contents of the program counter (PC) by 3 or 4, adds data r7-r0 to the PC, and places the result in the PC if the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) pointed to by dst is set. If the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) pointed to by dst is reset, the CPU increments the PC by 3 or 4.

The byte count and cycle count vary according to the address value of the operand (dst).

[Example 1]

			PC	Instruction Code	RAM 0010H
	MOV	#001H, 010H	00F1AH	441001H	01H
	BP	010H, 0, LA	00F1DH	08103FH	01H
LA:	INC	010H	00F5FH	8C10H	02H
	NOP		00F61H	00H	02H

The CPU causes a branch to the label LA because bit 0 of address 0010H is held at "1" when the BP instruction is executed.

[Example 2]

			PC	Instruction Code	A
	MOV	#080H, A	00F18H	430080H	80H
	BP	A, 0, LA	00F1DH	18003FH	80H
	DEC	A	00F20H	9B00H	7FH
LA:	INC	A			

Execution proceeds to the next instruction because bit 0 of A is held at "0" when the BP instruction is executed.

BPM dst, bit, r8 (Branch near relative address if Middle range direct bit is Positive)

Instruction code	0H≦dst＜2000H : [E7H][dst_L(7-0)][bit&&dst_H][r8(7-0)] (4bytes) E7H
Byte count	4
Cycle count	3
Function	if(dst).bit = 1 (PC) ← (PC) + 4 + r8
	else(PC) ← (PC) + 4
Affected flags	

[Description]

Increments the contents of the program counter (PC) by 4, adds data r7-r0 to the PC, and places the result in the PC if the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 is set. If the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 is reset, the CPU increments the PC by 4.

[Example 1]

			PC	Instruction Code	RAM 0010H
	MOV	#001H, 010H	00F19H	441001H	01H
	BPM	010H, 0, LA	00F1CH	E710003FH	01H
LA:	INC	010H	00F5FH	8C10H	02H
	NOP		00F61H	00H	02H

The CPU causes a branch to the label LA because bit 0 of address 0010H is held at "1" when the BPM instruction is executed.

[Example 2]

			PC	Instruction Code	RAM 0200H
	MOVL	#080H,00200H	00F18H	46000280H	80H
	BPM	00200H, 0, LA	00F1CH	E700023FH	80H
	DECL	00200H	00F20H	9E0002H	7FH
LA:	INCL	00200H			

Execution proceeds to the next instruction because bit 0 of address 0200H is held at "0" when the BPM instruction is executed.

BPC dst, bit, r8
(Branch near relative address and Clear bit if middle range direct bit is Positive)

Instruction code	0H≦dst＜2000H : [A7H][dst_L(7-0)][bit&&dst_H][r8(7-0)] (4bytes) A7H
Byte count	4
Cycle count	3
Function	if(dst).bit = 1 (dst).bit ← 0, (PC) ← (PC) + 4 + r8 else(PC) ← (PC) + 4
Affected flags	

[Description]
If the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 is set, the CPU resets the bit, increments the contents of the program counter (PC) by 4, adds data r7-r0 to the PC, and places the result in the PC. If the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 is reset, the CPU increments the PC by 4.

[Example 1]

			PC	Instruction Code	RAM 0010H
	MOV	#003H, 010H	00F19H	441003H	03H
	BPC	010H, 0, LA	00F1CH	A710003FH	02H
LA:	INC	010H	00F5FH	8C10H	03H
	NOP		00F61H	00H	03H

Because bit 0 of address 0010H is held at "1" when the BPC instruction is executed, the CPU causes a branch to the label LA.

[Example 2]

			PC	Instruction Code	RAM 0200H
	MOVL	#080H,00200H	00F18H	46000280H	80H
	BPC	00200H, 0, LA	00F1CH	A700023FH	80H
	DECL	00200H	00F20H	9E0002H	7FH
LA:	INCL	00200H			

Execution proceeds to the next instruction because bit 0 of address 0200H is held at "0" when the BPC instruction is executed.

BR r12 (BRanch relative address)

Instruction code	[0 1 1 r11 1 r10r9r8][r7r6r5r4r3r2r1r0]	68H - 6FH,78H - 7FH
Byte count	2	
Cycle count	2	
Function	(PC) ← (PC) + 2 + r12	
Affected flags		

[Description]

Increments the program counter (PC) by 2, then adds data r11-r0 to the PC and places the result in the PC.

[Example 1]

The value of label LA is 00F5FH.

			PC	Instruction Code
	NOP		00F1CH	00H
	NOP		00F1DH	00H
	BR	LA	00F1EH	683FH
LA:	INC	A	00F5FH	8B00H
	ROR		00F61H	C0H

[Example 2]

The value of label LA is 01F0EH.

			PC	Instruction Code
	NOP		01F0CH	00H
	NOP		01F0DH	00H
LA:	INC	A	01F0EH	8B00H
	ROR		01F10H	C0H
	NOP		01F11H	00H
	NOP		01F12H	00H
	BR	LA	01F13H	7FF9H

BZ r8 (Branch near relative address if accumulator is Zero)

Instruction code	[1 0 0 0 1 0 0 1][r7r6r5r4r3r2r1r0] 89H
Byte count	2
Cycle count	2
Function	if(A) = 0 (PC) ← (PC) + 2 + r8 else(PC) ← (PC) + 2
Affected flags	

[Description]

Adds the relative address (r) + 2 to the program counter (PC) and places the result in the program counter (PC) if the contents of the accumulator (A) are zeros. If the contents of the accumulator (A) are nonzero, the CPU adds 2 to the program counter (PC) and executes the next instruction.

[Example]

			PC	Instruction Code	A
	MOV	#01H, A	01F00H	430001H	01H
	BZ	LA	01F03H	8905H	01H
	DEC	A	01F05H	9B00H	00H
	BZ	LA	01F07H	8901H	00H
	NOP		01F09H	00H	
LA:	NOP		01F0AH	00H	00H

BZW r8 (Branch near relative address if Word data(BA) is Zero)

Instruction code	[0 1 0 0 0 0 0 1][r7r6r5r4r3r2r1r0] 41H
Byte count	2
Cycle count	2
Function	if(BA) = 0000H (PC) ← (PC) + 2 + r8 else(PC) ← (PC) + 2
Affected flags	

[Description]

Adds the relative address (r) + 2 to the program counter (PC) and places the result in the program counter (PC) if the contents of both the accumulator (A) and B register are zeros. If the contents of both the accumulator (A) and B register are nonzero, the CPU adds 2 to the program counter (PC) and executes the next instruction.

[Example]

			PC	Instruction Code	A	B
	LDW	#0001H	01F00H	470100H	01H	00H
	STW	A	01F03H	9700FEH	01H	00H
	BZW	LA	01F06H	4104H	01H	00H
	DEC	A	01F08H	9B00H	00H	00H
	BZW	LA	01F0AH	4100H	00H	00H
LA:	NOP		01F0CH	00H	00H	00H

BNZ r8 (Branch near relative address if accumulator is Not Zero)

Instruction code	[1 0 0 1 1 0 0 1][r7r6r5r4r3r2r1r0] 99H
Byte count	2
Cycle count	2
Function	if(A)≠0 (PC) ← (PC) + 2 + r8 else(PC) ← (PC) + 2
Affected flags	

[Description]

Adds the relative address (r) + 2 to the program counter (PC) and places the result in the program counter (PC) if the contents of the accumulator (A) is nonzero. If the contents of the accumulator (A) are zeros, the CPU adds 2 to the program counter (PC) and executes the next instruction.

[Example]

			PC	Instruction Code	A
	MOV	#00H, A	01F00H	430000H	00H
	BNZ	LA	01F03H	9905H	00H
	DEC	A	01F05H	9B00H	FFH
	BNZ	LA	01F07H	9901H	FFH
	NOP		01F09H	00H	
LA:	NOP		01F0AH	00H	FFH

5-41

BNZW r8 (Branch near relative address if Word data(BA) is Not Zero)

Instruction code	[0 1 0 1 0 0 0 1][r7r6r5r4r3r2r1r0] 51H
Byte count	2
Cycle count	2
Function	if(BA)≠0000H (PC) ← (PC) + 2 + r8 else(PC) ← (PC) + 2
Affected flags	

[Description]

Adds the relative address (r) + 2 to the program counter (PC) and places the result in the program counter (PC) if the contents of both the accumulator (A) and B register are nonzero. If the contents of both the accumulator (A) and B register are zeros, the CPU adds 2 to the program counter (PC) and executes the next instruction.

[Example]

			PC	Instruction Code	A	B
	LDW	#0000H	01F00H	470000H	00H	00H
	STW	A	01F03H	9700FEH	00H	00H
	BNZW	LA	01F06H	5104H	00H	00H
	DEC	A	01F08H	9B00H	FFH	00H
	BNZW	LA	01F0AH	5100H	FFH	00H
LA:	NOP		01F0CH	00H	FFH	00H

CALL a17 (CALL absolute address)

Instruction code	[0 0 1 1 0 0 0 a16][a7a6a5a4a3a2a1a0][a15a14a13a12a11a10a9a8]	30H, 31H
Byte count	3	
Cycle count	2	
Function	w(++(SP)) ← (PC), ++ (SP), (PC) ← a17	
Affected flags	RAMH8 ← PC (bit16), RAML8 ← BNK	

[Description]

Increments the stack pointer (SP), stores the address of the instruction following the CALL instruction (return address) in the data memory (RAM) pointed to by SP, and increments the SP. Finally, the instruction places the data a16-a0 (called address) in the PC. The CALL instruction is a subroutine call instruction that covers the entire ROM address on a bank (128KB).

The instruction transfers bit 16 of the program counter (PC) to bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP. It also transfers the bank flag bit (BNK) to bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP.

[Example 1]

The value of label LA is 00F0EH.

			PC	Instruction Code	SP	RAM 0021H	RAM 0020H
	MOV	#01FH, SPL	00FF6H	430A1FH	--	--	--
	MOV	#000H, SPH	00FF9H	430B00H	001FH	--	--
	CALL	LA	00FFCH	300E0FH	0021H	0FH	FFH
LA:	INC	A	00F0EH	8B00H	0021H	0FH	FFH
	RET		00F10H	A0H	001FH	0FH	FFH
	NOP		00FFFH	00H	001FH	0FH	FFH

[Example 2]

The value of label LA is 01F0EH.

			PC	Instruction Code	SP	RAM 0021H	RAM 0020H
	MOV	#01FH, SPL	00FF7H	430A1FH	--	--	--
	MOV	#000H, SPH	00FFAH	430B00H	001FH	--	--
	CALL	LA	00FFDH	300E1FH	0021H	10H	00H
LA:	INC	A	01F0EH	8B00H	0021H	10H	00H
	RET		01F10H	A0H	001FH	10H	00H
	INC	A	01000H	8B00H	001FH	10H	00H

CHGP1 n (CHanGe Psw bit1(P1) to n)

Instruction code	[1 1 n 1 1 0 0 1][0 0 0 0 0 1 1 0]	D9H, F9H
Byte count	2	
Cycle count	1	
Function	(PSW).1 ← n, (PC) ← (PC) + 2	
Affected flags		

[Description]
Resets bit 1 (P1) of the program status word (PSW) when n=0 and sets P1 when n=1.

CHGP3 n (CHanGe Psw bit3(LDCBNK) to n)

Instruction code	[1 1 n 1 1 0 1 1][0 0 0 0 0 1 1 0]	DBH, FBH
Byte count	2	
Cycle count	1	
Function	(PSW).3 ← n, (PC) ← (PC) + 2	
Affected flags		

[Description]
Resets the bank flag (LDCBNK) used for table lookup when n=0 and sets LDCBNK when n=1.

CLR1 dst, bit (CLeaR direct bit)

Instruction code			C8H - CFH, D8H - DFH, D7H	
	0H≦dst＜100H	: [bit][C8H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [bit][D8H][dst_L(7-0)]	(2bytes)	
	100H≦dst＜2000H	: [D7H][dst_L(7-0)][bit&&dst_H]	(3bytes)	
Byte count	2	3		
Cycle count	1	2		
Function	(dst).bit ← 0, (PC) ← (PC) + (2 or 3)			
Affected flags				

[Description]
Resets the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) pointed to by dst and subsequently increments the program counter (PC) by 2 or 3.
The byte count and cycle count vary according to the address value of the operand (dst).

[Example 1]

		RAM 0010H	
MOV	#001H, 010H	01H	0000 0001B
CLR1	010H, 0	00H	0000 0000B

[Example 2]

		A	
MOV	#001H, A	01H	0000 0001B
CLR1	A, 0	00H	0000 0000B

CLR1M dst, bit (CLeaR Middle range direct bit)

Instruction code	0H≦dst＜2000H : [D7H][dst_L(7-0)][bit&&dst_H] (3bytes) D7H
Byte count	3
Cycle count	2
Function	(dst).bit ← 0, (PC) ← (PC) + 3
Affected flags	

[Description]

Resets the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 (0≦dst＜2000H) and subsequently increments the program counter (PC) by 3.

[Example 1]

	RAM 0200H	
MOVL #001H, 00200H	01H	0000 0001B
CLR1M 00200H, 0	00H	0000 0000B

[Example 2]

	RAM 037FH	
MOVL #001H, 0037FH	01H	0000 0001B
CLR1M 0037FH, 0	00H	0000 0000B

DBNZ [Rn], r8
(Decrement indirect byte and Branch near relative address if indirect byte is Not Zero)

Instruction code	[0 0 1 0 0 0 1 0][0 n5n4n3n2n1n0 0][r7r6r5r4r3r2r1r0]	22H
Byte count	3	
Cycle count	3	
Function	If−−((Rn))≠0 (8-bit compare) (PC) ← (PC) + 3 + r8 else(PC) ← (PC) + 3	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]
Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn and, if the result of decrement is nonzero, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is zero, the CPU adds 3 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

			PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0055H
	LDW	#0055H	01EFAH	475500H	--	--	--
	STW	R1	01EFDH	970200H	55H	00H	--
	MOV	#01H, [R1]	01F00H	420201H	55H	00H	01H
	DBNZ	[R1], LA	01F03H	220204H	55H	00H	00H
	DBNZ	[R1], LA	01F06H	220201H	55H	00H	FFH
	NOP		01F09H	00H			
LA:	NOP		01F0AH	00H	55H	00H	FFH

<Programming Note>
This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

DBNZ [Rn, C], r8
(Decrement indirect(with C reg.) byte and Branch near relative address if indirect byte is Not Zero)

Instruction code	[0 0 1 0 0 0 1 0][0 n5n4n3n2n1n0 1][r7r6r5r4r3r2r1r0]	22H
Byte count	3	
Cycle count	3	
Function	If−−((Rn) + C)≠0 (8-bit compare)　　(PC) ← (PC) + 3 + r8 else(PC) ← (PC) + 3	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register and, if the result of decrement is nonzero, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is zero, the CPU adds 3 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data (-128$\leqq$C reg.$\leqq$127)

[Example]

R1 points to addresses 02H and 03H

			PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0059H	C
	MOV	#04H, C	01EF7H	430204H	--	--	--	04H
	LDW	#0055H	01EFAH	475500H	--	--	--	04H
	STW	R1	01EFDH	970200H	55H	00H	--	04H
	MOV	#01H, [R1,C]	01F00H	420301H	55H	00H	01H	04H
	DBNZ	[R1,C], LA	01F03H	220304H	55H	00H	00H	04H
	DBNZ	[R1,C], LA	01F06H	220301H	55H	00H	FFH	04H
	NOP		01F09H	00H				
LA:	NOP		01F0AH	00H	55H	00H	FFH	04H

<Programming Note>

This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

DBNZ [off], r8
(Decrement indirect (with displacement) byte and Branch near relative address if indirect byte is Not Zero)

Instruction code	[0 0 1 0 0 0 1 0][1 off6off5off4off3off2off1off0][r7r6r5r4r3r2r1r0]	22H
Byte count	3	
Cycle count	3	
Function	If−−((R0) + off)≠0 (8-bit compare) (PC) ← (PC) + 3 + r8	
	else(PC) ← (PC) + 3	
Affected flags	P1 ← REG8(9bitDEC)	

[Description]
Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off and, if the result of decrement is nonzero, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is zero, the CPU adds 3 to the program counter (PC) and executes the next instruction.
The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to address 00H and 01H

			PC	Instruction Code	RAM 0000H	RAM 0001H	RAM 0059H
	LDW	#0055H	01EFAH	475500H	--	--	--
	STW	R0	01EFDH	970000H	55H	00H	--
	MOV	#01H, [04H]	01F00H	428401H	55H	00H	01H
	DBNZ	[04H], LA	01F03H	228404H	55H	00H	00H
	DBNZ	[04H], LA	01F06H	228401H	55H	00H	FFH
	NOP		01F09H	00H			
LA:	NOP		01F0AH	00H	55H	00H	FFH

<Programming Note>
This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation
    100H (9 bits) − 1 = 0FFH (bit 8 = 0)
is carried out and the register is loaded with "0FFH" and P1 with "0."

DBNZ dst, r8
(Decrement direct byte and Branch near relative address if direct byte is Not Zero)

Instruction code	0000H≦dst＜0100H	: [24H][dst_L(7-0)][r8]	(3bytes)	23H-26H
	0100H≦dst＜0200H	: [25H][dst_L(7-0)][r8]	(3bytes)	
	fe00H≦dst＜ff00H	: [23H][dst_L(7-0)][r8]	(3bytes)	
	0200H≦dst＜fe00H	: [26H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
	ff00H≦dst≦ffffH	: [26H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
Byte count	3 : 4			
Cycle count	2 : 3			
Function	If−−(dst)≠0 (8-bit compare)　(PC) ← (PC) + (3 or 4) + r8			
	else(PC) ← (PC) + (3 or 4)			
Affected flags	P1 ← REG8 (9bitDEC)			

[Description]
Decrements the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if the result of decrement is nonzero, adds the relative address (r) + 3 or 4 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is zero, the CPU adds 3 or 4 to the program counter (PC) and executes the next instruction.
The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

			PC	Instruction Code	RAM 0055H
	MOV	#01H, 55H	01F00H	445501H	01H
	DBNZ	55H, LA	01F03H	245504H	00H
	DBNZ	55H, LA	01F06H	245501H	FFH
	NOP		01F09H	00H	
LA:	NOP		01F0AH	00H	FFH

<Programming Note>
This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation
　　100H (9 bits) − 1 = 0FFH (bit 8 = 0)
is carried out and the register is loaded with "0FFH" and P1 with "0."

DBNZL dst, r8
(Decrement Long range direct byte and Branch near relative address if direct byte is Not Zero)

Instruction code	0000H≦dst≦ffffH : [26H][dst_L(7-0)][dst_H(7-0)][r8] (4bytes) 26H
Byte count	4
Cycle count	3
Function	If−−(dst)≠0 (8-bit compare) (PC) ← (PC) + 4 + r8
	else(PC) ← (PC) + 4
Affected flags	P1 ← REG8 (9bitDEC)

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if the result of decrement is nonzero, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is zero, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

[Example]

			PC	Instruction Code	RAM 0200H
	MOV	#01H, 200H	01EFFH	46000201H	01H
	DBNZL	200H, LA	01F03H	26000205H	00H
	DBNZL	200H, LA	01F06H	26000201H	FFH
	NOP		01F09H	00H	
LA:	NOP		01F0AH	00H	FFH

<Programming Note>

This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

DBZ [Rn], r8
(Decrement indirect byte and Branch near relative address if indirect byte is Zero)

Instruction code	[0 0 1 1 0 0 1 0][0 n5n4n3n2n1n0 0][r7r6r5r4r3r2r1r0]	32H
Byte count	3	
Cycle count	3	
Function	If$--$((Rn)) = 0 (8-bit compare) (PC) ← (PC) + 3 + r8	
	else(PC) ← (PC) + 3	
Affected flags	P1← REG8 (9bitDEC)	

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn and, if the result of decrement is zero, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is nonzero, the CPU adds 3 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

		PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0055H
LDW	#0055H	01EFAH	475500H	--	--	--
STW	R1	01EFDH	970200H	55H	00H	--
MOV	#02H, [R1]	01F00H	420202H	55H	00H	02H
DBZ	[R1], LA	01F03H	320204H	55H	00H	01H
DBZ	[R1], LA	01F06H	320201H	55H	00H	00H
NOP		01F09H	00H			
LA: NOP		01F0AH	00H	55H	00H	00H

<Programming Note>

This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

DBZ [Rn, C], r8
(Decrement indirect (with C reg.) byte and Branch near relative address if indirect byte is Zero)

Instruction code	[0 0 1 1 0 0 1 0][0 n5n4n3n2n1n0 1][r7r6r5r4r3r2r1r0]	32H
Byte count	3	
Cycle count	3	
Function	If−−(Rn) + C = 0 (8-bit compare) (PC) ← (PC) + 3 + r8	
	else(PC) ← (PC) + 3	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]
Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register and, if the result of decrement is zero, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is nonzero, the CPU adds 3 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data (-128$\leqq$C reg.$\leqq$127)

[Example]
R1 points to addresses 02H and 03H

			PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0059H	C
	MOV	#04H, C	01EF7H	430204H	--	--	--	04H
	LDW	#0055H	01EFAH	475500H	--	--	--	04H
	STW	R1	01EFDH	970200H	55H	00H	--	04H
	MOV	#02H, [R1,C]	01F00H	420302H	55H	00H	02H	04H
	DBZ	[R1,C], LA	01F03H	320304H	55H	00H	01H	04H
	DBZ	[R1,C], LA	01F06H	320301H	55H	00H	00H	04H
	NOP		01F09H	00H				
LA:	NOP		01F0AH	00H	55H	00H	00H	04H

<Programming Note>
This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

    100H (9 bits) − 1 = 0FFH (bit 8 = 0)
is carried out and the register is loaded with "0FFH" and P1 with "0."

DBZ [off], r8
(Decrement indirect (with displacement) byte and Branch near relative address if indirect byte is Zero)

Instruction code	[0 0 1 1 0 0 1 0][1 off6off5off4off3off2off1off0][r7r6r5r4r3r2r1r0]	32H
Byte count	3	
Cycle count	3	
Function	If−−((R0)+off) = 0 (8-bit compare)　　(PC) ← (PC) + 3 + r8 else(PC) ← (PC) + 3	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]
Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off and, if the result of decrement is zero, adds the relative address (r) + 3 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is nonzero, the CPU adds 3 to the program counter (PC) and executes the next instruction.
The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to address 00H and 01H

			PC	Instruction Code	RAM 0000H	RAM 0001H	RAM 0059H
	LDW	#0055H	01EFAH	475500H	--	--	--
	STW	R0	01EFDH	970000H	55H	00H	--
	MOV	#02H, [R0]	01F00H	420202H	55H	00H	02H
	DBZ	[04H], LA	01F03H	328404H	55H	00H	01H
	DBZ	[04H], LA	01F06H	328401H	55H	00H	00H
	NOP		01F09H	00H			
LA:	NOP		01F0AH	00H	55H	00H	00H

<Programming Note>
This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation
　　100H (9 bits)　− 1 = 0FFH (bit 8 = 0)
is carried out and the register is loaded with "0FFH" and P1 with "0."

DBZ dst, r8 (Decrement direct byte and Branch near relative address if direct byte is Zero)

Instruction code	0000H≦dst＜0100H	: [34H][dst_L(7-0)][r8]	(3bytes)	33H-36H
	0100H≦dst＜0200H	: [35H][dst_L(7-0)][r8]	(3bytes)	
	fe00H≦dst＜ff00H	: [33H][dst_L(7-0)][r8]	(3bytes)	
	0200H≦dst＜fe00H	: [36H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
	ff00H≦dst＜ffffH	: [36H][dst_L(7-0)][dst_H(7-0)][r8]	(4bytes)	
Byte count	3	4		
Cycle count	2	3		
Function	If−−(dst) = 0 (8-bit compare)　(PC) ← (PC) + (3 or 4) + r8			
	else(PC) ← (PC) + (3 or 4)			
Affected flags	P1 ← REG8(9bitDEC)			

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if the result of decrement is zero, adds the relative address (r) + 3 or 4 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is nonzero, the CPU adds 3 or 4 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

			PC	Instruction Code	RAM 0055H
	MOV	#02H, 55H	01F00H	445502H	02H
	DBZ	55H, LA	01F03H	345504H	01H
	DBZ	55H, LA	01F06H	345501H	00H
	NOP		01F09H	00H	
LA:	NOP		01F0AH	00H	00H

<Programming Note>

This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

　　100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

DBZL dst, r8
(Decrement Long range direct byte and Branch near relative address if direct byte is Zero)

Instruction code	0000H≦dst≦ffffH : [36H][dst_L(7-0)][dst_H(7-0)][r8] (4bytes)	36H
Byte count	4	
Cycle count	3	
Function	If−−(dst) = 0 (8-bit compare) (PC) ← (PC) + 4 + r8	
	else(PC) ← (PC) + 4	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by dst and, if the result of decrement is zero, adds the relative address (r) + 4 to the program counter (PC) and places the result in the program counter (PC). If the result of decrement is nonzero, the CPU adds 4 to the program counter (PC) and executes the next instruction.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

[Example]

			PC	Instruction Code	RAM 0200H
	MOV	#02H, 200H	01EFFH	46000202H	02H
	DBZL	200H, LA	01F03H	36000205H	01H
	DBZL	200H, LA	01F07H	36000201H	00H
	NOP		01F0BH	00H	
LA:	NOP		01F0CH	00H	00H

<Programming Note>

This instruction performs "9-bit decrement processing" but tests only 8 bits (bits7-0) which the CPU can access. When addressing an 8-bit register in which bit 8 is nonexistent, the CPU performs 9-bit decrement processing assuming a 1 in bit 8 and places the resultant bit 8 in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

$100H$ (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

DEC [Rn] (DECrement indirect byte)

Instruction code	[1 0 0 1 1 0 1 0][0 n5n4n3n2n1n0 0]	9AH
Byte count	2	
Cycle count	2	
Function	((Rn)) ← ((Rn)) – 1, (PC) ← (PC) + 2	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leq n \leq 63$.

[Example]

R1 points to addresses 02H and 03H

	RAM 0002H	RAM 0003H	RAM 0051H
LDW #0051H	--	--	--
STW R1	51H	00H	--
MOV #10H, [R1]	51H	00H	10H
DEC [R1]	51H	00H	0FH
DEC [R1]	51H	00H	0EH
DEC [R1]	51H	00H	0DH

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit decrement processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) – 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

Consequently, decrement processing performed on an 8-bit register results in the following:

There is a carry if P1 is set to 0.

There is no carry if P1 is set to 1.

DEC [Rn,C] (DECrement indirect (with C register) byte)

Instruction code	[1 0 0 1 1 0 1 0][0 n5n4n3n2n1n0 1]	9AH
Byte count	2	
Cycle count	2	
Function	((Rn) + (C)) ← ((Rn) + (C)) − 1, (PC) ← (PC) + 2	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data $(-128 \leqq C \text{ reg.} \leqq 127)$

[Example]

		C	RAM 0002H	RAM 0003H	RAM 0056H
R1 points to addresses 02H and 03H					
LDW	#0051H	--	--	--	--
STW	R1	--	51H	00H	--
MOV	#05H, C	05H	51H	00H	--
MOV	#0FFH, [R1, C]	05H	51H	00H	FFH
DEC	[R1, C]	05H	51H	00H	FEH
DEC	[R1, C]	05H	51H	00H	FDH
DEC	[R1, C]	05H	51H	00H	FCH

\<Programming Note\>

The CPU assumes a 1 in bit position 8 when performing 9-bit decrement processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

   100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

Consequently, decrement processing performed on an 8-bit register results in the following:

   There is a carry if P1 is set to 0.

   There is no carry if P1 is set to 1.

DEC [off] (DECrement indirect (with displacement) byte)

Instruction code	[1 0 0 1 1 0 1 0][0 off6off5off4off3off2off1off0]	9AH
Byte count	2	
Cycle count	2	
Function	((R0) + off) ← ((R0) + off) − 1, (PC) ← (PC) + 2	
Affected flags	P1 ← REG8 (9bitDEC)	

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

		A	RAM 0000H	RAM 0001H	RAM 0056H	RAM 004CH
LDW	#0051H	51H	--	--	--	--
STW	R0	51H	51H	00H	--	--
ST	[05H]	51H	51H	00H	51H	--
ST	[-5]	51H	51H	00H	51H	51H
DEC	[05H]	51H	51H	00H	50H	51H
DEC	[-5]	51H	51H	00H	50H	50H

<div style="border:1px solid">

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit decrement processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) − 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

Consequently, decrement processing performed on an 8-bit register results in the following:

There is a carry if P1 is set to 0.

There is no carry if P1 is set to 1.

