TOSHIBA

TOSHIBA Original CMOS 16-Bit Microcontroller

TLCS-900/L1 Series

TMP91CP27

TOSHIBA CORPORATION

Semiconductor Company

Preface

Thank you very much for making use of Toshiba microcomputer LSIs. Before use this LSI, refer the section, “Points of Note and Restrictions”.

Especially, take care below cautions.

CAUTION

How to release the HALT mode

Usually, interrupts can release all halts stats. However, the interrupts = ($\overline{NMI}$, INT0, INTRTC), which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 5 clocks of f_{FPH}) with IDLE1 or STOP mode (IDLE2 is not applicable to this case). (In this case, an interupt request is kept on hold internally.)

If another interupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

CMOS 16-Bit Microcontrollers

TMP91CP27U

1. Outline and Features

TMP91CP27 is a high-speed 16-bit microcontroller designed for the control of various mid- to large-scale equipment.

TMP91CP27U comes in a 64-pin flat package. Listed below are the features.

(1) High-speed 16-bit CPU (900/L1 CPU)
- Instruction mnemonics are upward-compatible with TLCS-90/900
- 16 Mbytes of linear address space
- General-purpose registers and register banks
- 16-bit multiplication and division instructions; bit transfer and arithmetic instructions
- Micro DMA: 4 channels (1.0 μs/2 bytes at 16 MHz)

(2) Minimum instruction execution time: 148 ns (at 27 MHz)

(3) Built-in RAM: 4 Kbytes
Built-in ROM: 48 Kbytes

(4) External memory expansion
- Expandable up to 16 Mbytes (Shared program/data area)
- Can simultaneously support 8-/16-bit width external data bus (Dynamic data bus sizing)

(5) 8-bit timers: 6 channels

(6) 16-bit timers: 1 channel

(7) General-purpose serial interface: 2 channels
- UART/Synchronous mode: 2 channels
- IrDA Ver.1.0 (115.2 Kbps) mode selectable: 1 channel

(8) Serial bus interface: 1 channel
- I²C bus mode/clock synchronous mode selectable

(9) 10-bit AD converter (Sample hold circuit is inside): 4 channels

(10) Watchdog timer

(11) Timer for real time clock (RTC)

(12) Chip select/wait controller: 4 blocks

(13) Interrupts: 34 interrupts

- 9 CPU interrupts: Software interrupt instruction and illegal instruction
- 21 internal interrupts: 7 priority levels are selectable
- 4 external interrupts: 7 priority levels are selectable
  (among 3 interrupts are selectable edge mode)

(14) Input/output ports: 53 pins

(15) Standby function
Three HALT modes: IDLE2 (Programmable), IDLE1 and STOP

(16) Clock controller

- Clock gear function: Select a high-frequency clock fc to fc/16
- RTC (fs = 32.768 KHz)

(17) Operating voltage

- VCC = 2.7 V to 3.6 V (fc max = 27 MHz)
- VCC = 1.8 V to 3.6 V (fc max = 10 MHz)

(18) Package

- P-LQFP64-1010-0.50D

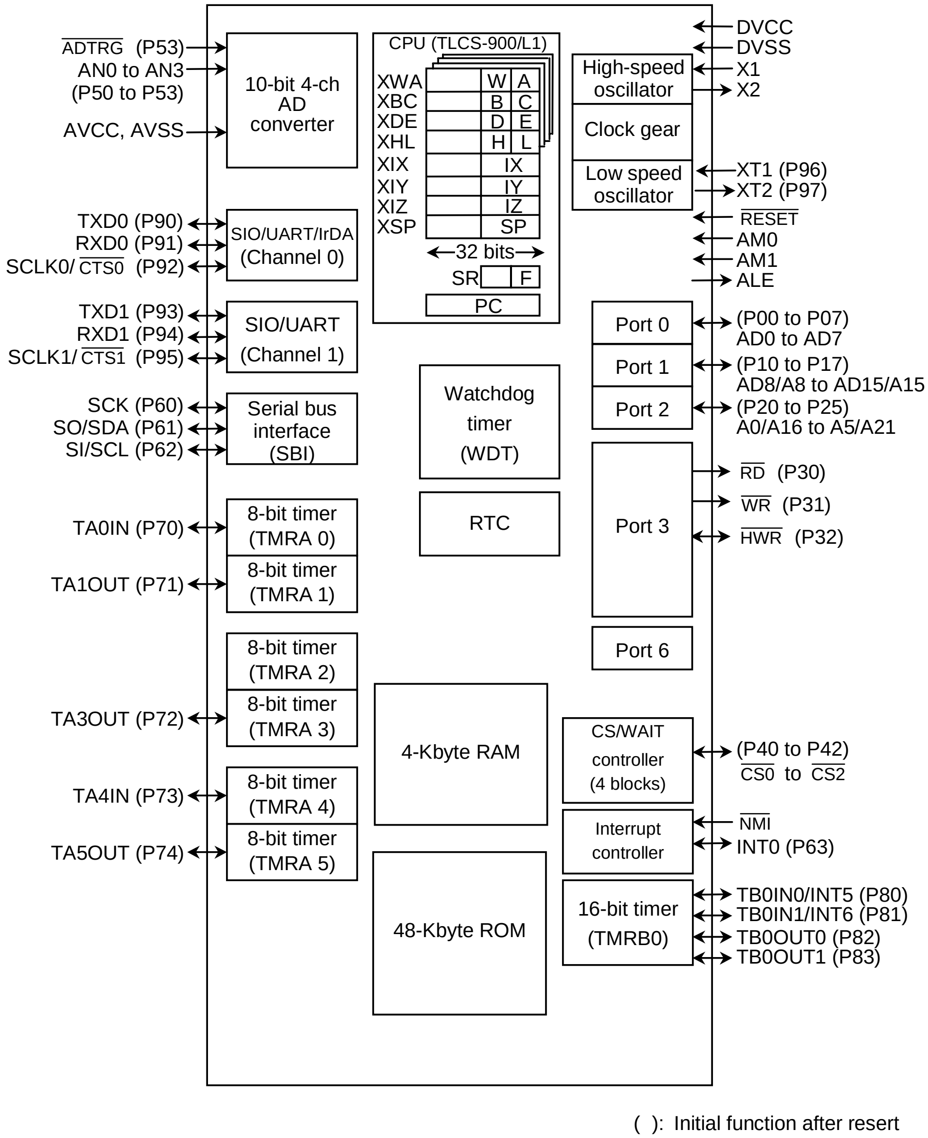


Figure 1.1 TMP91CP27 Block Diagram

2. Pin Assignment and Pin Functions

The assignment of input/output pins for the TMP91CP27U, their names and functions are as follows:

2.1 Pin Assignment Diagram

Figure 2.1 shows the pin assignment of the TMP91CP27U.

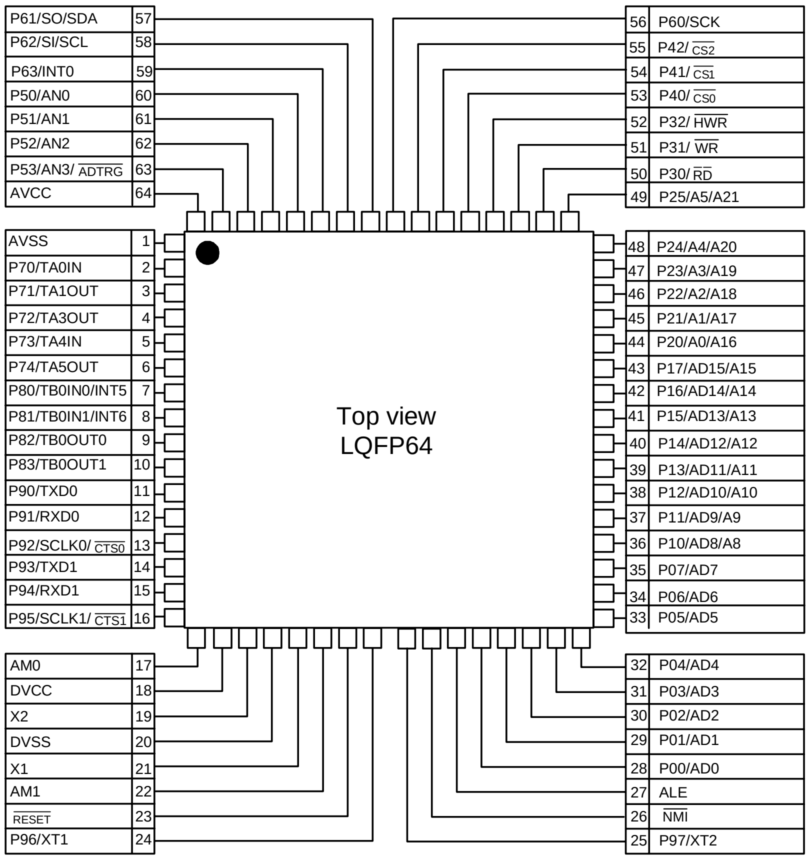


Figure 2.1 Pin Assignment Diagram (64-pin LQFP)

2.2 Pin names and Functions

The names of the input/output pins and their functions are described below.
Table 2.2.1 to Table 2.2.3 show pin names and functions.

Table 2.2.1 Pin Names and Functions (1/3)

Pin Names	Number of Pins	I/O	Functions
P00 to P07 AD0 to AD7	8	I/O I/O	Port 0: I/O port that allows I/O to be selected at the bit level Address data (Lower): 0 to 7 of address/data bus
P10 to P17 AD8 to AD15 A8 to A15	8	I/O I/O Output	Port1: I/O port that allows I/O to be selected at the bit level Address data (Upper): 8 to 15 of address/data bus Address: 8 to 15 of address bus
P20 to P25 A0 to A5 A16 to A21	6	I/O Output Output	Port 2: I/O port that allows I/O to be selected at the bit level Address: 0 to 5 of address bus Address: 16 to 21 of address bus
P30 $\overline{\text{RD}}$	1	Output Output	Port 30: Output port Read: Strobe signal for reading external memory when read internal area also, output $\overline{\text{RD}}$ by setting to P3<P30> = 0, P3FC<P30F> = 1.
P31 $\overline{\text{WR}}$	1	Output Output	Port 31: Output port Write: Strobe signal for writing data to pins AD0 to AD7
P32 $\overline{\text{HWR}}$	1	I/O Output	Port 32: I/O port (with pull-up resistor) High write: Strobe signal for writing data to pins AD8 to AD15
P40 $\overline{\text{CS0}}$	1	I/O Output	Port 40: I/O port (with pull-up resistor) Chip select 0: Outputs “0” when address is within specified address area.
P41 $\overline{\text{CS1}}$	1	I/O Output	Port41: I/O port (with pull-up resistor) Chip select 1: Outputs “0” when address is within specified address area.
P42 $\overline{\text{CS2}}$	1	I/O Output	Port 42: I/O port (with pull-up resistor) Chip select 2: Outputs “0” when address is within specified address area.
P50 to P53 AN0 to AN3 $\overline{\text{ADTRG}}$	4	Input Input Input	Port 5: Input port Analog input: Analog input pins of the AD converter AD trigger: Pin used to request AD start (Shared with P53).
P60 SCK	1	I/O I/O	Port 60: I/O port Serial bus interface clock I/O at SIO mode
P61 SO SDA	1	I/O Output I/O	Port 61: I/O port Serial bus interface send data at SIO mode Serial bus interface send/receive data at I^2C mode Open-drain output mode by programmable
P62 SI SCL	1	I/O Input I/O	Port 62: I/O port Serial bus interface receive data at SIO mode Serial bus interface clock I/O at I^2C mode Open-drain output mode by programmable
P63 INT0	1	I/O Input	Port 63: I/O port (Schmitt input) Interrupt request pin 0: Interrupt request pin with selectable level/rising/falling edge

Table 2.2.2 Pin Names and Functions (2/3)

Pin Names	Number of Pins	I/O	Functions
P70 TA0IN	1	I/O Input	Port 70: I/O port 8-bit timer 0 input: Input pin of 8-bit timer TMRA0
P71 TA1OUT	1	I/O Output	Port 71: I/O port 8-bit timer 1 output: Output pin of 8-bit timer TMRA0 or TMRA1
P72 TA3OUT	1	I/O Output	Port 72: I/O port 8-bit timer 3 output: Output pin of 8-bit timer TMRA2 or TMRA3
P73 TA4IN	1	I/O Input	Port 73: I/O port 8-bit timer 4 input: Input pin of 8-bit timer TMRA4
P74 TA5OUT	1	I/O Output	Port 74: I/O port 8-bit timer 5 output: Output pin of 8-bit timer TMRA4 or TMRA5
P80 TB0IN0 INT5	1	I/O Input Input	Port 80: I/O port 16-bit timer 0 Input 0: Input of count/capture trigger in 16-bit timer TMRB0 Interrupt request pin 5: Interrupt request pin with selectable rising/falling edge
P81 TB0IN1 INT6	1	I/O Input Input	Port 81: I/O port 16-bit timer 0 input 1: Input of count/capture trigger in 16-bit timer TMRB0 Interrupt request pin 6: Interrupt request pin of rising edge
P82 TB0OUT0	1	I/O Output	Port 82: I/O port 16-bit timer 0 output 0: Outpit pin of 16-bit timer TMRB0
P83 TB0OUT1	1	I/O Output	Port 83: I/O port 16-bit timer 0 output 1: Output pin of 16-bit timer TMRB0
P90 TXD0	1	I/O Output	Port 90: I/O port Serial 0 send data: Open-drain output pin by programmable
P91 RXD0	1	I/O Input	Port 91: I/O port Serial 0 receive data
P92 SCLK0 $\overline{\text{CTS0}}$	1	I/O I/O Input	Port 92: I/O port Serial 0 clock I/O Serial 0 data send enable (Clear to send)
P93 TXD1	1	I/O Output	Port 93: I/O port Serial 1 send data: Open-drain output pin by programmable
P94 RXD1	1	I/O Input	Port 94: I/O port Serial 1 receive data
P95 SCLK1 $\overline{\text{CTS1}}$	1	I/O I/O Input	Port 95: I/O port Serial 1 clock I/O Serial 1 data send enable (Clear to send)
P96 XT1	1	I/O Input	Port 96: I/O port. Open-drain output pin Low-frequency oscillator connection pin
P97 XT2	1	I/O Output	Port 97: I/O port. Open-drain output pin Low-frequency oscillator connection pin

Table 2.2.3 Pin Names and Functions (3/3)

Pin Names	Number of Pins	I/O	Functions
ALE	1	Output	Address latch enable (It can be set as prohibition of an output for noise reduction.)
$\overline{NMI}$	1	Input	Non-maskable interrupt request pin: Interrupt request pin with programmable falling edge level or with both edge levels programmable (Schmitt input).
AM0, AM1	2	Input	Operation mode: Fixed to AM1 = "1" and AM0 = "1".
$\overline{RESET}$	1	Input	Reset: Initialize LSI. (Schmitt input, with pull-up resistor)
AVCC	1		Pin used to both power supply pin for AD converter and standard power supply for AD converter (H).
AVSS	1		Pin used to both GND pin for AD converter (0 V) and standard power supply pin for AD converter (L).
X1/X2	2	I/O	High-frequency oscillator connection pin.
DVCC	1		Power supply pins (All DVCC pins should be connected with the power supply pin.)
DVSS	1		GND pins (All pins shuold be connected with GND (0 V).)

3. Operation

This following describes block by block the functions and operation of the TMP91CP27.

3.1 CPU

The TMP91CP27 incorporates a high-performance 16-bit CPU (The 900/L1-CPU). For CPU operation, see the "TLCS-900/L1 CPU".

The following describe the unique function of the CPU used in the TMP91CP27; these functions are not covered in the TLCS-900/L1 CPU section.

3.1.1 Reset

When resetting the TMP91CP27 microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input to low level at least for 10 system clocks (80 μs at 4 MHz).

Thus, when turn on the switch, be set to the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input to Low level at least for 10 system clocks.

Clock gear is initialized 1/16 mode by reset operation. It means that the system clock mode f_{SYS} is set to fc/32 (= fc/16 × 1/2).

When the reset is accept, the CPU:

- Sets as follows the program counter (PC) in accordance with the reset vector stored at address FFFF00H to FFFF02H:

  PC<7:0> ← Value at FFFF00H address
  PC<15:8> ← Value at FFFF01H address
  PC<23:16> ← Value at FFFF02H address

- Sets the stack pointer (XSP) to 100H.
- Sets bits<IFF2:0> of the status register (SR) to 111 (Sets the interrupt level mask register to level 7).
- Sets the <MAX> bit of the status register (SR) to 1 (MAX mode).
- Clears bits<RFP2:0> of the status register (SR) to 000 (Sets the register bank to 0).

When reset is released, the CPU starts executing instructions in accordance with the program counter settings. CPU internal registers not mentioned above do not change when the reset is released.

When the reset is accepted, the CPU sets internal I/O, ports, and other pins as follows.

- Initializes the internal I/O registers.
- Sets the port pins, including the pins that also act as internal I/O, to general-purpose input or output port mode.
- Sets ALE pin to high impedance.

Note: The CPU internal register (except to PC, SR, XSP in CPU) and internal RAM data do not change by resetting.

Figure 3.1.1 is a reset timing chart of the TMP91CP27.

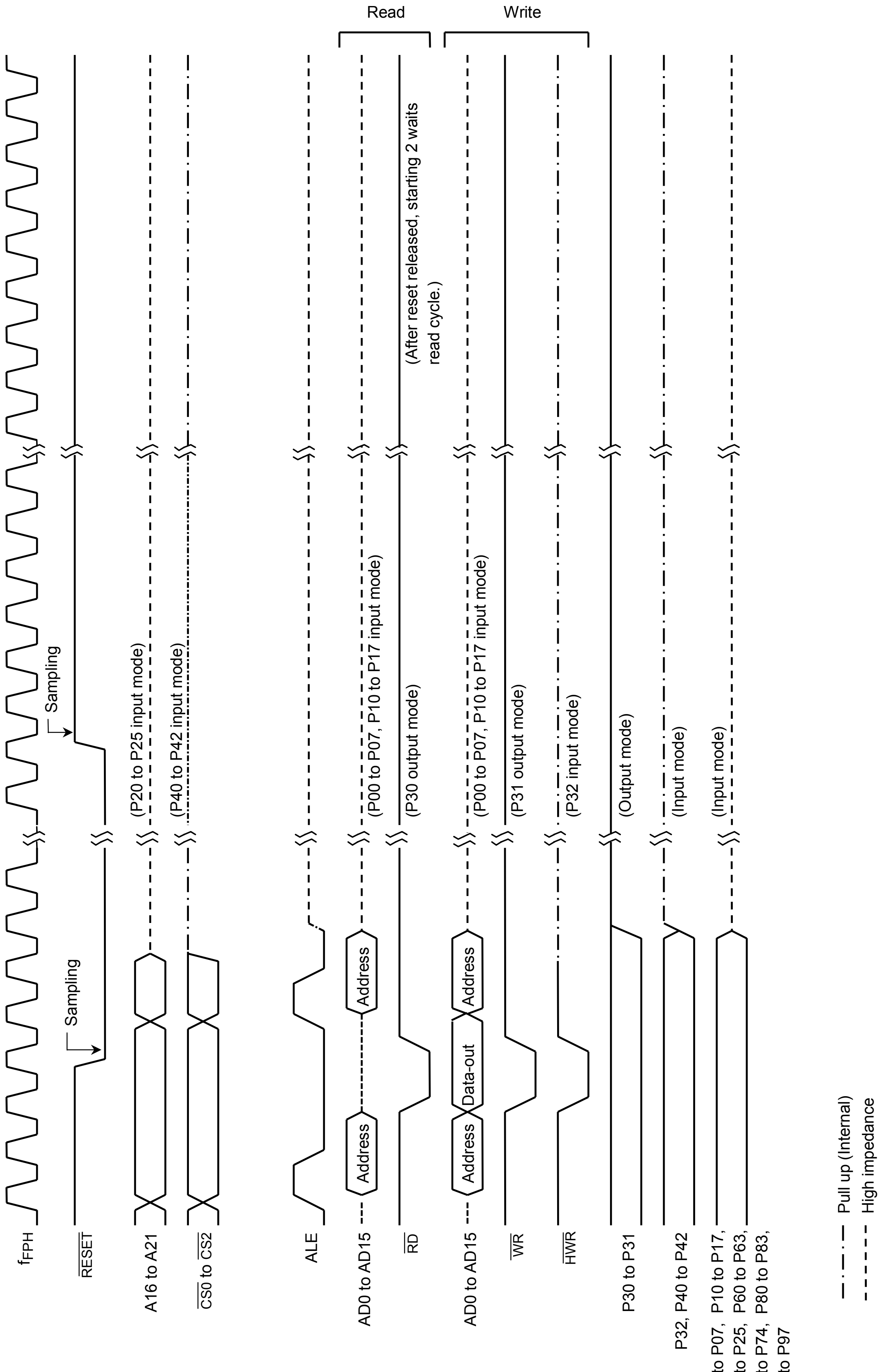


Figure 3.1.1 TMP91CP27 Reset Timing Chart

3.2 Memory Map

Figure 3.2.1 is a memory map of the TMP91CP27.

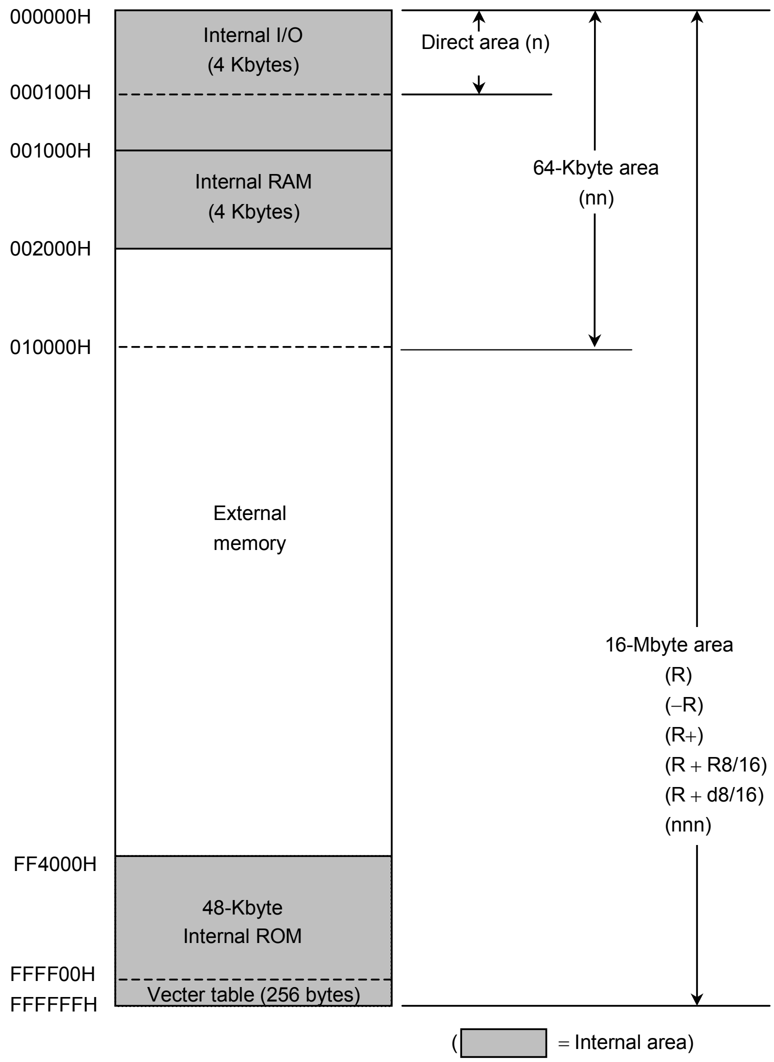


Figure 3.2.1 Memory Map

3.3 Diversity of TMP91CW12A and TMP91CP27

TMP91CP27 is article cut function and pin from TMP91CW12A fundamentally.

Specification of function shows section 3.3.1 to 3.3.6. Wide difference of AC/DC characteristic is AC characteristics (Shown section 3.3.7). For the details, please refer to Chapter 4, "Electrical characteristics".

3.3.1 Cut Internal I/O

TMP91CP27 is micro controller cut 8-bit timer (TMRA6 to TMRA7), 16-bit timer (TMRB1) and clock doublers circuit (DFM) from TMP91CW12A.

Please don't access to special function register address of above internal I/O in TMP91CW12A.

Please refer to "Table of special function register".

3.3.2 Cut Port Function

TMP91CP27 be cut below port function from TMP91CW12A.

- Port 2: P27 (A23/A7) and P26 (A22/A6)
- Port 3: P37, P36 ($R/\overline{W}$), P35 ($\overline{BUSAK}$), P34 ($\overline{BUSRQ}$) and P33 ($\overline{WAIT}$)
- Port 4: P43 ($\overline{CS3}$)
- Port 5: P57 to P54 (AN7 to AN4)
- Port 6: P66, P65 and P64 (SCOUT)
- Port 7: P75 (TA7OUT)
- Port 8: P87 (TB1OUT1), P86 (TB1OUT0), P85 (TB1IN1/INT8) and P84 (TB1IN0/INT7)
- Port A: PA7 to PA4, PA3 to PA0 (INT4 to INT1)

3.3.3 Cut factor of Interrupt

TMP91CP27 be cut factor of interrupt by cut internal I/O and port function (Refer to Table 3.5.1). Please don't access to interrupt priority setting register for cut factor of interrupt. Please refer to "table of special function register".

3.3.4 Bus Release Function

TMP91CP27 don't include bus release function by cutting bus request pin (P34) and bus acknowledge pin (P35).

3.3.5 CS/WAIT Controller

When set TMP91CP27 to BxCS<BxW2:0> = "010" (1 + N) WAIT mode, it operates as 1 wait by cutting $\overline{WAIT}$ pin.

And there is not $\overline{CS3}$ pin (P43), but when set MSAR3, MAMR3 and set B3CS<B3E> = "1", wait control is effective.

3.3.6 AD Converter

Analog input pin AN4 to AN7 be cut. Therefore please don’t select cutting channel in ADMOD1<ADCH2:0>.

3.3.7 AC Characteristic

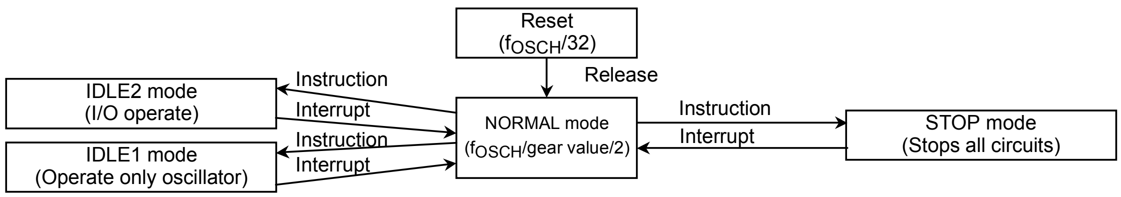
When accessing to external, AC characteristic don’t guarantee at 2 V operation.

3.4 System Clock Function and Standby Control

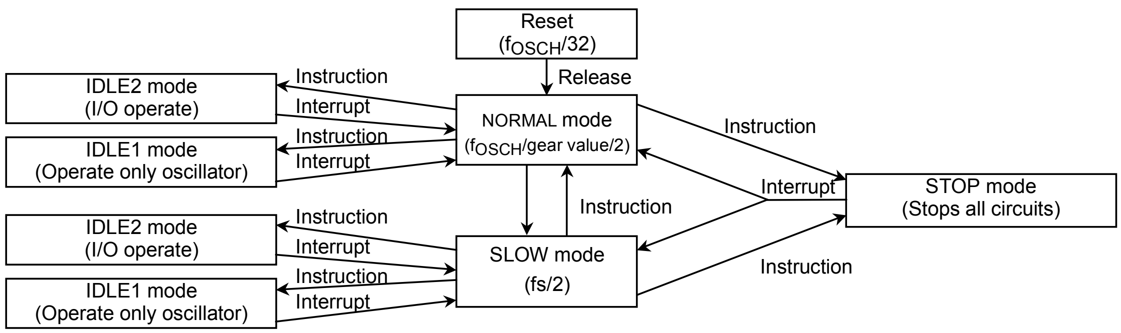
TMP91CP27 contains (1) a clock gear, (2) stand-by controller and (3) noise-reduction circuit. It is used for low-power and low-noise systems.

The clock operating modes are as follows: (a) Single clock mode (X1 and X2 pins only), (b) Dual clock mode (X1, X2, XT1 and XT2 pins).

Figure 3.4.1 shows a transition figure.



(a) Single clock mode transition figure



(b) Dual clock mode transition fiigure

Figure 3.4.1 Clock Operating Mode

Note: The clock frequency input from the X1 and X2 pins is called f_{OSCH} and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<SYSCK> is called f_{FPH}. The system clock f_{SYS} is defined as the divided clock of f_{FPH}, and one cycle of f_{SYS} is regret to as one state.

3.4.1 Block Diagram of System Clock

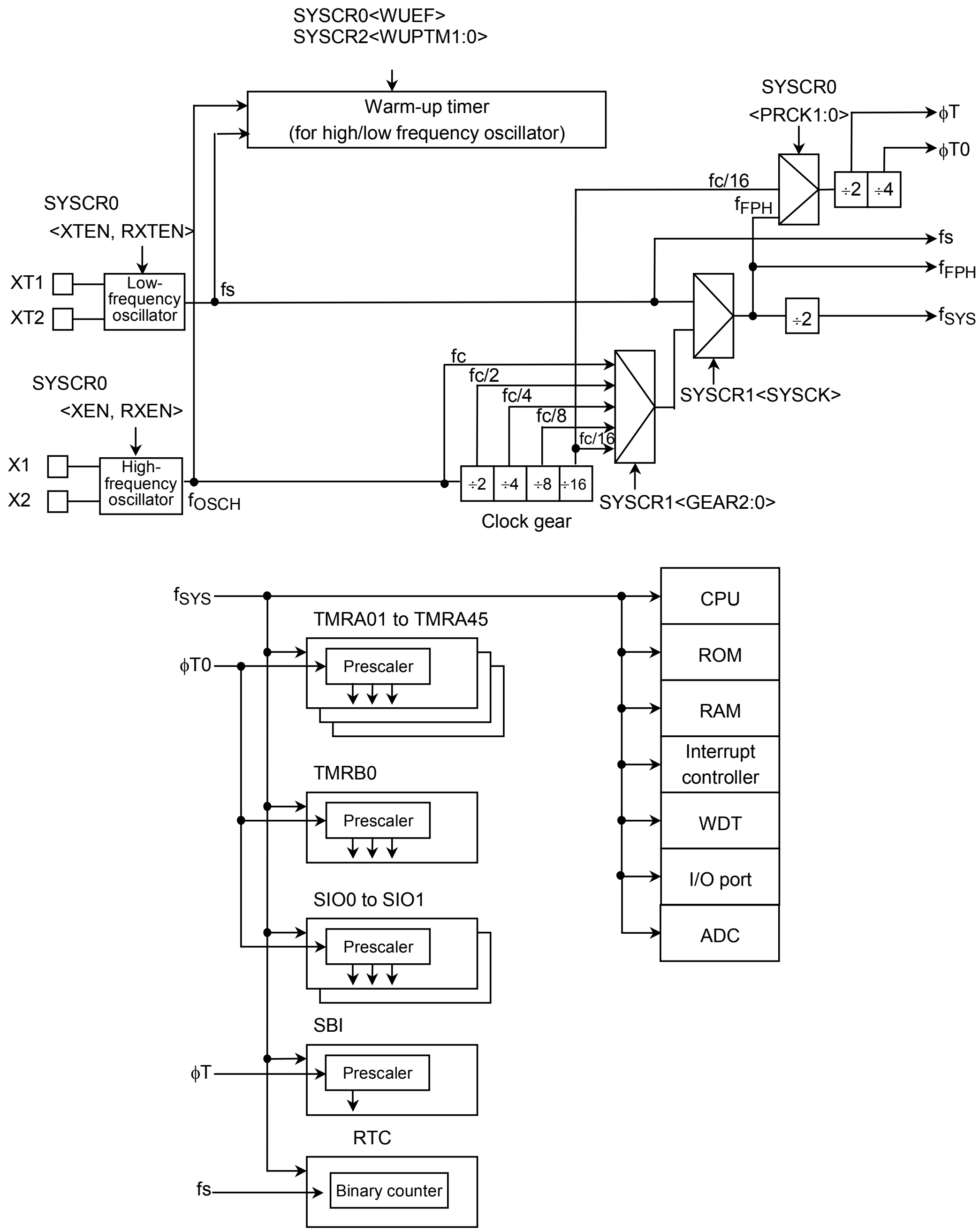


Figure 3.4.2 Block Diagram of System Clock

3.4.2 SFR

		7	6	5	4	3	2	1	0
SYSCR0 (00E0H)	Bit symbol	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	PRCK1	PRCK0
	Read/Write	R/W							
	After reset	1	0	1	0	0	0	0	0
	Function	High-frequency oscillator (fc) 0: Stop 1: Oscillation	Low-frequency oscillator (fs) 0: Stop 1: Oscillation	High-frequency oscillator (fc) after release of STOP mode 0: Stop 1: Oscillation	Low-frequency oscillator (fs) after release of STOP mode 0: Stop 1: Oscillation	Selects clock after release of STOP mode 0: fc 1: fs	Warm-up timer control 0 Write: Don't care 1 Write: Start warm-up 0 Read: End warm-up 1 Read: Do not end warm-up	Select prescaler clock 00: f_{FPH} 01: Reserved 10: fc/16 11: Reserved (Note 2)	

		7	6	5	4	3	2	1	0
SYSCR1 (00E1H)	Bit symbol					SYSCK	GEAR2	GEAR1	GEAR0
	Read/Write					R/W			
	After reset					0	1	0	0
	Function					Select system clock 0: fc 1: fs	Select gear value of high frequency (fc) 000: fc 001: fc/2 010: fc/4 011: fc/8 100: fc/16 101: (Reserved) 110: (Reserved) 111: (Reserved)		

		7	6	5	4	3	2	1	0
SYSCR2 (00E2H)	Bit symbol		–	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
	Read/Write		R/W	R/W	R/W	R/W	R/W		R/W
	After reset		0	1	0	1	1		0
	Function		Write to "0".	Select warm-up time for oscillator 00: Reserved 01: 2^{8}/inputted frequency 10: 2^{14}/inputted frequency 11: 2^{16}/inputted frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			Pin state control in STOP mode 0: I/O off 1: Remains the state before HALT

Note 1: SYSCR1<bit7:4>, SYSCR2<bit7,bit1> are read as undefined value.

Note 2: When using internal SBI, set prescaler clock select register SYSCR0<PRCK1:0> to f_{FPH}.

Figure 3.4.3 SFR for System Clock

		7	6	5	4	3	2	1	0
EMCCR0 (00E3H)	Bit symbol	PROTECT	–	–	–	ALEEN	EXTIN	DRVOSCH	DRVOSCL
	Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	1	0	0	0	1	1
	Function	Protect flag 0: OFF 1: ON	Write "0".	Write "1".	Write "0".	1: ALE output enable 0: ALE output disable	1: fc external clock	fc oscillator driver ability 1: NORMAL 0: WEAK	fs oscillator driver ability 1: NORMAL 0: WEAK
EMCCR1 (00E4H)	Bit symbol	Protect OFF by writing "1FH". Protect ON by writing except "1FH".							
	Read/Write								
	After reset								
	Function								

Figure 3.4.4 SFR for Noise Reduction

3.4.3 System Clock Controller

The system clock controller generates the system clock signal (f_{SYS}) for the CPU core and internal I/O. It contains two oscillation circuits and a clock gear circuit for high-frequency (fc) operation. The register SYSCR1<SYSCK> changes the system clock to either fc or fs, SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling of each oscillator, and SYSCR1<GEAR2:0> sets the high-frequency clock gear to either 1, 2, 4, 8 or 16 (fc, fc/2, fc/4, fc/8 or fc/16). These functions can reduce the power consumption of the equipment in which the device is installed.

The combination of settings <XEN> = “1”, <XTEN> = “0”, <SYSCK> = “0” and <GEAR2:0> = “100” will cause the system clock (f_{SYS}) to be set to fc/32 (fc/16 × 1/2) after a Reset.

For example, f_{SYS} is set to 0.5 MHz when the 16 MHz oscillator connected to the X1 and X2 pins.

(1) Switching from NORMAL mode to SLOW mode

When the resonator is connected to the X1 and X2 pins, or to the XT1 and XT2 pins, the warm-up timer can be used to change the operation frequency after stable oscillation has been attained.

The warm-up time can be selected using SYSCR2<WUPTM1:0>.

This warm-up timer can be programmed to start and stop as shown in the following examples 1 and 2.

Table 3.4.1 shows the warm-up time.

Note 1: When using an oscillator (other than a resonator) with stable oscillation, a warm-up timer is not needed.

Note 2: The warm-up timer is operated by an oscillation clock. Hence, there may be some variation in warm-up time.

Note 3: Note of using low-frequency oscillator

When connect low-frequency oscillator to ports 96 and 97, need below setting for cut consumption power.

(Case of resonators)

Set P9CR<P96C, P97C> = “11”, P9<P96, P97> = “00”

(Case of oscillator)

Set P9CR<P96C, P97C> = “11”, P9<P96, P97> = “10”

Table 3.4.1 Warm-up Times (when changing clock)

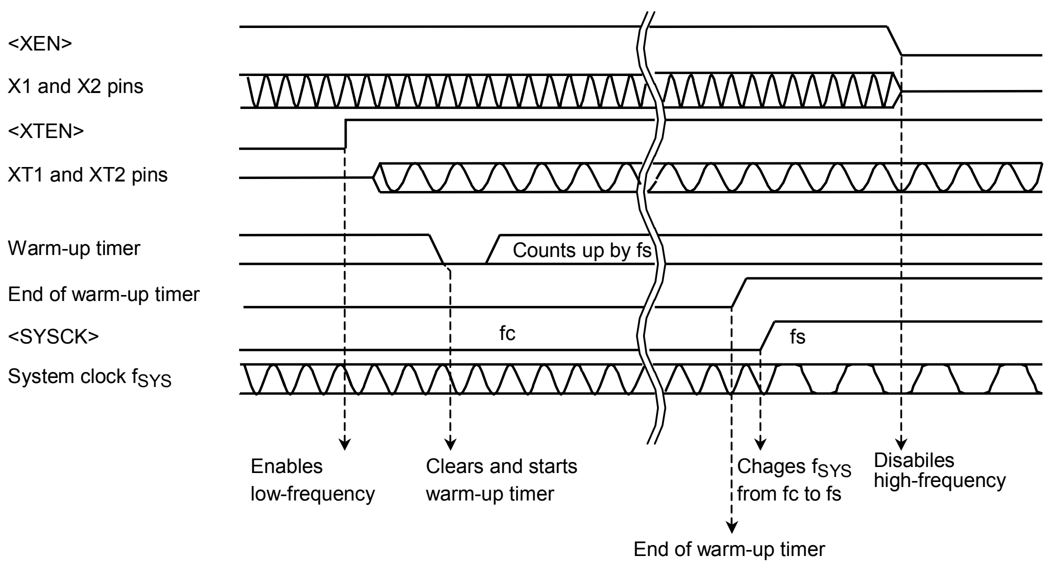
Select Warm-up Time SYSCR2<WUPTM1:0>	Change to NORMAL (fc)	Change to SLOW (fs)
01 (2^8/frequency)	16 [μs]	7.8 [ms]
10 (2^{14}/frequency)	1.024 [ms]	500 [ms]
11 (2^{16}/frequency)	4.096 [ms]	2000 [ms]

At f_{OSCH} = 16 MHz, fs = 32.768 kHz

Example 1: Setting the clock
Changing from high frequency (fc) to low frequency (fs).

```
SYSCR0   EQU   00E0H
SYSCR1   EQU   00E1H
SYSCR2   EQU   00E2H
         LD    (SYSCR2), X−11− −X−B   ;  Sets warm-up time to 2^16/fs.
         SET   6, (SYSCR0)            ;  Enables low-frequency oscillation.
         SET   2, (SYSCR0)            ;  Clears and starts warm-up timer.
WUP:     BIT   2, (SYSCR0)            ;} Detects stopping of warm-up timer.
         JR    NZ, WUP                ;}
         SET   3, (SYSCR1)            ;  Changes fSYS from fc to fs.
         RES   7, (SYSCR0)            ;  Disables high-frequency oscillation.
```

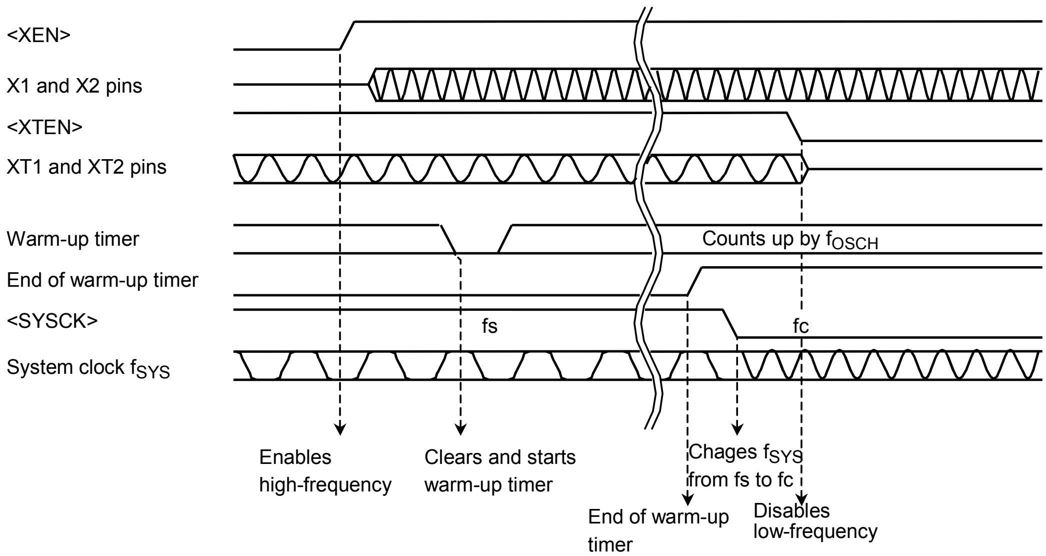
X: Don't care, −: No change



Example 2: Setting the clock
Changing from low frequency (fs) to high frequency (fc).

```
SYSCR0   EQU    00E0H
SYSCR1   EQU    00E1H
SYSCR2   EQU    00E2H
         LD     (SYSCR2), X–10– –X–B    ;   Sets warm-up time to 2^14/fc.
         SET    7, (SYSCR0)             ;   Enables high-frequency oscillation.
         SET    2, (SYSCR0)             ;   Clears and starts warm-up timer.
WUP:     BIT    2, (SYSCR0)             ;}  Detects stopping of warm-up timer.
         JR     NZ, WUP                 ;}
         RES    3, (SYSCR1)             ;   Changes fSYS from fs to fc.
         RES    6, (SYSCR0)             ;   Disables low-frequency oscillation.
```

X: Don’t care, –: No change



(2) Clock gear controller

When the high-frequency clock fc is selected by setting SYSCR1<SYSCK> = "0", f_{FPH} is set according to the contents of the clock gear select register SYSCR1<GEAR2:0> to either fc, fc/2, fc/4, fc/8 or fc/16. Using the clock gear to select a lower value of f_{FPH} reduces power consumption.

Below show example of changing clock gear.

Example 3: Changing to a clock gear

```
SYSCR1   EQU   00E1H
         LD    (SYSCR1), XXXX0000B     ;   Changes fSYS to fc/2.
      X: Don't care
```

(Clock gear changing)

To change the clock gear, write the register value to the SYSCR1<GEAR2:0> register. It is necessary the warm-up time until changing after writing the register value.

There is the possibility that the instruction next to the clock gear changing instruction is executed by the clock gear before changing. To execute the instruction next to the clock gear switching instruction by the clock gear after changing, input the dummy instruction as follows (instruction to execute the write cycle).

Example:

```
SYSCR1   EQU   00E1H
         LD    (SYSCR1), XXXX0001B     ;   Changes fSYS to fc/4.
         LD    (DUMMY), 00H            ;   Dummy instruction
         Instruction to be executed after clock gear has changed.
```

3.4.4 Prescaler Clock Controller

For the internal I/O (TMRA01 to TMRA45, TMRB0, SIO0 to SIO1 and SBI) there is a prescaler which can divide the clock.

The ϕT, $\phi T0$ clock input to the prescaler is either the clock f_{FPH} divided by 2 or the clock fc/16 divided by 4. The setting of the SYSCR0<PRCK1:0> register determines which clock signal is input.

When using internal SBI, set SYSCR0<PRCK1:0> to “00”.

3.4.5 Noise Reduction Circuits

Noise reduction circuits are built in, allowing implementation of the following features.

(1) Reduced drivability for high-frequency oscillator

(2) Reduced drivability for low-frequency oscillator

(3) Single drive for high-frequency oscillator

(4) Output ALE pin disable

(5) SFR protection of register contents

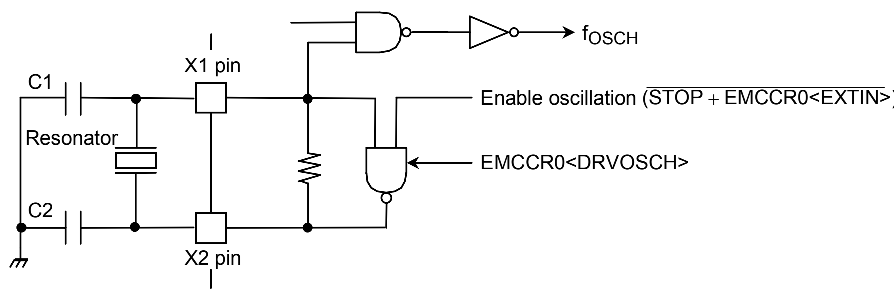
The above functions are performed by making the appropriate settings in the EMCCR0 to EMCCR1 registers.

(1) Reduced drivability for high-frequency oscillator

(Purpose)

Reduces noise and power for oscillator when a resonator is used.

(Block diagram)



(Setting method)

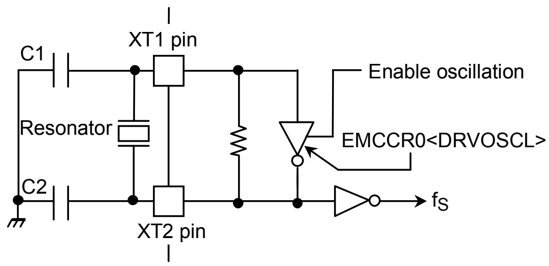
The drivability of the oscillator is reduced by writing “0” to EMCCR0<DRVOSCH> register. By reset, <DRVOSCH> is initialized to “1” and the oscillator starts oscillation by normal drivability when the power supply is on.

(2) Reduced drivability for low-frequency oscillator

(Purpose)

Reduces noise and power for oscillator when a resonator is used.

(Block diagram)



(Setting method)

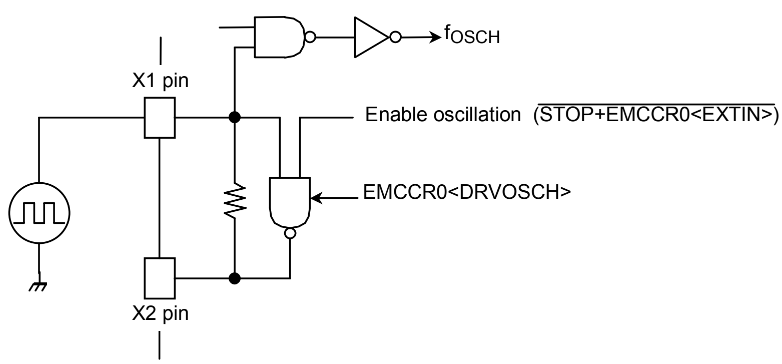
The drivability of the oscillator is reduced by programming 0 to the EMCCR0<DRVOSCL> register. By Reset, <DRVOSCL> is initialized to "1".

(3) Single drive for high-frequency oscillator

(Purpose)

Not need twin-drive and protect mistake operation by inputted noise to X2 pin when the external oscillator is used.

(Block diagram)



(Setting method)

The oscillator is disabled by programming "1" to EMCCR0<EXTIN> register. X2 pin is always outputted "1".

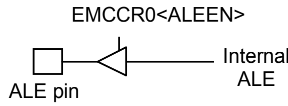
By reset, <EXTIN> is initialized to "0".

(4) Output ALE pin disable

(Purpose)

Not need noise of clock property case of don't access external area is reduced.

(Block diagram)



(Setting method)

Output buffer of ALE pin is output disable by programming "0" to EMCCR0<ALEEN>. And ALE pin is high impedance.

By resetting, <ALEEN> is initialized to "0".

When you access to external area, you must program "1" to <ALEEN> before access.

(5) Runaway provision with SFR protection register

(Purpose)

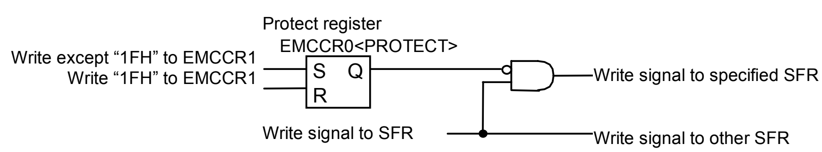
Provision in runaway of program by noise mixing.

Write operation to specified SFR is prohibited so that provision program in runaway prevents that it is it in the state which is fetch impossibility by stopping of clock, memory control register (CS/WAIT controller) is changed.

Specified SFR list

1. CS/WAIT controller
   B0CS, B1CS, B2CS, B3CS, BEXCS,
   MSAR0, MSAR1, MSAR2, MSAR3,
   MAMR0, MAMR1, MAMR2, MAMR3
2. Clock gear (write enable only EMCCR1)
   SYSCR0, SYSCR1, SYSCR2, EMCCR0

(Block diagram)



(Setting method)

If writing except "1FH" code to EMCCR1 register, it become protect ON. By this operation, write operation to specified SFR is disabling.

If writing "1FH" to EMCCR1 register, it become protect OFF. State of protect can to confirm by reading EMCCR0<PROTECT>.

3.4.6 Standby Controller

(1) HALT modes

When the HALT instruction is executed, the operating mode switches to IDLE2, IDLE1 or STOP mode, depending on the contents of the SYSCR2<HALTM1:0> register.

The subsequent actions performed in each mode are as follows:

a. IDLE2: Only the CPU halts.

The internal I/O is available to select operation during IDLE2 mode by setting the following register.

Shows the registers of setting operation during IDLE2 mode.

Table 3.4.2 SFR Setting Operation during IDLE2 Mode

Internal I/O	SFR
TMRA01	TA01RUN<I2TA01>
TMRA23	TA23RUN<I2TA23>
TMRA45	TA45RUN<I2TA45>
TMRB0	TB0RUN<I2TB0>
SIO0	SC0MOD1<I2S0>
SIO1	SC1MOD1<I2S1>
SBI	SBI0BR0<I2SBI0>
AD converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

b. IDLE1: Only the oscillator and the RTC (Real time clock) continue to operate.

c. STOP: All internal circuits stop operating.

The operation of each of the different HALT modes is described in Table 3.4.3.

Table 3.4.3 I/O Operation during HALT Modes

<table>
<tr><th colspan="2">HALT Mode</th><th>IDLE2</th><th>IDLE1</th><th>STOP</th></tr>
<tr><td colspan="2">SYSCR2<HALTM1:0></td><td>11</td><td>10</td><td>01</td></tr>
<tr><td rowspan="8">Block</td><td>CPU</td><td colspan="3">Stop</td></tr>
<tr><td>I/O port</td><td colspan="2">Keep the state when the HALT instruction was executed.</td><td>See Table 3.4.6, Table 3.4.7.</td></tr>
<tr><td>TMRA, TMRB</td><td rowspan="4">Available to select operation block</td><td colspan="2" rowspan="4">Stop</td></tr>
<tr><td>SIO, SBI</td></tr>
<tr><td>AD</td></tr>
<tr><td>WDT</td></tr>
<tr><td>RTC</td><td colspan="2">Operate enable</td><td rowspan="2"></td></tr>
<tr><td>Interrupt controller</td><td>Operate</td><td></td></tr>
</table>

(2) How to release the HALT mode

These halt states can be released by resetting or requesting an interrupt. The halt release sources are determined by the combination between the states of interrupt mask register <IFF2:0> and the HALT modes. The details for releasing the halt status are shown in Table 3.4.4.

- Released by requesting an interrupt

  The operating released from the HALT mode depends on the interrupt enabled status. When the interrupt request level set before executing the HALT instruction exceeds the value of interrupt mask register, the interrupt due to the source is processed after releasing the HALT mode, and CPU status executing an instruction that follows the HALT instruction. When the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, releasing the HALT mode is not executed. (In non-maskable interrupts, interrupt processing is processed after releasing the HALT mode regardless of the value of the mask register.) However only for INT0 and RTC interrupts, even if the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, releasing the the HALT mode is executed. In this case, interrupt processing, and CPU starts executing the instruction next to the HALT instruction, but the interrupt request flag is held at “1”.

  Note: Usually, interrupts can release all halts status. However, the interrupts = ($\overline{\text{NMI}}$, INT0, INTRTC) which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 5 clocks of f_{FPH}) with IDLE1 or STOP mode (IDLE2 is not applicable to this case). (In this case, an interrupt request is kept on hold internally.)
  If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficulty. The priority of this interrupt is compared with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

- Releasing by resetting

  Releasing all halt status is executed by resetting.

  When the STOP mode is released by RESET, it is necessry enough resetting time (See Table 3.4.5) to set the operation of the oscillator to be stable.

  When releasing the HALT mode by resetting, the internal RAM data keeps the state before the “HALT” instruction is executed. However the other settings contents are initialized. (Releasing due to interrupts keeps the state before the “HALT” instruction is executed.)

Table 3.4.4 Source of Halt State Clearance and Halt Clearance Operation

Status of Received Interrupt			Interrupt Enable (Interrupt level) ≥ (Interrupt mask)			Interrupt Disable (Interrupt level) < (Interrupt mask)		
HALT Mode			IDLE2	IDLE1	STOP	IDLE2	IDLE1	STOP
Source of Halt state clearance	Interrupt	NMI	♦	♦	♦ *1	–	–	–
		INTWD	♦	×	×	–	–	–
		INT0 (Note1)	♦	♦	♦ *1	○	○	○ *1
		INTRTC	♦	♦	×	○	○	×
		INT5 to INT6	♦ (Note 2)	×	×	×	×	×
		INTTA0 to INTTA5	♦	×	×	×	×	×
		INTTB00, INTTB01, OF0	♦	×	×	×	×	×
		INTRX0 to INTRX1, INTTX0 to INTTX1	♦	×	×	×	×	×
		INTSBI	♦	×	×	×	×	×
		INTAD	♦	×	×	×	×	×
	RESET		Initialize LSI					

♦: After clearing the HALT mode, CPU starts interrupt processing.

○: After clearing the HALT mode, CPU resumes executing starting from instruction following the HALT instruction. (Interrupt routine don't execute.)

×: It can not be used to release the HALT mode.

–: The priority level (Interrupt request level) of non-maskable interrupts is fixed to 7, the highest priority level. There is not this combination type.

*1: Releasing the HALT mode is executed after passing the warm-up time.

Note 1: When the HALT mode is cleared by an INT0 interrupt of the level mode in the interrupt enabled status, hold high level until starting interrupt process. If low level was set before interrupt process is stared, interrupt process is not started correctly.

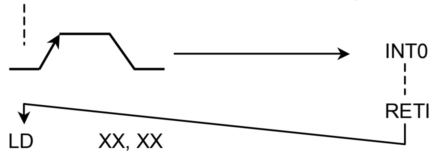
Note 2: If using external interrupt INT5 to INT6 in IDLE2 mode, set 16-bit timer RUN register TB0RUN<I2TB0> to "1".

Example: Clearing halt state
An INT0 interrupt clears the halt state when the device is in IDLE1 mode.

```
Address
8203H    LD      (IIMC), 00H        ; Selects INT0 interrupt rising edge.
8206H    LD      (INTE0AD), 06H     ; Sets INT0 interrupt level to 6.
8209H    EI      5                  ; Sets CPU interrupt level to 5.
820BH    LD      (SYSCR2), 28H      ; Sets HALT mode to IDLE1 mode.
820EH    HALT                       ; Halts CPU.

  INT0                              INT0 interrupt routine

                                    RETI
820FH    LD      XX, XX
```



(3) Operation

a. IDLE2 mode

In IDLE2 mode only specific internal I/O operations, as designated by the IDLE2 setting register, can take place. Instruction execution by the CPU stops.

Figure 3.4.5 illustrates an example of the timing for clearance of the IDLE2 mode halt state by an interrupt.

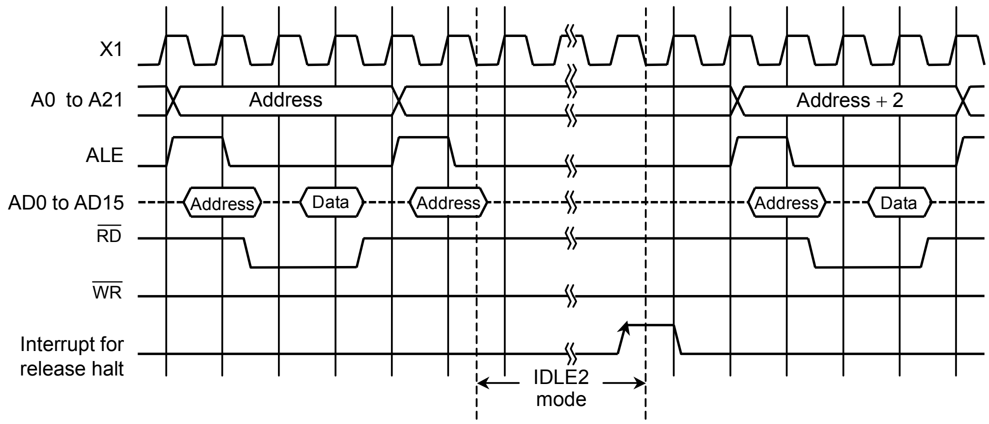


Figure 3.4.5 Timing Chart for IDLE2 Mode Halt State Cleared by Interrupt

b. IDLE1 mode

In IDLE1 mode, only the internal oscillator and the RTC continue to operate. The system clock in the MCU stops.

In the halt state, the interrupt request is sampled asynchronously with the system clock; however, clearance of the Halt state (e.g., restart of operation) is synchronous with it.

Figure 3.4.6 illustrates the timing for clearance of the IDLE1 mode halt state by an interrupt.

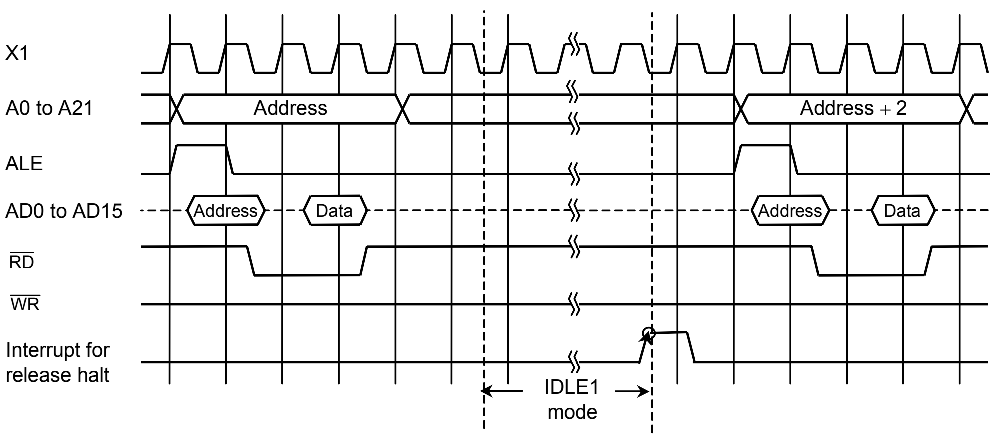


Figure 3.4.6 Timing Chart for IDLE1 Mode Halt State Cleared by Interrupt

c. STOP mode

When STOP mode is selected, all internal circuits stop, including the internal oscillator. Pin status in STOP mode depends on the settings in the SYSCR2<DRVE> register. Table 3.4.6 and Table 3.4.7 summarizes the state of these pins in STOP mode.

After STOP mode has been cleared, system clock output starts when the warm-up time has elapsed, in order to allow oscillation to stabilize. After STOP mode has been cleared, either NORMAL mode or SLOW mode can be selected using the SYSCR0<RSYSCK> register. Therefore, <RSYSCK>, <RXEN> and <RXTEN> must be set. See the sample warm-up times in Table 3.4.5.

Figure 3.4.7 illustrates the timing for clearance of the STOP mode halt state by an interrupt.

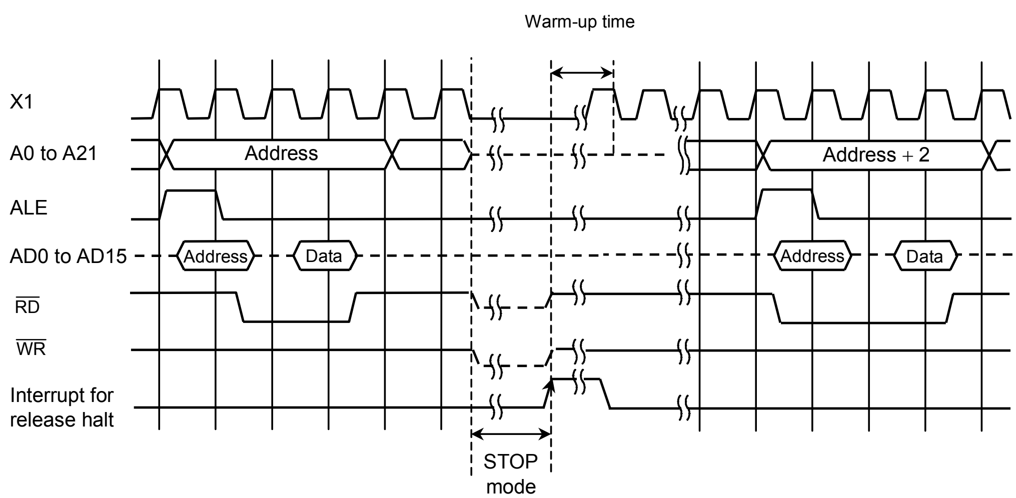


Figure 3.4.7 Timing Chart for STOP Mode Halt State Cleared by Interrupt

Table 3.4.5 Sample Warm-up Times after Clearance of STOP Mode

@ f_{OSCH} = 16 MHz, fs = 32.768 kHz

SYSCR0 <RSYSCK>	SYSCR2<WUPTM1:0>		
	01 (2^8)	10 (2^{14})	11 (2^{16})
0 (fc)	16 μs	1.024 ms	4.096 ms
1 (fs)	7.8 ms	500 ms	2000 ms

Example:

- The STOP mode is entered when the low-frequency operates, and high-frequency operates after releasing due to NMI.

```
Address
              SYSCR0    EQU     00E0H
              SYSCR1    EQU     00E1H
              SYSCR2    EQU     00E2H
8FFDH                   LD      (SYSCR1), 08H           ; fSYS = fs/2
9000H                   LD      (SYSCR2), X–1001X1B     ; Sets warm-up time to 2^14/fc
9002H                   LD      (SYSCR0), –11000 – – B  ; Operates high frequency after released.
9005H                   HALT
                                                        Clears and starts warm-up timer
              NMI pin input  ------------------------>  (High frequency)
                                                                 END
                                                          NMI interrupt routine
9006H                   LD     XX, XX                            RETI
```

–: No change

Note: When different modes are used before and after STOP mode as the above mentioned, there is possible to release the HALT mode without changing the operation mode by acceptance of the halt release interrupt request during execution of “HALT” instruction (during 6 state). In the system which accepts the interrupts during execution “HALT” instruction, set the same operation mode before and after the STOP mode.

