TOSHIBA

8 Bit Microcontroller
TLCS-870/C Series

TMP86FM48

TOSHIBA CORPORATION

CMOS 8-Bit Microcontroller
TMP86FM48UG/FG

The TMP86FM48 is the high-speed, high-performance and low power consumption 8-bit microcomputer, including FLASH, RAM, multi-function timer/counter, serial interface (UART, SIO, I²C), a 10-bit AD converter and two clock generators on chip.

Product No.	FLASH (Program area)	FLASH (Data area)	RAM	Package	Emulation Chip
TMP86FM48UG	32256 × 8 bits	512 × 8 bits	2.0 K × 8 bits	LQFP64-P-1010-0.50E	TMP86C948XB
TMP86FM48FG				QFP64-P-1414-0.80C	

Features

♦ 8-bit single chip microcomputer TLCS-870/C series

♦ Instruction execution time: 0.25 μs (at 16 MHz)
  122 μs (at 32.768 kHz)

♦ 132 types and 731 basic instructions

♦ 20 interrupt sources (External: 5, Internal: 15)

♦ Input/output ports (54 pins)

♦ 16-bit timer counter: 2 ch
  • Timer, Event counter,
    Pulse width measurement, External trigger timer,
    Window, PPG output modes

♦ 8-bit timer counter: 2 ch
  • Timer, Event counter, PWM output,
    Programmable divider output, Capture modes

♦ Time base timer

♦ Divider output function

♦ Watchdog timer
  • Interrupt source/internal reset generate (Programmable)

LQFP64-P-1010-0.50E

TMP86FM48UG

QFP64-P-1414-0.80C

TMP86FM48FG

♦ Serial interface
  • UART/SIO: 1ch
  • SIO: 1ch
  • I²C bus: 1ch

♦ 10-bit successive approximation type AD converter
  • Analog input: 16 ch

♦ Four Key-on wake-up pins

♦ Dual clock operation
  • Single/dual-clock mode

♦ Nine power saving operating modes
  • STOP mode: Oscillation stops. Battery/capacitor back-up.
                      Port output hold/High-impedance.
  • SLOW 1, 2 mode: Low-power consumption operation using low-frequency clock (32.768 kHz)
  • IDLE 0 mode: CPU stops, and peripherals operate using high-frequency clock of Time-Base-Timer. Release by falling edge of TBTCR<TBTCK> setting.
  • IDLE 1 mode: CPU stops, and peripherals operate using high-frequency clock.
                      Release by interruputs.
  • IDLE 2 mode: CPU stops, and peripherals operate using high and low-frequency clock.
                      Release by interruputs.
  • SLEEP 0 mode: CPU stops, and peripherals operate using low-frequency clock of time-base-timer. Release by falling edge of TBTCR<TBTCK> setting.
  • SLEEP 1 mode: CPU stops, and peripherals operate using low-frequency clock.
                      Release by interrupts.
  • SLEEP 2 mode: CPU stops, and peripherals operate using high- and low-frequency clock.
                      Release by interrupts.

♦ Wide operating voltage: 1.8 to 3.6 V at 8 MHz/32.768 kHz
                          2.7 to 3.6 V at 16 MHz/32.768 kHz

Pin Assignments (Top view)

LQFP64-P-1010-0.50E
QFP64-P-1414-0.80C

Top pins (left to right, 48→33):
P77(AIN17), P76(AIN16), P75(AIN15), P74(AIN14), P73(AIN13), P72(AIN12), P71(AIN11), P70(AIN10), P67(AIN07/STOP3), P66(AIN06/STOP2), P65(AIN05/STOP1), P64(AIN04/STOP0), P63(AIN03), P62(AIN02), P61(AIN01), P60(AIN00)

Left pins (49–64):
P80 ↔ 49
P81 ↔ 50
P82 ↔ 51
P83 ↔ 52
P84 ↔ 53
P85 ↔ 54
P86 ↔ 55
P87 ↔ 56
P30 ↔ 57
P31 ↔ 58
P32 ↔ 59
P33 ↔ 60
P34 ↔ 61
P35 ↔ 62
P36 ↔ 63
P37 ↔ 64

Right pins (32–17):
32 ← AVDD
31 ← VAREF
30 ← AVSS/VASS
29 ← BOOT
28 ↔ P52
27 ↔ P51 ($\overline{DVO}$ /SDA)
26 ↔ P50 ($\overline{PPG}$ /SCL)
25 ↔ P07 ($\overline{SCK1}$)
24 ↔ P06 (SO1/TXD)
23 ↔ P05 (SI1/RXD)
22 ↔ P04
21 ↔ P03 (TC2)
20 ↔ P02 (INT2)
19 ↔ P01 (INT1)
18 ↔ P00 ($\overline{INT0}$)
17 ↔ P17

Bottom pins (left to right, 1–16):
VSS →, XIN →, XOUT ←, TEST →, VDD →, P21 (XTIN) ↔, P22 (XTOUT) ↔, $\overline{RESET}$ →, P20 ($\overline{STOP}$/$\overline{INT5}$) ↔, P10 (SO2) ↔, P11 (SI2) ↔, P12 ($\overline{SCK2}$) ↔, P13 ($\overline{PWM5}$/PDO5/TC5) ↔, P14 (INT3/TC3) ↔, P15 (INT3/TC3) ↔, P16 (TC1) ↔

Block Diagram

Pin Functions (1/2)

Pin Name	Input/Output	Functions	
P07 ($\overline{SCK1}$)	I/O (I/O)	8-bit input/output port with latch. When used as a serial interface output or UART output, respective output latch (P0DR) should be set to "1". When used as an input port, an serial interface input, UART input, timer counter input or an external interrupt input, respective output control (P0OUTCR) should be cleared to "0" after setting P0DR to "1".	Serial clock input/output 1
P06 (TXD, SO1)	I/O (Output)		UART data output, Serial data output 1
P05 (RXD, SI1)	I/O (Input)		UART data input, Serial data input 1
P04	I/O		
P03 (TC2)	I/O (Input)		Timer counter 2 input
P02 (INT2)	I/O (Input)		External interrupt 2 input
P01 (INT1)	I/O (Input)		External interrupt 1 input
P00 ($\overline{INT0}$)	I/O (Input)		External interrupt 0 input
P17	I/O	8-bit input/output port with latch. When used as a timer/counter output or serial interface output, respective output latch (P1DR) should be set to "1". When used as an input port, a timer counter input, an external interrupt input or serial interface input, respective output control (P1OUTCR) should be cleared to "0" after setting P1DR to "1".	
P16	I/O		
P15 (TC1)	I/O (Input)		Timer counter 1 input
P14 (TC3,INT3)	I/O (Input)		Timer counter 3 input, External interrupt 3 input
P13 ($\overline{PWM5}$, $\overline{PDO5}$, TC5)	I/O (I/O)		PWM5 output, PDO5 output, Timer/counter 5 input
P12 ($\overline{SCK2}$)	I/O (I/O)		Serial clock input/output 2
P11 (SI2)	I/O (Input)		Serial data input 2
P10 (SO2)	I/O (Output)		Serial data output 2
P22 (XTOUT)	I/O (Output)	3-bit input/output port with latch. When used as an input port or an external interrupt input, respective output control (P2OUTCR) should be cleared to "0" after setting output latch (P2DR) to "1".	Resonator connecting pins (32.768 kHz) For inputting external clock, XTIN is used and XTOUT is opened.
P21 (XTIN)	I/O (Input)		
P20 ($\overline{INT5}$, $\overline{STOP}$)	I/O (Input)		External interrupt input 5 or STOP mode release signal input
P37 to P30	I/O	8-bit input/output port with latch (N-ch high-current output). When used as an input port, respective output control (P3OUTCR) should be cleared to "0" after setting output latch (P3DR) to "1".	
P52	I/O	3-bit input/output port with latch (N-ch high-current output). When used as an input port or I^{2}C bus interface input/output, respective output control (P5OUTCR) should be cleared to "0" after setting output latch (P5DR) to "1". When used as a PPG output or divider output, respective P5DR should be set to "1".	
P51 ($\overline{DVO}$, SDA)	I/O (Output,I/O)		Divider Output/I^{2}C bus serial data input/output
P50 ($\overline{PPG}$, SCL)	I/O (Output,I/O)		PPG Output/I^{2}C bus serial clock input/output
P67 (AIN07, STOP3)	I/O (Input)	8-bit programmable input/output port (tri-state). Each bit of this port can be individually configured as an input or an output under software control. When used as an input port, respective input/output control (P6CR1) should be cleared to "0" after setting input control (P6CR2) to "1". When used as an analog input or key on wake up input, respective P6CR1 should be cleared to "0" after clearing P6CR2 to "0". When used as a key on wake up input, STOPCR<STOPiEN > should be set to "1". (i = 0 to 3)	STOP 3 input	
P66 (AIN06, STOP2)	I/O (Input)		STOP 2 input	
P65 (AIN05, STOP1)	I/O (Input)		STOP 1 input	
P64 (AIN04, STOP0)	I/O (Input)		STOP 0 input	AD converter analog inputs
P63 (AIN03)	I/O (Input)			
P62 (AIN02)	I/O (Input)			
P61 (AIN01)	I/O (Input)			
P60 (AIN00)	I/O (Input)			

Pin Functions (2/2)

Pin Name	Input/Output	Functions	Pin Name
P77 (AIN17)	I/O (Input)	8-bit programmable input/output port (tri-state). Each bit of this port can be individually configured as an input or an output under software control. When used as an input port, respective input/output control (P7CR1) should be cleared to "0" after setting input control (P7CR2) to "1". When used as an analog input, respective P7CR1 should be cleared to "0" after clearing P7CR2 to "0".	AD converter analog inputs
P76 (AIN16)	I/O (Input)		
P75 (AIN15)	I/O (Input)		
P74 (AIN14)	I/O (Input)		
P73 (AIN13)	I/O (Input)		
P72 (AIN12)	I/O (Input)		
P71 (AIN11)	I/O (Input)		
P70 (AIN10)	I/O (Input)		
P87 to P80	I/O	8-bit input/output port with latch (N-ch high-current output). When used as an input port, respective output control (P8OUTCR) should be cleared to "0" after setting output latch (P8DR) to "1".	
XIN, XOUT	Input Output	Resonator connecting pins for high-frequency clock. For inputting external clock, XIN is used and XOUT is opened.	
$\overline{\text{RESET}}$	Input	Reset signal input	
TEST	Input	Test pin for out-going test. Be fixed to low.	
BOOT	Input	Serial prom mode control input. When writing to FLASH memory, BOOT pin should be fixed to high level.	
VDD, VSS	Power Supply	Power supply for operation	
VAREF		Analog reference voltage for AD conversion	
AVDD		AD circuit power supply	
AVSS/VASS		AD circuit power supply/Analog reference GND for AD conversion	

Operational Description

1. CPU Core Functions

The CPU core consists of a CPU, a system clock controller, and an interrupt controller.

This section provides a description of the CPU core, the program memory, the data memory, the external memory interface, and the reset circuit.

1.1 Memory Address Map

The TMP86FM48 memory consists of 5 blocks: FLASH memory, BOOT ROM, RAM, DBR (Data buffer register) and SFR (Special function register). They are all mapped in 64-Kbyte address space. Figure 1.1.1 shows the TMP86FM48 memory address map. The general-purpose registers are not assigned to the RAM address space.

SFR	0000_H / $003F_H$	64 bytes
	0040_H	
RAM		2048 bytes
	$083F_H$	
DBR	$1F80_H$ / $1FFF_H$	128 bytes
BOOT ROM	3800_H / $3FFF_H$	2048 bytes
FLASH memory (Data memory)	8000_H / $81FF_H$	512 bytes
	8200_H	
FLASH memory (Program memory)		32176 bytes
	$FFB0_H$ / $FFBF_H$	16 bytes — Vector table for interrupts (8 vectors)
	$FFC0_H$ / $FFDF_H$	32 bytes — Vector table for vector call instructions (16 vectors)
	$FFE0_H$ / $FFFF_H$	32 bytes — Vector table for interrupts/reset (16 vectors)

FLASH memory: FLASH memory includes:
Program memory (The area of 8000_H to $81FF_H$ can be used as data memory.)
Vector table
BOOT ROM: FLASH writing program
RAM: Random Access Memory includes:
Data memory
Stack
SFR: Special Function Register includes:
I/O ports
Peripheral control registers
Peripheral status registers
System control registers
Interrupt control registers
Program Status Word
DBR: Data Buffer Register includes:
Peripheral control registers
Peripheral Status registers

Figure 1.1.1 Memory Address Maps

1.2 Program Memory (FLASH)

The TMP86FM48 has a 32 K × 8 bits (Address 8000_H to $FFFF_H$) of program memory (FLASH). The area of 8000H to 81FFH can be used as a 512 × 8 bits data memory of FLASH.

1.3   Data Memory (RAM)

The TMP86FM48 has 2048 bytes of internal RAM. The first 192 bytes (0040_H to $00FF_H$) of the internal RAM are located in the direct area; instructions with shorten operations are available against such an area.

The data memory contents become unstable when the power supply is turned on; therefore, the data memory should be initialized by an initialization routine.

```
Example:  Clears RAM to "00H".
              LD      HL, 0040H            ;   Start address setup
              LD      A, H                 ;   Initial value (00H) setup
              LD      BC, 07FFH            ;
SRAMCLR:      LD      (HL), A
              INC     HL
              DEC     BC
              JRS     F, SRAMCLR
```

1.4 System Clock Controller

www.DataSheet4U.comThe system clock controller consists of a clock generator, a timing generator, and a standby controller.
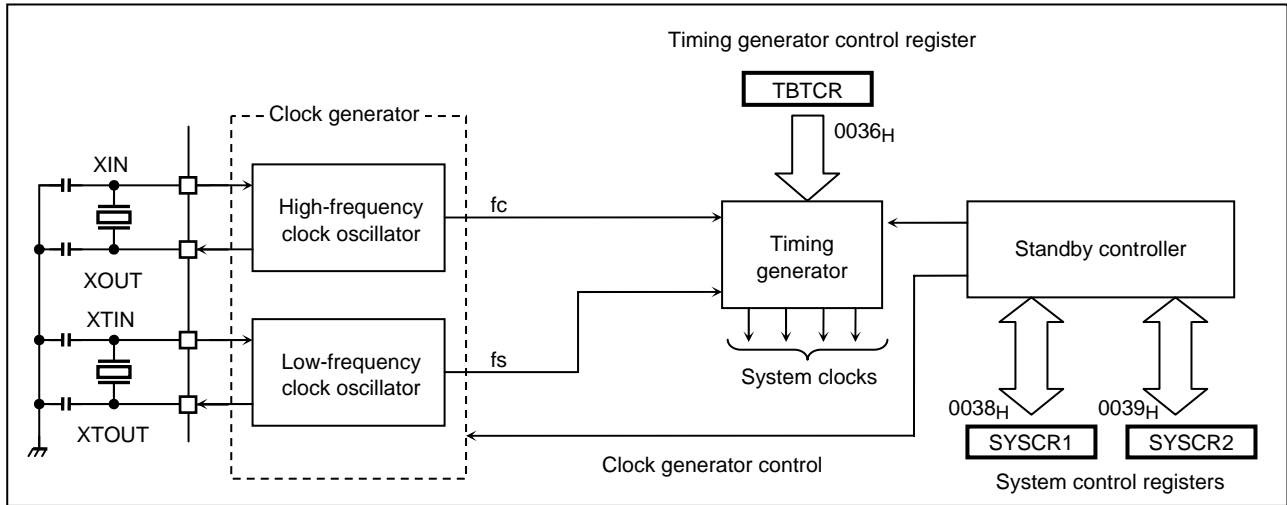
Figure 1.4.1 System Clock Control

1.4.1 Clock Generator

The clock generator generates the basic clock which provides the system clocks supplied to the CPU core and peripheral hardware. It contains two oscillation circuits: one for the high-frequency clock and one for the low-frequency clock. Power consumption can be reduced by switching of the standby controller to low-power operation based on the low-frequency clock.

The high-frequency (fc) and low-frequency (fs) clocks can easily be obtained by connecting a resonator between the XIN/XOUT and XTIN/XTOUT pins respectively. Clock input from an external oscillator is also possible. In this case, external clock is applied to XIN/XTIN pin with XOUT/XTOUT pin not connected.
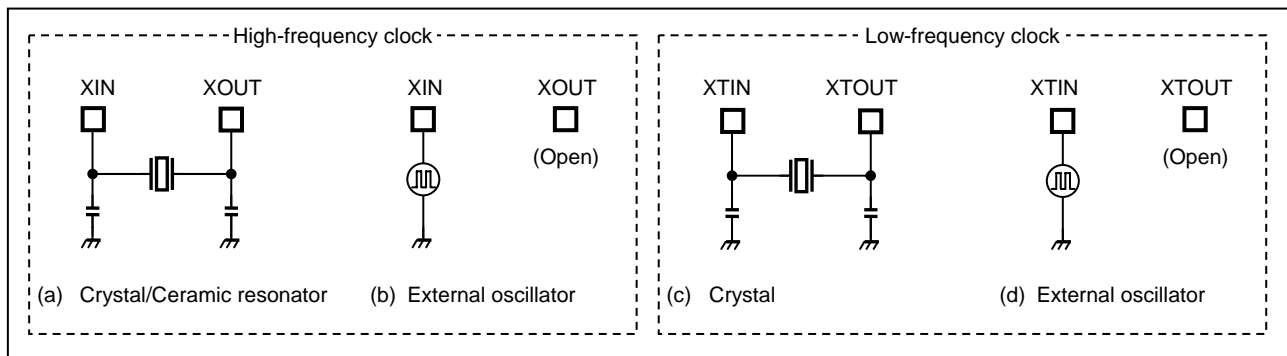
Figure 1.4.2 Examples of Resonator Connection

Note: The function to monitor the basic clock directly at external is not provided for hardware, however, with disabling all interrupts and watchdog timers, the oscillation frequency can be adjusted by monitoring the pulse which the fixed frequency is outputted to the port by the program.

The system to require the adjustment of the oscillation frequency should create the program for the adjustment in advance.

1.4.2   Timing Generator

The timing generator generates the various system clocks supplied to the CPU core and peripheral hardware from the basic clock (fc or fs). The timing generator provides the following functions.

    a.   Generation of main system clock

    b.   Generation of divider output ($\overline{\text{DVO}}$) pulses

    c.   Generation of source clocks for time base timer

    d.   Generation of source clocks for watchdog timer

    e.   Generation of internal source clocks for timer/counters and serial interface

    f.   Generation of warm-up clocks for releasing STOP mode

（1） Configuration of timing generator

The timing generator consists of a 2-stage prescaler, a 21-stage divider, a main system clock generator, and machine cycle counters.

An input clock to the 7th stage of the divider depends on the operating mode, TBTCR<DV7CK>, that is shown in Figure 1.4.4. As reset and STOP mode started/canceled, the prescaler and the divider are cleared to "0".
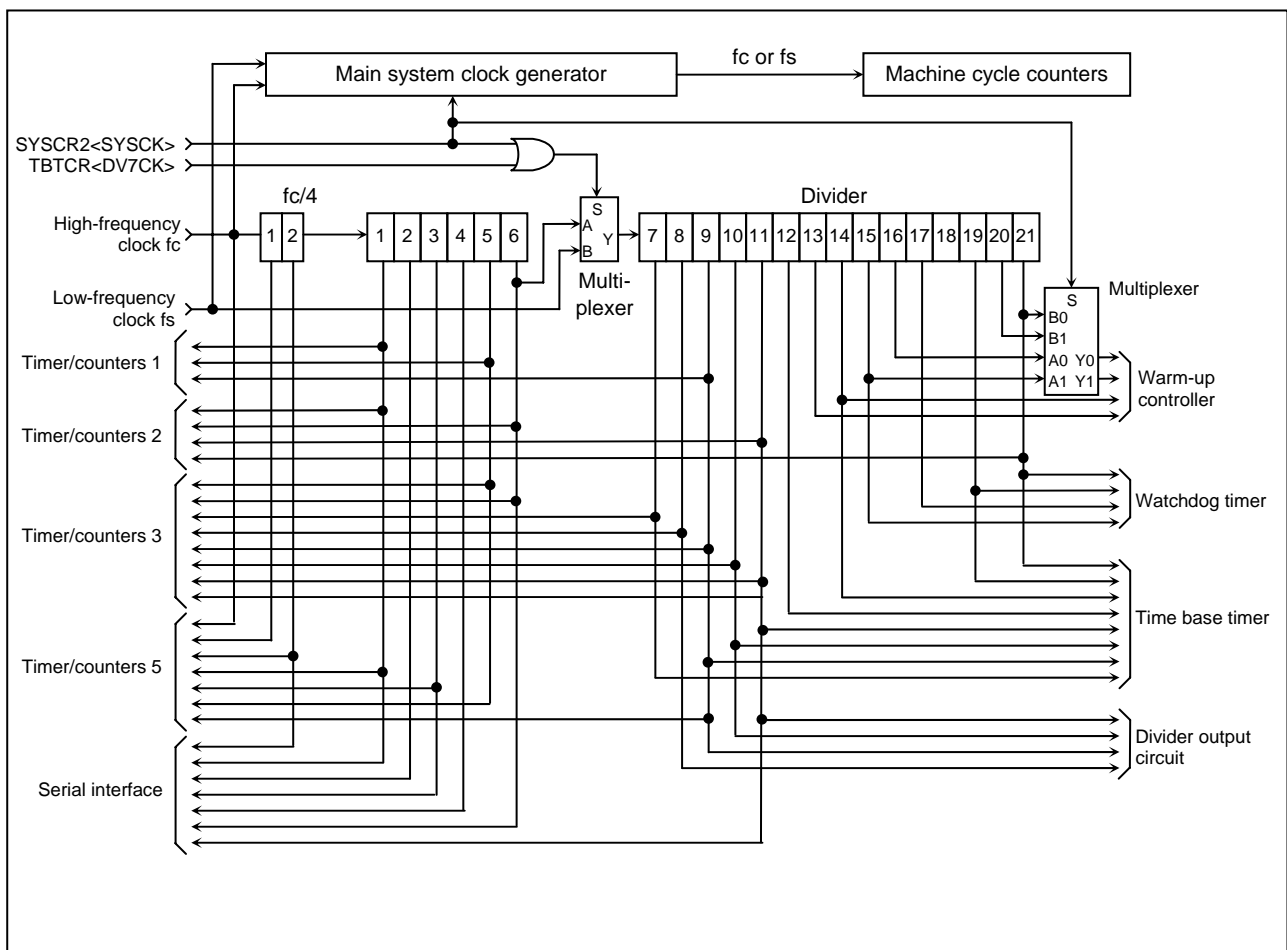
Figure 1.4.3 Configuration of Timing Generator

TBTCR	7	6	5	4	3	2	1	0	
(0036H)	(DVOEN)	(DVOCK)		DV7CK	(TBTEN)		(TBTCK)		(Initial value: 0000 0000)

DV7CK	Selection of input to the 7th stage of the divider	0: fc/2^8 [Hz] 1: fs	R/W

Note 1: In single clock mode, do not set DV7CK to "1".

Note 2: Do not set "1" on DV7CK while the low-frequency clock is not operated stably.

Note 3: fc: High-frequency clock [Hz], fc: Low-frequency clock [Hz], ∗: Don't care

Note 4: In SLOW1/2 and SLEEP1/2 modes, the DV7CK setting is ineffective, and fs is input to the 7th stage of the divider.

Note 5: When STOP mode is entered from NORMAL 1/2 mode, the DV7CK setting is ineffective during the warm-up period after release of STOP mode, and the 6th stage of the divider is input to the 7th stage during this period.

Figure 1.4.4 Timing Generator Control Register

(2) Machine cycle

Instruction execution and peripheral hardware operation are synchronized with the main system clock.

The minimum instruction execution unit is called an "machine cycle". There are a total of 10 different types of instructions for the TLCS-870/C series: ranging from 1-cycle instructions which require one machine cycle for execution to 10-cycle instructions which require 10 machine cycles for execution.

A machine cycle consists of 4 states (S0 to S3), and each state consists of one main system clock.
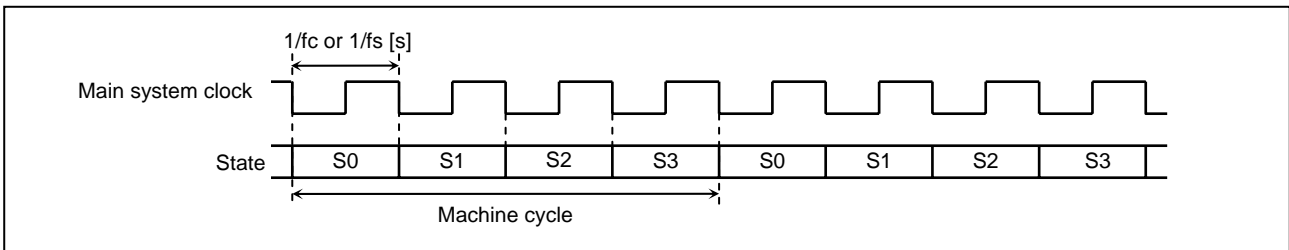
Figure 1.4.5 Machine Cycle

1.4.3 Operation Mode Control Circuit

The operation mode control circuit starts and stops the oscillation circuits for the high-frequency and low-frequency clocks, and switches the main system clock. There are two operating modes: single-clock and dual-clock. These modes are controlled by the system control registers (SYSCR1 and SYSCR2).

Figure 1.4.6 shows the operating mode transition diagram and Figure 1.4.7 shows the system control registers.

(1) Single-clock mode

Only the oscillation circuit for the high-frequency clock is used, and P21 (XTIN) and P22 (XTOUT) pins are used as input/output ports. The main-system clock is obtained from the high-frequency clock. In the single-clock mode, the machine cycle time is 4/fc [s].

a. NORMAL1 mode

In this mode, both the CPU core and on-chip peripherals operate using the high-frequency clock.

The TMP86FM48 is placed in this mode after reset.

b. IDLE1 mode

In this mode, the internal oscillation circuit remains active. The CPU and the watchdog timer are halted; however on-chip peripherals remain active (Operate using the high-frequency clock).

IDLE1 mode is started by SYSCR2<IDLE>, and IDLE1 mode is released to NORMAL1 mode by an interrupt request from the on-chip peripherals or external interrupt inputs. When the IMF (Interrupt master enable flag) is "1" (Interrupt enable), the execution will resume with the acceptance of the interrupt, and the operation will return to normal after the interrupt service is completed. When the IMF is "0" (Interrupt disable), the execution will resume with the instruction which follows the IDLE1 mode start instruction.

c. IDLE0 mode

In this mode, all the circuit, except oscillator and the time-base-timer, stops operation.

This mode is enabled by setting "1" on bit TGHALT on the system control register 2 (SYSCR2).

When IDLE0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

When returned from IDLE0 mode, the CPU restarts operating, entering NORMAL1 mode back again. IDLE0 mode is entered and returned regardless of how TBTCR<TBTEN> is set. When IMF = "1", EF7 (TBT interrupt individual enable flag) = "1", and TBTCR<TBTEN> = "1", interrupt processing is performed. When IDLE0 mode is entered while TBTCR<TBTEN> = "1", the INTTBT interrupt latch is set after returning to NORMAL1 mode.

(2) Dual-clock mode

www.DataSheet4U.com Both the high-frequency and low-frequency oscillation circuits are used in this mode. P21 (XTIN) and P22 (XTOUT) pins cannot be used as input/output ports. The main system clock is obtained from the high-frequency clock in NORMAL2 and IDLE2 modes, and is obtained from the low-frequency clock in SLOW and SLEEP modes. The machine cycle time is 4/fc [s] in the NORMAL2 and IDLE2 modes, and 4/fs [s] (122 μs at fs = 32.768 kHz) in the SLOW and SLEEP modes.

The TLCS-870/C is placed in the single-clock mode during reset. To use the dual-clock mode, the low-frequency oscillator should be turned on at the start of a program.

a. NORMAL2 mode

In this mode, the CPU core operates with the high-frequency clock. On-chip peripherals operate using the high-frequency clock and/or low-frequency clock.

b. SLOW2 mode

In this mode, the CPU core operates with the low-frequency clock, while both the high-frequency clock and the low-frequency clock are operated. On-chip peripherals are triggered by the low-frequency clock. As the SYSCK on SYSCR2 becomes "0", the hardware changes into NORMAL2 mode. As the XEN on SYSCR2 becomes "0", the hardware changes into SLOW1 mode. Do not clear XTEN to "0" during SLOW2 mode.

c. SLOW1 mode

This mode can be used to reduce power-consumption by turning off oscillation of the high-frequency clock. The CPU core and on-chip peripherals operate using the low-frequency clock.

Switching back and forth between SLOW1 and SLOW2 modes are performed by XEN bit on the system control register 2 (SYSCR2). In SLOW1 and SLEEP mode, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

d. IDLE2 mode

In this mode, the internal oscillation circuit remain active. The CPU and the watchdog timer are halted; however, on-chip peripherals remain active (Operate using the high-frequency clock and/or the low-frequency clock). Starting and releasing of IDLE2 mode are the same as for IDLE1 mode, except that operation returns to NORMAL2 mode.

e. SLEEP1 mode

In this mode, the internal oscillation circuit of the low-frequency clock remains active. The CPU, the watchdog timer, and the internal oscillation circuit of the high-frequency clock are halted; however, on-chip peripherals remain active (Operate using the low-frequency clock). Starting and releasing of SLEEP mode are the same as for IDLE1 mode, except that operation returns to SLOW mode. In SLOW and SLEEP mode, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

f.   SLEEP2 mode

The SLEEP2 mode is the IDLE mode corresponding to the SLOW2 mode. The status under the SLEEP2 mode is same as that under the SLEEP1 mode, except for the oscillation circuit of the high-frequency clock.

g.   SLEEP0 mode

In this mode, all the circuit, except oscillator and the time-base-timer, stops operation.

This mode is enabled by setting "1" on bit TGHALT on the system control register 2 (SYSCR2).

When SLEEP0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

When returned from SLEEP0 mode, the CPU restarts operating, entering SLOW1 mode back again. SLEEP0 mode is entered and returned regardless of how TBTCR<TBTEN> is set. When IMF = "1", EF7 (TBT interrupt individual enable flag) = "1", and TBTCR<TBTEN> = "1", interrupt processing is performed. When SLEEP0 mode is entered while TBTCR<TBTEN> = "1", the INTTBT interrupt latch is set after returning to SLOW1 mode.

(3)   STOP mode

In this mode, the internal oscillation circuit is turned off, causing all system operations to be halted. The internal status immediately prior to the halt is held with a lowest power consumption during STOP mode.

STOP mode is started by the system control register 1 (SYSCR1), and STOP mode is released by a inputting (either level-sensitive or edge-sensitive can be programmably selected) to the $\overline{\text{STOP}}$ pin or key on wake up pin input which is enabled by STOPCR. After the warm-up period is completed, the execution resumes with the instruction which follows the STOP mode start instruction.

Note 1: When the IDLE0/1/2 and SLEEP0/1/2 modes are started with the EEPCR<ATPWDW> = "0", the CPU wait period for stabilizing of the power supply of Flash control circuit is executed after being released from these mode.

Note 2: When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait period for stablizing of the power supply of Flash control circuit is executed after in the STOP warm-up time.

www.DataSheet4U.com

Note 1:   NORMAL1 and NORMAL2 modes are generically called NORMAL; SLOW1 and SLOW2 are called SLOW;
          IDLE0, IDLE1 and IDLE2 are called IDLE; SLEEP0, SLEEP1 and SLEEP2 are called SLEEP.

Note 2:   The mode is released by falling edge of TBTCR<TBTCK> setting.

Operating Mode		Oscillator		CPU Core	TBT	Other Peripherals	Machine Cycle Time
		High Frequency	Low Frequency				
Single Clock	RESET	Oscillation	Stop	Reset	Reset	Reset	4/fc [s]
	NORMAL1			Operate	Operate	Operate	
	IDLE1			Halt			
	IDLE0					Halt	–
	STOP	Stop			Halt		
Dual Clock	NORMAL2	Oscillation	Oscillation	Operate with high frequency	Operate	Operate	4/fc [s]
	IDLE2			Halt			
	SLOW2			Operate with low frequency			4/fs [s]
	SLEEP2			Halt			
	SLOW1	Stop		Operate with low frequency			
	SLEEP1			Halt			
	SLEEP0					Halt	–
	STOP		Stop		Halt		

Figure 1.4.6 Operating Mode Transition Diagram

System Control Register 1

SYSCR1   7   6   5   4   3   2   1   0
(0038$_H$) | STOP | RELM | RETM | OUTEN | WUT | | | (Initial value: 0000 00**)

		Return to NORMAL mode	Return to SLOW mode	
STOP	STOP mode start	0: CPU core and peripherals remain active 1: CPU core and peripherals are halted (Start STOP mode)		
RELM	Release method for STOP pin (P20)	0: Edge-sensitive release 1: Level-sensitive release		
RETM	Operating mode after STOP mode	0: Return to NORMAL1/2 mode 1: Return to SLOW1 mode		
OUTEN	Port output during STOP mode	0: High impedance 1: Output kept		R/W
WUT	Warm-up time at releasing STOP mode (Note 8)		Return to NORMAL mode	Return to SLOW mode
		00 01 10 11	$3 \times 2^{16}/fc + (2^{10}/fc)$ $2^{16}/fc + (2^{10}/fc)$ $3 \times 2^{14}/fc + (2^{10}/fc)$ $2^{14}/fc + (2^{10}/fc)$	$3 \times 2^{13}/fs + (2^3/fs)$ $2^{13}/fs + (2^3/fs)$ $3 \times 2^6/fs + (2^3/fs)$ $2^6/fs + (2^3/fs)$

Note 1: Always set RETM to "0" when transiting from NORMAL mode to STOP mode. Always set RETM to "1" when transiting from SLOW mode to STOP mode.

Note 2: When STOP mode is released with $\overline{\text{RESET}}$ pin input, a return is made to NORMAL1 regardless of the RETM contents.

Note 3: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 4: Bits 1 and 0 in SYSCR1 are read as undefined data when a read instruction is executed.

Note 5: As the hardware becomes STOP mode under OUTEN = "0", input value is fixed to "0"; therefore it may cause interrupt request on account of falling edge.

Note 6: When the key-on wake-up input (STOP0 to STOP3) is used, RELM should be set to "1".

Note 7: Port P20 is used as $\overline{\text{STOP}}$ pin. Therefore, when stop mode is started, OUTEN does not affect to P20, and P20 becomes High-Z mode.