</div>

DEC dst (DECrement direct byte)

Instruction code	0000H≦dst＜0100H	: [9CH][dst_L(7-0)]	(2bytes)	9BH-9EH
	0100H≦dst＜0200H	: [9DH][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [9BH][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [9EH][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	:[9EH][dst_L(7-0)][dstH(7-0)]	(3bytes)	
Byte count	2	3		
Cycle count	1	2		
Function	(dst) ← (dst) – 1, (PC) ← (PC) + (2 or 3)			
Affected flags	P1 ← REG8 (9bitDEC)			

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by dst.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0011H	RAM 0012H
MOV	#050H, A	50H	--	--
ST	11H	50H	50H	--
DEC	11H	50H	4FH	--
ST	12H	50H	4FH	50H
DEC	12H	50H	4FH	4FH

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit decrement processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) – 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

Consequently, decrement processing performed on an 8-bit register results in the following:

There is a carry if P1 is set to 0.

There is no carry if P1 is set to 1.

DECL dst (DECrement Long range direct byte)

Instruction code	0000H≦dst≦ffffH :[9EH][dst_L(7-0)][dst_H(7-0)] (3bytes) 9EH
Byte count	3
Cycle count	2
Function	(dst) ← (dst) – 1, (PC) ← (PC) + 3
Affected flags	P1 ← REG8 (9bitDEC)

[Description]

Decrements the contents of the data memory (RAM) or special function register (SFR) designated by dst.

The instruction also transfers bit 8 of the decremented data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

[Example]

		A	**RAM FE12H**	**RAM FE13H**
MOV	#050H, A	50H	--	--
ST	0FE12H	50H	50H	--
DECL	0FE12H	50H	4FH	--
ST	0FE13H	50H	4FH	50H
DECL	0FE13H	50H	4FH	4FH

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit decrement processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when decrement processing is performed on an 8-bit register having a value of "00H", the computation

100H (9 bits) – 1 = 0FFH (bit 8 = 0)

is carried out and the register is loaded with "0FFH" and P1 with "0."

Consequently, decrement processing performed on an 8-bit register results in the following:

There is a carry if P1 is set to 0.

There is no carry if P1 is set to 1.

DECW dst (DECrement long range direct Word)

Instruction code	[1 0 0 1 1 1 1 1][dst7dst6dst5dst4dst3dst2dst1dst0] 9FH
	[dst15dst14dst13dst12dst11dst10dst9dst8]
Byte count	3
Cycle count	3
Function	w17 (dst) −−, (PC) ← (PC) + 3
	w17 (dst) = [REGH8 (1bit) + REGH (8bit) + REGL (8bit)]
Affected flags	P1← REGH8, REGL8 ← not ｛LowByte − CY｝

[Description]

Decrements the contents of the word data memory (RAM) or special function register (SFR) designated by dst15-dst0 (17 bits (9 higher-order bits + 8 lower-order bits)) and subsequently increments the contents of the program counter (PC) by 3.

The instruction transfers bit 8 of the higher-order byte of the decremented word data memory (RAM) or special function register having bit 8 to bit 1 (P1) of the program status word (PSW). It also transfers the inverted CY out of the lower-order byte to bit 8 of the lower-order byte of the word data memory (RAM) or special function register (SFR) having bit 8 designated by dst15-dst0.

[Example 1]

		B	A
MOV	#002H, A	--	02H
MOV	#000H, B	00H	02H
DECW	0FE00H	00H	01H
DECW	0FE00H	00H	00H
DECW	0FE00H	FFH	FFH
DECW	0FE00H	FFH	FEH

[Example 2]

		RAM 007FH	RAM 007EH
MOV	#002H, 07EH	--	002H
MOV	#000H, 07FH	000H	002H
DECW	0007EH	000H	101H
DECW	0007EH	000H	100H
DECW	0007EH	1FFH	0FFH

DIV16 (DIVision 16bit/8bit=(16bit&8bit))

Instruction code	[0 1 0 0 0 0 0 0][0 0 0 0 0 0 0 0] 40H
Byte count	2
Cycle count	8
Function	quotient(B)(A)...remainder(C) ← ((SP) − 1)((SP) − 2) / ((SP)) if((SP) = 00H) {(OV)← 1, (B)(A) ← 0ffffH, (C) ← ((SP) − 2)} ;zero divide else {(OV) ← 0} , (SP) ← (SP) – 3, (PC) ← (PC) + 2
Affected flags	OV

[Description]
Divides the 16-bit data consisting of the contents of the data memory (RAM) pointed to by the stack pointer (SP)-1 (higher-order byte) and the contents of the data memory (RAM) pointed to by the stack pointer (SP)-2 (lower-order byte) by the contents of the data memory (RAM) pointed to by the stack pointer SP. The quotient is placed in the B register (B) (higher-order byte) and the accumulator (A) (lower-order byte) and the remainder in the C register (C). If the contents of the data memory (RAM) pointed to by SP are zero, Both B and A are loaded with FFH, the C register with the contents of the data memory (RAM) pointed to by SP, and the overflow flag (OV) is set. If the contents of the data memory (RAM) pointed to by SP are nonzero, the OV is reset. Subsequently, the CPU decrements the SP by 3 and finally increments the program counter (PC) by 2.

[Example 1]

		C	B	A	SP	RAM 0022H	RAM 0021H	RAM 0020H	OV
MOV	#004H, PSW	--	--	--	--	--	--	--	1
MOV	#01FH, SPL	--	--	--	--	--	--	--	1
MOV	#000H, SPH	--	--	--	001FH	--	--	--	1
PUSH	#005H	--	--	--	0020H	--	--	05H	1
PUSH	#079H	--	--	--	0021H	--	79H	05H	1
PUSH	#007H	--	--	--	0022H	07H	79H	05H	1
DIV16		06H	11H	49H	001FH	07H	79H	05H	0

[Example 2]

		C	B	A	SP	RAM 0022H	RAM 0021H	RAM 0020H	OV
MOV	#000H, PSW	--	--	--	--	--	--	--	0
MOV	#01FH, SPL	--	--	--	--	--	--	--	0
MOV	#000H, SPH	--	--	--	001FH	--	--	--	0
PUSH	#010H	--	--	--	0020H	--	--	10H	0
PUSH	#007H	--	--	--	0021H	--	07H	10H	0
PUSH	#000H	--	--	--	0022H	00H	07H	10H	0
DIV16		10H	FFH	FFH	001FH	00H	07H	10H	1

DIV24 (DIVision 24bit/16bit=(24bit&16bit))

Instruction code	[0 1 0 0 0 0 0 0][1 0 0 0 0 0 0 0] 40H
Byte count	2
Cycle count	12
Function	quotient(C)(B)(A)**...** remainder((SP) − 3)((SP) − 4) $\leftarrow$ ((SP) − 2)((SP) − 3)((SP) − 4) / ((SP))((SP) − 1) if((SP)((SP) − 1) = 0000H) {(OV)$\leftarrow$ 1, (C)(B)(A) $\leftarrow$ 0ffffffH, ((SP) − 3) ((SP) − 4) is invariable (Note)};zero divide else {(OV) $\leftarrow$ 0} , (SP) $\leftarrow$ (SP) − 3, (PC) $\leftarrow$ (PC) + 2
Affected flags	OV, RAM8 $\leftarrow$ 1

Note: Bit 8 of RAM is set to 1.

[Description]
Divides the 24-bit data consisting of the contents of the data memory (RAM) pointed to by the stack pointer (SP)-2 (highest-order byte), the contents of the data memory (RAM) pointed to by the stack pointer (SP)-3 (higher-order byte), and the contents of the data memory (RAM) pointed to by the stack pointer (SP)-4 (lower-order byte) by the contents of the data memory (RAM) pointed to by the stack pointer (SP) (higher-order byte) and the contents of the data memory (RAM) pointed to by the stack pointer SP-1 (lower-order byte). The quotient is placed in the C register (C) (highest-order byte), B register (B) (higher-order byte) and the accumulator (A) (lower-order byte), and the remainder in the data memory (RAM) locations pointed to by SP-3 and SP-4. If the contents of the data memory (RAM) pointed to by SP and the contents of the data memory (RAM) pointed to by SP-1 are zero, C, B, and A are loaded with FFH and the overflow flag (OV) is set. The contents of RAM locations pointed to by SP-3 and SP-4 remain unchanged. If the contents of RAM pointed to by SP (higher-order byte) and the contents of RAM pointed to by SP-1 (lower-order byte) are nonzero, the OV is reset.
Subsequently, the CPU decrements the SP by 3 and finally increments the program counter (PC) by 2.

[Example]

		C	B	A	SP	RAM 0024H	RAM 0023H	RAM 0022H	RAM 0021H	RAM 0020H	OV
MOV	#000H, PSW	--	--	--	--	--	--	--	--	--	0
MOV	#01FH, SPL	--	--	--	--	--	--	--	--	--	0
MOV	#000H, SPH	--	--	--	001FH	--	--	--	--	--	0
PUSH	#000H	--	--	--	0020H	--	--	--	--	00H	0
PUSH	#010H	--	--	--	0021H	--	--	--	10H	00H	0
PUSH	#007H	--	--	--	0022H	--	--	07H	10H	00H	0
PUSH	#000H	--	--	--	0023H	--	00H	07H	10H	00H	0
PUSH	#000H	--	--	--	0024H	00H	00H	07H	10H	00H	0
DIV24		FFH	FFH	FFH	0021H	00H	00H	07H	10H	00H	1

FADD (Function ADDition)

Instruction code	[1 0 0 1 1 0 0 0][0 0 0 0 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A) + ((SP)−−, (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Adds the contents of the data memory (RAM) pointed to by SP and the contents of the accumulator (A), places
the result in the accumulator (A), and decrements the SP. Finally, the instruction increments the program counter
(PC) by 2.

[Example 1]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#061H, A	61H	001FH	--	0	0	0
PUSH	#068H	61H	0020H	68H	0	0	0
FADD		C9H	001FH	68H	0	0	1
PUSH	#068H	C9H	0020H	68H	0	0	1
FADD		31H	001FH	68H	1	1	0

[Example 2]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#072H, A	72H	001FH	--	0	0	0
PUSH	#095H	72H	0020H	95H	0	0	0
FADD		07H	001FH	95H	1	0	0
PUSH	#095H	07H	0020H	95H	1	0	0
FADD		9CH	001FH	95H	0	0	0

FADDC (Function ADDition with Carry)

Instruction code	[1 0 0 1 1 0 0 0][0 0 0 1 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A) + (CY) + ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Adds the contents of the data memory (RAM) pointed to by SP, the carry flag (CY), and the contents of the accumulator (A), places the result in the accumulator (A), and decrements the SP. Finally, the instruction increments the program counter (PC) by 2.

[Example 1]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#061H, A	61H	001FH	--	0	0	0
PUSH	#068H	61H	0020H	68H	0	0	0
FADDC		C9H	001FH	68H	0	0	1
PUSH	#068H	C9H	0020H	68H	0	0	1
FADDC		31H	001FH	68H	1	1	0

[Example 2]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#072H, A	72H	001FH	--	0	0	0
PUSH	#095H	72H	0020H	95H	0	0	0
FADDC		07H	001FH	95H	1	0	0
PUSH	#095H	07H	0020H	95H	1	0	0
FADDC		9DH	001FH	95H	0	0	0

FADDW (Function Word ADDition)

Instruction code	[1 0 0 1 1 0 0 0][1 0 0 0 0 0 0 0] 98H
Byte count	2
Cycle count	2
Function	(SP)−−, (BA) ← (BA) + w ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Decrements the stack pointer (S), adds the contents of the BA register pair (BA) and the contents of the word data memory (RAM) pointed to by SP, places the result in the BA, and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
MOV	#000H, A	--	00H	001FH	--	--	0	0	0
MOV	#061H, B	61H	00H	001FH	--	--	0	0	0
PUSH	#000H	61H	00H	0020H	--	00H	0	0	0
PUSH	#068H	61H	00H	0021H	68H	00H	0	0	0
FADDW		C9H	00H	001FH	68H	00H	0	0	1

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
MOV	#000H, A	--	00H	001FH	--	--	0	0	0
MOV	#072H, B	72H	00H	001FH	--	--	0	0	0
PUSH	#000H	72H	00H	0020H	--	00H	0	0	0
PUSH	#095H	72H	00H	0021H	95H	00H	0	0	0
FADDW		07H	00H	001FH	95H	00H	1	0	0

FADDCW (Function Word ADDition with Carry)

Instruction code	[1 0 0 1 1 0 0 0][1 0 0 1 0 0 0 0] 98H
Byte count	2
Cycle count	2
Function	(SP)−−, (BA) ← (BA) + (CY) + w ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Decrements the stack pointer (S), adds the contents of the BA register pair (BA), the carry flag (CY), and the contents of the word data memory (RAM) pointed to by SP, places the result in the BA, and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
LDW	#6800H	68H	00H	001FH	--	--	0	0	0
PUSH_BA		68H	00H	0021H	68H	00H	0	0	0
LDW	#6100H	61H	00H	0021H	68H	00H	0	0	0
FADDCW		C9H	00H	001FH	68H	00H	0	0	1
SET1	PSW, 7	C9H	00H	001FH	68H	00H	1	0	1
PUSH	#000H	C9H	00H	0020H	68H	00H	1	0	1
PUSH	#068H	C9H	00H	0021H	68H	00H	1	0	1
FADDCW		31H	01H	001FH	68H	00H	1	1	0

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
LDW	#9500H	95H	00H	001FH	--	--	0	0	0
PUSH_BA		95H	00H	0021H	95H	00H	0	0	0
LDW	#7200H	72H	00H	0021H	95H	00H	0	0	0
FADDCW		07H	00H	001FH	95H	00H	1	0	0
PUSH	#000H	07H	00H	0020H	95H	00H	1	0	0
PUSH	#095H	07H	00H	0021H	95H	00H	1	0	0
FADDCW		9DH	00H	001FH	95H	00H	0	0	0

FAND (Function AND)

Instruction code	[1 0 0 1 1 0 0 0][0 1 0 1 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A) & ((SP)−−), (PC) ← (PC) + 2
Affected flags	

[Description]

Takes the AND of the contents of the accumulator (A) and the contents of the data memory (RAM) pointed to by SP, places the result in the accumulator (A), and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		A	SP	RAM 0020H
MOV	#0FFH, A	FFH	001FH	--
PUSH	#055H	FFH	0020H	55H
FAND		55H	001FH	55H
PUSH	#0AAH	55H	0020H	AAH
FAND		00H	001FH	AAH

[Example 2]

		A	SP	RAM 0020H
MOV	#0FFH, A	FFH	001FH	--
PUSH	#0FEH	FFH	0020H	FEH
FAND		FEH	001FH	FEH
PUSH	#0FDH	FEH	0020H	FDH
FAND		FCH	001FH	FDH

FANDW (Function Word AND)

Instruction code	[1 0 0 1 1 0 0 0][1 1 0 1 0 0 0 0]	98H
Byte count	2	
Cycle count	2	
Function	(SP)−−, (BA) ← (BA) & (w(SP)−−), (PC) ← (PC) + 2	
Affected flags		

[Description]
Decrements the SP, takes the AND of the contents of the BA register pair (BA) and the contents of the word data memory (RAM) pointed to by SP, places the result in the BA, and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	--	--	--	--	--
MOV	#000H, SPH	--	--	001FH	--	--
PUSH	#055H	--	--	0020H	--	55H
PUSH	#055H	--	--	0021H	55H	55H
MOV	#0FFH, A	--	FFH	0021H	55H	55H
MOV	#0FFH, B	FFH	FFH	0021H	55H	55H
FANDW		55H	55H	001FH	55H	55H

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	--	--	--	--	--
MOV	#000H, SPH	--	--	001FH	--	--
PUSH	#0FEH	--	--	0020H	--	FEH
PUSH	#0FEH	--	--	0021H	FEH	FEH
MOV	#0FFH, A	--	FFH	0021H	FEH	FEH
MOV	#0FFH, B	FEH	FFH	0021H	FEH	FEH
FANDW		FEH	FEH	001FH	FEH	FEH

FNOR (Function NOR)

Instruction code	[1 0 0 1 1 0 0 0][0 1 0 0 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← not ｛(A)｜((SP)−−)｝ , (PC) ← (PC) + 2
Affected flags	

[Description]

Takes the OR of the contents of the accumulator (A) and the contents of the data memory (RAM) pointed to by SP, places the inverted result in the accumulator (A), and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		A	SP	RAM 0020H
MOV	#000H, A	00H	001FH	--
PUSH	#055H	00H	0020H	55H
FNOR		AAH	001FH	55H
PUSH	#0AAH	AAH	0020H	AAH
FNOR		55H	001FH	AAH

[Example 2]

		A	SP	RAM 0020H
MOV	#000H, A	00H	001FH	--
PUSH	#001H	00H	0020H	01H
FNOR		FEH	001FH	01H
PUSH	#002H	FEH	0020H	02H
FNOR		01H	001FH	02H

FNORW (Function Word NOR)

Instruction code	[1 0 0 1 1 0 0 0][1 1 0 0 0 0 0 0]	98H
Byte count	2	
Cycle count	2	
Function	(SP)−−, (BA) ← not {(BA)∣w((SP)−−)} , (PC) ← (PC) + 2	
Affected flags		

[Description]
Decrements the SP, takes the OR of the contents of the BA register pair (BA) and the contents of the word data memory (RAM) pointed to by SP, places the inverted result in the BA, and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

	B	A	SP	RAM 0021H	RAM 0020H
MOV #01FH, SPL	--	--	--	--	--
MOV #000H, SPH	--	--	001FH	--	--
PUSH #055H	--	--	0020H	--	55H
PUSH #055H	--	--	0021H	55H	55H
MOV #000H, A	--	00H	0021H	55H	55H
MOV #000H, B	00H	00H	0021H	55H	55H
FNORW	AAH	AAH	001FH	55H	55H

[Example 2]

	B	A	SP	RAM 0021H	RAM 0020H
MOV #01FH, SPL	--	--	--	--	--
MOV #000H, SPH	--	--	001FH	--	--
PUSH #001H	--	--	0020H	--	01H
PUSH #001H	--	--	0021H	01H	01H
MOV #000H, A	--	00H	0021H	01H	01H
MOV #000H, B	00H	00H	0021H	01H	01H
FNORW	FEH	FEH	001FH	01H	01H

FOR (Function OR)

Instruction code	[1 0 0 1 1 0 0 0][0 1 1 0 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A) │ ((SP)−−), (PC) ← (PC) + 2
Affected flags	

[Description]

Takes the OR of the contents of the accumulator (A) and the contents of the data memory pointed to by SP, places the result in the accumulator (A), and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		A	SP	RAM 0020H
MOV	#000H, A	00H	001FH	--
PUSH	#055H	00H	0020H	55H
FOR		55H	001FH	55H
PUSH	#0AAH	55H	0020H	AAH
FOR		FFH	001FH	AAH

[Example 2]

		A	SP	RAM 0020H
MOV	#000H, A	00H	001FH	--
PUSH	#001H	00H	0020H	01H
FOR		01H	001FH	01H
PUSH	#002H	01H	0020H	02H
FOR		03H	001FH	02H

FORW (Function Word OR)

Instruction code	[1 0 0 1 1 0 0 0][1 1 1 0 0 0 0 0] 98H
Byte count	2
Cycle count	2
Function	(SP)−−,　(BA) ← (BA)∣(w(SP)−−),　(PC) ← (PC) + 2
Affected flags	

[Description]

Decrements the SP, takes the OR of the contents of the BA register pair (BA) and the contents of the word data memory (RAM) pointed to by SP, places the result in the BA, and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	--	--	--	--	--
MOV	#000H, SPH	--	--	001FH	--	--
PUSH	#055H	--	--	0020H	--	55H
PUSH	#055H	--	--	0021H	55H	55H
MOV	#000H, A	--	00H	0021H	55H	55H
MOV	#000H, B	00H	00H	0021H	55H	55H
FORW		55H	55H	001FH	55H	55H

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	--	--	--	--	--
MOV	#000H, SPH	--	--	001FH	--	--
PUSH	#001H	--	--	0020H	--	01H
PUSH	#001H	--	--	0021H	01H	01H
MOV	#000H, A	--	00H	0021H	01H	01H
MOV	#000H, B	00H	00H	0021H	01H	01H
FORW		01H	01H	001FH	01H	01H

FSUB (Function SUBtraction)

Instruction code	[1 0 0 1 1 0 0 0][0 0 1 0 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A) − ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Subtracts the contents of the data memory (RAM) pointed to by SP from the accumulator (A), places the result in the accumulator (A), and decrements the SP. The instruction finally increments the program counter by 2.

[Example 1]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#049H, A	49H	001FH	--	0	0	0
PUSH	#068H	49H	0020H	68H	0	0	0
FSUB		E1H	001FH	68H	1	0	0
PUSH	#068H	E1H	0020H	68H	1	0	0
FSUB		79H	001FH	68H	0	1	1

[Example 2]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#07EH, A	7EH	001FH	--	0	0	0
PUSH	#095H	7EH	0020H	95H	0	0	0
FSUB		E9H	001FH	95H	1	0	1
PUSH	#095H	E9H	0020H	95H	1	0	1
FSUB		54H	001FH	95H	0	0	0

FSUBC (Function SUBtraction with Carry)

Instruction code	[1 0 0 1 1 0 0 0][0 0 1 1 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A) – (CY) – ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Subtracts the contents of the data memory (RAM) pointed to by SP and the carry flag (CY) from the accumulator (A), places the result in the accumulator (A), and decrements the SP. The instruction finally increments the program counter by 2.

[Example 1]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#049H, A	49H	001FH	--	0	0	0
PUSH	#068H	49H	0020H	68H	0	0	0
FSUBC		E1H	001FH	68H	1	0	0
PUSH	#068H	E1H	0020H	68H	1	0	0
FSUBC		78H	001FH	68H	0	1	1

[Example 2]

		A	SP	RAM 0020H	CY	AC	OV
MOV	#07EH, A	7EH	001FH	--	0	0	0
PUSH	#095H	7EH	0020H	95H	0	0	0
FSUBC		E9H	001FH	95H	1	0	1
PUSH	#095H	E9H	0020H	95H	1	0	1
FSUBC		53H	001FH	95H	0	0	0

FSUBW (Function Word SUBtraction)

Instruction code	[1 0 0 1 1 0 0 0][1 0 1 0 0 0 0 0] 98H
Byte count	2
Cycle count	2
Function	(SP)−−, (BA) ← (BA) – w ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Decrements the stack pointer (SP), subtracts the contents of the word data memory (RAM) pointed to by SP
from the BA register pair (BA), places the result in the BA, and decrements the SP. The instruction finally
increments the program counter by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
MOV	#000H, A	--	00H	001FH	--	--	0	0	0
MOV	#049H, B	49H	00H	001FH	--	--	0	0	0
PUSH	#000H	49H	00H	0020H	--	00H	0	0	0
PUSH	#068H	49H	00H	0021H	68H	00H	0	0	0
FSUBW		E1H	00H	001FH	68H	00H	1	0	0

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
MOV	#000H, A	--	00H	001FH	--	--	0	0	0
MOV	#07EH, B	7EH	00H	001FH	--	--	0	0	0
PUSH	#000H	7EH	00H	0020H	--	00H	0	0	0
PUSH	#095H	7EH	00H	0021H	95H	00H	0	0	0
FSUBW		E9H	00H	001FH	95H	00H	1	0	1

5-78

FSUBCW (Function Word SUBtraction with Carry)

Instruction code	[1 0 0 1 1 0 0 0][1 0 1 1 0 0 0 0] 98H
Byte count	2
Cycle count	2
Function	(SP)−−, (BA) ← (BA) – (CY) – w ((SP)−−), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Decrements the stack pointer (SP), subtracts the contents of the word data memory (RAM) pointed to by SP and the carry flag (CY) from the BA register pair (BA), places the result in the BA, and decrements the SP. The instruction finally increments the program counter by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
MOV	#000H, A	--	00H	001FH	--	--	0	0	0
MOV	#049H, B	49H	00H	001FH	--	--	0	0	0
PUSH	#000H	49H	00H	0020H	--	00H	0	0	0
PUSH	#068H	49H	00H	0021H	68H	00H	0	0	0
FSUBCW		E1H	00H	001FH	68H	00H	1	0	0
PUSH	#000H	E1H	00H	0020H	68H	00H	1	0	0
PUSH	#068H	E1H	00H	0021H	68H	00H	1	0	0
FSUBCW		78H	FFH	001FH	68H	00H	0	1	1

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H	CY	AC	OV
MOV	#000H, A	--	00H	001FH	--	--	0	0	0
MOV	#07EH, B	7EH	00H	001FH	--	--	0	0	0
PUSH	#000H	7EH	00H	0020H	--	00H	0	0	0
PUSH	#095H	7EH	00H	0021H	95H	00H	0	0	0
FSUBCW		E9H	00H	001FH	95H	00H	1	0	1
PUSH	#000H	E9H	00H	0020H	95H	00H	1	0	1
PUSH	#095H	E9H	00H	0021H	95H	00H	1	0	1
FSUBCW		53H	00H	001FH	95H	00H	0	0	0

FXOR (Function eXclusive OR)

Instruction code	[1 0 0 1 1 0 0 0][0 1 1 1 0 0 0 0] 98H
Byte count	2
Cycle count	1
Function	(A) ← (A)^((SP)−−), (PC) ← (PC) + 2
Affected flags	

[Description]

Takes the XOR of the contents of the accumulator (A) and the data memory (RAM) pointed to by SP, places the result in the accumulator (A), and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		A	SP	RAM 0020H
MOV	#000H, A	00H	001FH	--
PUSH	#055H	00H	0020H	55H
FXOR		55H	001FH	55H
PUSH	#0FFH	55H	0020H	FFH
FXOR		AAH	001FH	FFH

[Example 2]

		A	SP	RAM 0020H
MOV	#0FFH, A	FFH	001FH	--
PUSH	#010H	FFH	0020H	10H
FXOR		EFH	001FH	10H
PUSH	#020H	EFH	0020H	20H
FXOR		CFH	001FH	20H

FXORW (Function Word eXclusive OR)

Instruction code	[1 0 0 1 1 0 0 0][1 1 1 1 0 0 0 0]	98H
Byte count	2	
Cycle count	2	
Function	(SP)−−, (BA) ← (BA)^(w(SP)−−), (PC) ← (PC) + 2	
Affected flags		

[Description]
Decrements the stack pointer, takes the XOR of the contents of the BA register pair (BA) and the word data memory (RAM) pointed to by SP, places the result in the BA, and decrements the SP. The instruction finally increments the program counter (PC) by 2.

[Example 1]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	--	--	--	--	--
MOV	#000H, SPH	--	--	001FH	--	--
PUSH	#055H	--	--	0020H	--	55H
PUSH	#055H	--	--	0021H	55H	55H
MOV	#000H, A	--	00H	0021H	55H	55H
MOV	#000H, B	00H	00H	0021H	55H	55H
FXORW		55H	55H	001FH	55H	55H

[Example 2]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	--	--	--	--	--
MOV	#000H, SPH	--	--	001FH	--	--
PUSH	#010H	--	--	0020H	--	10H
PUSH	#010H	--	--	0021H	10H	10H
MOV	#0FFH, A	--	FFH	0021H	10H	10H
MOV	#0FFH, B	FFH	FFH	0021H	10H	10H
FXORW		EFH	EFH	001FH	10H	10H

INC [Rn] (INCrement indirect byte)

Instruction code	[1 0 0 0 1 0 1 0][0 n5n4n3n2n1n0 0] 8AH
Byte count	2
Cycle count	2
Function	((Rn)) ← ((Rn)) + 1, (PC) ← (PC) + 2
Affected flags	P1← REG8 (9bitINC)

[Description]

Increments the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn.

The instruction also transfers bit 8 of the designated data memory or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

	R1 points to addresses 02H and 03H	RAM 0002H	RAM 0003H	RAM 0051H
LDW	#0051H	--	--	--
STW	R1	51H	00H	--
MOV	#10H, [R1]	51H	00H	10H
INC	[R1]	51H	00H	11H
INC	[R1]	51H	00H	12H
INC	[R1]	51H	00H	13H

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit increment processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when increment processing is performed on an 8-bit register having a value of "0FFH", the computation

    1FFH (9 bits) + 1 = 200H (bit 8 = 0)

is carried out and the register is loaded with "00H" and P1 with "0."

Consequently, increment processing performed on an 8-bit register results in the following:

    There is a carry if P1 is set to 0.
    There is no carry if P1 is set to 1.

INC [Rn,C] (INCrement indirect (with C register) byte)

Instruction code	[1 0 0 0 1 0 1 0][0 n5n4n3n2n1n0 1] 8AH
Byte count	2
Cycle count	2
Function	((Rn) + (C)) ← ((Rn) + (C)) + 1, (PC) ← (PC) + 2
Affected flags	P1← REG8 (9bitINC)

[Description]
Increments the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register.
The instruction also transfers bit 8 of the designated data memory or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).
The valid value range of n is $0 \leq n \leq 63$.
C register: 8-bit signed indirect displacement data $(-128 \leq C reg. \leq 127)$

[Example]

R1 points to addresses 02H and 03H

		C	RAM 0002H	RAM 0003H	RAM 0056H
LDW	#0051H	--	--	--	--
STW	R1	--	51H	00H	--
MOV	#05H, C	05H	51H	00H	--
MOV	#0FFH, [R1, C]	05H	51H	00H	FFH
INC	[R1, C]	05H	51H	00H	00H
INC	[R1, C]	05H	51H	00H	01H
INC	[R1, C]	05H	51H	00H	02H

<Programming Note>
The CPU assumes a 1 in bit position 8 when performing 9-bit increment processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when increment processing is performed on an 8-bit register having a value of "0FFH", the computation
    1FFH (9 bits) + 1 = 200H (bit 8 = 0)
is carried out and the register is loaded with "00H" and P1 with "0."
Consequently, increment processing performed on an 8-bit register results in the following:
    There is a carry if P1 is set to 0.
    There is no carry if P1 is set to 1.

