Low Voltage/Low Power

CMOS 16-Bit Microcontroller

TMP93CS20F

1. Outline and Device Characteristics

The TMP93CS20 is high-speed, advanced 16-bit microcontroller. It enables low-voltage and low-power-consumption operation.

The TMP93CS20 is housed in 144-pin flat packages.

The device characteristics are as follows:

(1) Original 16-bit CPU (900/L CPU)
- TLCS-90 instruction mnemonic upward compatible
- 16-Mbyte linear address space
- General-purpose registers, register bank system
- 16-bit multiplication, 16-bit division, bit transfer and bit manipulation instructions
- Micro DMA: 4 channels (1.6 μs per 2 bytes at 20 MHz)

(2) Minimum instruction execution time: 200 ns at 20 MHz

(3) Internal RAM: 2 Kbytes

Internal ROM: 64 Kbytes

(4) LCD driver
- Boosting circuit (Reference voltage external input)
- Maximum 40 segments × 4 commons
- 1/4, 1/3, 1/2 duty, static driving selection

(5) Timer for realtime clock: 1 channel

(6) 8-bit timer: 4 channels

(7) 16-bit timer: 4 channels

(8) Serial interface: 2 channels
- UART/synchronous mode: 1 channel
- UART: 1 channel

(9) Serial bus interface: 1 channel
- I^2C bus mode/clock-synchronous 8-bit SIO mode

(10) 10-bit AD converter: 8 channels

(11) Large current drive port: 8 ports (Port A)

(12) Watchdog timer

(13) Key-on wakeup (Key input interrupt)

(14) Interrupt functions
- 9 CPU interrupts (SWI instruction, and illegal instruction)
- 24 internal interrupts
- 12 external interrupts

] 7-level priority can be set.

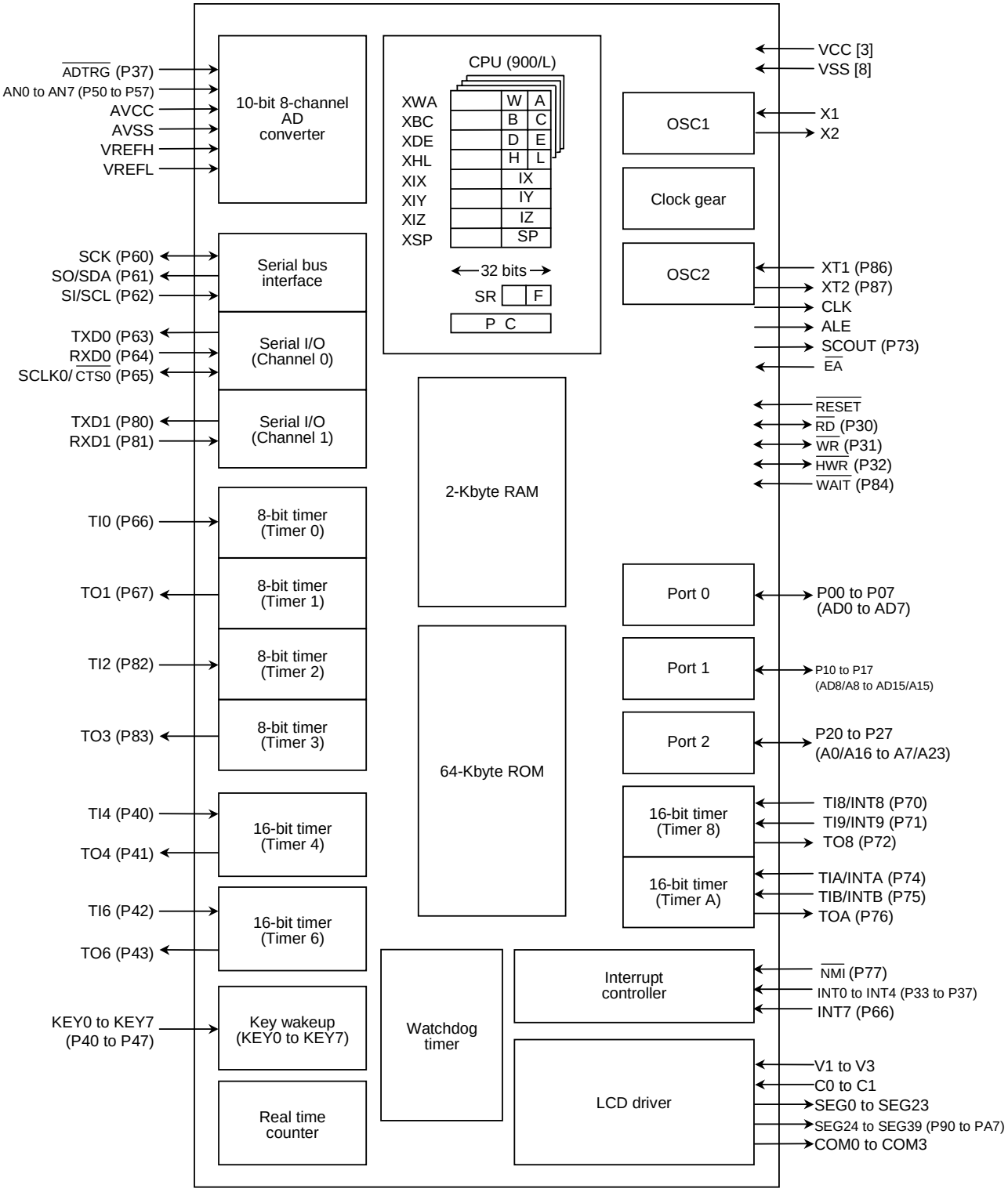
(15) I/O ports: 88 pins (including XT1, XT2)
- Large current output: 8 pins, LED can be directly driven.

(16) Standby function: 4 HALT modes (RUN, IDLE2, IDLE1, STOP)

(17) Clock gear function
- Clock gear: High-frequency clock can be changed from fc to fc/16.
- Dual clock operation

(18) Operating voltage
- Vcc = 2.7 to 5.5 V

(19) Package
- P-LQFP144-1616-0.40



Note: The item in parentheses ( ) are the initial setting after reset.

Figure 1.1 TMP93CS20 Block Diagram

2. Pin Assignment and Functions

The assignment of input and output pins for the TMP93CS20, their names and functions are described below.

2.1 Pin Assignment

Figure 2.1.1 shows pin assignment of the TMP93CS20F.

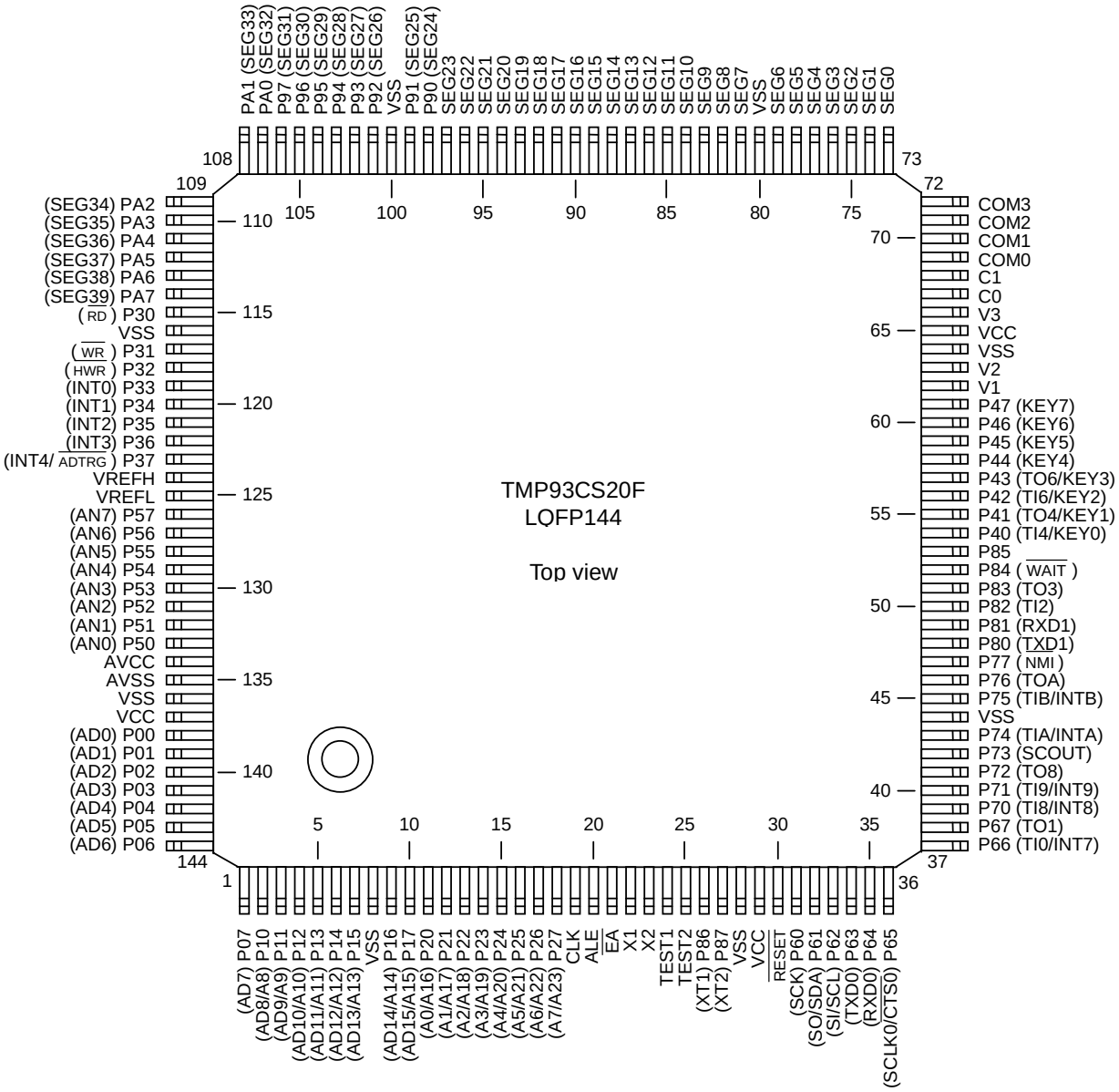


Figure 2.1.1 Pin Assignment (144-pin LQFP)

2.2 Pin Names and Functions

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

Table 2.2.1 Pin Names and Function (1/3)

Pin Names	Number of Pins	I/O	Functions
P00 to P07	8	I/O	Port 0: I/O port that allows I/O to be selected at the bit level.
AD0 to AD7		I/O	Address and data (Lower): Bits 0 to 7 for address and data bus.
P10 to P17	8	I/O	Port 1: I/O port that allows I/O to be selected at the bit level.
AD8 to AD15		I/O	Address and data (Upper): Bits 8 to 15 for address and data bus.
A8 to A15		Output	Address: Bits 8 to 15 for address bus.
P20 to P27	8	I/O	Port 2: I/O port that allows I/O to be selected at the bit level. (with pull-up resistor.)
A0 to A7		Output	Address: Bits 0 to 7 for address bus.
A16 to A23		Output	Address: Bits 16 to 23 for address bus.
P30	1	Output	Port 30: Output port.
$\overline{RD}$		Output	Read: Strobe signal for reading external memory. (Read when reading internal memory at P3<P30> = 0, P3FC<P30F> = 1.)
P31	1	Output	Port 31: Output port.
$\overline{WR}$		Output	Write: Strobe signal for writing data on pins AD0 to AD7.
P32	1	I/O	Port 32: I/O port (with pull-up resistor).
$\overline{HWR}$		Output	High write: Strobe signal for writing data on pins AD8 to AD15.
P33	1	I/O	Port 33: I/O port (with pull-up resistor).
INT0		Input	Interrupt request pin 0: Interrupt request pin with programmable level/rising/falling edge.
P34	1	I/O	Port 34: I/O port (with pull-up resistor).
INT1		Input	Interrupt request pin 1: Interrupt request pin with programmable rising/falling edge.
P35	1	I/O	Port 35: I/O port (with pull-up resistor).
INT2		Input	Interrupt request pin 2: Interrupt request pin with programmable rising/falling edge.
P36	1	I/O	Port 36: I/O port (with pull-up resistor).
INT3		Input	Interrupt request pin 3: Interrupt request pin with programmable rising/falling edge.
P37	1	I/O	Port 37: I/O port (with pull-up resistor).
INT4		Input	Interrupt request pin 4: Interrupt request pin with programmable rising/falling edge.
$\overline{ADTRG}$		Input	AD converter external start trigger input.
P40	1	I/O	Port 40: I/O port (with pull-up resistor).
TI4		Input	Timer input 4: 16-bit timer 4 input.
KEY0		Input	Key input 0: Key-on wakeup pin 0.
P41	1	I/O	Port 41: I/O port (with pull-up resistor).
TO4		Output	Timer output 4: 16-bit timer 4 output.
KEY1		Input	Key input 1: Key-on wakeup pin 1.
P42	1	I/O	Port 42: I/O port (with pull-up resistor).
TI6		Input	Timer input 6: 16-bit timer 6 input.
KEY2		Input	Key input 2: Key-on wakeup pin 2.
P43	1	I/O	Port 43: I/O port (with pull-up resistor).
TO6		Output	Timer output 6: 16-bit timer 6 output.
KEY3		Input	Key input 3: Key-on wakeup pin 3.
P44 to P47	4	I/O	Port 44 to 47: I/O port (with pull-up resistor).
KEY4 to KEY7		Input	Key input 4 to 7: Key-on wakeup pin 4 to 7.

Table 2.2.1 Pin Names and Function (2/3)

Pin Name	Number of Pins	I/O	Functions
P50 to P57	8	Input	Port 50 to 57: Pin used to input port.
AN0 to AN7		Input	Analog input 0 to 7.
P60	1	I/O	Port 60: I/O port.
SCK		I/O	Clock I/O pin in SIO mode of the serial bus interface.
P61	1	I/O	Port 61: I/O port (with programmable open drain).
SO		Output	Data send channel in SIO mode of the serial bus interface.
SDA		I/O	Data I/O pin in I^2C bus mode of the serial bus interface.
P62	1	I/O	Port 62: I/O port (with programmable open drain).
SI		Input	Data receive channel in SIO mode of the serial bus interface.
SCL		I/O	Clock I/O pin in I^2C bus mode of the serial bus interface.
P63	1	I/O	Port 63: I/O port (with programmable open drain).
TXD0		Output	Serial send data 0.
P64	1	I/O	Port 64: I/O port.
RXD0		Input	Serial receive data 0.
P65	1	I/O	Port 65: I/O port.
SCLK0		I/O	Serial clock I/O 0.
$\overline{CTS0}$		Input	Serial data send enable 0 (Clear to send).
P66	1	I/O	Port 66: I/O port.
TI0		Input	Timer input 0: 8-bit timer 0 input.
INT7		Input	Interrupt request pin 7: Interrupt request pin with programmable rising/falling edge.
P67	1	I/O	Port 67: I/O port.
TO1		Output	Timer output1: 8-bit timer 0 or timer 1 output.
P70	1	I/O	Port 70: I/O port (with pull-up resistor).
TI8		Input	Timer input 8: 16-bit timer 8 input.
INT8		Input	Interrupt request pin 8: Interrupt request pin with programmable rising/falling edge.
P71	1	I/O	Port 71: I/O port (with pull-up resistor).
TI9		Input	Timer input 9: 16-bit timer 8 input.
INT9		Input	Interrupt request pin 9: Interrupt request pin with rising edge.
P72	1	I/O	Port 72: I/O port (with pull-up resistor).
TO8		Output	Timer output 8: 16-bit timer 8 output.
P73	1	I/O	Port 73: I/O port (with pull-up resistor).
SCOUT		Output	System clock output: System clock or double system clock output to be synchronized with the external circuit.
P74	1	I/O	Port 74: I/O port (with pull-up resistor).
TIA		Input	Timer input A: 16-bit timer A input.
INTA		Input	Interrupt request pin A: Interrupt request pin with programmable rising/falling edge.
P75	1	I/O	Port 75: I/O port (with pull-up resistor).
TIB		Input	Timer input B: 16-bit timer B input.
INTB		Input	Interrupt request pin B: Interrupt request pin with rising edge.
P76	1	I/O	Port 76: I/O port (with pull-up resistor).
TOA		Output	Timer output A: 16-bit timer A output.
P77	1	I/O	Port 77: I/O port (with pull-up resistor).
$\overline{NMI}$		Input	Non-maskable Interrupt request pin: Interrupt request pin with programmable falling edge or both edges.

Table 2.2.1 Pin Names and Function (3/3)

Pin Name	Number of Pins	I/O	Functions
P80 TXD1	1	I/O Output	Port 80: I/O port (with programmable open drain). Serial send data 1.
P81 RXD1	1	I/O Input	Port 81: I/O port (with programmable open drain). Serial receive data 1.
P82 TI2	1	I/O Input	Port 82: I/O port (with programmable open drain). Timer input 2: 8-bit timer 2 input pin.
P83 TO3	1	I/O Output	Port 83: I/O port (with programmable open drain). Timer output 3: 8-bit timer 2, 3 output pin.
P84 $\overline{\text{WAIT}}$	1	I/O Input	Port 84: I/O port (with programmable open drain). Wait: Pin used to request CPU bus wait.
P85	1	I/O	Port 85: I/O port (with programmable open drain).
P86 XT1	1	I/O Input	Port 86: I/O port (Open drain). Low-frequency oscillator connecting pin.
P87 XT2	1	I/O Output	Port 87: I/O port (Open drain). Low-frequency oscillator connecting pin.
P90 to P97 SEG24 to SEG31	8	Output Output	Port 90 to 97: Output port (Open drain). Segment data output pin.
PA0 to PA7 SEG32 to SEG39	8	Output Output	Port A0 to A7: Output port, large current port (Open drain). LCD segment output pin.
SEG0 to SEG23	24	Output	LCD segment output.
COM0 to COM3	4	Output	LCD common output.
AVCC	1	Power supply	Power supply pin for AD converter.
AVSS	1	Power supply	GND pin for AD converter (0 V).
VREFH	1	Input	Pin for reference voltage input to AD converter (H).
VREFL	1	Input	Pin for reference voltage input to AD converter (L).
X1	1	Input	Oscillator connecting pin.
X2	1	Output	Oscillator connecting pin.
$\overline{\text{RESET}}$	1	Input	Reset: Initializes LSI.
ALE	1	Output	Address latch enable. Can be disabled for reducing noise.
CLK	1	Output	Clock output: Outputs “external input clock X1 ÷ 4” clock. Pulled-up during reset. Can be disabled for reducing noise.
$\overline{\text{EA}}$	1	Input	The VCC pin should be connected.
VCC	3	Power supply	Power supply pin (All Vcc pins should be connected with the power supply pin).
VSS	8	Power supply	GND pin (0 V) (All Vss pins should be connected with GND (0 V)).
TEST1 TEST2	2	Output Input	TEST1 should be connected with TEST2 pin. Do not connect to any other pins.
C0, C1, V1 to V3	5	LCD pin	LCD drive boosting pin. A condenser should be connected between C0 and C1, V1, V2, V3 and GND.

Note: All pins that have built-in pull-up resistors can be disconnected from the built-in pull-up resistor by software.

3. Operation

This section describes in blocks the functions and basic operations of TMP93CS20 device. Please also refer to section 7., "Points to Note and Restrictions", which describes some points requiring careful attention.

3.1 CPU

TMP93CS20 devices have a built-in high-performance 16-bit CPU (900/L CPU). (For basics of the CPU operation, see the information on the TLCS-900/L CPU in the previous chapter).

This section describes some CPU functions unique to the TMP93CS20, that are not described in the previous chapter, entitled TLCS-900/L CPU.

3.1.1 Reset

When resetting the TMP93CS20 microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then set the $\overline{\text{RESET}}$ input to low level at least for 10 system clocks (16 μs at 20 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 a reset signal is accepted, the CPU sets itself as follows:

- The program counter (PC) is set according to the reset vector that is stored from FFFF00H to FFFF02H.

  PC<7:0> ← Data in location FFFF00H

  PC<15:8> ← Data in location FFFF01H

  PC<23:16> ← Data in location FFFF02H

- The stack pointer (XSP) for system mode is set to 100H.
- The <IFF2:0> bits of the status register SR are set to 111. (Sets mask register to interrupt level 7.)
- The <MAX> bit of SR is set to 1. (Sets to maximum mode. See previous chapter.)
- The <RFP2:0> bits of SR are set to 000. (Clears register banks to 0.)

When the reset is released, instruction execution starts from PC (the reset vector). The reset makes no changes in any CPU internal registers other than those specifically mentioned above.

When a reset is received, signal and data processing for built-in I/Os, ports, and other pins is affected as follows:

- Initializes built-in I/O registers as per specifications.
- Sets port pins (including pins also used as built-in I/Os) to general-purpose input/output port mode.
- Pulls up the CLK pin to 1.
- Sets the ALE pin to high impedance (High-Z).

Note 1: Resetting makes no change in any register in the CPU except the program counter (PC), status register (SR), and stack pointer (XSP), nor in the data in the internal RAM.

Note 2: The CLK pin is pulled up to "H" level during reset. When the voltage is put down externally, there is possible to cause malfunctions.

Figure 3.1.1 shows the reset timing chart of TMP93CS20.

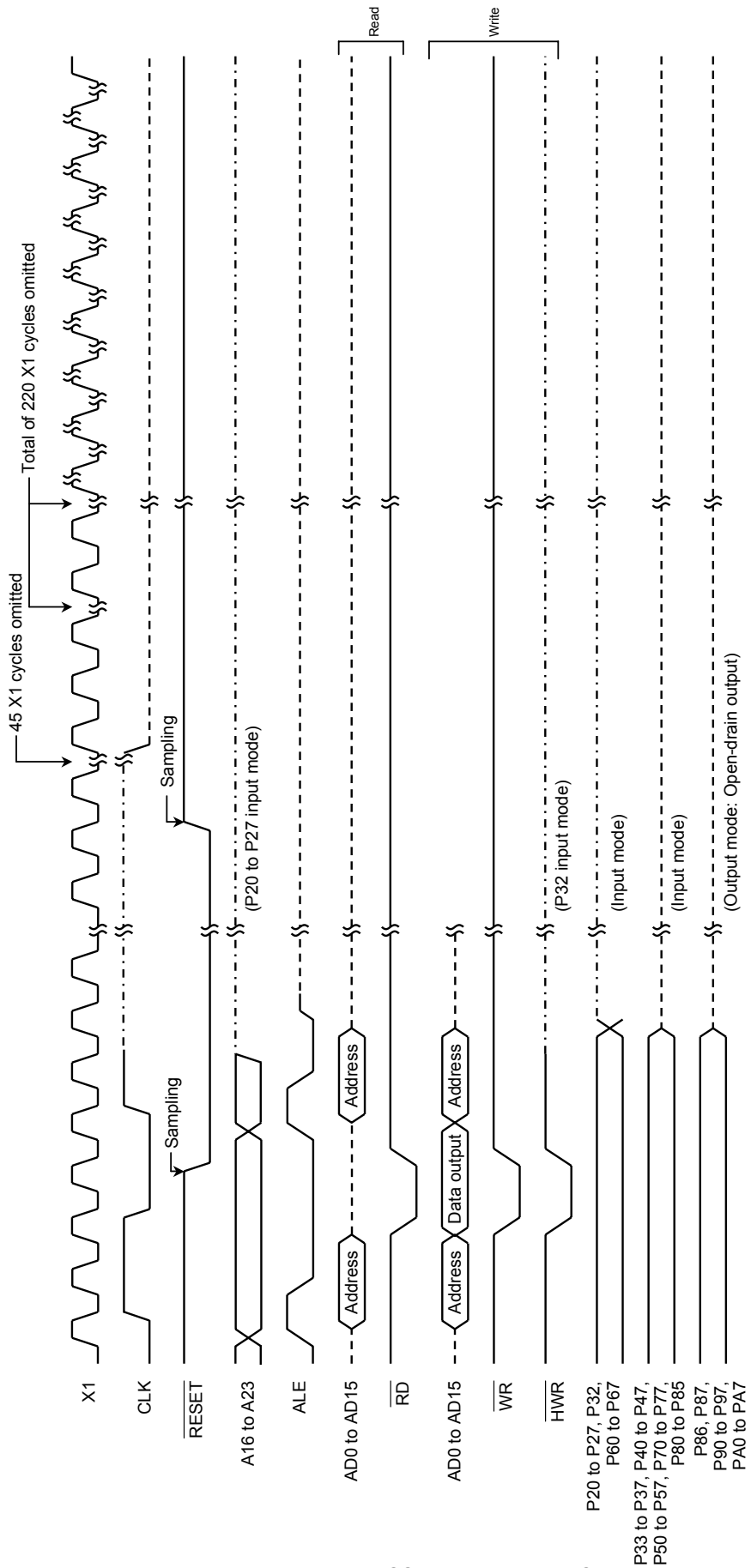


Figure 3.1.1 TMP93CS20 Reset Timing Chart

3.2 Memory Map

Figure 3.2.1 is a memory map of the TMP93CS20.

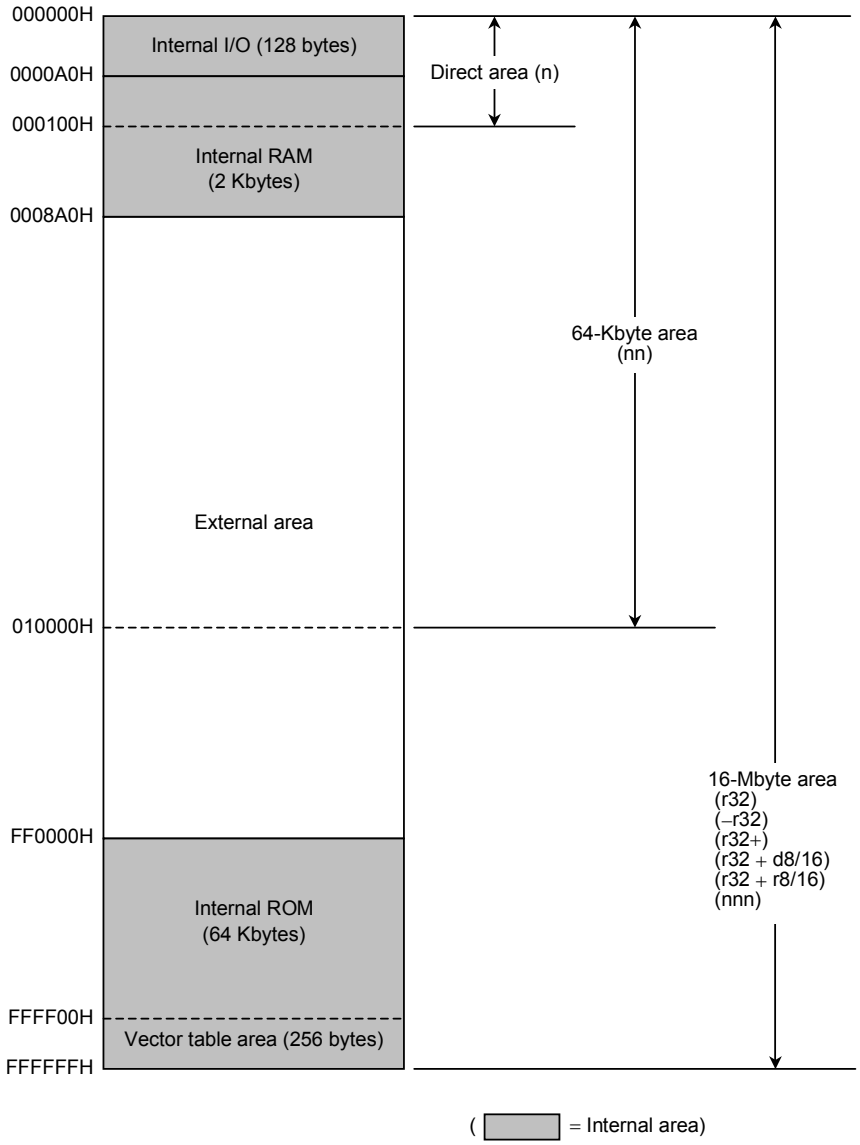


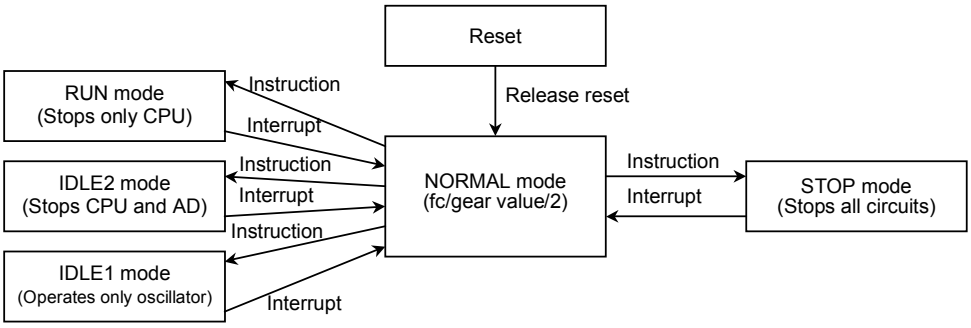
Figure 3.2.1 Memory Map

3.3 Dual Clock, Standby Function

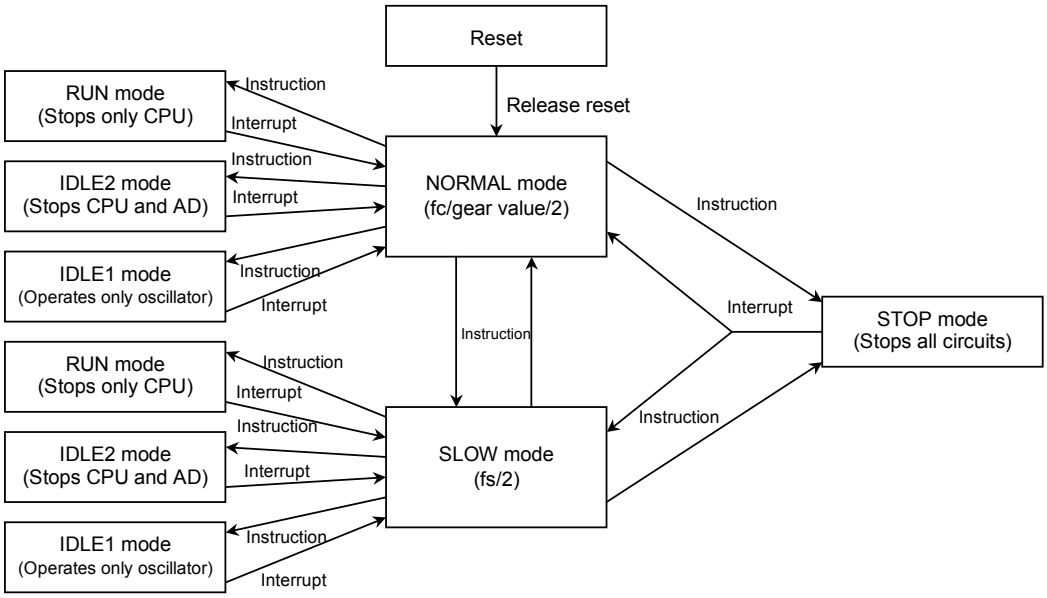
Dual clock, standby control circuits are comprised of a system clock controller, prescaler clock controller, internal clock pin output function and standby controller.

The oscillator operating modes are classified as either (a) single clock mode (Using only the X1 and X2 pins), or (b) dual clock mode (Using the X1, X2, XT1, and XT2 pins).

Figure 3.3.1 shows state diagrams for the two clock modes. Figure 3.3.2 shows the corresponding block diagram, Figure 3.3.3 displays functions of the I/O registers and Table 3.3.1 lists correspondences between alternative states of the system clock and those of the CPU, oscillator and internal I/O components.



(a) Single clock mode state diagram



(b) Dual clock mode state diagram

Figure 3.3.1 State Diagrams

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

Table 3.3.1 Relations between System Clock States and other Internal Operations

<table>
<tr><th colspan="2" rowspan="2">Operating Mode</th><th colspan="2">Oscillator</th><th rowspan="2">CPU</th><th rowspan="2">Internal I/O</th><th rowspan="2">System clock
f_{SYS}</th></tr>
<tr><th>High Frequency (fc)</th><th>Low Frequency (fs)</th></tr>
<tr><td rowspan="7">Single Clock</td><td>Reset</td><td rowspan="5">Oscillation</td><td rowspan="6">Stop</td><td>Reset</td><td>Reset</td><td>fc/32</td></tr>
<tr><td>NORMAL</td><td>Operate</td><td rowspan="2">Operate</td><td rowspan="4">Programmable
(fc/2, fc/4, fc/8,
fc/16, fc/32)</td></tr>
<tr><td>RUN</td><td rowspan="3">Stop</td></tr>
<tr><td>IDLE2</td><td>Stop only AD</td></tr>
<tr><td>IDLE1</td><td>*</td></tr>
<tr><td>STOP</td><td>Stop</td><td>Stop</td><td>Stop</td></tr>
<tr><td rowspan="7">Dual Clock</td><td>Reset</td><td rowspan="2">Oscillation</td><td>Stop</td><td>Reset</td><td>Reset</td><td>fc/32</td></tr>
<tr><td>NORMAL</td><td>Programmable</td><td rowspan="2">Operate</td><td rowspan="2">Operate</td><td>Programmable
(fc/2, fc/4, fc/8, fc/16, fc/32)</td></tr>
<tr><td>SLOW</td><td>Programmable</td><td>Oscillation</td><td>fs/2</td></tr>
<tr><td>RUN</td><td colspan="2" rowspan="3">Oscillator being used as system
Clock: Oscillation
Other oscillator: Programmable</td><td rowspan="4">Stop</td><td rowspan="3">Stop only AD</td><td rowspan="3">Programmable
(fc/2, fc/4, fc/8,
fc/16, fc/32, fs/2)</td></tr>
<tr><td>IDLE2</td></tr>
<tr><td>IDLE1</td><td>*</td></tr>
<tr><td>STOP</td><td colspan="2">Stop</td><td>Stop</td><td>Stop</td></tr>
</table>

* Only timer for realtime clock is operating.

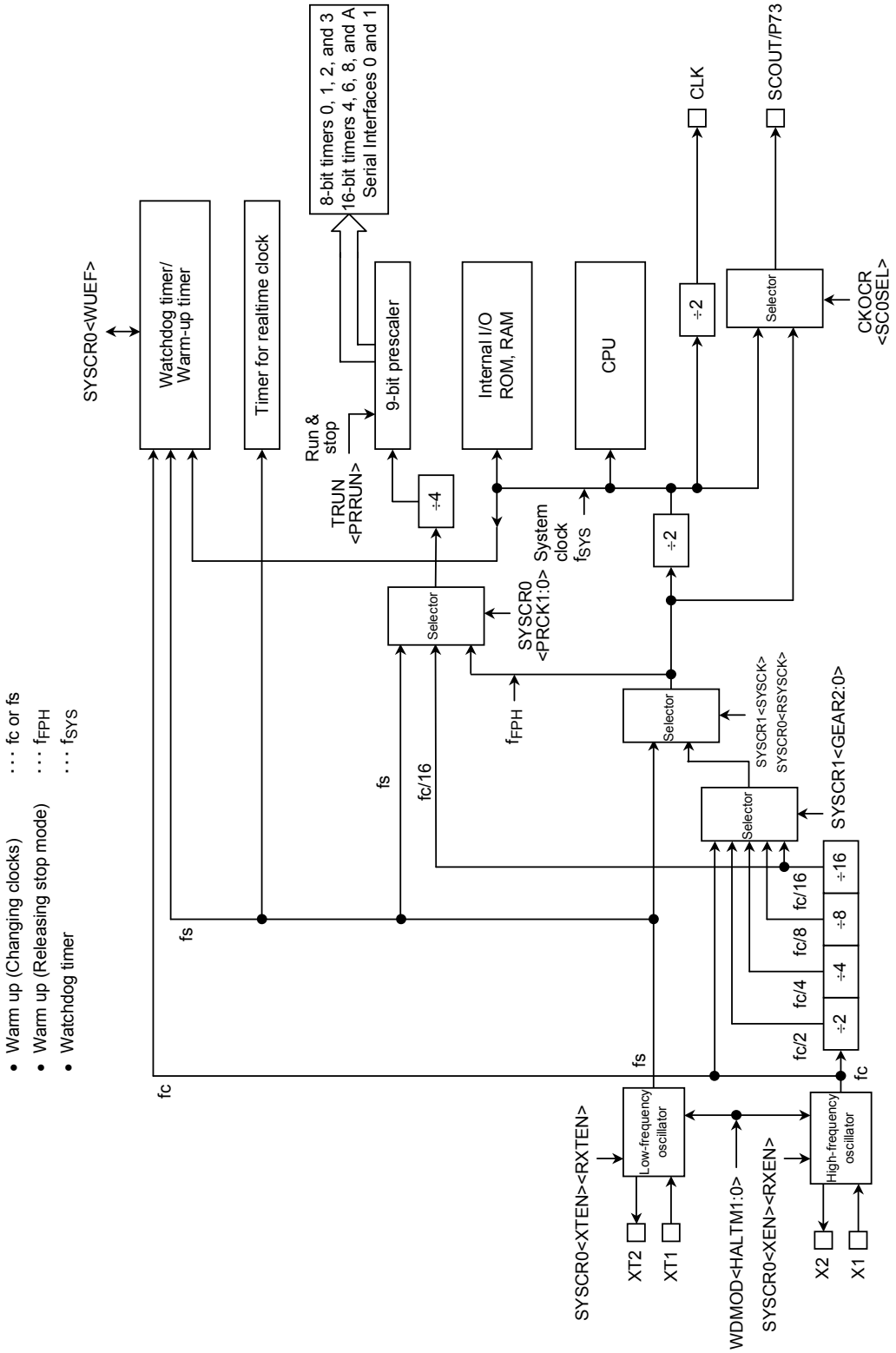


Figure 3.3.2 Block Diagram of Dual Clock and Standby Circuits

		7	6	5	4	3	2	1	0
SYSCR0 (006EH)	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 released Stop mode 0: Stop 1: Oscillation	Low-frequency oscillator (fs) after released Stop mode 0: Stop 1: Oscillation	Select clock after STOP mode is released 0: fc 1: fs	Warm-up timer (Write) 0: Don't care 1: Start timer (Read) 0: Warm-up complete 1: Continue warm up	Select prescaler clock 00: f_{FPH} 01: fs 10: fc/16 11: (Reserved)	
SYSCR1 (006FH)	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)		
CKOCR (006DH)	Bit symbol					SCOSEL	SCOEN	ALEEN	CLKEN
	Read/Write					R/W			
	After reset					0	0	0	0
	Function					SCOUT select 0: f_{FPH} 1: f_{SYS}	SCOUT output control 0: I/O port 1: SCOUT output	ALE pin output control 0: High-Z output 1: ALE output	CLK pin output control 0: High-Z output 1: CLK output
WDMOD (005CH)	Bit symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
	Read/Write	R/W							
	After reset	1	0	0	0	0	0	0	0
	Function	WDT control 0: Disable 1: Enable	WDT detection time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$		Warm-up timer 0: 2^{14}/ frequency input 1: 2^{16}/ frequency input	HALT mode 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		0: Don't care 1: Connects WDT output to RESET pin internally.	Control pin in STOP mode 0: I/O off 1: Maintain the state before HALT mode

Note 1: SYSCR1<Bit7:4> are always 1.

Note 2: The CLK pin is internally pulled up during reset, regardless of the product types.

Note 3: Programming 0 to SYSCR1<SYSCK> enables the high-frequency oscillator regardless of the value of SYSCR0<XEN>.
Additionally, programming 1 to SYSCR1<SYSCK> enables the low-frequency oscillator regardless of the value of SYSCR0<XTEN>.

Figure 3.3.3 I/O Register about Dual Clock, Standby

3.3.1 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). The register SYSCR1<SYSCK> changes the system clock to either fc or fs, SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling each oscillator, and SYSCR1<GEAR2:0> changes 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 system clock (f_{SYS}) is set to fc/32 (fc/16 × 1/2) by the setting of <XEN> = 1, <XTEN> = 0, <SYSCK> = 0, and <GEAR2:0> = 100 upon resetting.

For example, f_{SYS} is set to 0.625 MHz by resetting in the case where the 20 MHz oscillator is connected to the X1 and X2 pins.

The high-frequency (fc) and low-frequency (fs) clock signals can be easily obtained by connecting a resonator to the X1 and X2, XT1 and XT2 pins, respectively. Clock input from an external oscillator is also possible.

The XT1 and XT2 pins can also function as ports 86 and 87. Therefore in the case of single clock mode, the XT1 and XT2 pins can be used as I/O port pins.

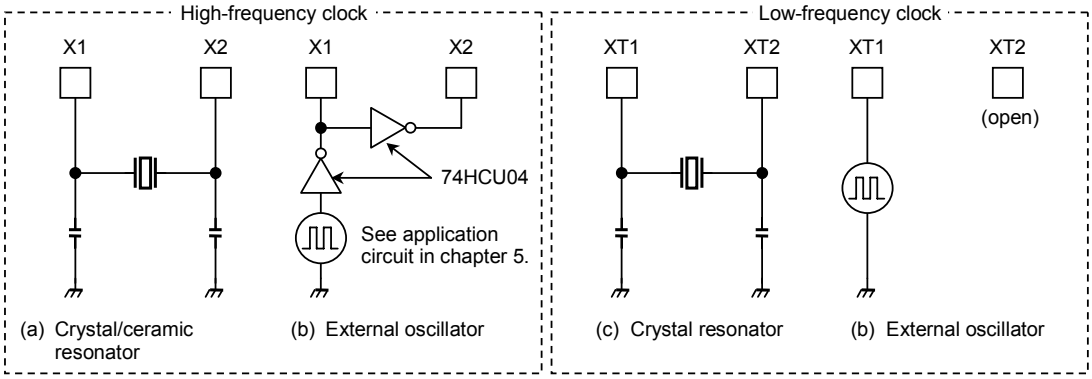


Figure 3.3.4 Examples of Resonator Connection

Note 1: Note on using the low frequency oscillation circuit.
In connecting the low frequency resonator to ports 86 and 87, it is necessary to make the following settings to reduce the power consumption.
(Connecting with resonators) P8CR<P86C, P87C> = 11, P8<P86:87> = 00
(Connecting with oscillators) P8CR<P86C, P87C> = 11, P8<P86:87> = 10

Note 2: Accurate adjustment of the oscillation frequency.
The CLK pin output at 1/2 the system clock frequency (f_{SYS}/2) is used to monitor the oscillation clock. With a system requiring adjustment of the oscillation frequency, an adjusting program must be written.

(1) Switching between normal and slow mode

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

The warm-up time can be selected by WDMOD<WARM>.

This starting and stopping of the warm-up timer are performed by programming as in the following examples 1 and 2.

Note 1: The warm-up timer is also used as a watchdog timer, so when it is to be used as a warm-up timer, the watchdog timer function must be disabled.

Note 2: In the case of using an oscillator (not resonator) with stable oscillation, a warm-up timer is not needed.

Note 3: The warm-up timer is operated by an oscillation clock. Therefore there is an error in, warm-up time.

Table 3.3.2 Warm-up Time

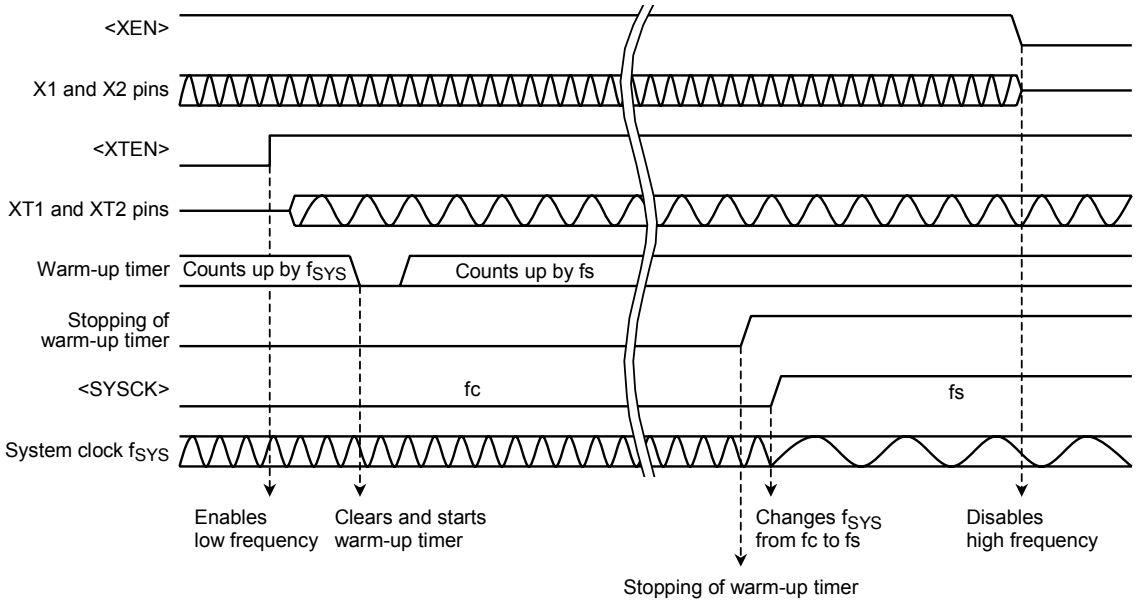
Warm-up Time WDMOD<WARM>	Change to Normal	Change to Slow
0 (2^{14}/frequency)	0.8192 ms	500 ms
1 (2^{16}/frequency)	3.2768 ms	2000 ms

at fc = 20 MHz, fs = 32.768 kHz

Clock setting example 1:

Changing from high frequency (fc) to low frequency (fs).

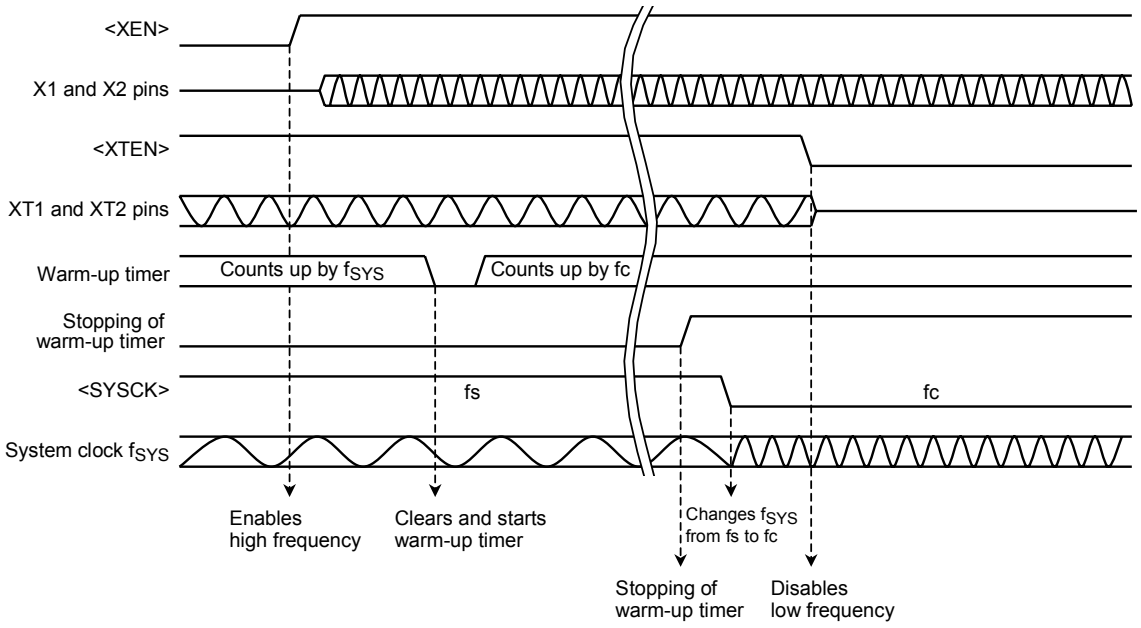
```
SYSCR0   EQU   006EH
SYSCR1   EQU   006FH
WDCR     EQU   005DH
WDMOD    EQU   005CH
         RES   7, (WDMOD)      ;
         LD    (WDCR), B1H     ;   Disables watchdog timer.
         SET   4, (WDMOD)      ;   Sets warm-up time to 2^16/fs.
         SET   6, (SYSCR0)     ;   Enables low-frequency oscillation.
         SET   2, (SYSCR0)     ;   Clears and starts warm-up timer.
WUP:     BIT   2, (SYSCR0)     ;
         JR    NZ, WUP         ;   Detects stopping of the warm-up timer.
         SET   3, (SYSCR1)     ;   Changes fSYS from fc to fs.
         RES   7, (SYSCR0)     ;   Disables high-frequency oscillation.
         SET   7, (WDMOD)      ;   Enables watchdog timer.
```



Clock setting example 2:

Changing from low frequency (fs) to high frequency (fc).

```
SYSCR0   EQU   006EH
SYSCR1   EQU   006FH
WDCR     EQU   005DH
WDMOD    EQU   005CH
         RES   7, (WDMOD)      ;
         LD    (WDCR), B1H     ;   Disables watchdog timer.
         RES   4, (WDMOD)      ;   Sets warm-up time to 2^14/fc.
         SET   7, (SYSCR0)     ;   Enables high-frequency oscillation (fc).
         SET   2, (SYSCR0)     ;   Clears and starts warm-up timer.
WUP:     BIT   2, (SYSCR0)     ;
         JR    NZ, WUP         ;   Detects stopping of the warm-up timer.
         RES   3, (SYSCR1)     ;   Changes fSYS from fs to fc.
         RES   6, (SYSCR0)     ;   Disables low-frequency oscillation.
         SET   7, (WDMOD)      ;   Enables watchdog timer.
```



(2) Clock gear controller

When the high-frequency clock fc is selected at SYSCR1<SYSCK> = 0, the clock gear select register SYSCR1<GEAR2:0> sets f_{FPH} to either fc, fc/2, fc/4, fc/8, or fc/16. Switching f_{FPH} with the clock gear reduces the power consumption.

Clock setting example 3:

Changing gear value of the high-frequency clock

```
SYSCR1  EQU  006FH
        LD   (SYSCR1), XXXX0000B  ;  Changes fSYS to fc/2.
        LD   (SYSCR1), XXXX0100B  ;  Changes fSYS to fc/32.
        X: Don't care
```

(High-frequency clock gear changing)

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

There is a possibility that the instruction immediately following the clock-gear-changing instruction will be executed by the clock gear before executing its gear change. To ensure that the instruction immediately following the clock-gear-changing instruction will only be executed by the clock gear after changing its gear ratio, input a dummy instruction (an instruction to execute a write cycle) as follows.

(Example)

Instruction to be executed by the clock gear after changing its gear ratio.

```
SYSCR1  EQU  00E6H
        LD   (SYSCR1), XXXX0001B  ;  Changes fSYS to fc/4.
        LD   (DUMMY), 00H         ;  Dummy instruction.
             Instruction to be executed by the clock gear after changing.
        X: Don't care
```

3.3.2 Prescaler Clock Controller

The 9-bit prescaler provides a clock signal to the 8-bit timers 0 to 3, 16-bit timer 4, timer 6, timer 8, and timer A, and serial interface 0 and serial interface 1.

The clock input to this prescaler is a clock signal which is selected as either f_{FPH}, fc/16, or fs according to the value in the SYSCR0<PRCK1:0> register.

The <PRCK1:0> register is initialized to 00 by resetting.

When the IDLE1 mode (Operating only the oscillator) is being used, set TRUN<PRRUN> to 0 to reduce the power consumption before a HALT instruction is executed.

3.3.3 Internal Clock Pin Output Function

(1) P73/SCOUT pin

The P73/SCOUT pin outputs the internal clock signals f_{FPH} or f_{SYS}.

One bit in the port 7 control register P7CR<P73C>, and two bits in the clock output control register CKOCR<SCOEN and SCOSEL> specify the clock and the pins. The P73/SCOUT pin is assigned as the input port in resetting.

Table 3.3.3 shows states of the SCOUT pin in the alternative operation modes which it can assume on condition that the P73/SCOUT pin is specified as SCOUT output.

Table 3.3.3 SCOUT Pin States in Alternative Operation Modes

Operation mode / Output clock	NORMAL, SLOW	HALT Mode	
		RUN, IDLE2, IDLE1	STOP
f_{FPH}	Outputs f_{FPH} clock.		Fixed to "0" or "1".
f_{SYS}	Outputs f_{SYS} clock.		

(2) CLK pin

The CLK pin outputs the internal clock signal f_{SYS} divided by 2.

The type of output is determined by one bit in the clock output control register, CKOCR<CLKEN>. Writing 1 sets the clock output, and writing 0 sets the CLK pin to high impedance. CKOCR<CLKEN> is set to 0 upon resetting.

During resetting, the CLK pin is internally pulled up regardless of the value of the <CLKEN> register. (See "TMP93CS20 Reset Timing Chart" in Figure 3.1.1.)

Table 3.3.4 States of the <CLKEN> Register, and CLK Pin Operation after Reset

Type No.	CKOCR<CLKEN>	CLK pin operation
TMP93CS20	0	High impedance

Note: To set <CLKEN> = 0 and set CLK pin to high impedance, pull up externally to prevent through current which follows to the input buffer of CLK pin.

3.3.4 Standby Controller

(1) HALT mode

When the HALT instruction is executed, the operating mode changes to RUN, IDLE2, IDLE1, or STOP mode depending on the contents of the HALT mode setting register WDMOD<HALTM1:0>. Figure 3.3.5 shows the alternative states of the watchdog timer mode registers.

Watchdog Timer Mode Register

		7	6	5	4	3	2	1	0
WDMOD (005CH)	Bit symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
	Read/Write	R/W							
	After reset	1	0	0	0	0	0	0	0
	Function	Watchdog timer control 0: Disable 1: Enable	Watchdog timer detect time selection 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$		Warm-up timer 0: 2^{14}/clock frequency selection 1: 2^{16}/clock frequency selection	HALT mode selection 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		Runaway detection internal reset control 1: Executes internal reset by runaway detection	STOP mode pin control 1: Drive pins in STOP mode

Pin state control in STOP mode

0	I/O off
1	Retains the state before halt

HALT mode setting

00	RUN mode (only CPU stops)
01	STOP mode (All circuits stop)
10	IDLE1 mode (only clock timer operating)
11	IDLE2 mode (Partial I/O operation)

Warm-up time selection at returning from the STOP mode (See Table 3.3.7)

0	$2/^{14}$ select clock frequency
1	$2/^{16}$ select clock frequency

Figure 3.3.5 Watchdog Timer Mode Register

The features of the RUN, IDLE2, IDLE1, and STOP modes are as follows.

1. RUN: Only the CPU halts; power consumption remains unchanged.
2. IDLE2: The built-in oscillator and the specified I/O operates.
   The power consumption is reduced to 1/2 than that during NORMAL operation.
3. IDLE1: Only the built-in oscillator and the clock timer operate, while all other built-in circuits stop. Consumption is reduced to 1/5 or less than that during NORMAL operation.
4. STOP: All internal circuits including the built-in oscillator stop. This greatly reduces power consumption.

The operations in the halt state are described in Table 3.3.5.

Table 3.3.5 I/O Operation during HALT Mode

HALT mode		RUN	IDLE2	IDLE1	STOP
WDMOD<HALTM1:0>		00	11	10	01
Block	CPU	Halt			
	I/O Port	Maintain the state when the HALT instruction was executed.			See Table 3.3.8
	8-bit timer	Operate		Stop	
	16-bit timer				
	Timer for realtime clock				
	Serial channel				
	Serial bus interface controller				
	AD converter				
	Watchdog timer				
	Interrupt controller				

(2) How to release the HALT mode

These halt states can be released by resetting or by requesting an interrupt. The halt release sources are determined by the combinations between the states of the interrupt mask register <IFF2:0> and the HALT modes. The details for releasing the halt status are shown in Table 3.3.6.

- Release by requesting an interrupt

  This method of releasing operation from the HALT mode depends on the interrupt-enabled status being in force. When the interrupt request level set before executing the HALT instruction exceeds the value of the interrupt mask register, the interrupt due to that source is processed after releasing the HALT mode, and then the CPU starts executing the next 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, release of the HALT mode is not executed. (In non-maskable interrupts, interrupt processing is preformed after releasing the HALT mode regardless of the value of the mask register.)

  INT0 interrupts are a special case in which release of the HALT mode is executed even if the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register. In this case interrupt processing is not performed, and the CPU starts executing the next instruction that follows the HALT instruction, but the interrupt request flag is held at 1.

  Note: Usually, interrupts can release all halt status. However, the interrupts ($\overline{NMI}$, INT0 to INT4, INTKEY, 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 3 clocks of f_{FPH}) with IDLE1 or STOP mode (RUN and IDLE2 are 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.

- Release by resetting

  Resetting releases all halt status settings.

  When the STOP mode is released by reset, it is necessary to allow enough

resetting time (3ms or more) for the operation of the oscillator to stabilize.

When the HALT mode is released by resetting, the internal RAM data maintains the state it was in before the HALT instruction was executed.

However the other setting contents are initialized. (Release of the HALT mode due to interrupts maintains all setting contents in their states before the HALT instruction was executed.)

Table 3.3.6 Halt Release Sources and Halt Release Operations

Interrupt Receiving Status			Interrupt Enabled (Interrupt level) ≥ (Interrupt mask)				Interrupt Disabled (Interrupt level) < (Interrupt mask)			
HALT mode			RUN	IDLE2	IDLE1	STOP	RUN	IDLE2	IDLE1	STOP
Halt release source	Interrupt	NMI	♦	♦	♦	♦*1	–	–	–	–
		INTWDT	♦	×	×	×	–	–	–	–
		INT0 to 4, INTKEY	♦	♦	♦	♦*1	○	○	○	○*1
		INTRTC	♦	♦	♦	×	○	○	○	×
		INT7 to B	♦	♦	×	×	×	×	×	×
		INTT0 to 3	♦	♦	×	×	×	×	×	×
		INTTR4 to B	♦	♦	×	×	×	×	×	×
		INTTO4, 6, 8, A	♦	♦	×	×	×	×	×	×
		INTRX0, TX0	♦	♦	×	×	×	×	×	×
		INTRX1, TX1	♦	♦	×	×	×	×	×	×
		INTS2	♦	♦	×	×	×	×	×	×
		INTAD	♦	×	×	×	×	×	×	×
	RESET		♦	♦	♦	♦	♦	♦	♦	♦

♦: After release of the HALT mode, the CPU starts interrupt processing. (RESET initializes LSI.)

○: After release of the HALT mode, the CPU starts executing the next instruction that follows the HALT instruction.

×: Cannot be used to release the HALT mode.

–: This combination type does not exist because the priority level (Interrupt request level) of non-maskable interrupts is fixed to the highest priority level 7.

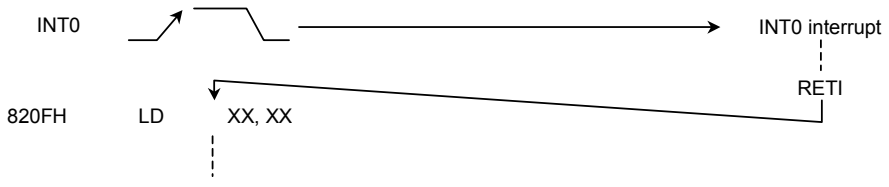
*1: Release the HALT mode is executed after the warm-up cycle is completed.

Note: When release of the HALT mode is executed by an INT0 interrupt of the level mode in the interrupt enabled status, maintain level “H” until the start of interrupt processing. If level “L” is set before the start of interrupt processing, interrupt processing is correctly started.

Example of releasing the RUN mode:

An INT0 interrupt releases the halt state when the RUN mode is on.

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



(3) Operation

1. RUN mode

In the RUN mode, the system clock in the MCU continues to operate even after a HALT instruction is executed. Only the CPU stops executing further instructions.

In the halt state, an interrupt request is accepted on the falling edge of the CLK signal.

Release of the RUN mode is executed by the external or internal interrupts. (See Table 3.3.6 "Halt Release Sources and Halt Release Operations".)

Figure 3.3.6 shows the timing for releasing the halt state by interrupts in the RUN or IDLE2 modes.

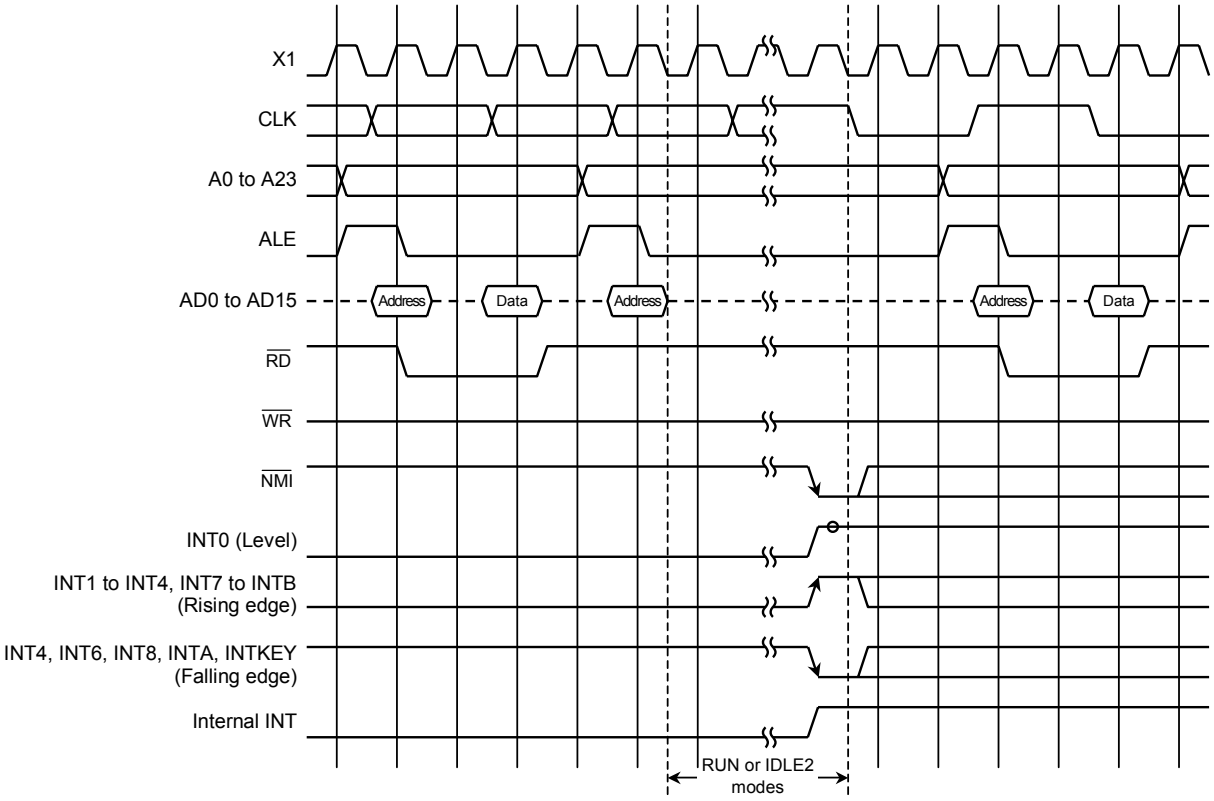


Figure 3.3.6 Timing Chart for Releasing the Halt State by Interrupt in RUN/IDLE2 Modes

2. IDLE2 mode

In the IDLE2 mode, the system clock signal is supplied only to specific internal I/O devices, and the CPU stops executing the current instruction. In the IDLE2 mode, the halt state is released by an interrupt with the same timing as in the RUN mode. The IDLE2 mode is released by external or internal interrupts, except for INTWDT and INTAD interrupts. (See Table 3.3.6 "Halt Release Sources and Halt Release Operations".)

In the IDLE2 mode, the watchdog timer should be disabled before entering the halt status, to prevent the watchdog timer interrupt from occurring just after release of the HALT mode.

3. IDLE1 mode

In the IDLE1 mode, the internal oscillator and the timer for realtime clock operates. The system clock in the MCU stops, and the CLK pin is fixed at the level "H" in the output enabled state. (CKOCR<CLKEN> = 1)

In the halt state, an interrupt request is sampled unsyncronously with the system clock, however the halt release (Restart of operation) is performed synchronously with it.

IDLE1 mode is released by external interrupts (NMI, INT0 to INT4, INTKEY) and internal interrupts (INTRTC). (See Table 3.3.6 "Halt Release Sources and Halt Release Operations".)

When the IDLE mode is used, set (TRUN<PRRUN> to 0) to stop the 9-bit prescaler before a HALT instruction is executed, to reduce the power consumption.

Figure 3.3.7 illustrates the timing for releasing the halt state by interrupts in the IDLE1 mode.

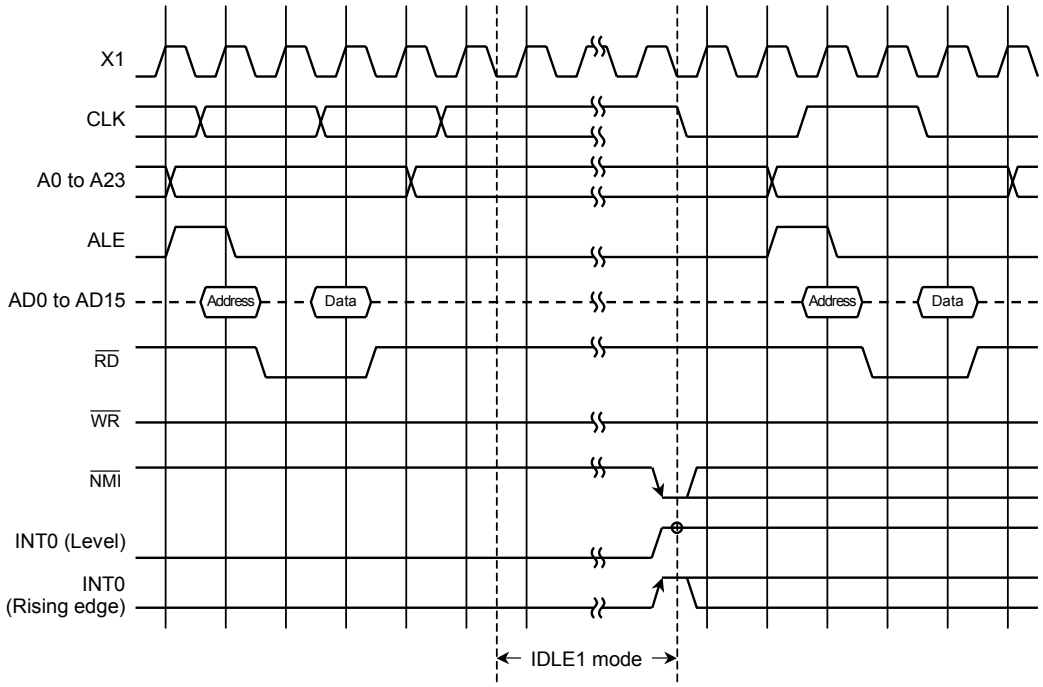


Figure 3.3.7 Timing Chart of Halt State Release by Interrupts in IDLE1 Mode

4. STOP mode

The STOP mode is selected to stop all internal circuits including the internal oscillator.

The pin status in the STOP mode depends on the setting of a bit in the watchdog timer mode register WDMOD<DRVE>. (See Figure 3.3.5 for setting of WDMOD<DRVE>.) Table 3.3.8 summarizes the state of these pins in the STOP mode.

The STOP mode is released by external interrupts (NMI, INT0 to INT4). When the STOP mode is released, the system clock output starts after the warm-up time required to attain stable oscillation.

The warm-up time can be set using WDMOD<WARM>. See the example of warm-up time setting (Table 3.3.7).

In a system which supplies a stable clock signal generated by an external oscillator, the warm-up time can be reduced by using the setting of T16CR<QCU>.

Figure 3.3.8 illustrates the timing for releasing the halt state by interrupts during the STOP mode.

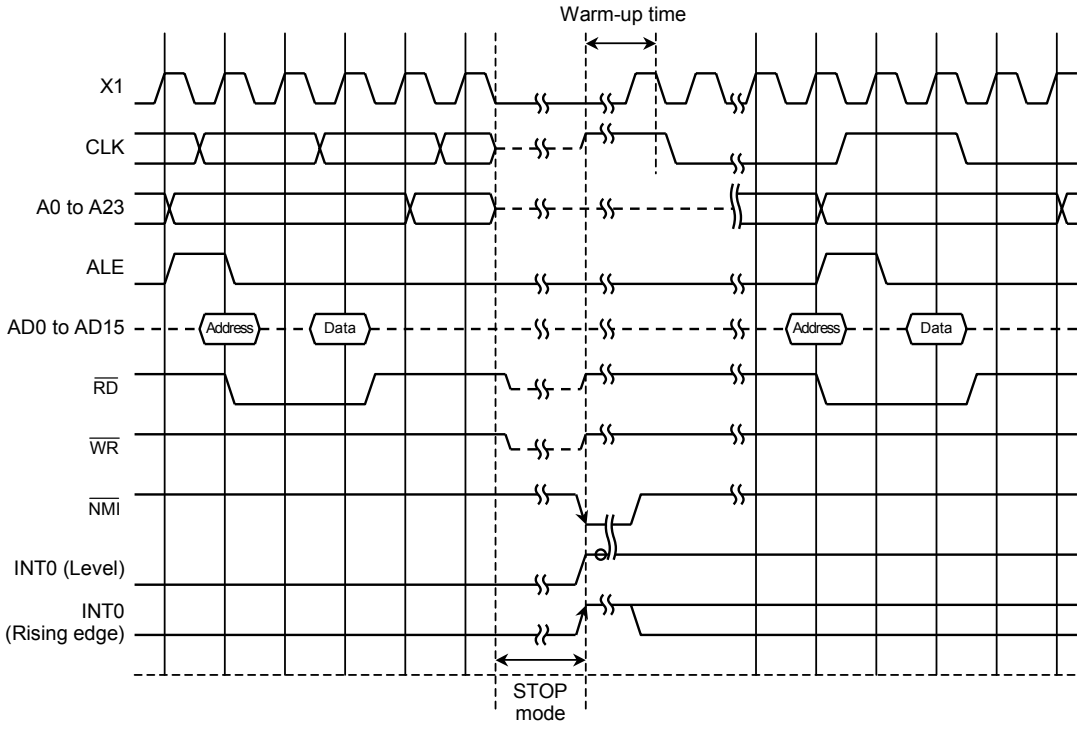


Figure 3.3.8 Timing Chart of Halt State Release by Interrupts in STOP mode

Table 3.3.7 Example of Warm-up Time after Releasing the STOP Mode

Clock Operation Frequency after the STOP Mode is Released	Warm-up Time [ms]		Clock Frequency
	WDMOD<WARM> = 0	WDMOD<WARM> = 1	
fc	0.8192	3.2768	fc = 20 MHz
fc/2	1.6384	6.5536	
fc/4	3.2768	13.1072	
fc/8	6.5536	26.2144	
fc/16	13.1072	52.4288	
fs	500	2000	fs = 32.768 kHz

How to calculate the warm-up time

WDMOD<WARM> = 0: Clock operation frequency after the 2^{14}/STOP mode.

WDMOD<WARM> = 1: Clock operation frequency after the 2^{16}/STOP mode.

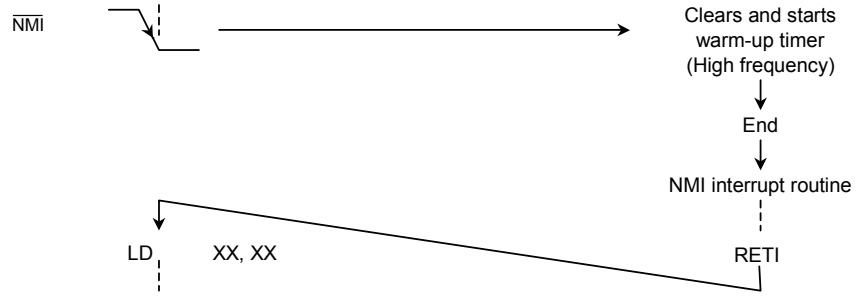
The selection of normal versus SLOW modes is possible after the STOP mode is released.

This selection is mode according to the contents of the SYSCR0<RSYSCK> register. Therefore, setting <RSYSCK>, <RXEN>, and <RXTEN> is necessary before the HALT instruction is executed.

Setting example:

In this illustrative case, the STOP mode is entered while the clock is operating at low frequency (fs). After the Stop mode is released by a NMI interrupt, the clock resumes operation at high frequency.

```
Address
         SYSCR0   EQU   006EH
         SYSCR1   EQU   006FH
         WDMOD    EQU   005CH
8FFDH    SYSCR1   LD    (SYSCR1), 08H          ;  fSYS = fs/2.
9000H             RES   4, (WDMOD)             ;  Sets warm-up time to 2^14/fc.
9002H             LD    (SYSCR0), –11000 – –B  ;  Operates at high frequency after STOP mode
                                                   is released.
9005H             HALT
```



–: No change

Note: When different operation modes are used before and after the STOP mode, and halt release interrupt request is accepted during execution of the HALT instruction (8 states), it is possible to release the HALT mode without changing the operation mode. In a system which accepts interrupts during execution of the HALT instruction, set the same operation mode before and after the STOP mode.

Table 3.3.8 Pin States in STOP Mode

Pin Name	I/O	<DRVE> = 0	<DRVE> = 1
P00 to P07	Input/Output mode	▲	▲
	Output mode	High-Z	Output
	AD8 to AD15	High-Z	High-Z
P10 to P17	Input/Output mode	▲	▲
	Output mode	High-Z	Output
	AD0 to AD7	High-Z	High-Z
P20 to P27	Input mode	▲	▲
	Output mode, A0 to A7/A16 to A23	▲	Output
P30 ($\overline{RD}$), P31 ($\overline{WR}$)	Output	High-Z	Output
P32	Input mode	PU*	Input
	Output mode	PU*	Output
P33	Input mode	PU	Input
	Output mode	PU	Output
	Input mode (INT0)	Input	Input
P34 to P37	Input mode	PU*	Input
	Output mode	PU*	Output
P4	Input mode	PU	Input
	Output mode	PU	Output
	Input mode (KEY0 to KEY7)	Input	Input
P5	Input mode	▲	▲
P6	Input mode	Invalid	Input
	Output mode	High-Z	Output
P70 to P72	Input mode	PU*	Input
	Output mode	PU*	Output
P73	Input mode	PU*	Input
	Output mode/SCOUT	PU*	Output
P74 to P76	Input mode	PU*	Input
	Output mode	PU*	Output
P77	Input mode	PU	Input
	Output mode	PU	Output
	NMI	Input	Input
P80 to P85	Input mode	Invalid	Input
	Output mode	High-Z	Output
P86	Input mode	Invalid	Input
	Output mode	High-Z	Output*
	XT1	Invalid	Invalid
P87	Input mode	Invalid	Input
	Output mode	High-Z	Output*
	XT2	Invalid	Invalid
P9	Output mode	High-Z	Output
	SEG24 to SEG31	"L"	"L"
PA	Output mode	High-Z	Output
	SEG32 to SEG39	"L"	"L"
ALE	Output (<ALEEN> = 1)	"L"	"L"
CLK	Output (<CLKEN> = 1)	High-Z	"H"
$\overline{RESET}$	Input	Input	Input
$\overline{EA}$	Input	Input	Input
X1	Input	Invalid	Invalid
X2	Output	"H"	"H"

(Align)

Input: Input gate in operation. Fix input voltage to low or high so that the input state pin stays constant.

Output: Output state

Output*: Open-drain output state. Input gate in operation. Set output to low or attach pull up on pin that the input gate stays constant.

Invalid: Input gate in operation.

High-Z: Output is at high impedance.

PU: Programmable pull-up pin. When a pull-up resistor is not set, fix the pin to avoid through current because the input gate always operates.

PU*: Programmable pull-up pin in input gate disable state. No through current flows even if the pin is set to high impedance.

▲: When a HALT instruction is executed and CPU stops at the address of the port register, an input gate operates. Fix the pin to avoid through current, and change the program. In all other cases, input is not accepted.

Note: Port registers are used for controlling programmable pull up/pull down. If a pin can be used for an output function (e.g., TO3) and the output function is specified, whether pull up or pull down is selected depends on the output function data. If a pin can be used for an input function, whether pull up is selected depends on the port register setting value only.

3.4 Interrupts

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

Altogether the TMP93CS20 has the following 45 interrupt sources:

- Interrupts from the CPU, 9 sources
  (Software interrupts, and illegal (Undefined) instruction execution)
- Interrupts from external pins ($\overline{NMI}$, INT0 to INT4, INT7 to INTB, and key-on wakeup), 12 sources
- Interrupts from built-in I/Os, 24 sources

A fixed individual interrupt vector number is assigned to each interrupt source; any one of 6 levels of priority can also be assigned to each maskable interrupt. Non-maskable interrupts have a fixed priority of 7.

When an interrupt is generated, the interrupt controller sends the value of the priority of the interrupt source to the CPU. When more than one interrupt is generated simultaneously, the interrupt controller sends the value of the highest priority (7 for non-maskable interrupts is the highest) to the CPU.