Table 3.4.6 Input Buffer State Table

<table>
<tr><th rowspan="4">Port Name</th><th rowspan="4">Input Function Name</th><th colspan="9">Input Buffer State</th></tr>
<tr><th rowspan="3">During Reset</th><th colspan="2" rowspan="2">When the CPU is Operating</th><th colspan="2" rowspan="2">In HALT mode (IDLE2/IDLE1)</th><th colspan="4">In HALT mode（STOP）</th></tr>
<tr><th colspan="2"><DRVE>=1</th><th colspan="2"><DRVE>=0</th></tr>
<tr><th>When Used as Function Pin</th><th>When Used as Input Port</th><th>When Used as Function Pin</th><th>When Used as Input Port</th><th>When Used as Function Pin</th><th>When Used as Input Port</th><th>When Used as Function Pin</th><th>When Used as Input Port</th></tr>
<tr><td>P00-07</td><td>AD0-7</td><td rowspan="4">OFF</td><td rowspan="2">ON upon external read</td><td rowspan="5">ON</td><td rowspan="2">OFF</td><td rowspan="4">OFF</td><td rowspan="2">OFF</td><td rowspan="4">OFF</td><td rowspan="2">OFF</td><td rowspan="11">OFF</td></tr>
<tr><td>P10-17</td><td>AD8-15</td></tr>
<tr><td>P20-25</td><td>-</td><td rowspan="4">-</td><td rowspan="4">-</td><td rowspan="4">-</td><td rowspan="4">-</td></tr>
<tr><td>P32(*1)</td><td>-</td></tr>
<tr><td>P40-42(*1)</td><td>-</td><td>ON</td><td>ON</td><td>ON</td></tr>
<tr><td>P50-52(*2)</td><td>-</td><td rowspan="2">OFF</td><td rowspan="2">ON upon port read</td><td rowspan="2">OFF</td><td rowspan="2">OFF</td></tr>
<tr><td>P53(*2)</td><td>/ADTRG</td><td rowspan="7">ON</td><td rowspan="7">ON</td><td rowspan="7">ON</td><td>ON</td></tr>
<tr><td>P60</td><td>SCK</td><td rowspan="24">ON</td><td rowspan="24">ON</td><td rowspan="24">ON</td><td rowspan="24">ON</td><td rowspan="4">OFF</td></tr>
<tr><td>P61</td><td>SDA</td></tr>
<tr><td rowspan="2">P62</td><td>SI</td></tr>
<tr><td>SCL</td></tr>
<tr><td>P63</td><td>INT0</td><td>ON</td><td>ON</td></tr>
<tr><td>P70</td><td>TA0IN</td><td>OFF</td><td rowspan="22">OFF</td></tr>
<tr><td>P71</td><td>-</td><td rowspan="2">-</td><td rowspan="2">-</td><td rowspan="2">-</td><td rowspan="2">-</td></tr>
<tr><td>P72</td><td>-</td></tr>
<tr><td>P73</td><td>TA4IN</td><td>ON</td><td>ON</td><td>ON</td><td>OFF</td></tr>
<tr><td>P74</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td rowspan="2">P80</td><td>TB0UN0</td><td rowspan="4">ON</td><td rowspan="4">ON</td><td rowspan="4">ON</td><td rowspan="4">OFF</td></tr>
<tr><td>INT5</td></tr>
<tr><td rowspan="2">P81</td><td>TB0IN1</td></tr>
<tr><td>INT6</td></tr>
<tr><td>P82</td><td>-</td><td rowspan="3">-</td><td rowspan="3">-</td><td rowspan="3">-</td><td rowspan="3">-</td></tr>
<tr><td>P83</td><td>-</td></tr>
<tr><td>P90</td><td>-</td></tr>
<tr><td>P91</td><td>RXD0</td><td rowspan="3">ON</td><td rowspan="3">ON</td><td rowspan="3">ON</td><td rowspan="3">OFF</td></tr>
<tr><td rowspan="2">P92</td><td>SCLK0</td></tr>
<tr><td>/CTS0</td></tr>
<tr><td>P93</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>P94</td><td>RXD1</td><td rowspan="4">ON</td><td rowspan="4">ON</td><td rowspan="3">ON</td><td rowspan="5">OFF</td></tr>
<tr><td rowspan="2">P95</td><td>SCLK1</td></tr>
<tr><td>/CTS1</td></tr>
<tr><td rowspan="2">P96</td><td>XT1 For oscillator</td><td>OFF</td><td>OFF</td><td>OFF</td><td rowspan="2">OFF</td><td>OFF</td></tr>
<tr><td>XT1 For port</td><td rowspan="6">ON</td><td>OFF</td><td rowspan="2">ON</td><td>OFF</td><td rowspan="2">ON</td><td rowspan="2">ON</td></tr>
<tr><td>P97</td><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td></tr>
<tr><td>/NMI</td><td>-</td><td rowspan="4">ON</td><td rowspan="4">-</td><td rowspan="4">ON</td><td rowspan="4">-</td><td rowspan="3">ON</td><td rowspan="4">-</td><td rowspan="3">ON</td><td rowspan="4">-</td></tr>
<tr><td>/RESET</td><td>-</td></tr>
<tr><td>AM0,AM1</td><td>-</td></tr>
<tr><td>X1</td><td>-</td><td>OFF</td><td>OFF</td></tr>
</table>

ON: The buffer is always turned on. A current flows the input buffer if the input pin is not driven.

OFF :The buffer is always turned off.

-: No applicable

*1:Port having a pull-up/pull-down resistor.

*2:AIN input does not cause a current to flow through the buffer.

Table 3.4.7 Output buffer State Table

Port Name	Output Function Name	During Reset	When the CPU is Operating: When Used as Function Pin	When the CPU is Operating: When Used as Output Port	In HALT mode (IDLE2/IDLE1): When Used as Function Pin	In HALT mode (IDLE2/IDLE1): When Used as Output Port	In HALT mode (STOP) <DRVE>=1: When Used as Function Pin	In HALT mode (STOP) <DRVE>=1: When Used as Output Port	In HALT mode (STOP) <DRVE>=0: When Used as Function Pin	In HALT mode (STOP) <DRVE>=0: When Used as Output Port
P00-07	AD0-7		ON upon external write		OFF		OFF			
P10-17	AD8-15	OFF								
	A8-15									
P20-25	A0-5									
	A16-21									
P30	/RD	ON								
P31	/WR								OFF	
P32(*1)	/HWR		ON		ON		ON			
P40-42(*1)	/CS0									
	/CS1									
	/CS2									
P60	SCK									
P61	SDA									
	SO									
P62	SCL									
P63	-		-	ON	-	ON	-	ON	-	
P70	-									OFF
P71	TA1OUT		ON		ON		ON		OFF	
P72	TA3OUT	OFF								
P73	-		-		-		-		-	
P74	TA5OUT		ON		ON		ON		OFF	
P80	-		-		-		-		-	
P81	-									
P82	TB0OUT0		ON		ON		ON		OFF	
P83	TB0OUT1									
P90	TXD0									
P91	-		-		-		-		-	
P92	SCLK0		ON		ON		ON		OFF	
P93	TSD1									
P94	-		-		-		-		-	
P95	SCLK1		ON		ON		ON		OFF	
P96	-	ON	-		-		-		-	
P97	XT2 For oscillator	OFF	ON	OFF	ON	OFF	OFF	OFF	OFF	
	XT2 For port	ON	OFF	ON	OFF	ON		ON		
ALE	-	OFF	ON	-	ON	-	ON	-	ON	-
X2	-	ON								

ON: The buffer is always turned on.

OFF: The buffer is always turned off.

-:No applicable

*1 : Port having a pull-up/pull-down resistor.

3.5 Interrupts

Interrupts are controlled by the CPU interrupt mask register SR<IFF2:0> and by the built-in interrupt controller.

The TMP91CP27 has a total of 34 interrupts divided into the following three types:

- Interrupts generated by CPU: 9 sources
  (Software interrupts, illegal instruction interrupt)
- Internal interrupts: 21 sources
- Interrupts on external pins ($\overline{\text{NMI}}$, INT0, INT5 and INT6): 4 sources

A (fixed) individual interrupt vector number is assigned to each interrupt.

One of six (Variable) priority level can be assigned to each maskable interrupt.

The priority level of non-maskable interrupts are fixed at 7 as the highest level.

When an interrupt is generated, the interrupt controller sends the piority of that interrupt to the CPU. If multiple interrupts are generated simultaneously, the interrupt controller sends the interrupt with the highest priority to the CPU. (The highest priority is level 7 using for non-maskable interrupts.)

The CPU compares the priority level of the interrupt with the value of the CPU interrupt mask register <IFF2:0>. If the priority level of the interrupt is higher than the value of the interrupt mask register, the CPU accepts the interrupt.

The interrupt mask register <IFF2:0> value can be updated using the value of the EI instruction ("EI num" sets <IFF2:0> data to num).

For example, specifying "EI3" enables the maskable interrupts which priority level set in the interrupt controller is 3 or higher, and also non-maskable interrupts.

Operationally, the DI instruction (<IFF2:0> = "7") is identical to the "EI7" instruction. DI instruction is used to disable maskable interrupts because of the priority level of maskable interrupts is 0 to 6. The EI instruction is vaild immediately after execution.

In addition to the above general-purpose interrupt processing mode, TLCS-900/L1 has a micro DMA interrupt processing mode as well. The CPU can transfer the data (1/2/4 bytes) automatically in micro DMA mode, therefore this mode is used for speed-up interrupt processing, such as transferring data to the internal or external peripheral I/O. Moreover, TMP91CP27 has software start function for micro DMA processing request by the software not by the hardware interrupt.

Figure 3.5.1 shows the overall interrupt processing flow.

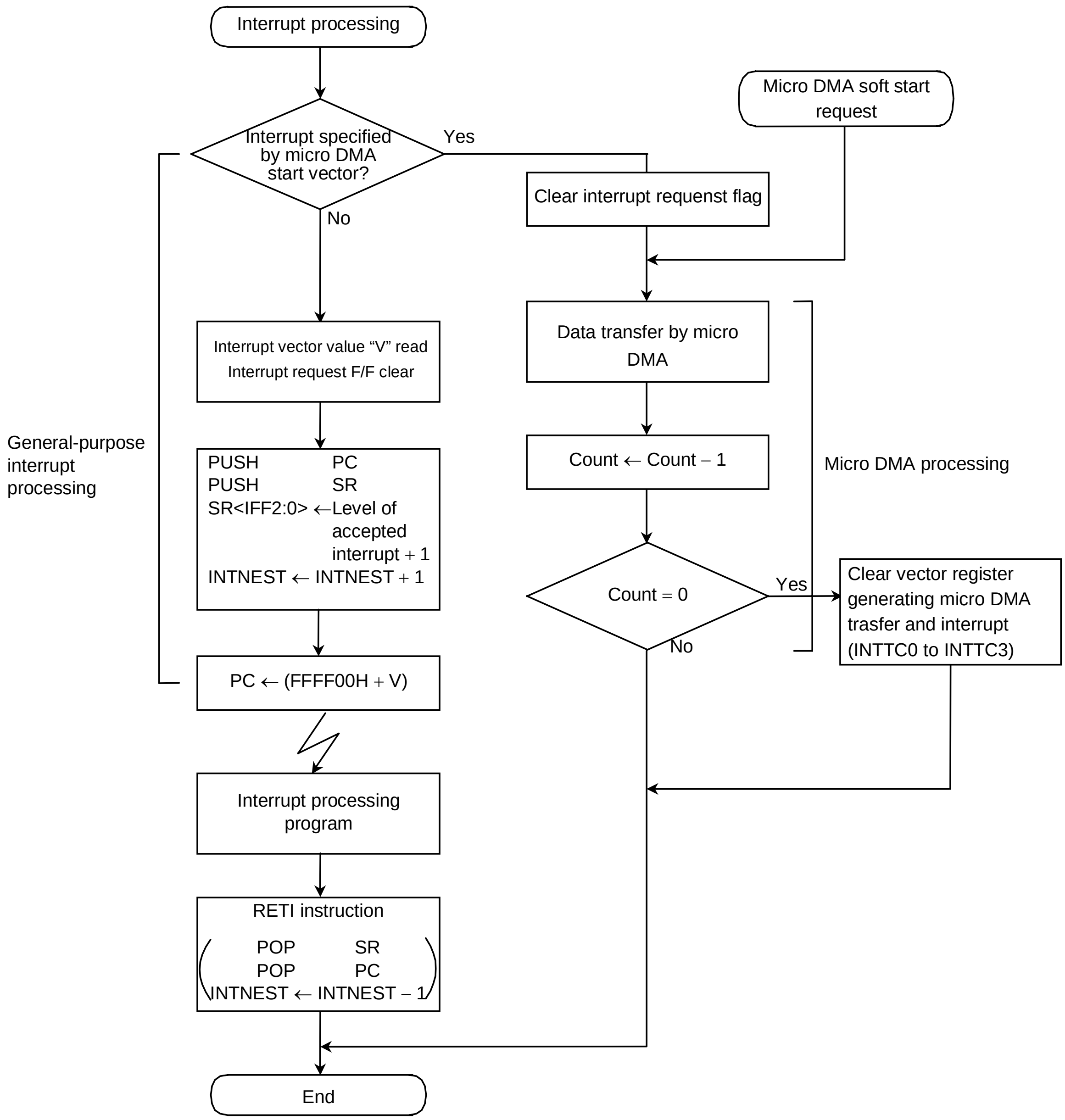


Figure 3.5.1 Overall Interrupt Processing Flow

3.5.1 General-Purpose Interrupt Processing

When the CPU accepts an interrupt, it usually performs the following sequence of operations. That is also the same as TLCS-900/L and TLCS-900/H.

(1) The CPU reads the interrupt vector from the interrupt controller.
If the same level interrupts occur simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt request.
(The default priority is already fixed for each interrupt: The smaller vector value has the higher priority level.)

(2) The CPU pushes the value of program counter (PC) and status register (SR) onto the stack area (indicated by XSP).

(3) The CPU sets the value which is the priority level of the accepted interrupt plus 1 (+1) to the interrupt mask register <IFF2:0>. However, if the priority level of the accepted interrupt is 7, the register's value is set to 7.

(4) The CPU increases the interrupt nesting counter INTNEST by 1 (+1).

(5) The CPU jumps to the address indicated by the data at address "FFFF00H + interrupt vector" and starts the interrupt processing routine.
The above processing time is 18-states (2.25 μs at 16 MHz) as the best case (16 bits data bus width and 0 waits).

When the CPU completd the interrupt processing, use the RETI instruction to return to the main routine. RETI restores the contents of program counter (PC) and status register (SR) from the stack and decreases the interrupt nesting counter INTNEST by 1 (–1).

Non-maskable interrupts cannot be disabled by a user program. Maskable interrupts, however, can be enabled or disabled by a user program. A program can set the priority level for each interrupt source.

If an interrupt request which has a priority level equal to or greater than the value of the CPU interrupt mask register <IFF2:0> comes out, the CPU accepts its interrupt. Then, the CPU interrupt mask register <IFF2:0> is set to the value of the priority level for the accepted interrupt plus 1 (+1).

Therefore, if an interrupt is generated with a higher level than the current interrupt during its processing, the CPU accepts the later interrupt and goes to the nesting status of interrupt processing.

Moreover, if the CPU receives another interrupt request while performing the said (1) to (5) processing steps of the current interrupt, the latest interrupt request is sampled immediately after execution of the first instruction of the current interrupt processing routine. Specifying DI as the start instruction disables maskable interrupt nesting.

A Reset initializes the interrupt mask register <IFF2:0> to "7", disabling all maskable interrupts.

Figure 3.5.1 shows the TMP91CP27 interrupt vectors and micro DMA start vectors. The address FFFF00H to FFFFFFH (256 bytes) is assigned for the interrupt vector area.

Table 3.5.1 TMP91CP27 Interrupt Vectors and Micro DMA Start Vectors

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value (V)	Vector Reference Address	Micro DMA Start Vector
1	Non-maskable	"RESET" or SWI0 instruction	0000H	FFFF00H	–
2		SWI1 instruction	0004H	FFFF04H	–
3		INTUNDEF: Illegal Instruction or SWI2 instruction	0008H	FFFF08H	–
4		SWI3 instruction	000CH	FFFF0CH	–
5		SWI4 instruction	0010H	FFFF10H	–
6		SWI5 instruction	0014H	FFFF14H	–
7		SWI6 instruction	0018H	FFFF18H	–
8		SWI7 instruction	001CH	FFFF1CH	–
9		$\overline{\text{NMI}}$ pin	0020H	FFFF20H	–
10		INTWD: Watchdog timer	0024H	FFFF24H	–
–	Maskable	(Micro DMA)	–	–	–
11		INT0 pin	0028H	FFFF28H	0AH
12		Reserved	–	–	–
13		Reserved	–	–	–
14		Reserved	–	–	–
15		Reserved	–	–	–
16		INT5 pin	003CH	FFFF3CH	0FH
17		INT6 pin	0040H	FFFF40H	10H
18		Reserved	–	–	–
19		Reserved	–	–	–
20		INTTA0: 8-bit timer 0	004CH	FFFF4CH	13H
21		INTTA1: 8-bit timer 1	0050H	FFFF50H	14H
22		INTTA2: 8-bit timer 2	0054H	FFFF54H	15H
23		INTTA3: 8-bit timer 3	0058H	FFFF58H	16H
24		INTTA4: 8-bit timer 4	005CH	FFFF5CH	17H
25		INTTA5: 8-bit timer 5	0060H	FFFF60H	18H
26		Reserved	–	–	–
27		Reserved	–	–	–
28		INTTB00: 16-bit timer 0 (TB0RG0)	006CH	FFFF6CH	1BH
29		INTTB01: 16-bit timer 0 (TB0RG1)	0070H	FFFF70H	1CH
30		Reserved	–	–	–
31		Reserved	–	–	–
32		INTTBOF0: 16-bit timer 0 (Over flow)	007CH	FFFF7CH	1FH
33		Reserved	–	–	–
34		INTRX0: Serial reception (Channel 0)	0084H	FFFF84H	21H
35		INTTX0: Serial transmission (Channel 0)	0088H	FFFF88H	22H
36		INTRX1: Serial reception (Channel 1)	008CH	FFFF8CH	23H
37		INTTX1: Serial transmission (Channel 1)	0090H	FFFF90H	24H
38		INTSBI: Serial bus interface interrupt	0094H	FFFF94H	25H
39		INTRTC: Interrupt for RTC	0098H	FFFF98H	26H
40		INTAD: AD conversion end	009CH	FFFF9CH	27H
41		INTTC0: End of Micro DMA (Channel 0)	00A0H	FFFFA0H	–
42		INTTC1: End of Micro DMA (Channel 1)	00A4H	FFFFA4H	–
43		INTTC2: End of Micro DMA (Channel 2)	00A8H	FFFFA8H	–
44		INTTC3: End of Micro DMA (Channel 3)	00ACH	FFFFACH	–
		(Reserved) : (Reserved)	00B0H : 00FCH	FFFFB0H : FFFFFCH	– : –

3.5.2 Micro DMA

In addition to general-purpose interrupt processing, the TMP91CP27 supports a micro DMA function. Interrupt requests set by micro DMA perform micro DMA processing at the highest priority level among maskable interrupts, regardless of the priority level of the particular interrupt source. The micro DMA has 4 channels and is possible continuous transmission by specifying the say later burst mode.

Because the micro DMA function has been implemented with the cooperative operation of CPU, when CPU goes to a standby mode by HALT instruction, the requirement of micro DMA will be ignored (Pending).

(1) Micro DMA operation

When an interrupt request specified by the micro DMA start vector register is generated, the micro DMA triggers a micro DMA request to the CPU at interrupt priority highest level and starts processing the request in spite of any interrupt source's level. The micro DMA is ignored on <IFF2:0> = "7".

The 4 micro DMA channels allow micro DMA processing to be set for up to 4 types of interrupts at any one time. When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared.

The data are automatically transferred once (1/2/4 bytes) from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decreased by 1 (−1).

If the decreased result is "0", the micro DMA transfer end interrupt (INTTCn) passes from the CPU to the interrupt controller. In addition, the micro DMA start vector register DMAnV is cleared to 0, the next micro DMA is disabled and micro DMA processing completes.

If the decreased result is other than "0", the micro DMA processing completes if it isn't specified the say later burst mode. In this case, the micro DMA transfer end interrupt (INTTCn) aren't generated.

If an interrupt request is triggered for the interrupt source in use during the interval between the clearing of the micro DMA start vector and the next setting, general-purpose interrupt processing executes at the interrupt level set. Therefore, if only using the interrupt for starting the micro DMA (Not using the interrupts as a general-purpose interrupt: Level 1 to 6), first set the interrupts level to 0 (Interrupt requests disabled).

If using micro DMA and general-purpose interrupts together, first set the level of the interrupt used to start micro DMA processing lower than all the other interrupt levels. In this case, the cause of general interrupt is limited to the edge interrupt.

The priority of the micro DMA transfer end interrupt is defined by the interrupt level and the default priority as the same as the other maskable interrupt.

If a micro DMA request is set for more than one channel at the same time, the priority is not based on the interrupt priority level but on the channel number. The smaller channel number has the higher priority (Channel 0 (High) > channel 3 (Low)).

While the register for setting the transfer source/transfer destination addresses is a 32-bit control register, this register can only effectively output 24-bit addresses. Accordingly, micro DMA can access 16 Mbytes (The upper eight bits of the 32 bits are not valid).

Three micro DMA transfer modes are supported: 1-byte transfer, 2-byte transfer, and 4-byte transfer. After a transfer in any mode, the transfer source/destination addresses are increased, decreased, or remain unchanged.

This simplifies the transfer of data from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the transfer modes, see 3.5.2 (4) "Detailed description of the transfer mode register: DMAM0 to DMAM3".

As the transfer counter is a 16-bit counter, micro DMA processing can be set for up to 65536 times per interrupt source. (The micro DMA processing count is maximized when the transfer counter initial value is set to 0000H.)

Micro DMA processing can be started by the 19 interrupts shown in the micro DMA start vectors of Figure 3.5.1 and by the micro DMA soft start, making a total of 20 interrupts.

Figure 3.5.2 shows the word transfer micro DMA cycle in transfer destination address INC mode (except for Counter mode, the same as for other modes).

(The conditions for this cycle are based on an external 16-bit bus, 0 waits, transfer source/transfer destination addresses both even-numbered values).

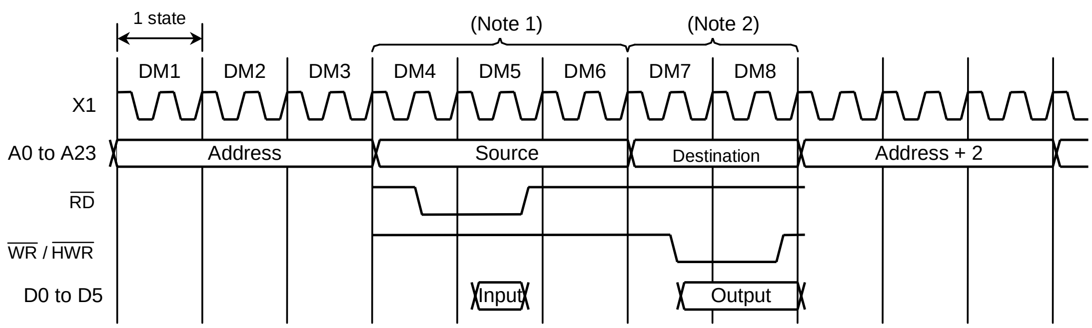


Figure 3.5.2 Timing for Micro DMA Cycle (Word transfer)

States 1 to 3: Instruction fetch cycle (gets next address code).
If 3 bytes and more instruction codes are inserted in the instruction queue buffer, this cycle becomes a dummy cycle.
States 4 to 5: Micro DMA read cycles
State 6: Dummy cycle (The address bus remains unchanged from state 5)
States 7 to 8: Micro DMA write cycle

Note 1: If the source address area is an 8-bit bus, it is increased by two states.
If the source address area is a 16-bit bus and the address starts from an odd number, it is increased by two states.

Note 2: If the destination address area is an 8-bit bus, it is increased by two states.
If the destination address area is a 16-bit bus and the address starts from an odd number, it is increased by two states.

(2) Soft start function

In addition to starting the micro DMA function by interrupts, TMP91CP27 includes a micro DMA software start function that starts micro DMA on the generation of the write cycle to the DMAR register.

Writing "1" to each bit of DMAR register causes micro DMA once. At the end of transfer, the corresponding bit of the DMAR register is automatically cleared to "0".

Only one-channel can be set once for micro DMA. (Do not write "1" to plural bits.)

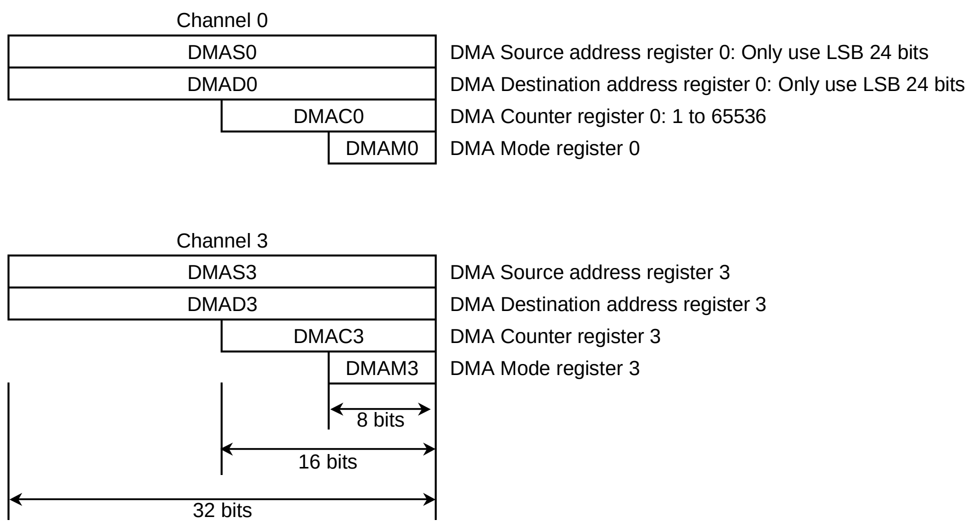
When writing again "1" to the DMAR register, check whether the bit is "0" before writing "1".

When a burst is specified by DMAB register, data is continuously transferred until the value in the micro DMA transfer counter is "0" after start up of the micro DMA.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA software request register	89H					DMA request			
							DMAR3	DMAR2	DMAR1	DMAR0
							R/W			
							0	0	0	0

(3) Transfer control registers

The transfer source address and the transfer destination address are set in the following registers in CPU. An instruction of the form "LDC cr,r" can be used to set these registers.



(4) Detailed description of the transfer mode register: DMAM0 to DMAM3

(DMAM0 to DMAM3)

0	0	0	Mode				

Note: The upper three bits of data programmed to these registers must always be 0.

ZZ: 0 = Byte transfer, 1 = Word transfer, 2 = 4-byte transfer, 3 = Reserved

Mode	Description	Execution time
0 0 0 Z Z	Transfer destination address INC mode.................... I/O to memory (DMADn+) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (1000 ns) @ byte/word transfer 12 states (1500 ns) @4-byte transfer
0 0 1 Z Z	Transfer destination address DEC mode .... I/O to memory (DMADn–) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (1000 ns) @ byte/word transfer 12 states (1500 ns) @4-byte transfer
0 1 0 Z Z	Transfer source address INC mode .... Memory to I/O (DMADn) ← (DMASn+) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (1000 ns) @ byte/word transfer 12 states (1500 ns) @4-byte transfer
0 1 1 Z Z	Transfer source address DEC mode......................... Memory to I/O (DMADn) ← (DMASn–) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (1000 ns) @ byte/word transfer 12 states (1500 ns) @4-byte transfer
1 0 0 Z Z	Address fixed mode.. I/O to I/O (DMADn) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (1000 ns) @ byte/word transfer 12 states (1500 ns) @4-byte transfer
1 0 1 0 0	Counter mode ･･for counting number of times interrupt is generated DMASn ← DMASn + 1 DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	5 states (625 ns)

Note 1: “n” is the corresponding micro DMA channels 0 to 3
DMADn+/DMASn+: Post increment (Increment register value after transfer)
DMADn–/DMASn–: Post decrement (Decrement register value after transfer)
The I/Os in the table mean fixed address and the memory means increment (INC) or decrement (DEC) addresses.

Note 2: Execution time is under the condition of:
16-bit bus width/0 waits.
fc = 16MHz/selected high frequency mode (fc × 1)

Note 3: Do not use an undefined code for the transfer mode register except for the defined codes listed in the above table.

3.5.3 Interrupt Controller Operation

The block diagram in Figure 3.5.3 shows the interrupt circuits. The left-hand side of the diagram shows the interrupt controller circuit. The right-hand side shows the CPU interrupt request signal circuit and the halt release circuit.

For each of the 25 interrupt channels there is an interrupt request flag (Consisting of a flip-flop), an interrupt priority setting register and a micro DMA start vector register. The interrupt request flag latches interrupt requests from the peripherals. The flag is cleared to zero in the following cases:

- When reset occurs
- When the CPU reads the channel vector after accepted its interrupt
- When executing an instruction that clears the interrupt (Program DMA start vector to INTCLR register)
- When the CPU receives a micro DMA request (When micro DMA is set.)
- When the micro DMA burst transfer is terminated

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g., INTE0AD or INTE56). 6 interrupt priorities levels (1 to 6) are provided. Setting an interrupt source's priority level to 0 (or 7) disables interrupt requests from that source. The priority of non-maskable interrupts ($\overline{\text{NMI}}$ pin interrupts and watchdog timer interrupts) is fixed at 7. If interrupt request with the same level are generated at the same time, the default priority (The interrupt with the lowest priority or, in other words, the interrupt with the lowest vector value) is used to determine which interrupt request is accepted first.

The 3rd and 7th bits of the interrupt priority setting register indicate the state of the interrupt request flag and thus whether an interrupt request for a given channel has occurred.

The interrupt controller sends the interrupt request with the highest priority among the simultaneous interrupts and its vector address to the CPU. The CPU compares the priority value <IFF2:0> in the status register by the interrupt request signal with the priority value set; if the latter is higher, the interrupt is accepted. Then the CPU sets a value higher than the priority value by 1 (+1) in the CPU SR<IFF2:0>. Interrupt request where the priority value equals or is higher than the set value are accepted simultaneously during the previous interrupt routine.

When interrupt processing is completed (after execution of the RETI instruction), the CPU restores the priority value saved in the stack before the interrupt was generated to the CPU SR<IFF2:0>.

The interrupt controller also has registers (4 channels) used to store the micro DMA start vector. Writing the start vector of the interrupt source for the micro DMA processing (See Table 3.5.1), enables the corresponding interrupt to be processed by micro DMA processing. The values must be set in the micro DMA parameter register (e.g., DMAS and DMAD) prior to the micro DMA processing.

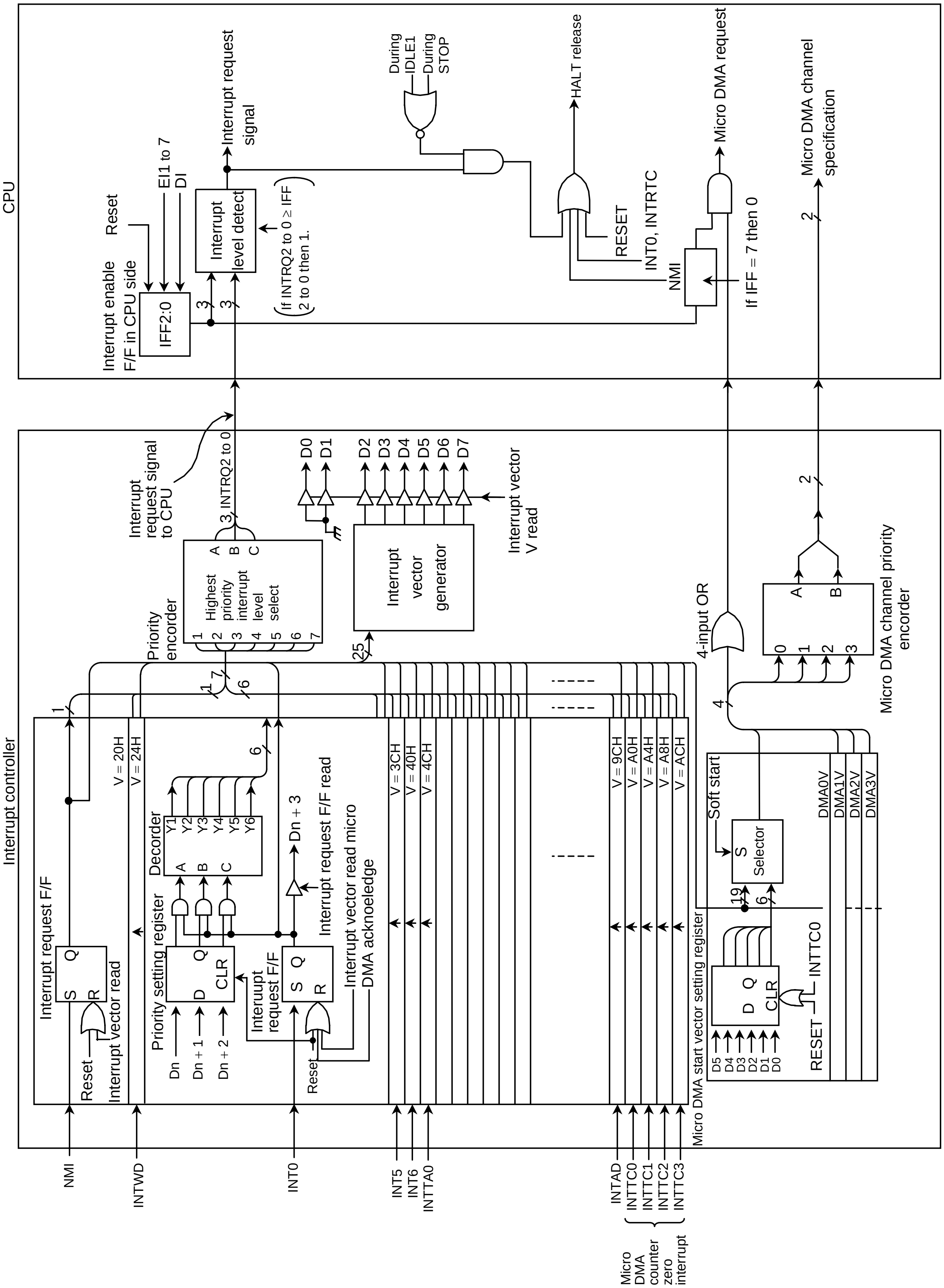


Figure 3.5.3 Block Diagram of Interrupt Controller

(1) Interrupt level setting registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	INT0 & INTAD enable	90h	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	93h	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	95h	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA23	INTTA2 & INTTA3 enable	96h	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA45	INTTA4 & INTTA5 enable	97h	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disable interrupt request
0	0	1	Setting interrupt priority level to “1”.
0	1	0	Setting interrupt priority level to “2”.
0	1	1	Setting interrupt priority level to “3”.
1	0	0	Setting interrupt priority level to “4”.
1	0	1	Setting interrupt priority level to “5”.
1	1	0	Setting interrupt priority level to “6”.
1	1	1	Disable interrupt request

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB0	Interrupt enable TMRB0	99H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB01V	Interrupt enable TMRB0 (over flow)	9BH					INTTBOF0 (TMRB0 over flow)			
							ITF0C	ITF0M2	ITF0M1	ITF0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES0	Interrupt enable Serial 0	9CH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES1	Interrupt enable serial 1	9DH	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES2RTC	Interrupt enable SBI/RTC	9EH	INTRTC				INTSBI			
			IRTCC	IRTCM2	IRTCM1	IRTCM0	IS2C	IS2M2	IS2M1	IS2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC01	INTTC0 & INTTC1 enable	A0H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23	INTTC2 & INTTC3 enable	A1H	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disable interrupt request
0	0	1	Setting interrupt priority level to “1”.
0	1	0	Setting interrupt priority level to “2”.
0	1	1	Setting interrupt priority level to “3”.
1	0	0	Setting interrupt priority level to “4”.
1	0	1	Setting interrupt priority level to “5”.
1	1	0	Setting interrupt priority level to “6”.
1	1	1	Disable interrupt request

(2) External interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt input mode control	8CH (Prohibit RMW)	–	–	–	–	–	I0EDGE	I0LE	NMIREE
			W							
			0	0	0	0	0	0	0	0
			Write “0”.					INT0 edge 0: Rising 1: Falling	INT0 mode 0: Edge 1: Level	1: Operate even on rising/ falling edge of $\overline{\text{NMI}}$

INT0 level enable

0	edge detect interrupt
1	“H” level interrupt

$\overline{\text{NMI}}$ rising edge enable

0	Interrupt request generation at falling edge
1	Interrupt request generation at rising/falling edge

(3) Interrupt request flag clear register

The interrupt request flag is cleared by writing the appropriate micro DMA start vector, as given in Table 3.5.1, to the register INTCLR.

For example, to clear the interrupt flag INT0, perform the following register operation after execution of the DI instruction.

INTCLR ← 0AH   Clears interrupt request flag INT0

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTCLR	Interrupt clear control	88H (Prohibit RMW)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
					W					
					0	0	0	0	0	0
					Interrupt vector					

(4) Micro DMA start vector registers

This register assigns micro DMA processing to which interrupt source. The interrupt source with a micro DMA start vector that matches the vector set in this register is assigned as the micro DMA start source.

When the micro DMA transfer counter value reaches zero, the micro DMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA start vector register is cleared, and the micro DMA start source for the channel is cleared. Therefore, to continue micro DMA processing, set the micro DMA start vector register again during the processing of the micro DMA transfer end interrupt.

If the same vector is set in the micro DMA start vector registers of more than one Accordingly, if the same vector is set in the micro DMA start vector registers of two channels, the interrupt generated in the channel with the lower number is executed until micro DMA transfer is complete. If the micro DMA start vector for this channel is not set again, the next micro DMA is started for the channel with the higher number. (Micro DMA chaining)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	80H			DMA0 start vector					
					DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
DMA1V	DMA1 start vector	81H			DMA1 start vector					
					DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
DMA2V	DMA2 start vector	82H			DMA2 start vector					
					DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
DMA3V	DMA3 start vector	83H			DMA3 start vector					
					DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0

(5) Micro DMA burst specification

Specifying the micro DMA burst continues the micro DMA transfer until the transfer counter register reaches zero after micro DMA start. Setting a bit which corresponds to the micro DMA channel of the DMAB registers mentioned below to “1” specifies a burst.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA software request register	89H					DMAR3	DMAR2	DMAR1	DMAR0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA software request			
DMAB	DMA burst request register	8AH					DMAB3	DMAB2	DMAB1	DMAB0
							R/W			
							0	0	0	0
							1: DMA request on Burst Mode			

(6) Attention point

The instruction execution unit and the bus interface unit of this CPU operate independently. Therefore, immediately before an interrupt is generated, if the CPU fetches an instruction that clears the corresponding interrupt request flag, the CPU may execute the instruction that clears the interrupt request flag (*1) between accepting and reading the interrupt vector. In this case, the CPU reads the default vector 0008H and reads the interrupt vector address FFFF08H.

To avoid the above program, place instructions that clear interrupt request flags after a DI instruction. And in the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing and more than 1 instructions (Example: "NOP" * 1 times). If placed EI instruction without waiting NOP instruction after execution of clearing instruction, interrupt will be enable before request flag is cleared.

In the case of changing the value of the interrupt mask register <IFF2:0> by execution of POP SR instruction, disable an interrupt by DI instruction before execution of POP SR instruction.

In addition, take care as the following 2 circuits are exceptional and demand special attention.

INT0 level mode	In level mode INT0 is not an edge-triggered interrupt. Hence, in level mode the interrupt request flip-flop for INT0 does not function. The peripheral interrupt request passes through the S input of the flip-flop and becomes the Q output. If the interrupt input mode is changed from edge mode to level mode, the interrupt request flag is cleared automatically.

	If the CPU enters the interrupt response sequence as a result of INT0 going from 0 to 1, INT0 must then be held at 1 until the interrupt response sequence has been completed. If INT0 is set to level mode so as to release a halt state, INT0 must be held at 1 from the time INT0 changes from 0 to 1 until the halt state is released. (Hence, it is necessary to ensure that input noise is not interpreted as a 0, causing INT0 to revert to 0 before the halt state has been released.)
	When the mode changes from level mode to edge mode, interrupt request flags which were set in level mode will not be cleared. Interrupt request flags must be cleared using the following sequence.
	DI
	LD (IIMC), 00H ; Switches interrupt input mode from level mode to edge mode.
	LD (INTCLR), 0AH ; Clears interrupt request flag
	NOP ; Wait EI instruction
	EI

INTRX	The interrupt request flip-flop can only be cleared by a reset or by reading the serial channel receive buffer. It cannot be cleared by an instruction.

Note: The following instructions or pin input state changes are equivalent to instructions that clear the interrupt request flag.

INT0: Instructions which switch to Level Mode after an interrupt request has been generated in edge mode.
The pin input change from high to low after interrupt request has been generated
in Level Mode. (H → L)

INTRX: Instruction which read the receive buffer

3.6 Port Function

The TMP91CP27 features 53 bit settings which relate to the various I/O ports.

As well as general-purpose I/O port functionality, the port pins also have I/O functions which relate to the built-in CPU and internal I/Os. Table 3.6.1 lists the functions of each port pin. Table 3.6.2 to Table 3.6.3 lists I/O registers and their specifications.

Table 3.6.1 Port Function

(R: ↑ = with pull-up resistor)

Port Names	Pin Names	Number of Pins	Direction	R	Direction Setting Unit	Pin Names for Built-in Functions
Port 0	P00 to P07	8	I/O	–	Bit	AD0 to AD7
Port 1	P10 to P17	8	I/O	–	Bit	AD8 to AD15/A8 to A15
Port 2	P20 to P25	6	I/O	–	Bit	A16 to A21/A0 to A5
Port 3	P30	1	Output	–	Bit	$\overline{\text{RD}}$
	P31	1	Output	–	Bit	$\overline{\text{WR}}$
	P32	1	I/O	↑	Bit	$\overline{\text{HWR}}$
Port 4	P40	1	I/O	↑	Bit	$\overline{\text{CS0}}$
	P41	1	I/O	↑	Bit	$\overline{\text{CS1}}$
	P42	1	I/O	↑	Bit	$\overline{\text{CS2}}$
Port 5	P50 to P53	4	Input	–	(Fixed)	AN0 to AN3, $\overline{\text{ADTRG}}$ (P53)
Port 6	P60	1	I/O	–	Bit	SCK
	P61	1	I/O	–	Bit	SO/SDA
	P62	1	I/O	–	Bit	SI/SCL
	P63	1	I/O	–	Bit	INT0
Port 7	P70	1	I/O	–	Bit	TA0IN
	P71	1	I/O	–	Bit	TA1OUT
	P72	1	I/O	–	Bit	TA3OUT
	P73	1	I/O	–	Bit	TA4IN
	P74	1	I/O	–	Bit	TA5OUT
Port 8	P80	1	I/O	–	Bit	TB0IN0/INT5
	P81	1	I/O	–	Bit	TB0IN1/INT6
	P82	1	I/O	–	Bit	TB0OUT0
	P83	1	I/O	–	Bit	TB0OUT1
Port 9	P90	1	I/O	–	Bit	TXD0
	P91	1	I/O	–	Bit	RXD0
	P92	1	I/O	–	Bit	SCLK0/ $\overline{\text{CTS0}}$
	P93	1	I/O	–	Bit	TXD1
	P94	1	I/O	–	Bit	RXD1
	P95	1	I/O	–	Bit	SCLK1/ $\overline{\text{CTS1}}$
	P96	1	I/O	–	Bit	XT1
	P97	1	I/O	–	Bit	XT2

Table 3.6.2 I/O Port Setting List (1/2)

Ports	Pin Names	Specifications	I/O Register Setting Values		
			Pn	PnCR	PnFC
Port 0	P00 to P07	Input port	×	0	None
		Output port	×	1	
		AD0 to AD7 bus	×	×	
Port 1	P10 to P17	Input port	×	0	0
		Output port	×	1	0
		AD8 to AD15 bus	×	0	1
		A8 to A15	×	1	1
Port 2	P20 to P25	Input port	×	0	0
		Output port	×	1	0
		A0 to A5 output	×	0	1
		A16 to A21 output	×	1	1
Port 3	P30	Output port	×	None	0
		$\overline{RD}$ output only when accessing an external area	1		1
		Always $\overline{RD}$ output	0		1
	P31	Output port	×	None	0
		$\overline{WR}$ output only when accessing an external area	×		1
	P32	Input port (without pull up)	0	0	0
		Input port (with pull up)	1	0	0
		Output port	×	1	0
		$\overline{HWR}$ output	×	1	1
Port 4	P40 to P42	Input port (without pull up)	0	0	0
		Input port (with pull up)	1	0	0
		Output port	×	1	0
	P40	$\overline{CS0}$ output	×	1	1
	P41	$\overline{CS1}$ output	×	1	1
	P42	$\overline{CS2}$ output	×	1	1
Port 5	P50 to P53	Input port	×	None	
		AN0 to AN3 input (Note 1)	×		
	P53	$\overline{ADTRG}$ input (Note 2)	×		
Port 6	P60 to P63	Input port	×	0	0
		Output port	×	1	0
	P60	SCK input	×	0	0
		SCK output	×	1	1
	P61	SDA input	×	0	0
		SDA output (Note 3)	×	1	1
		SO output	×	1	1
	P62	SI input	×	0	0
		SCL input	×	0	0
		SCL output (Note 3)	×	1	1
	P63	INT0 input	×	0	1

X: Don't care

Note 1: If use P50 to P53 as input channels of AD converter, channel selection set by using ADMODE1<ADCH2:0>.

Note 2: If use P53 as $\overline{ADTRG}$ input, enabling external trigger is set by using ADMODE1<ADTRGE>.

Note 3: If use P61 as open-drain output in SDA output and P62 as open-drain output in SCL output, please set ODE<ODE62:61>.

Table 3.6.3 I/O Port Setting List (2/2)

Ports	Pin Names	Specifications	I/O Register Setting Values		
			Pn	PnCR	PnFC
Port 7	P70 to P74	Input port	×	0	0
		Output port	×	1	0
	P70	TA0IN input	×	0	None
	P71	TA1OUT output	×	1	1
	P72	TA3OUT output	×	1	1
	P73	TA4IN input	×	0	None
	P74	TA5OUT output	×	1	1
Port 8	P80 to P83	Input port	×	0	0
		Output port	×	1	0
	P80	TB0IN0, INT5 input	×	0	1
	P81	TB0IN1, INT6 input	×	0	1
	P82	TB0OUT0 output	×	1	1
	P83	TB0OUT1 output	×	1	1
Port 9	P90 to P95	Input port	×	0	0
		Output port	×	1	0
	P90	TXD0 output	×	1	1
	P91	RXD0 input	×	0	None
	P92	SCLK0 input	×	0	0
		SCLK0 output	×	1	1
		$\overline{\text{CTS0}}$ input	×	0	0
	P93	TXD1 output	×	1	1
	P94	RXD1 input	×	0	None
	P95	SCLK1 input	×	0	0
		SCLK1 output	×	1	1
		$\overline{\text{CTS1}}$ input	×	0	0
	P96 to P97	Input port	×	0	None
		Output port (Note 4)	×	1	
		XT1 to XT2 (Note 5)	×	0	

X: Don't care

Note 4: If use P96 to P97 as output port, it be set open-drain buffer.

Note 5: If use P96 to P97 as XT1 to XT2, enabling oscillation and so on are set by using SYSCR0 register.

By resetting, these port pin become general-purpose input port.

And it become input port except P96/XT1 and P97/XT2 that input pin of input and output is programmable.

When use port pin to internal function, need setting by program.

3.6.1 Port 0 (P00 to P07)

Port 0 is an 8-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P0CR. Resetting, reset all bits of the control register P0CR to "0" and sets port 0 to input mode.

In addition to functioning as a general-purpose I/O port, port 0 can also function as address data bus (AD0 to AD7).

When access external memory, port 0 function as address data bus (AD0 to AD7) and P0CR be cleared to "0".

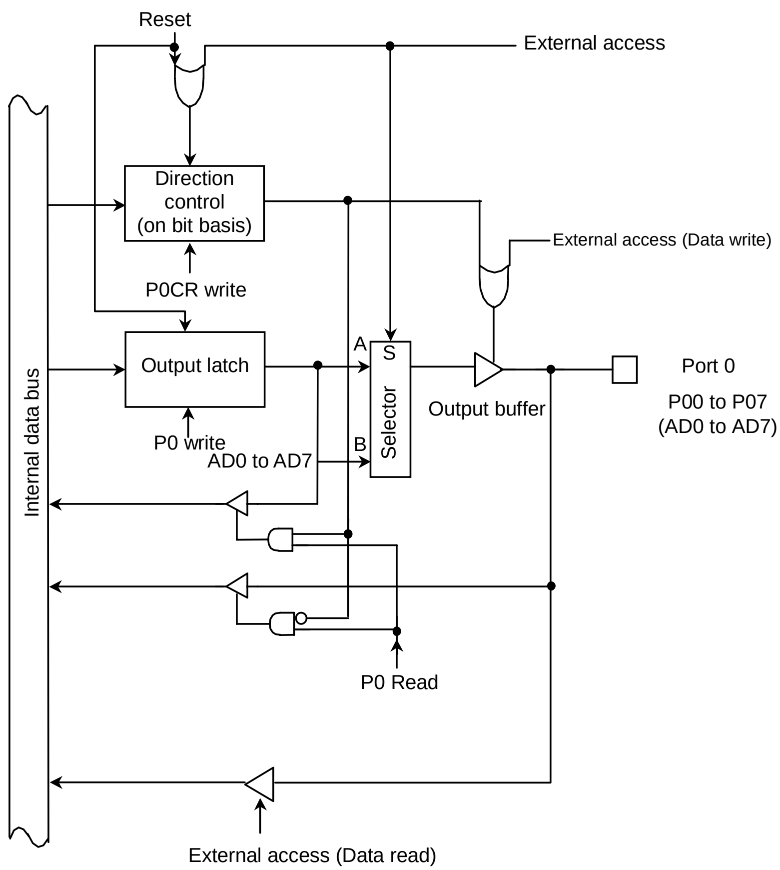


Figure 3.6.1 Port 0

3.6.2 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P1CR and function register P1FC. Resetting reset all bits of output latch P1, the control register P1CR and function register P1FC to “0” and sets port 1 to input mode.

In addition to functioning as a general-purpose I/O port, port 1 can also function as address data bus (AD8 to AD15) and address bus (A8 to A15).

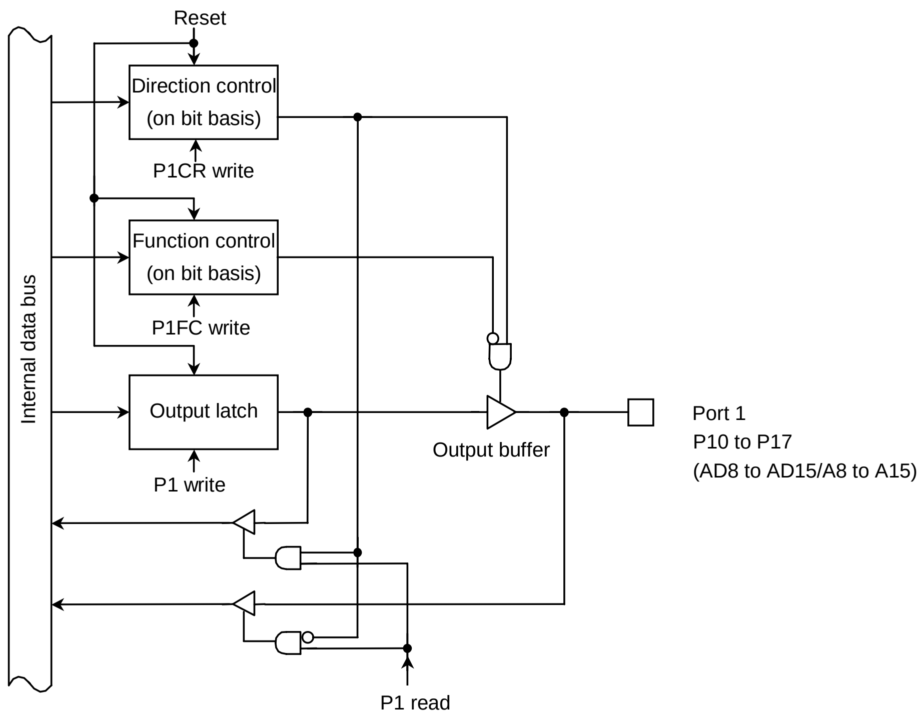


Figure 3.6.2 Port 1

Port 0 Register

P0 (0000H)

	7	6	5	4	3	2	1	0
Bit symbol	P07	P06	P05	P04	P03	P02	P01	P00
Read/Write	R/W							
After reset	Data from external port (Output latch register is undefined.)							

Port 0 Control Register

P0CR (0002H)

	7	6	5	4	3	2	1	0
Bit symbol	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output (When access to external, become AD7 to AD0 and this register is cleared to “0”.)							

Port I/O setting

0	Input
1	Output

Port 1 Register

P1 (0001H)

	7	6	5	4	3	2	1	0
Bit symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”.)							

Port 1 Control Register

P1CR (0004H)

	7	6	5	4	3	2	1	0
Bit symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	<<Refer to column of P1FC>>							

Port 1 Function Register

P1FC (0005H)

	7	6	5	4	3	2	1	0
Bit symbol	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	P1FC/P1CR = 00: Input, 01: Output, 10: AD15 to AD8, 11: A15 to A8							

Note: Read-modify-write instructions are prohibited for registers P0CR, P1CR and P1FC.

Port1 function setting

P1CR<P1XC> \ P1FC<P1XF>	0	1
0	Input Port	Address data bus (AD15 to AD8)
1	Output Port	Address bus (A15 to A8)

Note: <P1XF>/<P1XC> is bit X of each register P1FC/P1CR.

Figure 3.6.3 Register for Ports 0 and 1

3.6.3 Port 2 (P20 to P25)

Port 2 is an 6-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P2CR and function register P2FC. Resetting, set all bits of output latch P2 to "1", and reset the control register P2CR and function register P2FC to "0" and sets port 2 to input mode.

In addition to functioning as a general-purpose I/O port, port 2 can also function as address bus (A0 to A5) and address bus (A16 to A21).

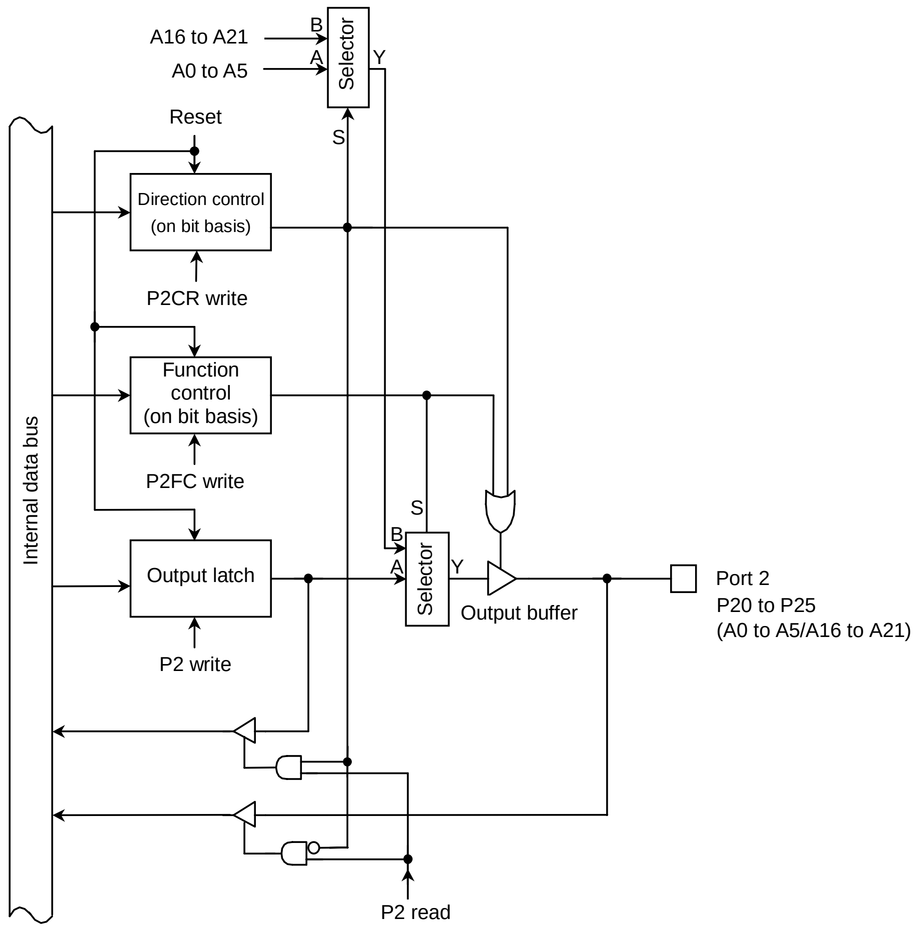


Figure 3.6.4 Port 2

Port 2 Register

		7	6	5	4	3	2	1	0
P2 (0006H)	Bit symbol			P25	P24	P23	P22	P21	P20
	Read/Write			R/W					
	After reset			Data from external port (Output latch register is set to “1”.)					

Port 2 Control Register

		7	6	5	4	3	2	1	0
P2CR (0008H)	Bit symbol			P25C	P24C	P23C	P22C	P21C	P20C
	Read/Write			W					
	After reset			0	0	0	0	0	0
	Function			<<Refer to column of P2FC>>					

Port 2 Function Register

		7	6	5	4	3	2	1	0
P2FC (0009H)	Bit symbol			P25F	P24F	P23F	P22F	P21F	P20F
	Read/Write			W					
	After reset			0	0	0	0	0	0
	Function			P2FC/P2CR = 00: Input, 01: Output, 10: A5 to A0, 11: A21 to A16					

Note: Read-modify-write instructions are prohibited for registers P2CR and P2FC.

Port 2 function setting

P2CR<P2XC> \ P2FC<P2XF>	0	1
0	Input Port	Address bus (A5 to A0)
1	Output Port	Address bus (A21 to A16)

Note: <P2XF>/<P2XC> is bit X of each register P2FC/P2CR. When set to address bus A21 to A16, set P2FC after set P2CR.

Figure 3.6.5 Register for Port 2

3.6.4 Port 3 (P30 to P32)

Port 3 is a 3-bit general-purpose I/O port (however P30 and P31 is only output port). Each bit can be set individually for input or output using the control register P3CR and function register P3FC. Resetting, all bits of output latch P3 is set to "1", and the control register P3CR (Bit0 and bit1 don't using) and function register P3FC are reset to "0". And P30 and P31 of port 3 output "High", and sets P32 to input mode with pull-up resister.

In addition to functioning as a general-purpose I/O port, port 3 can also function as the output for the CPU's control/status signal.

Case of P30 is defined as $\overline{RD}$ signal output mode (Case of <P30F> = "1"), when the output latch register <P30> clearing to "0", outputs the $\overline{RD}$ strobe (used for the pseudo static RAM) of the $\overline{RD}$ pin even when the internal addressed.

If the <P30 > remains "1", the $\overline{RD}$ strobe signal is output only when the external address area is accessed.

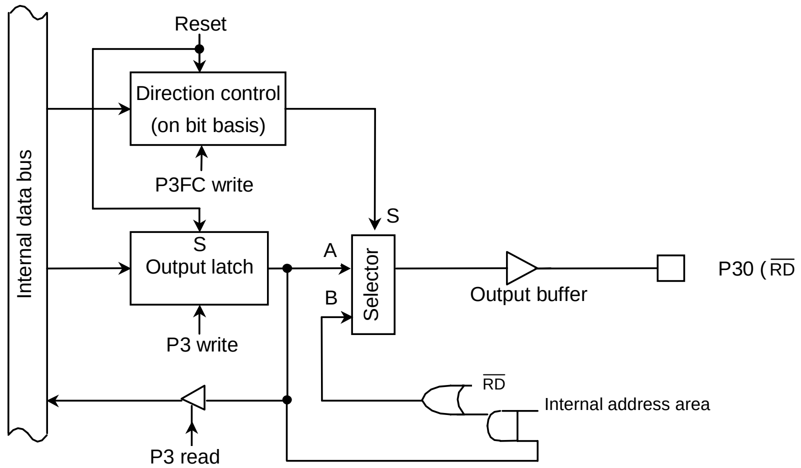


Figure 3.6.6 Port 3 (P30)

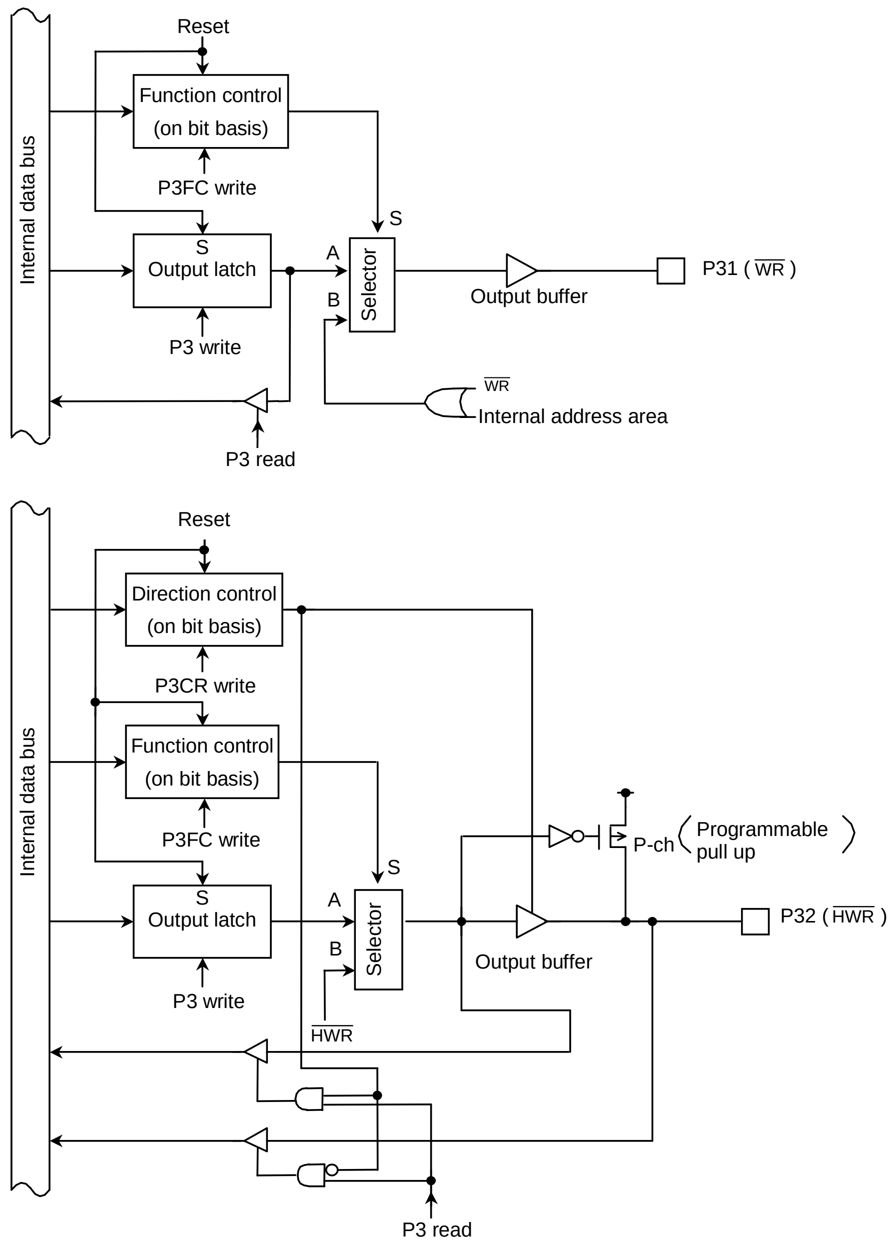


Figure 3.6.7 Port 3 (P31 and P32)

Port 3 Register

P3 (0007H)		7	6	5	4	3	2	1	0
	Bit symbol						P32	P31	P30
	Read/Write						R/W		
	After reset						Data from external port Note3	1	1
	Function						Pull-up resistor 0: OFF 1: ON		

Port 3 Control Register

P3CR (000AH)		7	6	5	4	3	2	1	0
	Bit symbol						P32C		
	Read/Write						W		
	After reset						0		
							0: Input 1: Output		

Input/ Output setting

0	Input
1	Output

Port 3 Function Register

P3FC (000BH)		7	6	5	4	3	2	1	0
	Bit symbol	–					P32F	P31F	P30F
	Read/Write	W					W		
	After reset	0					0	0	0
	Function	Write “0”.					0: Port 1: $\overline{HWR}$	0: Port 1: $\overline{WR}$	0: Port 1: $\overline{RD}$

P30 ($\overline{RD}$) function setting

<P30F> \ <P30>	0	1
0	“0” output	“1” output
1	$\overline{RD}$ output always. (Correspond to pseudo SRAM)	$\overline{RD}$ is only output during external accesses.

P31 ($\overline{WR}$) function setting

<P31F> \ <P31>	0	1
0	“0” output	“1” output
1	$\overline{WR}$ is only output during external accesses.	

P32 ($\overline{HWR}$) function setting

<P32F> \ <P32C>	0	1
0	Input port	Output port
1	–	$\overline{HWR}$ output

Note 1: Read-modify-write instrustions are prohibited for registers P3CR and P3FC.