INC [off] (INCrement indirect (with displacement) byte)

Instruction code	[1 0 0 0 1 0 1 0][0 off6off5off4off3off2off1off0] 8AH
Byte count	2
Cycle count	2
Function	((R0) + off) ← ((R0) + off) + 1, (PC) ← (PC) + 2
Affected flags	P1← REG8 (9bitINC)

[Description]

Increments the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off.

The instruction also transfers bit 8 of the designated data memory or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).

off: 7-bit signed indirect address offset data ($-64 \leqq off \leqq 63$)

[Example]

R0 points to addresses 00H and 01H

	A	RAM 0000H	RAM 0001H	RAM 0056H	RAM 004CH
LDW #0051H	51H	--	--	--	--
STW R0	51H	51H	00H	--	--
ST [05H]	51H	51H	00H	51H	--
ST [-5]	51H	51H	00H	51H	51H
INC [05H]	51H	51H	00H	52H	51H
INC [-5]	51H	51H	00H	52H	52H

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit increment processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when increment processing is performed on an 8-bit register having a value of "0FFH", the computation

1FFH (9 bits) + 1 = 200H (bit 8 = 0)

is carried out and the register is loaded with "00H" and P1 with "0."

Consequently, increment processing performed on an 8-bit register results in the following:

There is a carry if P1 is set to 0.

There is no carry if P1 is set to 1.

INC dst (INCrement direct byte)

Instruction code	0000H≦dst＜0100H	: [8CH][dst_L(7-0)]	(2bytes)	8BH-8EH
	0100H≦dst＜0200H	: [8DH][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [8BH][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [8EH][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [8EH][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2	3		
Cycle count	1	2		
Function	(dst) ← (dst) + 1, (PC) ← (PC) + (2 or 3)			
Affected flags	P1 ← REG8 (9bitINC)			

[Description]
Increments the contents of the data memory (RAM) or special function register (SFR) designated by dst.
The instruction also transfers bit 8 of the designated data memory or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

	A	RAM 0011H	RAM 0012H
MOV #050H, A	50H	--	--
ST 11H	50H	50H	--
INC 11H	50H	51H	--
ST 12H	50H	51H	50H
INC 12H	50H	51H	51H

<Programming Note>

The CPU assumes a 1 in bit position 8 when performing 9-bit increment processing on an 8-bit register in which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when increment processing is performed on an 8-bit register having a value of "0FFH", the computation
   1FFH (9 bits) + 1 = 200H (bit 8 = 0)
is carried out and the register is loaded with "00H" and P1 with "0."
Consequently, increment processing performed on an 8-bit register results in the following:
   There is a carry if P1 is set to 0.
   There is no carry if P1 is set to 1.

INCL dst (INCrement Long range direct byte)

Instruction code	0000H≦dst≦ffffH : [8EH][dst_L(7-0)][dst_H(7-0)] (3bytes) 8EH
Byte count	3
Cycle count	2
Function	(dst) ← (dst) + 1, (PC)← (PC) + 3
Affected flags	P1← REG8 (9bitINC)

[Description]
Increments the contents of the data memory (RAM) or special function register (SFR) designated by dst.
The instruction also transfers bit 8 of the designated data memory or special function register (SFR) having bit 8
to bit 1 of the program status word (PSW).

[Example]

		A	RAM FE12H	RAM FE13H
MOV	#050H, A	50H	--	--
ST	0FE12H	50H	50H	--
INCL	0FE12H	50H	51H	--
ST	0FE13H	50H	51H	50H
INCL	0FE13H	50H	51H	51H

<Programming Note>
The CPU assumes a 1 in bit position 8 when performing 9-bit increment processing on an 8-bit register in
which bit 8 is nonexistent. The resultant state of bit 8 is placed in P1 (bit 1 of the PSW). For example, when
increment processing is performed on an 8-bit register having a value of "0FFH", the computation
     1FFH (9 bits) + 1 = 200H (bit 8 = 0)
is carried out and the register is loaded with "00H" and P1 with "0."
Consequently, increment processing performed on an 8-bit register results in the following:
     There is a carry if P1 is set to 0.
     There is no carry if P1 is set to 1.

INCW dst (INCrement long range direct Word data)

Instruction code	[1 0 0 0 1 1 1 1][dst7dst6dst5dst4dst3dst2dst1dst0] 8FH [dst15dst14dst13dst12dst11dst10dst9dst8]
Byte count	3
Cycle count	3
Function	w17(dst) ++, (PC) ← (PC) + 3 w17(dst) = [REGH8(1bit) + REGH(8bit) + REGL(8bit)]
Affected flags	P1 ← REGH8, REGL8 ← (LowByte–CY)

[Description]

Increments the contents (17 bits (9 higher-order bits + 8 lower-order bits)) of the word data memory (RAM) or special function register (SFR) designated by dst15-dst0. The instruction subsequently increments the program counter (PC) by 3.

The instruction transfers bit 8 of the higher-order byte of the incremented word data memory or special function register (SFR) having bit 8 designated by dst15-dst0 to bit 1 (P1) of the program status word (PSW). It also transfers the CY out of the lower-order byte to bit 8 of the lower-order byte of the word data memory (RAM) or special function register (SFR) having bit 8 designated by dst15-dst0.

[Example 1]

	B	A
MOV #0FDH, A	--	FDH
MOV #0FFH, B	FFH	FDH
CLR1 PSW, 1	FFH	FEH
INCW 0FE00H	FFH	FEH
INCW 0FE00H	FFH	FFH
INCW 0FE00H	00H	00H
INCW 0FE00H	00H	01H

[Example 2]

	RAM 007FH	RAM 007EH
MOV #0FDH, 07EH	--	0FDH
MOV #0FFH, 07FH	0FFH	0FDH
CLR1 PSW, 1	0FFH	0FDH
INCW 0007EH	0FFH	0FEH
INCW 0007EH	0FFH	0FFH
INCW 0007EH	100H	100H

JMP a17 (JuMP absolute address)

Instruction code	[0 0 1 0 0 0 0 a16][a7a6a5a4a3a2a1a0][a15a14a13a12a11a10a9a8] 20H,21H
Byte count	3
Cycle count	2
Function	(PC) ← a17
Affected flags	

[Description]

Transfer data a16-a0 to the PC. The JMP instruction causes a jump to a location in the entire ROM address space on the same bank.

[Example 1]

The value of label LA is 00F0EH.

		PC	Instruction Code
NOP		00FFBH	00H
NOP		00FFCH	00H
JMP	LA	00FFDH	200E0FH
LA: INC	A	00F0EH	8B00H
ROR		00F10H	C0H

[Example 2]

The value of label LA is 01F0EH.

		PC	Instruction Code
NOP		00FFBH	00H
NOP		00FFCH	00H
JMP	LA	00FFDH	200E1FH
LA: INC	A	01F0EH	8B00H
ROR		01F10H	C0H

LD #I (LoaD immediate data to accumulator)

Instruction code	[1 0 0 0 0 0 0 1][i7i6i5i4i3i2i1i0] 81H
Byte count	2
Cycle count	1
Function	(A) ← i, (PC) ← (PC) + 2
Affected flags	

[Description]

Transfers immediate data (i) to the accumulator (A).

[Example]

	A
LD #050H	50H
LD #051H	51H
LD #052H	52H
LD #053H	53H
LD #054H	54H

LD [Rn] (LoaD indirect byte to accumulator)

Instruction code	[1 0 0 0 0 0 1 0][0 n5n4n3n2n1n0 0] 82H
Byte count	2
Cycle count	2
Function	(A) ← ((Rn)), (PC) ← (PC) + 2
Affected flags	P1 ← REG8

[Description]

Transfers the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn to the accumulator (A).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

		A	RAM 0002H	RAM 0003H	RAM 0051H
LDW	#0051H	51H	--	--	--
STW	R1	51H	51H	00H	--
MOV	#0FFH, A	FFH	51H	00H	--
MOV	#00H, [R1]	FFH	51H	00H	00H
LD	[R1]	00H	51H	00H	00H
INC	[R1]	00H	51H	00H	01H
LD	[R1]	01H	51H	00H	01H

LD [Rn,C] (LoaD indirect byte (with C register) to accumulator)

Instruction code	[1 0 0 0 0 0 1 0][0 n5n4n3n2n1n0 1] 82H
Byte count	2
Cycle count	2
Function	(A) ← ((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	P1 ← REG8

[Description]
Transfers the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register to the accumulator (A).
The instruction also transfers bit 8 of the designated data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).
The valid value range of n is 0≦n≦63.
C register: 8-bit signed indirect displacement data (-128≦C reg.≦127)

[Example]

R1 points to addresses 02H and 03H

		A	**C**	**RAM 0002H**	**RAM 0003H**	**RAM 0056H**
LDW	#0051H	51H	--	--	--	--
STW	R1	51H	--	51H	00H	--
MOV	#05H, C	51H	05H	51H	00H	--
MOV	#0FFH, [R1,C]	51H	05H	51H	00H	FFH
LD	[R1,C]	FFH	05H	51H	00H	FFH
INC	[R1,C]	FFH	05H	51H	00H	00H
LD	[R1,C]	00H	05H	51H	00H	00H

LD [off] (LoaD indirect byte (with displacement) to accumulator)

Instruction code	[1 0 0 0 0 0 1 0][1 off6off5off4off3off2off1off0] 82H
Byte count	2
Cycle count	2
Function	(A) ← ((R0) + off),　(PC) ← (PC) + 2
Affected flags	P1 ← REG8

[Description]

Transfers the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off to the accumulator (A).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

	A	RAM 0000H	RAM 0001H	RAM 0056H	RAM 004CH
LDW #0051H	51H	--	--	--	--
STW R0	51H	51H	00H	--	--
MOV #00H, 56H	51H	51H	00H	00H	--
MOV #88H, 4CH	51H	51H	00H	00H	88H
LD [05H]	00H	51H	00H	00H	88H
LD [-5]	88H	51H	00H	00H	88H

5-92

LD dst (LoaD direct byte to accumulator)

Instruction code	0000H≦dst＜0100H	: [84H][dst_L(7-0)]	(2bytes)	83H-86H
	0100H≦dst＜0200H	: [85H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [83H][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [86H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [86H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2 ┆ 3			
Cycle count	1 ┆ 2			
Function	(A) ← (dst), (PC) ← (PC) + (2 or 3)			
Affected flags	P1 ← REG8			

[Description]

Transfers the contents of the data memory (RAM) or special function register (SFR) designated by dst to the accumulator (A).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0011H	RAM 0012H
MOV	#050H, A	50H	--	--
MOV	#051H, 11H	50H	51H	--
MOV	#052H, 12H	50H	51H	52H
LD	11H	51H	51H	52H
LD	12H	52H	51H	52H

LDL dst (LoaD Long range direct byte to accumulator)

Instruction code	0000H≦dst≦ffffH : [86H][dst_L(7-0)][dst_H(7-0)]	86H
Byte count	3	
Cycle count	2	
Function	(A) ← (dst), (PC) ← (PC) + 3	
Affected flags	P1 ← REG8	

[Description]

Transfers the contents of the data memory (RAM) or special function register (SFR) designated by dst to the accumulator (A).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register having bit 8 to bit 1 of the program status word (PSW).

[Example]

		A	RAM 0201H	RAM 0202H
MOV	#050H, A	50H	--	--
MOVL	#051H, 0201H	50H	51H	--
MOVL	#052H, 0202H	50H	51H	52H
LDL	0201H	51H	51H	52H
LDL	0202H	52H	51H	52H

LDCW [Rn] (LoaD indirect rom Code Word to register pair BA)

Instruction code	[1 0 0 0 1 0 0 0][0 n5n4n3n2n1n0 0] 88H
Byte count	2
Cycle count	4
Function	(BA) ← ROMw((Rn)), (PC) ← (PC) + 2
Affected flags	

[Description]
This instruction is used to refer to the entire ROM data area on the bank designated by the table lookup bank flag (LDCBNK).
The instruction transfers the contents of the word program memory (ROM) that is designated by the contents of the indirect address pair register (Rn) designated by n5-n0 (17 bits (9 higher-order bits (Rn+1) + 8 lower-order bits (Rn)) to the BA register pair (BA). Subsequently, the CPU increments the program counter (PC) by 2.
The valid value range of n is $0 \leqq n < 63$.

[Example]

		B	A	RAM 0005H	RAM 0004H
CHGP1	0	--	--	--	--
MOV	#03EH, 004H	--	--	--	03EH
MOV	#001H, 005H	--	--	001H	03EH
LDCW	[R2]	30H	00H	001H	03EH
MOV	#00AH, PSW	30H	00H	001H	03EH
MOV	#002H, 005H	30H	00H	102H	03EH
LDCW	[R2]	EAH	00H	102H	03EH

BNK	PC	ROM
0	0013EH	00H
0	0013FH	30H
1	1023EH	00H
1	1023FH	EAH

LDCW [Rn,C] (LoaD indirect (with C reg.) rom Code Word to register pair BA)

Instruction code	[1 0 0 0 1 0 0 0][0 n5n4n3n2n1n0 1] 88H
Byte count	2
Cycle count	4
Function	(BA) ← ROMw((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	

[Description]

This instruction is used to refer to the entire ROM data area on the bank designated by the table lookup bank flag (LDCBNK).

The instruction transfers the contents of the word program memory (ROM) that is designated by the sum of the contents of the indirect address pair register (Rn) designated by n5-n0 (17 bits (9 higher-order bits (Rn+1) + 8 lower-order bits (Rn)) and the contents of the C register (C) (signed 8-bit data) to the BA register pair (BA). Subsequently, the CPU increments the program counter (PC) by 2.

The valid value range of n is $0 \leqq n < 63$.

C register: 8-bit signed indirect displacement data $(-128 \leqq C \text{ reg.} \leqq 127)$

[Example]

		B	A	C	RAM 0005H	RAM 0004H
CHGP1	0	--	--	--	--	--
MOV	#000H, 004H	--	--	--	--	000H
MOV	#001H, 005H	--	--	--	001H	000H
MOV	#-2, C	--	--	FEH	001H	000H
LDCW	[R2, C]	30H	00H	FEH	001H	000H
MOV	#00AH, PSW	30H	00H	FEH	001H	000H
MOV	#001H, 005H	30H	00H	FEH	101H	000H
MOV	#000H, C	30H	00H	00H	101H	000H
LDCW	[R2, C]	EAH	00H	00H	101H	000H

BNK	PC	ROM
0	000FEH	00H
0	000FFH	30H
1	10100H	00H
1	10101H	EAH

LDCW [off] (LoaD indirect (with displacement) rom Code Word to register pair BA)

Instruction code	[1 0 0 0 1 0 0 0][1 off7off6off5off4off3off2off1off0] 88H
Byte count	2
Cycle count	4
Function	(BA) ← ROMw((R0) + off), (PC) ← (PC) + 2
Affected flags	

[Description]

This instruction is used to refere to the entire ROM data area on the bank designated by the table lookup bank flag (LDCBNK).

The instruction transfers the contents of the word program memory (ROM) that is designated by the sum of the contents of the indirect address pair register (R0) (17 bits (9 higher-order bits (R0+1) + 8 lower-order bits (R0)) and the 7-bit signed indirect address offset data (off) to the BA register pair (BA). Subsequently, the CPU increments the program counter (PC) by 2.

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

			B	A	RAM 0001H	RAM 0000H
CHGP1	0		--	--	--	--
MOV	#000H, 000H		--	--	--	000H
MOV	#001H, 001H		--	--	001H	000H
LDCW	[-2]		30H	00H	001H	000H
LDCW	[00H]		FFH	00H	001H	000H
MOV	#00AH, PSW		FFH	00H	001H	000H
MOV	#001H, 001H		FFH	00H	101H	000H
LDCW	[-2]		57H	00H	101H	000H
LDCW	[00H]		EAH	00H	101H	000H

BNK	PC	ROM
0	000FEH	00H
0	000FFH	30H
0	00100H	00H
0	00101H	FFH
1	100FEH	00H
1	100FFH	57H
1	10100H	00H
1	10101H	EAH

LDW #wi (LoaD immediate Word data to register pair BA)

Instruction code	[0 1 0 0 0 1 1 1][wi_L(7-0)][wi_H(7-0)]	47H
Byte count	3	
Cycle count	1	
Function	(BA) ← wi, (PC) ← (PC) + 3	
Affected flags		

[Description]

Transfers the lower-order byte of immediate data (wi) to the accumulator (A) and the higher-order byte to the B register.

[Example]

		A	B
LDW	#01234H	34H	12H
LDW	#04321H	21H	43H
LDW	#0FF00H	00H	FFH
LDW	#000FFH	FFH	00H

LDW [Rn] (LoaD indirect Word data to register pair BA)

Instruction code	[0 0 0 0 0 1 1 1][0 n5n4n3n2n1n0 0] 07H
Byte count	2
Cycle count	2 or 3 (fe00H-ffffH)
Function	(BA) ← w((Rn)), (PC) ←(PC) + 2
Affected flags	P1 ← REGH8

[Description]
Transfers the lower- and higher-order bytes of the contents of the word data memory (RAM) or special function register (SFR) designated by the indirect address register Rn to the accumulator (A) and B register, respectively.
The instruction also transfers the bit 8 (highest-order address bit) of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).
The valid value range of n is $0 \leq n \leq 63$.

[Example]

R1 points to addresses 02H and 03H	PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0055H	RAM 0056H	A	B
LDW #0055H	01EFBH	475500H	--	--	--	--	55H	00H
STW R1	01EFEH	970200H	55H	00H	--	--	55H	00H
LDW #1234H	01F01H	473412H	55H	00H	--	--	34H	12H
STW [R1]	01F04H	1702H	55H	00H	34H	12H	34H	12H
LDW #5678H	01F06H	477856H	55H	00H	34H	12H	78H	56H
LDW [R1]	01F09H	0702H	55H	00H	34H	12H	34H	12H
NOP	01F0BH	00H	55H	00H	34H	12H	34H	12H

<Programming Note>

If a BA-to-BA transfer is attempted with the LDW instruction, the B register will be loaded with FFH and the A register with an unpredictable value.
[Example]:   LDW   #0FE00H
            STW   R5
            LDW   [R5]   Designates BA indirectly.

LDW [Rn,C] (LoaD indirect (with C reg.) Word data to register pair BA)

Instruction code	[0 0 0 0 0 1 1 1][0 n5n4n3n2n1n0 1] 07H
Byte count	2
Cycle count	2 or 3 (fe00H-ffffH)
Function	(BA) ← w((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	P1← REGH8

[Description]

Transfers the lower- and higher-order bytes of the contents of the word data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register to the accumulator (A) and B register, respectively.

The instruction also transfers the bit 8 (highest-order address bit) of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leqq$ C reg.$\leqq 127$)

[Example]

R1 points to addresses 02H and 03H

		PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0059H	RAM 005AH	A	B	C
MOV	#04H, C	01EF8H	430204H	--	--	--	--	--	--	04H
LDW	#0055H	01EFBH	475500H	--	--	--	--	55H	00H	04H
STW	R1	01EFEH	970200H	55H	00H	--	--	55H	00H	04H
LDW	#1234H	01F01H	473412H	55H	00H	--	--	34H	12H	04H
STW	[R1, C]	01F04H	1703H	55H	00H	34H	12H	34H	12H	04H
LDW	#5678H	01F06H	477856H	55H	00H	34H	12H	78H	56H	04H
LDW	[R1, C]	01F09H	0703H	55H	00H	34H	12H	34H	12H	04H
NOP		01F0BH	00H	55H	00H	34H	12H	34H	12H	04H

<Programming Note>
If a BA-to-BA transfer is attempted with the LDW instruction, the B register will be loaded with FFH and the A register with an unpredictable value.

[Example]:
```
           MOV      #00H,C
           LDW      #0FE00H
           STW      R5
           LDW      [R5,C]              Designates BA indirectly.
```

LDW [off] (LoaD indirect (with displacement) Word data to register pair BA)

Instruction code	[0 0 0 0 0 1 1 1][1 off6off5off4off3off2off1off0] 07H
Byte count	2
Cycle count	2 or 3 (fe00H-ffffH)
Function	(BA) ← w((R0) + off), (PC) ← (PC) + 2
Affected flags	P1 ← REGH8

[Description]

Transfers the lower- and higher-order bytes of the contents of the word data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off to the accumulator (A) and B register, respectively.

The instruction also transfers the bit 8 (highest-order address bit) of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H		PC	Instruction Code	RAM 0000H	RAM 0001H	RAM 0059H	RAM 005AH	A	B
LDW	#0055H	01EFBH	475500H	--	--	--	--	55H	00H
STW	R0	01EFEH	970000H	55H	00H	--	--	55H	00H
LDW	#1234H	01F01H	473412H	55H	00H	--	--	34H	12H
STW	[04H]	01F04H	1784H	55H	00H	34H	12H	34H	12H
LDW	#5678H	01F06H	477856H	55H	00H	34H	12H	78H	56H
LDW	[04H]	01F09H	0784H	55H	00H	34H	12H	34H	12H
NOP		01F0BH	00H	55H	00H	34H	12H	34H	12H

<Programming Note>
If a BA-to-BA transfer is attempted with the LDW instruction, the B register will be loaded with FFH and the A register with an unpredictable value.

[Example]:　　LDW　　　#0FE00H
　　　　　　　　STW　　　R0
　　　　　　　　LDW　　　[00H]　　　　　　　　　Designates BA indirectly.

LDW dst (LoaD long range direct Word data to register pair BA)

Instruction code	[1 0 0 0 0 1 1 1][dst_L(7-0)][dst_H(7-0)] 87H
Byte count	3
Cycle count	2 or 3 (fe00H-ffffH)
Function	(BA) ← w(dst), (PC) ← (PC) + 3
Affected flags	P1 ← REGH8

[Description]

Transfers the lower- and higher-order bytes of the contents of the word data memory (RAM) or special function register (SFR) designated by dst to the accumulator (A) and B register, respectively.

The instruction also transfers the bit 8 (highest-order address bit) of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).

[Example]

		PC	Instruction Code	RAM 00C0H	RAM 00C1H	RAM FE12H	RAM FE13H	A	B
LDW	#1234H	01EFBH	473412H	--	--	--	--	34H	12H
STW	00C0H	01EFEH	97C000H	34H	12H	--	--	34H	12H
LDW	#5678H	01F01H	477856H	34H	12H	--	--	78H	56H
STW	0FE12H	01F04H	9712FEH	34H	12H	78H	56H	78H	56H
LDW	00C0H	01F07H	87C000H	34H	12H	78H	56H	34H	12H
LDW	0FE12H	01F0AH	8712FEH	34H	12H	78H	56H	78H	56H
NOP		01F0DH	00H	34H	12H	78H	56H	78H	56H

<Programming Note>

If a BA-to-BA transfer is attempted with the LDW instruction, the B register will be loaded with FFH and the A register with an unpredictable value.

[Example]:   LDW   0FE00H   Designates BA directly.

LDX [Rn] (LoaD indirect eXternal memory byte to accumulator)

Instruction code	[0 1 1 1 0 0 0 1][0 n5n4n3n2n1n0 0] 71H	
Byte count	2	
Cycle count	4	
Function	(A) ← ext((Rn) + (B) * 10000H), (PC) ← (PC) + 2	
Affected flags		

[Description]

Transfers, to the accumulator (A), the contents of the external memory (EXT RAM) designated by the sum of the contents of the indirect address pair register (Rn) designated by n5-n0 and the contents of the B register (B) shifted 16 bits to the left.

Subsequently, the CPU increments the program counter (PC) by 2.

The valid value range of n is $0 \le n \le 63$.

[Example 1]

		A	B	RAM 0005H	RAM 0004H	EXT RAM 200100H
MOV	#000H, 004H	--	--	--	00H	--
MOV	#001H, 005H	--	--	01H	00H	--
MOV	#020H, B	--	20H	01H	00H	--
MOV	#0FDH, A	FDH	20H	01H	00H	--
STX	[R2]	FDH	20H	01H	00H	FDH
LDX	[R2]	FDH	20H	01H	00H	FDH

[Example 2]

		A	B	RAM 0007H	RAM 0006H	EXT RAM 20017FH
MOV	#07FH, 006H	--	--	--	7FH	--
MOV	#001H, 007H	--	--	01H	7FH	--
MOV	#020H, B	--	20H	01H	7FH	--
MOV	#0FDH, A	FDH	20H	01H	7FH	--
STX	[R3]	FDH	20H	01H	7FH	FDH
LDX	[R3]	FDH	20H	01H	7FH	FDH

LDX [Rn,C] (LoaD indirect (with C reg.) eXternal memory byte to accumulator)

Instruction code	[0 1 1 1 0 0 0 1][0 n5n4n3n2n1n0 1]	71H
Byte count	2	
Cycle count	4	
Function	(A) ← ext((Rn) + (C) + (B) * 10000H), (PC) ← (PC) + 2	
Affected flags		

[Description]

Transfers, to the accumulator (A), the contents of the external memory (EXT RAM) designated by the sum of the contents of the indirect address pair register (Rn) designated by n5-n0, the contents of the C register, and the contents of the B register (B) shifted 16 bits to the left.

Subsequently, the CPU increments the program counter (PC) by 2.

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data (-128 $\leqq$ C reg. $<$ 127)

[Example 1]

		A	B	C	RAM 0005H	RAM 0004H	EXT RAM 20013FH
MOV	#000H, 004H	--	--	--	--	00H	--
MOV	#001H, 005H	--	--	--	01H	00H	--
MOV	#020H, B	--	20H	--	01H	00H	--
MOV	#03FH, C	--	20H	3FH	01H	00H	--
MOV	#0FDH, A	FDH	20H	3FH	01H	00H	--
STX	[R2, C]	FDH	20H	3FH	01H	00H	FDH
LDX	[R2, C]	FDH	20H	3FH	01H	00H	FDH

[Example 2]

		A	B	C	RAM 0007H	RAM 0006H	EXT RAM 2000C0H
MOV	#000H, 006H	--	--	--	--	00H	--
MOV	#001H, 007H	--	--	--	01H	00H	--
MOV	#020H, B	--	20H	--	01H	00H	--
MOV	#-64, C	--	20H	C0H	01H	00H	--
MOV	#0FDH, A	FDH	20H	C0H	01H	00H	--
STX	[R3, C]	FDH	20H	C0H	01H	00H	FDH
LDX	[R3, C]	FDH	20H	C0H	01H	00H	FDH

LDX [off] (LoaD indirect (with displacement) eXternal memory byte to accumulator)

Instruction code	[0 1 1 1 0 0 0 1][1 off7off6off5off4off3off2off1off0] 71H
Byte count	2
Cycle count	4
Function	(A) ← ext((R0) + off + (B) ∗ 10000H), (PC) ← (PC) + 2
Affected flags	

[Description]

Transfers, to the accumulator (A), the contents of the external memory (EXT RAM) designated by the sum of the contents of the indirect address pair register (R0), 7-bit signed indirect address offset data (off), and the contents of the B register (B) shifted 16 bits to the left.

Subsequently, the CPU increments the program counter (PC) by 2.