Note 8: When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait period for stabilizing of the power supply of Flash control circuit is executed after in the STOP warm-up time.
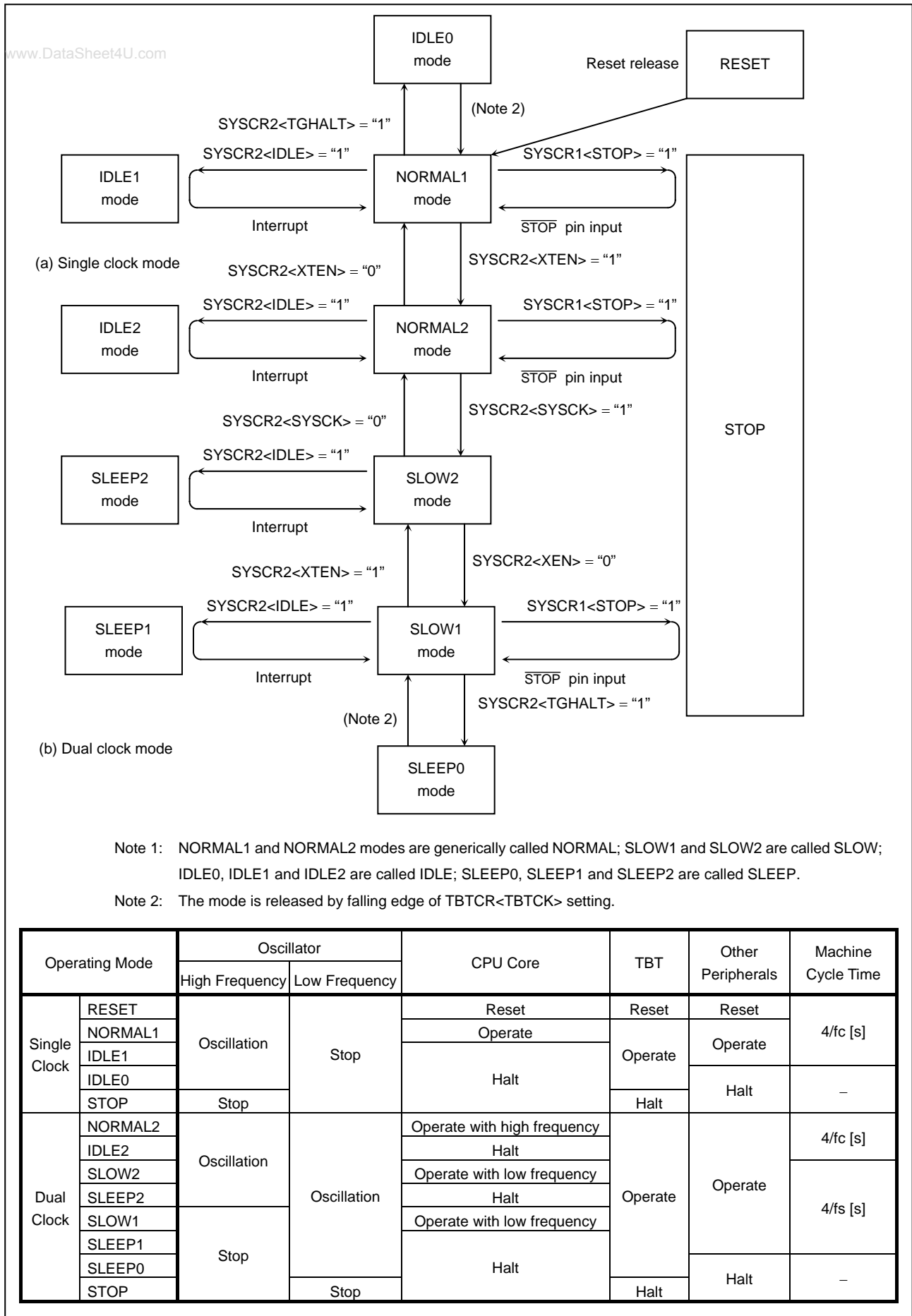(The CPU wait period for FLASH is shown in parentheses)

System Control Register 2

SYSCR2   7   6   5   4   3   2   1   0
(0039$_H$) | XEN | XTEN | SYSCK | IDLE | | TGHALT | | | (Initial value: 1000 *0**)
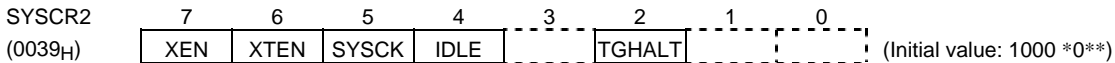
XEN	High-frequency oscillator control	0: Turn off oscillation 1: Turn on oscillation	
XTEN	Low-frequency oscillator control	0: Turn off oscillation 1: Turn on oscillation	
SYSCK	Main system clock select (write)/main system clock monitor (read)	0: High-frequency clock 1: Low-frequency clock	R/W
IDLE	CPU and watchdog timer control (IDLE1/2, SLEEP1/2 mode)	0: CPU and watchdog timer remain active 1: CPU and watchdog timer are stopped (start IDLE1/2, SLEEP1/2 mode)	
TGHALT	TG control (IDLE0, SLEEP0 mode)	0: Feeding clock to all peripherals from TG 1: Stop feeding clock to peripherals except TBT from TG. (Start IDLE0, SLEEP0 mode)	

Note 1: A reset is applied if both XEN and XTEN are cleared to "0", XEN is cleared to "0" when SYSCK = "0", or XTEN is cleared to "0" when SYSCK = "1".

Note 2: *: Don't care, TG: Timing generator

Note 3: Bits 3, 1and 0 in SYSCR2 are always read as undefined value.

Note 4: Do not set IDLE and TGHALT to "1" simultaneously.

Note 5: Because returning from IDLE0/SLEEP0 to NORMAL1/SLOW1 is executed by the asynchronous internal clock, the period of IDLE0/SLEEP0 mode might be shorter than the period setting by TBTCR<TBTCK>.

Note 6: When IDLE1/2 or SLEEP1/2 mode is released, IDLE is automatically cleared to "0".

Note 7: When IDLE0 or SLEEP0 mode is released, TGHALT is automatically cleared to "0".

Note 8: Before setting TGHALT to "1", be sure to stop peripherals. If peripherals are not stopped, the interrupt latch of peripherals may be set after IDLE0 or SLEEP0 mode is released.

Figure 1.4.7 System Control Registers

1.4.4 Operating Mode Control

(1) STOP mode

STOP mode is controlled by the system control register 1, the $\overline{\text{STOP}}$ pin input and key-on wake-up input (STOP0 to STOP3) which is controlled by the STOP mode release control register (STOPCR).

The $\overline{\text{STOP}}$ pin is also used both as a port P20 and an $\overline{\text{INT5}}$ (External interrupt input 5) pin.

STOP mode is started by setting SYSCR1<STOP> to "1". During STOP mode, the following status is maintained.

a. Oscillations are turned off, and all internal operations are halted.

b. The data memory, registers, the program status word and port output latches are all held in the status in effect before STOP mode was entered.

c. The prescaler and the divider of the timing generator are cleared to "0".

d. The program counter holds the address 2 ahead of the instruction (e.g. [SET (SYSCR1).7]) which started STOP mode.

STOP mode includes a level-sensitive mode and an edge-sensitive mode, either of which can be selected with the SYSCR1<RELM>. Do not use any STOPx (x: 0 to 3) pin input for releasing STOP mode in edge-sensitive mode.

When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait period for stabilizing of the power supply of Flash control circuit is executed after in the STOP warming-up time.
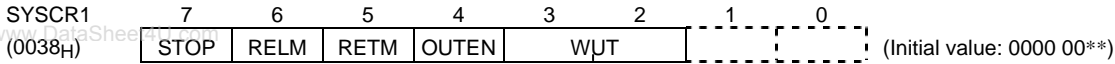
Note 1: The STOP mode can be released by either the STOP or key-on wake-up pin (STOP0 to STOP3). However, because the $\overline{\text{STOP}}$ pin is different from the key-on wake-up and can not inhibit the release input, the $\overline{\text{STOP}}$ pin must be used for releasing STOP mode.

Note 2: During stop period (from start of STOP mode to end of warm-up), due to changes in the external interrupt pin signal, interrupt latches may be set to "1" and interrupts may be accepted immediately after STOP mode is released. Before starting STOP mode, therefore, disable interrupts. Also, before enabling interrupts after STOP mode is released, clear unnecessary interrupt latches.

a. Level-sensitive release mode (RELM = "1")

In this mode, STOP mode is released by setting the $\overline{\text{STOP}}$ pin high or setting the STOPx (x: 0 to 3) pin input which is enabled by STOPCR. This mode is used for capacitor back-up when the main power supply is cut off and long term battery back-up.

When the $\overline{\text{STOP}}$ pin input is high, executing an instruction which starts STOP mode will not place in STOP mode but instead will immediately start the release sequence (Warm-up). Thus, to start STOP mode in the level-sensitive release mode, it is necessary for the program to first confirm that the $\overline{\text{STOP}}$ pin input is low. The following two methods can be used for confirmation.

a. Testing a port P20.

b. Using an external interrupt input $\overline{\text{INT5}}$ ($\overline{\text{INT5}}$ is a falling edge-sensitive input).

Example 1: Starting STOP mode from NORMAL mode by testing a port P20.

```
              LD        (SYSCR1), 01010000B   ;  Sets up the level-sensitive release mode
SSTOPH:       TEST      (P2PRD). 0            ;  Wait until the STOP pin input goes low
                                                 level

              JRS       F, SSTOPH
              SET       (SYSCR1).7            ;  Starts STOP mode
```

Example 2: Starting STOP mode from NORMAL mode with an INT5 interrupt.

```
PINT5:        TEST      (P2PRD). 0            ;  To reject noise, STOP mode does not
                                                 start if port P20 is at high

              JRS       F, SINT5
              LD        (SYSCR1), 01010000B   ;  Sets up the level-sensitive release mode.
              SET       (SYSCR1). 7           ;  Starts STOP mode
SINT5:        RETI
```


Note: When the STOP mode is started with the EEPCR<MNPWDW> = "1", the CPU wait for stabilizing of the power supply of Flash control circuit is executed after in the STOP warming-up time.
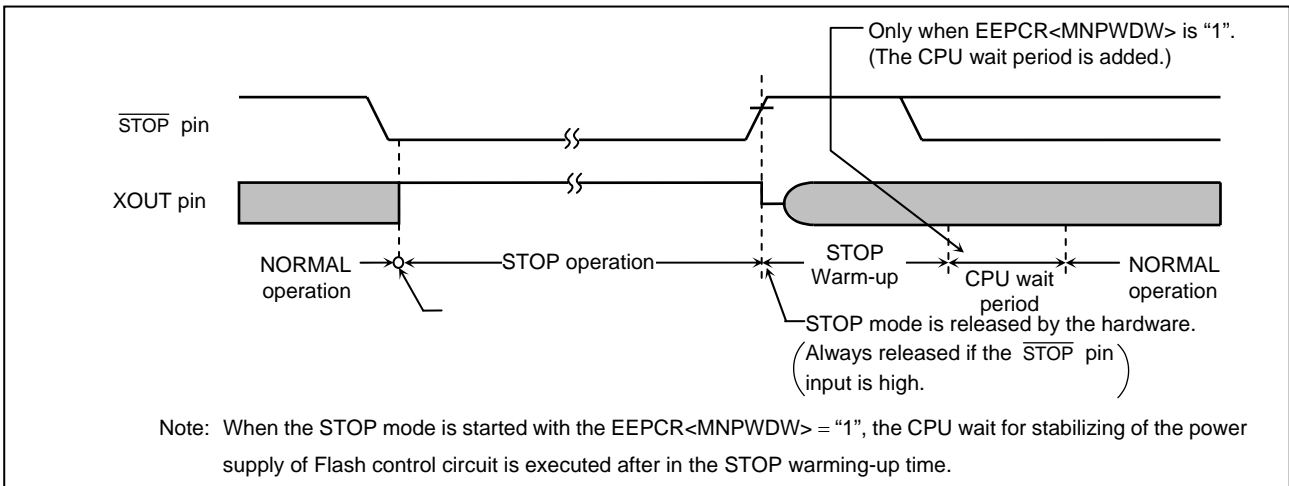
Figure 1.4.8 Level-sensitive Release Mode

Note 1: Even if the $\overline{\text{STOP}}$ pin input is low after warming up start, the STOP mode is not restarted.

Note 2: In this case of changing to the level-sensitive mode from the edge-sensitive mode, the release mode is not switched until a rising edge of the $\overline{\text{STOP}}$ pin input is detected.

b.   Edge-sensitive release mode (RELM = "0")

In this mode, STOP mode is released by a rising edge of the $\overline{\text{STOP}}$ pin input. This is used in applications where a relatively short program is executed repeatedly at periodic intervals. This periodic signal (For example, a clock from a low-power consumption oscillator) is input to the $\overline{\text{STOP}}$ pin. In the edge-sensitive release mode, STOP mode is started even when the $\overline{\text{STOP}}$ pin input is high level. Do not use any STOPx (x: 0 to 3) pin input for releasing STOP mode in edge-sensitive release mode.

Example: Starting STOP mode from NORMAL mode

```
              LD        (SYSCR1), 10010000B   ;  Starts after specified to the edge-sensitive
                                                 release mode
```
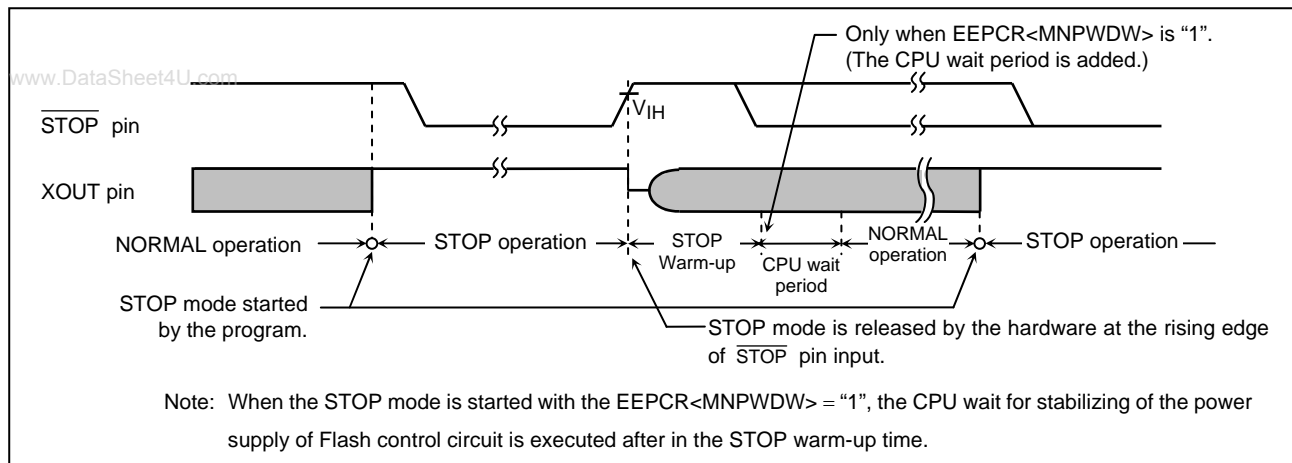
Figure 1.4.9 Edge-sensitive Release Mode

STOP mode is released by the following sequence.

a. In the dual-clock mode, when returning to NORMAL2, both the high-frequency and low-frequency clock oscillators are turned on; when returning to SLOW1 mode, only the low-frequency clock oscillator is turned on. In the single-clock mode, only the high-frequency clock oscillator is turned on.

b. A STOP warm-up period is inserted to allow oscillation time to stabilize. During STOP warm-up, all internal operations remain halted. Four different STOP warm-up times can be selected with the SYSCR1<WUT> in accordance with the resonator characteristics.

c. When the EEPCR<MNPWDW> is "1", the CPU wait period is inserted to stabilize the power supply of Flash control circuit. During CPU wait, though CPU operations remain halted, the peripheral function operation is resumed, and the counting of the timing generator is restarted. After the CPU wait is finished, normal operation resumes with the instruction following the STOP mode start instruction.

d. When the EEPCR<MNPWDW> is "0", normal operation resumes with the instruction following the STOP mode start instruction after the STOP Warm-up.

Note 1: When the STOP mode is released, the start is made after the prescaler and the divider of the timing generator are cleared to "0".

Note 2: STOP mode can also be released by inputting low level on the $\overline{\text{RESET}}$ pin, which immediately performs the normal reset operation.

Note 3: When STOP mode is released with a low hold voltage, the following cautions must be observed.
The power supply voltage must be at the operating voltage level before releasing STOP mode. The $\overline{\text{RESET}}$ pin input must also be "H" level, rising together with the power supply voltage. In this case, if an external time constant circuit has been connected, the $\overline{\text{RESET}}$ pin input voltage will increase at a slower pace than the power supply voltage. At this time, there is a danger that a reset may occur if input voltage level of the $\overline{\text{RESET}}$ pin drops below the non-inverting high-level input voltage (Hysteresis input).

Table 1.4.1 Warm-up Time Example (at fc = 16.0 MHz, fs = 32.768 kHz)

WUT	Warm-up Time [ms] (Note 2)	
	Return to NORMAL Mode	Return to SLOW Mode
00	12.288 + (0.064)	750 + (0.244)
01	4.096 + (0.064)	250 + (0.244)
10	3.072 + (0.064)	5.85 + (0.244)
11	1.024 + (0.064)	1.95 + (0.244)

Note 1: The warm-up time is obtained by dividing the basic clock by the divider: therefore, the warm-up time may include a certain amount of error if there is any fluctuation of the oscillation frequency when STOP mode is released. Thus, the warm-up time must be considered an approximate value.

Note 2: The CPU wait period for FLASH is shown in parentheses.
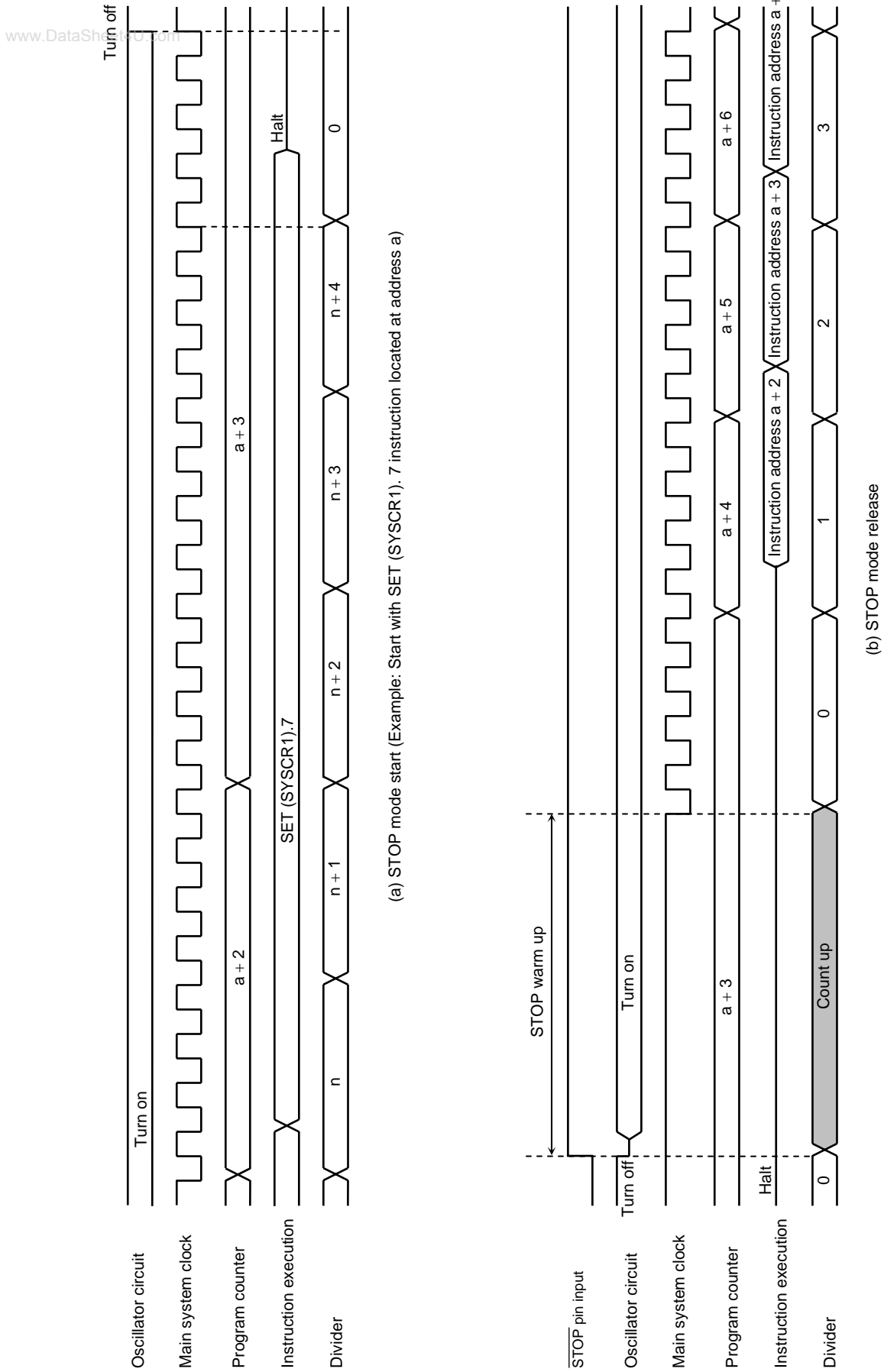
Figure 1.4.10 STOP Mode Start/Release (When EEPCR<MNPWDW> = "0")
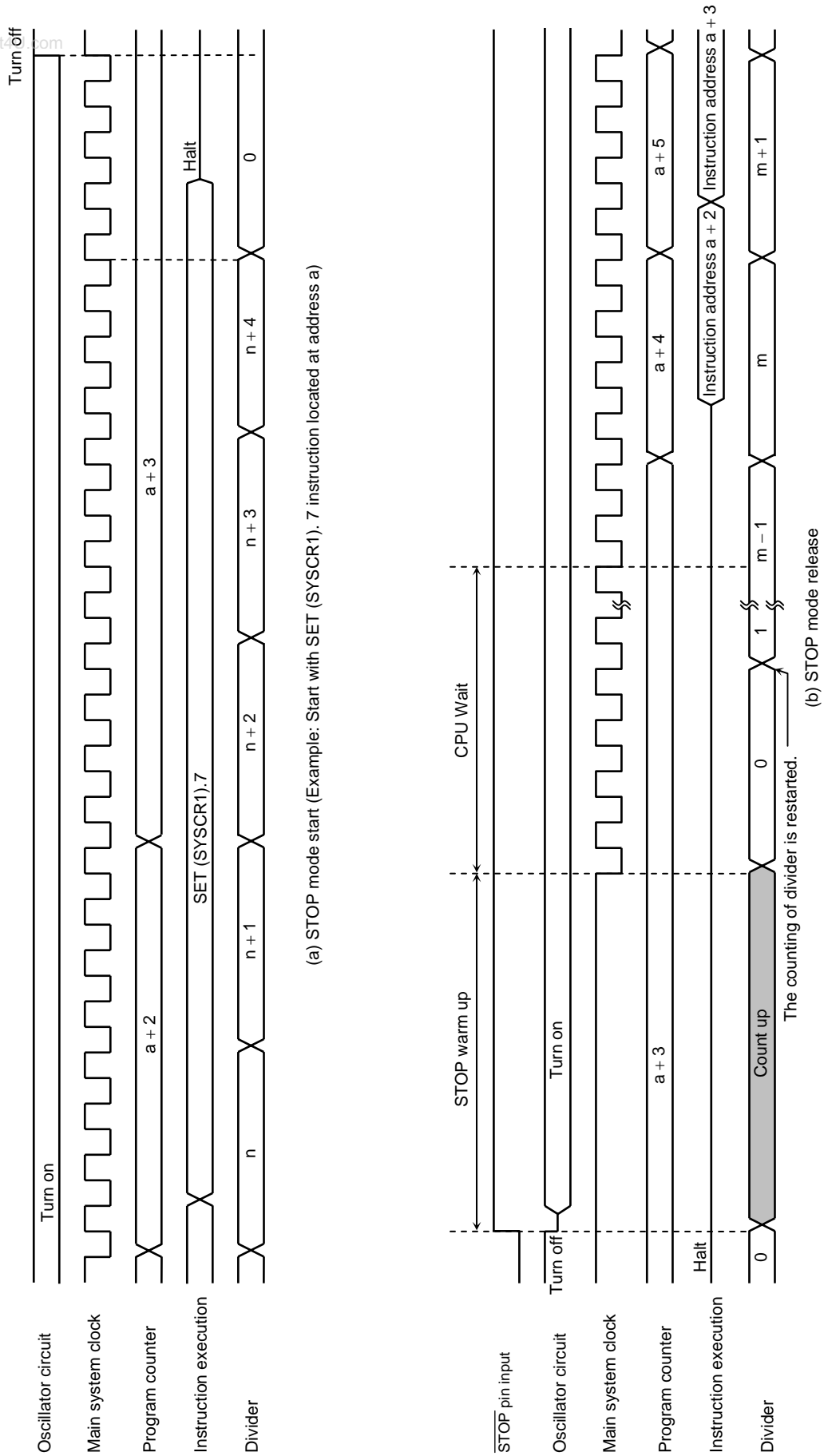
Figure 1.4.11 STOP Mode Start/Release (When EEPCR<MNPWDW> = "1")

(2) IDLE1/2 mode, SLEEP1/2 mode

IDLE1/2 and SLEEP1/2 modes are controlled by the system control register 2 (SYSCR2) and maskable interrupts. The following status is maintained during these modes.

a. Operation of the CPU and watchdog timer (WDT) is halted. On-chip peripherals continue to operate.

b. The data memory, CPU registers, program status word and port output latches are all held in the status in effect before these modes were entered.

c. The program counter holds the address 2 ahead of the instruction which starts these modes.

Note 1: EEPCR<ATPWDW> is a bit1 in EEPCR, which is a control bit of the power supply circuit for flash.

Note 2: During CPU wait, though CPU operations remain halted, the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.
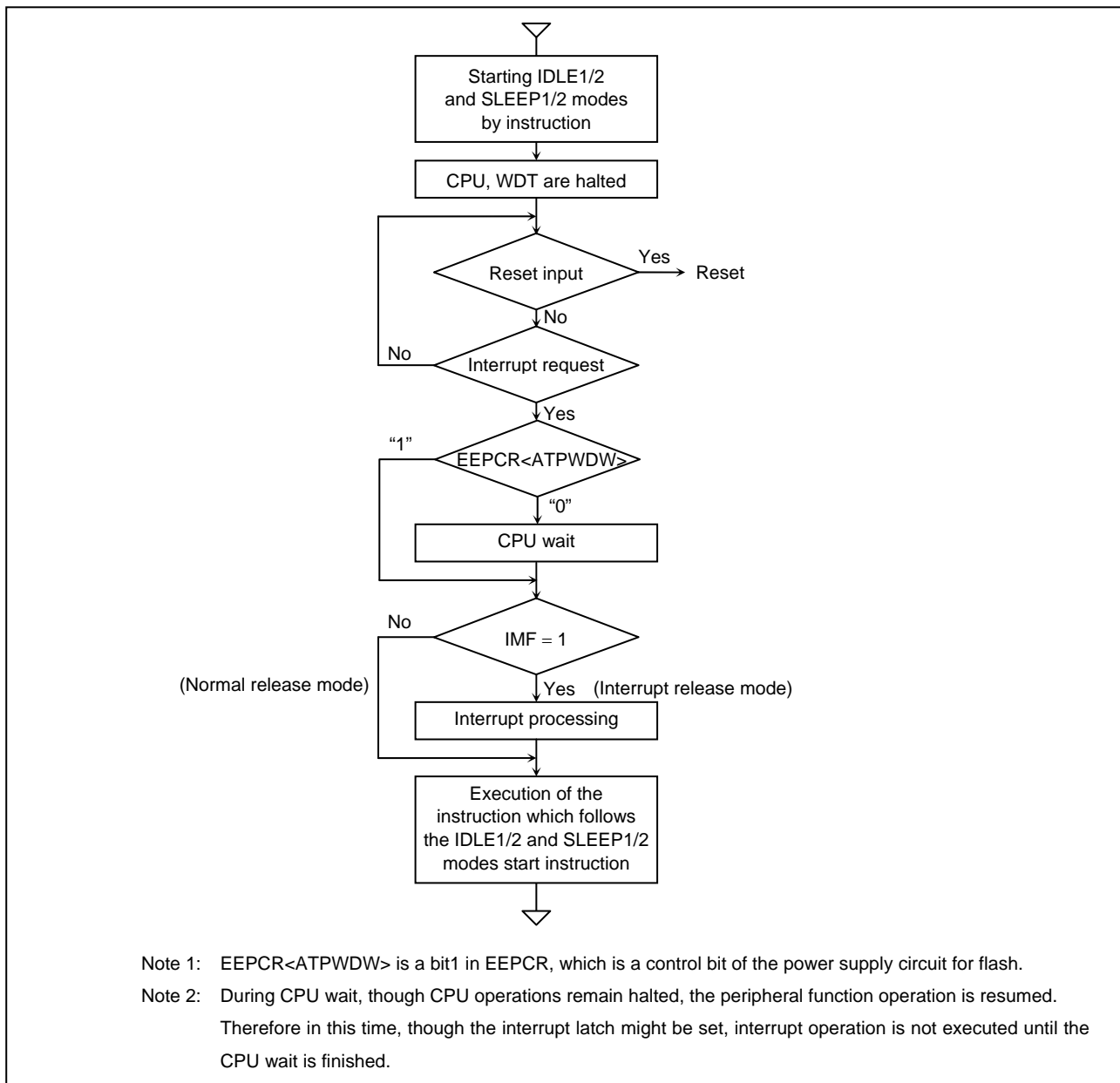
Figure 1.4.12 IDLE1/2, SLEEP1/2 Modes

- Start the IDLE1/2 and SLEEP1/2 modes

    When IDLE1/2 and SLEEP1/2 modes start, set SYSCR2<IDLE> to "1".

- Release the IDLE1/2 and SLEEP1/2 modes

    IDLE1/2 and SLEEP1/2 modes include a normal release mode and an interrupt release mode. These modes are selected by interrupt master enable flag (IMF).

    After releasing IDLE1/2 and SLEEP1/2 modes, the SYSCR2<IDLE> is automatically cleared to "0" and the operation mode is returned to the mode preceding IDLE1/2 and SLEEP1/2 modes.

    When the IDLE1/2 and SLEEP1/2 modes are started with the EEPCR<ATPWDW> = "0", the CPU wait period for stabilizing of the power supply of Flash control circuit is added before the operation mode is returned to the preceding modes. The CPU wait time of IDLE1/2 is $2^{10}/fc$ [s] and that of SLEEP1/2 mode is $2^3/fs$ [s].

    IDLE1/2 and SLEEP1/2 modes can also be released by inputting low level on the $\overline{\text{RESET}}$ pin. After releasing reset, the operation mode is started from NORMAL1 mode.

    Note: During CPU wait, though CPU operations remain halted, but the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.

(a) Normal release mode (IMF = "0")

    IDLE1/2 and SLEEP1/2 modes are released by any interrupt source enabled by the individual interrupt enable flag (EF). After the interrupt is generated, the program operation is resumed from the instruction following the IDLE1/2 and SLEEP1/2 modes start instruction. Normally, the interrupt latches (IL) of the interrupt source used for releasing must be cleared to "0" by load instructions.

(b) Interrupt release mode (IMF = "1")

    IDLE1/2 and SLEEP1/2 modes are released by any interrupt source enabled with the individual interrupt enable flag (EF). After the interrupt is processed, the program operation is resumed from the instruction following the instruction, which starts IDLE1/2 and SLEEP1/2 modes.

    Note: When a watchdog timer interrupts is generated immediately before IDLE1/2 and SLEEP1/2 mode are started, the watchdog timer interrupt will be processed but IDLE1/2 and SLEEP1/2 mode will not be started.

(a) IDLE1/2, SLEEP1/2 mode start (Example: starting with the SET instruction located at address a)

Main system clock

Interrupt request

Program counter a + 2 a + 3

Instruction execution SET (SYSCR2).4 Halt

Watchdog timer Operate

(1) Normal release mode (EEPCR<ATPWDW> = "1")

Main system clock

Interrupt request

Program counter a + 3 a + 4 Instruction address a + 2

Instruction execution Halt

Watchdog timer Halt Operate

(2) Interrupt release mode

Main system clock

Interrupt request

Program counter a + 3 Acceptance of interrupt

Instruction execution Halt

Watchdog timer Halt Operate

(b) IDLE1/2, SLEEP1/2 mode release (EEPCR<ATPWDW> = "1")

Figure 1.4.13 IDLE1/2, SLEEP1/2 Mode Start/Release

(3) IDLE0, SLEEP0 mode (IDLE0, SLEEP0)

IDLE0 and SLEEP0 modes are controlled by the system control register 2 (SYSCR2) and the time base timer control register (TBTCR). The following status is maintained during IDLE0 and SLEEP0 modes.

a.   Timing generator stops feeding clock to peripherals except TBT.

b.   The data memory, CPU registers, program status word and port output latches are all held in the status in effect before IDLE0 and SLEEP0 modes were entered.

c.   The program counter holds the address 2 ahead of the instruction which starts IDLE0 and SLEEP0 modes.

Note: Before starting IDLE0 or SLEEP0 mode, be sure to stop (Disable) periperals.

Note 1: EEPCR<ATPWDW> is a bit1 in EEPCR, which is a control bit of the power supply circuit for Flash.

Note 2: During CPU wait, though CPU operations remain halted, but the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.

Figure 1.4.14 IDLE0, SLEEP0 Mode

www.DataSheet4U.com

- Start the IDLE0 and SLEEP0 modes

  Stop (Disable) peripherals such as a timer counter.

  When IDLE0 and SLEEP0 modes start, set SYSCR2<TGHALT> to "1".

- Release the IDLE0 and SLEEP modes

  IDLE0 and SLEEP0 modes include a normal release mode and an interrupt release mode.

  These modes are selected by interrupt master flag (IMF), individual interrupt enable-flag (EF7) for INTTBT and TBTCR<TBTEN>.

  After releasing IDLE0 and SLEEP0 modes, the SYSCR2<TGHALT> is automatically cleared to "0" and the operation mode is returned to the mode preceding IDLE0 and SLEEP0 modes. Before starting the IDLE0 or SLEEP0 mode, when the TBTCR<TBTEN> is set to "1", INTTBT interrupt latch is set to "1".

  When the IDLE0 and SLEEP0 modes are started with the EEPCR<ATPWDW> = "0", the CPU wait period for stabilizing of the power supply of Flash control circuit is added before the operation mode is returned to the preceding modes. The CPU wait time of IDLE0 is $2^{10}/fc$ [s] and that of SLEEP0 mode is $2^3/fs$ [s].

  IDLE0 and SLEEP0 modes can also be released by inputting low level on the $\overline{\text{RESET}}$ pin. After releasing reset, the operation mode is started from NORMAL1 mode.

  Note 1: IDLE0 and SLEEP0 modes start/release without reference to TBTCR<TBTEN> setting.

  Note 2: During CPU wait, though CPU operations remain halted, but the peripheral function operation is resumed. Therefore in this time, though the interrupt latch might be set, interrupt operation is not executed until the CPU wait is finished.

a. Normal release mode (IMF・EF7・TBTCR<TBTEN> = "0")

  IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK>. After the falling edge is detected, the program operation is resumed from the instruction following the IDLE0 and SLEEP0 modes start instruction.

b. Interrupt release mode (IMF・EF7・TBTCR<TBTEN> = "1")

  IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK> and INTTBT interrupt processing is started.

  Note 1: Because returning from IDLE0, SLEEP0 to NORMAL1, SLOW1 is executed by the asynchronous internal clock, the period of IDLE0, SLEEP0 mode might be the shorter than the period setting by TBTCR<TBTCK>.

  Note 2: When a watchdog timer interrupt is generated immediately before IDLE0/SLEEP0 mode is started, the watchdog timer interrupt will be processed but IDLE0/SLEEP0 mode will not be started.