Note 2: When port 3 is used in Input mode, the P3 register controls the internal pull-up resistor. Read-modify-write instruction is prohibited in Input mode or I/O mode. Setting the internal pull-up resistor may be depend on the states of the input pin.

Note 3: Output latch register is set to “1”, and pull-up resistor is connected.

Figure 3.6.8 Register for Port 3

3.6.5 Port 4 (P40 to P42)

Port 4 is a 3-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P4CR and function register P4FC. Resetting, set P40 to P42 of output register to "1", the control register P4CR and function register P4FC reset to "0" and sets port 4 to input mode with pull-up resistor.

In addition to functioning as a general-purpose I/O port, port 4 can also function as chip select output signal ($\overline{\text{CS0}}$ to $\overline{\text{CS2}}$).

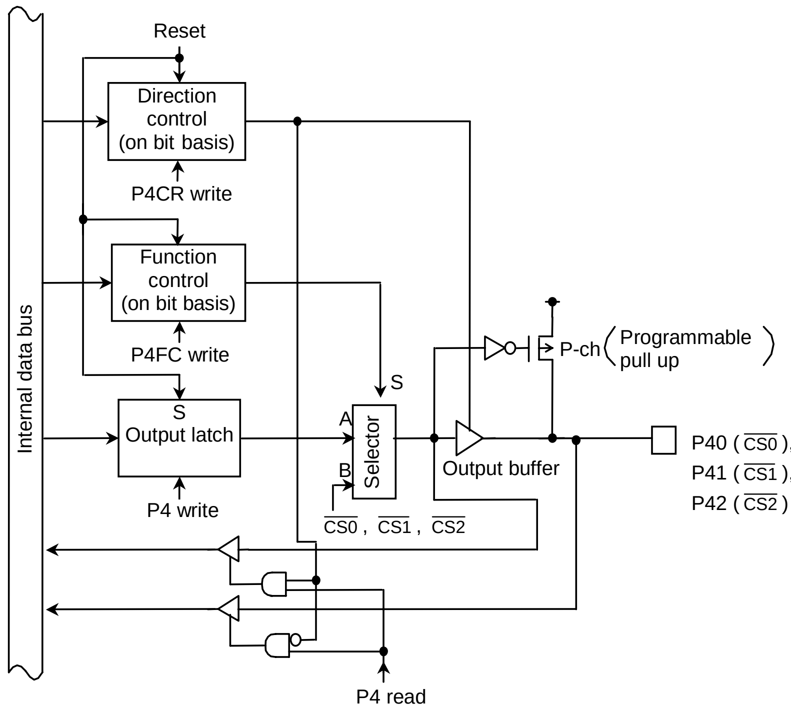


Figure 3.6.9 Port4

Port 4 Register

		7	6	5	4	3	2	1	0
P4 (000CH)	Bit symbol						P42	P41	P40
	Read/Write						R/W		
	After reset						Data from external port Note4		
	Function						Pull-up resistor 0: OFF 1: ON		

Port 4 Control Register

		7	6	5	4	3	2	1	0
P4CR (000EH)	Bit symbol						P42C	P41C	P40C
	Read/Write						W		
	After reset						0	0	0
							0: Input 1: Output		

Input/Output setting

0	Input
1	Output

Port 4 Function Register

		7	6	5	4	3	2	1	0
P4FC (000FH)	Bit symbol						P42F	P41F	P40F
	Read/Write						W		
	After reset						0	0	0
	Function						0: Port 1: $\overline{CS}$		

0	Port (P40)
1	$\overline{CS0}$

0	Port (P41)
1	$\overline{CS1}$

0	Port (P42)
1	$\overline{CS2}$

Note 1: Read-modify-write instructions are prohibited for registers, P4CR and P4FC.

Note 2: When port 4 is used in Input mode, the P4 register controls the internal pull-up resistor. Read-modify-write instruction is prohibited in Input mode or I/O mode. Setting the internal pull-up resistor may be depend on the states of the input pin.

Note 3: When output chip select signal ($\overline{CS0}$ to $\overline{CS2}$), set bit of control register (P4CR) to “1” after set bit of function register (P4FC) to “1”.

Note 4: Output latch register is set to “1”, and pull-up resistor is connected.

Figure 3.6.10 Register for Port 4

3.6.6 Port 5 (P50 to P53)

Port 5 is a 4-bit input port and can also be used as the analog input pin for the AD converter. P53 can also be used as AD trigger input pin for AD converter.

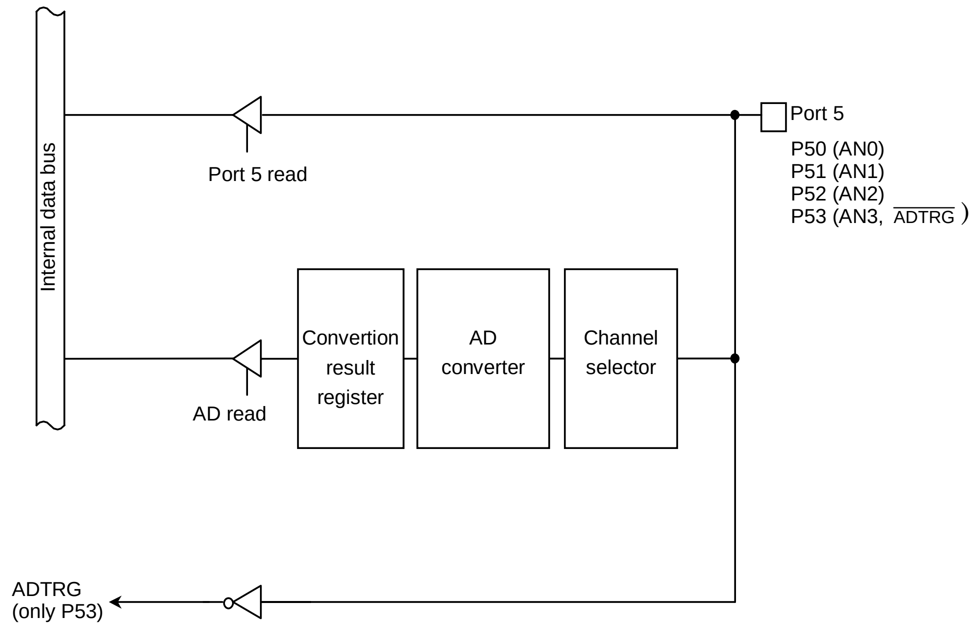


Figure 3.6.11 Port 5

Port 5 Register

		7	6	5	4	3	2	1	0
P5 (000DH)	Bit symbol					P53	P52	P51	P50
	Read/Write					R			
	After reset					Data from external port			

Figure 3.6.12 Register for Port 5

Note: The input channel selection of AD converter and the permission of AD trigger input of P53 set by AD converter mode register ADMOD1.

3.6.7 Port 6 (P60 to P63)

Port 60 to P63 are 4-bit general-purpose I/O ports. Resetting, set to input port. All bits of output latch register P6 are set to "1".

In addition to functioning as a I/O port, port 6 can also function as input or output function of serial bus interface. This function enable each function by writing "1" to applicable bit of Port 6 function register P6FC.

Resetting, P6CR and P6FC reset to "0", all bit set input port.

(1) Port 60 (SCK)

In addition to functioning as an I/O port, port 60 can also function as clock SCK I/O port in SIO mode of serial bus interface.

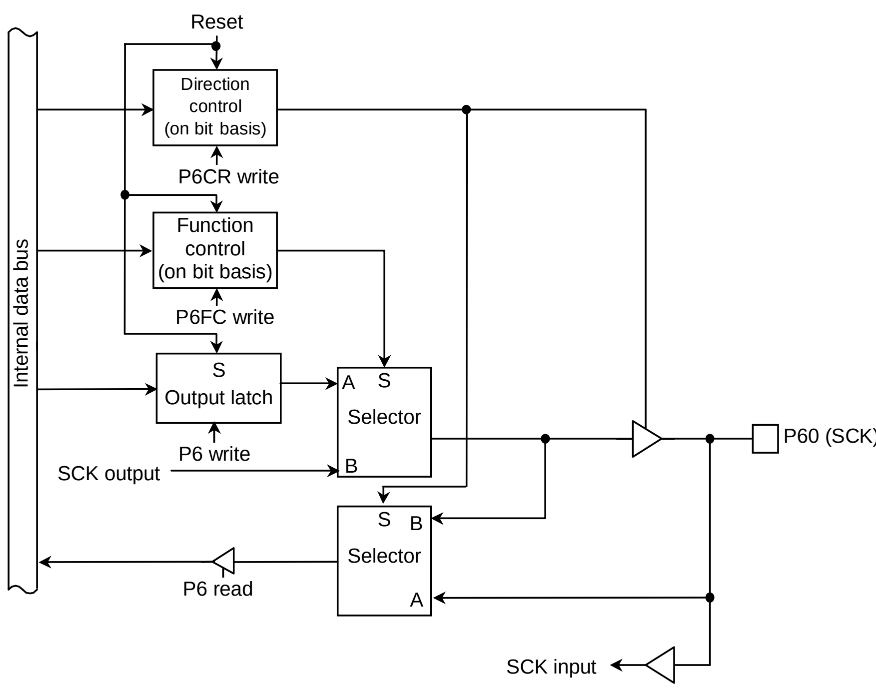


Figure 3.6.13 Port 60

(2) Port 61 (SO/SDA)

In addition to functioning as an I/O port, port 61 can also function as data SDA I/O port in I2C mode or data SO output pin in SIO mode of serial bus interface.

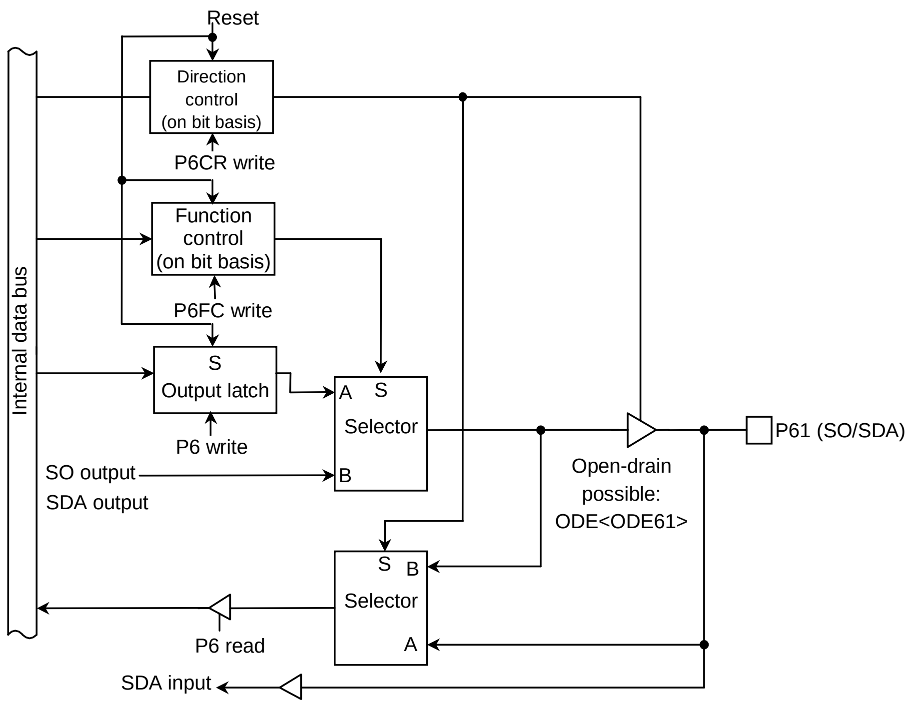


Figure 3.6.14 Port 61

(3) Port 62 (SI/SCL)

In addition to functioning as an I/O port, port 62 can also function as data receiving pin in SIO mode or clock SCL I/O pin in I²C bus mode of serial bus interface.

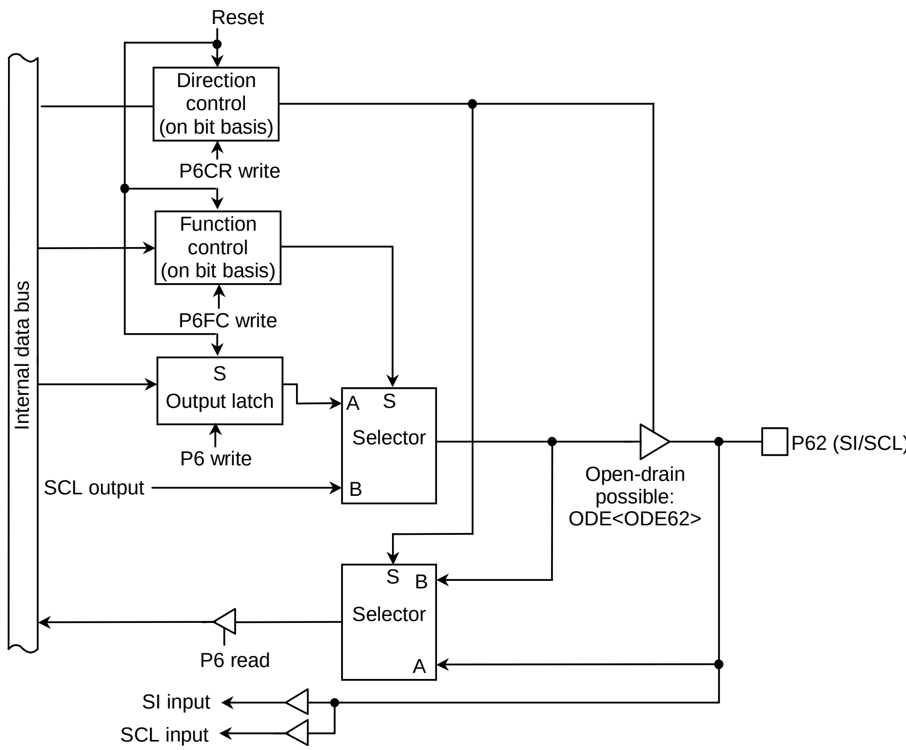


Figure 3.6.15 Port 62

(4) Port 63 (INT0)

In addition to functioning as an I/O port, port 63 can also function as INT0 input pin of external interrupt.

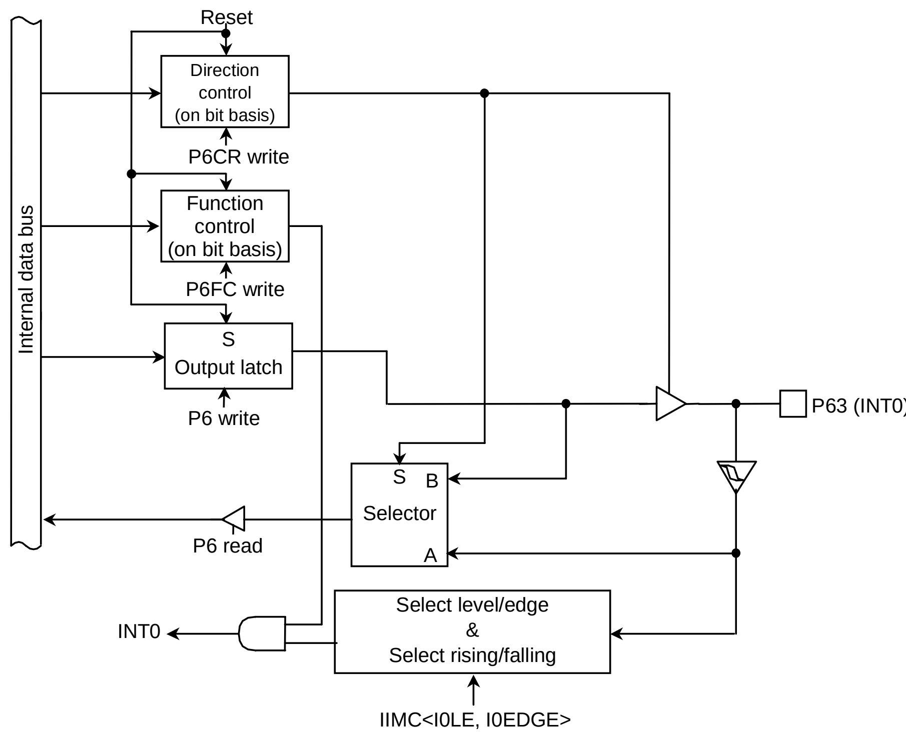


Figure 3.6.16 Port 63

Port 6 Registers

		7	6	5	4	3	2	1	0
P6 (0012H)	Bit symbol					P63	P62	P61	P60
	Read/Write					R/W			
	After reset					Data from external port (Output latch register is set to “1”)			

Port 6 Control Register

		7	6	5	4	3	2	1	0
P6CR (0014H)	Bit symbol					P63C	P62C	P61C	P60C
	Read/Write					W			
	After reset					0	0	0	0
	Function					0: Input 1: Output			

Port6 I/O setting

0	Input
1	Output

Port 6 Function Register

		7	6	5	4	3	2	1	0
P6FC (0015H)	Bit symbol					P63F	P62F	P61F	P60F
	Read/Write					W	W	W	W
	After reset					0	0	0	0
	Function					0: Port 1: INT0 input	0: Port 1: SCL output	0: Port 1: SDA/SO output	0: Port 1: SCK output

P60 SCK output setting

P6FC<P60F>	1
P6CR<P60C>	1

P61 SDA/SO output setting

P6FC<P61F>	1
P6CR<P61C>	1

P62 SCL output setting

P6FC<P62F>	1
P6CR<P62C>	1

P63 INT0 input setting

P6FC<P63F>	1
P6CR<P63C>	0

Note: Read-modify-write instructions are prohibited for registers P6CR and P6FC.

Figure 3.6.17 Register for Port 6

3.6.8 Port 7 (P70 to P74)

Port 7 is a 5-bit general-purpose I/O port. Resetting, set to input port.

In addition to functioning as a I/O port, port 70 and 73 can also function as clock input pin TA0IN, TA4IN of 8-bit timer 0, 4 and port 71, 72, 74 can also function 8-bit timer output pin TA1OUT, TA3OUT, TA5OUT. This timer output function enable each function by writing "1" to applicable bit of Port 7 function register P7FC.

Resetting, P7CR and P7FC reset to "0", all bit set input port.

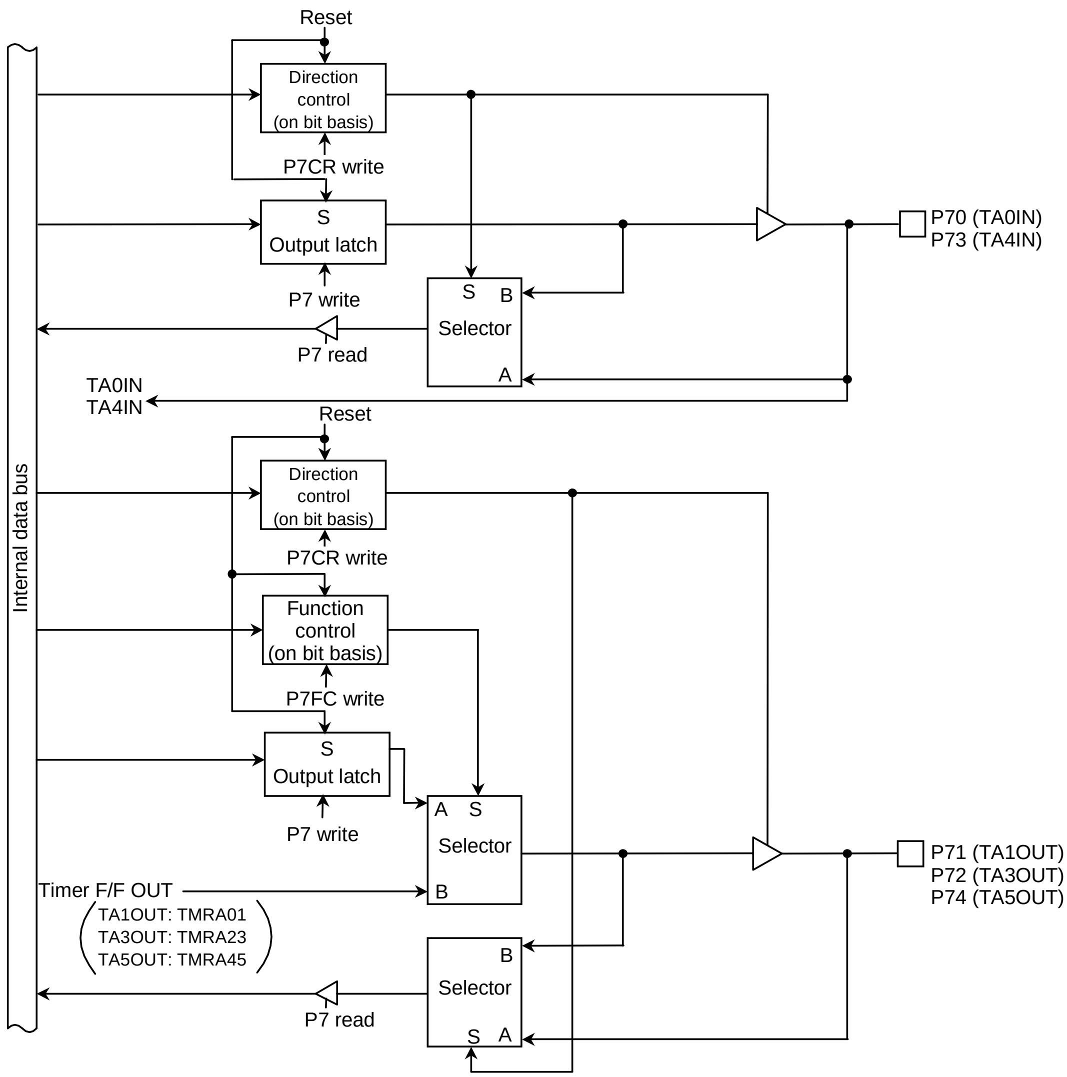


Figure 3.6.18 Port 7

Port 7 Register

P7 (0013H)		7	6	5	4	3	2	1	0
	Bit symbol				P74	P73	P72	P71	P70
	Read/Write				R/W				
	After reset				Data from external port (Output latch register is set to “1”.)				

Port 7 Control Register

P7CR (0016H)		7	6	5	4	3	2	1	0
	Bit symbol				P74C	P73C	P72C	P71C	P70C
	Read/Write				W				
	After reset				0	0	0	0	0
	Function				0: Input 1: Output				

Port 7 I/O setting

0	Input
1	Output

Port 7 Function Register

P7FC (0017H)		7	6	5	4	3	2	1	0
	Bit symbol				P74F		P72F	P71F	
	Read/Write				W		W		
	After reset				0		0	0	
	Function				0: Port 1: TA5OUT		0: Port 1: TA3OUT	0: Port 1: TA1OUT	

P71 timer out 1 output setting

P7FC<P71F>	1
P7CR<P71C>	1

P72 timer out 3 output setting

P7FC<P72F>	1
P7CR<P72C>	1

P74 timer out 5 output setting

P7FC<P74F>	1
P7CR<P74C>	1

Note 1: Read-Modify-Write instructions are prohibited for the registers P7CR and P7FC.

Note 2: P70/TA0IN and P73/TA4IN pin does not have a register changing Port/Function.
For example, when it is used as an input port, the input signal is inputted to 8-bit timer.

Figure 3.6.19 Register for Port 7

3.6.9 Port 8 (P80 to P83)

Port 8 is a 4-bit general-purpose I/O port. Resetting, set to input port. All bits of output latch register P8 are set to “1”.

In addition to functioning as a I/O port, port 8 can also function as clock input of 16-bit timer, output of 16-bit timer F/F and input function of INT5 to INT6. This function enable each function by writing “1” to applicable bit of port 8 function register P8FC.

Resetting, P8CR and P8FC reset to “0”, all bits set input port.

(1) P80 to P83

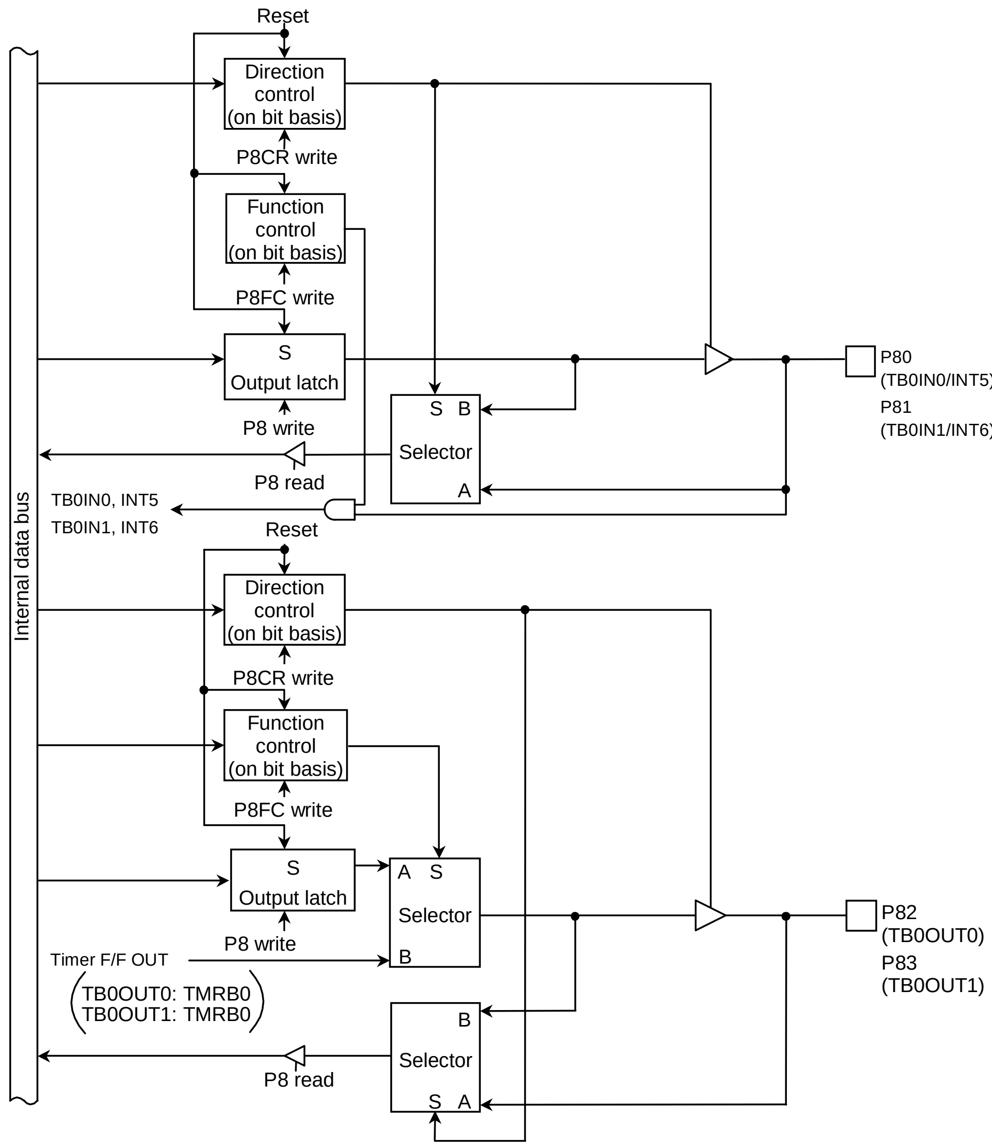


Figure 3.6.20 Port 8 (P80 to P83)

Port 8 Register

P8 (0018H)		7	6	5	4	3	2	1	0
	Bit symbol					P83	P82	P81	P80
	Read/Write					R/W			
	After reset					Data from external port (Output latch register is set to "1".)			

Port 8 Control Register

P8CR (001AH)		7	6	5	4	3	2	1	0
	Bit symbol					P83C	P82C	P81C	P80C
	Read/Write					W			
	After reset					0	0	0	0
						0: Input 1: Output			

Port 8 I/O setting

0	Input
1	Output

Port 8 Function Register

P8FC (001BH)		7	6	5	4	3	2	1	0
	Bit symbol					P83F	P82F	P81F	P80F
	Read/Write					W	W	W	W
	After reset					0	0	0	0
	Function					0: Port 1: TB0OUT1	0: Port 1: TB0OUT0	0: Port 1: TB0IN1 INT6 input	0: Port 1: TB0IN0 INT5 input

P82 TB0OUT0 output setting

P8FC<P82F>	1
P8CR<P82C>	1

P83 TB0OUT1 output setting

P8FC<P83F>	1
P8CR<P83C>	1

Note: Read-modify-write instructions are prohibited for registers P8CR and P8FC.

Figure 3.6.21 Register for Port 8

3.6.10 Port 9 (P90 to P97)

- Ports 90 to 95

Ports 90 to 95 are a 6-bit general-purpose I/O port. Resetting, set to input port. All bits of output latch register are set to “1”.

In addition to functioning as a I/O port, port 90 to 95 can also function as I/O of SIO0, SIO1. This function enable each function by writing “1” to applicable bit of port 9 function register P9FC.

Resetting, P9CR and P9FC reset to “0”, all bits set input port.

- Ports 96 to 97

Ports 96 to 97 are a 2-bit general-purpose I/O port. Case of output port, this is open drain output. Resetting, output latch register and control register set to “1”, and set to “High-Z” (High impedance).

In addition to functioning as a I/O port, ports 96 to 97 can also function as low-frequency oscilator connection pin (XT1 and XT2) during using low speed clock function. Therefore, dual clock function can use by setting of system clock control registers SYSCR0 and SYSCR1.

(1) Ports 90 and 93 (TXD0 and TXD1)

In addition to functioning as a I/O port, Ports 90 and 93 can also function as TXD output pin of serial channel.

And P90 and P93 have a programmable open-drain function which can be controlled by the ODE<ODE90, 93> register.

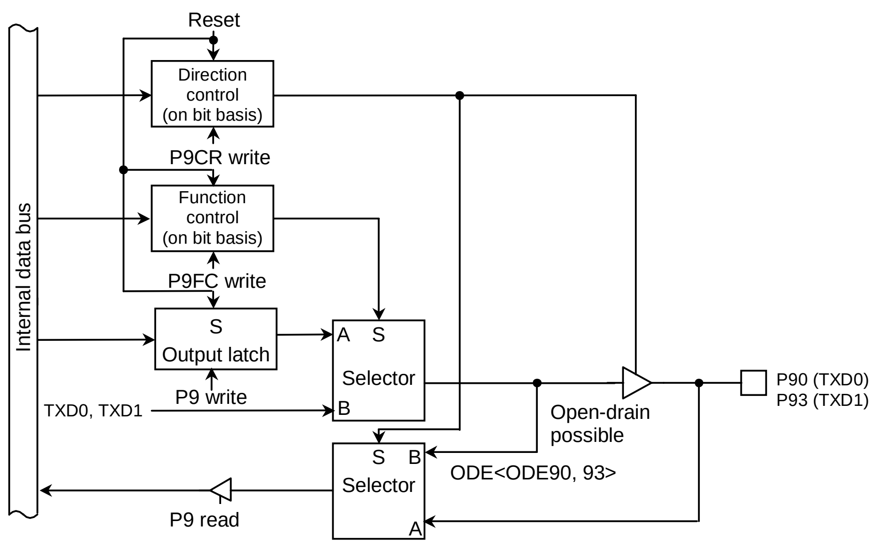


Figure 3.6.22 Ports 90 and 93

(2) Ports 91 and 94 (RXD0 and RXD1)

In addition to functioning as a I/O port, ports 91 and 94 can also function as RXD input pin of serial channel.

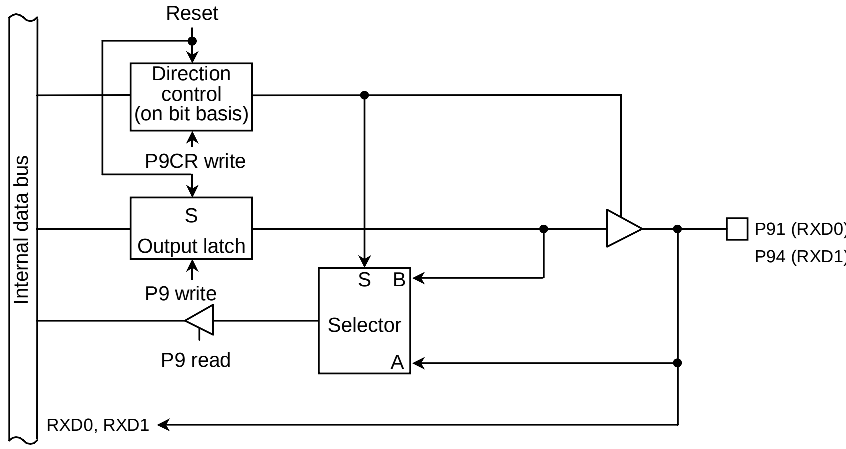


Figure 3.6.23 Ports 91 and 94

(3) Ports 92 and 95 ($\overline{\text{CTS0}}$ /SCLK0, $\overline{\text{CTS1}}$ /SCLK1)

In addition to functioning as a I/O port, ports 92 and 95 can also function as $\overline{\text{CTS}}$ input pin or SCLK I/O pin of serial channel.

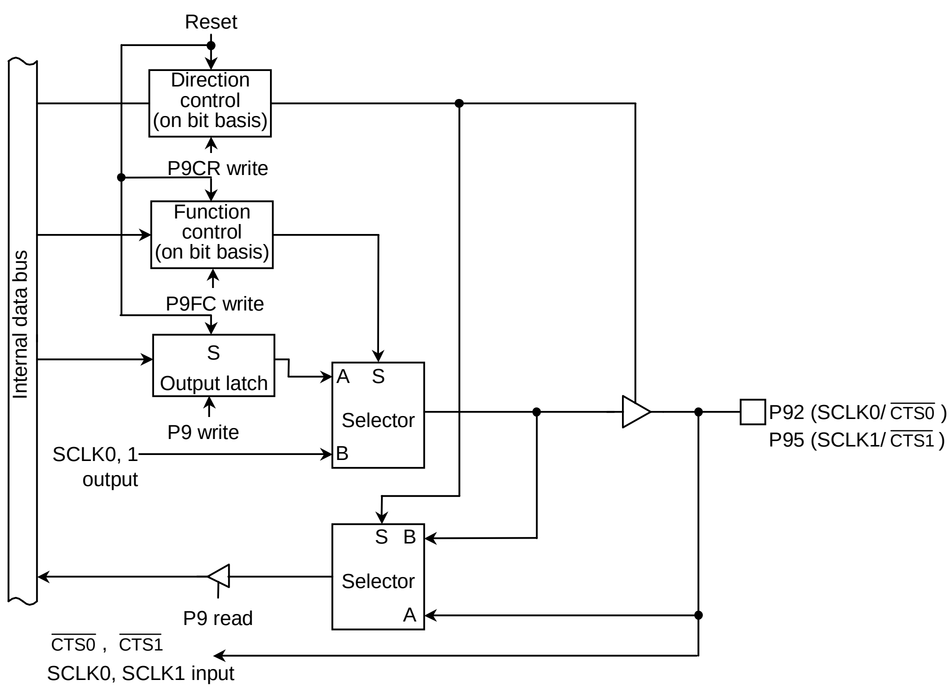


Figure 3.6.24 Port 92, 95

(4) Ports 96 (XT1) and 97 (XT2)

In addition to functioning as a I/O port, ports 96 and 97 can also function as low frequency oscillator connection pins.

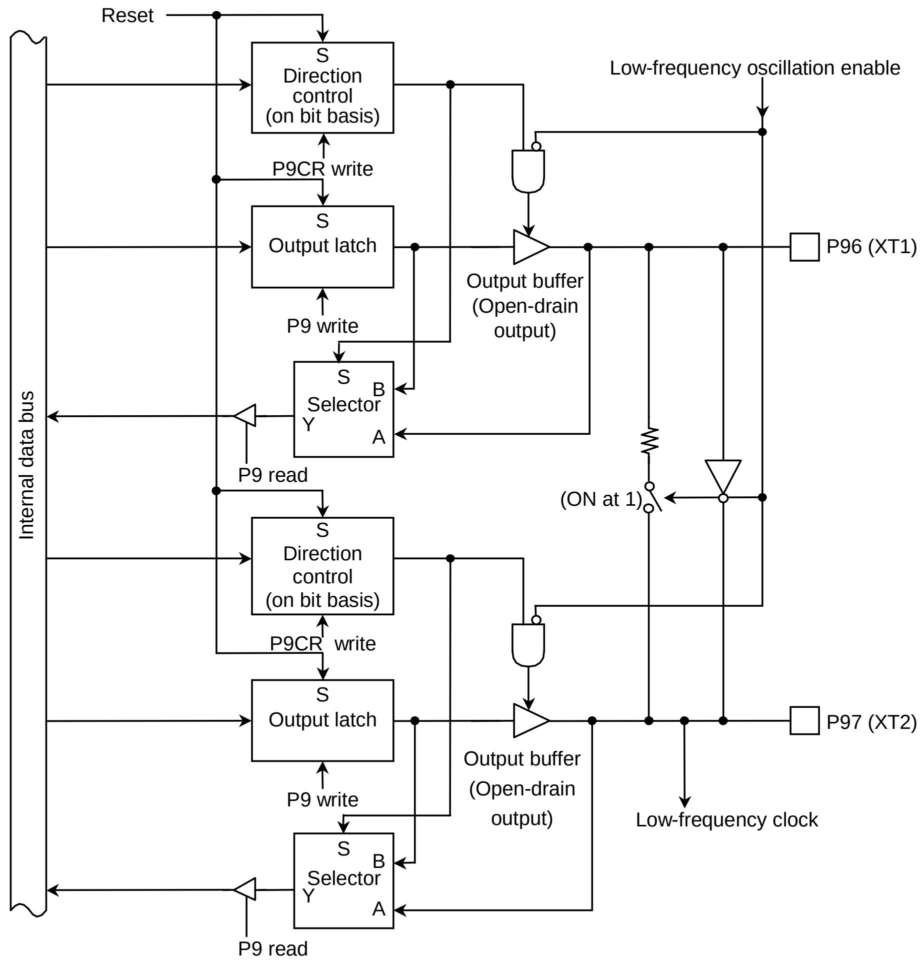


Figure 3.6.25 Ports 96 and 97

Port 9 Registers

		7	6	5	4	3	2	1	0
P9 (0019H)	Bit symbol	P97	P96	P95	P94	P93	P92	P91	P90
	Read/Write	R/W							
	After reset	1	1	Data from external port (Output latch register is set to “1”.)					

Port 9 Control Register

		7	6	5	4	3	2	1	0
P9CR (001CH)	Bit symbol	P97C	P96C	P95C	P94C	P93C	P92C	P91C	P90C
	Read/Write	W							
	After reset	1	1	0	0	0	0	0	0
		0: Input 1: Output							

Port9 I/O setting

0	Input
1	Output

Note: Ports 96 and 97 are open-drain output pins.

Port 9 Function Register

		7	6	5	4	3	2	1	0
P9FC (001DH)	Bit symbol			P95F		P93F	P92F		P90F
	Read/Write			W		W	W		W
	After reset			0		0	0		0
	Function			0: Port 1: SCLK1 output		0: Port 1: TXD1	0: Port 1: SCLK0 output		0: Port 1: TXD0

P90 TXD0 output setting

P9FC<P90F>	1
P9CR<P90C>	1

P92 SCLK0 output setting

P9FC<P92F>	1
P9CR<P92C>	1

P93 TXD1 output setting

P9FC<P93F>	1
P9CR<P93C>	1

P95 SCLK1 output setting

P9FC<P95F>	1
P9CR<P95C>	1

Note 1: Read-modify-write instructions are prohibited for the registers P9CR and P9FC.

Note 2: When set TXD pin to open-drain output, write “1” to bit0 of ODE register (for TXD0 pin), or bit1 (for TXD1 pin).
P91/RXD0 and P94/RXD1 pin does not have a register changing Port/Function.
For example, when it is also used as an input port, the input signal is inputted to SIO as serial receiving data.

Note 3: Low frequency oscillation circuit
To connect a low frequency resonator to ports 96 and 97, it is necessary to set a following procedure to reduce the consumption power supply.
(Case of resonator connection)
P9CR<P96C, P97C> = “11”, P9<P96:97> = “00”
(Case of oscillator connection)
P9CR<P96C, P97C> = “11”, P9<P96:97> = “10”

Figure 3.6.26 Register for Port 9

3.7 Chip select/Wait Controller

On the TM91CP27, four user-specifiable address areas (CS0 to CS3) can be set. The data bus width and the number of waits can be set independently for each address area (CS0 to CS3 and others).

The pins $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ (which can also function as port pins P40 to P42) are the respective output pins for the CS0 to CS2 areas. When the CPU specifies an address in one of these areas, the corresponding CS0 to CS2 pin outputs the chip select signal for the specified address area (in ROM or SRAM). However, in order for the chip select signal to be output, the port 4 control register P4CR and function register P4FC must be set.

The areas CS0 to CS3 are defined by the values in the memory start address registers MSAR0 to MSAR3 and the memory address mask registers MAMR0 to MAMR3.

The chip select/wait control registers B0CS to B3CS and BEXCS should be used to specify the master enable status, the data bus width and the number of waits for each address area.

3.7.1 Specifying an Address Area

The CS0 to CS3 address areas are specified using the start address registers (MSAR0 to MSAR3) and memory address mask registers (MAMR0 to MAMR3).

At each bus cycle, a compare operation is performed to determine if the address on the specified a location in the CS0 to CS3 area. If the result of the comparison is a match, this indicates an access to the corresponding CS area. In this case, the $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ pin outputs the chip select signal and the bus cycle operates in accordance with the settings in chip select/wait control register B0CS to B3CS. (See section 3.7.2, “Chip Select/Wait Control Registers”.)

(1) Memory start address registers

Figure 3.7.1 shows the memory start address registers. The memory start address registers MSAR0 to MSAR3 set the start addresses for the CS0 to CS3 areas. Set the upper 8 bits (A23 to A16) of the start address in <S23:S16>. The lower 16 bits of the start address (A15 to A0) are permanently set to 0. Accordingly, the start address can only be set in 64-Kbyte increments, starting from 000000H. shows the relationship between the start address and the start address register value.

Memory Start Address Register (CS0 to CS3 area)

MSAR0 (00C8H) / MSAR1 (00CAH)
MSAR2 (00CCH) / MSAR3 (00CEH)

	7	6	5	4	3	2	1	0
Bit symbol	S23	S22	S21	S20	S19	S18	S17	S16
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Determines A23 to A16 of start address							

Sets start addresses for areas CS0 to CS3

Figure 3.7.1 Memory Start Address Register

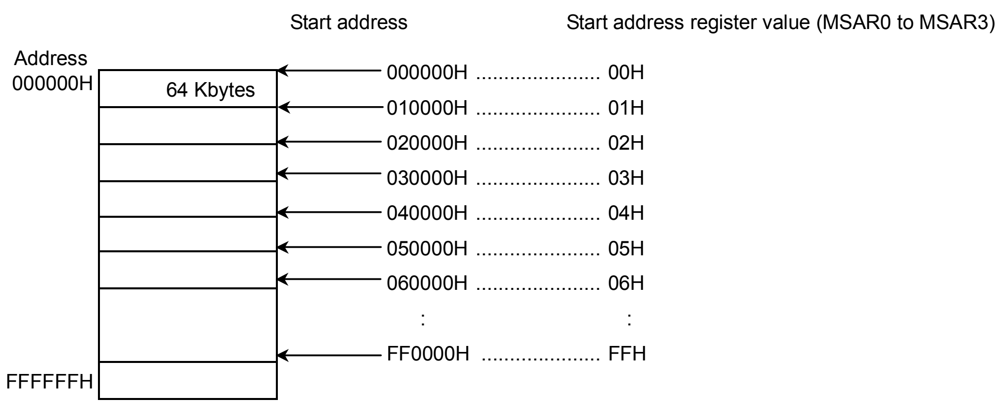


Figure 3.7.2 Start Address and Start Address Register Value

(2) Memory address mask registers

Figure 3.7.3 shows the memory address mask registers. Memory address mask registers MAMR0 to MAMR3 are used to set the size of the CS0 to CS3 areas by specifying a mask for each bit of the start address set in memory start address registers MSAR0 to MSAR3. The compare operation used to determine if an address is in the CS0 to CS3 areas is only performed for bus address bits corresponding to bits set to “0” in these registers. Also, the address bits that can be masked by MAMR0 to MAMR3 differ between CS0 to CS3 areas. Accordingly, the size that can be each area is different.

Memory Address Mask Register (CS0 Area)

MAMR0 (00C9H)

	7	6	5	4	3	2	1	0
Bit symbol	V20	V19	V18	V17	V16	V15	V14 to 9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets area size of CS0. 0: Used for address compare							

Range of possible settings for CS0 area size: 256 bytes to 2 Mbytes

Memory Address Mask Register (CS1 Area)

MAMR1 (00CBH)

	7	6	5	4	3	2	1	0
Bit symbol	V21	V20	V19	V18	V17	V16	V15 to 9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets area size of CS1. 0: Used for address compare							

Range of possible settings for CS1 area size: 256 bytes to 4 Mbytes.

Memory Address Mask Register (CS2 and CS3 Area)

MAMR2 (00CDH) / MAMR3 (00CFH)

	7	6	5	4	3	2	1	0
Bit symbol	V22	V21	V20	V19	V18	V17	V16	V15
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets area size of CS2 and CS3. 0: Used for address compare							

Range of possible settings for CS2 and CS3 area sizes: 32 Kbytes to 8 Mbytes.

Figure 3.7.3 Memory Address Mask Register

(3) Setting memory start address and address area

Figure 3.7.4 show an example of specifying a 64-Kbyte address area starting from 010000H using the CS0 areas.

Set “01H” in memory start address registers MSAR0<S23:16> (Corresponding to the upper 8 bits of the start address). Next, calculate the difference between the start address and the anticipated end address (01FFFFH). Bits 20 to 8 of the result correspond to the mask value to be set for the CS0 area. Setting this value in memory address mask register MAMR0<V20:8> sets the area size. This example sets “07H” in MAMR0 to specify a 64-Kbyte area.

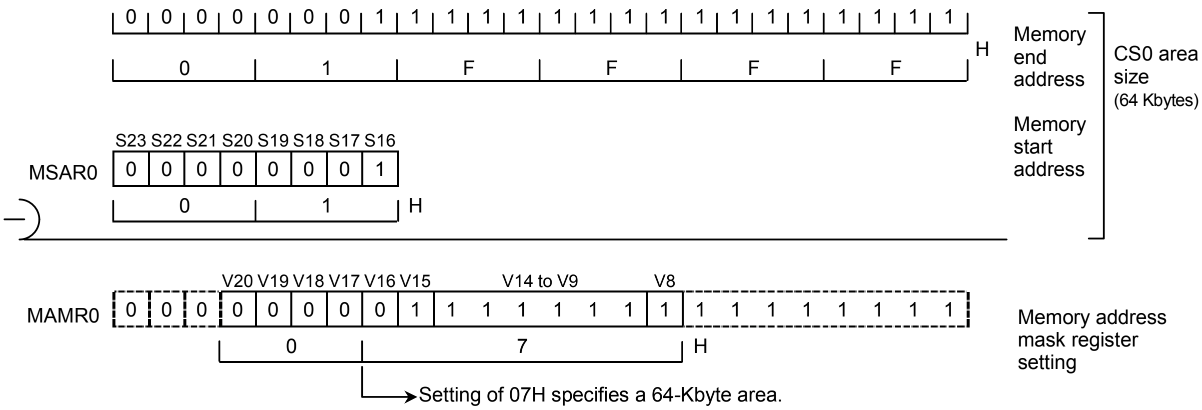


Figure 3.7.4 Example Showing How to Set the CS0 Area

After a reset, MSAR0 to MSAR3 and MAMR0 to MAMR3 are set to “FFH”. B0CS<B0E>, B1CS<B1E> and B3CS<B3E> are reset to “0”. Therefore this is disabling the CS0, CS1 and CS3 areas. However, set as B2CS<B2M> to “0” and B2CS<B2E> to “1”, CS2 is enabled from 002000H to FF3FFFH in TMP91CP27. Also, the bus width and number of waits specified in BEXCS are used for accessing addresses outside the specified CS0 to CS3 area. (See section 3.7.2, “Chip Select/Wait Control Registers”.)

(4) Address area size specification

Table 3.7.1 shows the relationship between CS area and area size. "Δ" indicates areas that cannot be set by memory start address register and address mask register combinations. When setting an area size using a combination indicated by "Δ", set the start address in the desired steps starting from 000000H.

If the CS2 area is set to 16 Mbytes or if two or more areas overlap, the smaller CS area number has the higher priority.

Example: To set the area size for CS0 to 128 Kbytes:

a. Valid start addresses

000000H
128 Kbytes
020000H
128 Kbytes
040000H
128 Kbytes
060000H

Any of these addresses may be set as the start address.

b. Invalid start addresses

000000H
64 Kbytes
010000H
128 Kbytes
030000H
128 Kbytes
050000H

This is not an integer multiple of the desired area size setting. Hence, none of these addresses can be set as the start address.

Table 3.7.1 Valid Area Sizes for Each CS Area

CS Area \ Size (Byte)	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	Δ	Δ	Δ	Δ	Δ		
CS1	○	○		○	Δ	Δ	Δ	Δ	Δ	Δ	
CS2			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ
CS3			○	○	Δ	Δ	Δ	Δ	Δ	Δ	Δ

Note: Δ indicates areas that cannot be set by memory start address register and memory address mask register combinations.

3.7.2 Chip Select/Wait Control Registers

Figure 3.7.5 lists the chip select/wait control registers.

The master enable/disable, chip select output waveform, data bus width and number of wait states for each address area (CS0 to CS3 and others) are set in their respective chip select/wait control registers, B0CS to B3CS and BEXCS.

Chip Select/Wait Control Register

Register		7	6	5	4	3	2	1	0
B0CS (00C0H)	Bit symbol	B0E		B0OM1	B0OM0	B0BUS	B0W2	B0W1	B0W0
	Read/Write	W		W					
	After reset	0		0	0	0	0	0	0
	Function	0: Disable 1: Enable		Chip select output waveform selection 00: For ROM/SRAM 01: Don't care 10: Don't care 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of waits 000: 2 waits 001: 1 wait 010: (1 + N) waits 011: 0 waits	100: Reserved 101: 3 waits 110: 4 waits 111: 8 waits	
B1CS (00C1H)	Bit symbol	B1E		B1OM1	B1OM0	B1BUS	B1W2	B1W1	B1W0
	Read/Write	W		W					
	After reset	0		0	0	0	0	0	0
	Function	0: Disable 1: Enable		Chip select output waveform selection 00: For ROM/SRAM 01: Don't care 10: Don't care 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of waits 000: 2 waits 001: 1 wait 010: (1 + N) waits 011: 0 waits	100: Reserved 101: 3 waits 110: 4 waits 111: 8 waits	
B2CS (00C2H)	Bit symbol	B2E	B2M	B2OM1	B2OM0	B2BUS	B2W2	B2W1	B2W0
	Read/Write	W							
	After reset	1	0	0	0	0	0	0	0
	Function	0: Disable 1: Enable	CS2 area selection 0: 16-Mbyte area 1: CS area	Chip select output waveform selection 00: For ROM/SRAM 01: Don't care 10: Don't care 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of waits 000: 2 waits 001: 1 wait 010: (1 + N) waits 011: 0 waits	100: Reserved 101: 3 waits 110: 4 waits 111: 8 waits	
B3CS (00C3H)	Bit symbol	B3E		B3OM1	B3OM0	B3BUS	B3W2	B3W1	B3W0
	Read/Write	W		W					
	After reset	0		0	0	0	0	0	0
	Function	0: Disable 1: Enable		Chip select output waveform selection 00: For ROM/SRAM 01: Don't care 10: Don't care 11: Don't care		Data bus width 0: 16 bits 1: 8 bits	Number of waits 000: 2 waits 001: 1 wait 010: (1 + N) waits 011: 0 waits	100: Reserved 101: 3 waits 110: 4 waits 111: 8 waits	
BEXCS (00C7H)	Bit symbol					BEXBUS	BEXW2	BEXW1	BEXW0
	Read/Write					W			
	After reset					0	0	0	0
	Function					Data bus width 0: 16 bits 1: 8 bits	Number of Waits 000: 2 waits 001: 1 wait 010: (1 + N) waits 011: 0 waits	100: Reserved 101: 3 waits 110: 4 waits 111: 8 waits	

Master enable bit

0	Disable CS area
1	Enable CS area

CS2 area selection

0	16-Mbyte area
1	Specified address area

Chip select output waveform selection

00	For ROM/SRAM
01	Don't care
10	Don't care
11	Don't care

Number of address area waits
(See section 3.7.2, "(3) Wait Control.")

Data bus width selection

0	16-bit data bus
1	8-bit data bus

Note: Read-modify-write instructions are prohibited for registers B0CS, B1CS, B2CS, B3CS and BEXCS.

Figure 3.7.5 Chip Select/Wait Control Register

(1) Master enable bits

Bit7 (<B0E>, <B1E>, <B2E> or <B3E>) of a chip select/wait control register is the master bit that is used to enable or disable settings for the corresponding address area. Writing “1” to this bit enables the settings. Reset disables (Sets to “0”) <B0E>, <B1E> and <B3E>, and enables (sets to “1”) <B2E>. This enables area CS2 only.

(2) Data bus width selection

Bit3 (<B0BUS>, <B1BUS>, <B2BUS>, <B3BUS> or <BEXBUS>) of a chip select/wait control register specifies the width of the data bus. This bit should be set to “0” when memory is to be accessed using a 16-bit data bus and to “1” when an 8-bit data bus is to be used.

This process of changing the data bus width according to the address being accessed is known as “dynamic bus sizing”. For details of this bus operation see Table 3.7.2.

Table 3.7.2 Dynamic Bus Sizing

Operand Data Bus Width	Operand Start Address	Memory Data Bus Width	CPU Address	CPU Data	
				D15 to D8	D7 to D0
8 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
		16 bits	2n + 0	xxxxx	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
		16 bits	2n + 1	b7 to b0	xxxxx
16 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
		16 bits	2n + 0	b15 to b8	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	xxxxx	b15 to b8
32 bits	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
			2n + 2	xxxxx	b23 to b16
			2n + 3	xxxxx	b31 to b24
		16 bits	2n + 0	b15 to b8	b7 to b0
			2n + 2	b31 to b24	b23 to b16
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
			2n + 3	xxxxx	b23 to b16
			2n + 4	xxxxx	b31 to b24
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	b23 to b16	b15 to b8
			2n + 4	xxxxx	b31 to b24

Note: “xxxxx” indicates that the input data from these bits are ignored during a read. During a write, indicates that the bus for these bits goes too high impedance; also, that the write strobe signal for the bus remains inactive.

(3) Wait control

Bits 0 to 2 (<B0W0:2>, <B1W0:2>, <B2W0:2>, <B3W0:2>, <BEXW0:2>) of a chip select/wait control register specify the number of waits that are to be inserted when the corresponding memory area is accessed.

The following types of wait operation can be specified using these bits. Bit settings other than those listed in the table should not be made.

Table 3.7.3 Wait Operation Setting

<BxW2:0>	Number of Waits	Wait Operation
000	2	Inserts a wait of 2 states.
001	1	Inserts a wait of 1 state.
010	(1 + N)	Same operation with 1 wait because of nothing $\overline{\text{WAIT}}$ pin.
011	0	Ends the bus cycle without a wait.
100	Reserved	Invalid setting
101	3	Inserts a wait of 3 states.
110	4	Inserts a wait of 4 states.
111	8	Inserts a wait of 8 states.

Note: A Reset sets these bits to “000” (2 waits).

(4) Bus width and wait control for an area other than CS0 to CS3

The chip select/wait control register BEXCS controls the bus width and number of waits when memory locations that are not in one of the four user-specified address areas (CS0 to CS3) are accessed. The BEXCS register settings are always enabled for areas other than CS0 to CS3.

(5) Selecting 16-Mbyte area/specified address area

Setting B2CS<B2M> (Bit6 of the chip select/wait control register for CS2) to “0” designates the 16-Mbyte areas 002000H to FF3FFFH as the CS2 area. Setting B2CS<B2M> to “1” designates the address area specified by the start address register MSAR2 and the address mask register MAMR2 as CS2 (e.g., if B2CS<B2M> = 1, CS2 is specified in the same manner as CS0, CS1 and CS3 are).

A Reset clears this bit to “0”, specifying CS2 as a 16-Mbytes address area.

(6) Procedure for setting chip select/wait control

When using the chip select/wait control function, set the registers in the following order:

a. Set the Memory Start Address Registers MSAR0 to MSAR3.
   Set the start addresses for CS0 to CS3.

b. Set the Memory Address Mask Registers MAMR0 to MAMR3.
   Set the sizes of CS0 to CS3.

c. Set the chip select/wait control registers B0CS to B3CS.
   Set the chip select output waveform, data bus width, number of waits and master enable/disable status for CS0 to CS3 areas.

The $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ pins can also function as pins P40 to P42. To output a chip select signal using one of these pins, set the corresponding bit in the port 4 function register P4FC and port 4 control register P4CR to "1".

If a CS0 to CS3 address is specified which is actually an internal I/O, RAM and ROM area address, the CPU accesses the internal address area and no chip select signal is output on any of the $\overline{\text{CS0}}$ to $\overline{\text{CS2}}$ pins.

Example:

In this example CS0 is set to be the 64-Kbyte area 010000H to 01FFFFH. The bus width is set to 16 bits and the number of waits is cleared to 0.

MSAR0 = 01H	Start address: 010000H
MAMR0 = 07H	Address area: 64 Kbytes
B0CS = 83H	ROM/SRAM, 16-bit data bus, zero waits, CS0 area settings enabled

3.7.3 Connecting External Memory

Figure 3.7.6 shows an example of how to connect external memory to the TMP91CP27.

In this example the ROM is connected using a 16-bit bus. The RAM and I/O are connected using an 8-bit bus.

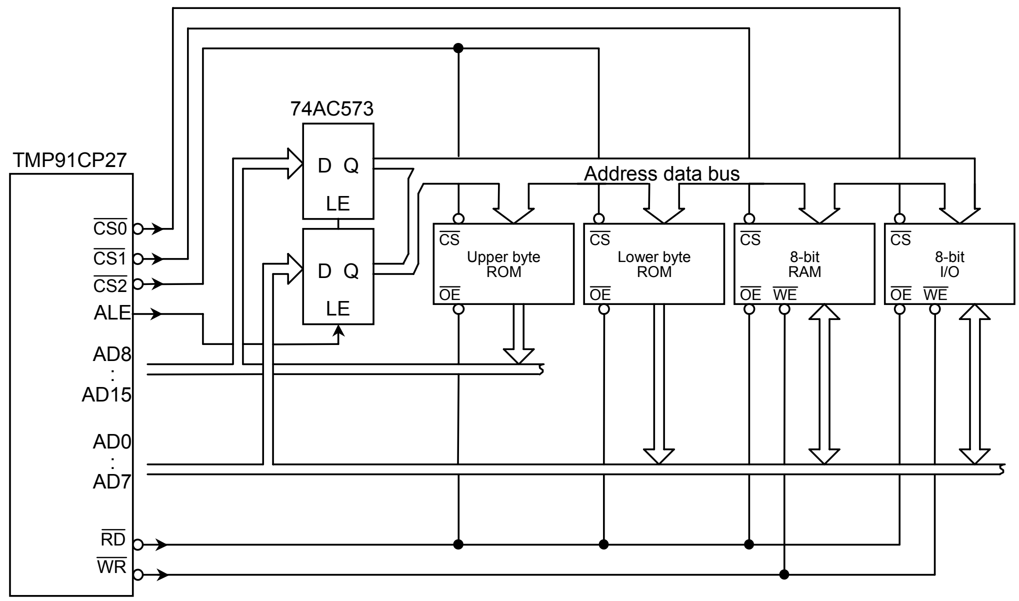


Figure 3.7.6 Example of External Memory Connection

(ROM uses 16-bit bus, RAM and I/O uses 8-bit bus)

A reset clears all bits of the port 4 control register P4CR and the port 4 function register P4FC to “0” and disables output of the CS signal. To output the CS signal, the appropriate bit must be set P4CR to “1” after set P4FC to “1”.

3.8 8-Bit Timers (TMRA)

The TMP91CP27 features 6 channels (TMRA0 to TMRA5) built-in 8-bit timers.

These timers are paired into 3 modules: TMRA01, TMRA23 and TMRA45. Each module consists of 2 channels and can operate in any of the following 4 operating modes.

- 8-bit interval timer mode
- 16-bit interval timer mode
- 8-bit programmable square wave pulse generation output mode (PPG: Variable duty cycle with variable period)
- 8-bit pulse width modulation output mode (PWM: Variable duty cycle with constant period)

Figure 3.8.1 to Figure 3.8.3 show block diagrams for TMRA01, TMRA23 and TMRA45.

Each channel consists of an 8-bit up counter, an 8-bit comparator and an 8-bit timer register. In addition, a timer flip-flop and a prescaler are provided for each pair of channels.

The operation mode and timer flip-flops are controlled by 5 bytes registers (SFRs: Special function registers).

Each of the three modules (TMRA01, TMRA23 and TMRA45) can be operated independently. All modules operate in the same manner; hence only the operation of TMRA01 is explained here.