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example 1]

		A	B	RAM 0001H	RAM 0000H	EXT RAM 20013FH
MOV	#000H, 000H	--	--	--	00H	--
MOV	#001H, 001H	--	--	01H	00H	--
MOV	#020H, B	--	20H	01H	00H	--
MOV	#0FDH, A	FDH	20H	01H	00H	--
STX	[3FH]	FDH	20H	01H	00H	FDH
LDX	[3FH]	FDH	20H	01H	00H	FDH

[Example 2]

		A	B	RAM 0001H	RAM 0000H	EXT RAM 2000C0H
MOV	#000H, 000H	--	--	--	00H	--
MOV	#001H, 001H	--	--	01H	00H	--
MOV	#020H, B	--	20H	01H	00H	--
MOV	#0FDH, A	FDH	20H	01H	00H	--
STX	[-64]	FDH	20H	01H	00H	FDH
LDX	[-64]	FDH	20H	01H	00H	FDH

MOV #i, [Rn] (MOVe immediate data to indirect byte)

Instruction code	[0 1 0 0 0 0 1 0][0 n5n4n3n2n1n0 0][i7i6i5i4i3i2i1i0] 42H
Byte count	3
Cycle count	2
Function	((Rn)) ← i, (PC) ← (PC) + 3
Affected flags	REG8 ← P1

[Description]

Transfers immediate data (i) to the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R8 points to addresses 10H and 11H

R9 points to addresses 12H and 13H

		RAM 0010H	RAM 0011H	RAM 0012H	RAM 0013H	RAM 0200H	RAM 0301H
LDW	#0200H	--	--	--	--	--	--
STW	R8	00H	02H	--	--	--	--
LDW	#0301H	00H	02H	--	--	--	--
STW	R9	00H	02H	01H	03H	--	--
MOV	#055H, [R8]	00H	02H	01H	03H	55H	--
MOV	#0AAH, [R9]	00H	02H	01H	03H	55H	AAH

MOV #i, [Rn,C] (MOVe immediate data to indirect(with C register) byte)

Instruction code	[0 1 0 0 0 0 1 0][0 n5n4n3n2n1n0 0][i7i6i5i4i3i2i1i0] 42H
Byte count	3
Cycle count	2
Function	$((Rn) + (C)) \leftarrow i$, $(PC) \leftarrow (PC) + 3$
Affected flags	REG8 $\leftarrow$ P1

[Description]

Transfers immediate data (i) to the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

The valid value range of n is $0 \leq n \leq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leq$ C reg. ≤ 127)

[Example]

	RAM 0010H	RAM 0011H	C	RAM 0105H	RAM 0106H
R8 points to addresses 10H and 11H					
LDW #0100H	--	--	--	--	--
STW R8	00H	01H	--	--	--
MOV #05H,C	00H	01H	05H	--	--
MOV #055H, [R8,C]	00H	01H	05H	55H	--
INC C	00H	01H	06H	55H	--
MOV #0AAH, [R8,C]	00H	01H	06H	55H	AAH

5-107

MOV #i, [off] (MOVe immediate data to indirect (withdisplacement) byte)

Instruction code	[0 1 0 0 0 0 1 0][1 off6off5off4off3off2off1off0][i7i6i5i4i3i2i1i0] 42H
Byte count	3
Cycle count	2
Function	$((R0) + off) \leftarrow i$, $(PC) \leftarrow (PC) + 3$
Affected flags	REG8 $\leftarrow$ P1

[Description]

Transfers immediate data (i) to the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

off: 7-bit signed indirect address offset data $(-64 \leqq off \leqq 63)$

[Example]

R0 points to addresses 00H and 01H

	RAM 0000H	RAM 0001H	RAM 0106H	RAM 00FAH	RAM 0100H
LDW #0100H	--	--	--	--	--
STW R0	00H	01H	--	--	--
MOV #055H, [06H]	00H	01H	55H	--	--
MOV #082H, [-6]	00H	01H	55H	82H	--
MOV #032H, [00H]	00H	01H	55H	82H	32H

MOV #i, dst (MOVe immediate data to direct byte)

Instruction code	0000H≦dst＜0100H : [44H][dst_L(7-0)][i(7-0)] (3bytes) 43H-46H 0100H≦dst＜0200H : [45H][dst_L(7-0)][i(7-0)] (3bytes) fe00H≦dst＜ff00H : [43H][dst_L(7-0)][i(7-0)] (3bytes) 0200H≦dst＜fe00H : [46H][dst_L(7-0)][dst_H(7-0)][i(7-0)] (4bytes) ff00H≦dst≦ffffH : [46H][dst_L(7-0)][dst_H(7-0)][i(7-0)] (4bytes)
Byte count	3 ┊ 4
Cycle count	1 ┊ 2
Function	(dst) ← i, (PC) ← (PC) + (3 or 4)
Affected flags	REG8 ← P1

[Description]

Transfers immediate data (i) to the data memory (RAM) or special function register (SFR) designated by dst.
The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		RAM 0010H	RAM 0011H	RAM 0012H	RAM 0013H	RAM 0014H
MOV	#050H, 10H	50H	--	--	--	--
MOV	#051H, 11H	50H	51H	--	--	--
MOV	#052H, 12H	50H	51H	52H	--	--
MOV	#053H, 13H	50H	51H	52H	53H	--
MOV	#054H, 14H	50H	51H	52H	53H	54H

MOVL #i, dst (MOVe immediate data to Long range direct byte)

Instruction code	0000H≦dst≦ffffH : [46H][dst_L(7-0)][dst_H(7-0)][i(7-0)] 46H
Byte count	4
Cycle count	2
Function	(dst) ← i, (PC) ← (PC) + 4
Affected flags	REG8 ← P1

[Description]
Transfers immediate data (i) to the data memory (RAM) or special function register (SFR) designated by dst.
The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

[Example]

	RAM 0200H	RAM 0201H	RAM 0202H	RAM 0203H	RAM 0204H
MOVL #050H, 0200H	50H	--	--	--	--
MOVL #051H, 0201H	50H	51H	--	--	--
MOVL #052H, 0202H	50H	51H	52H	--	--
MOVL #053H, 0203H	50H	51H	52H	53H	--
MOVL #054H, 0204H	50H	51H	52H	53H	54H

MUL16 (MULtiplication 16bit * 8bit = 24bit)

Instruction code	[0 1 0 0 0 0 0 0][0 1 0 0 0 0 0 0]	40H
Byte count	2	
Cycle count	5	
Function	(C)(B)(A) ← ((SP) −1) ((SP) −2) * ((SP))	
	if((C) = 00H) (OV) ← 0 else (OV) ← 1,	
	(SP) ← (SP) − 3, (PC) ← (PC) + 2	
Affected flags	OV	

[Description]

Multiplies 16-bit unsigned data consisting of the contents of the data memory (RAM) pointed to by the stack pointer (SP) −1 (higher-order byte) and the contents of RAM pointed to by SP2 (lower-order byte) by 8-bit unsigned data consisting of the contents of RAM pointed to by SP. The highest-order 8 bits of the 24-bit result of multiplication are placed in the C register (C), the higher-order 8 bits in the B register (B), and the lower-order 8 bits in the accumulator (A). The overflow flag (OV) is reset if the contents of C are zero and set if the contents of B are nonzero. Subsequently, the CPU decrements the SP by 3 and finally increments the program counter (PC) by 2.

[Example 1]

		C	B	A	SP	RAM 0022H	RAM 0021H	RAM 0020H	OV
MOV	#000H, PSW	--	--	--	--	--	--	--	0
MOV	#01FH, SPL	--	--	--	--	--	--	--	0
MOV	#000H, SPH	--	--	--	001FH	--	--	--	0
PUSH	#023H	--	--	--	0020H	--	--	23H	0
PUSH	#011H	--	--	--	0021H	--	11H	23H	0
PUSH	#052H	--	--	--	0022H	52H	11H	23H	0
MUL16		05H	7DH	36H	001FH	52H	11H	23H	1

[Example 2]

		C	B	A	SP	RAM 0022H	RAM 0021H	RAM 0020H	OV
MOV	#004H, PSW	--	--	--	--	--	--	--	1
MOV	#01FH, SPL	--	--	--	--	--	--	--	1
MOV	#000H, SPH	--	--	--	001FH	--	--	--	1
PUSH	#005H	--	--	--	0020H	--	--	05H	1
PUSH	#007H	--	--	--	0021H	--	07H	05H	1
PUSH	#010H	--	--	--	0022H	10H	07H	05H	1
MUL16		00H	70H	50H	001FH	10H	07H	05H	0

MUL24 (**MULtiplication 24bit * 16bit = 40bit**)

Instruction code	[0 1 0 0 0 0 0 0][1 1 0 0 0 0 0 0] 40H
Byte count	2
Cycle count	12
Function	$((SP) - 3)((SP) - 4)(C)(B)(A)$ $\leftarrow ((SP) - 2)((SP) - 3)((SP) - 4) * ((SP))((SP) - 1)$ if$(((SP) - 3)((SP) - 4) = 00H)$ $(OV) \leftarrow 0$ else $(OV) \leftarrow 1$, $(SP) \leftarrow (SP) - 3$, $(PC) \leftarrow (PC) + 2$
Affected flags	OV, RAM8 $\leftarrow$ 1

[Description]

Multiplies 24-bit unsigned data consisting of the contents of the data memory (RAM) pointed to by the stack pointer (SP)–2 (highest-order byte), the contents of RAM pointed to by SP–3 (higher-order byte), and the contents of RAM pointed to by SP–4 (lower-order byte) by 16-bit unsigned data consisting of the contents of RAM pointed to by SP (higher-order byte) and the contents of RAM pointed to by SP–1 (lower-order byte). Bits 39-32 of the 40-bit result of multiplication are placed in RAM pointed to by SP–3, bits 31-24 in RAM pointed to by SP–4, bits 23-16 in the C register (C), bits 15-8 in the B register (B), and bits 7-0 in the accumulator (A).
The overflow flag (OV) is reset if the contents of RAM designated by both SP–3 and S̄P4 are zero and set if they are nonzero. Subsequently, the CPU decrements the SP by 3 and finally increments the program counter (PC) by 2.

[Example]

		C	B	A	SP	RAM 0024H	RAM 0023H	RAM 0022H	RAM 0021H	RAM 0020H	OV
MOV	#000H, PSW	--	--	--	--	--	--	--	--	--	0
MOV	#01FH, SPL	--	--	--	--	--	--	--	--	--	0
MOV	#000H, SPH	--	--	--	001FH	--	--	--	--	--	0
PUSH	#000H	--	--	--	0020H	--	--	--	--	00H	0
PUSH	#005H	--	--	--	0021H	--	--	--	05H	00H	0
PUSH	#007H	--	--	--	0022H	--	--	07H	05H	00H	0
PUSH	#000H	--	--	--	0023H	--	00H	07H	05H	00H	0
PUSH	#010H	--	--	--	0024H	10H	00H	07H	05H	00H	0
MUL24		50H	00H	00H	0021H	10H	00H	07H	00H	70H	1

NOP (Non Operation)

Instruction code	[0 0 0 0 0 0 0 0]	00H
Byte count	1	
Cycle count	1	
Function	(PC) ← (PC) + 1	
Affected flags		

[Description]
Consumes one machine cycle and does nothing.

NOT1 dst, bit (Not direct bit)

Instruction code			A8H-AFH, B8H-BFH, B7H
	0H≦dst＜100H	: [bit][A8H][dst_L(7-0)]	(2bytes)
	fe00H≦dst＜ff00H	: [bit][B8H][dst_L(7-0)]	(2bytes)
	100H≦dst＜2000H	: [B7H][dst_L(7-0)][bit&&dst_H]	(3bytes)
Byte count	2	3	
Cycle count	1	2	
Function	(dst).bit ← not{(dst).bit},　(PC) ← (PC) + (2 or 3)		
Affected flags			

[Description]
Inverts the state of the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) pointed to by dst. Subsequently, the CPU increments the program counter (PC) by 2 or 3.
The byte count and cycle count vary according to the address value of the operand (dst).

[Example 1]

		RAM 0010H	
MOV	#000H, 010H	00H	0000 0000B
NOT1	010H, 7	80H	1000 0000B
NOT1	010H, 7	00H	0000 0000B

[Example 2]

		A	
MOV	#001H, A	01H	0000 0001B
NOT1	A, 6	41H	0400 0001B
NOT1	A, 6	01H	0000 0001B

NOT1M dst, bit (Not Middle range direct bit)

Instruction code	0H≦dst＜2000H ：[B7H][dst_L(7-0)][bit&&dst_H] (3bytes) B7H
Byte count	3
Cycle count	2
Function	(dst).bit ← not {(dst).bit} , (PC) ← (PC) + 3
Affected flags	

[Description]
Inverts the state of the bit designated by bits 2-0 of the data memory (RAM) pointed to by dst12-dst0 (0≦dst＜
2000H). Subsequently, the CPU increments the program counter (PC) by 3.

[Example 1]

		RAM 0200H	
MOVL	#000H, 00200H	00H	0000 0000B
NOT1M	00200H, 7	80H	1000 0000B
NOT1M	00200H, 7	00H	0000 0000B

[Example 2]

		RAM 037FH	
MOVL	#001H, 0037FH	01H	0000 0001B
NOT1M	0037FH, 6	41H	0100 0001B
NOT1M	0037FH, 6	01H	0000 0001B

OR #i (OR immediate data to accumulator)

Instruction code	[1 1 1 0 0 0 0 1][i7i6i5i4i3i2i1i0]	E1H
Byte count	2	
Cycle count	1	
Function	(A) ← (A)	i, (PC)← (PC) + 2	
Affected flags		

[Description]

Takes the OR of the contents of the accumulator (A) and immediate data (i) and places the result in the accumulator (A).

[Example]

		A
MOV	#00H, A	00H
OR	#03H	03H
OR	#0CH	0FH
OR	#30H	3FH
OR	#0C0H	FFH

5-116

OR [Rn] (OR indirect byte to accumulator)

Instruction code	[1 1 1 0 0 0 1 0][0 n5n4n3n2n1n0 0]	E2H
Byte count	2	
Cycle count	2	
Function	(A) ← (A) │ ((Rn)), (PC) ← (PC) + 2	
Affected flags		

[Description]

Takes the OR of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

		RAM 0002H	RAM 0003H	A	RAM 0055H
R1 points to addresses 02H and 03H					
LDW	#0055H	--	--	55H	--
STW	R1	55H	00H	55H	--
MOV	#22H, [R1]	55H	00H	55H	22H
OR	[R1]	55H	00H	77H	22H
ROL	[R1]	55H	00H	77H	44H
OR	[R1]	55H	00H	77H	44H
ROL	[R1]	55H	00H	77H	88H
OR	[R1]	55H	00H	FFH	88H

OR [Rn, C] (OR indirect byte(with C register) to accumulator)

Instruction code	[1 1 1 0 0 0 1 0][0 n5n4n3n2n1n0 1] E2H
Byte count	2
Cycle count	2
Function	(A) ← (A)	((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	

[Description]

Takes the OR of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leqq$ C reg. $\leqq 127$)

[Example]

R1 points to addresses 02H and 03H

		C	RAM 0002H	RAM 0003H	A	RAM 0059H
LDW	#0055H	--	--	--	55H	--
STW	R1	--	55H	00H	55H	--
MOV	#04H, C	04H	55H	00H	55H	--
MOV	#00H, A	04H	55H	00H	00H	--
MOV	#03H, [R1, C]	04H	55H	00H	00H	03H
OR	[R1, C]	04H	55H	00H	03H	03H
MOV	#0FCH, [R1, C]	04H	55H	00H	03H	FCH
OR	[R1, C]	04H	55H	00H	FFH	FCH

OR [off] (OR indirect byte (with displacement) to accumulator)

Instruction code	[1 1 1 0 0 0 1 0][1 off6off5off4off3off2off1off0]	E2H
Byte count	2	
Cycle count	2	
Function	(A) ← (A)	((R0) + off), (PC) ← (PC) + 2	
Affected flags		

[Description]
Takes the OR of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off and places the result in the accumulator (A).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

	RAM 0000H	RAM 0001H	A	RAM 0059H
R0 points to addresses 00H and 01H				
LDW #0055H	--	--	55H	--
STW R0	55H	00H	55H	--
MOV #00H, A	55H	00H	00H	--
MOV #15H, [04H]	55H	00H	00H	15H
OR [04H]	55H	00H	15H	15H
MOV #0EAH, [04H]	55H	00H	15H	EAH
OR [04H]	55H	00H	FFH	EAH

OR dst (OR direct byte to accumulator)

Instruction code	0000H≦dst<0100H	: [E4H][dst_L(7-0)]	(2bytes)	E3H-E6H
	0100H≦dst<0200H	: [E5H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst<ff00H	: [E3H][dst_L(7-0)]	(2bytes)	
	0200H≦dst<fe00H	: [E6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [E6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2	3		
Cycle count	1	2		
Function	(A) ← (A) │ (dst), (PC) ← (PC) + (2 or 3)			
Affected flags				

[Description]
Takes the OR of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0034H
MOV	#00H, A	00H	--
MOV	#03H, 34H	00H	03H
OR	34H	03H	03H
MOV	#3CH, 34H	03H	3CH
OR	34H	3FH	3CH
MOV	#0C0H, 34H	3FH	C0H
OR	34H	FFH	C0H

ORL dst (OR Long range direct byte to accumulator)

Instruction code	0000H≦dst≦ffffH : [E6H][dst_L(7-0)][dst_H(7-0)] (3bytes) E6H
Byte count	3
Cycle count	2
Function	(A) ← (A)	(dst), (PC) ← (PC) + (2 or 3)
Affected flags	

[Description]

Takes the OR of the contents of the accumulator (A) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

[Example]

		A	RAM 0200H
MOV	#00H, A	00H	--
MOV	#03H, 200H	00H	03H
ORL	200H	03H	03H
MOV	#3CH, 200H	03H	3CH
ORL	200H	3FH	3CH
MOV	#0C0H, 200H	3FH	C0H
ORL	200H	FFH	C0H

POP [Rn] (POP indirect byte from stack)

Instruction code	[0 1 1 1 0 0 1 0][0 n5n4n3n2n1n0 0] 72H
Byte count	2
Cycle count	2
Function	((Rn)) ← ((SP)−−), (PC) ← (PC) + 2
Affected flags	REG8 ← RAM8, P1← RAM8

[Description]

Transfers the contents of the data memory (RAM) pointed to by the stack pointer (SP) to the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn. Subsequently, the CPU decrements the stack pointer (SP).

The instruction also transfers bit 8 of the data memory (RAM) pointed to by the stack pointer (SP) to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

		SPL FE0AH	SPH FE0BH	RAM 0002H	RAM 0003H	RAM 0051H	RAM 00C8H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
LDW	#0051H	C7H	00H	--	--	--	--
STW	R1	C7H	00H	51H	00H	--	--
PUSH	#0BBH	C8H	00H	51H	00H	--	BBH
POP	[R1]	C7H	00H	51H	00H	BBH	BBH

POP [Rn, C] (POP indirect (with C register) byte from stack)

Instruction code	[0 1 1 1 0 0 1 0][0 n5n4n3n2n1n0 1] 72H
Byte count	2
Cycle count	2
Function	((Rn) + (C)) ← ((SP)−−), (PC) ← (PC) + 2
Affected flags	REG8 ← RAM8, P1 ← RAM8

[Description]

Transfers the contents of the data memory (RAM) pointed to by the stack pointer (SP) to the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register. Subsequently, the CPU decrements the stack pointer (SP).

The instruction also transfers bit 8 of the data memory (RAM) pointed to by the stack pointer (SP) to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leqq$ C reg.$\leqq 127$)

[Example]

R1 points to addresses 02H and 03H

		C	SPL FE0AH	SPH FE0BH	RAM 0002H	RAM 0003H	RAM 0056H	RAM 00C8H
MOV	#05H, C	05H	--	--	--	--	--	--
LDW	#00C7H	05H	--	--	--	--	--	--
STW	SP	05H	C7H	00H	--	--	--	--
LDW	#0051H	05H	C7H	00H	--	--	--	--
STW	R1	05H	C7H	00H	51H	00H	--	--
PUSH	#0BBH	05H	C8H	00H	51H	00H	--	BBH
POP	[R1,C]	05H	C7H	00H	51H	00H	BBH	BBH

POP [off] (POP indirect (with displacement) byte from stack)

Instruction code	[0 1 1 1 0 0 1 0][1 off6off5off4off3off2off1off0] 72H
Byte count	2
Cycle count	2
Function	((R0) + off) ← ((SP)−−), (PC) ← (PC) + 2
Affected flags	REG8 ← RAM8, P1← RAM8

[Description]

Transfers the contents of the data memory (RAM) pointed to by the stack pointer (SP) to the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off. Subsequently, the CPU decrements the stack pointer (SP).

The instruction also transfers bit 8 of the data memory (RAM) pointed to by the stack pointer (SP) to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

		SPL FE0AH	SPH FE0BH	RAM 0000H	RAM 0001H	RAM 0056H	RAM 00C8H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
LDW	#0051H	C7H	00H	--	--	--	--
STW	R0	C7H	00H	51H	00H	--	--
PUSH	#0AAH	C8H	00H	51H	00H	--	AAH
POP	[05H]	C7H	00H	51H	00H	AAH	AAH

POP dst (PUSH direct byte from stack)

Instruction code	0000H≦dst＜0100H : [74H][dst_L(7-0)] (2bytes) 73H-76H
	0100H≦dst＜0200H : [75H][dst_L(7-0)] (2bytes)
	fe00H≦dst＜ff00H : [73H][dst_L(7-0)] (2bytes)
	0200H≦dst＜fe00H : [76H][dst_L(7-0)][dst_H(7-0)] (3bytes)
	ff00H≦dst≦ffffH : [76H][dst_L(7-0)][dst_H(7-0)] (3bytes)
Byte count	2 ┊ 3
Cycle count	1 ┊ 2
Function	(dst) ← ((SP)−−), (PC) ← (PC) + (2 or 3)
Affected flags	REG8 ← RAM8, P1 ← RAM8

[Description]

Transfers the contents of the data memory (RAM) pointed to by the stack pointer (SP) to the data memory (RAM) or special function register (SFR) designated by dst. Subsequently, the CPU decrements the stack pointer (SP).

The instruction also transfers bit 8 of the data memory (RAM) pointed to by the stack pointer (SP) bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		SPL FE0AH	SPH FE0BH	RAM 0055H	RAM 0056H	RAM 00C8H	RAM 00C9H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
PUSH	#0AAH	C8H	00H	--	--	AAH	--
PUSH	#0BBH	C9H	00H	--	--	AAH	BBH
POP	55H	C8H	00H	BBH	--	AAH	BBH
POP	56H	C7H	00H	BBH	AAH	AAH	BBH

POPL dst (POP Long range direct byte from stack)

Instruction code	0000H≦dst≦ffffH　　: [76H][dst_L(7-0)][dst_H(7-0)]　　(3bytes)　　　　　76H
Byte count	3
Cycle count	2
Function	(dst) ← ((SP)−−),　　(PC) ← (PC) + 3
Affected flags	REG8 ← RAM8,　　P1 ← RAM8

[Description]

Transfers the contents of the data memory (RAM) pointed to by the stack pointer (SP) to the data memory (RAM) or special function register (SFR) designated by dst. Subsequently, the CPU decrements the stack pointer (SP).

The instruction also transfers bit 8 of the data memory (RAM) pointed to by the stack pointer (SP) bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8.

[Example]

		SPL FE0AH	SPH FE0BH	RAM 0200H	RAM 0201H	RAM 00C8H	RAM 00C9H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
PUSH	#0AAH	C8H	00H	--	--	AAH	--
PUSH	#0BBH	C9H	00H	--	--	AAH	BBH
POPL	0200H	C8H	00H	BBH	--	AAH	BBH
POPL	0201H	C7H	00H	BBH	AAH	AAH	BBH

POP__BA (POP register pair BA from stack)

Instruction code	[0 1 1 1 0 0 0 0]	70H
Byte count	1	
Cycle count	2	
Function	(SP)−−,　(BA) ← w((SP)−−),　(PC) ← (PC) + 1	
Affected flags	P1 ← RAMH8	

[Description]

Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the BA register pair (BA), then decrements the stack pointer (SP). The instruction finally increments the program counter (PC).

The instruction also transfers bit 8 of the word data memory (RAM) pointed to by SP to bit 1 (P1) of the program status word (PSW).

[Example]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH,SPL	--	--	--	--	--
MOV	#000H,SPH	--	--	001FH	--	--
MOV	#0AAH, A	--	AAH	001FH	--	--
MOV	#000H, B	00H	AAH	001FH	--	--
PUSH_BA		00H	AAH	0021H	00H	AAH
POP_BA		00H	AAH	001FH	00H	AAH

POP__P (POP PSW from stack)

Instruction code	[1 0 0 1 0 0 0 0]	90H
Byte count	1	
Cycle count	1	
Function	(P) ← ((SP)−−),　(PC) ← (PC) + 1	
Affected flags	P1 ← RAM (bit1)	

[Description]

Transfers the contents of the data memory (RAM) pointed to by SP to the program status word (PSW) and decrements the stack pointer (SP). The instruction finally increments the program counter (PC).

The instruction also transfers bit 1 of the data memory (RAM) pointed to by SP to bit 1 (P1) of the program status word (PSW).

[Example]

		PSW	SP	RAM 0020H
MOV	#01FH,SPL	--	--	--
MOV	#000H,SPH	--	001FH	--
MOV	#0AAH, PSW	AAH	001FH	--
PUSH_P		AAH	0020H	AAH
POP_P		AAH	001FH	AAH

5-128

POPW [Rn] (POP indirect Word from stack)

Instruction code	[0 0 1 1 0 1 1 1][0 n5n4n3n2n1n0 0] 37H
Byte count	2
Cycle count	3
Function	(SP)−−, w((Rn)) ← w((SP)−−), (PC) ← (PC) + 2
Affected flags	REGH8 ← RAMH8, REGL8 ← RAML8, P1 ← RAMH8

[Description]
Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the word data memory (RAM) or special function register (SFR) designated by the contents of the indirect address pair register (Rn), then decrements the stack pointer (SP). The instruction finally increments the program counter (PC) by 2.

The instruction also transfers bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the higher-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8 and to bit 1 (P1) of the program status word (PSW). It also transfers bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the lower-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8.

The valid value range of n is $0 \leq n \leq 63$.

[Example]

		SP	RAM 013FH	RAM 013EH	RAM 0021H	RAM 0020H	RAM 0005H	RAM 0004H
MOV	#01FH,SPL	--	--	--	--	--	--	--
MOV	#000H,SPH	001FH	--	--	--	--	--	--
MOV	#0AAH,13EH	001FH	--	AAH	--	--	--	--
MOV	#000H,13FH	001FH	00H	AAH	--	--	--	--
MOV	#03EH,004H	001FH	00H	AAH	--	--	--	3EH
MOV	#001H,005H	001FH	00H	AAH	--	--	01H	3EH
PUSHW	[R2]	0021H	00H	AAH	00H	AAH	01H	3EH
POPW	[R2]	001FH	00H	AAH	00H	AAH	01H	3EH

POPW [Rn,C] (POP indirect(with C reg.) Word from stack)

Instruction code	[0 0 1 1 0 1 1 1][0 n5n4n3n2n1n0 1] 37H
Byte count	2
Cycle count	3
Function	(SP)−−, w((Rn) + (C)) ← w((SP)−−), (PC) ← (PC) + 2
Affected flags	REGH8 ← RAMH8, REGL8 ← RAML8, P1← RAMH8

[Description]

Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the word data memory (RAM) or special function register (SFR) designated by the sum of the contents of the indirect address pair register (Rn) and the contents of the C register (C) (signed 8-bit data), then decrements the stack pointer (SP). The instruction finally increments the program counter (PC) by 2.

The instruction also transfers bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the higher-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8 and to bit 1 (P1) of the program status word (PSW). It also transfers bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the lower-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8.

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data (-128≦C reg.≦127)

[Example]

		C	SP	RAM 013FH	RAM 013EH	RAM 0021H	RAM 0020H	RAM 0005H	RAM 0004H
MOV	#01FH,SPL	--	--	--	--	--	--	--	--
MOV	#000H,SPH	--	001FH	--	--	--	--	--	--
MOV	#0AAH,13EH	--	001FH	--	AAH	--	--	--	--
MOV	#000H,13FH	--	001FH	00H	AAH	--	--	--	--
MOV	#000H,004H	--	001FH	00H	AAH	--	--	--	00H
MOV	#001H,005H	--	001FH	00H	AAH	--	--	01H	00H
MOV	#03EH,C	3EH	001FH	00H	AAH	--	--	01H	00H
PUSHW	[R2, C]	3EH	0021H	00H	AAH	00H	AAH	01H	00H
POPW	[R2, C]	3EH	001FH	00H	AAH	00H	AAH	01H	00H

POPW [off] (POP indirect (with displacement) Word from stack)

Instruction code	[0 0 1 1 0 1 1 1][1 off6off5off4off3off2off1off0]	37H
Byte count	2	
Cycle count	3	
Function	(SP)−−, w((R0) + off) ← w((SP)−−), (PC) ← (PC) + 2	
Affected flags	REGH8 ← RAMH8, REGL8 ← RAML8, P1 ← RAMH8	

[Description]
Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the word data memory (RAM) or special function register (SFR) designated by the sum of the contents of the indirect address pair register (R0) and 7-bit signed indirect address offset data (off), then decrements the stack pointer (SP). The instruction finally increments the program counter (PC) by 2.
The instruction also transfers bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the higher-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8 and to bit 1 (P1) of the program status word (PSW). It also transfers bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the lower-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8.
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

		SP	RAM 013FH	RAM 013EH	RAM 0021H	RAM 0020H	RAM 0001H	RAM 0000H
MOV	#01FH,SPL	--	--	--	--	--	--	--
MOV	#000H,SPH	001FH	--	--	--	--	--	--
MOV	#0AAH,13EH	001FH	--	AAH	--	--	--	--
MOV	#000H,13FH	001FH	00H	AAH	--	--	--	--
MOV	#000H,000H	001FH	00H	AAH	--	--	--	00H
MOV	#001H,001H	001FH	00H	AAH	--	--	01H	00H
PUSHW	[3EH]	0021H	00H	AAH	00H	AAH	01H	00H
POPW	[3EH]	001FH	00H	AAH	00H	AAH	01H	00H

POPW dst (POP Long range direct Word from stack)

Instruction code	[0 1 1 1 0 1 1 1][dst7dst6dst5dst4dst3dst2dst1dst0] 77H [dst15dst14dst13dst12dst11dst10dst9dst8]
Byte count	3
Cycle count	3
Function	(SP)−−, w(dst) ← w((SP)−−), (PC) ← (PC) + 3
Affected flags	REGH8 ← RAMH8, REGL8 ← RAML8, P1 ← RAMH8

[Description]

Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the word data memory (RAM) or special function register (SFR) designated by dst15-dst0, then decrements the stack pointer (SP). The instruction finally increments the program counter (PC) by 3.