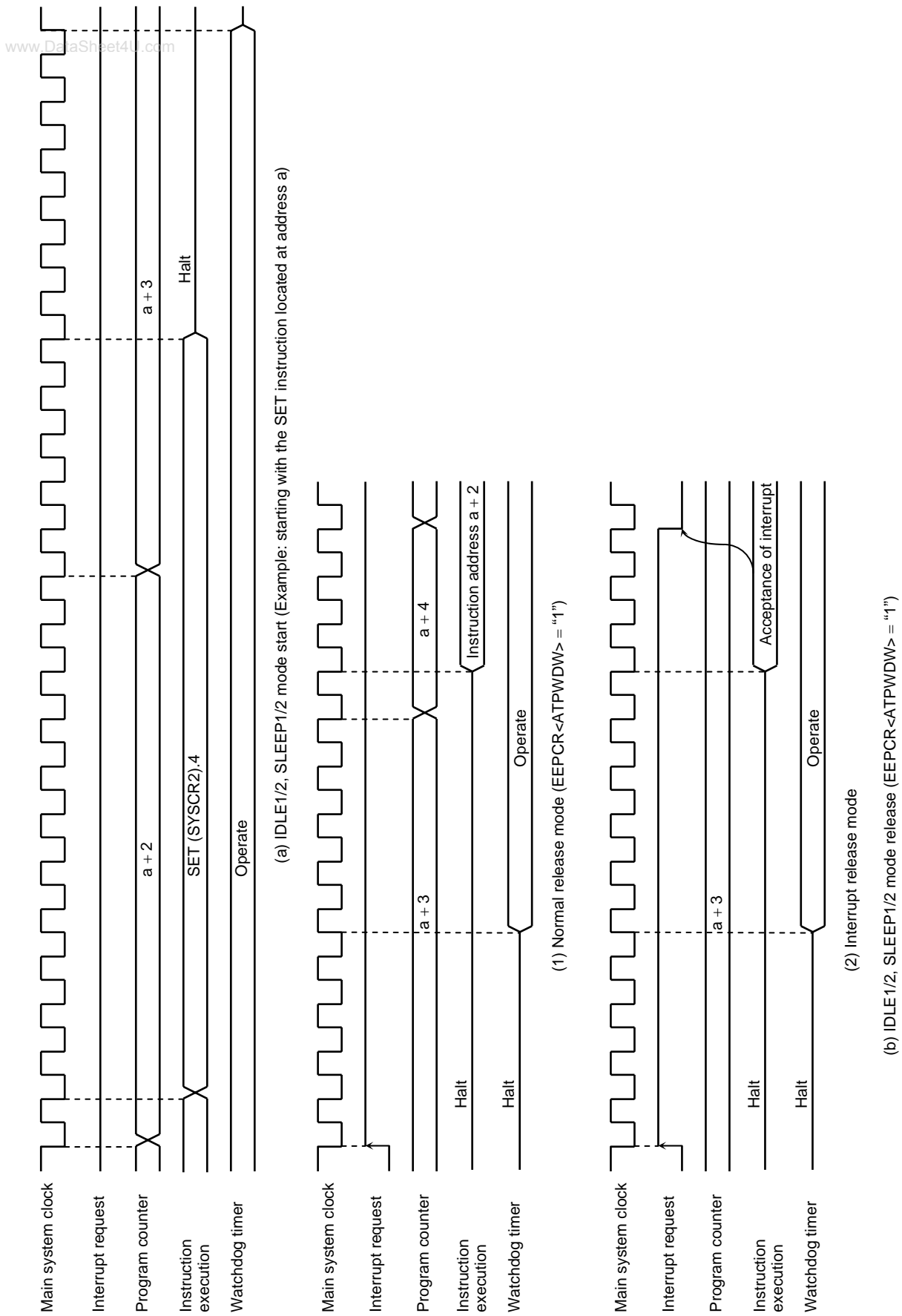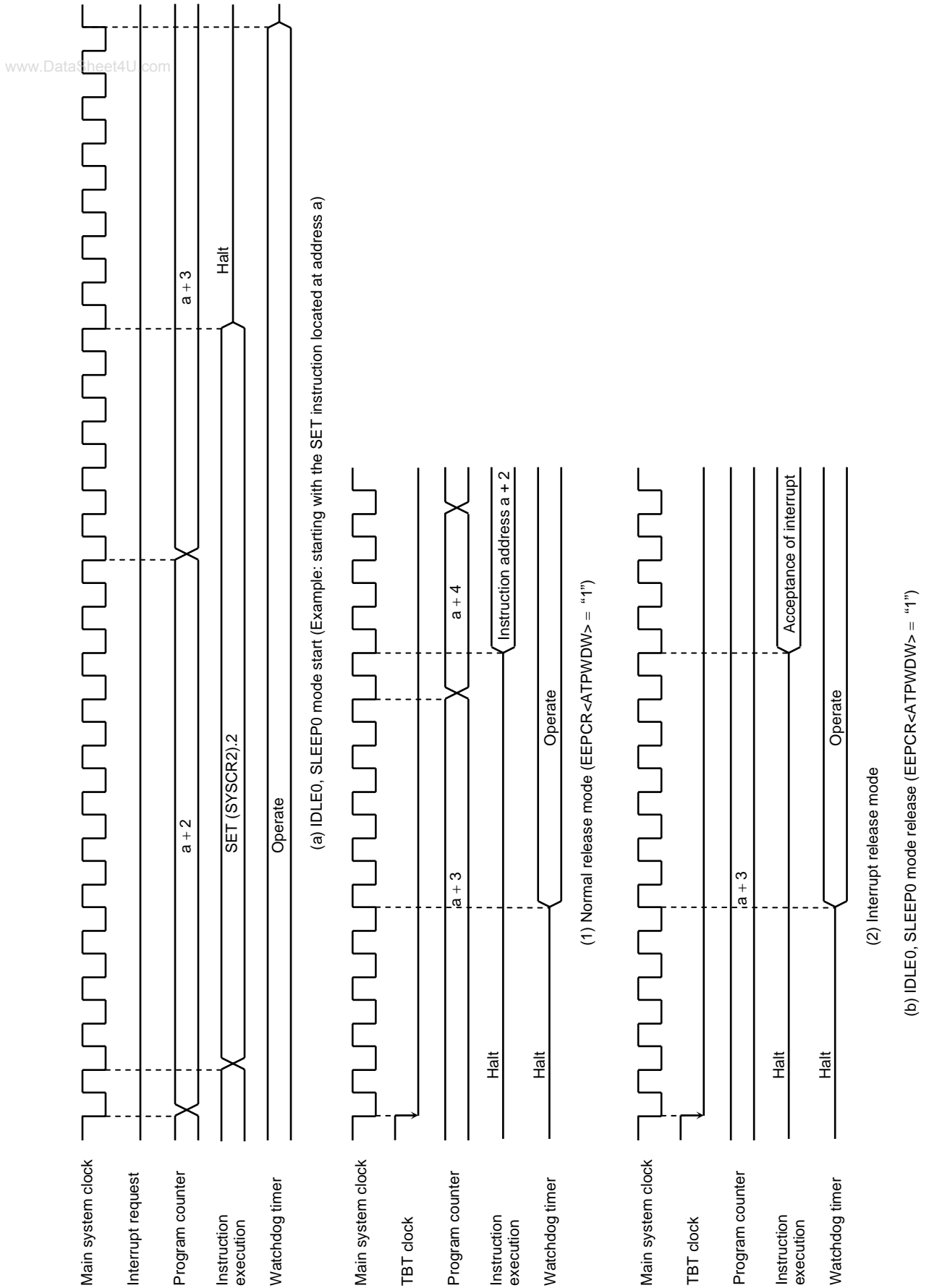Figure 1.4.15 IDLE0, SLEEP0 Mode Start/Release

(4) SLOW mode

SLOW mode is controlled by the system control register 2 (SYSCR2).

The following is the methods to switch the mode with the warm-up counter (TC2).

a. Switching from NORMAL2 mode to SLOW1 mode

First, set SYSCR2<SYSCK> to switch the main system clock to the low-frequency clock for SLOW2 mode.

Next, clear SYSCR2<XEN> to turn off high-frequency oscillation.

Note: The high-frequency clock oscillation can be continued to return quickly to NORMAL2 mode. But starting STOP mode while SLOW mode, the high-frequency oscillation must be stopped.

When the low-frequency clock oscillation is unstable, wait until oscillation stabilizes before performing the above operations. The timer/counter 2 (TC2) can conveniently be used to confirm that low-frequency clock oscillation has stabilized.

```
Example 1: Switching from NORMAL2 mode to SLOW1 mode.
            SET     (SYSCR2). 5      ;  SYSCR2<SYSCK> ← 1
                                        (Switches the main system clock to the
                                        low-frequency clock for SLOW2)
            CLR     (SYSCR2). 7      ;  SYSCR2<XEN> ← 0
                                        (Turns off high-frequency oscillation)

Example2: Switching to the SLOW1 mode after low-frequency clock has stabilized.
            SET     (SYSCR2). 6      ;  SYSCR2<XTEN> ← 1
            LD      (TC2CR), 14H     ;  Sets mode for TC2
            LDW     (TC2DRL), 8000H  ;  Sets warm-up time
                                        (Depend on oscillator accompanied)
            DI                       ;  IMF ← 0
            SET     (EIRE). 4        ;  Enables INTTC2
            EI                       ;  IMF ← 1
            SET     (TC2CR). 5       ;  Starts TC2
              :
PINTTC2:    CLR     (TC2CR). 5       ;  Stops TC2
            SET     (SYSCR2). 5      ;  SYSCR2<SYSCK> ← 1
                                        (Switches the main system clock to the
                                        low-frequency clock)
            CLR     (SYSCR2). 7      ;  SYSCR2<XEN> ← 0
                                        (Turns off high-frequency oscillation)
            RETI
              :
VINTTC2:    DW      PINTTC2          ;  INTTC2 vector table
```

b. Switching from SLOW1 mode to NORMAL2 mode

First, set SYSCR2<XEN> to turn on the high-frequency oscillation. When time for stabilization (Warm-up) has been taken by the timer/counter 2 (TC2), clear SYSCR2<SYSCK> to switch the main system clock to the high-frequency clock.

Note 1: After SYSCK is cleared to "0", executing the instructions is continued by the low-frequency clock for the period synchronized with low-frequency and high-frequency clocks.

High-frequency clock
Low-frequency clock
Main system clock
SYSCK
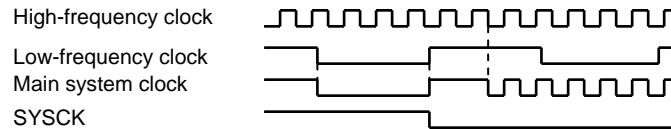
Note 2: SLOW mode can also be released by inputting low level on the $\overline{\text{RESET}}$ pin, which immediately performs the reset operation. After reset, the TMP86FM48 is placed in NORMAL1 mode.

Example: Switching from the SLOW1 mode to the NORMAL2 mode
(fc = 16 MHz, warm-up time is = 4.0 ms).

```
               SET     (SYSCR2). 7      ;  SYSCR2<XEN> ← 1
                                           (Starts high-frequency oscillation)
               LD      (TC2CR), 10H     ;  Sets mode for TC2
                                           (Timer mode, fc for source)
               LD      (TC2DRH), 0F8H   ;  Sets warm-up time
                                           (Depend on oscillator accompanied)
               DI                       ;  IMF ← 0
               SET     (EIRE). 4        ;  Enables INTTC2
               EI                       ;  IMF ← 1
               SET     (TC2CR). 5       ;  Starts TC2

PINTTC2:       CLR     (TC2CR). 5       ;  Stops TC2
               CLR     (SYSCR2). 5      ;  SYSCR2<SYSCK> ← 0
                                           (Switches the main system clock to the
                                           high-frequency clock)

               RETI

VINTTC2:       DW      PINTTC2          ;  INTTC2 vector table
```
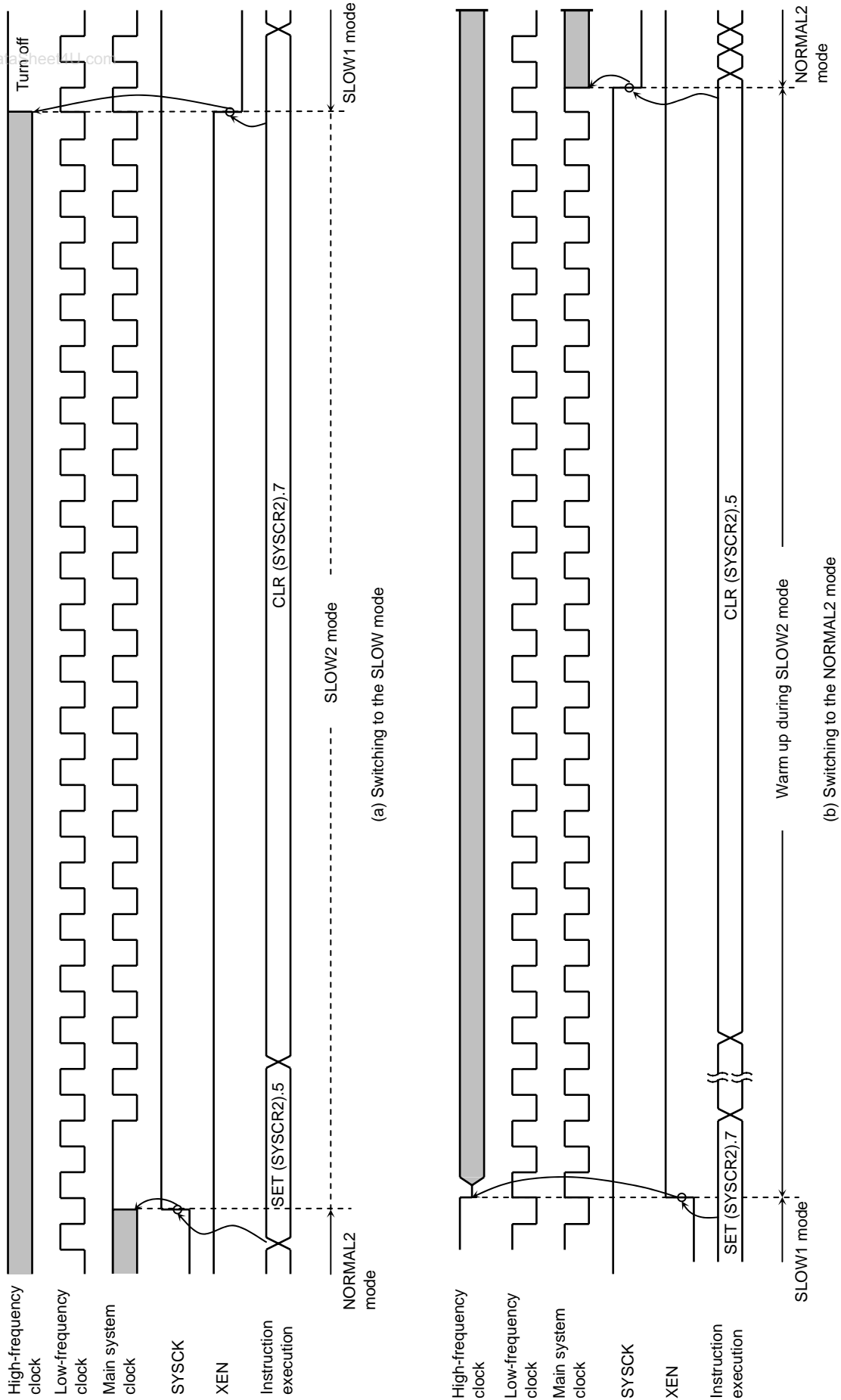
Figure 1.4.16 Switching between the NORMAL2 and SLOW Modes

1.5 Interrupt Control Circuit

The TMP86FM48 has a total (Reset is excluded) of 20 interrupt source: 5 externals and 15 internals. 4 of the internal sources are non-maskable interrupts, and the rest of them are maskable interrupts.

Interrupt sources are provided with interrupt latches (IL), which hold interrupt requests, and independent vectors. The interrupt latch is set to "1" by the generation of its interrupt request which requests the CPU to accept its interrupts. Interrupts are enabled or disabled by software using the interrupt master enable flag (IMF) and interrupt enable flag (EF). If more than one interrupts are generated simultaneously, interrupts are accepted in order which is dominated by hardware. However, there are no prioritized interrupt factors among non-maskable interrupts.

Table 1.5.1 Interrupt Sources

Interrupt Factors		Enable Condition	Interrupt Latch	Vector Address	Priority	
Internal/External	(Reset)	Non-maskable	–	$FFFE_H$	High	1
Internal	INTSWI (Software interrupt)	Non-maskable	–	$FFFC_H$		2
Internal	INTUNDEF (Executed the undefined instruction interrupt)	Non-maskable	–	$FFFC_H$		2
Internal	INTATRAP (Address trap interrupt)	Non-maskable	IL_2	$FFFA_H$		2
Internal	INTWDT (Watchdog timer interrupt)	Non-maskable	IL_3	$FFF8_H$		2
External	$\overline{INT0}$ (External interrupt 0)	$IMF \cdot EF_4 = 1$	IL_4	$FFF6_H$		5
Internal	INTTC1 (TC1 interrupt)	$IMF \cdot EF_5 = 1$	IL_5	$FFF4_H$		6
External	INT1 (External interrupt 1)	$IMF \cdot EF_6 = 1$	IL_6	$FFF2_H$		7
Internal	INTTBT (Time base timer interrupt)	$IMF \cdot EF_7 = 1$	IL_7	$FFF0_H$		8
External	INT2 (External interrupt 2)	$IMF \cdot EF_8 = 1$	IL_8	$FFEE_H$		9
Internal	INTTC3 (TC3 interrupt)	$IMF \cdot EF_9 = 1$	IL_9	$FFEC_H$		10
Internal	INTSIO1 (Serial interface 1 interrupt)	$IMF \cdot EF_{10} = 1$	IL_{10}	$FFEA_H$		11
Internal	INTSIO2 (Serial interface 2 interrupt)	$IMF \cdot EF_{11} = 1$	IL_{11}	$FFE8_H$		12
Internal	INTTC5 (TC5 interrupt)	$IMF \cdot EF_{12} = 1$	IL_{12}	$FFE6_H$		13
External	INT3 (External interrupt 3)	$IMF \cdot EF_{13} = 1$	IL_{13}	$FFE4_H$		14
Internal	INTADC (AD converter interrupt)	$IMF \cdot EF_{14} = 1$	IL_{14}	$FFE2_H$		15
	Reserved	$IMF \cdot EF_{15} = 1$	IL_{15}	$FFE0_H$		16
	Reserved	$IMF \cdot EF_{16} = 1$	IL_{16}	$FFBE_H$		17
Internal	INTSBI (Serial bus interface interrupt)	$IMF \cdot EF_{17} = 1$	IL_{17}	$FFBC_H$		18
Internal	INTRXD (UART received interrupt)	$IMF \cdot EF_{18} = 1$	IL_{18}	$FFBA_H$		19
Internal	INTTXD (UART transmitted interrupt)	$IMF \cdot EF_{19} = 1$	IL_{19}	$FFB8_H$		20
Internal	INTTC2 (TC2 interrupt)	$IMF \cdot EF_{20} = 1$	IL_{20}	$FFB6_H$		21
External	$\overline{INT5}$ (External interrupt 5)	$IMF \cdot EF_{21} = 1$	IL_{21}	$FFB4_H$		22
	Reserved	$IMF \cdot EF_{22} = 1$	IL_{22}	$FFB2_H$		23
	Reserved	$IMF \cdot EF_{23} = 1$	IL_{23}	$FFB0_H$		24

Note 1: To use the watchdog timer interrupt (INTWDT), clear WDTCR1<WDTOUT> to "0" (It is set for the "Reset request" after reset is released). For details, see 2.4 Watchdog Timer.

Note 2: To use the address trap interrupt (INTATRAP), clear WDTCR1<ATOUT> to "0" (It is set for the "Reset request" after reset is released). For details, see 2.4.5 Address Trap.

www.DataSheet4U.com
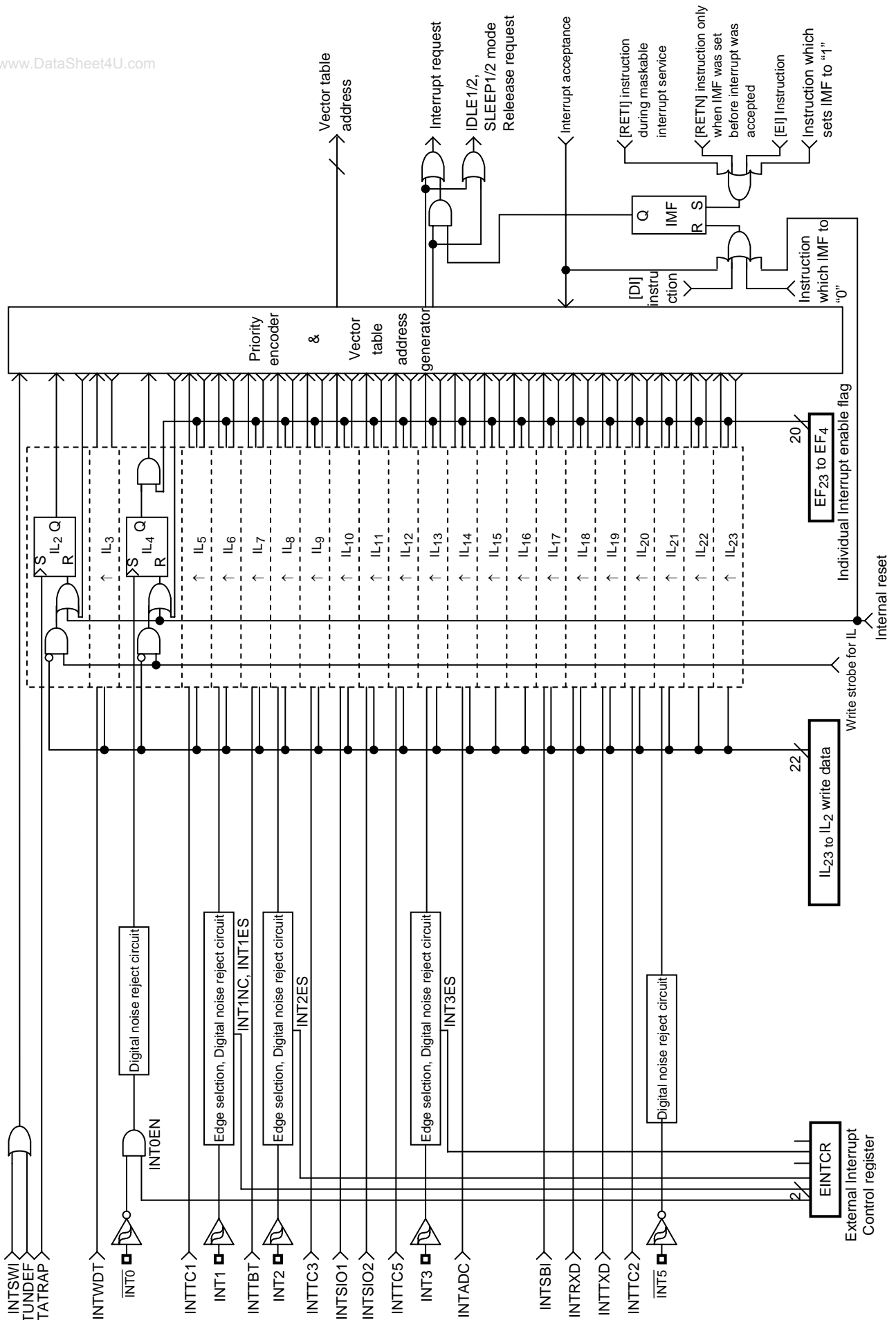
Figure 1.5.1 Interrupt Controller Block Diagram

(1) Interrupt latches (IL_{24} to IL_2)

An interrupt latch is provided for each interrupt source, except for a software interrupt. When interrupt request is generated, the latch is set to "1", and the CPU is requested to accept the interrupt if its interrupt is enabled. All interrupt latches are initialized to "0" during reset.

The interrupt latches are located on address $002E_H$, $003C_H$ and $003D_H$ in SFR area. Except for IL_3 and IL_2, each latch can be cleared to "0" individually by instruction. (However, the read-modify-write instructions such as bit manipulation or operation instructions cannot be used. Interrupt request would be cleared inadequately if interrupt is requested while such instructions are executed.) Thus interrupt request can be canceled/initialized by software.

Interrupt latches are not set to "1" by an instruction. Since interrupt latches can be read, the status for interrupt requests can be monitored by software.

Note: When manipulating IL, clear IMF (to disable interrupts) beforehand.

```
Example 1: Clears interrupt latches
          DI                                      ;   IMF ← 0
          LD        (ILE), 11110011B              ;   IL19, IL18 ← 0
          LDW       (ILL), 1110100000111111B      ;   IL12, IL10 to IL6 ← 0
          EI                                      ;   IMF ← 1
Example 2: Reads interrupt latches
          LD        WA, (ILL)                     ;   W ← ILH, A ← ILL
Example 3: Tests an interrupt latches
          TEST      (IL).7                        ;   IL7 = 1 then jump
          JR        F, SSET
```

(2) Interrupt enable register (EIR)

The interrupt enable register (EIR) enables and disables the acceptance of interrupts, except for the non-maskable interrupts (Software interrupt, undefined instruction interrupt, address trap interrupt and watchdog interrupt). Non-maskable interrupt is accepted regardless of the contents of the EIR.

The EIR consists of an interrupt master enable flag (IMF) and the individual interrupt enable flags (EF). These registers are located on address $002C_H$, $003A_H$ and $003B_H$ in SFR area, and they can be read and written by an instructions (Including read-modify-write instructions such as bit manipulation or operation instructions).

a. Interrupt master enable flag (IMF)

The interrupt enable register (IMF) enables and disables the acceptance of the whole maskable-interrupt. While IMF = "0", all maskable interrupts are not accepted regardless of the status on each individual interrupt enable flag (EF). By setting IMF to "1", the interrupt becomes acceptable if the individuals are enabled. When an interrupt is accepted, IMF is cleared to "0" after the latest status on IMF is stacked. Thus the maskable interrupts which follow are disabled. By executing return interrupt instruction [RETI/RETN], the stacked data, which was the status before interrupt acceptance, is loaded on IMF again.

The IMF is located on bit0 in EIRL (Address: $003A_H$ in SFR), and can be read and written by an instruction. The IMF is normally set and cleared by [EI] and [DI] instruction respectively. During reset, the IMF is initialized to "0", and maskable interrupts are not accepted until it is set to "1".

b. Individual interrupt enable flags (EF_{23} to EF_4)

Each of these flags enables and disables the acceptance of its maskable interrupt. Setting the corresponding bit of an individual interrupt enable flag to "1" enables acceptance of its interrupt, and setting the bit to "0" disables acceptance. The individual interrupt enable flags (EF_{23} to EF_4) are located on EIRE, EIRL to EIRH (address: $002C_H$, $003A_H$ to $003B_H$ in SFR), and can be read and written by an instruction. During reset, all the individual interrupt enable flags (EF_{23} to EF_4) are initialized to "0" and all maskable interrupts are not accepted until they are set to "1".

Note: Before manipulating EF, be sure to clear IMF (Interrupt disabled). Then set IMF newly again after operating on the interrupt enables flag (EF). Normally, IMF is clear to "0" automatically on service routine. When IMF is set to "1" for using a multiple interrupt on service routine, be sure to process as is the case with EF.

Example 1: Enables interrupts individually and sets IMF

```
        DI                                  ;   IMF ← 0
        LD          (EIRE), 00001100B       ;   EF19, EF18 ← "1"
        LDW         (EIRL),                 ;   EF14, EF13, EF11, EF7, EF5 ← "1"
                    0110100010100000B           Note: IMF is not set.
        EI                                  ;   IMF ← "1"
```

Example 2: C compiler description example

```
        unsigned int _io (3AH) EIRL;        ;   /* 3AH shows EIRL address */
        _DI ( );
        EIRL = 10100000B;

        _EI ( );
```

Interrupt Latches

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ILH, ILL (003C$_H$, 003D$_H$)	IL$_{15}$	IL$_{14}$	IL$_{13}$	IL$_{12}$	IL$_{11}$	IL$_{10}$	IL$_9$	IL$_8$	IL$_7$	IL$_6$	IL$_5$	IL$_4$	IL$_3$	IL$_2$		

IL$_H$ (003D$_H$)   IL$_L$ (003C$_H$)

(Initial value: 00000000 000000**)

	23	22	21	20	19	18	17	16
ILE (002E$_H$)	IL$_{23}$	IL$_{22}$	IL$_{21}$	IL$_{20}$	IL$_{19}$	IL$_{18}$	IL$_{17}$	IL$_{16}$

IL$_E$ (002E$_H$)

(Initial value: 00000000)

IL$_{23}$ to IL$_2$	Interrupt Latches	at RD 0: No interrupt request 1: Interrupt request	at WR Clears the interrupt request (Note 1) (Interrupt Latch is not set.)	R/W

Note 1: IL$_2$ and IL$_3$ are prohibited from clearing.

Note 2: When manipulating IL, clear IMF (to disable interrupts) beforehand.

Note 3: Do not clear IL with read-modify-write instructions such as bit operations.

Interrupt Enable Registers

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIRH, EIRL (003A$_H$, 003B$_H$)	EF$_{15}$	EF$_{14}$	EF$_{13}$	EF$_{12}$	EF$_{11}$	EF$_{10}$	EF$_9$	EF$_8$	EF$_7$	EF$_6$	EF$_5$	EF$_4$				IMF

EIR$_H$ (003B$_H$)   EIR$_L$ (003A$_H$)

(Initial value: 00000000 00000***0)

	23	22	21	20	19	18	17	16
EIRE (002C$_H$)	EF$_{23}$	EF$_{22}$	EF$_{21}$	EF$_{20}$	EF$_{19}$	EF$_{18}$	EF$_{17}$	EF$_{16}$

EIR$_E$ (002C$_H$)

(Initial value: 00000000)

EF$_{23}$ to EF$_4$	Individual-interrupt enable flag (specified for each bit)	0: Disable the acceptance of each maskable interrupt. 1: Enable the acceptance of each maskable interrupt.	R/W
IMF	Interrupt master enable flag	0: Disable the acceptance of all maskable interrupts. 1: Enable the acceptance of all maskable interrupts.	

Note 1: *: Don't care

Note 2: When manipulating EF, clear IMF (to disable interrupts) beforehand.

Note 3: Do not set IMF to 1 simultaneously with EF15 to EF4.

Figure 1.5.2 Interrupt Latch (IL), Interrupt Enable Registers (EIR)

1.5.1 Interrupt Sequence

An interrupt request, which raised interrupt latch, is held, until interrupt is accepted or interrupt latch is cleared to "0" by resetting or an instruction. Interrupt acceptance sequence requires 8-machine cycles (4 μs at 8.0 MHz) after the completion of the current instruction. The interrupt service task terminates upon execution of an interrupt return instruction [RETI] (for maskable interrupts) or [RETN] (for non-maskable interrupts). Figure 1.5.3 shows the timing chart of interrupt acceptance processing.

（1） Interrupt acceptance processing is packaged as follows.

1. The interrupt master enable flag (IMF) is cleared to "0" in order to disable the acceptance of any following interrupt.

2. The interrupt latch (IL) for the interrupt source accepted is cleared to "0".

3. The contents of the program counter (PC) and the program status word, including the interrupt master enable flag (IMF), are saved (Pushed) on the stack in sequence of PSW + IMF, PCH, PCL. Meanwhile, the stack pointer (SP) is decremented by 3.

4. The entry address (Interrupt vector) of the corresponding interrupt service program, loaded on the vector table, is transferred to the program counter.

5. The instruction stored at the entry address of the interrupt service program is executed.

Note: When the contents of PSW are saved on the stack, the contents of IMF are also saved.

Note 1: a: return address entry address, b: entry address, c: address which RETI instructrion is stored

Note 2: On condition that interrupt is enabled, it takes 38/fc [s] or 38/fs [s] at maximum (If the interrupt latch is set at the first machine cycle on 10 cycle instruction) to start interrupt acceptance processing since its interrupt latch is set.
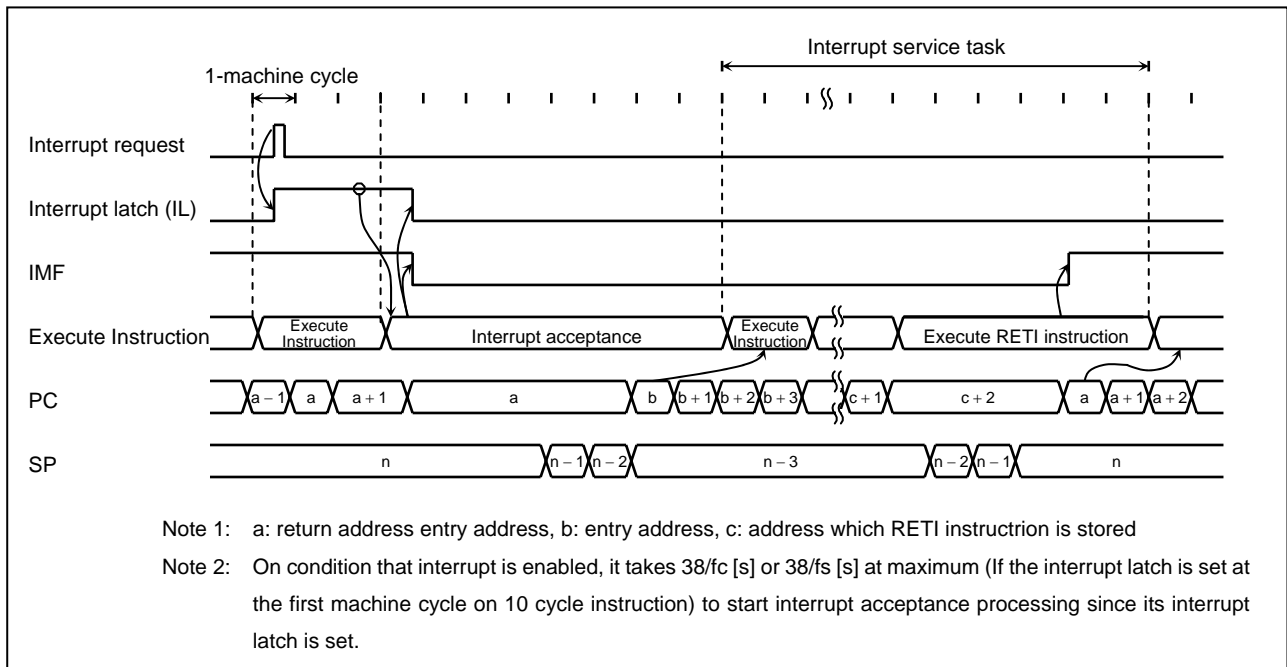
Figure 1.5.3 Timing Chart of Interrupt Acceptance/Return Interrupt Instruction

Example: Correspondence between vector table address for INTTBT and the entry address of the interrupt service program

Vector table address

FFF0H	03H
FFF1H	D2H

Vector

Entry address

D203H	0FH
D204H	06H

Interrupt service program
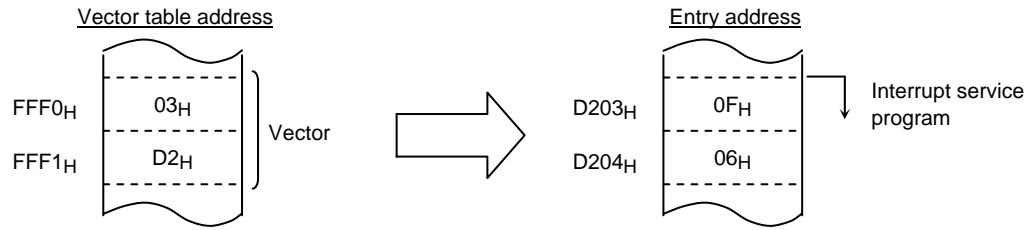
A maskable interrupt is not accepted until the IMF is set to "1" even if the maskable interrupt higher than the level of current servicing interrupt is requested.

In order to utilize nested interrupt service, the IMF is set to "1" in the interrupt service program. In this case, acceptable interrupt sources are selectively enabled by the individual interrupt enable flags.

To avoid overloaded nesting, clear the individual interrupt enable flag whose interrupt is currently serviced, before setting IMF to "1". As for non-maskable interrupt, keep interrupt service shorter compared with length between interrupt requests; otherwise the status cannot be recovered as non-maskable interrupt would simply nested.