Table 3.8.1 Registers and Pins for Each Module

Specification \ Module		TMRA01	TMRA23	TMRA45
External pins	Input pin for external clock	TA0IN (Shared with P70)	None	TA4IN (Shared with P73)
	Output pin for timer flip-flop	TA1OUT (Shared with P71)	TA3OUT (Shared with P72)	TA5OUT (Shared with P74)
SFR name (Address)	Timer RUN register	TA01RUN (0100H)	TA23RUN (0108H)	TA45RUN (0110H)
	Timer register	TA0REG (0102H) TA1REG (0103H)	TA2REG (010AH) TA3REG (010BH)	TA4REG (0112H) TA5REG (0113H)
	Timer mode register	TA01MOD (0104H)	TA23MOD (010CH)	TA45MOD (0114H)
	Timer flop-flop control register	TA1FFCR (0105H)	TA3FFCR (010DH)	TA5FFCR (0115H)

3.8.1 Block Diagram

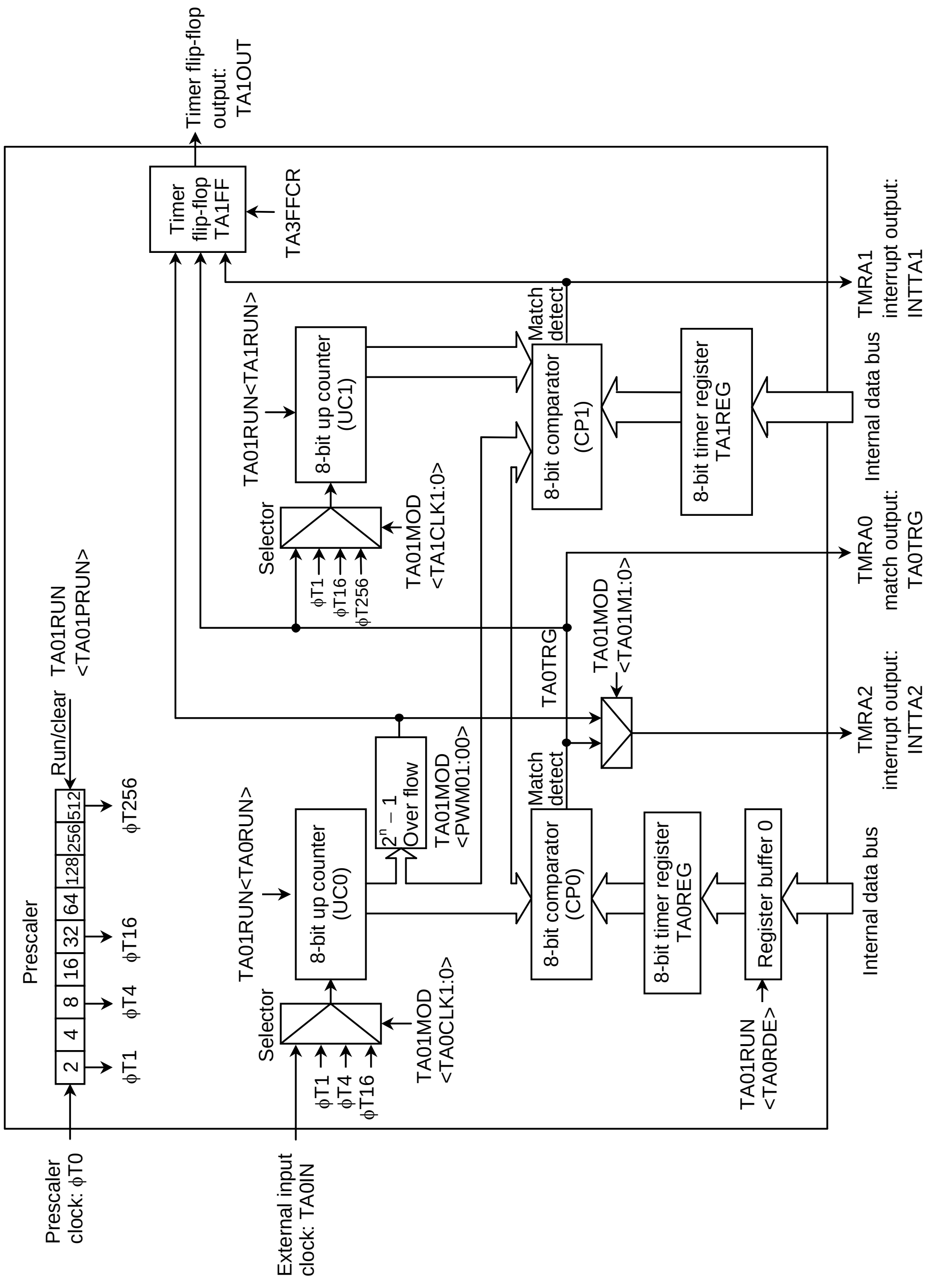


Figure 3.8.1 Block Diagram of TMRA01

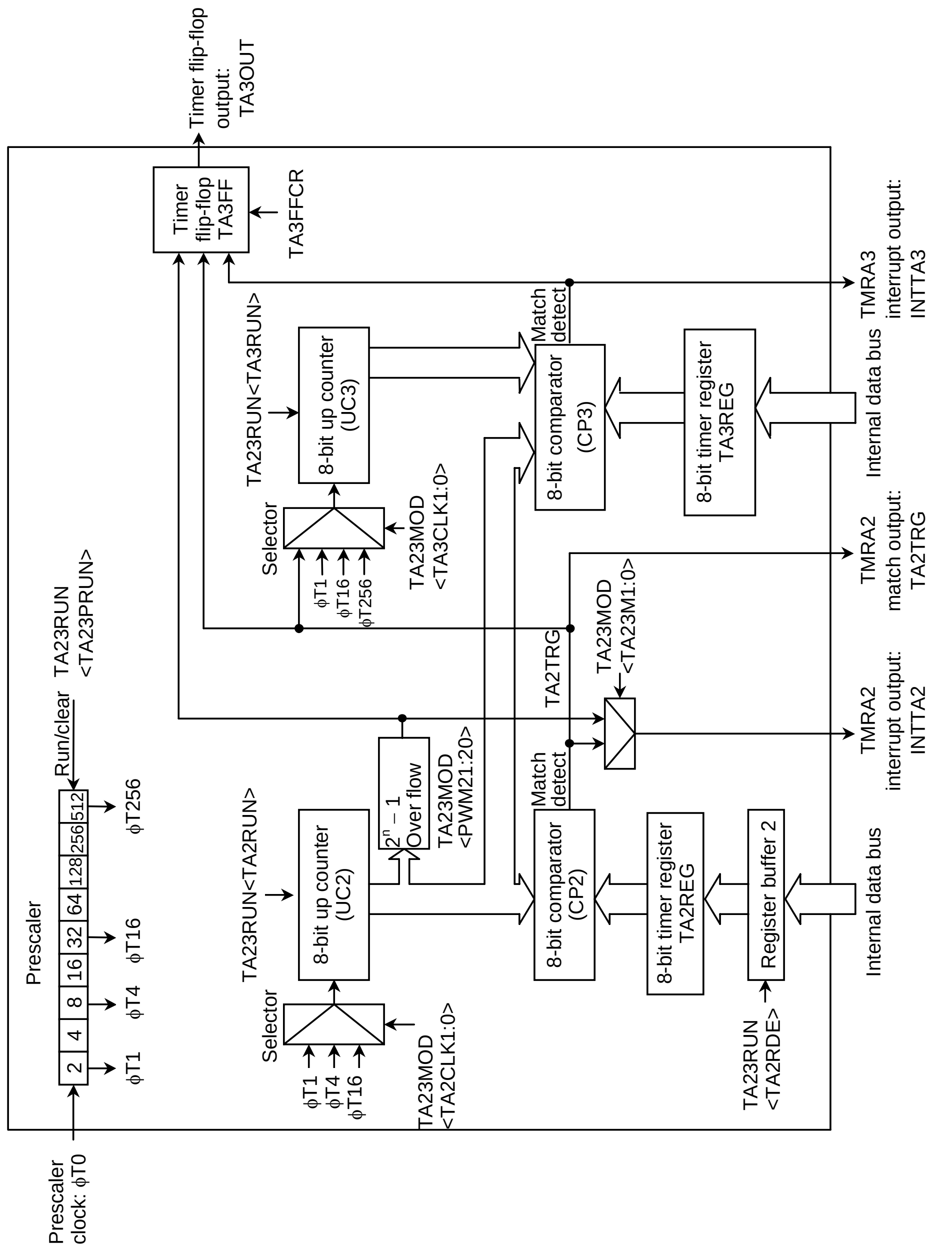


Figure 3.8.2 Block Diagram of TMRA23

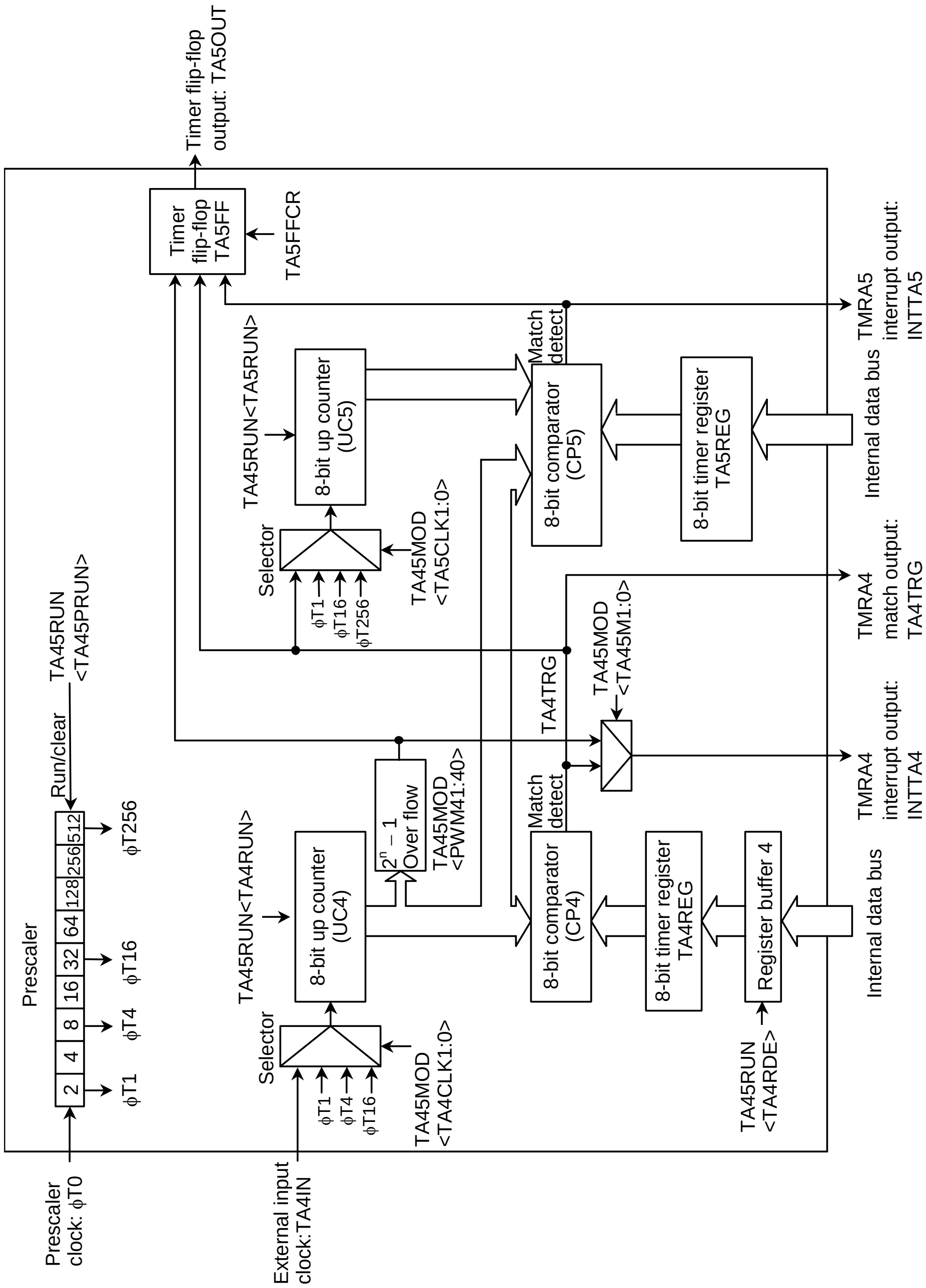


Figure 3.8.3 Block Diagram of TMRA45

3.8.2 Operation of Each Circuit

(1) Prescalers

A 9-bit prescaler generates the input clock to TMRA01.

The ϕT0 as the input clock to prescaler is a clock divided by 4, which selected using the prescaler clock selection register SYSCR0<PRCK1:0>.

The prescaler's operation can be controlled using TA01RUN<TA0PRUN> in the timer control register. Setting <TA0PRUN> to "1" starts the count; setting <TA0PRUN> to "0" clears the prescaler to zero and stops operation. Table 3.8.2 shows the various prescaler output clock resolutions.

Table 3.8.2 Prescaler Output Clock Resolution

@fc = 16MHz, fs = 32.768kHz

System Clock Selection <SYSCK>	Prescaler Clock Selection <PRCK1:0>	Clock Gear Value <GEAR2:0>	Prescaler Output Clock Resolution			
			ϕT1	ϕT4	ϕT16	ϕT256
1 (fs)	00 (f_{FPH})	XXX	fs/2^3 (244 μs)	fs/2^5 (977 μs)	fs/2^7 (3.9 ms)	fs/2^{11} (62.5 ms)
0 (fc)		000 (fc)	fc/2^3 (0.5 μs)	fc/2^5 (2.0 μs)	fc/2^7 (8.0 μs)	fc/2^{11} (128 μs)
		001 (fc/2)	fc/2^4 (1.0 μs)	fc/2^6 (4.0 μs)	fc/2^8 (16 μs)	fc/2^{12} (256 μs)
		010 (fc/4)	fc/2^5 (2.0 μs)	fc/2^7 (8.0 μs)	fc/2^9 (32 μs)	fc/2^{13} (512 μs)
		011 (fc/8)	fc/2^6 (4.0 μs)	fc/2^8 (16 μs)	fc/2^{10} (64 μs)	fc/2^{14} (1024 μs)
		100 (fc/16)	fc/2^7 (8.0 μs)	fc/2^9 (32 μs)	fc/2^{11} (128 μs)	fc/2^{15} (2048 μs)
	10 (fc/16 clocks)	XXX	fc/2^7 (8.0 μs)	fc/2^9 (32 μs)	fc/2^{11} (128 μs)	fc/2^{15} (2048 μs)

xxx: Don't care

(2) Up counters (UC0 and UC1)

These are 8-bit binary counters which count up the input clock pulses for the clock specified by TA01MOD.

The input clock for UC0 is selectable and can be either the external clock input via the TA0IN pin or one of the three internal clocks ϕT1, ϕT4, and ϕT16. The clock setting is specified by the value set in TA01MOD<TA0CLK1:0>.

The input clock for UC1 depends on the operation mode. In 16-bit timer mode, the overflow output from UC0 is used as the input clock. In any mode other than 16-bit timer mode, the input clock is selectable and can either be one of the internal clocks ϕT1, ϕT16, and ϕT256, or the comparator output (The match detection signal) from TMRA0 by setting TA01MOD<TA1CLK1:0>.

For each interval timer the timer operation control register bits TA01RUN<TA0RUN> and TA01RUN<TA1RUN> can be used to stop and clear the up counters and to control their count. A reset clears both up counters, stopping the timers.

(3) Timer registers (TA0REG and TA1REG)

These are 8-bit registers that can be used to set a time interval. When the value set in the timer register TA0REG or TA1REG matches the value in the corresponding up counter, the comparator match detect signal goes Active. If the value set in the timer register is 00H, the signal goes Active when the up counter overflows.

The TA0REG are double buffer structure, each of which makes a pair with register buffer.

The setting of the bit TA01RUN<TA0RDE> determines whether TA0REG's double buffer structure is enabled or disabled. It is disabled if <TA0RDE> = "0" and enabled if <TA0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer to the timer register when a $2^n - 1$ overflow occurs in PWM mode, or at the start of the PPG cycle in PPG mode. Hence the double buffer cannot be used in timer mode.

A Reset initializes <TA0RDE> to "0", disabling the double buffer. To use the double buffer, write data to the timer register, set <TA0RDE> to "1", and write the following data to the register buffer. Figure 3.8.4 shows the configuration of TA0REG.

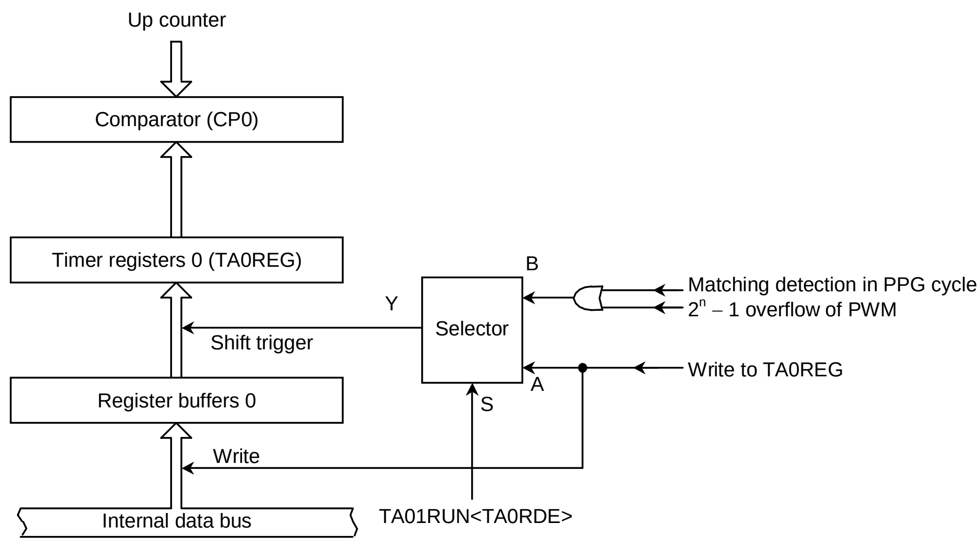


Figure 3.8.4 Configuration of Timer Register 0 (TA0REG)

Note: The same memory address is allocated to the timer register and the register buffer when write data to TA0REG. When <TA0RDE> = 0, the same value is written to the register buffer and the timer register; when <TA0RDE> = 1, only the register buffer is written to.

The address of each timer register is as follows.

TA0REG: 000102H TA1REG: 000103H

TA2REG: 00010AH TA3REG: 00010BH

TA4REG: 000112H TA5REG: 000113H

All these registers are write only and cannot be read.

(4) Comparator (CP0 and CP1)

The comparator compares the value in an up counter with the value set in a timer register. If they match, the up counter is cleared to zero and an interrupt signal (INTTA0 or INTTA1) is generated. If timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

(5) Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detects signal (8-bit comparator output) of each interval timer.

Whether inversion is enabled or disabled is determined by the setting of the bit TA1FFCR<TAFF1IE> in the Timer Flip-Flop Control Register.

A Reset clears the value of TA1FF to “0”.

Programming “01” or “10” to TA1FFCR<TAFF1C1:0> sets TA1FF to 1 or 0. Programming “00” to these bits inverts the value of TA1FF (This is known as software inversion).

The TA1FF signal is output via the TA1OUT pin (Concurrent with P71). When this pin is used as the timer output, the timer flip-flop should be set beforehand using the Port 7 relation registers P7CR and P7FC.

3.8.3 SFR

TMRA01 Run Register

		7	6	5	4	3	2	1	0
TA01RUN (0100H)	Bit symbol	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
	Read/Write	R/W				R/W			
	After reset	0				0	0	0	0
	Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	8-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		

TA0REG double buffer control

0	Disable
1	Enable

Timer run/stop control

0	Stop and clear
1	Run (Count up)

I2TA01: Operation in IDLE2 mode
TA01PRUN: Run prescaler
TA1RUN: Run TMRA1
TA0RUN: Run TMRA0

Note: The values of bits 4, 5 and 6 of TA01RUN are undefined when read.

TMRA23 Run Register

		7	6	5	4	3	2	1	0
TA23RUN (0108H)	Bit symbol	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
	Read/Write	R/W				R/W			
	After reset	0				0	0	0	0
	Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	8-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		

TA2REG double buffer control

0	Disable
1	Enable

Timer run/stop control

0	Stop and clear
1	Run (Count up)

I2TA23: Operation in IDLE2 mode
TA23PRUN: Run prescaler
TA3RUN: Run TMRA3
TA2RUN: Run TMRA2

Note: The values of bits 4, 5 and 6 of TA23RUN are undefined when read.

Figure 3.8.5 Register for TMRA

TMRA45 Run Register

		7	6	5	4	3	2	1	0
TA45RUN (0110H)	Bit symbol	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
	Read/Write	R/W				R/W			
	After reset	0				0	0	0	0
	Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	8-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		

TA4REG double buffer control

0	Disable
1	Enable

Count operate

0	Stop and clear
1	Run (Count up)

I2TA45: Operation in IDLE2 mode
TA45PRUN: Run prescaler
TA5RUN: Run TMRA5
TA4RUN: Run TMRA4

Note: The values of bits 4, 5 and 6 of TA45RUN are undefined when read.

Figure 3.8.6 Register for TMRA

TMRA01 Mode Register

TA01MOD (0104H)		7	6	5	4	3	2	1	0
	Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock for TMRA1 00: TA0TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA0 00: TA0IN pin 01: φT1 10: φT4 11: φT16	

TMRA0 source clock selection

00	TA0IN (External input)
01	φT1 (Prescaler)
10	φT4 (Prescaler)
11	φT16 (Prescaler)

TMRA1 source clock selection

	TA01MOD <TA01M1:0> ≠ 01	TA01MOD <TA01M1:0> = 01
00	Comparator output from TMRA0	Overflow output from TMRA0 (16-bit timer mode)
01	φT1	
10	φT16	
11	φT256	

PWM cycle selection

00	Reserved
01	$(2^6 - 1) \times$ Clock source
10	$(2^7 - 1) \times$ Clock source
11	$(2^8 - 1) \times$ Clock source

TMRA01 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA0) + 8-bit timer (TMRA1)

Figure 3.8.7 Register for TMRA

TMRA23 Mode Register

		7	6	5	4	3	2	1	0
TA23MOD (010CH)	Bit symbol	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock for TMRA3 00: TA2TRG 01: φT1 10: φT16 11: φT256		Source clock for TMRA2 00: Reserved 01: φT1 10: φT4 11: φT16	

TMRA2 source clock selection

00	Do not set
01	φT1 (Prescaler)
10	φT4 (Prescaler)
11	φT16 (Prescaler)

TMRA3 source clock selection

	TA23MOD <TA23M1:0> ≠ 01	TA23MOD <TA23M1:0> = 01
00	Comparator output from TMRA2	Overflow output from TMRA2 (16-bit timer mode)
01	φT1	
10	φT16	
11	φT256	

PWM cycle selection

00	Reserved
01	$(2^6 - 1) \times$ Clock source
10	$(2^7 - 1) \times$ Clock source
11	$(2^8 - 1) \times$ Clock source

TMRA23 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA2) + 8-bit timer (TMRA3)

Figure 3.8.8 Register for TMRA

TMRA45 Mode Register

TA45MOD (0114H)

	7	6	5	4	3	2	1	0
Bit symbol	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8-1		Source clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA4 00: TA4IN pin 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA4 source clock selection

00	TA4IN (External input)
01	ϕT1 (Prescaler)
10	ϕT4 (Prescaler)
11	ϕT16 (Prescaler)

TMRA5 source clock selection

	TA45MOD <TA45M1:0> $\neq$ 01	TA45MOD <TA45M1:0> $=$ 01
00	Comparator output from TMRA4	Overflow output from TMRA4 (16-bit timer mode)
01	ϕT1	
10	ϕT16	
11	ϕT256	

PWM cycle selection

00	Reserved
01	$(2^6-1) \times$ Clock source
10	$(2^7-1) \times$ Clock source
11	$(2^8-1) \times$ Clock source

TMRA45 operation mode selection

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG
11	8-bit PWM (TMRA4) + 8-bit timer (TMRA5)

Figure 3.8.9 Register for TMRA

TMRA1 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA1FFCR (0105H)	Bit symbol					TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
	Read/Write					R/W		R/W	
	After reset					1	1	0	0
Read-modify-write instruction is prohibited.	Function					00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don’t care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1: TMRA1

Inverse signal for timer flip-flop 1 (TA1FF)
(Don’t care except in 8-bit timer mode)

0	Inversion by TMRA0
1	Inversion by TMRA1

Inversion of TA1FF

0	Disabled
1	Enabled

Control of TA1FF

00	Inverts the value of TA1FF.
01	Sets TA1FF to “1”.
10	Clears TA1FF to “0”.
11	Don’t care

Note: The values of bits 4, 5, 6 and 7 of TA1FFCR are undefined when read.

Figure 3.8.10 Register for TMRA

TMRA3 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA3FFCR (010DH)	Bit symbol					TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
	Read/Write					R/W		R/W	
	After reset					1	1	0	0
Read-modify-write instruction is prohibited.	Function					00: Invert TA3FF 01: Set TA3FF 10: Clear TA3FF 11: Don't care		TA3FF control for inversion 0: Disable 1: Enable	TA3FF inversion select 0: TMRA2 1: TMRA3

Inverse signal for timer flip-flop 3 (TA3FF)
(Don't care except in 8-bit timer mode)

0	Inversion by TMRA2
1	Inversion by TMRA3

Inversion of TA3FF

0	Disabled
1	Enabled

Control of TA3FF

00	Inverts the value of TA3FF.
01	Sets TA3FF to "1".
10	Clears TA3FF to "0".
11	Don't care

Note: The values of bits 4, 5, 6 and 7 of TA3FFCR are undefined when read.

Figure 3.8.11 Register for 8-Bit Timer

TMRA5 Flip-Flop Control Register

		7	6	5	4	3	2	1	0
TA5FFCR (0115H)	Bit symbol					TAFF5C1	TAFF5C0	TAFF5IE	TAFF5IS
	Read/Write					R/W		R/W	
	After Reset					1	1	0	0
Read-modify-write instruction is prohibited.	Function					00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF control for inversion 0: Disable 1: Enable	TA5FF inversion select 0: TMRA4 1: TMRA5

Inverse signal for timer flip-flop 5 (TA5FF)
(Don't care except in 8-bit timer mode)

0	Inversion by TMRA4
1	Inversion by TMRA5

Inversion of TA5FF

0	Disabled
1	Enabled

Control of TA5FF

00	Inverts the value of TA5FF.
01	Sets TA5FF to "1".
10	Clears TA5FF to "0".
11	Don't care

Note: The values of bits 4, 5, 6 and 7 of TA5FFCR are undefined when read.

Figure 3.8.12 Register for TMRA

3.8.4 Operation in Each Mode

(1) 8-bit timer mode

Both TMRA0 and TMRA1 can be used independently as 8-bit interval timers.

When set function and counter data, be stopped operation of TMRA0 and TMRA1 registers beforehand.

a. Generating interrupts at a fixed interval (using TMRA1)

To generate interrupts at constant intervals using TMRA1 (INTTA1), first stop TMRA1 then set the operation mode, input clock and a cycle to TA01MOD and TA1REG register, respectively. Then, enable the interrupt INTTA1 and start TMRA1 counting.

Example: To generate an INTTA1 interrupt every 20 μs at fc = 16 MHz, set each register as follows:

* Clock state
- System clock: High frequency (fc)
- Clock gear: 1(fc)
- Prescaler clock: f_{FPH}

		7 (MSB)	6	5	4	3	2	1	0 (LSB)	
TA01RUN	←	–	–	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	←	0	0	X	X	0	1	X	X	Select 8-bit timer mode and select φT1 (0.5 μs at fc = 16 MHz) as the input clock.
TA1REG		0	0	1	0	1	0	0	0	Set TA1REG to 20 μs ÷ φT1 = 40 = 28H
INTETA01	←	X	1	0	1	–	–	–	–	Enable INTTA1 and set it to level 5.
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don't care, –: No change

Select the input clock using Table 3.8.2.

Note: The input clocks for TMRA0 and TMRA1 are different from as follows.
TMRA0: TA0IN input, φT1, φT4 or φT16
TMRA1: Comparator output from TMRA0, φT1, φT16 and φT256

b. Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 3.0 μs square wave pulse from the TA1OUT pin at fc = 16 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 may be used.

* Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

		7 (MSB)	6	5	4	3	2	1	0 (LSB)	
TA01RUN	←	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	←	0	0	X	X	0	1	–	–	Select 8-bit timer mode and select ϕT1 (0.5 μs at fc = 16 MHz) as the input clock.
TA1REG	←	0	0	0	0	0	0	1	1	Set the timer register to 3.0 μs ÷ ϕT1 ÷ 2 = 3
TA1FFCR	←	X	X	X	X	1	0	1	1	Clear TA1FF to "0" and set it to invert on the match detects signal from TMRA1.
P7CR	←	X	X	X	–	–	–	1	–	Set P71 to function as the TA1OUT pin.
P7FC	←	X	X	X	–	X	–	1	X	
TA01RUN	←	–	X	X	X	–	1	1	–	Start TMRA1 counting.

X: Don't care, –: No change

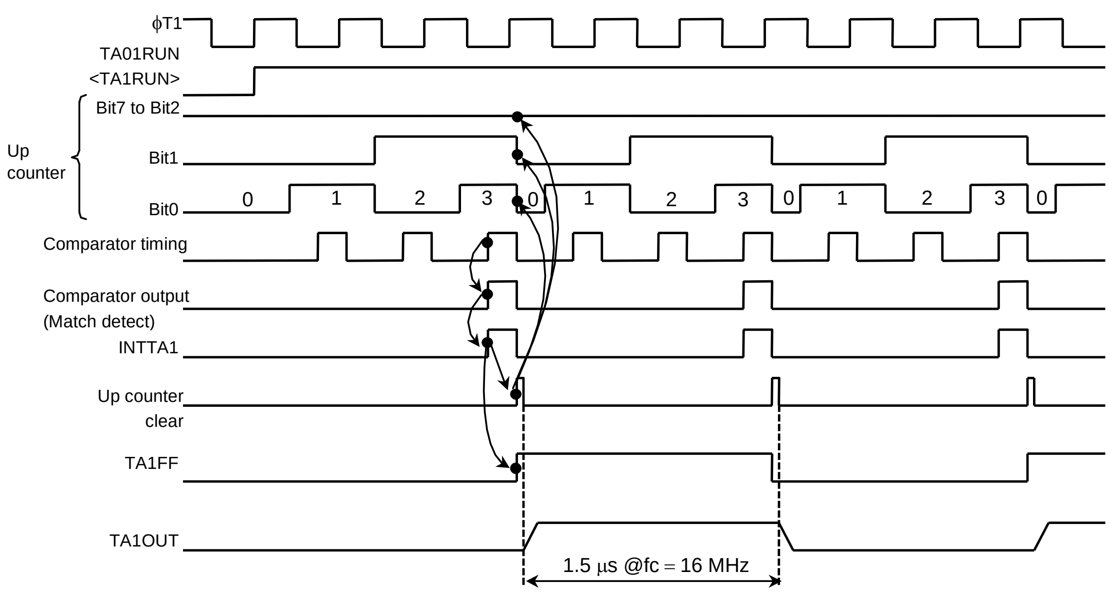


Figure 3.8.13 Square Wave Output Timing Chart (50% Duty)

c. Making TMRA1 count up on the match signal from the TMRA0 comparator

Select 8-bit timer mode and set the comparator output from TMRA0 to be the input clock to TMRA1.

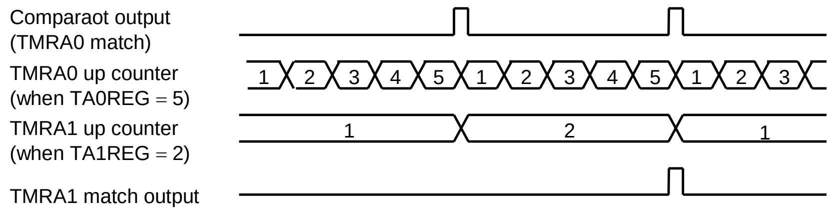


Figure 3.8.14 TMRA1 Count Up on Signal from TMRA0

(2) 16-bit timer mode

A 16-bit interval timer is configured by pairing the two 8-bit timers TMRA0 and TMRA1.

To make a 16-bit interval timer in which TMRA0 and TMRA1 are cascaded together, set TA01MOD<TA01M1:0> to "01".

In 16-bit timer mode, the overflow output from TMRA0 is used as the input clock for TMRA1, regardless of the value set in TA01MOD<TA1CLK1:0>. Table 3.8.2 shows the relationship between the timer (Interrupt) cycle and the input clock selection.

Timer interrupt cycle set lower 8 bits to TA0REG and set upper 8 bits to TA1REG. Please keep setting TA0REG first because setting data for TA0REG inhibit its compare function and setting data for TA1REG permit it.

Setting example: To generate an INTTA1 interrupt every 0.5 s at fc = 16 MHz, set the timer registers TA0REG and TA1REG as follows:

* Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

If ϕT16 (8.0 μs at 16 MHz) is used as the input clock for counting, set the following value in the registers:

0.5 s/8.0 μs = 62500 = F424H;

i.e. set TA1REG to F4H and TA0REG to 24H.

The comparator match signal is output from TMRA0 each time the up counter UC0 matches TA0REG, though the up counter UC0 is not be cleared and also INTTA0 is not generated.

In the case of the TMRA1 comparator, the match detect signal is output on each comparator pulse on which the values in the up counter UC1 and TA1REG match. When the match detect signal is output simultaneously from both the comparators TMRA0 and TMRA1, the up counters UC0 and UC1 are cleared to 0 and the interrupt INTTA1 is generated. Also, if inversion is enabled, the value of the timer flip-flop TA1FF is inverted.

Example: When TA1REG = 04H and TA0REG = 80H

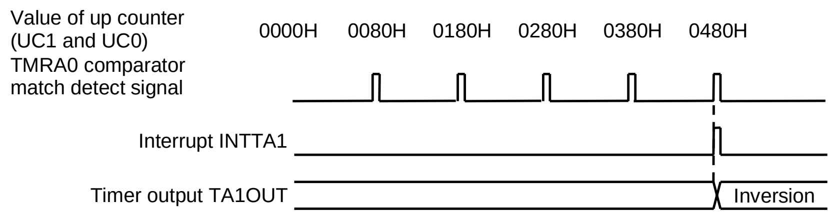


Figure 3.8.15 Timer Output by 16-Bit Timer Mode

(3) 8-bit PPG (Programmable pulse generation) output mode

Square wave pulses can be generated at any frequency and duty ratio by TMRA0. The output pulses may be active low or active high. In this mode TMRA1 cannot be used.

TMRA0 outputs pulses on the TA1OUT pin (shared with P71).

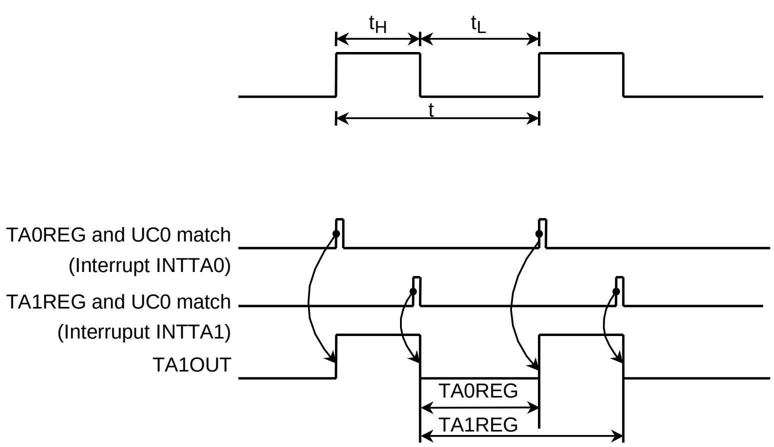


Figure 3.8.16 8-Bit PPG Output Waveforms

In this mode, a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC0) matches the value in one of the timer registers TA0REG or TA1REG.

The value set in TA0REG must be smaller than the value set in TA1REG.

Although the up counter for TMRA1 (UC1) is not used in this mode, TA01RUN<TA1RUN> should be set to "1", so that UC1 is set for counting.

Figure 3.8.17 shows a block diagram representing this mode.

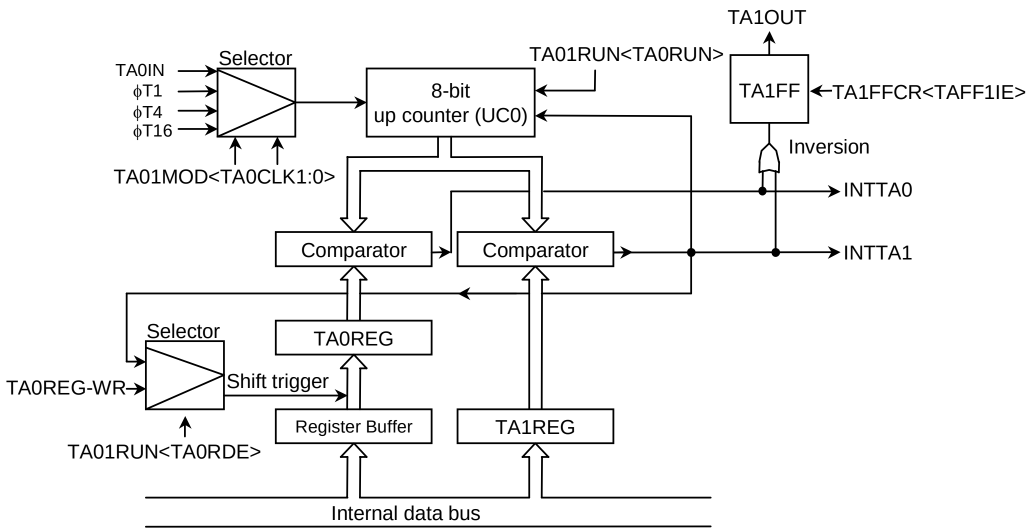


Figure 3.8.17 Block Diagram of 8-Bit PPG Output Mode

If the TA0REG double buffer is enabled in this mode, the value of the register buffer will be shifted into TA0REG each time TA1REG matches UC0.

Use of the double buffer facilitates the handling of low-duty waves (when duty is varied).

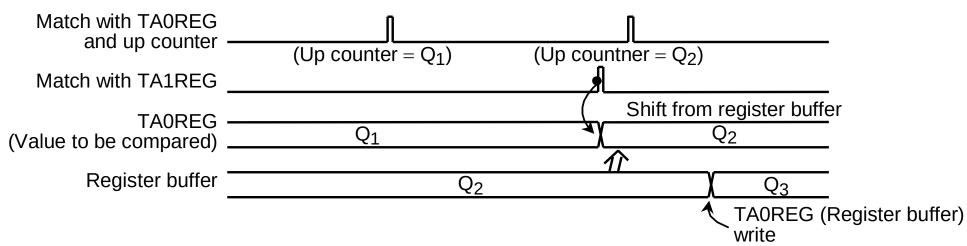
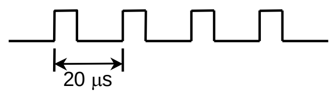


Figure 3.8.18 Operation of Register Buffer

Example: To generate 1/4-duty 50-kHz pulses (at fc = 16 MHz):



* Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

Calculate the value that should be set in the timer register.

To obtain a frequency of 50 kHz, the pulse cycle t should be: t = 1/50 kHz = 20 μs

ϕT1 = 0.5 μs (at 16 MHz);

20 μs/0.5 μs = 40

Therefore set TA1REG to 40 (28H)

The duty is to be set to 1/4: t × 1/4 = 20 μs × 1/4 = 5 μs

5 μs/0.5 μs = 10

Therefore, set TA0REG = 10 = 0AH.

		MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	←	0	X	X	X	–	0	0	0	Stop TMRA0 and TMRA1 and clear it to "0".
TA01MOD	←	1	0	X	X	X	X	0	1	Set the 8-bit PPG mode, and select ϕT1 as input clock.
TA0REG	←	0	0	0	0	1	0	1	0	Write 0AH
TA1REG	←	0	0	1	0	1	0	0	0	Write 28H
TA1FFCR	←	X	X	X	X	0	1	1	X	Set TA1FF, enabling inversion. Writing "10" provides negative logic pulse.
P7CR	←	X	X	X	–	–	–	1	–	Set P71 as the TA1OUT pin.
P7FC	←	X	X	X	–	X	–	1	X	Set P71 as the TA1OUT pin.
TA01RUN	←	1	X	X	X	–	1	1	1	Start TMRA0 and TMRA1 counting and enable double buffer.

X: Don't care, –: No change

(4) 8-bit PWM output mode

This mode is only valid for TMRA0. In this mode, a PWM pulse with the maximum resolution of 8 bits can be output.

When TMRA0 is used the PWM pulse is output on the TA1OUT pin (which is also used as P71). TMRA1 can also be used as an 8-bit timer.

The timer output is inverted when the up counter (UC0) matches the value set in the timer register TA0REG or when $2^n - 1$ counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01:00>). The up counter UC0 is cleared when $2^n - 1$ counter overflow occurs.

The following conditions must be satisfied before this PWM mode can be used.

Value set in TA0REG < Value set for $2^n - 1$ counter overflow

Value set in TA0REG ≠ 0

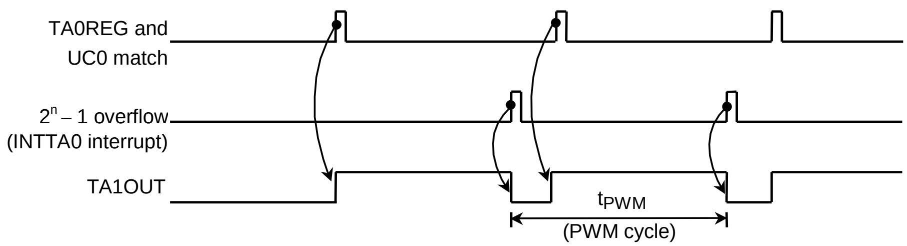


Figure 3.8.19 8-Bit PWM Output Wave Form

Figure 3.8.20 shows a block diagram representing this mode.

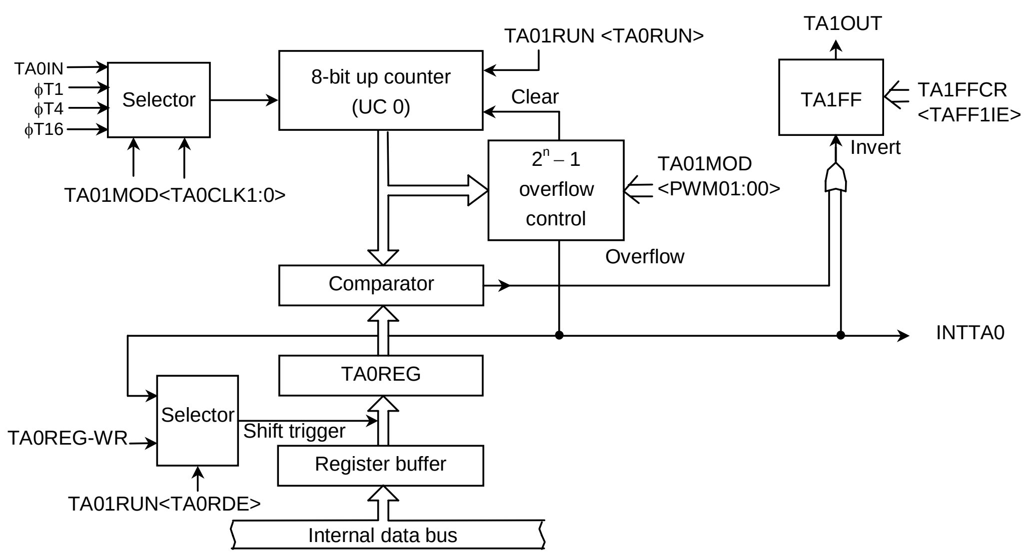


Figure 3.8.20 Block Diagram of 8-Bit PWM Output Mode

In this mode, the value of the register buffer will be shifted into TA0REG if $2^n - 1$ overflow is detected when the TA0REG double buffer is enabled.

Use of the double buffer facilitates the handling of low duty ratio waves.

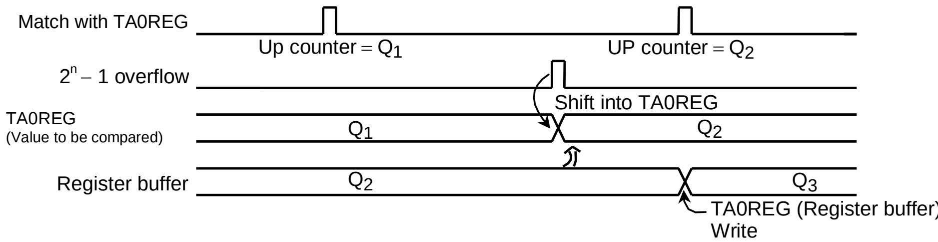
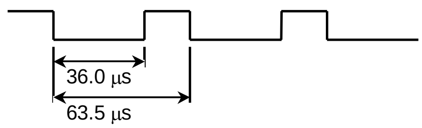


Figure 3.8.21 Operation of Register Buffer

Example: To output the following PWM waves on the TA1OUT pin at fc = 16 MHz:



* Clock state
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: f_{FPH}

To achieve a 63.5-μs PWM cycle by setting φT1 to 0.5 μs (at fc = 16 MHz):

$63.55\ \mu s / 0.5\ \mu s = 127 = 2^n - 1$

Therefore n should be set to 7.

Since the low-level period is 36.0 μs when φT1 = 0.5 μs,

set the following value for TA0REG:

$36.0\ \mu s / 0.5\ \mu s = 72 = 48H$

		7 (MSB)	6	5	4	3	2	1	0 (LSB)	
TA01RUN	←	–	X	X	X	–	–	–	0	Stop TMRA0 and clear it to 0.
TA01MOD	←	1	1	1	0	–	–	0	1	Select 8-bit PWM mode (cycle: $2^7 - 1$) and select φT1 as the input clock.
TA0REG	←	0	1	0	0	1	0	0	0	Write 48H.
TA1FFCR	←	X	X	X	X	1	0	1	X	Clear TA1FF to 0; enable the inversion.
P7CR	←	X	X	X	–	–	–	1	–	Set P71 to function as the TA1OUT pin.
P7FC	←	X	X	X	–	X	–	1	X	
TA01RUN	←	1	X	X	X	–	1	–	1	Start TMRA0 counting, and enable the double buffer.

X: Don't care, –: No change

Table 3.8.3 PWM Cycle

@ fc = 16 MHz, fs = 32.768 kHz

Select System Clock <SYSCK>	Select Prescaler Clock <PRCK1:0>	Gear Value <GEAR2:0>	PWM Cycle								
			$2^6 - 1$			$2^7 - 1$			$2^8 - 1$		
			ϕT1	ϕT4	ϕT16	ϕT1	ϕT4	ϕT16	ϕT1	ϕT4	ϕT16
1 (fs)	00 (f_{FPH})	XXX	15.4 ms	61.5 ms	246 ms	31.0 ms	124 ms	496 ms	62.3 ms	249 ms	996 ms
0 (fc)		000 (fc)	31.5 μs	126 μs	504 μs	63.5 μs	254 μs	1016 μs	127.5 μs	510 μs	2040 μs
		001 (fc/2)	63.0 μs	252 μs	1008 μs	127 μs	508 μs	2032 μs	255 μs	1020 μs	4080 μs
		010 (fc/4)	126 μs	504 μs	2016 μs	254 μs	1016 μs	4064 μs	510 μs	2040 μs	8160 μs
		011 (fc/8)	252 μs	1008 μs	4032 μs	508 μs	2032 μs	8128 μs	1020 μs	4080 μs	16.32 ms
		100 (fc/16)	504 μs	2016 μs	8064 μs	1016 μs	4064 μs	16.256 ms	2040 μs	8160 μs	32.64 ms
	10 (fc/16 clock)	XXX	504 μs	2016 μs	8064 μs	1016 μs	4064 μs	16.256 ms	2040 μs	8160 μs	32.64 ms

XXX: Don't care

(5) Settings for each mode

Table 3.8.4 shows the SFR settings for each mode.

Table 3.8.4 Timer Mode Setting Registers

Register Name	TA01MOD				TA1FFCR
<Bit symbol>	<TA01M1:0>	<PWM01:00>	<TA1CLK1:0>	<TA0CLK1:0>	<TAFF1IS>
Function	Timer Mode	PWM Cycle	Upper Timer Input Clock	Lower Timer Input Clock	Timer F/F Invert Signal Select
8-bit timer × 2 channels	00	–	Lower timer match, ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
16-bit timer mode	01	–	–	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PPG × 1 channel	10	–	–	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PWM × 1 channel	11	$2^6 - 1$, $2^7 - 1$, $2^8 - 1$ (01, 10, 11)	–	External clock, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit timer × 1 channel	11	–	ϕT1, ϕT16, ϕT256 (01, 10, 11)	–	Output disabled

–: Don't care

3.9 16-Bit Timer/Event Counters (TMRB)

The TMP91CP27 contains one multifunctional 16-bit timer/event counter (TMRB0) which have the following operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable square wave pulse generation output mode (PPG: Variable duty cycle with variable period)

Can be used following operation modes by capture function:

- Frequency measurement mode
- Pulse width measurement mode
- Time differential measurement mode

Figure 3.9.1 show block diagram of TMRB0. Timer/event counter consists of a 16-bit up counter, two 16-bit timer registers (One of them with a double-buffer structure), two 16-bit capture registers, two comparators, a capture input controller, a timer flip-flop and a control circuit.

Timer/Event counter is controlled by 11-byte control register (SFR).

Table 3.9.1 Registers and Pins for TMRB0

Spec \ Channel		TMRB0
External pin	External clock/ capture trigger input pin	TB0IN0 (Shared with P80) TB0IN1 (Shared with P81)
	Timer flip-flop output pin	TB0OUT0 (Shared with P82) TB0OUT1 (Shared with P83)
SFR name (Address)	Timer RUN register	TB0RUN (0180H)
	Timer mode register	TB0MOD (0182H)
	Timer flip-flop control register	TB0FFCR (0183H)
	Timer register	TB0RG0L (0188H) TB0RG0H (0189H) TB0RG1L (018AH) TB0RG1H (018BH)
	Capture register	TB0CP0L (018CH) TB0CP0H (018DH) TB0CP1L (018EH) TB0CP1H (018FH)

3.9.1 Block Diagram of TMRB0

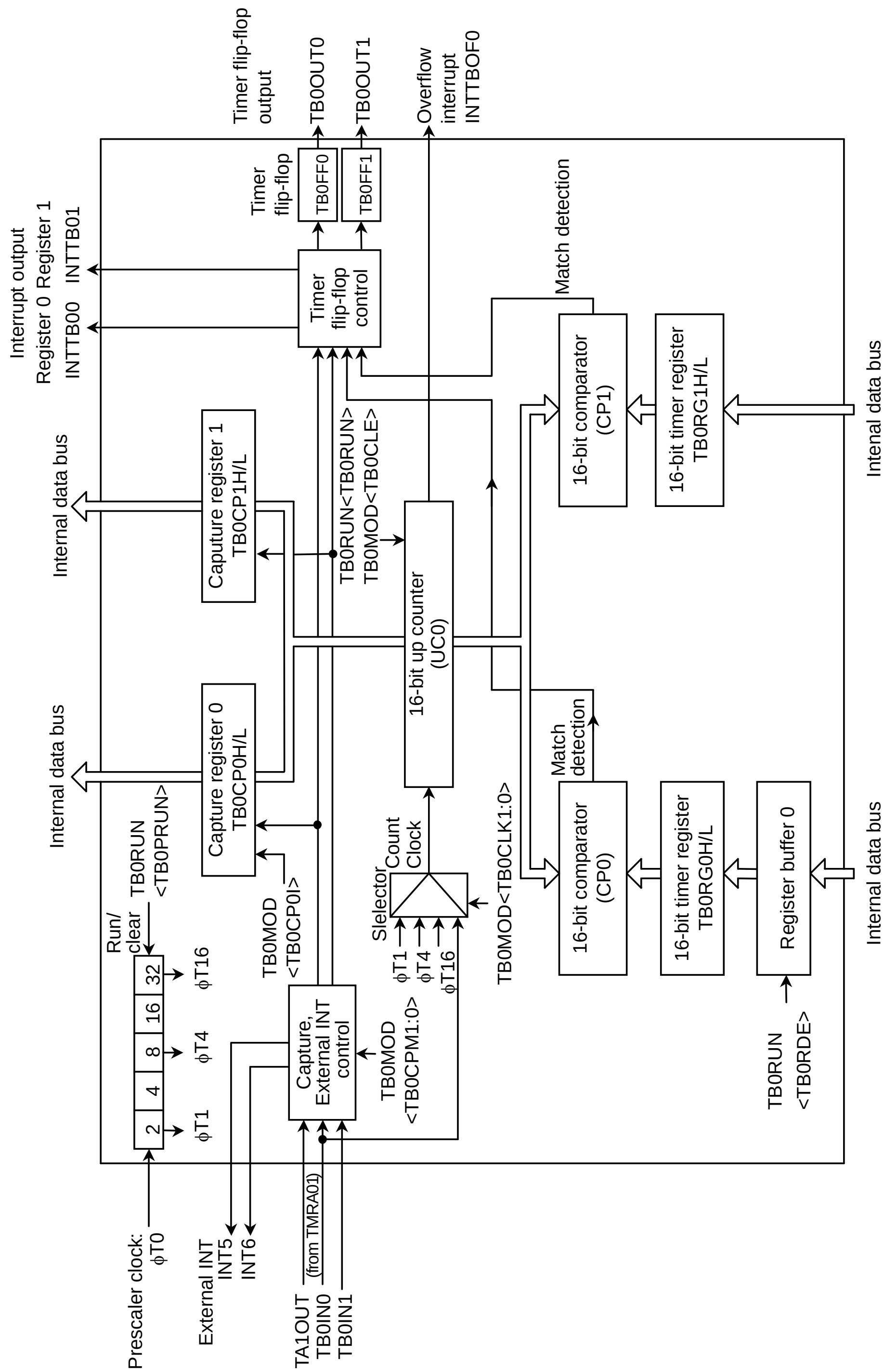


Figure 3.9.1 Block Diagram of TMRB0

3.9.2 Operation of Each Circuit

(1) Prescaler

The 5-bit prescaler generates the source clock for TMRB0. The prescaler clock (φT0) is divided clock (divided by 4) from selected clock by the register SYSCR0<PRCK1:0> of clock gear.

This prescaler can be started or stopped using TB0RUN<TB0PRUN>. Counting starts when <TB0PRUN> is set to 1; the prescaler is cleared to zero and stops operation when <TB0PRUN> is cleared to 0.

Table 3.9.2 show prescaler output clock resolution.

Table 3.9.2 Prescaler Output Clock Resolution

@fc = 16 MHz, fs = 32.768 KHz

System Clock Selection <SYSCK>	Prescaler Clock Selection <PRCK1:0>	Clock Gear Value <GEAR2:0>	Prescaler Output Clock Resolution		
			φT1	φT4	φT16
1 (fs)	00 (f_{FPH})	XXX	fs/2^3 (244 μs)	fs/2^5 (977 μs)	fs/2^7 (3.9 ms)
0 (fc)		000 (fc)	fc/2^3 (0.5 μs)	fc/2^5 (2.0 μs)	fc/2^7 (8.0 μs)
		001 (fc/2)	fc/2^4 (1.0 μs)	fc/2^6 (4.0 μs)	fc/2^8 (16.0 μs)
		010 (fc/4)	fc/2^5 (2.0 μs)	fc/2^7 (8.0 μs)	fc/2^9 (32.0 μs)
		011 (fc/8)	fc/2^6 (4.0 μs)	fc/2^8 (16.0 μs)	fc/2^{10} (64.0 μs)
		100 (fc/16)	fc/2^7 (8.0 μs)	fc/2^9 (32.0 μs)	fc/2^{11} (128 μs)
	10 (Note) (fc/16 clock)	XXX	fc/2^7 (8.0 μs)	fc/2^9 (32.0 μs)	fc/2^{11} (128 μs)

XXX: Don't care

(2) Up counter (UC0)

UC0 is a 16-bit binary counter which counts up according to input from the clock specified by TB0MOD<TB0CLK1:0> register.

As the input clock, one of the prescaler internal clocks φT1, φT4 and φT16 or an external clock from TB0IN0 pin can be selected. Counting or stopping and clearing of the counter is controlled by timer operation control register TB0RUN<TB0RUN>.

When clearing is enabled, the up counter UC0 will be cleared to zero each time its value matches the value in the timer register TB0RG1H/L. Clearing can be enabled or disabled using TB0MOD<TB0CLE>.

If clearing is disabled, the counter operates as a free-running counter.

A timer overflow interrupt (INTTBOF0) is generated when UC0 overflow occurs.

(3) Timer registers (TB0RG0H/L and TB0RG1H/L)

These two 16-bit registers are used to set the interval time. When the value in the up counter UC0 matches set value of timer register, the comparator match detect signal will be active.

Setting data for both upper and lower timer registers are always needed. For example, either using 2-byte data transfer instruction or using 1-byte date transfer instruction twice for lower 8 bits and upper 8 bits in order.

The TB0RG0 timer register has a double-buffer structure, which is paired with register buffer 0. The timer control register TB0RUN<TB0RDE> control whether the double buffer structure should be enabled or disabled: it is disabled when <TB0RDE> = 0, and enabled when <TB0RDE> = 1.

When the double buffer is enabled, data is transferred from the register buffer to the timer register when the values in the up counter (UC0) and the timer register TB0RG1 match.

After a Reset, TB0RG0 and TB0RG1 are undefined. To use the 16-bit timer after reset, data should be written beforehand.

When reset, <TB0RDE> is initialized to 0, whereby the double buffer is disabled. To use the double buffer, write data to the timer register, set <TB0RDE> to 1, then write following data to the register buffer.

TB0RG0 and the register buffer are allocated to the same memory address 0188H/0189H. When <TB0RDE> = 0, same value will be written to both the timer register and register buffer. When <TB0RDE> = 1, the value is written into only the register buffer.

Therefore, when write initial value to timer register, set register buffer to disable.

The addresses of the timer registers are as follows:

TMRB0

TB0RG0		TB0RG1	
Upper 8-bit (TB0RG0H)	Lower 8-bit (TB0RG0L)	Upper 8-bit (TB0RG1H)	Lower 8-bit (TB0RG1L)
000189H	000188H	00018BH	00018AH

The TB0RG0 to TB0RG1 are write-only registers and thus cannot be read.

(4) Capture registers (TB0CP0H/L, TB0CP1H/L)

These 16-bit registers are used to latch the values of the up counters.

Data in the capture register should be read all 16 bits. For example, using 2-byte data load instruction or using 1-byte date load instruction twice for lower 8 bits and upper 8 bits in order.

The addresses of the capture registers are as follows:

TMRB0

TB0CP0		TB0CP1	
Upper 8-bit (TB0CP0H)	Lower 8-bit (TB0CP0L)	Upper 8-bit (TB0CP1H)	Lower 8-bit (TB0CP1L)
00018DH	00018CH	00018FH	00018EH

The TB0CP0 to TB0CP1 are read-only registers and thus cannot be read.

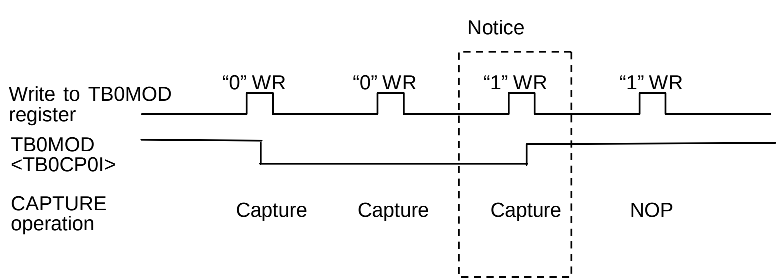
(5) Capture, external interrupt control

This circuit controls the timing to latch the value of up counter UC0 into TB0CP0, TB0CP1 and control generation of external interrupt. The latch timing of capture register and selection of edge for external interrupt is set in TB0MOD<TB0CPM1:0>.

The edge of external interrupt INT6 is fixed to rising edge.

Besides, the value of up counter can be loaded into a capture registers by software. Whenever 0 is written to TB0MOD<TB0CP0I>, the current value in the up counter is loaded into capture register TB0CP0. It is necessary to keep the prescaler in run mode (e.g., TB0RUN<TB0PRUN> must be held at a value of 1).

Note: As described above, whenever 0 is written to TB0MOD<TB0CP0I>, the current value in the up counter is loaded into capture register TB0CP0. However, note that the current value in the up counter is also loaded into capture register TB0CP0 when 1 is written to TB0MOD<TB0CP0I> while this bit is holding 0.



(6) Comparators (CP0 and CP1)

CP0 and CP1 are 16-bit comparators which compare the value in the up counter UC0 value with the value set of TB0RG0 or TB0RG1 respectively, in order to detect a match. If a match is detected, the comparators generate an interrupt (INTTB00 or INTTB01 respectively).

(7) Timer flip-flops (TB0FF0 and TB0FF1)

These flip-flops are inverted by the match detect signals from the comparators and the latch signals to the capture registers. Inversion can be enabled and disabled for each element using TB0FFCR<TB0C1T1, TB0C0T1, TB0E1T1, TB0E0T1>.

After a reset, the value of TB0FF0 and TB0FF1 is undefined. If "00" is written to TB0FFCR<TB0FF0C1:0> or <TB0FF1C1:0>, TB0FF0 or TB0FF1 will be inverted. If "01" is written to the flip-flops control registers, the value of TB0FF0 and TB0FF1 will be set to "1". If "10" is written to the flip-flops control registers, the value of TB0FF0 and TB0FF1 will be cleared to "0".

The values of TB0FF0 and TB0FF1 can be output to the timer output pins TB0OUT0 (which is shared with P82), TB0OUT1 (which is shared with P83). Timer output should be specified by using the port 8 function register P8FC and port 8 control register P8CR.

3.9.3 SFR

TMRB0 RUN Register

TB0RUN
(0180H)

	7	6	5	4	3	2	1	0
Bit symbol	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
Read/Write	R/W	R/W			R/W	R/W		R/W
After reset	0	0			0	0		0
Function	Double buffer 0: Disable 1: Enable	Write “0”.			IDLE2 0: Stop 1: Operation	16-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		

Count operation

0	Stop and clear
1	Count

I2TB0: Operation of IDLE2 mode
TB0PRUN: Operation of prescaler
TB0RUN: Operation of TMRB0

Note: The values of bits 1, 4 and 5 of TB0RUN are undefined when read.

Figure 3.9.2 Register for TMRB

TMRB0 Mode Register

TB0MOD (0182H)

Read-Modify-write instruction is prohibited

	7	6	5	4	3	2	1	0
Bit symbol	TB0CT1	TB0ET1	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
Read/Write	R/W		W*	R/W				
After reset	0	0	1	0	0	0	0	0
Function	TB0FF1 Invertion trigger 0: Trigger disable 1: Trigger enable		Software capture control	Capture timing 00: Disable INT5 is rising edge 01: TB0IN0 ↑ TB0IN1 ↑ INT5 is rising edge 10: TB0IN0 ↑ TB0IN0 ↓ INT5 is falling edge 11: TA1OUT ↑ TA1OUT ↓ INT5 is rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB0 source clock select 00: TB0IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
	Invert when capture to capture register 1	Invert when match UC0 with timer register 1	0: Software capture 1: Undefined					

TMRB0 source closk

00	External input clock (TB0IN0 pin input)
01	ϕT1
10	ϕT4
11	ϕT16

Clear of up counter 0 (UC0)

0	Disable clear of up counter
1	Clear by match with TB0RG1

Capture/Interrupt timing

	Capture control	INT5 control
00	Capture disable	INT5 generate by rising TB0IN0
01	TB0CP0 by rising TB0IN0 TB0CP1 by rising TB0IN1	
10	TB0CP0 by rising TB0IN0 TB0CP1 by falling TB0IN0	INT5 generate by falling TB0IN0
11	TB0CP0 by rising TA1OUT TB0CP1 by falling TA1OUT	INT5 generate by rising TB0IN0

Software capture

0	Capture value of up counter to TB0CP0.
1	Undefined (Note)

Note: Whenever programming “0” to TB0MOD<TB0CP0I> bit, present value of up counter is received to capture register TB0CP0. But, write “1” to TB0MOD<TB0CP0I> in condition of written “0” to TB0MOD<TB0CP0I> bit, present value of up counter is received to capture register TB0CP0. Therefore you must to regard.

Figure 3.9.3 Register for TMRB

TMRB0 Flip-Flop Control Register

<table>
<tr><td></td><td></td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td rowspan="3">TB0FFCR
(0183H)</td><td>Bit symbol</td><td>TB0FF1C1</td><td>TB0FF1C0</td><td>TB0C1T1</td><td>TB0C0T1</td><td>TB0E1T1</td><td>TB0E0T1</td><td>TB0FF0C1</td><td>TB0FF0C0</td></tr>
<tr><td>Read/Write</td><td colspan="2">W*</td><td colspan="4">R/W</td><td colspan="2">W*</td></tr>
<tr><td>After reset</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>1</td></tr>
<tr><td rowspan="2">Read-Modify
-write
instruction is
prohibited</td><td rowspan="2">Function</td><td colspan="2" rowspan="2">TB0FF1 control
00: Invert
01: Set
10: Clear
11: Don’t care
* Always read as “11”.</td><td colspan="4">TB0FF0 inversion trigger
0: Trigger disable
1: Trigger enable</td><td colspan="2" rowspan="2">TB0FF0 Control
00: Invert
01: Set
10: Clear
11: Don’t care
* Always read as “11”.</td></tr>
<tr><td>Invert when the UC value is loaded in to TB0CP1.</td><td>Invert when the UC value is loaded in to TB0CP0.</td><td>Invert when the UC matches with TB0RG1.</td><td>Invert when the UC match with TB0RG0.</td></tr>
</table>

Timer Flip-Flop (TB0FF0) control

00	Invert to TB0FF0 (Software inversion).
01	Set TB0FF0 to “1”.
10	Clear TB0FF0 to “0”.
11	Don’t care

Inversion trigger of TB0FF0 when the UC match with TB0RG0

0	Trigger disable (Disable inversion)
1	Trigger enable (Enable inversion)

Inversion trigger of TB0FF0 when the UC match with TB0RG1

0	Trigger disable (Disable inversion)
1	Trigger enable (Enable inversion)

Inversion trigger of TB0FF0 When the UC value is loaded in to TB0CP0

0	Trigger disable (Disable inversion)
1	Trigger enable (Enable inversion)

Inversion trigger of TB0FF0 When the UC value is loaded in to TB0CP1

0	Trigger disable (Disable inversion)
1	Trigger enable (Enable inversion)

Figure 3.9.4 Register for TMRB

3.9.4 Operation in Each Mode

(1) 16-bit interval timer mode

Generating interrupts at fixed intervals

In this example,the interval time is set the timer register TB0RG1 to generate the interrupt INTTB01.

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Stop TMRB0.
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01 and set interrupt level 4. Disable INTTB00.
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable the trigger.
TB0MOD	←	0	0	1	0	0	1	*	*	Select source clock and
						(** = 01, 10, 11)				Disable the capture function.
TB0RG1	←	*	*	*	*	*	*	*	*	Set the interval time (16 bits).
		*	*	*	*	*	*	*	*	
TB0RUN	←	0	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

(2) 16-bit event counter mode

In 16-bit timer mode as described in above, the timer can be used as an event counter by selecting the external clock (TB0IN0 pin input) as the input clock.

Up counter counting up by rising edge of TB0IN0 pin input. And execution software capture and reading capture value enable reading count value.

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Stop TMRB0.
P8CR	←	X	X	X	X	–	–	–	0	Set P80 to TB0IN0 input mode.
P8FC	←	X	X	X	X	–	–	–	1	
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01 and set interrupt level 4. Disable INTTB00.
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable trigger.
TB0MOD	←	0	0	1	0	0	1	0	0	Set input clock to TB0IN0 pin input.
TB0RG1	←	*	*	*	*	*	*	*	*	Set number of count. (16 bits)
		*	*	*	*	*	*	*	*	
TB0RUN	←	0	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

When used as an event counter, set the prescaler to "RUN".
(TB0RUN<TB0PRUN> = "1")

(3) 16-bit programmable pulse generation (PPG) output mode

Square wave pulses can be generated at any frequency and duty ratio. The output pulse may be either low active or high active.

The PPG mode is obtained by inversion of the timer flip-flop TB0FF0 that is to be enabled by the match of the up counter UC0 with timer register TB0RG0 or TB0RG1 and to be output to TB0OUT0. In this mode, the following conditions must be satisfied.