The instruction also transfers bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the higher-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8 and to bit 1 (P1) of the program status word (PSW). It also transfers bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP to bit 8 of the lower-order byte of the Rn-designated word data memory (RAM) or special function register (SFR) having bit 8.

[Example]

	SP	RAM 013FH	RAM 013EH	RAM 0021H	RAM 0020H
MOV #01FH,SPL	--	--	--	--	--
MOV #000H,SPH	001FH	--	--	--	--
MOV #0AAH,13EH	001FH	--	AAH	--	--
MOV #000H,13FH	001FH	00H	AAH	--	--
PUSHW 0013EH	0021H	00H	AAH	00H	AAH
POPW 0013EH	001FH	00H	AAH	00H	AAH

PUSH #I (PUSH immediate data to stack)

Instruction code	[0 1 1 0 0 0 0 1][i7i6i5i4i3i2i1i0] 61H
Byte count	2
Cycle count	1
Function	(++(SP)) ← i, (PC)← (PC) + 2
Affected flags	RAM8 ← P1

[Description]

Increments the stack pointer (SP), then transfers immediate data (i) to the data memory (RAM) pointed to by the stack pointer (SP).

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory (RAM).

[Example]

		SPL FE0AH	SPH FE0BH	RAM 00C8H	RAM 00C9H	RAM 00CAH
LDW	#00C7H	--	--	--	--	--
STW	SP	C7H	00H	--	--	--
PUSH	#12H	C8H	00H	12H	--	--
PUSH	#34H	C9H	00H	12H	34H	--
PUSH	#56H	CAH	00H	12H	34H	56H

PUSH [Rn] (PUSH indirect byte to stack)

Instruction code	[0 1 1 0 0 0 1 0][0 n5n4n3n2n1n0 0]	62H
Byte count	2	
Cycle count	2	
Function	$(++(SP)) \leftarrow ((Rn))$, $(PC) \leftarrow (PC) + 2$	
Affected flags	RAM8 $\leftarrow$ REG8, P1 $\leftarrow$ REG8	

[Description]

Increments the stack pointer (SP), then transfers the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn to the data memory (RAM) pointed to by the stack pointer (SP).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

		SPL FE0AH	SPH FE0BH	RAM 0002H	RAM 0003H	RAM 0051H	RAM 00C8H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
LDW	#0051H	C7H	00H	--	--	--	--
STW	R1	C7H	00H	51H	00H	--	--
MOV	#0BBH, [R1]	C7H	00H	51H	00H	BBH	--
PUSH	[R1]	C8H	00H	51H	00H	BBH	BBH

5-134

PUSH [Rn, C] (PUSH indirect (with C register) byte to stack)

Instruction code	[0 1 1 0 0 0 1 0][0 n5n4n3n2n1n0 1] 62H
Byte count	2
Cycle count	2
Function	$(++(SP)) \leftarrow ((Rn) + (C))$, $(PC) \leftarrow (PC) + 2$
Affected flags	RAM8 $\leftarrow$ REG8, P1 $\leftarrow$ REG8

[Description]

Increments the stack pointer (SP), then transfers the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register to the data memory (RAM) pointed to by the stack pointer (SP).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data $(-128 \leqq C\ reg. \leqq 127)$

[Example]

R1 points to addresses 02H and 03H

		C	SPL FE0AH	SPH FE0BH	RAM 0002H	RAM 0003H	RAM 0056H	RAM 00C8H
MOV	#05H, C	05H	--	--	--	--	--	--
LDW	#00C7H	05H	--	--	--	--	--	--
STW	SP	05H	C7H	00H	--	--	--	--
LDW	#0051H	05H	C7H	00H	--	--	--	--
STW	R1	05H	C7H	00H	51H	00H	--	--
MOV	#0BBH, [R1,C]	05H	C7H	00H	51H	00H	BBH	--
PUSH	[R1,C]	05H	C8H	00H	51H	00H	BBH	BBH

PUSH [off] (PUSH indirect (with displacement) byte to stack)

Instruction code	[0 1 1 0 0 0 1 0][1 off6off5off4off3off2off1off0] 62H
Byte count	2
Cycle count	2
Function	(++(SP)) ← ((R0) + off), (PC) ← (PC) + 2
Affected flags	RAM8 ← REG8, P1 ← REG8

[Description]

Increments the stack pointer (SP), then transfers the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off to the data memory (RAM) pointed to by the stack pointer (SP).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM).

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

	SPL FE0AH	SPH FE0BH	RAM 0000H	RAM 0001H	RAM 0056H	RAM 00C8H
LDW #00C7H	--	--	--	--	--	--
STW SP	C7H	00H	--	--	--	--
LDW #0051H	C7H	00H	--	--	--	--
STW R0	C7H	00H	51H	00H	--	--
MOV #0AAH, [05H]	C7H	00H	51H	00H	AAH	--
PUSH [05H]	C8H	00H	51H	00H	AAH	AAH

PUSH dst (PUSH direct byte to stack)

Instruction code	0000H≦dst＜0100H : [64H][dst_L(7-0)] (2bytes) 63H-66H
	0100H≦dst＜0200H : [65H][dst_L(7-0)] (2bytes)
	fe00H≦dst＜ff00H : [63H][dst_L(7-0)] (2bytes)
	0200H≦dst＜fe00H : [66H][dst_L(7-0)][dst_H(7-0)] (3bytes)
	ff00H≦dst≦ffffH : [66H][dst_L(7-0)][dst_H(7-0)] (3bytes)
Byte count	2 ┊ 3
Cycle count	1 ┊ 2
Function	(++(SP)) ← (dst), (PC) ← (PC) + (2 or 3)
Affected flags	RAM8 ← REG8, P1 ← REG8

[Description]

Increments the stack pointer (SP), then transfers the contents of the data memory (RAM) or special function register (SFR) designated by dst to the data memory (RAM) pointed to by the stack pointer (SP).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM).

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		SPL FE0AH	SPH FE0BH	RAM 0055H	RAM 0056H	RAM 00C8H	RAM 00C9H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
MOV	#0AAH, 55H	C7H	00H	AAH	--	--	--
MOV	#0BBH, 56H	C7H	00H	AAH	BBH	--	--
PUSH	55H	C8H	00H	AAH	BBH	AAH	--
PUSH	56H	C9H	00H	AAH	BBH	AAH	BBH

PUSHL dst (PUSH Long range byte to stack)

Instruction code	0000H≦dst≦ffffH : [66H][dst_L(7-0)][dst_H(7-0)] (3bytes) 66H
Byte count	3
Cycle count	2
Function	(++(SP)) ← (dst), (PC) ← (PC) + 3
Affected flags	RAM8 ← REG8, P1 ← REG8

[Description]

Increments the stack pointer (SP), then transfers the contents of the data memory (RAM) or special function register (SFR) designated by dst to the data memory (RAM) pointed to by the stack pointer (SP).

The instruction also transfers bit 8 of the designated data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the designated data memory (RAM).

[Example]

		SPL FE0AH	SPH FE0BH	RAM 0200H	RAM 0201H	RAM 00C8H	RAM 00C9H
LDW	#00C7H	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--
MOV	#0AAH, 200H	C7H	00H	AAH	--	--	--
MOV	#0BBH, 201H	C7H	00H	AAH	BBH	--	--
PUSHL	200H	C8H	00H	AAH	BBH	AAH	--
PUSHL	201H	C9H	00H	AAH	BBH	AAH	BBH

PUSH__BA (PUSH register pair BA to stack)

Instruction code	[0 1 1 0 0 0 0 0] 60H
Byte count	1
Cycle count	1
Function	w(++(SP)) ← (BA), ++(SP), (PC) ← (PC) + 1
Affected flags	RAMH8 ←P1, RAML8 ← P1

[Description]

Increments the stack pointer (SP), transfers the contents of the BA register pair (BA) to the word data memory (RAM) pointed to by SP, then increments the SP. The instruction finally increments the program counter (PC).
The instruction also transfers bit 1 (P1) of the program status word (PSW) to bit 8 of the higher- and lower-order bytes of the word data memory (RAM) pointed to by SP.

[Example]

		B	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH,SPL	--	--	--	--	--
MOV	#000H,SPH	--	--	001FH	--	--
MOV	#055H, A	--	55H	001FH	--	--
MOV	#000H, B	00H	55H	001FH	--	--
CLR1	PSW, 1	00H	55H	001FH	--	--
PUSH_BA		00H	55H	0021H	00H	55H
POP_BA		00H	55H	001FH	00H	55H

5-139

PUSH__P (PUSH PSW to stack)

Instruction code	[1 0 0 0 0 0 0 0] 80H
Byte count	1
Cycle count	1
Function	(++(SP)) ← (P), (PC) ← (PC) + 1
Affected flags	RAM8 ← P1

[Description]

Increments the stack pointer (SP), transfers the contents of the program status word (PSW) to the data memory (RAM) pointed to by SP, then increments the program counter (PC).

The instruction also transfers bit 1 (P1) of the program status word (PSW) to bit 8 of the data memory (RAM) pointed to by SP.

[Example]

		PSW	SP	RAM 0020H
MOV	#01FH,SPL	--	--	--
MOV	#000H,SPH	--	001FH	--
MOV	#055H, PSW	55H	001FH	--
PUSH_P		55H	0020H	55H
POP_P		55H	001FH	55H

PUSHW [Rn] (PUSH indirect Word to stack)

Instruction code	[0 0 1 0 0 1 1 1][0 n5n4n3n2n1n0 0] 27H
Byte count	2
Cycle count	3
Function	w (++(SP)) ← w ((Rn)), ++(SP), (PC) ← (PC) + 2
Affected flags	RAMH8 ← REGH8, RAML8 ← REGL8, P1← REGH8

[Description]
Increments the stack pointer (SP), transfers the contents of the word data memory (RAM) or special function register (SFR) designated by the indirect address register Rn to the data memory (RAM) pointed to by SP, then increments the SP. The instruction finally increments the program counter (PC) by 2.
The instruction transfers bit 8 of the higher-order byte of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the higher-order byte of the word data memory (RAM). It also transfers bit 8 of the lower-order byte to bit 8 of the lower-order byte of the word data memory (RAM).
The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

		SPL FE0AH	SPH FE0BH	RAM 0002H	RAM 0003H	RAM 0051H	RAM 0052H	RAM 00C8H	RAM 00C9H
R1 points to addresses 02H and 03H									
LDW	#00C7H	--	--	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--	--	--
LDW	#0051H	C7H	00H	--	--	--	--	--	--
STW	R1	C7H	00H	51H	00H	--	--	--	--
LDW	#1234H	C7H	00H	51H	00H	--	--	--	--
STW	[R1]	C7H	00H	51H	00H	34H	12H	--	--
PUSHW	[R1]	C9H	00H	51H	00H	34H	12H	34H	12H

<Programming Note>

The PUSHW instruction will not push the correct data if it is executed on the stack pointer itself (SP).
Exercise care not to manipulate the stack pointer itself with the PUSHW instruction.
[Example]:   LDW   #0FE0AH   <------- References the SP.
                STW   R3
                PUSHW   [R3]   <------- NG (The SP is manipulated.)

PUSHW [Rn,C] (PUSH indirect (with C reg.) Word to stack)

Instruction code	[0 0 1 0 0 1 1 1][1 n5n4n3n2n1n0 1]	27H
Byte count	2	
Cycle count	3	
Function	w(++(SP)) ← w((Rn) + (C)), ++(SP), (PC) ← (PC) + 2	
Affected flags	RAMH8 ← REGH8, RAML8 ← REGL8, P1← REGH8	

[Description]

Increments the stack pointer (SP), transfers the contents of the word data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register to the data memory (RAM) pointed to by SP, then increments the SP. The instruction finally increments the program counter (PC) by 2.

The instruction transfers bit 8 of the higher-order byte of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the higher-order byte of the word data memory (RAM). It also transfers bit 8 of the lower-order byte to bit 8 of the lower-order byte of the word data memory (RAM).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data (-128 $\leqq$ C reg. $\leqq$ 127)

[Example]

R1 points to addresses 02H and 03H	SPL FE0AH	SPH FE0BH	RAM 0002H	RAM 0003H	RAM 0055H	RAM 0056H	RAM 00C8H	RAM 00C9H	C
MOV #04H,C	--	--	--	--	--	--	--	--	04H
LDW #00C7H	--	--	--	--	--	--	--	--	04H
STW SP	C7H	00H	--	--	--	--	--	--	04H
LDW #0051H	C7H	00H	--	--	--	--	--	--	04H
STW R1	C7H	00H	51H	00H	--	--	--	--	04H
LDW #1234H	C7H	00H	51H	00H	--	--	--	--	04H
STW [R1,C]	C7H	00H	51H	00H	34H	12H	--	--	04H
PUSHW [R1,C]	C9H	00H	51H	00H	34H	12H	34H	12H	04H

<Programming Note>

The PUSHW instruction will not push the correct data if it is executed on the stack pointer itself (SP). Exercise care not to manipulate the stack pointer itself with the PUSHW instruction.
[Example]:   LDW   #0FE00H
               STW   R3
               MOV   #0AH,C
               PUSHW [R3,C]   <-------- NG (The SP is manipulated.)

PUSHW [off] (PUSH indirect (with displacement) Word to stack)

Instruction code	[0 0 1 0 0 1 1 1][1 off6off5off4off3off2off1off0]	27H
Byte count	2	
Cycle count	3	
Function	w(++(SP)) ← w((R0) + off), ++(SP), (PC) ← (PC) + 2	
Affected flags	RAMH8 ← REGH8, RAML8 ← REGL8, P1 ← REGH8	

[Description]
Increments the stack pointer (SP), transfers the contents of the word data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off to the data memory (RAM) pointed to by SP, then increments the SP. The instruction finally increments the program counter (PC) by 2.
The instruction transfers bit 8 of the higher-order byte of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the higher-order byte of the word data memory (RAM). It also transfers bit 8 of the lower-order byte to bit 8 of the lower-order byte of the word data memory (RAM).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

		SPL FE0AH	SPH FE0BH	RAM 0000H	RAM 0001H	RAM 0055H	RAM 0056H	RAM 00C8H	RAM 00C9H
R0 points to addresses 00H and 01H									
LDW	#00C7H	--	--	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--	--	--
LDW	#0051H	C7H	00H	--	--	--	--	--	--
STW	R0	C7H	00H	51H	00H	--	--	--	--
LDW	#1234H	C7H	00H	51H	00H	--	--	--	--
STW	[04H]	C7H	00H	51H	00H	34H	12H	--	--
PUSHW	[04H]	C9H	00H	51H	00H	34H	12H	34H	12H

<Programming Note>
The PUSHW instruction will not push the correct data if it is executed on the stack pointer itself (SP).
Exercise care not to manipulate the stack pointer itself with the PUSHW instruction.

[Example]:	LDW	#0FE00H	
	STW	R0	
	PUSHW	[0AH]	<-------- NG (The SP is manipulated.)

PUSHW dst (PUSH Long range direct Word to stack)

Instruction code	[0 1 1 0 0 1 1 1][dst_L(7-0)][dst_H(7-0)]	67H
Byte count	3	
Cycle count	3	
Function	w(++(SP)) ← w(dst), ++(SP), (PC) ← (PC) + 3	
Affected flags	RAMH8 ← REGH8, RAML8 ← REGL8, P1 ← REGH8	

[Description]

Increments the stack pointer (SP), transfers the contents of the word data memory (RAM) or special function register (SFR) designated by dst to the data memory (RAM) pointed to by SP, then increments the SP. The instruction finally increments the program counter (PC) by 3.

The instruction transfers bit 8 of the higher-order byte of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW) and bit 8 of the higher-order byte of the word data memory (RAM). It also transfers bit 8 of the lower-order byte to bit 8 of the lower-order byte of the word data memory (RAM).

[Example]

		SPL FE0AH	SPH FE0BH	RAM 00C0H	RAM 00C1H	RAM 00C8H	RAM 00C9H	RAM 00CAH	RAM 00CBH
LDW	#00C7H	--	--	--	--	--	--	--	--
STW	SP	C7H	00H	--	--	--	--	--	--
LDW	#1234H	C7H	00H	--	--	--	--	--	--
STW	00C0H	C7H	00H	34H	12H	--	--	--	--
PUSHW	00C0H	C9H	00H	34H	12H	34H	12H	--	--
XCH	B	C9H	00H	34H	12H	34H	12H	--	--
STW	00C0H	C9H	00H	12H	34H	34H	12H	--	--
PUSHW	00C0H	CBH	00H	12H	34H	34H	12H	12H	34H

<Programming Note>

The PUSHW instruction will not push the correct data if it is executed on the stack pointer itself (SP).

Exercise care not to manipulate the stack pointer itself with the PUSHW instruction.

[Example]: PUSHW   0FE0AH   <--------NG (The SP is manipulated.)

RCALL r12 (CALL Relative address)

Instruction code	[0 1 0 r11 1 r10r9r8][r7r6r5r4r3r2r1r0]	48H-4FH,58H-5FH
Byte count	2	
Cycle count	2	
Function	w(++(SP)) ← (PC), ++(SP), (PC) ← (PC) + 2 + r12	
Affected flags	RAMH8 ← PC (bit16), RAML8 ← BNK	

[Description]

Increments the stack pointer (SP), stores the address of the instruction immediately following the RCALL instruction (return address) in the word data memory (RAM) pointed to by SP, then increments the SP. The instruction finally adds the address data of the instruction immediately following the RCALL instruction to data designated by r11-r0 and places the result in the program counter (PC).

The instruction also transfers bit 16 of the program counter (PC) to bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP and the bank flag (BNK) to bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP.

[Example 1]

The value of label LA is 01100H.

			PC	Instruction Code	SP	RAM 0021H	RAM 0020H
	MOV	#01FH, SPL	00FF7H	430A1FH	--	--	--
	MOV	#000H, SPH	00FFAH	430B00H	001FH	--	--
	RCALL	LA	00FFDH	4901H	0021H	0FH	FFH
LA:	INC	A	01100H	8B00H	0021H	0FH	FFH
	RET		01102H	A0H	001FH	0FH	FFH
	NOP		00FFFH	00H	001FH	0FH	FFH

[Example 2]

The value of label LA is 01100H.

			PC	Instruction Code	SP	RAM 0021H	RAM 0020H
	MOV	#01FH, SPL	00FF8H	430A1FH	--	--	--
	MOV	#000H, SPH	00FFBH	430B00H	001FH	--	--
	RCALL	LA	00FFEH	4900H	0021H	10H	00H
LA:	INC	A	01100H	8B00H	0021H	10H	00H
	RET		01102H	A0H	001FH	10H	00H
	INC	A	01000H	8B00H	001FH	10H	00H

RCALLA (CALL Relative address with Accumulator)

Instruction code	[0 0 0 1 0 0 0 0] 10H
Byte count	1
Cycle count	2
Function	w(++(SP)) ← (PC), ++(SP), (PC) ← (PC) + 1 + (A)
	(A): unsigned byte hanndling
Affected flags	RAMH8 ← PC (bit16), RAML8 ← BNK

[Description]

Increments the stack pointer (SP), stores the address of the instruction immediately following the RCALLA instruction (return address) in the word data memory (RAM) pointed to by SP, then increments the SP. The instruction finally adds the address data of the instruction immediately following the RCALLA instruction to the 8-bit unsigned data in the accumulator (A) and places the result in the program counter (PC).

The instruction also transfers bit 16 of the program counter (PC) to bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP and the bank flag (BNK) to bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP.

[Example 1]

		PC	Instruction Code	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	00FF5H	430A1FH	--	--	--	--
MOV	#000H, SPH	00FF8H	430B00H	--	001FH	--	--
MOV	#081H, A	00FFBH	430081H	81H	001FH	--	--
RCALLA		00FFEH	10H	81H	0021H	0FH	FFH
INC	A	01080H	8B00H	81H	0021H	0FH	FFH
RET		01082H	A0H	81H	001FH	0FH	FFH
NOP		00FFFH	00H	81H	001FH	0FH	FFH

[Example 2]

		PC	Instruction Code	A	SP	RAM 0021H	RAM 0020H
MOV	#01FH, SPL	00FF6H	430A1FH	--	--	--	--
MOV	#000H, SPH	00FF9H	430B00H	--	001FH	--	--
MOV	#081H, A	00FFCH	430081H	81H	001FH	--	--
RCALLA		00FFFH	10H	81H	0021H	10H	00H
INC	A	01080H	8B00H	81H	0021H	10H	00H
RET		01082H	A0H	81H	001FH	10H	00H
INC	A	01000H	8B00H	81H	001FH	10H	00H

RET (RETurn from subroutine)

Instruction code	[1 0 1 0 0 0 0 0]	A0H
Byte count	1	
Cycle count	2	
Function	(SP)−−, (PC) ← w((SP)−−)	
Affected flags	BNK ← RAML8, PC (bit16) ← RAMH8	

[Description]
Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the program counter (PC), then decrements the SP.
The instruction also transfers bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP to the bank flag bit (BNK) and bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP to bit 16 of the PC.

[Example 1]
The value of label LA is 00F0EH.

			BNK	PC	Instruction Code	SP	RAM 0021H	RAM 0020H
	MOV	#01FH, SPL	0	00FF6H	430A1FH	--	--	--
	MOV	#000H, SPH	0	00FF9H	430B00H	001FH	--	--
	CALL	LA	0	00FFCH	300E0FH	0021H	00FH	0FFH
LA:	INC	A	0	00F0EH	8B00H	0021H	00FH	0FFH
	RET		0	00F10H	A0H	001FH	00FH	0FFH
	NOP		0	00FFFH	00H	001FH	00FH	0FFH

[Example 2]
The value of label LA is 00F0EH.

			BNK	PC	Instruction Code	SP	RAM 0021H	RAM 0020H
	MOV	#01FH, SPL	0	00FF7H	430A1FH	--	--	--
	MOV	#000H, SPH	0	00FFAH	430B00H	001FH	--	--
	CALL	LA	0	00FFDH	300E0FH	0021H	010H	000H
LA:	INC	A	0	00F0EH	8B00H	0021H	010H	000H
	RET		0	00F10H	A0H	001FH	010H	000H
	INC	A	0	01000H	8B00H	001FH	010H	000H

RETI (RETurn from Interrupt)

Instruction code	[1 0 1 1 0 0 0 0] B0H
Byte count	1
Cycle count	2
Function	(SP)−−, (PC) ← w((SP)−−), pop interrupt level
Affected flags	BNK ← RAML8, PC (bit16) ← RAMH8

[Description]

Decrements the stack pointer (SP), transfers the contents of the word data memory (RAM) pointed to by SP to the program counter (PC), decrements the SP, and enables interrupts that were disabled when an interrupt was accepted.

The instruction also transfers bit 8 of the lower-order byte of the word data memory (RAM) pointed to by SP to the bank flag bit (BNK) and bit 8 of the higher-order byte of the word data memory (RAM) pointed to by SP to bit 16 of the PC.

[Example 1]

		BNK	PC	Instruction Code	
NOP		0	00FFAH	00H	
NOP		0	00FFBH	00H	
MOV	#001H, A	0	00FFCH	430001H	← An external interrupt 0
INC	A	0	00003H	8B00H	occurs.
RETI		0	00005H	B0H	
NOP		0	00FFFH	00H	

[Example 2]

		BNK	PC	Instruction Code	
NOP		0	00FFCH	00H	
MOV	#00EH, B	0	00FFDH	43010EH	← An external interrupt 1
INC	A	0	0000BH	8B00H	occurs.
RETI		0	0000DH	B0H	
INC	A	0	01000H	8B00H	

ROL (ROtate accumulator Left)

Instruction code	[1 1 1 0 0 0 0 0]	E0H
Byte count	1	
Cycle count	1	
Function	┌─ A7←A6←A5←A4←A3←A2←A1←A0←┐	(PC) ← (PC) + 2
Affected flags		

[Description]
Rotates the 8-bit data in the accumulator 1 bit to the left.
The bits of the accumulator (A) are shifted as follows:

bit 0 to bit 1
bit 1 to bit 2
bit 2 to bit 3
bit 3 to bit 4
bit 4 to bit 5
bit 5 to bit 6
bit 6 to bit 7
bit 7 to bit 0

[Example]

		A	
MOV	#01H, A	01H	0000 0001B
ROL		02H	0000 0010B
ROL		04H	0000 0100B
ROL		08H	0000 1000B
ROL		10H	0001 0000B
ROL		20H	0010 0000B
ROL		40H	0100 0000B
ROL		80H	1000 0000B
ROL		01H	0000 0001B

ROLC (ROtate accumulator Left through the Carry)

Instruction code	[1 1 1 1 0 0 0 0] F0H
Byte count	1
Cycle count	1
Function	CY←A7←A6←A5←A4←A3←A2←A1←A0← (PC) ← (PC) + 2
Affected flags	CY

[Description]

Rotates 8-bit data in the accumulator 1 bit to the left through the carry flag.

The bits of the accumulator (A) are shifted as follows:

bit 0 to bit 1
bit 1 to bit 2
bit 2 to bit 3
bit 3 to bit 4
bit 4 to bit 5
bit 5 to bit 6
bit 6 to bit 7
bit 7 to CY
CY to bit 0

[Example]

		A		CY
MOV	#01H, A	01H	0000 0001B	-
CLR1	PSW, 7	01H	0000 0001B	0
ROLC		02H	0000 0010B	0
ROLC		04H	0000 0100B	0
ROLC		08H	0000 1000B	0
ROLC		19H	0001 0000B	0
ROLC		20H	0010 0000B	0
ROLC		40H	0100 0000B	0
ROLC		80H	1000 0000B	0
ROLC		00H	0000 0000B	1
ROLC		01H	0000 0001B	0

ROR (ROtate accumulator Right)

Instruction code	[1 1 0 0 0 0 0 0]	C0H
Byte count	1	
Cycle count	1	
Function	→ A7→A6→A5→A4→A3→A2→A1→A0 ─┐	(PC) ← (PC) + 2
Affected flags		

[Description]
Rotates the 8-bit data in the accumulator 1 bit to the right.
The bits of the accumulator (A) are shifted as follows:

bit 0 to bit 7
bit 1 to bit 0
bit 2 to bit 1
bit 3 to bit 2
bit 4 to bit 3
bit 5 to bit 4
bit 6 to bit 5
bit 7 to bit 6

[Example]

		A	
MOV	#01H, A	01H	0000 0001B
ROR		80H	1000 0000B
ROR		40H	0100 0000B
ROR		20H	0010 0000B
ROR		10H	0001 0000B
ROR		08H	0000 1000B
ROR		04H	0000 0100B
ROR		02H	0000 0010B
ROR		01H	0000 0001B

RORC (ROtate accumulator Right through the Carry)

Instruction code	[1 1 0 1 0 0 0 0] D0H
Byte count	1
Cycle count	1
Function	$\rightarrow$CY$\rightarrow$A7$\rightarrow$A6$\rightarrow$A5$\rightarrow$A4$\rightarrow$A3$\rightarrow$A2$\rightarrow$A1$\rightarrow$A0 $\quad$ (PC) $\leftarrow$ (PC) + 2
Affected flags	CY

[Description]

Rotates 8-bit data in the accumulator 1 bit to the right through the carry flag.

The bits of the accumulator (A) are shifted as follows:

bit 0 to CY
bit 1 to bit 0
bit 2 to bit 1
bit 3 to bit 2
bit 4 to bit 3
bit 5 to bit 4
bit 6 to bit 5
bit 7 to b it 6
CY to bit 7

[Example]

		A	CY
MOV	#01H, A	01H	0000 0001B	-
CLR1	PSW, 7	01H	0000 0001B	0
RORC		00H	0000 0000B	1
RORC		80H	1000 0000B	0
RORC		40H	0100 0000B	0
RORC		20H	0010 0000B	0
RORC		10H	0001 0000B	0
RORC		08H	0000 1000B	0
RORC		04H	0000 0100B	0
RORC		02H	0000 0010B	0
RORC		01H	0000 0001B	0

SET1 dst,bit (SET direct bit)

Instruction code		E8H-EFH, F8H-FFH, F7H
	0H≦dst＜100H : [bit][E8H][dst_L(7-0)]	(2bytes)
	fe00H≦dst＜ff00H : [bit][F8H][dst_L(7-0)]	(2bytes)
	100H≦dst＜2000H : [F7H][dst_L(7-0)][bit&&dst_H]	(3bytes)
Byte count	2 ┊ 3	
Cycle count	1 ┊ 2	
Function	(dst).bit ← 1, (PC) ← (PC) + (2 or 3)	
Affected flags		

[Description]
Sets the bit designated by bits 2-0 of the data memory (RAM) or special function register (SFR) designated by dst. Subsequently, the CPU increments the program counter (PC) by 2 or 3.
The byte count and cycle count vary according to the address value of the operand (dst).