(2) Saving/restoring general-purpose registers

During interrupt acceptance processing, the program counter (PC) and the program status word (PSW, includes IMF) are automatically saved on the stack, but the accumulator and others are not. These registers are saved by software if necessary. When multiple interrupt services are nested, it is also necessary to avoid using the same data memory area for saving registers. The following methods are used to save/restore the general-purpose registers.

a. Using PUSH and POP instructions

To save only a specific register, PUSH and POP instructions are available.

```
Example:  Save/store register using PUSH and POP instructions
PINTxx:        PUSH      WA                    ;   Save WA register
               (interrupt processing)
               POP       WA                    ;   Restore WA register
               RETI                            ;   RETURN
```

Address
(Example)

	023AH
A	023B
W	023C
PCL	023D
PCH	023E
PSW	023F

At Acceptance of an Interrupt ⇨ At Execution of PUSH instructin ⇨ At Execution of POP instructin ⇨ At Execution of an RETI instruction
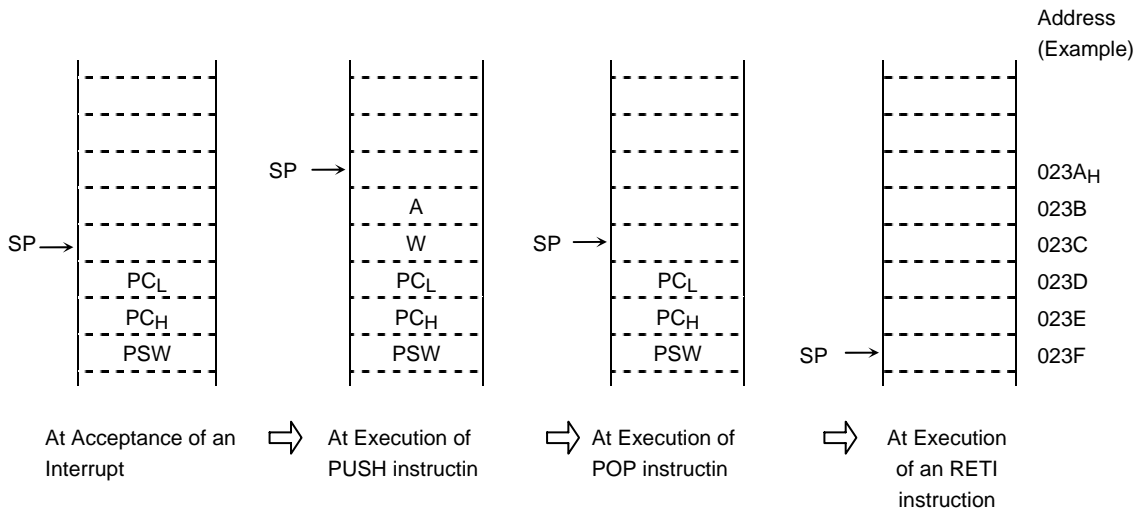
b. Using data transfer instructions

To save only a specific register without nested interrupts, data transfer instructions are available.

Example: Save/store register using data transfer instructions

```
PINTxx:      LD      (GSAVA), A              ;   Save A register
             (interrupt processing)
             LD      A, (GSAVA)              ;   Restore A register
             RETI                            ;   RETURN
```


Saving/restoring general-purpose registers using PUSH/POP instruction
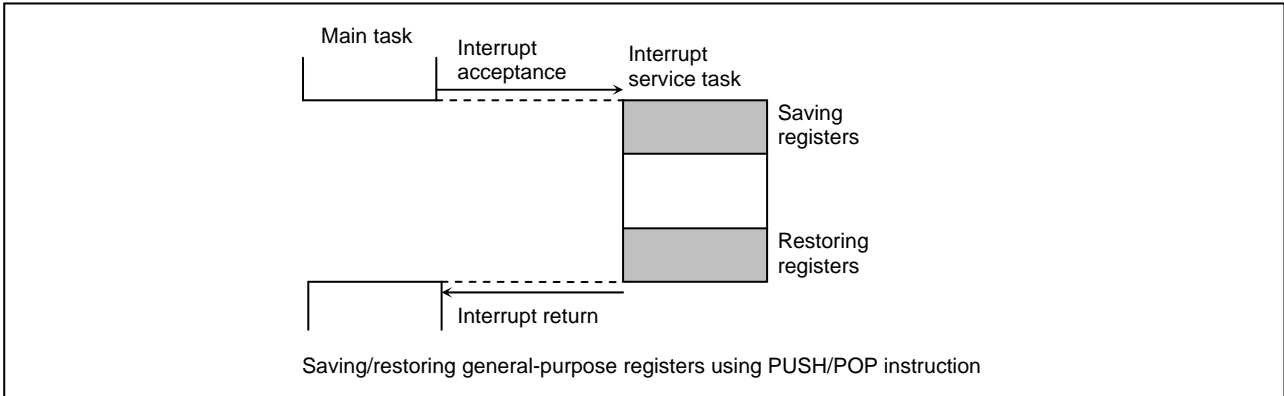
Figure 1.5.4 Saving/Restoring General-purpose Registers under Interrupt Processing

(3) Interrupt return

Interrupt return instructions [RETI]/[RETN] perform as follows.

[RETI]/[RETN] Interrupt Return
1. Program Counter (PC) and program status word (PSW, includes IMF) are restored from the stack.
2. Stack pointer (SP) is incremented by 3.

As for Address Trap interrupt (INTARTAP), it is required to alter stacked data for program counter (PC) to restarting address, during interrupt service program. Otherwise returning interrupt causes INTATRAP again. When interrupt acceptance processing has completed, stacked data for PC_L and PC_H are located on address (SP + 1) and (SP + 2) respectively.

Note: If [RETN] is executed with the above data unaltered, the program returns to the address trap area and INTATRAP occurs again.

Example 1: Returning from address trap interrupt (INTATRAP) service program

PINTxx:	POP	WA	;	Recover SP by 2
	LD	WA, Return Address	;	
	PUSH	WA	;	Alter stacked data
	(interrupt processing)			
	RETN		;	RETURN

Example 2: Restarting without returning interrupt (In this case, PSW (includes IMF) before interrupt
        acceptance is discarded.)

PINTxx	INC	SP	;	Recover SP by 3
	INC	SP	;	
	INC	SP	;	
	(interrupt processing)			
	LD	EIRL, data	;	Set IMF to "1" or clear it to "0"
	JP	Restart Address	;	Jump into restarting address

Note: It is recommended that stack pointer be return to rate before INTATRAP (increment
        3 times), if return interrupt instruction [RETN] is not utilized during interrupt service
        program under INTATRAP (such as Example 2).

Interrupt requests are sampled during the final cycle of the instruction being
executed. Thus, the next interrupt can be accepted immediately after the interrupt
return instruction is executed.

Note: When the interrupt processing time is longer than the interrupt request generation
        time, the interrupt service task is performed but not the main task.

1.5.2   Software Interrupt (INTSW)

Executing the [SWI] instruction generates a software interrupt and immediately starts interrupt processing (INTSW is highest prioritized interrupt).

Use the [SWI] instruction only for detection of the address error or for debugging.

(1) Address error detection

FF_H is read if for some cause such as noise the CPU attempts to fetch an instruction from a non-existent memory address during single chip mode. Code FF_H is the SWI instruction, so a software interrupt is generated and an address error is detected. The address error detection range can be further expanded by writing FF_H to unused areas of the program memory. Address trap reset is generated in case that an instruction is fetched from RAM or SFR areas.

(2) Debugging

Debugging efficiency can be increased by placing the SWI instruction at the software break point setting address.

1.5.3   Undefined Instruction Interrupt (INTUNDEF)

Taking code which is not defined as authorized instruction for instruction causes INTUNDEF. INTUNDEF is generated when the CPU fetches such a code and tries to execute it. INTUNDEF is accepted even if non-maskable interrupt is in process. Contemporary process is broken and INTUNDEF interrupt process starts, soon after it is requested.

Note: The undefined instruction interrupt (INTUNDEF) forces CPU to jump into vector address, as software interrupt (SWI) does.

1.5.4   Address Trap Interrupt (INTATRAP)

Fetching instruction from unauthorized area for instructions (Address trapped area) causes reset-output or address trap interrupt (INTATRAP). INTATRAP is accepted even if non-maskable interrupt is in process. Contemporary process is broken and INTATRAP interrupt process starts, soon after it is requested.

Note: The operating mode under address trapped, whether to be reset-output or interrupt processing, is selected on watchdog timer control register (WDTCR).

1.5.5   External Interrupts

The TMP86FM48 has five external interrupt inputs. These inputs are equipped with digital noise reject circuits (Pulse inputs of less than a certain time are eliminated as noise).

Edge selection is also possible with INT1 to INT3. $\overline{INT0}$ /P00 pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.

Edge selection, noise reject control and $\overline{INT0}$ /P00 pin function selection are performed by the external interrupt control register (EINTCR).

Table 1.5.2 External Interrupts

Source	Pin	Secondary Function Pin	Enable Conditions	Edge	Digital Noise Reject
INT0	$\overline{\text{INT0}}$	P00	IMF = 1, EF$_4$ = 1, INT0EN = 1	Falling edge	Pulses of less than 2/fc [s] are eliminated as noise. Pulses of 7/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT1	INT1	P01	IMF•EF$_6$ = 1	Falling edge or Rising edge	Pulses of less than 15/fc or 63/fc [s] are eliminated as noise. Pulses of 49/fc or 193/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT2	INT2	P02	IMF•EF$_8$ = 1		Pulses of less than 7/fc [s] are eliminated as noise. Pulses of 25/fc [s] or more are considered to be signals.
INT3	INT3	P14/TC3	IMF•EF$_{13}$ = 1		In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT5	$\overline{\text{INT5}}$	P20/$\overline{\text{STOP}}$	IMF•EF$_{21}$ = 1	Falling edge	Pulses of less than 2/fc [s] are eliminated as noise. Pulses of 7/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.

Note 1: If a noiseless signal is input to the external interrupt pin in the NORMAL 1/2 or IDLE 1/2 mode, the maximum time from the edge of input signal until the IL is set is as follows:

    (1) INT1 pin   55/fc [s] (INT1NC = 1), 199/fc [s] (INT1NC = 0)

    (2) INT2, INT3 pin   31/fc [s]

Note 2: Even if the falling edge of $\overline{\text{INT0}}$ pin input is detected at INT0EN = 0, the interrupt latch IL$_4$ is not set.

Note 3: When data changed and did a change of I/O when used external interrupt ports as a normal ports, interrupt request signal occurs incorrectly. Handling of prohibition of interrupt enable register (EIR) is necessary.

Note 4: The maximum time from modifying INT1NC until a noise reject time is changed is 2^6/fc.

External interrupt control register

EINTCR
(0037$_H$)

	7	6	5	4	3	2	1	0	
	INT1NC	INT0EN			INT3ES	INT2ES	INT1ES		(Initial value: 00** 000*)

INT1NC	Noise reject time select	0: Pulses of less than 63/fc [s] are eliminated as noise 1: Pulses of less than 15/fc [s] are eliminated as noise	
INT0EN	P00/$\overline{\text{INT0}}$ pin configuration	0: P00 input/output port 1: $\overline{\text{INT0}}$ pin (Port P00 should be set to an input mode)	R/W
INT3ES INT2ES INT1ES	INT3 to INT1 edge select	0: Rising edge 1: Falling edge	

Note 1: fc: High-frequency clock [Hz], *: Don't care

Note 2: When the system clock frequency is switched between high and low or when the external interrupt control register (EINTCR) is overwritten, the noise canceller may not operate normally. It is recommended that external interrupts are disabled using the interrupt enable register (EIR).

Figure 1.5.5 External Interrupt Control Register

1.6　Reset Circuit

The TMP86FM48 has four types of reset generation procedures: an external reset input, an address trap reset, a watchdog timer reset and a system clock reset. Table 1.6.1 shows on-chip hardware initialization by reset action.

Since the reset circuit has an 11-stage counter for generation of flash reset, which is the reset counter for stabilizing of the power supply for Flash, the reset period is $2^{10}/fc$ [s] (64 μs at 16.0 MHz).

Because the malfunction reset circuit such as watchdog timer reset, address trap reset and system clock reset is not initialized when power is turned on, the reset operation occur for the maximum 24/fc [s] (1.5 μs at 16.0 MHz).

Therefore, the maximum reset period is 24/fc [s] + $2^{10}/fc$ [s] (65.5 μs at 16.0 MHz).

Table 1.6.1 shows on-chip hardware initialization by reset action.

Table 1.6.1　Initializing Internal Status by Reset Action

On-chip Hardware		Initial Value	On-chip Hardware	Initial Value
Program counter	(PC)	(FFFE$_H$)	Prescaler and Divider of timing generator	0
Stack pointer	(SP)	Not initialized		
General-purpose registers (W, A, B, C, D, E, H, L, IX, IY)		Not initialized		
Jump status flag	(JF)	Not initialized	Watchdog timer	Enable
Zero flag	(ZF)	Not initialized	Output latches of I/O ports	Refer to I/O port circuitry
Carry flag	(CF)	Not initialized		
Half carry flag	(HF)	Not initialized		
Sign flag	(SF)	Not initialized		
Overflow flag	(VF)	Not initialized		
Interrupt master enable flag	(IMF)	0		
Interrupt individual enable flags	(EF)	0	Control registers	Refer to each of control register
Interrupt latches	(IL)	0		
			RAM	Not initialized

1.6.1　External Reset Input

The $\overline{RESET}$ pin contains a Schmitt trigger (Hysteresis) with an internal pull-up resistor. When the $\overline{RESET}$ pin is held at "L" level for at least 3 machine cycles (12/fc [s]) with the power supply voltage within the operating voltage range and oscillation stable, a reset is applied and the internal state is initialized.

When $2^{10}/fc$ (65.5 μs at 16 MHz) period passes after the $\overline{RESET}$ pin input goes high, the reset operation is released and the program execution starts at the vector address stored at addresses FFFE$_H$ to FFFF$_H$.
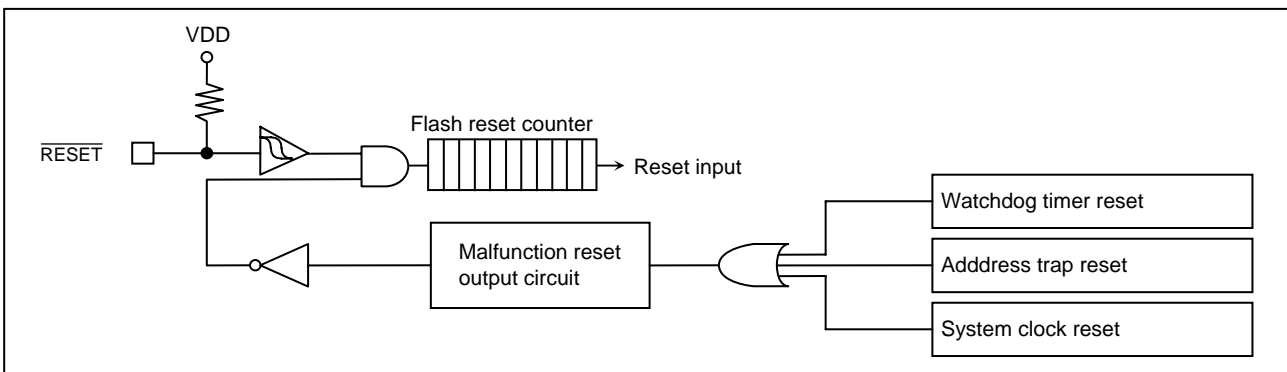
Figure 1.6.1　Reset Circuit

1.6.2 Address-Trap-Reset

If the CPU should start looping for some cause such as noise and an attempt be made to fetch an instruction from the on-chip RAM (when WDTCR1<ATAS> is set to "1") or the SFR area, address-trap-reset and the Flash reset will be generated. The reset time is maximum $24/fc$ [s] $+ 2^{10}/fc$ [s] (65.5 μs at 16.0 MHz).


```
Instruction
execution ──< JP   a >──
                      Address trap is occurred          < Reset release >< Instruction at address r
Internal reset ─────────────────────────────────────────────────────────────────

              |← max 24/fc [s] →|← 2^10/fc [s] →|← 4/fc to 12/fc [s] →|← 16/fc [s] →|
                                 for Flash reset
```

Note 1:   Address "a" is in the SFR or on-chip RAM (WDTCR1<ATAS> = "1") space.

Note 2:   During reset release, reset vector "r" is read out, and an instruction at address "r" is fetched and decoded.
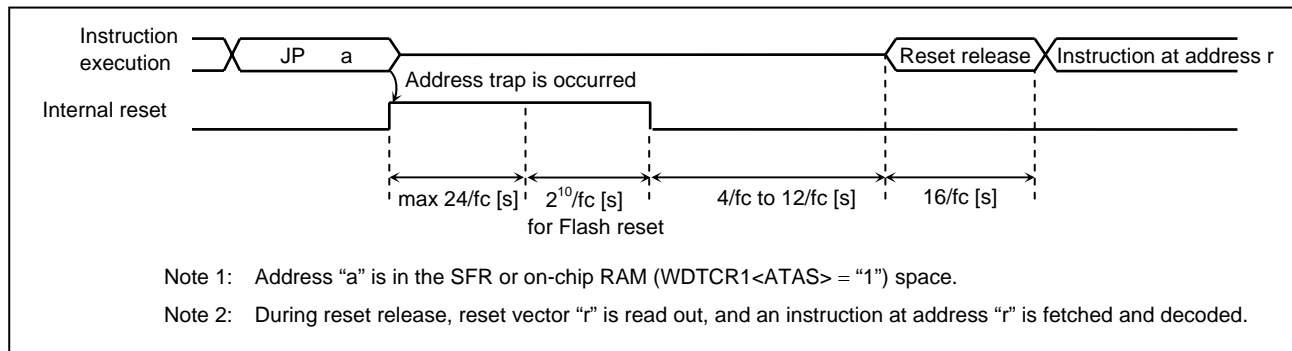
Figure 1.6.2 Address-Trap-Reset

Note: The operating mode under address trapped is alternative of reset or interrupt. Address trap or no address trap can be selected by WDTCR1<ATAS> for the internal RAM.

1.6.3 Watchdog Timer Reset

Refer to Section "2.4 Watchdog Timer".

1.6.4 System-Clock-Reset

If the condition as follows is detected, the system clock reset occurs automatically to prevent dead lock of the CPU. (The oscillation is continued without stopping.)

- In case of clearing SYSCR2<XEN> and SYSCR2<XTEN> simultaneously to "0".
- In case of clearing SYSCR2<XEN> to "0", when the SYSCR2<SYSCK> is "0".
- In case of clearing SYSCR2<XTEN> to "0", when the SYSCR2<SYSCK> is "1".

When the system clock reset is generated, the flash reset is also generated. Therefore, the maximum reset period is $24/fc$ [s] $+ 2^{10}/fc$ [s] (65.5 μs at 16.0 MHz).

2. On-Chip Peripherals Functions

2.1 Special Function Register (SFR)

The TMP86FM48 adopts the memory mapped I/O system, and all peripheral control and data transfers are performed through the special function register (SFR). The SFR is mapped on address 0000_H to $003F_H$, DBR is mapped on address $1F80_H$ to $1FFF_H$.

Figure 2.1.1 to

Figure 2.1.2 indicate the special function register (SFR) and data buffer register (DBR) for TMP86FM48.

Address	Read	Write	Address	Read	Write
0000H	P0DR (P0 Port output latch)		0020H	TC1DRAL (Timer register 1A)	
01	P1DR (P1 Port output latch)		21	TC1DRAH (Timer register 1A)	
02	P2DR (P2 Port output latch)		22	TC1DRBL (Timer register 1B)	
03	P3DR (P3 Port output latch)		23	TC1DRBH (Timer register 1B)	
04	Reserved		24	TC2DRL (Timer register 2)	
05	P5DR (P5 Port output latch)		25	TC2DRH (Timer register 2)	
06	P6DR (P6 Port output latch)		26	ADCDR2 (AD result register 2)	−
07	P7DR (P7 Port output latch)		27	ADCDR1 (AD result register 1)	−
08	P8DR (P8 Port output latch)		28	P6CR2 (P6 Port input control)	
09	Reserved		29	Reserved	
0A	P0OUTCR (P0 Port output control)		2A	P3OUTCR (P3 Port output control)	
0B	P1OUTCR (P1 Port output control)		2B	Reserved	
0C	P6CR1 (P6 Port input/output control)		2C	EIRE (Interrupt enable register)	
0D	P5OUTCR (P5 Port output control)		2D	Reserved	
0E	ADCCR1 (AD control register 1)		2E	ILE (Interrupt latch)	
0F	ADCCR2 (AD control register 2)		2F	Reserved	
10	TC3DRA (Timer register 3A)		30	Reserved	
11	TC3DRB (Timer register 3B)	−	31	Reserved	
12	TC3CR (Timer Counter 3 control)		32	Reserved	
13	TC2CR (Timer Counter 2 control)		33	Reserved	
14	TC5CR (Timer Counter 5 control)		34	−	WDTCR1 (Watchdog timer control)
15	TC5DR (Timer register 5)		35	−	WDTCR2 (Watchdog timer control)
16	Reserved		36	TBTCR (TBT/TG/DVO control)	
17	SIO1CR (SIO1 control)		37	EINTCR (External interrupt control)	
18	SIO1SR (SIO1 status)	−	38	SYSCR1 (System control 1)	
19	SIO1BUF (SIO1 data buffer)		39	SYSCR2 (System control 2)	
1A	Reserved		3A	EIRL (Interrupt enable register)	
1B	SIO2CR (SIO2 control)		3B	EIRH (Interrupt enable register)	
1C	SIO2SR (SIO2 status)	−	3C	ILL (Interrupt latch)	
1D	SIO2BUF (SIO2 data buffer)		3D	ILH (Interrupt latch)	
1E	Reserved		3E	Reserved	
1F	TC1CR (Timer counter 1 control)		3F	PSW (Program status word)	

Note 1:   Do not access reserved areas by the program.

Note 2:   −: Cannot be accessed.

Note 3:   Write-only registers and interrupt latches cannot use the read-modify-write instructions (Bit manipulation instructions such as SET, CLR, etc. and logical operation instructions such as AND, OR, etc.).

Figure 2.1.1 The Special Function Register (SFR) for TMP86FM48 (1/2)

Address

Address	Read	Write
1F80H	Reserved	
D8	Reserved	
D9	−	SBICRA (SBI control 1)
DA	SBIDBR (SBI data buffer)	
DB	−	I2CAR (I2C address)
DC	SBISR (SBI status)	SBICRB (SBI control 2)
DD	UARTSR (UART status)	UARTCR1 (UART control 1)
DE	−	UARTCR2 (UART control 2)
DF	RDBUF (UART received data buffer)	TDBUF (UART transmit data buffer)
E0	EEPCR (FLASH control)	
E1	EEPSR (FLASH status)	−
E2	EEPEVA (FLASH write emulation time control)	
E3	Reserved	
E4	P2OUTCR (P2 Port output control)	
E5	P7CR1 (P7 Port input/output control)	
E6	P7CR2 (P7 Port input control)	
E7	P8CR (P8 Port input/output control)	
E8	Reserved	
E9	Reserved	
EA	Reserved	
EB	Reserved	
EC	Reserved	
ED	P0PRD (P0 Terminal input)	−
EE	P1PRD (P1 Terminal input)	−
EF	P2PRD (P2 Terminal input)	−
F0	P3PRD (P3 Terminal input)	−
F1	Reserved	
F2	P5PRD (P5 Terminal input)	−
F3	Reserved	
F4	Reserved	
F5	Reserved	
F6	Reserved	
F7	Reserved	
F8	Reserved	
F9	Reserved	
FA	Reserved	
FB	Reserved	
FC	Reserved	
FD	Reserved	
FE	−	STOPCR (Key-on wake-up control)
FF	Reserved	

Note 1: Do not access reserved areas by the program.

Note 2: −: Cannot be accessed.

Note 3: Write-only registers and interrupt latches cannot use the read-modify-write instructions (Bit manipulation instructions such as SET, CLR, etc. and logical operation instructions such as AND, OR, etc.).

Figure 2.1.2 The Special Function Register (SFR) for TMP86FM48 (2/2)

2.2   I/O Ports

The TMP86FM48 has 8 parallel input/output ports (54 pins) as follows.

	Primary Function	Secondary Functions
Port P0	8-bit I/O port	External interrupt input, serial interface input/output, UART input/output and Timer/Counter input .
Port P1	8-bit I/O port	External interrupt input, serial interface input/output and Timer/Counter input/output.
Port P2	3-bit I/O port	Low-frequency resonator connections, external interrupt input, STOP mode release signal input.
Port P3	8-bit I/O port	
Port P5	3-bit I/O port	Divider output, Timer/Counter output and Serial Bus Interface input/output.
Port P6	8-bit I/O port	Analog input and STOP mode release signal input.
Port P7	8-bit I/O port	Analog input.
Port P8	8-bit I/O port	

Each output port contains a latch, which holds the output data. All input ports do not have latches, so the external input data should be externally held until the input data is read from outside or reading should be performed several times before processing. Figure 2.2.1 shows input/output timing examples.

External data is read from an I/O port in the S1 state of the read cycle during execution of the read instruction. This timing cannot be recognized from outside, so that transient input such as chattering must be processed by the program.

Output data changes in the S2 state of the write cycle during execution of the instruction which writes to an I/O port.
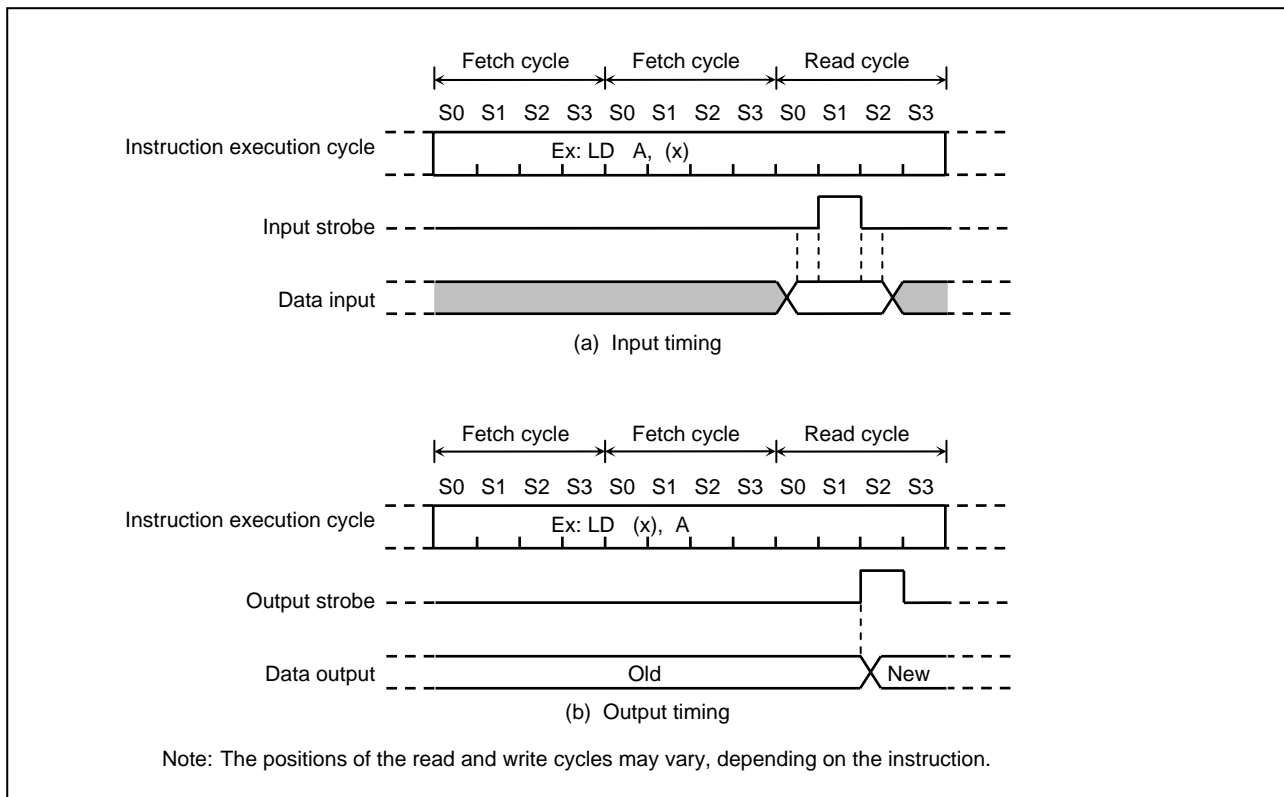
Figure 2.2.1 Input/Output Timing (Example)

2.2.1   Port P0 (P07 to P00)

Port P0 is an 8-bit input/output port which is also used as an external interrupt input, serial interface input/output, timer/counter input and UART input/output. It can be selected whether output circuit of P0 port is CMOS output or a sink open drain individually, by setting the output circuit control (P0OUTCR). When a corresponding bit of P0OUTCR is cleared to "0", the output circuit is selected to a sink open drain and when a corresponding bit of P0OUTCR is set to "1", the output circuit is selected to a CMOS output.

When used as an input port or a secondary function input (External interrupt input, serial interface input, timer/counter input or UART input), the respective output latch (P0DR) should be set to "1" and its corresponding P0OUTCR bit should be cleared to "0".

When used as a secondary function output (Serial interface output or UART output), the respective P0DR should be set to "1".

During reset, the P0DR is initialized to "1" and P0OUTCR is initialized to "0".

P0 port output latch (P0DR) and P0 port terminal input (P0PRD) are located on their respective address. When read the output latch data, the P0DR should be read and when read the terminal input data, the P0PRD register should be read.

	7	6	5	4	3	2	1	0	
P0DR (0000H) R/W	P07 $\overline{SCK1}$	P06 TxD SO1	P05 RxD SI1	P04	P03 TC2	P02 INT2	P01 INT1	P00 $\overline{INT0}$	(Initial value: 1111 1111)

P0OUTCR (000AH)									(Initial value: 0000 0000)

P0OUTCR	Port P0 output circuit control (Set for each bit individually)	0: Sink open-drain output 1: CMOS output	R/W

	7	6	5	4	3	2	1	0
P0PRD (1FEDH) Read only	P07	P06	P05	P04	P03	P02	P01	P00

Figure 2.2.2 Port 0

2.2.2 Port P1 (P17 to P10)

Port P1 is a 8-bit input/output port which is also used as an external interrupt input, serial interface input/output and timer/counter input/output. It can be selected whether output circuit of P1 port is CMOS output or a sink open drain individually, by setting the output circuit control (P1OUTCR). When a corresponding bit of P1OUTCR is cleared to "0", the output circuit is selected to a sink open drain and when a corresponding bit of P1OUTCR is set to "1", the output circuit is selected to a CMOS output.

When used as an input port or a secondary function input (External interrupt input, serial interface input, timer/counter input), the respective output latch (P1DR) should be set to "1" and its corresponding P1OUTCR bit should be cleared to "0".

When used as a secondary function output (Serial interface output or timer/counter output), the respective P1DR should be set to "1".

During reset, the P1DR is initialized to "1" and P1OUTCR is initialized to "0".

P1 port output latch (P1DR) and P1 port terminal input (P1PRD) are located on their respective address. When read the output latch data, the P1DR should be read and when read the terminal input data, the P1PRD register should be read.
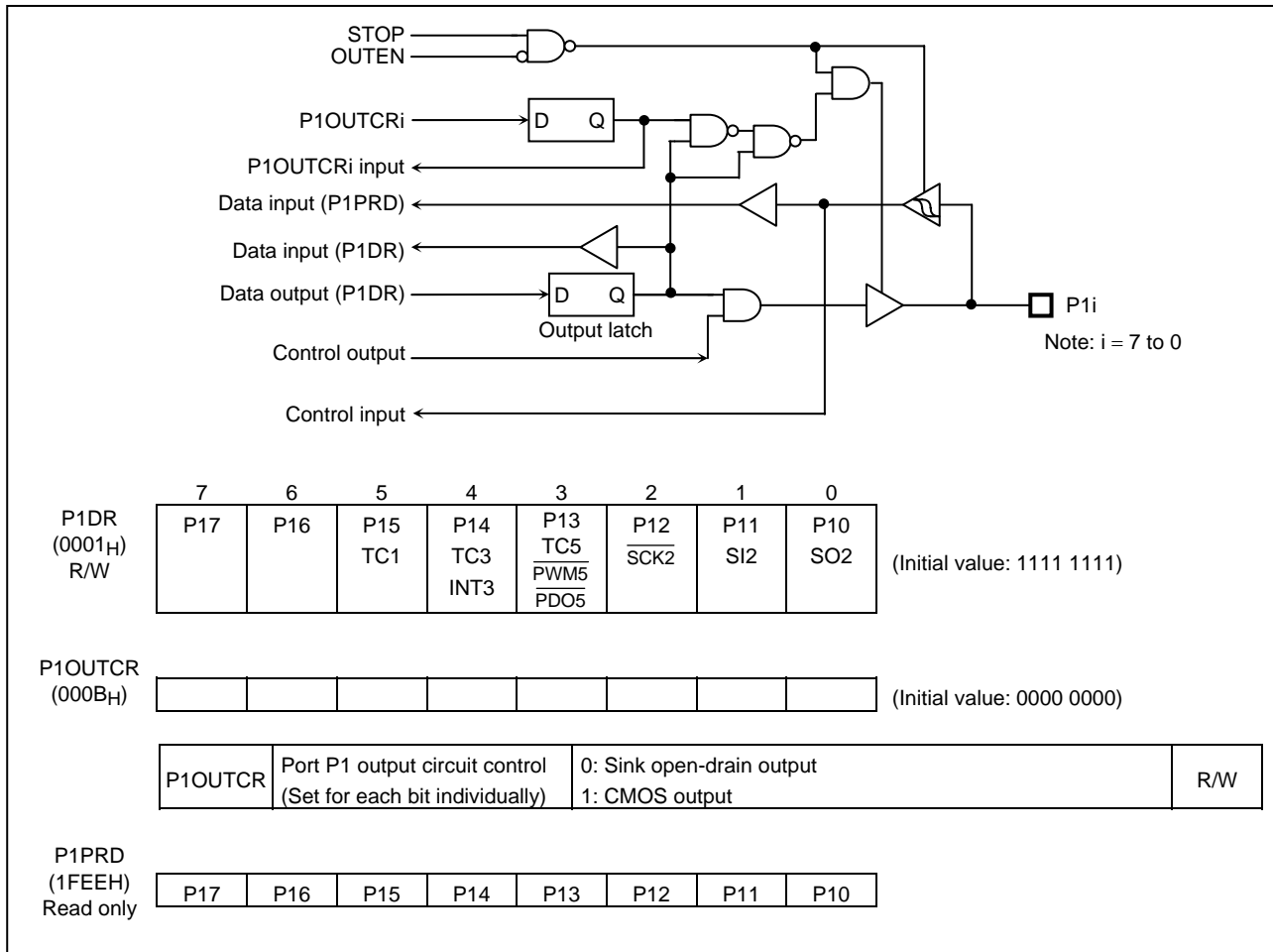
Figure 2.2.3 Port 1

2.2.3 Port P2 (P22 to P20)

Port P2 is a 3-bit input/output port. It is also used as an external interrupt, a STOP mode release signal input, and low-frequency crystal oscillator connection pins. It can be selected whether output circuit of P2 port is CMOS (P21 and P22 have a pull-up resistor) output or a sink open drain individually, by setting the output circuit control (P2OUTCR). When a corresponding bit of P2OUTCR is cleared to "0", the output circuit is selected to a sink open drain and when a corresponding bit of P2OUTCR is set to "1", the output circuit is selected to a CMOS output. (In case of P21 and P22, the pull-up resistor is connected.)

When used as an input port or an external interrupt input, the respective output latch (P2DR) should be set to "1".

During reset, the P2DR initialized to "1" and P2OUTCR is initialized to "0".

A low-frequency crystal oscillator (32.768 kHz) is connected to pins P21 (XTIN) and P22 (XTOUT) in the dual-clock mode. In the single-clock mode, pins P21 and P22 can be used as normal input/output ports.

It is recommended that pin P20 should be used as an external interrupt input, a STOP mode release signal input, or an input port. If it is used as an output port, the interrupt latch is set on the falling edge of the output pulse.