(Set value of TB0RG0) < (Set value of TB0RG1)

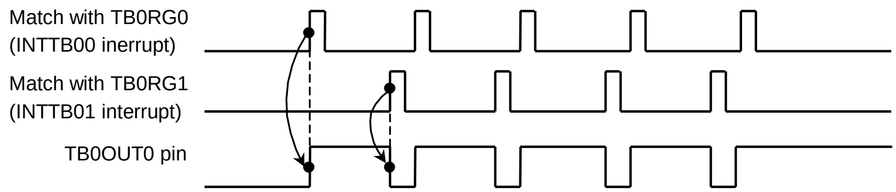


Figure 3.9.5 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0 double buffer is enabled in this mode, the value of register buffer 0 will be shifted into TB0RG0 at match with TB0RG1. This feature makes easy the handling of low-duty waves.

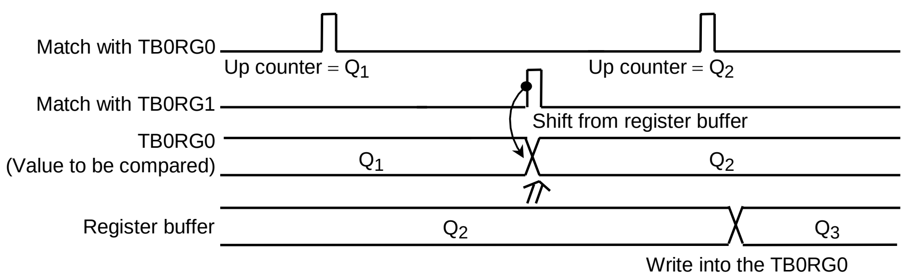


Figure 3.9.6 Operation of Register Buffer

The following block diagram illustrates this mode.

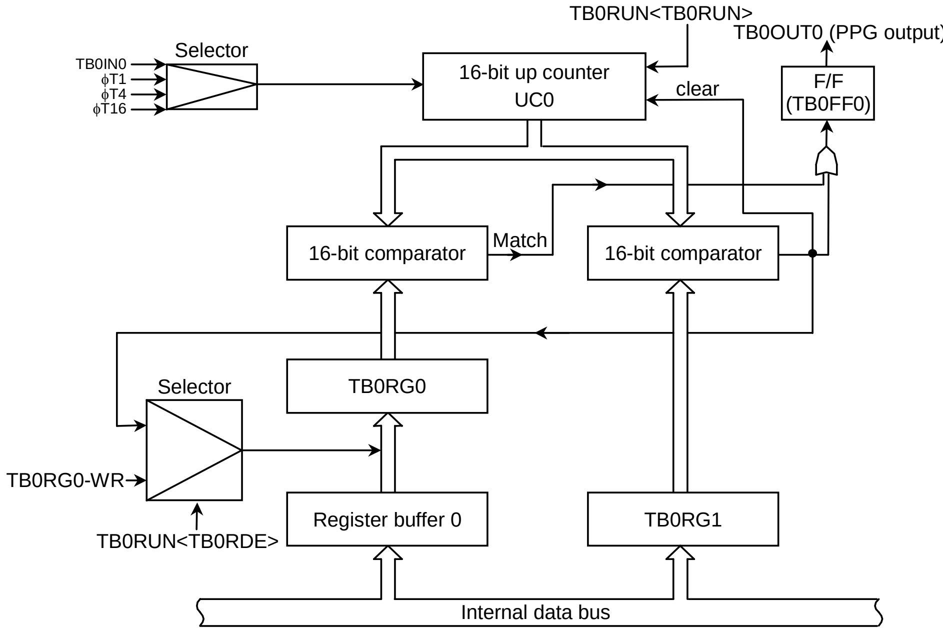


Figure 3.9.7 Block Diagram of 16-Bit PPG Mode

The following example shows how to set 16-bit PPG output mode:

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Disable the TB0RG0 double buffer and stop TMRB0.
TB0RG0	←	*	*	*	*	*	*	*	*	Set the duty ratio (16 bits).
TB0RG1	←	*	*	*	*	*	*	*	*	Set the frequency (16 bits).
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0 double buffer. (The duty and frequency are changed on an INTTB01 interrupt.)
TB0FFCR	←	X	X	0	0	1	1	1	0	Set the mode to invert TB0FF0 at the match with TB0RG0/TB0RG1. Clear TB0FF0 to 0.
TB0MOD	←	0	0	1	0	0	1	*	*	Select the source clock and disable the capture function.
						(** = 01, 10, 11)				
P8CR	←	X	X	X	X	–	1	–	–	Set P82 to function as TB0OUT0.
P8FC	←	X	X	X	X	–	1	–	–	
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

X: Don’t care, –: No change

(4) Capture function examples

Used capture function, they can be applicabled in many ways, for example:

a. One-shot pulse output from external trigger pulse

b. For frequency measurement

c. For pulse width measurement

d. For time difference measurement

a. One-shot pulse output from external trigger pulse

Set the up counter UC0 in free-running mode with the internal input clock, input the external trigger pulse from TB0IN0 pin, and load the value of up-counter into capture register TB0CP0 at the rise edge of the TB0IN0 pin.

When the interrupt INT5 is generated at the rise edge of TB0IN0 input, set the TB0CP0 value (c) plus a delay time (d) to TB0RG0 (= c + d), and set the above set value (c + d) plus a one-shot width (p) to TB0RG1 (= c + d + p). And, set "11" to timer flip-flop control register TB0FFCR<TB0E1T1, TB0E0T1>. Set to trigger enable for be inverted timer flip-flop TB0FF0 by UC0 matching with TB0RG0 and with TB0RG1. When interrupt INTTB01 occurs, this inversion will be disabled after one-shot pulse is output.

The (c), (d) and (p) correspond to c, d and p figure 3.9.8.

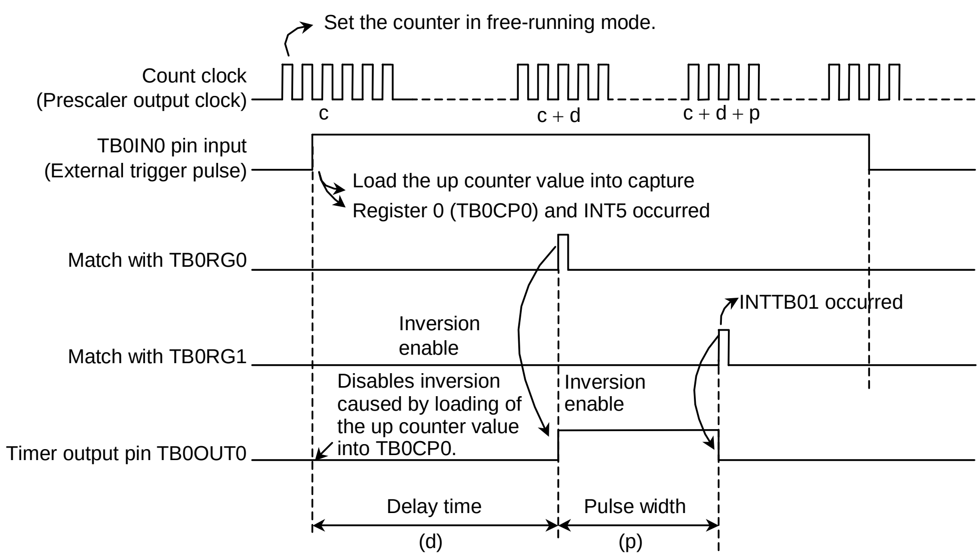


Figure 3.9.8 One-shot Pulse Output (with delay)

Example: To output a 2-ms one-shot pulse with a 3-ms delay to the external trigger pulse via the TB0IN0 pin.

* Clock state
- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: f_{FPH}

Main setting

		7	6	5	4	3	2	1	0	
TB0MOD	←	X	X	1	0	1	0	0	1	Set free running. Count using φT1. Load the up counter value into TB0CP0 at the rising edge of TB0IN0 pin input.
TB0FFCR	←	X	X	0	0	0	0	1	0	Clear TB0FF0 to 0. Disable inversion of TB0FF0.
P8CR	←	X	X	X	X	–	1	–	0	Set P82 to function as the TB0OUT0 pin.
P8FC	←	X	X	X	X	–	1	–	1	Set P80 to TB0IN0 input mode.
INTE56	←	X	–	–	–	X	1	0	0	Enable INT5. Disable INTTB00 and INTTB01.
INTETB0	←	X	0	0	0	X	0	0	0	
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0.

X: Don't care, –: No change

Setting in INT5

TB0RG0	←	TB0CP0 + 3 ms/φT1								
TB0RG1	←	TB0RG0 + 2 ms/φT1								
TB0FFCR	←	X	X	–	–	1	1	–	–	Enable inversion of TB0FF0 when the up counter value match with value of TB0RG0 or TB0RG1.
INTETB0	←	X	1	0	0	X	–	–	–	Enable INTTB01.

X: Don't care, –: No change

Setting in INTTB01

TB0FFCR	←	X	X	–	–	0	0	–	–	Disable inversion of TB0FF0 when the up counter value match with value of TB0RG0 or TB0RG1.
INTETB0	←	X	0	0	0	X	–	–	–	Disable INTTB01.

X: Don't care, –: No change

When delay time is unnecessary, invert timer flip-flop TB0FF0 when up-counter value is loaded into capture register (TB0CP0), and set the TB0CP0 value (c) plus the one-shot pulse width (p) to TB0RG1 when the interrupt INT5 occurs. The TB0FF0 inversion should be enable when the up counter (UC0) value matches TB0RG1, and disabled when generating the interrupt INTTB01.

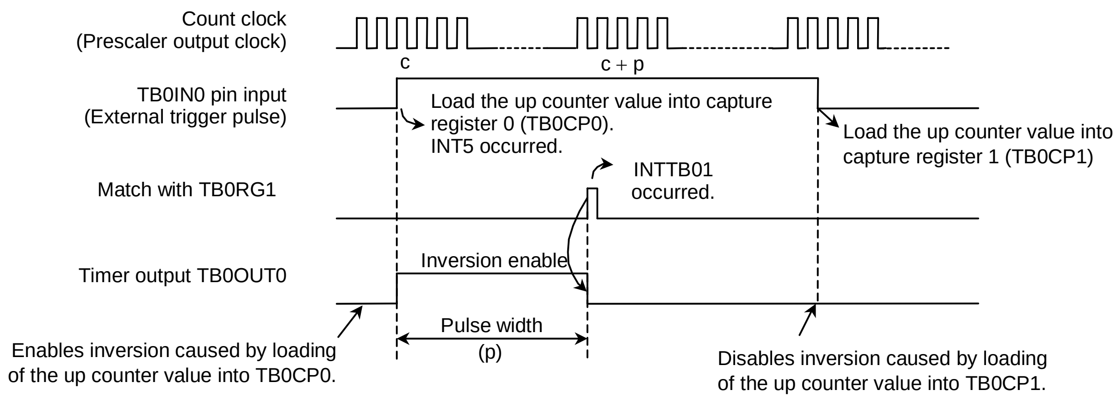


Figure 3.9.9 One-shot Pulse Output of External Trigger Pulse (without delay)

b. Frequency measurement

The frequency of the external clock can be measured in this mode. The clock is input through the TB0IN0 pin, and its frequency is measured by the 8-bit timers TMRA01 and the 16-bit timer/event counter (TMRB0). (TMRA01 is used to setting of measurement time by inversion TA1FF.)

The TB0IN0 pin input should be for the input clock of TMRB0. Set to TB0MOD <TB0CPM1:0> = “11”. The value of the up counter (UC0) is loaded into the capture register TB0CP0 at the rise edge of the timer flip-flop TA1FF of 8-bit timers (TMRA01), and into TB0CP1 at its fall edge.

The frequency is calculated by difference between the loaded values in TB0CP0 and TB0CP1 when the interrupt (INTTA0 or INTTA1) is generates by either 8-bit timer.

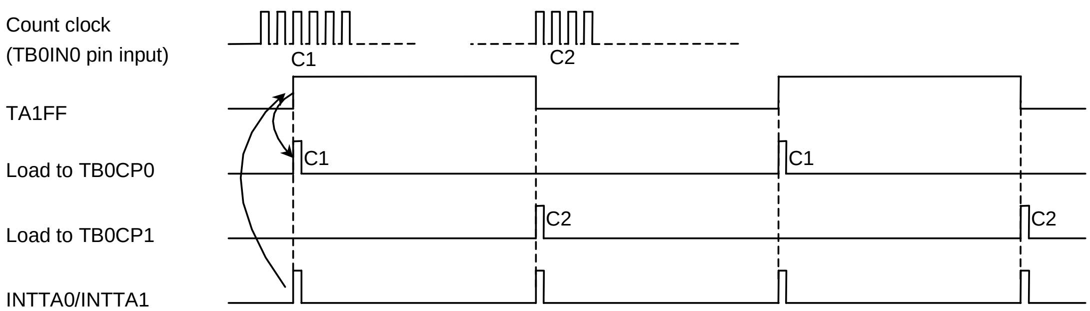


Figure 3.9.10 Frequency Measurement

For example, if the value for the level 1 width of TA1FF of the 8-bit timer is set to 0.5 s and the difference between the values in TB0CP0 and TB0CP1 is 100, the frequency is 100 ÷ 0.5 s = 200 Hz.

c. Pulse width measurement

This mode allows to measure the high-level width of an external pulse. While keeping the 16-bit timer/event counter counting (Free running) with the internal clock input, external pulse is input through the TB0IN0 pin. Then the capture function is used to load the UC0 values into TB0CP0 and TB0CP1 at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT5 occurs at the falling edge of TB0IN0.

The pulse width is obtained from the difference between the values of TB0CP0 and TB0CP1 and the internal clock cycle.

For example, if the internal clock is 0.8 μs and the difference between TB0CP0 and TB0CP1 is 100, the pulse width will be 100 × 0.8 μs = 80 μs.

Additionally, the pulse width which is over the UC0 maximum count time specified by the clock source, can be measured by changing software.

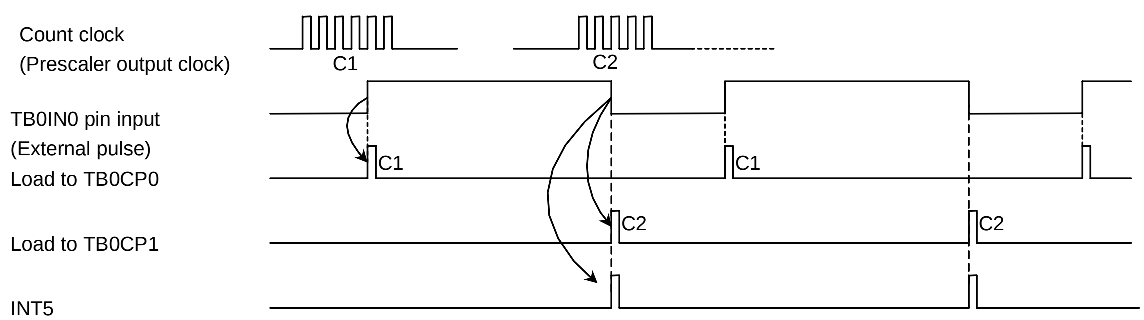


Figure 3.9.11 Pulse Width Measurement

Note: Pulse width measure by setting “10” to TB0MOD<TB0CPM1:0>. The external interrupt INT5 is generated in timing of falling edge of TB0IN0 input. In other modes, it is generated in timing of rising edge of TB0IN0 input.

The width of low-level can be measured from the difference between the first C2 and the second C1 at the second INT5 interrupt.

d. Measurement of difference time

This mode is used to measure the difference in time between the rising edges of external pulses input through TB0IN0 and TB0IN1.

Keep the 16-bit timer/event counter (TMRB0) counting (Free running) with the internal clock, and load the UC0 value into TB0CP0 at the rising edge of the input pulse to TB0IN0. Then the interrupt INT5 is generated.

Similarly, the UC0 value is loaded into TB0CP1 at the rising edge of the input pulse to TB0IN1, generating the interrupt INT6.

The time difference between these pulses can be obtained by multiplying the value subtracted TB0CP0 from TB0CP1 and the internal clock cycle together at which loading the up counter value into TB0CP0 and TB0CP1 has been done.

.

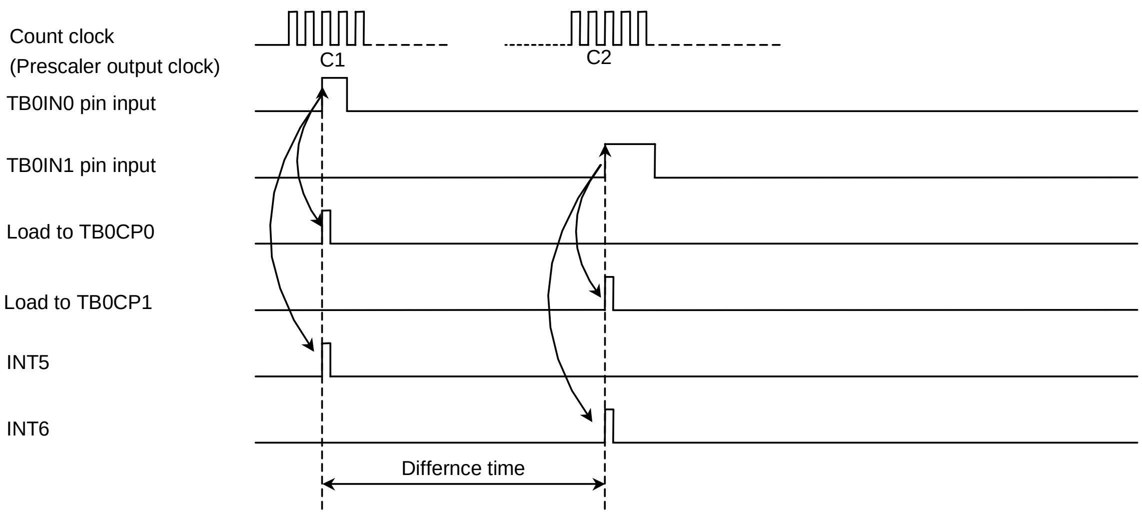


Figure 3.9.12 Measurement of Difference Time

3.10 Serial Channels

TMP91CP27 includes 2 serial I/O channels. Each channel is called SIO0 and SIO1. For both channels either UART mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission) can be selected.

- I/O interface mode —— Mode 0: For transmitting and receiving I/O data using the synchronizing signal SCLK for extending I/O.
- UART mode
  - Mode 1: 7-bit data
  - Mode 2: 8-bit data
  - Mode 3: 9-bit data

In mode 1 and mode 2 a parity bit can be added. Mode 3 has a wakeup function for making the master controller start slave controllers via a serial link (A multi-controller system).

Figure 3.10.2 and Figure 3.10.3 are block diagrams for each channel. Each channel is structured in prescaler, serial clock generation circuit, receiving buffer and control circuit, transfer buffer and control circuit.

Serial channels 0 and 1 can be used independently.

Both channels operate in the same fashion except for the following points; hence only the operation of channel 0 is explained below.

Table 3.10.1 Differences Between Channels 0 to 1

	SIO0	SIO1
Pin name	TXD0 (P90) RXD0 (P91) $\overline{\text{CTS0}}$ /SCLK0 (P92)	TXD1 (P93) RXD1 (P94) $\overline{\text{CTS1}}$ /SCLK1 (P95)
IrDA mode	Yes	No

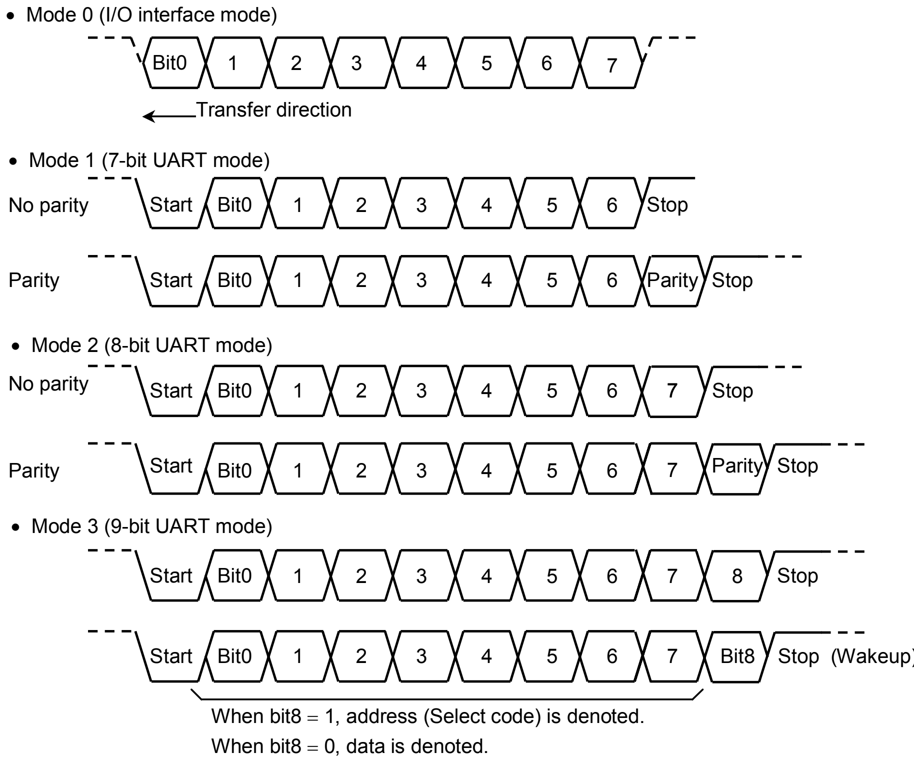


Figure 3.10.1 Data Format

3.10.1 Block Diagram of Each Channels

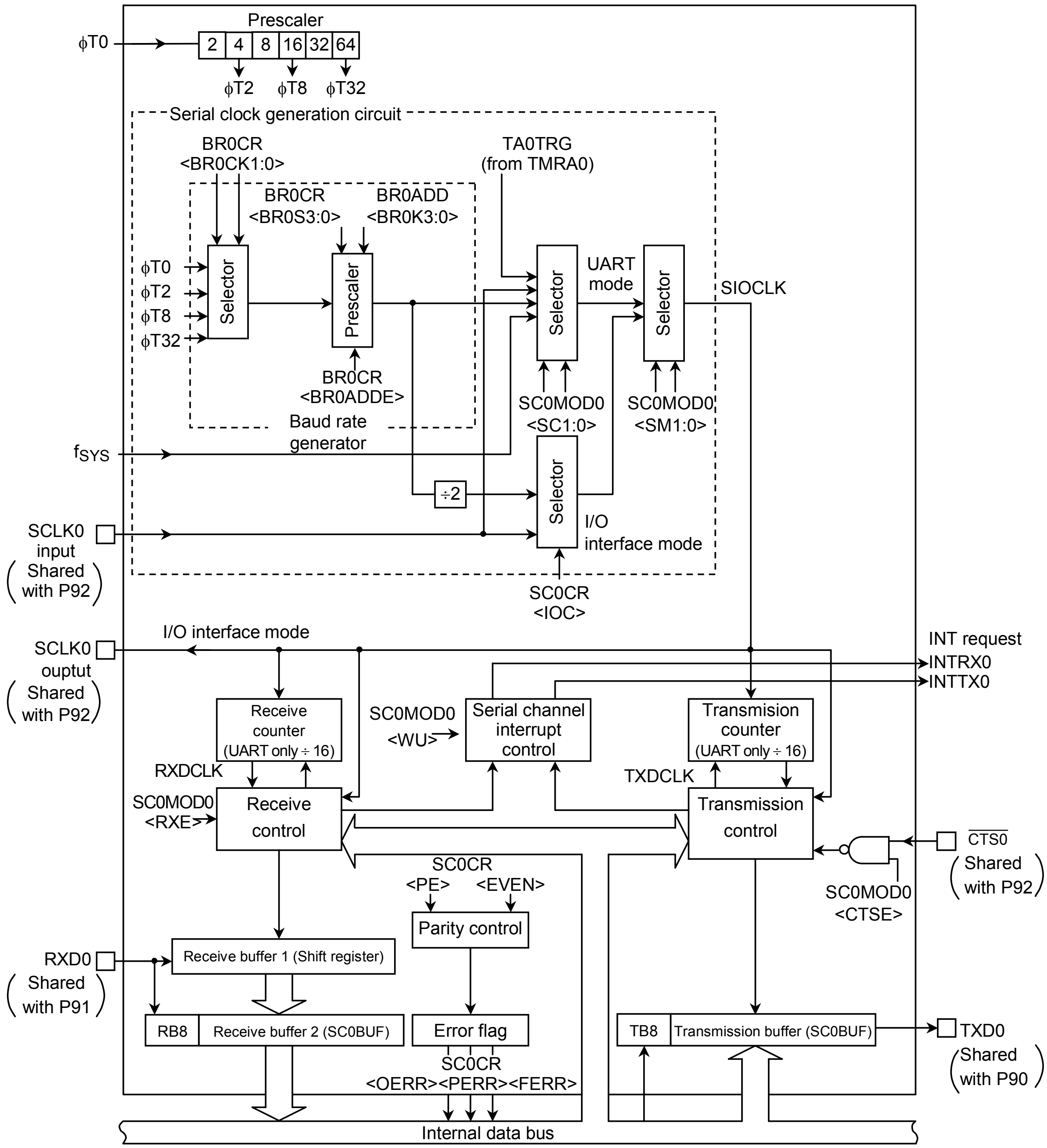


Figure 3.10.2 Block Diagram of SIO0

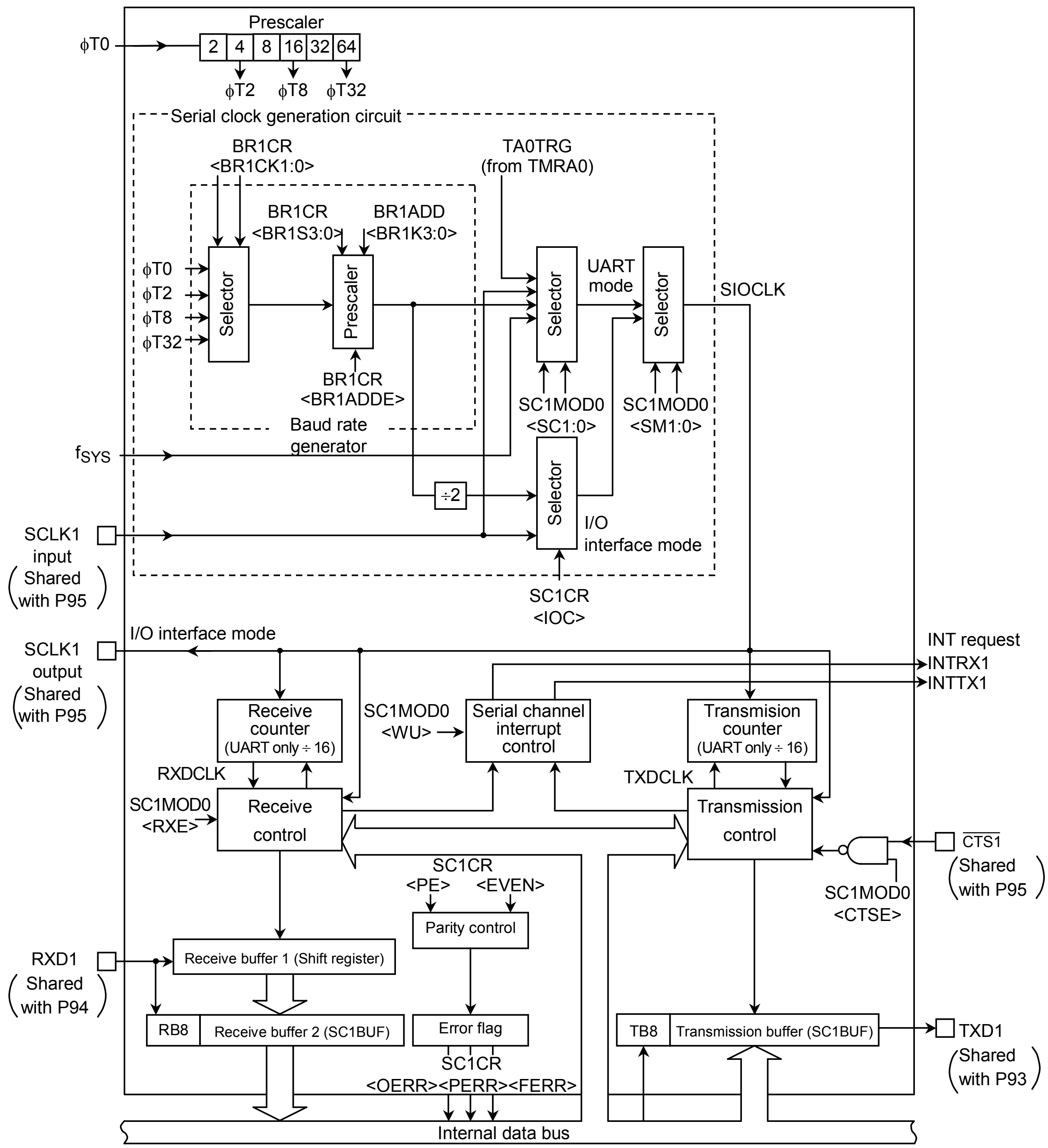


Figure 3.10.3 Block Diagram of SIO1

3.10.2 Operation of Each Circuit

(1) Prescaler

There is a 6-bit prescaler for generating a clock to SIO0. The clock selected using SYSCR0<PRCK1:0> is divided by 4 and input to the prescaler as ϕT0. The prescaler can be run only case of selecting the baud rate generator as the serial transfer clock.

Table 3.10.2 shows prescaler clock resolution into the baud rate generator.

Table 3.10.2 Prescaler Clock Resolution to Baud Rate Generator

Select System Clock <SYSCK>	Select Prescaler Clock <PRCK1:0>	Clock Gear Value <GEAR2:0>	Prescaler Input Clock Resolution			
			ϕT0	ϕT2	ϕT8	ϕT32
1 (fs)	00 (f_{FPH})	XXX	$fs/2^{2}$	$fs/2^{4}$	$fs/2^{6}$	$fs/2^{8}$
0 (fc)		000 (fc)	$fc/2^{2}$	$fc/2^{4}$	$fc/2^{6}$	$fc/2^{8}$
		001 (fc/2)	$fc/2^{3}$	$fc/2^{5}$	$fc/2^{7}$	$fc/2^{9}$
		010 (fc/4)	$fc/2^{4}$	$fc/2^{6}$	$fc/2^{8}$	$fc/2^{10}$
		011 (fc/8)	$fc/2^{5}$	$fc/2^{7}$	$fc/2^{9}$	$fc/2^{11}$
		100 (fc/16)	$fc/2^{6}$	$fc/2^{8}$	$fc/2^{10}$	$fc/2^{12}$
	10 (fc/16 Clock)	XXX	–	$fc/2^{8}$	$fc/2^{10}$	$fc/2^{12}$

XXX: Don’t care, –: Can not be used

The serial interface baud rate generator selects between 4 clock inputs: ϕT0, ϕT2, ϕT8, and ϕT32 among the prescaler outputs.

(2) Baud rate generator

The baud rate generator is a circuit that generates transmission and receiving clocks that determine the transfer rate of the serial channels.

The input clock to the baud rate generator, $\phi T0$, $\phi T2$, $\phi T8$ or $\phi T32$, is generated by the 6-bit prescaler which is shared by the timers. One of these input clocks is selected using the BR0CR<BR0CK1:0> field in the baud rate generator control register.
The baud rate generator includes a frequency divider, which divides the frequency by 1 or N + (16 – K)/16 to 16 values, determining the transfer rate.

The transfer rate is determined by the settings of BR0CR<BR0ADDE, BR0S3:0> and BR0ADD<BR0K3:0>.

- In UART mode

(1) When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3:0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CR<BR0S3:0>. (N = 1, 2, 3 ··· 16)

(2) When BR0CR<BR0ADDE> = 1

The N + (16 – K)/16 division function is enabled. The baud rate generator divides the selected prescaler clock by N + (16 – K)/16 using the value of N set in BR0CR<BR0S3:0> and the value of K set in BR0ADD<BR0K3:0>. (N = 2, 3 ··· 15, K = 1, 2, 3 ··· 15)

Note: If N = 1 or N = 16, the N + (16 – K)/16 division function is disabled.
Clear BR0CR<BR0ADDE> register to “0”.

- In I/O interface mode

The N + (16 – K)/16 division function is not available in I/O Interface Mode. Set BR0CR<BR0ADDE> to 0 before dividing by N.

The method for calculating the transfer rate when the baud rate generator is used is explained below.

- In UART mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

- In I/O interface mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 2$$

- Integer divider (N divider)

For example, when the source clock frequency (fc) = 12.288 MHz, the input clock frequency = ϕT2 (fc/16), the frequency divider N (BR0CR<BR0S3:0>) = 5, and BR0CR<BR0ADDE> = 0, the baud rate in UART Mode is as follows:

* Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

$$\text{Baud rate} = \frac{fc/16}{5} \div 16$$

$$= 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600 \text{ (bps)}$$

Note: The N + (16 – K)/16 division function is disabled and setting BR0ADD<BR0K3:0> is invalid.

- N + (16 – K)/16 divider (UART mode only)

Accordingly, when the source clock frequency (fc) = 4.8 MHz, the input clock frequency = ϕT0, the frequency divider N (BR0CR<BR0S3:0>) = 7, K (BR0ADD<BR0K3:0>) = 3, and BR0CR<BR0ADDE> = 1, the baud rate in UART Mode is as follows:

* Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

$$\text{Baud rate} = \frac{fc/4}{7 + (16 - 3)/16} \div 16$$

$$= 4.8 \times 10^6 \div 4 \div (7 + 13/16) \div 16 = 9600 \text{ (bps)}$$

Table 3.10.3, Table 3.10.4 show examples of UART mode transfer rates.

Additionally, the external clock input is available in the serial clock. (Serial channels 0 and 1). The method for calculating the baud rate is explained below:

- In UART mode

  Baud rate = external clock input frequency ÷ 16

  It is necessary to satisfy (external clock input cycle) ≥ 4/fc

- In I/O interface mode

  Baud rate = External clock input frequency

  It is necessary to satisfy (external clock input cycle) ≥ 16/fc

Table 3.10.3 Transfer Rate Selection

(when baud rate generator is used and BR0CR<BR0ADDE> = 0.) Unit (kbps)

fc [MHz]	Input Clock / Divider N (Set to BR0CR<BR0S3:0>)	ϕT0	ϕT2	ϕT8	ϕT32
9.830400	2	76.800	19.200	4.800	1.200
↑	4	38.400	9.600	2.400	0.600
↑	8	19.200	4.800	1.200	0.300
↑	0	9.600	2.400	0.600	0.150
12.288000	5	38.400	9.600	2.400	0.600
↑	A	19.200	4.800	1.200	0.300
14.745600	2	115.200			
↑	3	76.800	19.200	4.800	1.200
↑	6	38.400	9.600	2.400	0.600
↑	C	19.200	4.800	1.200	0.300

Note 1: Transfer rates in I/O interface mode are eight times faster than the values given above.

Note 2: The values in this table are calculated for when fc is selected as the system clock, fc/1 is selected as the clock gear and f_{FPH} is selected as the clock for prescaler.

Table 3.10.4 UART Baud Rate Selection

(when TMRA0 with input clock ϕT1 is used and trigger output of TMRA0 is used.)

Unit (kbps)

fc / TA0REG	12.288 MHz	12 MHz	9.8304 MHz	8 MHz	6.144 MHz
1H	96		76.8	62.5	48
2H	48		38.4	31.25	24
3H	32	31.25			16
4H	24		19.2		12
5H	19.2				9.6
8H	12		9.6		6
AH	9.6				4.8
10H	6		4.8		3
14H	4.8				2.4

Method for calculating the transfer rate (when TMRA0 is used):

$$\text{Transfer rate} = \frac{\text{Clock frequency determined by SYSCR0<PRCK1:0>}}{\text{TA0REG} \times \underbrace{8}_{\text{(When TMRA0 with input clock } \phi \text{T1 is used)}} \times 16}$$

Note 1: The TMRA0 match detect signal cannot be used as the transfer clock in I/O interface mode.

Note 2: The values in this table are calculated for when fc is selected as the system clock, fc/1 is selected as the clock gear, and f_{FPH} is selected as the clock for prescaler.

(3) Serial clock generation circuit

This circuit generates the basic clock for transmitting and receiving data.

- In I/O interface mode

  In SCLK output mode with the setting SC0CR<IOC> = 0, the basic clock is generated by dividing the output of the baud rate generator by 2, as described previously.

  In SCLK input mode with the setting SC0CR<IOC> = 1, the rising edge or falling edge will be detected according to the setting of the SC0CR<SCLKS> register to generate the basic clock.

- In UART mode

  The SC0MOD0<SC1:0> setting determines whether the baud rate generator clocks, the internal system clock f_{SYS}, the trigger output signal from timer TMRA0 or the external clock (SCLK0) is used to generate the basic clock SIOCLK.

(4) Receiving counter

The receiving counter is a 4-bit binary counter used in UART mode that counts up the pulses of the SIOCLK clock. It takes 16 SIOCLK pulses to receive 1 bit of data; each data bit is sampled three times – on the 7th, 8th and 9th clock cycles.

The value of the data bit is determined from these three samples using the majority rule.

For example, if the data bit is sampled respectively as 1, 0 and 1 on 7th, 8th and 9th clock cycles, the received data bit is taken to be 1. A data bit sampled as 0, 0 and 1 are taken to be 0.

(5) Receiving control

- In I/O interface mode

  In SCLK output mode with the setting SC0CR<IOC> = 0, the RXD0 signal is sampled on the rising edge of the shift clock which is output on the SCLK0 pin.

  In SCLK input mode with the setting SC0CR<IOC> = 1, the RXD0 signal is sampled on the rising or falling edge of the SCLK input, according to the SC0CR<SCLKS> setting.

- In UART mode

  The receiving control block has a circuit that detects a start bit using the majority rule. Received bits are sampled three times; when two or more out of three samples are 0, the bit is recognized as the start bit and the receiving operation commences.

  The values of the data bits that are received are also determined using the majority rule.

(6) The receiving buffers

To prevent overrun errors, the receiving buffers are arranged in a double-buffer structure.

Received data is stored one bit at a time in receiving buffer 1 (which is a shift register). When 7 or 8 bits of data have been stored in receiving buffer 1, the stored data is transferred to receiving buffer 2 (SC0BUF); this causes an INTRX0 interrupt to be generated.

The CPU only reads receiving buffer 2 (SC0BUF). Even before the CPU reads receiving buffer 2 (SC0BUF), the received data can be stored in receiving buffer 1. However, unless receiving buffer 2 (SC0BUF) is read before all bits of the next data are received by receiving buffer 1, an overrun error occurs. If an overrun error occurs, the contents of receiving buffer 1 will be lost, although the contents of receiving buffer 2 and SC0CR<RB8> will be preserved.

SC0CR<RB8> is used to store either the parity bit – added in 8-bit UART mode – or the most significant bit (MSB) – in 9-bit UART mode.

In 9-bit UART mode the wake-up function for the slave controller is enabled by setting SC0MOD0<WU> to 1; in this mode INTRX0 interrupts occur only when the value of SC0CR<RB8> is 1.

(7) Transmission counter

The transmission counter is a 4-bit binary counter that is used in UART mode and which, like the receiving counter, counts the SIOCLK clock pulses; a TXDCLK pulse is generated every 16 SIOCLK clock pulses.

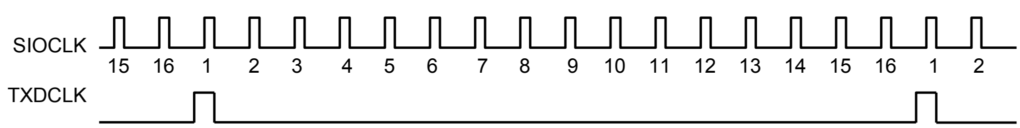


Figure 3.10.4 Generation of Transmission Clock

(8) Transmission controller

- In I/O interface mode

  In SCLK output mode with the setting SC0CR<IOC> = 0, the data in the transmission buffer is output one bit at a time to the TXD0 pin on the rising edge of the shift clock which is output on the SCLK0 pin.

  In SCLK input mode with the setting SC0CR<IOC> = 1, the data in the transmission buffer is output one bit at a time on the TXD0 pin on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

- In UART mode

  When transmission data sent from the CPU is written to the transmission buffer, transmission starts on the rising edge of the next TXDCLK, generating a transmission shift clock TXDSFT.

Handshake function

Serial channels 0 and 1 each has a $\overline{\text{CTS}}$ pin. Use of this pin allows data can be sent in units of one data format; thus, overrun errors can be avoided. The handshake functions is enabled or disabled by the SC0MOD0<CTSE> setting.

When the $\overline{\text{CTS0}}$ pin condition is High level, after completed the current data send, data transmission is halted until the $\overline{\text{CTS0}}$ pin condition is low again. However, the INTTX0 Interrupt is generated, it requests the next send data to the CPU. The next data is written in the transmission buffer and data sending is halted.

Though there is no $\overline{\text{RTS}}$ pin, a handshake function can be easily configured by setting any port assigned to be the $\overline{\text{RTS}}$ function. The $\overline{\text{RTS}}$ should be output "High" to request send data halt after data receive is completed.

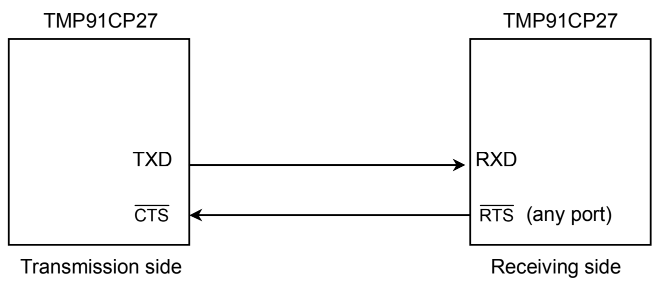
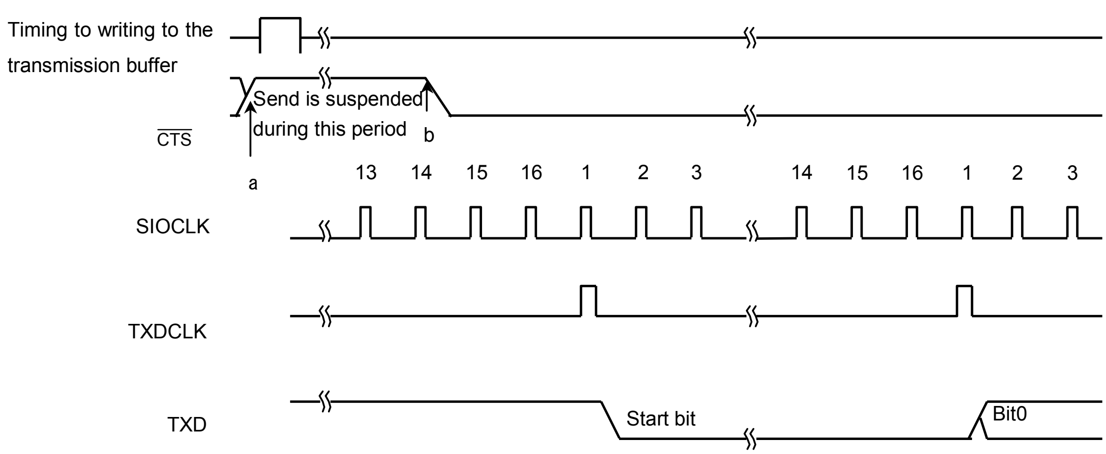


Figure 3.10.5 Hand Shake Function



Note 1: If the $\overline{\text{CTS}}$ signal goes high during transmission, no more data will be sent after completion of the current transmission.

Note 2: Transmission starts on the first falling edge of the TXDCLK clock after the $\overline{\text{CTS}}$ signal has fallen.

Figure 3.10.6 $\overline{\text{CTS}}$ (Clear to send) Timing

(9) Transmission buffer

The transmission buffer (SC0BUF) shifts out and sends the transmission data written from the CPU form the least significant bit (LSB) in order. When all the bits are shifted out, the transmission buffer becomes empty and generates an INTTX0 interrupt.

(10) Parity control circuit

When SC0CR<PE> in the serial channel control register is set to 1, it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART mode or 8-bit UART mode. The SC0CR<EVEN> field in the serial channel control register allows either even or odd parity to be selected.

In the case of transmission, parity is automatically generated when data is written to the transmission buffer SC0BUF. The data is transmitted after the parity bit has been stored in SC0BUF<TB7> in 7-bit UART mode or in SC0MOD0<TB8> in 8-bit UART mode. SC0CR<PE> and SC0CR<EVEN> must be set before the transmission data is written to the transmission buffer.

In the case of receiving, data is shifted into receiving buffer 1, and the parity is added after the data has been transferred to receiving buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> in 7-bit UART mode or with SC0CR<RB8> in 8-bit UART mode. If they are not equal, a parity error is generated and the SC0CR<PERR> flag is set.

(11) Error flags

Three error flags are provided to increase the reliability of data reception.

1. Overrun error<OERR>

   If all the bits of the next data item have been received in receiving buffer 1 while valid data still remains stored in receiving buffer 2 (SC0BUF), an overrun error is generated.

   The below is a recommended flow when the overrun error is generated.

   (INTRX interrupt routine)

   1) Read receiving buffer
   2) Read error flag
   3) If<OERR> = 1
      then
      a. Set to disable receiving (write 0 to SC0MOD0<RXE>)
      b. Wait to terminate current frame
      c. Read receiving buffer
      d. Read error flag
      e. Set to enable receiving (write 1 to SC0MOD0<RXE>)
      f. Request to transmit again
   4) Other

2. Parity error<PERR>

The parity generated for the data shifted into receiving buffer 2 (SC0BUF) is compared with the parity bit received via the RXD pin. If they are not equal, a parity error is generated.

3. Framing error<FERR>

The stop bit for the received data is sampled three times around the center. If the majority of the samples are 0, a framing error is generated.

(12) Timing generation

a. In UART mode

Receiving

Mode	9 bits	8 bits + Parity	8 bits, 7 bits + Parity, 7 bits
Interrupt generation timing	Center of last bit (Bit8)	Center of last bit (Parity bit)	Center of stop bit
Framing error generation timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error generation timing	–	Center of last bit (Parity bit)	Center of last bit (Parity bit)
Overrun error generation timing	Center of last bit (Bit8)	Center of last bit (Parity bit)	Center of stop bit

Note: In 9 bits and 8 bits + Parity modes, interrupts coincide with the ninth bit pulse. Thus, when servicing the interrupt, it is necessary to wait for a 1-bit period (to allow the stop bit to be transferred) to allow checking for a framing error.

Transmitting

Mode	9 bits	8 bits + Parity	8 bits, 7 bits + Parity, 7 bits
Interrupt timing	Just before stop bit is transmitted	Just before stop bit is transmitted	Just before stop bit is transmitted

b. I/O interface

Transmission interrupt timing	SCLK output mode	Immediately after rise of last SCLK signal. (See Figure 3.10.19.)
	SCLK input mode	Immediately after rise of last SCLK signal rising mode, or immediately after fall in falling mode. (See Figure 3.10.20.)
Receiving interrupt timing	SCLK output mode	Timing used to transfer received to data receive buffer 2 (SC0BUF) (e.g., immediately after last SCLK). (See Figure 3.10.21.)
	SCLK input mode	Timing used to transfer received data to receive buffer 2 (SC0BUF) (e.g., immediately after last SCLK). (See Figure 3.10.22.)

3.10.3 SFR

SC0MOD0 (0202H)		7	6	5	4	3	2	1	0
	Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Transfer data bit8	Hand shake function control 0: CTS disable 1: CTS enable	Receive control 0: Receive disable 1: Receive enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clcok (SCLK0 input)	

Serial transmission clock source (UART)

00	Timer TMRA0 trigger output signal
01	Baud rate generator
10	Internal clock f_{SYS}
11	External clock (SCLK0 input)

Note: The clock selection for the I/O interface mode is controlled by the serial control register (SC0CR).

Serial transmission mode

00	I/O Interface mode	
01	UART mode	7-bit mode
10		8-bit mode
11		9-bit mode

Wakeup function

	9-bit UART	Other modes
0	Interrupt generated when data is received	Don't care
1	Interrupt generated only when RB8 = 1	

Receiving function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{CTS}$ pin)

0	Disabled (Always transferable)
1	Enabled

Transmission data bit8

Figure 3.10.7 Serial Mode Control Register 0 (for SIO0 and SC0MOD0)

SC1MOD0
(020AH)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transfer data bit8	Handshake function 0: CTS disable 1: CTS enable	Receive control 0: Receive disable 1: Receive enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: TMRA0 trigger 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clcok (SCLK1 input)	

Serial transmission clock source (for UART)

00	Timer TMRA0 trigger output signal
01	Baud rate generator
10	Internal clock f_{SYS}
11	External clock (SCLK1 input)

Note: The clock selection for the I/O interface mode is controlled by the serial control register (SC1CR).

Serial transmission mode

00	I/O interface mode	
01	UART mode	7-bit mode
10		8-bit mode
11		9-bit mode

Wakeup function

	9-bit UART	Other modes
0	Interrupt generated when data is received	Don't care
1	Interrupt generated only when RB8 = 1	

Receiving function

0	Receive disabled
1	Receive enabled

Handshake function ($\overline{CTS}$ pin)

0	Disabled (Always transferable)
1	Enabled

Transmission data bit8

Figure 3.10.8 Serial Mode Control Register 0 (for SIO1 and SC1MOD0)

SC0CR (0201H)

	7	6	5	4	3	2	1	0
Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (Cleared to 0 when read)			R/W	
After reset	Undefined	0	0	0	0	0	0	0
Function	Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK0 1: SCLK0	0:Baud rate generator 1:SCLK0 pin input

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Edge selection for SCLK0 pin (I/O mode)

0	Transmits and receivers data on rising edge of SCLK0
1	Transmits and receivers data on falling edge SCLK0.

Framing error flag
Parity error flag
Overrun error flag
} Cleared to 0 when read

Parity addition enables

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data 8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.10.9 Serial Control Register (for SIO0 and SC0CR)

SC1CR (0209H)

	7	6	5	4	3	2	1	0
Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
Read/Write	R	R/W		R (Cleared to 0 when read)			R/W	
After reset	Undefined	0	0	0	0	0	0	0
Function	Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	1: Error Parity	1: Error Framing	0: SCLK1 1: SCLK1	0:Baud rate generator 1:SCLK1 pin input

I/O interface input clock selection

0	Baud rate generator
1	SCLK1 pin input

Edge selection for SCLK1 pin (I/O mode)

0	Transmits and receivers data on rising edge of SCLK1.
1	Transmits and receivers data on falling edge SCLK1.

Framing error flag
Parity error flag
Overrun error flag
} Cleared to 0 when read

Parity addition enables

0	Disabled
1	Enabled

Even parity addition/check

0	Odd parity
1	Even parity

Received data bit8

Note: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.10.10 Serial Control Register (for SIO1 and SC1CR)

BR0CR (0203H)

	7	6	5	4	3	2	1	0
Bit symbol	−	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Always write “0”.	+(16 − K)/16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Setting of the divided frequency			

\+ (16 − K)/16 divisions enable

0	Disable
1	Enable

Setting the input clock of baud rate generator

00	Internal clock ϕT0
01	Internal clock ϕT2
10	Internal clock ϕT8
11	Internal clock ϕT32

BR0ADD (0204H)

	7	6	5	4	3	2	1	0
Bit symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write					R/W			
After reset					0	0	0	0
Function					Sets frequency divisor “K” (divided by N + (16 − K)/16)			

Sets baud rate generator frequency divisor seting

BR0ADD <BR0K3:0> \ BR0CR <BR0S3:0>	BR0CR<BR0ADDE> = 1: 0000 (N = 16) or 0001 (N = 1)	BR0CR<BR0ADDE> = 1: 0010 (N = 2) to 1111 (N = 15)	BR0CR<BR0ADDE> = 0: 0001 (N = 1) (UART only) to 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by N + (16 − K)/16	Divided by N

Note: When N + (16 − K)/16 division function is used or not

N	UART Mode	I/O Mode
2 to 15	○	×
1, 16	×	×

Set BR0CR<BR0ADDE> to 1 after setting K (K = 1 to 15) to BR0ADD<BR0K3:0> when + (16 − K)/16 division function is used.

Figure 3.10.11 Baud Rate Generator Control (for SIO0, BR0CR and BR0ADD)

BR1CR (020BH)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Always write "0".	+(16 − K)/16 division 0: Disable 1: Enable	00: ϕT0 01: ϕT2 10: ϕT8 11:ϕT32		Divided frequency setting			

+ (16 − K)/16 divisions enable

0	Disabled
1	Enabled

Input clock selection for baud rate generator

00	Internal clock ϕT0
01	Internal clock ϕT2
10	Internal clock ϕT8
11	Internal clock ϕT32

BR1ADD (020CH)

	7	6	5	4	3	2	1	0
Bit symbol					BR1K3	BR1K2	BR1K1	BR1K0
Read/Write					R/W			
After reset					0	0	0	0
Function					Set frequency divisor K (Divided by N + (16 − K)/16)			

Baud rate generator frequency divisor setting

BR1CR <BR1S3:0> / BR1ADD <BR1K3:0>	BR1CR<BR1ADDE> = 1		BR1CR<BR1ADDE> = 0
	0000 (N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) (UART only) to 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	Divided by N
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by N + (16 − K)/16	

Note: When N + (16 − K)/16 division function is used or not

N	UART Mode	I/O Mode
2 to 15	○	×
1, 16	×	×

Set BR1CR<BR1ADDE> to 1 after setting K (K = 1 to 15) to BR1ADD<BR1K3:0> when + (16 − K)/16 division function is used.

Figure 3.10.12 Baud Rate Generator Control (for SIO1, BR1CR and BR1ADD)

SC0BUF (0200H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

Note: Prohibit read modify write for SC0BUF.

Figure 3.10.13 Serial Transmission/Receiving Buffer Register (for SIO0 and SC0BUF)

SC0MOD1 (0205H)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0						
Read/Write	R/W	R/W						
After reset	0	0						
Function	IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full						

Figure 3.10.14 Serial Mode Control Register1 (for SIO0 and SC0MOD1)

SC1BUF (0208H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

Note: Prohibit read modify write for SC1BUF.

Figure 3.10.15 Serial Transmission/Receiving Buffer Register (for SIO1 and SC1BUF)

SC1MOD1 (020DH)

	7	6	5	4	3	2	1	0
Bit symbol	I2S1	FDPX1						
Read/Write	R/W	R/W						
After reset	0	0						
Function	IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full						

Figure 3.10.16 Serial Mode Control Register1 (for SIO1 and SC1MOD1)

3.10.4 Operation of Each Mode

(1) Mode 0 (I/O interface mode)

This mode allows an increase in the number of I/O pins available for transmitting data to or receiving data from an external shift register.

This mode includes the SCLK output mode to output synchronous clock SCLK and SCLK input mode to input external synchronous clock SCLK.

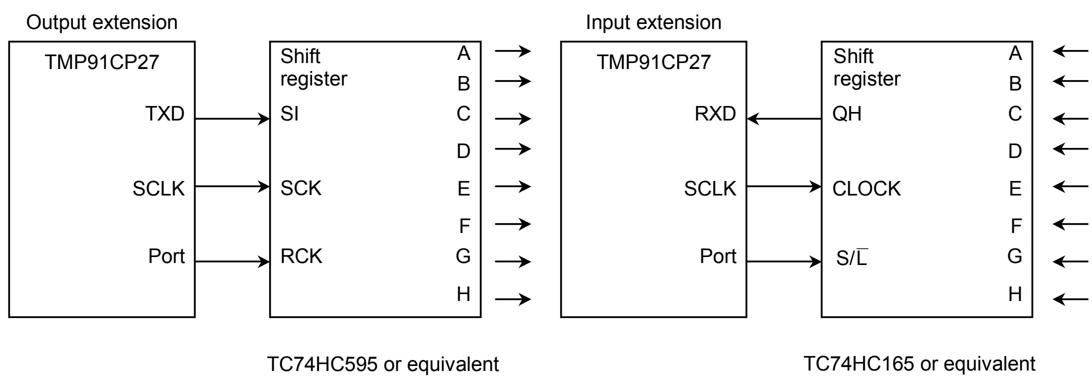


Figure 3.10.17 Example of SCLK Output Mode Connection

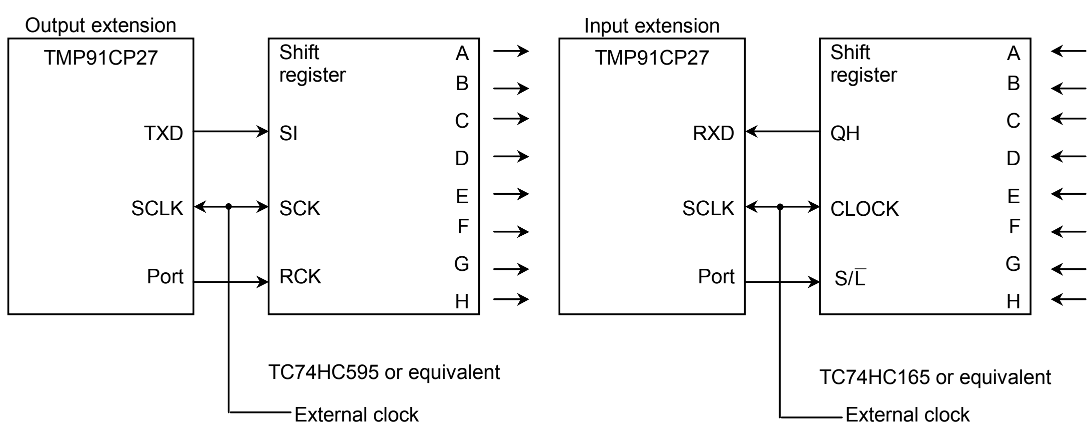


Figure 3.10.18 Example of SCLK Input Mode Connection

a. Transmission

In SCLK output mode 8-bit data and a synchronous clock are output on the TXD0 and SCLK0 pins respectively each time the CPU writes the data to the transmission buffer. When all data is outputted, INTES0<ITX0C> will be set to generate the INTTX0 interrupt.

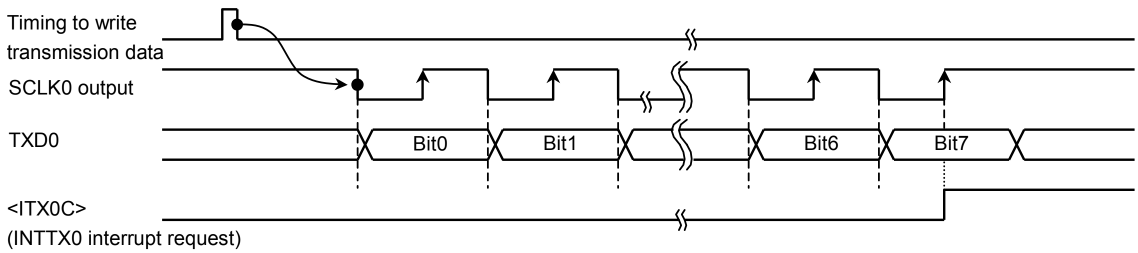


Figure 3.10.19 Transmission Operation in I/O Interface Mode (SCLK0 output mode)

In SCLK Input Mode, 8-bit data is output from the TXD0 pin when the SCLK0 input becomes active after the data has been written to the transmission buffer by the CPU.

When all data is outputted, INTES0<ITX0C> will be set to generate INTTX0 interrupt.

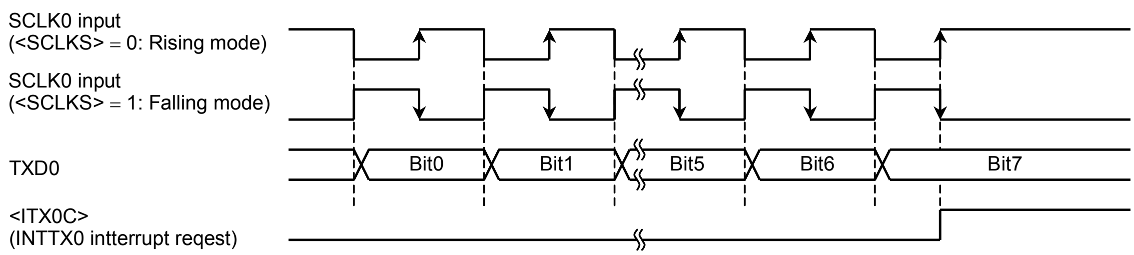


Figure 3.10.20 Transmission Operation in I/O Interface Mode (SCLK0 input mode)

b. Receiving

In SCLK output mode, the synchronous clock is outputted from SCLK0 pin and the data is shifted to receiving buffer 1. This starts when the receive interrupt flag INTES0<IRX0C> is cleared by reading the received data. When 8-bit data are received, the data will be transferred to receiving buffer 2 (SC0BUF according to the timing shown below) and INTES0<IRX0C> will be set to generate INTRX0 interrupt.

The outputting for the first SCLK0 starts by setting SC0MOD0<RXE> to 1.

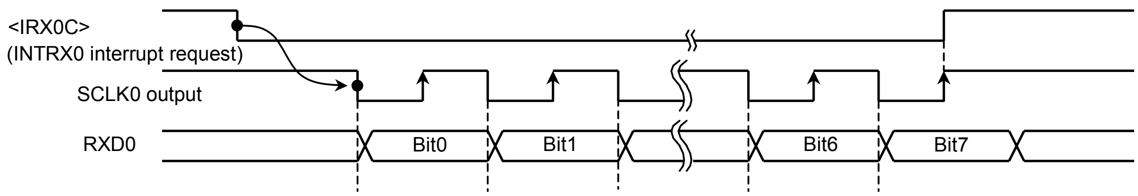


Figure 3.10.21 Receiving Operation in I/O Interface Mode (SCLK0 output mode)

In SCLK input mode, the data is shifted to receiving buffer 1 when the SCLK input becomes active after the receive interrupt flag INTES0<IRX0C> is cleared by reading the received data. When 8-bit data is received, the data will be shifted to receiving buffer 2 (SC0BUF according to the timing shown below) and INTES0<IRX0C> will be set again to be generate INTRX0 interrupt.

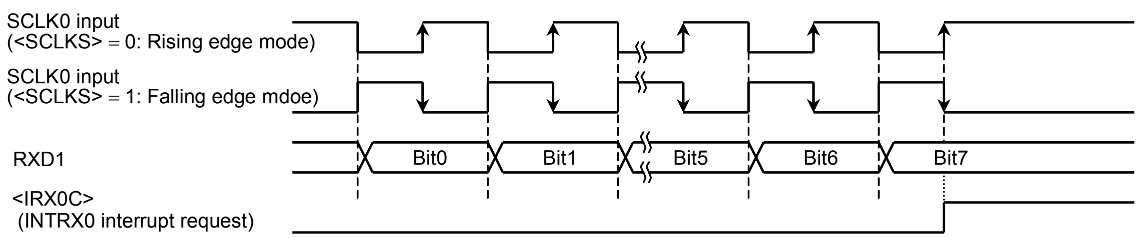


Figure 3.10.22 Receiving Operation in I/O Interface Mode (SCLK0 input mode)

Note: If receiving, set to the receive enable state (SC0MOD0<RXE> = 1) in both SCLK input mode and output mode.

c. Transmission and receiving (Full duplex mode)

When the full duplex mode is used, set the level of receive interrupt to “0” and set enable the interrupt level (1 to 6) to the Transfer interrupts. In the transfer interrupt program, the receiving operation should be done like the below example before setting the next transfer data.

Example: Channel 0, SCLK output
Baud rate = 9600 bps
fc = 14.7456 MHz

* Clock state
System clock: High frequency (fs)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

Main routine

	7	6	5	4	3	2	1	0	
INTES0	0	0	0	1	0	0	0	0	Set transmission interrupt level, and disable receiving interrupt.
P9CR	–	–	–	–	–	1	0	1	Set to P90 (TXD0) and P92 (SCLK0).
P9FC	–	–	–	–	–	1	–	1	
SC0MOD0	0	0	0	0	0	0	0	0	Set to I/O interface mode.
SC0MOD1	1	1	0	0	0	0	0	0	Set to full duplex mode.
SC0CR	0	0	0	0	0	0	0	0	SCLK out, transmit on negative edge, receive on positive edge
BR0CR	0	0	0	1	0	1	1	0	Set to 9600 bps.
SC0MOD0	0	0	1	0	0	0	0	0	Enable receiving.
SC0BUF	*	*	*	*	*	*	*	*	Set the transfer data.

INTTX0 interrupt routine

Acc←SC0BUF									Read the receiving data.
SC0BUF	*	*	*	*	*	*	*	*	Set the next transfer data.