The CPU compares the value of the priority sent, with the value in the CPU interrupt mask register (IFF2 to IFF0). If the value sent is greater than in that the CPU interrupt mask register, the interrupt is accepted. However, software interrupts and illegal instruction execution interrupts are not compared with the <IFF2:0> register. They are given top priority. The value in the CPU interrupt mask register (IFF2 to IFF0) can be changed using the EI instruction. Executing EI n changes the contents of <IFF2:0> to n. For example, programming EI 3 enables acceptance of maskable interrupts with a priority of 3 or greater, and non-maskable interrupts which are set in the interrupt controller. When EI or EI 0 is programmed, maskable interrupts with a priority of 1 or greater, and non-maskable interrupts are enabled in the interrupt instructions. (In the same way as the EI 1)

The DI instruction operates in the same way as the EI 7 instruction, setting <IFF2:0> = 7. Since the priority values for maskable interrupts are 0 to 6, the DI instruction is used to disable acceptance of maskable interrupts. The EI instruction becomes effective immediately after execution. (with the TLCS-90, the EI instruction becomes effective after execution of the next following instruction.)

In addition to the general-purpose interrupt processing mode described above, there is also a micro DMA processing mode. Micro DMA is a mode used by the CPU to automatically transfer byte or word data. It enables the CPU to process interrupts such as data saves to built-in I/Os at high speed.

Figure 3.4.1 is a flowchart showing overall interrupt processing.

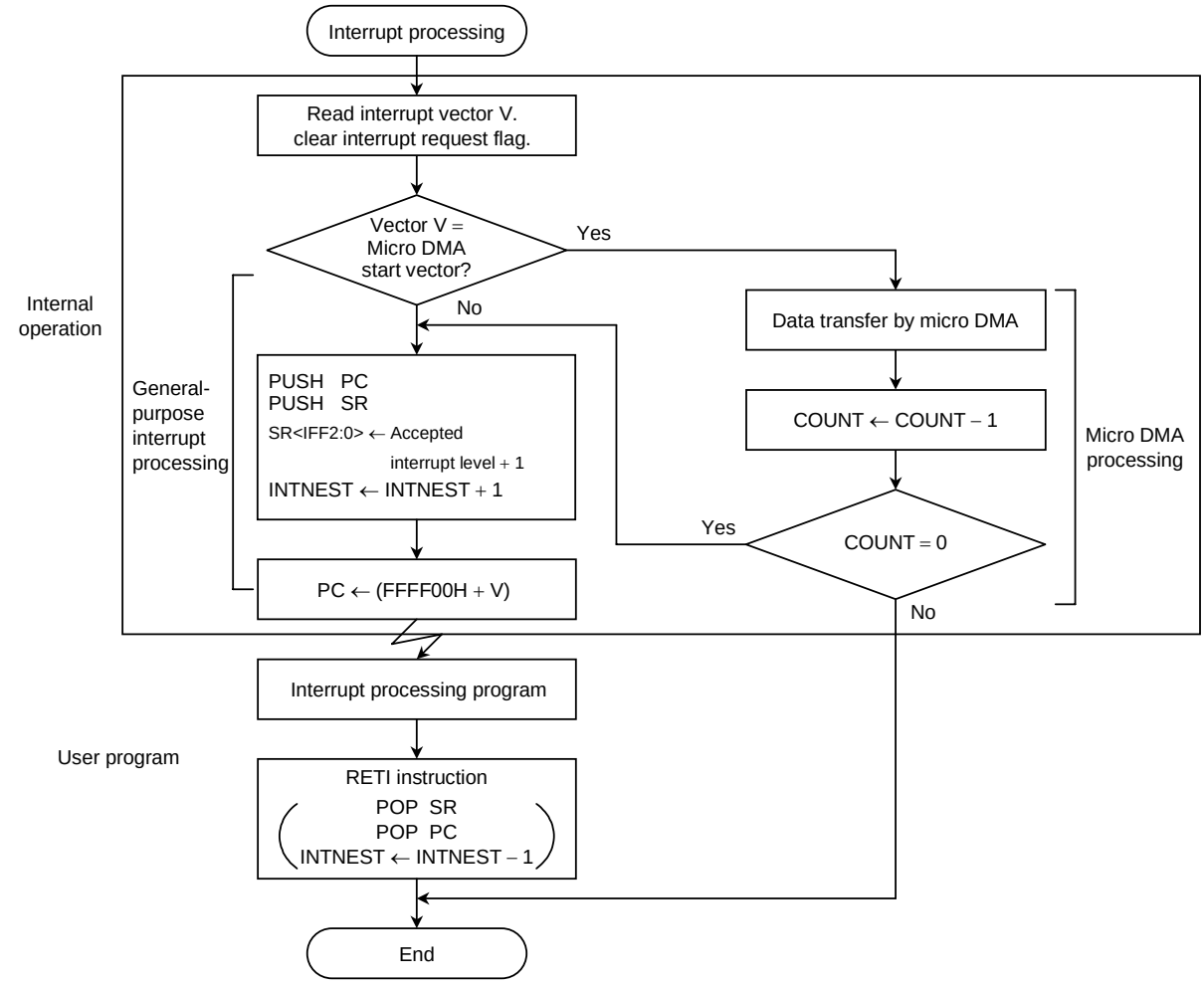


Figure 3.4.1 Interrupt Processing Flowchart

3.4.1 General-purpose Interrupt Processing

When accepting an interrupt, the CPU operates as follows. In the cases of software interrupts or interrupts generated by the CPU because of attempts to execute illegal instructions, the following steps (1) and (3) are not executed.

(1) The CPU reads the interrupt vector from the interrupt controller. When more than one interrupt with the same priority level is generated simultaneously, the interrupt controller generates interrupt vectors in accordance with the default priority, then clears the interrupt request. The default priority is fixed as follows: the smaller the vector value, the higher the priority.

(2) The CPU pushes the program counter and the status register to the system stack area (Area indicated by the system mode stack pointer (XSP)).

(3) The CPU sets a value in the CPU interrupt mask register <IFF2:0> that is higher by 1 than the priority level value of the accepted interrupt. However, if the accepted interrupt's priority value is 7, 7 is set without an increment.

(4) The CPU increments the interrupt nesting counter (INTNEST).

(5) The CPU jumps to an address stored in the FFFF00H + interrupt vector, then starts the interrupt processing routine.

The following table shows the number of processing states corresponding to steps 1 to 5 above.

Bus Width of Stack Area	Bus Width of Interrupt Vector Area	Number of Interrupt Processing States
8 bits	8 bits	35
	16 bits	31
16 bits	8 bits	29
	16 bits	25

The RETI instruction is usually used to complete the interrupt processing. Executing this instruction restores the contents of the program counter and the status registers, and decrements the interrupt nesting counter (INTNEST).

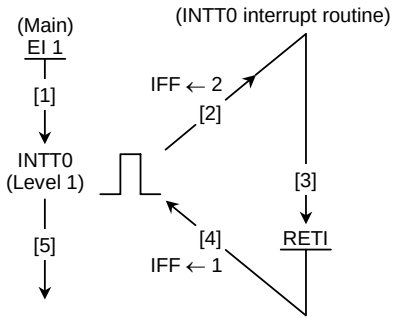
Though acceptance of non-maskable interrupts cannot be disabled by programming, acceptance of maskable interrupts can. A priority can be set for each source of maskable interrupts. The CPU accepts any interrupt request with a priority higher than the current value in the CPU mask register <IFF2:0>. The CPU mask register <IFF2:0> is then set to a value higher by 1 than the priority of the accepted interrupt. Thus, if another interrupt is generated with a priority level higher than the interrupt currently being processed, the CPU accepts the interrupt with the higher level, causing interrupt processing to nest.

If an interrupt request with a priority higher than the currently-processed interrupt is generated during the time that CPU is processing the above steps (1) to (5), and is accepted before the first instruction in the interrupt processing routine is executed, this will cause interrupt processing to nest. (The nesting process is the same as in the case of overlapping each non-maskable interrupt (Level 7).) The CPU does not accept an interrupt request of the same priority level as that of the interrupt currently being processed.

Resetting initializes the CPU mask registers <IFF2:0> to the value 7, therefore, acceptance of all maskable interrupts is disabled.

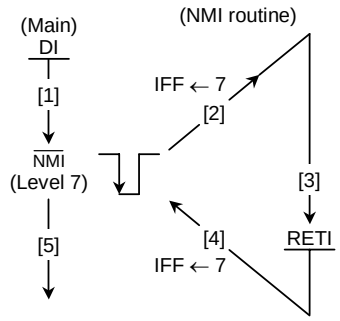
The following (1) to (5) show a flowchart of interrupt processing.

(1) Maskable interrupt



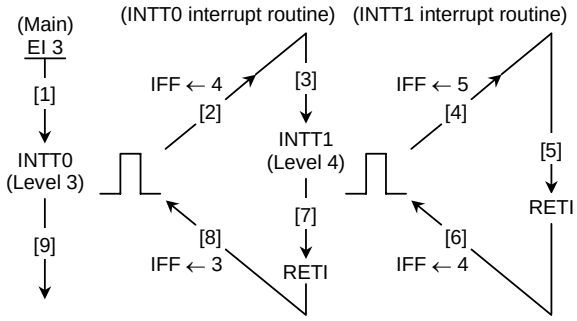
During execution of the main program, the CPU accepts an interrupt request. The CPU then increments IFF so that no new interrupts of priority level 1 will be accepted during processing of the interrupt routine.

(2) Non-maskable interrupt



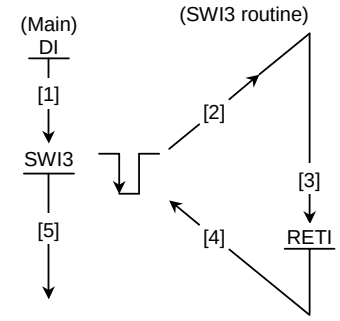
The DI instruction is executed in the main program, so that only interrupts of priority level 7 are accepted. In this state the CPU does not increment the IFF even if the CPU accepts an interrupt request of level 7.

(3) Interrupt nesting



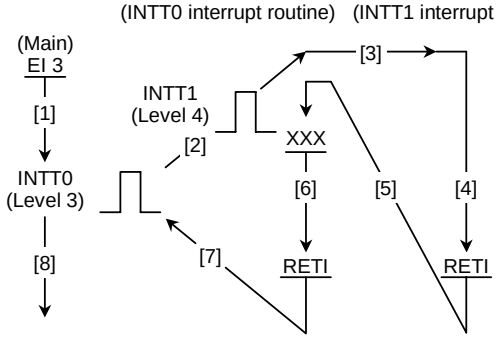
During processing an interrupt of priority level 3, the IFF is set to 4. When an interrupt with a level higher than 4 is generated, the CPU accepts the interrupt with the higher priority level, causing interrupt processing to nest.

(4) Software interrupt



The CPU accepts a software interrupt request during DI status (IFF = 7) because the request has a priority of level 7. The IFF is not changed by the software interrupt.

(5) Timing of interrupt acceptance



If an interrupt with a priority level higher than the interrupt currently being processed is generated, the CPU accepts the interrupt with the higher level. The program counter which returns at [5] is the state address of the INTT0 interrupt routine.

Note: ___ (underline): Instruction
[1], [2],……: Execution flow

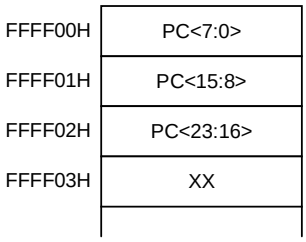
The addresses FFFF00H to FFFFFFH (256 bytes) of the TMP93CS20 are assigned as interrupt vector areas.

Table 3.4.1 TMP93CS20 Interrupt Table

Default Priority	Type	Interrupt Source	Vector Value "V"	Address Referring to Vector	Micro DMA Start Vector
1	Non-maskable	Reset, or SWI0 instruction	0000H	FFFF00H	–
2		SWI 1 instruction	0004H	FFFF04H	–
3		Illegal instruction, or SWI2	0008H	FFFF08H	–
4		SWI 3 instruction	000CH	FFFF0CH	–
5		SWI 4 instruction	0010H	FFFF10H	–
6		SWI 5 instruction	0014H	FFFF14H	–
7		SWI 6 instruction	0018H	FFFF18H	–
8		SWI 7 instruction	001CH	FFFF1CH	–
9		NMI: $\overline{\text{NMI}}$ pin input	0020H	FFFF20H	08H
10		INTWD: Watchdog timer	0024H	FFFF24H	09H
11	Maskable	INT0: INT0 pin input	0028H	FFFF28H	0AH
12		INT1: INT1 pin input	002CH	FFFF2CH	0BH
13		INT2: INT2 pin input	0030H	FFFF30H	0CH
14		INT3: INT3 pin input	0034H	FFFF34H	0DH
15		INT4: INT4 pin input	0038H	FFFF38H	0EH
16		INT7: INT7 pin input	003CH	FFFF3CH	0FH
17		INT8: INT8 pin input	0040H	FFFF40H	10H
18		INT9: INT9 pin input	0044H	FFFF44H	11H
19		INTA: INTA pin input	0048H	FFFF48H	12H
20		INTB: INTB pin input	004CH	FFFF4CH	13H
21		INTKEY: Key wakeup	0050H	FFFF50H	14H
22		INTT0: 8-bit timer 0 (TREG0)	0054H	FFFF54H	15H
23		INTT1: 8-bit timer 1 (TREG1)	0058H	FFFF58H	16H
24		INTT2: 8-bit timer 2 (TREG2)	005CH	FFFF5CH	17H
25		INTT3: 8-bit timer 3 (TREG3)	0060H	FFFF60H	18H
26		INTTR4: 16-bit timer 4 (TREG4)	0064H	FFFF64H	19H
27		INTTR5: 16-bit timer 4 (TREG5)	0068H	FFFF68H	1AH
28		INTTR6: 16-bit timer 6 (TREG6)	006CH	FFFF6CH	1BH
29		INTTR7: 16-bit timer 6 (TREG7)	0070H	FFFF70H	1CH
30		INTTR8: 16-bit timer 8 (TREG8)	0074H	FFFF74H	1DH
31		INTTR9: 16-bit timer 8 (TREG9)	0078H	FFFF78H	1EH
32		INTTRA: 16-bit timer A (TREGA)	007CH	FFFF7CH	1FH
33		INTTRB: 16-bit timer A (TREGB)	0080H	FFFF80H	20H
34		INTTO4: 16-bit timer 4 (Overflow)	0084H	FFFF84H	21H
35		INTTO6: 16-bit timer 6 (Overflow)	0088H	FFFF88H	22H
36		INTTO8: 16-bit timer 8 (Overflow)	008CH	FFFF8CH	23H
37		INTTOA: 16-bit timer A (Overflow)	0090H	FFFF90H	24H
38		INTRX0: Serial receive (channel 0)	0094H	FFFF94H	25H
39		INTTX0: Serial send (channel 0)	0098H	FFFF98H	26H
40		INTRX1: Serial receive (channel 1)	009CH	FFFF9CH	27H
41		INTTX1: Serial send (channel 1)	00A0H	FFFFA0H	28H
42		INTS2: Serial send/receive/send and receive, I^2C bus	00A4H	FFFFA4H	29H
43		INTAD: AD conversion completion	00A8H	FFFFA8H	2AH
44		INTRTC: Timer for realtime clock	00ACH	FFFFACH	2BH
–		(Reserved)	00B0H	FFFFB0H	–
to		to	to	to	to
–		(Reserved)	00FCH	FFFFFCH	–

Setting to reset and interrupt vectors

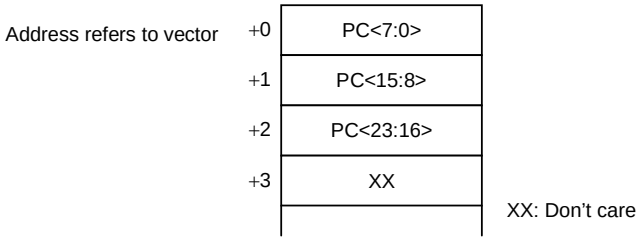
1. Reset vector



The vector base addresses are dependent on the products.

Type No.	Vector Base Address	PC Setting Sequence after Reset	Notes
TMP93CS44 TMP93CS45 TMP93PS44 TMP93CS32 TMP93PW32 TMP93CS20 TMP93PW20A	FFFF00H	PC<7:0> ← Data in location FFFF00H PC<15:8> ← Data in location FFFF01H PC<23:16> ← Data in location FFFF02H	P27 to P20 and A23 to A16 pins are defined as input ports and are pulled down in resetting. The logic data item is FFH. When Port 2 is used for the A23 to A16 output to access the program ROM, set PC<23:16> to FFH and set the reset vector to lie within the area FF0000H to FFFFFFH. (This is applicable mainly to products without ROM.)

2. Interrupt vector (except reset vector)



Setting example:

Set the reset vector to FF0000H, $\overline{\text{NMI}}$ vector to FF9ABCH and INTAD vector to 123456H.

```
ORG     FFFF00H
DL      FF0000H         ;   Reset = FF0000H

ORG     FFFF20H
DL      FF9ABCH         ;   NMI = FF9ABCH

ORG     FFFFA8H
DL      123456H         ;   INTAD = 123456H

ORG     FF0000H
LD      A, B
        :
        :

ORG     FF9ABCH
LD      B, C
        :
        :

ORG     123456H
LD      C, A
        :
        :
```

Note:
ORG and DL are assembler directives.
ORG: Control location counter.
DL: Defines long word (32 bits) data.

3.4.2 Micro DMA

In addition to the conventional interrupt processing, the TMP93CS20 also has a micro DMA function. When an interrupt is accepted, in addition to an interrupt vector, the CPU receives data indicating whether it is to be processed in micro DMA mode or in general-purpose interrupt mode. The CPU performs micro DMA processing only if that mode is requested.

The micro DMA of the TMP93CS20 can process at very high speed compared with the TLCS-90 micro DMA because it has transfer parameters in dedicated registers in the CPU. Since those dedicated registers are assigned as CPU control registers, they can only be accessed by the LDC instruction.

(1) Micro DMA operation

Micro DMA operation starts when the accepted interrupt vector value matches the micro DMA start vector value set in the interrupt controller. The micro DMA has four channels, so that it can be set for up to four types of interrupt sources at the same time.

When a micro DMA interrupt is accepted, data are automatically transferred from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decremented. If the value in the counter after decrementing is other than 0, micro DMA processing is completed; if the value in the counter after decrementing is 0, general-purpose interrupt processing is performed.

32-bit control registers are used for setting transfer source and destination addresses. However, the TMP93CS20 has only 24 address pins for output. A 16-Mbyte space is available for the micro DMA.

There are two data transfer modes: 1-byte mode and 1-word mode. Incrementing, decrementing, and fixing the transfer source and destination addresses after transfer can be done in both modes. Therefore data can easily be transferred between I/O and memory and between different I/Os. For details of transfer modes, see the description of transfer mode registers.

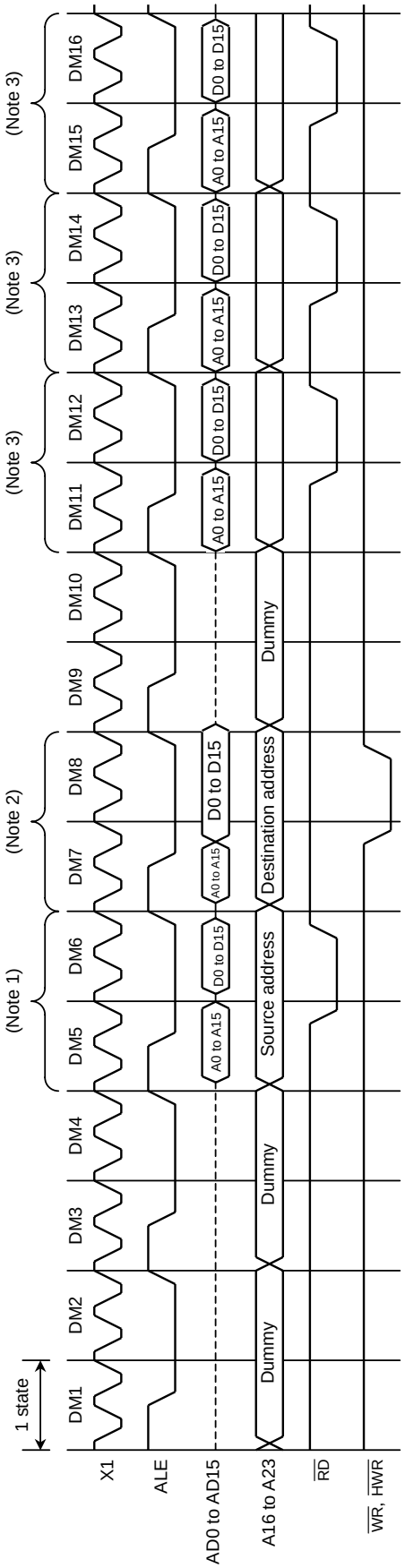
The transfer counter has 16 bits, so up to 65536 transfers (the maximum when the initial value of the transfer counter is 0000H) can be performed for one interrupt source by micro DMA processing.

When the transfer counter is decremented to 0 after data are transferred by the micro DMA, general-purpose interrupt processing is performed. After processing the general-purpose interrupt, restarting the interrupts of the same channel restarts the transfer counter from 65536. It is necessary to reset the transfer counter in the general-purpose interrupt processing routine.

Interrupt sources handled by micro DMA processing are 36 in total, and the micro DMA start vectors are listed in Table 3.4.1.

The following timing chart is a micro DMA cycle of the transfer address increment (INC) mode.

(Conditions: 16-bit bus width for 16 Mbytes, 0 waits.)

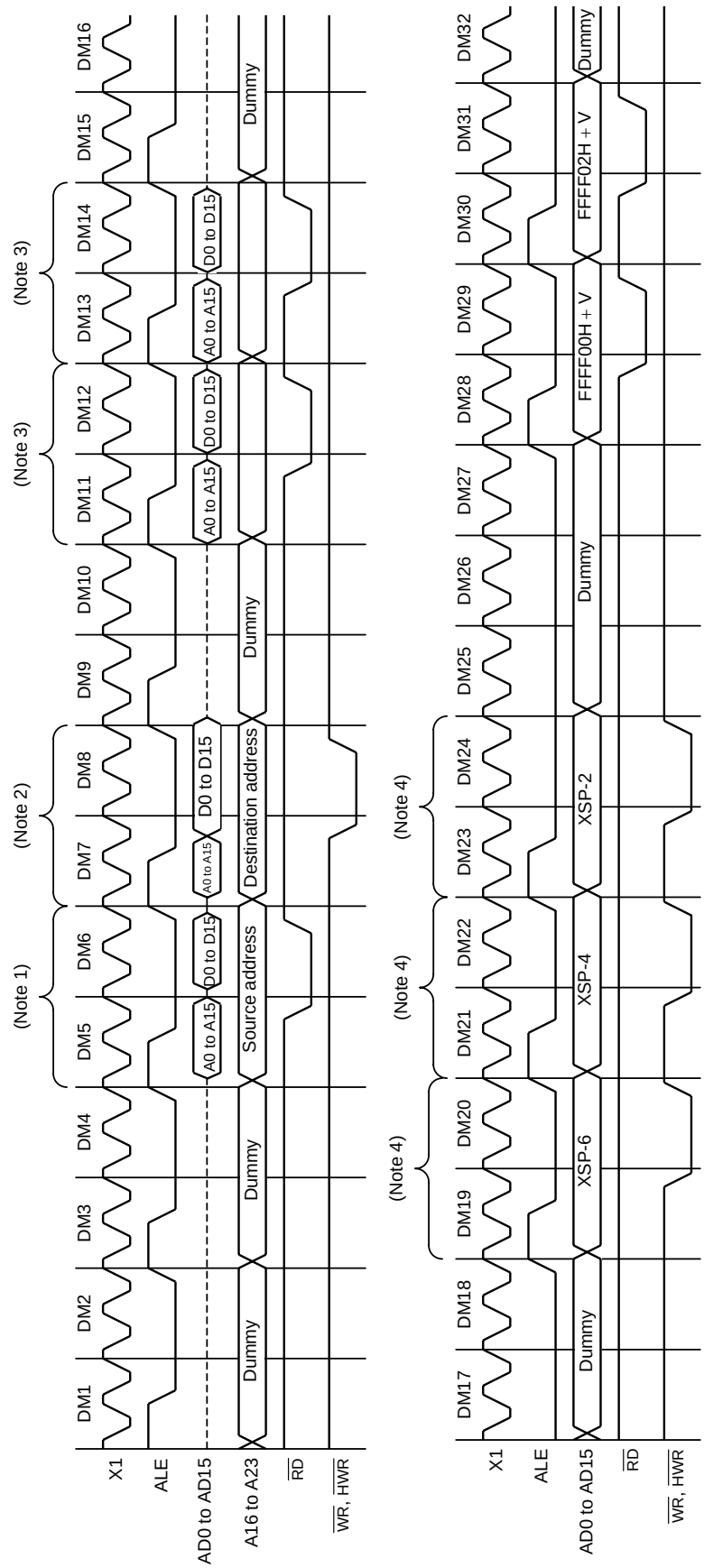


Note 1: These 2 states are added in the case that the bus width of the source address area is 8 bits or the address starts from an odd number.

Note 2: These 2 states are added in the case that the bus width of the destination address area is 8 bits or the address starts from an odd number.

Note 3: This may be a dummy cycle with an instruction queue buffer.

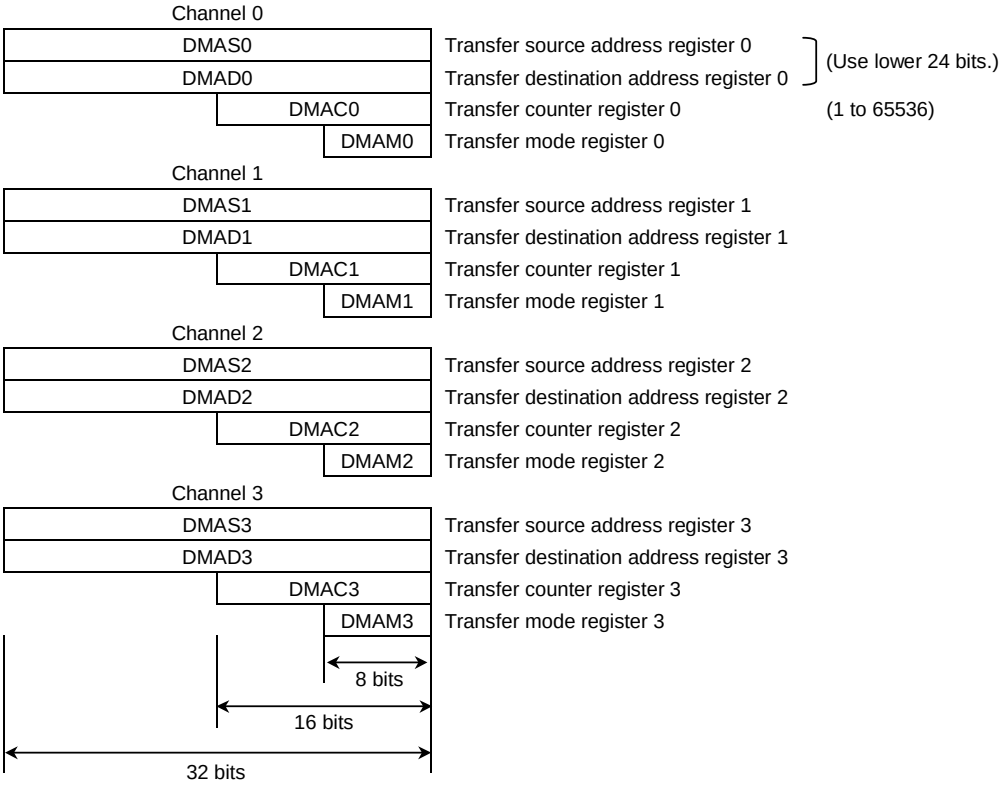
Figure 3.4.2 Micro DMA Cycle (COUNT ≠ 0)



Note 1: These 2 states are added in the case that the bus width of the source address area is 8 bits or the address starts from an odd number.
Note 2: These 2 states are added in the case that the bus width of the destination address area is 8 bits or the address starts from an odd number.
Note 3: This may be a dummy cycle with an instruction queue buffer.
Note 4: These 2 states are added in the case that the bus width of the stack address area is 8 bits or the stack pointer starts from an odd number.

Figure 3.4.3 Micro DMA Cycle (COUNT = 0)

(2) Register configuration (CPU control registers)

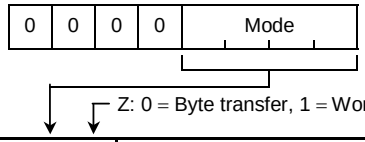


These control registers can only be set with the "LDC cr, r" instruction.

Example:

```
LD      XWA, 100H
LDC     DMAS0, XWA
LD      XWA, 50H
LDC     DMAD0, XWA
LD      WA, 40H
LDC     DMAC0, WA
LD      A, 05H
LDC     DMAM0, A
```

(3) Transfer mode register details



Note: When setting values for this register, set the upper 4 bits to 0.

					Execution time
0	0	0	Z	Transfer destination address INC mode ..... for I/O to memory (DMADn+) ← (DMASn) DMACn ← DMACn – 1 if DMACn = 0 then INT.	16 states (1.6 μs)
0	0	1	Z	Transfer destination address DEC mode..... for I/O to memory (DMADn–) ← (DMASn) DMACn ← DMACn – 1 if DMACn = 0 then INT.	16 states (1.6 μs)
0	1	0	Z	Transfer source address INC mode ..... for memory to I/O (DMADn) ← (DMASn+) DMACn ← DMACn – 1 if DMACn = 0 then INT.	16 states (1.6 μs)
0	1	1	Z	Transfer source address DEC mode..... for memory to I/O (DMADn) ← (DMASn–) DMACn ← DMACn – 1 if DMACn = 0 then INT.	16 states (1.6 μs)
1	0	0	Z	Fixed address mode ..... I/O to I/O (DMADn) ← (DMASn) DMACn ← DMACn – 1 if DMACn = 0 then INT.	16 states (1.6 μs)
1	0	1	1	Counter mode..... for interrupt counter DMASn ← DMASn + 1 DMACn ← DMACn – 1 if DMACn = 0 then INT.	11 states (1.1 μs)

Note 1: n: Corresponds to micro DMA channels 0 to 3.
DMADn+/DMASn+: Post-increment (Increments register value after transfer).
DMADn–/DMASn–: Post-decrement (Decrements register value after transfer).

Note 2: Execution time: When setting source address/destination address area to 16-bit bus, 0 waits.

Note 3: Do not use any codes for transfer mode registers other than those indicated above.

3.4.3 Interrupt Controller

Figure 3.4.4 is a block diagram of the interrupt circuits. The left half of the diagram shows the interrupt controller; the right half includes the CPU interrupt request signal circuit and the halt release signal circuit.

Each interrupt channel (Total of 36 channels) in the interrupt controller has an interrupt request flag, interrupt priority setting register, and a register for storing the micro DMA start vector.

The flag is cleared to 0 when any of the following conditions are met.

- upon resetting
- when the CPU reads the interrupt vector after acceptance of an interrupt
- when the CPU executes an instruction that clears the interrupt from that channel (Writes 0 in <IxxC> of the interrupt priority setting register)

For example, to clear the INT0 interrupt request, after the DI instruction set the register INTE0AD as follows.

```
LD      (INTE0AD), ----0---B
```

The status of the interrupt request flag is detected by reading the corresponding clear bit. This also allows the interrupt to be identified by the software.

The interrupt priority can be set by writing the priority in the interrupt priority setting register (e.g., INTE0AD or INTE43) provided for each interrupt source. Interrupt priority levels to be set range from or 1 to 6. Except for NMIs (Non-maskable interrupts), writing 0 or 7 as the interrupt priority disables the corresponding interrupt request. The priority of non-maskable interrupt sources ($\overline{\text{NMI}}$ pin, watchdog timer, etc.) is fixed to 7. If interrupt requests with the same interrupt level are generated simultaneously, interrupts are accepted in accordance with the default ranking of priorities.

The interrupt controller selects the interrupt request with the highest priority among the simultaneous interrupts, and sends it and its vector address to the CPU. The CPU compares the priority value <IFF2:0> set in the status register, with the priority value sent by the interrupt request signal; if the latter is higher, the interrupt is accepted. Then the CPU sets in CPU SR<IFF2:0> a value equal to one plus the priority value of the interrupt request just received. Interrupt requests whose priority values equal or are higher than the value set in the register are accepted concurrently with execution of the previous interrupt routine. When interrupt processing is completed (after execution of the RETI instruction), the CPU restores to CPU SR<IFF2:0> the priority value saved in the stack before the interrupt was generated.

The interrupt controller also has four registers used to store the micro DMA start vector. Unlike other micro DMA registers (DMAS, DMAD, DMAM, and DMAC), these are I/O registers. Writing the start vector of the interrupt source for micro DMA processing (See Table 3.4.1), enables the corresponding interrupt to be processed by micro DMA. Please note that appropriate values must be set in the micro DMA parameter registers (e.g., DMAS and DMAD) prior to the beginning of micro DMA processing.

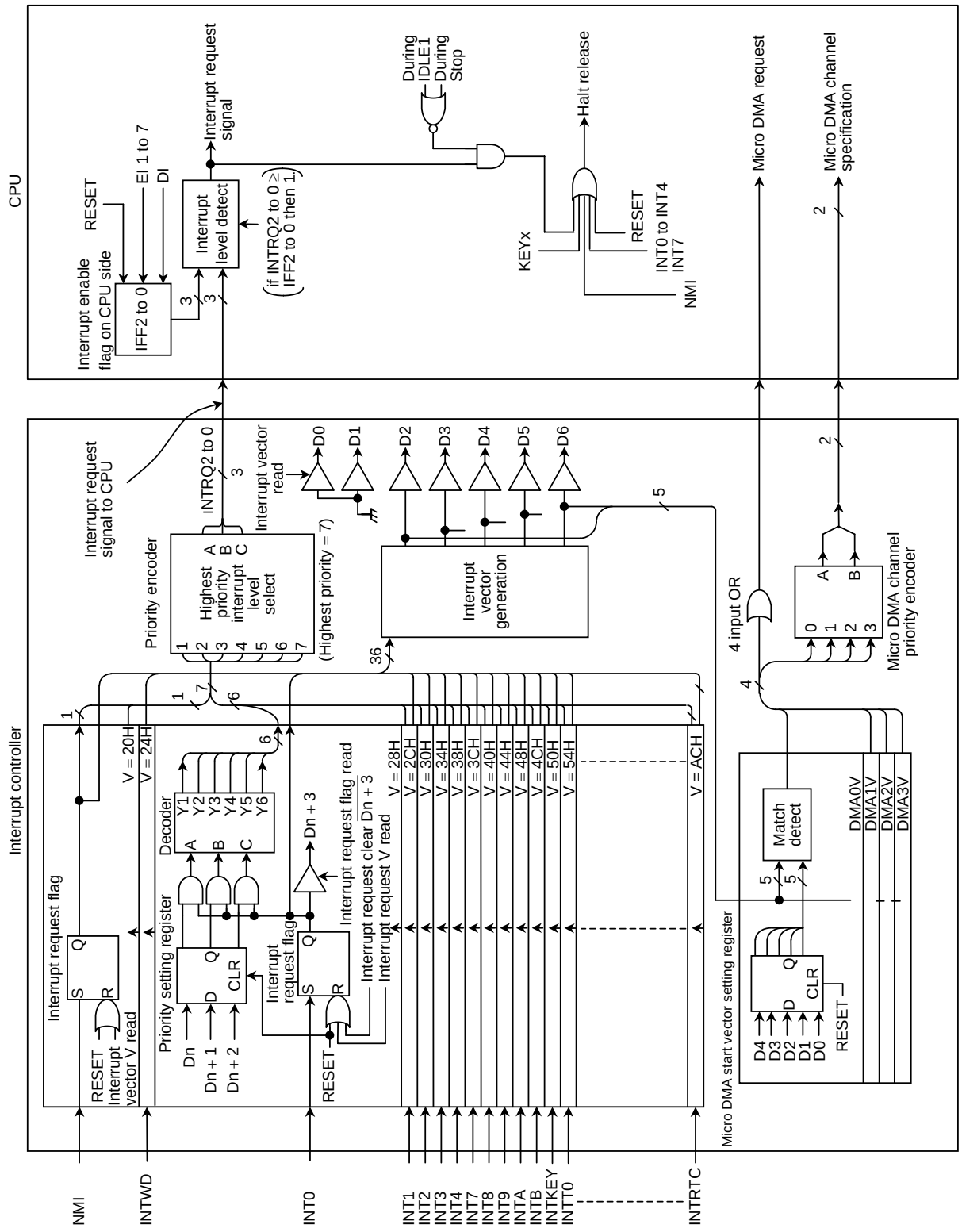


Figure 3.4.4 Block Diagram of Interrupt Controller

(1) Interrupt priority setting register

Symbol	Address	7	6	5	4	3	2	1	0	
INTE0AD	0070H	INTAD				INT0				←Interrupt source
		IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0	←Bit symbol
		R/W	W			R/W	W			←Read/Write
		0	0	0	0	0	0	0	0	←After reset
INTE21	0071H	INT2				INT1				
		I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE43	0072H	INT4				INT3				
		I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE98	0073H	INT9				INT8				
		I9C	I9M2	I9M1	I9M0	I8C	I8M2	I8M1	I8M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTEBA	0074H	INTB				INTA				
		IBC	IBM2	IBM1	IBM0	IAC	IAM2	IAM1	IAM0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTET10	0075H	INTT1 (TREG1)				INTT0 (TREG0)				
		IT1C	IT1M2	IT1M1	IT1M0	IT0C	IT0M2	IT0M1	IT0M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTET32	0076H	INTT3 (TREG3)				INTT2 (TREG2)				
		IT3C	IT3M2	IT3M1	IT3M0	IT2C	IT2M2	IT2M1	IT2M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTES0	0077H	INTTX0				INTRX0				
		ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTES1	0078H	INTTX1				INTRX1				
		ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE7S2	0079H	INT7				INTS2				
		I7C	I7M2	I7M1	I7M0	IS2C	IS2M2	IS2M1	IS2M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Prohibits interrupt request.
0	0	1	Sets interrupt request level to 1.
0	1	0	Sets interrupt request level to 2.
0	1	1	Sets interrupt request level to 3.
1	0	0	Sets interrupt request level to 4.
1	0	1	Sets interrupt request level to 5.
1	1	0	Sets interrupt request level to 6.
1	1	1	Prohibits interrupt request.

IxxC	Function (Read)	Function (Write)
0	Indicates no interrupt request.	Clears interrupt request flag.
1	Indicates interrupt request.	Don't care

Note 1: Read-modify-write is prohibited.

Note 2: This note is about clearing interrupt request flags. The interrupt request flags of INTRX0 and INTRX1 are not cleared by writing 0 to IxxC because they are level-sense interrupts.

These interrupt request flags are cleared by clearing, resetting, or reading of the converted result or the receive buffer.

Figure 3.4.5 Interrupt Priority Setting Register (1)

Symbol	Address	7	6	5	4	3	2	1	0	
INTET54	007AH	INTTR5 (TREG5)				INTTR4 (TREG4)				←Interrupt source
		IT5C	IT5M2	IT5M1	IT5M0	IT4C	IT4M2	IT4M1	IT4M0	←Bit symbol
		R/W	W			R/W	W			←Read/Write
		0	0	0	0	0	0	0	0	←After reset
INTET76	0044H	INTTR7 (TREG7)				INTTR6 (TREG6)				
		IT7C	IT7M2	IT7M1	IT7M0	IT6C	IT6M2	IT6M1	IT6M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTET98	0045H	INTTR9 (TREG9)				INTTR8 (TREG8)				
		IT9C	IT9M2	IT9M1	IT9M0	IT8C	IT8M2	IT8M1	IT8M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTETBA	0046H	INTTRB (TREGB)				INTTRA (TREGA)				
		ITBC	ITBM2	ITBM1	ITBM0	ITAC	ITAM2	ITAM1	ITAM0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTEO64	0047H	INTTO6				INTTO4				
		ITO6C	ITO6M2	ITO6M1	ITO6M0	ITO4C	ITO4M2	ITO4M1	ITO4M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTEOA8	005AH	INTTOA				INTTO8				
		ITOAC	ITOAM2	ITOAM1	ITOAM0	ITO8C	ITO8M2	ITO8M1	ITO8M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTEKR	005BH	INTKEY				INTRTC				
		IKEYC	IKEYM2	IKEYM1	IKEYM0	IRTCC	IRTCM2	IRTCM1	IRTCM0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Prohibits interrupt request.
0	0	1	Sets interrupt request level to 1.
0	1	0	Sets interrupt request level to 2.
0	1	1	Sets interrupt request level to 3.
1	0	0	Sets interrupt request level to 4.
1	0	1	Sets interrupt request level to 5.
1	1	0	Sets interrupt request level to 6.
1	1	1	Prohibits interrupt request.

IxxC	Function (Read)	Function (Write)
0	Indicates no interrupt request.	Clears interrupt request flag.
1	Indicates interrupt request.	Don't care

Note: Read-modify-write is prohibited.

Figure 3.4.6 Interrupt Priority Setting Register (2)

(2) External interrupt control

Interrupt Input Mode Control Register

IIMC (007BH)

Read-modify-write is prohibited

	7	6	5	4	3	2	1	0
Bit symbol	I7FE	I4FE	I3FE	I2FE	I1FE	I0FE	I0LE	NMIREE
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: INT7 rising edge 1: INT7 falling edge	0: INT4 rising edge 1: INT4 falling edge	0: INT3 rising edge 1: INT3 falling edge	0: INT2 rising edge 1: INT2 falling edge	0: INT1 rising edge 1: INT1 falling edge	0: INT0 rising edge 1: INT0 falling edge	0: INT0 edge-sense mode 1: INT0 level-sense mode	1: Can be accepted in $\overline{\text{NMI}}$ rising edge.

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

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

INT0 level enable (Note 1)

0	Rising edge detect interrupt
1	High level interrupt

Note 1: This is a case of changing from level-sence to edge-sence for INT0 pin mode. (Changing from 1 to 0 for <IOIE>.)

Execution example:

```
LD    (INTE0AD), XXXX0000B    ;  INT0 disable, clear the request flag.
LD    (IIMC), XXXXX10XB       ;  Change from level to edge.
LD    (INTE0AD), XXXX0nnnB    ;  Set interrupt level "n" for INT0, clear the request flag.
```

Note 2: See electrical characteristics in section 4 for external interrupt input pulse.

Figure 3.4.7 Interrupt Input Mode Control Register

Table 3.4.2 Setting of External Interrupt Pin Function

Interrupt	Pin Name	Mode		Setting Method
$\overline{NMI}$	P77		Falling edge	IIMC<NMIREE> = 0, P7FC<NMIC> = 1
			Falling and rising edges	IIMC<NMIREE> = 1, P7FC<NMIC> = 1
INT0	P33		Rising edge	IIMC<I0LE> = 0, <I0FE> = 0, P3FC<I0IE> = 1
			Falling edge	IIMC<I0LE> = 0, <I0FE> = 1, P3FC<I0IE> = 1
			Level	IIMC<I0LE> = 1, P3FC<I0IE> = 1
INT1	P34		Rising edge	IIMC<I1FE> = 0, P3FC<I1IE> = 1
			Falling edge	IIMC<I1FE> = 1, P3FC<I1IE> = 1
INT2	P35		Rising edge	IIMC<I2FE> = 0, P3FC<I2IE> = 1
			Falling edge	IIMC<I2FE> = 1, P3FC<I2IE> = 1
INT3	P36		Rising edge	IIMC<I3FE> = 0, P3FC<I3IE> = 1
			Falling edge	IIMC<I3FE> = 1, P3FC<I3IE> = 1
INT4	P37		Rising edge	IIMC<I4FE> = 0, P3FC<I4IE> = 1
			Falling edge	IIMC<I4FE> = 1, P3FC<I4IE> = 1
INT7	P66/TI0		Rising edge	IIMC<I7FE> = 0, P6FC<I7IE> = 1
			Falling edge	IIMC<I7FE> = 1, P6FC<I7IE> = 1
INT8	P70		Rising edge	T8MOD<CAP89M1:0> = 0, 0 or 0, 1 or 1, 1
			Falling edge	T8MOD<CAP89M1:0> = 1, 0
INT9	P71		Rising edge	–
INTA	P74		Rising edge	TAMOD<CAPABM1:0> = 0, 0 or 0, 1 or 1, 1
			Falling edge	TAMOD<CAPABM1:0> = 1, 0
INTB	P75		Rising edge	–

(3) Micro DMA start vector

When the CPU reads the interrupt vector after accepting an interrupt, it simultaneously compares the interrupt vector with each channel's micro DMA start vector (Bits 2 to 6 of the interrupt vector). When the two match, the interrupt from the channel whose value matched is processed in micro DMA mode.

If the interrupt vector matches more than one channel, the channel with the lower channel number has a higher priority.

Micro DMA 0 State Vector

DMA0V (007CH)

	7	6	5	4	3	2	1	0
Bit symbol			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
Read/Write			W					
After reset			0	0	0	0	0	0
Function	Micro DMA channel 0 processed by matching bits 2 to 6 of the interrupt vector.							

Micro DMA 1 State Vector

DMA1V (007DH)

	7	6	5	4	3	2	1	0
Bit symbol			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
Read/Write			W					
After reset			0	0	0	0	0	0
Function	Micro DMA channel 1 processed by matching bits 2 to 6 of the interrupt vector.							

Micro DMA 2 State Vector

DMA2V (007EH)

	7	6	5	4	3	2	1	0
Bit symbol			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
Read/Write			W					
After reset			0	0	0	0	0	0
Function	Micro DMA channel 2 processed by matching bits 2 to 6 of the interrupt vector.							

Micro DMA 3 State Vector

DMA3V (007FH)

	7	6	5	4	3	2	1	0
Bit symbol			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
Read/Write			W					
After reset			0	0	0	0	0	0
Function	Micro DMA channel 3 processed by matching bits 2 to 6 of the interrupt vector.							

Note: Read-modify-write is not possible for DMA0V to DMA3V.

Figure 3.4.8 Micro DMA State Vector Register

(4) Notes

The instruction execution unit and the bus interface unit of this CPU operate independently of each other. Therefore, if the instruction used to clear the interrupt request flag of an interrupt is fetched before the interrupt is generated, it is possible that the CPU might accept the interrupt and execute the fetched instruction to clear the interrupt request flag while reading the interrupt vector. If so, the CPU would start the interrupt processing from the address "FFFF28H".

To avoid this, make sure that the instruction used to clear the interrupt request flag comes after the DI instruction. In the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing instruction and following more than one instruction are executed. When EI instruction is placed immediately after clearing instruction, an interrupt becomes enable before interrupt request flags are 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.

3.5 Functions of Ports

The TMP93CS20 has 88 bits for I/O ports.

These port pins have I/O functions for the built-in CPU and internal I/Os as well as general-purpose I/O port functions. Table 3.5.1 lists the function of each port pin. Table 3.5.2 lists I/O registers and their specifications.

Table 3.5.1 Functions of Ports

Port No.	Pin No.	Number of Pins	Direction	R	Direction Setting Unit	Pin Name for Built-in Function
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 P27	8	I/O	PU	Bit	A0 to A7/A16 to A23
Port 3	P30	1	Output	–	(Fixed)	$\overline{RD}$
	P31	1	Output	–	(Fixed)	$\overline{WR}$
	P32	1	I/O	PU	Bit	$\overline{HWR}$
	P33 to P36	4	I/O	PU	Bit	INT0 to INT3
	P37	1	I/O	PU	Bit	INT4/ $\overline{ADTRG}$
Port 4	P40	1	I/O	PU	Bit	TI4/KEY0
	P41	1	I/O	PU	Bit	TO4/KEY1
	P42	1	I/O	PU	Bit	TI6/KEY2
	P43	1	I/O	PU	Bit	TO6/KEY3
	P44	1	I/O	PU	Bit	KEY4
	P45	1	I/O	PU	Bit	KEY5
	P46	1	I/O	PU	Bit	KEY6
	P47	1	I/O	PU	Bit	KEY7
Port 5	P50 to P57	8	Input	–	(Fixed)	AN0 to AN7
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	TXD0
	P64	1	I/O	–	Bit	RXD0
	P65	1	I/O	–	Bit	SCLK0/ $\overline{CTS0}$
	P66	1	I/O	–	Bit	TI0/INT7
	P67	1	I/O	–	Bit	TO1
Port 7	P70	1	I/O	PU	Bit	TI8/INT8
	P71	1	I/O	PU	Bit	TI9/INT9
	P72	1	I/O	PU	Bit	TO8
	P73	1	I/O	PU	Bit	SCOUT
	P74	1	I/O	PU	Bit	TIA/INTA
	P75	1	I/O	PU	Bit	TIB/INTB
	P76	1	I/O	PU	Bit	TOA
	P77	1	I/O	PU	Bit	$\overline{NMI}$
Port 8	P80	1	I/O	–	Bit	TXD1
	P81	1	I/O	–	Bit	RXD1
	P82	1	I/O	–	Bit	TI2
	P83	1	I/O	–	Bit	TO3
	P84	1	I/O	–	Bit	$\overline{WAIT}$
	P85	1	I/O	–	Bit	
	P86	1	I/O	–	Bit	XT1
	P87	1	I/O	–	Bit	XT2
Port 9	P90 to P97	8	Output	–	Bit	SEG24 to SEG31
Port A	PA0 to PA7	8	Output	–	Bit	SEG32 to SEG39

PU = With programmable pull-up resistor

Table 3.5.2 I/O Registers and Specifications (1/2)

Port No.	Pin No.	Function	I/O Register		
			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 output	×	1	1
Port 2	P20 to P27	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
		A0 to A7 output	0	0	1
		A16 to A23 output	0	1	1
Port 3	P30	Output port	×	None	0
		Outputs $\overline{RD}$ only when accessing external space	1		1
		Always outputs $\overline{RD}$	0		1
	P31	Output port	×	None	0
		Outputs $\overline{WR}$ only when accessing external space	×		1
	P32 to P37	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
	P32	$\overline{HWR}$ output	×	1	1
	P33 to P36	INT0 to INT3 input	×	0	1
	P37	INT4 input	×	0	1
		$\overline{ADTRG}$ input	×	0	×
Port 4	P40 to P47	Input port (without PU)	0	0	0
		Input port (with PU)	1	0	0
		Output port	×	1	0
	P40	TI4 input port (without PU)	0	0	None
		TI4 input port (with PU)	1	0	
		KEY0 input (Note 1)	×	0	
	P42	TI6 input port (without PU)	0	0	None
		TI6 input port (with PU)	1	0	
		KEY2 input (Note 1)	×	0	
	P41	TO4 output	×	1	1
		KEY1 input (Note 1)	×	0	×
	P43	TO6 output	×	1	1
		KEY3 input (Note 1)	×	0	×
	P44 to P47	KEY4 to KEY7 input (Note 1)	×	0	None

X: Don't care, n: Corresponding port number

Note 1: When P40 to P47 are used as KEY input, KEY input is enabled by the KEYCR register.

Table 3.5.2 I/O Registers and Specifications (2/2)

Port No.	Pin No.	Function	I/O Register		
			Pn	PnCR	PnFC
Port 5	P50 to P57	Input port	×	None	
		AN0 to AN7 input (Note 2)	×		
Port 6	P60 to P67	Input port	×	0	0
		Output port	×	1	0
	P60	SCK input	×	0	×
		SCK output	×	1	1
	P61	SO output/SDA input/output	×	1	1
	P62	SI input	×	0	×
		SCL input/output	×	1	1
	P63	TXD0 output	×	1	1
	P64	RXD0 input	×	0	None
	P65	SCLK0/$\overline{\text{CTS0}}$ input	×	0	1
		SCLK0 output	×	1	1
	P66	TI0 input	×	0	×
		INT7 input	×	0	1
	P67	TO1 output	×	1	1
Port 7	P70 to P77	Input port	×	0	0
		Output port	×	1	0
	P70	TI8/INT8 input	×	0	None
	P71	TI9/INT9 input	×	0	
	P72	TO8 output	×	1	1
	P73	SCOUT output (Note 3)	×	1	None
	P74	TIA/INTA input	×	0	
	P75	TIB/INTB input	×	0	
	P76	TOA output	×	1	1
	P77	$\overline{\text{NMI}}$ input	×	×	1
Port8	P80 to P87	Input port	×	0	0
		Output port	×	1	0
	P80	TXD1 output	×	1	1
	P81	RXD1 input	×	0	None
	P82	TI2 input	×	0	
	P83	TO3 output	×	1	1
	P84	$\overline{\text{WAIT}}$ input	×	0	None
	P86, P87	Input port	×	0	
		Output port (Note 4)	×	1	
		XT1, XT2 (Note 5)	×	0	
Port9	P90 to P97	Output port	×	None	0
		SEG24 to SEG31 output	×		1
PortA	PA0 to PA7	Output port	×	None	0
		SEG32 to SEG39 output	×		1

X: Don't care, n: Corresponding port number

Note 2: The channnel for AD input is selected by ADMOD1<ADCH2:0> for P50 to P57 pins.

Note 3: When using P73 as SCOUT, the CKOCR must be set to enable it.

Note 4: When using P86 to P87 as output ports, output goes through the open-drain buffer.

Note 5: When using P86 to P87 as XT1 to XT2, the SYSCR0 must be set to enable oscillating.

Resetting makes the port pins listed below function as general-purpose I/O ports.

I/O pins programmable for input or output are then set to function as input ports, except for P86/XT1 and P87/XT2.

A program is needed to set port pins for built-in functions.

3.5.1 Port 0 (P00 to P07)

Port 0 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis using the control register P0CR. Resetting resets all bits of P0CR to 0 and sets port 0 to input mode.

In addition to functioning as a general-purpose I/O port, port 0 also functions as an address data bus (AD0 to AD7). To access external memory, port 0 functions as an address data bus (AD0 to AD7), and all bits of the control register P0CR are cleared to 0.

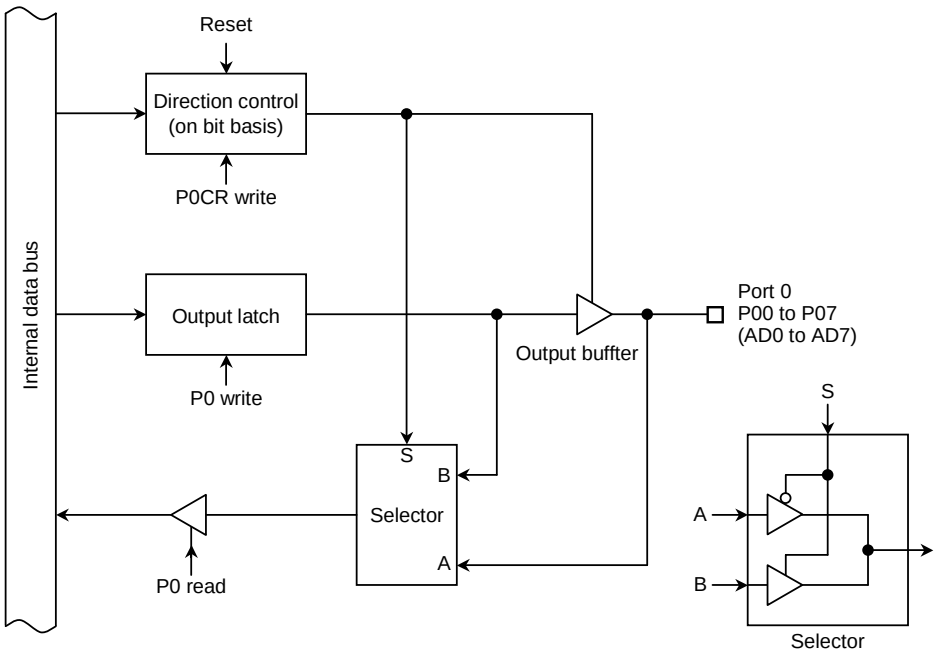


Figure 3.5.1 Port 0

3.5.2 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis using control register P1CR and function register P1FC. Resetting resets all bits of output latch P1, 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 also functions as an address data bus (AD8 to AD15) or an address bus (A8 to A15).

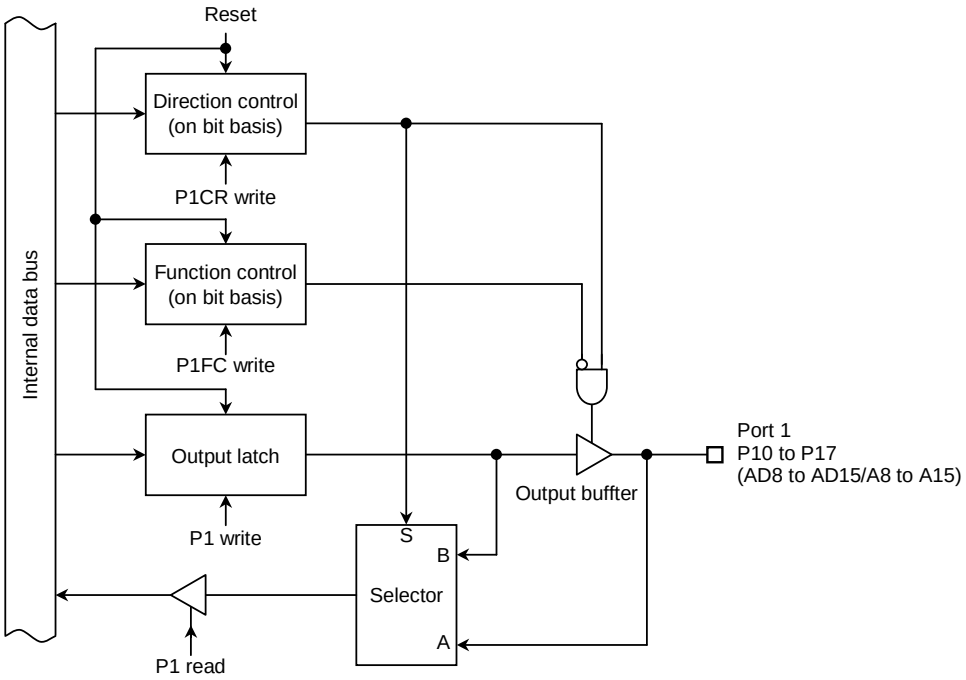


Figure 3.5.2 Port 1

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	Input mode (Output latch register becomes 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	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:Input 1:Output (when externally accessed, port 0 becomes AD7 to AD0 and P0CR is cleared to 0.)							

Port 0 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	Input mode (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	R/W							
After reset	0	0	0	0	0	0	0	0
Function	<<See table below>>							

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	R/W							
After reset	0	0	0	0	0	0	0	0
Function	<<See table below>>							

Port 1 function setting

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

Note: <P1XF> is bit X in register P1FC; <P1XC> is bit X in register P1CR.

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

Figure 3.5.3 Registers for Ports 0 and 1

3.5.3 Port 2 (P20 to P27)

Port 2 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis using the control register P2CR and function register P2FC. Resetting resets all bits of output latch P2 to P1, control register P2CR and function register P2FC to 0. It also sets port 2 to input mode and connects a pull-up resistor.

In addition to functioning as a general-purpose I/O port, port 2 also functions as an address bus (A0 to A7) or (A16 to A23). To use port 2 as an address bus, write 0 to the output latches to turn off the programmable pull-up resistors.

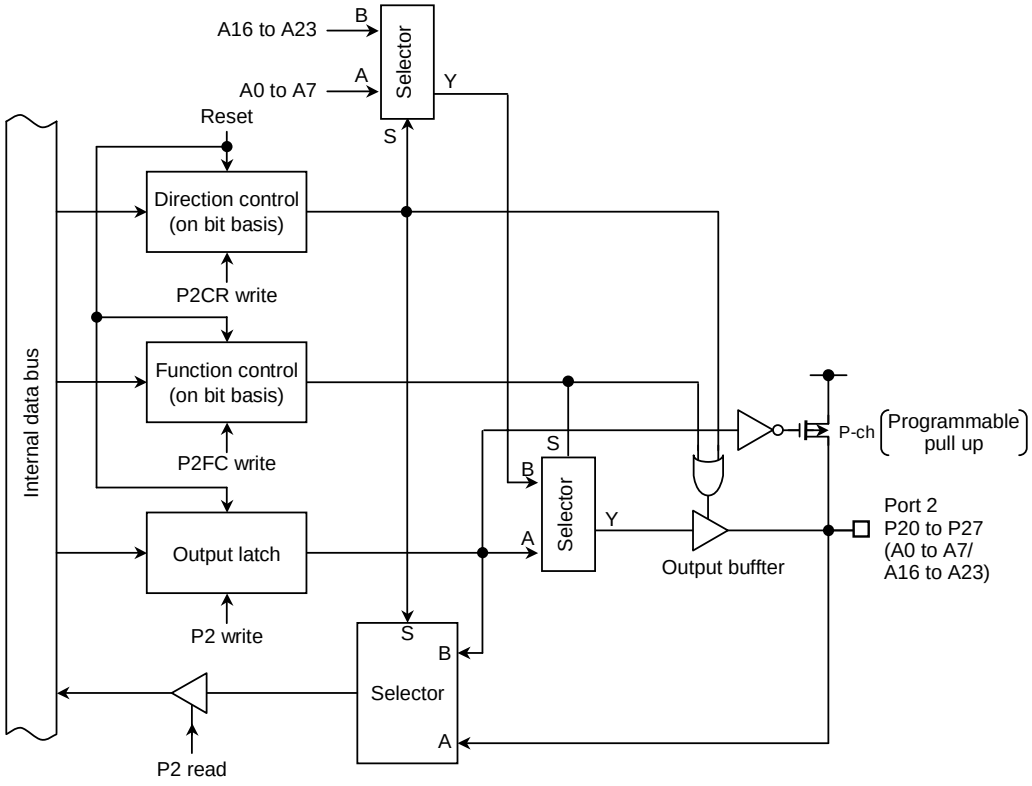


Figure 3.5.4 Port 2

Port 2 Register

P2 (0006H)

	7	6	5	4	3	2	1	0
Bit symbol	P27	P26	P25	P24	P23	P22	P21	P20
Read/Write	R/W							
After reset	Input mode (Output latch register is set to 1.)							

Port 2 Control Register

P2CR (0008H)

	7	6	5	4	3	2	1	0
Bit symbol	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	<<See table below>>							

Port 2 Function Register

P2FC (0009H)

	7	6	5	4	3	2	1	0
Bit symbol	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	<<See table below>>							

Port 2 function setting

P2CR<P2XC> \ P2FC<P2XF>	0	1
0	Input port	Address data bus (A7 to A0)
1	Output port	Address bus (A23 to A16)

Note: <P2XF> is bit X in register P2FC; <P2XC> is bit X in register P2CR. To set as an address bus A23 to A16, set P2FC after setting P2CR.

Note 1: Read-modify-write is prohibited for registers P2CR and P2FC.

Note 2: When port P2 is used in the input mode, P2 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up resistor.

Figure 3.5.5 Registers for Port 2

3.5.4 Port 3 (P30 to P37)

Port 3 is an 8-bit general-purpose I/O port.

I/O can be set on a bit basis, but note that P30 and P31 are used for output only. I/O is set using control register P3CR and function register P3FC. Resetting sets all bits of output latch P3 to 1, and control register P3CR (Bits 0 and 1 are unused) and function register P3FC to 0. Resetting also outputs 1 from P30 and P31, sets P32 to P37 to input mode, and connects a pull-up resistor.

In addition to functioning as a general-purpose I/O port, port 3 also functions as an I/O for the CPU's control/status signal, an input for INT0 to INT4 and an analog conversion external trigger input $\overline{\text{ADTRG}}$.

When the P30 pin is defined as $\overline{\text{RD}}$ signal output mode (<P30F> = 1), in the external ROM is used, clearing the output latch register <P30> to 0 outputs the $\overline{\text{RD}}$ strobe (Used for the pseudo-static RAM) from the P30 pin even when the internal address area is accessed.

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

(1) P30 ($\overline{\text{RD}}$), P31 ($\overline{\text{WR}}$)

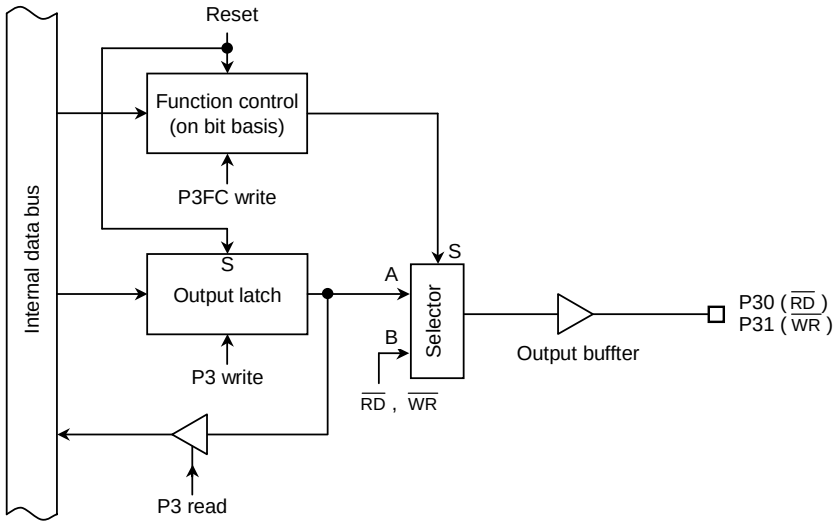
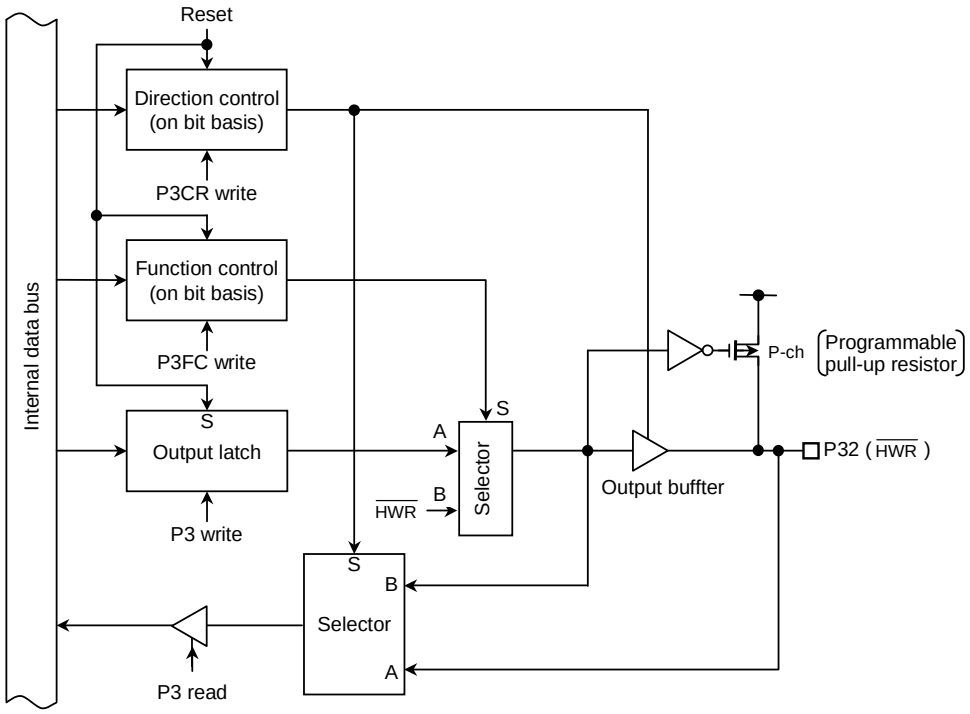


Figure 3.5.6 Port 3 (P30 and P31)

(2) P32 ($\overline{HWR}$)



(3) P33 (INT0)

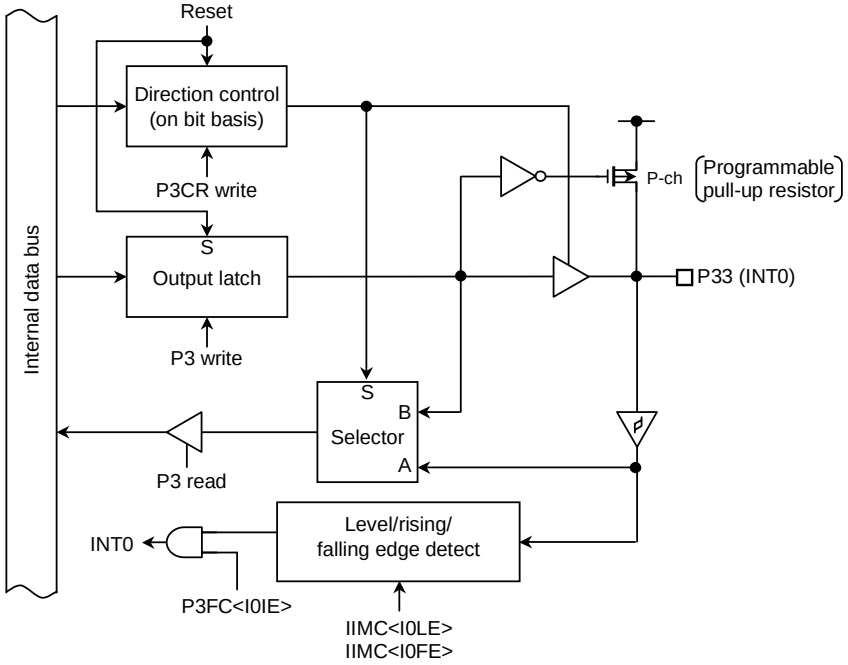


Figure 3.5.7 Port 3 (P32 and P33)

(4) P34 (INT1), P35 (INT2), P36 (INT3), P37 (INT4/$\overline{\text{ADTRG}}$)

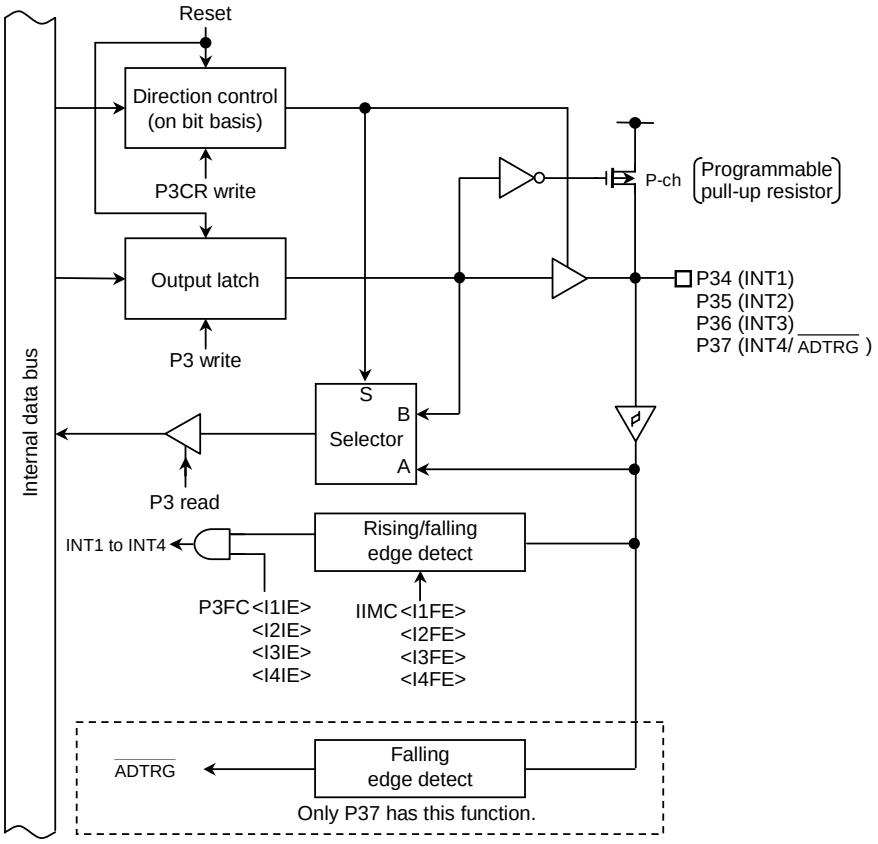


Figure 3.5.8 Port3 (P34 to P37)

Port 3 Register

P3 (0007H)

	7	6	5	4	3	2	1	0
Bit symbol	P37	P36	P35	P34	P33	P32	P31	P30
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
	Input mode (pulled up)						Output mode	

Port 3 Control Register

P3CR (000AH)

	7	6	5	4	3	2	1	0
Bit symbol	P37C	P36C	P35C	P34C	P33C	P32C		
Read/Write	W							
After reset	0	0	0	0	0	0		
Function	0: Input 1: Output							

I/O setting

0	Input
1	Output

Port 3 Function Register

P3FC (000BH)

	7	6	5	4	3	2	1	0
Bit symbol	I4IE	I3IE	I2IE	I1IE	I0IE	P32F	P31F	P30F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: INT4	0: Port 1: INT3	0: Port 1: INT2	0: Port 1: INT1	0: Port 1: INT0	0: Port 1: $\overline{HWR}$	0: Port 1: $\overline{WR}$	0: Port 1: $\overline{RD}$

0	INT0 Input disable (Only P33)
1	INT0 Input enable

0	INT1 Input disable
1	INT1 Input enable

0	INT2 Input disable
1	INT2 Input enable

0	INT3 Input disable
1	INT3 Input enable

0	INT4 Input disable
1	INT4 Input enable

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

<P30> / <P30F>	0	1
0	0 output	1 output
1	Always $\overline{RD}$ output (for pseudo SRAM)	$\overline{RD}$ output only for external access

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

<P31> / <P31F>	0	1
0	0 output	1 output
1	$\overline{WR}$ output only for external access	

$\overline{HWR}$ setting

P3FC<P32F>	1
P3CR<P32C>	1

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

Note 2: When port P3 is used in the input mode, the P3 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up resistor.

Note 3: The INT0 to INT4 pins can be also used to release the standby state. When these pins are not used for releasing the standby, port P3 maintains the port function by setting the P3 register to 0 during standby.

Figure 3.5.9 Registers for Port 3

3.5.5 Port 4 (P40 to P47)

Port 4 is an 8-bit general-purpose I/O port. I/O can be set on a bit basis. Resetting sets port 4 as an input port and connects a pull-up resistor. It also sets all bits of the output latch register P4 to 1. In addition to functioning as a general-purpose I/O port, port 4 also functions a 16-bit timer input clock pin, a 16-bit timer output pin, and a key-on wakeup input signal. Writing 1 in the appropriate bit of the function register P4FC enables a timer output signal. Resetting resets P4CR and P4FC to 0, and all bits to input ports.

(1) Port 40, 42 (TI4/KEY0, TI6/KEY2)

In addition to functioning as a general-purpose I/O port, port 40 also functions as an event count input TI4 of the timer 4 and a key-on wakeup input KEY0. Port 42 functions as a general-purpose I/O port, an event count input TI6 of the timer 6, and a key-on wakeup input KEY2.

When using key input KEY0, KEY2:

When a general-purpose I/O port or a timer event count input is used during using the key input KEY0 and KEY2, an interrupt request occurs at the falling edge of these I/O signals.

When using timer event count input TI4, TI6:

When a general-purpose I/O port or the key-on input KEY0 and KEY2 are used during event count operating due to TI4 and TI6, timer event counting is executed by these I/O signals.

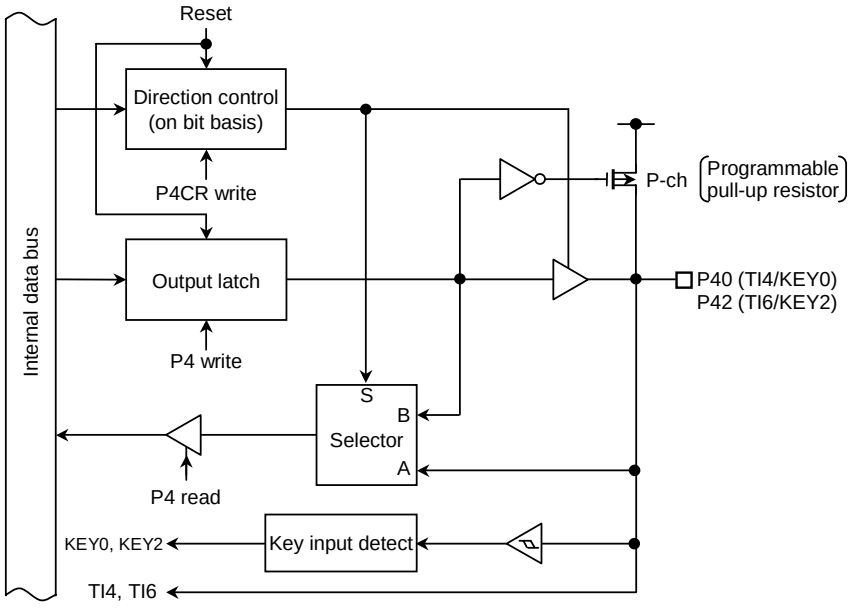


Figure 3.5.10 Port 4 (P40 and P42)

(2) Port 41, 43 (TO4/KEY1, TO6/KEY3)

Port 41 is a general-purpose I/O port, and is also used as an output TO4 and KEY1 of a 16-bit timer 4. Port 43 is a general-purpose I/O port, and is also used as an output TO6 and KEY3 of a 16-bit timer 6.