P2 port output latch (P2DR) and P2 port terminal input (P2PRD) are located on their respective address. When read the output latch data, the P2DR should be read and when read the terminal input data, the P2PRD register should be read.
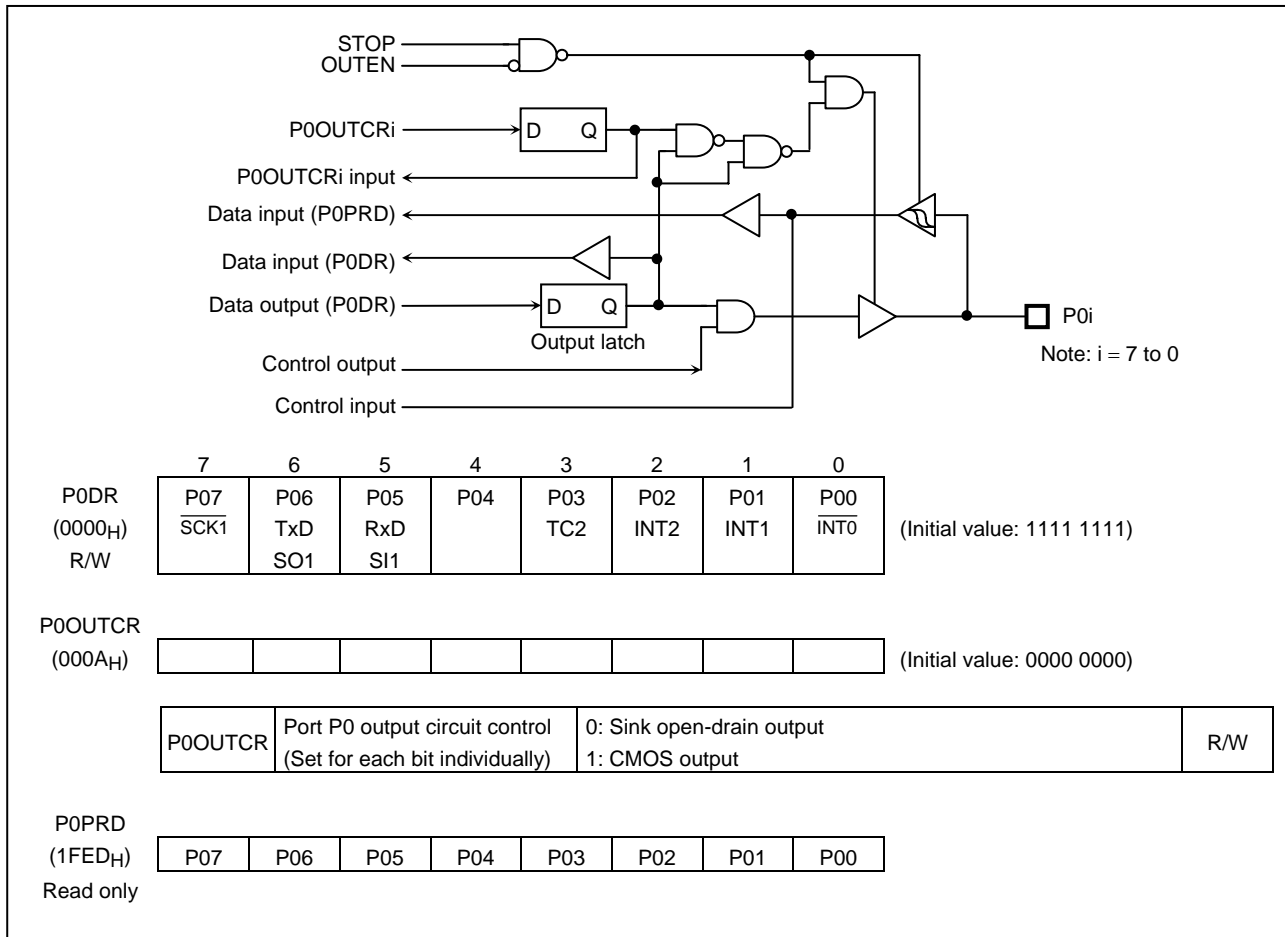
If a read instruction is executed for port P2DR, P2OUTCR and P2PRD, read data of bits 7 to 3 are unstable.

Note: When XTEN<SYSCR2> is set to "1", P21 and P22 become a high impedance state.

Figure 2.2.4 Port 2 (P21 and P22)

Note: Port P20 is used as $\overline{\text{STOP}}$ pin. Therefore, when stop mode is started, OUTEN does not affect to P20, and P20 becomes High-Z state.

	7	6	5	4	3	2	1	0	
P2DR (0002H) R/W						P22 XTOUT	P21 XTIN	P20 $\overline{\text{INT5}}$ $\overline{\text{STOP}}$	(Initial value: **** *111) *: Don't care

	7	6	5	4	3	2	1	0	
P2OUTCR (1FE4H)									(Initial value: **** *000) *: Don't care

P2OUTCR	Port P2 output circuit control (Set for each bit individually)	0:	Sink open-drain output		R/W
		1:	P20 pin	P21, P22 ports	
			CMOS output	CMOS output with pull-up resistor	

	7	6	5	4	3	2	1	0
P2PRD (1FEFH) Read only						P22	P21	P20

Figure 2.2.5 Port 2 (P20)

2.2.4   Port P3 (P37 to P30)

Port P3 is an 8-bit input/output port. It can be selected whether output circuit of P3 port is CMOS output or a sink open drain individually, by setting P3OUTCR. (N-ch high current output) When a corresponding bit of P3OUTCR is cleared to "0", the output circuit is selected to a sink open drain and when a corresponding bit of P3OUTCR is set to "1", the output circuit is selected to a CMOS output.

When used as an input port, the respective output latch (P3DR) should be set to "1" and its corresponding P3OUTCR bit should be cleared to "0".

During reset, the P3DR is initialized to "1", and the P3OUTCR is initialized to "0".

P3 port output latch (P3DR) and P3 port terminal input (P3PRD) are located on their respective address. When read the output latch data, the P3DR should be read and when read the terminal input data, the P3PRD register should be read.
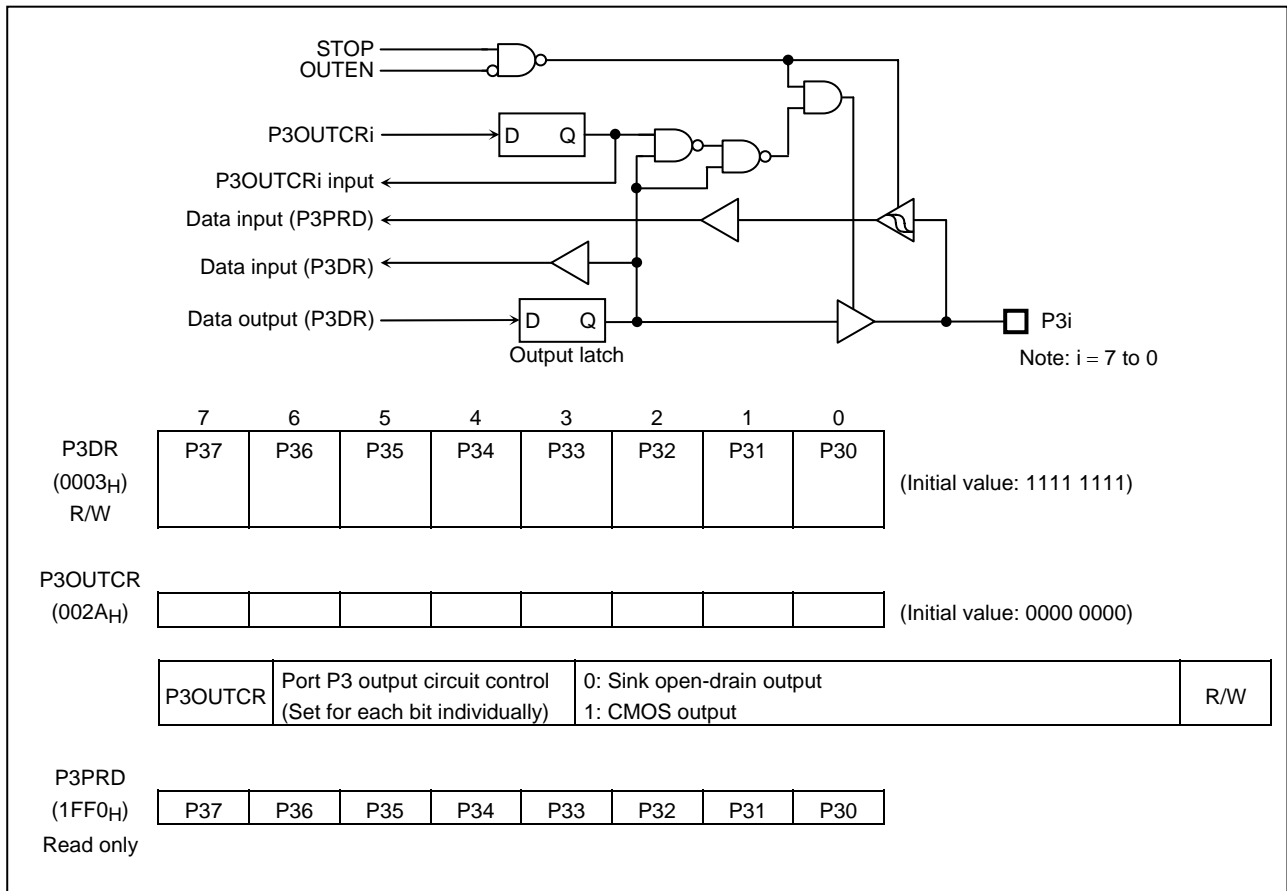

Figure 2.2.6 Port 3

2.2.5   Port P5 (P52 to P50)

Port P5 is an 3-bit input/output port which is also used as a timer/counter output, divider output and serial bus interface input/output. (N-ch high current output) It can be selected whether output circuit of P5 port is CMOS output or a sink open drain individually, by setting the output circuit control (P5OUTCR). When a corresponding bit of P5OUTCR is cleared to "0", the output circuit is selected to a sink open drain and when a corresponding bit of P5OUTCR is set to "1", the output circuit is selected to a CMOS output.

When used as an input port or a serial bus interface input/output, the respective output latch (P5DR) should be set to "1" and its corresponding P5OUTCR bit should be cleared to "0".

When used as a secondary function output (Timer/counter output or divider output), the respective P5DR should be set to "1".

When used as a serial bus interface input/output, P5DR of P50 and P51 should be set to "1" and P5OUTCR of P50 and P51 should be cleared to "0" as a sink open drain output.

During reset, the P5DR is initialized to "1" and P5OUTCR is initialized to "0".

P5 port output latch (P5DR) and P5 port terminal input (P5PRD) are located on their respective address. When read the output latch data, the P5DR should be read and when read the terminal input data, the P5PRD register should be read.

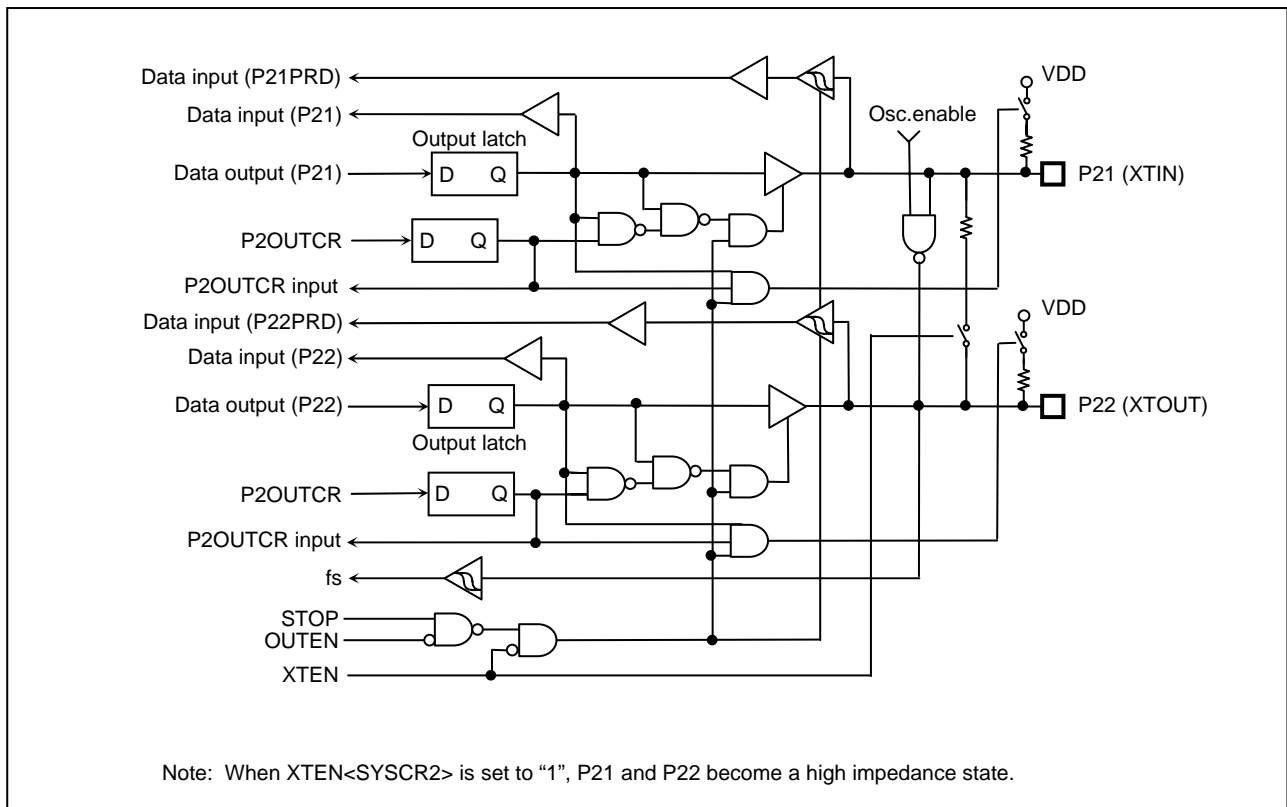If a read instruction is executed for P5DR, P5OUTCR and P5PRD, read data of bits 7 to 3 are unstable.
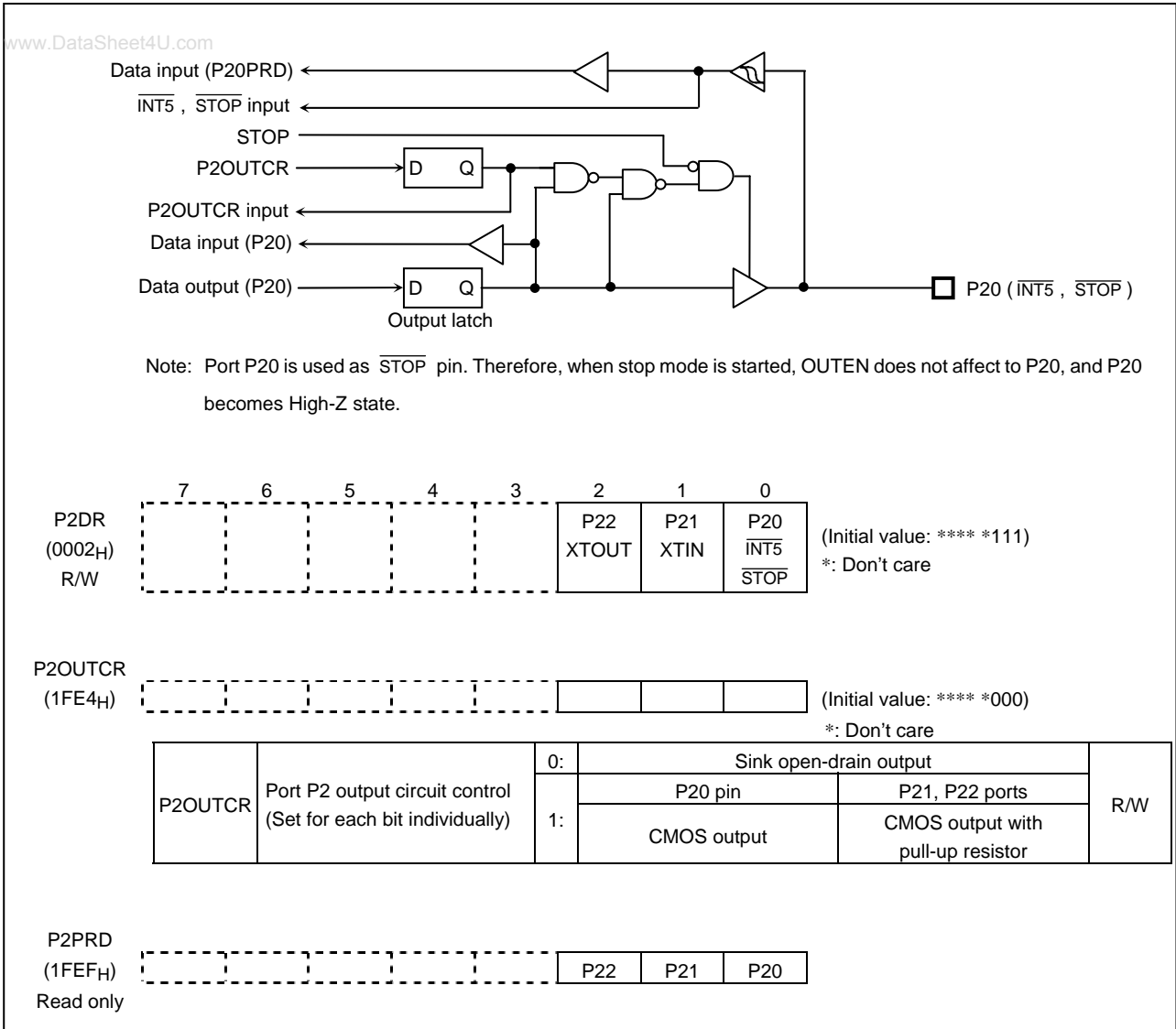

Figure 2.2.7 Port 5

2.2.6   Port P6 (P67 to P60)

Port P6 is an 8-bit input/output port which can be configured as an input or an output in one-bit unit. Port P6 is also used as an analog input and key-on wake-up input. Input/output mode is specified by the P6 control register (P6CR1). P6 port input is controlled by the input control register (P6CR2).

When used as an output port, respective P6CR1 should be set to "1".

When used as an input port, respective P6CR1 should be cleared to "0" and respective P6CR2 should be set to "1".

When used as an analog input, respective P6CR2 should be cleared to "0" after respective P6CR1 is cleared to "0".

When used as a key on wake up input, respective STOPkEN<STOPCR> should be set to "1". (k = 3 to 0)

During reset, the P6CR1 and P6DR are initialized to "0", and the P6CR2 is initialized to "1". Table 2.2.1 and Table 2.2.2 show a P6 state.

Table 2.2.1 P63 to P60 State

P6CR1	P6CR2	P6DR	P6DR Read	Output	Remark
0	0	*	"0"	High-Z	–
0	1	*	Terminal input	High-Z	Input mode
1	*	0	"0" (Output latch)	Low	Output mode
1	*	1	"1" (Output latch)	High	Output mode

*: Don't care.

Table 2.2.2 P67 to P64 State

STOPkEN	P6CR1	P6CR2	P6DR	P6DR read	Output	Remark
0	0	0	*	"0"	High-Z	–
0	0	1	*	Terminal input	High-Z	Input mode
0	1	*	0	"0" (Output latch)	Low	Output mode
0	1	*	1	"1" (Output latch)	High	Output mode
1	*	*	*	Terminal input	High-Z	Key on wake up

*: Don't care.

Note: STOPkEN is bit7 to 4 in STOPCR.
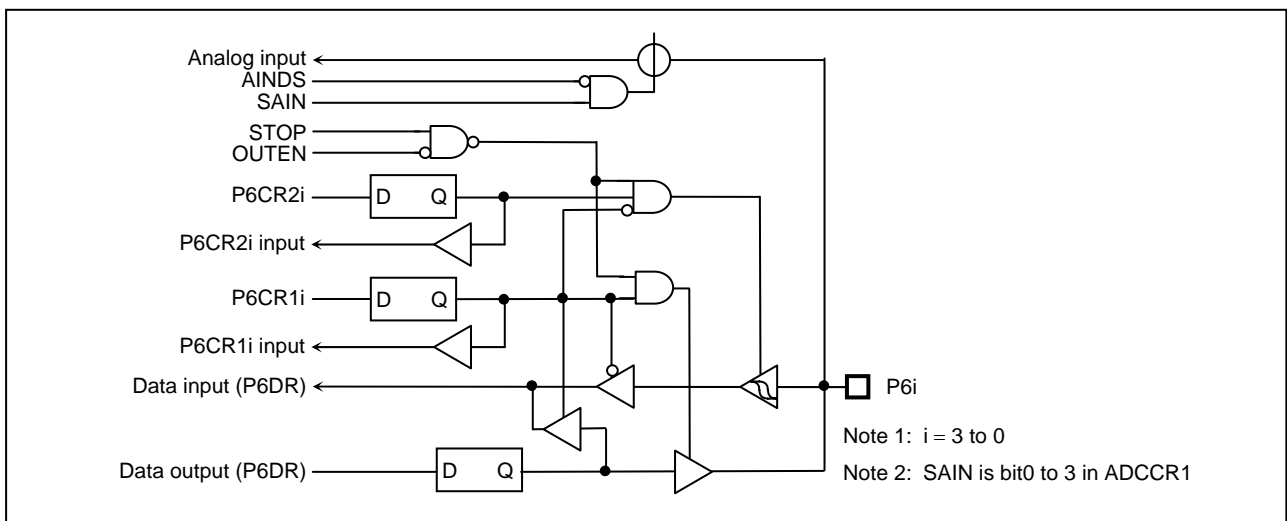
Figure 2.2.8 Port 6 (P63 to P60)

Note 1: j = 7 to 4, k = 3 to 0

Note 2: SAIN is bit0 to 3 in ADCCR1

Note 3: STOPkEN is bit 7 to 4 in STOPCR.

	7	6	5	4	3	2	1	0	
P6DR (0006$_H$) R/W	P67 AIN07 STOP3	P66 AIN06 STOP2	P65 AIN05 STOP1	P64 AIN04 STOP0	P63 AIN03	P62 AIN02	P61 AIN01	P60 AIN00	(Initial value: 0000 0000)

P6CR1
(000C$_H$)   (Initial value: 0000 0000)

P6CR1	Port P6 I/O control (Set for each bit individually)	0: Input mode or Analog input 1: Output mode	R/W

P6CR2
(0028$_H$)   (Initial value: 1111 1111)

P6CR2	Port P6 input control (Set for each bit individually)	0: Input disable 1: Input enable	R/W

Note 1:   Do not set output mode to pin which is used for an analog input.

Note 2:   If both P6CR1 and P6CR2 are cleared to "0", the read value of P6DR is always "0".
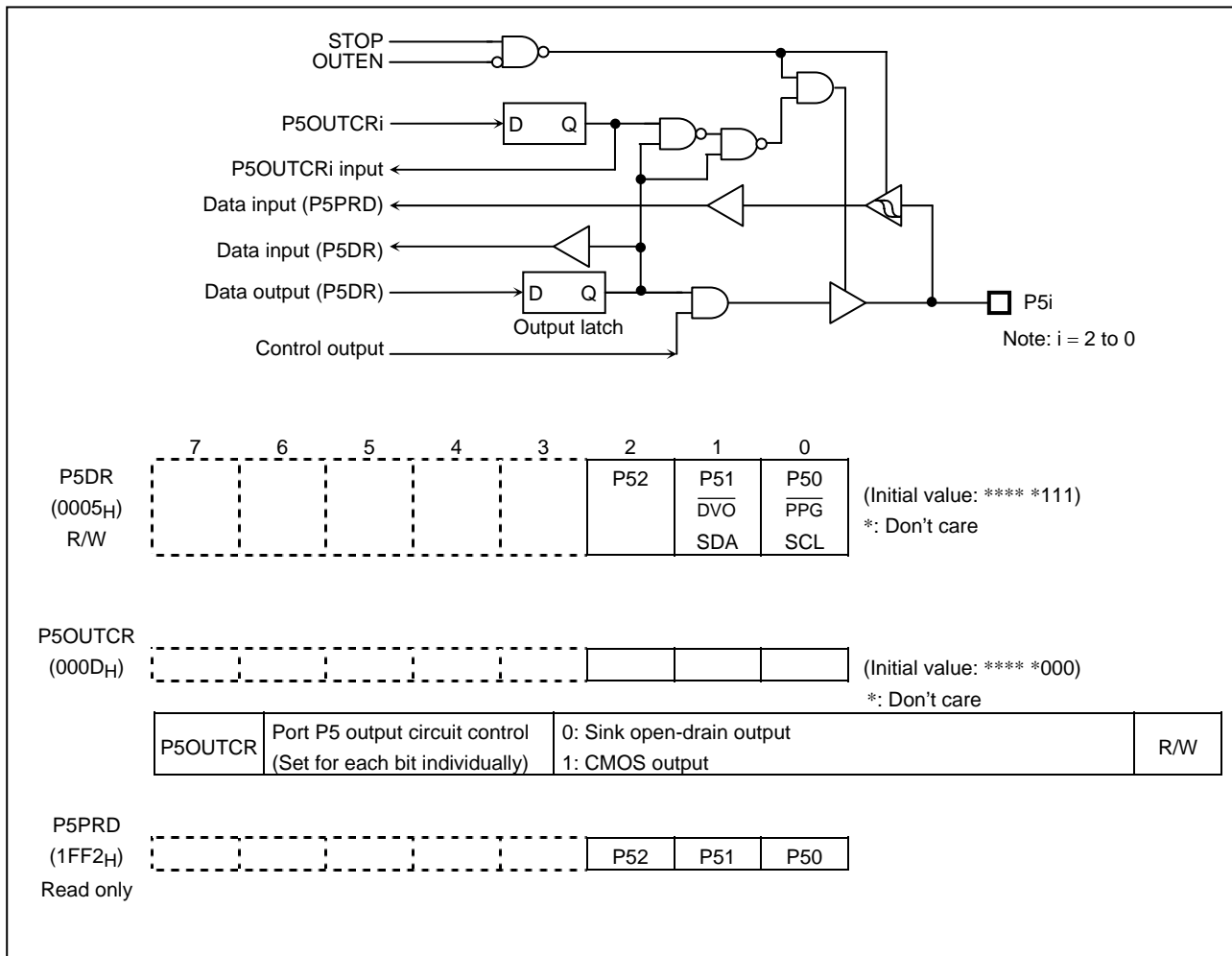
Figure 2.2.9 Port 6 (P67 to P64)

2.2.7 Port P7 (P77 to P70)

Port P7 is an 8-bit input/output port which can be configured as an input or an output in one-bit unit. Port P7 is also used as an analog input. Input/output mode is specified by the P7 control register (P7CR1). P7 port input is controlled by the input control register (P7CR2).

When used as an output port, respective P7CR1 should be set to "1".

When used as an input port, respective P7CR1 should be cleared to "0" and respective P7CR2 should be set to "1".

When used as an analog input, respective P7CR2 should be cleared to "0" after respective P7CR1 is cleared to "0".

During reset, the P7CR1 and P7DR are initialized to "0", and the P7CR2 is initialized to "1". Table 2.2.3 shows a P7 state.

Table 2.2.3 P7 Port State

P7CR1	P7CR2	P7DR	P7DR Read	Output	Remark
0	0	*	"0"	High-Z	–
0	1	*	Terminal input	High-Z	Input mode
1	*	0	"0" (Output latch)	Low	Output mode
1	*	1	"1" (Output latch)	High	Output mode

*: Don't care.
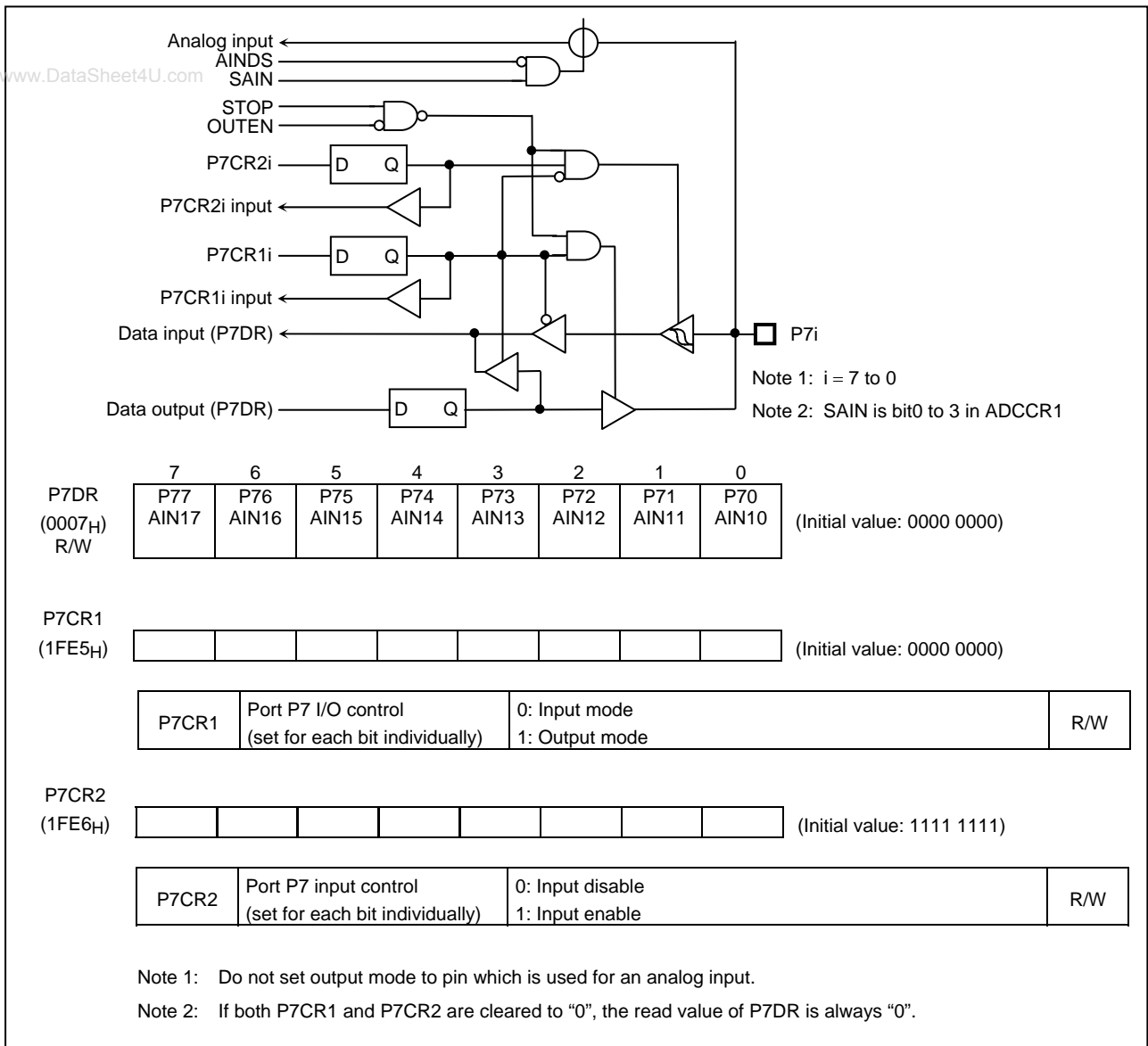
Figure 2.2.10 Port 7

2.2.8 Port P8 (P87 to P80)

Port P8 is an 8-bit input/output port which can be configured as an input or an output in one-bit unit. Input/output mode is specified by the P8 control register (P8CR).

When used as an output port, respective P8CR should be set to "1".

When used as an input port, respective P8CR should be cleared to "0".

During reset, the P8CR and P8DR are initialized to "0". Table 2.2.4 shows a P8 state.

Table 2.2.4 P8 Port State

P8CR	P8DR	P8DR read	Output	Remark
0	*	Terminal input	High-Z	Input mode
1	0	"0" (Output latch)	Low	Output mode
1	1	"1" (Output latch)	High	Output mode

*: Don't care.

	7	6	5	4	3	2	1	0	
P8DR (0008H) R/W	P87	P86	P85	P84	P83	P82	P81	P80	(Initial value: 0000 0000)

P8CR (1FE7H) (Initial value: 0000 0000)

P8CR	Port P8 I/O control (Set for each bit individually)	0: Input mode or Analog input 1: Output mode	R/W
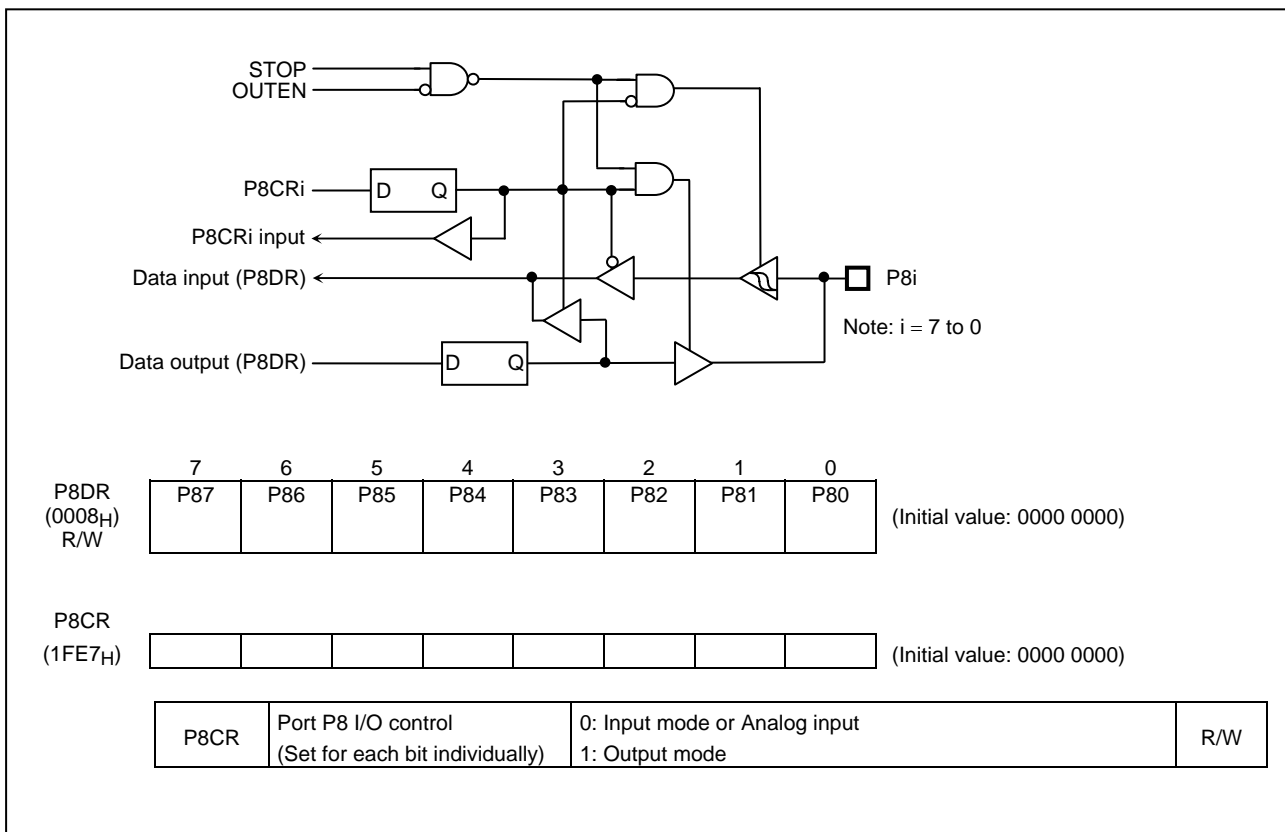
Figure 2.2.11 Port 8

2.3 Time Base Timer (TBT)

The time base timer generates time base for key scanning, dynamic displaying, etc. It also provides a time base timer interrupt (INTTBT).

An INTTBT is generated on the first falling edge of source clock (The divider output of the timing generator) after the time base timer has been enabled. The divider is not cleared by the program; therefore, only the first interrupt may be generated ahead of the set interrupt period (Figure 2.3.1 (b)).

The interrupt frequency (TBTCK) must be selected with the time base timer disabled (the interrupt frequency must not be changed with the disable from the enable state). Both frequency selection and enabling can be performed simultaneously.