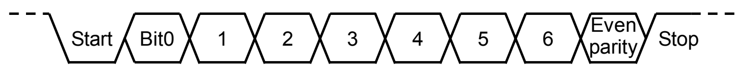
X: Don’t care, –: No change

(2) Mode 1 (7-bit UART mode)

7-bit UART mode is selected by setting serial channel mode register SC0MOD0<SM1:0> to 01.

In this mode, a parity bit can be added. Use of a parity bit is enabled or disabled by the setting of the serial channel control register SC0CR<PE> bit; whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (Enabled).

Setting example: When transmitting data of the following format, the control registers should be set as described below. This explanation applies to channel 0.



← Transmission direction (Transmission rate: 2400 bps at fc = 12.288 MHz)

* Clock state

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: f_{FPH}

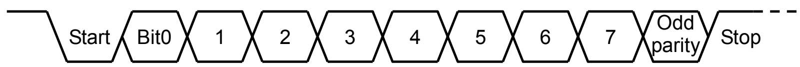
		7	6	5	4	3	2	1	0	
P9CR	←	–	–	–	–	–	–	–	1	Set P90 to function as the TXD0 pin.
P9FC	←	–	–	–	–	–	–	–	1	
SC0MOD0	←	X	0	–	X	0	1	0	1	Select 7-bit UART mode.
SC0CR	←	X	1	1	X	X	X	0	0	Add even parity.
BR0CR	←	0	0	1	0	0	1	0	1	Set the transfer rate to 2400 bps.
INTES0	←	X	1	0	0	–	–	–	–	Enable the INTTX0 interrupt and set it to interrupt level 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

X: Don’t care, –: No change

(3) Mode 2 (8-bit UART mode)

8-bit UART mode is selected by setting SC0MOD0<SM1:0> to 10. In this mode, a parity bit can be added (use of a parity bit is enabled or disabled by the setting of SC0CR<PE>); whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to 1 (Enabled).

Setting example: When receiving data of the following format, the control registers should be set as described below.



← Transmission direction (Transmission rate: 9600 bps at fc = 12.288 MHz)

* Clock state

- System clock: High frequency (fc)
- Clock gear: 1 (fc)
- Prescaler clock: f_{FPH}

Main settings

	7	6	5	4	3	2	1	0	
P9CR ←	−	−	−	−	−	−	0	−	Set P91 (RXD0) pin to input pin.
SC0MOD0 ←	−	0	1	X	1	0	0	1	Enable receiving in 8-bit UART mode.
SC0CR ←	X	0	1	X	X	X	0	0	Add odd parity.
BR0CR ←	0	0	0	1	0	1	0	1	Set to 9600 bps.
INTES0 ←	−	−	−	−	X	1	0	0	Enable the INTRX0 interrupt and set it to interrupt level 4.

Interrupt processing

Acc	← SC0CR AND 00011100	Check for errors.
if Acc	≠ 0 then ERROR	
Acc	← SC0BUF	Read the received data.

X: Don’t care, −: No change

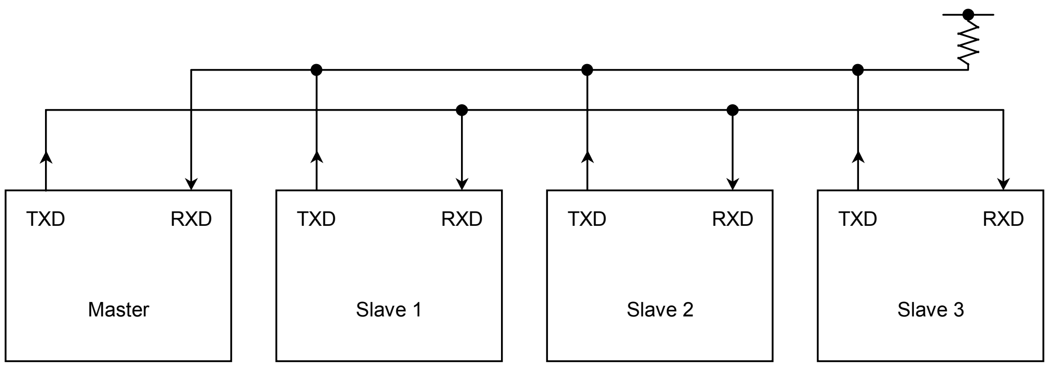
(4) Mode 3 (9-bit UART mode)

9-bit UART mode is selected by setting SC0MOD0<SM1:0> to 11. In this mode parity bit cannot be added.

In the case of transmission the MSB (9th bit) is written to SC0MOD0<TB8>. In the case of receiving it is stored in SC0CR<RB8>. When the buffer is written and read, the MSB is read or written first, before the rest of the SC0BUF data.

Wakeup function

In 9-bit UART mode, the wakeup function for slave controllers is enabled by setting SC0MOD0<WU> to 1. The interrupt INTRX0 occurs only when <RB8> = 1.



Note: The TXD pin of each slave controller must be in open-drain output mode.

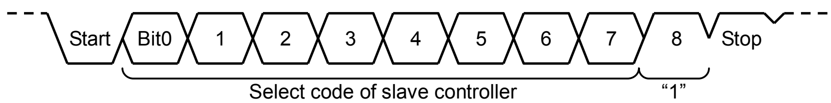
Figure 3.10.23 Serial Link Using Wakeup Function

Protocol

a. Select 9-bit UART mode on the master and slave controllers.

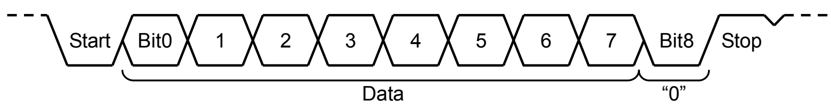
b. Set the SC0MOD0<WU> bit on each slave controller to 1 to enable data receiving.

c. The master controller transmits one-frame data including the 8-bit select code for the slave controllers. The MSB (Bit8)<TB8> is set to “1”.



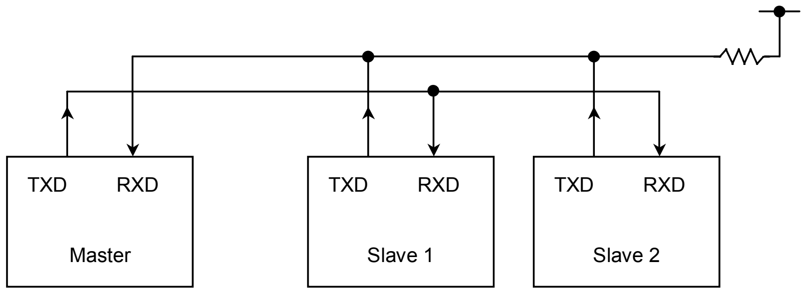
d. Each slave controller receives the above frame. If it matches with own select code, clears<WU> bit to “0”.

e. The master controller transmits data to the specified slave controller whose SC0MOD0<WU> bit is cleared to “0”. The MSB (Bit8)<TB8> is cleared to “0”.



f. The other slave controllers (whose<WU> bits remain at 1) ignore the received data because their MSB (Bit8 or <RB8>) are set to “0”, disabling INTRX0 interrupts.
The slave controller (<WU> bit = “0”) can transmit data to the master controller, and it is possible to indicate the end of data receiving to the master controller by this transmission.

Example: To link two slave controllers serially with the master controller using the system clock f_{SYS} as the transfer clock.



- Master controller setting

```
Main routine

P9CR     ← – – – – – – 0 1   } Set P90 to TXD0 pin. Set P91 to RXD0 pin.
P9FC     ← – – – – – – X 1   }
INTES0   ← X 1 0 0 X 1 0 1     Set INTTX0 to enable, set interrupts level to level 4.
                               Set INTRX0 to enable, set interrupt level to level 5.
SC0MOD0  ← 1 0 1 0 1 1 1 0     Set to 9-bit UART mode, and set transfer clock to fSYS.
SC0BUF   ← 0 0 0 0 0 0 0 1     Set select code of slave 1.

Interrupt routine  (INTTX0)

SC0MOD0  ← 0 – – – – – – –     Clear<TB8> to "0".
SC0BUF   ← * * * * * * * *     Set transmission data.
```

- Slave setting

```
Main routine

P9CR     ← – – – – – – 0 1   }
P9FC     ← – – – – – – X 1   } Set P90 to TXD0 (Open-drain output) and P91 to RXD0.
ODE      ← X X X X – – – 1   }
INTES0   ← X 1 0 1 X 1 1 0     Set INTTX0 and INTRX0 to enable.
SC0MOD0  ← 0 0 1 1 1 1 1 0     Set to<WU> = "1" in 9-bit UART mode, and set transfer clock
                               fSYS.
Interrupt routine (INTRX0)

Acc ← SC0BUF
if Acc = Select code
Then SC0MOD0 ← – – – 0 – – – – –   Clear<WU> to "0".
```

3.10.5 Support for IrDA

SIO0 includes support for the IrDA 1.0 infrared data communication specification. Figure 3.10.24 shows the block diagram.

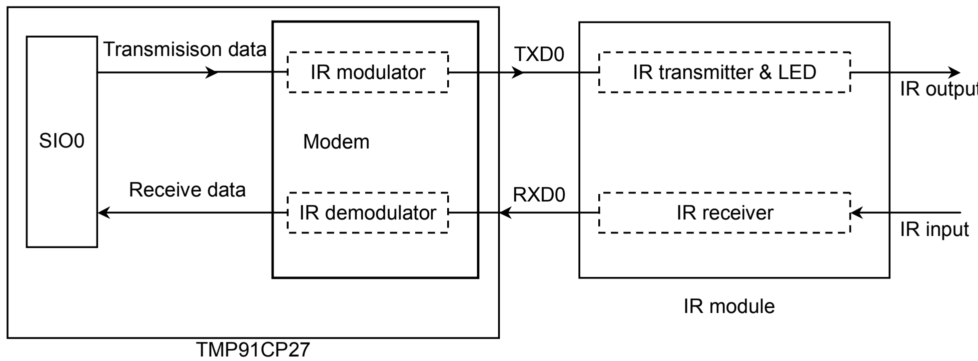


Figure 3.10.24 Block Diagram of IrDA

(1) Modulation of transmission data

When the transmission data is 0, output “H” level with either 3/16 or 1/16 times for width of baud rate (Selectable in software). When data is “1”, modem output “L” level.

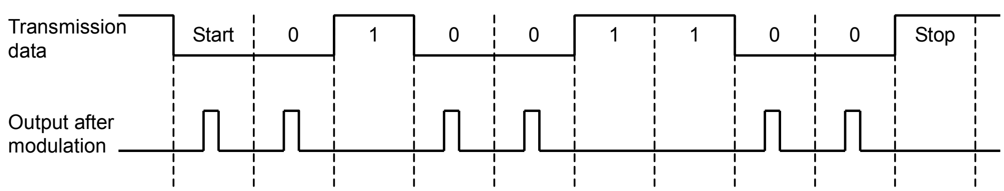


Figure 3.10.25 Example of Modulation of Transmission Data

(2) Modulation of receiving data

When the receive data has the effective high level pulse width (Software selectable), the modem outputs “0” to SIO0. Otherwise modem outputs “1” to SIO0. Receive pulse logic is selectable by SIRCR<RXSEL>.

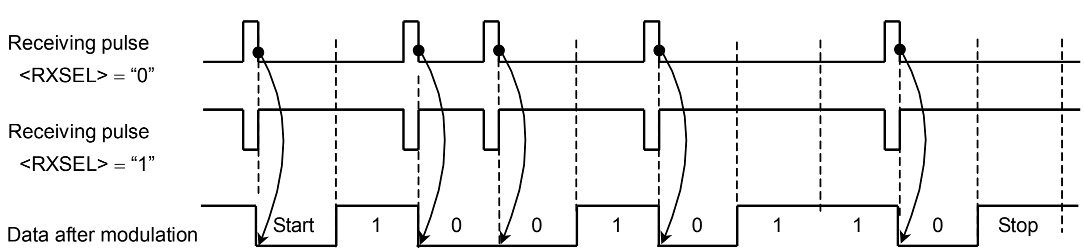


Figure 3.10.26 Example of Modulation of Receiving Data

(3) Data format

Format of transmission/receiving must set to data length 8 bits, without parity bit, 1-bit of stop bit.

Any other settings don't guarantee the normal operation.

(4) SFR

Figure 3.10.27 shows the control register SIRCR. If change setting this register, must set it after set operation of transmission/receiving to disable (Both <TXEN> and <RXEN> of this register should be cleared to 0).

Any changing for this register during transmission or receiving operation doesn't guarantee the normal operation.

The following example describes how to set this register:

```
1)  SIO setting                      ;  Set SIO side.
          ↓
2)  LD      (SIRCR), 07H             ;  Set receiving effect pulse width to 8/16.
3)  LD      (SIRCR), 37H             ;  Enable transmission/receiving by setting<TXEN>, <RXEN> bit to “1”.
          ↓
4)  Start of transmission/receiving  ;  The modem operates as follows:
                                           • SIO0 starts transmitting.
                                           • IR receiver starts receiving.
```

(5) Notes

1. Making baud rate when using IrDA
   In baud rate during using IrDA, must set “01” to SC0MOD0<SC1:0> in SIO by using baud rate generator.
   TA0TRG, f_{SYS}, SCLK0 input of except for it can not using.

2. Output pulse width and baud rate generator during transmission IrDA
   As the IrDA 1.0 physical layer specification, the data transfer speed and infra-red pulse width is specified.

Table 3.10.5 Specification of Transfer Rate and Pulse Width

Transfer Rate	Modulation	Transfer Rate Tolerance (% of Rate)	Minimum of Pulse Width	Typical of Pulse Width 3/16	Maximum of Pulse Width
2.4 kbps	RZI	±0.87	1.41 μs	78.13 μs	88.55 μs
9.6 kbps	RZI	±0.87	1.41 μs	19.53 μs	22.13 μs
19.2 kbps	RZI	±0.87	1.41 μs	9.77 μs	11.07 μs
38.4 kbps	RZI	±0.87	1.41 μs	4.88 μs	5.96 μs
57.6 kbps	RZI	±0.87	1.41 μs	3.26 μs	4.34 μs
115.2 kbps	RZI	±0.87	1.41 μs	1.63 μs	2.23 μs

The infra-red pulse width is specified either baud rate T × 3/16 or 1.6 μs (1.6 μs is equal to T × 3/16 pulse width when baud rate is 115.2 kbps).

The TMP91CP27 has function that selectable the pulse width in during transmitting to either 3/16 or 1/16. But T × 1/16 pulse width can be selected when the baud rate is equal or less than 38.4 kbps only. When 57.6 kbps and 115.2 kbps, the output pulse width should not be set to T × 1/16.

As the same reason, + (16 − K)/16 division function in the baud rate generator of SIO0 cannot be used to generate 115.2 kbps baud rate.
Also when the 38.4 kbps and 1/16 pulse width, + (16 − K)/16 division function can not be used. Table 3.10.6 shows baud rate and pulse width for (16 − K)/16 division function.

Table 3.10.6 Baud Rate and Pulse Width For (16 − K)/16 Division Function

Pulse Width	Baud Rate					
	115.2 kbps	57.6 kbps	38.4 kbps	19.2 kbps	9.6 kbps	2.4 kbps
T × 3/16	×	○	○	○	○	○
T × 1/16	−	−	×	○	○	○

○: Can be used (16 − K)/16 division function.
×: Cannot be used (16 − K)/16 division function.
−: Cannot be set to T × 1/16 pulse width.

SIRCR (0207H)

	7	6	5	4	3	2	1	0
Bit symbol	PLSEL	RXSEL	TXEN	RXEN	SIRWD3	SIRWD2	SIRWD1	SIRWD0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Selection transmission pulse width 0: 3/16 1: 1/16	Receiving data logic 0: "H" pulse 1: "L" pulse	Transmission operation 0: Disable 1: Enable	Receiving operation 0: Disable 1: Enable	Select receiving effective pulse width. Set effective pulse width for equal or more than 2x × (Value + 1). Can be set: 1 to14 Cannot be set: 0, 15			

Select receiving effective pulse width Formula:
Receiving effective pulse width ≥ 2x × (Setting value + 1)
$x = 1/f_{FPH}$

0000	Cannot be set.
0001	Pulse width of equal or more than 4x + 100 ns is effective.
⁒	
1110	Pulse width of equal or more than 30x + 100 ns is effective.
1111	Cannot be set.

Enable of receiving operation

0	Disable receiving operation (Received input is ignored.)
1	Enable receiving operation.

Enable of transmission operation

0	Disable transmission operation
1	Enable transmission operation

Select transmission pulse width

0	Pulse width of 3/16
1	Pulse width of 1/16

Note: If baud rate is late and secure of pulse width of IrDA 1.0 standard (Min 1.6 μs) is enable, reducing infrared ray lighting time and consumption power by setting this bit to "1"are enable.

Figure 3.10.27 IrDA Control Register

3.11 Serial Bus Interface (SBI)

The TMP91CP27 has a 1-channel serial bus interface which employs a clocked-synchronous 8-bit SIO mode and an I^{2}C bus mode (Multi muster).

The serial bus interface is connected to an external device through P61 (SDA) and P62 (SCL) in the I^{2}C bus mode; and through P60 (SCK), P61 (SO), and P62 (SI) in the clocked-synchronous 8-bit SIO mode.

Each pin is specified as follows.

	ODE <ODE62, 61>	P6CR <P62C:60C>	P6FC <P62F:60F>
I^{2}C bus mode	11	11X	11X
Clock synchronous 8-bit SIO mode	XX	011 010	X11

X: Don't care

3.11.1 Configuration

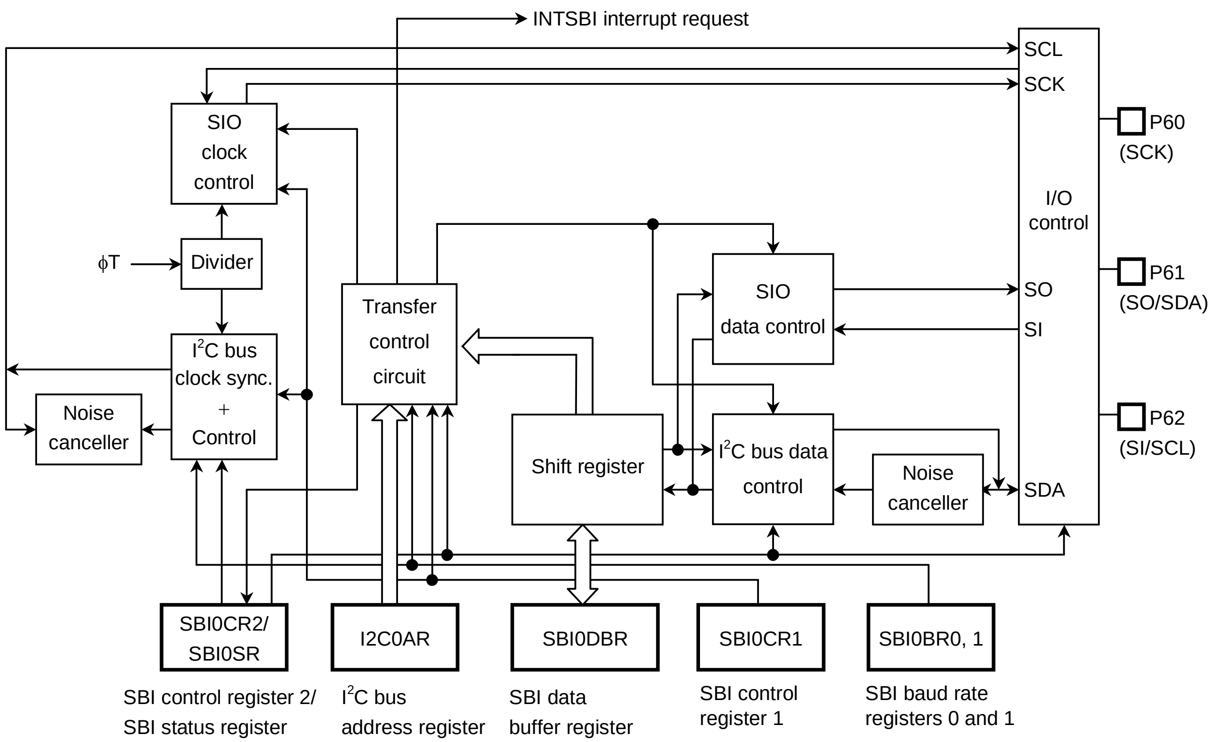


Figure 3.11.1 Serial Bus Interface (SBI)

3.11.2 Control

The following registers are used to control the serial bus interface and monitor the operation status.

- Serial bus interface control register 1 (SBI0CR1)
- Serial bus interface control register 2 (SBI0CR2)
- Serial bus interface data buffer register (SBI0DBR)
- I^2C bus address register (I2C0AR)
- Serial bus interface status register (SBI0SR)
- Serial bus interface baud rate register 0 (SBI0BR0)
- Serial bus interface baud rate register 1 (SBI0BR1)

The above registers differ depending on a mode to be used.

Refer to Section, "3.11.4 I^2C Bus Mode Control Register" and "3.11.7 Clocked Synchronous 8-Bit SIO Mode Control."

3.11.3 Data Format in I^2C Bus Mode

Data format in I^2C bus mode is shown Figure 3.11.2.

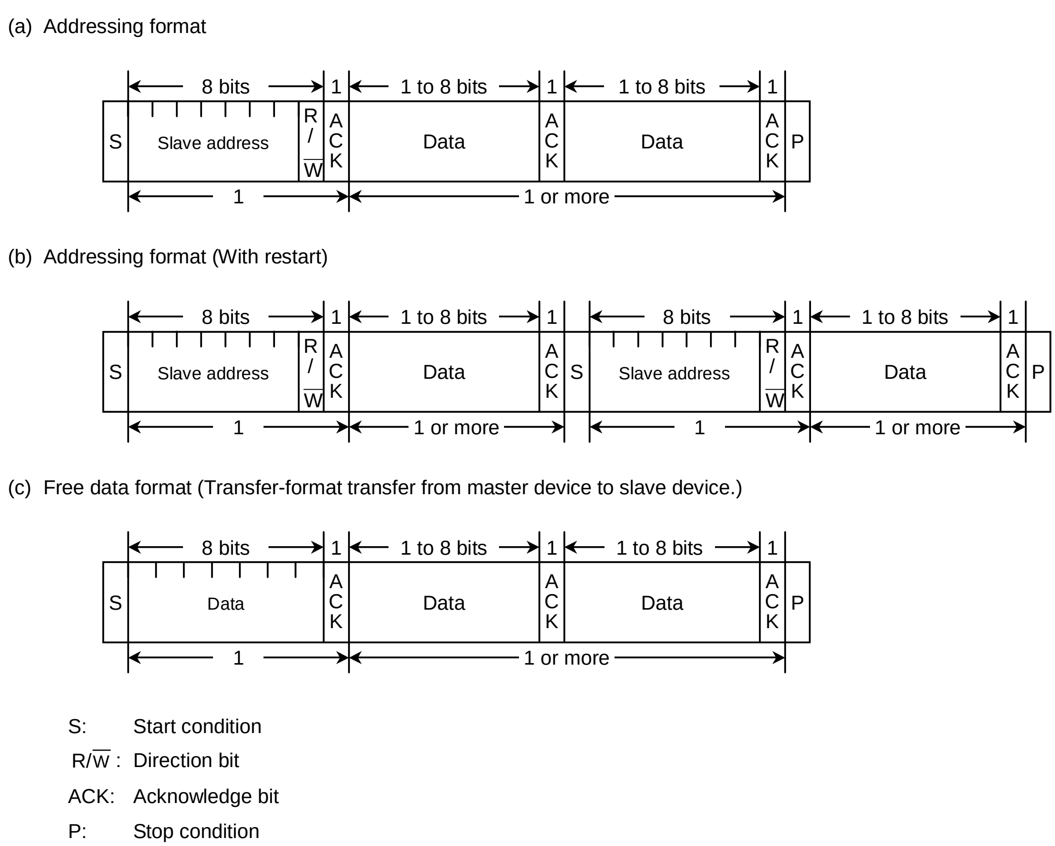


Figure 3.11.2 Data Format in I^2C Bus Mode

3.11.4 I²C Bus Mode Control Register

The following registers are used to control and monitor the operation status when using the serial bus interface (SBI) in the I²C bus mode.

Serial Bus Interface Control Register 1

		7	6	5	4	3	2	1	0
SBI0CR1 (0240H)	Bit symbol	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0/ SWRMON
	Read/Write	W			R/W		W		R/W
	After reset	0	0	0	0		0	0	0/1 (Note3)
Read-modify-write instruction is prohibited.	Function	Select number of transferred bits (Note 1)			Acknowledge mode specification 0: Not generate 1: Generate		Internal serial clock selection and software reset monitor (Note 2)		

Internal serial clock selection <SCK2:0> @ write

000	n = 5	400 kHz	System clock: fc Clock gear: fc/1 fc = 16 MHz (Output to internal SCL) Frequency = $\frac{fc}{2^n + 8}$ [Hz]
001	n = 6	222 kHz	
010	n = 7	118 kHz	
011	n = 8	60.6 kHz	
100	n = 9	30.8 kHz	
101	n = 10	15.5 kHz	
110	n = 11	7.78 kHz	
111	Reserved	(Reserved)	

Software reset state monitor <SWRMON> @ read

0	During software reset
1	(Default value)

Acknowledge mode selection

0	Not generate clock pulse for acknowledge signal
1	Generate clock pulse for acknowledge signal

Select number of bits transferred

<BC2:0>	<ACK> = 0		<ACK> = 1	
	Number of clock pulses	Data length	Number of clock pulses	Data length
000	8	8	9	8
001	1	1	2	1
010	2	2	3	2
011	3	3	4	3
100	4	4	5	4
101	5	5	6	5
110	6	6	7	6
111	7	7	8	7

Note 1: Set the <BC2:0> to “000” before switching to a clock-synchronous 8-bit SIO mode.

Note 2: For the frequency of the SCL line clock, see section 3.11.5, (3) “Serial clock”.

Note 3: After reset, default value of <SCK0> is cleared to “0”. Also, default value of <SWRMON> is set to “1”.

Figure 3.11.3 Register for I²C Bus Mode

Serial Bus Interface Control Register 2

SBI0CR2 (0243H)

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
Read/Write	W				W (Note 1)		W (Note 1)	
After reset	0	0	0	1	0	0	0	0
Function	Master/ slave selection	Transmitter/ receiver selection	Start/stop condition generation	Cancel INTSBI interrupt request	Serial bus interface operation mode selection (Note 2) 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)		Software reset control write "10" and "01" in order, then an internal reset signal is generated.	

Read-modify-write instruction is prohibited.

Serial bus interface operating mode selection (Note 2)

00	Port mode (Serial bus interface output disabled)
01	Clocked synchronous 8-bit SIO mode
10	I^2C bus mode
11	(Reserved)

INTSBI interrupt request

0	–
1	Cancel interrupt request

Start/stop condition generation

0	Generates the stop condition
1	Generates the start condition

Transmitter/receiver selection

0	Receiver
1	Transmitter

Master/slave selection

0	Slave
1	Master

Note 1: Reading this register function as SBI0SR register.

Note 2: Switch a mode to port mode after confirming that the bus is free.
Switch a mode between I^2C bus mode and clock-synchronous 8-bit SIO mode after confirming that input signals via port are high level.

Figure 3.11.4 Register for I^2C Bus Mode

Serial Bus Interface Status Register

SBI0SR (0243H)

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
Read/Write	R							
After reset	0	0	0	1	0	0	0	0
Function	Master/ slave status selection monitor	Transmitter/ receiver status selection monitor	I^2C bus status monitor	INTSBI interrupt request monitor	Arbitration lost detection monitor 0: – 1: Detected	Slave address match detection monitor 0: Undetected 1: Detected	GENERAL CALL detection monitor 0: Undetected 1: Detected	Last received bit monitor 0: 0 1: 1

Read-modify-write instruction is prohibited.

Last received bit monitor

0	Last received bit was 0.
1	Last received bit was 1.

GENERAL CALL detection monitor

0	Undetected
1	GENERAL CALL detected

Slave address match detection monitor

0	Undetected
1	Slave address match or GENERAL CALL detected

Arbitration lost detection monitor

0	–
1	Arbitration lost

INTSBI interrupt request monitor

0	Interrupt requested
1	Interrupt canceled

I^2C bus status monitor

0	Free
1	Busy

Transmitter/receiver status monitor

0	Receiver
1	Transmitter

Master/slave status monitor

0	Slave
1	Master

Note: Writing in this register functions as SBI0CR2.

Figure 3.11.5 Register for I^2C Bus Mode

Serial Bus Interface Baud Rate Regster 0

SBI0BR0 (0244H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI0						
Read/Write	W	R/W						
After reset	0	0						
Function	Write "0".	IDLE2 0: Stop 1: Run						

Operation during IDLE2 mode

0	Stop
1	Run

Serial Bus Interface Baud Rate Register 1

SBI0BR1 (0245H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	P4EN	–						
Read/Write	W	W						
After reset	0	0						
Function	Internal clock 0: Stop 1: Run	Write "0".						

Baud rate clock control

0	Stop
1	Run

Sirial Bus Interface Data Buffer Register

SBI0DBR (0241H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receiving)/W (Transmission)							
After reset	Undefined							

Note 1: When writing transmitted data, start from the MSB (Bit7). Receiving data is placed from LSB (Bit0).

Note 2: SBI0DBR can't be read the written data. Therefore read-modify-write insturction (e.g., "BIT" instruction) is prohibitted.

Note 3: Written data in SBI0DBR is cleared by INTSBI signal.

I^2C Bus Address Register

I2C0AR (0242H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Slave address selection for when device is operating as slave device							Address recognition mode specification

Address recognition mode specification

0	Slave address recognition
1	Non slave address recognition

Figure 3.11.6 Register for I^2C Bus Mode

3.11.5 Control in I²C Bus Mode

(1) Acknowledge mode specification

Set the SBI0CR1<ACK> to 1 for operation in the acknowledge mode. The TMP91CP27 generates an additional clock pulse for an acknowledge signal when operating in master mode. In the transmitter mode during the clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the low in order to generate the acknowledge signal.

Clear the <ACK> to 0 for operation in the non-acknowledge mode, the TMP91CP27 does not generate a clock pulse for the acknowledge signal when operating in the master mode.

(2) Select number of transfer bits

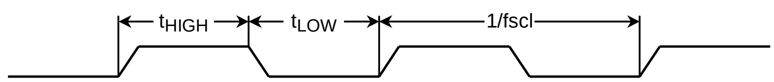
The SBI0CR1<BC2:0> is used to select a number of bits for next transmission and receiving data.

Since the <BC2:0> is cleared to 000 as a start condition, a slave address and direction bit transmission are executed in 8 bits. Other than these, the <BC2:0> retains a specified value.

(3) Serial clock

a. Clock source

The SBI0CR1<SCK2:0> is used to select a maximum transfer frequency outputted on the SCL pin in master mode.



$$t_{LOW} = 2^{n-1}/f_{SBI}$$
$$t_{HIGH} = 2^{n-1}/f_{SBI} + 8/f_{SBI}$$
$$fscl = 1/(t_{Low} + t_{HIGH}) = \frac{f_{SBI}}{2^n + 8}$$

SBI0CR1<SCK2:0>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

Note 1: f_{SBI} shows f_{FPH}.

Note 2: It's prohibit to use fc/16 prescaler clock (SYSCR0<PRCK1:0> = "10") when using SBI block. (I²C bus and clocked synchronous)

Figure 3.11.7 Clock Source

b. Clock synchronization

In the I²C bus mode, in order to wired-AND a bus, a master device which pulls down a clock line to low-level, in the first place, invalidate a clock pulse of another master device which generates a high-level clock pulse. The master device with a high-level clock pulse needs to detect the situation and implement the following procedure.

The TMP91CP27 has a clock synchronization function for normal data transfer even when more than one master exists on the bus.

The example explains the clock synchronization procedures when two masters simultaneously exist on a bus.

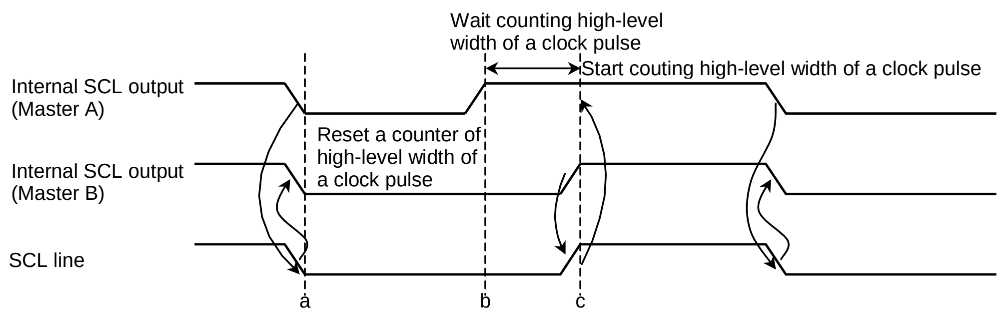


Figure 3.11.8 Clock Synchronization

As master A pulls down the internal SCL output to the low level at point “a”, the SCL line of the bus becomes the low level. After detecting this situation, Master B resets a counter of high-level width of an own clock pulse and sets the internal SCL output to the low level.

Master A finishes counting low-level width of an own clock pulse at point “b” and sets the internal SCL output to the high level. Since master B holds the SCL line of the bus at the low level, master A wait for counting high-level width of an own clock pulse. After master B finishes counting low-level width of an own clock pulse at point “c” and master A detects the SCL line of the bus at the high level, and starts counting high level of an own clock pulse. The clock pulse on the bus is determined by the master device with the shortest high-level width and the master device with the longest low-level width from among those master devices connected to the bus.

(4) Slave address and address recognition mode specification

When the TMP91CP27 is used as a slave device, set the slave address <SA6:0> and <ALS> to the I2C0AR. Clear the <ALS> to “0” for the address recognition mode.

(5) Master/slave selection

Set the SBI0CR2<MST> to “1” for operating the TMP91CP27 as a master device. Clear the SBI0CR2<MST> to “0” for operation as a slave device. The <MST> is cleared to “0” by the hardware after a stop condition on the bus is detected or arbitration is lost.

(6) Transmitter/receiver selection

Set the SBI0CR2<TRX> to “1” for operating the TMP91CP27 as a transmitter. Clear the <TRX> to “0” for operation as a receiver. In slave mode, when transfer data in addressing format, when received slave address is same value with setting value to I2C0AR, or GENERAL CALL is received (All 8-bit data are “0” after a start condition), the <TRX> is set to “1” by the hardware if the direction bit (R/$\overline{W}$) sent from the master device is “1”, and <TRX> is cleared to “0” by the hardware if the bit is “0”. In the Master Mode, after an Acknowledge signal is returned from the slave device, the <TRX> is cleared to “0” by the hardware if a transmitted direction bit is “1”, and is set to “1” by the hardware if it is “0”. When an acknowledge signal is not returned, the current condition is maintained.

The <TRX> is cleared to “0” by the hardware after a stop condition on the bus is detected or arbitration is lost.

(7) Start/stop condition generation

When the SBI0SR<BB> is “0”, slave address and direction bit which are set to SBI0DBR are output on a bus after generating a start condition by writing “1” to the SBI0CR2<MST, TRX, BB, PIN>. It is necessary to set transmitted data to the data buffer register (SBI0DBR) and set “1” to <ACK> beforehand.

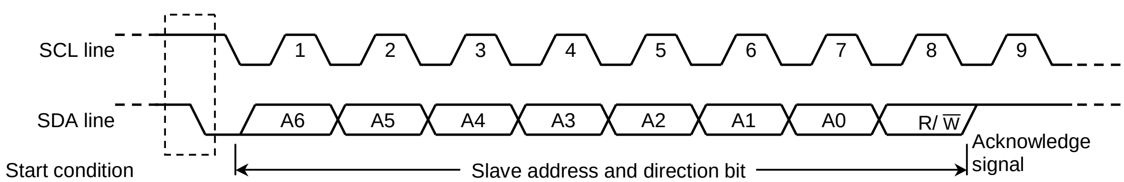


Figure 3.11.9 Generation of Start Condition and Slave Address

When the <BB> is “1”, a sequence of generating a stop condition is started by writing “1” to the <MST, TRX, PIN>, and “0” to the <BB>. Do not modify the contents of <MST, TRX, BB, PIN> until a stop condition is generated on a bus.

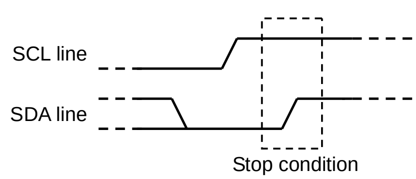


Figure 3.11.10 Generation of Stop Condition

The state of the bus can be ascertained by reading the contents of SBI0SR<BB>. SBI0SR<BB> will be set to 1 if a start condition has been detected on the bus, and will be cleared to 0 if a stop condition has been detected (Bus free status).

And about generation of stop condition in master mode, there are some limitation points. Please refer to section 3.11.6, (4) “Stop condition generation”.

(8) Interrupt service requests and interrupt cancellation

When a serial bus interface interrupt request (INTSBI) occurs, the SBI0CR2<PIN> is cleared to "0". During the time that the SBI0CR2<PIN> is "0", the SCL line is pulled down to the low level.

The <PIN> is cleared to "0" when a 1 word of data is transmitted or received. Either writing/reading data to/from SBI0DBR sets the <PIN> to "1".

The time from the <PIN> being set to "1" until the SCL line is released takes t_{LOW}.

In the address recognition mode (<ALS> = 0), <PIN> is cleared to "0" when the received slave address is the same as the value set at the I2C0AR or when a GENERAL CALL is received (All 8-bit data are "0" after a start condition). Although SBI0CR2 <PIN> can be set to "1" by the program, the <PIN> is not clear it to "0" when it is written "0".

(9) Serial bus interface operation mode selection

SBI0CR2<SBIM1:0> is used to specify the serial bus interface operation mode. Set SBI0CR2<SBIM1:0> to "10" when the device is to be used in I^2C bus mode after confirming pin condition of serial bus interface to "H".

Switch a mode to port after confirming a bus is free.

(10) Arbitration lost detection monitor

Since more than one master device can exist simultaneously on the bus in I^2C bus mode, a bus arbitration procedure has been implemented in order to guarantee the integrity of transferred data.

Data on the SDA line is used for I^2C bus arbitration.

The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on the bus. Master A and master B output the same data until point "a". After master A outputs "L" and master B, "H", the SDA line of the bus is wire-AND and the SDA line is pulled down to the low level by master A. When the SCL line of the bus is pulled up at point b, the slave device reads the data on the SDA line, that is, data in master A. A data transmitted from master B becomes invalid. The state in master B is called "ARBITRATION LOST". Master B device that loses arbitration releases the internal SDA output in order not to affect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.

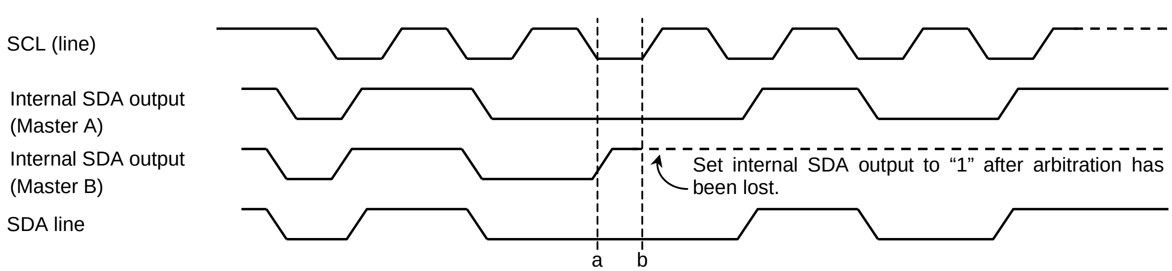


Figure 3.11.11 Arbitration Lost

The TMP91CP27 compares the levels on the bus's SDA line with those of the internal SDA output on the rising edge of the SCL line. If the levels do not match, arbitration is lost and SBI0SR<AL> is set to "1".

When SBI0SR<AL> is set to "1", SBI0SR<MST, TRX> are cleared to "0" and the mode is switched to slave receiver mode. Thus, clock output is stopped in data transfer after setting <AL> = "1".

SBI0SR<AL> is cleared to "0" when data is written to or read from SBI0DBR or when data is written to SBI0CR2.

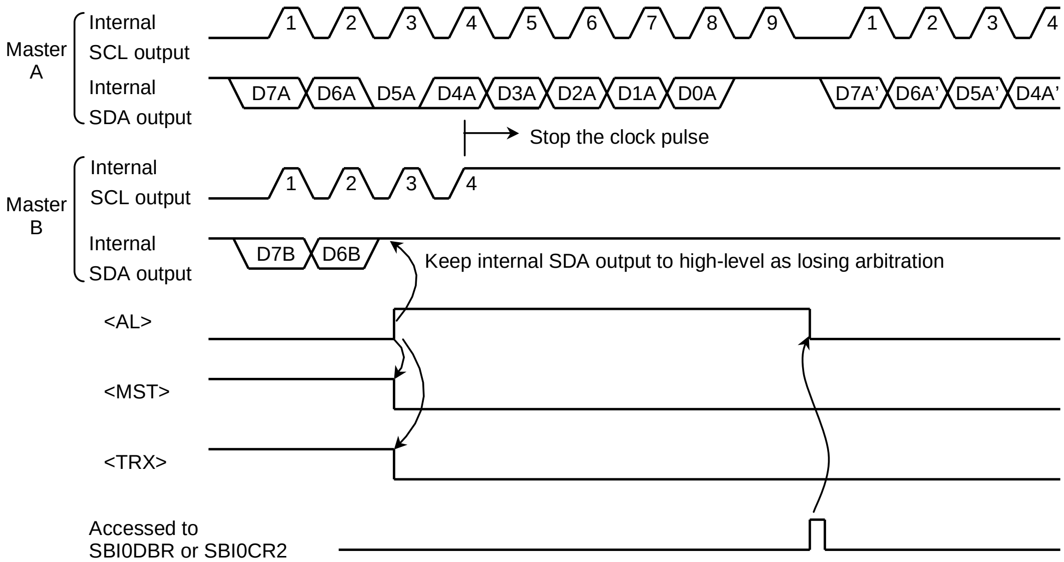


Figure 3.11.12 Example of When TMP91CP27 is a Master Device B (D7A = D7B, D6A = D6B)

(11) Slave address match detection monitor

SBI0SR<AAS> is set to "1" in slave mode, in address recognition mode (e.g., when I2C0AR<ALS> = "0"), when a GENERAL CALL is received, or when a slave address matches the value set in I2C0AR. When I2C0AR<ALS> = "1", SBI0SR<AAS> is set to "1" after the first word of data has been received. SBI0SR<AAS> is cleared to "0" when data is written to or read from the data buffer register SBI0DBR.

(12) General call detection monitor

SBI0SR<AD0> is set to "1" in slave mode, when a GENERAL CALL is received (All 8-bit received data is "0" after a start condition). SBI0SR<AD0> is cleared to "0" when a start condition or stop condition is detected on the bus.

(13) Last received bit monitor

The SDA line value stored at the rising edge of the SCL line is set to the SBI0SR<LRB>. In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the SBI0SR<LRB>.

(14) Software reset function

The software reset function is used to initialize the SBI circuit, when SBI is rocked by external noises, etc.

An internal reset signal pulse can be generated by setting SBI0CR2<SWRST1:0> to "10" and "01". This initializes the SBI circuit internally. All command (except SBI0CR2<SBIM1:0>) registers and status registers are initialized as well.

SBI0CR1<SWRMON> is automatically set to "1" after the SBI circuit has been initialized.

(15) Serial bus interface data buffer register (SBI0DBR)

The received data can to read and transmission data can to write by reading or writing SBI0DBR.

In the master mode, after the start condition is generated the slave address and the direction bit are set in this register.

(16) I^2C bus address register (I2C0AR)

I2C0AR<SA6:0> is used to set the slave address when the TMP91CP27 functions as a slave device.

The slave address outputted from the master device is recognized by setting the I2C0AR<ALS> to "0". The data format is the addressing format. When the slave address is not recognized at the <ALS> = "1", the data format is the free data format.

(17) Baud rate register (SBI0BR1)

Write "1" to SBI0BR1<P4EN> before operation commences.

(18) Setting register for IDLE2 mode operation (SBI0BR0)

SBI0BR0<I2SBI0> is the register setting operation/stop during IDLE2 mode. Therefore, setting <I2SBI0> is necessary before the HALT instruction is executed.

3.11.6 Data Transfer In I^2C Bus Mode

(1) Device initialization

Set the SBI0BR1<P4EN>, SBI0CR1<ACK, SCK2:0>, set SBI0BR1 to “1” and clear bits 7 to 5 and 3 in the SBI0CR1 to “0”.

Set a slave address <SA6:0> and the <ALS> (<ALS> = “0” when an addressing format) to the I2C0AR.

For specifying the default setting to a slave receiver mode, clear “0” to the <MST, TRX, BB> and set “1” to the <PIN>, “10” to the <SBIM1:0>.

(2) Start condition and slave address generation

a. Master mode

In the master mode, the start condition and the slave address are generated as follows.

Check a bus free status (when <BB> = “0”).

Set the SBI0CR1<ACK> to “1” (Acknowledge mode) and specify a slave address and a direction bit to be transmitted to the SBI0DBR.

When SBI0CR2<BB> = “0”, the start condition are generated by writing “1111” to SBI0CR2<MST, TRX, BB, PIN>. Subsequently to the start condition, nine clocks are output from the SCL pin. While eight clocks are output, the slave address and the direction bit which are set to the SBI0DBR. At the 9th clock, the SDA line is released and the acknowledge signal is received from the slave device.

An INTS2 interrupt request occurs at the falling edge of the 9th clock. The <PIN> is cleared to “0”. In the master mode, the SCL pin is pulled down to the low level while <PIN> is “0”. When an interrupt request occurs, the <TRX> is changed according to the direction bit only when an acknowledge signal is returned from the slave device.

b. Slave mode

In the slave mode, the start condition and the slave address are received.

After the start condition is received from the master device, while eight clocks are output from the SCL pin, the slave address and the direction bit that are output from the master device are received.

When a GENERAL CALL or the same address as the slave address set in I2C0AR is received, the SDA line is pulled down to the low level at the 9th clock, and the acknowledge signal is output.

An INTSBI interrupt request generate on the falling edge of the 9th clock. The <PIN> is cleared to “0”. In slave mode the SCL line is pulled down to the low level while the <PIN> = “0”.

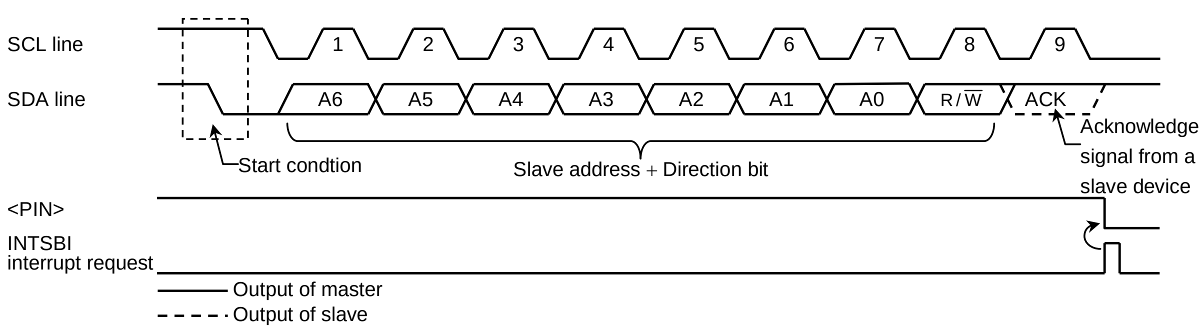


Figure 3.11.13 Start Condition and Slave Address Generation

(3) 1-word data transfer

Check the <MST> by the INTSBI interrupt process after the 1-word data transfer is completed, and determine whether the mode is a master or slave.

a. If <MST> = “1” (Master mode)

Check the <TRX> and determine whether the mode is a transmitter or receiver.

When the <TRX> = “1” (Transmitter mode)

- Check the <LRB>. When <LRB> is “1”, a receiver does not request data. Implement the process to generate a stop condition (Refer to 3.11.6 (4)) and terminate data transfer.
  When the <LRB> is “0”, the receiver requests new data. When the next transmitted data is 8 bits, write the transmitted data to SBI0DBR. When the next transmitted data is other than 8 bits, set the <BC2:0> <ACK> and write the transmitted data to SBI0DBR. After written the data, <PIN> becomes “1”, a serial clock pulse is generated for transferring a new 1 word of data from the SCL pin, and then the 1-word data is transmitted. After the data is transmitted, an INTSBI interrupt request generates. The <PIN> becomes “0” and the SCL line is pulled down to the low level. If the data to be transferred is more than one word in length, repeat the procedure from the <LRB> checking above.

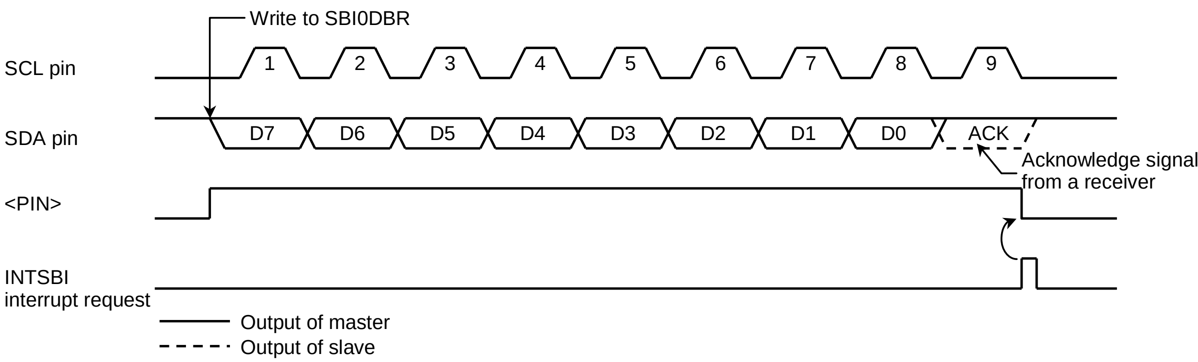


Figure 3.11.14 Example in Which <BC2:0> = “000” and <ACK> = “1” (Transmitter mode)

When the <TRX> is "0" (Receiver mode)

When the next transmitted data is other than 8 bits, set <BC2:0> <ACK> and read the received data from SBI0DBR to release the SCL line (Data which is read immediately after a slave address is sent is undefined). After the data is read, <PIN> becomes "1". Serial clock pulse for transferring new 1 word of data is defined SCL and outputs "L" level from SDA pin with acknowledge timing.

An INTSBI interrupt request then generates and the <PIN> becomes "0", Then the TMP91CP27 pulls down the SCL pin to the low level. The TMP91CP27 outputs a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBI0DBR.

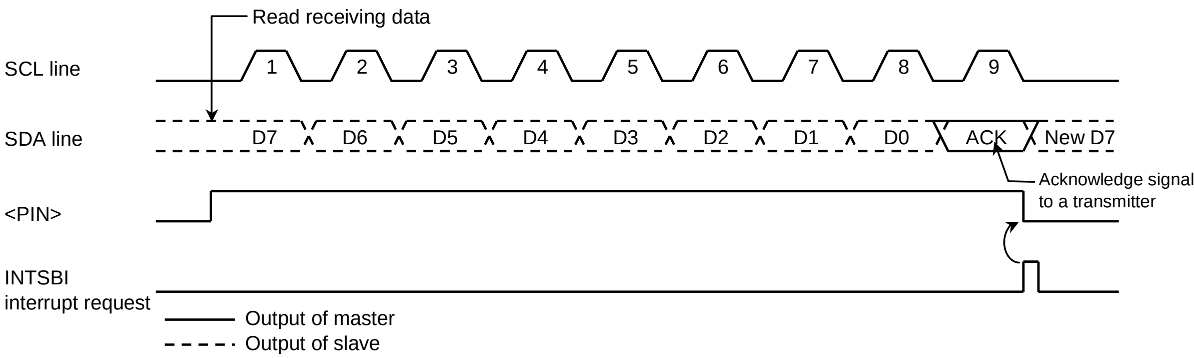


Figure 3.11.15 Example of When <BC2:0> = "000" and <ACK> = "1" (Receiver mode)

In order to terminate the transmission of data to a transmitter, clear <ACK> to "0" before reading data which is 1-word before the last data to be received. The last data word does not generate a clock pulse as the acknowledge signal. After the data has been transmitted and an interrupt request has been generated, set <BC2:0> to "001" and read the data. The TMP91CP27 generates a clock pulse for a 1-bit data transfer. Since the master device is a receiver, the SDA line on the bus remains high. The transmitter receives the high signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete.

After the one data bit has been received and an interrupt request been generated, the TMP91CP27 generates a stop condition (See Section 3.11.6 (4)) and terminates data transfer.

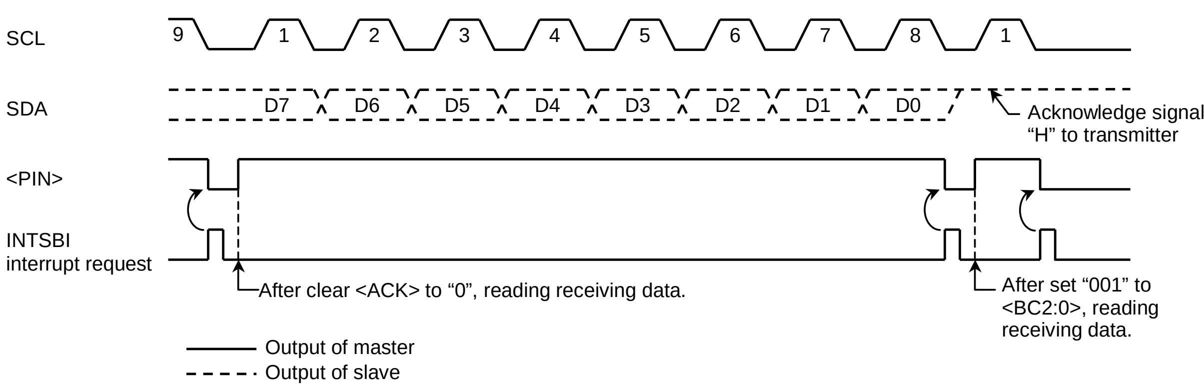


Figure 3.11.16 Termination of Data Transfer (Master receiver mode)

b. If <MST> = 0 (Slave mode)

In the slave mode the TMP91CP27 operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, an INTSBI interrupt request occurs when the TMP91CP27 receives a slave address or a GENERAL CALL from the master device, or when a GENERAL CALL is received and data transfer is complete, or after matching received address. In the master mode, the TMP91CP27 operates in a slave mode if it losing arbitration. An INTSBI interrupt request generate when a word data transfer terminates after losing arbitration. When an INTSBI interrupt request generate the <PIN> is cleared to “0” and the SCL pin is pulled down to the low level. Either reading/writing from/to the SBI0DBR or setting the <PIN> to “1” will release the SCL pin after taking t_{LOW} time.

Check the SBI0SR<AL>, <TRX>, <AAS>, and <AD0> and implements processes according to conditions listed in the next table.

Table 3.11.1 Operation in the Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	Conditions	Process
1	1	1	0	The TMP91CP27 loses arbitration when transmitting a slave address, and receives a slave address for which the value of the direction bit sent from another master is “1”.	Set the number of bits of single word to <BC2:0>, and write the transmit data to SBI0DBR
	0	1	0	In slave receiver mode, the TMP91CP27 receives a slave address for which the value of the direction bit sent from the master is “1”.	
		0	0	In slave transmitter mode, transmission of data of single word is terminated.	Check the <LRB> setting. If <LRB> is set to “1”, set <PIN> to “1” since the receiver win no request the data which follows. Then, clear <TRX> to “0” to release the bus. If <LRB> is cleared to “0” of and write the transmitted data to SBI0DBR since the receiver requests next data.
0	1	1	1/0	The TMP91CP27 loses arbitration when transmitting a slave address, and receives a slave address or GENERAL CALL for which the value of the direction bit sent from another master is “0”.	Read the SBI0DBR for setting the <PIN> to “1” (reading dummy data) or set the <PIN> to “1”.
		0	0	The TMP91CP27 loses arbitration when transmitting a slave address or data, and terminates word data transfer.	
	0	1	1/0	In slave receiver mode the TMP91CP27 receives a slave address or GENERAL CALL for which the value of the direction bit sent from the master is “0”.	
		0	1/0	In slave receiver mode the TMP91CP27 terminates receiving word data.	Set <BC2:0> to the number of bits in a word and read the received data from SBI0DBR.

(4) Stop condition generation

When SBI0SR<BB> = 1, the sequence for generating a stop condition start by writing “1” to SBI0CR2<MST, TRX, PIN> and “0” to SBI0CR2<BB>. Do not modify the contents of SBI0CR2<MST, TRX, PIN, BB> until a stop condition has been generated on the bus. When the bus’s SCL line has been pulled Low by another device, the TMP91CP27 generates a stop condition when the other device has released the SCL line and SDA pin rising.

When SBI0CR2<MST, TRX, PIN> are written “1” and <BB> is written “0” (Generate stop condition in master mode), <BB> changes to “0” by internal SCL changes to “1”, without waiting stop condition.To check whether SCL and SDA pin are “1” by sensing their ports is needed to detect bus free condition.

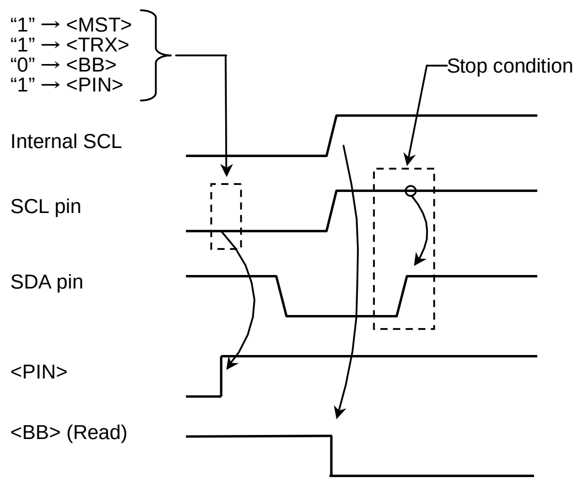


Figure 3.11.17 Stop Condition Generation (Single master)

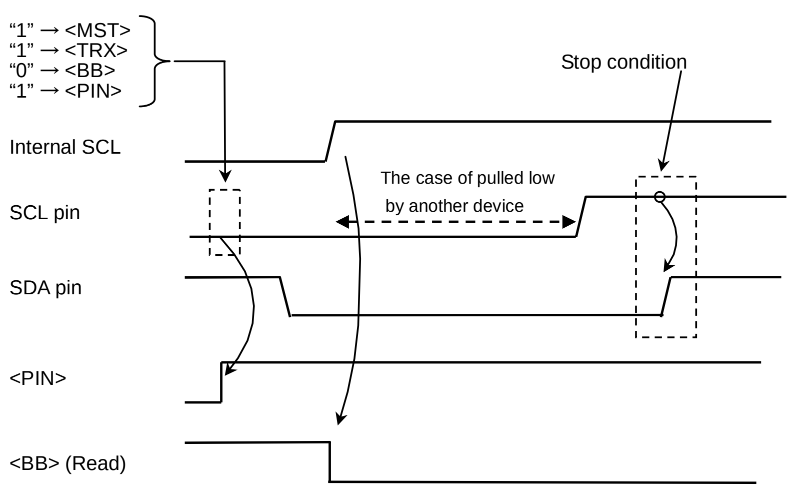


Figure 3.11.18 Stop Condition Generation (Multi master)

(5) Restart

Restart is used during data transfer between a master device and a slave device to change the data transfer direction.

The following description explains how to restart when the TMP91CP27 is in master mode.

Clear SBI0CR2<MST, TRX, BB> to 0 and set SBI0CR2<PIN> to 1 to release the bus. The SDA line remains high and the SCL pin is released. Since a stop condition has not been generated on the bus, other devices assume the bus to be in busy state.

And confirm SCL pin, that SCL pin is released and become bus-free state by SBI0SR <BB> = "0" or signal level "1" of SCL pin in port mode. Check the <LRB> until it becomes 1 to check that the SCL line on a bus is not pulled down to the low level by other devices. After confirming that the bus remains in a free state, generate a start condition using the procedure described in 3.11.6 (2).

In order to satisfy the setup time requirements when restarting, take at least 4.7 μs of waiting time by software from the time of restarting to confirm that the bus is free until the time to generate the start condition.

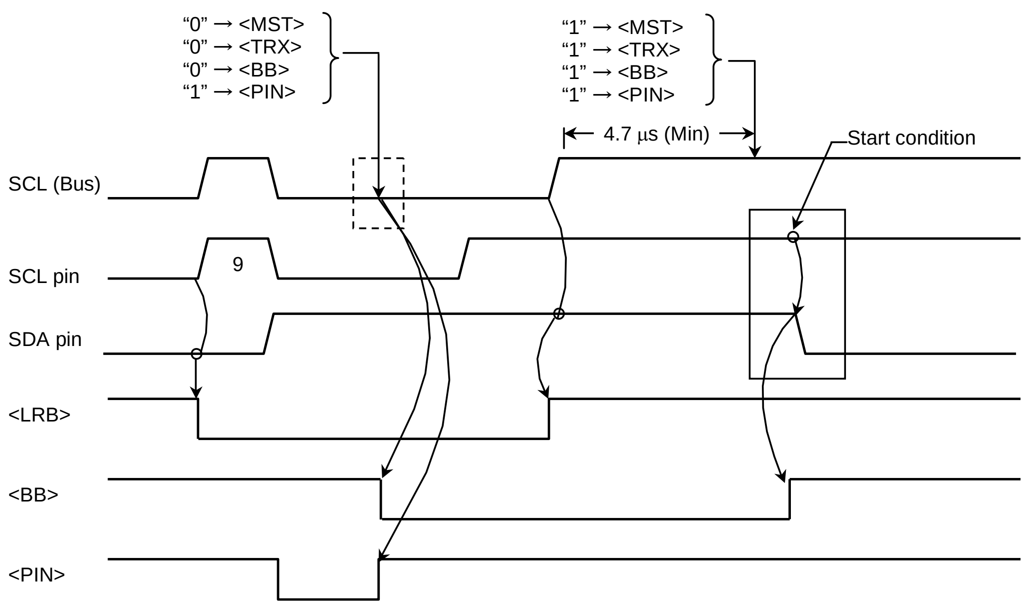


Figure 3.11.19 Timing Chart for Generate Restart

3.11.7 Clocked Synchronous 8-Bit SIO Mode Control

The following registers are used to control and monitor the operation status when the serial bus interface (SBI) is being operated in clocked synchronous 8-bit SIO mode.

Serial Bus Interface Control Register 1

SBI0CR1 (0240H)

		7	6	5	4	3	2	1	0
SBI0CR1 (0240H)	Bit symbol	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
	Read/Write	W					W		
	After reset	0	0	0	0		0	0	0
Read-modify-write instruction is prohibited.	Function	Transfer start/stop 0: Stop 1: Start	Continue/ abort transfer 0: Continue transfer 1: Abort transfer	Transfer mode selection 00: Transmission mode 01: (Reserved) 10: Transmission/ receiving mode 11: Receiving mode			Selection of serial clock frequency		

Serial clock selection <SCK2:0>

000	n = 4	1 MHz	System clcok: fc Clock gear: fc/1 fc = 16 MHz (Output to SCK pin) frequency = $\frac{fc}{2^n}$ [Hz]
001	n = 5	500 kHz	
010	n = 6	250 kHz	
011	n = 7	125 kHz	
100	n = 8	62.5 kHz	
101	n = 9	31.25 kHz	
110	n = 10	15.625 kHz	
111	–	External clock	(Input from SCK Pin)

Transfer mode selection

00	8-bit transmission mode
01	(Reserved)
10	8-bit transmission/receiving mode
11	8-bit receiving mode

Continue/Abort transfer

0	Continue transfer
1	Abort transfer (Automatically cleared after transfer aborted)

Transfer start/stop

0	Stopped
1	Started

Note: Set the tranfer mode and the serial clock after setting <SIOS> to “0” and <SIOINH> to “1”.

Serial Bus Interface Data Buffer Register

		7	6	5	4	3	2	1	0
SBI0DBR (0241H)	Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read-modify-write instruction is prohibited.	Read/Write	R (Receiver)/W (Transfer)							
	After reset	Undefined							

Figure 3.11.20 Register for SIO Mode

Serial Bus Interface Control Register 2

SBI0CR2 (0243H)		7	6	5	4	3	2	1	0
	Bit symbol					SBIM1	SBIM0	–	–
	Read/Write					W		W	W
	After reset					0	0	0	0
Read-modify-write instruction is prohibited.	Function					Serial bus interface operation mode selection 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)		Write “0”.	