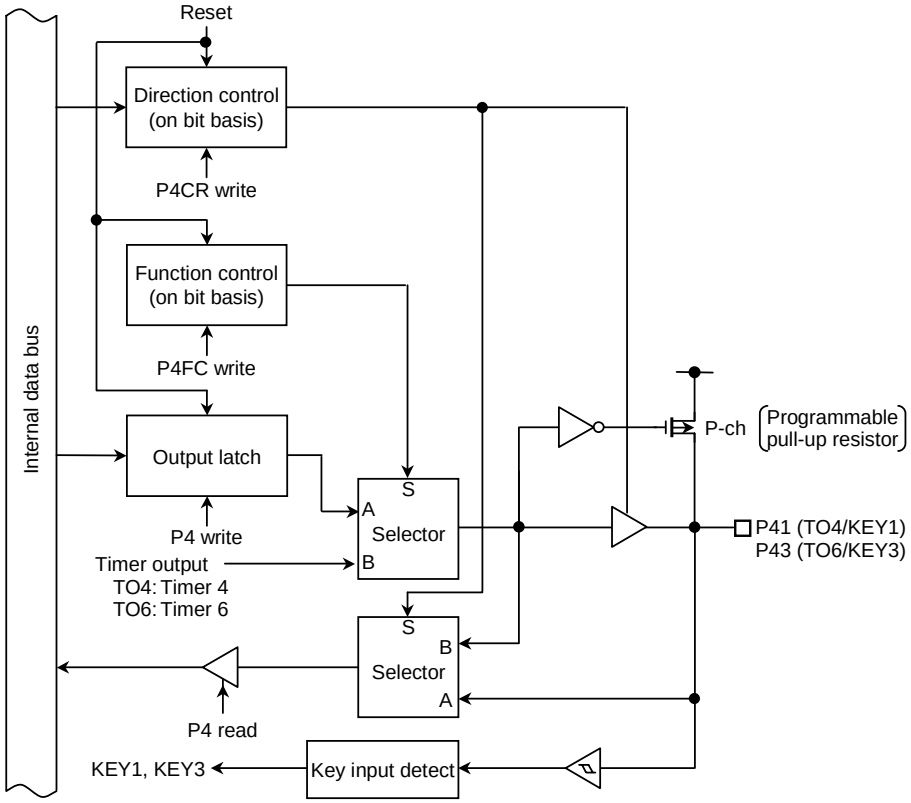


Figure 3.5.11 Port 4 (P41 and P43)

(3) Port 44 to 47 (KEY4 to KEY7)

Port 44 to 47 are general-purpose I/O ports, and are also used as KEY4 to KEY7.

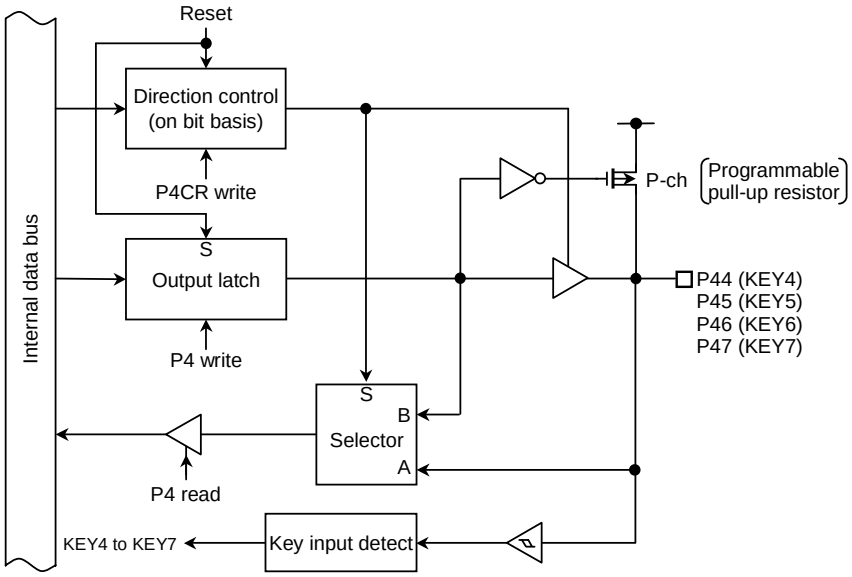


Figure 3.5.12 Port 4 (P44 to P47)

Port 4 Register

P4 (000CH)

	7	6	5	4	3	2	1	0
Bit symbol	P47	P46	P45	P44	P43	P42	P41	P40
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
	Input mode (Pull up)							

Port 4 Control Register

P4CR (000EH)

	7	6	5	4	3	2	1	0
Bit symbol	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 4 Function Register

P4FC (0010H)

	7	6	5	4	3	2	1	0
Bit symbol					P43F		P41F	
Read/Write					W		W	
After reset					0		0	
Function					0: Port 1: TO6		0: Port 1: TO4	

Setting P41 as timer out 4 output

P4FC<P41F>	1
P4CR<P41C>	1

Setting P43 as timer out 6 output

P4FC<P43F>	1
P4CR<P43C>	1

Note 1: Read-modify-write is prohibited for registers P4CR and P4FC.

Note 2: When port P4 is used in the input mode, the P4 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up resistor.

Note 3: P40/TI4 and P42/TI6 pins do not pave a register changing it from port to function. For example, when they are used as input ports, the incoming signal is input to 16-bit timer 4 and 6 as timer input 4 and 6.

Figure 3.5.13 Registers for Port 4

3.5.6 Port 5 (P50 to P57)

Port 5 is an 8-bit input port, also used as an analog input pin for the internal AD converter.

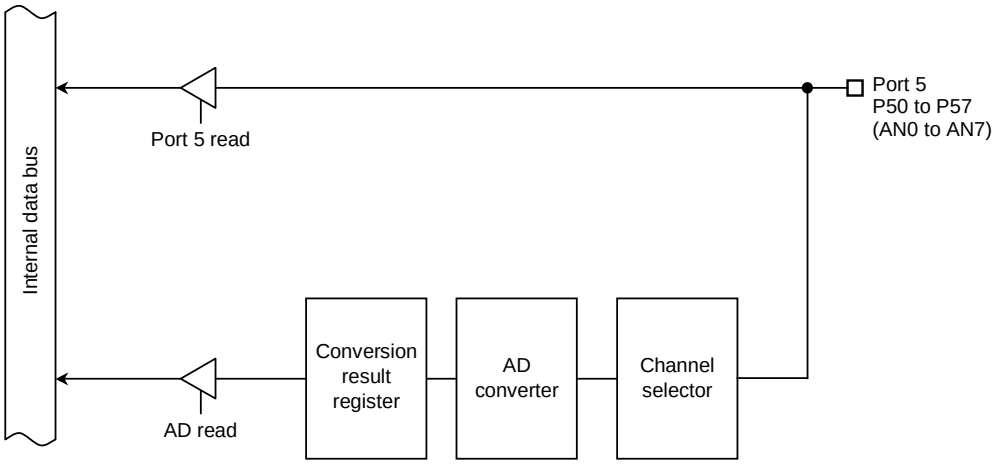


Figure 3.5.14 Port 5

Port 5 Register

		7	6	5	4	3	2	1	0
P5 (000DH)	Bit symbol	P57	P56	P55	P54	P53	P52	P51	P50
	Read/Write	R							
	After reset	Input mode							

Note: The input channel selection of the AD converter is set by AD converter mode register ADMOD1.

Figure 3.5.15 Registers for Port 5

3.5.7 Port 6 (P60 to P67)

Port 60 to 65 are 8-bit general-purpose I/O ports. I/O can be set on a bit basis.

In addition to functioning as general-purpose I/O ports, port 60 to 65 also function as serial bus interface I/O signals, 8-bit timer I/O signals. and external interrupt INT7 input signals. Setting 1 in the appropriate bit of the port 6 function register P6FC enables each functions.

To use P61 and P62 as serial bus interface I/O pins (I^2C bus mode), set these ports to open-drain output mode. (P6FC<P61F, P62F> = 11)

Resetting sets all bits of the output latch P6 to 1, initializes all bits of P6CR and P6FC to 0, and sets all bits of the port 6 as an input port.

(1) Port 60, 65 (SCK, $\overline{CTS0}$ /SCLK0)

In addition to functioning as an I/O port, port 60 also functions as a SCK I/O pins when the serial bus interface is in the SIO mode. Port 65 is used as a $\overline{CTS}$ input pin of the serial channel 0, or a SCLK I/O pin.

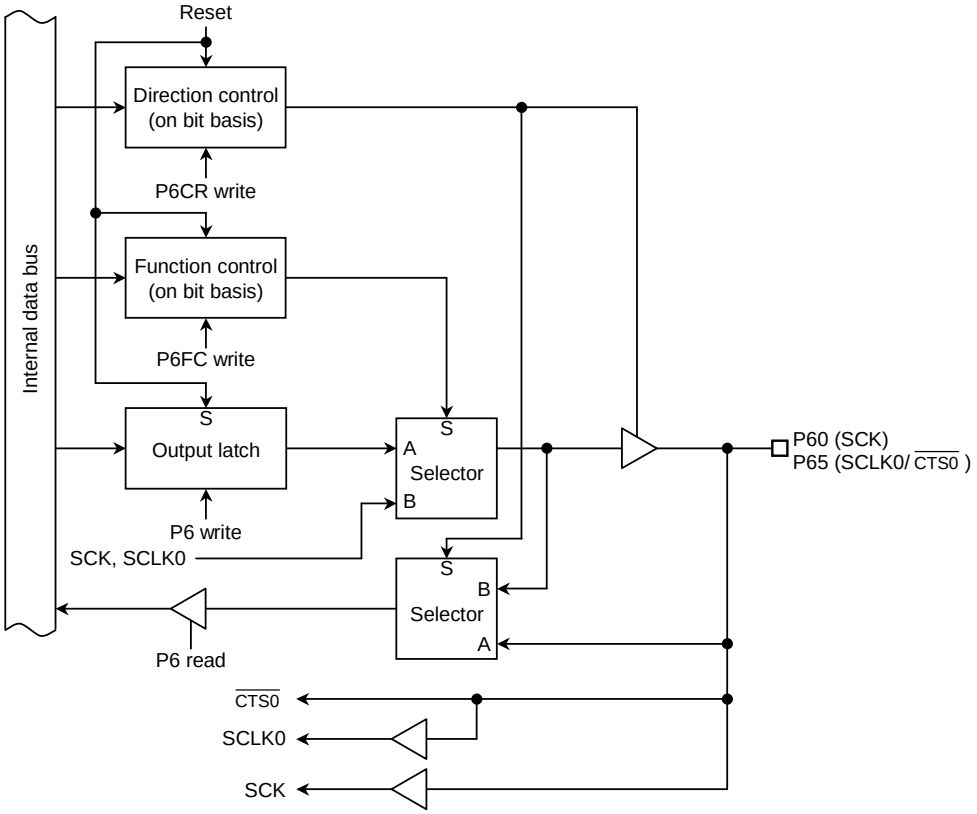


Figure 3.5.16 Port 6 (P60 and P65)

(2) Port 61, 62, and 63 (SO/SDA, SI/SCL, TXD0)

In addition to functioning as a general-purpose I/O port, port 61 also functions as a SO output pin when the serial bus interface is in the SIO mode, and a SDA I/O pin in the I²C mode. Port 62 is used as a SI input pin when the serial bus interface is in the SIO mode, and SCL I/O pin in the I²C mode. Port 63 is used as a TXD output pin of the serial channel 0.

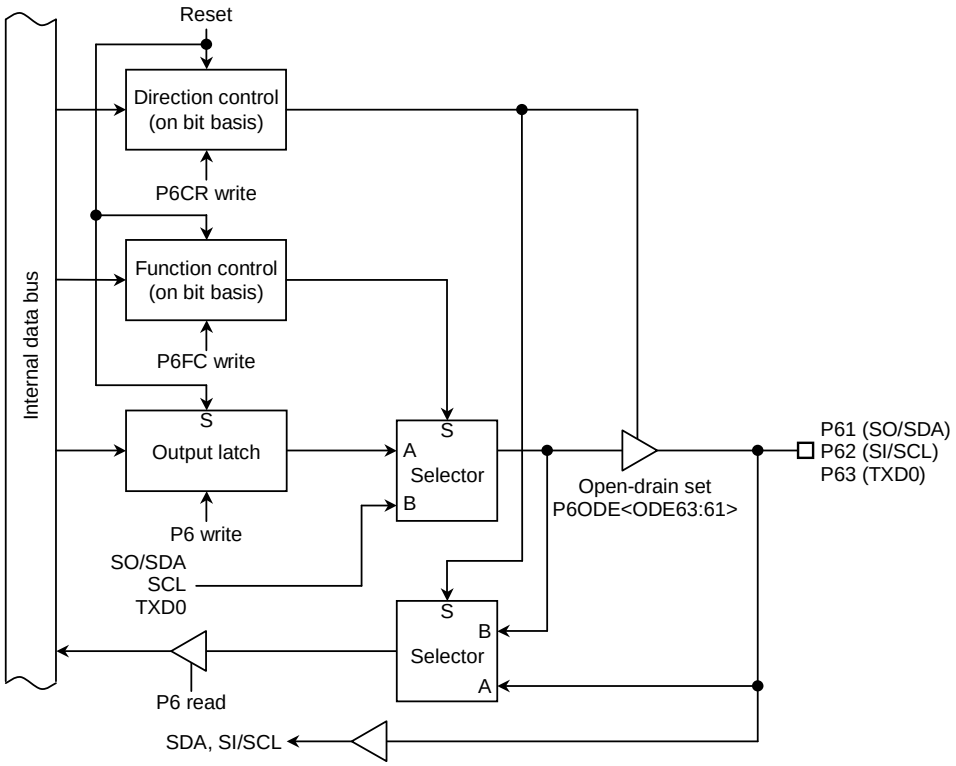


Figure 3.5.17 Port 6 (P61 to P63)

(3) Port 64 (RXD0)

In addition to functioning as a general-purpose I/O port, port 64 also functions as a RXD input pin of the serial channel 0.

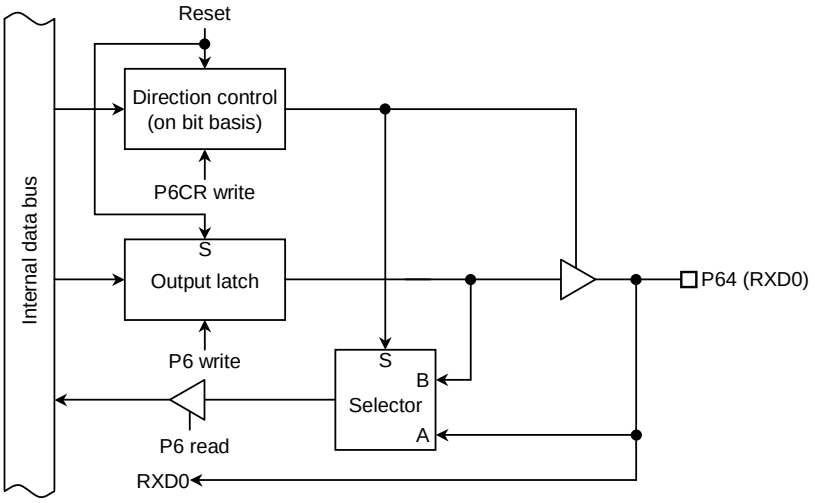


Figure 3.5.18 Port 64

(4) Port 66, 67 (TI0/INT7, TO1)

In addition to functioning as a general-purpose I/O port, port 66 also functions as an event count input TI0 of the timer 0 and an external interrupt request input INT7. Port 67 is an general-purpose I/O port, and is also used as an output TO1 of the timer 0 and 1.

When using INT7 interrupt:

When INT7 interrupt function is used, and a general-purpose I/O port or a timer event count input is used, an interrupt request occurs at the rising and the falling edge of these I/O signals.

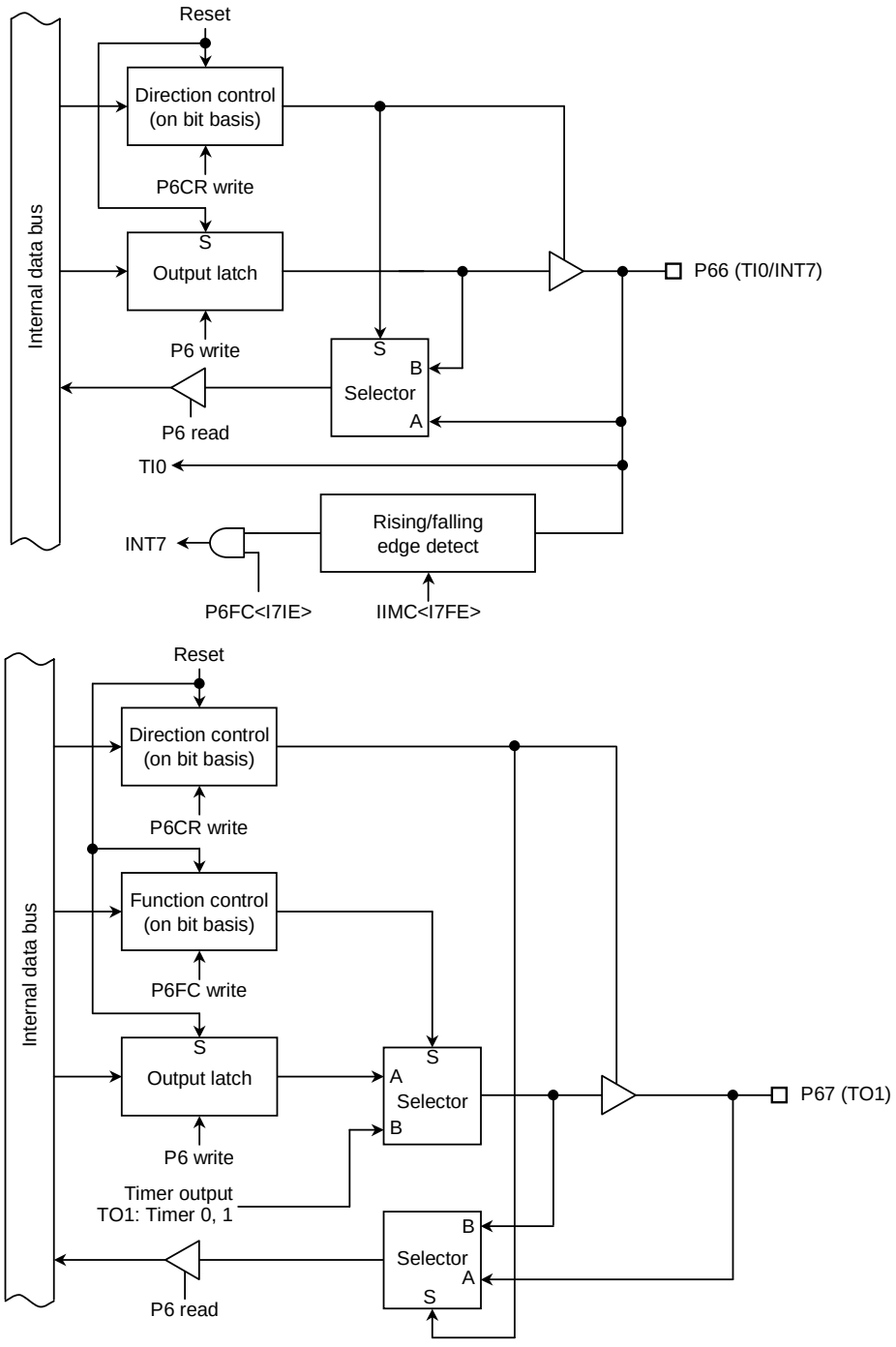


Figure 3.5.19 Port 6 (P66 and P67)

Port 6 Register

P6 (0012H)

	7	6	5	4	3	2	1	0
Bit symbol	P67	P66	P65	P64	P63	P62	P61	P60
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
	Input mode							

Port 6 Control Register

P6CR (0014H)

	7	6	5	4	3	2	1	0
Bit symbol	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 6 Function Register

P6FC (0016H)

	7	6	5	4	3	2	1	0
Bit symbol	P67F	I7IE	P65F		P63F	P62F	P61F	P60F
Read/Write	W				W			
After reset	0	0	0		0	0	0	0
Function	0: Port 1: TO1	0: Port 1: INT7	0: Port 1: SCLK0		0: Port 1: TXD0	0: Port 1: SCL/SI	0: Port 1: SDA/SO	0: Port 1: SCK

0	INT7 input disable
1	INT7 input enable

Setting P63 as TXD0 output

P6FC<P63F>	1
P6CR<P63C>	1

Setting P60 as SCK output

P6FC<P60F>	1
P6CR<P60C>	1

Setting P67 as TO1 output

P6FC<P67F>	1
P6CR<P67C>	1

Setting P65 as SCLK output

P6FC<P65F>	1
P6CR<P65C>	1

Setting P61 as SDA I/O or SO output (Note 2)

P6FC<P61F>	1
P6CR<P61C>	1

P62 function setting (Note 2)

<P62C> / <P62F>	0	1
0	Port input	Port output
1	SI Input	SCL I/O

Note 1: Read-modify-write is prohibited for register P6CR and P6FC.

Note 2: To use P61 and P62 as serial bus interface I/O pins (I^2C bus mode), set P6ODE<ODE62:61> to 11 and these ports to open-drain output mode.

Note 3: P64/RXD0 pin does not have a register changing if from port to function. For example, when it is used as an input port, this pin is set to the serial channel 0 as a serial receive data.

Figure 3.5.20 Registers for Port 6

3.5.8 Port 7 (P70 to P77)

Port 7 is a 8-bit general-purpose I/O port. I/O can be set on a bit basis. Resetting sets port 7 as an input port and connects a pull-up resistor. In addition to functioning as a general-purpose I/O port, port 7 also functions as an input for 16-bit timer 8 and A clocks, an output for 16-bit timer F/F 8 and A output, internal clock output, and a non-maskable interrupt input. Resetting resets the function register P7CR, P7FC value to 0, and sets all bits to input ports.

(1) P70 (TI8/INT8), P71 (TI9/INT9), P72 (TO8), P74 (TIA/INTA), P75 (TIB/INTB), P76 (TOA)

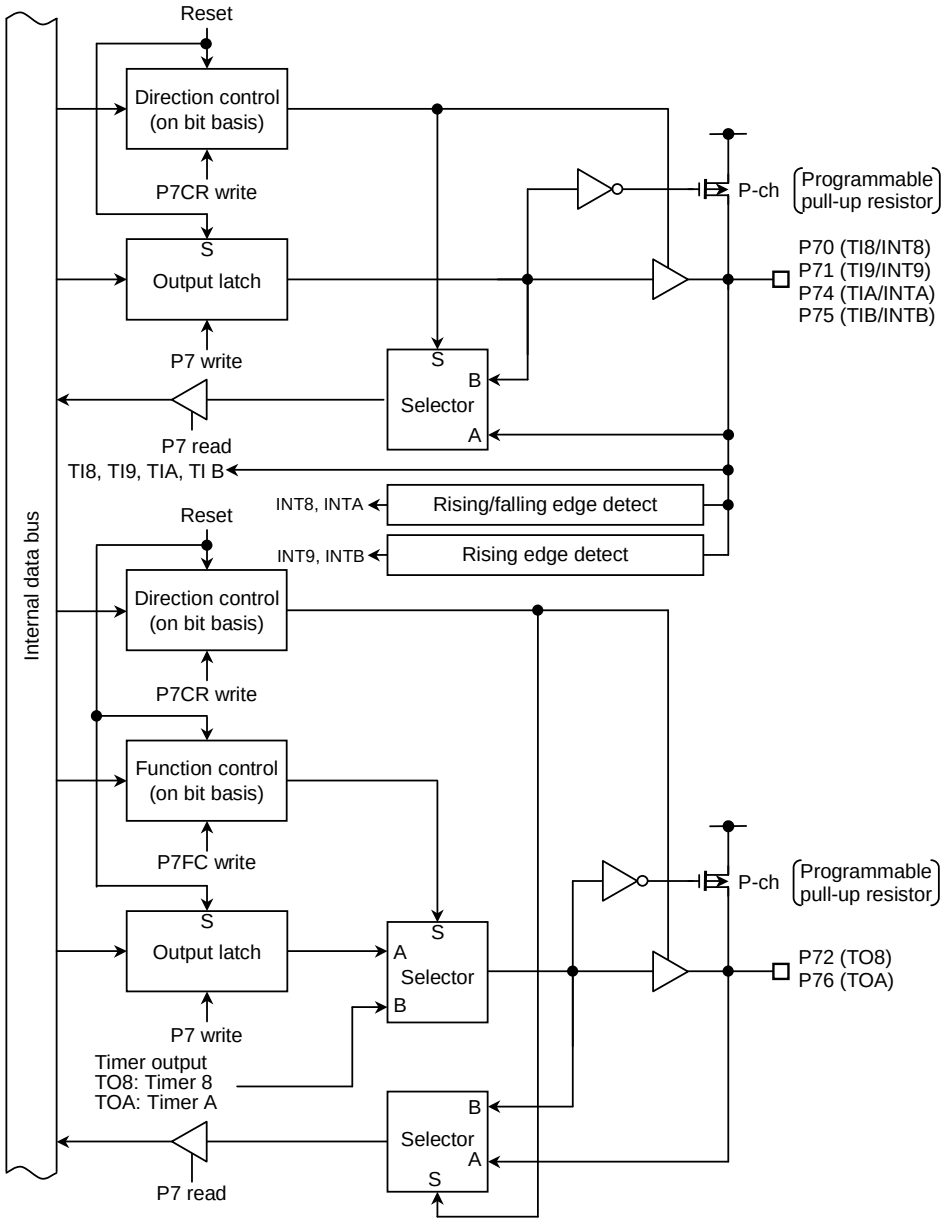


Figure 3.5.21 Port 7 (P70, P71, P72, P74, P75, and P76)

When using INT8 to INTB interrupts:

When a general-purpose I/O port or a timer event count input is used during using INT8 to INTB interrupt functions, an interrupt request occurs at the rising and the falling edges of these I/O signals.

When using Timer event count input TI8 to TIB:

When a general-purpose I/O port or INT8 to INTB interrupts are used during event count operating due to TI8 and TIB, timer event counting is executed by these I/O signals.

In addition to functioning as a general-purpose I/O port, port 73 also functions as an internal clock output. The output clock is f_{FPH} or f_{SYS}, which is selected by the clock output control register CKOCR<SCOSEL>. Writing 1 in P7CR<P73C> and CKOCR<SCOEN> enables the internal clock output function.

(2) P73 (SCOUT)

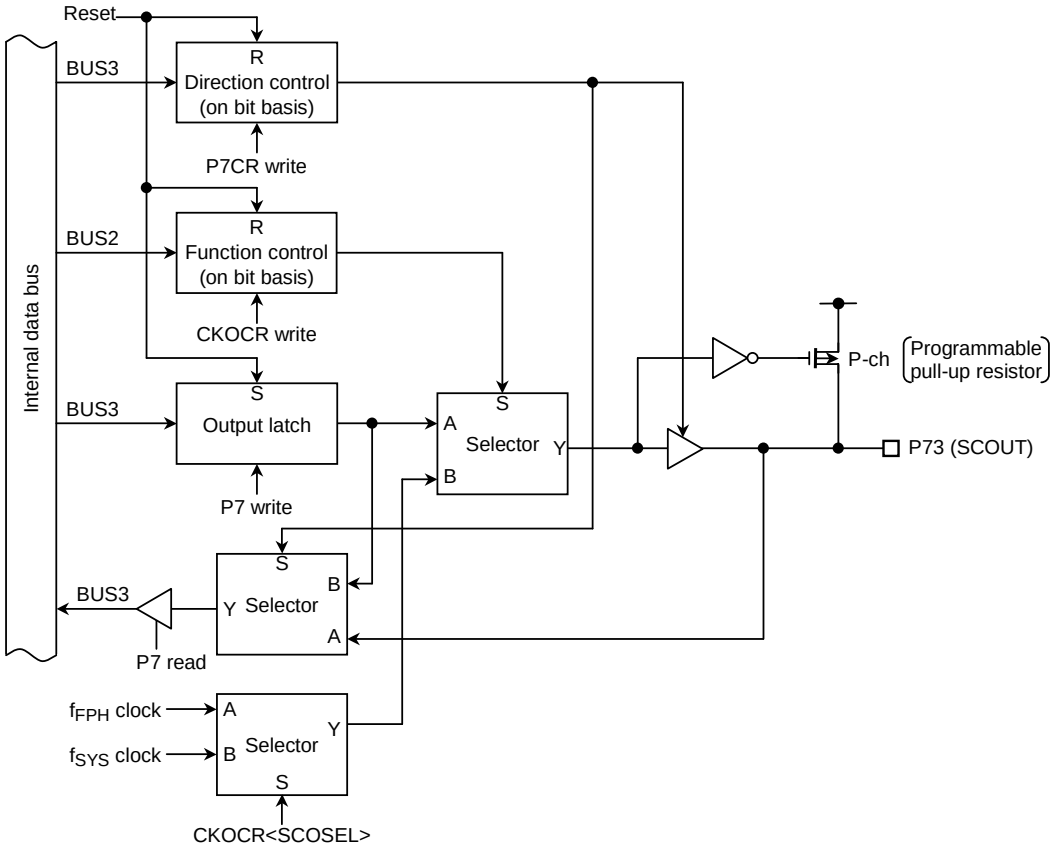


Figure 3.5.22 Port 73

(3) P77 ($\overline{NMI}$)

P77 is a general-purpose I/O port, and is also used as a non-maskable interrupt ($\overline{NMI}$) input pin. The $\overline{NMI}$ pin is set by the function register P7FC<NMIC>. Writing NMIC to 1 sets $\overline{NMI}$ input pin. This bit is prior to the value set in the I/O control register (P7CR). The $\overline{NMI}$ in is set one time only. Changing from the $\overline{NMI}$ pin to the general-purpose port is disabled. Resetting resets NMIC to 0, and sets to the general-purpose port mode.

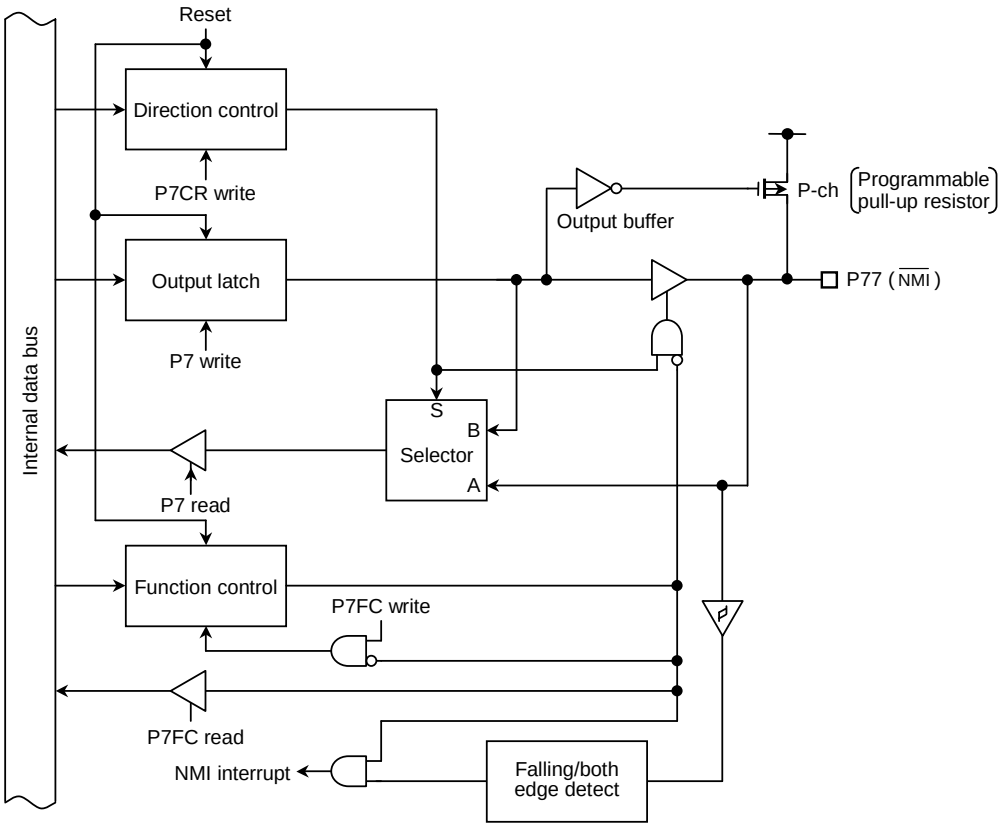


Figure 3.5.23 Port 77

Port 7 Register

P7 (0013H)

	7	6	5	4	3	2	1	0
Bit symbol	P77	P76	P75	P74	P73	P72	P71	P70
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
	Input mode (Pull up)							

Port 7 Control Register

P7CR (0015H)

	7	6	5	4	3	2	1	0
Bit symbol	P77C	P76C	P75C	P74C	P73C	P72C	P71C	P70C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 7 Function Register

P7FC (0017H)

	7	6	5	4	3	2	1	0
Bit symbol	NMIC	P76F				P72F		
Read/Write	R/W	W				W		
After reset	0	0				0		
Function	0: Port 1: $\overline{NMI}$	0: Port 1: TOA				0: Port 1: TO8		

Setting P72 as timer out 8 output

P7FC<P72F>	1
P7CR<P72C>	1

Setting P76 as timer out A output

P7FC<P76F>	1
P7CR<P76C>	1

Setting P77 as $\overline{NMI}$ input

P7CR <P77C> \ P7FC <NMIC>	0	1
0	Input	$\overline{NMI}$
1	Output	$\overline{NMI}$

Note 1: Read-modify-write is prohibited for registers P7CR and P7FC.

Note 2: When port P7 is used in the input mode, the P7 register controls the built-in pull-up resistor. Read-modify-write is prohibited in the input mode or the I/O mode, as it may affect the states of the pull-up resistors.

Note 3: The P7x/TIx pin does not have a register changing it from port to function. For example, when it is used as an input port, the incoming signal is input to 16-bit timer as a timer input.

Figure 3.5.24 Registers for Port 7

3.5.9 Port 8 (P80 to P87)

- Port 80 to 85

  Port 80 to 85 are 6-bit general-purpose I/O ports. I/O can be set on a bit basis.

  Resetting sets port 80 to 85 as input ports. All bits of the output latch register are set to 1. In addition to functioning as I/O ports, port 80 to 85 also function as I/Os for the serial channel 2 and I/Os for an 8-bit timer. Writing 1 in the corresponding bit of the port 8 function register P8FC enables those functions. Resetting resets P8CR and P8FC to 0 and sets all bits to input ports.

- Port 86 to 87

  Port 86 to 87 are 2-bit I/O ports. I/O can be set on a bit basis. When using as output ports, port 86 to 87 are set to open-drain outputs.

  Resetting sets the output latch register and the control register value to 1, and sets the high impedance output.

  In addition to functioning as I/O ports, port 86 to 87 also function as low-frequency oscillator serial pins (XT1 and XT2). These ports are also used as a dual clock mode by setting of the system clock control register SYSCR0 and SYSCR1.

(1) Port 80, 83 (TXD1, TO3)

In addition to functioning as I/O ports, port 80 and 83 also function as TXD1 output pin of the serial channel 1 and output TO3 pin of an 8-bit timer.

These ports are also used as the programmable open-drain mode.

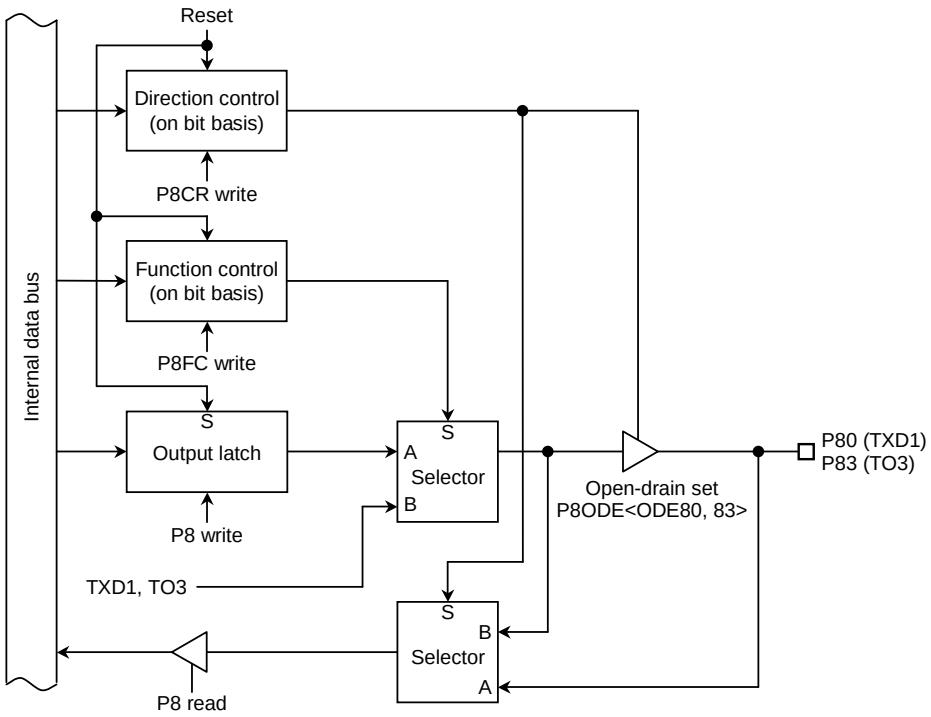


Figure 3.5.25 Port 8 (P80 and P83)

(2) Port 81, 82 (RXD1, TI2)

In addition to functioning as I/O ports, port 81 and 82 also function as RXD1 input pin of the serial channel 1 and input TI2 pin of an 8-bit timer.

These ports are also used as the programmable open-drain mode.

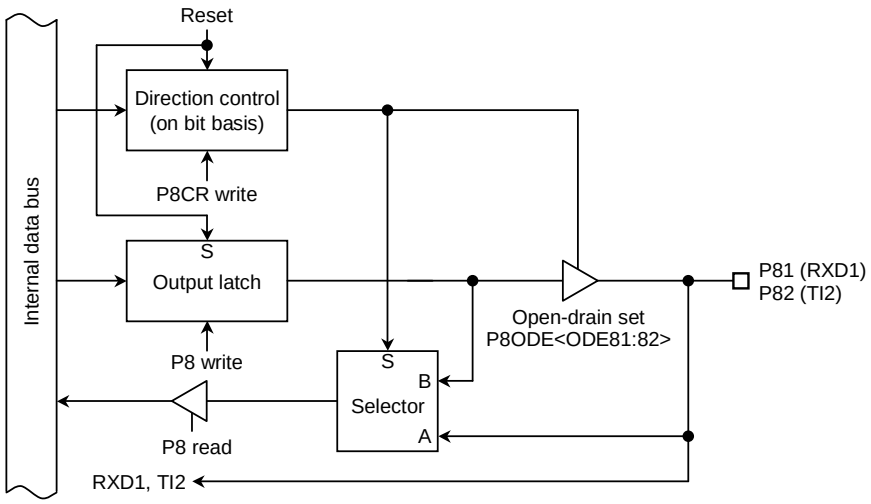


Figure 3.5.26 Port 8 (P81 and P82)

(3) Port 84, 85

Port 84 and 85 are general-purpose I/O ports. These ports are used as the programmable open-drain mode. Port 84 also functions as a bus wait request pin.

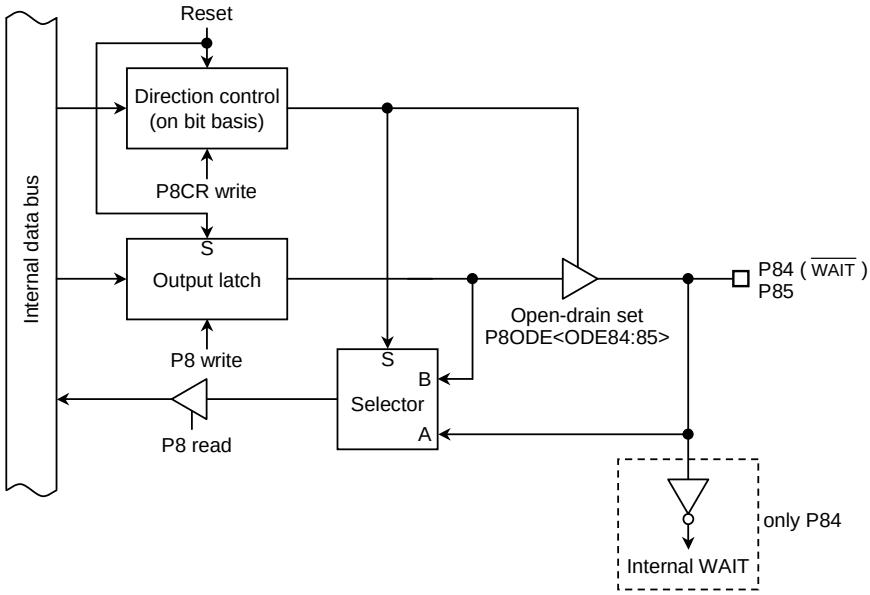


Figure 3.5.27 Port 8 (P84 and P85)

(4) Port 86 (XT1), 87 (XT2)

In addition to functioning as I/O ports, port 86 and 87 also function as low-frequency oscillator serial pins.

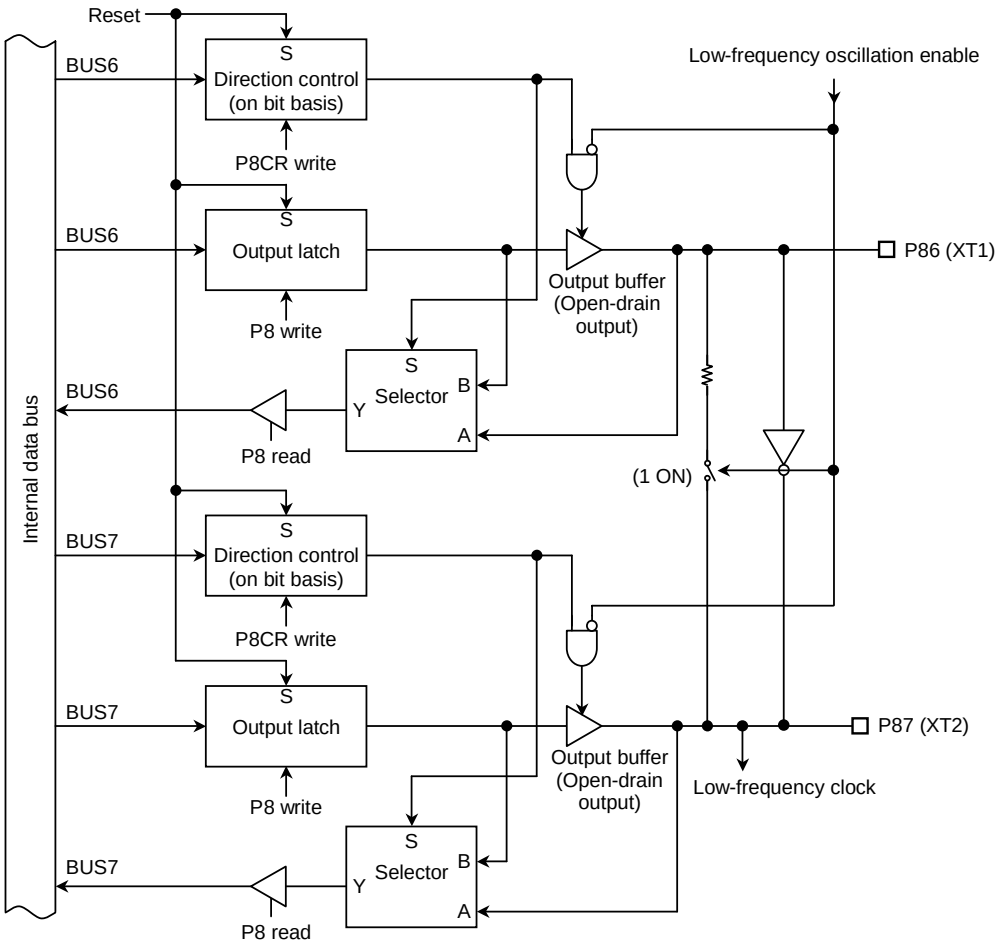


Figure 3.5.28 Port 8 (P86 and P87)

Port 8 Register

P8 (0018H)

	7	6	5	4	3	2	1	0
Bit symbol	P87	P86	P85	P84	P83	P82	P81	P80
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
	Output mode		Input mode					

Port 8 Control Register

P8CR (001AH)

	7	6	5	4	3	2	1	0
Bit symbol	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
Read/Write	W							
After reset	1	1	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 8 I/O setting

0	Input
1	Output

Port 8 Function Register

P8FC (001CH)

	7	6	5	4	3	2	1	0
Bit symbol					P83F			P80F
Read/Write					W			W
After reset					0			0
Function					0: Port 1: TO3			0: Port 1: TXD1

Setting P80 as TXD1 output

P8FC <P82F>	1
P8CR <P82C>	1

Setting P83 as timer out 3 output

P8FC <P83F>	1
P8CR <P83C>	1

Note 1: Read-modify-write is prohibited for registers P8CR and P8FC.

Note 2: The P81/RXD1, and P82/T12 pins do not have a register changing them from port to function. For example, when they are used as an input port, the incoming signal is input to the 8-bit timer as timer input.

Note 3: To set the P80 to P85 to open drain, write 1 in bit0 to 5 of the P8ODE register.

Note 4: Notes on using low-frequency oscillation circuit. To connect a low-frequency resonator to port 86, 87, it is necessary to set the following procedures to reduce the consumption power supply.

(Connecting to a resonator)

Set P8CR<P86C, P87C> = 11, P8<P86:87> = 00

(Connecting to a oscillator)

Set P8CR<P86C, P87C> = 11, P8<P86:87> = 10

Note 5: If you can enter STOP mode on either of the following conditions, fix port 8 to VCC or GND.

- When setting the pin control to “I/O off” at STOP mode.
- When setting the pin control to “keep the condition prior to halt” at STOP mode and also when port 8 is input mode.

Figure 3.5.29 Registers for Port 8

3.5.10 Port 9 (P90 to P97), Port A (PA0 to PA7)

Port 9 and port A are 8-bit open-drain output ports. In addition to functioning as output ports, port 9 and port A also function as segment outputs of the LCD driver. Writing 1 in the corresponding bit of the port 9 function register P9FC and port A function register PAFC enable s this function.

Resetting resets all bits of the output latch P9 to 0, initializes all bits of the PA to 1 and all bits of P9FC and PAFC to 0, and sets all bits of P9 and PA ports to output ports.

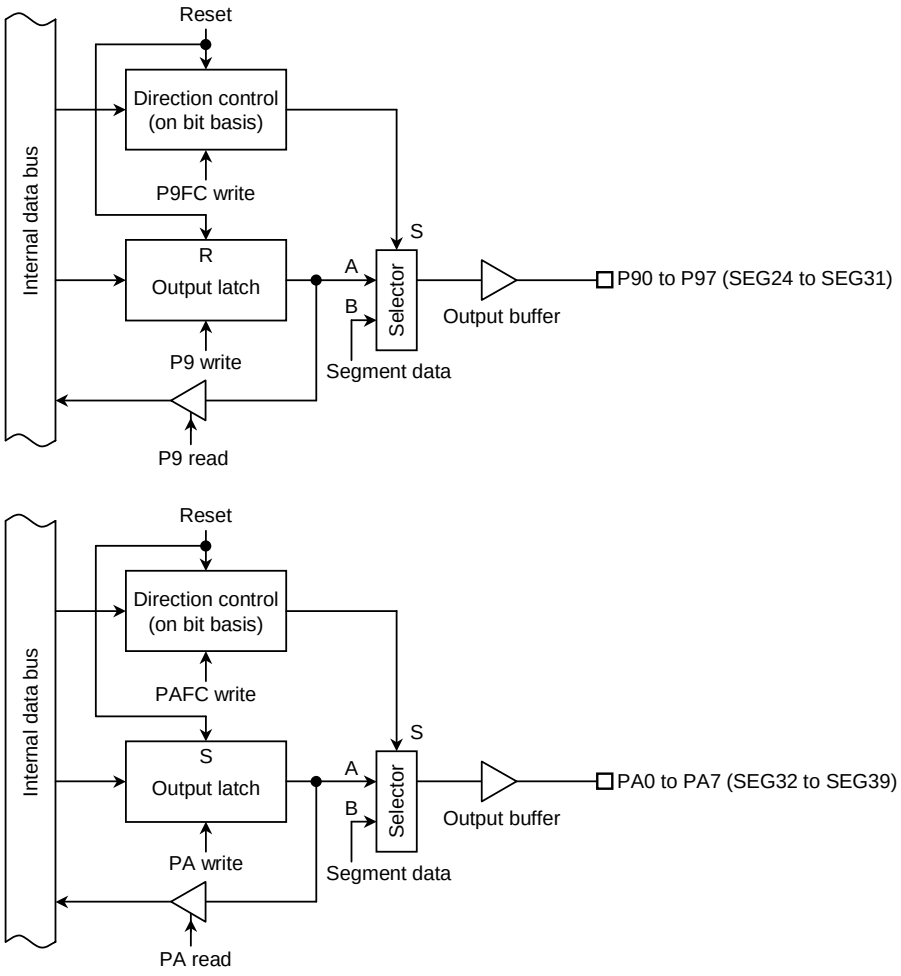


Figure 3.5.30 Port 9 and A

Port 9

P9 (0019H)

	7	6	5	4	3	2	1	0
Bit symbol	P97	P96	P95	P94	P93	P92	P91	P90
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	Output mode							

Port A Register

PA (001EH)

	7	6	5	4	3	2	1	0
Bit symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
	Output mode							

Port 9 Function Register

P9FC (001DH)

	7	6	5	4	3	2	1	0
Bit symbol	P97F	P96F	P95F	P94F	P93F	P92F	P91F	P90F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SEG31	0: Port 1: SEG30	0: Port 1: SEG29	0: Port 1: SEG28	0: Port 1: SEG27	0: Port 1: SEG26	0: Port 1: SEG25	0: Port 1: SEG24

Port A Function Register

PAFC (001FH)

	7	6	5	4	3	2	1	0
Bit symbol	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Port 1: SEG39	0: Port 1: SEG38	0: Port 1: SEG37	0: Port 1: SEG36	0: Port 1: SEG35	0: Port 1: SEG34	0: Port 1: SEG33	0: Port 1: SEG32

Note 1: Read-modify-writes is prohibited for registers P9FC and PAFC.

Note 2: Port 9 and port A are open-drain output pins.

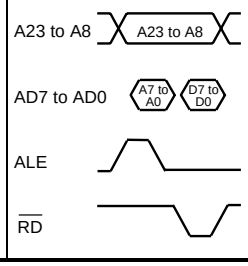
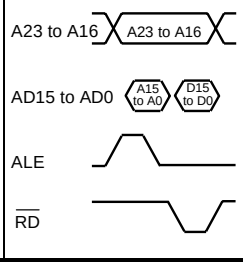
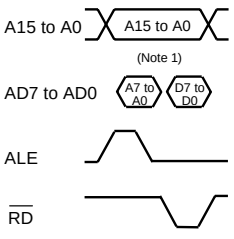
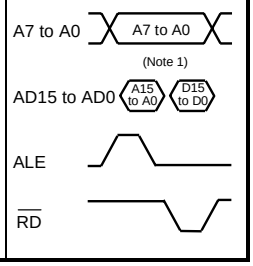
Figure 3.5.31 Registers for Ports 9 and A

3.6 Bus Width/Wait Controller

The TMP93CS20 has a built-in bus width/wait controller used to control chip select, wait ($\overline{WAIT}$ pin), and data bus size (8 or 16 bits) for any of the three block address areas.

3.6.1 Address/Data Bus Pins

Port 0/AD0 to AD7, port 1/AD8 to AD15, and port 2/AD16 to AD23/A0 to A7 function as an address and data bus for connecting the microcontroller to the external memories and I/O peripherals.

		1	2	3	4
Number of address bus pins		Max 24 (to 16 Mbytes)	Max 24 (to 16 Mbytes)	Max 16 (to 64 Kbytes)	Max 8 (to 256 bytes)
Number of data bus pins		8	16	8	16
Number of multiplexed pins		8	16	0	0
Mode pins	$\overline{EA}$	V_{IH}			
Port function	Port 0	AD0 to AD7	AD0 to AD7	AD0 to AD7	AD0 to AD7
	Port 1	A8 to A15	AD8 to AD15	A8 to A15	AD8 to AD15
	Port 2	A16 to A23	A16 to A23	A0 to A7	A0 to A7
Timing chart					

Note 1: In the cases of 3. and 4., the data bus signals output the addresses because the signals are also used as the address bus. By writing 0 to bit CKOCR<ALEEN>, the ALE signal can be prevented from outputting.

Note 2: After a reset operation, port 0, port 1, and port 2 of the TMP93CS20 function as input ports.

3.6.2 Bus Width/Wait Control Registers

Table 3.6.1 shows control registers.

One block of the address areas is controlled by each of the 1-byte bus width/wait control registers WAITC0, WAITC1, and WAITC2.

(1) Data bus size select

Bit4 (B0BUS, B1BUS, and B2BUS) of the control register is used to specify data bus size. Setting this bit to 0 accesses the memory in 16-bit data bus mode; setting it to 1 accesses the memory in 8-bit data bus mode.

Changing data bus size depending on the access address is called dynamic bus sizing. Table 3.6.1 shows the details of the bus operation.

(2) Wait control

Control register bits 3 and 2 <B0W1:0, B1W1:0, and B2W1:0> are used to specify the number of waits. Setting these bits to 00 inserts a 2-state wait regardless of the $\overline{\text{WAIT}}$ pin status. Setting them to 01 inserts a 1-state wait regardless of the $\overline{\text{WAIT}}$ status. Setting them to 10 inserts a 1-state wait and samples the $\overline{\text{WAIT}}$ pin status. If the pin is low, inserting the wait maintains the bus cycle until the pin goes high. Setting them to 11 completes the bus cycle without a wait, regardless of the $\overline{\text{WAIT}}$ pin status.

Resetting sets these bits to 00 (2-state wait mode).

(3) Address area specification

Control register bits 1 and 0 <B0C1:0, B1C1:0, and B2C1:0> are used to specify the target address area. Setting these bits to 00 enables settings as follows:

* The WAITC0 setting is enabled when the address space 7F00H to 7FFFH is accessed.
* The WAITC1 setting is enabled when the address space 8A0H to 7FFFH is accessed.
* The WAITC2 setting is enabled when the address space 8000H to 3FFFFFH is accessed,

Setting these bits to 01 enables settings for all WAITC's blocks when the address space 400000H to 7FFFFFH is accessed. Setting these bits to 10 enables when the address space 800000H to BFFFFFH to be accessed. Setting these bits to 11 enables when the address space C00000H to FFFFFFH to be accessed.

Table 3.6.1 Bus Width Control Register

		7	6	5	4	3	2	1	0
WAITC0 (0068H)	Bit symbol				B0BUS	B0W1	B0W0	B0C1	B0C0
	Read/Write				W				
	After reset				0	0	0	0	0
	Function				0:16-bit bus 1: 8-bit bus	00: 2 waits 01: 1 wait 10: (1 + N) waits 11: 0 waits		00: 7F00H to 7FFFH 01: 400000H to 10: 800000H to 11: C00000H to	
WAITC1 (0069H)	Bit symbol				B1BUS	B1W1	B1W0	B1C1	B1C0
	Read/Write				W				
	After reset				0	0	0	0	0
	Function				0:16-bit bus 1: 8-bit bus	00: 2 waits 01: 1 wait 10: (1 + N) waits 11: 0 waits		00: 8A0H to 7FFFH 01: 400000H to 10: 800000H to 11: C00000H to	
WAITC2 (006AH)	Bit symbol				B2BUS	B2W1	B2W0	B2C1	B2C0
	Read/Write				W				
	After reset				0	0	0	1	1
	Function				0:16-bit bus 1: 8-bit bus	00: 2 waits 01: 1 wait 10: (1 + N) waits 11: 0 waits		00: 8000H to 01: 400000H to 10: 800000H to 11: C00000H to	

Note: Read-modify-write is prohibited for WAITC0, WAITC1, and WAITC2.

Table 3.6.2 Dynamic Bus Sizing

Operand Data Size	Operand Start Address	Memory Data Size	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 2n + 1	xxxxx xxxxx	b7 to b0 b15 to b8
		16 bits	2n + 0	b15 to b8	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1 2n + 2	xxxxx xxxxx	b7 to b0 b15 to b8
		16 bits	2n + 1 2n + 2	b7 to b0 xxxxx	xxxxx b15 to b8
32 bits	2n + 0 (Even number)	8 bits	2n + 0 2n + 1 2n + 2 2n + 3	xxxxx xxxxx xxxxx xxxxx	b7 to b0 b15 to b8 b23 to b16 b31 to b24
		16 bits	2n + 0 2n + 2	b15 to b8 b31 to b24	b7 to b0 b23 to b16
	2n + 1 (Odd number)	8 bits	2n + 1 2n + 2 2n + 3 2n + 4	xxxxx xxxxx xxxxx xxxxx	b7 to b0 b15 to b8 b23 to b16 b31 to b24
		16 bits	2n + 1 2n + 2 2n + 4	b7 to b0 b23 to b16 xxxxx	xxxxx b15 to b8 b31 to b24

×××××: During a read, data input to the bus is ignored. While writing, the bus is at high impedance and the write strobe signal remains in active.

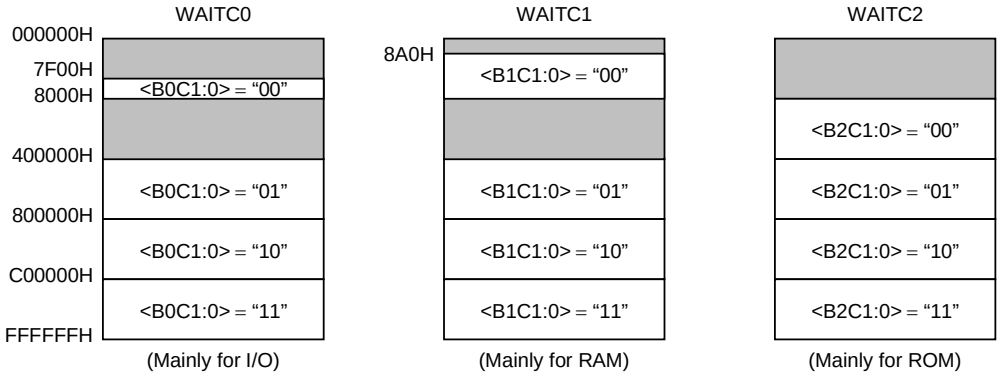
3.6.3 Bus Width/Wait Control

An image of the actual addresses which can be specified by chip select is shown below. Out of the whole memory area, address areas that can be specified are divided into four parts. Addresses from 000000H to 3FFFFFH are further divided as follows: 7F00H to 7FFFH is specified for WAITC0; 8A0H to 7FFFH, for WAITC1; and 8000H to 3FFFFFH, for WAITC2. The reason is that a device other than ROM (e.g., RAM or I/O) might be connected externally.

7F00H to 7FFFH (256 bytes) designated as WAITC0 are mapped mainly for possible expansions to external I/O.

8A0H to 7FFFH (Approx. 30 Kbytes) designated as WAITC1 are mapped mainly for possible extensions to external RAM.

8000H to 3FFFFFH (Approx. 4 Mbytes) designated as WAITC2 are mapped mainly for possible extensions to external ROM. After resetting, WAITC2 is enabled in a 16-bit bus and 2-wait configuration. With the TMP93CS20, which does have a built-in ROM, addresses from FF0000H to FFFFFFH are used as the internal ROM area; WAITC2 is disabled in this area. After resetting, the CPU reads the program from the built-in ROM in 16-bit bus, 0 WAIT mode.



Note 1: Access priority is highest for built-in I/O, then built-in memory, and lowest for the chip select/wait controller.

Note 2: External areas other than WAITC0 to WAITC2 are accessed in 0 WAIT mode.
When using the chip select/wait controller, do not specify the same address area more than once. (However, when specifications overlap, only one of them will be utilized. For example, when addresses 7F00H to 7FFFH for WAITC0 are specified at the same time as 8A0H to 7FFFH for WAITC1, only the WAITC0 setting and pin will be active.)

Example of Usage

Figure 3.6.1 is an example in which an external memory is connected to the TMP93CS20. In this example, a 128-Kbyte ROM is connected using a 16-bit bus, and a 256 Kbyte RAM is connected using a 16-bit bus.

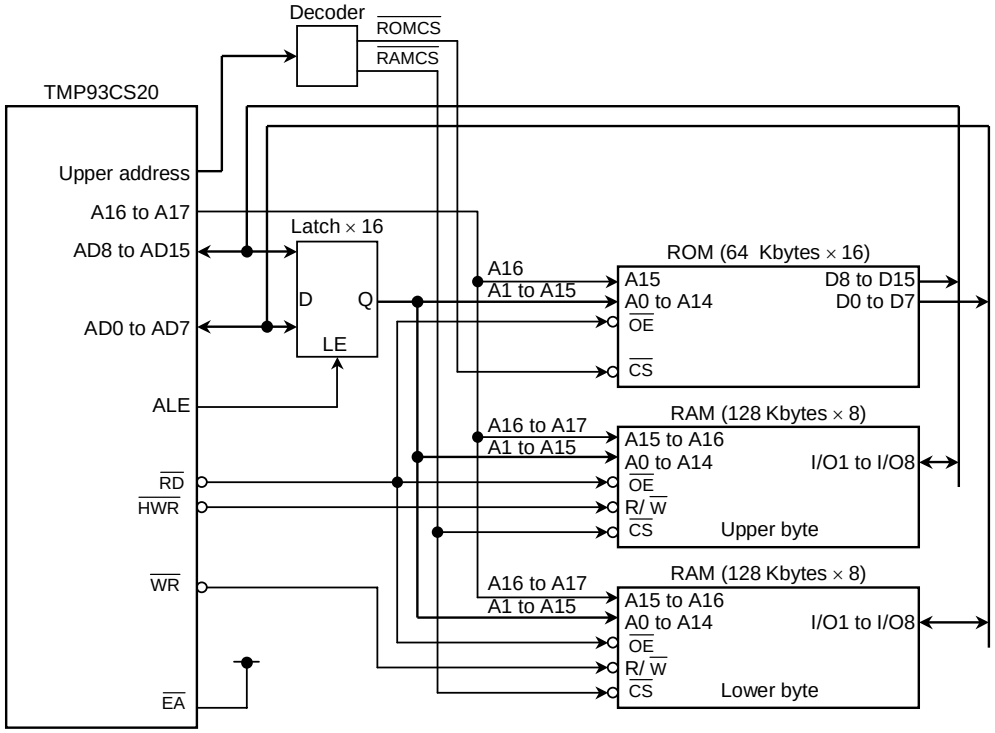


Figure 3.6.1 Example of External Memory Connection (ROM and RAM = 16 bits)

The TMP93CS20 has built-in ROM and RAM. When ROM and RAM have insufficient capacity, it is possible to connect an external memory following the usage examples for this purpose. In this example, the memory configuration is as follows.

Memory		Memory Size	Address	Data Bus
ROM	Internal	64 Kbytes	FF0000H to FFFFFFH	16 bits
	External	128 Kbytes	400000H to 41FFFFH	16 bits
SRAM	Internal	2 Kbytes	0000A0H to 00089FH	16 bits
	External	256 Kbytes	800000H to 83FFFFH	16 bits

3.7 8-Bit Timer

The TMP93CS20 contains four 8-bit timers (timers 0 to 3), each of which can be operated independently. The cascade connection also allows these timers to be used together as two 16-bit timers. The following four operating modes are supported for the 8-bit timers:

- 8-bit interval timer mode (4 timers)
- 16-bit interval timer mode (2 timers)

} Changeable combination (8 bits × 2 timers, 16 bits × 1 timer, etc.)

- 8-bit programmable square wave pulse generation (PPG: Variable duty with variable cycle) output mode (2 timers)
- 8-bit pulse width modulation (PWM: Variable duty with constant cycle) output mode (2 timers)

Figure 3.7.1 shows the block diagram of the 8-bit timers (timer 0 and timer 1).

Figure 3.7.2 shows that of the 8-bit timers (timer 2 and timer 3).

Each timer consists of an 8-bit up counter, 8-bit comparator and 8-bit timer register. Besides, one timer flip-flop (TFF1) is provided for the pair consisting of timer 0 and timer 1, and TFF3 is provided for the pair consisting of timer 2 and timer 3.

Among the input clock sources for the timers, the internal clocks of ϕT1, ϕT4, ϕT16, and ϕT256 are obtained from the 9-bit prescaler shown in Figure 3.7.3.

The operation modes and timer flip-flops of the 8-bit timers are controlled by the five control registers T10MOD, T32MOD, TFFCR, TRUN, and TRDC.

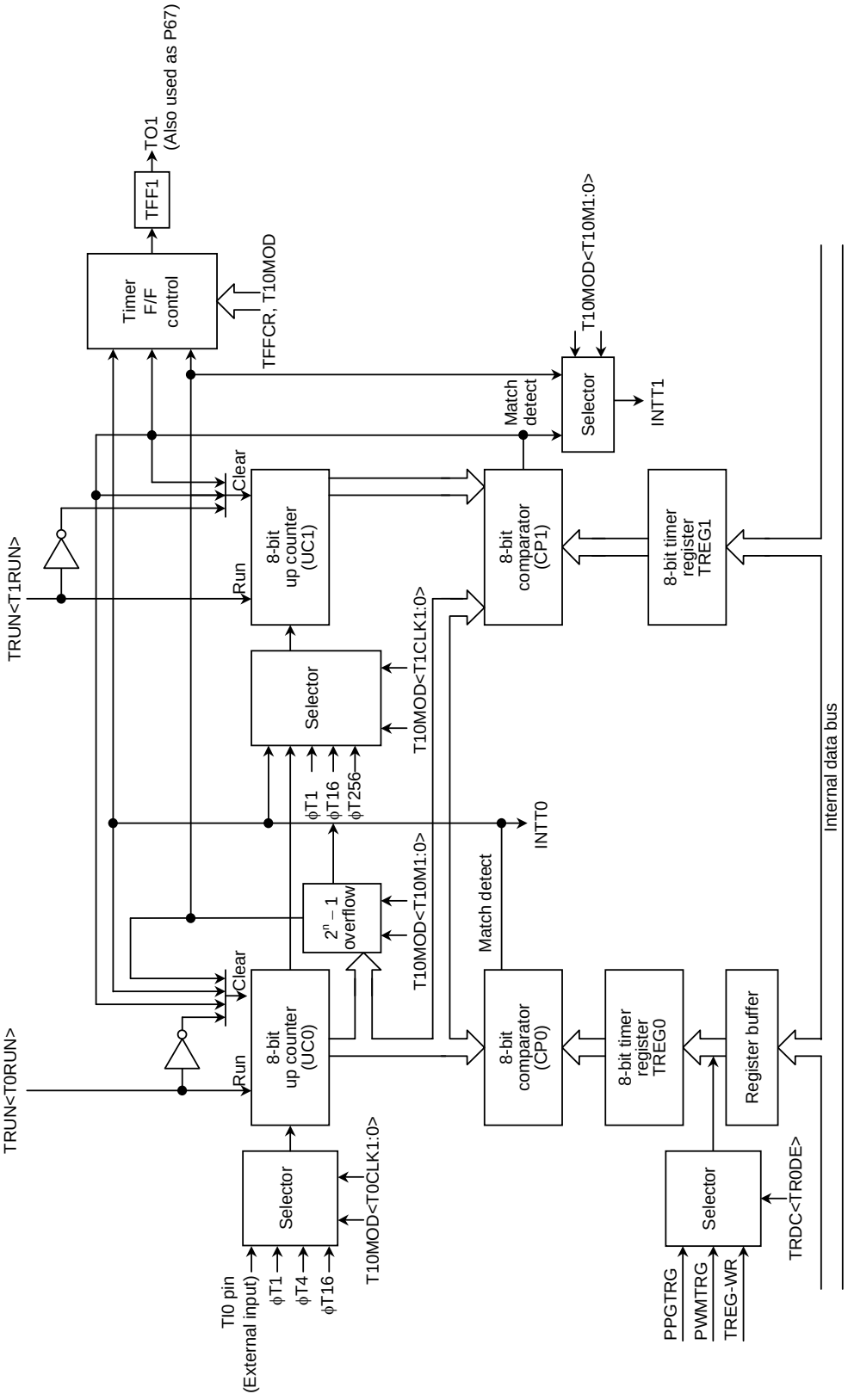


Figure 3.7.1 Block Diagram of 8-Bit Timers (Timers 0 and 1)

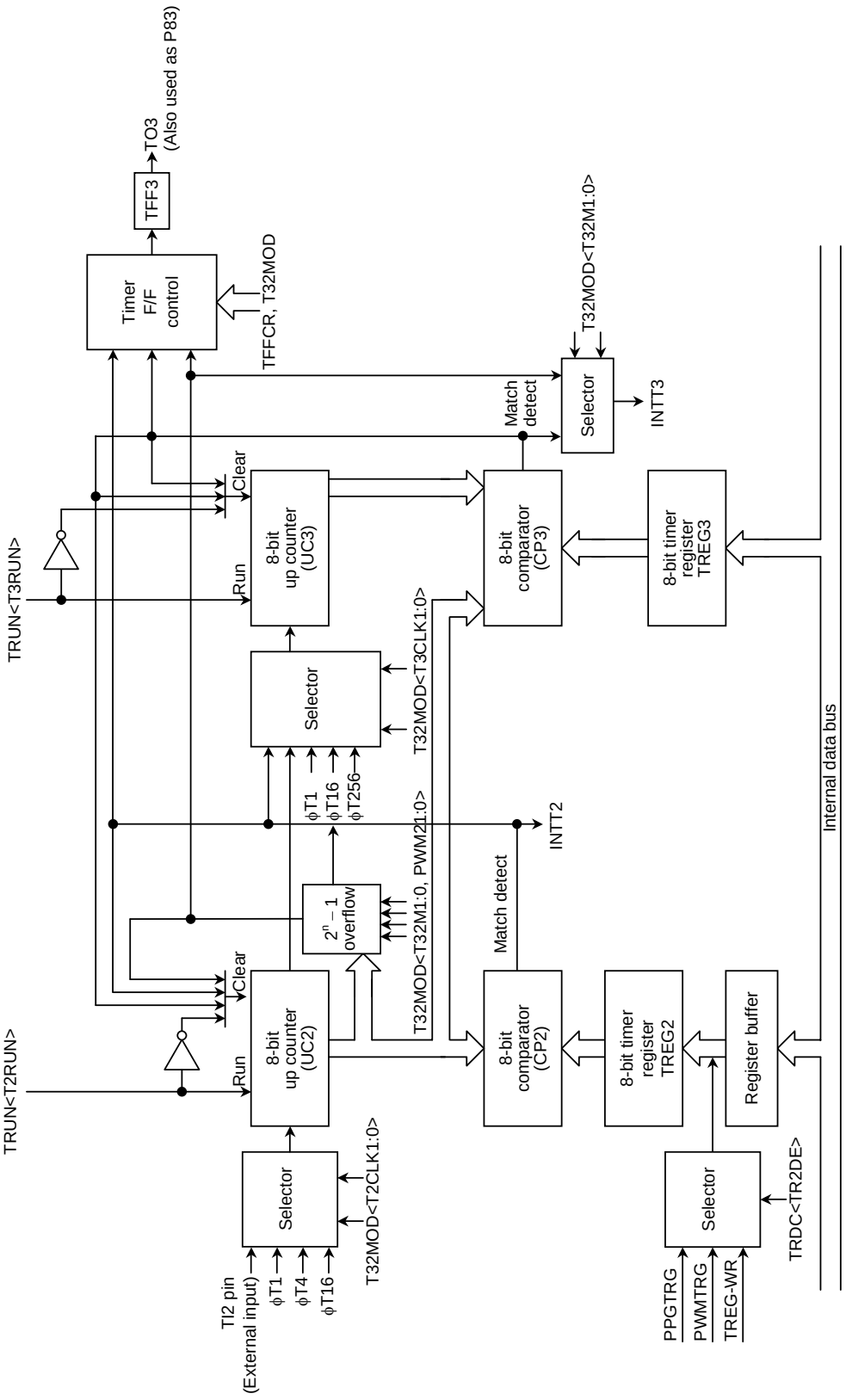


Figure 3.7.2 Block Diagram of 8-Bit Timers (Timers 2 and 3)

1. Prescaler

There are 9-bit prescaler and prescaler clock selection registers to generate input clock signals for the 8-bit timers 0 to 3, the 16-bit timers 4, 6, 8, and A, and the serial interfaces 0 and 1.

Figure 3.7.3 shows the corresponding block diagram, and Table 3.7.1 shows prescaler clock signal resolution into 8- and 16-bit timers.

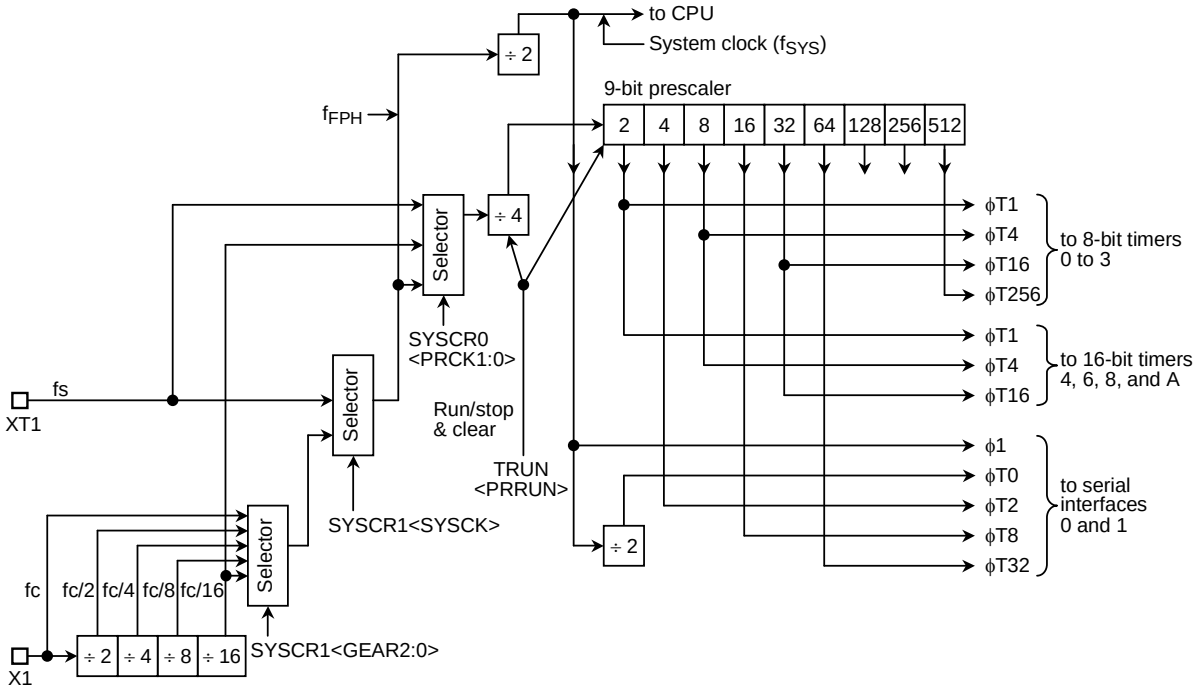


Figure 3.7.3 Block Diagram of the Prescaler

Table 3.7.1 Prescaler Clock Resolution to 8- and 16-Bit Timers

at fc = 20 MHz, fs = 32.768 kHz

Select System Clock <SYSCK>	Select Prescaler Clock <PRCK1:0>	Gear Value <GEAR2:0>	Prescaler 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 (0.4 μs)	fc/2^5 (1.6 μs)	fc/2^7 (6.4 μs)	fc/2^{11} (102.4 μs)
		001 (fc/2)	fc/2^4 (0.8 μs)	fc/2^6 (3.2 μs)	fc/2^8 (12.8 μs)	fc/2^{12} (204.8 μs)
		010 (fc/4)	fc/2^5 (1.6 μs)	fc/2^7 (6.4 μs)	fc/2^9 (25.6 μs)	fc/2^{13} (409.6 μs)
		011 (fc/8)	fc/2^6 (3.2 μs)	fc/2^8 (12.8 μs)	fc/2^{10} (51.2 μs)	fc/2^{14} (819.2 μs)
		100 (fc/16)	fc/2^7 (6.4 μs)	fc/2^9 (25.6 μs)	fc/2^{11} (102.4 μs)	fc/2^{15} (1.6384 ms)
XXX	01 (Low-frequency clock)	XXX	fs/2^3 (244 μs)	fs/2^5 (977 μs)	fs/2^7 (3.9 ms)	fs/2^{11} (62.5 ms)
XXX	10 (Note) (fc/16 clock)	XXX	fc/2^7 (6.4 μs)	fc/2^9 (25.6 μs)	fc/2^{11} (102.4 μs)	fc/2^{15} (1.6384 ms)

xxx: Don't care

16-bit timer

8-bit timer

Note: The fc/16 clock cannot be used as a prescaler clock when the fs is used as a system clock.