[Example 1]

	RAM 0010H
MOV #000H, 010H	00H	0000 0000B
SET1 010H, 7	80H	1000 0000B

	RAM 0010H	
MOV #000H, 010H	00H	0000 0000B
SET1 010H, 7	80H	1000 0000B

[Example 2]

	A	
MOV #001H, A	01H	0000 0001B
SET1 A, 6	41H	0100 0001B

SET1M dst, bit (SET Middle range direct bit)

Instruction code	0H≦dst＜2000H　　　: [F7H][dst_L(7-0)][bit&&dst_H]　　　(3bytes)　　F7H
Byte count	3
Cycle count	2
Function	(dst)**.**bit ← 1,　(PC) ← (PC) + 3
Affected flags	

[Description]
Sets the bit designated by bits 2-0 of the data memory (RAM) designated by dst12-dst0 (0≦dst＜2000H).
Subsequently, the CPU increments the program counter (PC) by 3.

[Example 1]

	RAM 0200H	
MOV #000H, 00200H	00H	0000 0000B
SET1M 00200H, 7	80H	1000 0000B

[Example 2]

	RAM 037FH	
MOVL #001H, 0037FH	01H	0000 0001B
SET1M 0037FH, 6	41H	0100 0001B

ST [Rn] (STore accumulator to indirect byte)

Instruction code	[1 0 0 1 0 0 1 0][0 n5n4n3n2n1n0 0] 92H
Byte count	2
Cycle count	2
Function	((Rn)) ← (A), (PC) ← (PC) + 2
Affected flags	REG8 ← P1

[Description]

Transfers the contents of the accumulator (A) to the data memory (RAM) or special function register designated by the indirect address register Rn.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory or special function register (SFR) having bit 8.

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

	A	RAM 0002H	RAM 0003H	RAM 0051H
LDW #0051H	51H	--	--	--
STW R1	51H	51H	00H	--
MOV #0FFH, A	FFH	51H	00H	--
MOV #00H, [R1]	FFH	51H	00H	00H
ST [R1]	FFH	51H	00H	FFH
INC A	00H	51H	00H	FFH
ST [R1]	00H	51H	00H	00H

ST [Rn, C] (STore accumulator to indirect (with C register) byte)

Instruction code	[1 0 0 1 0 0 1 0][0 n5n4n3n2n1n0 1] 92H
Byte count	2
Cycle count	2
Function	$((Rn) + (C)) \leftarrow (A)$, $(PC) \leftarrow (PC) + 2$
Affected flags	REG8 $\leftarrow$ P1

[Description]

Transfers the contents of the accumulator (A) to the data memory (RAM) or special function register designated by the result of arithmetic operation between the indirect address register Rn and C register.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory or special function register (SFR) having bit 8.

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data (-128 $\leqq$ C reg. $\leqq$ 127)

[Example]

R1 points to addresses 02H and 03H

	A	C	RAM 0002H	RAM 0003H	RAM 0056H
LDW #0051H	51H	--	--	--	--
STW R1	51H	--	51H	00H	--
MOV #05H, C	51H	05H	51H	00H	--
MOV #0FFH, A	FFH	05H	51H	00H	--
ST [R1,C]	FFH	05H	51H	00H	FFH
INC A	00H	05H	51H	00H	FFH
ST [R1,C]	00H	05H	51H	00H	00H

ST [off] (STore accumulator to indirect (with displacement) byte)

Instruction code	[1 0 0 1 0 0 1 0][1 off6off5off4off3off2off1off0] 92H
Byte count	2
Cycle count	2
Function	((R0) + off) ← (A), (PC) ← (PC) + 2
Affected flags	REG8 ← P1

[Description]

Transfers the contents of the accumulator (A) to the data memory (RAM) or special function register designated by the result of arithmetic operation between the indirect address register R0 and off.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory or special function register (SFR) having bit 8.。

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

	A	RAM 0000H	RAM 0001H	RAM 0056H	RAM 004CH
LDW #0051H	51H	--	--	--	--
STW R0	51H	51H	00H	--	--
ST [05H]	51H	51H	00H	51H	--
ST [-5]	51H	51H	00H	51H	51H

ST dst (STore accumulator to direct byte)

Instruction code	0000H≦dst＜0100H : [94H][dst_L(7-0)] (2bytes) 93H-96H			
	0100H≦dst＜0200H : [95H][dst_L(7-0)] (2bytes)			
	fe00H≦dst＜ff00H : [93H][dst_L(7-0)] (2bytes)			
	0200H≦dst＜fe00H : [96H][dst_L(7-0)][dst_H(7-0)] (3bytes)			
	ff00H≦dst≦ffffH : [96H][dst_L(7-0)][dst_H(7-0)] (3bytes)			
Byte count	2	3		
Cycle count	1	2		
Function	(dst) ← (A), (PC) ← (PC) + (2 or 3)			
Affected flags	REG8 ← P1			

[Description]
Transfers the contents of the accumulator (A) to the data memory (RAM) or special function register designated by dst.
The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory or special function register (SFR) having bit 8.
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0011H	RAM 0012H
MOV	#050H, A	50H	--	--
ST	11H	50H	50H	--
INC	A	51H	50H	--
ST	12H	51H	50H	51H

STL dst (STore accumulator to Long range direct byte)

Instruction code	0000H ≦ dst ≦ ffffH : [96H][dst_L(7-0)][dst_H(7-0)] (3bytes) 96H
Byte count	3
Cycle count	2
Function	(dst) ← (A), (PC) ← (PC) + 3
Affected flags	REG8 ← P1

[Description]
Transfers the contents of the accumulator (A) to the data memory (RAM) or special function register designated by dst.
The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the designated data memory or special function register (SFR) having bit 8.

[Example]

		A	RAM 0201H	RAM 0202H
MOV	#88H, A	88H	--	--
MOVL	#051H, 0201H	88H	51H	--
MOVL	#052H, 0202H	88H	51H	52H
STL	0201H	88H	88H	52H
INC	A	89H	88H	52H
STL	0202H	89H	88H	89H

STW [Rn] (STore register pair BA to indirect Word)

Instruction code	[0 0 0 1 0 1 1 1][0 n5n4n3n2n1n0 0] 17H
Byte count	2
Cycle count	2 or 3 (fe00H-ffffH)
Function	w((Rn)) ← (BA), (PC) ← (PC) + 2
Affected flags	REGH8 ← P1, REGL8 ← P1

[Description]

Transfers the contents of the accumulator (A) and B register to the data memory (RAM) or special function register designated by the indirect address register Rn.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the higher- and lower-order bytes of the designated data memory or special function register (SFR) having bit 8.

The valid value range of n is $0 \le n \le 63$.

[Example]

R1 points to addresses 02H and 03H

	PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0055H	RAM 0056H	A	B
LDW #0055H	01EFBH	475500H	--	--	--	--	55H	00H
STW R1	01EFEH	970200H	55H	00H	--	--	55H	00H
LDW #1234H	01F01H	473412H	55H	00H	--	--	34H	12H
STW [R1]	01F04H	1702H	55H	00H	34H	12H	34H	12H
LDW #5678H	01F06H	477856H	55H	00H	34H	12H	78H	56H
STW [R1]	01F09H	1702H	55H	00H	78H	56H	78H	56H
NOP	01F0BH	00H	55H	00H	78H	56H	78H	56H

STW [Rn,C] (STore register pair BA to indirect (with C) Word)

Instruction code	[0 0 0 1 0 1 1 1][0 n5n4n3n2n1n0 1]	17H
Byte count	2	
Cycle count	2 or 3 (fe00H-ffffH)	
Function	w((Rn) + (C)) ← (BA), (PC) ← (PC) + 2	
Affected flags	REGH8 ← P1, REGL8 ← P1	

[Description]

Transfers the contents of the accumulator (A) and B register to the data memory (RAM) or special function register designated by the result of arithmetic operation between the indirect address register Rn and the C register.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the higher- and lower-order bytes of the designated data memory or special function register (SFR) having bit 8.

The valid value range of n is $0 \leq n \leq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leq C$ reg.≤ 127)

[Example]

R1 points to addresses 02H and 03H

		PC	Instruction Code	RAM 0002H	RAM 0003H	RAM 0059H	RAM 005AH	A	B	C
MOV	#04H, C	01EF8H	430204H	--	--	--	--	--	--	04H
LDW	#0055H	01EFBH	475500H	--	--	--	--	55H	00H	04H
STW	R1	01EFEH	970200H	55H	00H	--	--	55H	00H	04H
LDW	#1234H	01F01H	473412H	55H	00H	--	--	34H	12H	04H
STW	[R1, C]	01F04H	1703H	55H	00H	34H	12H	34H	12H	04H
LDW	#5678H	01F06H	477856H	55H	00H	34H	12H	78H	56H	04H
STW	[R1, C]	01F09H	1703H	55H	00H	78H	56H	78H	56H	04H
NOP		01F0BH	00H	55H	00H	78H	56H	78H	56H	04H

STW [off] (STore register pair BA to indirect (with displacement) Word)

Instruction code	[0 0 0 1 0 1 1 1][1 off6off5off4off3off2off1off0]	17H
Byte count	2	
Cycle count	2 or 3 (fe00H-ffffH)	
Function	w((R0) + off) ← (BA), (PC) ← (PC) + 2	
Affected flags	REGH8 ← P1, REGL8 ← P1	

[Description]

Transfers the contents of the accumulator (A) and B register to the data memory (RAM) or special function register designated by the result of arithmetic operation between the indirect address register R0 and off.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the higher- and lower-order bytes of the designated data memory or special function register (SFR) having bit 8.

off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

		PC	Instruction Code	RAM 0000H	RAM 0001H	RAM 0059H	RAM 005AH	A	B
LDW	#0055H	01EFBH	475500H	--	--	--	--	55H	00H
STW	R0	01EFEH	970000H	55H	00H	--	--	55H	00H
LDW	#1234H	01F01H	473412H	55H	00H	--	--	34H	12H
STW	[04H]	01F04H	1784H	55H	00H	34H	12H	34H	12H
LDW	#5678H	01F06H	477856H	55H	00H	34H	12H	78H	56H
STW	[04H]	01F09H	1784H	55H	00H	78H	56H	78H	56H
NOP		01F0BH	00H	55H	00H	78H	56H	78H	56H

5-162

STW dst (STore register pair BA to long range direct Word)

Instruction code	[1 0 0 1 0 1 1 1][dst_L(7-0)][dst_H(7-0)] 97H
Byte count	3
Cycle count	2 or 3 (fe00H-ffffH)
Function	w(dst) ← (BA), (PC) ← (PC) + 3
Affected flags	REGH8 ← P1, REGL8 ← P1

[Description]

Transfers the contents of the accumulator (A) and B register to the data memory (RAM) or special function register designated by dst.

The instruction also transfers bit 1 of the program status word (PSW) to bit 8 of the higher- and lower-order bytes of the designated data memory or special function register (SFR) having bit 8.

[Example]

		PC	Instruction Code	RAM 00C0H	RAM 00C1H	RAM FE12H	RAM FE13H	A	B
LDW	#1234H	01EFBH	473412H	--	--	--	--	34H	12H
STW	00C0H	01EFEH	97C000H	34H	12H	--	--	34H	12H
LDW	#5678H	01F01H	477856H	34H	12H	--	--	78H	56H
STW	0FE12H	01F04H	9712FEH	34H	12H	78H	56H	78H	56H
STW	00C0H	01F07H	97C000H	78H	56H	78H	56H	78H	56H

5-163

STX [Rn] (STore accumulator to indirect eXternal memory byte)

Instruction code	[1 0 0 1 0 0 0 1][0 n5n4n3n2n1n0 0] 91H
Byte count	2
Cycle count	4
Function	ext ((Rn) + (B) ∗ 10000H) ← (A), (PC) ← (PC) + 2
Affected flags	

[Description]

Transfers the contents of the accumulator (A) to the external memory (EXT RAM) designated by the sum of the contents of the indirect address pair register (Rn) designated by n5-n0 and the contents of the B register (B) shifted 16 bits to the left. Subsequently, the CPU increments the program counter (PC) by 2.

The valid value range of n is $0 \leq n \leq 63$.

[Example 1]

		A	B	RAM 0005H	RAM 0004H	EXT RAM 200100H
MOV	#000H, 004H	--	--	--	00H	--
MOV	#001H, 005H	--	--	01H	00H	--
MOV	#020H, B	--	20H	01H	00H	--
MOV	#0FDH, A	FDH	20H	01H	00H	--
STX	[R2]	FDH	20H	01H	00H	FDH

[Example 2]

		A	B	RAM 0007H	RAM 0006H	EXT RAM 20017FH
MOV	#07FH, 006H	--	--	--	7FH	--
MOV	#001H, 007H	--	--	01H	7FH	--
MOV	#020H, B	--	20H	01H	7FH	--
MOV	#0FDH, A	FDH	20H	01H	7FH	--
STX	[R3]	FDH	20H	01H	7FH	FDH

STX [Rn,C] (STore accumulator to indirect (with C reg.) eXternal memory byte)

Instruction code	[1 0 0 1 0 0 0 1][0 n5n4n3n2n1n0 1]	91H
Byte count	2	
Cycle count	4	
Function	ext((Rn) + (C) + (B) ∗ 10000H) ← (A), (PC) ← (PC) + 2	
Affected flags		

[Description]

Transfers the contents of the accumulator (A) to the external memory (EXT RAM) designated by the sum of the contents of the indirect address pair register (Rn) designated by n5-n0, the contents of the C register, and the contents of the B register (B) shifted 16 bits to the left. Subsequently, the CPU increments the program counter (PC) by 2.

The valid value range of n is 0≦n≦63.

C register: 8-bit signed indirect displacement data (-128≦C reg.＜127)

[Example 1]

		A	B	C	RAM 0005H	RAM 0004H	EXT RAM 20013FH
MOV	#000H, 004H	--	--	--	--	00H	--
MOV	#001H, 005H	--	--	--	01H	00H	--
MOV	#020H, B	--	20H	--	01H	00H	--
MOV	#03FH, C	--	20H	3FH	01H	00H	--
MOV	#0FDH, A	FDH	20H	3FH	01H	00H	--
STX	[R2, C]	FDH	20H	3FH	01H	00H	FDH

[Example 2]

		A	B	C	RAM 0007H	RAM 0006H	EXT RAM 2001C0H
MOV	#000H, 006H	--	--	--	--	00H	--
MOV	#001H, 007H	--	--	--	01H	00H	--
MOV	#020H, B	--	20H	--	01H	00H	--
MOV	#-64, C	--	20H	C0H	01H	00H	--
MOV	#0FDH, A	FDH	20H	C0H	01H	00H	--
STX	[R3, C]	FDH	20H	C0H	01H	00H	FDH

STX [off] (STore accumulator to indirect (with displacement) eXternal memory byte)

Instruction code	[1 0 0 1 0 0 0 1][1 off7off6off5off4off3off2off1off0] 91H
Byte count	2
Cycle count	4
Function	ext((R0) + off + (B) * 10000H) ← (A), (PC) ← (PC) + 2
Affected flags	

[Description]
Transfers the contents of the accumulator (A) to the external memory (EXT RAM) designated by the sum of the contents of the indirect address pair register (R0), 7-bit signed indirect address offset data (off), and the contents of the B register (B) shifted 16 bits to the left. Subsequently, the CPU increments the program counter (PC) by 2.
off: 7-bit signed indirect address offset data ($-64 \leqq off \leqq 63$)

[Example 1]

		A	B	RAM 0001H	RAM 0000H	EXT RAM 20013FH
MOV	#000H, 000H	--	--	--	00H	--
MOV	#001H, 001H	--	--	01H	00H	--
MOV	#020H, B	--	20H	01H	00H	--
MOV	#0FDH, A	FDH	20H	01H	00H	--
STX	[3FH]	FDH	20H	01H	00H	FDH

[Example 2]

		A	B	RAM 0001H	RAM 0000H	EXT RAM 2000C0H
MOV	#000H, 000H	--	--	--	00H	--
MOV	#001H, 001H	--	--	01H	00H	--
MOV	#020H, B	--	20H	01H	00H	--
MOV	#0FDH, A	FDH	20H	01H	00H	--
STX	[-64]	FDH	20H	01H	00H	FDH

SUB #i (SUBtract immediate data to accumulator)

Instruction code	[1 1 0 0 0 0 0 1][i7i6i5i4i3i2i1i0] C1H
Byte count	2
Cycle count	1
Function	(A) ← (A) – i, (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Subtracts immediate data (i) from the contents of the accumulator (A) and places the result in the accumulator (A).

[Example]

		A	**CY**	**AC**	**OV**
MOV	#55H, A	55H	--	--	--
SUB	#13H	42H	0	0	0
SUB	#03H	3FH	0	1	0
SUB	#3FH	00H	0	0	0
SUB	#02H	FEH	1	1	0

SUB [Rn] (SUBtract indirect byte to accumulator)

Instruction code	[1 1 0 0 0 0 1 0][0 n5n4n3n2n1n0 0] C2H
Byte count	2
Cycle count	2
Function	(A) ← (A) – ((Rn)), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Subtracts the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn from the contents of the accumulator (A) and places the result in the accumulator (A).
The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

R1 points to addresses 02H and 03H

		RAM 0002H	RAM 0003H	A	RAM 0055H	CY	AC	OV
LDW	#0055H	--	--	55H	--	--	--	--
STW	R1	55H	00H	55H	--	--	--	--
ST	[R1]	55H	00H	55H	55H	--	--	--
SUB	[R1]	55H	00H	00H	55H	0	0	0
SUB	[R1]	55H	00H	ABH	55H	1	1	0
SUB	[R1]	55H	00H	56H	55H	0	0	1
SUB	[R1]	55H	00H	01H	55H	0	0	0

SUB [Rn,C] (SUBtract indirect byte (with C register) to accumulator)

Instruction code	[1 1 0 0 0 0 1 0][0 n5n4n3n2n1n0 1] C2H
Byte count	2
Cycle count	2
Function	(A) ← (A) – ((Rn) + (C)), (PC) ←(PC) + 2
Affected flags	CY,AC,OV

[Description]
Subtracts, from the contents of the accumulator (A), the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register Rn and the C register and places the result in the accumulator (A).
The valid value range of n is 0≦n≦63.
C register: 8-bit signed indirect displacement data (-128≦C reg.≦127)

[Example]

R1 points to addresses 02H and 03H	C	RAM 0002H	RAM 0003H	A	RAM 0059H	CY	AC	OV
LDW #0055H	--	--	--	55H	--	--	--	--
STW R1	--	55H	00H	55H	--	--	--	--
MOV #04H, C	04H	55H	00H	55H	--	--	--	--
ST [R1, C]	04H	55H	00H	55H	55H	--	--	--
SUB [R1, C]	04H	55H	00H	00H	55H	0	0	0
SUB [R1, C]	04H	55H	00H	ABH	55H	1	1	0
SUB [R1, C]	04H	55H	00H	56H	55H	0	0	1
SUB [R1, C]	04H	55H	00H	01H	55H	0	0	0

SUB [off] (SUBtract indirect byte(with displacement) to accumulator)

Instruction code	[1 1 0 0 0 0 1 0][1 off6off5off4off3off2off1off0] C2H
Byte count	2
Cycle count	2
Function	(A ← (A) − ((R0) + off), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Subtracts, from the contents of the accumulator (A), the contents of the data memory (RAM) or special function register (SFR) designated by the result of arithmetic operation between the indirect address register R0 and off and places the result in the accumulator (A).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

		RAM 0000H	RAM 0001H	A	RAM 0059H	CY	AC	OV
LDW	#0055H	--	--	55H	--	--	--	--
STW	R0	55H	00H	55H	--	--	--	--
ST	[04H]	55H	00H	55H	55H	--	--	--
SUB	[04H]	55H	00H	00H	55H	0	0	0
SUB	[04H]	55H	00H	ABH	55H	1	1	0
SUB	[04H]	55H	00H	56H	55H	0	0	1
SUB	[04H]	55H	00H	01H	55H	0	0	0

SUB dst (SUBtract direct byte to accumulator)

Instruction code	0000H≦dst＜0100H : [C4H][dst_L(7-0)] (2bytes) C3H-C6H
	0100H≦dst＜0200H : [C5H][dst_L(7-0)] (2bytes)
	fe00H≦dst＜ff00H : [C3H][dst_L(7-0)] (2bytes)
	0200H≦dst＜fe00H : [C6H][dst_L(7-0)][dst_H(7-0)] (3bytes)
	ff00H≦dst≦ffffH : [C6H][dst_L(7-0)][dst_H(7-0)] (3bytes)
Byte count	2 ┊ 3
Cycle count	1 ┊ 2
Function	(A) ← (A) – (dst), (PC) ← (PC) + (2 or 3)
Affected flags	CY, AC, OV

[Description]
Subtracts, from the contents of the accumulator (A), the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0034H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	34H	55H	55H	--	--	--
SUB	34H	00H	55H	0	0	0
SUB	34H	ABH	55H	1	1	0
SUB	34H	56H	55H	0	0	1
SUB	34H	01H	55H	0	0	0

SUBL dst (SUBtract Long range direct byte to accumulator)

Instruction code	0000H ≦ dst ＜ ffffH : [C6H][dst_L(7-0)][dst_H(7-0)] (3bytes) C6H
Byte count	3
Cycle count	2
Function	(A) ← (A) − (dst), (PC) ← (PC) + 3
Affected flags	CY, AC, OV

[Description]

Subtracts, from the contents of the accumulator (A), the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

[Example]

		A	RAM 0200H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	200H	55H	55H	--	--	--
SUBL	200H	00H	55H	0	0	0
SUBL	200H	ABH	55H	1	1	0
SUBL	200H	56H	55H	0	0	1
SUBL	200H	01H	55H	0	0	0

SUBC #i (SUBtract immediate data and Carry to accumulator)

Instruction code	[1 1 0 1 0 0 0 1][i7i6i5i4i3i2i1i0] D1H
Byte count	2
Cycle count	1
Function	(A) ← (A) – (CY) – i, (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Subtracts the carry flag (CY) and immediate data (i) from the contents of the accumulator (A) and places the result in the accumulator (A).

[Example]

		A	CY	AC	OV
MOV	#55H, A	55H	--	--	--
SUB	#13H	42H	0	0	0
SUBC	#0AH	38H	0	1	0
SUBC	#0FH	29H	0	1	0
SUBC	#80H	A9H	1	0	1
SUBC	#2AH	7EH	0	1	1

SUBC [Rn] (SUBtract indirect byte and Carry to accumulator)

Instruction code	[1 1 0 1 0 0 1 0][0 n5n4n3n2n1n0 0] D2H
Byte count	2
Cycle count	2
Function	(A) ← (A) – (CY) – ((Rn)), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Subtracts, from the contents of the accumulator (A), the carry flag (CY) and the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

[Example]

	RAM 0002H	RAM 0003H	A	RAM 0055H	CY	AC	OV
R1 points to addresses 02H and 03H							
LDW #0055H	--	--	55H	--	--	--	--
STW R1	55H	00H	55H	--	--	--	--
ST [R1]	55H	00H	55H	55H	--	--	--
SUB [R1]	55H	00H	00H	55H	0	0	0
SUBC [R1]	55H	00H	ABH	55H	1	1	0
SUBC [R1]	55H	00H	55H	55H	0	0	1
SUBC [R1]	55H	00H	00H	55H	0	0	0

5-174

SUBC [Rn,C] (SUBtract indirect byte (with C register) and Carry to accumulator)

Instruction code	[1 1 0 1 0 0 1 0][0 n5n4n3n2n1n0 1] D2H
Byte count	2
Cycle count	2
Function	(A) ← (A) – (CY) – ((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]

Subtracts, from the contents of the accumulator (A), the carry flag (CY) and the contents of the data memory (RAM) or special function register (SFR) designated by the arithmetic operation between the indirect address register Rn and C register and places the result in the accumulator (A).

The valid value range of n is $0 \leqq n \leqq 63$.

C register: 8-bit signed indirect displacement data ($-128 \leqq$ C reg. $\leqq 127$)

[Example]

R1 points to addresses 02H and 03H	C	RAM 0002H	RAM 0003H	A	RAM 0059H	CY	AC	OV
LDW #0055H	--	--	--	55H	--	--	--	--
STW R1	--	55H	00H	55H	--	--	--	--
MOV #04H, C	04H	55H	00H	55H	--	--	--	--
ST [R1, C]	04H	55H	00H	55H	55H	--	--	--
SUB [R1, C]	04H	55H	00H	00H	55H	0	0	0
SUBC [R1, C]	04H	55H	00H	ABH	55H	1	1	0
SUBC [R1, C]	04H	55H	00H	55H	55H	0	0	1
SUBC [R1, C]	04H	55H	00H	00H	55H	0	0	0

SUBC [off] (SUBtract indirect byte (with off) and Carry to accumulator)

Instruction code	[1 1 0 1 0 0 1 0][1 off6off5off4off3off2off1off0] D2H
Byte count	2
Cycle count	2
Function	(A) ← (A) – (CY) – ((R0) + off), (PC) ← (PC) + 2
Affected flags	CY, AC, OV

[Description]
Subtracts, from the contents of the accumulator (A), the carry flag (CY) and the contents of the data memory (RAM) or special function register (SFR) designated by the arithmetic operation between the indirect address register R0 and off and places the result in the accumulator (A).
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

		RAM 0000H	RAM 0001H	A	RAM 0059H	CY	AC	OV
R0 points to addresses 00H and 01H								
LDW	#0055H	--	--	55H	--	--	--	--
STW	R0	55H	00H	55H	--	--	--	--
ST	[04H]	55H	00H	55H	55H	--	--	--
SUB	[04H]	55H	00H	00H	55H	0	0	0
SUBC	[04H]	55H	00H	ABH	55H	1	1	0
SUBC	[04H]	55H	00H	55H	55H	0	0	1
SUBC	[04H]	55H	00H	00H	55H	0	0	0

5-176

SUBC dst (SUBtract direct byte and Carry to accumulator)

Instruction code	0000H≦dst＜0100H	: [D4H][dst_L(7-0)]	(2bytes)	D3H-D6H
	0100H≦dst＜0200H	: [D5H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [D3H][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [D6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [D6H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2 ┊ 3			
Cycle count	1 ┊ 2			
Function	(A) ← (A) – (CY) – (dst), (PC) ← (PC) + (2 or 3)			
Affected flags	CY, AC, OV			

[Description]
Subtracts, from the contents of the accumulator (A), the carry flag (CY) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).
The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0034H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	34H	55H	55H	--	--	--
SUB	34H	00H	55H	0	0	0
SUBC	34H	ABH	55H	1	1	0
SUBC	34H	55H	55H	0	0	1
SUBC	34H	00H	55H	0	0	0

SUBCL dst (SUBtract Long range direct byte and Carry to accumulator)

Instruction code	0000H≦dst＜ffffH : [D6H][dst_L(7-0)][dst_H(7-0)] (3bytes) D6H
Byte count	3
Cycle count	2
Function	(A) ← (A) – (CY) – (dst), (PC) ← (PC) + 3
Affected flags	CY, AC, OV

[Description]

Subtracts, from the contents of the accumulator (A), the carry flag (CY) and the contents of the data memory (RAM) or special function register (SFR) designated by dst and places the result in the accumulator (A).

[Example]

		A	RAM 0200H	CY	AC	OV
MOV	#55H, A	55H	--	--	--	--
ST	200H	55H	55H	--	--	--
SUBL	200H	00H	55H	0	0	0
SUBCL	200H	ABH	55H	1	1	0
SUBCL	200H	55H	55H	0	0	1
SUBCL	200H	00H	55H	0	0	0

XCH [Rn] (eXCHange indirect byte with accumulator)

Instruction code	[0 1 0 1 0 0 1 0][0 n5n4n3n2n1n0 0]	52H
Byte count	2	
Cycle count	2	
Function	(A) ←→ ((Rn)), (PC) ← (PC) + 2	
Affected flags	P1 ←→ REG8	

[Description]
Exchanges the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by the indirect address register Rn.
The instruction also exchanges bit of the program status word with bit 8 of the designated data memory (RAM) or special function register having bit 8.
The valid value range of n is $0 \leq n \leq 63$.