Serial bus interface operation mode selection

00	Port mode (serial bus interface output disabled)
01	Clocked-synchronous 8-bit SIO mode
10	I^2C bus mode
11	(Reserved)

Note 1: Set the SBI0CR1<BC2:0> to “000” before switching to a clocked-synchronous 8-bit SIO mode.

Note 2: Please always write “00” to SBICR2<1:0>.

Serial Bus Interface Status Register

SBI0SR (0243H)		7	6	5	4	3	2	1	0
	Bit symbol					SIOF	SEF		
	Read/Write					R			
	After reset					0	0		
	Function					Serial transfer operation status monitor	Shift operation status monitor		

Shift operation status monitor

0	Shift operation terminated
1	Shift operation in progress

Serial transfer operating status monitor

0	Transfer terminated
1	Transfer in progress

Figure 3.11.21 Register for SIO Mode

Serial Bus Interface Baud Rate Register 0

SBI0BR0 (0244H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI0						
Read/Write	W	R/W						
After reset	0	0						
Function	Write "0".	IDLE2 0: Stop 1: Operate						

Operation in IDLE2 mode

0	Stop
1	Operate

Serial Bus Interface Baud Rate Register 1

SBI0BR1 (0245H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	P4EN	–						
Read/Write	W	W						
After reset	0	0						
Function	Internal clock 0: Stop 1: Operate	Write "0".						

Baud rate clock control

0	Stop
1	Operate

Figure 3.11.22 Register for SIO Mode

(1) Serial clock

a. Clock source

SBI0CR1<SCK2:0> is used to select the following functions:

Internal clock

In internal clock mode one of seven frequencies can be selected. The serial clock signal is output to the outside on the SCK pin. The SCK pin goes high when data transfer starts. When the device is writing (in transmit mode) or reading (in receive mode), data cannot follow the serial clock rate, so an automatic wait function is executed which automatically stops the serial clock and holds the next shift operation until reading or writing has been completed.

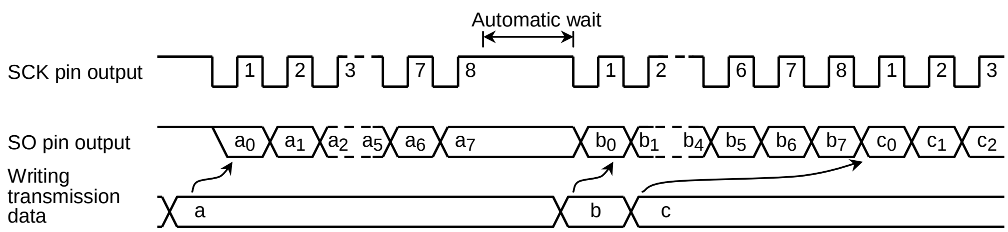


Figure 3.11.23 Automatic Wait Function

External clock (<SCK2:0> = “111”)

An external clock input via the SCK pin is used as the serial clock. In order to ensure the integrity of shift operations, both the high and low-level serial clock pulse widths shown below must be maintained. The maximum data transfer frequency is 1 MHz (when fc = 16 MHz).

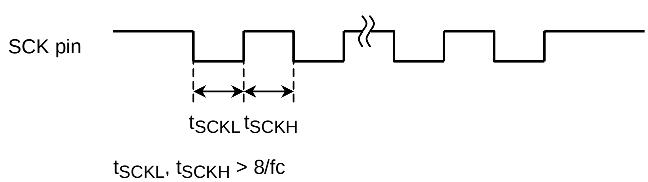


Figure 3.11.24 Maximum Data Transfer Frequency When External Clock Input Used

b. Shift edge

Data is transmitted on the leading edge of the clock and received on the trailing edge.

Leading edge shift

Data is shifted on the leading edge of the serial clock (on the falling edge of the SCK pin input/output).

Trailing edge shift

Data is shifted on the trailing edge of the serial clock (on the rising edge of the SCK pin input/output).

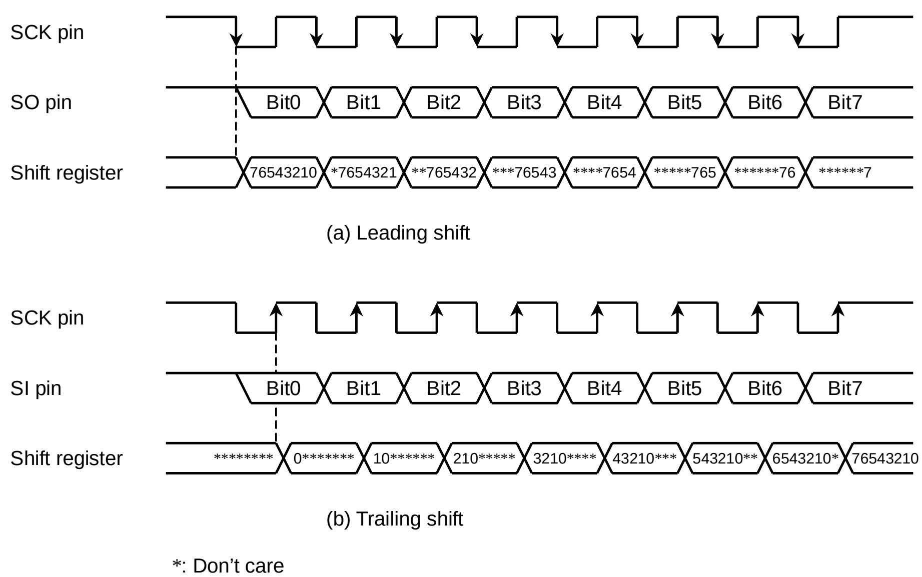


Figure 3.11.25 Shift Edge

(2) Transfer modes

The SBI0CR1<SIOM1:0> is used to select a transmit, receive or transmit/receive mode.

a. 8-bit transmit mode

Set a control register to a transmit mode and write transmit data to the SBI0DBR.

After the transmit data is written, set the SBI0CR1<SIOS> to "1" to start data transfer. The transmitted data is transferred from SBI0DBR to the shift register and output to the SO pin in synchronized with the serial clock, starting from the least significant bit (LSB). When the transmission data is transferred to the shift register, the SBI0DBR becomes empty. An INTSBI (Buffer empty) interrupt request is generated to request new data.

When the internal clock is used, the serial clock will stop and automatic-wait function will be initiated if new data is not loaded to the data buffer register after the specified 8-bit data is transmitted. When new transmit data is written, automatic-wait function is canceled.

When the external clock is used, data should be written to SBI0DBR before new data is shifted. The transfer speed is determined by the maximum delay time between the time when an interrupt request is generated and the time when data is written to SBI0DBR by the interrupt service program.

When the transmit is started, after the SBI0SR<SIOF> goes "1" output from the SO pin holds final bit of the last data until falling edge of the SCK.

Transmitting data is ended by clearing the <SIOS> to "0" by the buffer empty interrupt service program or setting the <SIOINH> to "1". When the <SIOS> is cleared, the transmitted mode ends when all data is output. In order to confirm if data is surely transmitted by the program, set the <SIOF> (Bit3 of SBI0SR) to be sensed. The SBI0SR<SIOF> is cleared to "0" when transmitting is complete.

When the <SIOINH> is set to "1", transmitting data stops. SBI0SR<SIOF> turns "0".

When an external clock is used, it is also necessary to clear SBI0CR1<SIOS> to "0" before new data is shifted; otherwise, dummy data is transmitted and operation ends.

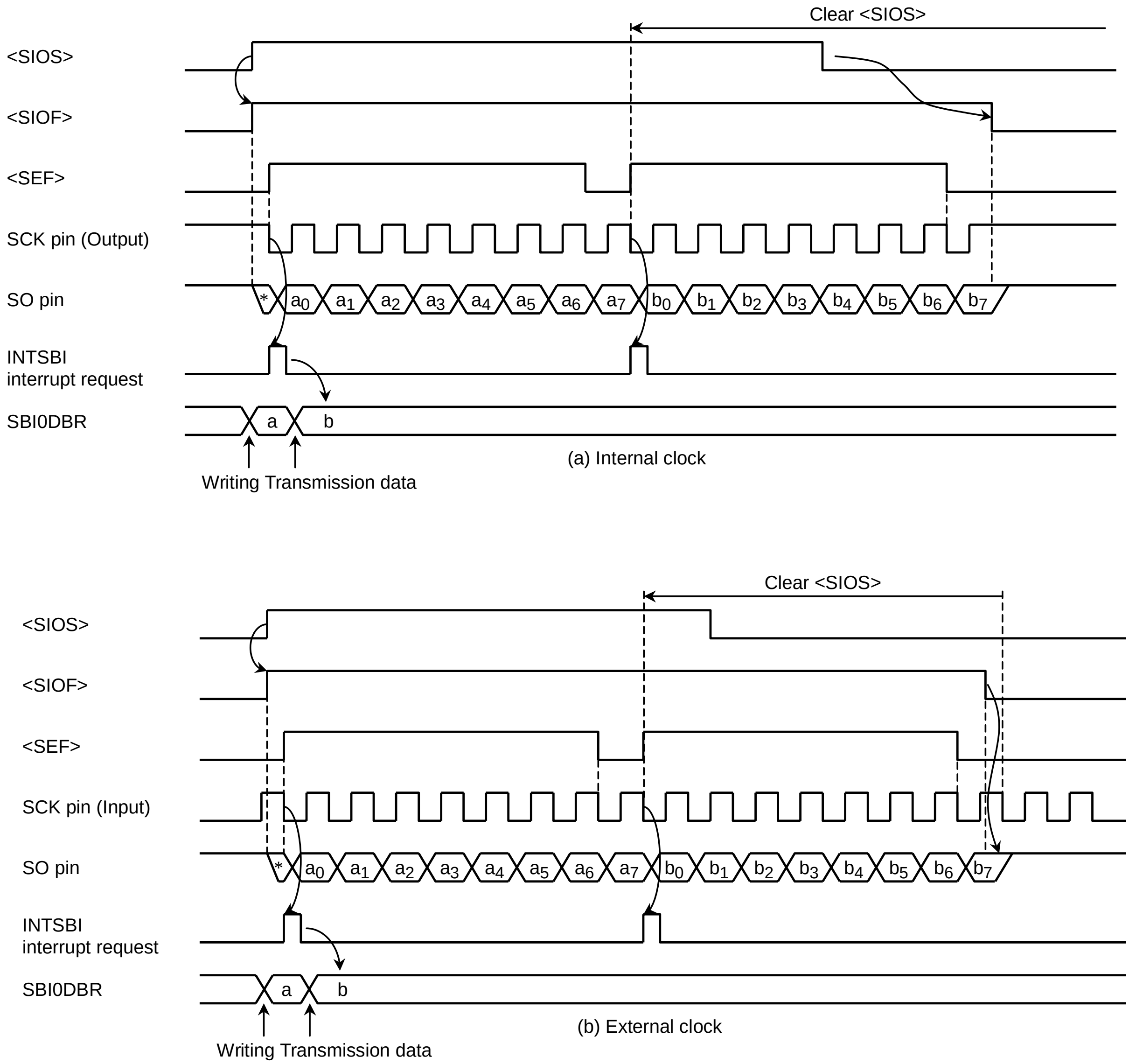


Figure 3.11.26 Transmission Mode

Example: Program to stop data transmission (when an external clock is used)

```
STEST1  :  BIT   SEF, (SBI0SR)              ;   If <SEF> = 1 then loop
           JR    NZ, STEST1
STEST2  :  BIT   0, (P6)                    ;   If SCK = 0 then loop
           JR    Z, STEST2
           LD    (SBI0CR1), 00000111B       ;   <SIOS> ← 0
```

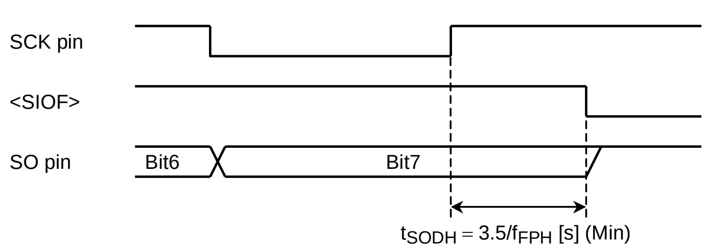


Figure 3.11.27 Transmission Data Hold Time at End Transmit

b. 8-bit receive mode

Set the control register to receive mode and set SBI0CR1<SIOS> to "1" for switching to receive mode. Data is received into the shift register via the SI pin and synchronized with the serial clock, starting from the least significant bit (LSB). When 8-bit data is received, the data is transferred from the shift register to SBI0DBR. An INTSBI (Buffer full) interrupt request is generated to request that the received data be read. The data is then read from SBI0DBR by the interrupt service program.

When an internal clock is used, the serial clock will stop and the automatic wait function will be in effect until the received data has been read from SBI0DBR.

When an external clock is used, since shift operation is synchronized with an external clock pulse, the received data should be read from SBI0DBR before the next serial clock pulse is input. If the received data is not read, any further data which is to be received is canceled. The maximum transfer speed when an external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when the received data is read.

Receiving of data ends when <SIOS> is cleared to "0" by the buffer full interrupt service program or when <SIOINH> is set to "1". If <SIOS> is cleared to "0", received data is transferred to SBI0DBR in complete blocks. The received mode ends when the transfer is complete. In order to confirm whether data is being received properly by the program, set SBI0SR<SIOF> to be sensed. <SIOF> is cleared to "0" when receiving has been completed. When it is confirmed that receiving has been completed, the last data is read. When <SIOINH> is set to "1", data receiving stops. <SIOF> is cleared to "0" (The received data becomes invalid, therefore no need to read it).

Note: When the transfer mode is changed, the contents of SBI0DBR will be lost. If the mode must be changed, conclude data receiving by clearing <SIOS> to "0", read the last data, then change the mode.

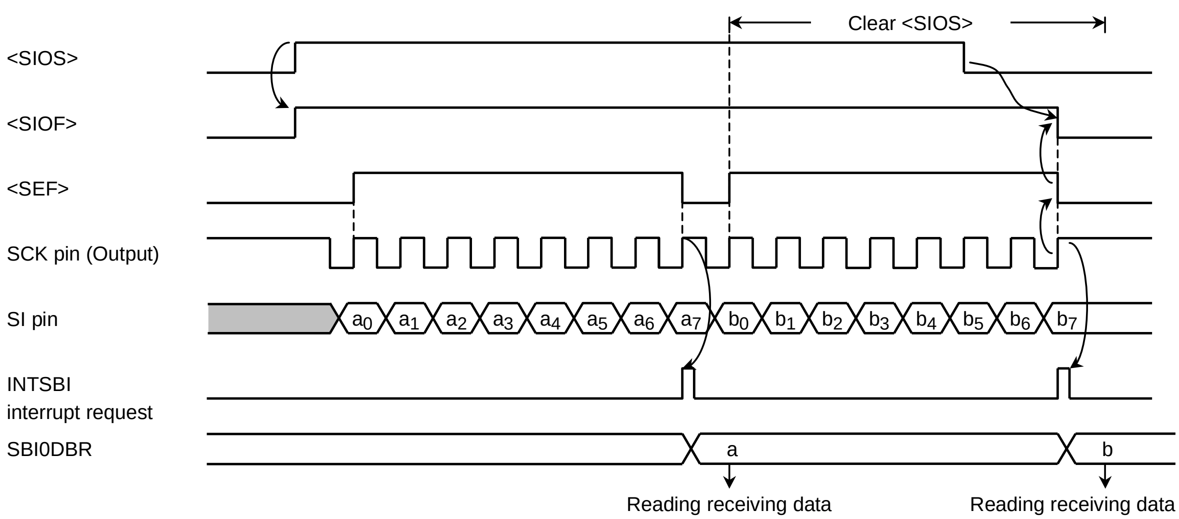


Figure 3.11.28 Receiving Mode (when an internal clock is used)

c. 8-bit transmit/receive mode

Set a control register to a transmit/receive mode and write data to SBI0DBR. After the data has been written, set SBI0CR1<SIOS> to “1” to start transmitting/receiving. When data is transmitted, the data is output via the SO pin, starting from the least significant bit (LSB) and synchronized with the leading edge of the serial clock signal. When data is received, the data is input via the SI pin on the trailing edge of the serial clock signal. 8-bit data is transferred from the shift register to SBI0DBR and an INTSBI interrupt request is generated. The interrupt service program reads the received data from the data buffer register and writes the data that is to be transmitted. SBI0DBR is used for both transmitting and receiving. Transmitted data should always be written after received data has been read.

When an internal clock is used, the automatic wait function will be in effect until the received data has been read and the next data has been written.
When an external clock is used, since the shift operation is synchronized with the external clock, received data is read and transmitted data is written before a new shift operation is executed. The maximum transfer speed when an external clock is used is determined by the delay time between the time when an interrupt request is generated and the time at which received data is read and transmitted data is written.

When transmission is started, after the SBI0SR<SIOF> goes “1” output from the SO pin holds final bit of the last data until falling edge of the SCK.

Transmitting/receiving data ends when <SIOS> is cleared to “0” by the INTSBI interrupt service program or when SBI0CR1<SIOINH> is set to “1”. When <SIOS> is cleared to “0”, received data is transferred to SBI0DBR in complete blocks. The transmit/receive mode ends when the transfer is complete. In order to confirm whether data is being transmitted/received properly by the program, set SBI0SR<SIOF> to be sensed. <SIOF> is set to “0” when transmitting/receiving has been completed. When <SIOINH> is set to 1, data transmitting/receiving stops. SBI0SR<SIOF> is then cleared to 0.

Note: When the transfer mode is changed, the contents of SBI0DBR will be lost. If the mode must be changed, conclude data transmitting/receiving by clearing <SIOS> to “0”, read the last data, then change the transfer mode.

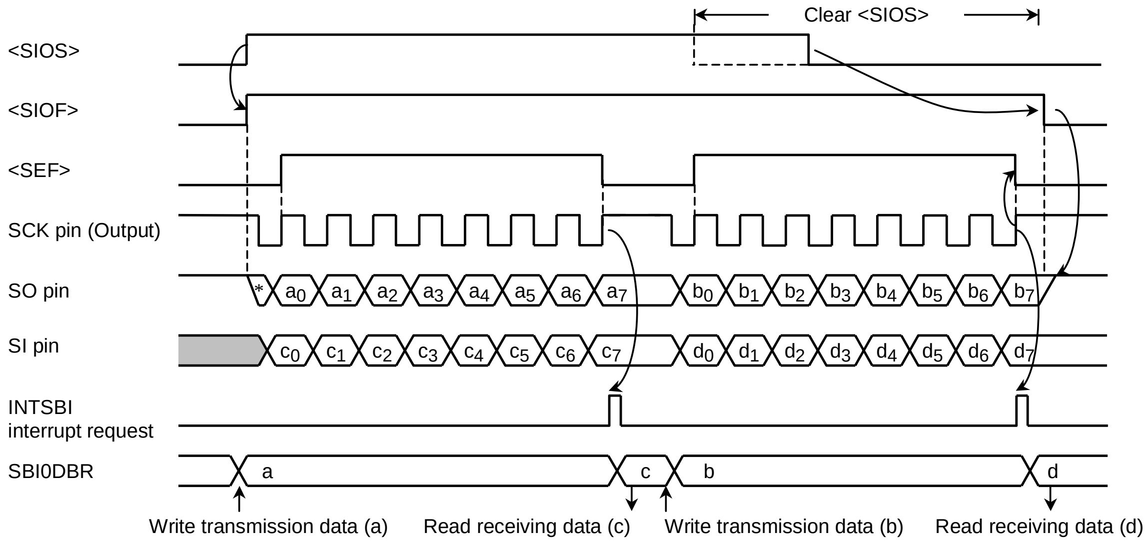


Figure 3.11.29 Transmission/Receiving Mode (when an internal clock is used)

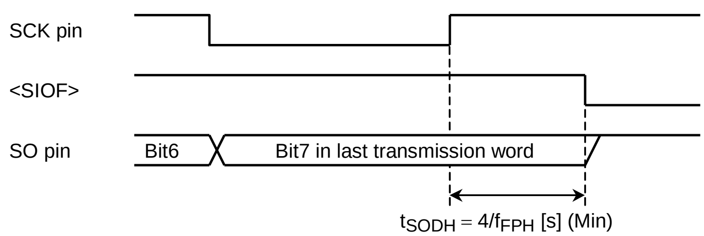


Figure 3.11.30 Transmission Data Hold Time at End of Transmission/Receiving

(Transmission/receiving mode)

3.12 Analog/Digital Converter

The TMP91CP27 incorporates a 10-bit successive approximation-type analog/digital converter (AD converter) with 4-channel analog input.

Figure 3.12.1 is a block diagram of the AD converter. The 4-channel analog input pins (AN0 to AN3) are shared with the input-only port 5 and can thus be used as an input port.

Note: When IDLE2, IDLE1 or STOP mode is selected, so as to reduce the power, with some timing the system may enter a stand-by mode even though the internal comparator is still enabled. Therefore be sure to check that AD converter operations are halted before a HALT instruction is executed.

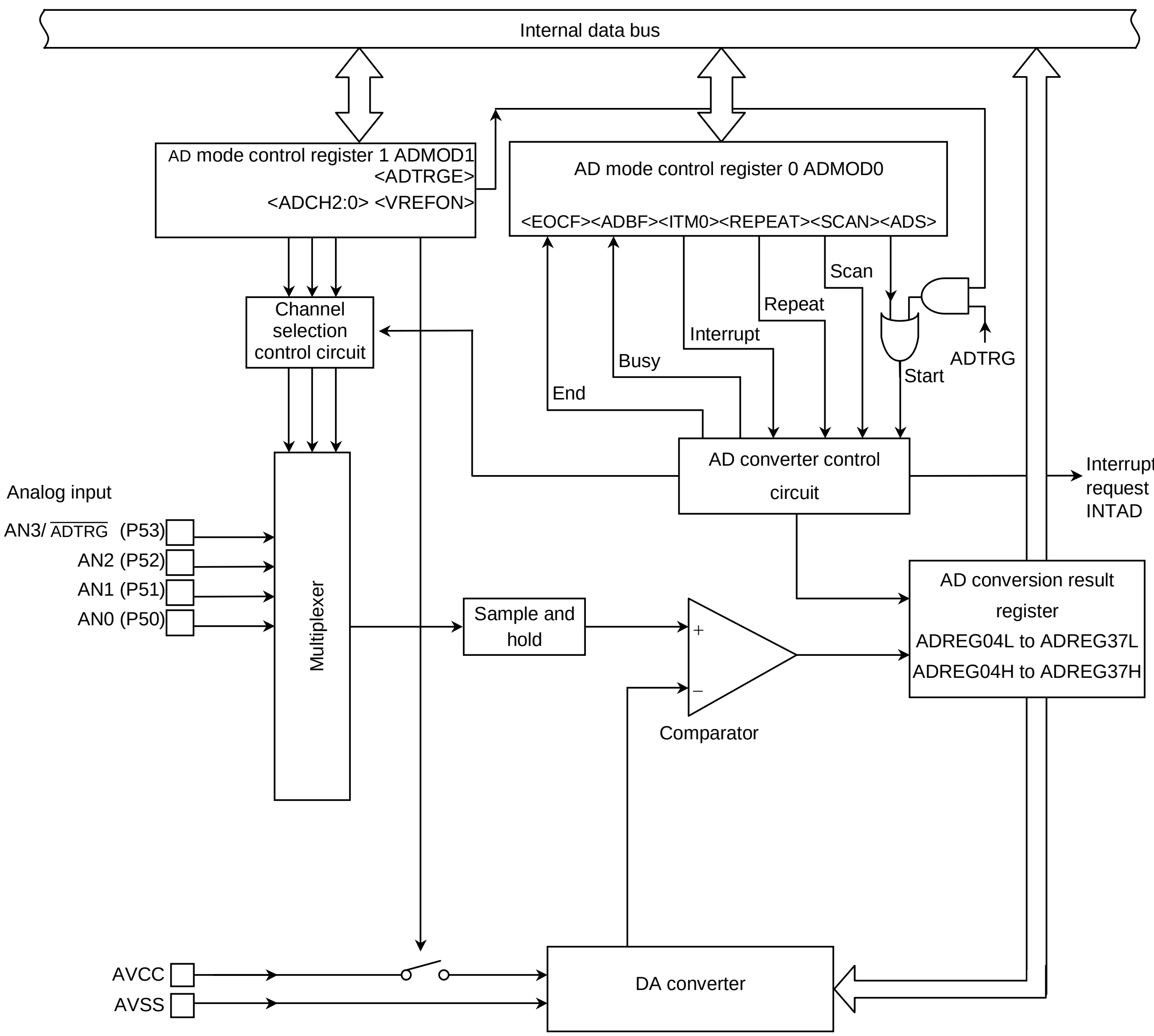


Figure 3.12.1 Block Diagram of AD Converter

3.12.1 Control Register

The AD converter is controlled by the two AD mode control registers: ADMOD0 and ADMOD1. The AD conversion results are stored in 8 kinds of AD conversion data upper and lower registers: ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L.

Figure 3.12.2 to Figure 3.12.5 shows the registers related to the AD converter.

AD Mode Control Register 0

ADMOD0 (02B0H)

Read-modify-write instruction is prohibited.

		7	6	5	4	3	2	1	0
ADMOD0 (02B0H)	Bit symbol	EOCF	ADBF	–	–	ITM0	REPEAT	SCAN	ADS
	Read/Write	R		R/W					
	After reset	0	0	0	0	0	0	0	0
	Function	AD conversion end flag 0: Conversion in progress 1: Conversion complete	AD conversion busy flag 0: Conversion stopped 1: Conversion in progress	Note: Always write 0.	Note: Always write 0.	Interrupt specification in conversion channel fixed repeat mode 0: Every conversion 1: Every fourth conversion	Repeat mode specification 0: Single conversion 1: Repeat conversion mode	Scan mode specification 0: Conversion channel fixed mode 1: Conversion channel scan mode	AD conversion start 0: Don't care 1: Start conversion Always 0 When read.

AD conversion start

0	Don't care
1	Start AD conversion

Note: Always read as 0.

AD scan mode setting

0	AD conversion channel fixed mode
1	AD conversion channel scan mode

AD repeat mode setting

0	AD single conversion mode
1	AD repeat conversion mode

Specify AD conversion interrupt for cannel fixed repeat conversion mode

	Channel fixed repeat conversion mode <SCAN> = "0", <REPEAT> = "1"
0	Generates interrupt every conversion.
1	Generates interrupt every fourth conversion.

AD conversion busy flag

0	AD conversion stopped
1	AD conversion in progress

AD conversion end flag

0	Before or during AD conversion
1	AD conversion complete

Figure 3.12.2 Register for AD Converter

AD Mode Control Register 1

ADMOD1 (02B1H)		7	6	5	4	3	2	1	0
	Bit symbol	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
	Read/Write	R/W	R/W			R/W			
	After reset	0	0			0	0	0	0
	Function	VREF application control 0: OFF 1: ON	IDLE2 0: Stop 1: Operate			AD external trigger start control 0: Disable 1: Enable	Analog input channel selection		

Analog input channel selection

<ADCH2:0> \ <SCAN>	0 (Channel fixed)	1 (Channel scanned)
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011 (Note)	AN3	AN0 → AN1 → AN2 → AN3
100	Don't select.	
101		
110		
111		

AD conversion start control by external trigger ($\overline{\text{ADTRG}}$ input)

0	Disabled
1	Enabled

IDLE2 control

0	Stopped
1	In operation

Control of application of reference voltage to AD converter

0	OFF
1	ON

Before starting conversion (Before writing "1" to ADMOD0<ADS>), set the <VREFON> bit to "1".

Note: As pin AN3 also functions as the $\overline{\text{ADTRG}}$ input pin, do not set <ADCH2:0> = "011" when using $\overline{\text{ADTRG}}$ with <ADTRGE> = "1".

Figure 3.12.3 Register for AD Converter

AD Conversion Data Lower Register 0/4

ADREG04L (02A0H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR01	ADR00						ADR0RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD conversion data storage flag 1: Conversion result stored

AD Conversion Data Upper Register 0/4

ADREG04H (02A1H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
After reset	Undefined							
Function	Stores upper 8 bits AD conversion result.							

AD Conversion Data Lower Register 1/5

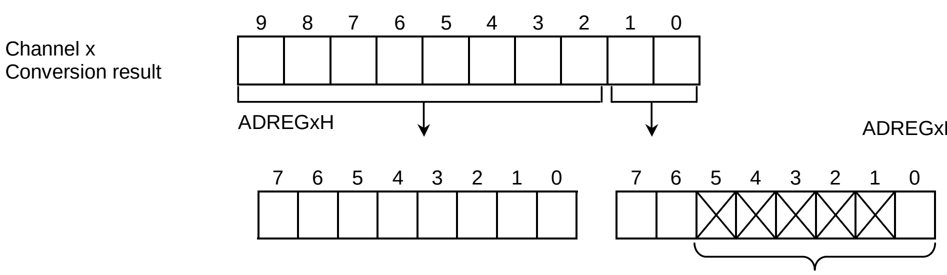
ADREG15L (02A2H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10						ADR1RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD conversion result flag 1: Conversion result stored

AD Conversion Data Upper Register 1/5

ADREG15H (02A3H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After reset	Undefined							
Function	Stores upper 8 bits of AD conversion result.							



- Bits 5 to 1 are always read as “1”.
- Bit0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to “1”. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to “0”.

Figure 3.12.4 Register for AD Converter

AD Conversion Result Lower Register 2/6

		7	6	5	4	3	2	1	0
ADREG26L (02A4H)	Bit symbol	ADR21	ADR20						ADR2RF
	Read/Write	R							R
	After reset	Undefined							0
	Function	Stores lower 2 bits of AD conversion result.							AD conversion data storage flag 1: Conversion result stored

AD Conversion Data Upper Register 2/6

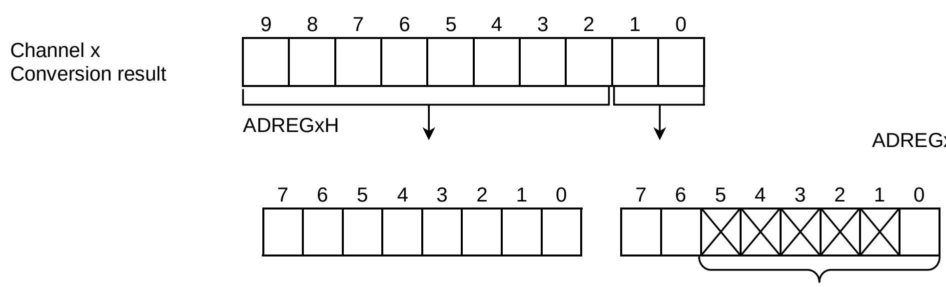
		7	6	5	4	3	2	1	0
ADREG26H (02A5H)	Bit symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R							
	After reset	Undefined							
	Function	Stores upper 8 bits of AD conversion result.							

AD Conversion Data Lower Register 3/7

		7	6	5	4	3	2	1	0
ADREG37L (02A6H)	Bit symbol	ADR31	ADR30						ADR3RF
	Read/Write	R							R
	After reset	Undefined							0
	Function	Stores lower 2 bits of Ad conversion result.							AD cnversion data storage flag 1: Conversion result stored

AD Conversion Result Upper Register 3/7

		7	6	5	4	3	2	1	0
ADREG37H (02A7H)	Bit symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
	Read/Write	R							
	After reset	Undefined							
	Function	Stores upper 8 bits of AD conversion result.							



- Bits 5 to 1 are always read as “1”.
- Bit0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to “1”. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to “0”.

Figure 3.12.5 Register for AD Converter

3.12.2 Operation

(1) Analog reference voltage

A high-level analog reference voltage is applied to the AVCC pin; a low-level analog reference voltage is applied to the AVSS pin. To perform AD conversion, the reference voltage as the difference between AVCC and AVSS, is divided by 1024 using string resistance. The result of the division is then compared with the analog input voltage.

To turn off the switch between AVCC and AVSS, write “0” to ADMOD1<VREFON> in AD mode control register 1. To start AD conversion in the OFF state, first write “1” to ADMOD1<VREFON>, wait 3 μs until the internal reference voltage stabilizes (this is not related to fc), then set ADMOD0< ADS> to “1”.

(2) Analog input channel selection

The analog input channel selection varies depends on the operation mode of the AD converter.

- In analog input channel fixed mode (ADMOD0<SCAN> = “0”)
  Setting ADMOD1<ADCH2:0> selects one of the analog input pins AN0 to AN3 as the input channel.
- In analog input channel scan mode (ADMOD0<SCAN> = “1”)
  Setting ADMOD1<ADCH2:0> selects one of the 4 scan modes.

Table 3.12.1 Illustrates analog input channel selection in each operation mode.

After Reset, ADMOD0<SCAN> = “0” and ADMOD1<ADCH2:0> = “000”. Thus pin AN0 is selected as the fixed input channel. Pins not used as analog input channels can be used as standard input port pins.

Table 3.12.1 Analog Input Channel Selection

<ADCH2:0>	Channel Fixed <SCAN> = “0”	Channel Scan <SCAN> = “1”
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011	AN3	AN0 → AN1 → AN2 → AN3
100	Don’t select.	
101		
110		
111		

(3) Starting AD conversion

To start AD conversion, write "1" to ADMOD0<ADS> in AD mode control register 0, or ADMOD1<ADTRGE> in AD mode control register 1 and input falling edge on $\overline{\text{ADTRG}}$ pin. When AD conversion starts, the AD conversion busy flag ADMOD0<ADBF> will be set to "1", indicating that AD conversion is in progress.

Writing "1" to ADMOD0<ADS> during AD conversion restarts conversion. At that time, to determine whether the AD conversion results have been preserved, check the value of the conversion data storage flag ADREGxL<ADRxRF>.

During AD conversion, a falling edge input on the $\overline{\text{ADTRG}}$ pin will be ignored.

(4) AD conversion modes and the AD conversion end interrupt

The 4 AD conversion modes are:

- Channel fixed single conversion mode
- Channel scan single conversion mode
- Channel fixed repeat conversion mode
- Channel scan repeat conversion mode

The ADMOD0<REPEAT> and ADMOD0<SCAN> settings in AD mode control register 0 determine the AD mode setting.

Completion of AD conversion triggers an AD conversion end interrupt request INTAD. Also, ADMOD0<EOCF> will be set to "1" to indicate that AD conversion has been completed.

a. Channel fixed single conversion mode

Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to "00" selects channel fixed single conversion mode.

In this mode, data on one specified channel is converted once only. When the conversion has been completed, the ADMOD0<EOCF> flag is set to "1", ADMOD0<ADBF> is cleared to "0", and an INTAD interrupt request is generated.

b. Channel scan single conversion mode

Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to "01" selects channel scan single conversion mode.

In this mode, data on the specified scan channels is converted once only. When scan conversion has been completed, ADMOD0<EOCF> is set to "1", ADMOD0<ADBF> is cleared to "0", and an INTAD interrupt request is generated.

c. Channel fixed repeat conversion mode

Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to “10” selects channel fixed repeat conversion mode.

In this mode, data on one specified channel is converted repeatedly. When conversion has been completed, ADMOD0<EOCF> is set to “1” and ADMOD0<ADBF> is not cleared to “0” but held “1”. INTAD interrupt request generation timing is determined by the setting of ADMOD0<ITM0>.

Clearing <ITM0> to “0” generates an interrupt request every time an AD conversion is completed.

Setting <ITM0> to “1” generates an interrupt request on completion of every fourth conversion.

d. Channel scaen repeat conversion mode

Setting ADMOD0<REPEAT> and ADMOD0<SCAN> to “11” selects channel scan repeat conversion mode.

In this mode, data on the specified scan channels is converted repeatedly. When each scan conversion has been completed, ADMOD0<EOCF> is set to “1” and an INTAD interrupt request is generated. ADMOD0<ADBF> is not cleared to “0” but held “1”.

To stop conversion in a repeat conversion mode (e.g., in cases c. and d.), program a “0” to ADMOD0<REPEAT>. After the current conversion has been completed, the repeat conversion mode terminates and ADMOD0<ADBF> is cleared to “0”.

Switching to a halt state (IDLE2 mode with ADMOD1<I2AD> cleared to “0”, IDLE1 mode or STOP mode) immediately stops operation of the AD converter even when AD conversion is still in progress. In repeat conversion modes (e.g., in cases c. and d.), when the halt is released, conversion restarts from the beginning. In single conversion modes (e.g., in cases a. and b.), conversion does not restart when the halt is released (The converter remains stopped).

Table 3.12.2 shows the relationship between the AD conversion modes and interrupt requests.

Table 3.12.2 Relationship between the AD Conversion Modes and Interrupt Requests AD

Mode	Generation of Interrupt Request	ADMOD0		
		<ITM0>	<REPEAT>	<SCAN>
Channel fixed single conversion mode	After completion of conversion	X	0	0
Channel scan single conversion mode	After completion of scan conversion	X	0	1
Channel fixed repeat conversion mode	Every conversion	0	1	0
	Every forth conversion	1		
Cannel scan repeat conversion mode	After completion of every scan conversion	X	1	1

X: Don’t care

(5) AD conversion time

84 states (10.5 μs @ f_{FPH} = 16MHz) are required for the AD conversion for one channel.

(6) Storing and reading the results of AD conversion

The AD conversion data upper and lower registers (ADREG04H/L to ADREG37H/L) store the AD conversion results. (ADREG04H/L to ADREG37H/L are read-only registers.)

In channel fixed repeat conversion mode with ADMOD0<ITM0> = “1”, the conversion results are stored successively in registers ADREG04H/L to ADREG37H/L. In other modes, the AN0, AN1, AN2 and AN3 conversion results are stored in ADREG04H/L, ADREG15H/L, ADREG26H/L and ADREG37H/L respectively.

Table 3.12.3 shows the correspondence between the analog input channels and the registers that are used to hold the results of AD conversion.

Table 3.12.3 Correspondence between Analog Input Channel and AD Conversion Result Register

Analog Input Channel (Port 5)	AD Conversion Result Register	
	Conversion Mode Other Than Right	Channel Fixed Repeat Conversion Mode (Every forth conversion)
AN0	ADREG04H/L	ADREG04H/L
AN1	ADREG15H/L	ADREG15H/L
AN2	ADREG26H/L	ADREG26H/L
AN3	ADREG37H/L	ADREG37H/L

<ADRxRF> “bit0” of the AD conversion data lower register is used as the AD conversion data storage flag. The storage flag indicates whether the AD conversion result register has been read or not. When a conversion result is stored in the AD conversion result register, the flag is set to “1”. When either of the AD conversion result registers (ADREGxH or ADREGxL) is read, the flag is cleared to “0”.

Reading the AD conversion result also clears the AD conversion end flag ADMOD0<EOCF> to “0”.

Example:

a. Convert the analog input voltage on the AN3 pin and write the result, to memory address 1800H using the AD interrupt (INTAD) processing routine.

Setting of main routine

		7	6	5	4	3	2	1	0	
INTE0AD	←	X	1	0	0	–	–	–	–	Enable INTAD and set it to interrupt level 4.
ADMOD1	←	1	1	X	X	0	0	1	1	Set pin AN3 to the analog input channel.
ADMOD0	←	X	X	0	0	0	0	0	1	Start conversion in channel fixed single conversion mode.

Interrupt routine processing example

WA	← ADREG37	Read value of ADREG37L, ADREG37H to general purpose register WA (16-bit).
WA	> > 6	Shift contents read into WA six times to right and zero-fill upper bits.
(1800H)	← WA	Write contents of WA to memory address 1800H.

b. Converts repeatedly the analog input voltages on the three pins AN0, AN1 and AN2, using channel scan repeat conversion mode.

INTE0AD	←	X	0	0	0	–	–	–	–	Disable INTAD.
ADMOD1	←	1	1	X	X	0	0	1	0	Set pins AN0 to AN2 to be the analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Start conversion in channel scan repeat conversion mode.

X: Don’t care, –: No change

3.13 Watchdog Timer (Runaway detection timer)

The TMP91CP27 features a watchdog timer for detecting runaway.

The watchdog timer (WDT) is used to return the CPU to normal state when it detects that the CPU has started to malfunction (Runaway) due to causes such as noise.

When the watchdog timer detects a malfunction, it generates a non-maskable interrupt INTWD to notify the CPU. Connecting the watchdog timer out to the reset pin internally forces a reset.

3.13.1 Configuration

Figure 3.13.1 is a block diagram of watchdog timer.

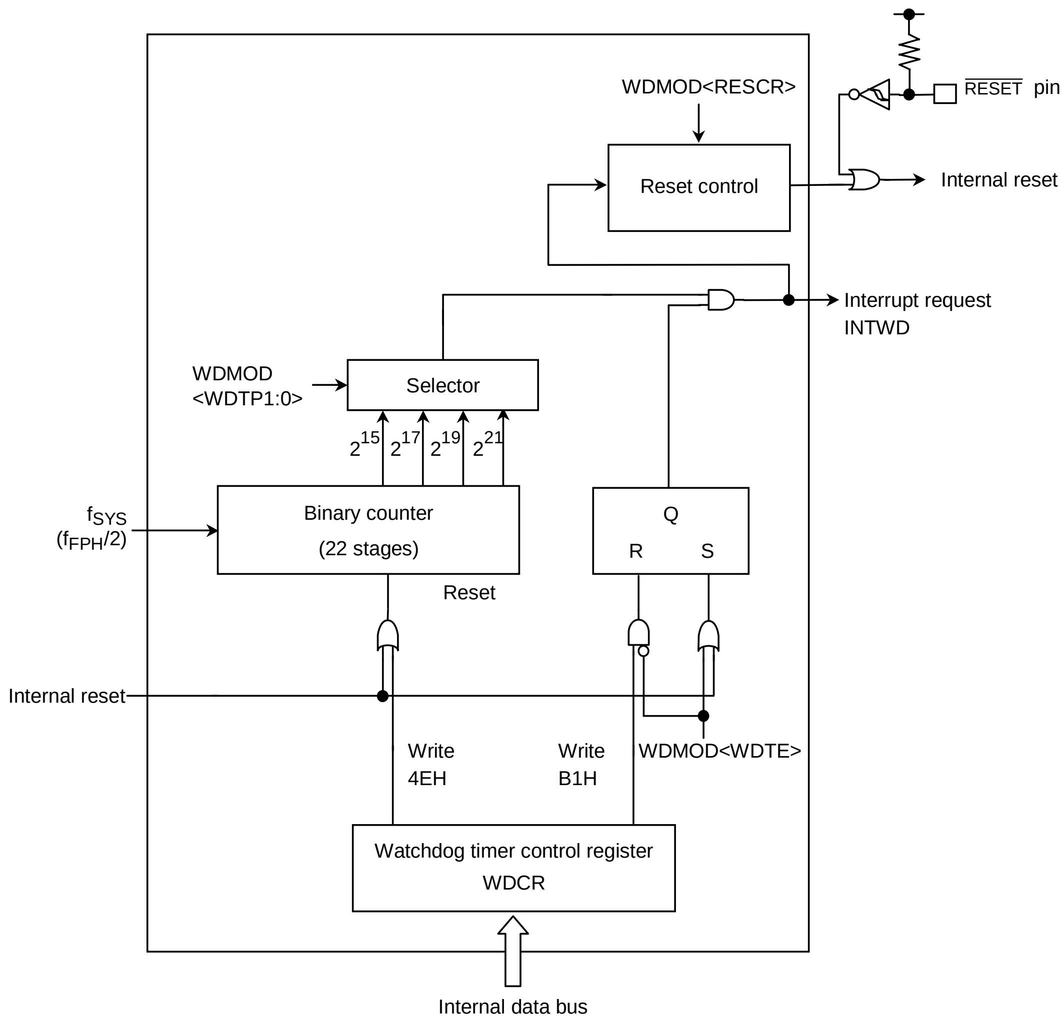


Figure 3.13.1 Block Diagram of Watchdog Timer

Note: It needs to care designing the machine set, because watchdog timer can't operate completely by external noise.

The watchdog timer consists of a 22-stage binary counter which uses the system clock (f_{SYS}) as the input clock. The binary counter can output $f_{SYS}/2^{15}$, $f_{SYS}/2^{17}$, $f_{SYS}/2^{19}$ and $f_{SYS}/2^{21}$. Selecting one of the outputs using WDMOD<WDTP1:0> generates a watchdog interrupt and outputs watchdog timer out when an overflow occurs as shown in Figure 3.12.2.

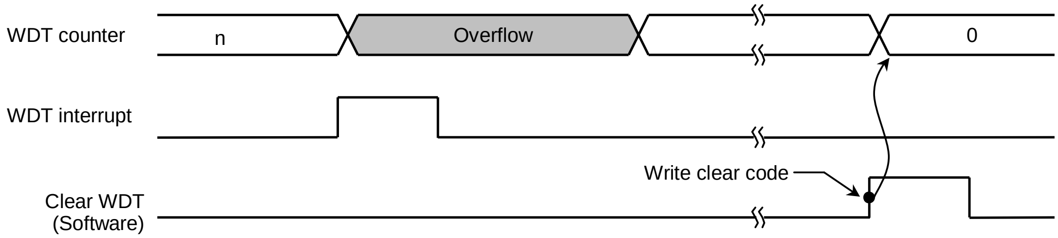


Figure 3.13.2 Normal Mode

The runaway detection result can also be connected to RESET pin internally.

In this case, the reset time will be between 22 and 29 states (44 to 58 μs at f_{OSCH} = 16 MHz, f_{FPH} = 1 MHz) as shown Figure 3.13.3. Also, system clock f_{SYS} (1 cycle = 1 state) which generated clock by dividing it into 2, that clock f_{FPH} divide clock f_{OSCH} high-frequency oscillator into 16 is used to resetting.

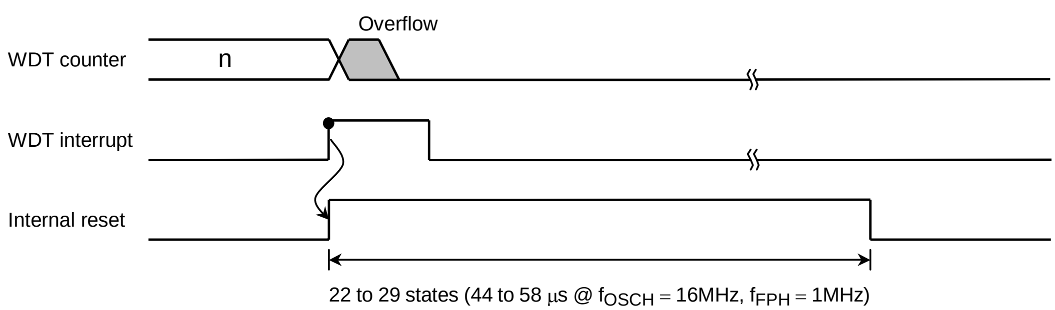


Figure 3.13.3 Reset Mode

3.13.2 Control Register

The watchdog timer WDT is controlled by two controls registers WDMOD and WDCR.

(1) Watchdog timer mode register (WDMOD)

a. Setting the detection time for the watchdog timer in <WDTP1:0>

This 2-bit register is used for setting the watchdog timer interrupt time used when detecting runaway. During resetting, this register is initialized to WDMOD<WDTP1:0> = "00".

The detection times for WDT are shown in Figure 3.13.4.

b. Watchdog timer enable/disable control register <WDTE>

During resetting, WDMOD<WDTE> is initialized to "1", enabling the watchdog timer.

To disable the watchdog timer, it is necessary to set this bit to "0" and to write the disable code (B1H) to the watchdog timer control register WDCR. This makes it difficult for the watchdog timer to be disabled by runaway.

However, it is possible to return the watchdog timer from the disabled state to the enabled state merely by setting <WDTE> to "1".

c. Watchdog timer out reset connection <RESCR>

This register is used to connect the output of the watchdog timer with the RESET terminal internally. Since WDMOD<RESCR>is initialized to "0" on Reset, a Reset by the watchdog timer will not be performed.

(2) Watchdog timer control register (WDCR)

This register is used to disable and clear the binary counter for the watchdog timer.

- Disable control

Disable control the watchdog timer can be disabled by clearing WDMOD<WDTE> to "0" and then writing the disable code (B1H) to the WDCR register.

```
WDMOD  ← 0 – – 0 0 – – 0    Clear WDMOD<WDTE> to "0".
WDCR   ← 1 0 1 1 0 0 0 1    Write the disable code (B1H).
```

- Enable control

Set WDMOD<WDTE> to "1".

- Watchdog timer clear control

To clear the binary counter and cause counting to resume, write the clear code (4EH) to the WDCR register.

```
WDCR   ← 0 1 0 0 1 1 1 0    Write the clear code (4EH).
```

WDMOD (0300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0	–	–	I2WDT	RESCR	–
Read/Write	R/W	R/W		R/W	R/W	R/W		R/W
After reset	1	0	0	0	0	0	0	0
Function	WDT control 0: Stop 1: Enable	WDT detection time selection 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$		Write “0”.	Write “0”.	IDLE2 0: Stop 1: Operate	1:Internally connects WDT out to the reset pin	Write “0”.

Watchdog timer out control

0	–
1	Connect WDT out to reset

IDLE2 control

0	Stop
1	Operate

Watchdog timer detection time

@ fc = 16 MHz, fs = 32.768 kHz

System Clock Selection SYSCR1 <SYSCK>	Clock Gear Value SYSCR1 <GEAR2:0>	Watchdog Timer Detection Time WDMOD<WDTP1:0> 00	01	10	11
1 (fs)	XXX	2.0 s	8.0 s	32.0 s	128.0 s
0 (fc)	000 (fc)	4.096 ms	16.384 ms	65.536 ms	262.144 ms
	001 (fc/2)	8.192 ms	32.768 ms	131.072 ms	524.288 ms
	010 (fc/4)	16.384 ms	65.536 ms	262.144 ms	1.049 s
	011 (fc/8)	32.768 ms	131.072 ms	524.288 ms	2.097 s
	100 (fc/16)	65.536 ms	262.144 ms	1.049 s	4.194 s

Watchdog timer enable/disable control

0	Disable
1	Enable

Figure 3.13.4 Watchdog Timer Mode Register

WDCR (0301H)

Read-modify-write instruction is prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	–							
Read/Write	W							
After reset	–							
Function	B1H: WDT disable code 4EH: WDT clear code							

Disable/clear WDT

B1H	Disable code
4EH	Clear code
Others	–

Figure 3.13.5 Watchdog Timer Control Register

3.13.3 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1:0> has elapsed. The binary counter for the watchdog timer must be cleared to "0" by software (Instruction) before an INTWD interrupt will be generated. If the CPU malfunctions (e.g., if runaway occurs) due to causes such as noise and does not execute the instruction used to clear the binary counter, the binary counter will overflow and an INTWD interrupt will be generated. The CPU will detect malfunction (Runaway) due to the INTWD interrupt and in this case it is possible to return the CPU to normal operation by means of an anti-malfunction program.

The watchdog timer works immediately after reset.

The watchdog timer does not operate in IDLE1 or STOP mode. When the device is in IDLE2 mode, the operation of WDT depends on the WDMOD<I2WDT> setting. Ensure that WDMOD<I2WDT> is set before the device enters IDLE2 mode.

Example:

a. Clear binary counter.

WDCR ← 0 1 0 0 1 1 1 0 Write the clear code (4EH).

b. Set watchdog timer detection time to $2^{17}/f_{SYS}$.

WDMOD ← 1 0 1 0 0 – – 0

c. Disable watchdog timer.

WDMOD ← 0 – – 0 0 – – 0 Clear <WDTE> to "0".
WDCR ← 1 0 1 1 0 0 0 1 Write the disable code (B1H).

3.14 Timer for Real Time Clock (RTC)

The TMP91CP27 includes a timer that is used for a clock operation.

An interrupt (INTRTC) can be generated each 0.0625 [s] or 0.125 [s] or 0.25 [s] or 0.50 [s] by using a low frequency clock of 32.768 kHz. A clock function can be easily used.

A timer for real time clock can operate in all modes in which a low-frequency oscillation is operated.

In addition, INTRTC can return from each standby mode except STOP mode.

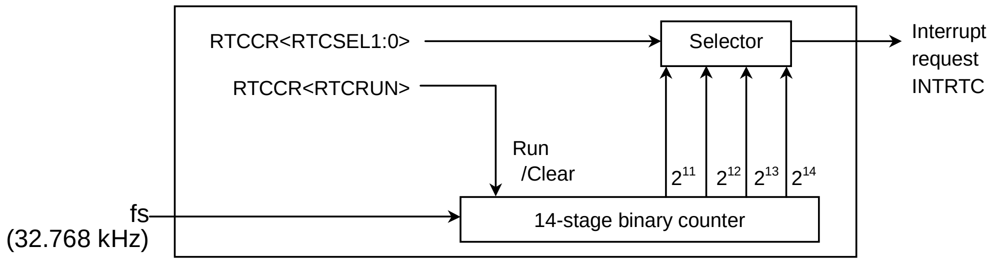


Figure 3.14.1 Block Diagram for Real Time Clock

The timer for real time clock is controlled by the real time clock control register (RTCCR) as shown in Figure 3.14.2.

RTCCR (0310H)

	7	6	5	4	3	2	1	0
Bit symbol	–					RTCSEL1	RTCSEL0	RTCRUN
Read/Write	R/W					R/W		R/W
After reset	0					0	0	0
Function	Write “0”.					00: 2^{14}/fs 01: 2^{13}/fs 10: 2^{12}/fs 11: 2^{11}/fs		0: Stop & clear 1: Count

Counting operation

0	Stop & clear
1	Count

Interrupt generation cycle (fs = 32.768 kHz)

00	0.50 s
01	0.25 s
10	0.125 s
11	0.0625 s

Figure 3.14.2 Real Time Clock Control Register

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Power supply voltage	Vcc	−0.5 to 4.0	V
Input voltage	VIN	−0.5 to Vcc + 0.5	
Output current (1 pin)	IOL	2	mA
Output current (1 pin)	IOH	−2	
Output current (Total)	ΣIOL	80	
Output current (Total)	ΣIOH	−80	
Power dissipation (Ta = 85°C)	PD	600	mW
Soldering temperature (10 s)	TSOLDER	260	°C
Storage temperature	TSTG	−65 to 150	
Operation temperature	TOPR	−40 to 85	

Note: The absolute maximum ratings are rated values that must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products that include this device, ensure that no absolute maximum rating value will ever be exceeded.

4.2 DC Characteristics (1/2)

Parameter		Symbol	Condition		Min	Typ.(Note)	Max	Unit
Power supply voltage (AVCC = DVCC, AVSS = DVSS = 0 V)		VCC	fc = 4 to 27 MHz	fs = 30 to 34 kHz	2.7		3.6	V
			fc = 2 to 10 MHz		1.8			
Input low voltage	P00 to P17 (AD0 to AD15)	VIL	Vcc ≥ 2.7 V		−0.3		0.6	V
			Vcc < 2.7 V				0.2 Vcc	
	P20 to P97 (Except P63)	VIL1	Vcc ≥ 2.7 V				0.3 Vcc	
			Vcc < 2.7 V				0.2 Vcc	
	$\overline{RESET}$, $\overline{NMI}$ P63 (INT0)	VIL2	Vcc ≥ 2.7 V				0.25 Vcc	
			Vcc < 2.7 V				0.15 Vcc	
	AM0 and AM1	VIL3	Vcc ≥ 2.7 V				0.3	
			Vcc < 2.7 V				0.3	
	X1	VIL4	Vcc ≥ 2.7 V				0.2 Vcc	
			Vcc < 2.7 V				0.1 Vcc	
Input high voltage	P00 to P17 (AD0 to AD15)	VIH	Vcc ≥ 2.7 V		2.0		Vcc + 0.3	V
			Vcc < 2.7 V		0.7 Vcc			
	P20 to P97 (Except P63)	VIH1	Vcc ≥ 2.7 V		0.7 Vcc			
			Vcc < 2.7 V		0.8 Vcc			
	$\overline{RESET}$, $\overline{NMI}$, P63 (INT0)	VIH2	Vcc ≥ 2.7 V		0.75 Vcc			
			Vcc < 2.7 V		0.85 Vcc			
	AM0 and AM1	VIH3	Vcc ≥ 2.7 V		Vcc − 0.3			
			Vcc < 2.7 V		Vcc − 0.3			
	X1	VIH4	Vcc ≥ 2.7 V		0.8 Vcc			
			Vcc < 2.7 V		0.9 Vcc			
Output low voltage		VOL	IOL = 1.6mA	Vcc ≥ 2.7 V			0.45	V
			IOL = 0.4mA	Vcc < 2.7 V			0.15 Vcc	
Output high voltage		VOH	IOH = −400 μA	Vcc ≥ 2.7 V	Vcc − 0.3			
			IOH = −200 μA	Vcc < 2.7 V	0.8 Vcc			

Note: Typical values are for when Ta = 25°C and V_{CC} = 3.0 V uncles otherwise noted.

4.2 DC Characteristics (2/2)

Parameter	Symbol	Condition	Min	Typ. (Note1)	Max	Unit
Input leakage current	ILI	$0.0 \le V_{IN} \le Vcc$		0.02	±5	μA
Output leakage current	ILO	$0.2 \le V_{IN} \le Vcc - 0.2$		0.05	±10	μA
Power down voltage (@STOP, RAM back up)	VSTOP	V IL2 = 0.2 Vcc, V IH2 = 0.8 Vcc	1.8		3.6	V
$\overline{RESET}$ pull-up resistor	RRST	Vcc = 2.7 V to 3.6 V	100		400	kΩ
		Vcc = 2 V ± 10%	200		1000	
Pin capacitance	CIO	fc = 1 MHz			10	PF
Schmitt width $\overline{RESET}$, $\overline{NMI}$, INT0	VTH	Vcc ≥ 2.7 V	0.4	1.0		V
		Vcc < 2.7 V	0.3	0.8		
Programmable pull-up resistor	RKH	Vcc = 2.7 V to 3.6 V	100		400	kΩ
		Vcc = 2 V ± 10%	200		1000	
NORMAL (Note 2), (Note 3)	Icc	Vcc = 2.7 V to 3.6 V fc = 27 MHz		11.5 (10.8)	19.0	mA
IDLE2 (Note 3)				5.5 (4.8)	8.0	
IDLE1 (Note 3)				2.5 (1.8)	4.0	
NORMAL (Note 2), (Note 3)		Vcc = 3 V ± 10% fc = 27 MHz		11.5 (10.8)	16.0	mA
IDLE2 (Note 3)				5.5 (4.8)	7.5	
IDLE1 (Note 3)				2.5 (1.8)	3.5	
NORMAL (Note 2), (Note 3)		Vcc = 2 V ± 10 % fc = 10 MHz (Typ. Vcc = 2.0 V)		3.5 (3.0)	5.0	mA
IDLE2 (Note 3)				2.0 (1.5)	3.0	
IDLE1 (Note 3)				0.9 (0.4)	1.8	
SLOW (Note 2)		Vcc = 2.7 V to 3.6 V fs = 32.768 kHz		14.5	30	μA
IDLE2				7.0	19	
IDLE1				5.0	15	
SLOW (Note 2)		Vcc = 2 V ± 10 % fs = 32.768 kHz (Typ. Vcc = 2.0 V)		10	20	μA
IDLE2				5.0	13	
IDLE1				3.0	10	
STOP		Vcc = 1.8 V to 3.6 V		0.1	10	μA

Note 1: Typical values are for when Ta = 25°C and V_{CC} = 3.0 V unless otherwise noted.

Note 2: Icc measurement conditions (NORMAL, SLOW):
All functions are operational; output pins are open and input pins are fixed.

Note 3: Power supply cuurent from AVCC pin is included in power supply current of VCC pin. Also, AVCC pin share with AD reference power supply in TMP91CP27. Therefore, it is included in power supply current of VCC pin that not only power supply current from AVCC pin but also current to ladder resitster. Insert of ( ) is current value when VREF is Off.

4.3 AC Characteristics

(1) Vcc = 2.7 V to 3.6 V

No.	Parameter	Symbol	Valiable		f_{FPH} = 27 MHz		Unit
			Min	Max	Min	Max	
1	f_{FPH} period (= x)	t_{FPH}	37.0	31250	37.0		ns
2	A0 to A15 valid → ALE falling	t_{AL}	0.5x – 6		12		ns
3	ALE falling → A0 to A15 hold	t_{LA}	0.5x – 16		2		ns
4	ALE high pulse width	t_{LL}	x – 20		17		ns
5	ALE falling → $\overline{RD}$ / $\overline{WR}$ falling	t_{LC}	0.5x – 14		4		ns
6	$\overline{RD}$ rising → ALE rising	t_{CLR}	0.5x – 10		8		ns
7	$\overline{WR}$ rising → ALE rising	t_{CLW}	x – 10		27		ns
8	A0 to A15 vlalid → $\overline{RD}$ / $\overline{WR}$ falling	t_{ACL}	x – 23		14		ns
9	A0 to A21 valid → $\overline{RD}$ / $\overline{WR}$ falling	t_{ACH}	1.5x – 26		29		ns
10	$\overline{RD}$ rising → A0 to A21 hold	t_{CAR}	0.5x – 13		5		ns
11	$\overline{WR}$ rising → A0 to A21 hold	t_{CAW}	x – 13		24		ns
12	A0 to A15 valid → D0 to D15 input	t_{ADL}		3.0x – 38		73	ns
13	A0 to A21 valid → D0 to D15 input	t_{ADH}		3.5x – 41		88	ns
14	$\overline{RD}$ falling → D0 to D15 input	t_{RD}		2.0x – 30		44	ns
15	$\overline{RD}$ low puse width	t_{RR}	2.0x – 15		59		ns
16	$\overline{RD}$ rising → D0 to D15 hold	t_{HR}	0		0		ns
17	$\overline{RD}$ rising → A0 to A15 output	t_{RAE}	x – 15		22		ns
18	$\overline{WR}$ low pulse width	t_{WW}	1.5x – 15		40		ns
19	D0 to D15 valid → $\overline{WR}$ rising	t_{DW}	1.5x – 35		20		ns
20	$\overline{WR}$ rising → D0 to D15 hold	t_{WD}	x – 25		12		ns
21	A0 to A21 valid → Port input	t_{APH}		3.5x – 89		40	ns
22	A0 to A21 valid → Port hold	t_{APH2}	3.5x		129		ns
23	A0 to A21 valid → Port valid	t_{AP}		3.5x + 80		209	ns

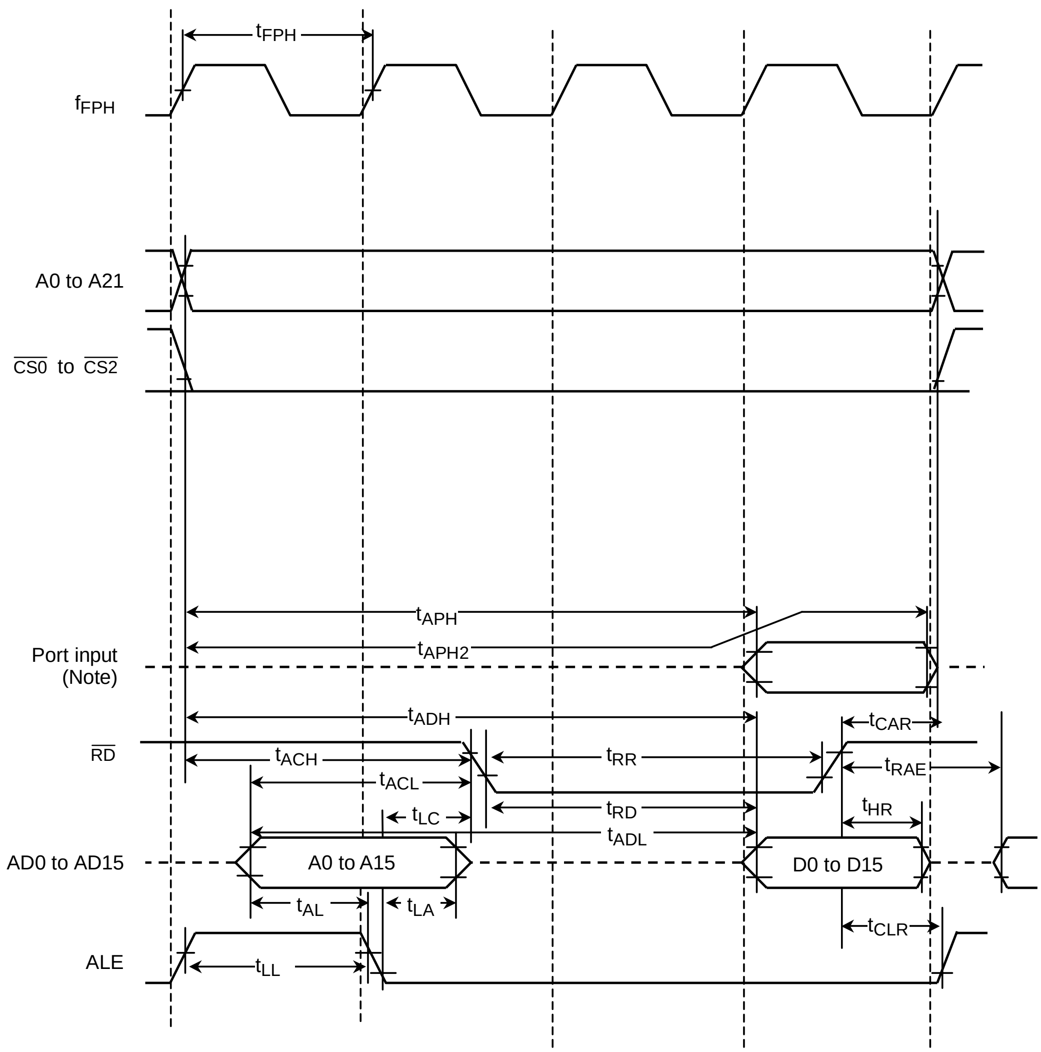
AC measurement conditions

- Output level: High 0.7 × Vcc/Low 0.3 × V_{CC}, C_L = 50 pF
- Input level: High 0.9 × Vcc/Low 0.1 × V_{CC}

Note: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

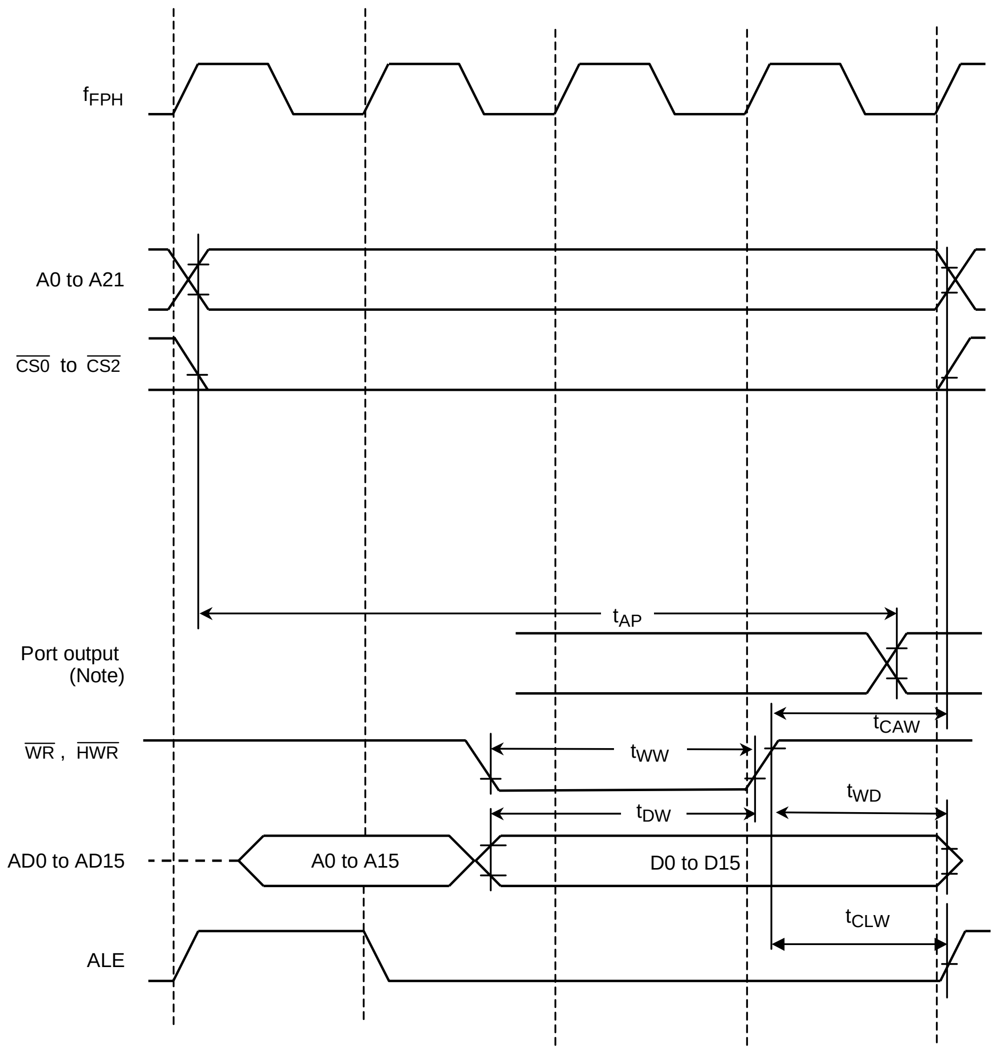
The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.