MPX

$fc/2^{23}$ or $fs/2^{15}$ — A
$fc/2^{21}$ or $fs/2^{13}$ — B
$fc/2^{16}$ or $fs/2^{8}$ — C
$fc/2^{14}$ or $fs/2^{6}$ — D
$fc/2^{13}$ or $fs/2^{5}$ — E
$fc/2^{12}$ or $fs/2^{4}$ — F
$fc/2^{11}$ or $fs/2^{3}$ — G
$fc/2^{9}$ or $fs/2$ — H

Source clock
Falling edge detector
IDLE0/SLEEP0 release request
INTTBT interrupt request

TBTCK   TBTEN
TBTCR
Time base timer control register

(a) Configuration

Source clock
TBTEN
INTTBT
Interrupt period
Enable TBT
MPX: Multiplexer

(b) Time base timer interrupt

Figure 2.3.1 Time Base Timer

Example: Sets the time base timer frequency to $fc/2^{16}$ [Hz] and enables an INTTBT interrupt.

```
LD        (TBTCR), 00000010B      ;   TBTCK ← 010
LD        (TBTCR), 00001010B      ;   TBTEN ← 1
DI                                ;   IMF ← 0
SET       (EIRL). 6
```

TBTCR	7	6	5	4	3	2	1	0	
(0036H)	(DVOEN)	(DVOCK)		(DV7CK)	TBTEN		TBTCK		(Initial value: 0000 0000)

TBTEN	Time base timer enable/disable	0: Disable 1: Enable				
TBTCK	Time base timer interrupt frequency select [Hz]		NORMAL1/2, IDLE1/2 Mode		SLOW, SLEEP Mode	R/W
			$DV7CK = 0$	$DV7CK = 1$		
		000	$fc/2^{23}$	$fs/2^{15}$	$fs/2^{15}$	
		001	$fc/2^{21}$	$fs/2^{13}$	$fs/2^{13}$	
		010	$fc/2^{16}$	$fs/2^{8}$	–	
		011	$fc/2^{14}$	$fs/2^{6}$	–	
		100	$fc/2^{13}$	$fs/2^{5}$	–	
		101	$fc/2^{12}$	$fs/2^{4}$	–	
		110	$fc/2^{11}$	$fs/2^{3}$	–	
		111	$fc/2^{9}$	$fs/2$	–	

Note: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Figure 2.3.2 Time Base Timer Control Register

Table 2.3.1 Time Base Timer Interrupt Frequency (Example: fc = 16 MHz, fs = 32.768 kHz)

TBTCK	Time Base Timer Interrupt Frequency [Hz]		
	NORMAL1/2, IDLE1/2 Mode		SLOW, SLEEP Mode
	$DV7CK = 0$	$DV7CK = 1$	
000	1.91	1	1
001	7.63	4	4
010	244.14	128	–
011	976.56	512	–
100	1953.13	1024	–
101	3906.25	2048	–
110	7812.5	4096	–
111	31250	16384	–

2.4   Watchdog Timer (WDT)

The watchdog timer is a fail-safe system to rapidly detect the CPU malfunctions such as endless looping caused by noise or the like, or deadlock and resume the CPU to the normal state.

The watchdog timer signal for detecting malfunction can be selected either a "reset request" or a non-maskable "interrupt request". However, selection is possible only once after reset. At first the "reset request" is selected.

When the watchdog timer is not being used for malfunction detection, it can be used as a timer to generate an interrupt at fixed intervals.

Note: Care must be given in system design so as to protect the watchdog timer from disturbing noise. Otherwise the Watchdog Timer may not fully exhibit its functionality.

2.4.1   Watchdog Timer Configuration

Figure 2.4.1 Watchdog Timer Configuration

2.4.2 Watchdog Timer Control

Figure 2.4.2 shows the watchdog timer control registers (WDTCR1, WDTCR2). The watchdog timer is automatically enabled after reset.

(1) Malfunction detection methods using the watchdog timer

The CPU malfunction is detected as follows.

1. Setting the detection time, selecting output, and clearing the binary counter.
2. Repeatedly clearing the binary counter within the setting detection time

If the CPU malfunctions such as endless looping or deadlock occur for any cause, the watchdog timer output will become active at the rising of an overflow from the binary counters unless the binary counters are cleared. At this time, when WDTCR1<WDTOUT> = "1", a reset is generated and the internal hardware is reseted. When WDTCR1<WDTOUT> = "0", a watchdog timer interrupt (INTWDT) is generated.

The watchdog timer temporarily stops counting in STOP mode including warm-up or IDLE mode, and automatically restarts (Continues counting) when the STOP/IDLE mode is released.

Note: The watchdog timer consists of an internal divider and a two-stage binary counter. When clear code $4E_H$ is written, only the binary counter is cleared, not the internal divider. Depending on the timing at which clear code $4E_H$ is written on the WDTCR2 register, the overflow time of the binary counter may be at minimum 3/4 of the time set in WDTCR1 <WDTT>. Thus, write the clear code using a shorter cycle than 3/4 of the time set in WDTCR1 <WDTT>.

Example: Sets the watchdog timer detection time to 2^{21}/fc [s] and resets the CPU malfunction.

```
SYSCR1        LD          (WDTCR2), 4EH          ;  Clears the binary counters
              LD          (WDTCR1), 00001101B    ;  WDTT ← 10, WDTOUT ← 1
              LD          (WDTCR2), 4EH          ;  Clears the binary counters (Always clear
Within 3/4 of                                      immediately before and after changing
WDT detection                                      WDTT)
time
Within 3/4 of LD          (WDTCR2), 4EH          ;  Clears the binary counters
WDT detection
time          LD          (WDTCR2), 4EH          ;  Clears the binary counters
```

Watchdog Timer Register 1

WDTCR1	7	6	5	4	3	2	1	0	
(0034H)			(ATAS)	(ATOUT)	WDTEN	WDTT		WDTOUT	(Initial value: **11 1001)

WDTEN	Watchdog timer enable/disable	0: Disable (It is necessary to write the disable code to WDTCR2) 1: Enable				Write only
WDTT	Watchdog timer detection time [s]		NORMAL1/2 mode		SLOW mode	
			DV7CK = 0	DV7CK = 1		
		00	2^{25}/fc	2^{17}/fs	2^{17}/fs	
		01	2^{23}/fc	2^{15}/fs	2^{15}/fs	
		10	2^{21}/fc	2^{13}/fs	2^{13}/fs	
		11	2^{19}/fc	2^{11}/fs	2^{11}/fs	
WDTOUT	Watchdog timer output select	0: Interrupt request 1: Reset request				

Note 1: WDTOUT cannot be set to "1" by program after clearing WDTOUT to "0".

Note 2: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 3: WDTCR1 is a write-only register and must not be used with any of read-modify-write instructions.

Note 4: The watchdog timer must be disabled or the counter must be cleared immediately before entering to the STOP mode. When the counter is cleared, the counter must be cleared again immediately after releasing the STOP mode.

Note 5: To disable the watchdog timer, always write "4EH" (Clear code) to WDTCR2 for clearing the binary counter before writing "0" to WDTEN, and then write "B1H" (Disable code) to WDTCR2.

Also, immediately before these procedure, disable the interrupt mater flag (IMF) by DI instruction.

Watchdog Timer Register 2

WDTCR2	7	6	5	4	3	2	1	0	
(0035H)									(Initial value: **** ****)

WDTCR2	Watchdog timer control code write register	4EH: Watchdog timer binary counter clear (Clear code) B1H: Watchdog timer disable (Disable code) D2H: Enable assigning address trap area Others: Invalid	Write only

Note 1: The disable code is invalid unless written when WDTCR1<WDTEN> = 0.

Note 2: *: Don't care

Note 3: The binary counter of the watchdog timer must not be cleared by the interrupt task.

Note 4: Write clear code 4EH within 3/4 of the time set in WDTCR1<WDTT>.
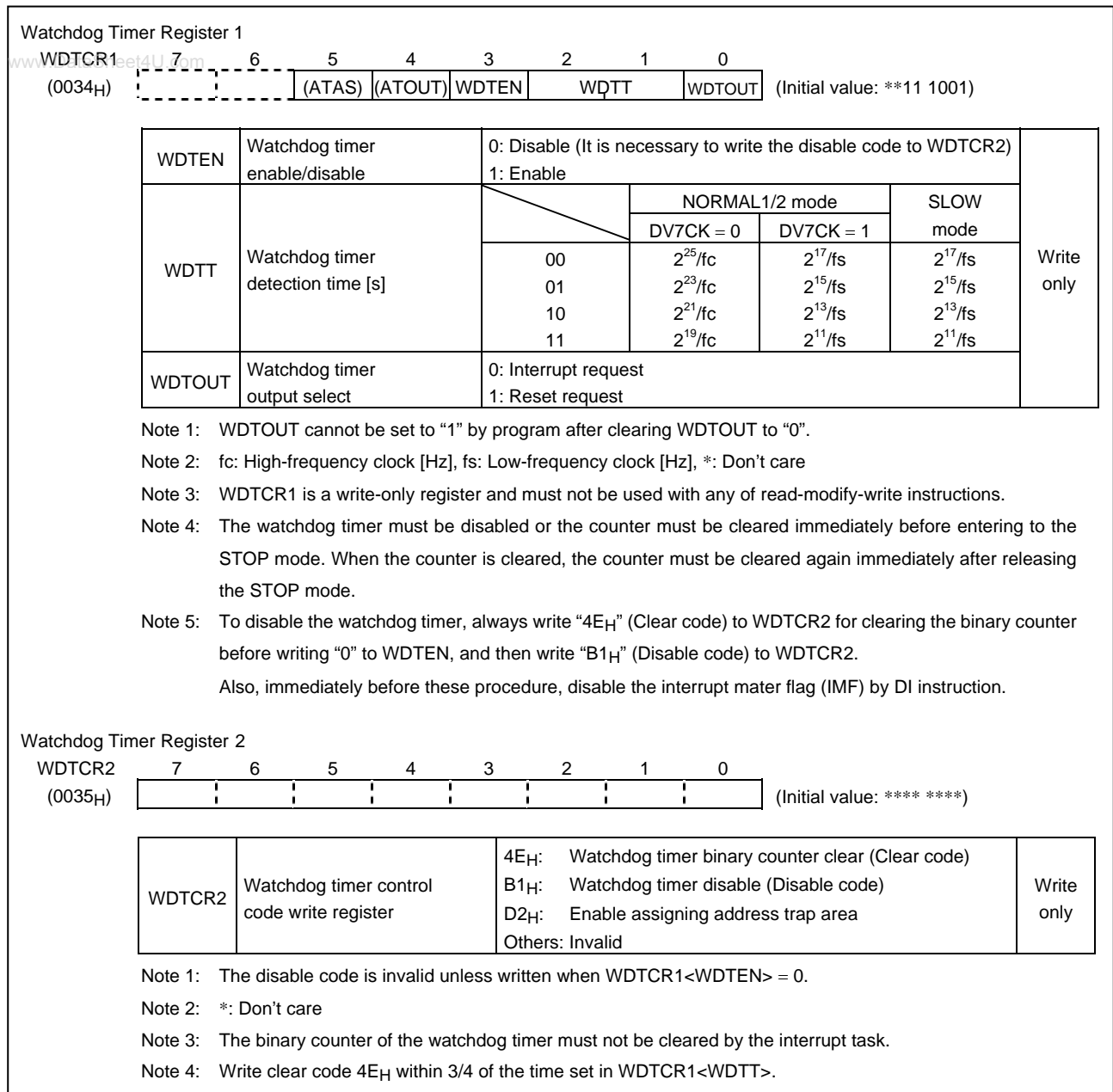
Figure 2.4.2 Watchdog Timer Control Registers

(2) Watchdog timer enable

The watchdog timer is enabled by setting WDTCR1<WDTEN> to "1".
WDTCR1<WDTEN> is initialized to "1" during reset, so the watchdog timer operates immediately after reset is released.

(3) Watchdog timer disable

To disable the watchdog time, write "4EH" (Clear code) to WDTCR2 for clearing the binary counter before writing "0" to WDTCR1<WDTEN>, and then write "B1H" (Disable code) to WDTCR2. The watchdog timer is not disabled if this procedure is reversed and the disable code is written to WDTCR2 before WDTCR1<WDTEN> is cleared to "0". Also, immediately before these procedure, disable the interrupt master flag (IMF) by DI instruction. During disabling the watchdog timer, the binary counters are cleared to "0".

Example: Disables watchdog timer

DI		;	IMF ← 0
LD	(WDTCR2), 4EH	;	Clear the binary counter
LDW	(WDTCR1), 0B101H	;	WDTEN ← 0, WDTCR2 ← Disable code

Table 2.4.1 Watchdog Timer Detection Time (Example: fc = 16 MHz, fs = 32.768 kHz)

WDTT	Watchdog Timer Detection Time [s]		
	NORMAL1/2 Mode		SLOW Mode
	DV7CK = 0	DV7CK = 1	
00	2.097	4	4
01	524.288 m	1	1
10	131.072 m	250 m	250 m
11	32.768 m	62.5 m	62.5 m

2.4.3 Watchdog Timer Interrupt (INTWDT)

This is a non-maskable interrupt which can be accepted regardless of the contents of the EIR. If a watchdog timer interrupt or a software interrupt is already accepted, however, the new watchdog timer interrupt waits until the previous interrupt processing is completed (The end of the [RETN] instruction execution).

The stack pointer (SP) should be initialized before using the watchdog timer output as an interrupt source with WDTOUT.

Example: Watchdog timer interrupt setting up

LD	SP, 023FH	;	Sets the stack pointer
LD	(WDTCR1), 00001000B	;	WDTOUT ← 0

2.4.4 Watchdog Timer Reset

If the watchdog timer reset request occur, a reset is generated and the internal hardware is reseted. When the watchdog timer reset is generated, the flash reset is also generated. Therefore, the maximum reset period is $24/fc$ [s] + $2^{10}/fc$ [s] (65.5 μs at 16.0 MHz).

Note: The high-frequency clock oscillator also immediately turns on when a watchdog timer reset is generated in SLOW mode. In this case, the reset time may include a certain amount of error if there is any fluctuation of the oscillation frequency at starting the high-frequency clock oscillation. Therefore, the reset time must be considered an approximated value.
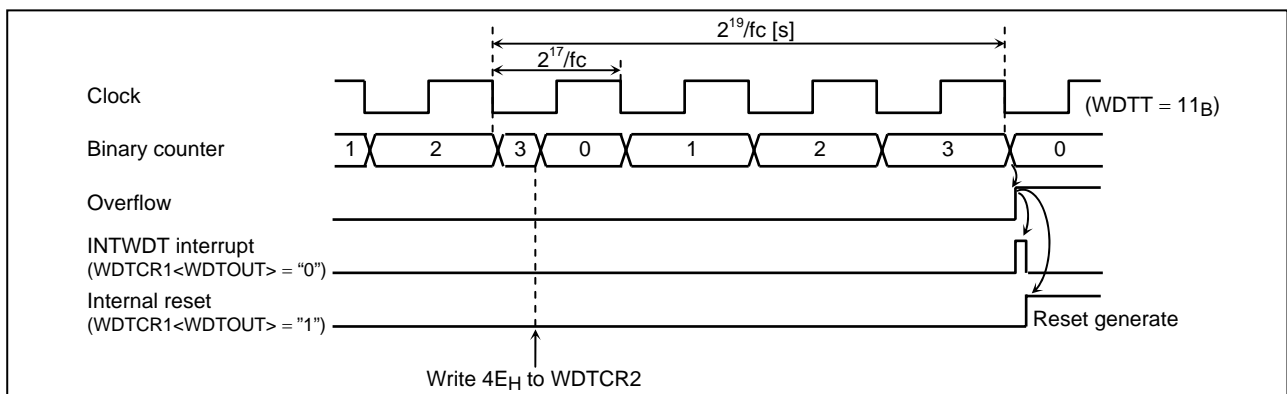

Figure 2.4.3 Watchdog Timer Interrupt/Reset

2.5   Address Trap

The watchdog timer control register 1, 2 shares its addresses with the control registers in case of address trap. These control registers for address trap are shown on Figure 2.5.1.
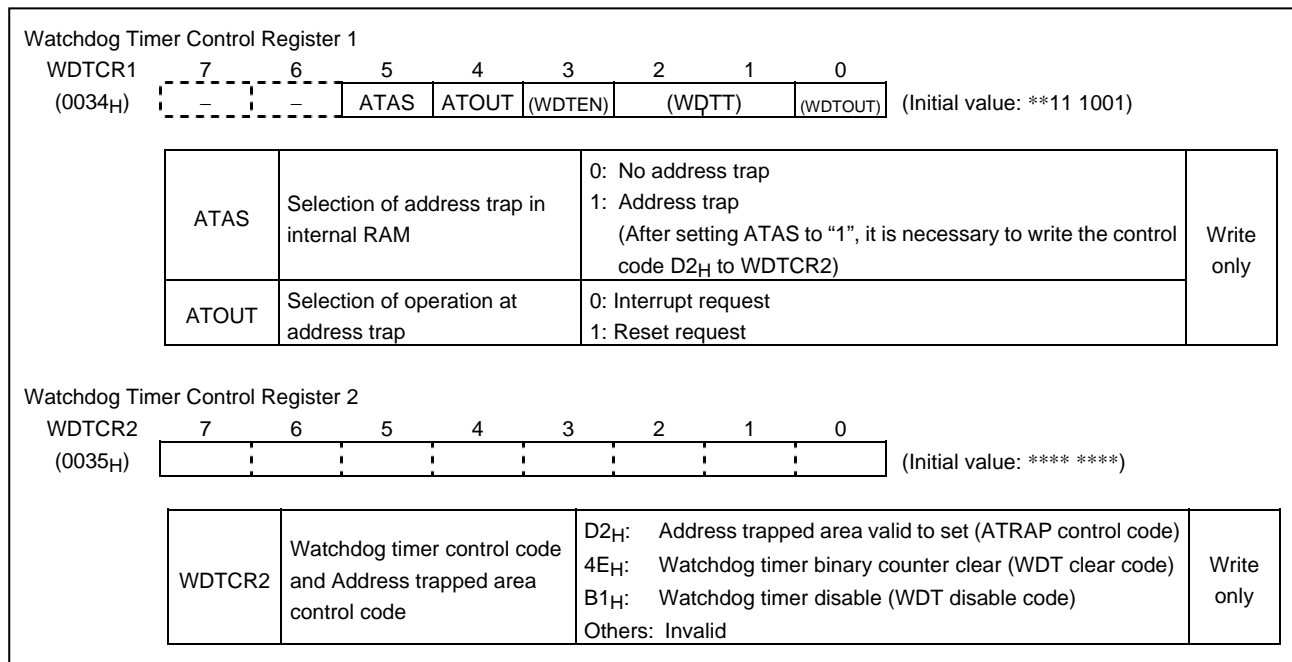

Figure 2.5.1 Watchdog Timer Control Registers

(1) Selection of address trap in internal RAM (ATAS)

Using WDTCR1<ATAS>, address trap or no address trap can be selected for the internal RAM area. To execute an instruction in the internal RAM area, set "0" in WDTCR1<ATAS>. Setting in WDTCR1<ATAS> becomes valid after control code D2H is written in WDTCR2. Executing an instruction in the SFR/DBR area generates an address trap unconditionally regardless of the setting in WDTCR1<ATAS>.

(2) Selection of operation at address trap (ATOUT)

As the operation at address trap either interrupt request or reset request can be selected by WDTCR1<ATOUT>.

2.6 Divider Output (DVO)

Approximately 50% duty pulse can be output using the divider output circuit, which is useful for piezoelectric buzzer drive. Divider output is from pin P51 ($\overline{\text{DVO}}$). The P51 output latch should be set to "1".

Note: Selection of divider output frequency must be made while divider output is disabled.
Also, in other words, when changing the state of the divider output frequency from enabled to disable, do not change the setting of the divider output frequency.

TBTCR	7	6	5	4	3	2	1	0	
(0036H)	DVOEN	DVOCK		(DV7CK)	(TBTEN)		(TBTCK)		(Initial value: 0000 0000)

	DVOEN	Divider output enable/disable	0: Disable 1: Enable				

				NORMAL1/2 Mode		SLOW, SLEEP Mode	
				DV7CK = 0	DV7CK = 1		
DVOCK	Divider output ($\overline{\text{DVO}}$) frequency selection [Hz]		00	$fc/2^{13}$	$fs/2^5$	$fs/2^5$	R/W
			01	$fc/2^{12}$	$fs/2^4$	$fs/2^4$	
			10	$fc/2^{11}$	$fs/2^3$	$fs/2^3$	
			11	$fc/2^{10}$	$fs/2^2$	$fs/2^2$	

Note: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], ∗: Don't care

Figure 2.6.1 Divider Output Control Register

Example: 1.95 kHz pulse output (at fc = 16.0 MHz)
SET   (P5DR).1   ;   P51 output latch ← "1"
LD   (TBTCR), 00000000B   ;   DVOCK ← "00"
LD   (TBTCR), 10000000B   ;   DVOEN ← "1"

Table 2.6.1 Divider Output Frequency (Example: at fc = 16.0 MHz, fs = 32.768 kHz)

DVOCK	Divider Output Frequency [Hz]		
	NORMAL1/2, IDLE1/2 Mode		SLOW, SLEEP Mode
	DV7CK = 0	DV7CK = 1	
00	1.953 k	1.024 k	1.024 k
01	3.906 k	2.048 k	2.048 k
10	7.813 k	4.096 k	4.096 k
11	15.625 k	8.192 k	8.192 k

(a) Configuration   (b) Timing Chart
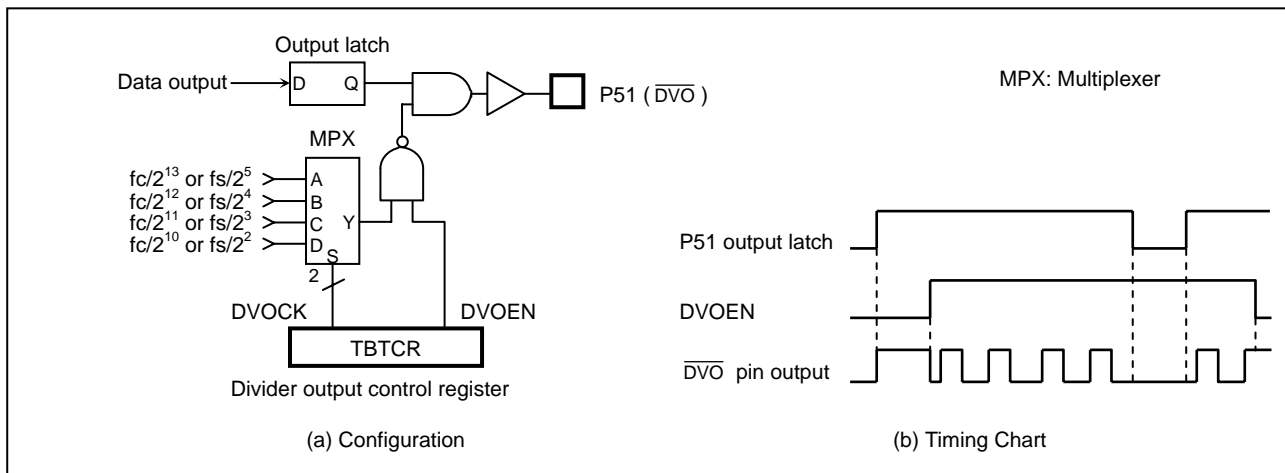
Figure 2.6.2 Divider Output

2.7   16-Bit Timer/Counter 1

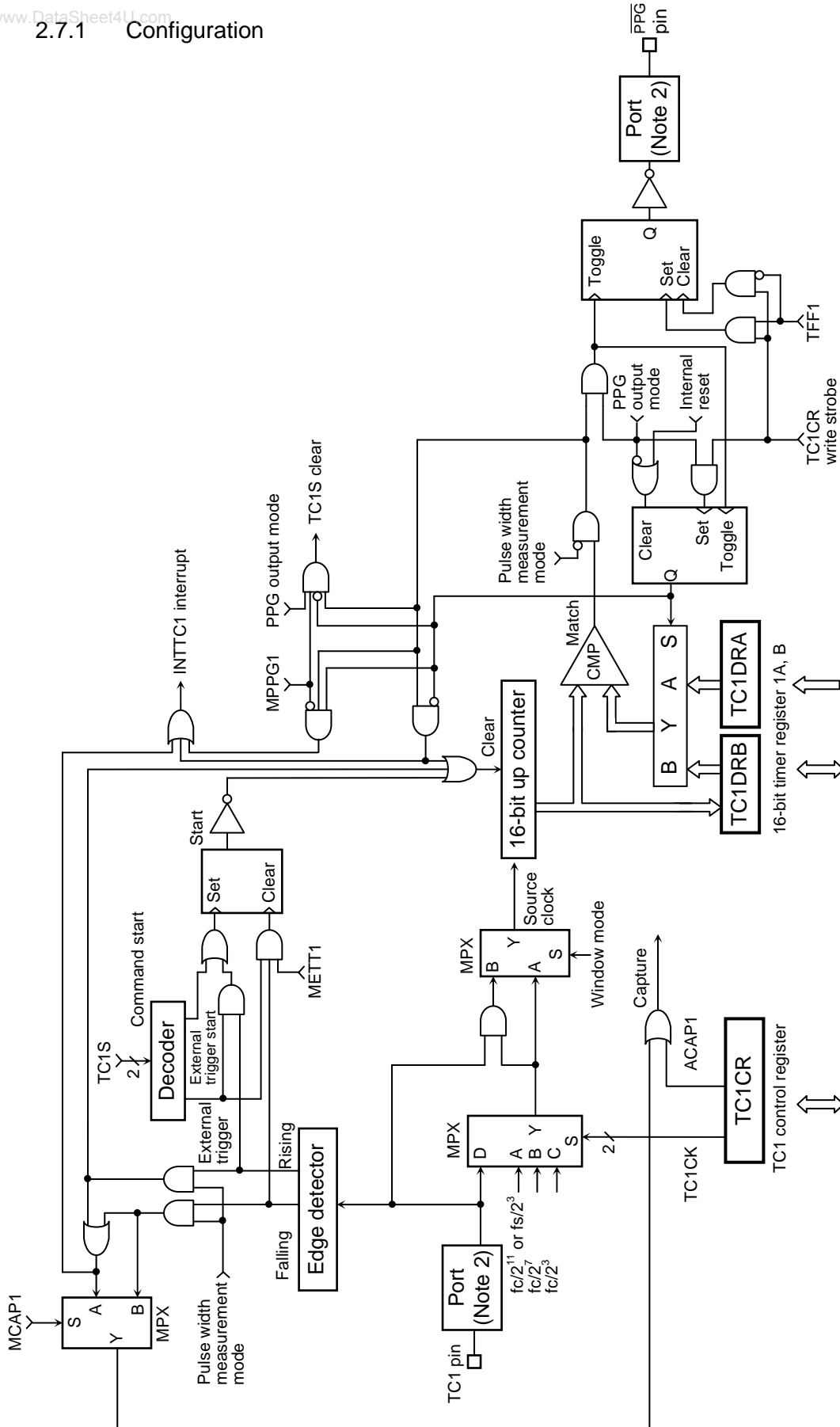### 2.7.1   Configuration

Figure 2.7.1 Timer/Counter 1 (TC1)

Note 1:   MPX: Multiplexer
          CMP: Comparator

Note 2:   When control input /output is used, I/O port setting should be set correctly.
          For details, refer to "2.2 I/O ports".

2.7.2 Control

The timer/counter 1 is controlled by a timer/counter 1 control register (TC1CR) and two 16-bit timer registers (TC1DRA and TC1DRB).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TC1DRA (0021,0020H) R/W				TC1DRAH (0021H)							TC1DRAL (0020H)				

(Initial value: 1111 1111 1111 1111)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TC1DRB (0023,0022H) R/W				TC1DRBH (0023H)							TC1DRBL (0022H)				

(Initial value: 1111 1111 1111 1111)

Note: TC1DRB should not be written except PPG mode.

	7	6	5	4	3	2	1	0
TC1CR (001FH)	TFF1	ACAP1 MCAP1 METT1 MPPG1	TC1S		TC1CK		TC1M	

(Initial value: 0000 0000)

TC1M	TC1 operating mode select	00: Timer/external trigger timer/event counter mode 01: Window mode 10: Pulse width measurement mode 11: PPG (Programmable pulse generate) output mode			

			NORMAL1/2, IDLE1/2 mode		SLOW1/2, SLEEP1/2 mode
TC1CK	TC1 source clock select [Hz]		DV7CK = 0	DV7CK = 1	
		00	$fc/2^{11}$	$fs/2^3$	$fs/2^3$
		01	$fc/2^7$	$fc/2^7$	–
		10	$fc/2^3$	$fc/2^3$	–
		11	External clock (TC1 pin input)		

			Timer	Extend	Event	Window	Pulse	PPG	
TC1S	TC1 start control	00: Stop and counter clear	○	○	○	○	○	○	R/W
		01: Command start	○	×	×	×	×	○	
		10: External trigger start at the rising edge	×	○	○	○	○	○	
		11: External trigger start at the falling edge	×	○	○	○	○	○	

ACAP1	Auto capture control	0: Auto-capture disable	1: Auto-capture enable
MCAP	Pulse width measurement mode control	0: Double edge capture	1: Single edge capture
METT1	External trigger timer mode control	0: Trigger start	1: Trigger start and stop
MPPG1	PPG output control	0: Continuous pulse generation	1: One-shot
TFF1	Time F/F1 control	0: Clear	1: Set

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz]

Note 2: The timer register consists of two shift registers. A value set in the timer register is put in effect at the rising edge of the first source clock pulse that occurs after the upper data (TC1DRAH and TC1DRBH) are written. Therefore, the lower byte must be written before the upper byte (it is recommended that a 16-bit access instruction be used in writing). Writing only the lower data (TC1DRAL and TC1DRBL) does not put the setting of the timer register in effect.

Note 3: Set the mode, source clock, PPG control and timer F/F control when TC1 stops (TC1S = 00).

Note 4: Auto-capture can be used in only timer, event counter, and window modes.

Note 5: Values to be loaded to timer registers must satisfy the following condition.
TC1DRA > TC1DRB > 1 (PPG output mode), TC1DRA > 1 (others)

Note 6: Always write "0" to TFF1 except PPG output mode.

Note 7: Writing to the TC1DRB is not possible unless TC1 is set to the PPG output mode.

Note 8: On entering STOP mode, the TC1 start control (TC1S) is cleared to "00" automatically. So, the timer stops. Once the STOP mode has been released, to start using the timer counter, set TC1S again.

Note 9: Use the auto-capture function in the operative condition of TC1. A captured value may not be fixed if it's read after the execution of the timer stop or auto-capture disable. Read the capture value in a capture enabled condition.

> Note 10: Since the up-counter value is captured into TC1DRB by the source clock of up-counter after setting
> TC1CR<ACAP1> to "1". Therefore, to read the captured value, wait at least one cycle of the internal source
> clock before reading TC1DRB for the first time.

Figure 2.7.2 Timer Registers and TC1 Control Register

2.7.3   Function

Timer/counter 1 has six operating modes: timer, external trigger timer, event counter, window, pulse width measurement, programmable pulse generator output mode.

(1) Timer mode

In this mode, counting up is performed using the internal clock. The contents of TC1DRA are compared with the contents of up counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared to "0". Counting up resumes after the counter is cleared. The current contents of up counter can be transferred to TC1DRB by setting TC1CR<ACAP1> to "1" (Auto capture function). Use the auto-capture function in the operative condition of TC1. A captured value may not be fixed if it's read after the execution of the timer stop or auto-capture disable. Read the capture value in a capture enabled condition. Since the up-counter value is captured into TC1DRB by the source clock of up-counter after setting TC1CR<ACAP1> to "1". Therefore, to read the captured value, wait at least one cycle of the internal source clock before reading TC1DRB for the first time.

Table 2.7.1 Source Clock (internal clock) for Timer/Counter 1 (Example: at fc = 16 MHz, fs = 32.768kHz)

| TC1CK | NORMAL1/2, IDLE1/2 Mode | | | | SLOW1/2, SLEEP1/2 Mode | |
| | DV7CK = 0 | | DV7CK = 1 | | | |
	Resolution [µs]	Maximum Time Setting [s]	Resolution [µs]	Maximum Time Setting [s]	Resolution [µs]	Maximum Time Setting [s]
00	128	8.39	244.14	16.0	244.14	16.0
01	8.0	0.524	8.0	0.524	–	–
10	0.5	32.77 m	0.5	32.77 m	–	–

Example 1: Sets the timer mode with source clock $fc/2^{11}$ [Hz] and generates an interrupt 1 second later
(at fc = 16 MHz, DV7CK = 0)

```
            LDW        (TC1DRA), 1E84H           ;  Sets the timer register
                                                    (1 s ÷ 2¹¹/fc = 1E84H)
            DI                                   ;  IMF = "0"
            SET        (EIRL). 5                 ;  Enable INTTC1
            EI                                   ;  IMF = "1"
            LD         (TC1CR), 00000000B        ;  TFF1 ← "0", TC1CK ← "00", TC1M ← "00"
            LD         (TC1CR), 00010000B        ;  Starts TC1
Example 2: Auto-capture
            LD         (TC1CR), 01010000B        ;  ACAP1 ← "1" (Capture)
            LD       WA, (TC1DRB)                ;  Reads the capture value
```

Note :   Since the up-counter value is captured into TC1DRB by the source clock of up-counter after setting
TC1CR<ACAP1> to "1". Therefore, to read the captured value, wait at least one cycle of the internal source
clock before reading TC1DRB for the first time.

Figure 2.7.3 Timer Mode Timing Chart

(2) External trigger timer mode

In this mode, counting up is started by an external trigger. This trigger is the edge of the TC1 pin input. Either the rising or falling edge can be selected with TC1S. Source clock is an internal clock. The contents of TC1DRA is compared with the contents of up counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared to "0" and halted. The counter is restarted by the selected edge of the TC1 pin input.

When TC1CR<METT1> is "1", inputting the edge to the reverse direction of the trigger edge to start counting clears the counter, and the counter is stopped. Inputting a constant pulse width can generate interrupts. When TC1CR<METT1> is "0", the reverse directive edge input is ignored. The TC1 pin input edge before a match detection is also ignored.

The TC1 pin input has the noise rejection; therefore, pulses of 4/fc [s] or less are rejected as noise. A pulse width of 12/fc [s] or more is required for edge detection in NORMAL1/2 or IDLE1/2 mode. The noise rejection circuit is turned off in SLOW1/2 and SLEEP1/2 modes. But, a pulse width of one machine cycle or more is required.

Example 1: Detects rising edge in TC1 pin input and generates an interrupt 100 μs later.
(at fc = 16 MHz, DV7CK = 0)

DI		;	IMF = "0"
LDW	(TC1DRA), 00C8H	;	100 μs ÷ 2^3/fc = C8H
SET	(EIRL). 5	;	INTTC1 interrupt enable
EI		;	IMF = "1"
LD	(TC1CR), 00001000B	;	TFF1 = "0", TC1CK = "10", TC1M = "00"
LD	(TC1CR), 00101000B	;	TC1 external trigger start, METT1 ="0"

Example 2: Generates an interrupt, inputting "L" level pulse (pulse width: 4 ms or more) to the TC1 pin.
(at fc = 16 MHz)

DI		;	IMF = "0"
LDW	(TC1DRA), 1F40H	;	4 ms ÷ 2^3/fc = 1F40H
SET	(EIRL). 5	;	INTTC1 interrupt enable
EI		;	IMF = "1"
LD	(TC1CR), 01001000B	;	TFF1 = "0", TC1CK = "10", TC1M = "00"
LD	(TC1CR), 01111000B	;	TC1 external trigger start, METT1 = 1
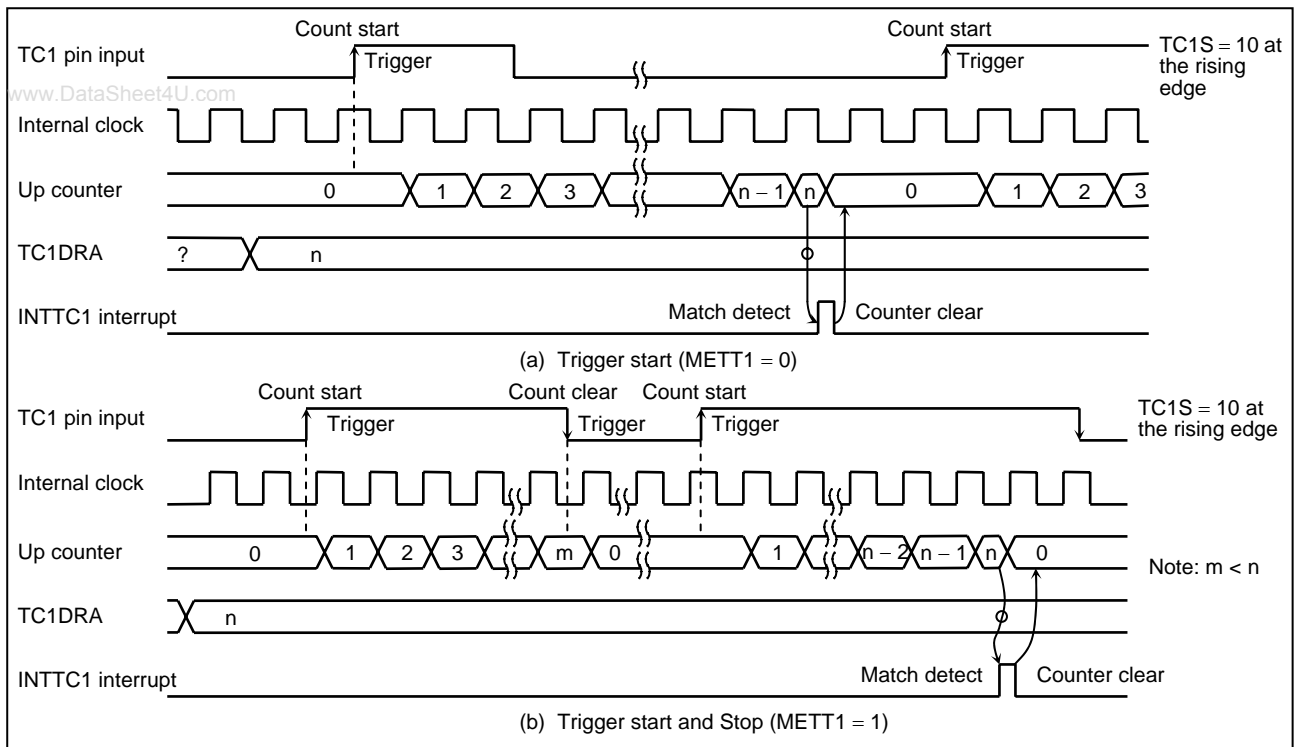
Figure 2.7.4 External Trigger Timer Mode Timing Chart

(3) Event counter mode

In this mode, events are counted at the edge of the TC1 pin input (either the rising or falling edge can be selected with the external trigger TC1CR<TC1S>). The contents of TC1DRA are compared with the contents of up counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. After the counter is cleared, the up counter starts counting by TC1 input edge. Match detect is executed on other edge of count-up. A match can not be detected and INTTC1 is not generated when the pulse is still in same state. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width.