[Example]

R1 points to addresses 02H and 03H

	A	RAM 0002H	RAM 0003H	RAM 0051H
LDW #0051H	51H	--	--	--
STW R1	51H	51H	00H	--
MOV #0FFH, A	FFH	51H	00H	--
MOV #00H, [R1]	FFH	51H	00H	00H
XCH [R1]	00H	51H	00H	FFH
XCH [R1]	FFH	51H	00H	00H

XCH [Rn, C] (eXCHange indirect byte (with C register) with accumulator)

Instruction code	[0 1 0 1 0 0 1 0][0 n5n4n3n2n1n0 1] 52H
Byte count	2
Cycle count	2
Function	(A) ←→ ((Rn) + (C)), (PC) ← (PC) + 2
Affected flags	P1 ←→ REG8

[Description]

Exchanges the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by the arithmetic operation between the indirect address register Rn and the C register.
The instruction also exchanges bit of the program status word with bit 8 of the designated data memory (RAM) or special function register having bit 8.
The valid value range of n is $0 \leqq n \leqq 63$.
C register: 8-bit signed indirect displacement data $(-128 \leqq C\ reg. \leqq 127)$

[Example]

R1 points to addresses 02H and 03H

	A	C	RAM 0002H	RAM 0003H	RAM 0056H
LDW #0051H	51H	--	--	--	--
STW R1	51H	--	51H	00H	--
MOV #05H, C	51H	05H	51H	00H	--
MOV #0FFH, A	FFH	05H	51H	00H	--
MOV #00H, [R1,C]	FFH	05H	51H	00H	00H
XCH [R1,C]	00H	05H	51H	00H	FFH
XCH [R1,C]	FFH	05H	51H	00H	00H

XCH [off] (eXCHange indirect byte (with displacement) with accumulator)

Instruction code	[0 1 0 1 0 0 1 0][1 off6off5off4off3off2off1off0] 52H
Byte count	2
Cycle count	2
Function	(A) $\leftrightarrow$ ((R0) + off), (PC) $\leftarrow$ (PC) + 2
Affected flags	P1 $\leftrightarrow$ REG8

[Description]
Exchanges the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by the arithmetic operation between the indirect address register R0 and off.
The instruction also exchanges bit of the program status word with bit 8 of the designated data memory (RAM) or special function register having bit 8.
off: 7-bit signed indirect address offset data (-64≦off≦63)

[Example]

R0 points to addresses 00H and 01H

		A	**RAM 0000H**	**RAM 0001H**	**RAM 0056H**	**RAM 004CH**
LDW	#0051H	51H	--	--	--	--
STW	R0	51H	51H	00H	--	--
MOV	#0FFH, A	FFH	51H	00H	--	--
MOV	#00H, 56H	FFH	51H	00H	00H	--
MOV	#88H, 4CH	FFH	51H	00H	00H	88H
XCH	[05H]	00H	51H	00H	FFH	88H
XCH	[-5]	88H	51H	00H	FFH	00H

XCH dst (eXCHange direct byte with accumulator)

Instruction code	0000H≦dst＜0100H	: [54H][dst_L(7-0)]	(2bytes)	53H-56H
	0100H≦dst＜0200H	: [55H][dst_L(7-0)]	(2bytes)	
	fe00H≦dst＜ff00H	: [53H][dst_L(7-0)]	(2bytes)	
	0200H≦dst＜fe00H	: [56H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
	ff00H≦dst≦ffffH	: [56H][dst_L(7-0)][dst_H(7-0)]	(3bytes)	
Byte count	2 ┊ 3			
Cycle count	1 ┊ 2			
Function	(A) ←→ (dst), (PC) ← (PC) + (2 or 3)			
Affected flags	P1 ←→ REG8			

[Description]

Exchanges the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by dst.

The instruction also exchanges bit of the program status word with bit 8 of the designated data memory (RAM) or special function register having bit 8.

The byte count and cycle count vary according to the address value of the operand (dst).

[Example]

		A	RAM 0011H	RAM 0012H
MOV	#050H, A	50H	--	--
MOV	#051H, 11H	50H	51H	--
MOV	#052H, 12H	50H	51H	52H
XCH	11H	51H	50H	52H
XCH	12H	52H	50H	51H

XCHL dst (eXCHange Long range direct byte with accumulator)

Instruction code	0000H≦dst≦ffffH : [56H][dst_L(7-0)][dst_H(7-0)]	56H
Byte count	3	
Cycle count	2	
Function	(A) ←→ (dst), (PC) ← (PC) + 3	
Affected flags	P1 ←→ REG8	

[Description]

Exchanges the contents of the accumulator (A) with the contents of the data memory (RAM) or special function register (SFR) designated by dst.

The instruction also exchanges bit of the program status word with bit 8 of the designated data memory (RAM) or special function register having bit 8.

[Example]

	A	RAM 0201H	RAM 0202H
MOV #050H, A	50H	--	--
MOVL #051H, 0201H	50H	51H	--
MOVL #052H, 0202H	50H	51H	52H
XCHL 0201H	51H	50H	52H
XCHL 0202H	52H	50H	51H

XCHW dst (eXCHange long range direct word with register pair BA)

Instruction code	[0 1 0 1 0 1 1 1][dst_L(7-0)][dst_H(7-0)] 57H
Byte count	3
Cycle count	3
Function	(BA) ←→ w(dst), (PC) ← (PC) + 3
Affected flags	REGH8 ← P1, REGL8 ←P1, P1 ← REGH8

[Description]
Exchanges the contents of the accumulator (A) and B register with the contents of the data memory (RAM) or special function register (SFR) designated by dst.
The instruction also transfers bit 1 of the program status word to bit 8 of the higher- and lower-order bytes of the designated data memory (RAM) or special function register having bit 8. It also transfers bit 8 of the higher-order byte of the designated word data memory (RAM) or special function register (SFR) having bit 8 to bit 1 of the program status word (PSW).

[Example]

		PC	Instruction Code	RAM 00C0H	RAM 00C1H	RAM FE12H	RAM FE13H	A	B
LDW	#1234H	01EFBH	473412H	--	--	--	--	34H	12H
STW	00C0H	01EFEH	97C000H	34H	12H	--	--	34H	12H
LDW	#5678H	01F01H	477856H	34H	12H	--	--	78H	56H
XCHW	00C0H	01F04H	57C000H	78H	56H	--	--	34H	12H
XCHW	0FE12H	01F07H	5712FEH	78H	56H	34H	12H	--	--

5-184

5.3　Instruction Set Chart

Mnemonic	Operation	Addressing							Flag			Page
		#	Rn	Rn,C	off	Dst	mid	1ng	CY	AC	OV	
ADD	(A)<-(A)+(operand)	○	○	○	○	○		○	○	○	○	3-8
ADDC	(A)<-(A)+(operand)+CY	○	○	○	○	○		○	○	○	○	9-14
AND	(A)<-(A) AND (operand)	○	○	○	○	○		○				15-20
BE	Branch if equal	○	○	○	○	○		○	○			21-26
BN	Branch if operand and bit are 0					○	○					27-28
BNE	Branch if not equal	○	○	○	○	○		○	○			29-34
BP	Branch if operand and bit are 1					○	○					35-36
BPC	Branch and reset bit					○						37
BR	Unconditional relative blanch (12-bit address)											38
BZ	Branch if (A)=0											39
BZW	Branch if (BA)=0											40
BNZ	Branch if (A)≠0											41
BNZW	Branch if (BA)≠0											42
CALL	Call absolute address within 128K bytes (bank)											43
CHGP1,3	Change PSW bit 1 or 3											44-45
CLR1	Clear 1 bit designated by (operand) and (bit)					○	○					46-47
DBNZ	Branch if ((operand)-1)≠0		○	○	○	○		○				48-52
DBZ	Branch if ((operand)-1)=0		○	○	○	○		○				53-57
DEC	(operand)<-(operand)-1		○	○	○	○		○				58-62
DECW	word(operand)<-word (operand)-1					○						63
DIV16／24	16÷8, 24÷16										○	64-65
FUNC(8/16)	ADD, ADDC, SUB, SUBC, NOR, AND, OR, XOR	Affected only during ADD(C) and SUB(C) →							○	○	○	66-81
INC	(operand)<-(operand)+1		○	○	○	○		○				82-86
INCW	word(operand)<-word (operand)+1					○						87
JMP	Unconditional absolute jump within 128K bytes (bank)											88
LD	(A)<-(operand (8 bits))	○	○	○	○	○		○				89-94
LDCW	(BA)<-ROMw (operand) Read ROM table		○	○	○							95-97
LDW	(BA)<-(operand (16 bits))	○	○	○	○	○						98-102
LDX	(A)<-external RAM (operand)		○	○	○							103-105
MOV	(operand)<-8-bit immediate data		○	○	○	○		○				106-110
MUL16／24	16×8, 24×16										○	111-112
NOP	Consume 1 machine cycle											113
NOT1	Invert 1 bit designated by (operand) and (bit)					○	○					114-115
OR	(A)<-(A) OR (operand)	○	○	○	○	○		○				116-121
POP	Restore data from stack into operand		○	○	○	○		○				122-126
POP__BA	Restore data from stack into (BA)											127
POP__P	Restore data from stack into (PSW)											128
POPW	Restore data from stack into word operand		○	○	○	○						129-132

Mnemonic	Operation	Addressing							Flag			Page
		#	Rn	Rn,C	off	dst	mid	1ng	CY	AC	OV	
PUSH	Save (operand) into stack	○	○	○	○	○		○				133-138
PUSH__BA	Save (BA) into stack											139
PUSH__P	Save (PSW) into stack											140
PUSHW	Save word (operand) into stack		○	○	○	○						141-144
RCALL	Relative call (12-bit address)											145
RCALLA	Relative call using (A)											146
RET,RETI	Return from call or interrupt											147-148
ROL	Rotate (A) 1 bit to left											149
ROLC	Rotate (A) and CY 1 bit to left								○			150
ROR	Rotate (A) 1 bit to right											151
RORC	Rotate (A) and CY 1 bit to right								○			152
SET1	Rotate 1 bit designated by operand and bit					○	○					153-154
ST	(operand (8 bits))<-(A)		○	○	○	○		○				155-159
STW	(operand (16 bits))<-(BA)		○	○	○	○						160-163
STX	external RAM (operand)<-(A)		○	○	○							164-166
SUB	(A)<-(A)-(operand)	○	○	○	○	○		○	○	○	○	167-172
SUBC	(A)<-(A)-(operand)-CY	○	○	○	○	○		○	○	○	○	173-178
XCH	(A)<->(operand (8 bits))		○	○	○	○		○				179-183
XCHW	(BA)<->(operand (16 bits))					○						184

Appendixes

Table of Contents

Appendix I

- Special Functions Register (SFR) Map

Appendix-II

- Port 0 Block Diagram
- Port 1 Block Diagram
- Port 2 Block Diagram
- Port 3 Block Diagram
- Port 7 Block Diagram

Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
0〜00FF	XXXX XXXX	R/W	RAM256B	9 bits long									
FE00	0000 0000	R/W	AREG		–	AREG7	AREG6	AREG5	AREG4	AREG3	AREG2	AREG1	AREG0
FE01	0000 0000	R/W	BREG		–	BREG7	BREG6	BREG5	BREG4	BREG3	BREG2	BREG1	BREG0
FE02	0000 0000	R/W	CREG		–	CREG7	CREG6	CREG5	CREG4	CREG3	CREG2	CREG1	CREG0
FE03													
FE04													
FE05													
FE06	0000 0000	R/W	PSW		–	CY	AC	PSWB5	PSWB4	LDCBNK	0V	P1	PARITY
FE07	HHHH H000	R/W	PCON		–	–	–	–	–	–	XTIDLE	PDN	IDLE
FE08	0000 HH00	R/W	IE		–	IE7	XFLG	HFLG	LFLG	–	–	XCNT1	XCNT0
FE09	0000 0000	R/W	IP		–	IP4B	IP43	IP3B	IP33	IP2B	IP23	IP1B	IP13
FE0A	0000 0000	R/W	SPL		–	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
FE0B	0000 0000	R/W	SPH		–	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP
FE0C	HHHH H000	R/W	CLKDIV		–	–	–	–	–	–	CLKDV2	CLKDV1	CLKDV0
FE0D	00HX XXXX	R/W	MRCR		–	MRCSEL	MRCST	–	RCCTD4	RCCTD3	RCCTD2	RCCTD1	RCCTD0
FE0E	0000 XX00	R/W	OCR	XT1 and XT2 read at bits 2 and 3	–	CLKSGL	EXTOSC	CLKCB5	CLKCB4	XT2IN	XT1IN	RCSTOP	CFSTOP
FE0F	0H00 H000	R/W	WDT		–	WDTFLG	–	WDTB5	WDTHLT	–	WDTCLR	WDTRST	WDTRUN
FE10	0000 0000	R/W	T0CNT		–	T0HRUN	T0LRUN	T0LONG	T0LEXT	T0HCMP	T0HIE	T0LCMP	T0LIE
FE11	0000 0000	R/W	T0PRR	Prescaler is 8 bits long. (max. 256 Tcyc).	–	T0PRR7	T0PRR6	T0PRR5	T0PRR4	T0PRR3	T0PRR2	T0PRR1	T0PRR0
FE12	0000 0000	R	T0L		–	T0L7	T0L6	T0L5	T0L4	T0L3	T0L2	T0L1	T0L0
FE13	0000 0000	R	T0H		–	T0H7	T0H6	T0H5	T0H4	T0H3	T0H2	T0H1	T0H0
FE14	0000 0000	R/W	T0LR		–	T0LR7	T0LR6	T0LR5	T0LR4	T0LR3	T0LR2	T0LR1	T0LR0
FE15	0000 0000	R/W	T0HR		–	T0HR7	T0HR6	T0HR5	T0HR4	T0HR3	T0HR2	T0HR1	T0HR0
FE16	XXXX XXXX	R	T0CAL	Timer 0 capture register L	–	T0CAL7	T0CAL6	T0CAL5	T0CAL4	T0CAL3	T0CAL2	T0CAL1	T0CAL0
FE17	XXXX XXXX	R	T0CAH	Timer 0 capture register H	–	T0CAH7	T0CAH6	T0CAH5	T0CAH4	T0CAH3	T0CAH2	T0CAH1	T0CAH0
FE18	0000 0000	R/W	T1CNT		–	T1HRUN	T1LRUN	T1LONG	T1PWM	T1HCMP	T1HIE	T1LCMP	T1LIE
FE19	0000 0000	R/W	T1PRR		–	T1HPRE	T1HPRC2	T1HPRC1	T1HPRC0	T1LPRE	T1LPRC2	T1LPRC1	T1LPRC0
FE1A	0000 0000	R	T1L		–	T1L7	T1L6	T1L5	T1L4	T1L3	T1L2	T1L1	T1L0
FE1B	0000 0000	R	T1H		–	T1H7	T1H6	T1H5	T1H4	T1H3	T1H2	T1H1	T1H0
FE1C	0000 0000	R/W	T1LR		–	T1LR7	T1LR6	T1LR5	T1LR4	T1LR3	T1LR2	T1LR1	T1LR0
FE1D	0000 0000	R/W	T1HR		–	T1HR7	T1HR6	T1HR5	T1HR4	T1HR3	T1HR2	T1HR1	T1HR0

Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE1E													
FE1F													
FE20													
FE21													
FE22													
FE23													
FE24													
FE25													
FE26													
FE27													
FE28													
FE29													
FE2A													
FE2B													
FE2C													
FE2D													
FE2E													
FE2F													
FE30	0000 0000	R/W	SCON0		−	FIX0	FIX0	SI0RUN	FIX0	SI0DIR	SI00VR	SI0END	SI0IE
FE31	0000 0000	R/W	SBUF0		−	SBUF07	SBUF06	SBUF05	SBUF04	SBUF03	SBUF02	SBUF01	SBUF00
FE32	0000 0000	R/W	SBR0		−	SBRG07	SBRG06	SBRG05	SBRG04	SBRG03	SBRG02	SBRG01	SBRG00
FE33	0000 0000	R/W	SCTR0		−	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0
FE34	0000 0000	R/W	SCON1		−	SI1M1	SI1M0	SI1RUN	SI1REC	SI1DIR	SI10VR	SI1END	SI1IE
FE35	00000 0000	R/W	SBUF1	9−bit REG	SBUF18	SBUF17	SBUF16	SBUF15	SBUF14	SBUF13	SBUF12	SBUF11	SBUF10
FE36	0000 0000	R/W	SBR1		−	SBRG17	SBRG16	SBRG15	SBRG14	SBRG13	SBRG12	SBRG11	SBRG10
FE37	0000 0000	R/W	SWCON0	Controls suspension of continuous SI00 transmission.	−	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0	FIX0
FE38													
FE39													
FE3A													
FE3B													
FE3C													
FE3D													

Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE3E													
FE3F													
FE40	0000 0000	R/W	P0		–	P07	P06	P05	P04	P03	P02	P01	P00
FE41	0000 0000	R/W	P0DDR		–	P0HPUS	P0LPUS	P0FLG	P0IE	P0HPU	P0LPU	P0HDDR	P0LDDR
FE42	00HH 0000	R/W	P0FCR		–	T70E	T60E	–	–	CLK0EN	CLK0DV2	CLK0DV1	CLK0DV0
FE43	0000 0000	R/W	XT2PC	Oscillator pin/general-purpose port input control	–	XT2PCB7	XT2PCB6	XT2PCB5	XT2PCB4	XTCFIN	XT2PCB2	XT2PCB1	XT2PCB0
FE44	0000 0000	R/W	P1		–	P17	P16	P15	P14	P13	P12	P11	P10
FE45	0000 0000	R/W	P1DDR		–	P17DDR	P16DDR	P15DDR	P14DDR	P13DDR	P12DDR	P11DDR	P10DDR
FE46	0000 0000	R/W	P1FCR		–	P17FCR	P16FCR	P15FCR	P14FCR	P13FCR	P12FCR	P11FCR	P10FCR
FE47	0000 H0H0	R/W	P1TST		–	FIX0	FIX0	MRCSHIFT	FIX0	–	DSNK0T	–	FIX0
FE48	HHHH HH00	R/W	P2		–	–	–	–	–	–	–	P21	P20
FE49	HHHH HH00	R/W	P2DDR		–	–	–	–	–	–	–	P21DDR	P20DDR
FE4A	0000 0000	R/W	I45CR		–	INT5HEG	INT5LEG	INT5IF	INT5IE	INT4HEG	INT4LEG	INT4IF	INT4IE
FE4B	0000 0000	R/W	I45SL		–	I5SL3	I5SL2	I5SL1	I5SL0	I4SL3	I4SL2	I4SL1	I4SL0
FE4C	HHHH HH00	R/W	P3		–	–	–	–	–	–	–	P31	P30
FE4D	HHHH HH00	R/W	P3DDR		–	–	–	–	–	–	–	P31DDR	P30DDR
FE4E													
FE4F													
FE50													
FE51													
FE52													
FE53													
FE54													
FE55													
FE56													
FE57	HHH0 HH00	R/W	CFLVM		–	–	–	–	CFMON	–	–	FIX0	FIX0
FE58	0000 0000	R/W	ADCRC	12-bit AD control	–	ADCHSEL3	ADCHSEL2	ADCHSEL1	ADCHSEL0	ADCR3	ADSTART	ADENDF	ADIE
FE59	0000 0000	R/W	ADMRC	12-bit AD mode ˚	–	ADMD4	ADMD3	ADMD2	ADMD1	ADMD0	ADMR2	ADTM1	ADTM0
FE5A	0000 0000	R/W	ADRLC	12-bit AD conversion results L	–	DATAL3	DATAL2	DATAL1	DATAL0	ADRL3	ADRL2	ADRL1	ADTM2
FE5B	0000 0000	R/W	ADRHC	12-bit AD conversion results H	–	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
FE5C	0000 0000	R/W	P7	4-bit IO (7-4: DDR 3:0: DATA)	–	P73DDR	P72DDR	P71DDR	P70DDR	P73DT	P72DT	P71DT	P70DT
FE5D	0000 0000	R/W	I01CR		–	INT1LH	INT1LV	INT1IF	INT1IE	INT0LH	INT0LV	INT0IF	INT0IE

A1-3

Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE5E	0000 0000	R/W	I23CR		–	INT3HEG	INT3LEG	INT3IF	INT3IE	INT2HEG	INT2LEG	INT2IF	INT2IE
FE5F	0000 0000	R/W	ISL	Bits 2, 6, and 7 added	–	STOHCP	STOLCP	BTIMC1	BTIMC0	BUZON	NFSEL	NFON	STOIN
FE60													
FE61													
FE62													
FE63													
FE64													
FE65													
FE66													
FE67													
FE68													
FE69													
FE6A													
FE6B													
FE6C													
FE6D													
FE6E													
FE6F													
FE70													
FE71													
FE72	0000 HHHH	R/W	PWM4L	PWM4 compare L (additional)	–	PWM4L3	PWM4L2	PWM4L1	PWM4L0	–	–	–	–
FE73	0000 0000	R/W	PWM4H	PWM4 compare H (base)	–	PWM4H7	PWM4H6	PWM4H5	PWM4H4	PWM4H3	PWM4H2	PWM4H1	PWM4H0
FE74	0000 HHHH	R/W	PWM5L	PWM5 compare L(additional)	–	PWM5L3	PWM5L2	PWM5L1	PWM5L0	–	–	–	–
FE75	0000 0000	R/W	PWM5H	PWM5 compare H (base)	–	PWM5H7	PWM5H6	PWM5H5	PWM5H4	PWM5H3	PWM5H2	PWM5H1	PWM5H0
FE76	0000 0000	R/W	PWM4C	PWM4/PWM5 control	–	PWM4C7	PWM4C6	PWM4C5	PWM4C4	ENPWM5	ENPWM4	PWM4OV	PWM4IE
FE77													
FE78	0000 0000	R/W	T67CNT		–	T7C1	T7C0	T6C1	T6C0	T7OV	T7IE	T6OV	T6IE
FE79													
FE7A	0000 0000	R/W	T6R		–	T6R7	T6R6	T6R5	T6R4	T6R3	T6R2	T6R1	T6R0
FE7B	0000 0000	R/W	T7R		–	T7R7	T7R6	T7R5	T7R4	T7R3	T7R2	T7R1	T7R0
FE7C	HHHH H000	R/W	SLWRC		–	–	–	–	–	–	CFLAMP	SLRCSEL	SLRCSTAT

Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE7D	0000 0000	R/W	NKREG		−	NKEN	NKCMP2	NKCMP1	NKCMP0	NKCOV	NKCAP2	NKCAP1	NKCAP0
FE7E	0000 0000	R/W	FSR0	FLASH control (bit 4 is R/O.)	−	FSR0B7 Fix to 0	FSR0B6 Fix to 0	FSAERR	FSWOK	INTHIGH	FSLDAT	FSPGL	FSWREQ
FE7F	0000 0000	R/W	BTCR	Base timer control	−	BTFST	BTON	BTC11	BTC10	BTIF1	BTIE1	BTIF0	BTIE0
FE80													
FE81													
FE82													
FE83													
FE84													
FE85													
FE86													
FE87													
FE88													
FE89													
FE8A													
FE8B													
FE8C													
FE8D													
FE8E													
FE8F													
FE90													
FE91													
FE92													
FE93													
FE94													
FE95													
FE96													
FE97													
FE98													
FE99													
FE9A													
FE9B													

A1-5

Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FE9C													
FE9D													
FE9E													
FE9F													
FEA0													
FEA1													
FEA2													
FEA3													
FEA4													
FEA5													
FEA6													
FEA7													
FEA8													
FEA9													
FEAA													
FEAB													
FEAC													
FEAD													
FEAE													
FEAF													
FEB0													
FEB1													
FEB2													
FEB3													
FEB4													
FEB5													
FEB6													
FEB7													
FEB8													
FEB9													
FEBA													
FEBB													

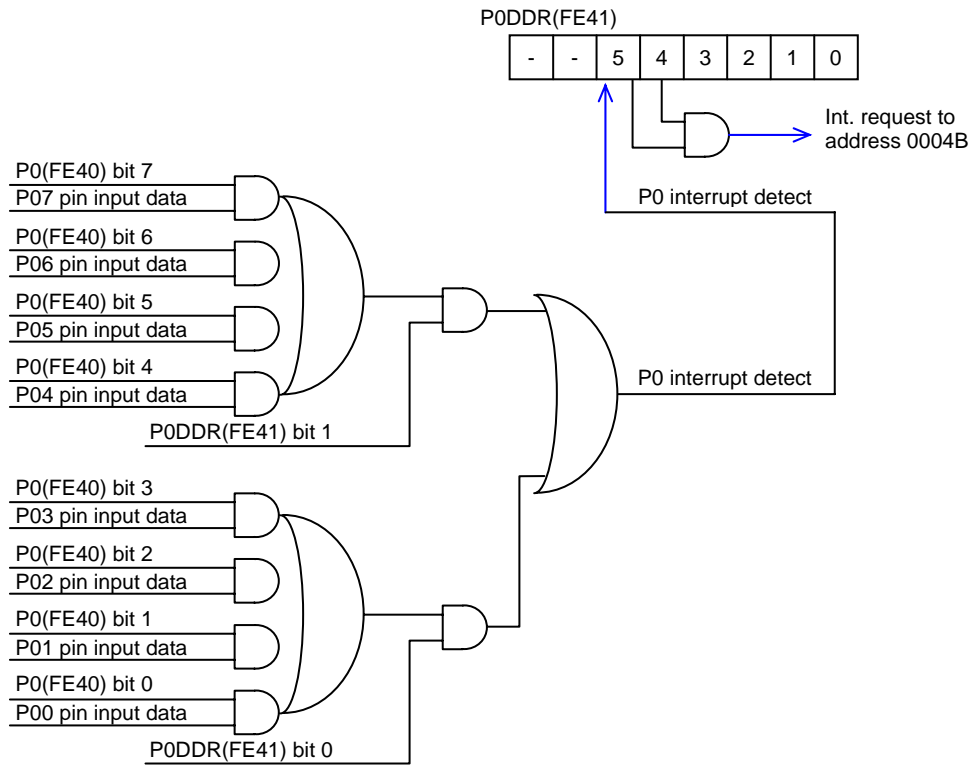| Address	Initial value	R/W	LC872H00	Remarks	BIT8	BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0
FEBC													
FEBD													
FEBE													
FEBF													
FEC0													
FEC1													
FEC2													
FEC3													
FEC4													
FEC5													
FEC6													
FEC7													
FEC8													
FEC9													
FECA													
FECB													
FECC													
FECD													
FECE													
FECF													
FED0	0000 0000	R/W	UCON0		–	UBRSEL	STRDET	RECRUN	STPERR	UOB3	RBIT8	RECEND	RECIE
FED1	0000 0000	R/W	UCON1		–	TRUN	8/9BIT	TDDR	TCMOS	7/8BIT	TBIT8	TEPTY	TRNSIE
FED2	0000 0000	R/W	UBR		–	UBRG7	UBRG6	UBRG5	UBRG4	UBRG3	UBRG2	UBRG1	UBRG0
FED3	0000 0000	R/W	TBUF		–	T1BUF7	T1BUF6	T1BUF5	T1BUF4	T1BUF3	T1BUF2	T1BUF1	T1BUF0
FED4	0000 0000	R/W	RBUF		–	R1BUF7	R1BUF6	R1BUF5	R1BUF4	R1BUF3	R1BUF2	R1BUF1	R1BUF0
FED5													
FED6													
FED7													
FED8													
FED9													
FEDA													
FEDB													
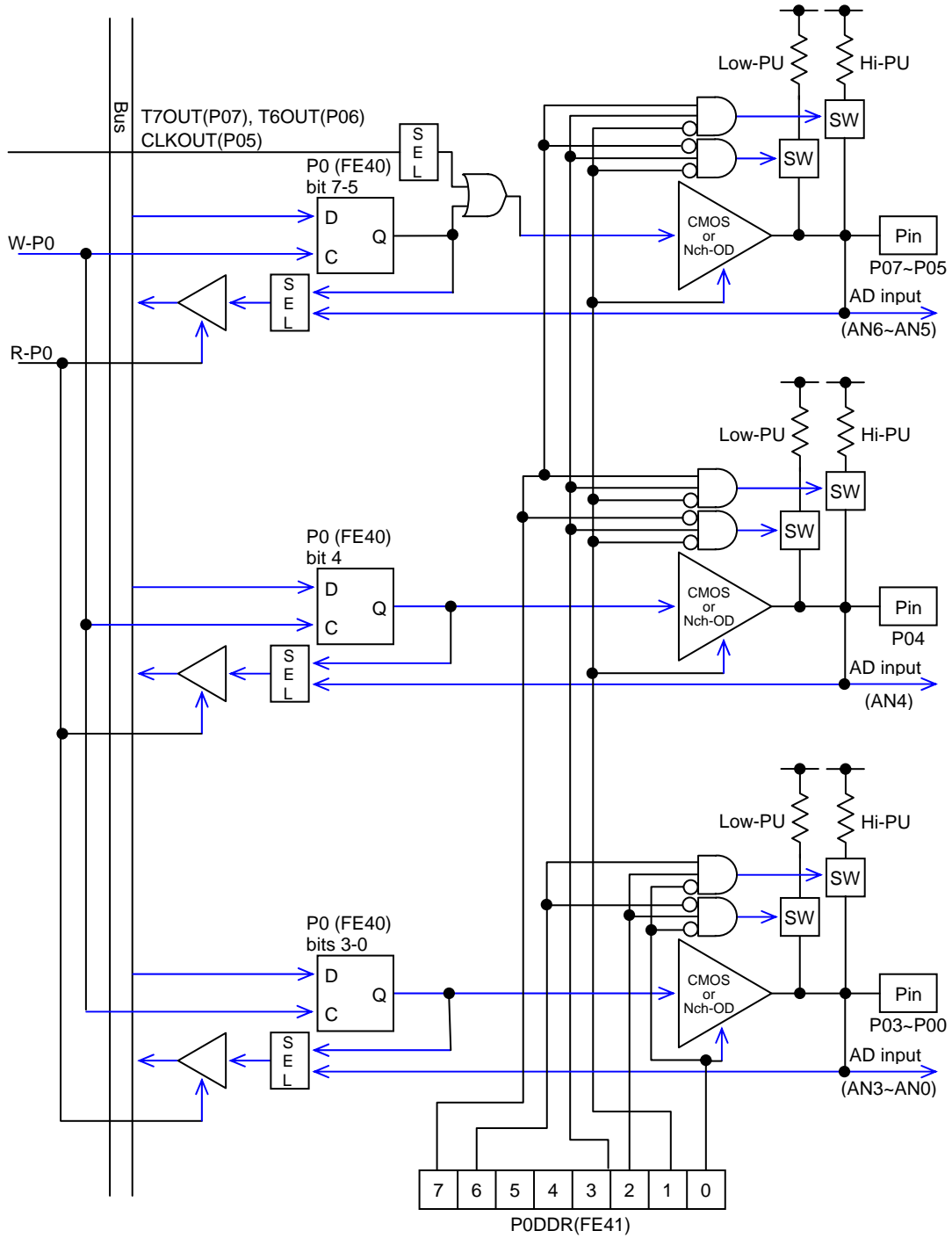
Pull-up resistor is:

Not attached if N-channel-OD option is selected.
Programmable if CMOS option is selected.