(2) Read cycle



Note: Since the CPU accesses the internal area to read data from a port, the control signals of external pins such as $\overline{RD}$ and $\overline{CS}$ are not enabled. Therefore, the above waveform diagram should be regarded as depicting internal operation. Please also note that the timing and AC characteristics of port input/output shown above are typical representation. For details, contact your local Toshiba sales representative.

(3) Write cycle



Note: Since the CPU accesses the internal area to write data to a port, the control signals of external pins such as $\overline{WR}$ and $\overline{CS}$ are not enabled. Therefore, the above waveform diagram should be regarded as depicting internal operation. Please also note that the timing and AC characteristics of port input/output shown above are typical representation. For details, contact your local Toshiba sales representative.

4.4 AD Conversion Characteristics

AVCC = VCC, AVSS = VSS

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog intput voltage	VAIN		AVSS		AVCC	V
Error (Not including quantization errors)	−	V_{CC} = 2.7 V to 3.6 V		±1.0	±4.0	LSB
		V_{CC} = 2 V ± 10%		±1.0	±4.0	

Note 1: 1 LSB = (AVCC − AVSS)/1024 [V]

Note 2: Minimum operation frequency:
The operaion of AD converter is guranteed only using fc (High frequency oscillator).
fs (Low frequency oscillator) is not guranteed. But When frequency of clock selected by clock gear is more than and eqaull 4 MHz in using fc, it is guranteed (f_{FPH} ≥ 4 MHz).

Note 3: The value for Icc (Current of VCC pin) includes the current which flows through the AVCC pin.

4.5 Serial Channel Timing (I/O interface mode)

(1) SCLK input mode

Parameter	Symbol	Valiable Min	Valiable Max	10 MHz Min	10 MHz Max	27 MHz Min	27 MHz Max	Unit
SCLK period	t_{SCY}	16X		1.6		0.59		μs
Output data → SCLK rising/falling	t_{OSS}	$t_{SCY}/2 - 4X - 110$ ($V_{CC} = 2.7$ V to 3.6 V)		290		38		ns
		$t_{SCY}/2 - 4X - 180$ ($V_{CC} = 2$ V ± 10%)		220		–		
SCLK rising/falling → Output data hold	t_{OHS}	$t_{SCY}/2 + 2X + 0$		1000		370		ns
SCLK rising/falling → Input data hold	t_{HSR}	$3X + 10$		310		121		ns
SCLK rising/falling → Valid data input	t_{SRD}		$t_{SCY} - 0$		1600		592	ns
Valid data input → SCLK rising/falling	t_{RDS}	0		0		0		ns

(2) SCLK ouptut mode

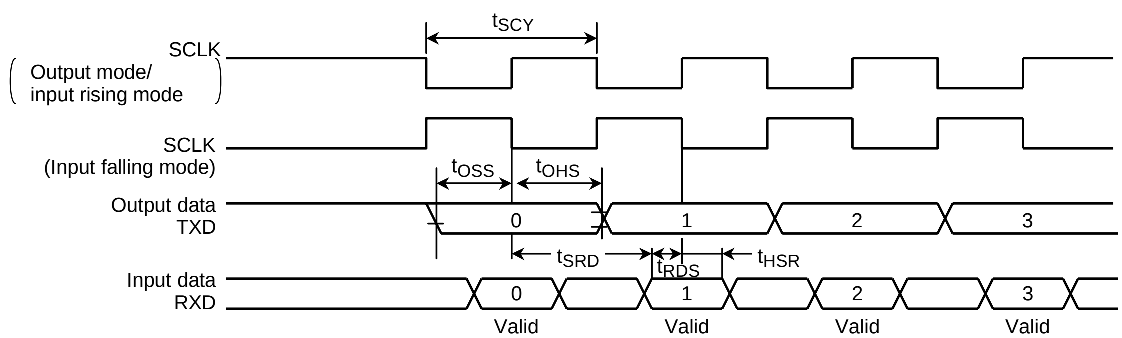
Parameter	Symbol	Valiable Min	Valiable Max	10 MHz Min	10 MHz Max	27 MHz Min	27 MHz Max	Unit
SCLK period	t_{SCY}	16X	8192X	1.6	819	0.59	303	μs
Output data → SCLK rising/falling	t_{OSS}	$t_{SCY}/2 - 40$		760		256		ns
SCLK rising/falling → Output data hold	t_{OHS}	$t_{SCY}/2 - 40$		760		256		ns
SCLK rising/falling → Input data hold	t_{HSR}	0		0		0		ns
SCLK rising/falling → Valid data input	t_{SRD}		$t_{SCY} - 1X - 180$		1320		375	ns
Valid data input → SCLK rising/falling	t_{RDS}	$1X + 180$		280		217		ns

Note 1: SCLK rising/falling: The rising edge is used in SCLK rising mode.
The falling edge is used in SCLK falling mode.

Note 2: 27 MHz and 10 MHz values are calculated from $t_{SCY} = 16X$ case.

Note 3: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.



4.6 Event Counter (TA0IN, TA4IN, TB0IN0 and TB0IN1)

Parameter	Symbol	Valiable		10 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock period	t_{VCK}	8X + 100		900		396		ns
Clock low level pulse width	t_{VCKL}	4X + 40		440		188		ns
Clock high level pulse width	t_{VCKH}	4X + 40		440		188		ns

4.7 Interrupt and Capture

(1) $\overline{\text{NMI}}$ and INT0 Interrupts

Parameter	Symbol	Variable		10 MHz		27 MHz		Unit
		Min	Max	Min	Max	Min	Max	
$\overline{\text{NMI}}$ and INT0 low level pulse width	t_{INTAL}	4X + 40		440		188		ns
$\overline{\text{NMI}}$ and INT0 high level pulse width	t_{INTAH}	4X + 40		440		188		ns

(2) INT5 and INT6 interrupts, capture

INT5 and INT6 input pulse width depend on the system clock selection and clock selection for prescaler. Below table show pulse width of each operation clock.

System Clock Selection SYSCR1 <SYSCK>	Clock Selection for Prescaler SYSCR0 <PRCK1:0>	t_{INTBL} (INT5 and INT6 low level pulse width)		t_{INTBH} (INT5 and INT6 high level pulse width)		Unit
		Valiable	f_{FPH} = 27 MHz	Valiable	f_{FPH} = 27MHz	
		Min	Min	Min	Min	
0 (fc)	00 (f_{FPH})	8X + 100	396	8X + 100	396	ns
	10 (fc/16)	128Xc + 0.1	4.8	128Xc + 0.1	4.8	μs
1 (fs)	00 (f_{FPH})	8X + 0.1	244.3	8X + 0.1	244.3	

Note 1: "Xc" shows period of clock fc in high frequency oscillator.

Note 2: Symbol [x] in the above table means the period of clock f_{FPH}. It's half period the system clock f_{SYS} for CPU core.

The period of clock f_{FPH} depends on the clock gear setting or the selection of high/low oscillator frequency.

4.8 Recommended Oscillation Circuit

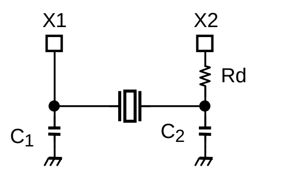
TMP91CP27 has been evaluated by murata manufacturing Co., Ltd. Please refer to murata manufacturing Co., Ltd.

Note 1: Total loads value of oscillator is sum of external loads (C1 and C2) and floating loads of actual assemble board. There is a possibility of miss-operating. When designing board, it should design minimum length pattern around oscillator. And we recommend that oscillator evaluation try on your actual using board.

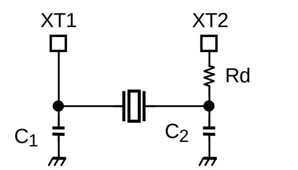
Note 2: The product numbers and specifications of the resonators by murata manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following URL:

http://www.murata.co.jp/search/index.html

(1) Connection example



High-frequency oscillation connection



Low-frequency oscillation connection

(2) Recommended ceramic oscillator

Oscillation Frequency [MHz]	Item of Oscillator	Parameter of Elements				Running Condition	
		C1 [pF]	C2 [pF]	Rf [Ω]	Rd [Ω]	Voltage of Power [V]	Temperature [°C]
4.0	CSTLS4M00G56-B0 (CSTS0400MG06)	(47)	(47)	Open	0	2.7~3.6	−40~85
8.0	CSTLS8M00G56-B0 (CSTS0800MG06)	(47)	(47)	Open	0		
10.0	CSTLS10M0G53-B0 (CSTS1000MG03)	(15)	(15)	Open	0		
12.0	CSTLA12M0T55-B0 (CST12. 0MTW)	(30)	(30)	Open	0		
16.0	CSTLS16M0X51-B0 (New and old is same product No.)	(5)	(5)	Open	0		
20.0	CSTLS20M0X51-B0 (New and old is same product No.)	(5)	(5)	Open	0		
27.0	CSALS27M0X51-B0 (New and old is same product No.)	Open	Open	10 k	0		

Oscillation Frequency [MHz]	Item of Oscillator	Parameter of Elements				Running Condition	
		C1 [pF]	C2 [pF]	Rf [Ω]	Rd [Ω]	Voltage of Power [V]	Temperature [°C]
4.0	CSTLS4M00G56U-B0 (CSTS0400MG06-951)	(47)	(47)	Open	0	1.8~2.2	−40~85
8.0	CSTLS8M00G56U-B0 (CSTS0800MG06-951)	(47)	(47)	Open	0		
10.0	CSTLS10M0G53U-B0 (CSTS1000MG03-951)	(15)	(15)	Open	0		

Note: In CST*** type oscillator, capacitance C1, C2 is builtin.

5. Table of SFRs

The SFRs (Special function registers) include the I/O ports and peripheral control registers allocated to the 4-Kbyte address space from 000000H to 000FFFH.

(1) I/O port
(2) I/O port control
(3) Interrupt control
(4) Chip select/wait control
(5) Clock control
(6) 8-bit timer control
(7) 16-bit timer control
(8) UART/serial channel control
(9) I²C bus/serial channel control
(10) AD converter control
(11) Watchdog timer control
(12) RTC (Real time clock) control

Table layout

Symbol	Name	Address	7	6			1	0	
									→ Bit symbol
									→ Read/Write
									→ Initial value after reset
									→ Remarks

Note: "Prohibit RMW" in the table means that you cannot use RMW instructions on these registers.

Example: When setting only bit0 of the register P0CR to "1", the instruction "SET 0, (0002H)" cannot be used. The LD (Transfer) instruction must be used to write all eight bits.

Read/Write

R/W: Both read and write are possible.

R: Only read is possible

W: Only write is possible

W*: Both read and write are possible (when this bit is read as 1.)

Prohibit RMW: Read-modify-write instructions are prohibited. (The EX, ADD, ADC, BUS, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD and RRD instruction are read-modify-write instructions.)

*R/W: Read-modify-write instructions are prohibited when controlling the pull-up resistor.

Table 5.1 Address Map for SFRs

[1] Port

Address	Name
0000H	P0
1H	P1
2H	P0CR
3H	
4H	P1CR
5H	P1FC
6H	P2
7H	P3
8H	P2CR
9H	P2FC
AH	P3CR
BH	P3FC
CH	P4
DH	P5
EH	P4CR
FH	P4FC

Address	Name
0010H	
1H	
2H	P6
3H	P7
4H	P6CR
5H	P6FC
6H	P7CR
7H	P7FC
8H	P8
9H	P9
AH	P8CR
BH	P8FC
CH	P9CR
DH	P9FC
EH	
FH	

Address	Name
0020H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	ODE

[2] INTC

Address	Name
0080H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	
5H	
6H	
7H	
8H	INTCLR
9H	DMAR
AH	DMAB
BH	
CH	IIMC
DH	
EH	
FH	

Address	Name
0090H	INTE0AD
1H	
2H	
3H	INTE56
4H	
5H	INTETA01
6H	INTETA23
7H	INTETA45
8H	
9H	INTETB0
AH	
BH	INTETB01V
CH	INTES0
DH	INTES1
EH	INTES2RTC
FH	

Address	Name
00A0H	INTETC01
1H	INTETC23
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[3] CS/WAIT

Address	Name
00C0H	B0CS
1H	B1CS
2H	B2CS
3H	B3CS
4H	
5H	
6H	
7H	BEXCS
8H	MSAR0
9H	MAMR0
AH	MSAR1
BH	MAMR1
CH	MSAR2
DH	MAMR2
EH	MSAR3
FH	MAMR3

Note: Do not access to the unnamed addresses, e.g., addresses to which no register has been allocated.

Table 5.2 Address Map for SFRs

[4] CGEAR

Address	Name	Address	Name
00E0H	SYSCR0	00F0H	
1H	SYSCR1	1H	
2H	SYSCR2	2H	
3H	EMCCR0	3H	
4H	EMCCR1	4H	
5H		5H	
6H		6H	
7H		7H	
8H		8H	
9H		9H	
AH		AH	
BH		BH	
CH		CH	
DH		DH	
EH		EH	
FH		FH	

[5] TMRA

Address	Name	Address	Name
0100H	TA01RUN	0110H	TA45RUN
1H		1H	
2H	TA0REG	2H	TA4REG
3H	TA1REG	3H	TA5REG
4H	TA01MOD	4H	TA45MOD
5H	TA1FFCR	5H	TA5FFCR
6H		6H	
7H		7H	
8H	TA23RUN	8H	
9H		9H	
AH	TA2REG	AH	
BH	TA3REG	BH	
CH	TA23MOD	CH	
DH	TA3FFCR	DH	
EH		EH	
FH		FH	

[6] TMRB

Address	Name	Address	Name
0180H	TB0RUN	0190H	
1H		1H	
2H	TB0MOD	2H	
3H	TB0FFCR	3H	
4H		4H	
5H		5H	
6H		6H	
7H		7H	
8H	TB0RG0L	8H	
9H	TB0RG0H	9H	
AH	TB0RG1L	AH	
BH	TB0RG1H	BH	
CH	TB0CP0L	CH	
DH	TB0CP0H	DH	
EH	TB0CP1L	EH	
FH	TB0CP1H	FH	

Note: Do not access to the unnamed addresses, e.g., addresses to which no register has been allocated.

Table 5.3 Address Map for SFRs

[7] UART/SIO

Address	Name
0200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	SIRCR
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	
FH	

[8] I^2C bus/SIO

Address	Name
0240H	SBI0CR1
1H	SBI0DBR
2H	I2C0AR
3H	SBI0CR2/SBI0SR
4H	SBI0BR0
5H	SBI0BR1
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[9] 10-bit ADC

Address	Name
02A0H	ADREG04L
1H	ADREG04H
2H	ADREG15L
3H	ADREG15H
4H	ADREG26L
5H	ADREG26H
6H	ADREG37L
7H	ADREG37H
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
02B0H	ADMOD0
1H	ADMOD1
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] WDT

Address	Name
0300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[11] RTC

Address	Name
0310H	RTCCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses, e.g., addresses to which no register has been allocated.

(1) I/O port

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0	Port 0	00H	P07	P06	P05	P04	P03	P02	P01	P00
			R/W							
			Data from external port (Output latch register is undefined)							
P1	Port 1	01H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Data from external port (Output latch register is cleared to "0")							
P2	Port 2	06H			P25	P24	P23	P22	P21	P20
					R/W					
					Data from external port (Output latch register is set to "1")					
P3	Port 3	07H (Prohibit RMW)						P32	P31	P30
								*R/W		
								Data from external port Note1	1	1
								Pull-up resistor 0: OFF 1: ON	–	–
P4	Port 4	0CH (Prohibit RMW)						P42	P41	P40
								*R/W		
								Data from external port Note1		
								Pull-up resistor 0: OFF 1: ON		
P5	Port 5	0DH					P53	P52	P51	P50
							R			
							Data from external port			
P6	Port 6	12H					P63	P62	P61	P60
							R/W			
							Data from external port (Output latch register is set to "1")			
P7	Port 7	13H				P74	P73	P72	P71	P70
						R/W				
						Data from external port (Output latch register is set to "1")				
P8	Port 8	18H					P83	P82	P81	P80
							R/W			
							Data from external port (Output latch register is set to "1")			
P9	Port 9	19H	P97	P96	P95	P94	P93	P92	P91	P90
			R/W	R/W	R/W					
			1	1	Data from external port (Output latch register is set to "1")					

Note: Output latch is set to "1", and pull-up resistor is connected.

(2) I/O port control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0CR	Port 0 control	02H (Prohibit RMW)	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output (When access to external, become AD7 to AD0 and this register is cleared to "0".)							
P1CR	Port 1 control	04H (Prohibit RMW)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			<<Refer to column of P1FC>>							
P1FC	Port 1 function	05H (Prohibit RMW)	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
			W							
			0	0	0	0	0	0	0	0
			P1FC/P1CR = 00: Input port, 01: Output port, 10: AD8 to AD15, 11: A8 to A15							
P2CR	Port 2 control	08H (Prohibit RMW)			P25C	P24C	P23C	P22C	P21C	P20C
					W					
					0	0	0	0	0	0
					<<Refer to column of P2FC>>					
P2FC	Port 2 function	09H (Prohibit RMW)			P25F	P24F	P23F	P22F	P21F	P20F
					W					
					0	0	0	0	0	0
					P2FC/P2CR = 00: Input port, 01: Output port, 10: A0 to A5, 11: A16 to A21					
P3CR	Port 3 control	0AH (Prohibit RMW)						P32C		
								W		
								0		
								0: Input 1: Output		
P3FC	Port 3 function	0BH (Prohibit RMW)	–					P32F	P31F	P30F
			W					W		
			0					0	0	0
			Write "0".					0: Port 1: $\overline{HWR}$	0: Port 1: $\overline{WR}$	0: Port 1: $\overline{RD}$
P4CR	Port 4 control	0EH (Prohibit RMW)						P42C	P41C	P40C
								W		
								0	0	0
								0: Input 1: Output		
P4FC	Port 4 function	0FH (Prohibit RMW)						P42F	P41F	P40F
								W		
								0	0	0
								0: Port 1: $\overline{CS2}$	0: Port 1: $\overline{CS1}$	0: Port 1: $\overline{CS0}$

Note 1: When port 2 is used as address bus A21 to A16 or A5 to A0, set P2FC after set P2CR.

Note 2: "L" level is outputted from P30 pin also during reading internal area by setting P3<P30> to "0", set P3FC<P30F> to "1".

Note 3: When port 4 is used as chip select signal $\overline{CS0}$ to $\overline{CS2}$ set P4CR to "1" after set P4FC to "1".

I/O port control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P6CR	Port 6 control	14H (Prohibit RMW)					P63C	P62C	P61C	P60C
							W			
							0	0	0	0
							0: Input 1: Output			
P6FC	Port 6 function	15H (Prohibit RMW)					P63F	P62F	P61F	P60F
							W			
							0	0	0	0
							0: Port 1: INT0	0: Port 1: SCL	0: Port 1: SDA/SO	0: Port 1: SCK out
P7CR	Port 7 control	16H (Prohibit RMW)				P74C	P73C	P72C	P71C	P70C
						W				
						0	0	0	0	0
						0: Input 1: Output				
P7FC	Port 7 function	17H (Prohibit RMW)				P74F		P72F	P71F	
						W		W	W	
						0		0	0	
						0: Port 1: TA5OUT		0: Port 1: TA3OUT	0: Port 1: TA1OUT	
P8CR	Port 8 control	1AH (Prohibit RMW)					P83C	P82C	P81C	P80C
							W			
							0	0	0	0
							0: Input 1: Output			
P8FC	Port 8 function	1BH (Prohibit RMW)					P83F	P82F	P81F	P80F
							W			
							0	0	0	0
							0: Port 1: TB0OUT1	0: Port 1: TB0OUT0	0: Port 1: INT6/ TB0IN1	0: Port 1: INT5/ TB0IN0
P9CR	Port 9 control	1CH (Prohibit RMW)	P97C	P96C	P95C	P94C	P93C	P92C	P91C	P90C
			W	W	W					
			1	1	0	0	0	0	0	0
			0: Input 1: Output							
P9FC	Port 9 function	1DH (Prohibit RMW)			P95F		P93F	P92F		P90F
					W		W	W		W
					0		0	0		0
					0: Port 1: SCLK1		0: Port 1: TXD1	0: Port 1: SCLK0		0: Port 1: TXD0
ODE	Open-drain enable	2FH					ODE62	ODE61	ODE93	ODE90
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: P62ODE	1: P61ODE	1: P93ODE	1: P90ODE

Note 1: External interrupt INT0:
Input enable is controlled by P6FC<P63F>. Level/edge selection and rising/falling selection is controlled by IIMC<I0LE, I0EDGE>.

Note 2: External interrupts INT5 and INT6:
Input enable is set by P8FC<P81F, P80F>. The setting of edge is controlled by TB0MOD.

Note 3: When P70 and P73 is used as an input port, the input signal is inputted to 8bit-timer (TMRA0 and TMRA4) as TA0IN and TA4IN inputs.

Note 4: When P91 and P94 is used as an input port, the input signal is inputted to SIO as serial receiving data RXD0 and RXD1.

(3) Interrupt control (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	Interrupt enable INT0 & AD	90H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTAD	Interrupt request level			1: INT0	Interrupt request level		
INTE56	Interrupt enable INT6/5	93H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT6	Interruput requeset level			1: INT5	Interrupt request level		
INTETA01	Interrupt enable TMRA 1/0	95H	INTTA1 (TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA1	Interrupt request level			1: INTTA0	Interrupt request level		
INTETA23	Interrupt enable TMRA 3/2	96H	INTTA3 (TMRA3)				INTTA2 (TMRA2)			
			ITA3C	ITA3M2	ITA3M1	ITA3M0	ITA2C	ITA2M2	ITA2M1	ITA2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA3	Interrupt request level			1: INTTA2	Interrupt request level		
INTETA45	Interrupt enable TMRA 5/4	97H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA5	Interrupt request level			1: INTTA4	Interrupt request level		

Interrupt control (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB0	Interrupt enable TMRB0	99H	INTTB01 (TMRB0)				INTTB00 (TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB01	Interrupt request level			1: INTTB00	Interrupt request level		
INTETB01V	Interrupt enable TMRB0 (Overflow)	9BH					INTTBOF0 (TMRB0 over flow)			
			–	–	–	–	ITF0C	ITF0M2	ITF0M1	ITF0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
				Write “0”.			1: INTTBOF0	Interrupt request level		
INTES0	Interrupt enable serial 0	9CH	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX0	Interrupt request level			1: INTRX0	Interrupt request level		
INTES1	Interrupt enable serial 1	9DH	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX1	Interrupt request level			1: INTRX1	Interrupt request level		
INTES2RTC	Interrupt enable SBI/RTC	9EH	INTRTC				INTSBI			
			IRTCC	IRTCM2	IRTCM1	IRTCM0	IS2C	IS2M2	IS2M1	IS2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTRTC	Interrupt request level			1: INTSBI	Interrupt request level		
INTETC01	Interrupt enable INTTC0/1	A0H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23	Interrupt enable INTTC2/3	A1H	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

Interrupt control (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 start vector	80H			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					R/W					
					0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 start vector	81H			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					R/W					
					0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 start vector	82H			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					R/W					
					0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 start vector	83H			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					R/W					
					0	0	0	0	0	0
					DMA3 start vector					
INTCLR	Interrupt clear control	88H (Prohibit RMW)			CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
					W					
					0	0	0	0	0	0
					Clear interrupt request flag by writing DMA start vector					
DMAR	DMA software request register	89H					DMAR3	DMAR2	DMAR1	DMAR0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA request in software			
DMAB	DMA burst request register	8AH					DMAB3	DMAB2	DMAB1	DMAB0
							R/W	R/W	R/W	R/W
							0	0	0	0
							1: DMA requesst on burst mode			
IIMC	Interrupt input mode control	8CH (Prohibit RMW)	–	–	–	–	–	I0EDGE	I0LE	NMIREE
			W	W	W	W	W	W	W	W
			0	0	0	0	0	0	0	0
			Write "0".					INT0 edge 0: Rising 1: Falling	INT0 0: Edge 1: Level	1: Operation even on $\overline{NMI}$ rising edge

Note: Only one channel can be set once for DMAR register. (Don't write "1" to plural bits.)

(4) Chip select/wait control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B0CS	Block 0 CS/WAIT control register	C0H (Prohibit RMW)	B0E		B0OM1	B0OM0	B0BUS	B0W2	B0W1	B0W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		00: ROM/SRAM 01: } Reserved 10: 11:		Data bus width selection 0: 16 bits 1: 8 bits	Set number of wait 000: 2 waits 100: Reserved 001: 1 wait 101: 3 waits 010: (1 + N) waits 110: 4 waits 011: 0 waits 111: 8 waits		
B1CS	Block 1 CS/WAIT control register	C1H (Prohibit RMW)	B1E		B1OM1	B1OM0	B1BUS	B1W2	B1W1	B1W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		00: ROM/SRAM 01: } Reserved 10: 11:		Data bus width selection 0: 16 bits 1: 8 bits	Set number of wait 000: 2 waits 100: Reserved 001: 1 wait 101: 3 waits 010: (1 + N) waits 110: 4 waits 011: 0 waits 111: 8 waits		
B2CS	Block 2 CS/WAIT control register	C2H (Prohibit RMW)	B2E	B2M	B2OM1	B2OM0	B2BUS	B2W2	B2W1	B2W0
			W	W	W	W	W	W	W	W
			1	0	0	0	0	0	0	0
			0: Disable 1: Enable	0:16-MB area 1: Area setting	00: ROM/SRAM 01: } Reserved 10: 11:		Data bus width selection 0: 16 bits 1: 8 bits	Set number of wait 000: 2 waits 100: Reserved 001: 1 wait 101: 3 waits 010: (1 + N) waits 110: 4 waits 011: 0 waits 111: 8 waits		
B3CS	Block 3 CS/WAIT control register	C3H (Prohibit RMW)	B3E		B3OM1	B3OM0	B3BUS	B3W2	B3W1	B3W0
			W		W	W	W	W	W	W
			0		0	0	0	0	0	0
			0: Disable 1: Enable		00: ROM/SRAM 01: } Reserved 10: 11:		Data bus width selection 0: 16 bits 1: 8 bits	Set number of wait 000: 2 waits 100: Reserved 001: 1 wait 101: 3 waits 010: (1 + N) waits 110: 4 waits 011: 0 waits 111: 8 waits		
BEXCS	External CS/WAIT control register	C7H (Prohibit RMW)					BEXBUS	BEXW2	BEXW1	BEXW0
							W	W	W	W
							0	0	0	0
							Data bus width selection 0: 16 bits 1: 8 bits	Set number of wait 000: 2 waits 100: Reserved 001: 1 wait 101: 3 waits 010: (1 + N) waits 110: 4 waits 011: 0 waits 111: 8 waits		
MSAR0	Memory start address register 0	C8H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR0	Memory address mask register 0	C9H	V20	V19	V18	V17	V16	V15	V14~9	V8
			R/W							
			1	1	1	1	1	1	1	1
			CS0 area size setting 0: Address comparison							
MSAR1	Memory start address register 1	CAH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR1	Memory address mask register 1	CBH	V21	V20	V19	V18	V17	V16	V15~9	V8
			R/W							
			1	1	1	1	1	1	1	
			CS1 area size setting 0: Address comparison							

Note: TMP91CP27 don't include $\overline{\text{WAIT}}$ pin. Therefore, when select "(1 + N) waits", operation is same with "1 wait".

Chip select/wait control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MSAR2	Memory start address register 2	CCH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR2	Memory address mask register 2	CDH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			CS2 area size setting 0: Address comparison							
MSAR3	Memory start address register 3	CEH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR3	Memory address mask register 3	CFH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			CS3 area size setting 0: Address comparison							

(5) Clock control

Symbol	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System clock control register 0	E0H	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	PRCK1	PRCK0
			R/W							
			1	0	1	0	0	0	0	0
			High-frequency oscillator (fc) 0: Stopped 1: Oscillation	Low-frequency oscillator (fs) 0: Stopped 1: Oscillation	High-frequency oscillator(fc) after release of stop mode 0: Stopped 1: Oscillation	Low-frequency oscillator(fs) after release of stop mode 0: Stopped 1: Oscillation	Clock after release of STOP mode 0: fc 1: fs	Warm-up (WUP) 0 write: Don't care 1 write: Warm-up start 0 read: End of WUP 1 read: Don't end WUP	Select prescaler clock 00: f_{FPH} 01: Reserved 10: fc/16 11: Reserved	
SYSCR1	System clock control register 1	E1H					SYSCK	GEAR2	GEAR1	GEAR0
							R/W			
							0	1	0	0
							Clock selection 0: fc 1: fs	Select gear of high frequency clock 000: fc 001: fc /2 010: fc /4 011: fc /8 100: fc /16 Others: Reserved		
SYSCR2	System clock control register 2	E2H		–	WUPTM1	WUPTM0	HALTM1	HALTM0		DRVE
				R/W	R/W	R/W	R/W	R/W		R/W
				0	1	0	1	1		0
				Write "0".	WUP time for oscillator 00: Reserved 01: 2^{8}/input frequency 10: 2^{14}/input frequency 11: 2^{16}/input frequency		HALT mode 00: Reserved 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode			1: Drive the pin in the stop mode
EMCCR0	EMC control register 0	E3H	PROTECT	–	–	–	ALEEN	EXTIN	DRVOSCH	DRVOSCL
			R	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			0	0	1	0	0	0	1	1
			Protect flag 0: OFF 1: ON	Write "0".	Write "1".	Write "0".	ALE Output 0: Disable 1: Enable	1: fc external clock	fc oscillator drive ability 1: Normal 0: Weak	fs oscillator drive abillity 1: Normal 0: Weak
EMCCR1	EMC control register 1	E4H	Write "1FH": Protect OFF Write except "1FH": Protect ON							

Note 1: If protection is on by writing except "1FH" code to EMCCR1 register, write operations to the following SFRs are not possible.

(1) CS/WAIT controller
B0CS, B1CS, B2CS, B3CS, BEXCS,
MSAR0, MSAR1, MSAR2, MSAR3,
MAMR0, MAMR1, MAMR2, MAMR3

(2) Clock gear (EMCCR1 can be written to)
SYSCR0, SYSCR1, SYSCR2, EMCCR0

Note 2: When using internal SBI, set SYSCR0<PRCK1:0> to "00".

(6) 8-bit timer control (1/2)

(6 – 1) TMRA01

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA01RUN	TMRA01 RUN	100H	TA0RDE				I2TA01	TA01PRUN	TA1RUN	TA0RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stopped 1: Operation	8-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		
TA0REG	TMRA0 register	102H (Prohibit RMW)	–							
			W							
			Undefined							
TA1REG	TMRA1 register	103H (Prohibit RMW)	–							
			W							
			Undefined							
TA01MOD	TMRA01 source CLK & mode	104H	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA0 00: TA0IN pin input 01: ϕT1 10: ϕT4 11: ϕT16	
TA1FFCR	TMRA01 flip-flop control	105H (Prohibit RMW)					TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
							R/W		R/W	
							1	1	0	0
							00: Invert TA1FF 01: SET TA1FF 10: Clear TA1FF 11: Don't care		1: TA1FF invert enable	Invertion by 0: TMRA0 1: TMRA1

(6 – 2) TMRA23

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA23RUN	TMRA23 RUN	108H	TA2RDE				I2TA23	TA23PRUN	TA3RUN	TA2RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stopped 1: Operation	8-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		
TA2REG	TMRA2 register	10AH (Prohibit RMW)	–							
			W							
			Undefined							
TA3REG	TMRA3 register	10BH (Prohibit RMW)	–							
			W							
			Undefined							
TA23MOD	TMRA23 source CLK & mode	10CH	TA23M1	TA23M0	PWM21	PWM20	TA3CLK1	TA3CLK0	TA2CLK1	TA2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock for TMRA3 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA2 00: Reserved 01: ϕT1 10: ϕT4 11: ϕT16	
TA3FFCR	TMRA23 flip-flop control	10DH (Prohibit RMW)					TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
							R/W		R/W	
							1	1	0	0
							00: Invert TA3FF 01: SET TA3FF 10: Clear TA3FF 11: Don't care		1:TA3FF invert enable	Invert by 0: TMRA2 1: TMRA3

8-bit timer control (2/2)

(6 – 3) TMRA45

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA45RUN	TMRA45 RUN	110H	TA4RDE				I2TA45	TA45PRUN	TA5RUN	TA4RUN
			R/W				R/W	R/W	R/W	R/W
			0				0	0	0	0
			Double buffer 0: Disable 1: Enable				IDLE2 0: Stopped 1: Operation	8-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		
TA4REG	TMRA4 register	112H (Prohibit RMW)	–							
			W							
			Undefined							
TA5REG	TMRA5 register	113H (Prohibit RMW)	–							
			W							
			Undefined							
TA45MOD	TMRA45 source CLK & mode	114H	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00 : Reserved 01: $2^6 – 1$ 10: $2^7 – 1$ 11: $2^8 – 1$		Source clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock for TMRA4 00: TA4IN Pin input 01: ϕT1 10: ϕT4 11: ϕT16	
TA5FFCR	TMRA45 flip-flop control	115H (Prohibit RMW)					TAFF5C1	TAFF5C0	TAFF5IE	TAFF5IS
							R/W		R/W	
							1	1	0	0
							00: Invert TA5FF 01: SET TA5FF 10: Clear TA5FF 11: Don't care		1: TA5FF Invert enable	Invert by 0: TMRA4 1: TMRA5

(7) 16-bit timer control

(7 – 1) TMRB0

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB0RUN	TMRB0 Run	180H	TB0RDE	–			I2TB0	TB0PRUN		TB0RUN
			R/W	R/W			R/W	R/W		R/W
			0	0			0	0		0
			Double buffer 0: Disable 1: Enable	Write “0”.			IDLE2 0: Stopped 1: Operation	16-bit timer run/stop control 0: Stop and clear 1: Run (Count up)		
TB0MOD	TMRB0 source CLK & mode	182H (Prohibit RMW)	TB0CT1	TB0ET1	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB0FF1 Inversion trigger 0: Disable 1: Enable		0: Software capture 1: Undefined	Capture timming 00: Disable 01: ↑, ↑ (TB0IN0, TB0IN1) 10: ↑, ↓ (TB0IN0) 11: ↑, ↓ (TA1OUT)		1:UC0 clear enable	Source clock 00: TB0IN0 input 01: ϕT1 10: ϕT4 11: ϕT16	
			Capture to TB0CP1	TB0RG1 matching						
TB0FFCR	TMRB0 flip-flop control	183H (Prohibit RMW)	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			00: Invert TB0FF1 01: Set TB0FF1 10: Clear TB0FF1 11: Don’t care		TB0FF0 invert trigger 0: Disable 1: Enable				00: Invert TB0FF0 01: Set TB0FF0 10: Clear TB0FF0 11: Don’t care	
					Invert when the UC value is loaded into TB0CP1	Invert when the UC value is loaded into TB0CP0	Invert when the UC matches with TB0RG1	Invert when the UC matches with TB0RG0		
TB0RG0L	TMRB0 register 0 Low	188H (Prohibit RMW)	–							
			W							
			Undefined							
TB0RG0H	TMRB0 register 0 High	189H (Prohibit RMW)	–							
			W							
			Undefined							
TB0RG1L	TMRB0 register 1 Low	18AH (Prohibit RMW)	–							
			W							
			Undefined							
TB0RG1H	TMRB0 register 1 High	18BH (Prohibit RMW)	–							
			W							
			Undefined							
TB0CP0L	TMRB0 capture register 0 Low	18CH	–							
			R							
			Undefined							
TB0CP0H	TMRB0 capture register 0 High	18DH	–							
			R							
			Undefined							
TB0CP1L	TMRB0 capture register 1 Low	18EH	–							
			R							
			Undefined							
TB0CP1H	TMRB0 capture register 1 High	18FH	–							
			R							
			Undefined							

Note: When programming “1” to TB0MOD<TB0CP0I> in condition of programmed “0”, present value of up-counter is captured to TB0CP0 register.

(8) UART/serial channel control (1/2)

(8 – 1) UART/SIO channel 0

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial channel 0 buffer	200H (prohibit RMW)	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	RB0/TB0
			R (Receiving)/W (Transmission)							
			Undefined							
SC0CR	Serial channel 0 control	201H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to 0 by reading)			R/W	
			Undefined	0	0	0	0	0	0	0
			Receiving data bit8	Parity 0: Odd 1: Even	1: Parity enable	1: Error			0: SCLK0 ↑ 1: SCLK0 ↓	1: SCLK0 pin input
						Overrun	Parity	Framing		
SC0MOD0	Serial channel 0 mode0	202H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit8	1: CTS enable	1: Receive enable	1: Wakeup enable	00: I/O interface 01: 7-bit UART 10: 8-bit UART 11: 9-bit UART		00: TA0TRG 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock SCLK0	
BR0CR	Serial channel 0 baud rate control	203H	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W							
			0	0	0	0	0	0	0	0
			Write "0".	1: (16 – K)/16 divided enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Set the dividing value "N" (0 to F).			
BR0ADD	Serial channel 0 K setting register	204H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
							0	0	0	0
							Set the value of "K" (1 to F).			
SC0MOD1	Serial channel 0 mode1	205H	I2S0	FDPX0						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Operation	I/O interface 1: Full duplex 0: Half duplex						

(8 – 2) IrDA

Symbol	Name	Address	7	6	5	4	3	2	1	0
SIRCR	IrDA control register	207H	PLSEL	RXSEL	TXEN	RXEN	SIRWD3	SIRWD2	SIRWD1	SIRWD0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission pulse width 0: 3/16 1: 1/16	Receiving data logic 0: "H" pulse 1: "L" pulse	Transmission operation 0: Disable 1: Enable	Receiving operation 0: Disable 1: Enable	Select effective pulse width Pulse width of more than and equal "2x × (Setting value + 1)" Possible: 1 to 14 Not possible: 0, 15			

UART/serial channel control(2/2)

(8 – 3) UART/SIO channel 1

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC1BUF	Serial channel 1 buffer	208H (Prohibit RMW)	RB7/TB7	RB6/TB6	RB5/TB5	RB4/TB4	RB3/TB3	RB2/TB2	RB1/TB1	RB0/TB0
			R (Receiving)/W (Transmission)							
			Undefined							
SC1CR	Serial channel 1 control	209H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to 0 by reading)			R/W	
			Undefined	0	0	0	0	0	0	0
			Receiving data bit 8	Parity 0: Odd 1: Even	1: Parity enable	1: Error Overrun	Parity	Framing	0: SCLK1 ↑ 1: SCLK1 ↓	1:SCLK1 pin input
SC1MOD0	Serial channel 1 mode 0	20AH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit8	1: CTS enable	1: Receive enable	1: Wakeup enable	00: I/O interface 01: 7-bit UART 10: 8-bit UART 11: 9-bit UART		00: TA0TRG 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock SCLK1	
BR1CR	Serial channel 1 baud rate control	20BH	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0	0	0	0	0	0
			Write “0”.	1: (16 – K)/16 divided enable	00: φT0 01: φT2 10: φT8 11: φT32		Set the dividing value “N” (0 to F).			
BR1ADD	Serial channel 1 K setting register	20CH					BR1K3	BR1K2	BR1K1	BR1K0
							R/W			
							0	0	0	0
							Set the value of “K” (1 to F).			
SC1MOD1	Serial channel 1 mode1	20DH	I2S1	FDPX1						
			R/W	R/W						
			0	0						
			IDLE2 0: Stop 1: Operation	I/O interface 1: Full duplex 0: Half duplex						

Note 1: As all error flags SCxCR<OERR, PERR,FERR> are cleared after reading, do not test only a single bit with a bit-testing instruction.

Note 2: The baud rate genetrator can be set N = “1” when UART mode and disable + (16 − K)/16 division function. Don’t use in I/O interface mode.

Note 3: Set BRxCR<BRxADDE> to “0” and disable + (16 − K)/16 division function in I/O interface mode.

(9) I²C bus/serial channel control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SBI0CR1	Serial bus interface control register 1	240H (I²C bus mode) (Prohibit RMW)	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0 /SWRMON
			W			R/W		W	W	R/W
			0	0	0	0		0	0	0/1
			Select number of transfer bit 000: 8 001: 1 010: 2 011: 3 100: 4 101: 5 110: 6 111: 7			Acknowledge mode 0: Disable 1: Enable		Internal serial clock selector (When writing) 000: 5 001: 6 010: 7 011: 8 100: 9 101: 10 110: 11 111: Reserved		
		240H (SIO mode) (Prohibit RMW)	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
			W	W	W	W		W		
			0	0	0	0		0	0	0
			Transfer control 0: Stop 1: Start	Forcing stop of transfer 0: Continue 1: Stop	Select transfer mode 00: 8-bit transmit 01: Reserved 10: 8-bit transmit/receiving 11: 8-bit receiving			Select frequency of serial clock 000: 4 001: 5 010: 6 011: 7 100: 8 101: 9 110: 10 111: External SCK input		
SBI0DBR	SBI data buffer register	241H (Prohibit RMW)	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
			R (Receiving)/W (Transmission)							
			Undefined							
I2C0AR	I²C bus address register	242H (Prohibit RMW)	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
			W							
			0	0	0	0	0	0	0	0
			Setting slave address							Address recognition 0: Enable 1: Disable

Note 1: When use built-in SBI, set SYSCR0<PRCK1:0> to f_{FPH}.

Note 2: Set the SBI0CR1<BC2:0> to “000” before switching to a clock-synchronous 8-bit SIO mode.

Note 3: Switch a mode to Port mode after confirming that the bus is free. And, switch from port mode to I²C bus mode or SIO mode after confirming port conditon = “H”.

Note 4: Set the transfer mode and the serial clock in SIO mode after clearing SBI0CR1<SIOS> to “0” and <SIOINH> to “1”.

Note 5: After reset, default value of SBI0CR1<SCK0> is cleared “0”, and default value of <SWRMON> is set “1”.

I^{2}C bus/serial channel control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SBI0CR2 (In writing)	Serial bus interface control register 2	243H (I^{2}C bus mode) (Prohibit RMW)	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
			W							
			0	0	0	1	0	0	0	0
			0: Slave 1: Master	0: Receiver 1: Transmitter	0: Generates the stop condition 1: Generates the start condition	1: Cancel INTSBI interrupt request	Operation mode selection 00:Port mode 01:Clocked synchronous 8-bit SIO mode 10:I^{2}C bus mode 11:Reserved		Software reset control Write “10 “ and “01” in order, then an internal reset signal is generated.	
		243H (SIO mode) (Prohibit RMW)					SBIM1	SBIM0	–	–
							W		W	W
							0	0	0	0
							Operation mode selection 00:Port mode 01:SIO mode 10:I^{2}C bus mode 11:Reserved		Write “0”.	
SBI0SR (In reading)	Serial bus interface status register	243H (I^{2}C bus mode)	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
			R							
			0	0	0	1	0	0	0	0
			0: Slave 1: Master	0: Receiver 1: Transmitter	0: Bus free 1: Bus busy	Interrupt request 0: Requested 1: Canceled	0: Undetected 1: Detect arbitration lost	0: – 1: Detect slave address match or general call	0: Undetected 1: Detect general call	Last received bit monitor 0: “0” 1: “1”
		243H (SIO mode)					SIOF	SEF		
							R			
							0	0		
							Transfer status monitor 0: Terminate 1: In progress	Shift operation status monitor 0: Terminate 1: In progress		
SBI0BR0	Serial bus interface baud rate register 0	244H (Prohibit RMW)	–	I2SBI0						
			W	R/W						
			0	0						
			Write “0”.	IDLE2 0: Stop 1: Operation						
SBI0BR1	Serial bus interface baud rate register 1	245H (Prohibit RMW)	P4EN	–						
			W	W						
			0	0						
			Clock control 0: Stop 1: Operation	Write “0”.						

(10) AD converter control

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD0	AD mode register 0	2B0H (Prohibit RMW)	EOCF	ADBF	–	–	ITM0	REPEAT	SCAN	ADS
			R		R/W	R/W	R/W	R/W	R/W	R/W
			0	0	0	0	0	0	0	0
			End flag 1: Conversion complete	Busy flag 1: Conversion in progress	Write “0”.	Write “0”.	Interrupt specification channel fixed repeat mode 0: Every conversion 1: Every fourth conversion	0: Single conversion 1: Repeat conversion	0: Channel fixed mode 1: Channel scan mode	1: Start conversion
ADMOD1	AD mode register 1	2B1H	VREFON	I2AD			ADTRGE	ADCH2	ADCH1	ADCH0
			R/W	R/W			R/W	R/W		
			0	0			0	0	0	0
			1: VREF on	IDLE2 0: Stop 1: Operation			External trigger start 0: Disable 1: Enable	Input channel selection Fixed/Scan 000: AN0/AN0 001: AN1/AN0 → AN1 010: AN2/AN0 → AN1 → AN2 011: AN3/AN0 → AN1 → AN2 → AN3 100: 101: Don’t select 110: 111:		
ADREG04L	AD result register 0/4 low	2A0H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
ADREG04H	AD result register 0/4 high	2A1H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
ADREG15L	AD result register 1/5 low	2A2H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
ADREG15H	AD result register 1/5 high	2A3H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
ADREG26L	AD result register 2/6 low	2A4H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
ADREG26H	AD result register 2/6 high	2A5H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
ADREG37L	AD result register 3/7 low	2A6H	ADR31	ADR30						ADR3RF
			R							R
			Undefined							0
ADREG37H	AD result register 3/7 high	2A7H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			Undefined							

Note 1: ADMOD0<ADS> is always read as “0”.

Note 2: When using $\overline{\text{ADTRG}}$ with ADMOD1<ADTRGE> = “1”, do not set ADMOD1<ADCH2:0> = “011.

Note 3: When set ADMOD1<I2AD> to ”0”, operation is different by AD conversion mode after released Halt mode.

(11) Watchdog timer control

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	WDT mode register	300H	WDTE	WDTP1	WDTP0	–	–	I2WDT	RESCR	–
			R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
			1	0	0	0	0	0	0	0
			1: WDT enable	00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$		Write "0".	Write "0".	IDLE2 0: Stop 1: Operation	1: Internaly connects WDT out to the reset pin	Write "0".
WDCR	WDT control	301H (Prohibit RMW)	–							
			W							
			–							
			B1H: WDT disable 4EH: WDT clear							

(12) RTC (Real time clock) control

Symbol	Name	Address	7	6	5	4	3	2	1	0
RTCCR	RTC control register	310H	–					RTCSEL1	RTCSEL0	RTCRUN
			R/W					R/W		R/W
			0					0	0	0
			Write "0".					00: 2^{14}/fs 01: 2^{13}/fs 10: 2^{12}/fs 11: 2^{11}/fs		0: Stop and clear 1: RUN

6. Port Section Equivalent Circuit Diagram

- Reading the circuit diagram

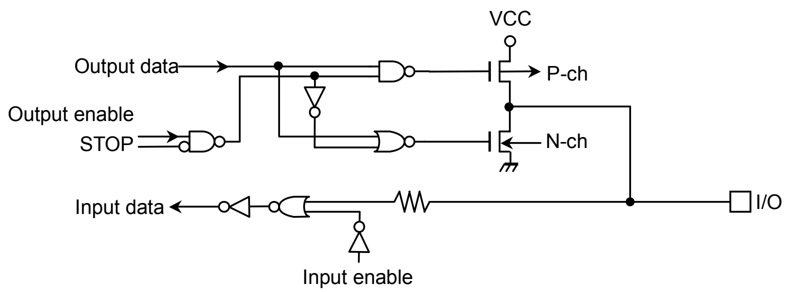
Basically, the gate symbols written are the same as those used for the standard CMOS logic IC “74HC××” series.

The dedicated signal is described below.

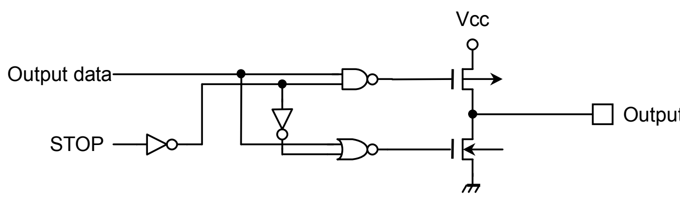
STOP: This signal becomes active “1” when the halt mode setting register is set to the STOP mode (SYSCR2<HALTM1:0> = 0, 1) and the CPU executes the HALT instruction. When the drive enable bit SYSCR2<DRVE> is set to “1”, stop remains at “0”.

- The input protection resistance ranges from several tens of ohms to several hundreds of ohms.

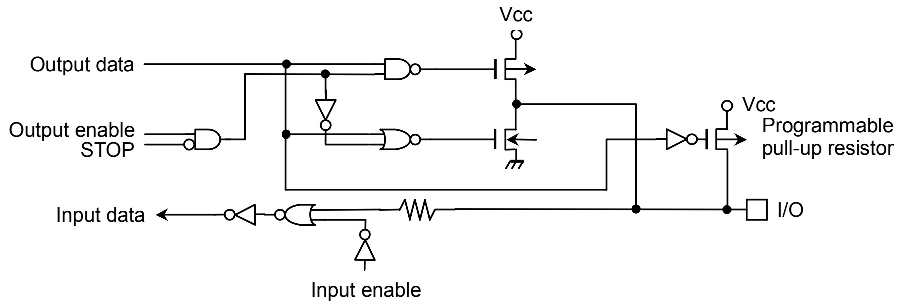
■ P0 (AD0 to AD7), P1 (AD8 to AD15, A8 to A15), P2 (A16 to A21, A0 to A5), P60, P70 to P74, P80 to P83, P91 to P92 and P94 to P95



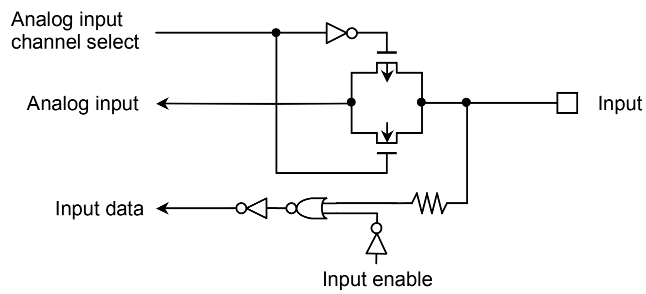
■ P30 ($\overline{\text{RD}}$), P31 ($\overline{\text{WR}}$)



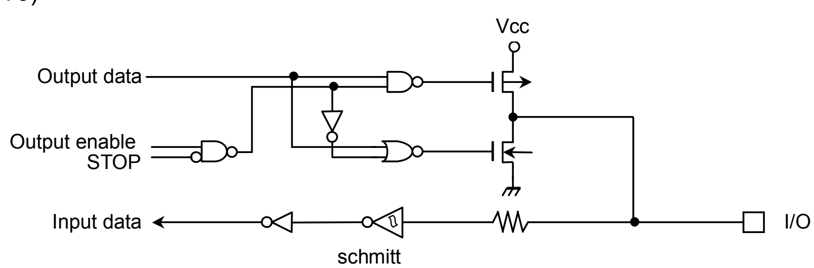
- P32, P40 to P42



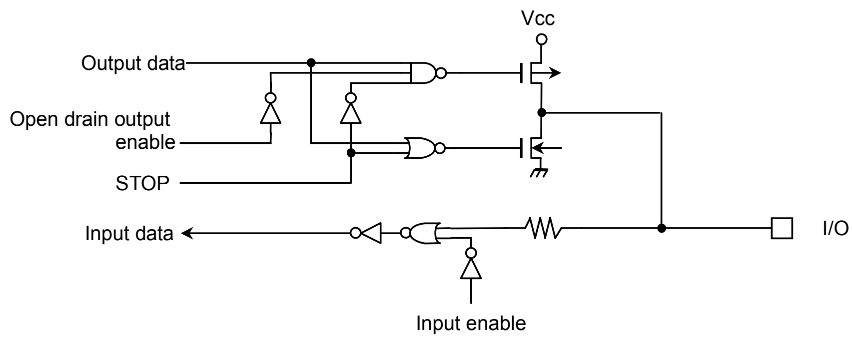
- P5 (AN0 to AN3)



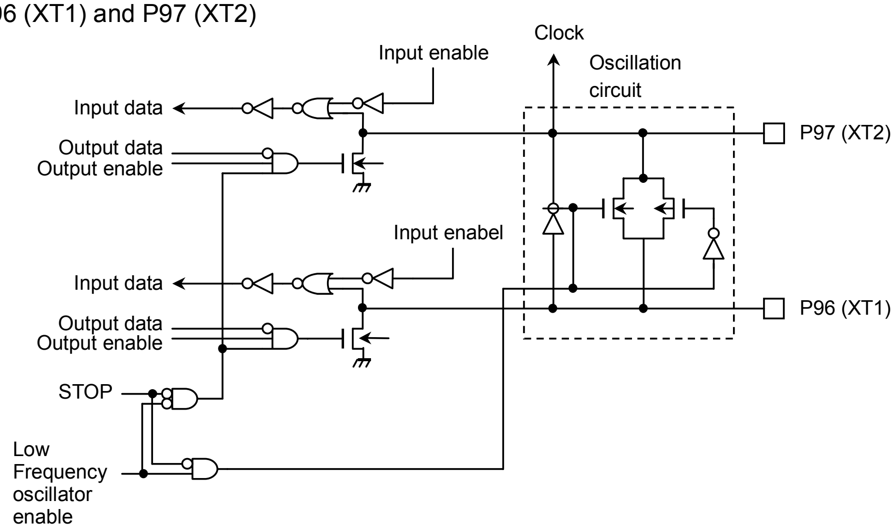
- P63 (INT0)



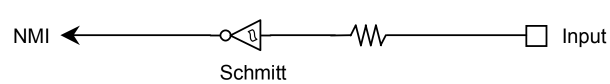
- P61 (SO/SDA), P62 (SI/SCL), P90 (TXD0) and P93 (TXD1)



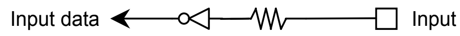
- P96 (XT1) and P97 (XT2)



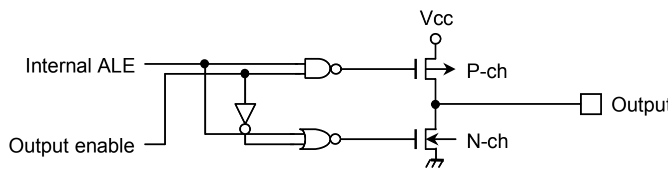
- $\overline{\text{NMI}}$



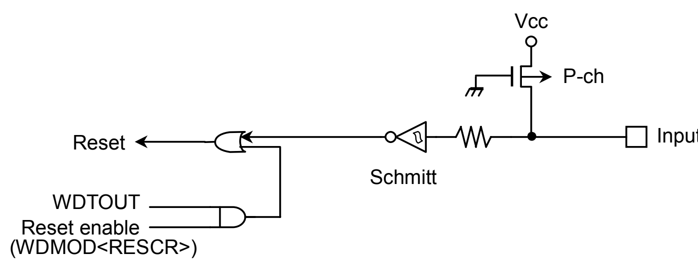
■ AM0 and AM1



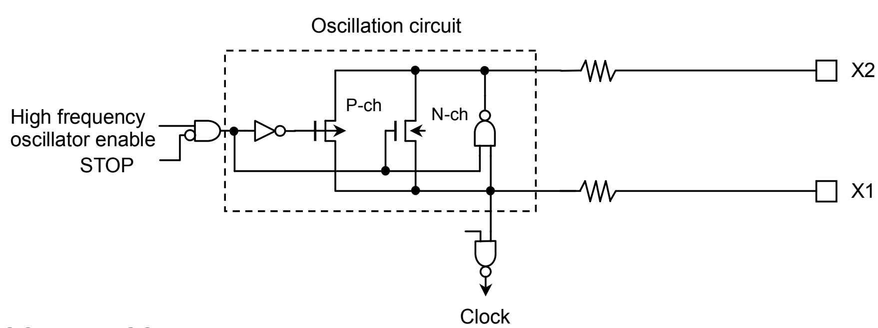
■ ALE



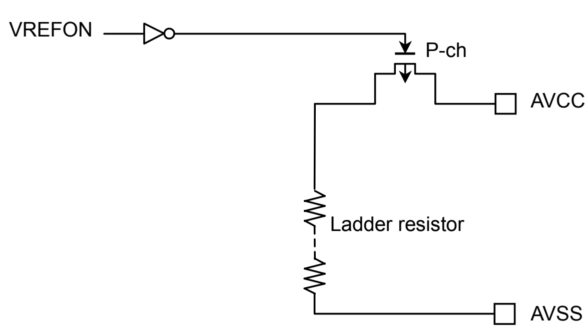
■ $\overline{\text{RESET}}$



■ X1 and X2



■ AVCC and AVSS



7. Points to Note and Restrictions

(1) Notation

1. The notation for built-in I/O registers is as follows register symbol<Bit symbol>

   Example: TA01RUN<TA0RUN> denotes bit TA0RUN of register TA01RUN.

2. Read-modify-write instructions

   An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

   Example 1: SET 3, (TA01RUN) .... Set bit 3 of TA01RUN.

   Example 2: INC 1, (100H).......................... Increment the data at 100H.

- Examples of read-modify-write instructions on the TLCS-900

Exchange instruction

```
EX      (mem), R
```

Arithmetic operations

```
ADD  (mem), R/#      ADC  (mem), R/#
SUB  (mem), R/#      SBC  (mem), R/#
INC  #3, (mem)       DEC  #3, (mem)
```

Logic operations

```
AND  (mem), R/#      OR   (mem), R/#
XOR  (mem), R/#
```

Bit manipulation operations

```
STCF #3/A, (mem)     RES  #3, (mem)
SET  #3, (mem)       CHG  #3, (mem)
TSET #3, (mem)
```

Rotate and shift operations

```
RLC  (mem)           RRC  (mem)
RL   (mem)           RR   (mem)
SLA  (mem)           SRA  (mem)
SLL  (mem)           SRL  (mem)
RLD  (mem)           RRD  (mem)
```

3. f_{OSCH}, fc, fs, f_{FPH}, f_{SYS} and one state

   The clock frequency input on ins X1 and X2 is called f_{OSCH}. TMP91CP27 have not DFM. Therefore, become fc equal f_{OSCH}. It called fs that clock frequency is inputted from XT1/XT2 pin.

   The clock selected by SYSCR1<SYSCK> is called f_{FPH}. The clock frequency give by f_{FPH} divided by 2 is called f_{SYS}.

   One cycle of f_{SYS} is referred to as one state.

(2) Points to note

a. AM0 and AM1 pins

This pin is connected to the VCC pin. Do not alter the level when the pin is active.

b. Warm-up counter

The warm-up counter operates when STOP Mode is released, even if the system is using an external oscillator. As a result a time equivalent to the warm-up time elapses between input of the release request and output of the system clock.

c. Programmable pull-up resistor

The programmable pull-up resistor can be turned ON/OFF by a program when the ports are set for use as input ports. When the ports are set for use as output ports, they cannot be turned ON/OFF by a program.

The data registers (e.g., P4 register) are used to turn the pull-up resistors ON/OFF. Consequently read-modify-write instructions are prohibited. Therefore, use Transfer instruction.

d. Watchdog timer

The watchdog timer starts operation immediately after a reset is released. When the watchdog timer is not to be used, disable it.

e. AD converter

TMP91CP27 include function that can cut or connect the string resistor between the reference pins (Share with AVCC and AVSS pins in TMP91CP27) by program. (ADMOD1 <VREFON>)

When STOP mode is used as reduce consumption power supply, disable the resistor using the program before the HALT instruction is executed.

f. RTC

If set IDLE1 mode (Operate only oscillator) and execute HALT instruction, built-in RTC is operation enable condition. If want to stop operation, stop operation by setting control register RTCCR <RTCRUN> to “0”.

g. CPU (Micro DMA)

Only the LDC cr, r and LDC r, cr instructions can be used to access the control registers in the CPU (e.g., the transfer source address register (DMASn)).

h. Undefined SFR

The value of an undefined bit in an SFR (Special function register) is undefined when read.

i. POP SR instruction

Please execute the POP SR instruction during DI condition.

j. Releasing the HALT mode by requesting an interruption

Usually, interrupts can release all halts status. However, the interrupts ($\overline{\text{NMI}}$, INT0, INTRTC) which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 5 clocks of f_{FPH}) with IDLE1 or STOP mode (IDLE2 is not applicable to this case).(In this case, an interrupt request is kept on hold internally)

If another interrupts is generated after it has shifted to HALT mode completely, halt status can be released without difficulty. The priority of this interrupt is compared with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

8. Package Dimensions

P-LQFP64-1010-0.50D

Unit: mm