Setting TC1CR<ACAP1> to "1" transfers the current contents of up counter to TC1DRB (Auto-capture function). Use the auto-capture function in the operative condition of TC1. A captured value may not be fixed if it's read after the execution of the timer stop or auto-capture disable. Read the capture value in a capture enabled condition. Since the up-counter value is captured into TC1DRB by the source clock of up-counter after setting TC1CR<ACAP1> to "1". Therefore, to read the captured value, wait at least one cycle of the internal source clock before reading TC1DRB for the first time.
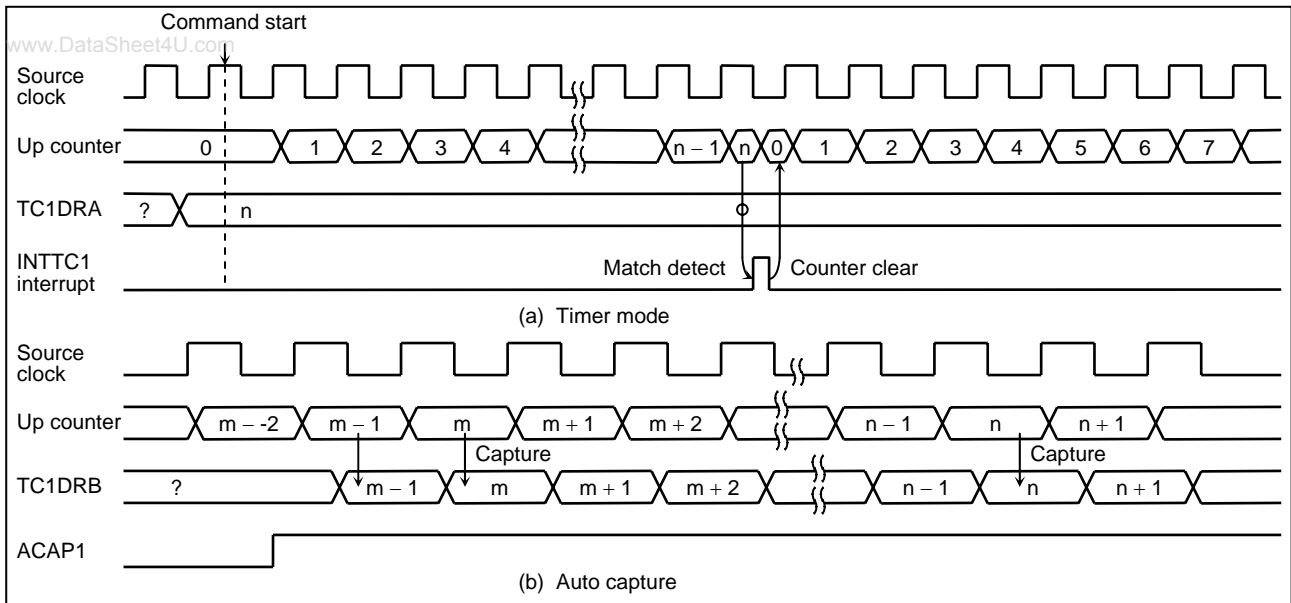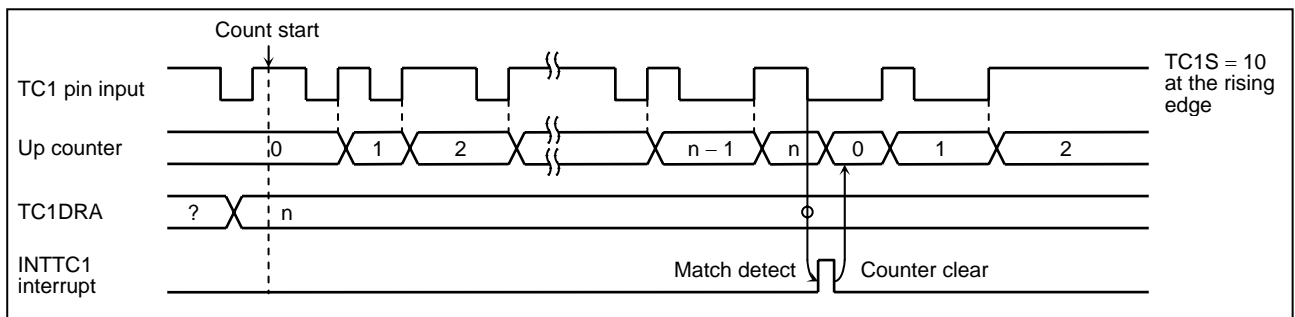

Figure 2.7.5 Event Counter Mode Timing Chart

Table 2.7.2 Timer/Counter 1 External Clock Source

	Minimum Input Pulse Width [s]	
	NORMAL1/2, IDLE1/2 Mode	SLOW1/2, SLEEP1/2 Mode
"H" width	2^3/fc	2^3/fs
"L" width	2^3/fc	2^3/fs

（4） Window mode

In this mode, counting up is performed on the rising edge of the pulse that is the logical AND-ed product of the TC1 pin input (Window pulse) and an internal clock. The contents of TC1DRA are compared with the contents of up counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. It is possible to select either positive logic or negative logic for the TC1 pin input (by using the TC1 start control TC1CR<TC1S>).

The maximum frequency that can be applied to the pin must be such that the related count can be analyzed by program. To put another way, the frequency of the applied pulse must be sufficiently low, compared with that of the internally set source clock.
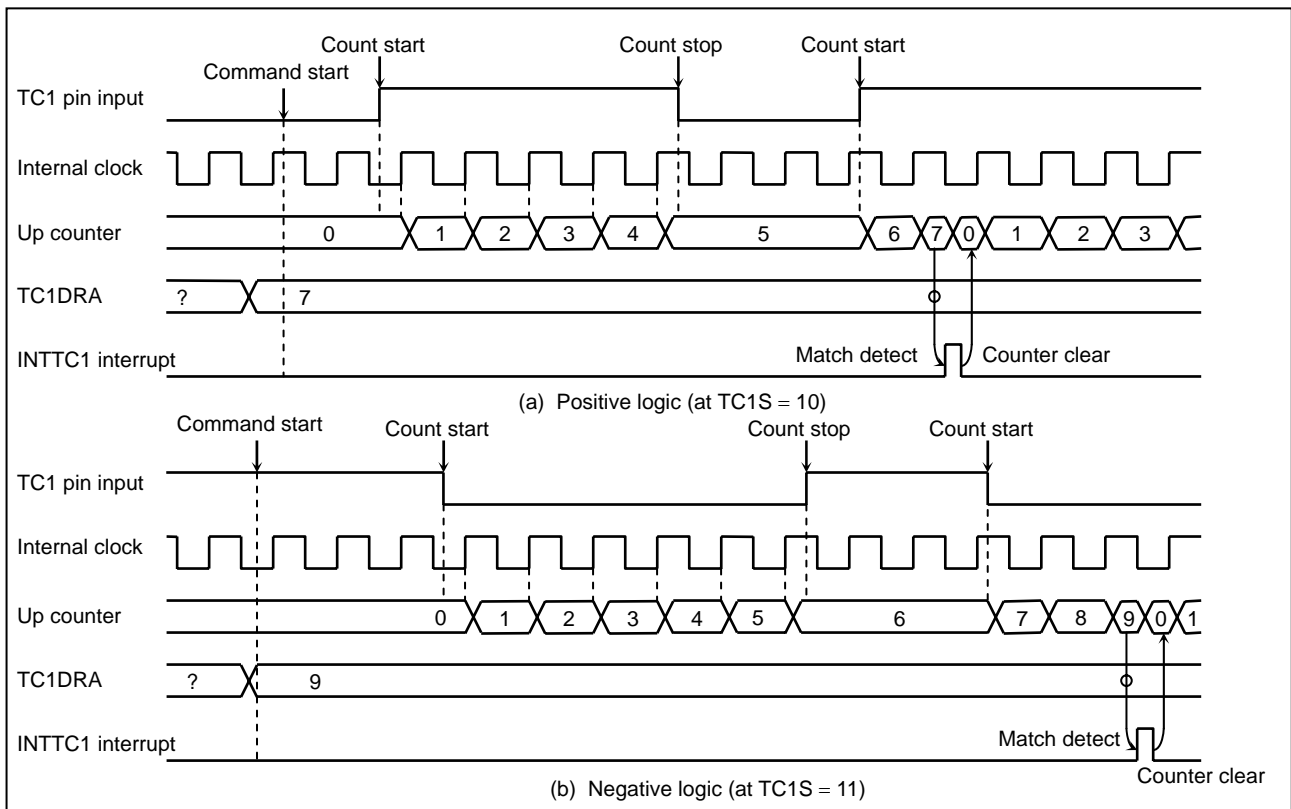
Figure 2.7.6 Window Mode Timing Chart

(5) Pulse width measurement mode

In this mode, counting is started by the external trigger (Set to external trigger start by TC1CR<TC1S>). The trigger can be selected either the rising or falling edge of the TC1 pin input. The source clock is used an internal clock. On the next falling (Rising) edge, the counter contents are transferred to TC1DRB and an INTTC1 interrupt is generated. The counter is cleared when the single edge capture mode (TC1CR<MCAP> = "1") is set. When double edge capture (TC1CR<MCAP> = "0") is set, the counter continues and, at the next rising (Falling) edge, the counter contents are again transferred to TC1DRB. If a falling (Rising) edge capture value is required, it is necessary to read out TC1DRB contents until a rising (Falling) edge is detected. Falling or rising edge is selected with the external trigger TC1CR<TC1S>, and single edge or double edge is selected with TC1CR<MCAP1>.

Note 1: Be sure to read the captured value from TC1DRB before the next trigger edge is detected. If fail to read it, it becomes undefined. It is recommended that a 16-bit access instruction be used to read from TC1DRB.

Note 2: If either the falling or rising edge is used in capturing values, the counter stops at "1" after a value has been captured until the next edge is detected. So, the value captured next will become "1" larger than the value captured right after capturing starts.

Note 3: The first captured value after the timer starts may be read incorrectively, therefore, ignore the first captured value.

```
Example:  Duty measurement (resolution fc/2^7 [Hz])
                 CLR      (INTTC1SW). 0          ;   INTTC1 service switch initial setting
                 LD       (TC1CR), 00000110B     ;   Sets the TC1 mode and source clock
                 DI                              ;   IMF = "0"
                 SET      (EIRL). 5              ;   Enables INTTC1
                 EI                              ;   IMF = "1"
                 LD       (TC1CR), 00100110B     ;   Starts TC1 with an external trigger at
                                                 :   MCAP1 = 0
PINTTC1:         CPL      (INTTC1SW). 0          ;   Inverts INTTC1 service switch
                 JRS      F, SINTTC1
                 LD       A, (TC1DRBL)           ;   Reads TC1DRB ("H" level pulse width)
                 LD       W, (TC1DRBH)
                 RETI
SINTTC1:         LD       L, (TC1DRBL)           ;   Reads TC1DRB (Period)
                 LD       H, (TC1DRBH)
                                                 ;   Duty calculation
                 RETI

VINTTC1:         DW       PINTTC1
```
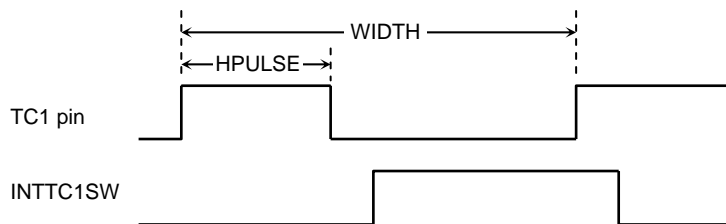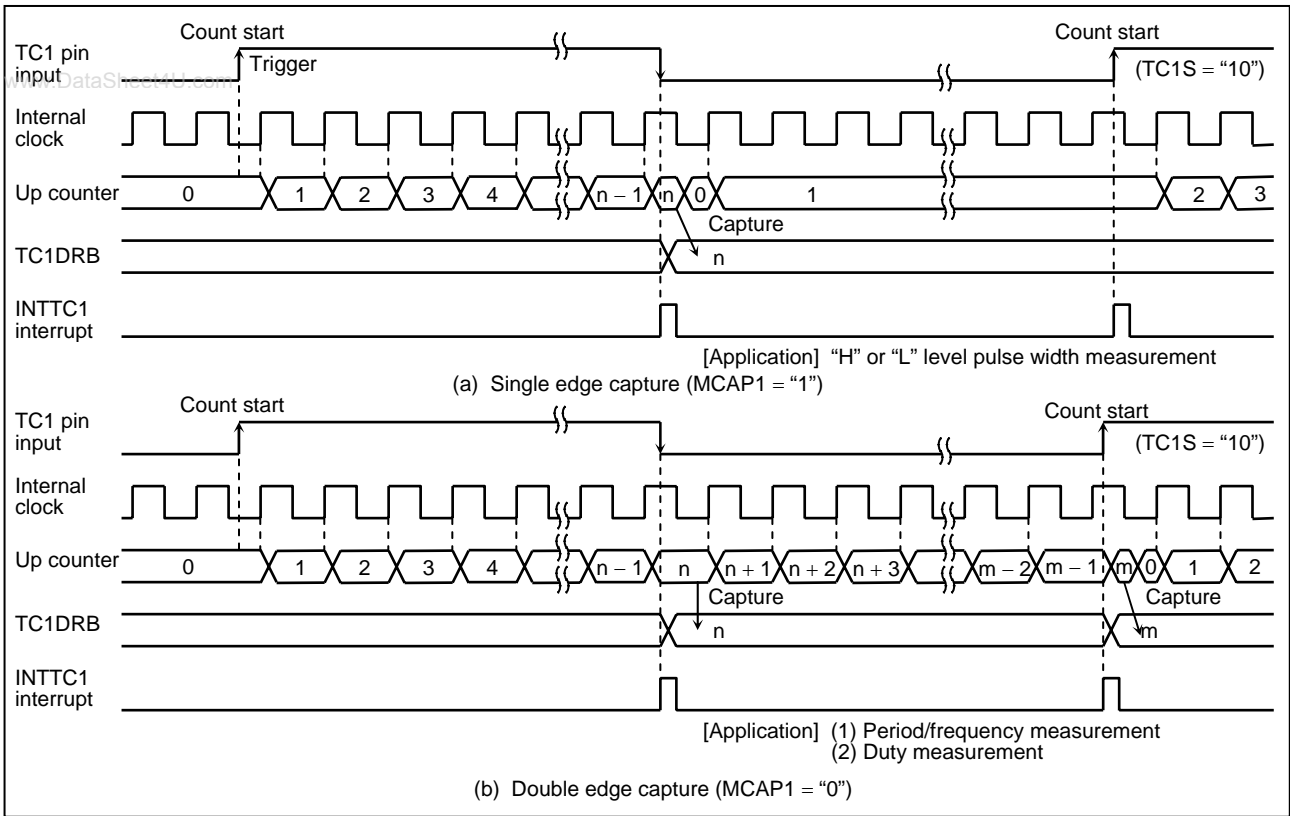
Figure 2.7.7 Pulse Measurement Mode Timing Chart

(6) Programmable pulse generate (PPG) output mode

The PPG output mode is intended to output pulses having an arbitrary duty cycle selected using two timer registers.

The timer starts at an edge (Rising or falling edge, that is, the same edge type as selected with the external trigger edge select bits (TC1CR<TC1S>) or on a command. Its source clock is an internal clock. Once the timer starts running, the timer F/F1 is inverted when the counter matches TC1DRB, generating the INTTC1 interrupt. The counter keeps up-counting, and when counter matches TC1DRA, the timer F/F1 is inverted, generating an INTTC1 interrupt. If TC1CR<MPPG1> was previously set to "1" (One shot), TC1S is cleared to "00" automatically, causing the timer to stop. If TC1CR<MPPG1> was previously cleared to "0" (Continuous pulse generation), the counter is cleared, resulting in the counter keeping to run and the PPG output being continued. If TC1CR<TC1S> is reset to "00" (One-shot-based automatic stop is included) during PPG output, the P50 ($\overline{\text{PPG}}$) pin holds the same level that it does just before the counter stops. In PPG output mode, set the output latch of port P50 to "1". The timer F/F1 is cleared to "0" at a reset. In addition, a positive or negative pulse can be output because the output level can be set up at a start, using TC1CR<TFF1>. The P50 ($\overline{\text{PPG}}$) pin outputs an inversion of the timer F/F1 output level. It is impossible to write to TC1DRB unless the PPG output mode is set.

Note 1: To change the content of the timer register when the timer is running, change it to a sufficiently large value, compared with the current count. If the timer register content is changed to a value smaller than the current count when the timer is running, it is likely that unintended pulses may be output.

Note 2: Do not change TC1CR<TFF1> when the timer is running.
TC1CR<TFF1> can be set correctly only at initialization (after a reset). When the timer is stopped during PPG output, if the PPG output is at a logic state opposite to the PPG that when the timer starts, it will become impossible to set TC1CR<TFF1> correctly (An attempt to program TC1CR<TFF1> will cause a state opposite to the programmed one to be set in the bit). Once the timer has stopped, putting the PPG output securely on an arbitrary level requires initializing the timer F/F1. To initialize it, put TC1CR<TC1M> in the timer mode again (It is unnecessary to start the timer mode), and then put it in the PPG output mode again. At the same time, set TC1CR<TFF1>.

Note 3: In the PPG output mode, a value set in the timer register must satisfy: TC1DRA > TC1DRB

Example: Pulse output "H" level 800 µs, "L" level 200 µs (at fc = 16 MHz, DV7CK = 0)

SET	(P5DR). 0	;	P50 output latch ← 1
LD	(TC1CR), 10001011B	;	Sets the PPG output mode
LDW	(TC1DRA), 07D0H	;	Sets the period (1 ms ÷ 2^3/fc = 07D0H)
LDW	(TC1DRB), 0190H	;	Sets "L" level pulse width
			(200 µs ÷ 2^3/fc = 0190H)
LD	(TC1CR), 10011011B	;	Starts

Figure 2.7.8 $\overline{PPG}$ Output

(a) Continuous pulse generation (with TC1S = 01)

(b) One-shot (with TC1S = 10)
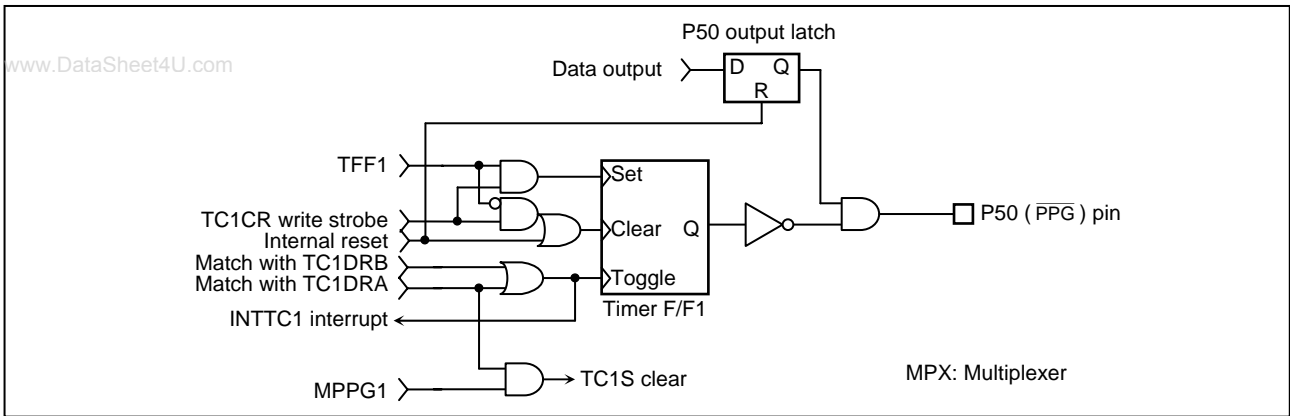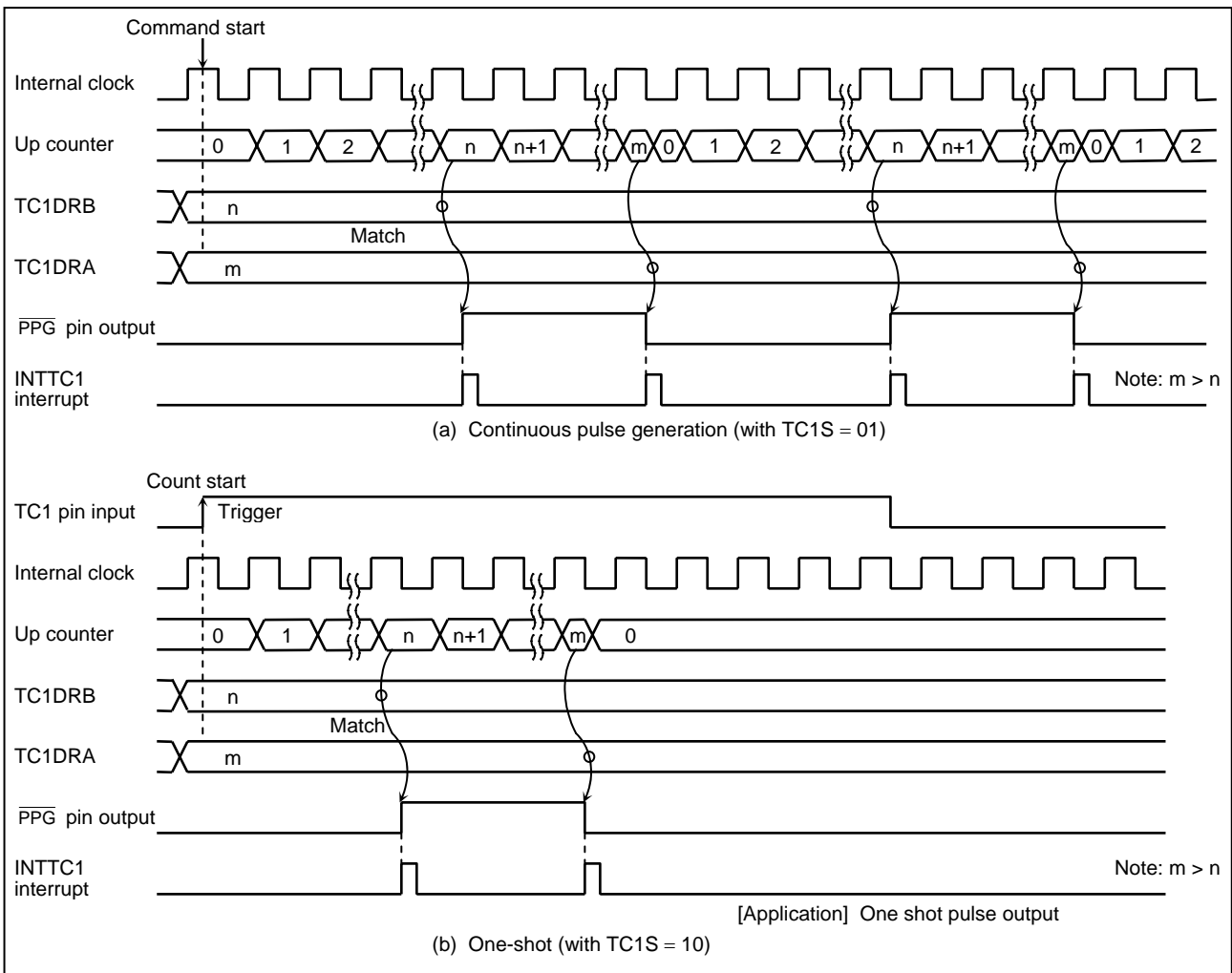
Figure 2.7.9 PPG Output Mode Timing Chart

2.8 16-Bit Timer/Counter 2

2.8.1 Configuration
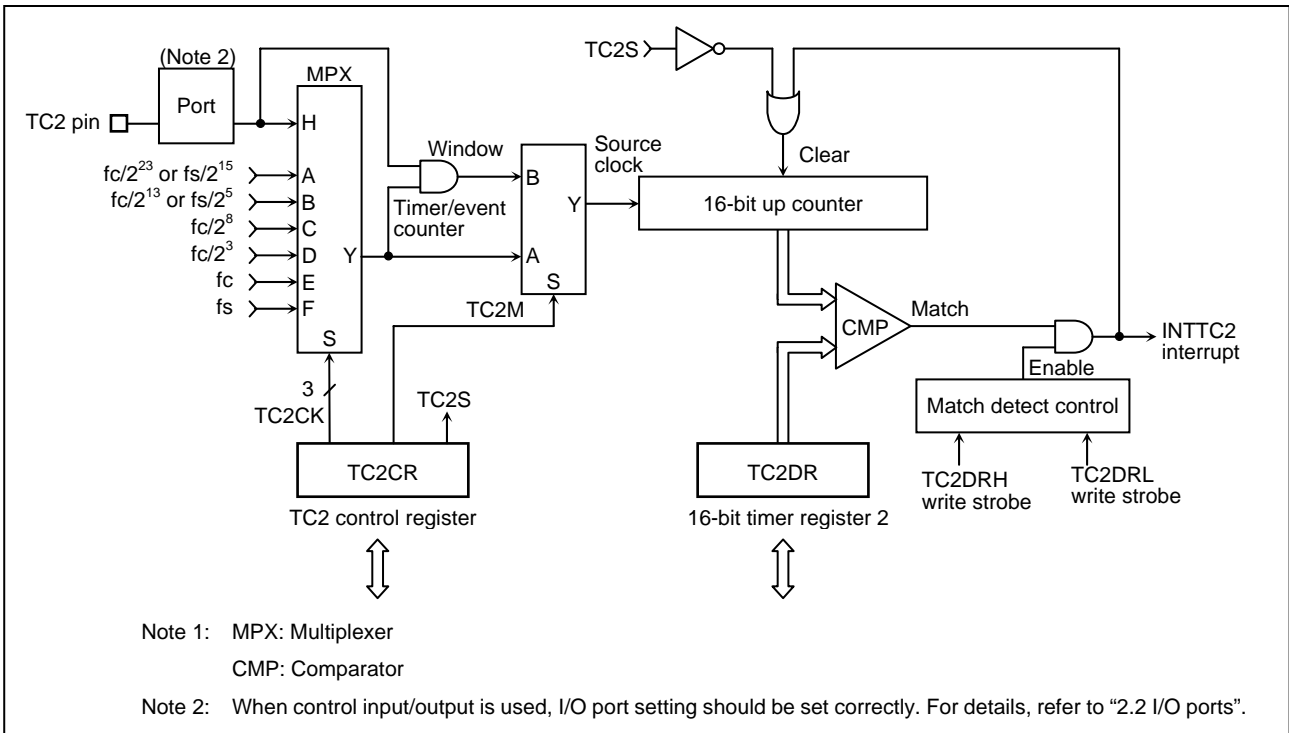
Figure 2.8.1 Timer/Counter 2 (TC2A)

2.8.2   Control

The timer/counter 2 is controlled by a timer/counter 2 control register (TC2CR) and a 16-bit timer register 2 (TC2DR). Reset does not affect TC2DR.

TC2DR (0025, 0024H) R/W	Bits 15–9: TC2DRH (0025H); Bits 8–0: TC2DRL (0024H)
TC2CR (0013H)	Bit 5: TC2S; Bits 4–2: TC2CK; Bit 0: TC2M (Initial value: **00 00*0)

TC2M	TC2 operating mode select	0: Timer/event counter mode 1: Window mode				R/W

			NORMAL1/2, IDLE1/2 mode		SLOW1/2 mode	SLEEP1/2 mode
			DV7CK $= 0$	DV7CK $= 1$		
TC2CK	TC2 source clock select [Hz]	000	$fc/2^{23}$	$fs/2^{15}$	$fs/2^{15}$	$fs/2^{15}$
		001	$fc/2^{13}$	$fs/2^{5}$	$fs/2^{5}$	$fs/2^{5}$
		010	$fc/2^{8}$	$fc/2^{8}$	–	–
		011	$fc/2^{3}$	$fc/2^{3}$	–	–
		100	–	–	fc (Note 7)	–
		101	fs	fs	–	–
		110	Reserved			
		111	External clock (TC2 pin input)			

TC2S	TC2 start control	0: Stop and counter clear 1: Start

Note 1:   fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 2:   When writing to the Timer Register 2 (TC2DR), always write to the lower side (TC2DRL) and then the upper side (TC2DRH) in that order. Writing to only the lower side (TC2DRL) or the upper side (TC2DRH) has no effect.

Note 3:   The timer register 2 (TC2DR) uses the value previously set in it for coincidence detection until data is written to the upper side (TC2DRH) after writing data to the lower side (TC2DRL).

Note 4:   Set the mode and source clock when the TC2 stops (TC2S = 0).

Note 5:   Values to be loaded to the timer register must satisfy the following condition.

$$TC2DR > 1 \ (TC2DR_{15 \ to} \ TC2DR_{11} > 1 \text{ at warm up})$$

Note 6:   If a read instruction is executed for TC2CR, read data of bit 7, 6 and 1 are unstable.

Note 7:   The high-frequency clock(fc) can be selected only when the timer mode at SLOW2 mode is selected.

Note 8:   On entering STOP mode, the TC2 start control (TC2S) is cleared to "0" automatically. So, the timer stops. Once the STOP mode has been released, to start using the timer counter, set TC2S again.

Figure 2.8.2 Timer Register 2 and TC2 Control Register

2.8.3 Function

The timer/counter 2 has three operating modes: timer, event counter and window modes.

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of TC2DR are compared with the contents of up counter. If a match is found, a timer/counter 2 interrupt (INTTC2) is generated, and the counter is cleared. Counting up is resumed after the counter is cleared.

When fc is selected for source clock at SLOW2 mode, lower 11-bits of TC2DR are ignored and generated a interrupt by matching upper 5-bits. Though, in this situation, it is necessary to set TC2DRH only.

Table 2.8.1 Source Clock (Internal clock) for Timer/Counter 2 (at fc = 16 MHz)

TC2CK	NORMAL1/2, IDLE1/2 Mode				SLOW1/2 Mode		SLEEP1/2 Mode	
	DV7CK = 0		DV7CK = 1		Resolution	Maximum Time Setting	Resolution	Maximum Time Setting
	Resolution	Maximum Time Setting	Resolution	Maximum Time Setting				
000	524.29 ms	9.54 h	1.00 s	18.20 h	1.00 s	18.20 h	1.00 s	18.20 h
001	512.00 μs	33.55 s	0.98 ms	1.07 min	0.98 ms	1.07 min	0.98 ms	1.07 min
010	16.00 μs	1.05 s	16.00 μs	1.05 s	–	–	–	–
011	0.50 μs	32.77 ms	0.50 μs	32.77 ms	–	–	–	–
100	–	–	–	–	62.5 ns (Note)	–	–	–
101	30.52 μs	2.00 s	30.52 μs	2.00 s	–	–	–	–

Note: When fc is selected as the source clock in timer mode, it is used at warm-up for switching from SLOW2 mode to NORMAL2 mode.

Example: Sets the timer mode with source clock $fc/2^3$ [Hz] and generates an interrupt every 25 ms (at fc = 16 MHz).

LDW	(TC2DR), 0C350H	;	Sets TC2DR (25 ms ÷ 2^3/fc = C350H)
DI		;	IMF = "0"
SET	(EIRE). 4	;	Enables INTTC2 interrupt
EI		;	IMF = "1"
LD	(TC2CR), 00001100B	;	TC2CK ← "011", TC2M ← "0"
LD	(TC2CR), 00101100B	;	Starts TC2

(2) Event counter mode

    In this mode, events are counted on the rising edge of the TC2 pin input. The contents of TC2DR are compared with the contents of the up counter. If a match is found, an INTTC2 interrupt is generated, and the counter is cleared. The minimum input pulse width of TC2 pin is shown in Table 2.8.2. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width. Match detect is executed on the falling edge of the TC2 pin. A match can not be detected and INTTC2 is not generated when the pulse is still in a falling state.

Example: Sets the event counter mode and generates an INTTC2 interrupt 640 counts later.

```
LDW     (TC2DR), 640            ; Sets TC2DR
DI                              ; IMF = "0"
SET     (EIRE). 4               ; Enables INTTC2 interrupt
EI                              ; IMF = "1"
LD      (TC2CR), 00011100B      ; TC2CK ← "111", TC2M ← "0"
LD      (TC2CR), 00111100B      ; Starts TC2
```

Table 2.8.2 Timer/Counter 2 External Clock Source

	Minimum Input Pulse Width [s]	
	NORMAL1/2, IDLE1/2 Mode	SLOW1/2, SLEEP1/2 Mode
"H" width	2^3/fc	2^3/fs
"L" width	2^3/fc	2^3/fs

(3) Window mode

In this mode, counting up performed on the rising edge of an internal clock during TC2 external pin input (Window pulse) is "H" level. The contents of TC2DR are compared with the contents of up counter. If a match found, an INTTC2 interrupt is generated, and the up-counter is cleared.

The maximum applied frequency (TC2 input) must be considerably slower than the selected internal clock.

Note: In the window mode, before the SLOW/SLEEP mode is entered, the timer should be halted by setting TC2CR<TC2S> to "0".