The timer clock selected among f_{FPH}, fc/16, and fs is divided by 4 and input to this prescaler. The selection is made by system clock control register SYSCR0<PRCK1:0>.

Resetting sets <PRCK1:0> to 00, which selects the f_{FPH} clock input to be divided by 4.

The 8-bit timers 0 and 1 select among 4 clock inputs: ϕT1, ϕT4, ϕT16, and ϕT256 of the prescaler output.

This prescaler can be run or stopped by the timer control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to 1. The prescaler is cleared to 0 and stops operation when <PRRUN> is set to 0.

Resetting clears <PRRUN> to 0 and stops the prescaler.

When the Idle1 mode (in which only the oscillator operates) is used, set TRUN<PRRUN> to "0" to reduce the power consumption of the prescaler before the "HALT" instruction is executed.

2. Up counter

This is an 8-bit binary counter which counts up by the input clock pulse specified by T10MOD or T32MOD.

The input clocks of timer 0 and timer 2 are selected from among the external clock from the TI0 and TI2 pins, and the three internal clocks ϕT1, ϕT4, and ϕT16, according to the value set in T10MOD<T0CLK1:0> or T32MOD<T2CLK1:0>.

The input clocks used by timer 1 and timer 3 depend on the operation mode. When 16-bit timer mode is set, the overflow outputs of timer 0 and timer 2 are used as the input clock. When any mode other than 16-bit timer mode is set, the input clock is selected from among the internal clocks ϕT1, ϕT16, and ϕT256 as well as the comparator outputs (Match detection signal) of timer 0 and timer 2 according to the set value of the T10MOD and te T32MOD registers.

Example: When T10MOD<T10M1:0> = 01, the overflow output of timer 0 becomes the input clock of timer 1 (16-bit timer mode).
When T10MOD<T10M1:0> = 00 and T10MOD<T1CLK1:0> = 01, ϕT1 becomes the input of timer 1 (8-bit timer mode).

Similarly, operation mode is also set by the T10MOD and the T32MOD registers. When reset, it is initialized to T10MOD<T10M1:0> = 00 and T32MOD<T32M1:0> = 00 whereby the up counter is placed in the 8-bit timer mode.

The counting and stop and clear of the up counter can be controlled for each interval timer by the timer operation control register TRUN. When reset, all up counters will be cleared to stop the timers.

3. Timer registers

These are 8-bit registers for setting a time interval. When the values of the timer registers match the values of the corresponding up counters, the comparator match detect signal becomes active. If the set value is 00H, this signal becomes active when the up counter overflows.

Timer registers TREG0 and TREG2 have a double buffer. The following describes an example of the TREG2.

The 8-bit timer double buffer control register TRDC<TR2DE> bit controls whether the double buffer structure should be enabled or disabled. It is disabled when <TR2DE> = 0 and enabled when it is set to 1.

In the double buffer enable state, the data set in the register buffer are transferred to the timer register when the $2^n - 1$ overflow occurs in PWM mode, or at the PPG cycle in PPG mode. Therefore, during timer mode, the double buffer cannot be used.

Upon resetting, TRDC<TR2DE> will be initialized to 0, disabling the double buffer. To use the double buffer, write data in the timer register, set <TR2DE> to 1, and write the following data in the register buffer.

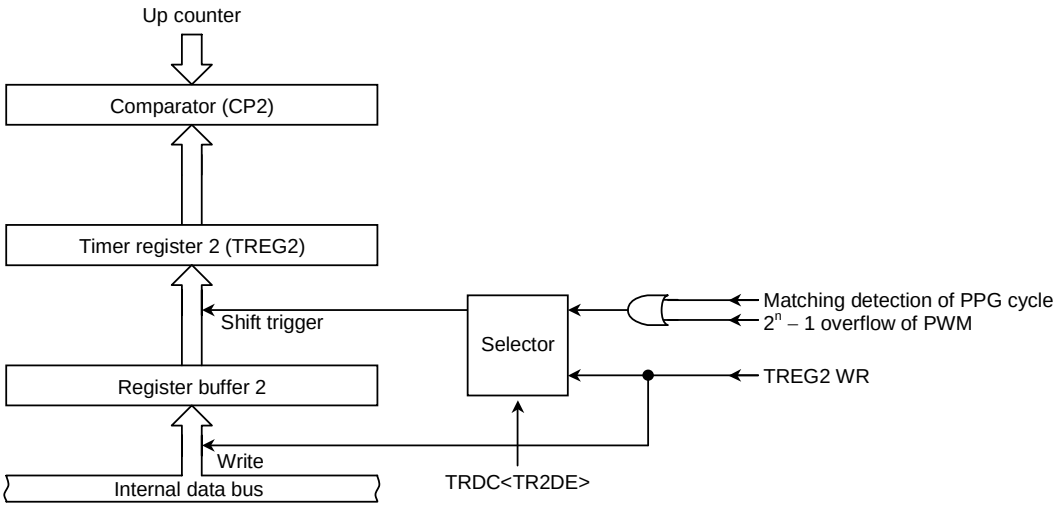


Figure 3.7.4 Configuration of Timer Register 2

Note: The timer register and the register buffer are allocated at the same memory address. When <TR2DE> = 0, the same value is written in the register buffer and in the timer register, while when <TR2DE> = 1 the value is written only into the register buffer.

The memory address of each timer register is as follows.

TREG0: 000022H TREG2: 000026H
TREG1: 000023H TREG3: 000027H

Both of these registers are write only and cannot be read.

4. Comparator

   A comparator compares the value in the up counter with the values to which the timer register is set. When they match, the up counter is cleared to 0 and an interrupt signal (INTT0, INTT1, INTT2, and INTT3) is generated. If the timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

5. Timer flip-flop

   The timer flip-flop (TFF1 and TFF3) is a flip-flop inverted by the match detect signal (8-bit comparator output).

   Inverting is enabled or disabled by the timer flip-flop control register TFFCR<TFF3IE>, <TFF1IE>.

   After a reset operation, the value of TFF1 and TFF3 are undefined. Writing 01 or 10 to TFFCR<TFF3C1:0> <TFF1C1:0> sets 0 or 1 to TFF1 and TFF3. Additionally, writing 00 to this bit inverts the value of TFF1 and TFF3 (Software inversion).

   The values in TFF1 and TFF3 can be output to the TO1 pin (also used as P67) and TO3 pin (also used as P83). When using the TFF1 and TFF3 contents as the timer output, the timer flip-flop should be set by the port 6 and 8 function registers P6FC and P8FC beforehand.

Timer Operation Control Register

TRUN (0020H)

	7	6	5	4	3	2	1	0
Bit symbol	PRRUN		T6RUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
Read/Write	R/W		R/W					
After reset	0		0	0	0	0	0	0
Function	Prescaler and timer run/stop control 0: Stop and clear 1: Run (count up)							

Count operation

0	Stop and clear
1	Count

PRRUN: Operation of prescaler
T6RUN: Operation of 16-bit timer (Timer 6)
T4RUN: Operation of 16-bit timer (Timer 4)
T3RUN: Operation of 8-bit timer (Timer 3)
T2RUN: Operation of 8-bit timer (Timer 2)
T1RUN: Operation of 8-bit timer (Timer 1)
T0RUN: Operation of 8-bit timer (Timer 0)

Note: TRUN<Bit6> is read as 1.

System Clock Control Register

SYSCR0 (006EH)

	7	6	5	4	3	2	1	0
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 the STOP mode 0: Stop 1: Oscillation	Low-frequency oscillator (fs) after release of the STOP mode 0: Stop 1: Oscillation	Clock selection after release of the Stop mode 0: fc 1: fs	Warm-up timer (Write) 0: Don't care 1: Start timer (Read) 0: End warm up 1: Continue warm up	Select prescaler clock 00: f_{FPH} 01: fs 10: fc/16 11: (Reserved)	

Select prescaler clock setting

00	f_{FPH}
01	fs
10	fc/16
11	(Reserved)

Clock divided by 4

Figure 3.7.5 Registers for 8-Bit Timers (1/5)

Timer 1 and Timer 0 Mode Control Register

T10MOD (0024H)

	7	6	5	4	3	2	1	0
Bit symbol	T10M1	T10M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock of timer 1 00: TO0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock of timer 0 00: TI0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	

Input clock signal of timer 0

00	External Input (TI0 pin input)
01	ϕT1 (Prescaler)
10	ϕT4 (Prescaler)
11	ϕT16 (Prescaler)

Input clock of timer 1

	T10MOD<T10M1:0> ≠ 01	T10MOD<T10M1:0> = 01
00	Comparator output of timer "0"	Overflow output of timer "0" (16-bit timer mode)
01	Internal clock ϕT1	
10	Internal clock ϕT16	
11	Internal clock ϕT256	

Select PWM cycle

00	Don't care
01	$(2^6 - 1) \times$ Timer 0 input clock cycle
10	$(2^7 - 1) \times$ Timer 0 input clock cycle
11	$(2^8 - 1) \times$ Timer 0 input clock cycle

Set the operation modes of timers 0 and 1.

00	Two 8-bit timers (Timer 0 and timer 1)
01	16-bit timer
10	8-bit PPG output
11	8-bit PWM output (Timer 0) 8-bit timer (Timer 1)

Figure 3.7.6 Registers for 8-Bit Timers (2/5)

Timer 3 and Timer 2 Mode Control Register

T32MOD (0028H)

	7	6	5	4	3	2	1	0
Bit symbol	T32M1	T32M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Source clock of timer 3 00: TO2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Source clock of timer 2 00: TI2 pin input 01: ϕT1 10: ϕT4 11: ϕT16	

Input clock signal of timer 2

00	External Input (TI2 pin input)
01	ϕT1 (Prescaler)
10	ϕT4 (Prescaler)
11	ϕT16 (Prescaler)

Input clock of timer 3

	T32MOD<T32M1:0> ≠ 01	T32MOD<T32M1:0> = 01
00	Comparator output of timer 2	Overflow output of timer 2 (16-bit timer mode)
01	Internal clock ϕT1	
10	Internal clock ϕT16	
11	Internal clock ϕT256	

Select PWM cycle

00	Don't care
01	$(2^6 - 1) \times$ Timer 2 input clock cycle
10	$(2^7 - 1) \times$ Timer 2 input clock cycle
11	$(2^8 - 1) \times$ Timer 2 input clock cycle

Set the operation modes of timers 2 and 3.

00	Two 8-bit timers
01	16-bit timer
10	8-bit PPG output
11	8-bit PWM output (Timer 2) 8-bit timer (Timer 3)

Figure 3.7.7 Registers for 8-Bit Timers (3/5)

Timer Flip-flop Control Register

TFFCR (0025H)

	7	6	5	4	3	2	1	0
Bit symbol	TFF3C1	TFF3C0	TFF3IE	TFF3IS	TFF1C1	TFF1C0	TFF1IE	TFF1IS
Read/Write	W		R/W		W		R/W	
After reset	1	1	0	0	1	1	0	0
Function	00: Invert TFF3 01: Set TFF3 10: Clear TFF3 11: Don't care		TFF3 Inversion trigger 0: Disable 1: Enable	TFF3 Inversion source 0: Timer 2 1: Timer 3	00: Invert TFF1 01: Set TFF1 10: Clear TFF1 11: Don't care		TFF1 Inversion trigger 0: Disable 1: Enable	TFF1 Inversion source 0: Timer 0 1: Timer 1

Select inverse signal of timer F/F3 (TFF3)
("Don't care" except in 8-bit timer mode)

0	Inversion by timer 2
1	Inversion by timer 3

Inversion of timer flip-flop 3 (TFF3)

0	Disable inversion
1	Enable inversion

Control of timer flip-flop 3 (TFF3)

00	Invert the value of TFF3 (Software inversion)
01	Set TFF3 to 1.
10	Clear TFF3 to 0.
11	Don't care (Always read as 11)

Select inverse signal of timer F/F1 (TFF1)
("Don't care" except in 8-bit timer mode)

0	Inversion by timer 0
1	Inversion by timer 1

Inversion of timer flip-flop 1 (TFF1)

0	Disable inversion
1	Enable inversion

Control of timer flip-flop 1 (TFF1)

00	Invert the value of TFF1 (Software inversion)
01	Set TFF1 to 1.
10	Clear TFF1 to 0.
11	Don't care (Always read as 11)

TFFCR<TFF3C1:0>, <TFF1C1:0> are read as 1.

Figure 3.7.8 Registers for 8-Bit Timers (4/5)

Timer Register Double Buffer Control Register

TRDC (0029H)

	7	6	5	4	3	2	1	0
Bit symbol							TR2DE	TR0DE
Read/Write							R/W	
After reset							0	0
Function							Double buffer of timer register 2	Double buffer of timer register 0

Control of timer register double buffer

0	Disable
1	Enable

Figure 3.7.9 Registers for 8-Bit Timers (5/5)

The operation of 8-bit timers will be described below:

(1) 8-bit timer mode

Four interval timers, designated 0 to 3, can be used independently as 8-bit interval timers. When setting functions and count data, stop operation of timer 0 to 3.

1. Generating interrupts in a fixed cycle

The operation of timer 1 will be explained below.

To generate timer 1 interrupts at constant intervals using timer 1 (INTT1), first stop timer 1. Set the operation mode and input clock speed by setting T10MOD, and the cycle time by setting TREG1. Then, enable interrupt INTT1 and start the counting of timer 1.

Example: To generate timer 1 interrupt every 1 second at fs = 32 kHz, set each register in the following manner.

* Clock condition — System clock: Low frequency (fs); Prescaler clock: Low frequency (fs)

	MSB 7	6	5	4	3	2	1	LSB 0	
TRUN	← –	X	–	–	–	–	0	–	Stop timer 1, and clear it to 0.
T10MOD	← 0	0	X	X	1	0	–	–	Set the 8-bit timer mode, and select ϕT16 (4 ms at fs = 32 kHz) as the input clock.
TREG1	← 1	1	1	1	1	0	1	0	Set the timer register 1, 1 s ÷ ϕT16 = 250 = FAH (H signifies hexadecimal)
INTET10	← 1	1	0	1	–	–	–	–	Enable INTT1, and set it to “level 5”.
TRUN	← 1	X	–	–	–	–	1	–	Start timer 1 counting.

X: Don’t care, –: No change

Use Table 3.7.1 for selecting the input clock.

Note: The input clock choices available for timer 0 and timer 1 differ from each other as follows.
Timer 0: TI0 input, ϕT1, ϕT4, ϕT16
Timer 1: Match detect signal of timer 0, ϕT1, ϕT16, ϕT256

2. Generating a 50% duty, square wave pulse

Timer 1 and timer 3 have a timer flip-flop respectively.

The timer flip-flop (TFF1 and TFF3) is inverted at constant intervals, and its status is output to timer output pin (TO1 and TO3).

Example: To output a 2.4 μs square wave pulse from the TO3 pin at fc = 20 MHz, set each register by the following procedures. Either timer 2 or timer 3 may be used, but this example uses timer 3.

* Clock condition

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

		7	6	5	4	3	2	1	0	
TRUN	←	–	X	–	–	0	–	–	–	Stop timer 3, and clear it to 0.
T32MOD	←	0	0	X	X	0	1	–	–	Set the 8-bit timer mode, and select φT1 (0.4 μs at fc = 20 MHz) as the input clock cycle time.
TREG3	←	0	0	0	0	0	0	1	1	Set the timer register at 2.4 μs ÷ φT1 ÷ 2 = 3.
TFFCR	←	1	0	1	1	–	–	–	–	Clear TFF3 to 0, and set to invert by the match detect signal from timer 3.
P8CR	←	–	–	–	–	1	–	–	–	Select P83 as TO3 pin.
P8FC	←	X	X	X	X	1	X	X	–	
TRUN	←	1	X	–	–	1	–	–	–	Start timer 3 counting.

X: Don't care, –: No change

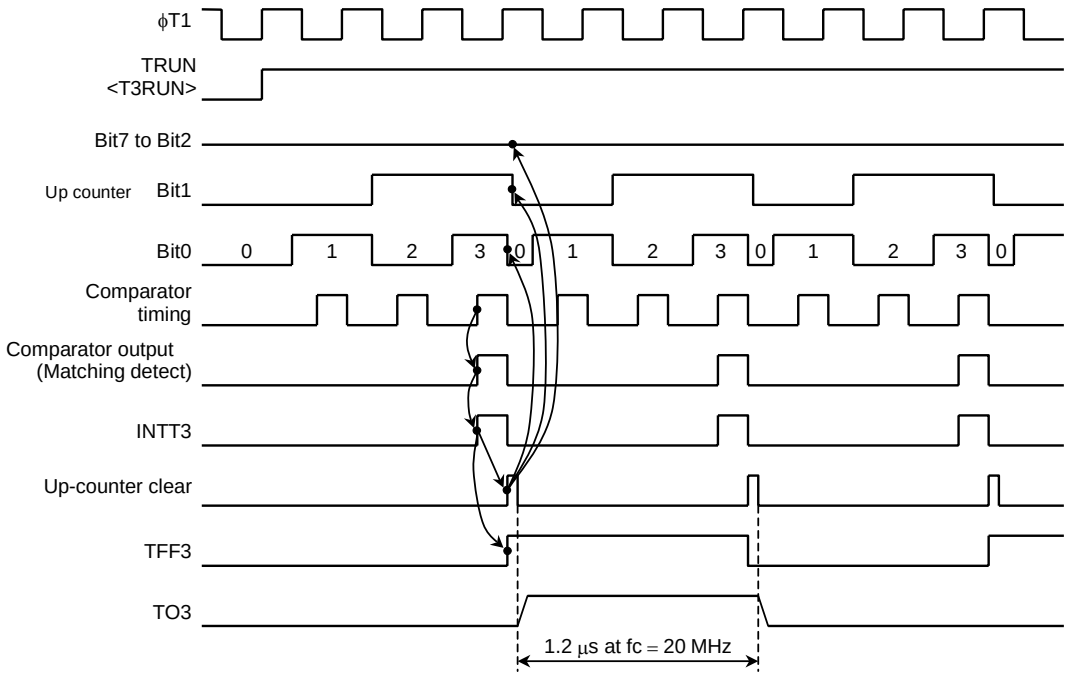


Figure 3.7.10 Square Wave (50% duty) Output Timing Chart

3. Making timer 1 count up by matching the signal from the timer 0 comparator. (Same function is achieved by using timer 3 and timer 2.)

   Set the 8-bit timer mode, and set the comparator output of timer 0 as the input clock to timer 1.

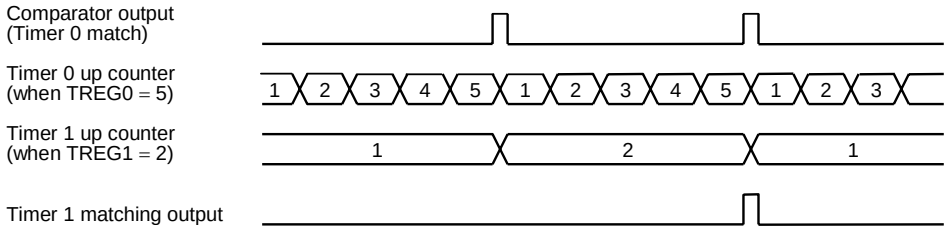


Figure 3.7.11 Timer 1 Count-up Regulated by Timer 0

(2) 16-bit timer mode

A 16-bit interval timer is configured by using timer 0 and timer 1 or timer 2 and timer 3 as a pair.

Setting timer mode register T10MOD<T10M1:0> or T32MOD<T32M1:0> of timer 0 and timer 1 to 01 establishes a 16-bit timer mode.

When set in 16-bit timer mode, the overflow outputs of timer 0 and timer 2 will become the input clocks of timer 1 and timer 3, regardless of the set value of T10MOD<T1CLK1:0> or T32MOD<T3CLK1:0>. Table 3.7 (1) shows the selection of timer 0 and timer 2 input clocks.

The lower 8 bits of the timer (Interrupt) cycle are set by the timer registers TREG0 and TREG2, and the upper 8 bits are set by TREG1 and TREG3. Note that TREG0 or TREG2 always must be set first. (Writing data into TREG0 or TREG2 disables the comparator temporarily, and the comparator is restarted by writing data into TREG1 or TREG3.)

Setting example: To generate an interrupt INTT3 every 0.4 seconds at fc = 20 MHz, set the following values for timer registers TREG2 and TREG3.

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

When counting with the ϕT16 input clock (6.4 μs at 20 MHz)

0.4 s ÷ 6.4 μs = 62500 = F424H

Therefore, set TREG3 = F4H and TREG2 = 24H, respectively.

The comparator signal is output from timer 2 each time the up-counter UC2 matches TREG2, when the up counter UC2 is not to be cleared.

With the timer 3 comparator, the match detect signal is output at each comparator check when the up counter UC3 and TREG3 values are found to match. When the match detect signal is output simultaneously from the comparators of both timer 2 and timer 3, the up counters UC2 and UC3 are cleared to 0, and the interrupt INTT3 is generated. If inversion is enabled, the value of the timer flip-flop TFF3 is inverted.

Example: When TREG3 = 04H and TREG2 = 80H

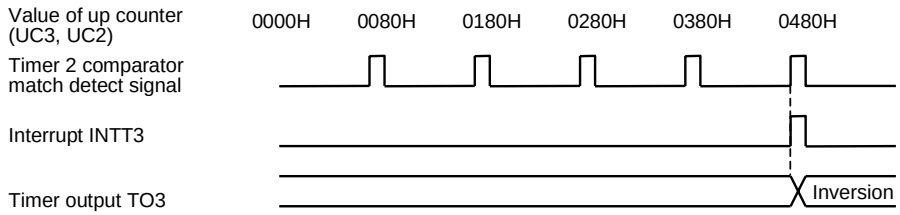


Figure 3.7.12 Timer Output by 16-Bit Timer Mode

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

Square wave pulse can be generated at any frequency and duty by timer 0 or timer 2. The output pulse may be either low-active or high-active. In this mode, timer 1 or timer 3 cannot be used.

Timer 0 outputs a pulse to the TO1 pin (also used as P67), or the TO3 pin (also used as P83).

The operation of timer 2 will be explained below.

In this mode, a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC2) matches the timer registers TREG2 and TREG3.

However, it is necessary for the set value of TREG2 to be smaller than that of TREG3.

Though the up counter (UC3) of timer 3 is not used in this mode, UC3 should be set for counting by setting TRUN<T3RUN>to 1.

Figure 3.7.14 shows the block diagram for this mode.

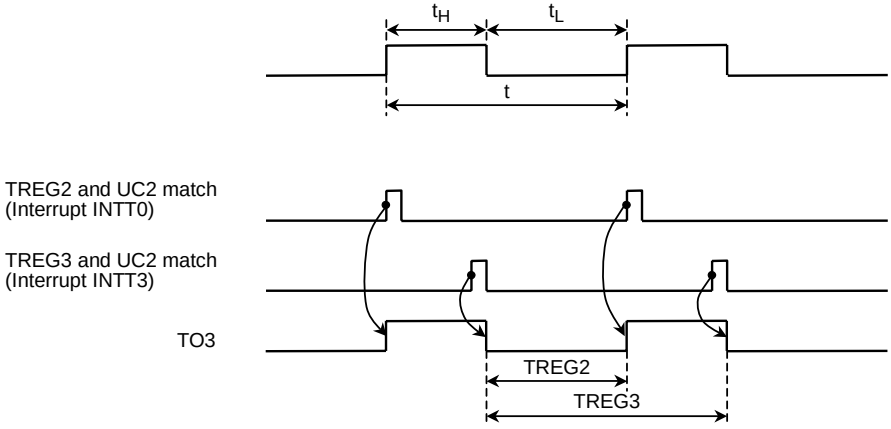


Figure 3.7.13 8-Bit PPG Output Waveforms

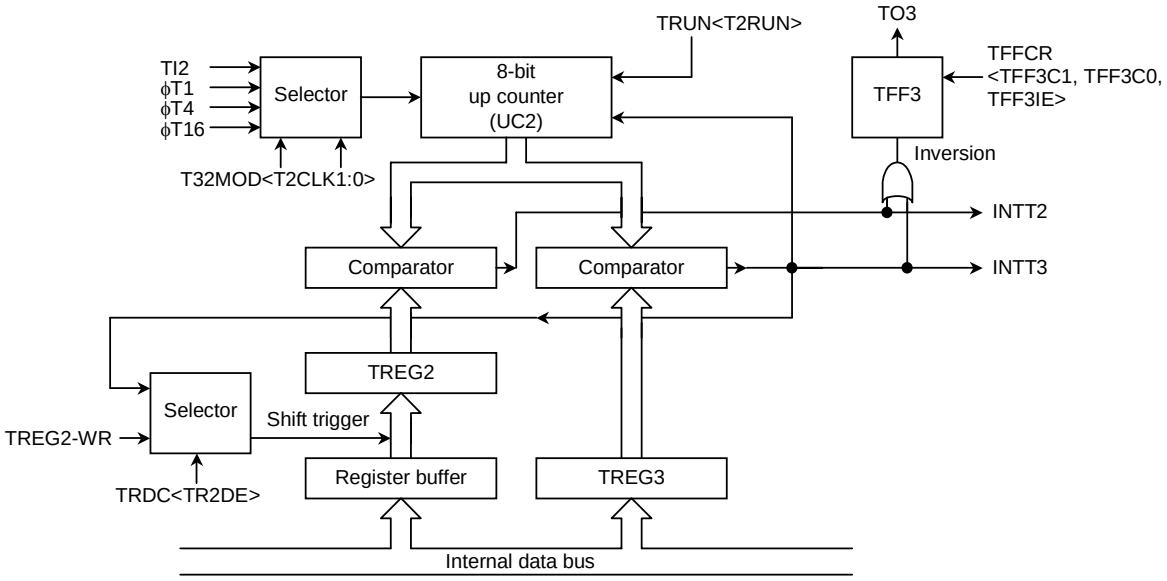


Figure 3.7.14 Block Diagram of 8-Bit PPG Output Mode

When the double buffer of TREG2 is enabled in this mode, the value of the register buffer will be shifted in TREG2 each time TREG3 matches UC2.

Use of the double buffer makes the handling of low duty waves easy (when duty is varied).

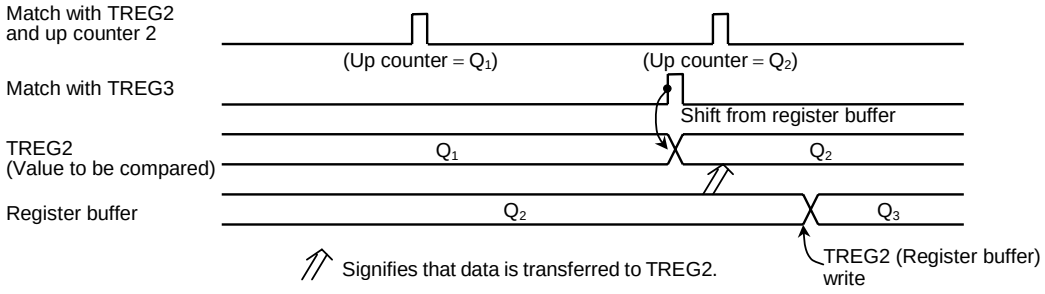
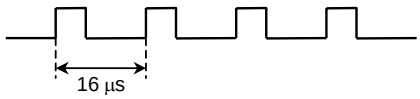


Figure 3.7.15 Operation of Register Buffer

Example: Generating 1/4 duty 62.5 kHz pulse (at fc = 20 MHz)



* Clock condition

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

- Calculate the value to be set in the timer register.

  To obtain a frequency of 62.5 kHz, the pulse cycle time t should be:

  t = 1/62.5 kHz = 16 μs.

  Given ϕT1 = 0.4 μs (at 20 MHz),

  16 μs ÷ 0.4 μs = 40

  Consequently, set the timer register 3 (TREG3) to TREG3 = 40 = 28H

  and then to obtain a duty of 1/4, t × 1/4 = 16 μs × 1/4 = 4 μs

  4 μs ÷ 0.4 μs = 10

  Therefore, set timer register 2 (TREG2) to TREG2 = 10 = 0AH.

		7	6	5	4	3	2	1	0	
TRUN	←	–	X	–	–	0	0	–	–	Stop timers 2 and 3, and clear then.
T32MOD	←	1	0	X	X	X	X	0	1	Set the 8-bit PPG mode, and select ϕT1 as input clock.
TREG2	←	0	0	0	0	1	0	1	0	Write 0AH.
TREG3	←	0	0	1	0	1	0	0	0	Write 28H.
TFFCR	←	0	1	1	X	–	–	–	–	Set TFF3 and enable the inversion. Writing 10 provides negative logic pulse.
P8CR	←	–	–	–	–	1	–	–	–	Set P83 as the TO3 pin.
P8FC	←	X	X	X	X	1	X	X	X	
TRUN	←	1	X	–	–	1	1	–	–	Start timer 2 and timer 3 counting.

X: Don't care, –: No change

(4) 8-bit PWM output mode

This mode is valid only for timer 0 and timer 2. In this mode, the maximum 8-bit resolution of the PWM pulse can be output.

When using timer 0, the PWM pulse is output to the TO1 pin (also used as P67). Timer 1 can also be used as an 8-bit timer. When using timer 2, the PWM pulse is output to the TO3 pin (also used as P83). The operation of timer 2 will be explained below. Timer 3 can also be used as an 8-bit timer.

Timer output is inverted when the up counter (UC2) matches the set value of timer register TREG2, or when $2^n - 1$ (n = 6, 7, or 8; specified by T32MOD<PWM21:0>) counter overflow occurs. Up counter UC2 is cleared when $2^n - 1$ counter overflow occurs.

To use this PWM mode, the following conditions must be satisfied.

(Set value of timer register) < (Set value of $2^n - 1$ counter overflow)

(Set value of timer register) ≠ 0

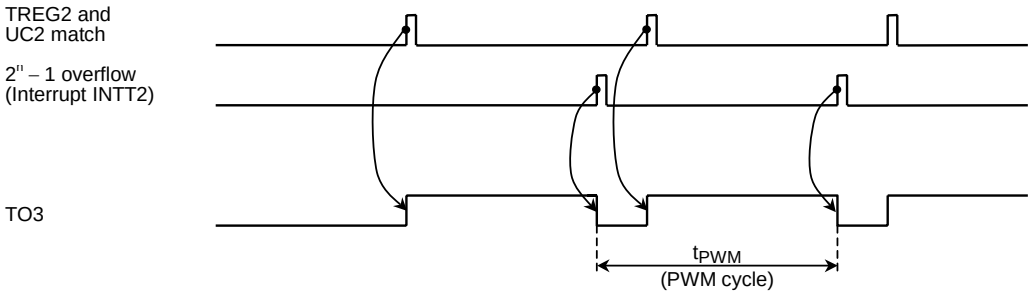


Figure 3.7.16 8-Bit PWM Waveforms

Figure 3.7.17 shows the block diagram of operations in this mode.

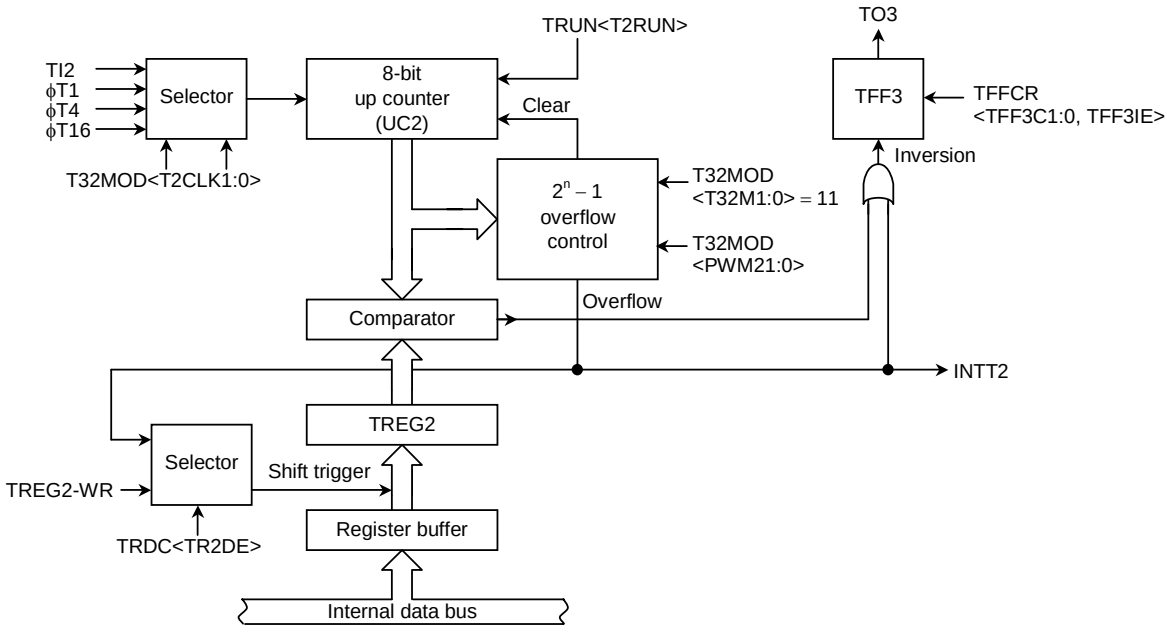


Figure 3.7.17 Block Diagram of Operations in 8-Bit PWM Mode

In this mode, the value of the register buffer will be shifted into TREG2 if a $2^n - 1$ overflow is detected while the double buffer of TREG2 is enabled.

Use of the double buffer makes easy the handling of small-duty waves.

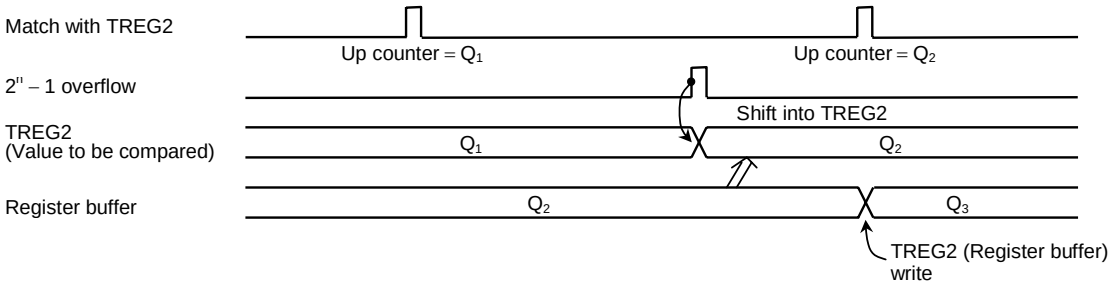
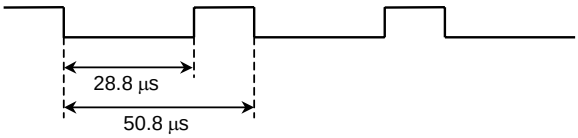


Figure 3.7.18 Operation of Register Buffer

Example: To output the following PWM waves to the TO3 pin at fc = 20 MHz.



* Clock condition

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

To realize a PWM cycle of 50.8 μs by utilizing ϕT1 = 0.4 μs (at fc = 20 MHz),

50.8 μs ÷ 0.4 μs = 127 = $2^n - 1$

Consequently, n should be set to 7.

As the period of low level is 28.8 μs, for ϕT1 = 0.4 μs,

set the following value for TREG2.

28.8 μs ÷ 0.4 μs = 72 = 48H

	MSB 7	6	5	4	3	2	1	LSB 0	
TRUN	← –	X	–	–	–	0	–	–	Stop timer 2 and clear it.
T32MOD	← 1	1	1	0	–	–	0	1	Set 8-bit PWM mode (cycle: $2^7 - 1$) and select ϕT1 as the input clock.
TREG2	← 0	1	0	0	1	0	0	0	Writes 48H.
TFFCR	← 1	0	1	X	–	–	–	–	Clear TFF3, enable the inversion and double buffer.
P8CR	← –	–	–	–	1	–	–	–	Set P83 as the TO3 pin.
P8FC	← X	X	X	X	1	X	X	–	
TRUN	← 1	X	–	–	–	1	–	–	Start timer 2 counting.

X: Don't care, –: No change

Table 3.7.2 PWM Cycle

at fc = 20 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)	25.2 μs	100.8 μs	403.2 μs	50.8 μs	203.2 μs	812.8 μs	102.0 μs	408.0 μs	1.63 ms
		001 (fc/2)	50.4 μs	201.6 μs	806.4 μs	101.6 μs	406.4 μs	1.63 ms	204.0 μs	816.0 μs	3.26 ms
		010 (fc/4)	100.8 μs	403.2 μs	1.61 ms	203.2 μs	812.8 μs	3.26 ms	408.0 μs	1.63 ms	6.53 ms
		011 (fc/8)	201.6 μs	806.4 μs	3.23 ms	406.4 μs	1.63 ms	6.52 ms	816.0 μs	3.26 ms	13.06 ms
		100 (fc/16)	403.2 μs	1.61 ms	6.45 ms	812.8 μs	3.25 ms	13.04 ms	1.63 ms	6.53 ms	26.11 ms
XXX	01 (Low-frequency clock)	XXX	15.4 ms	61.5 ms	246 ms	31.0 ms	124 ms	496 ms	62.3 ms	249 ms	996 ms
XXX	10 (fc/16 clock)	XXX	403.2 μs	1.61 ms	6.45 ms	812.8 μs	3.25 ms	13.04 ms	1.63 ms	6.53 ms	26.11 ms

XXX: Don't care

(5) Timer mode setting registers

Table 3.7.3 shows the list of 8-bit timer modes.

Table 3.7.3 Timer Mode Setting Registers

Register Name	T10MOD/T32MOD				TFFCR
Bit Name	T10M/T32M	PWM2	T1CLK/T3CLK	T0CLK/T2CLK	TFF1IS/TFF3IS
Function	Timer Mode	PWM Cycle	Upper Timer Input Clock	Lower Timer Input Clock	Timer F/F Invert Signal Select
16-bit timer mode	01	–	–	External clock, φT1, φT4, φT16 (00, 01, 10, 11)	–
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
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

3.8 16-Bit Timer/Event Counter

The TMP93CS20 contain four multifunctional 16-bit timer/event counters (Timers 4, 6, 8, and A) which support the following operation modes.

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable pulse generation (PPG) mode

Timers 8 and A can be used as the following operation modes using the capture function.

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

3.8.1 Timer/Event Counter 4 and 6

Each timer/event counter consists of a 16-bit up counter, two 16-bit timer registers (One with a double buffer), a 16-bit capture register, two comparators, a timer flip-flop and the control circuit.

Timer/event counters are controlled by 4 control registers: T4MOD/T6MOD, T4FFCR/T6FFCR, TRUN, and T16CR.

Figure 3.8.1, 3.8.2 shows the block diagram of the 16-bit timer/event counters (Timer 4 and timer 6).

Timers 4 and 6 can be used independently.

All timers operate in the same manner except for the following points, and thus only the operation of timer 4 will be explained below.

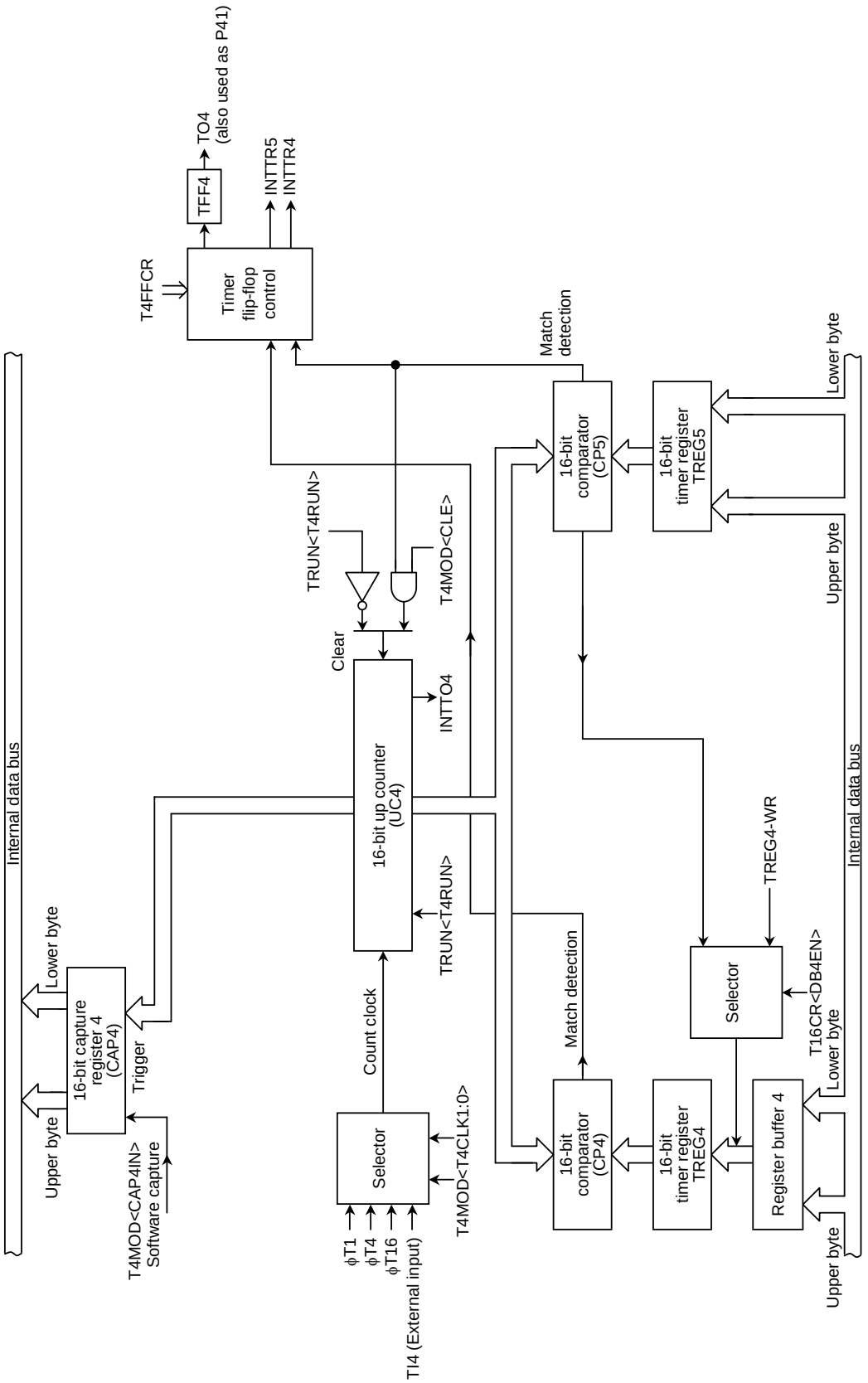


Figure 3.8.1 Block Diagram of 16-Bit Timer (Timer 4)

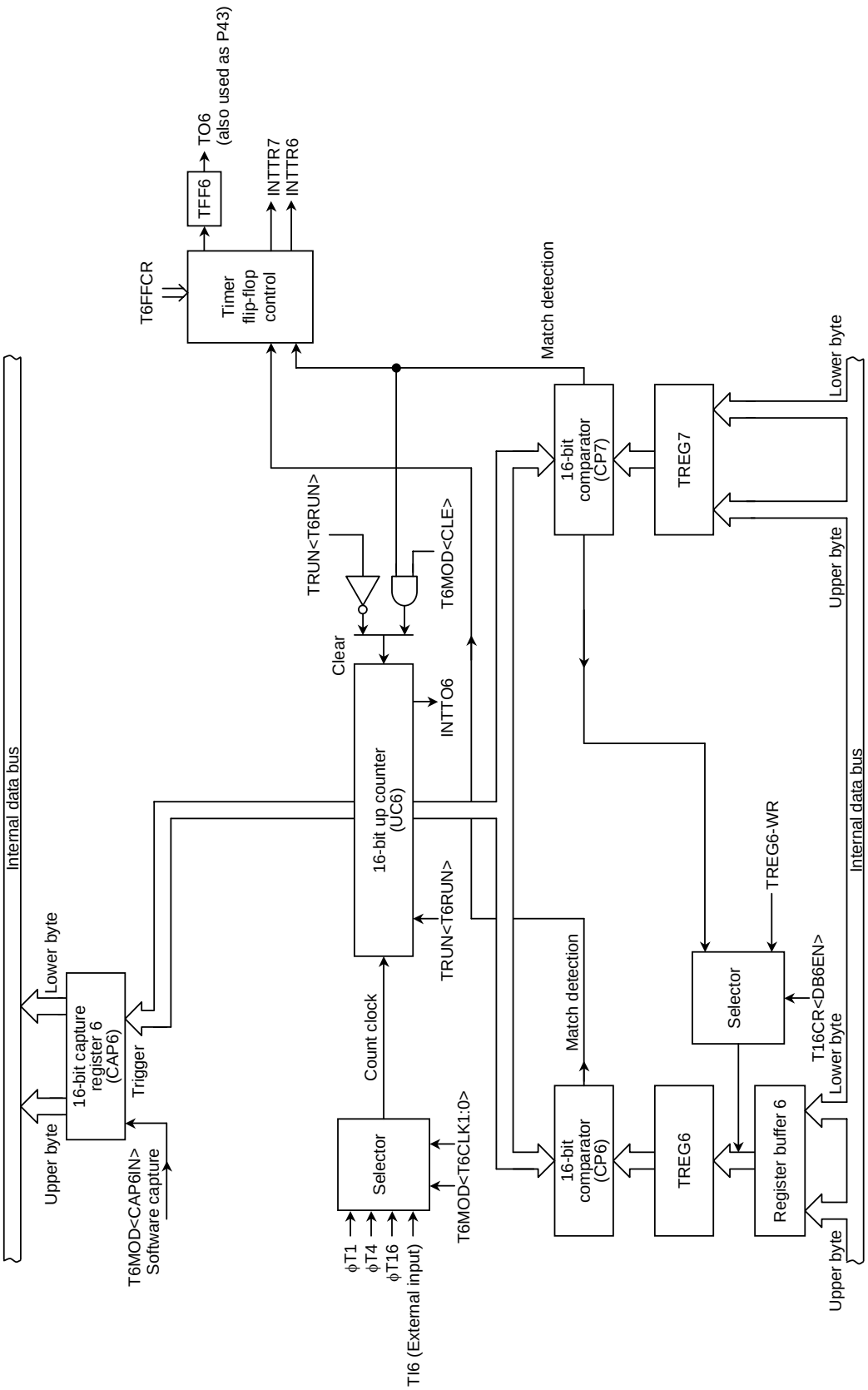


Figure 3.8.2 Block Diagram of 16-Bit Timer (Timer 6)

Timer 4 Mode Control Register

T4MOD (002EH)

	7	6	5	4	3	2	1	0
Bit symbol			CAP4IN	–	–	CLE	T4CLK1	T4CLK0
Read/Write			W	R/W				
After reset			1	0	0	0	0	0
Function			Control of software capture 0: Software capture 1: Don't care	Write "0".		Control of timer 4 0: Clear disable 1: Clear enable	Timer 4 source clock 00: TI4 01: φT1 10: φT4 11: φT16	

Timer 4 input clock

00	External clock (TI4)
01	φT1
10	φT4
11	φT16

Clearing the up counter UC4

0	Clear disable
1	Clear by match with TREG5.

Software capture

0	The up counter 4 value is loaded to CAP4
1	Don't care

Figure 3.8.3 Registers for 16-Bit Timer/Event Counter (1/6)

Timer 4 Flip-flop Control Register

T4FFCR (002FH)

	7	6	5	4	3	2	1	0
Bit symbol			–	–	EQ5T4	EQ4T4	TFF4C1	TFF4C0
Read/Write			R/W	R/W	R/W		W	
After reset			0	0	0	0	1	1
Function			Write “0”.	Write “0”.	TFF4 invert trigger 0: Disable trigger 1: Enable trigger		TFF4 control 00: Invert TFF4 01: Set TFF4 10: Clear TFF4 11: Don’t care * Always read as 11	
					Inverted when the UC value matches TREG5	Inverted when the UC value matches TREG4		

Timer flip-flop 4 (TFF4) control

00	Inverts the TFF4 value (Software inversion).
01	Sets TFF4 to 1.
10	Clear TFF4 to 0.
11	Don’t care (Always read as 11)

Timer flip-flop 4 (TFF4) invert trigger when up counter matches TREG4

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

Timer flip-flop 4 (TFF4) invert trigger when up counter matches TREG5

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

Figure 3.8.4 Registers for 16-Bit Timer/Event Counter (2/6)

Timer 6 Flip-flop Control Register

T6MOD (0036H)

	7	6	5	4	3	2	1	0
Bit symbol			CAP6IN	–	–	CLE	T6CLK1	T6CLK0
Read/Write			W	R/W				
After reset			1	0	0	0	0	0
Function			Control of software capture 0: Software capture 1: Don't care	Write "0".		Control of timer 6 up counter 0: Clear disable 1: Clear enable	Timer 6 source clock 00: TI6 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 6 input clock

00	External clock (TI6)
01	ϕT1
10	ϕT4
11	ϕT16

Clearing the up counter UC6

0	Clear disable
1	Clear by match with TREG7

Software capture

0	The up counter 6 value is loaded to CAP6
1	Don't care

Figure 3.8.5 Registers for 16-Bit Timer/Event Counter (3/6)

Timer 6 Flip-flop Control Register

T6FFCR (0037H)

	7	6	5	4	3	2	1	0
Bit symbol			–	–	EQ7T6	EQ6T6	TFF6C1	TFF6C0
Read/Write			R/W	R/W	R/W		W	
After reset			0	0	0	0	1	1
Function			Write “0”.	Write “0”.	TFF6 invert trigger 0: Disable trigger 1: Enable trigger		00: Invert TFF6 01: Set TFF6 10: Clear TFF6 11: Don’t care	
					Inverted when the UC value matches TREG7	Inverted when the UC value matches TREG6		

Timer flip-flop 6 (TFF6) control

00	Inverts the TFF6 value (Software inversion)
01	Sets TFF6 to 1
10	Clear TFF6 to 0
11	Don’t care (Always read as 11)

Timer flip-flop 6 (TFF6) invert trigger when up counter matches TREG6

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

Timer flip-flop 6 (TFF6) invert trigger when up counter matches TREG7

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

Figure 3.8.6 Reigsters for 16-Bit Timer/Event Counter (4/6)

16-Bit Timer Countrol Register

T16CR (003AH)

	7	6	5	4	3	2	1	0
Bit symbol	QCU		TARUN	T8RUN	DBAEN	DB8EN	DB6EN	DB4EN
Read/Write	R/W		R/W					
After reset	0		0	0	0	0	0	0
Function	Warm-up timer control		Prescaler and timer run/stop control 0: Stop and clear 1: Run (Count up)		Double buffer 0: Disable 1: Enable			
					Double buffer of TREGA	Double buffer of TREG8	Double buffer of TREG6	Double buffer of TREG4

Double buffer control

0	Disable
1	Enable

DBAEN: Double buffer of TREGA
DB8EN: Double buffer of TREG8
DB6EN: Double buffer of TREG6
DB4EN: Double buffer of TREG4

Count operation

0	Stop and clear
1	Count

TARUN: 16-bit timer (Timer A)
T8RUN: 16-bit timer (Timer 8)

Warm-up timer input control

0	Uses 7 stage binary counter
1	Does not use 7 stage binary counter (Note 1)

Note 1: In case of not using the 7 stage binary counter as a warm-up timer, a stable clock signal must be input from an external circuit.

Note 2: T16CR<Bit6> is always read as 1.

Figure 3.8.7 Registers for 16-Bit Timer/Event Counter (5/6)

Timer Operation Control Register

TRUN (0020H)

	7	6	5	4	3	2	1	0
Bit symbol	PRRUN		T6RUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
Read/Write	R/W		R/W					
After reset	0		0	0	0	0	0	0
Function	Prescaler and timer run/stop control 0: Stop and clear 1: Run (Count up)							

Count operation

0	Stop and clear
1	Count

PRRUN: Operation of prescaler.
T6RUN: Operation of 16-bit timer (Timer 6).
T4RUN: Operation of 16-bit timer (Timer 4).
T3RUN: Operation of 8-bit timer (Timer 3).
T2RUN: Operation of 8-bit timer (Timer 2).
T1RUN: Operation of 8-bit timer (Timer 1).
T0RUN: Operation of 8-bit timer (Timer 0).

Note: TRUN<Bit6> is always read as 1.

System Clock Control Register

SYSCR0 (006EH)

	7	6	5	4	3	2	1	0
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 STOP mode 0: Stop 1: Oscillation	Low-frequency oscillator (fs) after release STOP mode 0: Stop 1: Oscillation	Select clock after release STOP mode 0: fc 1: fs	Warm-up timer (Write) 0: Don't care 1: Start timer (Read) 0: End warm up 1: Continue warm up	Select prescaler clock input 00: f_{FPH} 01: fs 10: fc/16 11: (Reserved)	

Select prescaler clock input

00	f_{FPH}
01	fs
10	fc/16
11	(Reserved)

Clock divided by 4.

Figure 3.8.8 Registers for 16-Bit Timer/Event Counters (6/6)

(1) Prescaler

There are 9-bit prescaler and prescaler clock-selection registers to generate input clock signals for 8-bit timers 0 to 3, 16-bit timers 4, 6, 8, and A, and serial interfaces 0 and 1.

Figure 3.8.9 shows the block diagram. Table 3.8.1 shows prescaler clock resolution into 8 and 16-bit timers.

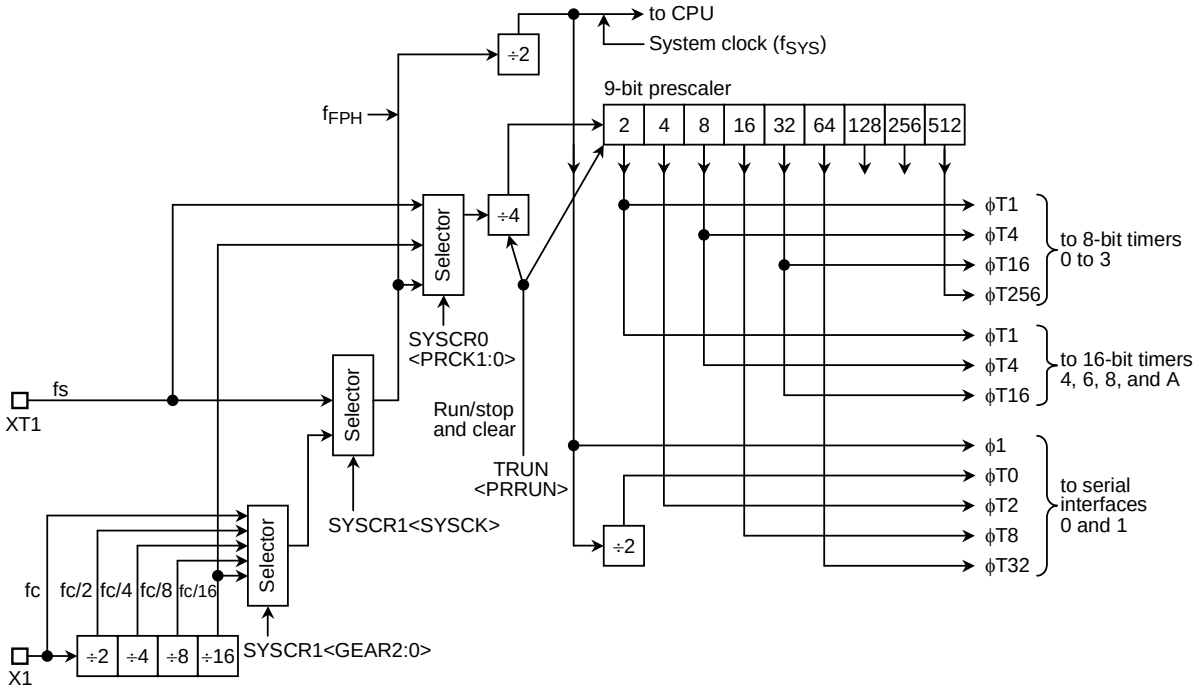


Figure 3.8.9 Block Diagram of Prescaler

Table 3.8.1 Prescaler Clock Resolution into 8- and 16-Bit Timers

at fc = 20 MHz, fs = 32.768 kHz

Select System Clock <SYSCK>	Select Prescaler Clock <PRCK1:0>	Gear Value <GEAR2:0>	Prescaler 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)	$fc/2^3$ (0.4 μs)	$fc/2^5$ (1.6 μs)	$fc/2^7$ (6.4 μs)	$fc/2^{11}$ (102.4 μs)
		001 (fc/2)	$fc/2^4$ (0.8 μs)	$fc/2^6$ (3.2 μs)	$fc/2^8$ (12.8 μs)	$fc/2^{12}$ (204.8 μs)
		010 (fc/4)	$fc/2^5$ (1.6 μs)	$fc/2^7$ (6.4 μs)	$fc/2^9$ (25.6 μs)	$fc/2^{13}$ (409.6 μs)
		011 (fc/8)	$fc/2^6$ (3.2 μs)	$fc/2^8$ (12.8 μs)	$fc/2^{10}$ (51.2 μs)	$fc/2^{14}$ (819.2 μs)
		100 (fc/16)	$fc/2^7$ (6.4 μs)	$fc/2^9$ (25.6 μs)	$fc/2^{11}$ (102.4 μs)	$fc/2^{15}$ (1.6384 ms)
XXX	01 (low-frequency clock)	XXX	$fs/2^3$ (244 μs)	$fs/2^5$ (977 μs)	$fs/2^7$ (3.9 ms)	$fs/2^{11}$ (62.5 ms)
XXX	10 (fc/16 clock)	XXX	$fc/2^7$ (6.4 μs)	$fc/2^9$ (25.6 μs)	$fc/2^{11}$ (102.4 μs)	$fc/2^{15}$ (1.6384 ms)
XXX: Don't care			← 16-bit timer →			
			← 8-bit timer →			

Note: The fc/16 clock cannot be used as a prescaler clock when the fs is used as a system clock.

The clock selected from among f_{FPH}, fc/16, and fs is divided by 4 and input to this prescaler. This selection is made by prescaler clock selection register SYSCR0 <PRCK1:0>.

Resetting sets <PRCK1:0> to 00, selecting the f_{FPH} clock input divided by 4.

The 16-bit timers 4 and 6 select among 3 clock inputs: ϕT1, ϕT4, and ϕT16 among the prescaler outputs.

This prescaler can be run or stopped by the timer operation control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to 1. The prescaler is cleared to 0 and stops operation when <PRRUN> is set to 0.

Resetting clears <PRRUN> to 0 and stops the prescaler.

When the IDLE1 mode (in which only the oscillator operates) is used, set TRUN<PRRUN> to 0 to reduce the power consumption of the prescaler before the HALT instruction is executed.

(2) Up counter

The up counter is a 16-bit binary counter which counts up according to the input clock specified by T4MOD<T4CLK1:0> register.

The alternatives for selecting the input clock include any one of the internal clocks ϕT1, ϕT4, and ϕT16 from the 9-bit prescaler (which is also used as an 8-bit timer), as will as the external clock from the TI4 pin (which itself can also be used as the P40 or KEY0 pin). When reset, <T4CLK1:0> will be initialized to 00; this selects TI4 input mode. Counting or stop and clear of the counter are controlled by timer operation control register TRUN<T4RUN>.

When clearing is enabled, up counter UC4 will be cleared each time it matches the timer register TREG5. The "clear enable/disable" setting is made by T4MOD<CLE>.

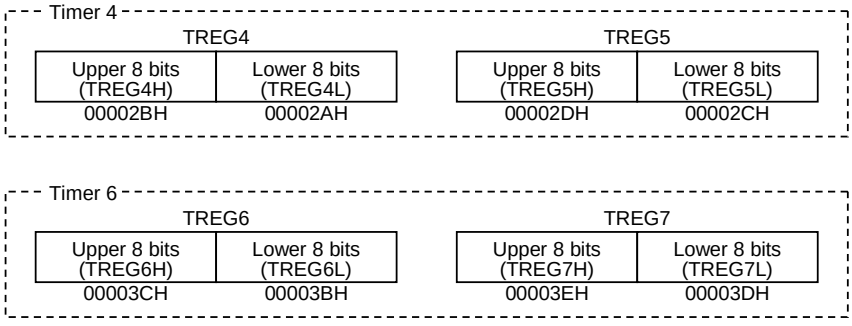
If clearing is disabled, the counter operates as a free-running counter.

When an overflow of UC4 occurs, the interrupt request INTTO4 occurs.

(3) Timer registers

These two 16-bit registers are used to set the interval time. When the value of up counter UC4 matches the value set in this timer register, the comparator match detect signal will be activated.

Setting data in the both upper and lower timer registers TREG4 and TREG5 is always needed. For example, either by using a 2-byte data transfer instruction, or by using 1-byte data transfer instructions twice: once for the lower 8 bits and once for the upper 8 bits in that order.



TREG4 to TREG7 can not be read out, because they are write only registers.

The TREG4 timer register is a double buffer structure, which is paired with a register buffer. The timer control register T16CR<DB4EN> controls whether the double buffer structure should be enabled or disabled. It is disabled when <DB4EN> = 0, and enabled when <DB4EN> = 1.

When the double buffer is enabled, the timing of data transfer from the register buffer to the timer register is at the match between the up counter (UC4) and timer register TREG5.

After reset, TREG4 and TREG5 are undefined. When using a 16-bit timer, write the data beforehand.

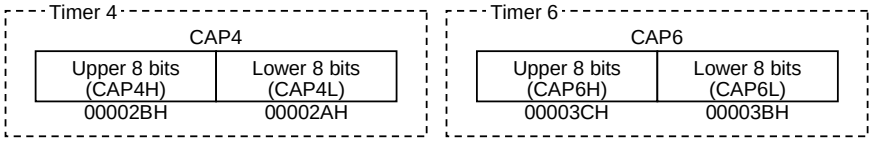
When reset, <DB4EN> will be initialized to 0; this disables the double buffer. To use the double buffer, write data in the timer register, set <DB4EN> = 1, and then write the data which follows to the register buffer.

TREG4 and the register buffer are allocated to the same memory addresses 00002AH/00002BH. When <DB4EN> = 0, the same value will be written in both the timer register and in the register buffer. When <DB4EN> = 1, the value is written only into the register buffer. Therefore, to write the initial value into the timer register, the register buffer should be disabled.

(4) Capture registers

These 16-bit registers are used to hold the values of the up counter.

Data in the capture registers should be read all 16 bits. For example, using a 2-byte data load instruction or by two 1-byte data load instructions, starting from the lower 8 bits followed by the upper 8 bits.



CAP4 and CAP6 can not be written by software, because they are read only registers.

(5) Chapter input control

The value of up counter can be loaded to capture registers by software. Whenever 0 is written in T4MOD<CAP1IN> the current value of up counter will be loaded to capture register CAP4. It is necessary to keep the prescaler in RUN mode (TRUN<PRRUN> to be 1).

(6) Comparator

There are 16-bit comparators which compare the up counter UC4 value with the values set in TREG4 and TREG5 to detect matches. When a match is detected, these comparators generate interrupts INTTR4 or INTTR5 respectively. The up counter UC4 is cleared only when UC4 matches TREG5. The clearing of up counter UC4 can be disabled by setting T4MOD<CLE> = 0.

(7) Timer flip-flop (TFF4)

This flip-flop is inverted by the match detect signal from the comparators. Disable or enable of inversion can be set for each element by T4FFCR<EQ5T4 and EQ4T4>. TFF4 is undefined on resetting. TFF4 will be inverted when 00 is written in T4FFCR<TFF4C1:0>. Also it is set to 1 when 01 is written, and cleared when 10 is written. The value of TFF4 can be output to the timer output pin TO4 (which is also used as P41 or KEY1).

When using the TFF4 contents as the timer output, the timer flip-flop should be set by the port 4 function register P4FC beforehand (See registers for port 4).

a. 16-bit interval timer mode

Generating interrupts at fixed intervals:

In this example, the interval time is set in the timer register TREG5 to generate the interrupt INTTR5.

		7	6	5	4	3	2	1	0	
TRUN	←	–	X	–	0	–	–	–	–	Stop timer 4.
INTET54	←	1	1	0	0	1	0	0	0	Enable INTTR5 and set interrupt level 4. Disable INTTR4.
T4FFCR	←	X	X	0	0	0	0	1	1	Disable trigger.
T4MOD	←	0	0	1	0	0	1	*	*	Select internal clock for input and disable the capture function.
					(** = 01, 10, 11)					
TREG5	←	*	*	*	*	*	*	*	*	Set the interval time (16 bits).
		*	*	*	*	*	*	*	*	
TRUN	←	1	X	–	1	–	–	–	–	Start timer 4.

X: Don't care, –: No change

b. 16-bit event counter mode

In 16-bit timer mode as described in (a.) above, the timer can be used as an event counter by selecting the external clock (TI4 pin input) as the input clock. The counter counts at the rising edge of the TI4 pin input. To read the value of the counter, first perform "software capture" once and read the captured value.

The TI4 pin can also be used as P40 or KEY0. However these functions can not be selected.

Starting timer 4 sets always event counting operation.

		7	6	5	4	3	2	1	0	
TRUN	←	–	X	–	0	–	–	–	–	Stop timer 4.
P4CR	←	–	–	–	–	–	–	–	0	Set P40 to input mode.
INTET54	←	1	1	0	0	1	0	0	0	Enable INTTR5 and sets interrupt level 4, while disables INTTR4.
T4FFCR	←	X	X	0	0	0	0	1	1	Disable trigger.
T4MOD	←	0	0	1	0	0	1	0	0	Select TI4 as the input clock.
TREG5	←	*	*	*	*	*	*	*	*	Set the number of counts (16 bits).
		*	*	*	*	*	*	*	*	
TRUN	←	1	X	–	1	–	–	–	–	Start timer 4.

X: Don't care, –: No change

When using this set up as an event counter, set the prescaler in RUN mode.
(TRUN<PRRUN> = 1)

c. 16-bit programmable pulse generation (PPG) output mode

Square wave pulse can be generated at any frequency and duty by timer 4. The output pulse may be either low-active or high-active.

Timer 4 outputs a pulse to the TO4 pin.

In this mode, a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC4) matches the timer registers TREG4 and TREG5. However, the following conditions must be satisfied:

(Set value of TREG4) < (Set value of TREG5)

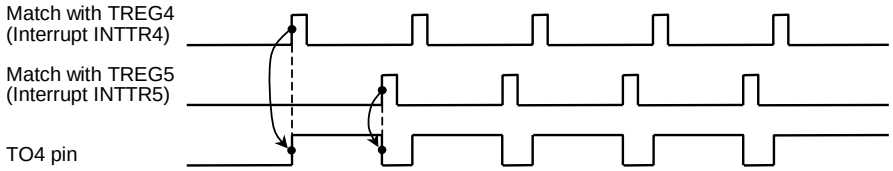


Figure 3.8.10 Programmable Pulse Generation (PPG) Output Waveforms

When the double buffer of TREG4 is enabled in this mode, the value of register buffer 4 will be shifted into TREG4 on finding a match with TREG5. This feature makes the handling of low duty waves easy.

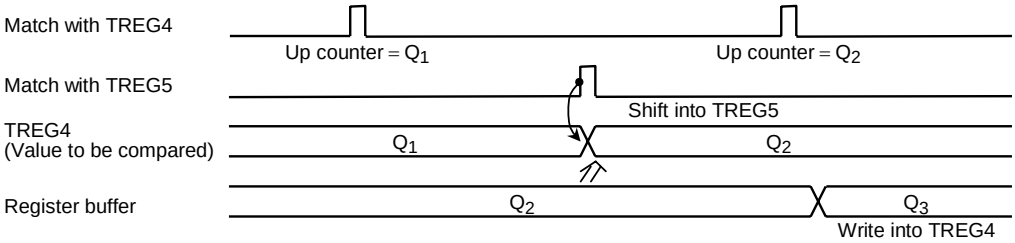


Figure 3.8.11 Operation of Register Buffer

Figure 3.8.12 shows the block diagram of this mode.

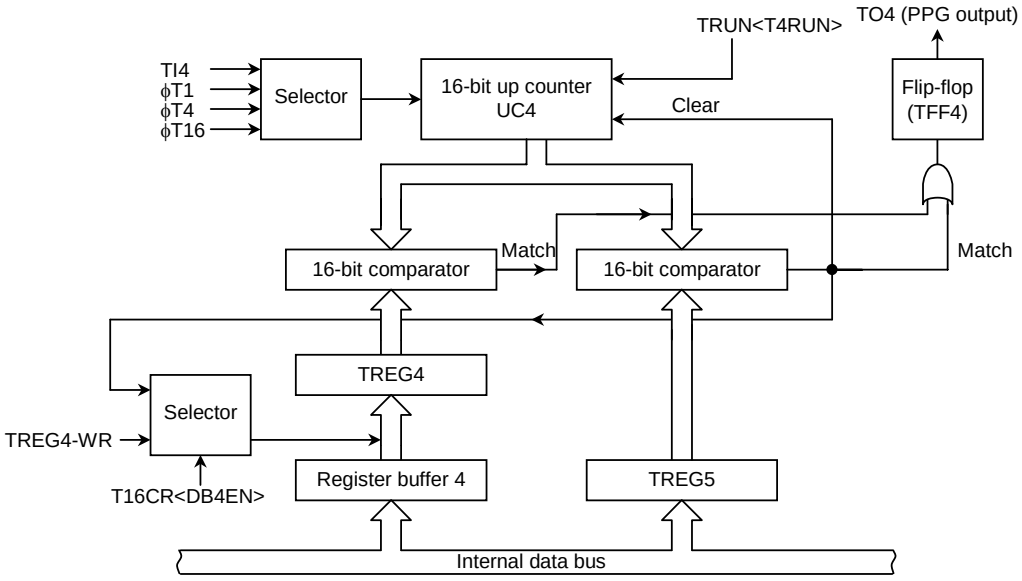


Figure 3.8.12 Block Diagram of 16-Bit PPG Mode

		7	6	5	4	3	2	1	0	
T16CR	←	0	X	X	X	–	–	–	0	Disable double buffer of TREG4.
TRUN	←	–	X	–	0	–	–	–	–	Stop timer 4.
TREG4	←	*	*	*	*	*	*	*	*	Set the duty (16 bits).
		*	*	*	*	*	*	*	*	
TREG5	←	*	*	*	*	*	*	*	*	Set the cycle (16 bits).
		*	*	*	*	*	*	*	*	
T16CR	←	0	X	X	X	–	–	–	1	Enable double buffer of TREG4. (Change the duty and cycle at the interrupt INTTR5.)
T4FFCR	←	X	X	0	0	1	1	1	0	Set the mode to invert TFF4 at the match with TREG4 or TREG5, and also set TFF4 to 0.
T4MOD	←	0	0	1	0	0	1	*	*	Select the internal clock for the input, and disable the capture function.
					(** = 01, 10, 11)					
P4CR	←	–	–	–	–	–	–	1	–	Assign P41 as TO4.
P4FC	←	X	X	X	X	–	X	1	X	
TRUN	←	1	X	–	1	–	–	–	–	Start timer 4.

X: Don't care, –: No change

3.8.2 Timer/Event Counter 8 and A

Each of timer/event counter 8 and A consists of a 16-bit up counter, two 16-bit timer registers (One with a double buffer), two 16-bit capture registers, two comparators, a capture input controller, a timer flip-flop and the control circuit.

Timer/event counters are controlled by 4 control registers: T8MOD/TAMOD, T8FFCR/TAFFCR, TRUN, and T16CR.

Figure 3.8.13 and Figure 3.8.14 show the block diagram of the 16-bit timer/event counters (Timer 8 and timer A).

Timers 8 and A can be used independently.

All timers operate in the same manner, and thus only the operation of timer 8 will be explained below.

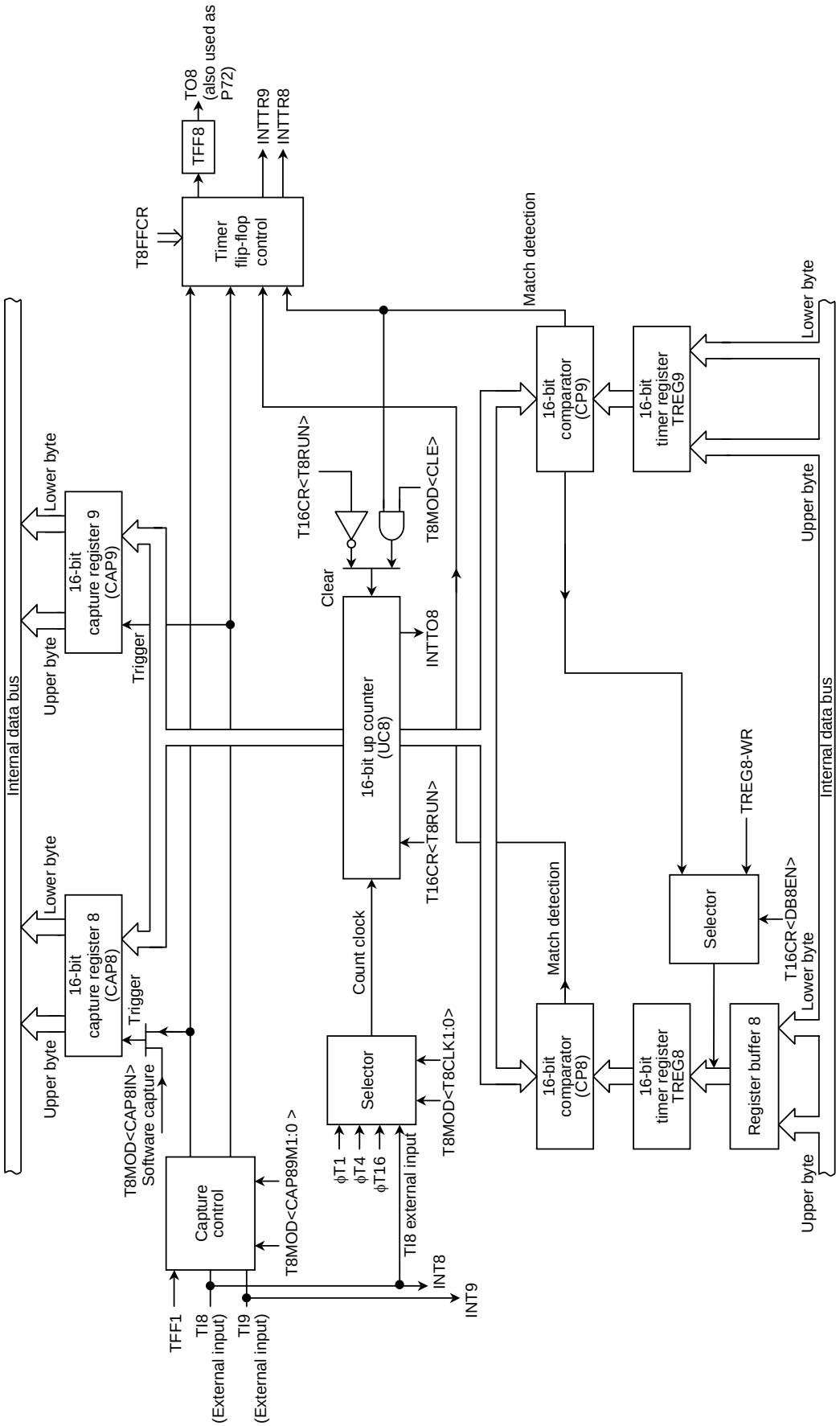


Figure 3.8.13 Block Diagram of 16-Bit Timer (Timer 8)

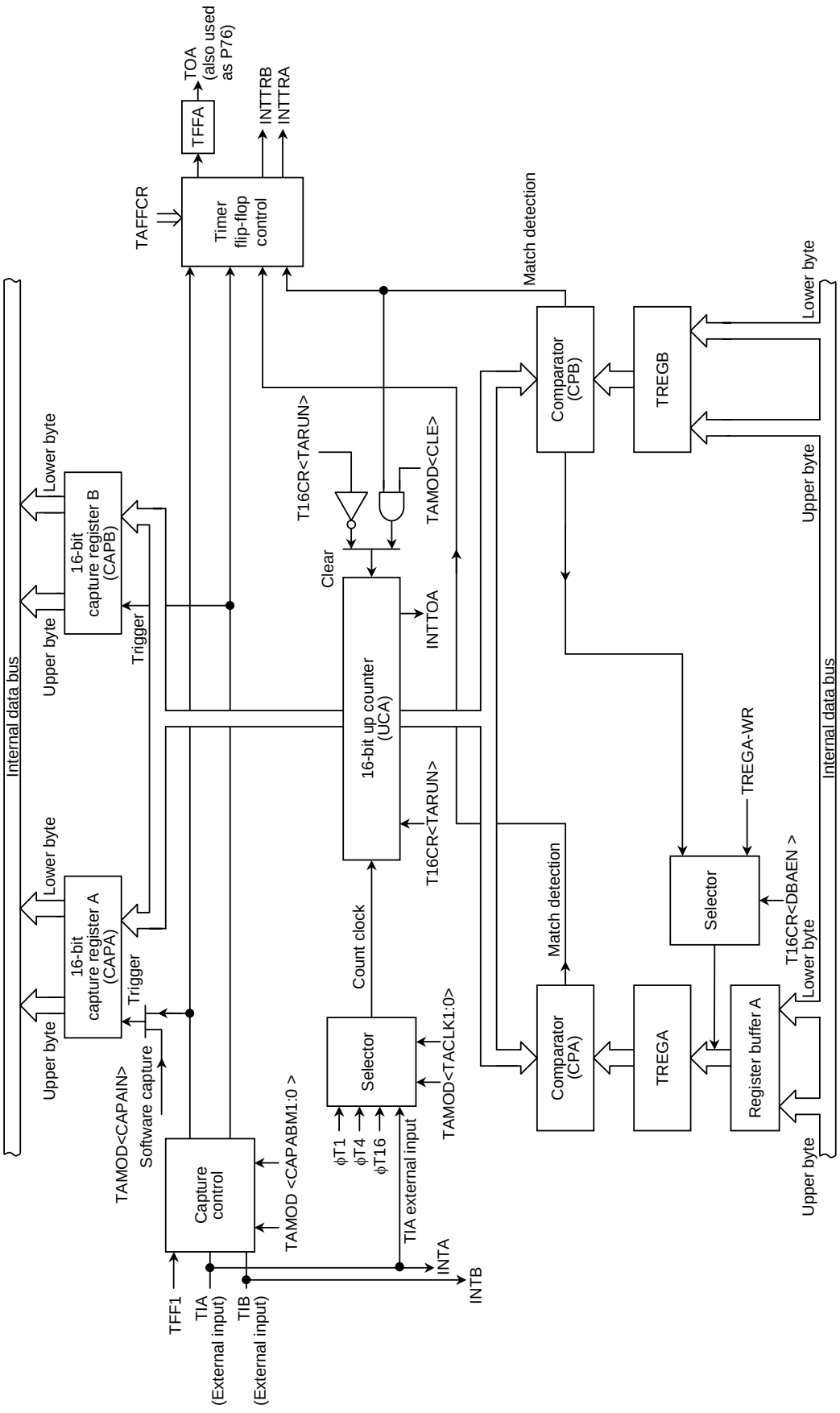


Figure 3.8.14 Block Diagram of 16-Bit Timer (Timer A)

Timer 8 Mode Control Register

T8MOD (0038H)

	7	6	5	4	3	2	1	0
Bit symbol			CAP8IN	CAP89M1	CAP89M0	CLE	T8CLK1	T8CLK0
Read/Write			W	R/W				
After reset			1	0	0	0	0	0
Function			Control of software capture 0: Software capture 1: Don't care	Capture timing 00: Disable INT8 occurs at rising edge. 01: TI8↑ TI9↑ INT8 occurs at rising edge. 10: TI8↑ TI8↓ INT8 occurs at falling edge. 11: TFF1↑ TFF1↓ INT8 occurs at rising edge.		Control of timer 8 0: Clear disable 1: Clear enable	Timer 8 source clock 00: TI8 01: φT1 10: φT4 11: φT16	

Timer 8 input clock

00	External clock (TI8)
01	φT1
10	φT4
11	φT16

Clearing the up counter UC8

0	Clear disable
1	Clear by match with TREG9

Capture timing of timer 8

	Capture control	INT8 control
00	Capture disable	Interrupt occurs at the rising edge of TI8 (INT8) input.
01	CAP8 at TI8 rise CAP9 at TI9 rise	
10	CAP8 at TI8 rise CAP9 at TI8 fall	Interrupt occurs at the faling edge of TI8 (INT8) input.
11	CAP8 at TFF1 rise CAP9 at TFF1 fall	Interrupt occurs at the rising edge of TI8 (INT8) input.