Port	Shared port pin function
P07	Timer 7 toggle output
P06	Timer 6 toggle output
P05	Clock output (system/subclock selectable)

Port 0 Block Diagram

Option: Output type (CMOS or N-channel OD) selectable on a bit basis.

Port Block Diagrams

P0DDR(FE41)

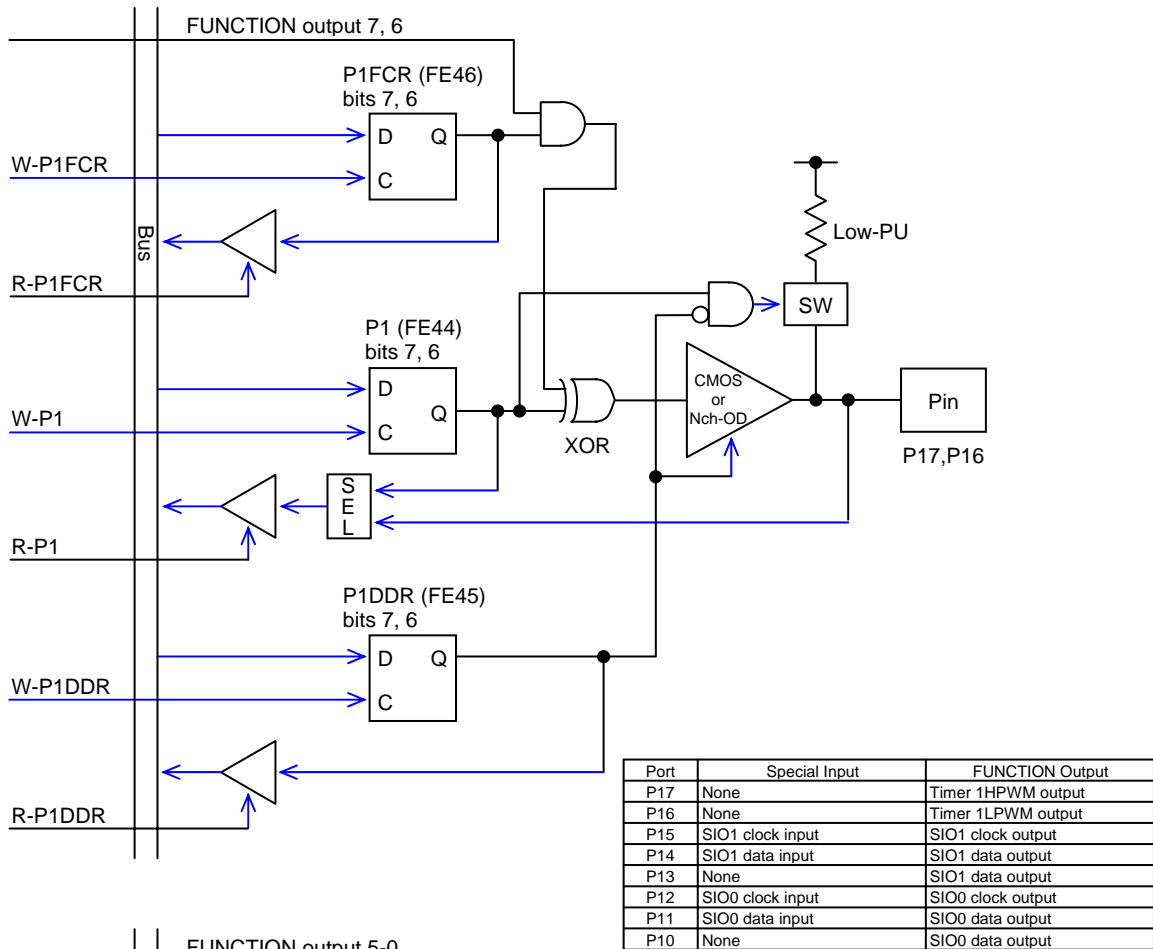| - | - | 5 | 4 | 3 | 2 | 1 | 0 |

Int. request to
address 0004B

P0 interrupt detect

P0(FE40) bit 7
P07 pin input data

P0(FE40) bit 6
P06 pin input data

P0(FE40) bit 5
P05 pin input data

P0(FE40) bit 4
P04 pin input data

P0DDR(FE41) bit 1

P0 interrupt detect

P0(FE40) bit 3
P03 pin input data

P0(FE40) bit 2
P02 pin input data

P0(FE40) bit 1
P01 pin input data

P0(FE40) bit 0
P00 pin input data

P0DDR(FE41) bit 0

Port 0（Interrupt）Block Diagram

FUNCTION output 7, 6

P1FCR (FE46)
bits 7, 6

W-P1FCR

R-P1FCR

Bus

Low-PU

SW

P1 (FE44)
bits 7, 6

W-P1

XOR

CMOS
or
Nch-OD

Pin

P17,P16

SEL

R-P1

P1DDR (FE45)
bits 7, 6

W-P1DDR

R-P1DDR

Port	Special Input	FUNCTION Output
P17	None	Timer 1HPWM output
P16	None	Timer 1LPWM output
P15	SIO1 clock input	SIO1 clock output
P14	SIO1 data input	SIO1 data output
P13	None	SIO1 data output
P12	SIO0 clock input	SIO0 clock output
P11	SIO0 data input	SIO0 data output
P10	None	SIO0 data output

Table of Port 1 Shared Pin

FUNCTION output 5-0

P1FCR (FE46)
bits 5-0

W-P1FCR

R-P1FCR

Bus

Low-PU

SW

P1 (FE44)
bits 5-0

W-P1

OR

CMOS
or
Nch-OD

Pin

P15～P10

SEL

R-P1

P1DDR (FE45)
bits 5-0

W-P1DDR

R-P1DDR

Port 1 Block Diagram

Option: Output type (CMOS or N-channel OD) selectable on a bit basis.

AII-3

Port Block Diagrams
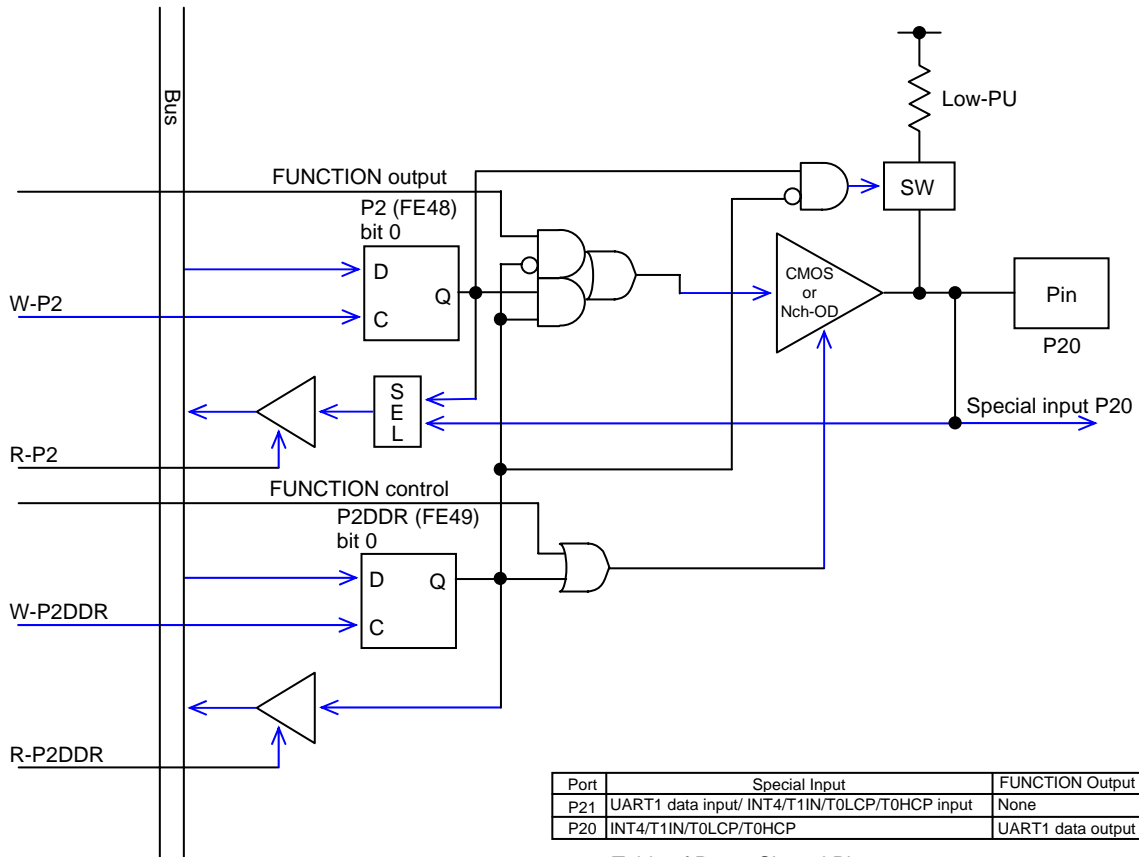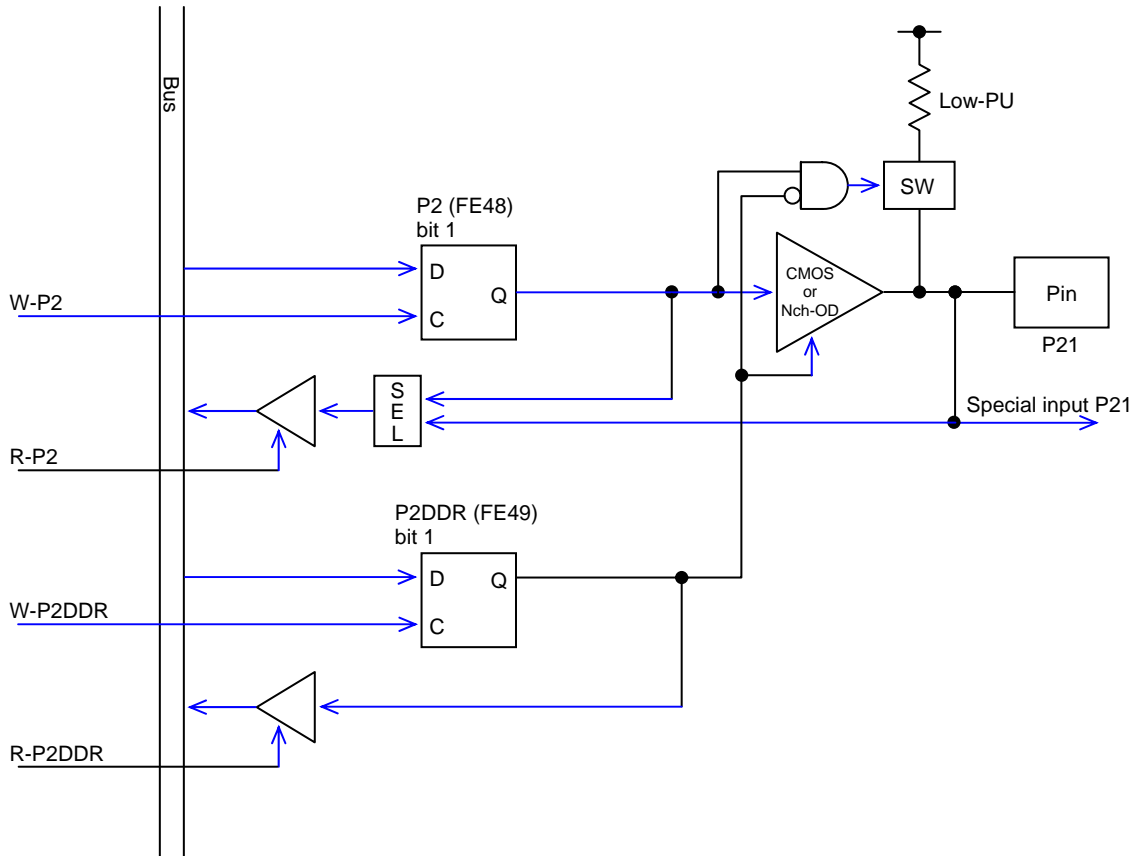
Port	Special Input	FUNCTION Output
P21	UART1 data input/ INT4/T1IN/T0LCP/T0HCP input	None
P20	INT4/T1IN/T0LCP/T0HCP	UART1 data output

Table of Port 2 Shared Pin

Port 2 Block Diagram

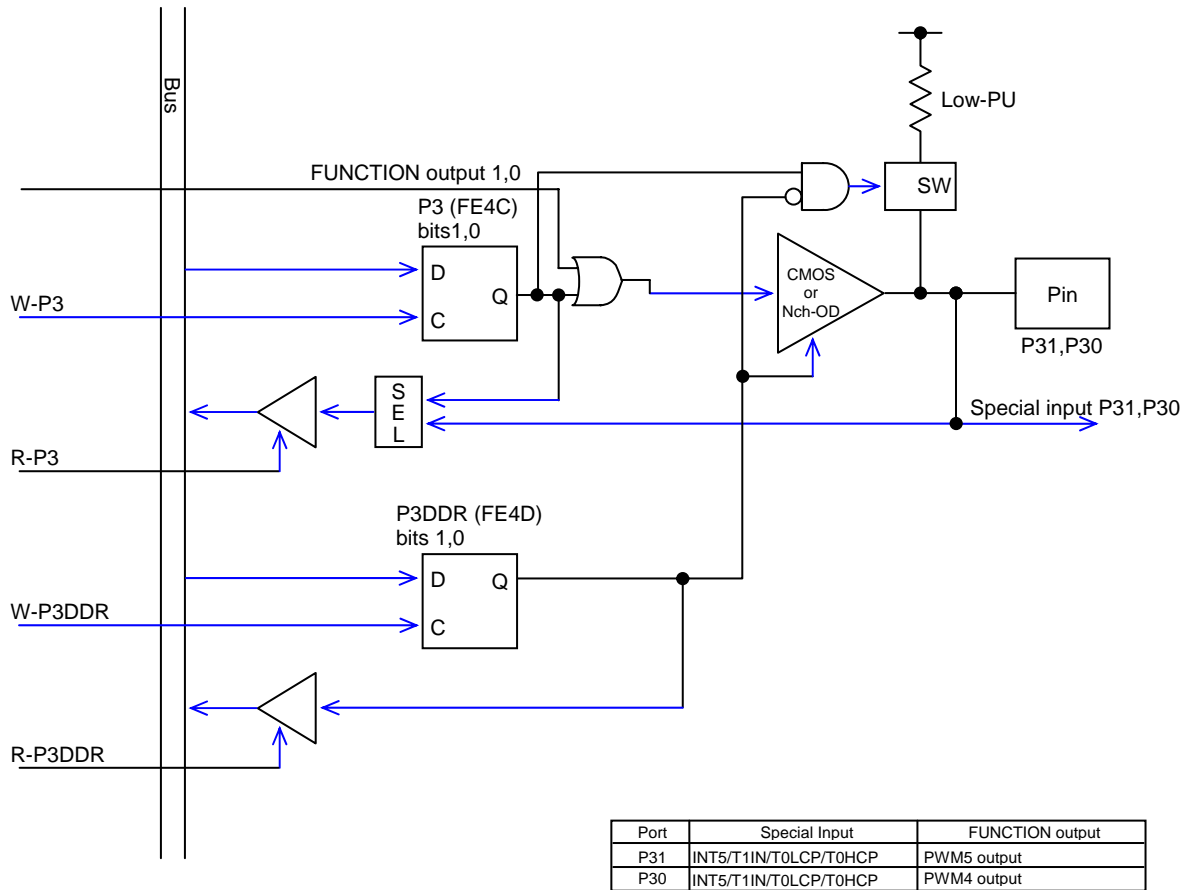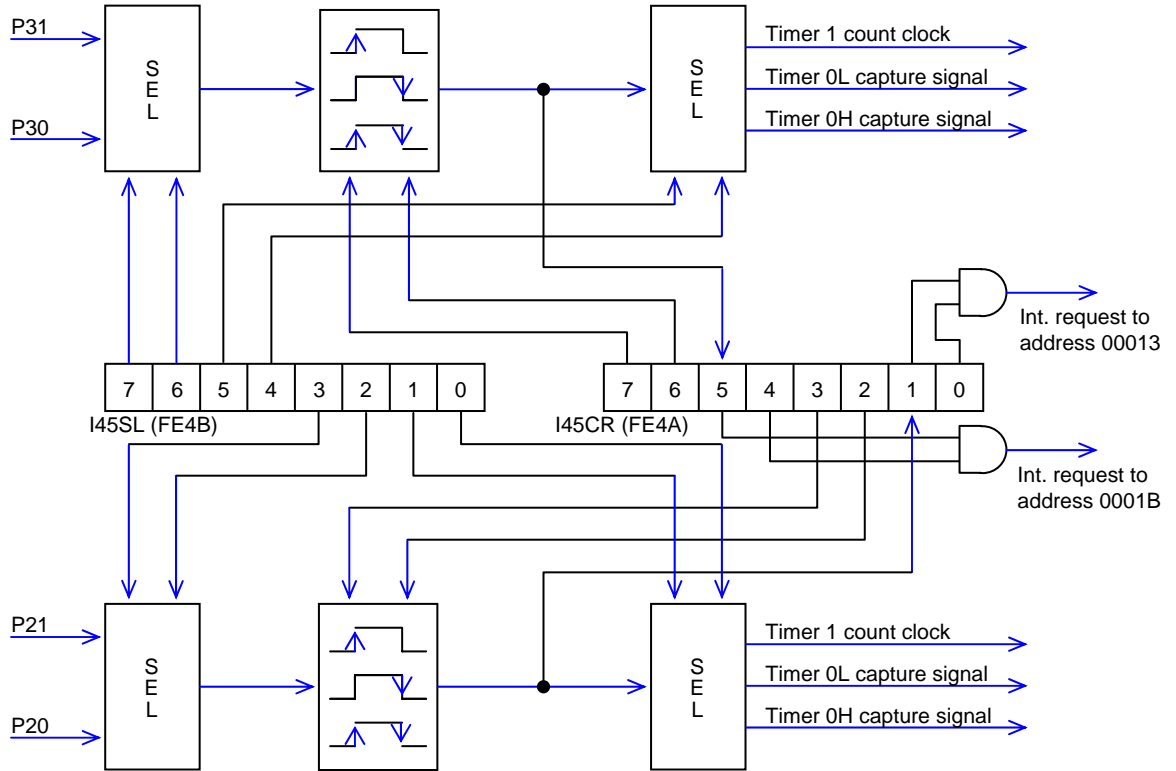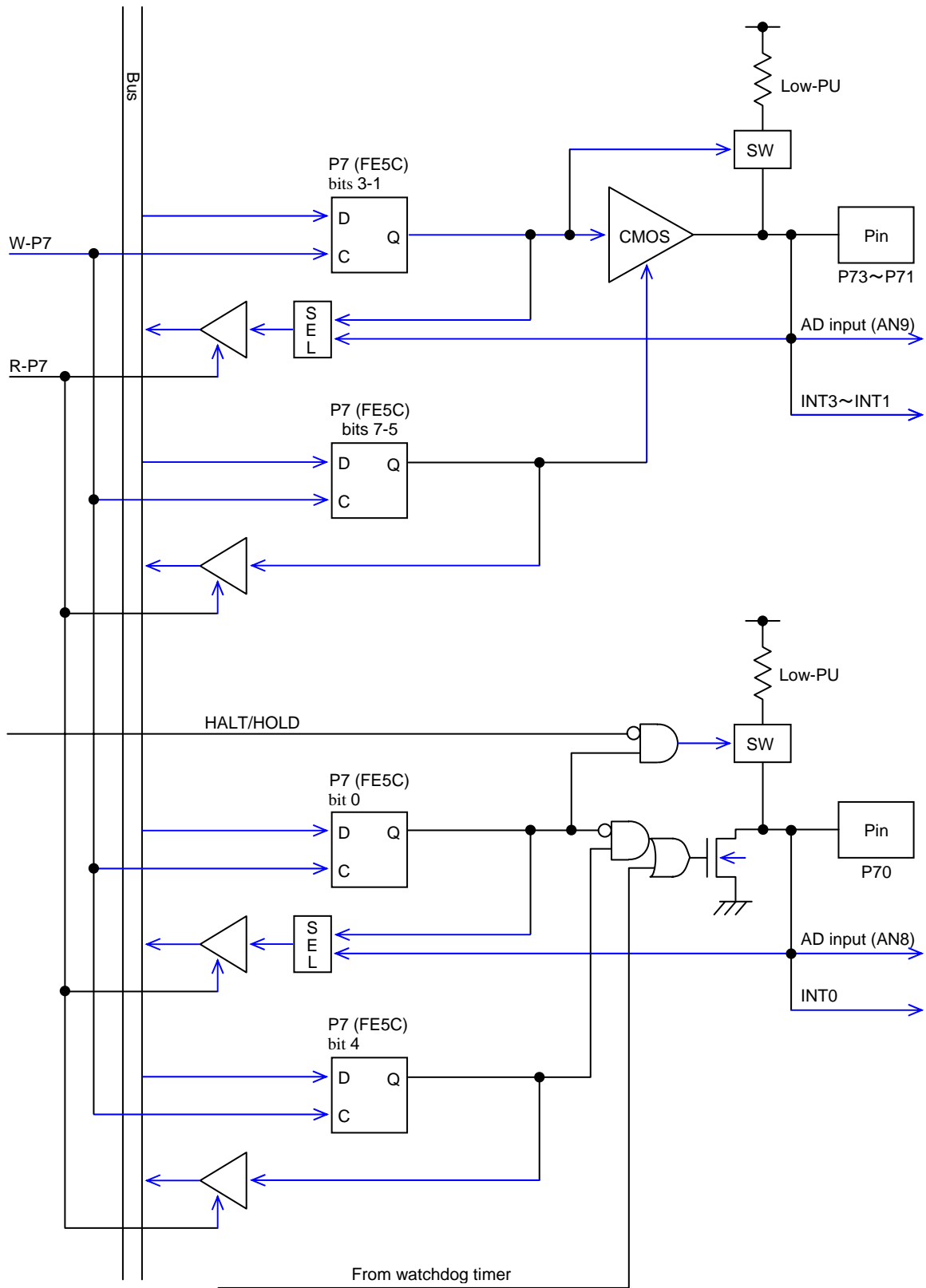**Option: Output type (CMOS or N-channel OD) selectable on a bit basis.**

AⅡ-4

Port	Special Input	FUNCTION output
P31	INT5/T1IN/T0LCP/T0HCP	PWM5 output
P30	INT5/T1IN/T0LCP/T0HCP	PWM4 output

Table of Port 3 Shared Pin

Port 3 Block Diagram
Option: Output type (CMOS or N-channel OD) selectable on a bit basis.
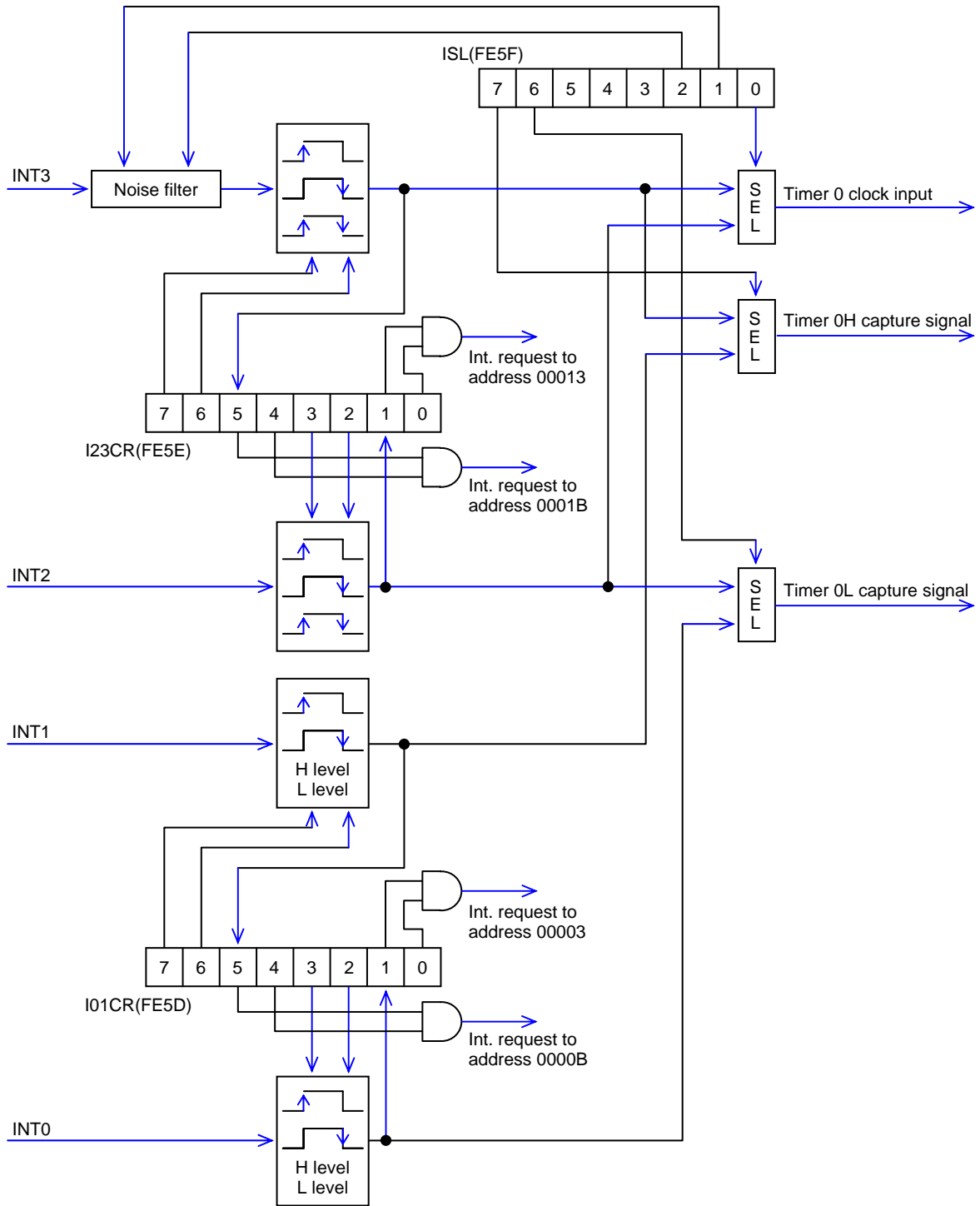
Port Block Diagrams

Ports 2 and 3 (Interrupt) Block Diagram

Port 7 Block Diagram
Option: None

Port Block Diagrams

Port 7 (Interrupt) Block Diagram

Important Note

This document is designed to provide the reader with accurate information in easily understandable form regarding the device features and the correct device implementation procedures.

The sample configurations included in the various descriptions are intended for reference only and should not be directly incorporated in user product configurations.

ON Semiconductor shall bear no responsibility for obligations concerning patent infringements, safety or other legal disputes arising from prototypes or actual products created using the information contained herein.