Software capture

0	The up counter 8 value is loaded to CAP8
1	Don't care

Figure 3.8.15 Registers for 16-Bit Timer/Event Counter (1/4)

Timer 8 Flip-flop Control Register

T8FFCR (0039H)

	7	6	5	4	3	2	1	0
Bit symbol			CAP9T8	CAP8T8	EQ9T8	EQ8T8	TFF8C1	TFF8C0
Read/Write			R/W				W	
After reset			0	0	0	0	1	1
Function			TFF8 invert trigger 0: Disable trigger 1: Enable trigger				TFF8 control 00: Invert TFF8 01: Set TFF8 10: Clear TFF8 11: Don't care * Always read as 11	
			Inverted when the UC value is latched to CAP9	Inverted when the UC value is latched to CAP8	Inverted when the UC value matches TREG9	Inverted when the UC value matches TREG8		

Timer flip-flop 8 (TFF8) control

00	Inverts the TFF8 value (Software inversion).
01	Sets TFF8 to 1.
10	Clear TFF8 to 0.
11	Don't care

Timer flip-flop 8 (TFF8) invert trigger when up counter matches TREG8

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

Timer flip-flop 8 (TFF8) invert trigger when up counter matches TREG9

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

Timer flip-flop 8 (TFF8) invert trigger when the up counter value is latched to CAP8

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

Timer flip-flop 8 (TFF8) invert trigger when the up counter value is latched to CAP9

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

Figure 3.8.16 Registers for 16-Bit Timer/Event Counter (2/4)

Timer A Mode Control Register

TAMOD (0048H)

	7	6	5	4	3	2	1	0
Bit symbol			CAPAIN	CAPABM1	CAPABM0	CLE	TACLK1	TACLK0
Read/Write			W	R/W				
After reset			1	0	0	0	0	0
Function			Control of software capture 0: Software capture 1: Don't care	Capture timing 00: Disable INTA occurs at rising edge. 01: TIA↑ TIB↑ INTA occurs at rising edge. 10: TIA↑ TIA↓ INTA occurs at falling edge. 11: TFF1↑ TFF1↓ INTA occurs at rising edge.		Control of timer A 0: Clear disable 1: Clear enable	Timer A source clock 00: TIA 01: φT1 10: φT4 11: φT16	

Timer A input clock

00	External clock (TIA)
01	φT1
10	φT4
11	φT16

Clearing the up counter UCA

0	Clear disable
1	Clear by match with TREGB

Capture timing of timer A

	Capture control	INTA control
00	Capture disable	Interrupt occurs at the rising edge of TIA (INTA) input.
01	CAPA at TIA rise CAPB at TIB rise	
10	CAPA at TIA rise CAPB at TIA fall	Interrupt occurs at the faling edge of TIA (INTA) input.
11	CAPA at TFF1 rise CAPB at TFF1 fall	Interrupt occurs at the rising edge of TIA (INTA) input.

Software capture

0	The up counter A value is loaded to CAPA
1	Don't care

Figure 3.8.17 Registers for 16-Bit Timer/Event Counter (3/4)

Timer A Flip-flop Control Register

TAFFCR (0049H)

	7	6	5	4	3	2	1	0
Bit symbol			CAPBTA	CAPATA	EQBTA	EQATA	TFFAC1	TFFAC0
Read/Write			R/W				W	
After reset			0	0	0	0	1	1
Function			TFFA invert trigger 0: Enable trigger 1: Disable trigger				TFFA control 00: Invert TFFA 01: Set TFFA 10: Clear TFFA 11: Don't care * Always read as 11	
			Inverted when the UC value is latched to CAPB	Inverted when the UC value is latched to CAPA	Inverted when the UC value matches TREGB	Inverted when the UC value matches TREGA		

Timer flip-flop A (TFFA) control

00	Inverts the TFFA value (Software inversion).
01	Sets TFFA to 1.
10	Clear TFFA to 0.
11	Don't care

Timer flip-flop A (TFFA) invert trigger when up counter matches TREGA

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

Timer flip-flop A (TFFA) invert trigger when up counter matches TREGB

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

Timer flip-flop A (TFFA) invert trigger when the up counter value is latched to CAPA

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

Timer flip-flop A (TFFA) invert trigger when the up counter value is latched to CAPB

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

Figure 3.8.18 Registers for 16-Bit Timer/Event Counter (4/4)

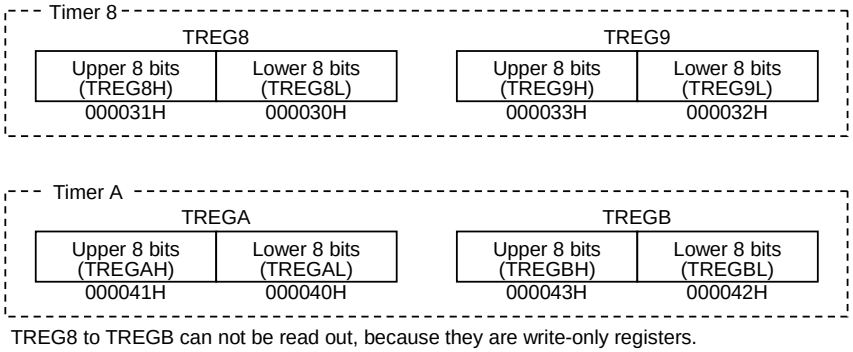
The 16-bit timer control register, the timer operation control register, and the system clock control registers operate in the same way as that of the timer/event counters 4 and 6 described in section 3.8.1. See figure 3.8.3.

(1) Prescaler and (2) Up counter have the same configuration as that of the timer/event counter 4 and 6 described in section 3.8.1.

(3) Timer register

These two 16-bit registers are used to set the interval time. When the value of up counter UC8 matches the value set in this timer register, the comparator match detect signal will be activated.

Setting data in the both upper and lower timer registers TREG8 and TREG9 is always needed. For example, either by using a 2-byte data transfer instruction, or by using 1-byte data transfer instructions twice: once for the lower 8 bits and once for the upper 8 bits in that order.



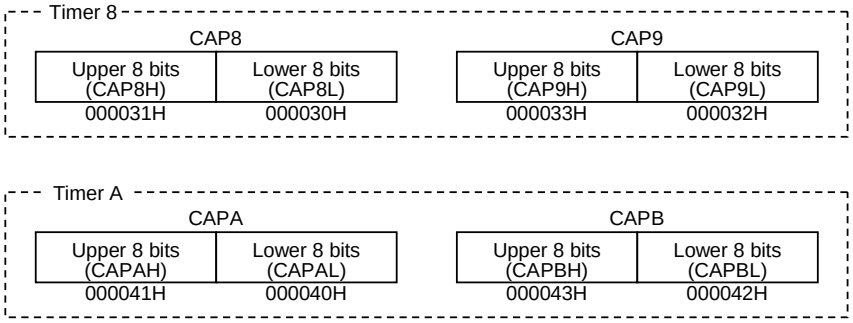
TREG8 to TREGB can not be read out, because they are write-only registers.

The configuration of this timer register is the same as that of the timer/event counters 4 and 6 described in section 3.8.1.

(4) Capture registers

These 16-bit registers are used to hold the values of the up counter.

Data in the capture registers should be read all 16 bits. For example, using a 2-byte data load instruction or by two 1-byte data load instructions, starting from the lower 8 bits followed by the upper 8 bits.



CAP8 to CAPB can not be written by software, because they are read-only registers.

(5) Capture input control

This circuit controls the timing of latching the value of up counter UC8 into CAP8 and CAP9. The latch timing of the capture register is controlled by register T8MOD<CAP89M1:0>. There are four possible settings:

- When T8MOD<CAP89M1:0> = 00

  Capture function is disabled. Disable is the default on resetting.

- When T8MOD<CAP89M1:0> = 01

  Data are loaded to CAP8 at the rising edge of the TI8 pin (which is also used as P70 or INT8) input, while data are loaded to CAP9 at the rising edge of the TI9 pin (also used as P71 or INT9) input.

- When T8MOD<CAP89M1:0> = 10

  Data are loaded to CAP8 at the rising edge of the TI8 pin input, and to CAP9 at the falling edge. Only in this setting, interrupt INT8 occurs at the falling edge.

- When T8MOD<CAP89M1:0> = 11

  Data are loaded to CAP8 at the rising edge of timer flip-flop TFF1, and to CAP9 at the falling edge.

Besides, the value of the up counter can be loaded to capture registers by software. Whenever 0 is written in T8MOD<CAP8IN>, the current value of the up counter will be loaded to capture register CAP8. It is necessary to keep the prescaler in RUN mode (TRUN<PRRUN> has to be 1).

(6) Comparator

There are 16-bit comparators which compare the up counter UC8 value with the values set in TREG8 and TREG9 to detect matches. When a match is detected, these comparators generate interrupts INTTR8 or INTTR9 respectively. The up counter UC8 is cleared only when UC8 matches TREG9. The clearing of up counter UC8 can be disabled by setting T8MOD<CLE> = 0.

(7) Timer flip-flop (TFF8)

This flip-flop is inverted by the match detect signal from the comparators or the latch signals to the capture registers. Disable or enable of inversion can be set for each element by T8FFCR<CAP9T8, CAP8T8, EQ9T8, and EQ8T8>. TFF8 will be inverted when 00 is written in T8FFCR<TFF8C1:0>. Also it is set to 1 when 01 is written, and cleared when 10 is written. The value of TFF8 can be output to the timer output pin TO8 (which is also used as P72). To output to the timer output pin, TFF8 must be set by the Port 7 function register, P7FC beforehand (See “Registers for Port 7”).

a. 16-bit interval timer mode

Generating interrupts at fixed intervals:

In this example, the interval time is set in the timer register TREG9 to generate the interrupt INTTR9.

	7	6	5	4	3	2	1	0	
T16RUN	← –	X	–	0	–	–	–	–	Stop timer 8.
INTET98	← 1	1	0	0	1	0	0	0	Enable INTTR9 and set interrupt level 4. Disable INTTR8.
T8FFCR	← X	X	0	0	0	0	1	1	Disable trigger.
T8MOD	← 0	0	1	0	0	1	*	*	Select internal clock for input and disable the capture function. (** = 01, 10, 11)
TREG9	← *	*	*	*	*	*	*	*	Set the interval time (16 bits).
	*	*	*	*	*	*	*	*	
TRUN	← 1	X	–	–	–	–	–	–	Start prescaler.
T16CR	← –	X	–	1	–	–	–	–	Start timer 8.

X: Don't care, –: No change

b. 16-bit event counter mode

In 16-bit timer mode as described in (a.) above, the timer can be used as an event counter by selecting the external clock (TI8 pin input) as the input clock. To read the value of the counter, first perform "software capture" once and read the captured value.

The counter counts at the rising edge of the TI8 pin input.

The TI8 pin can also be used as P70 or INT8.

However these function can not be selected.

Starting timer 8 set always event counter.

	7	6	5	4	3	2	1	0	
T16CR	← –	X	–	0	–	–	–	–	Stop timer 8.
P7CR	← –	–	–	–	–	–	–	0	Set P70 to input mode.
INTET98	← 1	1	0	0	1	0	0	0	Enable INTTR9 and sets interrupt level 4, while disables INTTR8.
T8FFCR	← X	X	0	0	0	0	1	1	Disable trigger.
T8MOD	← 0	0	1	0	0	1	0	0	Select TI8 as the input clock.
TREG9	← *	*	*	*	*	*	*	*	Set the number of counts (16 bits).
	*	*	*	*	*	*	*	*	
TRUN	← 1	X	–	–	–	–	–	–	Start prescaler.
T16CR	← –	X	–	1	–	–	–	–	Start timer 8.

X: Don't care, –: No change

When using this set up as an event counter, set the prescaler in RUN mode.

c. 16-bit programmable pulse generation (PPG) output mode

Square wave pulse can be generated at any frequency and duty by timer 8.

The output pulse may be either low-active or high-active.

The PPG mode is obtained by inversion of the timer flip-flop TFF8 that is enabled by the match of the up counter UC8 with either of the timer registers TREG8 or 9. TFF8 is also output to TO8. In this mode, the following conditions must be satisfied:

(Set value of TREG8) < (Set value of TREG9)

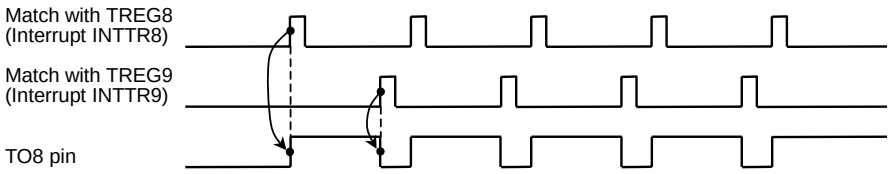


Figure 3.8.19 Programmable Pulse Generation (PPG) Output Waveforms

When the double buffer of TREG8 is enabled in this mode, the value of register buffer 8 will be shifted into TREG8 on finding a match with TREG9. This feature makes the handling of low duty waves easy.

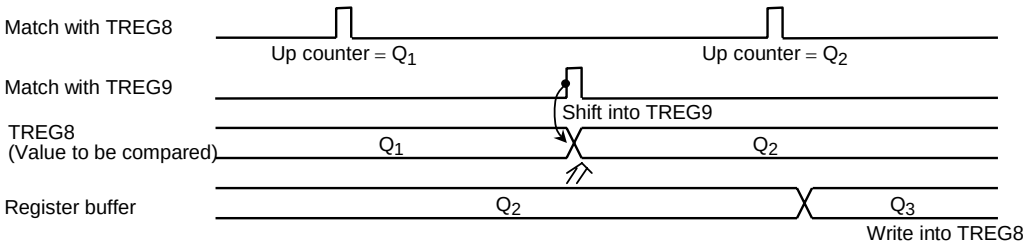


Figure 3.8.20 Operation of Register Buffer

Figure 3.8.21 shows the block diagram of this mode.

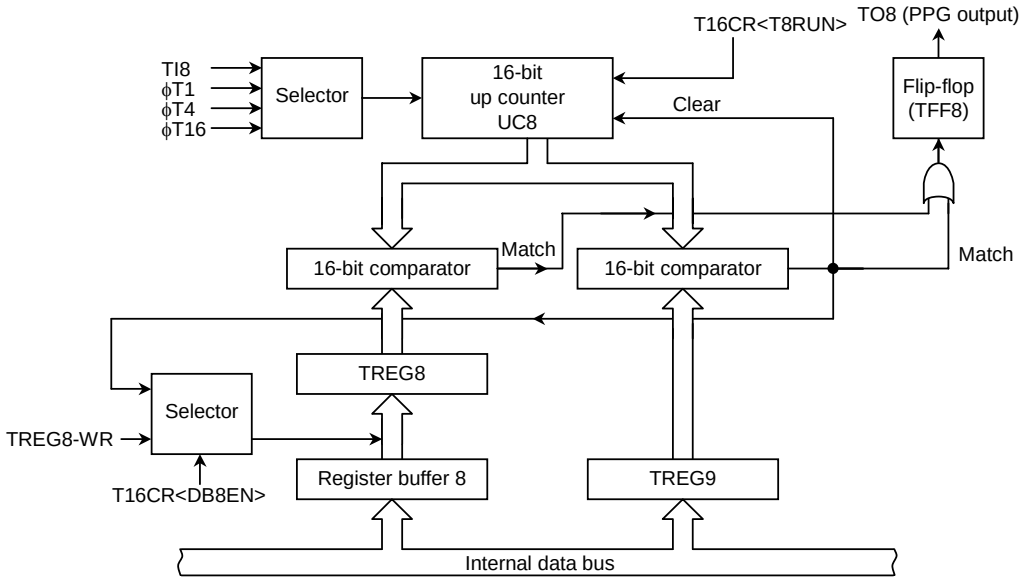


Figure 3.8.21 Block Diagram of 16-Bit PPG Mode

		7	6	5	4	3	2	1	0	
T16CR	←	–	X	–	0	–	0	–	–	Disable double buffer of TREG8. Stop timer 8.
TREG8	←	*	*	*	*	*	*	*	*	Set the duty (16 bits).
		*	*	*	*	*	*	*	*	
TREG9	←	*	*	*	*	*	*	*	*	Set the cycle (16 bits).
		*	*	*	*	*	*	*	*	
T16CR	←	–	X	–	0	–	1	–	–	Enable double buffer of TREG8. (Change the duty and cycle at the interrupt INTTR9.)
T8FFCR	←	X	X	0	0	1	1	1	0	Set the mode to invert TFF8 at the match with TREG8 or TREG9, and also set TFF8 to 0.
T8MOD	←	0	0	1	0	0	1	*	*	Select the internal clock for the input, and disable the capture function.
					(** = 01, 10, 11)					
P7CR	←	–	–	–	–	–	1	–	–	Assign P72 as TO8.
P7FC	←	–	–	X	X	X	1	X	X	
TRUN	←	1	X	–	–	–	–	–	–	Start prescaler.
T16CR	←	–	X	–	1	–	1	–	–	Start timer 8.

X: Don't care, –: No change

d. Application examples of the capture function

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

1. One-shot pulse output from external trigger pulse
2. Frequency measurement
3. Pulse width measurement
4. Time difference measurement

These four application examples are described in detail below.

1. One-shot pulse output from external trigger pulse

To program this application, set the up counter UC8 in free-running mode with the internal input clock, input the external trigger pulse from the TI8 pin, and load the value of the up counter into capture register CAP8 at the rising edge of the TI8 pin.

When the interrupt INT8 is generated at the rising edge of the TI8 input, set the CAP8 value (c) plus a delay time (d) to TREG8 (= c + d), and set the above set value (c + d) plus a one-shot pulse width (p) to TREG9 (= c + d + p). When the interrupt INT8 occurs, the T8FFCR<EQ9T8 and EQ8T8>register should be set to 11 so that the TFF8 inversion is enabled only when the up counter value matches TREG8 or TREG9. When interrupt INTTR9 occurs, this inversion will be disabled.

(c), (d), and (p) are described in Figure 3.8.22 “One-shot Pulse Output (with delay)”.

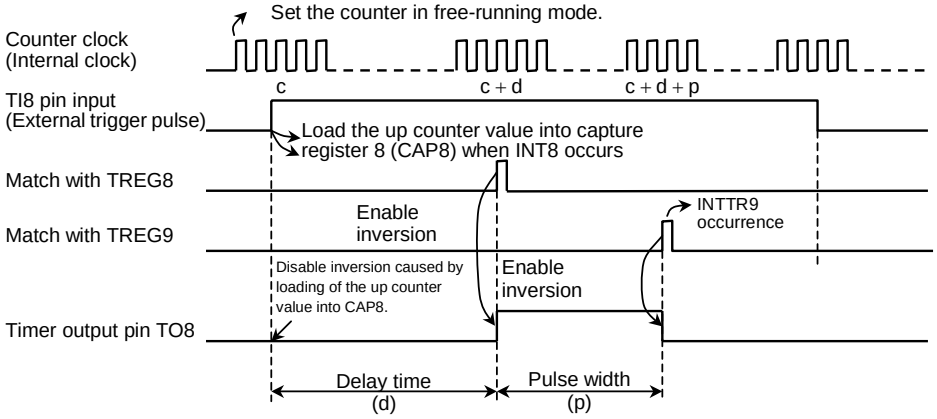
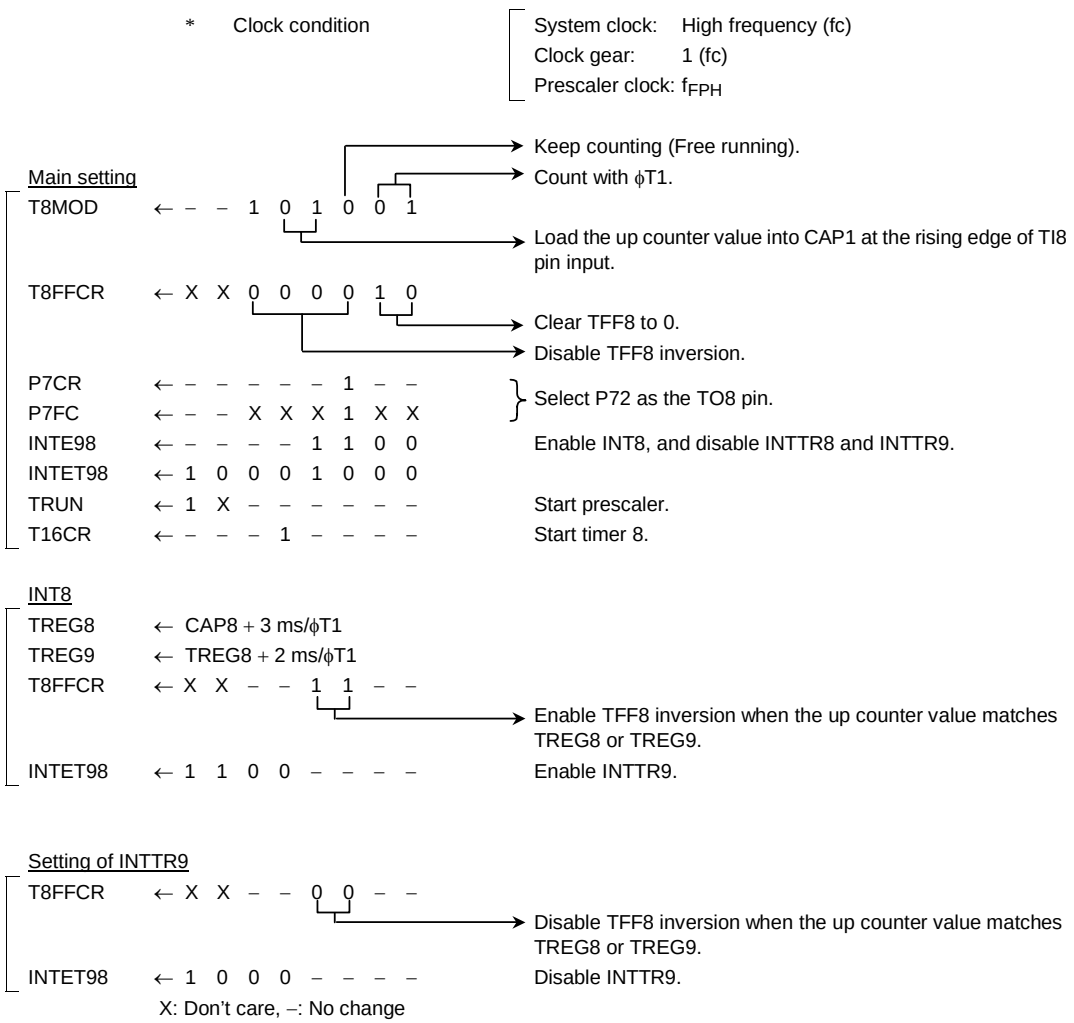


Figure 3.8.22 One-shot Pulse Output (with delay)

Setting example: To output a 2 ms one-shot pulse to the external trigger pulse from TI8 pin, with 3 ms delay



When a delay time is unnecessary, invert the timer flip-flop TFF8 by loading the up counter value into capture register 8 (CAP8). Then set TREG9 to the CAP8 value (c) plus the one-shot pulse width (p) when an INT8 interrupt occurs. The TFF8 inversion should be enabled when the up counter (UC8) value matches TREG9, and disabled when generating the interrupt INTTR9.

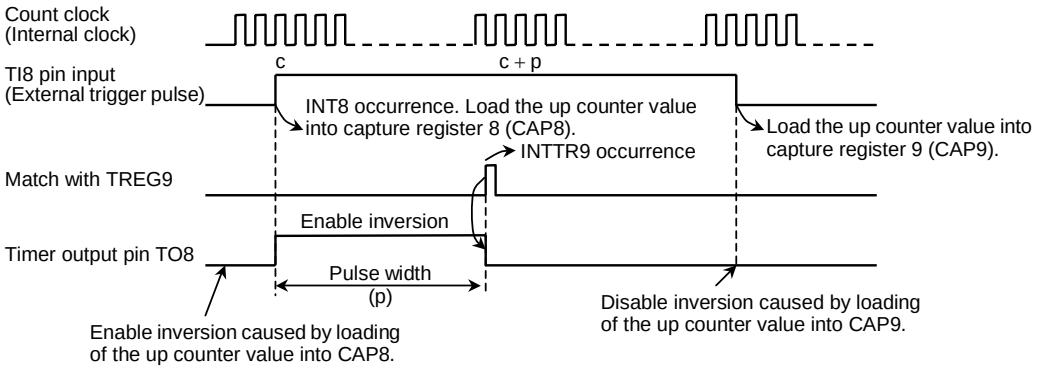


Figure 3.8.23 One-shot Pulse Output (without delay)

2. Frequency measurement

Using the capture function, the frequency of the external clock can be measured in this mode. The frequency is measured by combining the 16-bit event counter mode with the 8-bit timer (Timer 0 and timer 1). (Timer 0 and timer 1 set the measurement time by the TFF1 inversion.)

The TI8 pin input should be selected for the count clock of timer 8. The external clock input starts counting. Timer mode control register, T8MOD<CAP89M1:0> should be set to 11. The value of the 16-bit up counter, UC8 is loaded into the capture register CAP8 at the rising edge of the timer flip-flop TFF1 of the 8-bit timers (Timer 0 and timer 1). Similarly, the up counter value is loaded into CAP9 at the falling edge of the TFF1 flip-flop.

The frequency is calculated by the difference between the values loaded in CAP8 and CAP9, at the moment when an interrupt (INTT0 or INTT1) is generated by either 8-bit timer.

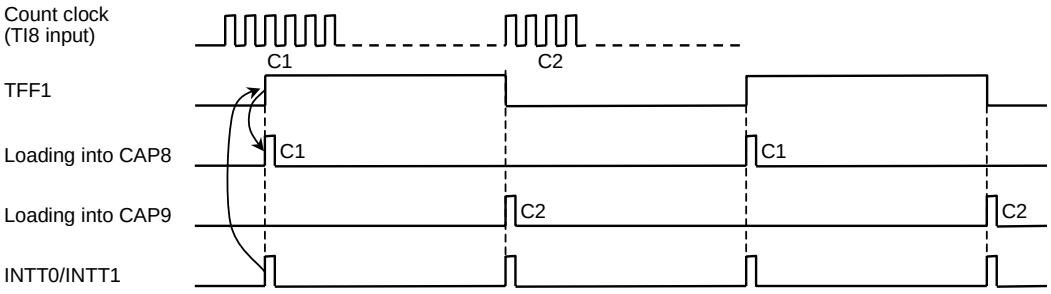


Figure 3.8.24 Frequency Measurement

For example, if the value for the level 1 width of TFF1 of the 8-bit timer is set to 0.5 s and the difference between CAP8 and CAP9 is 100, the frequency will be 100 ÷ 0.5 s = 200 Hz.

3. Pulse width measurement

This mode allows measurement of the H level width of an external pulse. While keeping the 16-bit up counter UC8 counting (Free running) with the internal clock input, the external pulse is input via the TI8 pin. Then the capture function is used to load the UC8 values into CAP8 and CAP9 on the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT8 occurs at the falling edge of TI8.

The pulse width is obtained from the difference between the values of CAP8 and CAP9 and the internal clock cycle.

For example, if the internal clock is 0.8 μs and the difference between CAP8 and CAP9 is 100, the pulse width will be 100 × 0.8 μs = 80 μs.

When measuring the pulse width which exceeds the maximum counting time of UC8 that is determined with the clock source, execute the process of software.

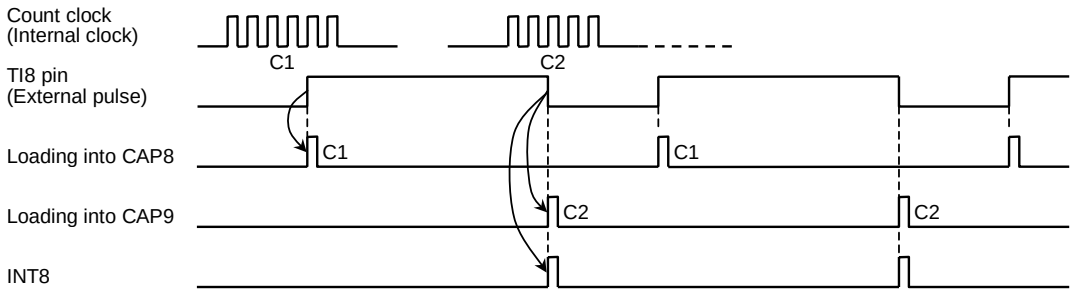


Figure 3.8.25 Pulse Width Measurement

Note: External interrupt INT8 occurs at the falling edge of the TI8 pin input only in this pulse width measuring mode (T8MOD<CAP89M1:0> = 10). In other modes, it occurs at the rising edge.

The width of the L level can be measured from the difference between the first C2 and the second C1 at the second INT8 interrupt. See "Time Difference Measurement" in Figure 3.8.26.

4. Time difference measurement

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

While keeping the 16-bit up counter UC8 counting (Free running) with the internal clock, the UC8 value is loaded into CAP8 on the rising edge of the input pulse to TI8. Then the interrupt INT8 is generated.

Similarly, the UC8 value is loaded into CAP9 on the rising edge of the input pulse to TI9, generating the interrupt INT9.

The time difference between these pulses can be obtained from the difference between the time counts at which loading the up counter value into CAP8 and CAP9 was performed. (= (CAP9 – CAP8) × the internal clock cycle)

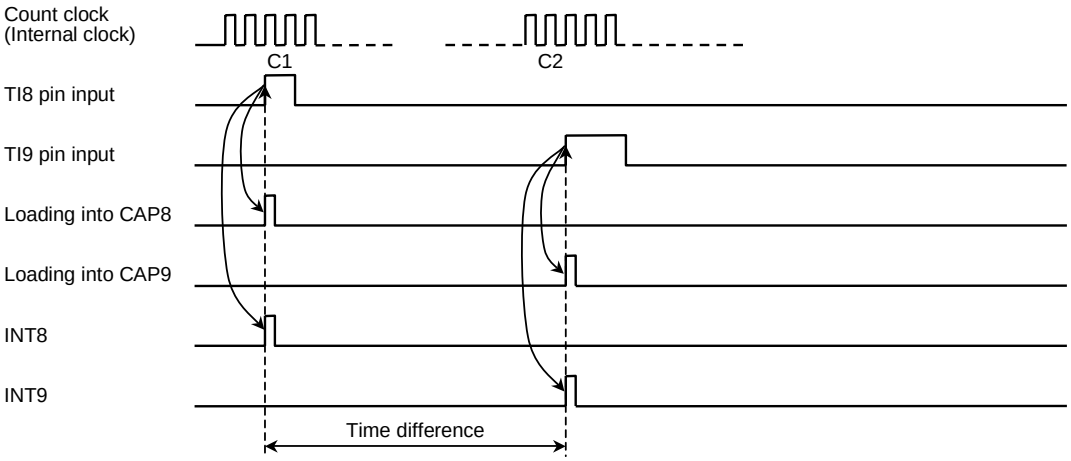


Figure 3.8.26 Time Difference Measurement

3.9 Serial Channel

TMP93CS20 contains 2 serial I/O channels. Channel 0 select UART mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission). Channel 1 is used only in UART mode.

The serial channel has the following operation modes.

- I/O interface mode (Channel 0)
  - Mode 0: To transmit and receive I/O data using the synchronizing signal SCLK for extending I/O.
- UART mode (Channel 0, 1)
  - 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 wake up function for making the master controller start slave controllers in a serial link (Multi-controller) system.

Figure 3.9.1 shows the data format in each mode.

Serial channels 0, 1 can be used independently.

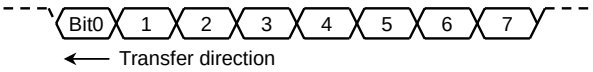
All channels have the same operations except the following points, thus only the operation of channel 0 will be explained below.

Different Points among Channel 0, 1

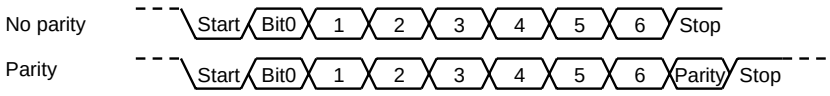
	Channel 0	Channel 1
Pin name	TXD0 (P63) RXD0 (P64) $\overline{\text{CTS0}}$ /SCLK0 (P65)	TXD1 (P80) RXD1 (P81)
UART mode	Yes	Yes
I/O interface mode	Yes	No
Handshake function	Yes	No (No $\overline{\text{CTS1}}$ pin)

Note: Using the handshake function can transmit in units of one data format. Thus over run error is prevented. See “Handshake function” for details.

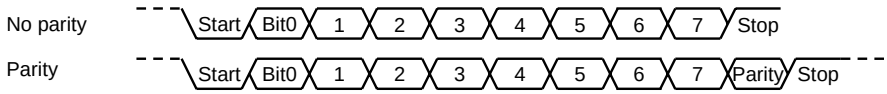
- Mode 0 (I/O interface mode)



- Mode 1 (7-bit UART mode)



- Mode 2 (8-bit UART mode)



- Mode 3 (9-bit UART mode)

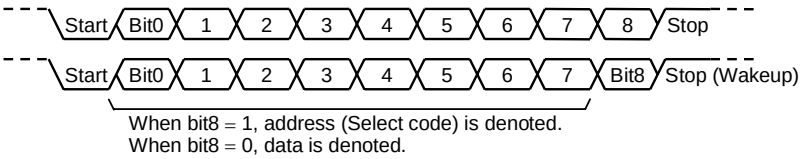


Figure 3.9.1 Data Formats

The serial channel has buffer registers for transmitting and receiving operations in order to temporarily store transmitted or received data. This is done so that transmitting and receiving operations can be done independently (Full duplex).

However, in I/O interface mode, the SCLK (Serial clock) pin is used for both transmitting and receiving, the channel becomes half duplex.

The receiving data register is a double buffer structure to prevent the occurrence of an overrun error and it provides one data format of margin before the CPU reads the received data. The receiving data register stores the previously received data while the buffer register receives the next frame data.

By using $\overline{CTS}$ and $\overline{RTS}$ (There is no $\overline{RTS}$ pin, so any single port must be controlled by software.) it is possible to halt data send until the CPU finishes reading receive data every time a frame is received (Handshake function).

In the UART mode, a check function is added to not start the receiving operation by erroneous start bits due to noise. The channel starts receiving data only when the start bit is detected to be normal at least twice in three samplings of the start bit.

When the transmission buffer becomes empty and requests the CPU to send the next transmission data, or when data is stored in the receiving data register and the CPU is requested to read the data, INTTX (Transmit interrupt) or INTRX (Receive interrupt) interrupt occurs. If an overrun error, parity error, or framing error occurs during receiving operation, flag SC0CR<OERR, PERR, FERR> will be set.

The serial channels 0, 1 include a special baud rate generator, which can set to any baud rate by dividing the frequency of 4 clocks (ϕT0, ϕT2, ϕT8, and ϕT32) from the 9-bit prescaler shared by the 8-bit/16-bit timers by the value 1, 2 + n/16 to 15 + n/16, 16 (n = 0 to 15).

Not only clocks from the internal baud rate generator but also the external input clock (SCLK0) can set to any baud rate. (Serial channel 0)

In I/O interface mode, it is possible to input synchronous signals as well as to transmit or receive data by using an external clock.

3.9.1 Control Registers

The serial channels are controlled by 4 control registers SC0CR, SC0MOD, BR0CR, and BRADD0. Transmitted and received data are stored in register SC0BUF.

Note: The number of the control register name is equaled to the channel number.

SC0MOD (0052H)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	Undefined	0	0	0	0	0	0	0
Function	Transfer data bit8	Hand shake 0: CTS disable 1: CTS enable	Receiving function 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: TO0 trigger 01: Baud rate generator 10: Internal clock ϕ1 11: External clock (SCLK0 input)	

Serial transmission clock source (UART)

00	Timer 0 match detect signal
01	Baud rate generator
10	Internal clock ϕ1
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 length
10		8-bit length
11		9-bit length

Wakeup function

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

Receiving function

0	Receive disable
1	Receive enable

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

0	Disable (Always transferable)
1	Enable

Transmission data bit8

8-bit UART mode (Parity)	Stores transmission parity bit
9-bit UART mode	Stores transmission data bit8

Figure 3.9.2 Serial Mode Control Register (Channel 0, SC0MOD)

SC0CR (0051H)

	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			0: SCLK0 (↑) 1: SCLK0 (↓)	0: Baud rate generator 1: SCLK0 pin input
				Overrun	Parity	Framing		

Select I/O interface input clock

0	Baud rate generater (Note 1)
1	SCLK0 pin input

Edge selection for SCLK pin (Input mode only)

0	Transmits and receives data at rise edge of SCLK0 (↑)
1	Transmits and receives data at fall edge of SCLK0 (↓)

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

Enable parity addition

0	Disable
1	Enable

Addition/check of even parity

0	Odd parity
1	Even parity

Received data bit8

8-bit UART mode (Parity)	Stores received parity bit
9-bit UART mode	Stores received data bit8

Note 1: To use baud rate generator, set TRUN<PRRUN> to 1, putting the prescaler in RUN mode.

Note 2: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.

Figure 3.9.3 Serial Control Register (Channel 0, SC0CR)

BR0CR (0053H)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR0ADD	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Always fixed to 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 division enable

0	Disable
1	Enable

Selecting the input clock of baud rate generator

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

BRADD0 (006BH)

	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

BRADD0 <BR0K3:0> \ BR0CR <BR0S3:0>	BR0CR<BR0ADD> = 1		BR0CR<BR0ADD> = 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	
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by N + (16 – K)/16	Divided by N

Note 1: Set TRUN<PRRUN> to 1 when the baud rate generator is used.

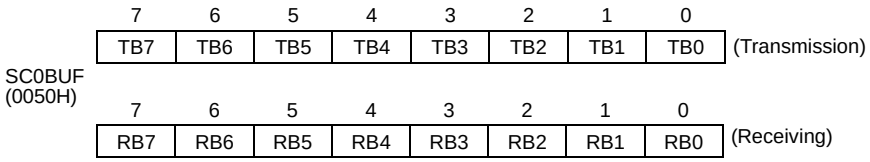
Note 2: Set BR0CE<BR0ADD> to 1 after setting K (K = 1 to 15) to BRADD0<BR0K3:0> when + (16 – K)/16 division function is used. However, don't use + (16 – K)/16 division function when you set <BR0K3:0> to 0000 or 0001.

Note 3: + (16 – K)/16 division function is possible to use in only UART mode.
Set BR0CR<BR0ADD> to 0 and disable + (16 – K)/16 division function in I/O interface mode.

Note 4: BRADD0<Bit7:4> is always read as 1.

Note 5: Don't read from or write to BR0CR and BRADD0 register during sending or receiving.

Figure 3.9.4 Baud Rate Generator Control (Channel 0, BR0CR, BRADD0)



Note: Prohibit read-modify-write for SC0BUF.

Figure 3.9.5 Serial Transmission/Receiving Buffer Registers (Channel 0, BR0CR)

SC1MOD (0056H)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	–	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	Undefined	0	0	0	0	0	0	0
Function	Transferred data bit8	Fix at "0"	Receiving function 0: Receive disable 1: Receive enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: Reserved 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: TO0 trigger 01: Baud rate generator 10: Internal clock ϕ1 11: Don't care	

Serial transmission clock source (For UART)

00	Timer 0 match detect signal
01	Baud rate generator
10	Internal clock ϕ1
11	Don't care

Serial transmission mode

00	Reserved	
01	UART mode	7-bit length
10		8-bit length
11		9-bit length

Wakeup function

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

Receiving control

0	Receive disable
1	Receive enable

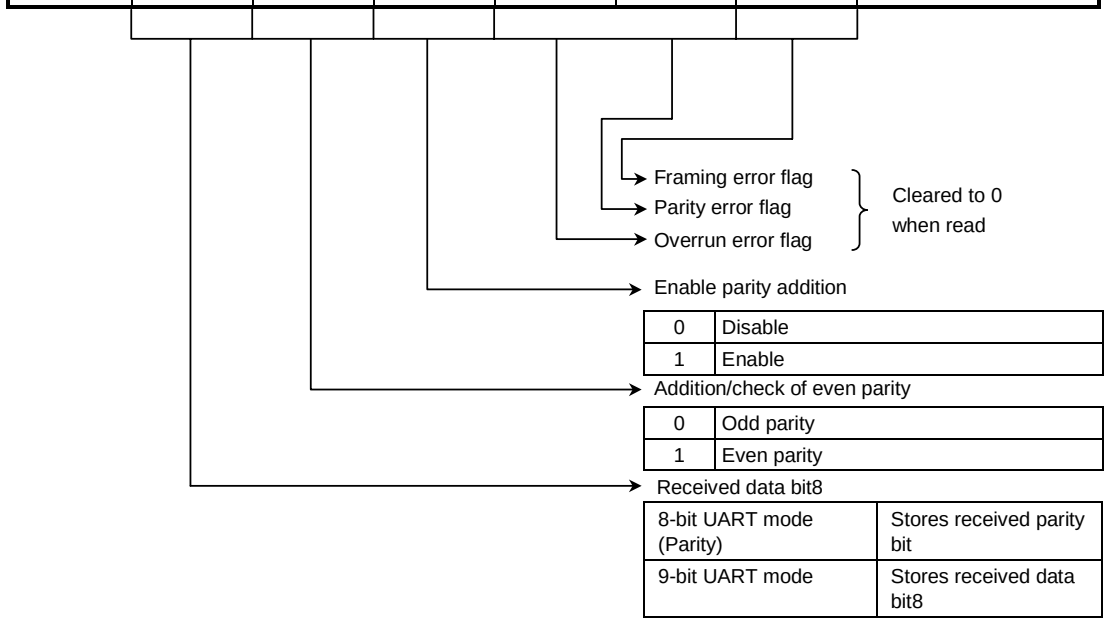
Transmission data bit8

8-bit UART mode (Parity)	Stores transmission parity bit
9-bit UART mode	Stores transmission data bit8

Figure 3.9.6 Serial Mode Control Register (Channel 1, SC1MOD)

SC1CR (0055H)

	7	6	5	4	3	2	1	0
Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	–	–
Read/Write	R	R/W		R (Clear 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			Always fixed to 0	
				Overrun	Parity	Framing		



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

Figure 3.9.7 Serial Control Register (Channel 1, SC1CR)

BR1CR (0057H)

	7	6	5	4	3	2	1	0
Bit symbol	–	BR1ADD	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Always fixed to 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 division enable

0	Disable
1	Enable

Selecting the input clock of baud rate generator

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

BRADD1 (006CH)

	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					Sets frequency divisor "K" (divided by N + (16 – K)/16)			

Sets baud rate generator frequency divisor

BRADD1 <BR1K3:0> \ BR1CR <BR1S3:0>	BR1CR<BR1ADD> = 1		BR1CR<BR1ADD> = 0
	0000 (N = 16) or 0001 (N = 1)	0010 (N = 2) to 1111 (N = 15)	0001 (N = 1) 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 1: Set TRUN<PRRUN> to 1 when the baud rate generator is used.

Note 2: Set BR1CR<BR1ADD> to 1 after setting K (K = 1 to 15) to BRADD1<BR1K3:0> when + (16 – K)/16 division function is used. However, don't use + (16 – K)/16 division function when <BR1K3:0>set to 0000 or 0001.

Note 3: Channel 1 is used only in UART mode, I/O interface mode is not possible to use.

Note 4: BRADD1<Bit7:4> is always read as 1.

Note 5: Don't read from or write to BR1CR and BRADD1 register during sending or receiving.

Figure 3.9.8 Baud Rate Generator Control (Channel 1, BR1CR, BRADD1)

SC1BUF (0054H)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

Note: Prohibit read-modify-write for SC1BUF.

Figure 3.9.9 Serial Transmission/Receiving Buffer Registers (Channel 1, SC1BUF)

P6FC (0016H)

	7	6	5	4	3	2	1	0
Bit symbol	P67F	I7IE	P65F		P63F	P62F	P61F	P60F
Read/Write	W	W	W		W	W	W	W
After reset	0	0	0		0	0	0	0
Function	0: Port 1: TO1	0: Port 1: INT7	0: Port 1: SCLK0		0: Port 1: TXD0	0: Port 1: SCL/SI	0: Port 1: SDA/SO	0: Port 1: SCK

Prohibit read-modify-write.

Setting P63 as TXD0 output

0	Port output
1	TXD0 output (Channel 0)

Setting P65 as SCLK0 output

0	Port output
1	SCLK0 output (Channel 0)

Figure 3.9.10 Port 6 Function Register (P6FC)

P6ODE (0058H)

	7	6	5	4	3	2	1	0
Bit symbol					ODE63	ODE62	ODE61	
Read/Write					R/W			
After reset					0	0	0	
Function					P63 0: CMOS 1: Open drain	P62 0: CMOS 1: Open drain	P61 0: CMOS 1: Open drain	

Setting P63 as open-drain output

0	CMOS output
1	Open-drain output

Setting P61 as open-drain output

0	CMOS output
1	Open-drain output

Setting P62 as open-drain output

0	CMOS output
1	Open-drain output

Note: P6ODE<Bit7:4> and <Bit0> are read as 1.

Figure 3.9.11 Port 6 Open-drain Enable Register (P6ODE)

P8FC (001CH)

	7	6	5	4	3	2	1	0
Bit symbol					P83F			P80F
Read/Write					W			W
After reset					0			0
Function					0: Port 1: TO3			0: Port 1: TXD1

Prohibit read-modify-write.

Setting P80 as TXD1 output

0	Port output
1	TXD1 output (Channel 1)

Figure 3.9.12 Port 8 Function Register (P8FC)

P8ODE (0021H)

	7	6	5	4	3	2	1	0
Bit symbol			ODE85	ODE84	ODE83	ODE82	ODE81	ODE80
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			P85 0: CMOS 1: Open drain	P84 0: CMOS 1: Open drain	P83 0: CMOS 1: Open drain	P82 0: CMOS 1: Open drain	P81 0: CMOS 1: Open drain	P80 0: CMOS 1: Open drain

Setting P80 to P85 as open-drain output

0	CMOS output
1	Open-drain output

Note: P8ODE<Bit7:6> is read as 1.

Figure 3.9.13 Port 8 Open-drain Enable Register (P8ODE)

3.9.2 Configuration

Figure 3.9.14 shows the block diagram of the serial channel 0.

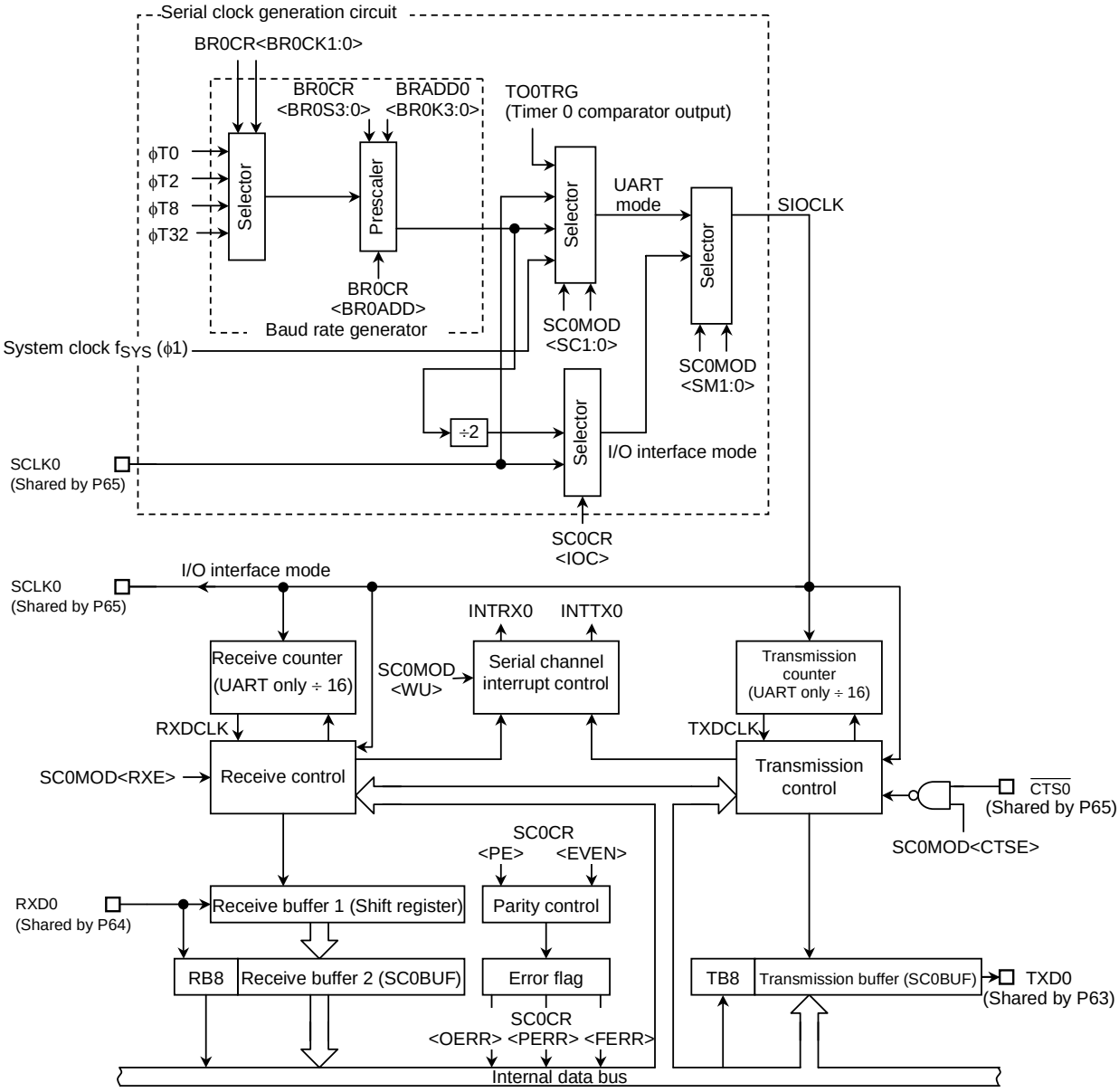
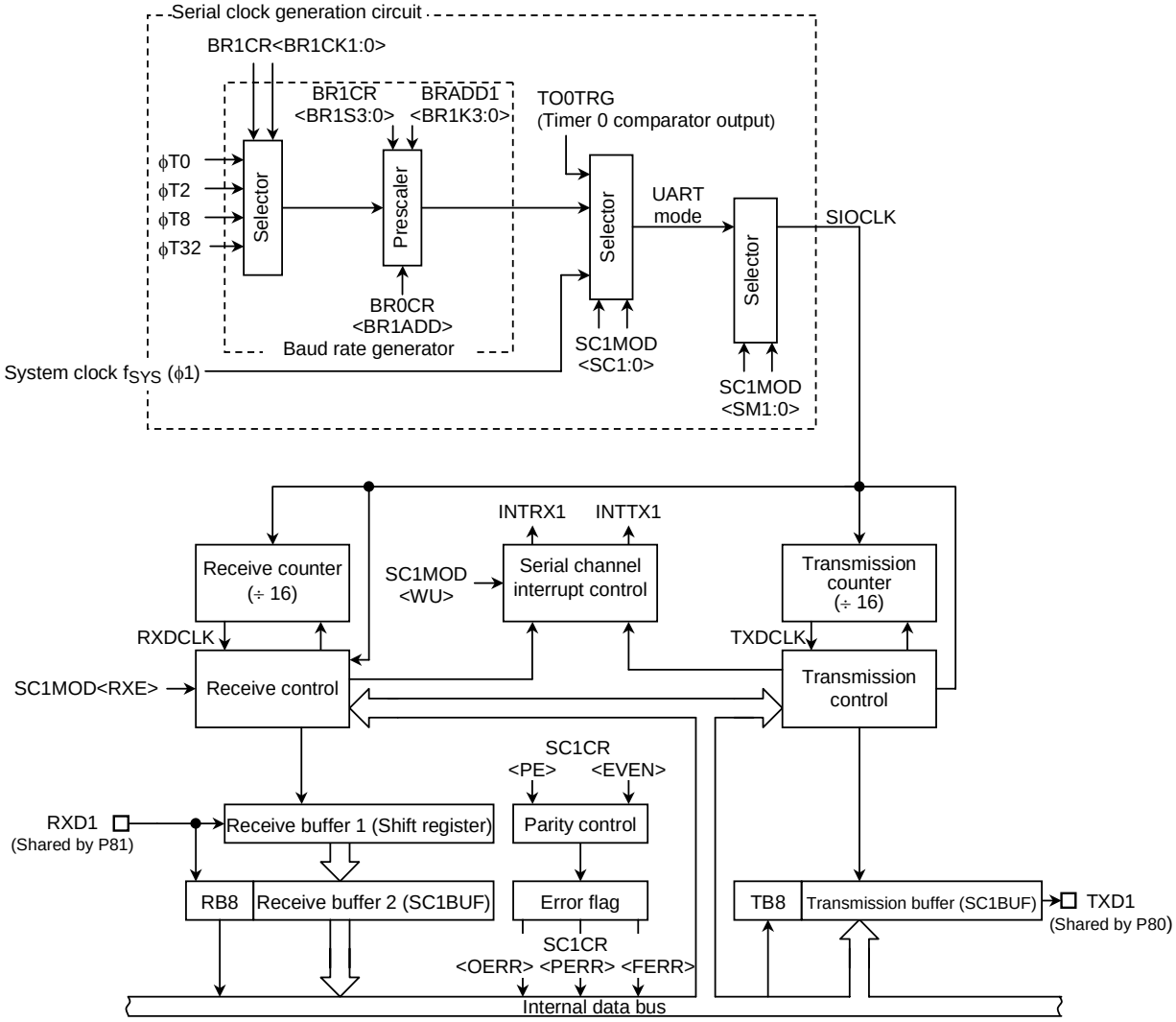


Figure 3.9.14 Block Diagram of the Serial Channel 0

Figure 3.9.15 shows the block diagram of the serial channel 1.



Note 1: No I/O interface mode in Channel 1.

Note 2: No handshake function in Channel 1.

Figure 3.9.15 Block Diagram of the Serial Channel 1

(1) Prescaler, prescaler clock select

There are 9-bit prescaler and prescaler clock selection registers to generate input clock for 8-bit timers 0 to 3, 16-bit timer/event counters 4, 6, 8, and A and serial interfaces 0, 1.

Figure 3.9.16 shows the block diagram. Table 3.9.1 shows prescaler clock resolution into the baud rate generator.

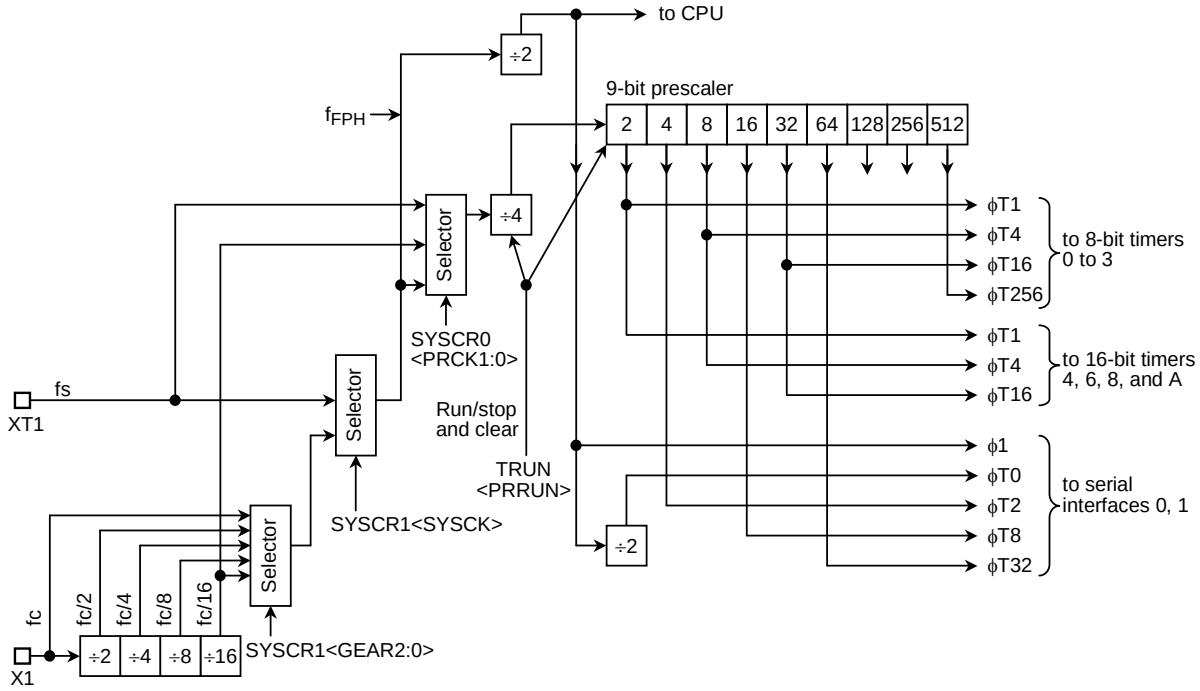


Figure 3.9.16 Block Diagram of Prescaler

Table 3.9.1 Prescaler Clock Resolution to Baud Rate Generator

at fc = 20 MHz, fs = 32.768 KHz

Select System Clock <SYSCK>	Select Prescaler Clock <PRCK1:0>	Gear Value <GEAR2:0>	Prescaler Output Clock Resolution			
			φT0	φT2	φT8	φT32
1 (fs)	00 (f_{FPH})	XXX	$fs/2^2$ (122 μs)	$fs/2^4$ (488 μs)	$fs/2^6$ (1.95 ms)	$fs/2^8$ (7.81 ms)
0 (fc)		000 (fc)	$fc/2^2$ (0.2 μs)	$fc/2^4$ (0.8 μs)	$fc/2^6$ (3.2 μs)	$fc/2^8$ (12.8 μs)
		001 (fc/2)	$fc/2^3$ (0.4 μs)	$fc/2^5$ (1.6 μs)	$fc/2^7$ (6.4 μs)	$fc/2^9$ (25.6 μs)
		010 (fc/4)	$fc/2^4$ (0.8 μs)	$fc/2^6$ (3.2 μs)	$fc/2^8$ (12.8 μs)	$fc/2^{10}$ (51.2 μs)
		011 (fc/8)	$fc/2^5$ (1.6 μs)	$fc/2^7$ (6.4 μs)	$fc/2^9$ (25.6 μs)	$fc/2^{11}$ (102.4 μs)
		100 (fc/16)	$fc/2^6$ (3.2 μs)	$fc/2^8$ (12.8 μs)	$fc/2^{10}$ (51.2 μs)	$fc/2^{12}$ (204.8 μs)
XXX	01 (Low-frequency clock)	XXX	–	$fs/2^4$ (488 μs)	$fs/2^6$ (1.95 ms)	$fs/2^8$ (7.81 ms)
XXX	10 (fc/16 clock)	XXX	–	$fc/2^8$ (12.8 μs)	$fc/2^{10}$ (51.2 μs)	$fc/2^{12}$ (204.8 μs)

XXX: Don't care, –: Can not use

Note: The fc/16 clock cannot be used as a prescaler clock when the fs is used as a system clock.

The clock selected among f_{FPH} clock, fc/16 clock, and fs clock is divided by 4 and input to this prescaler. This is selected by prescaler clock selection register SYSCR0<PRCK1:0>.

Resetting sets <PRCK1:0> to 00 and selects the f_{FPH} clock input divided by 4.

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

The prescaler can be run or stopped by the timer operation control register TRUN<PRRUN>. Counting starts when <PRRUN> is set to 1. The prescaler is cleared to 0 and stops operation when <PRRUN> is set to 0.

When the IDLE1 mode (Only the oscillator operates) is used, set TRUN<PRRUN> to 0 to stop this prescaler before the HALT instruction is executed.

(2) Baud rate generator

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

The input clock to the baud rate generator, ϕT0, ϕT2, ϕT8, or ϕT32, is generated by the 9-bit prescaler which is shared by the timers. One of these input clocks is selected by the baud rate generator control register BR0CR<BR0CK1:0>.

The baud rate generator includes a frequency divider, which divides frequency by 1, n + m/16 (n = 2 to 15, m = 0 to 15) to 16 values to determine the transfer rate.

The transfer rate is determined by setting BR0CR<BR0ADD, BR0S3:0> and BRADD0<BR0K3:0>.

- UART mode

  (1) BR0CR<BR0ADD> = 0

  Setting BRADD0<BR0K3:0> is ignored. The baud rate generator divides the selected prescaler clock by N which is set to BR0CK<BR0S3:0> (N = 1, 2, 3...16).

  (2) BR0CR<BR0ADD> = 1

  N + (16 − K)/16 division function is enabled. The baud rate generator divides the selected prescaler clock by N + (16 − K)/16 according to N set to BR0CR<BR0S3:0> (N = 2, 3...15) and K set to BRADD0<BR0K3:0> (K = 1, 2, 3...15).

  Note: At N = 1 or 16, N + (16 − K)/16 division function is disabled. Set BR0CR<BR0ADD> to 0.

- I/O interface mode

  N + (16 − K)/16 division function is not available in I/O interface mode. Set BK0CR<BR0ADD> to 0 before dividing by N.

How to calculate a transfer rate when the baud rate generator is used is explained below.

- UART mode

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

- I/O interface mode

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

- Integer divisor (N divisor)

  For example, when the source clock (fc) is 12.288 MHz, the input clock is ϕT2 (fc/16), and frequency divisor is N(BR0CR<BR0S3:0>) = 5 BR0CR<BRADD0> = 0, the baud rate in UART mode becomes as follows:

  * Clock condition
    - System clock: High frequency (fc)
    - Clock gear: 1 (fc)
    - Prescaler clock: System clock

$$\text{Baud rate} = \frac{fc/16}{5} \div 16$$
$$= 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600 \text{ (bps)}$$

  Note: + (16 − K)/16 division function is disabled, and setting BRADD0<BR0K3:0> is invalid.

- N + (16 − K)/16 divisor (only in UART mode)

  Accordingly, when source clock (fc) is 4.8 MHz, the input clock is ϕT0, and frequency divisor is N (BR0CR<BR0S3:0>) = 7, "K" (BRADD0<BR0K3:0>) = 3, BR0CR<BRADD0> = 1, the baud rate in UART mode becomes as follows.

  * Clock condition
    - System clock: High frequency (fc)
    - Clock gear: 1 (fc)
    - Prescaler clock: System clock

$$\text{Baud rate} = \frac{fc/4}{7 + (16 - 3)/16} \div 16$$
$$= 4.8 \times 10^6 \div 4 \div (7 + 13/16) \div 16 = 9600 \text{ (bps)}$$

  Table 3.9.2, Table 3.9.3 show an example of the transfer rate in UART mode.

  Additionally, the external clock input is available in the serial clock. (Serial channel 0 to 3.) How to calculate a baud rate is explained below.

  - UART mode

    Baud rate = External clock input ÷ 16

    It is necessary to satisfy (External clock input cycle) ≥ 4/fc + 20 [ns]

  - I/O interface mode

    Baud rate = External clock input

    It is necessary to satisfy (External clock input cycle) ≥ 16/fc [ns].

Table 3.9.2 Selection of Transfer Rate (when baud rate generator is used)

Unit (Kbps)

fc [MHz]	Input Clock / Frequency Divisor	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	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 8 times faster than the values given in the above table.

Note 2: This table is calculated when fc is selected as a system clock, the clock gear is set for fc, and the system clock as the prescaler clock input.

Table 3.9.3 Selection of Transfer Rate (when timer 0 with input clock φT1 is used)

Unit (Kbps)

TREG0 \ fc	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

How to calculate the transfer rate (when timer 0 is used):

$$\text{Transfer rate} = \frac{\text{The clock frequency selected by the register SYSCR0<PRCK1:0>}}{\text{TREG0} \times \underline{8} \times 16}$$

(when Timer 0 (input clock φT1) is used)

Note 1: Timer 0 match detect signal cannot be used as the transfer clock in I/O interface mode.

Note 2: This table is calculated when fc is selected as a system clock, the clock gear is set for fc, and system clock as the prescaler clock input.

(3) Serial clock generation circuit

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

- I/O interface mode

  When in SCLK output mode with the setting of SC0CR<IOC> = 0, the basic clock will be generated by dividing the output of the baud rate generator by 2 as described before. When in SCLK input mode with the setting of 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.

- UART mode

  The setting of SC0MOD<SC1:0> will select between the baud rate generator clock, internal clock ϕ1 (Max 625 kbps at fc = 20 MHz), or the match detect signal from timer 0 or the external clock (Channel 0) to generate the basic clock SIOCLK.

(4) Receiving counter

The receiving counter is a 4-bit binary counter used in UART mode and counts up according to the SIOCLK clock. 16 pulses of SIOCLK are used for receiving 1 bit of data, and the data bit is sampled three times at the 7th, 8th, and 9th clock.

With these three samples, the received data bit is evaluated by the majority rule.

For example, if the sampled data bit is 1, 0, and 1 at 7th, 8th, and 9th clock respectively, the received data is evaluated as 1. The sampled data 0, 0, and 1 is evaluated such that the received data bit is determined to be 0.

(5) Receiving control

- I/O interface mode

  When in SCLK output mode with the setting of SC0CR<IOC> = 0, the RXD0 signal will be sampled at the rising edge of the shift clock which is output to the SCLK0 pin.

  When in SCLK input mode with the setting SC0CR<IOC> = 1, the RXD0 signal will be sampled at the rising edge or falling edge of the SCLK0 input according to the setting of the SC0CR<SCLKS> register.

- UART mode

  The receiving control block has a circuit for detecting the start bit by the rule of majority. When two or more 0 are detected during the 3 samples, it is recognized as start bit and the receiving operation is started.

  The data being received is also evaluated by the majority rule.

(6) Receiving buffer

To prevent an overrun error, the receiving buffer has a double buffer structure.

Received data is stored bit by bit in receiving buffer 1 (Shift register type). When 7 bits or 8 bits of data are stored in receiving buffer 1, the stored data is transferred to receiving buffer 2 (SC0BUF) generating an interrupt INTRX0. The CPU reads only 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 the receiving buffer 2 and SC0CR<RB8> are still preserved.

The parity bit added in 8-bit UART mode and the most significant bit (MSB) in 9-bit UART mode are stored in SC0CR<RB8>.

When in 9-bit UART mode, the wake-up function of the slave controller is enabled by setting SC0MOD<WU> to 1, and interrupt INTRX0 occurs only when SC0CR<RB8> is set to 1.

(7) Transmission counter

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

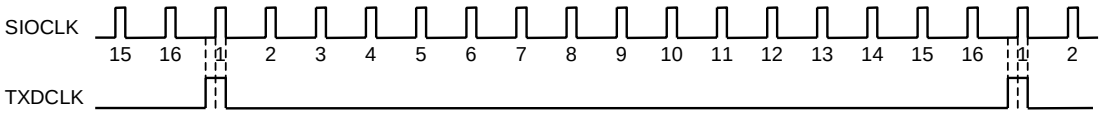


Figure 3.9.17 Generation of Transmission Clock

(8) Transmission controller

- I/O interface mode

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

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

- UART mode

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

Handshake function

Serial channel 0 has a $\overline{CTS}$ pin. Using this pin, data can be sent in units of one frame, thus, overrun errors can be avoided. The handshake function is enabled/disabled by SC0MOD<CTSE>.

When the $\overline{CTS0}$ pin goes high, after completion of the current data send, data send is halted until the $\overline{CTS0}$ pin goes low again. However the INTTX0 interrupt is generated, it requests the next data send to the CPU. The next data is written in the transmission buffer and data send is halted.

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

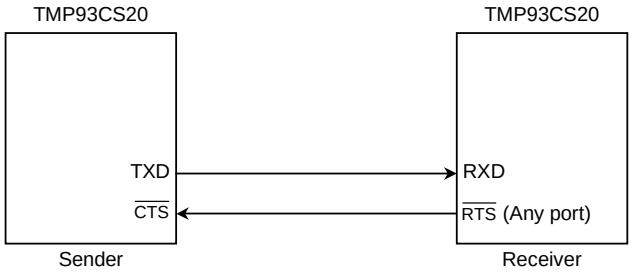
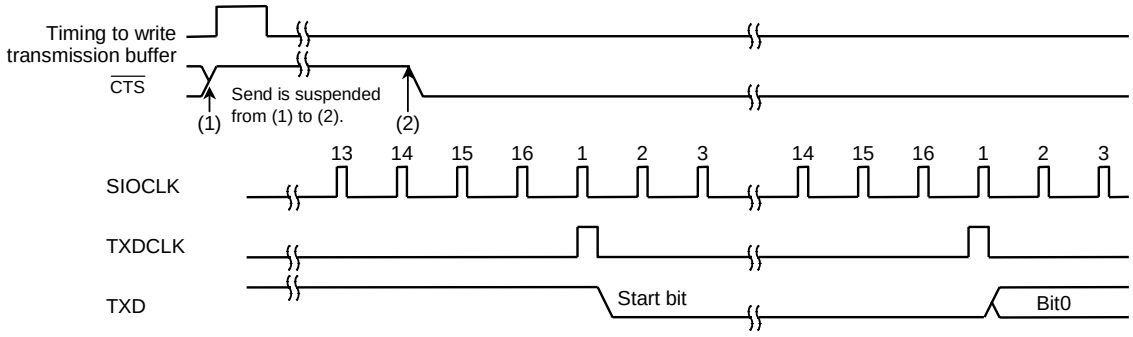


Figure 3.9.18 Handshake Function



Note 1: If the $\overline{CTS}$ signal rises during transmission, the next data is not sent after the completion of the current transmission.

Note 2: Transmission starts at the first TXDCLK clock falling edge after the $\overline{CTS}$ signal falls.

Figure 3.9.19 Timing of $\overline{CTS}$ (Clear to send)

(9) Transmission buffer

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

(10) Parity control circuit

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

For transmission, parity is automatically generated according to the data written in the transmission buffer SC0BUF. The data is transmitted after the parity bit is stored in SC0BUF<TB7> when in 7-bit UART mode or in SC0MOD<TB8> when in 8-bit UART mode. <PE> and <EVEN> must be set before the transmission data is written to the transmission buffer.

For receiving, data are shifted in the receiving buffer 1, the parity is added after the data is transferred to receiving buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> when in 7-bit UART mode and with SC0MOD<RB8> when in 8-bit UART mode. If they are not equal, a parity error occurs and SC0CR<PERR> flag is set.

(11) Error flag

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

1. Overrun error <OERR>

   If all bits of the next data are received in receiving buffer 1 while valid data is stored in receiving buffer 2 (SC0BUF), an overrun error will occur.

2. Parity error <PERR>

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

3. Framing error <FERR>

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

(12) Generating timing

1) UART mode

Receiving

Mode	9 Bits (Note)	8 Bits + Parity (Note)	8 Bits, 7 Bits + parity, 7 Bits
Interrupt timing	Center of last bit (Bit8)	Center of last bit (Parity bit)	Center of stop bit
Framing error timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error timing	−	Center of last bit (Parity bit)	Center of stop bit
Overrun error timing	Center of last bit (Bit8)	Center of last bit (Parity bit)	Center of stop bit

Note: In 9-bit and 8-bit + party 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.	←	←

2) I/O interface

Transmission interrupt timing	SCLK output mode	Immediately after rise of last SCLK signal. (See figure 3.9.22)
	SCLK input mode	Immediately after rise of last SCLK signal (Rising mode), or immediately after fall in falling mode. (See figure 3.9.23)
Receiving interrupt timing	SCLK output mode	Timing used to transfer received data to data receive buffer 2 (SC0BUF) (that is, immediately after last SCLK). (See figure 3.9.24)
	SCLK input mode	Timing used to transfer received data to data receive buffer 2 (SC0BUF) (that is, immediately after last SCLK). (See figure 3.9.25)

3.9.3 Operational Description

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

This mode is used to increase the number I/O pins for transmitting or receiving data to or from an external shifter register.

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

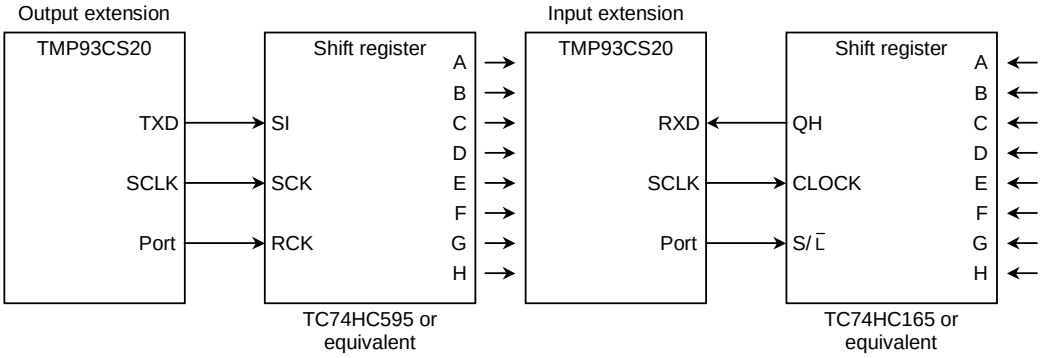


Figure 3.9.20 Example of SCLK Output Mode Connection

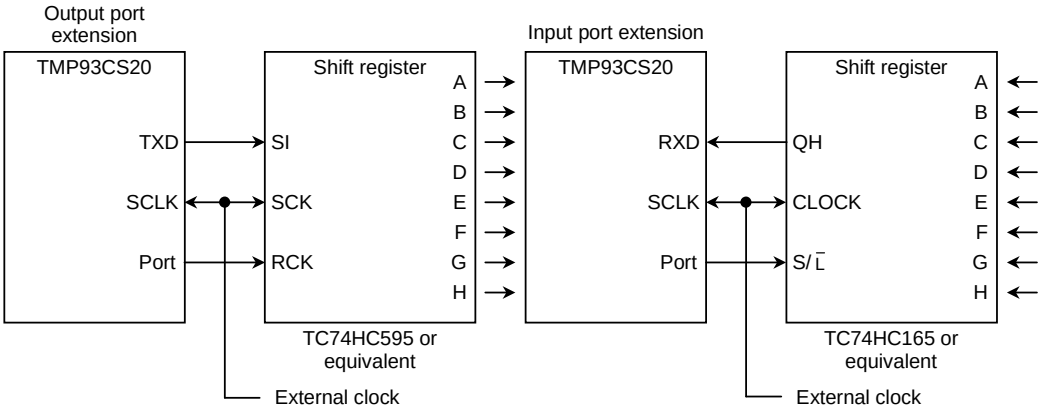


Figure 3.9.21 Example of SCLK Input Mode Connection

Note: Serial channel 1 has no I/O interface mode.

a. Transmission

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

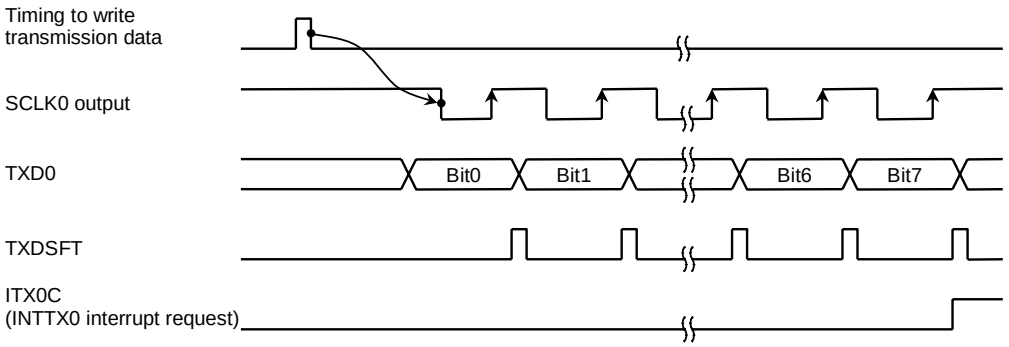


Figure 3.9.22 Transmitting Operation in I/O Interface Mode (SCLK0 output mode) (Channel 0)

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

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

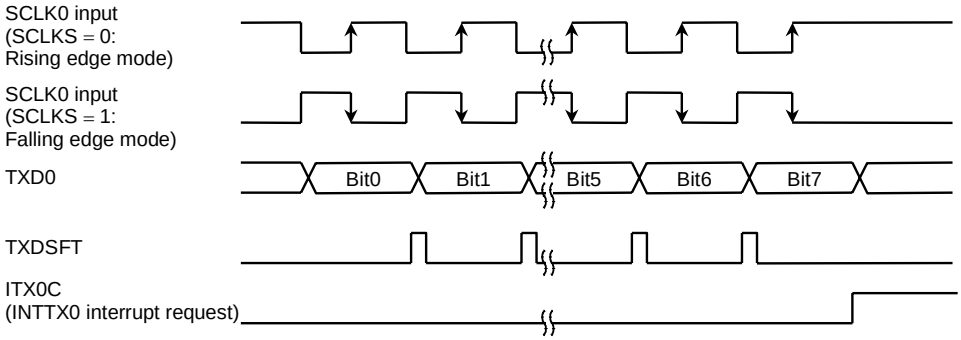


Figure 3.9.23 Transmitting Operation in I/O Interface Mode (SCLK0 input mode) (Channel 0)

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 again to generate INTRX0 interrupt.

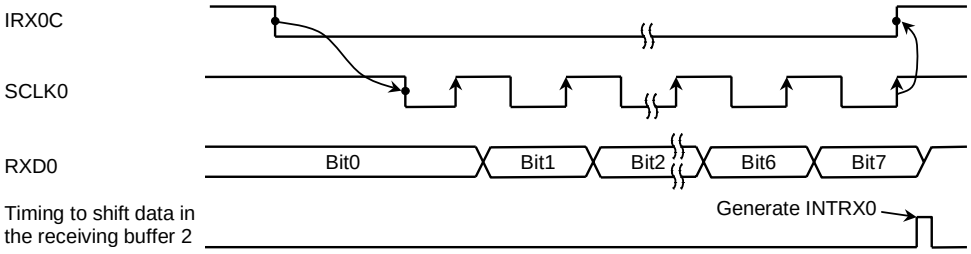


Figure 3.9.24 Receiving Operation in I/O Interface Mode (SCLK0 output mode) (Channel 0)

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 generate INTRX0 interrupt.

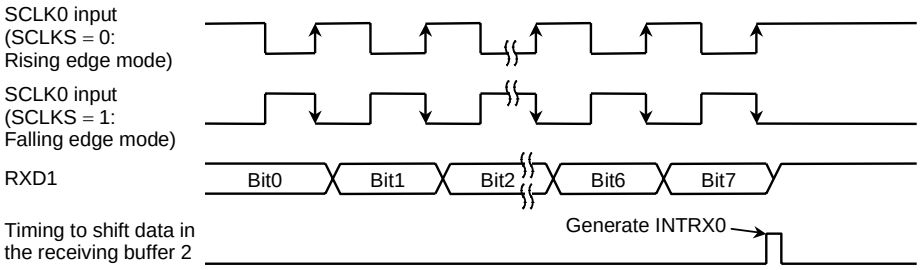


Figure 3.9.25 Receiving Operation in I/O Interface Mode (SCLK0 input mode) (Channel 0)

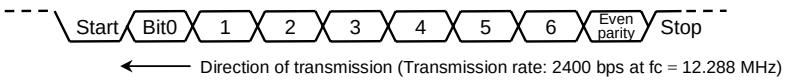
Note: For data receiving, the system must be placed in the receive enable state (SC0MOD<RXE> = 1)

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

The 7-bit mode can be set by setting serial channel mode register SC0MOD<SM1:0> to 01.

In this mode, a parity bit can be added, and the addition of the parity bit can be enabled or disabled by serial channel control register SC0CR<PE>, and even parity or odd parity is selected by SC0CR<EVEN> when <PE> is set to 1 (Enable).

Setting example: When transmitting data with the following format, the control registers should be set as described below. Channel 0 is explained here.



* Clock condition

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

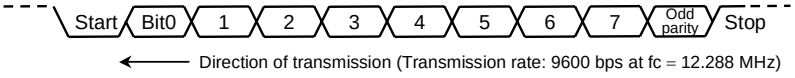
	7	6	5	4	3	2	1	0	
P6CR	–	–	–	–	1	–	–	–	Select P63 as the TXD0 pin.
P6FC	–	–	–	X	1	–	–	–	
SC0MOD	X	0	–	X	0	1	0	1	Set 7-bit UART mode.
SC0CR	X	1	1	X	X	X	0	0	Add even parity.
BR0CR	0	0	1	0	0	1	0	1	Set transfer rate at 2400 bps.
TRUN	1	X	–	–	–	–	–	–	Start the prescaler for the baud rate generator.
INTES0	1	1	0	0	–	–	–	–	Enable INTTX0 interrupt and set interrupt level 4.
SC0BUF	*	*	*	*	*	*	*	*	Set data for transmission.

X: Don't care, –: No change

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

The 8-bit UART mode can be specified by setting SC0MOD<SM1:0> to 10. In this mode, the parity bit can be added (the addition of a parity bit is enabled or disabled by SC0CR<PE>) and even parity or odd parity is selected by SC0CR<EVEN> when <PE> is set to 1 (Enable).

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



* Clock condition

System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

Main setting

		7	6	5	4	3	2	1	0	
P6CR	←	–	–	–	0	–	–	–	–	Select P64 (RXD0) as the input pin.
SC0MOD	←	–	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 transfer rate at 9600 bps.
TRUN	←	1	X	–	–	–	–	–	–	Start the prescaler for the baud rate generator.
INTES0	←	–	–	–	–	1	1	0	0	Enable INTRX0 interrupt and set interrupt level 4.

Interrupt processing

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

X: Don't care, –: No change

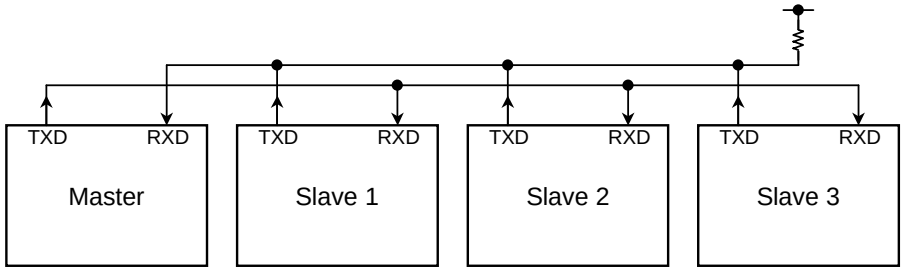
(4) Mode 3 (9-bit UART mode)

9-bit UART mode can be specified by setting SC0MOD<SM1:0> to 11. In this mode, parity bit cannot be added.

For transmission, the MSB (9th bit) is written in SC0MOD<TB8>. For receiving it is stored in SC0CR<RB8>. For writing and reading of the buffer, the MSB is read or written first then the rest of the data from SC0BUF.

Wakeup function

In 9-bit UART mode, the wakeup function of slave controllers is enabled by setting SC0MOD<WU> to 1. The interrupt INTRX0 occurs only when<RB8> = 1.

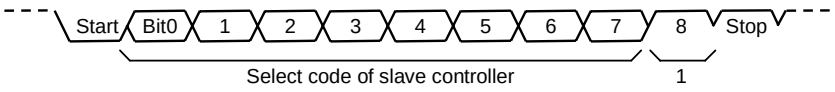


Note: TXD pin of the slave controllers must be in open-drain output mode.

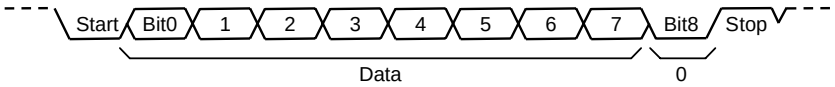
Figure 3.9.26 Serial Link Using Wakeup Function

Protocol

1. Select the 9-bit UART mode for the master and slave controllers.
2. Set SC0MOD<WU> bit of each slave controller to 1 to enable data receiving.
3. 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.

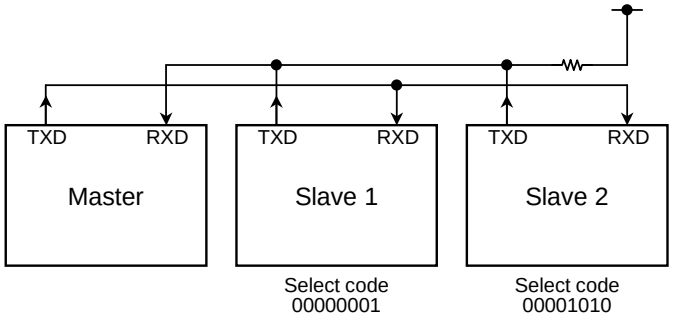


4. Each slave controller receives the above frame, and clears WU bit to 0 if the above select code matches its own select code.
5. The master controller transmits data to the specified slave controller whose SC0MOD<WU> bit is cleared to 0. The MSB (Bit8)<TB8> is cleared to 0.



6. The other slave controllers (with the <WU> bit remaining at 1) ignore the receiving data because their MSBs (Bit8 or <RB8>) are set to 0 to disable the interrupt INTRX0.
   The slave controllers (WU = 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.

Setting example: To link two slave controllers serially with the master controller, and use the internal clock ϕ1 as the transfer clock.



Since serial channels 0 and 1 operate in exactly the same way, channel 0 is used for the purposes of explanation.

- Setting the master controller

```
Main
P6CR     ← – – – 0 1 – – –     } Select P63 as TXD0 pin and P64 as RXD0 pin.
P6FC     ← – – – X 1 – – –     }
INTES0   ← 1 1 0 0 1 1 0 1     Enable INTTX0 and set the interrupt level 4.
                               Enable INTRX0 and set the interrupt level 5.
SC0MOD   ← 1 0 1 0 1 1 1 0     Set φ1 as the transmission clock in 9-bit UART mode.
SC0BUF   ← 0 0 0 0 0 0 0 1     Set the select code for slave controller 1.

INTTX0 interrupt
SC0MOD   ← 0 – – – – – – –     Sets TB8 to 0.
SC0BUF   ← * * * * * * * *     Set data for transmission.
```

- Setting the slave controller

```
Main
P6CR     ← – – – 0 1 – – –     } Select P64 as RXD0 pin and P63 as TXD0 pin (Open-drain
P6FC     ← – – – X 1 – – –     }  output).
P6ODE    ← X X X 1 – – – X     }
INTES0   ← 1 1 0 1 1 1 1 0     Enable INTRX0 and INTTX0.
SC0MOD   ← 0 0 1 1 1 1 1 0     Set <WU> to 1 in the 9-bit UART transmission mode with
                               transfer clock φ1.

INTRX0 interrupt
Acc      ← SC0BUF
if Acc = Select code
Then SC0MOD ← – – – 0 – – – –  Clear <WU> to 0.
```

3.10 Serial Bus Interface (SBI)

The TMP93CS20 has a 1-channel serial bus interface which employs a clocked-synchronous 8-bit serial bus interface and an I²C bus.

The serial bus interface is connected to an external device through P61 (SDA) and P62 (SCL) in the I²C bus mode; and through P60 (SCK), P61 (SO), and P62 (SI) in the clocked-synchronous 8-bit SIO mode.

TMP93CS20 has no an arbitration function which is necessary when two or more master devices scramble for the bus control. In master mode, other devices which are connected on the same bus need be slave devices (Single master).

Setting of every pins are as follows.

	ODE<ODE62:61>	P6CR<P62C, P61C, P60C>	P6FC<P62F, P61F, P60F>
I^2C bus mode	11	11X	110
Clock synchronous 8-bit SIO mode	XX	011 010	111

X: Don't care

3.10.1 Configuration

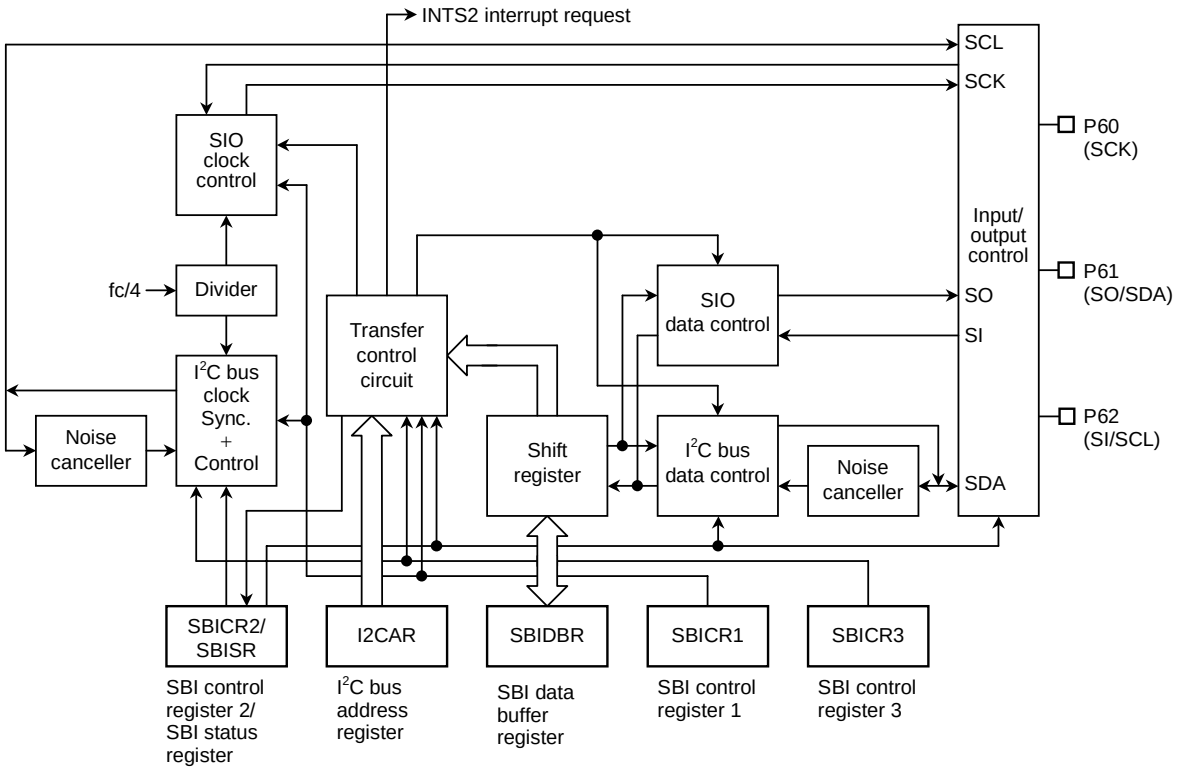


Figure 3.10.1 Serial Bus Interface (SBI)

3.10.2 Serial Bus Interface (SBI) Control

The following reginsters are used for control and operation status monitoring when using the serial bus interface (SBI).

- Serial bus interface control register 1 (SBICR1)
- Serial bus interface control register 2 (SBICR2)
- Serial bus interface control register 3 (SBICR3)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register (SBISR)

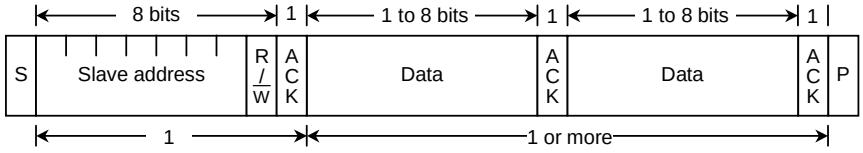
The above registers differ depending on a mode to be used.

Refer to section 3.10.4 “I²C Bus Mode Control” and 3.10.6 “Clocked-synchronous 8-Bit SIO Mode Control”.

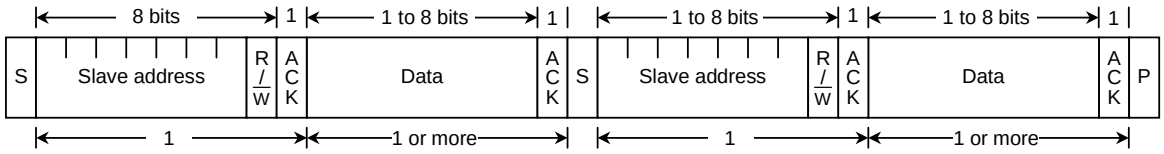
3.10.3 The Data Formats in the I²C Bus Mode

The data formats when using the TMP93CS20 in the I²C bus mode are shown below.

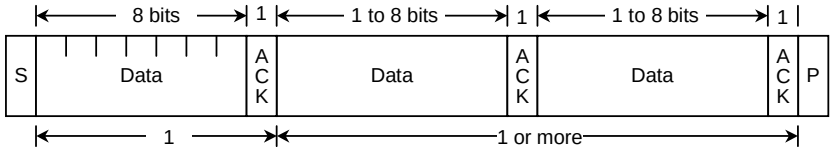
(a) Addressing format



(b) Addressing format (with restart)



(c) Free data format



S: Start condition

R/$\overline{W}$: Direction bit

ACK: Acknowledge bit

P: Stop condition

Figure 3.10.2 Data Format in the I²C Bus Mode

3.10.4 I²C Bus Mode Control

The following registers are used for control and operation status monitoring 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
SBICR1 (004BH)	Bit symbol	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0
	Read/Write	W			R/W		W		
	After reset	0	0	0	0		0	0	0
Prohibit read-modify-write	Function	Number of transferred bits (Note 1)			Acknowledge mode specification		Serial clock selection (Note 2)		

Serial clock selection

000	N = 4	– kHz	System clock: fc Clock gear: fc/1 fc = 20 MHz (Output on SCL pin)
001	N = 5	– kHz	
010	N = 6	– kHz	
011	N = 7	74.6 kHz	
100	N = 8	38.2 kHz	
101	N = 9	19.3 kHz	
110	N = 10	9.71 kHz	
111		(Reserved)	

Acknowledge mode specification

0	Not generate clock pulse for acknowledge signal
1	Generate clock pulse for acknowledge signal

Number of transferred bits

<BC2:0>	<ACK> = 0		<ACK> = 1	
	Number of Clock	Bits	Number of Clock	Bits
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 <BC2:0> to 000 before switching to a clock-synchronous 8-bit SIO mode.

Note 2: Refer to the sentence of 3.10.4 (3) “Serial clock”.

Note 3: This I²C bus circuit does not support high-speed mode. It supports standard mode only.

Figure 3.10.3 Register for I²C Bus Mode (1/4)

Serial Bus Interface Control Register 2

SBICR2 (004EH)

Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0		
Read/Write	W (R/W Note 1)				W			
After reset	0	0	0	1	0	0		
Function	Master/ slave selection	Transmitter/ receiver selection	Start/stop generation	Cancel INTS2 request	Serial bus interface operating mode selection (Note 2)			

SBI operating mode selection

00	Port mode (Serial bus interface output disable)
01	SIO mode
10	I^2C bus mode
11	(Reserved)

Cancel INTS2 request

0	−
1	Cancel interrupt service request

Stare/stop generation

0	Generate the stop condition
1	Generate the start condition

Transmitter/receiver selection

0	Receiver
1	Transmitter

Master/slave selection

0	Slave
1	Master

Note 1: This register functions as the SBISR by reading.

Note 2: Switch a mode to the port mode after confirming that the bus is free.
Switch a mode to the I^2C bus mode and the clocked-synchronous 8-bit SIO mode after confirming that input signals via port are high level.

Figure 3.10.4 Register for I^2C Bus Mode (2/4)

Serial Bus Interface Status Register

SBISR (004EH)

Prohibit read-modify-write

	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 selection status monitor	Transmitter /receiver selection status monitor	I^2C bus status monitor	INTS2 request status monitor	Noise detection monitor	Slave address match detection monitor	"GENERAL CALL" detection monitor	Last received bit monitor

Last received bit monitor

0	Last received bit 0
1	Last received bit 1

"GENERAL CALL" detection monitor

0	Non "GENERAL CALL" detected
1	"GENERAL CALL" detected

Slave address match detection monitor

0	Non slave address match or "GENERAL CALL" detected
1	Slave address match or "GENERAL CALL" detected

Noise detection monitor

0	Non noise detected
1	Noise detected

INTS2 request status

0	Interrupt service requested
1	Interrupt service canceled

I^2C bus status monitor

0	Bus free
1	Bus busy

Transmitter/receiver selection status monitor

0	Receiver
1	Transmitter

Master/slave selection status monitor

0	Slave
1	Master

Note: Bits 7 to 2 of this register function as the SBICR2 by writing.

Figure 3.10.5 Register for I^2C Bus Mode (3/4)

Serial Bus Interface Control Register 3

SBICR3 (004FH)

	7	6	5	4	3	2	1	0
Bit symbol								SWRTS
Read/Write								R/W
After reset								0
Function								Software reset 0: Don't care 1: Initialize SBI

Software reset

0	Don't care
1	Initialize SBI (after initializing SBI, SWRST is automatically cleared to 0).

Serial Bus Interface Data Buffer Register

SBIDBR (004CH)
Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Send)							
After reset	Undefined							

Note 1: When writing the send data, start from the MSB (Bit7). Redeiving data is placed from LSB (Bit0).

Note 2: SBIDBR can't be read the written data. Therefore read-modify-write instruction (e.g., "BIT" instruction) is prohibitted.

Note 3: Written data in SBIDBR is cleared by INTS2 signal.

I^2C Bus Address Register

I2CAR (004DH)
Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	W							
Read/Write	0	0	0	0	0	0	0	0
Function	Slave address selection.							Address recognition mode specification

Address recognition mode specification

0	Slave address recognition
1	Non slave address recognition

Figure 3.10.6 Registers for I^2C Bus Mode (4/4)

(1) Acknowledge mode specification

Set SBICR1<ACK> to 1 for operation in the acknowledge mode. The TMP93CS20 generates an additional clock pulse for an acknowledge signal when operating in the 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 level in order to generate the acknowledge signal.

Set <ACK> to 0 for operation in the non-acknowledge mode. The TMP93CS20 does not generate a clock pulse for the acknowledge signal when operating in the master mode.

In the acknowledgment mode, when the TMP93CS20 is the slave mode, clocks are counted for the acknowledge signal. During the clock for the acknowledge signal, when a received slave address matches to a slave address set to the I2CAR or a "GENERAL CALL" is received, the SDA pin is set to low level generating an acknowledge signal.

After a received slave address matches to a slave address set to the I2CAR and a "GENERAL CALL" is received, in the transmitter mode during the clock for the acknowledge signal, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode, the SDA pin is set to low level generating an acknowledge signal.

In the non-acknowledgment mode, when the TMP93CS20 is the slave mode, clocks for the acknowledge signal are not counter.

(2) Number of transfer bits

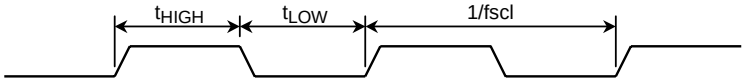
SBICR1<BC2:0> are used to select a number of bits for transmitting and receiving data.

Since <BC2:0> are cleared to 000 as a start condition, a slave address and direction bit transmissions are executed in 8 bits. Other than these <BC2:0> retain a specified value.

(3) Serial clock

a. Clock source

SBICR1<SCK2:0> are used to select a maximum transfer frequency output on the SCL pin in the master mode. Set the baud rates, which have been calculated according to the formula below, to meet the specifications of the I²C bus, such as the smallest pulse width of t_{LOW}.



$t_{LOW} = 2^n/f_{FPH}$
$t_{HIGH} = 2^n/f_{FPH} + 12/f_{FPH}$
$fscl = 1/(t_{LOW} + t_{HIGH})$
$= \frac{f_{FPH}}{2 \times 2^n + 12}$

<SCK2:0> (Bits 2 to 0 in the SBICR1)	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10

Figure 3.10.7 Clock Source

b. Clock synchronization

The I²C bus has a clock synchronization function to meet the transfer speed to a slow processing device when a transfer is performed between devices which have different process speed.

The clock synchronization functions when the SCL pin is high level and the SCL line of the bus is low level in the serial bus interface circuit. The serial bus interface circuit waits counting a clock pulse in high level until the SCL line of the bus is high level. When the SCL line of the bus is high level, the serial bus interface circuit starts counting during high level. The clock synchronization function holds clocks which are output from the serial interface circuit to be high level.

The slave device can stop the clock output of the master device on one word or one bit basis.

Additionally, the transfer speed by the master device matches to the process speed of the slave device.

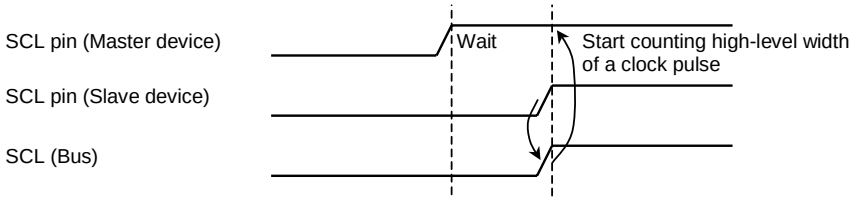


Figure 3.10.8 Clock Synchronization

(4) Slave address and address recognition mode specification

To operate the TMP93CS20 in the addressing format which recognizes the slave address, set I2CAR<ALS> to 0 and set the slave address to the I2CAR<SA6:0>.

To operate the serial bus interface circuit in the free data format which does not recognize the slave address, set <ALS> to 1. When the TMP93CS20 used in the free data format, the slave address and the direction bit are not recognized. They are handled as data just after generation of start conditions.

(5) Master/slave selection

Set SBICR2<MST> to 1 for operating the TMP93CS20 as a master device. <MST> is cleared to 0 by the hardware after a stop condition on a bus is detected or arbitration is lost.

(6) Transmitter/receiver selection

Set SBICR2<TRX> to 1 for operating the TMP93CS20 as a transmitter. Set <TRX> to 0 for operation as a receiver. When data with an addressing format is transferred in the slave mode, when a slave address with the same value that an I2CAR or the GENERAL CALL is received (All 8-bit data are 0 after the start condition), <TRX> is set to 1 by the hardware if the direction bit (R/W) sent from the master device is 1, and is set to 0 by the hardware if the bit is 0. In the master mode, after the acknowledge signal is returned from the slave device, <TRX> is set to 0 by the hardware if a transmitted direction bit is 1, and set to 1 by the hardware if it is 0. When the acknowledge signal is not returned, the current condition is maintained.

<TRX> is cleared to 0 by the hardware after the stop condition on the I²C bus is detected or arbitration is lost.

The following shows <TRX> change conditions in each mode and <TRX> after changing.

Mode	Direction Bit	Change Condition	<TRX> After Changing
Slave mode	0	A received slave address is the same as a value set to I2CAR.	0
	1		1
Master mode	0	ACK signal is returned.	1
	1		0

When the TMP93CS20 operates in the free data format, the slave address and the direction bit are not recognized. They are handled as data just after generating a start condition. The TRX was not changed by the hardware.

(7) Start/stop condition generation

When SBISR<BB> is 0, the start condition and slave address and direction bit are output by writing 1 to SBICR2<MST, TRX, BB, PIN>. It is necessary to set 1 to SBICR1<ACK> beforehand.

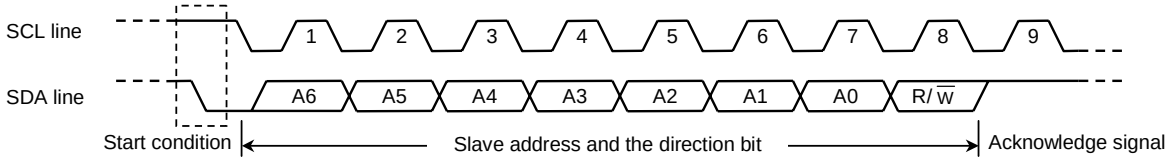


Figure 3.10.9 Start Condition Generation and Slave Address Generation

When SBISR<BB> is 1, a sequence of generating the stop condition is started by writing 1 to <MST, TRX, PIN> and 0 to <BB>. Do not modify the contents of <MST, TRX, BB, PIN> until the stop condition is generated on a bus.

When a stop condition is generated and the SCL line on the bus is set to low level by another device, a stop condition is generated after releasing the SCL line.

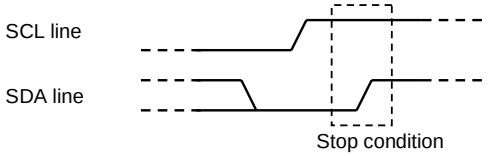


Figure 3.10.10 Stop Condition Generation

The bus condition can be indicated by reading the contents of <BB>. <BB> is set to 1 when the start condition on a bus is detected, and is set to 0 when the stop condition is detected.

(8) Cancel interrupt service request

When the TMP93CS20 is the master mode and transferring a number of clocks set by the SBICR1<BC2:0> and the SBICR1<ACK> is complete, a serial bus interface interrupt request (INTS2) is generated.

In the slave mode, the INTS2 is generated when the received slave address is the same as the value set to the I2CAR and an acknowledge signal is output, when a "GENERAL CALL" is received and an acknowledge signal is output, or when transferring/receiving data is complete after the received slave address is the same as the value set to the I2CAR and a "GENERAL CALL" is received.

When the serial bus interface interrupt request occurs, the SBISR<PIN> is cleared to 0. During the time that the PIN is 0, the SCL pin is set to low level.

Either writing or reading data to or from the SBIDBR sets the <PIN> to 1.

The time from the <PIN> being set to 1 until the SCL pin is released takes t_{LOW}.

Although the <PIN> can be set to 1 by the program, the <PIN> is not cleared to 0 when it is written 0.

(9) Serial bus interface operation mode selection

SBICR2<SBIM1:0> is used to specify the serial bus interface operation mode. Set <SBIM1:0> to 10 when used in the I^2C bus mode after confirming that input signal via port is high level.

Switch a mode to port after making sure that a bus is free.

(10) Noise detection monitor

The I^2C bus is easy to be affected by noise, because the bus is driven by the open drain and the pull-up resistor.

With the TMP93CS20, the SDA pin output and the SDA line level are compared at a rise of the SCL line on the bus, and whether data are output correctly on the bus is detected only in the master transmitter mode.

When the SDA pin output differs from the SDA line level, the SBISR<AL> is set to 1.

When the AL is set to 1, the SDA pin is released and the MST and the TRX are cleared to 0 by the hardware. The TMP93CS20 changes to the slave receiver mode, and continues outputting clocks unit transferring data when the AL was set to 1 is completed.

Either writing or reading data to or from the SBIDBR, or writing data to the SBICR2 clears to the AL to 0.

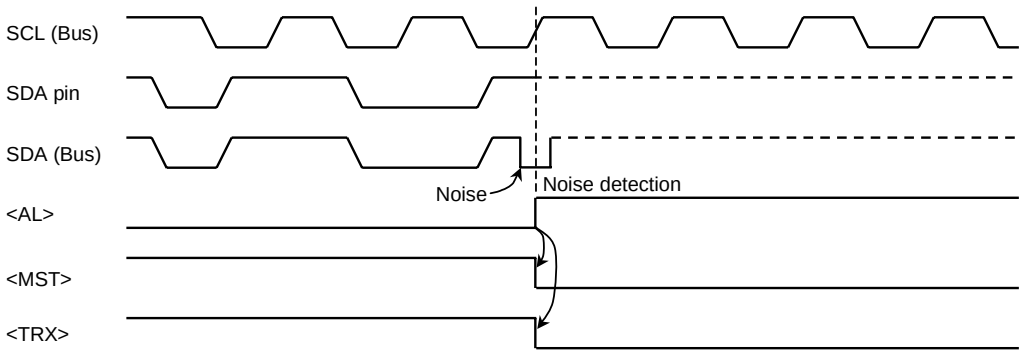


Figure 3.10.11 Noise Detection Monitor

(11) Slave address match detection monitor

SBISR<AAS> is set to 1 in the slave mode, in the address recognition mode (I2CAR<ALS> = 0) when receiving the GENERAL CALL or the slave address with the same value that is set to the I2CAR. When <ALS> is 1, <AAS> is set to 1 after receiving the first one-word of data. <AAS> is set to 0 by writing/reading data to/from a data buffer register.

(12) GENERAL CALL detection monitor

SBISR<AD0> is set to 1 in the slave mode, when the GENERAL CALL is received (All 8-bit data are 0 after the start condition). <AD0> is set to 0 when the start or stop condition is detected on a bus.

(13) Last received bit monitor

The SDA line value stored at the rising edge of the SCL line is sent to SBISR<LRB>. In the acknowledge mode, immediately after the INTS2 interrupt request is generated, the acknowledge signal is read by reading the contents of <LRB>.

(14) Software reset function

Software reset function is used to initialize the SBI which is rocked by external noise, etc. When SBICR3<SWRST> is set to 1, the internal reset signal pulse is generated and inputted into the SBI circuit.

All command registers and state registers are initialized to initial values. <SWRST> is automatically set to 0 after the SBI circuit is initialized.

(15) Serial bus interface data buffer register (SBIDBR)

The SBIDBR register can read out the receiving data and write the sending data.

After the start condition generated in the master mode, set the slave address and the direction bit in this register.

(16) I^2C bus address register (I2CAR)

I2CAR<SA6:0> sets the slave address when the TMP93CS20 are operated as the slave devices. Setting I2CAR<ALS> = 0, the slave address output from master device is recognized, and the data format is changed to the addressing format. Setting I2CAR<ALS> = 1, the slave address is not recognized, and the data format is changed to the free data format.

3.10.5 Data Transfer in I²C Bus Mode

(1) Device initialization

First, set SBICR1<ACK, SCK2:0>. Specify 0 to bits 7 to 5 and 3 in the SBICR1.

Set the slave address <SA6:0> and <ALS> to I2CAR (<ALS> = 0 when the addressing format).

Subsequently, set 0 to <MST, TRX, BB>; 1 to <PIN>; 10 to <SBIM1:0>; and 0 to bits 0 and 1 in the SBICR2. The slave receiver mode is set.

Note: The initialization of the serial bus interface circuit must be complete within the time from all devices which are connected to the bus have initialized to any device does not generate a start condition. If not, there is a possibility that another device starts transferring before an end of the initialization of the serial bus interface circuit. Data can not be received correctly.

(2) Start condition and slave address generation

Confirm a bus free status (when SIBSR<BB> = 0).

Set the SBICR1<ACK> to 1 and specify a slave address and a direction bit to be transmitted to the SBIDBR.

When the SBISR<BB> is 0, the start condition are generated and the slave address and the direction bit which are set to the SBIDBR are output on a bus by wiring 1 to the SBICR2<MST, TRX, BB> and PIN. An INTS2 interrupt request occurs at the 9th falling edge of the SCL clock cycle, and the <PIN> is cleared to 0. The SCL pin is pulled down to the low-level while the <PIN> is 0. When an interrupt request occurs, the <TRX> changes by the hardware according to the direction bit only when an acknowledge signal is returned from the slave device.

Note 1: Do not write a slave address to be output to the SBIDBR while data are transferred. If data is written to the SBIDBR, data to been outputting may be destroyed.

Note 2: Do not start transferring due to another master from writing a slave address to be output to the SBIDBR to writing a start condition generation command to the SBICR2. The serial bus interface circuit malfunctions because it has not an arbitration function.

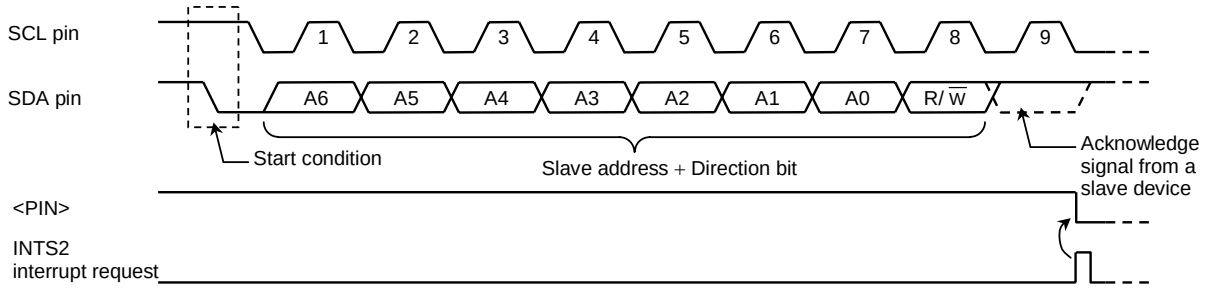


Figure 3.10.12 Start Condition Generation and Slave Address Transfer

(3) One-word data transfer

Test SBISR<MST> by the INTS2 interrupt process after a one-word data transfer is completed, and determine whether the mode is a master or slave.

a. When <MST> is 1 (Master mode)

Test SBISR<TRX> and determine whether the mode is a transmitter or receiver.

When <TRX> is 1 (Transmitter mode)

Check SBISR<LRB>. When <LRB> is 1, a receiver does not request data. Implement the process to generate the stop condition (described later) and terminate data transfer.

When <LRB> is 0, the receiver requests new data. When the next transmitted data is 8-bits, write it to the SBIDBR. When the next transmitted data is other than 8 bits, set SBICR1<BC2:0>, set SBICR1<ACK> to 1, and write the transmitted data to the SBIDBR. After writing the data, SBISR<PIN> becomes 1, the serial clock pulse is generated for transferring a new one-word of data from the SCL pin, and then the one-word data is transmitted. After the data is transmitted, the INTS2 interrupt request occurs. <PIN> becomes 0 and the SCL pin is set to the low level. If the data to be transferred is more than one word in length, repeat the procedures from <LRB> test mentioned above.

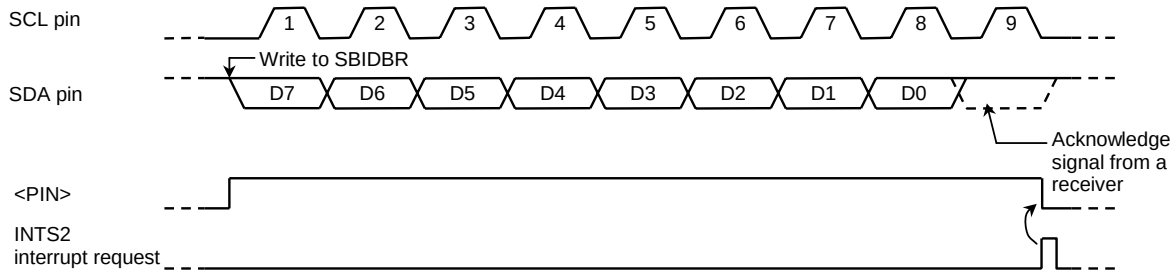


Figure 3.10.13 Example of when <BC2:0> = 000, <ACK> = 1 (Transmitter mode)

When <TRX> is 0 (Receiver mode)

When the next transmitted data is other than 8 bits, set SBICR1<BC2:0> again. Set <ACK> to 1 and read the received data from the SBIDBR to release the SCL line. The read data is undefined immediately after the slave address is set.) 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.

The INTS2 interrupt request then occurs and <PIN> becomes 0. The SCL pin is set to the low level. The TMP93CS20 outputs a clock pulse for one-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

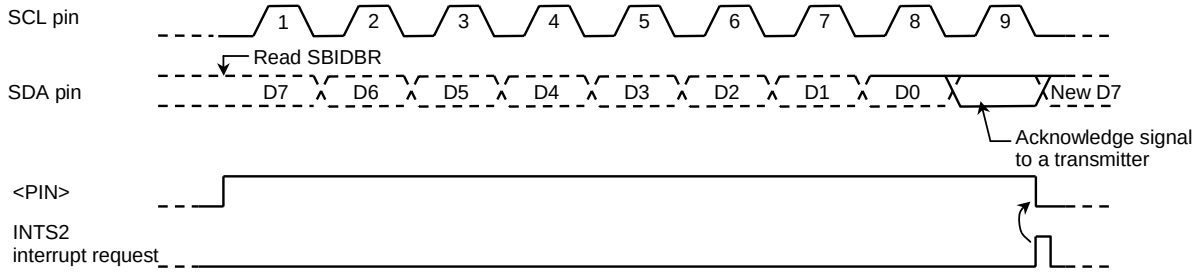


Figure 3.10.14 Example of when <BC2:0> = 000, <ACK> = 1 (Receiver mode)

In order to terminate the transmitting data to the transmitter, set <ACK> to 0 before reading data which is one-word before the last data to be received. The last data does not generate a clock pulse for the acknowledge signal. After the data is transmitted and an interrupt request has occurred, set <BC2:0> to 001 and read the data. The TMP93CS20 generates a clock pulse for a 1-bit data transfer. Since the master device is a receiver, the SDA line of the bus keeps the high level. The transmitter receives the high-level signal as the ACK signal. The receiver indicates to the transmitter that data transfer is complete.

After 1-bit data is received and the interrupt request has occurred, the TMP93CS20 generates the stop condition and terminates data transfer.

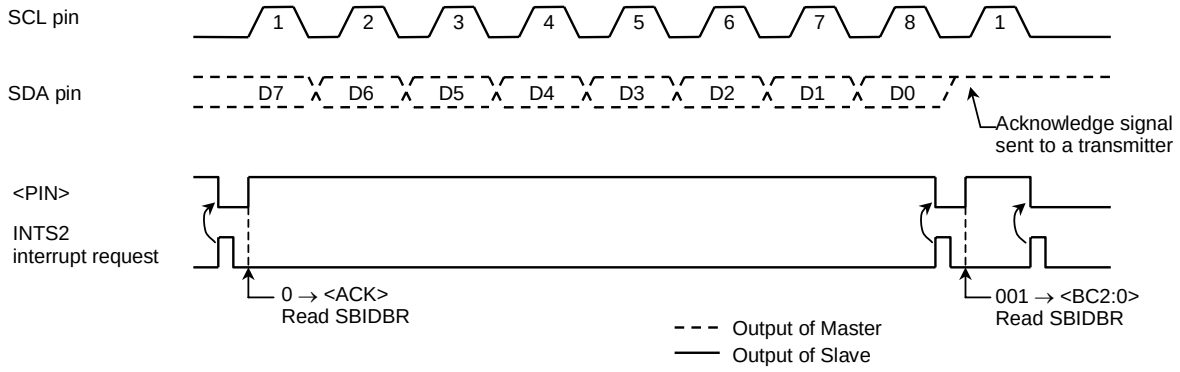


Figure 3.10.15 Termination of Data Transfer in Master Receiver Mode

b. When < MST> is 0 (Slave mode)

In the slave mode, the TMP93CS20 operates either in normal slave mode or in recovery process after a noise detection.

In the slave mode, an INTS2 interrupt request occurs when the serial bus interface circuit receives a slave address or a "GENERAL CALL" from the master device, or when a "GENERAL CALL" is received and data transfer is complete after matching a received slave address. In the master mode, the TMP93CS20 operates in a slave mode if a noise is detected. An INTS2 interrupt request occurs when word data transfer terminates after a noise detection. When an INTS2 interrupt request occurs, the SBISR<PIN> is reset, and the SCL pin is set to low level. Either reading or writing from or to the SBIDBR or setting the <PIN> to 1 releases the SCL pin after taking t_{LOW} time.

The TMP93CS20 tests the SBISR<AL>, the SBISR<TRX>, the <AAS>, and the <AD0> and implements processes according to conditions listed in the next table.

Table 3.10.1 Operation in the Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	Conditions	Process
1	0	1	0	In the slave receiver mode, the TMP93CS20 receives a slave address of which the direction bit sent from the master is "1".	Set the number of bits in one word to <BC2:0> and write transmitted data to the SBIDBR.
		0	0	In the slave transmitter mode, one-word data is transmitted.	Check <LRB>. If <LRB> is set to "1", set <PIN> to "1" since the receiver does not request next data. Then, clear <TRX> to "0" release the bus. If <LRB> is set to "0", set the number of bits in a word to <BC2:0> and write transmitted data to the SBIDBR since the receiver requests next data.
0	0	1	1/0	In the slave receiver mode, the TMP93CS20 receives a slave address or GENERAL CALL of which the direction bit sent from the master is "0".	Read the SBIDBR for setting <PIN> to 1 (Reading dummy data) or write "1" to <PIN>.
		0	1/0	In the slave receiver mode, the TMP93CS20 terminates receiving of one-word data.	Set the number of bits in a word to <BC2:0> and read received data from the SBIDBR.

(4) Stop condition generation

When SBISR<BB> is 1, a sequence of generating a stop condition is started by writing 1 to SBICR2<MST, TRX, PIN>, and 0 to <BB>. Do not modify the contents of <MST, TRX, BB, PIN> until the stop condition is generated on the bus. When SBICR2<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.

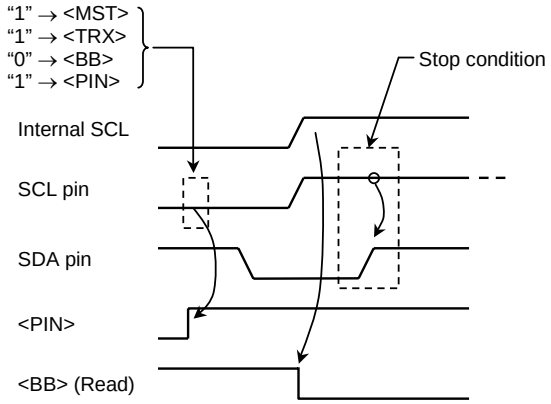


Figure 3.10.16 Stop Condition Generation

(5) Restart

Restart is used to change the direction of data transfer between a master device and a slave device during transferring data. The following explains how to restart the TMP93CS20.

Clear 0 to the <MST>, <TRX>, and <BB> and set 1 to the <PIN>. The SDA pin retains the high level and the SCL pin is released. Since a stop condition is not generated on the bus, the bus is assumed to be in a busy state from other devices. And confirm SCL pin, that SCL pin is released and become bus-free state by SBISR<BB> = "0" or signal level "1" of SCL pin in port mode. Test the <LRB> until it becomes 1 to check that the SCL line of the bus is not set to low level by other devices. After confirming that the bus stays in a free state, generate a start condition with procedure (2).

In order to meet setup time 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.

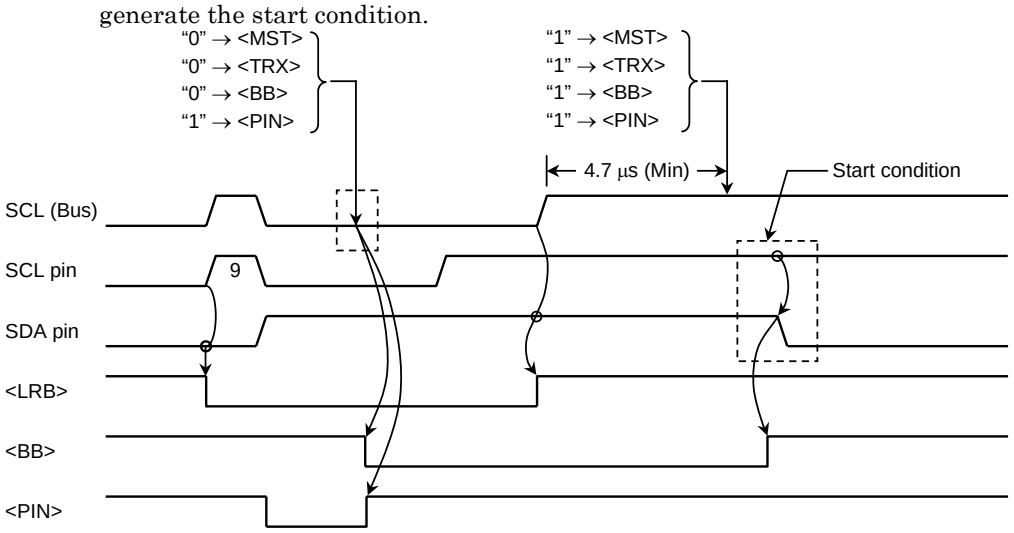


Figure 3.10.17 Timing Diagram when Restarting the TMP93CS20

3.10.6 Clock-synchronous 8-Bit SIO Mode Control

The following registers are used for control and operation status monitoring when using the serial bus interface (SBI) in the clock-synchronous 8-bit SIO mode.

Serial Bus Interface Control Register 1

SBICR1 (004BH)

Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
Read/Write	W					W		
After reset	0	0	0	0		0	0	0
Function	Indication transfer start/stop 0: Stop 1: Start	Continue/ abort transfer 0: Continue 1: Abort	Transfer mode select 00: 8-bit transmit 01: (Reserved) 10: 8-bit transmit/receive 11: 8-bit receive			Serial clock selection		

Serial clock selection

000	$fc/2^5$ (625 kHz)
001	$fc/2^6$ (312.5 kHz)
010	$fc/2^7$ (156.3 kHz)
011	$fc/2^8$ (78.13 kHz)
100	$fc/2^9$ (39.06 kHz)
101	$fc/2^{10}$ (19.53 kHz)
110	$fc/2^{11}$ (9.77 kHz)
111	External clock (input from SCK pin)

(System clock: fc / Clock gear: fc/1 / fc = 20 MHz (Output on SCK pin))

Transfer mode select

00	8-bit transmit mode
01	(Reserved)
10	8-bit transmit/receive mode
11	8-bit receive mode

Continue/abort transfer

0	Continue transfer
1	Abort transfer (Automatically cleared after abort)

Indicate transfer start/stop

0	Stop
1	Start

Set <SIO> to 0 and <SIOINH> to 1 when setting the transfer mode and the serial clock.

Serial Bus Interface Data Buffer Register

SBIDBR (004CH)

Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Receive)/W (Send)							
After reset	Undefined							

Figure 3.10.18 Registers for SIO Mode (1/2)

Serial Bus Interface Control Register 2

SBICR2 (004EH)

Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol					SBIM1	SBIM0		
Read/Write					W			
After reset					0	0		
Function					Serial bus interface operation mode selection 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)			

Serial bus interface operation mode selection

00	Port mode (Serial bus interface output disable)
01	SIO mode
10	I^2C bus mode
11	(Reserved)

Note 1: Switch a mode to port after data transfer is complete.

Note 2: Switch a mode to SIO mode after confirming that input signals via port are high level.

Serial Bus Interface Status Register

SBISR (004EH)

Prohibit read-modify-write

	7	6	5	4	3	2	1	0
Bit symbol					SIOF	SEF		
Read/Write					R			
After reset					0	0		
Function					Serial transfer operating status monitor	Shift operating status monitor		

Sift operating status monitor

0	Shift operation terminated
1	Shift operation in process

Serial transfer operating status monitor

0	Transfer terminated
1	Transfer in process

Figure 3.10.19 Registers for SIO Mode (2/2)

(1) Serial clock

a. Clock source

SBICR1<SCK2:0> are used to select the following functions.

Internal clock

In the internal clock mode, any of seven frequencies can be selected. The serial clock is output to the outside on the SCK pin. The SCK pin becomes a high level when data transfer starts. When writing (in the transmit mode) or reading (in the receive mode) data cannot follow the serial clock rate, an automatic-wait function is executed to stop the serial clock automatically and hold the next shift operation until reading or writing is complete.

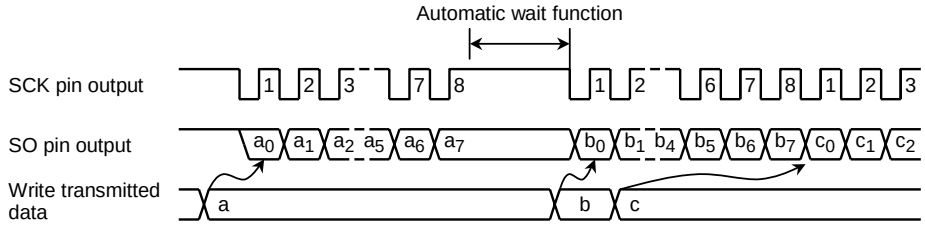
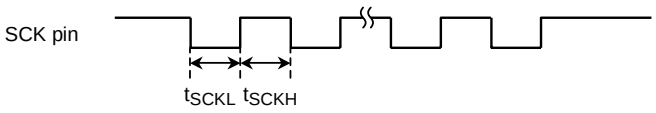


Figure 3.10.20 Automatic Wait Function

External clock (<SCK2:0> = 111)

An external clock supplied to the SCK pin is used as the serial clock. In order to ensure shift operation, a pulse width of at least 4 machine cycles is required for both high and low levels in the serial clock. The maximum data transfer frequency is 625 kHz (at fc = 20 MHz).



t_{SCKL}, t_{SCKH} > 4 tcyc Note: tcyc = $4/f_{FPH}$ (in NORMAL mode, IDLE mode)

Figure 3.10.21 External Clock

b. Shift edge

The leading edge is used to transmit data, and the trailing edge is used to receive data.

Leading edge shift:

Data is shifted on the leading edge of the serial clock (at the falling edge of the SCK pin input/output).

Trailing edge shift:

Data is shifted on the trailing edge of the serial clock (at the rising edge of the SCK pin input/output).

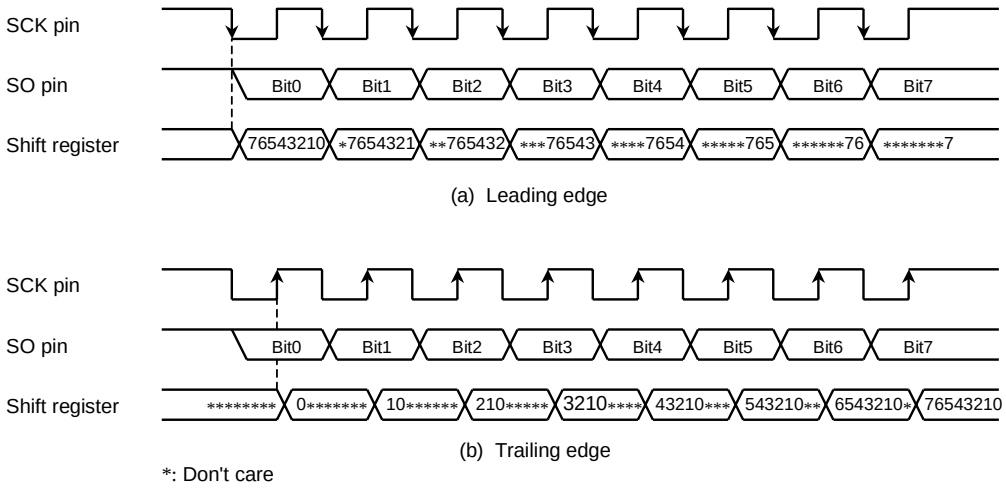


Figure 3.10.22 Shift Edge

(2) Transfer mode

SBICR1<SIOM1:0> are 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 data to the SBIDBR.

After the data is written, set SBICR1<SIOS> to 1 to start data transfer. The transmitted data is transferred from the SBIDBR to the shift register and output to the SO pin in synchronous with the serial clock, starting from the least significant bit (LSB). When the data is transferred to the shift register, the SBIDBR becomes empty. The INTS2 (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 data is written, automatic-wait function is canceled.

When the external clock is used, data should be written to the SBIDBR 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 the SBIDBR by the interrupt service program.

When the transmit is started, after SBISR<SIOF> goes 1 the same value as the final bit of the last data is output until the falling edge of the SCK.

Transmitting data is ended by clearing <SIOS> to 0 with the buffer empty interrupt service program or setting SBICR1<SIOINH> to 1. When <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 <SIOF> to be sensed. <SIOF> is cleared to 0 when transmitting is complete. When <SIOINH> is set to 1, transmitting data stops. <SIOF> turns 0.

When the external clock is used, it is also necessary to clear <SIOS> to 0 before new data is shifted, otherwise, dummy data is transmitted and operation ends.

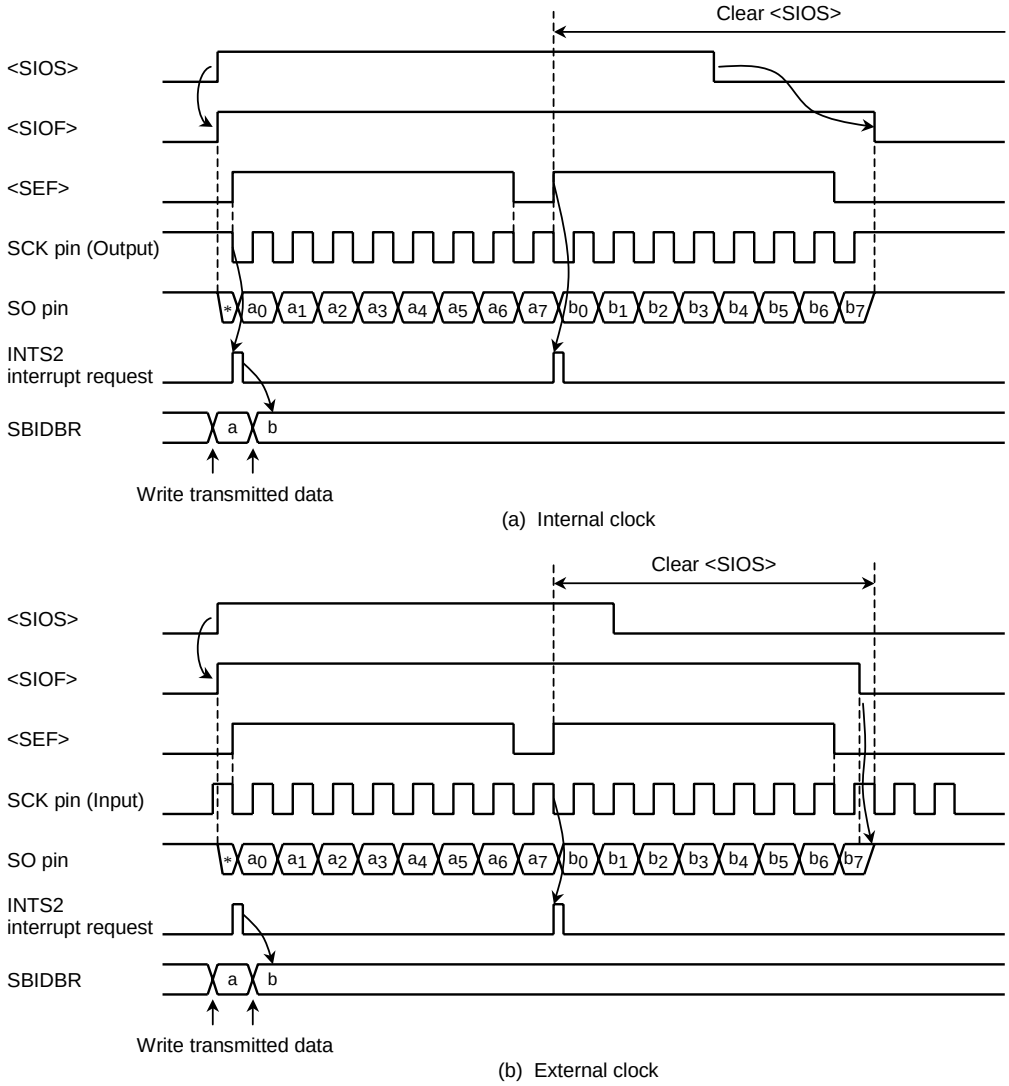


Figure 3.10.23 Transfer Mode

```
Example:   Program to stop transmitting data (when external clock is used)
STEST1:    BIT   SEF, (SBISR)          ;   If <SEF> = 1 then loop.
           JR    NZ, STEST1
STEST2:    BIT   0, (P6)               ;   If SCK = 0 then loop.
           JR    Z, STEST2
           LD    (SBICR1), 00000111B   ;   <SIOS> ← 0.
```

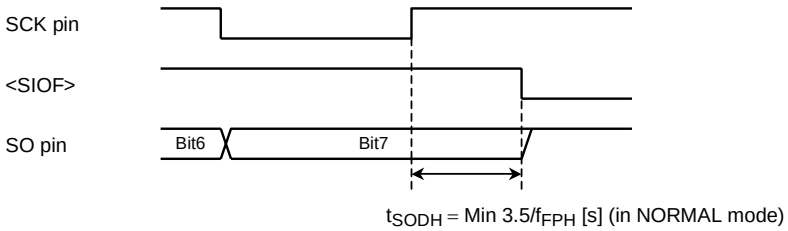


Figure 3.10.24 Transmitted Data Hold Time at End of Transmit

b. 8-bit receive mode

Set the control register to a receive mode and SBICR1<SIOS> to 1 for switching to the receive mode. Data is received from the SI pin to the shift register in synchronous with the serial clock, starting from the least significant bit (LSB). When the 8-bit data is received, the data is transferred from the shift register to the SBIDBR. The INTS2 (Buffer full) interrupt request is generated to request to read the received data. The data is then read from the SBIDBR by the interrupt service program.

When the internal clock is used, the serial clock will stop and automatic wait function will be initiated until the received data is read from the SBIDBR.

When the external clock is used, since shift operation is synchronized with the clock pulse provided externally, the received data should be read from the SBIDBR before next serial clock input. If the received data is not read, further data to be received is canceled. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when received data is read.

Receiving data is ended by clearing <SIOS> to 0 with the buffer full interrupt service program or setting SBICR1<SIOINH> to 1. When <SIOS> is cleared, received data is transferred to the SBIDBR in complete blocks. The received mode ends when the transfer is complete. In order to confirm if data is surely received by the program, set SBISR<SIOF> to be sensed. <SIOF> is cleared to 0 when receiving is complete. After confirming that receiving has ended, the last data is read. When <SIOINH> is set to 1, receiving data stops, <SIOF> turns 0. (The received data becomes invalid, therefore no need to read it.)

Note: When the transfer mode is switched, the SBIDBR contents are lost. In case that the mode needs to be switched, receiving data is concluded by clearing <SIOS> to 0, read the last data, and then switch the mode.

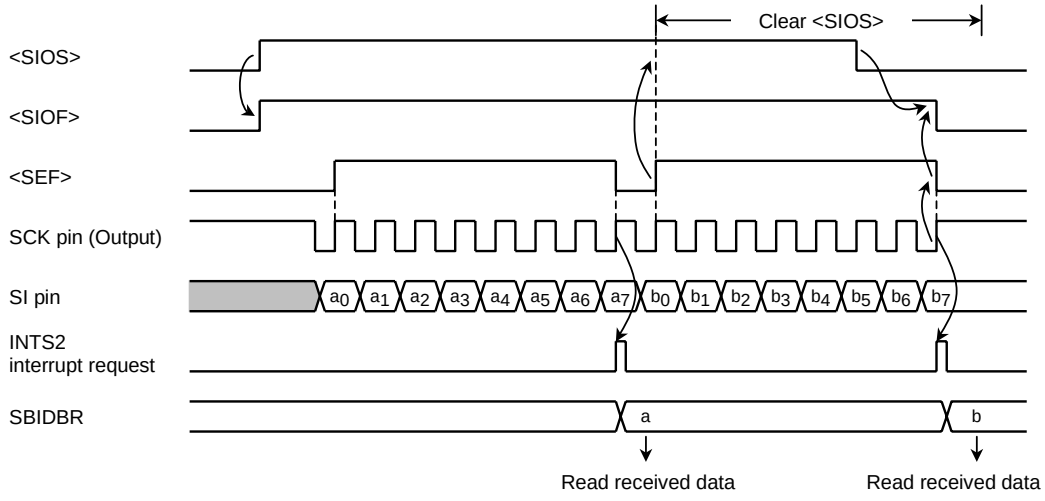


Figure 3.10.25 Receive Mode (Example: Internal clock)

c. 8-bit transmit/receive mode

Set a control register to a transmit/receive mode and write data to the SBIDBR. After the data is written, set SBICR1<SIOS> to 1 to start transmitting/receiving. When transmitting, the data is output from the SO pin on the leading edges in synchronous with the serial clock, starting from the least significant bit (LSB). When receiving, the data is input to the SI pin on the trailing edges of the serial clock. The 8-bit data is transferred from the shift register to the SBIDBR, and the INTS2 interrupt request occurs. The interrupt service program reads the received data from the data buffer register and writes data to be transmitted. The SBIDBR is used for both transmitting and receiving. Transmitted data should always be written after received data is read.

When the internal clock is used, automatic wait function is initiated until received data is read and next data is written.

When the external clock is used, since the shift operation is synchronized with the external clock, received data is read and transmitted data is written before new shift operation is executed. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when received data is read and transmitted data is written.

When the transmit is started, after SBISR<SIOF> 1 output from the SO pin holds final bit of last data until falling edge of the SCK.

Transmitting/receiving data is ended by clearing <SIOS> to 0 by the INTS2 interrupt service program or setting SBICR1<SIOINH> to 1. When <SIOS> is cleared, received data is transferred to the SBIDBR in complete blocks. The transmit/receive mode ends when the transfer is complete. In order to confirm if data is surely transmitted/received by the program, set SBISR<SIOF> to be sensed. <SIOF> becomes 0 after transmitting/receiving is complete. When <SIOINH> is set, transmitting/receiving data stops, <SIOF> turns 0.

Note: When the transfer mode is switched, the SBIDBR contents are lost. In case that the mode needs to be switched, transmitting/receiving data is concluded by clearing <SIOS> to 0, read the last data, and then switch the transfer mode.

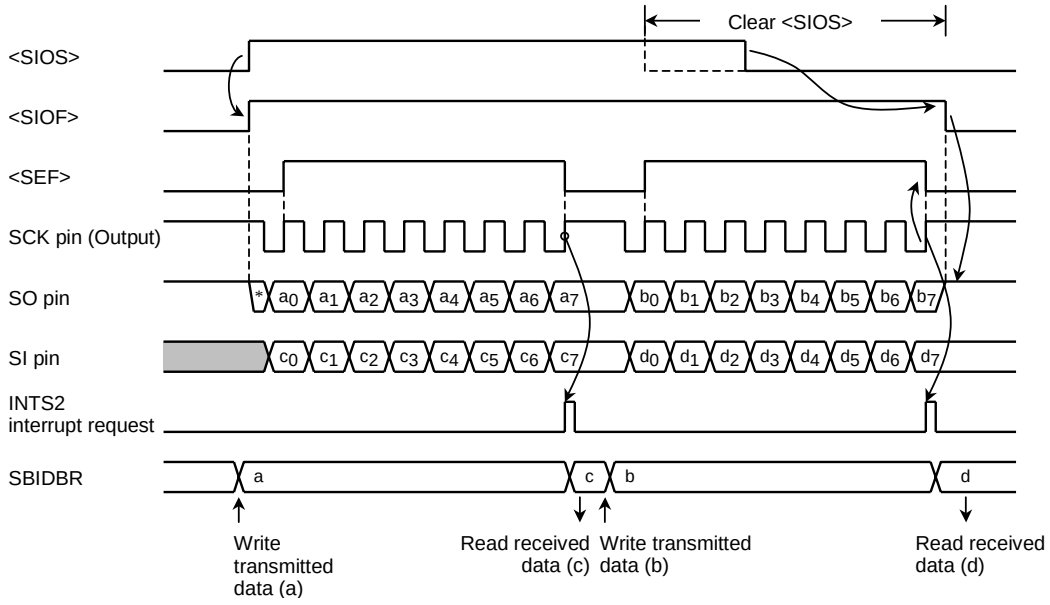


Figure 3.10.26 Transmit/Receive Mode (Example: Internal clock)

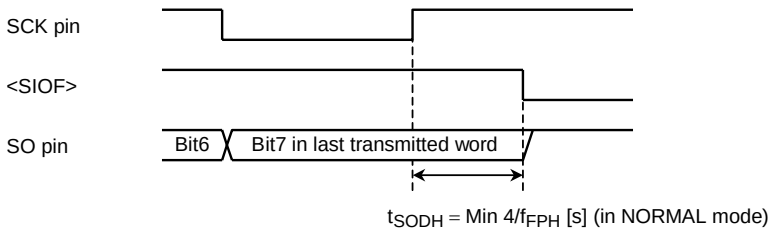


Figure 3.10.27 Transmitted Data Hold Time at End of Transmit/Receive

3.11 Analog/Digital Converter

TMP93CS20 incorporate a high-speed, high-precision 10-bit analog/digital converter (AD converter) with 8-channel analog input.

Figure 3.11.1 is a block diagram of the AD converter. The 8-channel analog input pins (AN0 to AN7) are also used as input only port 5 and can be also used as input ports.

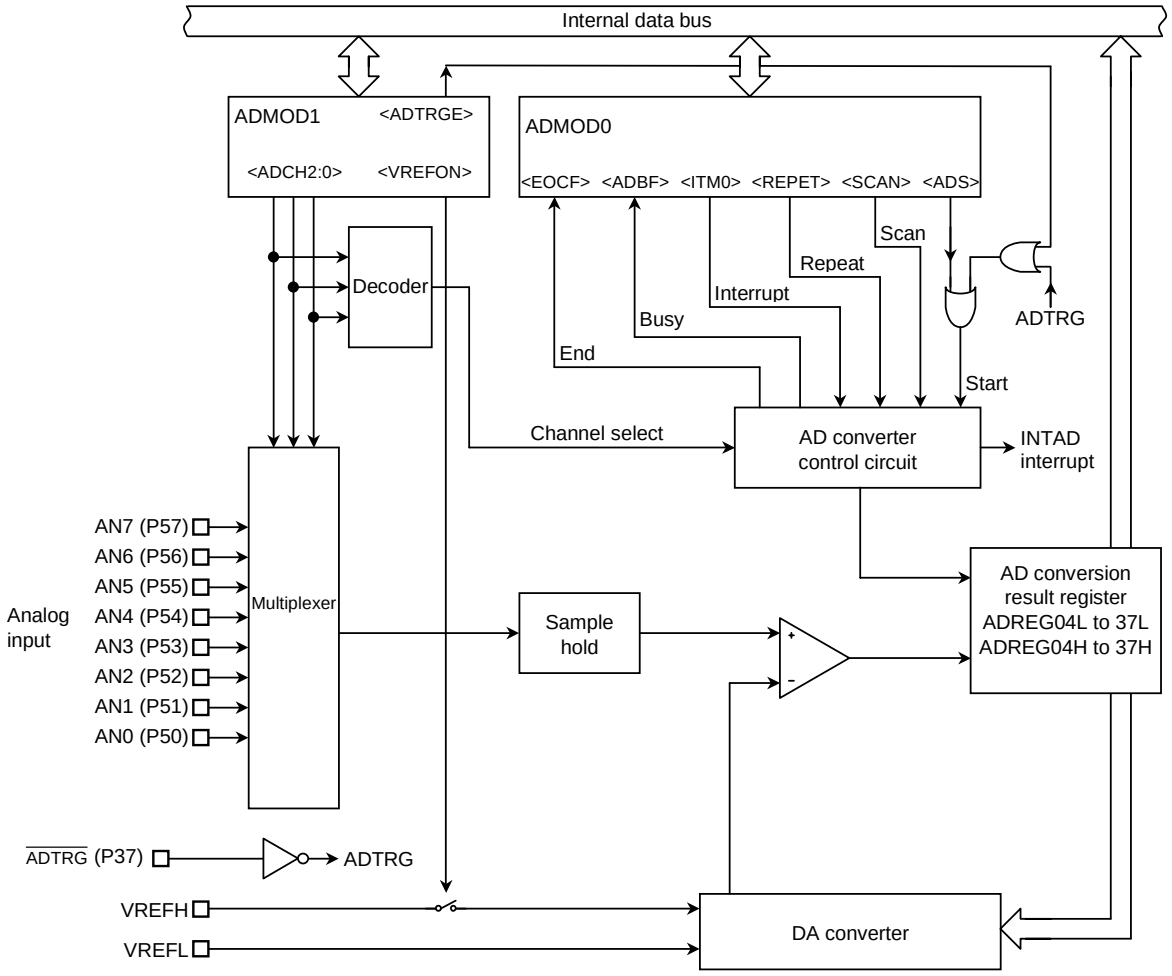


Figure 3.11.1 Block Diagram of AD Converter

Note 1: When the power supply current is reduced in IDLE2, IDLE1, STOP mode, there is possible to set a standby enabling the internal comparator due to a timing. Stop operation of AD converter before execution of HALT instruction.

Note 2: In regard to the lowest operation frequency
The operation of AD converter is guaranteed with clock of $f_{FPH} \geq 4$ MHz (Used fc clock), but not guaranteed with fs clock.

3.11.1 Analog/Digital Converter Registers

AD converter is controlled by two AD mode control registers (ADMOD0 and ADMOD1). AD conversion result is stored in eight AD conversion result registers (ADREG04H/L, ADREG15H/L, ADREG26H/L, ADREG37H/L).

AD Mode Control Register 0

		7	6	5	4	3	2	1	0
ADMOD0 (005EH)	Bit symbol	EOCF	ADBF	–	–	ITM0	REPET	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 end	AD conversion busy flag 1: Conversion is idle 1: Conversion in progress	Always write "0".	Always write "0".	Specifies interrupts for fixed channel /repeat-conversion mode. 0: Every conversion 1: Every four conversions	Repeat mode specification 0: Single conversion mode 1: Repeat conversion mode	Scan mode specification 0: Fixed-channel mode 1: Channel scan mode	AD conversion start 0: Don't care 1: Start conversion (Note)

AD conversion start

0	Don't care
1	Start AD conversion

Note: Always read as 0.

AD scan mode specification

0	AD conversion fixed-channel mode
1	AD conversion channel scan mode

AD repeat mode specification

0	AD single conversion mode
1	AD repeat conversion mode

AD conversion interrupt specification for fixed-channel/repeated-conversion mode

	Fixed channel/repeated conversion mode <SCAN> = 0, <REPET> = 1
0	Generates interrupt every conversion.
1	Generates interrupt every four conversions.

AD conversion busy flag

0	AD conversion is idle
1	AD conversion in progress

AD conversion end flag

0	AD conversion in progress
1	AD conversion end

Figure 3.11.2 Register for AD Converter (1/4)

AD Mode Control Register 1

		7	6	5	4	3	2	1	0
ADMOD1 (005FH)	Bit symbol	VREFON				ADTRGE	ADCH2	ADCH1	ADCH0
	Read/Write	R/W				R/W			
	After reset	1				0	0	0	0
	Function	String resistor 0: OFF 1: ON				External trigger start control 0: Disable 1: Enable	Analog input channel selection		

Analog input channel selection

<ADCH2:0> \ <SCAN>	0 (Channel fix)	1 (Channel scan)
000	AN0	AN0
001	AN1	AN0 → AN1
010	AN2	AN0 → AN1 → AN2
011	AN3	AN0 → AN1 → AN2 → AN3
100	AN4	AN4
101	AN5	AN4 → AN5
110	AN6	AN4 → AN5 → AN6
111	AN7	AN4 → AN5 → AN6 → AN7

Conversion start control by external trigger ($\overline{\text{ADTRG}}$ pin input)

0	Disable
1	Enable

Analog reference voltage control

0	OFF
1	ON

Note: Set the <VREFON> bit to 1 before starting conversion (before writing 1 to ADMOD0<ADS>).

Figure 3.11.3 Register for AD Converter (2/4)

AD Conversion Result Register 0/4 Low

ADREG04L (0060H)

	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.							Conversion result stored flag 1: Exist result

AD Conversion Result Register 0/4 High

ADREG04H (0061H)

	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 of AD conversion result.							

AD Conversion Result Register 1/5 Low

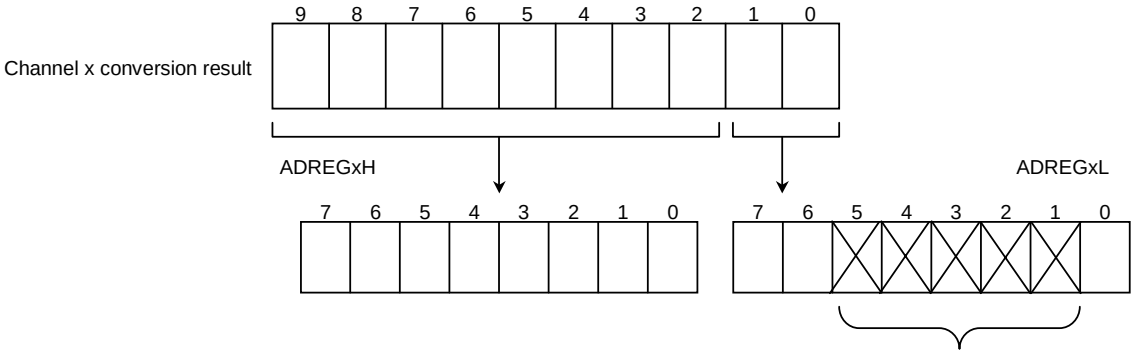
ADREG15L (0062H)

	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.							Conversion result stored flag 1: Exist result

AD Conversion Result Register 1/5 High

ADREG15H (0063H)

	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 conversion result stored flag bit <ADRxRF>.
  <ADRxRF> is set to 1 when the AD conversion result is stored.
  Reading either the ADREGxH or the ADREGxL registers clears <ADRxRF> to 0.

Figure 3.11.4 Registers for AD Converter (3/4)

AD Conversion Result Register 2/6 Low

ADREG26L (0064H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR21	ADR20						ADR2RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result.							Conversion result stored flag 1: Exist result

AD Conversion Result Register 2/6 High

ADREG26H (0065H)

	7	6	5	4	3	2	1	0
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 Result Register 3/7 Low

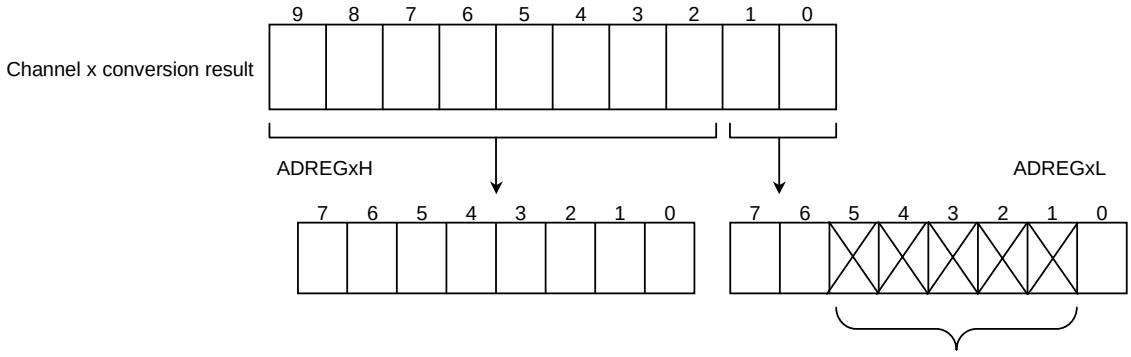
ADREG37L (0066H)

	7	6	5	4	3	2	1	0
Bit symbol	ADR31	ADR30						ADR3RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result.							Conversion result stored flag 1: Exist result

AD Conversion Result Register 3/7 High

ADREG37H (0067H)

	7	6	5	4	3	2	1	0
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 conversion result stored flag bit <ADRxRF>.
  <ADRxRF> is set to 1 when the AD conversion result is stored.
  Reading either the ADREGxH or the ADREGxL registers clears <ADRxRF> to 0.

Figure 3.11.5 Registers for AD Converter (4/4)

3.11.2 Operation

(1) Analog reference voltage

High analog reference voltage is applied to the VREFH pin, and low analog reference voltage is applied to the VREFL pin. The voltage between VREFH and VREFL is divided into 1024 increments using a string resistor. AD conversion is based on comparing the analog input voltage with these reference voltage increments.

To turn the switch between VREFH and VREFL off, write 0 to the ADMOD1<VREFON> bit.

To start AD conversion when the switch is off, first write 1 to <VREFON>. After that, wait at 3 μs long enough to get the stabilized oscillation, write 1 to ADMOD0<ADS>.

(2) Selecting analog input channels

The procedure for selecting analog input channels depends on the operating mode of the AD converter.

- When analog input channel is used to fix (ADMOD0<SCAN> = 0)

  To set ADMOD1<ADCH2:0>, selecting one channel from analog input pins AN0 to AN7.

- When analog input channel is used to scan (ADMOD0<SCAN> = 1)

  To set ADMOD1<ADCH2:0>, selecting one channel from 8 scan mode.

Table 3.11.1 shows the analog input channel selection each operating mode.

A reset initializes ADMOD0<SCAN> to 0 and ADMOD1<ADCH2:0> to 000, selecting pin AN0 for the AD converter input.

The pins not used as analog input channels can be used as general-purpose input ports (P5).

Table 3.11.1 Analog Input Channel Selection

<ADCH2:0>	Fixed Channel <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	AN4	AN4
101	AN5	AN4 → AN5
110	AN6	AN4 → AN5 → AN6
111	AN7	AN4 → AN5 → AN6 → AN7

(3) Starting AD conversion

AD conversion starts when ADMOD0<ADS> to 1, or ADMOD1<ADTRGE> is set to 1 and the falling edge is input through $\overline{\text{ADTRG}}$ pin.

When AD conversion starts, AD conversion busy flag ADMOD0<ADBF> is set to 1, indicating AD conversion is in progress.

Writing 1 to <ADS> while conversion is in progress restarts the conversion. Check the conversion result stored flag ADREGxxL<ADRxRF> to determine whether the AD conversion data are valid at this time.

Inputting the falling edge to the $\overline{\text{ADTRG}}$ pin while conversion is in progress is invalid.

(4) AD conversion modes and completion interrupt

Follow the four AD conversion modes are supported.

- Fixed channel single conversion mode
- Channel scan single conversion mode
- Fixed channel repeat conversion mode
- Channel scan repeat conversion mode

AD conversion mode can selected by setting AD mode control register ADMOD0<REPET, SCAN>.

When AD conversion end, AD conversion completion interrupt INTAD request occurs. And the ADMOD0<EOCF> flag is set to 1 to indicate that AD conversion has completed.

a. Fixed channel single conversion mode

Fixed channel single conversion mode can be specified by setting ADMOD0<REPET, SCAN> to 00.

In this mode, conversion of the specified single channel is executed once only. After conversion is completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0 and occurs INTAD interrupt request.

b. Channel scan single conversion mode

Channel scan single conversion mode can be specified by setting ADMOD0<REPET, SCAN> to 01.

In this mode, conversion of the specified channel are executed once only. After conversion is completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0 and occurs INTAD interrupt request.

c. Fixed channel repeat conversion mode

Fixed channel repeat conversion mode can be specified by setting ADMOD0<REPET, SCAN> to 10.

In this mode, conversion of the specified single channel is executed repeatedly. After conversion is completed, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> remains 1 not changed to 0. The timing of INTAD interrupt request can selected by setting of ADMOD0<ITM0>.

When <ITM0> is set to 0, interrupt request occurs after every conversion.

When <ITM0> is set to 1, interrupt request occurs after every fourth conversion.

d. Channel scan repeat conversion mode

Channel scan repeat conversion mode can be specified by setting ADMOD0<REPET, SCAN> to 11.

In this mode, specified channels are converted repeatedly. After every scan convert completion, ADMOD0<EOCF> is set to 1 and INTAD interrupt request occurs. ADMOD0<ADBF> remains 1, not changed to 0.

To stop the repeat conversion mode (modes c. and d.), write 0 to ADMOD0<REPET>. After the current conversion is completed, repeat conversion mode is terminated, and ADMOD0<ADBF> is cleared to 0.

If the device enters the IDLE2, IDLE1, or STOP modes during AD conversion, the conversion halt immediately. After the halt mode is released, AD conversion restarts from the beginning in repeat conversion mode (c. and d.), it does not restart in single conversion mode (a. and b.).

Table 3.11.2 shows the relations between AD conversion modes and interrupt request.

Table 3.11.2 Relation between AD Conversion Modes and Interrupt Request

Mode	Interrupt Request Timing	ADMOD0		
		<ITM0>	<REPET>	<SCAN>
Fixed channel single conversion mode	After conversion	X	0	0
Channel scan single conversion mode	After conversion	X	0	1
Fixed channel repeat conversion mode (Every conversion)	After every conversion	0	1	0
Fixed channel repeat conversion mode (Every fourth conversion)	After every fourth conversion	1		
Channel scan repeat conversion mode	After every scan conversion	X	1	1

X: Don't Care

(5) AD conversion time

80 states (8.0 μs at fc = 20 MHz) are required for AD conversion of one channel.

(6) Storing and reading the AD conversion result

AD conversion results are stored in AD conversion result registers high/low (ADREG04H/L to ADREG37H/L). These registers are read only.

In fixed channel repeat conversion mode, AD conversion results are stored in order from ADREG04H/L to ADREG37H/L. Except in this mode, AD conversion results for channel AN0 and AN4, AN1 and AN5, AN2 and AN6, AN3 and AN7 are stored severally ADREG04H/L, ADREG15H/L, ADREG26H/L, ADREG37H/L.

Figure 3.11.3 shows correspondence between analog input channels and AD conversion result registers.

Table 3.11.3 Correspondence between Analog Input Channels and AD Conversion Result Registers

Analog Input Channel (Port 5)	AD Conversion Result Registers	
	Conversion Modes Except Right	Fixed Channel Repeat Conversion Mode (Every fourth conversion)
AN0	ADREG04H/L	ADREG04H/L → ADREG15H/L → ADREG26H/L → ADREG37H/L → ADREG04H/L
AN1	ADREG15H/L	
AN2	ADREG26H/L	
AN3	ADREG37H/L	
AN4	ADREG04H/L	
AN5	ADREG15H/L	
AN6	ADREG26H/L	
AN7	ADREG37H/L	

AD conversion result registers bit 0 is AD conversion result stored flag <ADRxRF>. The flag shows that whether those registers are read or not. When AD conversion results are stored in those registers (ADREGxH or ADREGxL), this flag is set to 1. When each register is read, this flag is cleared to 0, and AD conversion end flag ADMOD0<EOCF> is also cleared to 0.

Setting example:

a. This example converts the analog input voltage at the AN3 pin. The INTAD interrupt routine writes the result to memory address 0800H.

Main routine setting:

		7	6	5	4	3	2	1	0	
INTE0AD	←	0	0	0	0	1	1	0	0	Enables INTAD and sets level 4.
ADMOD1	←	1	X	X	X	0	0	1	1	Sets analog input channel to AN3.
ADMOD0	←	X	X	0	0	0	0	0	1	Starts AD conversion in fixed channel single conversion mode.

Example of interrupt routine processing:

WA	←	ADREG37	Reads ADREG37L and ADREG37H values and writes them to WA (16 bits).
WA	>>	6	Shifts right WA six times and zero-fills the upper bits.
(0800H)	←	WA	Writes contents of WA to memory address 0800H.

b. This example repeatedly converts the analog input voltages at pins AN0 to AN2, using channel scan repeat conversion mode.

		7	6	5	4	3	2	1	0	
INTE0AD	←	0	0	0	0	1	0	0	0	Disables INTAD.
ADMOD1	←	1	X	X	X	0	0	1	0	Sets AN0 to AN2 as analog input channels.
ADMOD0	←	X	X	0	0	0	1	1	1	Starts AD conversion in channel scan repeat conversion mode.

X: Don't care, –: No change

3.12 Watchdog Timer (Runaway detection timer) and Warm-up Timer

The TMP93CS20 contains a watchdog timer for runaway detection.

The watchdog timer (WDT) is used to return the CPU to a 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 INTWD interrupt (Non-maskable) to notify the CPU of the malfunction.

Using the malfunction detected result forces a reset.

The watchdog timer consists of 7-stage and 15-stage binary counters.

These binary counters are also used as a warm-up timer for internal oscillator stabilization. This is used for releasing stop and also before changing the system clock.

3.12.1 Configuration

Figure 3.12.1 shows the block diagram for the watchdog timer (WDT) and warm-up timer.

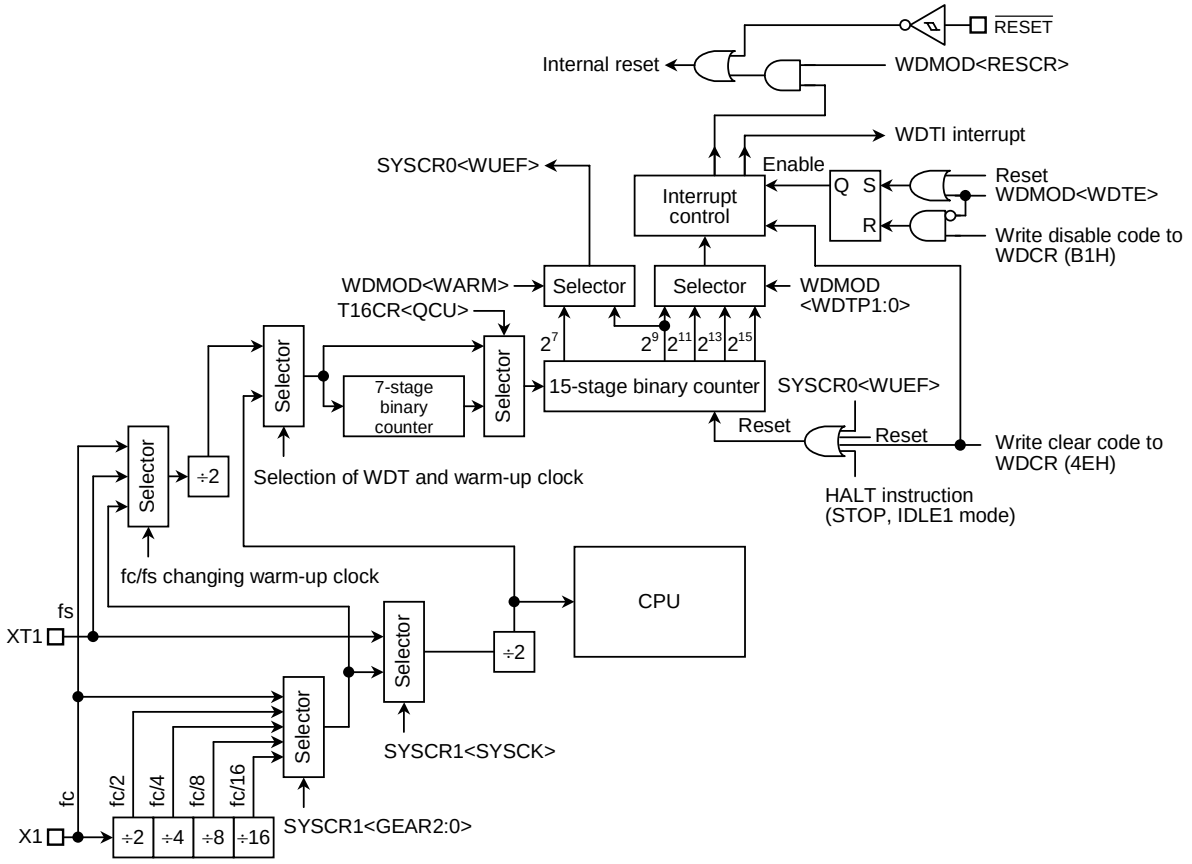


Figure 3.12.1 Block Diagram for Watchdog Timer/Warm-up Timer

The watchdog timer consists of 7-stage and 15-stage binary counters which use system clock (f_{SYS}) as the input clock. The 15-stage binary counter has $f_{SYS}/2^{15}$, $f_{SYS}/2^{17}$, $f_{SYS}/2^{19}$, and $f_{SYS}/2^{21}$ output. Selecting one of the outputs with the WDMOD<WDTP1:0> register generates a watchdog interrupt and outputs watchdog timer out when an overflow occurs. The binary counter for the watchdog timer should be cleared to 0 with runaway detecting result software (Instruction) before an interrupt occurs.

```
LDW    (WDMOD), B100H    ;    Disable.
LD     (WDCR), 4EH       ;    Write clear code.
SET    7, (WDMOD)        ;    Enable again.
```

The runaway detecting result can also be connected to the reset pin internally. In this case, the watchdog timer resets itself.

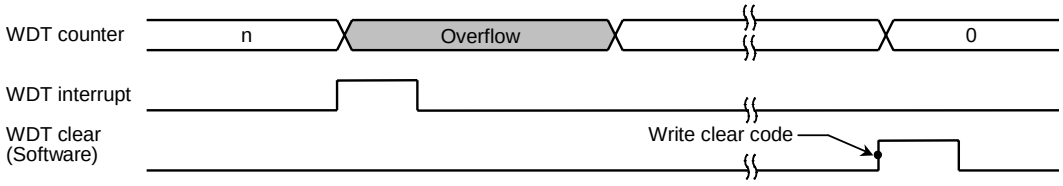


Figure 3.12.2 Normal Mode

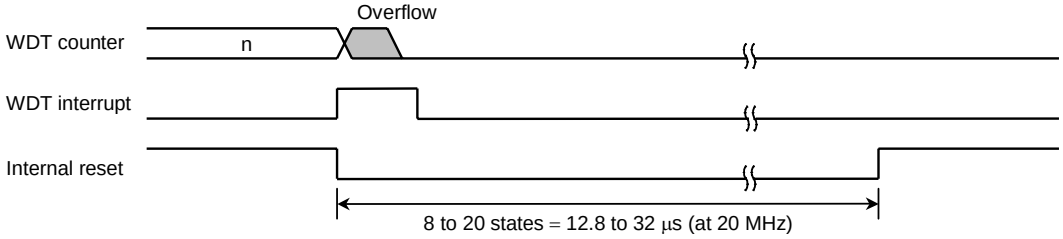


Figure 3.12.3 Reset Mode

For the warm-up counter, either a 2^7 or a 2^9 output from the 15-stage binary counter can be selected using the WDMOD<WARM> register. When a stable-external oscillator is used, a shorter warm-up time is available using the T16CR<QCU> register. When <QCU> = 1, a counting value 2^7 is selected.

When the watchdog timer is in operation, this shorter warm-up time function cannot be selected. The warm-up counter function can be made available by setting <QCU> = 0.

3.12.2 Control Registers

The watchdog timer WDT is controlled by two control registers WDMOD and WDCR.

(1) Watchdog timer mode register (WDMOD)

a. Setting the detection time for the watchdog timer <WDTP>

This 2-bit register is used to set the watchdog timer interrupt time for detecting runaway. This register is initialized to WDMOD<WDTP1:0> = 00 when reset.

The detection time for WDT is shown in Table 3.12.1.

b. Watchdog timer enable/disable control <WDTE>

When reset, WDMOD<WDTE> is initialized to 1 to enable the watchdog timer.

To disable the timer, it is necessary to clear this bit to 0 and write the disable code (B1H) into 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 from the disabled state to the enabled state simply by setting <WDTE> to 1.

c. Watchdog timer out reset connection<RESCR>

This bit is used to connect the output of the watchdog timer with $\overline{\text{RESET}}$ internally. Since WDMOD<RESCR> is initialized to 0 at reset, a watchdog timer reset is not performed.

(2) Watchdog timer control register (WDCR)

This register is used to disable and clear of binary counter the watchdog timer function.

- Disable control

The watchdog timer can be disabled by writing the disable code (B1H) to the WDCR register after clearing WDMOD<WDTE> to 0.

```
WDMOD   ← 0  –  –  –  –  –  X  X    Clear WDMOD<WDTE> to 0.
WDCR    ← 1  0  1  1  0  0  0  1    Write the disable code (B1H).
          X: Don't care, –: No change
```

- Enable control

This sets WDMOD<WDTE> to 1.

- Watchdog timer clear control

The binary counter can be cleared and made to resume counting by writing the clear code (4EH) into the WDCR register.

```
WDCR    ← 0  1  0  0  1  1  1  0    Write the clear code (4EH).
```

WDMOD (005CH)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
Read/Write	R/W							
After reset	1	0	0	0	0	0	0	0
Function	WDT control 0: Disable 1: Enable	Select detection time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$ See Table 3.12.1		Warm-up time	HALT mode 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		1: Internally connects WDT out to the reset pin	1: Drives the pin even in STOP mode

See 3.3 “Dual Clock, Standby Function”.

Watchdog timer enable/disable control

0	Disable
1	Enable

Watchdog timer out control

0	Don’t care
1	Connects WDT out to reset

Selection of warm-up time when changing clock at fc = 20 MHz, fs = 32.768 kHz

WDMOD<WARM>	Changing to Normal (fc)	Changing to Slow (fc)
0 (2^{14}/oscillation frequency)	0.8192 ms	500 ms
1 (2^{16}/oscillation frequency)	3.2768 ms	2000 ms

Selection of warm-up time when returning from the STOP mode at fc = 20 MHz, fs = 32.768 kHz

System Clock Selection <SYSCK>	Gear Value <GEAR2:0>	T45CR<QCU> = 0		T45CR<QCU> = 1
		<WARM> = 0	<WARM> = 1	<WARM> = X
1 (fs)	XXX (Don't care)	0.500 s	2.000 s	3.9 ms
0 (fc)	000 (fc)	0.8192 ms	3.277 ms	6.4 μs
	001 (fc/2)	1.638 ms	6.554 ms	12.8 μs
	010 (fc/4)	3.277 ms	13.107 ms	25.6 μs
	011 (fc/8)	6.554 ms	26.214 ms	51.2 μs
	100 (fc/16)	13.107 ms	52.429 ms	102.4 μs

Note: When using the register as a watchdog timer, write 0 to T16CR<QCU> bit.

Figure 3.12.4 Registers for Watchdog Timer (1/2)

WDCR (005DH)

	7	6	5	4	3	2	1	0
Bit symbol	–							
Read/Write	W							
After reset	–							
Function	Watchdog timer and clear B1H: WDT disable code 4EH: WDT clear code							

→ Disable/clear WDT

B1H	Disable code
4EH	Clear code
Others	Don't set

Note: When using the register as the watchdog timer, write 0 to the T16CR<QCU> bit.

Figure 3.12.5 Registers for Watchdog Timer (2/2)

Table 3.12.1 Watchdog Timer Detection Time

at fc = 20 MHz, fs = 32.768 kHz

System Clock Selection <SYSCK>	Gear Value <GEAR2:0>	Watchdog Timer Detecting Time			
		WDMOD<WDTP1:0>			
		00	01	10	11
1 (fs)	XXX	2.000 s	8.000 s	32.000 s	128.000 s
0 (fc)	000 (fc)	3.277 ms	13.107 ms	52.429 ms	209.715 ms
	001 (fc/2)	6.554 ms	26.214 ms	104.858 ms	419.430 ms
	010 (fc/4)	13.107 ms	53.429 ms	209.715 ms	838.861 ms
	011 (fc/8)	26.214 ms	104.858 ms	419.430 ms	1.678 s
	100 (fc/16)	52.429 ms	209.715 ms	838.861 ms	3.355 s

3.12.3 Operation

The watchdog timer generates interrupt INTWD after the detection time set in the WDMOD<WDTP1:0>. The watchdog timer must be cleared to 0 by software before an INTWD interrupt is generated. If the CPU malfunctions (Undergoes runaway) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter overflows and an INTWD interrupt is generated. The CPU detects malfunction (Runaway) from the INTWD interrupt and it is possible to return to normal operation using a recovery program. By connecting the watchdog timer out pin to peripheral devices resets, a CPU malfunction can also be acknowledged to other devices.

The watchdog timer restarts operation immediately after reset is released.

The watchdog timer does not operate in IDLE1 or STOP mode.

The watchdog timer is enabled in RUN or IDLE2 mode. In IDLE2 mode, disable the watchdog timer before entering the halt state to prevent a watchdog timer interrupt.

Example:

a. Clear the binary counter.

WDCR	← 0 1 0 0 1 1 1 0	Write clear code (4EH).

b. Set the watchdog timer detecting time to $2^{17}/f_{SYS}$.

WDMOD	← 1 0 1 – – – X X	

c. Disable the watchdog timer.

WDMOD	← 0 – – – – – X X	Clear WDTE to 0.
WDCR	← 1 0 1 1 0 0 0 1	Write disable code (B1H).

d. Set IDLE1 mode.

WDMOD	← 0 – – – 1 0 X X	Disable WDT and set IDLE1 mode.
WDCR	← 1 0 1 1 0 0 0 1	
Executes Halt command		Set HALT mode.

e. Set the STOP mode (warm-up time: $2^{16}/f_{SYS}$)

WDMOD	← – – – 1 0 1 X X	Set the STOP mode.
Executes Halt command		Set HALT mode.

X: Don't care, –: No change

3.13 Timer for Real Time Clock

The TMP93CS20 includes a timer which is used for a clock operation.

An interrupt (INTRTC) can be generated each 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 mode in which a low-frequency oscillation is operated.

In addition, INTRTC can return from each standby mode except STOP mode.

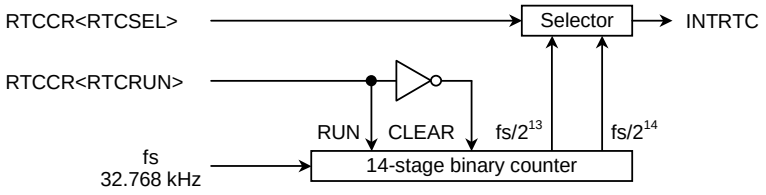


Figure 3.13.1 Block Diagram for Timer for Real Time Clock

The timer for real time clock is controlled by a timer for real time clock control register (RTCCR).

Figure 3.13.2 shows the timer for real time clock control register.

RTCCR (0059H)

	7	6	5	4	3	2	1	0
Bit symbol	–						RTCSEL	RTCRUN
Read/Write	R/W						R/W	R/W
After reset	0						0	0
Function	Write “0”.						0: $fs/2^{14}$ 1: $fs/2^{13}$	0: Stop and clear 1: Run

Counting operation

0	Stop and clear
1	Count

Interrupt generation cycle (fs = 32.768 kHz)

0	0.50 s
1	0.25 s

Figure 3.13.2 Timer for Real Time Clock Control Register

3.14 LCD Driver

The TMP93CS20 includes a driver which drives a liquid crystal display (LCD) directly and the control circuit.

The pins to be connected to the LCD are as follows:

a. Segment output pin 24 pins (SEG23 to SEG0)

b. Segment output/P9 and PA I/O port pin 16 pins (SEG39 to SEG24)

c. Common output pin 4 pins (COM3 to COM0)

Additionally, the TMP93CS20 includes C0, C1, V1, V2, and V3 pins which are used for the LCD drive constant voltage boosting circuit.

The following LCDs can drive directly.

a. 1/4 duty (1/3 bias) LCD ..... maximum 160 pixels (8 segments × 20 digits)

b. 1/3 duty (1/3 bias) LCD ..... maximum 120 pixels (8 segments × 15 digits)

c. 1/2 duty (1/3 bias) LCD ..... maximum 80 pixels (8 segments × 10 digits)

d. Static LCD ..... maximum 40 pixels (8 segments × 5 digits)

3.14.1 Configuration

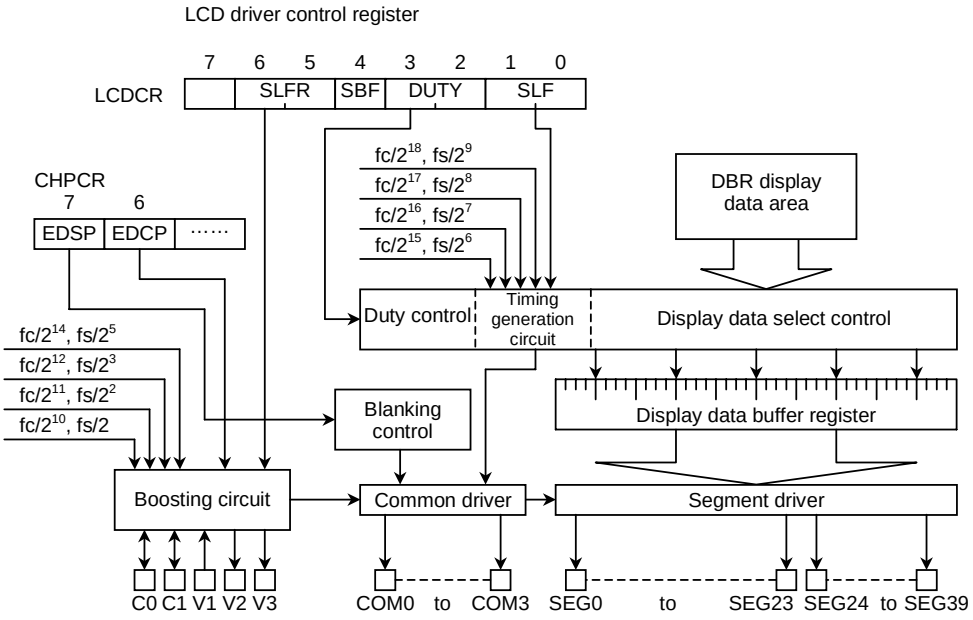


Figure 3.14.1 LCD Driver

3.14.2 LCD Driver Control

The LCD driver is controlled by a LCD control register (LCDCR) and a LCD boosting circuit control register (CHPCR). The LCD driver display is enabled by the CHPCR<EDSP>.

LCD Control Register

		7	6	5	4	3	2	1	0
LCDCR (0094H)	Bit symbol		SLFR1	SLFR0	SBF	DUTY1	DUTY0	SLF1	SLF0
	Read/Write		W						
	After reset		0	0	0	0	0	0	0
Read-modify-write is prohibited	Function		Boosting frequency 00: $fc/2^{14}$ or $fs/2^5$ 01: $fc/2^{12}$ or $fs/2^3$ 10: $fc/2^{11}$ or $fs/2^2$ 11: $fc/2^{10}$ or $fs/2$		Reference clock 0: fc 1: fs	LCD drive 00: 1/4 duty 01: 1/3 duty 10: 1/2 duty 11: Static		Base frequency 00: $fc/2^{18}$ or $fs/2^9$ 01: $fc/2^{17}$ or $fs/2^8$ 10: $fc/2^{16}$ or $fs/2^7$ 11: $fc/2^{15}$ or $fs/2^6$	

Selection of base frequency

00	$fc/2^{18}$ or $fs/2^9$
01	$fc/2^{17}$ or $fs/2^8$
10	$fc/2^{16}$ or $fs/2^7$
11	$fc/2^{15}$ or $fs/2^6$

Setting of LCD drive

00	1/4 duty (1/3 bias)
01	1/3 duty (1/3 bias)
10	1/2 duty (1/3 bias)
11	Static

Selection of reference clock frequency for base and boosting frequencies

00	fc
01	fs

Selection of boosting frequency

00	$fc/2^{14}$ or $fs/2^5$
01	$fc/2^{12}$ or $fs/2^3$
10	$fc/2^{11}$ or $fs/2^2$
11	$fc/2^{10}$ or $fs/2$

Figure 3.14.2 LCD Driver Control Registers

LCD Boosting Circuit Control Register

CHPCR (0095H)

	7	6	5	4	3	2	1	0
Bit symbol	EDSP	EDCP						
Read/Write	R/W							
After reset	0	0						
Function	LCD display 0: Blanking 1: Display enable	Boosting operation 0: Disable 1: Enable						

Boosting operation enable/disable

0	Disable
1	Enable

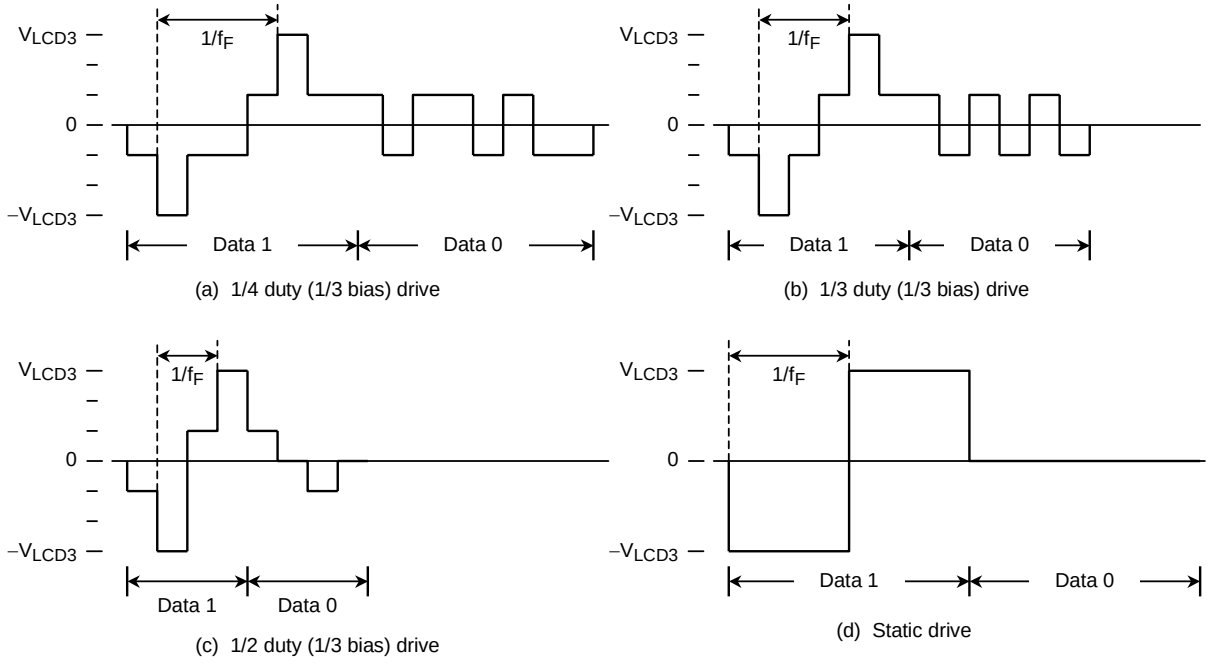
LCD display control

0	Blanking
1	Display enable

Figure 3.14.3 LCD Boosting Circuit Control Register

(1) LCD drive

There are four types to drive the LCD, which are selected by the LCDCR<DUTY1:0>. The driving method is initialized according to the LCD to be used in the initial program.



f_F: Frame frequency
V_{LCD3}: LCD drive voltage

Figure 3.14.4 LCD Drive Pulse (Electrical difference between COM and SEG)

(2) Frame frequency

The frame frequency (f_F) is specified according to a driving method and a base frequency, which is shown in Table 3.14.1.

The base frequency is selected by the LCDCR<SLF1:0>. It depends on the reference clock frequencies (fc and fs) to be used.

Table 3.14.1 Frame Frequency (1/2) at <SBF> = 0

<SLF1:0>	Base Frequency [Hz]	Frame Frequency [Hz]			
		1/4 Duty	1/3 Duty	1/2 Duty	Static
00	$\frac{fc}{2^{18}}$	$\frac{fc}{2^{18}}$	$\frac{4}{3}\bullet\frac{fc}{2^{18}}$	$\frac{4}{2}\bullet\frac{fc}{2^{18}}$	$\frac{fc}{2^{18}}$
	(at fc = 20 MHz)	76	101	152	76
01	$\frac{fc}{2^{17}}$	$\frac{fc}{2^{17}}$	$\frac{4}{3}\bullet\frac{fc}{2^{17}}$	$\frac{4}{2}\bullet\frac{fc}{2^{17}}$	$\frac{fc}{2^{17}}$
	(at fc = 16 MHz)	122	162	244	122
10	$\frac{fc}{2^{16}}$	$\frac{fc}{2^{16}}$	$\frac{4}{3}\bullet\frac{fc}{2^{16}}$	$\frac{4}{2}\bullet\frac{fc}{2^{16}}$	$\frac{fc}{2^{16}}$
	(at fc = 8 MHz)	122	162	244	122
11	$\frac{fc}{2^{15}}$	$\frac{fc}{2^{15}}$	$\frac{4}{3}\bullet\frac{fc}{2^{15}}$	$\frac{4}{2}\bullet\frac{fc}{2^{15}}$	$\frac{fc}{2^{15}}$
	(at fc = 4 MHz)	122	162	244	122

fc: High frequency clock [Hz]

Table 3.14.2 Frame Frequency (2/2) at <SBF> = 1

<SLF1:0>	Base Frequency [Hz]	Frame Frequency [Hz]			
		1/4 Duty	1/3 Duty	1/2 Duty	Static
00	$\frac{fs}{2^{9}}$	$\frac{fs}{2^{9}}$	$\frac{4}{3}\bullet\frac{fs}{2^{9}}$	$\frac{4}{2}\bullet\frac{fs}{2^{9}}$	$\frac{fs}{2^{9}}$
	(at fs = 32.768 kHz)	64	85	128	64
01	$\frac{fs}{2^{8}}$	$\frac{fs}{2^{8}}$	$\frac{4}{3}\bullet\frac{fs}{2^{8}}$	$\frac{4}{2}\bullet\frac{fs}{2^{8}}$	$\frac{fs}{2^{8}}$
	(at fs = 32.768 kHz)	128	171	256	128
10	$\frac{fs}{2^{7}}$	$\frac{fs}{2^{7}}$	$\frac{4}{3}\bullet\frac{fs}{2^{7}}$	$\frac{4}{2}\bullet\frac{fs}{2^{7}}$	$\frac{fs}{2^{7}}$
	(at fs = 32.768 kHz)	256	341	512	256
11	$\frac{fs}{2^{6}}$	$\frac{fs}{2^{6}}$	$\frac{4}{3}\bullet\frac{fs}{2^{6}}$	$\frac{4}{2}\bullet\frac{fs}{2^{6}}$	$\frac{fs}{2^{6}}$
	(at fs = 32.768 kHz)	512	683	1024	512

fs: Low frequency clock [Hz]

(3) LCD drive constant voltage boosting circuit

The TMP93CS20 includes a LCD drive boosting circuit ($V_{LCD1} \times 3$) to prevent a flicker on the LCD display when the power supply voltage changes.

The boosting circuit increased an input voltage of a reference voltage input pin by 2 times (V_{LCD2}) and 3 times (V_{LCD3}), and generates an output voltage for segment and common signals.

An example of the LCD drive boosting circuit is shown in Figure 3.14.5.

The reference frequency of the boosting circuit is $fc/2^{14}$ initially. The reference frequency is selected by the LCDCR<SLFR1:0> to increase a driving ability of the segment and common. The reference frequency of the boosting circuit is shown in Table 3.14.2.

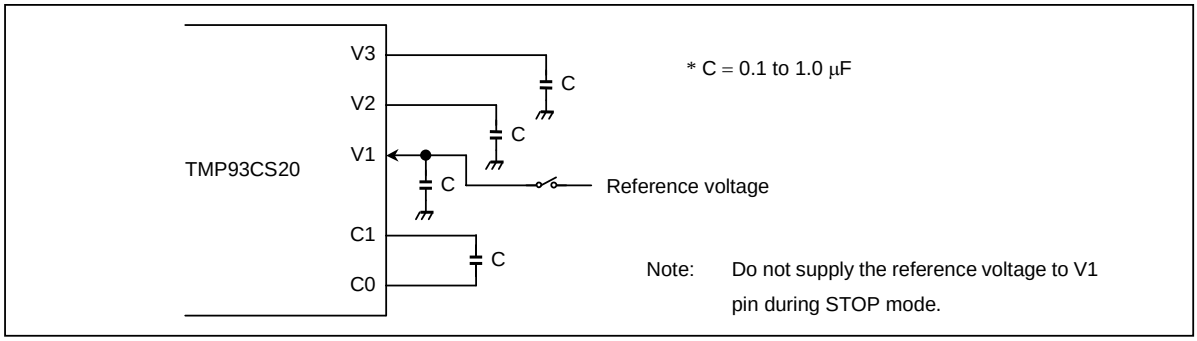


Figure 3.14.5 An Example of Constant Voltage Boosting Circuit

Table 3.14.3 Boosting Frequency

<SLFR1:0>	Frequency	fc = 20 MHz (<SBF> = 0)	fs = 32.768 kHz (<SBF> = 1)
00	$fc/2^{14}$ or $fs/2^5$	1.22 kHz	1.02 kHz
01	$fc/2^{12}$ or $fs/2^3$	4.88 kHz	4.09 kHz
10	$fc/2^{11}$ or $fs/2^2$	9.76 kHz	8.19 kHz
11	$fc/2^{10}$ or fs/2	19.53 kHz	16.38 kHz

3.14.3 LCD Display

(1) Setting of display data

A display data is stored in a display data area which is allocated in 20 bytes address area between addresses 000080H to 000093H.

The display data stored in the display data area is automatically read by hardware, and transferred to the LCD driver. The LCD driver generates a segment signal and a common signal according to the display data and the driving method. The display pattern can be changed by changing data stored in the display data area in the program.

Figure 3.14.6 shows SEG and COM pins in the display data areas. These pins turn on when a display data is 1. They turn off when a display data is 0.

In addition to a number of pixels, a number of bits in the display data area depend on the LCD driving methods.

Address		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
DBR00	0080H	SEG1				SEG0			
DBR01	0081H	SEG3				SEG2			
DBR02	0082H	SEG5				SEG4			
DBR03	0083H	SEG7				SEG6			
DBR04	0084H	SEG9				SEG8			
DBR05	0085H	SEG11				SEG10			
DBR06	0086H	SEG13				SEG12			
DBR07	0087H	SEG15				SEG14			
DBR08	0088H	SEG17				SEG16			
DBR09	0089H	SEG19				SEG18			
DBR0A	008AH	SEG21				SEG20			
DBR0B	008BH	SEG23				SEG22			
DBR0C	008CH	SEG25				SEG24			
DBR0D	008DH	SEG27				SEG26			
DBR0E	008EH	SEG29				SEG28			
DBR0F	008FH	SEG31				SEG30			
DBR10	0090H	SEG33				SEG32			
DBR11	0091H	SEG35				SEG34			
DBR12	0092H	SEG37				SEG36			
DBR13	0093H	SEG39				SEG38			
		COM3	COM2	COM1	COM0	COM3	COM2	COM1	COM0

SEG0 to SEG39 can be read and written.

Figure 3.14.6 LCD Display Data Area

Table 3.14.4 A Number of Bits in the Display Data Area

Driving Method	Bit 7/3	Bit 6/2	Bit 5/1	Bit 4/0
1/4 duty	COM3	COM2	COM1	COM0
1/3 duty	–	COM2	COM1	COM0
1/2 duty	–	–	COM1	COM0
Static	–	–	–	COM0

–: Bits not to be used in the display data area

(2) Blanking

When the CHPCR<EDSP> is cleared to 0, a blanking occurs. The GND level is output to the COM and SEG pins, and the LCD turns off.

Note: Although a pin output only for segments (SEG0 to SEG23), common output, and output port/segment combination pin (P9 port) output serves as GND level at the time of reset, an output port/segment combination pin (PA port) will benefit an open-drain output in a high impedance state. Therefore, when port A is used as a segment and the reset input from the exterior becomes remarkably long, there is a possibility of doing the influence to which the display of LCD spreads etc.

3.14.4 LCD Driver Control Method

(1) Initialization

A flow chart of an initialization is shown in Figure 3.14.7.

Example: When 40 segments × 4 commons and 1/4 duty LCD is operated at the frame frequency $fs/2^9$ [Hz] and the boosting frequency $fs/2^3$ [Hz].

```
LD      (LCDCR), 00110000B      ;   Sets a LCD driving method, a frame frequency, and
                                    a boosting frequency.
SET     6, (CHPCR)              ;   Starts boosting.
LD      (P9FC), 0FFH            ;   Sets P9 and PA port as segment outputs.
LD      (PAFC), 0FFH            ;   Sets an initial value of a display data. Waits for
                                    boosting.
SET     7, (CHPCR)              ;   Display enable.
```

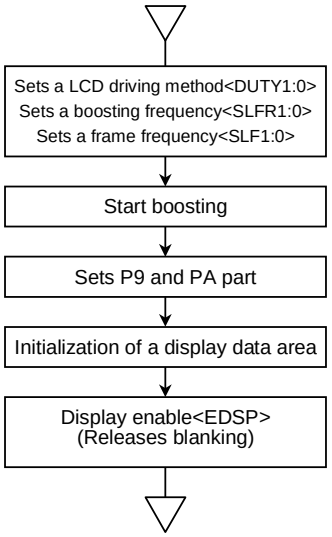


Figure 3.14.7 Initialization of the LCD Driver

(2) Storing of display data

Usually, the display data is provided as a constant data in a program memory (ROM). It is stored by the load instruction in the display data area.

Example 1: To display the BCD data by the 1/4 duty LCD which is connected to the COM and SEG pins as Figure 3.14.8, the display data is shown in Table 3.14.4.

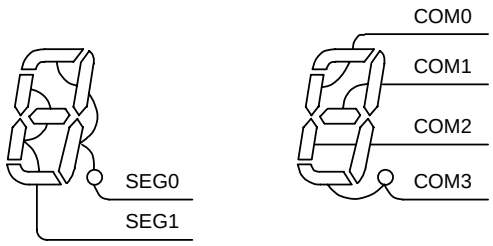
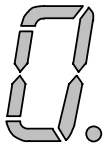
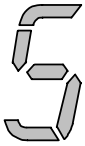
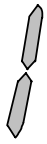
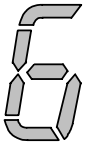
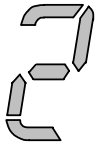
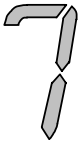
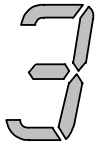
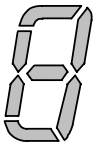
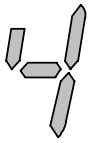
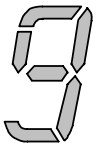


Figure 3.14.8 An Example of Connecting of COM and SEG Pins

Table 3.14.5 An Example of Display Data (1/4 Duty)

Number	Display	Display Data	Number	Display	Display Data
0		11011111	5		10110101
1		00000110	6		11110101
2		11100011	7		00000111
3		10100111	8		11110111
4		00110110	9		10110111

Example 2: To display the BCD data by the 1/2 duty LCD which is connected to the COM and SEG pins as Figure 3.14.9, the display data is shown in Table 3.14.5.

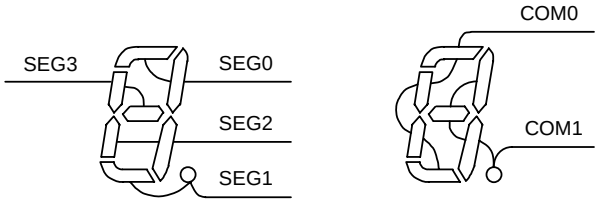


Figure 3.14.9 An Example of Connection of COM and SEG Pins

Table 3.14.6 An Example of Display Data (1/2 duty)

Number	Display Data		Number	Display Data	
	Upper Address	Lower Address		Upper Address	Lower Address
0	**01**11	**01**11	5	**11**10	**01**01
1	**00**10	**00**10	6	**11**11	**01**01
2	**10**01	**01**11	7	**01**10	**00**11
3	**10**10	**01**11	8	**11**11	**01**11
4	**11**10	**00**10	9	**11**10	**01**11

*: Don't care

(3) Example of LCD driving

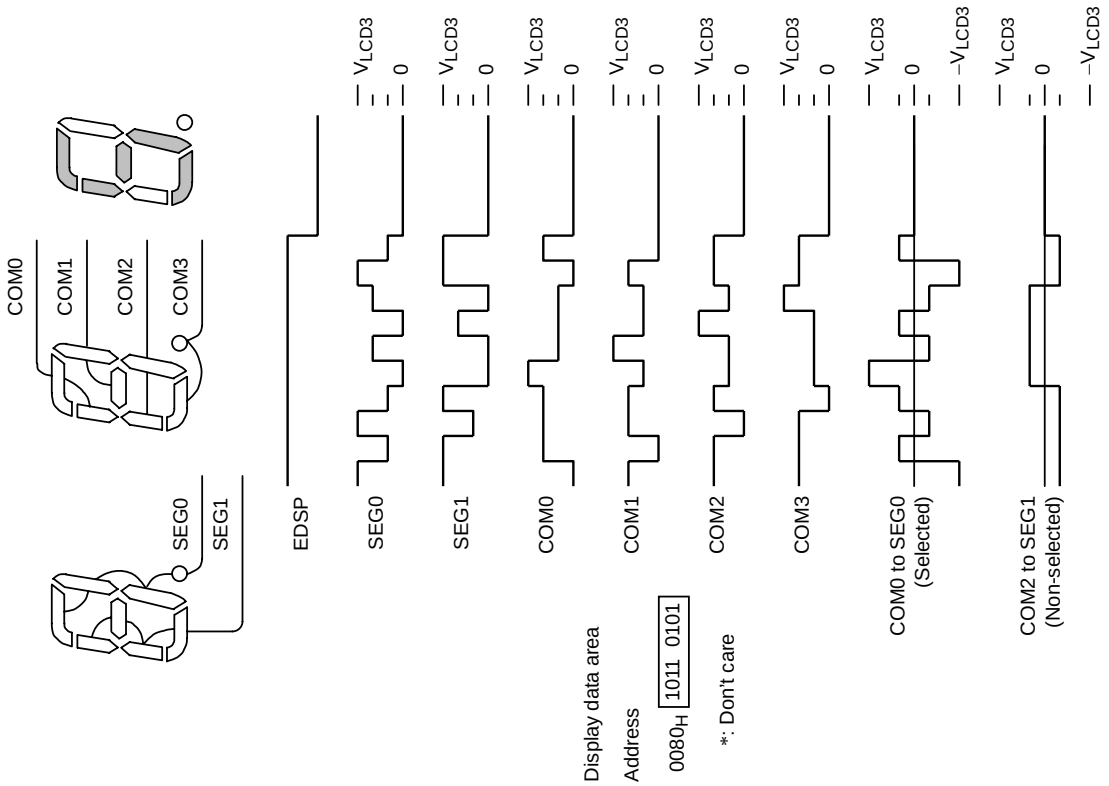


Figure 3.14.10 1/4 Duty Drive (1/3 bias)

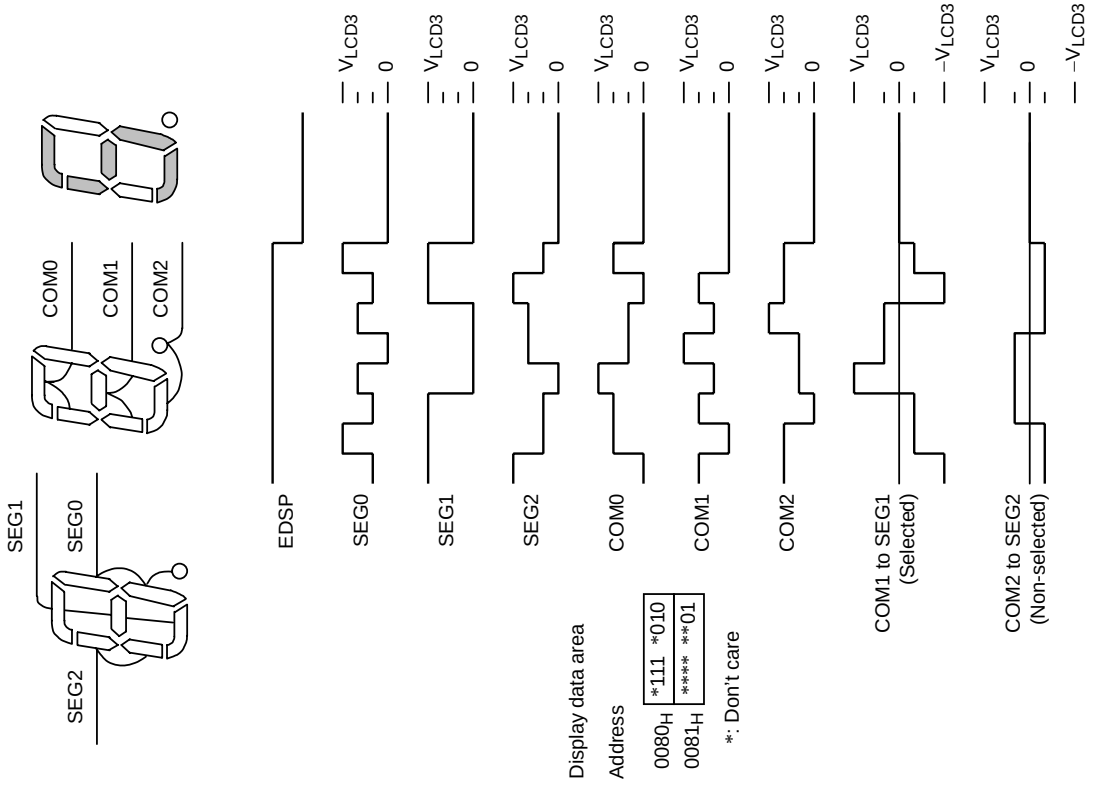


Figure 3.14.11 1/3 Duty Drive (1/3 bias)

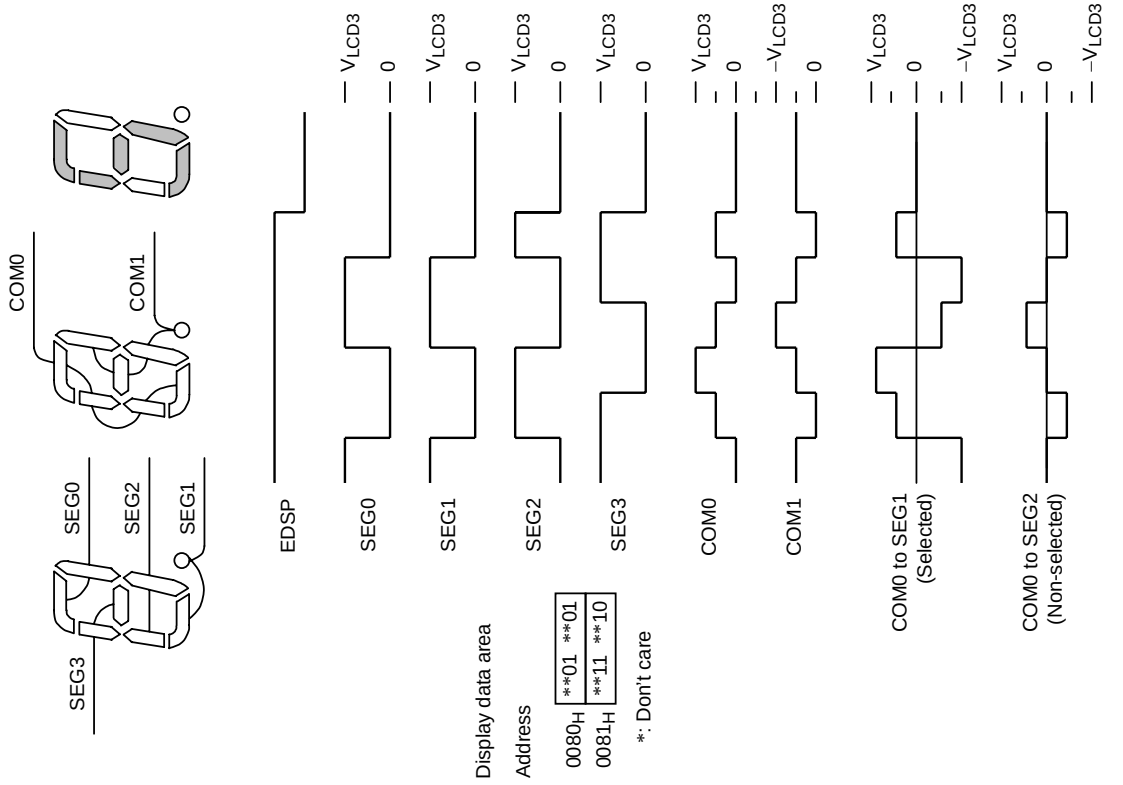


Figure 3.14.12 1/2 Duty Drive (1/3 bias)

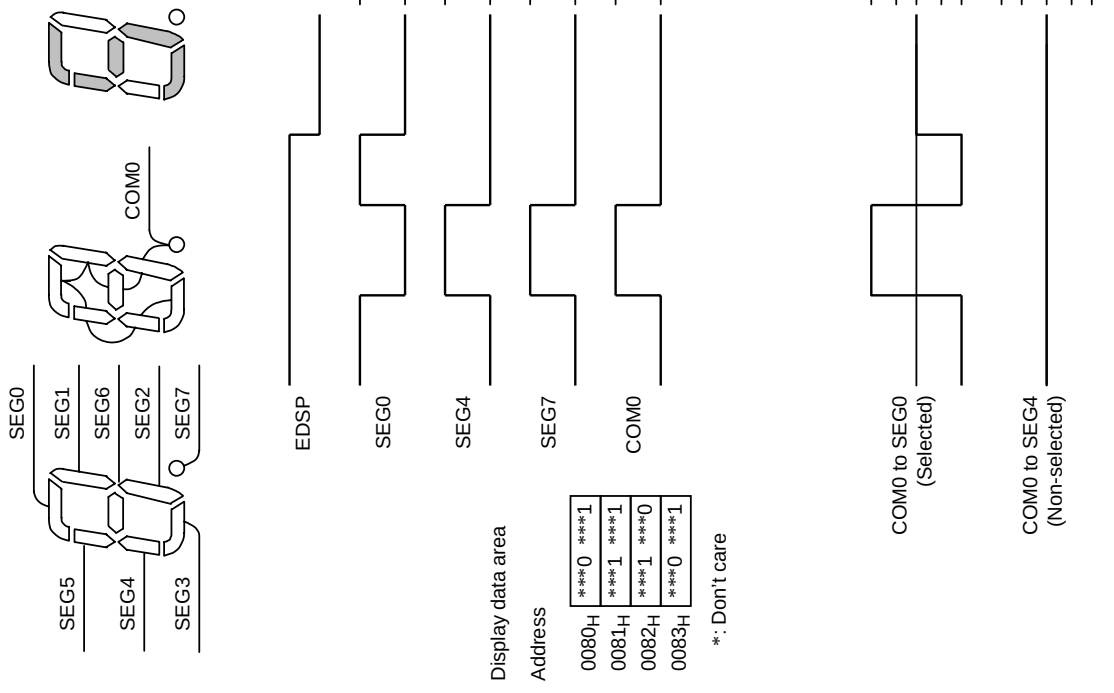


Figure 3.14.13 Static Drive

3.15 Key-on Wakeup (Key input interrupt)

The TMP93CS20 has eight key-on wakeup pins.
A key input interrupt (INTKEY) is generated at the falling edge of a key-on wakeup pin.
A key input interrupt can return from each standby mode and execute an interrupt process.

Note: The key-on wakeup pins are also used as P40 to P47. The pins which are not used as a key-on wakeup pin must be disabled.

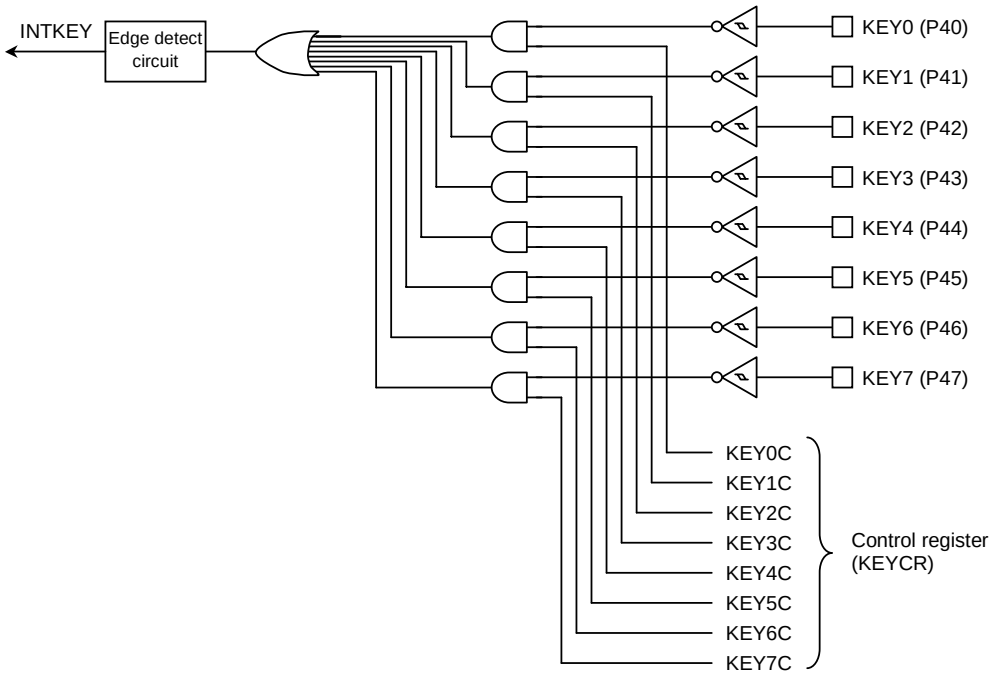


Figure 3.15.1 Block Diagram of Key-on Wakeup

The key-on wakeup control register (KEYCR) controls the key-on wakeup pins each 1 bit.

KEYCR (004AH)

	7	6	5	4	3	2	1	0
Bit symbol	KEY7C	KEY6C	KEY5C	KEY4C	KEY3C	KEY2C	KEY1C	KEY0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Key-on wakeup disable 1: Key-on wakeup enable							

Figure 3.15.2 Key-on Wakeup Control Register

4. Electrical Characteristics

"X" used in an expression shows a frequency for the clock f_{FPH} selected by SYSCR1<SYSCK>. The value of X changes according to whether a clock gear or a low speed oscillator is selected. An example value is calculated for fc, with gear = fc/1 (SYSCR1<SYSCK, GEAR2:0> = 0000).

4.1 Maximum Ratings (TMP93CS20F)

Parameter	Symbol	Rating	Unit
Power supply voltage	V_{CC}	−0.5 to 6.5	V
Input voltage	V_{IN}	−0.5 to V_{CC} + 0.5	
Output current (Per one pin), large current port	I_{OL1}	20	mA
Output current (Per one pin)	I_{OL2}	2	
Output current (Total of large current port)	ΣI_{OL1}	80	
Output current (Total)	ΣI_{OL}	120	
Output current (Total)	ΣI_{OH}	−80	
Power dissipation (Ta = 85°C)	P_D	600	mW
Soldering temperature (10 s)	T_{SOLDER}	260	°C
Storage temperature	T_{STG}	−65 to 150	
Operating temperature	T_{OPR}	−40 to 85	

Note: The maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any 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 which include this device, ensure that no maximum rating value will ever be exceeded.

4.2 DC Characteristics (1/2)

Ta = −40 to 85°C

Parameter		Symbol	Condition		Min	Typ. (Note)	Max	Unit
Power supply voltage (AV_{CC} = V_{CC}, AV_{SS} = V_{SS} = 0 V)		V_{CC}	fc = 4 to 20 MHz	fs = 30 to 34 kHz	4.5		5.5	V
			fc = 4 to 12.5 MHz		2.7			
Input low voltage	AD0 to AD15	V_{IL}	$V_{CC} \geq 4.5$ V		−0.3		0.8	V
		V_{IL1}	$V_{CC} < 4.5$ V				0.6	
	Port	V_{IL1}	V_{CC} = 2.7 to 5.5 V				0.3 V_{CC}	
	KEY0 to KEY7, $\overline{NMI}$, INT0 to INT4	V_{IL2}					0.25 V_{CC}	
	$\overline{EA}$	V_{IL3}					0.3	
	X1	V_{IL4}					0.2 V_{CC}	
	$\overline{RESET}$	V_{IL5}					0.1 V_{CC}	
Input high voltage	AD0 to AD15	V_{IH}	$V_{CC} \geq 4.5$ V		2.2		V_{CC} + 0.3	
		V_{IH}	$V_{CC} \geq 4.5$ V		2.0			
	Port	V_{IH1}	V_{CC} = 2.7 to 5.5 V		0.7 V_{CC}			
	KEY0 to KEY7, $\overline{NMI}$, INT0 to INT4	V_{IH2}			0.75 V_{CC}			
	$\overline{EA}$	V_{IH3}			V_{CC} − 0.3			
	X1	V_{IH4}			0.8 V_{CC}			
	$\overline{RESET}$	V_{IH5}			0.6 V_{CC}			

Note: Typical values are for Ta = 25°C and V_{CC} = 5 V unless otherwise noted.

4.2 DC Characteristics (2/2)

Parameter	Symbol	Condition		Min	Typ.(Note 1)	Max	Unit
Output low voltage	V_{OL}	I_{OL} = 1.6 mA (V_{CC} = 2.7 to 5.5 V)				0.45	V
Output low current (PA0 to PA7)	I_{OLA}	V_{OL} = 1.0 V	(V_{CC} = 5 V ± 10%)	16			mA
			(V_{CC} = 3 V ± 10%)	7			
Output high voltage	V_{OH1}	I_{OH} = −400 μA (V_{CC} = 3 V ± 10%)		2.4			V
	V_{OH2}	I_{OH} = −400 μA (V_{CC} = 5 V ± 10%)		4.2			
Darlington drive current (8 output pins max)	I_{DAR} (Note 2)	V_{EXT} = 1.5 V, R_{EXT} = 1.1 kΩ (V_{CC} = 5 V ± 10% only)		−1.0		−3.5	mA
Input leakage current	I_{LI}	0.0 ≤ V_{IN} ≤ V_{CC}			0.02	±5	μA
Output leakage current	I_{LO}	0.2 ≤ V_{IN} ≤ V_{CC} − 0.2			0.05	±10	
Power down voltage (at Stop, RAM backup)	V_{STOP}	V_{IL2} = 0.2 V_{CC}, V_{IH2} = 0.8 V_{CC}		2.0		6.0	V
Pin capacitance	C_{IO}	fc = 1 MHz				10	pF
Schmitt width KEYx, $\overline{NMI}$, INT0 to INT4, $\overline{RESET}$	V_{TH}			0.4	1.0		V
Programmable pull-up resistance	R_{KH}	V_{CC} = 5 V ± 10%		50		150	kΩ
		V_{CC} = 3 V ± 10%		100		300	
NORMAL	I_{CC}	V_{CC} = 5 V ± 10% fc = 20 MHz			25	28	mA
RUN					20	25	
IDLE2					14	17	
IDLE1					3.5	5	
NORMAL		V_{CC} = 3 V ± 10% fc = 12.5 MHz (Typ. Vcc = 3.0 V)			11	14	mA
RUN					9	12	
IDLE2					6	7.5	
IDLE1					1.5	2.0	
SLOW		V_{CC} = 3 V ± 10% fs = 32.768 kHz (Typ. V_{CC} = 3.0 V) at boosting frequency = 1 kHz			28	37	μA
RUN					21	27	
IDLE2					14	19	
IDLE1					7	9	
STOP		Ta ≤ 50°C			0.2	10	μA
		Ta ≤ 70°C				20	
		Ta ≤ 85°C				50	

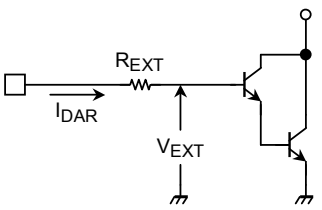
Note 1: Typical values are for Ta = 25°C and V_{CC} = 5 V unless otherwise noted.

Note 2: I_{DAR} is guaranteed for up to eight ports.

Note 3: Segment or common output is not loaded.

Note 4: I_{CC} measurement conditions (NORMAL, SLOW). Only CPU is operational; output pins are open and input pins are fixed.

e.g., Diagram of I_{DAR}



4.3 AC Electrical Characteristics

(1) $V_{CC} = 5\ V \pm 10\%$

No.	Parameter	Symbol	Variable		16 MHz		20 MHz		Unit
			Min	Max	Min	Max	Min	Max	
1	Osc. period (= x)	t_{OSC}	50	31250	62.5		50		ns
2	CLK width	t_{CLK}	2x – 40		85		60		ns
3	A0 to A23 valid → CLK hold	t_{AK}	0.5x – 20		11		5		ns
4	CLK valid → A0 to A23 hold	t_{KA}	1.5x – 70		24		5		ns
5	A0 to A15 valid → ALE fall	t_{AL}	0.5x – 15		16		10		ns
6	ALE fall → A0 to A15 hold	t_{LA}	0.5x – 20		11		5		ns
7	ALE high width	t_{LL}	x – 40		23		10		ns
8	ALE fall → $\overline{RD}$ / $\overline{WR}$ fall	t_{LC}	0.5x – 25		6		0		ns
9	$\overline{RD}$ / $\overline{WR}$ rise → ALE rise	t_{CL}	0.5x – 20		11		5		ns
10	A0 to A15 valid → $\overline{RD}$ / $\overline{WR}$ fall	t_{ACL}	x – 25		38		25		ns
11	A0 to A23 valid → $\overline{RD}$ / $\overline{WR}$ fall	t_{ACH}	1.5x – 50		44		25		ns
12	$\overline{RD}$ / $\overline{WR}$ rise → A0 to A23 hold	t_{CA}	0.5x – 25		6		0		ns
13	A0 to A15 valid → D0 to D15 input	t_{ADL}		3.0x – 55		133		95	ns
14	A0 to A23 valid → D0 to D15 input	t_{ADH}		3.5x – 65		154		110	ns
15	$\overline{RD}$ fall → D0 to D15 input	t_{RD}		2.0x – 60		65		40	ns
16	$\overline{RD}$ low pulse width	t_{RR}	2.0x – 40		85		60		ns
17	$\overline{RD}$ rise → D0 to D15 hold	t_{HR}	0		0		0		ns
18	$\overline{RD}$ rise → A0 to A15 output	t_{RAE}	x – 15		48		35		ns
19	$\overline{WR}$ low pulse width	t_{WW}	2.0x – 40		85		60		ns
20	D0 to D15 valid → $\overline{WR}$ rise	t_{DW}	2.0x – 55		70		45		ns
21	$\overline{WR}$ rise → D0 to D15 hold	t_{WD}	0.5x – 15		16		10		ns
22	A0 to A23 valid → $\overline{WAIT}$ input ($^{(1+N)}_{WAIT\ mode}$)	t_{AWH}		3.5x – 90		129		85	ns
23	A0 to A15 valid → $\overline{WAIT}$ input ($^{(1+N)}_{WAIT\ mode}$)	t_{AWL}		3.0x – 80		108		70	ns
24	$\overline{RD}$ / $\overline{WR}$ fall → $\overline{WAIT}$ hold ($^{(1+N)}_{WAIT\ mode}$)	t_{CW}	2.0x + 0		125		100		ns
25	A0 to A23 valid → Port input	t_{APH}		2.5x – 120		36		5	ns
26	A0 to A23 valid → Port hold	t_{APH2}	2.5x + 50		206		175		ns
27	$\overline{WR}$ rise → Port valid	t_{CP}		200		200		200	ns

AC measuring conditions

- Output level: High 2.2 V/Low 0.8 V, CL = 50 pF
  (However, CL = 100 pF for AD0 to AD15, A0 to A23, ALE, $\overline{RD}$, $\overline{WR}$, $\overline{HWR}$, CLK)
- Input level: High 2.4 V/Low 0.45 V (AD0 to AD15)
  High 0.8 × V_{CC}/Low 0.2 × V_{CC} (except for AD0 to AD15)

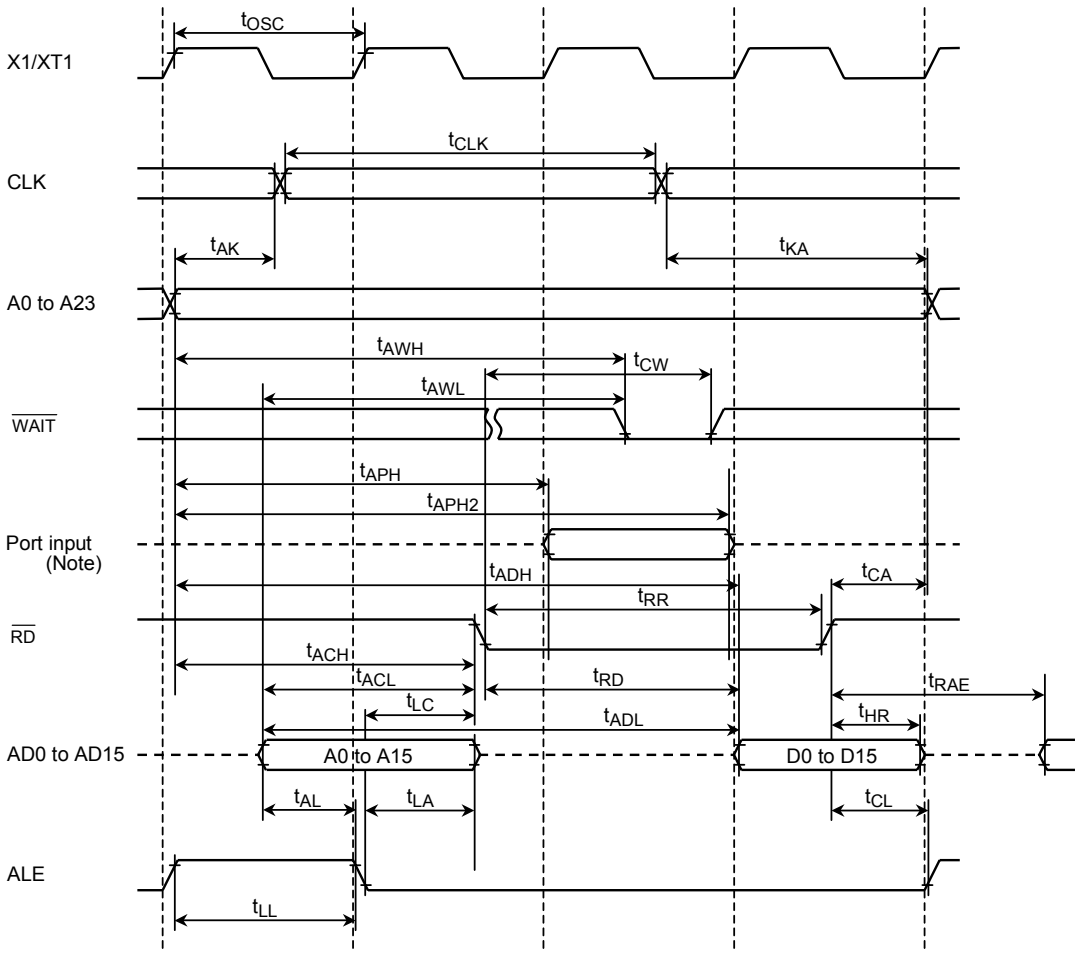
(2) V_{CC} = 3 V ± 10%

No.	Parameter	Symbol	Variable		12.5 MHz		Unit
			Min	Max	Min	Max	
1	Osc. period (= x)	t_{OSC}	80	31250	80		ns
2	CLK width	t_{CLK}	2x – 40		120		ns
3	A0 to A23 valid → CLK hold	t_{AK}	0.5x – 30		10		ns
4	CLK valid → A0 to A23 hold	t_{KA}	1.5x – 80		40		ns
5	A0 to A15 valid → ALE fall	t_{AL}	0.5x – 35		5		ns
6	ALE fall → A0 to A15 hold	t_{LA}	0.5x – 35		5		ns
7	ALE high width	t_{LL}	x – 60		20		ns
8	ALE fall → $\overline{RD}$ / $\overline{WR}$ fall	t_{LC}	0.5x – 35		5		ns
9	$\overline{RD}$ / $\overline{WR}$ rise → ALE Rise	t_{CL}	0.5x – 40		0		ns
10	A0 to A15 valid → $\overline{RD}$ / $\overline{WR}$ fall	t_{ACL}	x – 50		30		ns
11	A0 to A23 valid → $\overline{RD}$ / $\overline{WR}$ fall	t_{ACH}	1.5x – 50		70		ns
12	$\overline{RD}$ / $\overline{WR}$ rise → A0 to A23 hold	t_{CA}	0.5x – 40		0		ns
13	A0 to A15 valid → D0 to D15 input	t_{ADL}		3.0x – 110		130	ns
14	A0 to A23 valid → D0 to D15 input	t_{ADH}		3.5x – 125		155	ns
15	$\overline{RD}$ fall → D0 to D15 input	t_{RD}		2.0x – 115		45	ns
16	$\overline{RD}$ low pulse width	t_{RR}	2.0x – 40		120		ns
17	$\overline{RD}$ rise → D0 to D15 hold	t_{HR}	0		0		ns
18	$\overline{RD}$ rise → A0 to A15 output	t_{RAE}	x – 25		55		ns
19	$\overline{WR}$ low pulse width	t_{WW}	2.0x – 40		120		ns
20	D0 to D15 valid → $\overline{WR}$ rise	t_{DW}	2.0x – 120		40		ns
21	$\overline{WR}$ rise → D0 to D15 hold	t_{WD}	0.5x – 40		0		ns
22	A0 to A23 valid → $\overline{WAIT}$ input (1 + N) WAIT mode	t_{AWH}		3.5x – 130		150	ns
23	A0 to A15 valid → $\overline{WAIT}$ input (1 + N) WAIT mode	t_{AWL}		3.0x – 100		140	ns
24	$\overline{RD}$ / $\overline{WR}$ fall → $\overline{WAIT}$ hold (1 + N) WAIT mode	t_{CW}	2.0x + 0		160		ns
25	A0 to A23 valid → Port input	t_{APH}		2.5x – 120		80	ns
26	A0 to A23 valid → Port hold	t_{APH2}	2.5x + 50		250		ns
27	$\overline{WR}$ rise → Port valid	t_{CP}		200		200	ns

AC measuring conditions

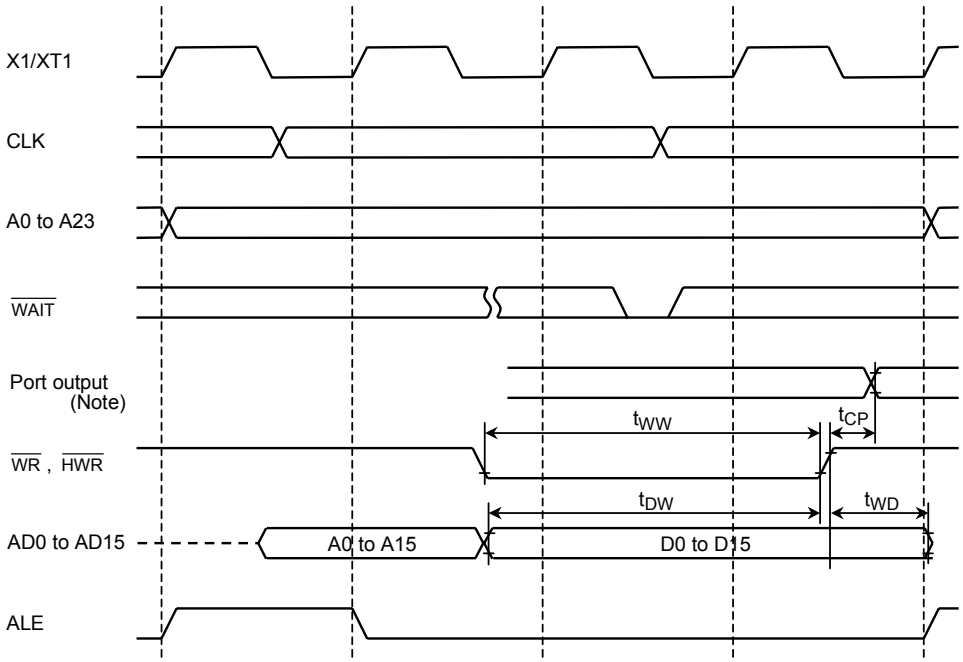
- Output level: High 0.7 × V_{CC}/Low 0.3 × V_{CC}, CL = 50 pF
- Input level: High 0.9 × V_{CC}/Low 0.1 × V_{CC}

(1) 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.

(2) 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 Serial Channel Timing

(1) I/O interface mode

a. SCLK input mode

Parameter	Symbol	Variable		32.768 MHz (Note)		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
SCLK cycle	t_{SCY}	16x		488 μs		1.28		0.8		μs
Output data → Rising edge or falling edge* of SCLK	t_{OSS}	$t_{SCY}/2-5x-50$		91.5 μs		190		100		ns
SCLK rising edge or falling edge* → Output data hold	t_{OHS}	5x − 100		152 μs		300		150		ns
SCLK rising edge or falling edge* → Input data hold	t_{HSR}	0		0		0		0		ns
SCLK rising edge or falling edge* → Effective data input	t_{SRD}		$t_{SCY}-5x-100$		336 μs		780		450	ns

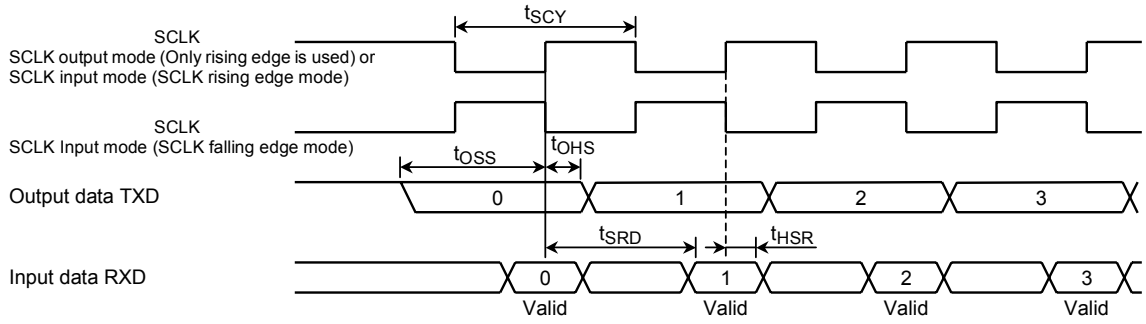
Note: System clock is fs, or input clock to prescaler is divisor clock of fs.

*) The rising edge is used in SCLK rising mode.
The falling edge is used SCLK falling mode.

b. SCLK output mode

Parameter	Symbol	Variable		32.768 MHz (Note)		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
SCLK cycle (Programmable)	t_{SCY}	16X	8192X	488 μs	250 ms	1.28	655.36	0.8	409.6	μs
Output data → SCLK rising edge	t_{OSS}	$t_{SCY}-2X-150$		427 μs		970		550		ns
SCLK rising edge → Output data hold	t_{OHS}	2X − 80		60 μs		80		20		ns
SCLK rising edge → Input data hold	t_{HSR}	0		0		0		0		ns
SCLK rising edge → Effective data input	t_{SRD}		$t_{SCY}-2X-150$		428 μs		970		550	ns

Note: System clock is fs, or input clock to prescaler is divisor clock of fs.



(2) UART mode (SCLK0, SCLK1 external input)

Parameter	Symbol	Variable		32.768 kHz (Note)		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	Min	Max	
SCLK cycle	t_{SCY}	4x + 20		122 μs		340		220		ns
SCLK low level pulse width	t_{SCYL}	2x + 5		6 μs		165		105		ns
SCLK high level pulse width	t_{SCYH}	2x + 5		6 μs		165		105		ns

Note: System clock is fs, or input clock to prescaler is divisor clock of fs.

4.5 AD Conversion Characteristics

(V_{SS} = 0 V, AV_{CC} = V_{CC}, AV_{SS} = V_{SS}, Ta = −40 to 85°C)

$AVcc = Vcc$, $AV_{SS} = V_{SS}$

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage (+)	V_{REFH}	V_{CC} = 5 V ± 10 %	V_{CC} − 1.5 V	V_{CC}	V_{CC}	V
		V_{CC} = 3 V ± 10%	V_{CC} − 0.2 V	V_{CC}	V_{CC}	
Analog reference voltage (−)	V_{REFL}	V_{CC} = 5 V ± 10%	V_{SS}			
		V_{CC} = 3 V ± 10%	V_{SS}			
Analog input voltage range	V_{AIN}		V_{REFL}		V_{REFH}	
Analog current for analog reference voltage	I_{REF} (V_{REFL} = 0 V)	V_{CC} = 5 V ± 10%		1.6	2.0	mA
<VREFON> = 1		V_{CC} = 3 V ± 10%		1.0	1.5	
<VREFON> = 0		V_{CC} = 2.7 to 5.5 V		0.02	5.0	μA
Error (Not including quantizing errors)	−	V_{CC} = 5 V ± 10%		±1.0	±3.0	LSB
		V_{CC} = 3 V ± 10%		±1.0	±5.0	

Note 1: 1LSB = $(V_{REFH} - V_{REFL})/2^{10}$ [V]

Note 2: Minimum operation frequency

The operation of the AD converter is guaranteed only when fc (High-frequency oscillator) is used. (It is not guaranteed when fs is used.) Additionally, it is guaranteed when the clock frequency which is selected by the clock gear is 4 MHz or more.

Note 3: The value I_{CC} includes the current which flows through the AV_{CC} pin.

4.6 LCD Driver Characteristics

Charge and Pump Characteristics		Symbol	Min	Typ.	Max	Unit
Reference input voltage		V_{L1}	0.9		1.83	V
Output voltage	V2 pin	V_{L2}		2 × V_{L1}		
	V3 pin	V_{L3}		3 × V_{L1}		
External capacity	C0, C1	C_{PMP}	0.1		1.0	μF
	V1 pin	C_{VL1}	0.1		1.0	
	V2 pin	C_{VL2}	0.1		1.0	
	V3 pin	C_{VL3}	0.1		1.0	

Note: Output voltage and external capacity are not loaded.

4.7 Event Counter (TI0, TI2, TI4, TI6, TI8 to TIB)

Parameter	Symbol	Variable		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock cycle	t_{VCK}	8X + 100		740		500		ns
Low level clock pulse width	t_{VCKL}	4X + 40		360		240		ns
High level clock pulse width	t_{VCKH}	4X + 40		360		240		ns

4.8 Interrupt and Capture

(1) $\overline{NMI}$, INT0 to INT4 interrupts, INTKEY interrupt

Parameter	Symbol	Variable		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low level pulse width	t_{INTAL}	4X		320		200		ns
High level pulse width	t_{INTAH}	4X		320		200		ns

(2) INT7 to INTB interrupts, capture

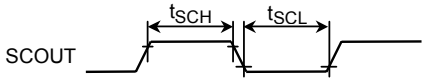
Parameter	Symbol	Variable		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low level pulse width	t_{INTBL}	4X+100		420		300		ns
High level pulse width	t_{INTBH}	4X+100		420		300		ns

4.9 SCOUT Pin AC Characteristics

Parameter	Symbol	Variable		12.5 MHz		20 MHz		Unit
		Min	Max	Min	Max	Min	Max	
High level pulse width V_{CC} = 5 V ± 10%	t_{SCH}	0.5X – 10		30		15		ns
High level pulse width V_{CC} = 3 V ± 10%		0.5X – 20		20		–	–	
Low level pulse width V_{CC} = 5 V ± 10%	t_{SCL}	0.5X – 10		30		15		ns
Low level pulse width V_{CC} = 3 V ± 10%		0.5X – 20		20		–	–	

Measurement condition

- Output level: High 2.2 V/Low 0.8 V, CL = 10 pF

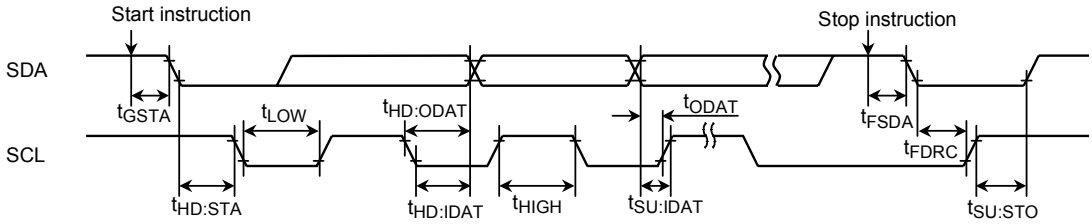


4.10 Timing Chart for Serial Bus Interface

(1) I²C bus mode

Parameter	Symbol	Variable			Unit
		Min	Typ.	Max	
START instruction → SDA falling edge	t_{GSTA}	3X			s
Start condition hold time	$t_{HD:STA}$	2^nX			s
SCL low level pulse width	t_{LOW}	2^nX			s
SCL high level pulse width	t_{HIGH}	$2^nX + 12X$			s
Data hold time (Input)	$t_{HD:IDAT}$	0			ns
Data setup time (Input)	$t_{SU:IDAT}$	250			ns
Data hold time (Output)	$t_{HD:ODAT}$	7X		11X	s
Data valid → SCL rising edge	t_{ODAT}		$2^nX - t_{HD:ODAT}$		s
STOP instruction → SDA falling edge	t_{FSDA}	3X			s
SDA falling edge → SCL rising edge	t_{FDRC}	2^nX			s
Stop condition hold time	$t_{SU:STO}$	$2^nX + 16X$			s

SBICR1<SCK2:0> sets n.



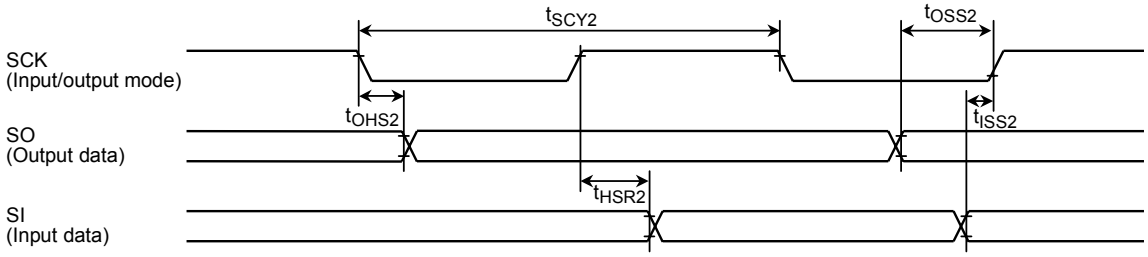
(2) Clock-synchronous 8-bit SIO mode (Serial bus interface)

a. SCK input mode

Parameter	Symbol	Variable		Unit
		Min	Max	
SCK cycle	t_{SCY2}	2^5X		s
SCK falling edge → Output data hold	t_{OHS2}	6X		s
Output data valid → SCK rising edge	t_{OSS2}	$t_{SCY2} - 6X$		s
SCK rising edge → Input data hold	t_{HSR2}	6X		ns
Input data valid → SCK rising edge	t_{ISS2}	0		ns

b. SCK output mode

Parameter	Symbol	Variable		Unit
		Min	Max	
SCK cycle	t_{SCY2}	2^5X	$2^{11}X$	s
SCK falling edge → Output data hold	t_{OHS2}	2X		s
Output data valid → SCK rising edge	t_{OSS2}	$t_{SCY2} - 2X$		s
SCK rising edge → Input data hold	t_{HSR2}	2X		s
Input data valid → SCK rising edge	t_{ISS2}	0		ns



5. Table of Special Function Registers (SFRs)

The special function registers (SFRs) include the I/O ports and peripheral control registers allocated to the 160 bytes whose addresses run from 000000H to 00009FH.

(1) I/O ports

(2) I/O port control

(3) Clock control

(4) Interrupt control

(5) Wait control

(6) Timer control

(7) Serial channel control

(8) Serial bus interface control

(9) Watchdog timer control

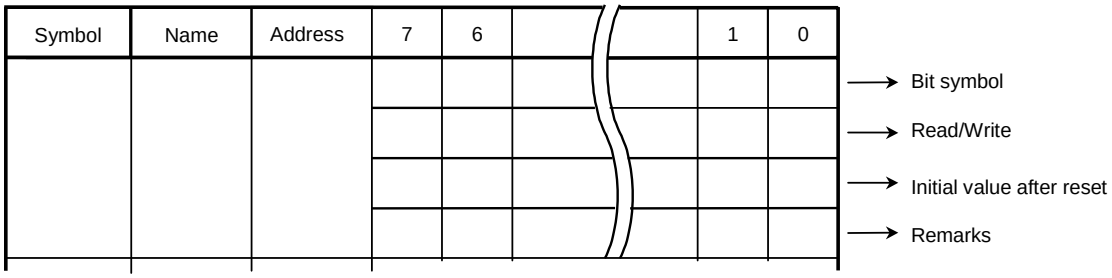
(10) AD converter control

(11) Timer for real time clock control

(12) LCD driver control

(13) Key-on wakeup control

Configuration of the table



Note: "Prohibit RMW" in the table means that you cannot use RMW instructions on these registers.

(Example) When setting only bit0 of register P0CR, "SET 0, (0002H)" cannot be used. The LD (transfer) instruction must be used to write all 8 bits.

Table 5 I/O Register Address Map

Address	Name	Address	Name	Address	Name	Address	Name	Address	Name
000000H	P0	20H	TRUN	40H	TREGAL/CAPAL	60H	ADREG04L	80H	DBR00
1H	P1	21H	P8ODE	41H	TREGAH/CAPAH	61H	ADREG04H	81H	DBR01
2H	P0CR	22H	TREG0	42H	TREGBL/CAPBL	62H	ADREG15L	82H	DBR02
3H	(Reserved)	23H	TREG1	43H	TREGBH/CAPBH	63H	ADREG15H	83H	DBR03
4H	P1CR	24H	T10MOD	44H	INTET76	64H	ADREG26L	84H	DBR04
5H	P1FC	25H	TFFCR	45H	INTET98	65H	ADREG26H	85H	DBR05
6H	P2	26H	TREG2	46H	INTETBA	66H	ADREG37L	86H	DBR06
7H	P3	27H	TREG3	47H	INTEO64	67H	ADREG37H	87H	DBR07
8H	P2CR	28H	T32MOD	48H	TAMOD	68H	WAITC0	88H	DBR08
9H	P2FC	29H	TRDC	49H	TAFFCR	69H	WAITC1	89H	DBR09
AH	P3CR	2AH	TREG4L/CAP4L	4AH	KEYCR	6AH	WAITC2	8AH	DBR0A
BH	P3FC	2BH	TREG4H/CAP4H	4BH	SBICR1	6BH	BRADD0	8BH	DBR0B
CH	P4	2CH	TREG5L	4CH	SBIDBR	6CH	BRADD1	8CH	DBR0C
DH	P5	2DH	TREG5H	4DH	I2CAR	6DH	CKOCR	8DH	DBR0D
EH	P4CR	2EH	T4MOD	4EH	SBICR2	6EH	SYSCR0	8EH	DBR0E
FH	(Reserved)	2FH	T4FFCR	4FH	SBICR3	6FH	SYSCR1	8FH	DBR0F
10H	P4FC	30H	TREG8L/CAP8L	50H	SC0BUF	70H	INTE0AD	90H	DBR10
11H	(Reserved)	31H	TREG8H/CAP8H	51H	SC0CR	71H	INTE21	91H	DBR11
12H	P6	32H	TREG9L/CAP9L	52H	SC0MOD	72H	INTE43	92H	DBR12
13H	P7	33H	TREG9H/CAP9H	53H	BR0CR	73H	INTE98	93H	DBR13
14H	P6CR	34H	(Reserved)	54H	SC1BUF	74H	INTEBA	94H	LCDCR
15H	P7CR	35H		55H	SC1CR	75H	INTET10	95H	CHPCR
16H	P6FC	36H	T6MOD	56H	SC1MOD	76H	INTET32	96H	(Reserved)
17H	P7FC	37H	T6FFCR	57H	BR1CR	77H	INTES0	97H	
18H	P8	38H	T8MOD	58H	P6ODE	78H	INTES1	98H	
19H	P9	39H	T8FFCR	59H	RTCCR	79H	INTE7S2	99H	
1AH	P8CR	3AH	T16CR	5AH	INTEOA8	7AH	INTET54	9AH	
1BH	(Reserved)	3BH	TREG6L/CAP6L	5BH	INTEKR	7BH	IIMC	9BH	
1CH	P8FC	3CH	TREG6H/CAP6H	5CH	WDMOD	7CH	DMA0V	9CH	
1DH	P9FC	3DH	TREG7L	5DH	WDCR	7DH	DMA1V	9DH	
1EH	PA	3EH	TREG7H	5EH	ADMOD0	7EH	DMA2V	9EH	
1FH	PAFC	3FH	(Reserved)	5FH	ADMOD1	7FH	DMA3V	9FH	

Note: Do not access addresses which do not have register names 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							
			Undefined							
			Input mode							
P1	Port 1	01H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			0	0	0	0	0	0	0	0
			Input mode							
P2	Port 2	06H (Prohibit RMW*)	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			1	1	1	1	1	1	1	1
			Input mode							
P3	Port 3	07H (Prohibit RMW*)	P37	P36	P35	P34	P33	P32	P31	P30 (Note 1)
			R/W							
			1	1	1	1	1	1	1	1
			Input mode						Output mode	
P4	Port 4	0CH (Prohibit RMW*)	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			1	1	1	1	1	1	1	1
			Input mode							
P5	Port 5	0DH	P57	P56	P55	P54	P53	P52	P51	P50
			R							
			Input mode							
P6	Port 6	12H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			1	1	1	1	1	1	1	1
			Input mode							
P7	Port 7	13H (Prohibit RMW*)	P77	P76	P75	P74	P73	P72	P71	P70
			R/W							
			1	1	1	1	1	1	1	1
			Input mode							
P8	Port 8	18H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			1	1	1	1	1	1	1	1
			Output mode		Input mode					
P9	Port 9	19H	P97	P96	P95	P94	P93	P92	P91	P90
			R/W							
			0	0	0	0	0	0	0	0
			Output mode							
PA	Port A	1EH	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R/W							
			1	1	1	1	1	1	1	1
			Output mode							

Note 1: When P30 pin is defined as $\overline{RD}$ signal output mode (P30F = 1), clearing the output latch register P30 to 0 outputs the $\overline{RD}$ strobe from P30 pin for PSRAM, even when the internal address is accessed. If the output latch register P30 remains 1, the $\overline{RD}$ strobe is output only when the external address is accessed.

Note 2: Port 86, 87 are also used as XT1, XT2. Therefore these pins are open drain output type.

Read/Write

R/W: Either read or write is possible

R: Only read is possible

W: Only write is possible

Prohibit RMW: Prohibit read-modify-write.
(Prohibit RES/SET/TSET/CHG/STCF/ANDCF/ORCF/XORCF instruction)

Prohibit RMW*:Read-modify-write is prohibited when controlling the PU resistors.

(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 external access, set as AD7 to AD0 and 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 the 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, 01: Output, 10: AD15 to AD8, 11: A15 to A8							
P2CR	Port 2 control	08H (Prohibit RMW)	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			<<Refer to the P2FC>>							
P2FC	Port 2 function	09H (Prohibit RMW)	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
			W							
			0	0	0	0	0	0	0	0
			P2FC/P2CR = 00: Input, 01: Output, 10: A7 to A0, 11: A23 to A16							
P3CR	Port 3 control	0AH (Prohibit RMW)	P37C	P36C	P35C	P34C	P33C	P32C		
			W							
			0	0	0	0	0	0		
			0: Input 1: Output							
P3FC	Port 3 function	0BH (Prohibit RMW)	I4IE	I3IE	I2IE	I1IE	I0IE	P32F	P31F	P30F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: INT4	0: Port 1: INT3	0: Port 1: INT2	0: Port 1: INT1	0: Port 1: INT0	0: Port 1: $\overline{HWR}$	0: Port 1: $\overline{WR}$	0: Port 1: $\overline{RD}$
P4CR	Port 4 control	0EH (Prohibit RMW)	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P4FC	Port 4 function	10H (Prohibit RMW)					P43F		P41F	
							W		W	
							0		0	
							0: Port 1: TO6		0: Port 1: TO4	

I/O port control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P6CR	Port 6 control	14H (Prohibit RMW)	P67C	P66C	P65C	P64C	P63C	P62C	P61	P60C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P7CR	Port 7 control	15H (Prohibit RMW)	P77C	P76C	P75C	P74C	P73C	P72C	P71C	P70C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P6FC	Port 8 function	16H (Prohibit RMW)	P67F	I7IE	P65F		P63F	P62F	P61F	P60F
			W				W			
			0	0	0		0	0	0	0
			0: Port 1: TO1	0: Port 1: INT7	0: Port 1: SCLK0		0: Port 1: TXD0	0: Port 1: SCL/SI	0: Port 1: SDA/SO	0: Port 1: SCK
P7FC	Port 7 function	17H (Prohibit RMW)	NMIC	P76F				P72W		
			R/W	W				W		
			0	0				0		
			0: Port 1: $\overline{\text{NMI}}$ input	0: Port 1: TOA				0: Port 1: TO8		
P8CR	Port 8 control	1AH (Prohibit RMW)	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
			W							
			1	1	0	0	0	0	0	0
			0: Input 1: Output							
P8FC	Port 8 function	1CH (Prohibit RMW)					P83F			P80F
							W			W
							0			0
							0: Port 1: TO3			0: Port 1: TXD1
P9FC	Port 9 function	1DH (Prohibit RMW)	P97F	P96F	P95F	P94F	P93F	P92F	P91F	P90F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: SEG31	0: Port 1: SEG30	0: Port 1: SEG29	0: Port 1: SEG28	0: Port 1: SEG27	0: Port 1: SEG26	0: Port 1: SEG25	0: Port 1: SEG24
PAFC	Port A function	1FH (Prohibit RMW)	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
			W							
			0	0	0	0	0	0	0	0
			0: Port 1: SEG39	0: Port 1: SEG38	0: Port 1: SEG37	0: Port 1: SEG36	0: Port 1: SEG35	0: Port 1: SEG34	0: Port 1: SEG33	0: Port 1: SEG32
P8ODE	Port 8 open-drain enable	21H			ODE85	ODE84	ODE83	ODE82	ODE81	ODE80
					R/W					
					0	0	0	0	0	0
					1: P85 Open drain	1: P84 Open drain	1: P83 Open drain	1: P82 Open drain	1: P81 Open drain	1: P80 Open drain
P6ODE	Port 6 open-drain enable	58H					ODE63	ODE62	ODE61	
							R/W			
							0	0	0	
							1: P63 Open drain	1: P62 Open drain	1: P61 Open drain	

(3) Clock control

Symbol	Name	Address	7	6	5	4	3	2	1	0
CKOCR	Clock output control register	006DH					SCOSEL	SCOEN	ALEEN	CLKEN
							R/W			
							0	0	0	0
							SCOUT 0: f_{FPH} 1: f_{SYS}	SCOUT output control 0: I/O port 1: SCOUT	ALE pin control 0: High-Z output 1: ALE output	CLK pin control 0: High-Z output 1: CLK output
SYSCR0	System clock control register 0	006EH	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	PRCK1	PRCK0
			R/W							
			1	0	1	0	0	0	0	0
			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	Selected clock after release of STOP mode 0: fc 1: fs	Warm-up timer (Write) 0: Don't care 1: Start timer (Read) 0: End warm up 1: Continue warm up	Select prescaler clock 00: f_{FPH} 01: fs 10: fc/16 11: (Reserved)	
SYSCR1	System clock control register 1	006FH					SYSCK	GEAR2	GEAR1	GEAR0
							R/W			
							0	1	0	0
							Select system clock 0: fc 1: fs (Note)	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)		

Note: The high frequency oscillator will be enabled when SYSCR1<SYSCK> is set to 0 regardless of the value of SYSCR0<XEN>.
The low frequency oscillator will be enabled when SYSCR1<SYSCK> is set to 1 regardless of the value of SYSCR0<XTEN>.

(4) Interrupt control (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	INT 0/AD enable register	0070H (Prohibit RMW)	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE21	INT 2/1 enable register	0071H (Prohibit RMW)	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE43	INT 4/3 enable register	0072H (Prohibit RMW)	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE98	INT 9/8 enable register	0073H (Prohibit RMW)	INT9				INT8			
			I9C	I9M2	I9M1	I9M0	I8C	I8M2	I8M1	I8M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTEBA	INT B/A enable register	0074H (Prohibit RMW)	INTB				INTA			
			IBC	IBM2	IBM1	IBM0	IAC	IAM2	IAM1	IAM0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET10	INTT 1/0 enable register	0075H (Prohibit RMW)	INTT1 (TREG1)				INTT0 (TREG0)			
			IT1C	IT1M2	IT1M1	IT1M0	IT0C	IT0M2	IT0M1	IT0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET32	INTT 3/2 enable register	0076H (Prohibit RMW)	INTT3 (TREG3)				INTT2 (TREG2)			
			IT3C	IT3M2	IT3M1	IT3M0	IT2C	IT2M2	IT2M1	IT2M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTES0	INT TX0/RX0 enable register	0077H (Prohibit RMW)	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTES1	INT TX1/RX1 enable register	0078H (Prohibit RMW)	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE7S2	INT 7/S2 enable register	0079H (Prohibit RMW)	INT7				INTS2			
			I7C	I7M2	I7M1	I7M0	IS2C	IS2M2	IS2M1	IS2M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Prohibit interrupt request.
0	0	1	Set interrupt request level to 1.
0	1	0	Set interrupt request level to 2.
0	1	1	Set interrupt request level to 3.
1	0	0	Set interrupt request level to 4.
1	0	1	Set interrupt request level to 5.
1	1	0	Set interrupt request level to 6.
1	1	1	Prohibit interrupt request.

IxxC	Function (Read)	Function (Write)
0	Indicate no interrupt request.	Clear interrupt request flag.
1	Indicate interrupt request.	- - - - - Don't care - - - - -

Interrupt control (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTET54	INTT 5/4 enable register	007AH (Prohibit RMW)	INTTR5 (TREG5)				INTTR4 (TREG4)			
			IT5C	IT5M2	IT5M1	IT5M0	IT4C	IT4M2	IT4M1	IT4M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET76	INTT 7/6 enable register	0044H (Prohibit RMW)	INTTR7 (TREG7)				INTTR6 (TREG6)			
			IT7C	IT7M2	IT7M1	IT7M0	IT6C	IT6M2	IT6M1	IT6M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET98	INTT 9/8 enable register	0045H (Prohibit RMW)	INTTR9 (TREG9)				INTTR8 (TREG8)			
			IT9C	IT9M2	IT9M1	IT9M0	IT8C	IT8M2	IT8M1	IT8M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTETBA	INTT B/A enable register	0046H (Prohibit RMW)	INTTRB (TREGB)				INTTRA (TREGA)			
			ITBC	ITBM2	ITBM1	ITBM0	ITAC	ITAM2	ITAM1	ITAM0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTEO64	INTO 6/4 enable register	0047H (Prohibit RMW)	INTTO6				INTTO4			
			ITO6C	ITO6M2	ITO6M1	ITO6M0	ITO4C	ITO4M2	ITO4M1	ITO4M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTEOA8	INTT OA/8 enable register	005AH (Prohibit RMW)	INTTOA				INTTO8			
			ITOAC	ITOAM2	ITOAM1	ITOAM0	ITO8C	ITO8M2	ITO8M1	ITO8M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTEKR	INT KEY/RTC enable register	005BH (Prohibit RMW)	INTKEY				INTRTC			
			IKEYC	IKEYM2	IKEYM1	IKEYM0	IRTCC	IRTCM2	IRTCM1	IRTCM0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Prohibit interrupt request.
0	0	1	Set interrupt request level to 1.
0	1	0	Set interrupt request level to 2.
0	1	1	Set interrupt request level to 3.
1	0	0	Set interrupt request level to 4.
1	0	1	Set interrupt request level to 5.
1	1	0	Set interrupt request level to 6.
1	1	1	Prohibit interrupt request.

IxxC	Function (Read)	Function (Write)
0	Indicate no interrupt request.	Clear interrupt request flag.
1	Indicate interrupt request.	- - - - - Don't care - - - - -

Interrupt control (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	Micro DMA 0 request vector	7CH (Prohibit RMW)			DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
					W					
					0	0	0	0	0	0
					Micro DMA0 start vector					
DMA1V	Micro DMA 1 request vector	7DH (Prohibit RMW)			DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
					W					
					0	0	0	0	0	0
					Micro DMA1 start vector					
DMA2V	Micro DMA 2 request vector	7EH (Prohibit RMW)			DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
					W					
					0	0	0	0	0	0
					Micro DMA2 start vector					
DMA3V	Micro DMA 3 request vector	7FH (Prohibit RMW)			DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
					W					
					0	0	0	0	0	0
					Micro DMA3 start vector					
IIMC	Interrupt input mode control	7BH (Prohibit RMW)	I7FE	I4FE	I3FE	I2FE	I1FE	I0FE	I0LE	NMIREE
			W							
			0	0	0	0	0	0	0	0
			0: INT7 rising edge 1: INT7 falling edge	0: INT4 rising edge 1: INT4 falling edge	0: INT3 rising edge 1: INT3 falling edge	0: INT2 rising edge 1: INT2 falling edge	0: INT1 rising edge 1: INT1 falling edge	0: INT0 rising edge 1: INT0 falling edge	0: INT0 edge mode 1: INT0 level mode	1: Operation even at $\overline{\text{NMI}}$ rising edge

(5) Wait controller

Symbol	Name	Address	7	6	5	4	3	2	1	0
WAITC0	Block 0 wait control register	68H (Prohibit RMW)				B0BUS	B0W1	B0W0	B0C1	B0C0
						W				
						0	0	0	0	0
						0: 16-bit bus 1: 8-bit bus	00: 2 waits 01: 1 wait 10: (1 + N) waits 11: 0 waits		00: 7F00H to 7FFFH 01: 400000H to 10: 800000H to 11: C00000H to	
WAITC1	Block 1 wait control register	69H (Prohibit RMW)				B1BUS	B1W1	B1W0	B1C1	B1C0
						W				
						0	0	0	0	0
						0: 16-bit bus 1: 8-bit bus	00: 2 waits 01: 1 wait 10: (1 + N) waits 11: 0 waits		00: 8A0H to 7FFFH 01: 400000H to 10: 800000H to 11: C00000H to	
WAITC2	Block 2 wait control register	6AH (Prohibit RMW)				B2BUS	B2W1	B2W0	B2C1	B2C0
						W				
						0	0	0	1	1
						0: 16-bit bus 1: 8-bit bus	00: 2 waits 01: 1 wait 10: (1 + N) waits 11: 0 waits		00: 8000H to 01: 400000H to 10: 800000H to 11: C00000H to	

(6) Timer control (1/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TRUN	Timer control	20H	PRRUN		T6RUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
			R/W		R/W					
			0		0	0	0	0	0	0
			Prescaler and timer run/stop control 0: Stop and clear 1: Run (Count up)							
TREG0	8-bit timer register 0	22H (Prohibit RMW)	–							
			W							
			Undefined							
TREG1	8-bit timer register 1	23H (Prohibit RMW)	–							
			W							
			Undefined							
T10MOD	8-bit timer 0,1 source CLK and mode	24H	T01M1	T01M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		00: – 01: 2^6 – 1PWM 10: 2^7 – 1 11: 2^8 – 1		00: TO0TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI0 Input 01: ϕT1 10: ϕT4 11: ϕT16	
TFFCR	8-bit timer flip flop control	25H	TFF3C1	TFF3C0	TFF3IE	TFF3IS	TFF1C1	TFF1C0	TFF1IE	TFF1IS
			W		R/W		W		R/W	
			1	1	0	0	1	1	0	0
			00: Invert TFF3 01: Set TFF3 10: Clear TFF3 11: Don't care		1: TFF3 invert enable	1: Timer 3 invert enable	00: Invert TFF1 01: Set TFF1 10: Clear TFF1 11: Don't care		1: TFF1 invert enable	1: Timer 1 invert enable
TREG2	8-bit timer register 2	26H (Prohibit RMW)	–							
			W							
			Undefined							
TREG3	8-bit timer register 3	27H (Prohibit RMW)	–							
			W							
			Undefined							
T32MOD	8-bit timer 2,3 source CLK and mode	28H	T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		00: – 01: 2^6 – 1 PWM 10: 2^7 – 1 11: 2^8 – 1		00: TO2TRG 01: ϕT1 10: ϕT16 11: ϕT256		00: TI2 input 01: ϕT1 10: ϕT4 11: ϕT16	
TRDC	Timer register double buffer control register	29H							TR2DE	TR0DE
									R/W	
									0	0
									0: Double buffer disable 1: Double buffer enable	0: Double buffer disable 1: Double buffer enable

Timer control (2/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREG4L	16-bit timer register 4 low	2AH (Prohibit RMW)	–							
			W							
			Undefined							
TREG4H	16-bit timer register 4 high	2BH (Prohibit RMW)	–							
			W							
			Undefined							
TREG5L	16-bit timer register 5 low	2CH (Prohibit RMW)	–							
			W							
			Undefined							
TREG5H	16-bit timer register 5 high	2DH (Prohibit RMW)	–							
			W							
			Undefined							
CAP4L	Capture register 4 low	2AH	–							
			R							
			Undefined							
CAP4H	Capture register 4 high	2BH	–							
			R							
			Undefined							
T4MOD	16-bit timer 4 source CLK and mode	2EH			CAP4IN	–	–	CLE	T4CLK1	T4CLK0
					W	R/W				
					1	0	0	0	0	0
					0: Soft-capture 1: Don't care	Write “0”.		1: UC4 clear enable	Source clock 00: TI4 input 01: ϕT1 10: ϕT4 11: ϕT16	
T4FFCR	16-bit timer 4 flip-flop control	2FH			–	–	EQ5T4	EQ4T4	TFF4C1	TFF4C0
					R/W	R/W	R/W		W	
					0	0	0	0	1	1
					Write “0”.		TFF4 invert trigger 0: Trigger disable 1: Trigger enable		TFF4 control 00: Invert TFF4 01: Set TFF4 10: Clear TFF4 11: Don’t care * Always read as 11.	
							Inverted when the UC value matches TREG5	Inverted when the UC value matches TREG4		

Timer control (3/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
T16CR	16-bit timer control	3AH	QCU		TARUN	T8RUN	DBAEN	DB8EN	DB6EN	DB4EN
			R/W		R/W					
			0		0	0	0	0	0	0
			Warm-up timer control		Prescaler and timer run/stop control 0: Stop and clear 1: Run (Count up)		Double buffer 0: Disable 1: Enable			
							Double buffer of TREGA	Double buffer of TREG8	Double buffer of TREG6	Double buffer of TREG4
TREG6L	16-bit timer register 6 low	3BH (Prohibit RMW)	–							
			W							
			Undefined							
TREG6H	16-bit timer register 6 high	3CH (Prohibit RMW)	–							
			W							
			Undefined							
TREG7L	16-bit timer register 7 low	3DH (Prohibit RMW)	–							
			W							
			Undefined							
TREG7H	16-bit timer register 7 high	3EH (Prohibit RMW)	–							
			W							
			Undefined							
CAP6L	Capture register 6 low	3BH	–							
			R							
			Undefined							
CAP6H	Capture register 6 high	3CH	–							
			R							
			Undefined							
T6MOD	16-bit timer 6 source CLK and mode	36H			CAP6IN	–	–	CLE	T6CLK1	T6CLK0
					W	R/W				
					1	0	0	0	0	0
					0: Soft-capture 1: Don't care	Write "0".		1: UC6 clear enable	Source clock 00: TI6 input 01: ϕT1 10: ϕT4 11: ϕT16	
T6FFCR	16-bit timer 6 flip-flop control	37H			–	–	EQ7T6	EQ6T6	TFF6C1	TFF6C0
					R/W				W	
					0	0	0	0	1	1
					Write "0".		TFF6 invert trigger 0: Trigger disable 1: Trigger enable		00: Invert TFF6 01: Set TFF6 10: Clear TFF6 11: Don't care	
							Inverted when the UC value matches TREG7	Inverted when the UC value matches TREG6		

Timer control (4/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREG8L	16-bit timer register 8 low	30H (Prohibit RMW)	–							
			W							
			Undefined							
TREG8H	16-bit timer register 8 high	31H (Prohibit RMW)	–							
			W							
			Undefined							
TREG9L	16-bit timer register 9 low	32H (Prohibit RMW)	–							
			W							
			Undefined							
TREG9H	16 -bit timer register 9 high	33H (Prohibit RMW)	–							
			W							
			Undefined							
CAP8L	Capture register 8 low	30H	–							
			R							
			Undefined							
CAP8H	Capture register 8 high	31H	–							
			R							
			Undefined							
CAP9L	Capture register 9 low	32H	–							
			R							
			Undefined							
CAP9H	Capture register 9 high	33H	–							
			R							
			Undefined							
T8MOD	16-bit timer 8 source CLK and mode	38H			CAP8IN	CAP89M1	CAP89M0	CLE	T8CLK1	T8CLK0
					W	R/W				
					1	0	0	0	0	0
					0: Soft-capture 1: Don't care	Capture timing 00: Disable 01: TI8 ↑ TI9 ↑ 10: TI8 ↑ TI8 ↓ 11: TFF1 ↑ TFF1 ↓		1: UC8 clear enable	Source clock 00: TI8 input 01: ϕT1 10: ϕT4 11: ϕT16	
T8FFCR	16-bit timer 8 flip-flop control	39H			CAP9T8	CAP8T8	EQ9T8	EQ8T8	TFF8C1	TFF8C0
					R/W				W	
					0	0	0	0	1	1
					TFF8 invert trigger 0: Trigger disable 1: Trigger enable				00: Invert TFF8 01: Set TFF8 10: Clear TFF8 11: Don't care	
					Inverted when the UC value is latched to CAP9	Inverted when the UC value is latched to CAP8	Inverted when the UC value matches TREG9	Inverted when the UC value matches TREG8		

Timer control (5/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREGAL	16-bit timer register A low	40H (Prohibit RMW)	–							
			W							
			Undefined							
TREGAH	16-bit timer register A high	41H (Prohibit RMW)	–							
			W							
			Undefined							
TREGBL	16-bit timer Register B low	42H (Prohibit RMW)	–							
			W							
			Undefined							
TREGBH	16-bit timer register B high	43H (Prohibit RMW)	–							
			W							
			Undefined							
CAPAL	Capture register A low	40H	–							
			R							
			Undefined							
CAPAH	Capture register A high	41H	–							
			R							
			Undefined							
CAPBL	Capture register B low	42H	–							
			R							
			Undefined							
CAPBH	Capture register B high	43H	–							
			R							
			Undefined							
TAMOD	16-bit timer A source CLK and mode	48H			CAPAIN	CAPABM1	CAPABM0	CLE	TACLK1	TACLK0
					W	R/W				
					1	0	0	0	0	0
					0: Soft-capture 1: Don't care	Capture timing 00: Disable 01: TIA ↑ TIB ↑ 10: TIA ↑ TIA ↓ 11: TFF1 ↑ TFF1 ↓		1: UC5 clear enable	Source clock 00: TIA input 01: ϕT1 10: ϕT4 11: ϕT16	
TAFFCR	16-bit timer A flip-flop control	49H			CAPBTA	CAPATA	EQBTA	EQATA	TFFAC1	TFFAC0
					R/W				W	
					0	0	0	0	1	1
					TFFA invert trigger 0: Trigger disable 1: Trigger enable				00: Invert TFFA 01: Set TFFA 10: Clear TFFA 11: Don't care	
					Inverted when the UC value is latched to CAPB	Inverted when the UC value is latched to CAPA	Inverted when the UC value matches TREGB	Inverted when the UC value matches TREGA		

(7) Serial channel control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial channel 0 buffer	50H (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	51H	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: Input SCLK0 pin
						Overrun	Parity	Framing		
SC0MOD	Serial channel 0 Mode	52H	TB8	CTSE0	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			Undefined	0	0	0	0	0	0	0
			Transmisson data bit8	1: CTS enable	1: Receive enable	1: Wake-up enable	00: I/O Interface 01: UART 7 bits 10: UART 8 bits 11: UART 9 bits		00: TO0 trigger 01: Baud rate generator 10: Internal clock ϕ1 11: External clock SCLK0	
BR0CR	Baud rate 0 control	53H	—	BR0ADD	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W	R/W						
			0	0	0	0	0	0	0	0
			Always fixed to "0".	1: + (16 − K) /16 divisor enable	00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Set frequency divisor 0 to F			
BRADD0	Baud rate control	68H					BR0K3	BR0K2	BR0K1	BR0K0
							R/W			
							0	0	0	0
							Set N + (16 − K)/16 divisor 1 to F (0 prohibited)			
SC1BUF	Serial channel 1 buffer	54H (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	55H	RB8	EVEN	PE	OERR	PERR	FERR	–	–
			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			Always fixed to "0".	
						Overrun	Parity	Framing		
SC1MOD	Serial channel 1 mode	56H	TB8	–	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			Undefined	0	0	0	0	0	0	0
			Transmisson data bit8	Always fixed to "0".	1: Receive enable	1: Wakeup enable	00: Reserved 01: UART 7 bits 10: UART 8 bits 11: UART 9 bits		00: TO0 trigger 01: Baud rate generator 10: Internal clock ϕ1 11: Don't care	

Serial channel control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
BR1CR	Baud rate 1 control	57H	–	BR1ADD	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0	0	0	0	0	0
			Always fixed to "0".	1: + (16 − K)/16 divisor enable	00: φT0 01: φT2 10: φT8 11: φT32		Set frequency divisor 0 to F			
BRADD1	Baud rate control	6CH					BR1K3	BR1K2	BR1K1	BR1K0
							R/W			
							0	0	0	0
							Set N + (16 − K)/16 divisor 1 to F ("0" prohibited)			

(8) Serial bus interface control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SBICR1	Serial bus interface control register 1	4BH (I^2C bus mode) (Prohibit RMW)	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0
			W			R/W		W		
			0	0	0	0		0	0	0
			Select number of bits to be transferred 000: 8 100: 4 001: 1 101: 5 010: 2 110: 6 011: 3 111: 7			Set acknowledge mode 0: Disable 1: Enable		Select serial clock frequency 000: 4 100: 8 001: 5 101: 9 010: 6 110: 10 011: 7 111: (Reserved)		
		4BH (SIO mode) (Prohibit RMW)	SIOS	SIOINH	SIONH	SIOM0		SCK2	SCK1	SCK0
			W					W		
			0	0	0	0		0	0	0
			Indicate transfer start/stop 0: Stop 1: Start	Continue/ abort transfer 0: Continue 1: Abort	Transfer mode select 00: 8-bit transfer 01: (Reserved) 10: 8-bit transmit/receive 11: 8-bit receive			Select serial clock frequency 000: $f_{FPH}/2^5$ 100: $f_{FPH}/2^9$ 001: $f_{FPH}/2^6$ 101: $f_{FPH}/2^{10}$ 010: $f_{FPH}/2^7$ 110: $f_{FPH}/2^{11}$ 011: $f_{FPH}/2^8$ 111: External clock (SCK input)		
SBICR2	Serial bus interface control register 2	4EH (I^2C bus mode) (Prohibit RMW)	MST	TRX	BB	PIN	SBIM1	SBIM0		
			W							
			0	0	0	1	0	0		
			Select slave/ master 0: Slave 1: Master	Select Transmit/ receive 0: Receiver 1: Transmitter	Generate start stop condition 0: Stop 1: Start	Release INT2 interrupt service request 0: Don't care 1: Release	Select serial bus interface mode 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)			
		4 EH (SIO mode) (Prohibit RMW)					SBIM1	SBIM0		
							W			
							0	0		
							Select serial bus interface mode 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)			

Serial bus interface control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SBISR	Serial bus interface status register	4EH (I^2C bus mode) (Prohibit RMW)	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
			R							
			0	0	0	1	0	0	0	0
			Monitor select status 0: Slave 1: Master	Monitor select status 0: Receiver 1: Transmitter	Monitor bus status 0: Bus free 1: Bus busy	Monitor INTS2 interrumt service request status 0: Requesting 1: Release	Monitor arbitration lost detect 1: Detect	Monitor slave address match detect 1: Detect	Monitor general call detect 1: Detect	Monitor last receive bit 0: 0 1: 1
		4EH (SIO mode) (Prohibit RMW)					SIOF	SEF		
							R			
							0	0		
							Monitor serial transferring status 0: End 1: Transferring	Monitor shift operation 0: End 1: Setting		
SBICR3	Serial bus interface status register 3	4FH								SWRST
										R/W
										0
										Initialize SBI internally 0: – 1: Initialization
SBIDBR	Serial bus interface data buffer register	4CH (Prohibit RMW)	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
			R (Receiving)/W (Transmission)							
			Undefined							
I2CAR	I^2C bus address register	4DH (Prohibit RMW)	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
			W							
			0	0	0	0	0	0	0	0
			Set slave address							Speify address recogntion mode 0: Acknowledges slave address 1: Not acknowledge slave address

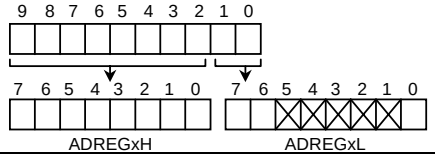
(9) Watchdog timer control

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	Watchdog timer mode	5CH	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
			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: $2^{21}/f_{SYS}$		Warm-up time 0: 2^{14}/input frequency 1: 2^{16}/input frequency	HALT mode 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		1: Connect internally WDT out to reset pin	1: Drive the pin in STOP mode
WDCR	Watchdog timer control register	5DH	—							
			W							
			—							
			B1H: WDT disable code 4EH: WDT clear code							

(10) AD converter control

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD0	AD mode control register 0	5EH	EOCF	ADBF	—	—	ITM0	REPET	SCAN	ADS
			R		R/W					
			0	0	0	0	0	0	0	0
			AD conversion 0: Not end 1: End	AD conversion busy flag 0: Not busy 1: Busy	Always fixed to "0".	Always fixed to "0".	AD conversion interrupt in fixed channel repeat mode 0: Each 1 time 1: Each 4 times	Repeat mode 0: Single mode 1: Repeat mode	Scan mode 0: Fixed channel mode 1: Channel scan mode	AD conversion start 0: Don't Care 1: Start Always "0" when data is read.
ADMOD1	AD mode control register 1	5FH	VREFON				ADTRGE	ADCH2	ADCH1	ADCH0
			R/W				R/W			
			1				0	0	0	0
			VREF control 0: OFF 1: ON				External trigger start control 0: Disable 1: Enable	Analog input channel selection		
AD REG04L	AD conversion result register 0/4 low	60H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
			Lower 2 bits of AD result are stored							AD conversion result storage flag
AD REG04H	AD conversion result register 0/4 high	61H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
			Upper 8 bits of AD result are stored							
AD REG15L	AD conversion result register 1/5 Low	62H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
			Lower 2 bits of AD result are stored							AD conversion result storage flag
AD REG15H	AD conversion result register 1/5 high	63H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
			Upper 8 bits of AD result are stored							
AD REG26L	AD conversion result register 2/6 Low	64H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
			Lower 2 bits of AD result are stored							AD conversion result storage flag
AD REG26H	AD conversion result register 2/6 high	65H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
			Upper 8 bits of AD result are stored							
AD REG37L	AD conversion result register 3/7 low	66H	ADR31	ADR30						ADR3RF
			R							R
			Undefined							0
			Lower 2 bits of AD result are stored							AD conversion result storage flag
AD REG37H	AD conversion result register 3/7 high	67H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			Undefined							
			Upper 8 bits of AD result are stored							

- Bits 5 to 1 are always read as 1.
  Bit0 is the AD conversion result storage flag <ADRxRF>. Bit0 is set to 1 when the conversion result is stored, and it is cleared by reading either ADREGxH or ADREGxL.



(11) Timer for real time clock control

Symbol	Name	Address	7	6	5	4	3	2	1	0
RTCCR	Real time counter control register	59H	–						RTCSEL	RTCRUN
			R/W						R/W	
			0						0	0
			Always fixed to "0".						Interrupt cycle 0: $fs/2^{14}$ 1: $fs/2^{13}$	Counting 0: Stop and clear 1: Count

(12) LCD driver control

Symbol	Name	Address	7	6	5	4	3	2	1	0
LCDCR	LCD control register	94H		SLFR1	SLFR0	SBF	DUTY1	DUTY0	SLF1	SLF0
				W						
				0	0	0	0	0	0	0
				Boosting frequency 00: $fc/2^{14}$ or $fs/2^5$ 01: $fc/2^{12}$ or $fs/2^3$ 10: $fc/2^{11}$ or $fs/2^2$ 11: $fc/2^{10}$ or fs/2		Reference clock 0: fc 1: fs	LCD drive 00: 1/4 duty 01: 1/3 duty 10: 1/2 duty 11: Static		Base frequency 00: $fc/2^{18}$ or $fs/2^9$ 01: $fc/2^{17}$ or $fs/2^8$ 10: $fc/2^{16}$ or $fs/2^7$ 11: $fc/2^{15}$ or $fs/2^6$	
CHPCR	Charge pump control register	95H	EDSP	EDCP						
			R/W							
			0	0						
			LCD display 0: Blanking 1: Enable	Boosting 0: Disable 1: Enable						

(13) Key-on wakeup control

Symbol	Name	Address	7	6	5	4	3	2	1	0
KEYCR	KEY on wakeup control register	4AH	KEY7C	KEY6C	KEY5C	KEY4C	KEY3C	KEY2C	KEY1C	KEY0C
			R/W							
			0	0	0	0	0	0	0	0
			0: Key-on wakeup disable 1: Key-on wakeup enable							

6. Port Section Equivalent Circuit Diagram

- Reading the circuit diagram

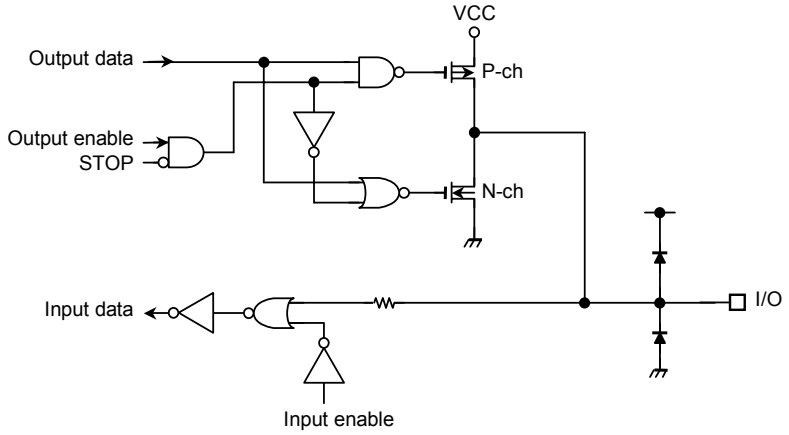
  Basically, the gate symbols written are the same as those used for the standard CMOS logic IC [74HCXX] series.

  The dedicated signal is described below.

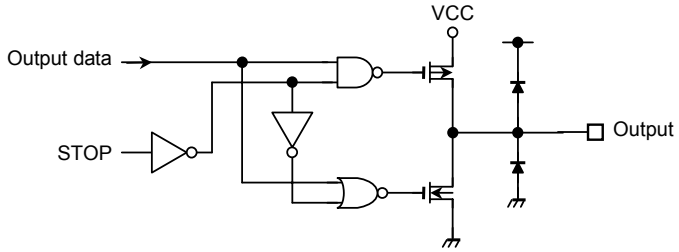
  Stop: This signal becomes active 1 when the HALT mode setting register is set to the Stop mode and the CPU executes the HALT instruction. When the drive enable bit [DRVE] is set to 1, however, stop remains at 0.

- The input protection resistance ranges from several tens of ohms to several hundreds of ohms.

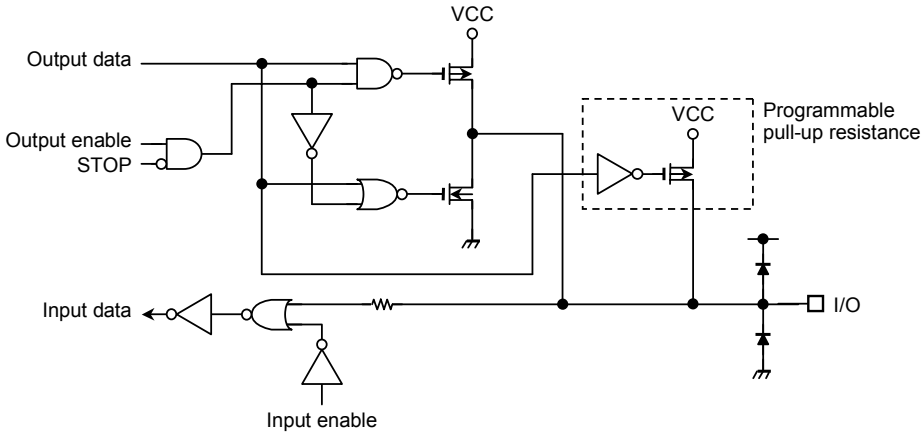
■ P0 (AD0 to AD7), P1 (AD8 to AD15, A8 to A15), P67 (TO1)



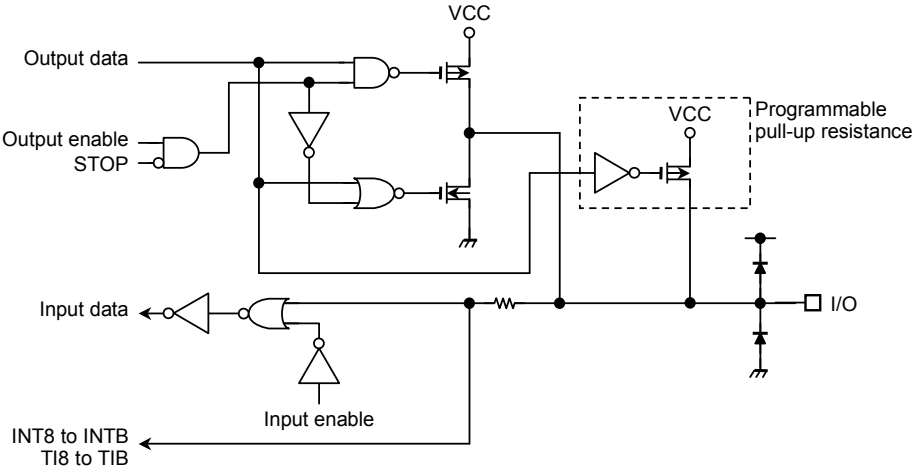
■ P30 ($\overline{RD}$), P31 ($\overline{WR}$)



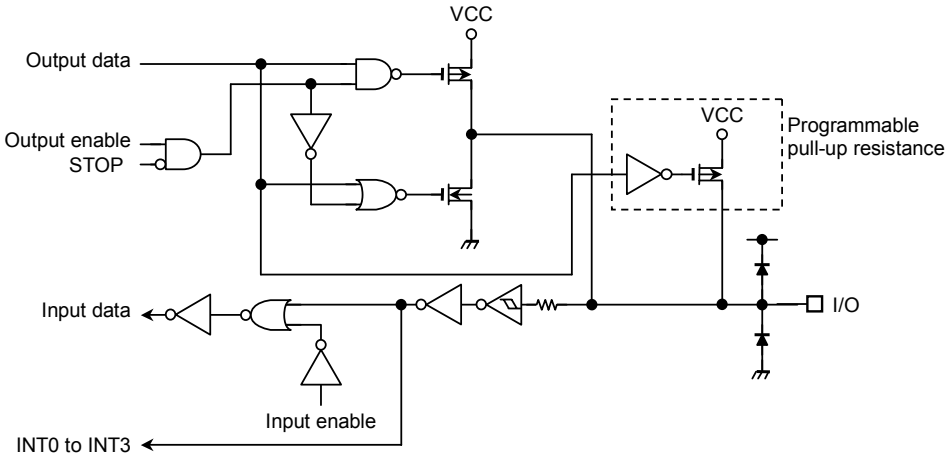
■ P2 (A16 to A23, A0 to A7), P32 ($\overline{HWR}$), P72 (TO8), P73 (SCOUT), P76 (TOA)



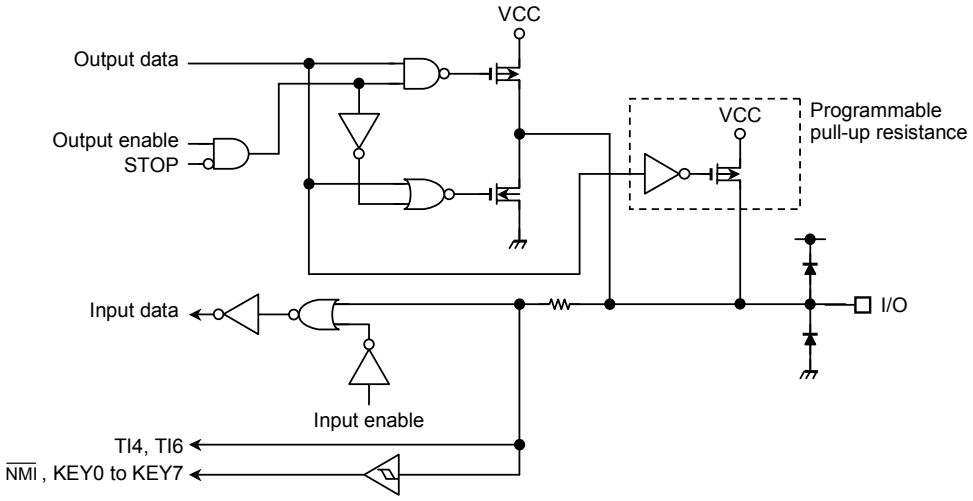
■ P70 (TI8/INT8), P71 (TI9/INT9), P74 (TIA/INTA), P75 (TIB/INTB)



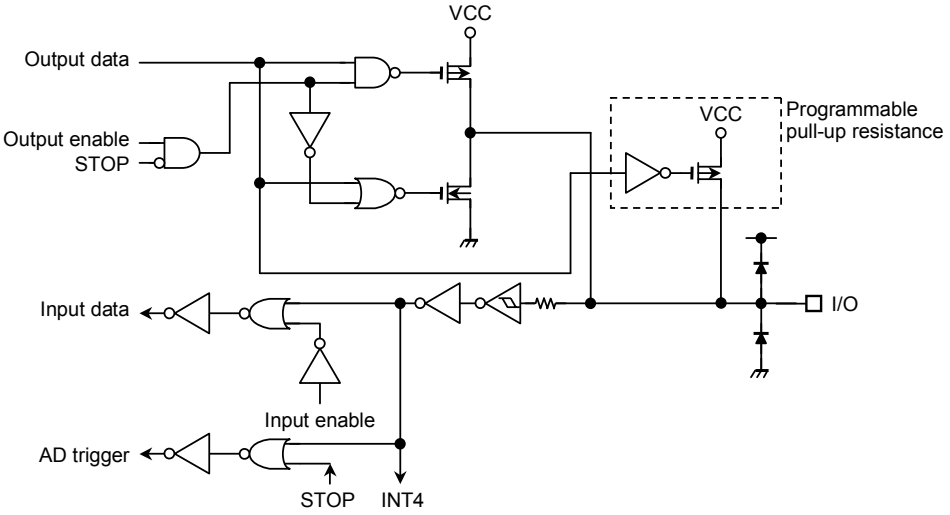
■ P33 to P36 (INT0 to INT3)



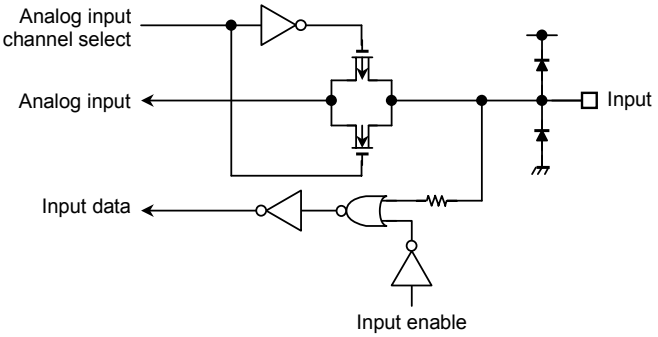
■ P4 (TIx, TOx, KEYx), P77 ($\overline{\text{NMI}}$)



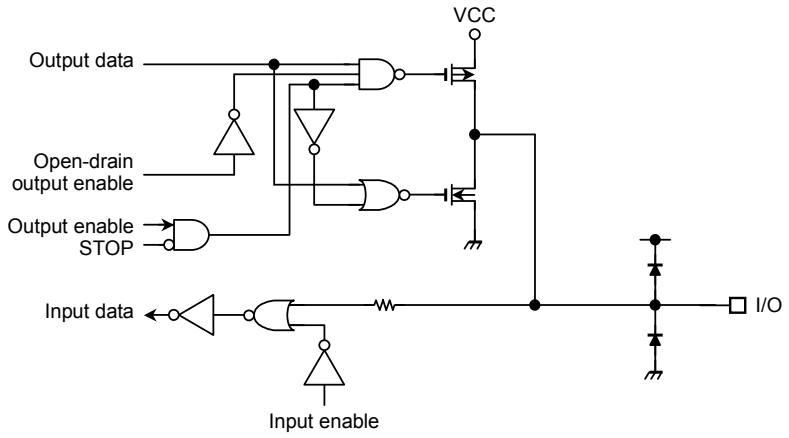
■ P37 (INT4/$\overline{\text{ADTRG}}$)



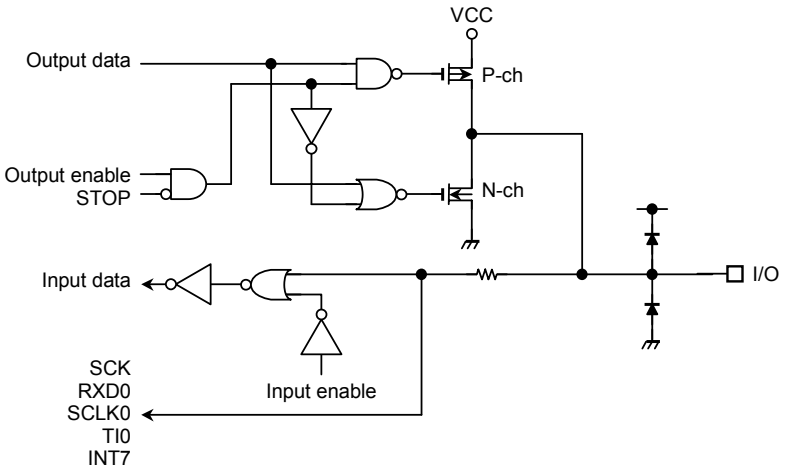
■ P5 (AN0 to AN7)



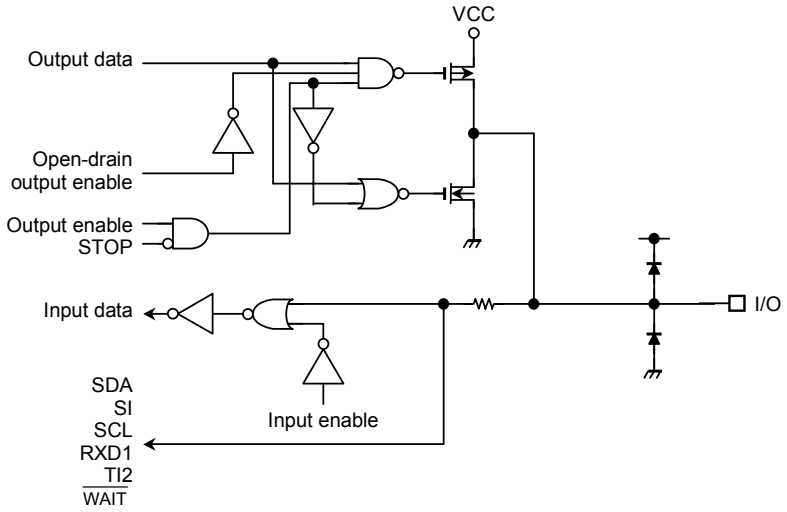
■ P63 (TXD0), P80 (TXD1), P83 (TO3), P85



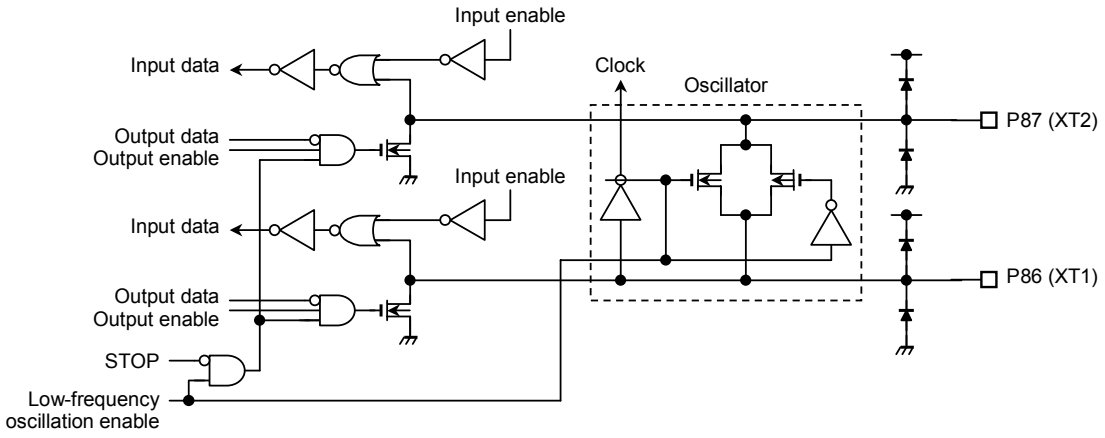
■ P60 (SCK), P64 (RXD0), P65 (SCLK0), P66 (TI0/INT7)



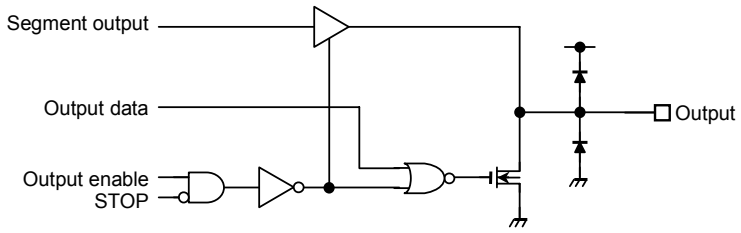
■ P61 (SO/SDA), P62 (SI/SCL), P81 (RXD1), P82 (TI2), P84 ($\overline{\text{WAIT}}$)



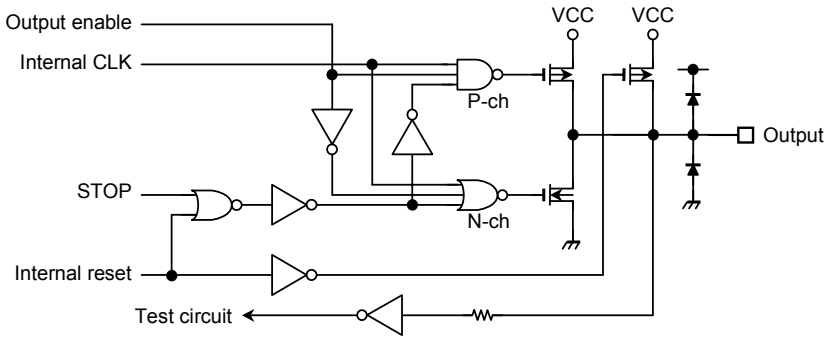
■ P86 (XT1), P87 (XT2)



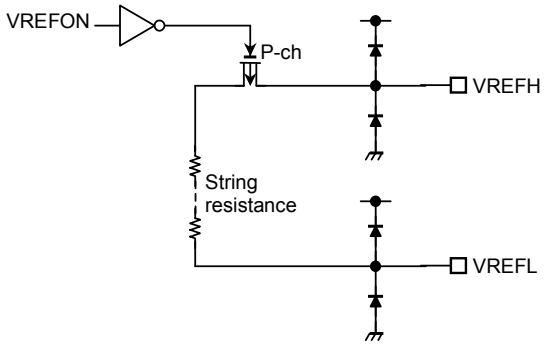
■ P90 to P97 (SEG24 to SEG31), PA0 to PA7 (SEG32 to SEG39)



■ CLK

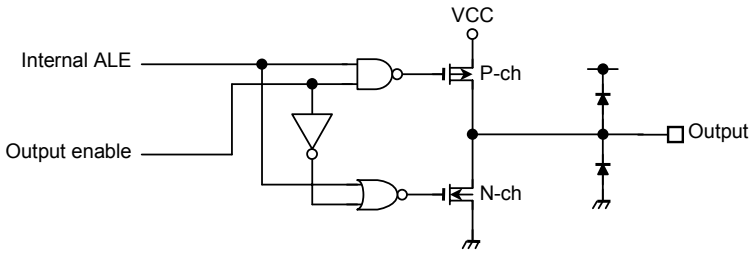


■ VREFH, VREFL

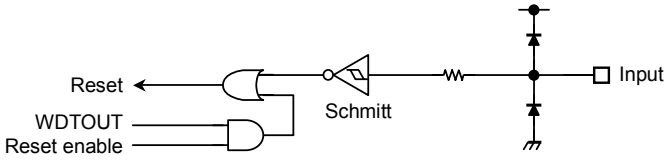


■ $\overline{EA}$

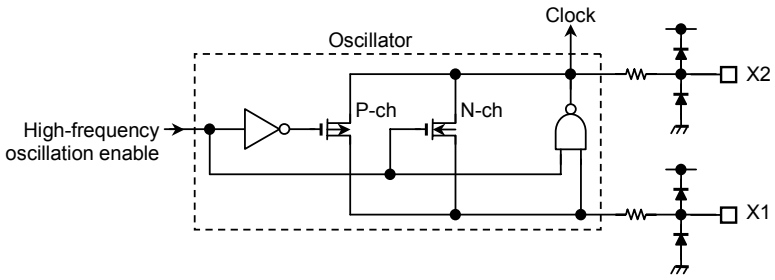
■ ALE



■ $\overline{\text{RESET}}$



■ X1, X2



7. Points to Note and Restrictions

(1) Notation

a. How a built-in I/O register is denoted: Register Symbol <Bit symbol>

Example: TRUN<T0RUN> … Bit T0RUN of register TRUN

b. Read-modify-write instruction

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, (TRUN) ···Set bit3 of TRUN
Example 2: INC 1, (100H)·····Increment the data at 100H

- A sample read-modify-write instructions using the TLCS-900

Exchange instruction

EX (mem), R

Arithmetic operation

ADD	(mem), R/#	ADC	(mem), R/#
SUB	(mem), R/#	SBC	(mem), R/#
INC	#3, (mem)	DEC	#3, (mem)

Logic operation

AND	(mem), R/#	OR	(mem), R/#
XOR	(mem), R/#		

Bit manipulation

STCF	#3/A, (mem)	RES	#3, (mem)
SET	#3, (mem)	CHG	#3, (mem)
TSET	#3, (mem)		

Rotate, shift

RLC	(mem)	RRC	(mem)
RL	(mem)	RR	(mem)
SLA	(mem)	SRA	(mem)
SLL	(mem)	SRL	(mem)
RLD	(mem)	RRD	(mem)

c. fc, fs, f_{FPH}, f_{SYS}, one state

The clock frequency input from pins X1 and X2 is called fc, the clock frequency input from pins XT1 and XT2 is called fs, the clock selected by SYSCR<SYSCK and GEAR2:0> is called f_{FPH}, and the clock frequency given by f_{FPH} divided by 2 is called f_{SYS}. One cycle of f_{SYS} is called one state.

(2) Points to note

a. $\overline{EA}$

Fix these pins to VCC unless changing voltage.

b. TEST1, TEST2 pin

Connect the TEST1 pin with the TEST2 pin. Do not connect to any other pins.

c. HALT mode (IDLE1)

When IDLE1 mode (Oscillator operation only) is used, set TRUN<PRRUN> to 0 to stop the prescaler before the HALT instruction is executed.

d. 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 warm-up time elapses from input of the release request to output of the system clock.

e. Programmable pull-up resistor

The programmable pull up resistor can be turned ON/OFF by the program when the ports are used as input ports. When the ports are used as outputs, they cannot be turned ON/OFF by the program.

The data registers (e.g., P6) are used for the pull-up/down resistors ON/OFF. Consequently read-modify-write instructions are prohibited.

f. Watchdog timer

The watchdog timer starts operation immediately after the reset is released. When the watch dog timer will not be used, disable it.

g. AD converter

The string resistor between the VREFH and VREFL pins can be cut by a program to reduce power consumption. When standby mode is used, disable the resistor using the program before the HALT instruction is executed.

h. CPU (Micro DMA)

Only the “LDC cr, r” and “LDC r, cr” instructions can be used to access the control registers in the CPU (Like the transfer source address register (DMASn)).

i. POP SR instruction

Please execute POP SR instruction during DI condition.

j. HALT mode (STOP)

If you can enter STOP mode on either of the following conditions, fix port 8 to VCC or GND.

- When setting the pin control to “I/O off” at STOP mode.
- When setting the pin control to “keep the condition prior to halt” at STOP mode and also when port 8 is input mode.

k. Supply of reference voltage to V1 pin

Do not supply the reference voltage to V1 pin during STOP mode.

l. Releasing the HALT mode by requesting an interruption.

Usually, interrupt can release all halt status. However, the interrupts ($\overline{NMI}$, INT0 to INT4, INTKEY, and 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 3 clocks of f_{FPH}) with IDLE1 or STOP mode (RUN and IDLE2 are not applicable to this case). In this case, an interrupt request is kept on hold internally.

If another interruption 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-LQFP144-1616-0.40

Unit: mm