Example: Generates an interrupt, inputting "H" level pulse width of 120 ms or more.
(at fc = 16 MHz, DV7CK = 0)

LDW	(TC2DR), 00EAH	;	Sets TC2DR (120 ms ÷ 2^{13}/fc = 00EAH)
DI		;	IMF = "0"
SET	(EIRE). 4	;	Enables INTTC2 interrupt
EI		;	IMF = "1"
LD	(TC2CR), 00000101B	;	TC2CK ← "001", TC1M ← "1"
LD	(TC2CR), 00100101B	;	Starts TC2
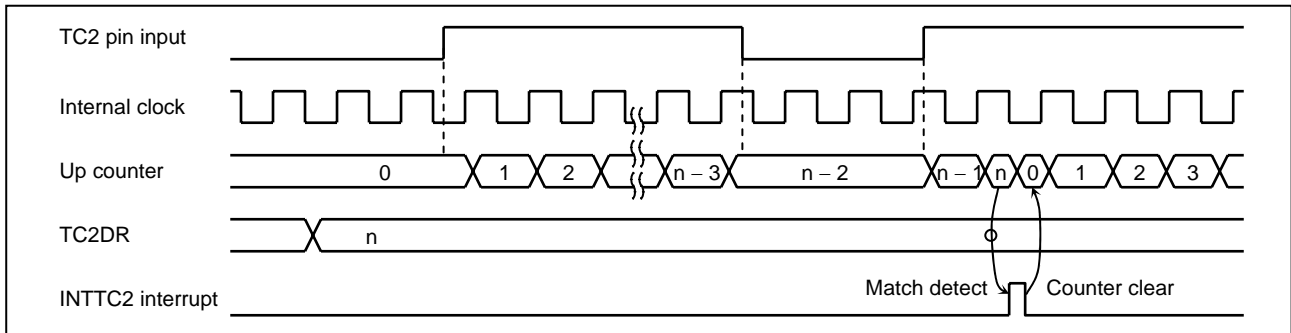
Figure 2.8.3 Window Mode Timing Chart

2.9 8-Bit Timer/Counter 3

2.9.1 Configuration
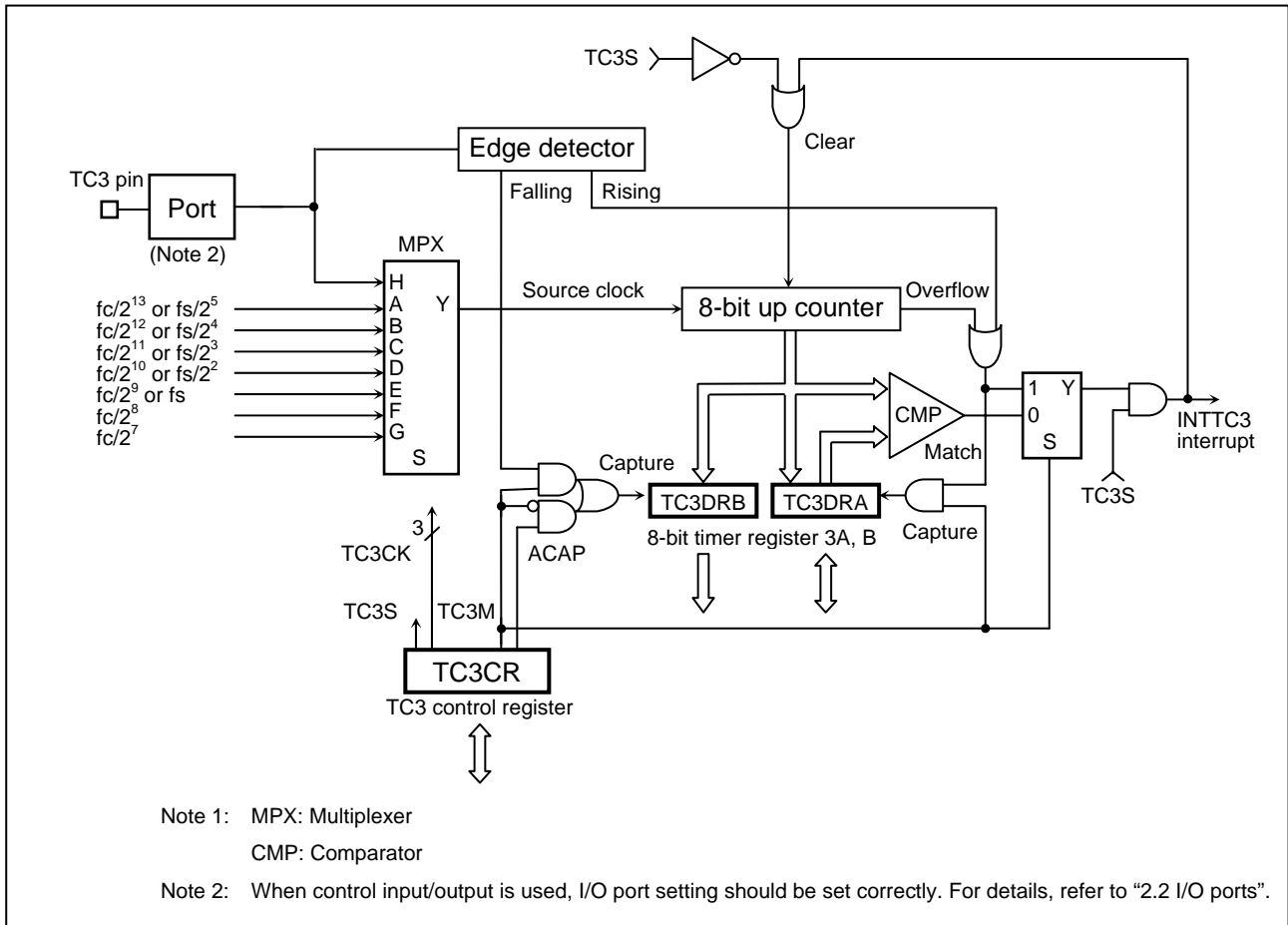
Figure 2.9.1 Timer/Counter 3 (TC3)

2.9.2　Control

The timer/counter 3 is controlled by a timer/counter 3 control register (TC3CR) and two 8-bit timer registers (TC3DRA and TC3DRB).

TC3M	TC3 operation mode set	0: Timer/event counter 1: Capture				
TC3CK	TC3 source clock select [Hz]		NORMAL1/2, IDLE1/2 mode		SLOW1/2, SLEEP1/2 mode	R/W
			DV7CK = 0	DV7CK = 1		
		000	$fc/2^{13}$	$fs/2^5$	$fs/2^5$	
		001	$fc/2^{12}$	$fs/2^4$	$fs/2^4$	
		010	$fc/2^{11}$	$fs/2^3$	$fs/2^3$	
		011	$fc/2^{10}$	$fs/2^2$	$fs/2^2$	
		100	$fc/2^9$	$fs/2$	$fs/2$	
		101	$fc/2^8$	$fc/2^8$	−	
		110	$fc/2^7$	$fc/2^7$	−	
		111	External clock (TC3 pin input)			
TC3S	TC3 start select	0: Stop and clear 1: Start				
ACAP	Auto-capture control	0: − 1: Auto capture enable				

Note 1:　fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], ∗: Don't care

Note 2:　Set the mode and the source clock when the TC3 stops (TC3S = 0).

Note 3:　Values to be loaded into timer register 3A must satisfy the following condition.

　　　　　TC3DRA > 1 (in the timer and event counter mode)

Note 4:　Auto-capture can be used only in the timer and event counter mode.

Note 5:　If a read instruction is executed for TC3CR, read data for bits 7 and 5 are unstable.

Note 6:　During TC3 operation, do not change TC3DRA.

Note 7:　On entering STOP mode, TC3 start control (TC3S) is cleared to "0" automatically, so the timer stops.

　　　　　Once the STOP mode has been released, to start using the timer counter, set TC3S again.

Figure 2.9.2　Timer Register 3 and TC3 Control Register

2.9.3 Function

The timer/counter 3 has three operating modes: timer, event counter, and capture mode.

（1） Timer mode

In this mode, the internal clock is used for counting up. The contents of TC3DRA are compared with the contents of up counter. If a match is found, a timer/counter 3 interrupt (INTTC3) is generated, and the up counter is cleared.

The current contents of up counter are loaded into TC3DRB by setting TC3CR<ACAP> to "1" (Auto-capture function). The contents of up counter can be easily confirmed by executing the read instruction (RD instruction) of TC3DRB. Loading the contents of up counter is not synchronized with counting up. The contents of over flow (FFH) and 00H can not be loaded correctly. It is necessary to consider the count cycle.

Table 2.9.1 Source Clock (Internal clock) for Timer/Counter 3 (Example: at fc = 16 MHz)
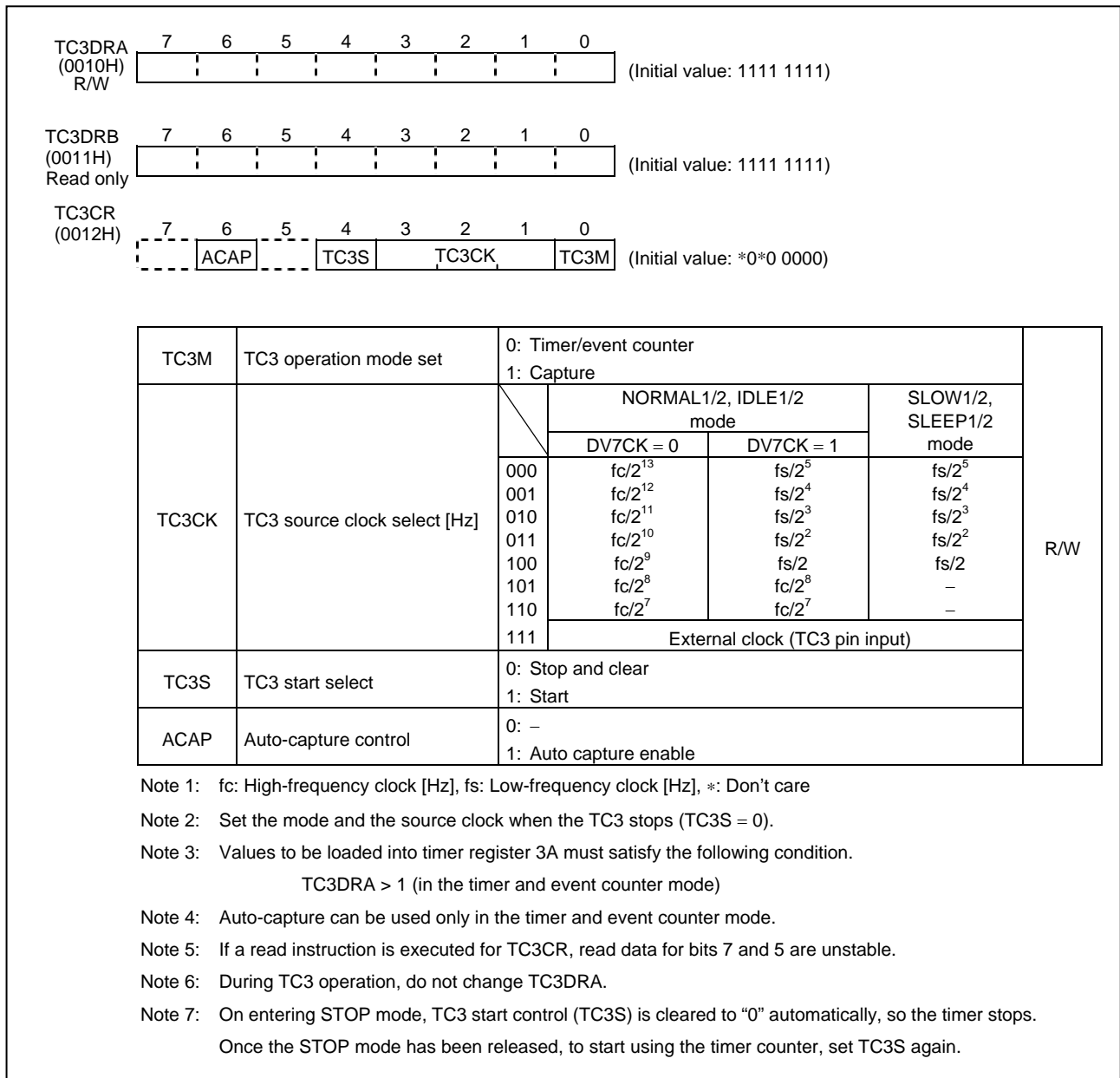
TC3CK	NORMAL1/2, IDLE1/2 Mode				SLOW1/2 Mode	
	DV7CK = 0		DV7CK = 1			
	Resolution [μs]	Maximum Time Setting [ms]	Resolution [μs]	Maximum Time Setting [ms]	Resolution [μs]	Maximum Time Setting [ms]
000	512.0	130.6	976.6	249.0	976.6	249.0
001	256.0	65.3	488.3	124.5	488.3	124.5
010	128.0	32.6	244.1	62.3	244.1	62.3
011	64.0	16.3	122.0	31.1	122.0	31.1
100	32.0	8.2	61.0	15.6	61.0	15.6
101	16.0	4.1	16.0	4.1	–	–
110	8.0	2.0	8.0	2.0	–	–

(2) Event counter mode

In this mode, events are counted on the edge of the TC3 pin input. The counter counts up on the rising edge of the TC3 pin input and when its value matches the TC3DRA set value, it is cleared while at the same time generating an INTTC3 interrupt.

The detection of match is executed at the falling edge of the TC3 pin. Therefore, if the TC3 pin keeps high level after the rising, the detection of match is not executed and INTTC3 is not generated until the level of TC3 pin becomes low.

The minimum input pulse width of the TC3 pin is shown in Table 2.9.2. One or more machine cycles are required for both the "H" and "L" levels of the pulse width.

The current contents of up counter are loaded into TC3DRB by setting TC3CR<ACAP> to "1" (Auto-capture function).

The contents of up counter can be easily confirmed by executing the read instruction (RD instruction) of TC3DRB. Loading the contents of up counter is not synchronized with counting up. The contents of over flow (FFH) and 00H can not be loaded correctly. It is necessary to consider the count cycle.
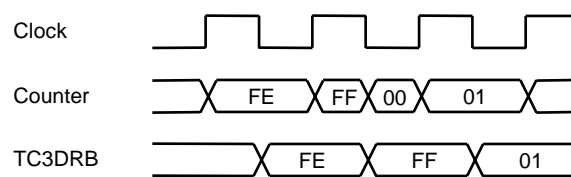
Table 2.9.2 Source Clock (External clock) for Timer/Counter

	Minimum Input Pulse Width [s]	
	NORMAL1/2, IDLE1/2 Mode	SLOW1/2, SLEEP1/2 Mode
"H" width	$2^2/fc$	$2^2/fs$
"L" width	$2^2/fc$	$2^2/fs$

TOSHIBA

(3) Capture mode

In this mode, the pulse width, period and duty of the TC3 pin input are measured in this mode, which can be used in decoding the remote control signals or distinguishing AC 50/60 Hz, etc.

Once command operation has started, the counter free-runs on an internal source clock.

When the falling edge of the TC3 pin input is detected, the counter value is loaded into TC3DRB. When the rising edge is detected, the counter value is loaded into TC3DRA, and the counter is cleared, generating an INTTC3 interrupt.

If the rising edge is detected right after command operation has started, no capture to TC3DRB and an INTTC3 interrupt occurs only on capture to TC3DRA. If a read instruction is executed for TC3DRB, the value that exists at the end of the previous capture (Immediately after a reset, "FF") is read.

The minimum acceptable input pulse width is equal to the length of one source clock period selected by TC3CR<TC3CK>.

Table 2.9.3 Capture Input Edges

Capture into TC3DRB	Capture into TC3DRA	INTTC3 Interrupt
Falling edge	Rising edge	

When the overflow occurs before detecting the edge, the INTTC3 interrupt is generated, setting "FFH" to TC3DRA and clearing the counter. It is possible to confirm whether the overflow has occurred or not by reading TC3DRA in interrupt routine. After generating of interrupt, the capture function and overflow detection stop until the TC3DRA is read, but the counting is continued. Because the capture function and overflow detection are restarted by reading TC3DRA, read the TC3DRB before the reading TC3DRA.

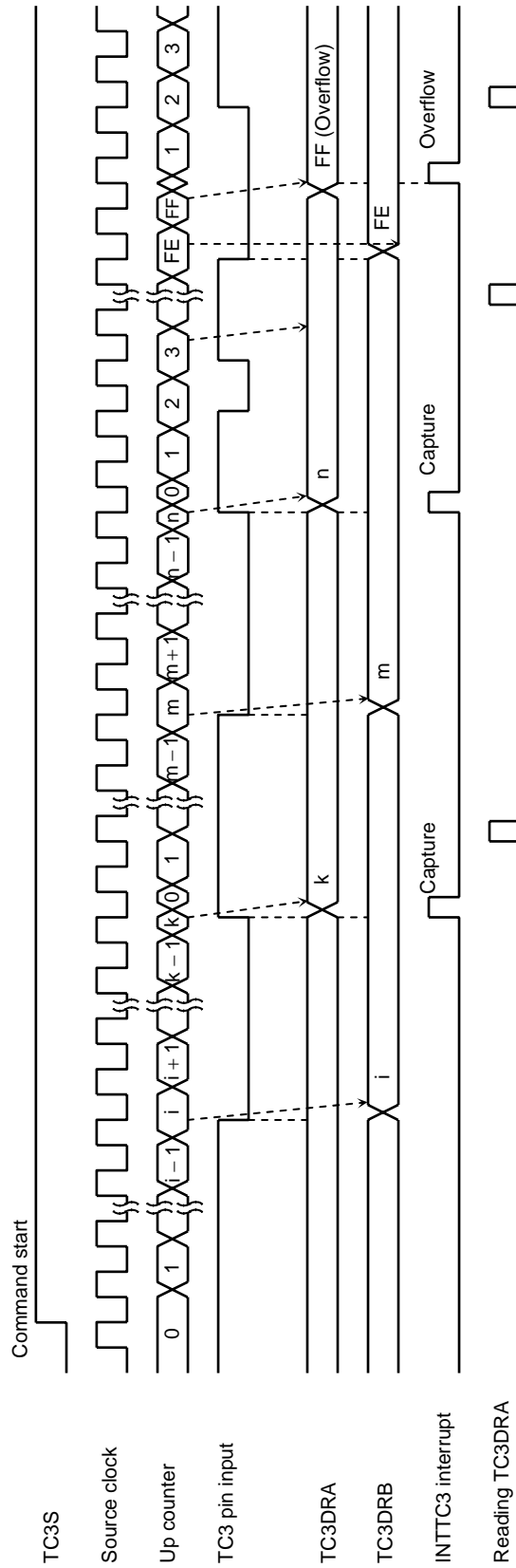Figure 2.9.3 Capture Mode Timing Chart

2.10   8-Bit Timer/Counter 5
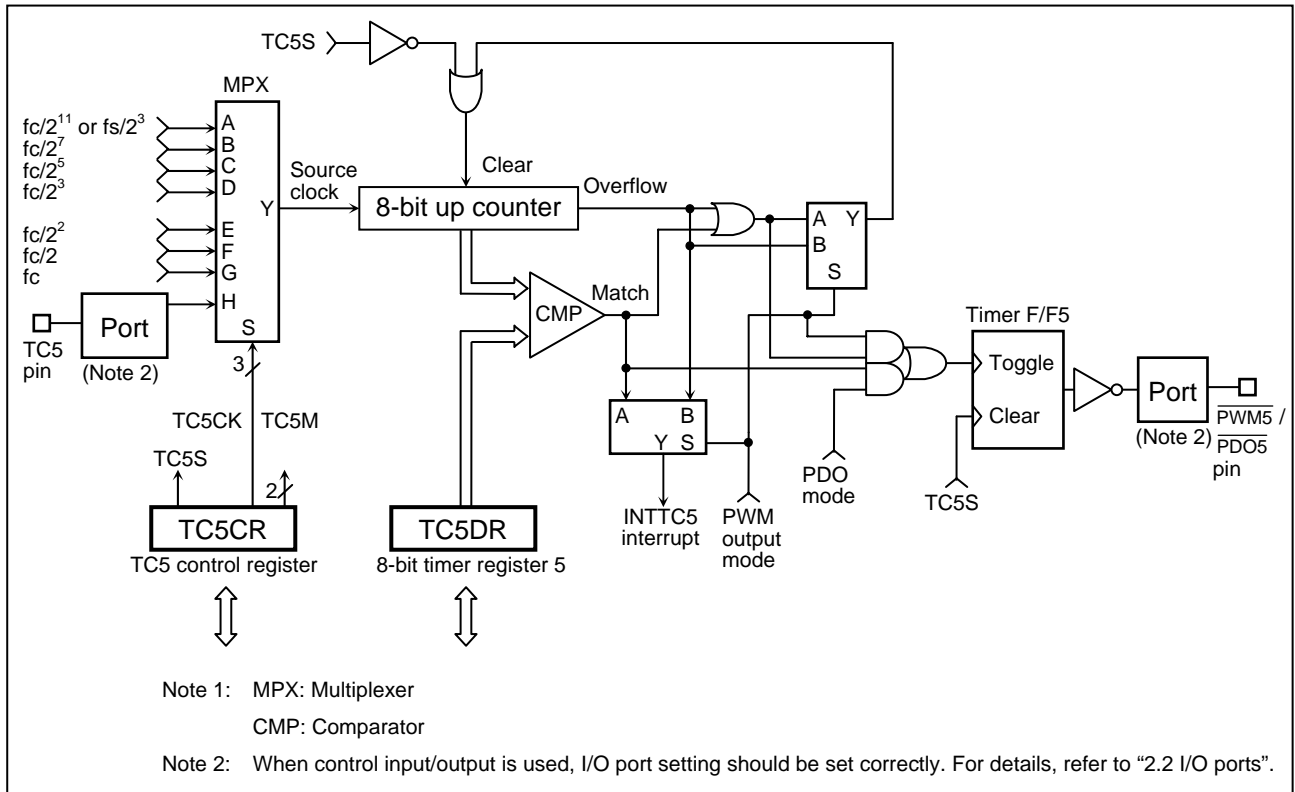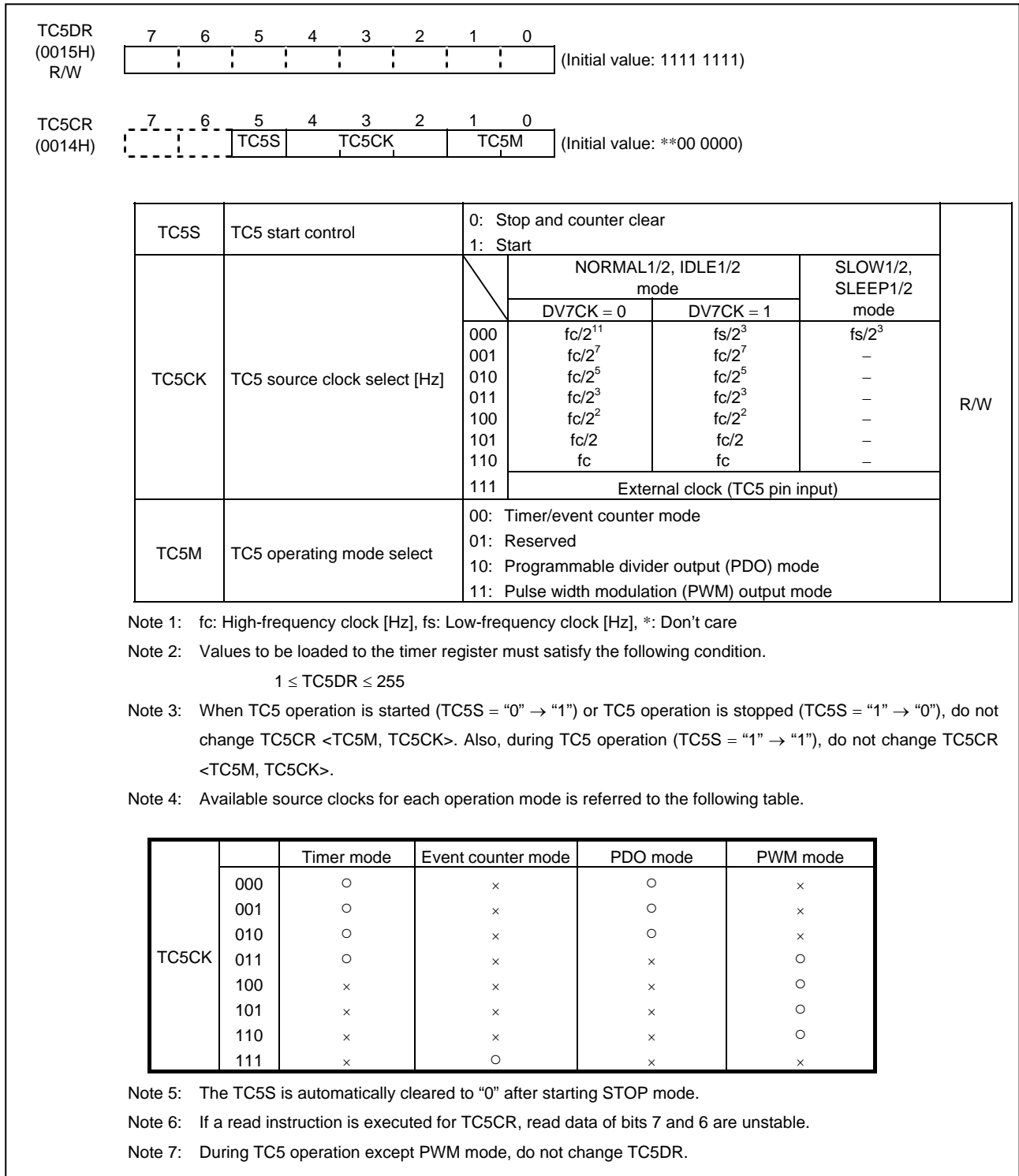
2.10.1   Configuration

Figure 2.10.1 Timer/Counter 5 (TC5)

2.10.2 Control

www.DataSheet4U.c The timer/counter 5 is controlled by a timer/counter 5 control register (TC5CR) and an 8-bit timer register 5 (TC5DR). Reset does not affect TC5DR.

TC5DR (0015H) R/W	7	6	5	4	3	2	1	0	(Initial value: 1111 1111)

TC5CR (0014H)	7	6	5	4	3	2	1	0	(Initial value: **00 0000)
			TC5S	TC5CK			TC5M		

TC5S	TC5 start control	0: Stop and counter clear 1: Start				R/W
TC5CK	TC5 source clock select [Hz]		NORMAL1/2, IDLE1/2 mode		SLOW1/2, SLEEP1/2 mode	
			DV7CK = 0	DV7CK = 1		
		000	$fc/2^{11}$	$fs/2^3$	$fs/2^3$	
		001	$fc/2^7$	$fc/2^7$	–	
		010	$fc/2^5$	$fc/2^5$	–	
		011	$fc/2^3$	$fc/2^3$	–	
		100	$fc/2^2$	$fc/2^2$	–	
		101	$fc/2$	$fc/2$	–	
		110	fc	fc	–	
		111	External clock (TC5 pin input)			
TC5M	TC5 operating mode select	00: Timer/event counter mode 01: Reserved 10: Programmable divider output (PDO) mode 11: Pulse width modulation (PWM) output mode				

Note 1: fc: High-frequency clock [Hz], fs: Low-frequency clock [Hz], *: Don't care

Note 2: Values to be loaded to the timer register must satisfy the following condition.

$$1 \leq TC5DR \leq 255$$

Note 3: When TC5 operation is started (TC5S = "0" → "1") or TC5 operation is stopped (TC5S = "1" → "0"), do not change TC5CR <TC5M, TC5CK>. Also, during TC5 operation (TC5S = "1" → "1"), do not change TC5CR <TC5M, TC5CK>.

Note 4: Available source clocks for each operation mode is referred to the following table.

TC5CK		Timer mode	Event counter mode	PDO mode	PWM mode
	000	○	×	○	×
	001	○	×	○	×
	010	○	×	○	×
	011	○	×	×	○
	100	×	×	×	○
	101	×	×	×	○
	110	×	×	×	○
	111	×	○	×	×

Note 5: The TC5S is automatically cleared to "0" after starting STOP mode.

Note 6: If a read instruction is executed for TC5CR, read data of bits 7 and 6 are unstable.

Note 7: During TC5 operation except PWM mode, do not change TC5DR.

Figure 2.10.2 Timer Register 5 and TC5 Control Register

2.10.3   Function

The timer/counter 5 has four operating modes: timer, event counter, programmable divider output, and PWM output mode.

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of TC5DR is compared with the contents of up counter. If a match is found, an INTTC5 interrupt is generated and the up-counter is cleared to "0". Counting up resumes after the up-counter is cleared.

Table 2.10.1   Source Clock (Internal clock) for Timer/Counter 5 (Example: at fc = 16 MHz)

TC5CK	NORMAL1/2, IDLE1/2 Mode				SLOW1/2 Mode	
	DV7CK = 0		DV7CK = 1			
	Resolution [μs]	Maximum Time Setting [ms]	Resolution [μs]	Maximum Time Setting [ms]	Resolution [μs]	Maximum Time Setting [ms]
000	128.0	32.6	244.14	62.3	244.14	62.3
001	8.0	2.0	8.0	2.0	–	–
010	2.0	0.510	2.0	0.510	–	–
011	0.5	0.128	0.5	0.128	–	–

(2) Event counter mode

In this mode, events are counted on the rising edge of the TC5 pin input (External clock).

The contents of the TC5DR is compared with the contents of the up counter. If a match is found, an INTTC5 interrupt is generated and the counter is cleared. Counting up resumes after the up counter is cleared. The minimum input pulse width of the TC5 pin is shown in Table 2.10.2. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width.

Match detect is executed on the falling edge of the TC5 pin. A match can not be detected and INTTC5 interrupt is not generated when the pulse is still in a falling state.

Table 2.10.2   Timer/Counter 5 External Clock Source

	Minimum Input Pulse Width [s]	
	NORMAL1/2, IDLE1/2 Mode	SLOW1/2, SLEEP1/2 Mode
"H" width	$2^3/fc$	$2^3/fs$
"L" width	$2^3/fc$	$2^3/fs$

(3) Programmable divider output (PDO) mode

The programmable divider output (PDO) mode is intended to output a pulse having a duty cycle of about 50%. The counter counts up on an internal source clock. If the timer value matches TC5DR, the timer F/F5 is inverted, and the counter is cleared, generating an INTTC5 interrupt. The counter keeps counting up, and the timer F/F5 is inverted each time the timer value matches TC5DR. The P13 ($\overline{\text{PDO5}}$) pin outputs an inversion of the timer F/F5 output level.

At a reset or when the timer stops, the timer F/F5 is cleared to "0". So, stopping the timer when the PDO output is low may cause the duty cycle to become smaller than the set value.

To use the programmable divider output mode, set the output latch of the P13 port to "1".

Example: Output a 1024 Hz pulse (at fc = 16 MHz)

```
            LD      (TC5CR), 00000110B    ;   Sets PDO mode
                                              (TC5M = 10, TC5CK = 001)
            SET     (P1DR). 3             ;   P13 output latch ← 1
            LD      (TC5DR), 3DH          ;   1/1024 ÷ 2^7/fc ÷ 2 = 3DH
            LD      (TC5CR), 00100110B    ;   Starts TC5
```
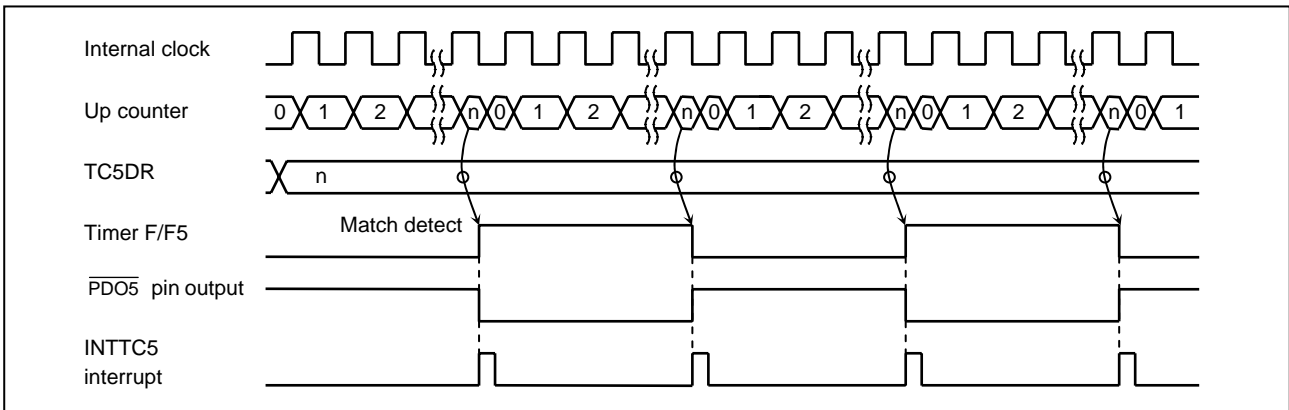

Figure 2.10.3 PDO Mode Timing Chart

(4) Pulse width modulation (PWM) output mode

The pulse width modulation (PWM) output mode is intended to output pulses at constant intervals with a resolution of 8 bits. The counter counts up on the internal source clock. If the timer value matches TC5DR, the timer F/F5 is inverted, and the counter keeps-up counting. If an overflow is detected, the timer F/F5 is inverted again, generating an INTTC5 interrupt. The P13 ($\overline{PWM5}$) pin outputs an inversion of the timer F/F5 output level.

At a reset or when the timer stops, the timer F/F5 is cleared to "0". So, stopping the timer when the PWM output is low may cause one cycle to become smaller than the set value.

To use the pulse width modulation (PWM) output mode, set the output latch of the P13 port to "1".

TC5DR is configured a 2-stage shift register and, during pulse width, will not switch until one output cycle is completed even if TC5DR is overwritten; therefore, pulse width can be altered continuously. Also, the first time, TC5DR is shifted by setting TC5CR<TC5S> to "1" after data are loaded to TC5DR.

Note: In PWM mode, writing to the timer register TC5DR should be performed only right after an INTTC5 interrupt occurs (Usually, within the INTTC5 interrupt service routine). If writing to the timer register TC5DR occurs at the same timing as the INTTC5 interrupt, pulses having a value other than the set value may be output before another INTTC5 interrupt occurs, because an unstable value that is being written is shifted.
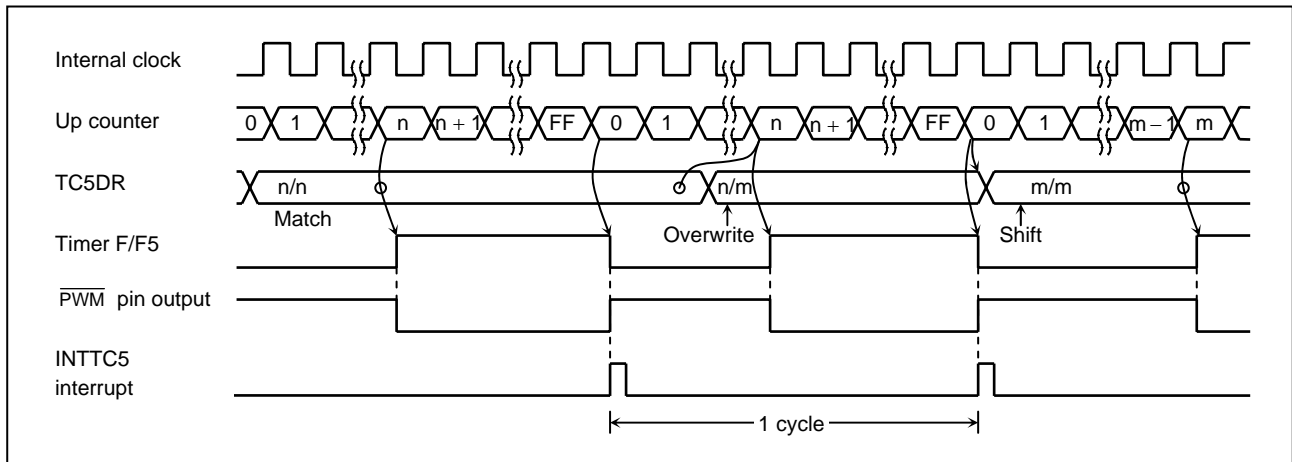

Figure 2.10.4 PWM Output Mode Timing Chart

Table 2.10.3 PWM Output Mode (Example: fc = 16 MHz)

TC5CK	NORMAL1/2, IDLE1/2 Mode	
	Resolution [ns]	Repeat Cycle [μs]
000	–	–
001	–	–
010	–	–
011	500	128
100	250	64
101	125	32
110	62.5	16

2.11 UART (Asynchronous serial interface)

The TMP86FM48 has 1 channel of UART (Asynchronous serial interface).

The UART is connected to external devices via RXD and TXD. RXD is also used as P05; TXD, as P06. To use P05 or P06 as the RXD or TXD pin, set P0 port output latches to "1".
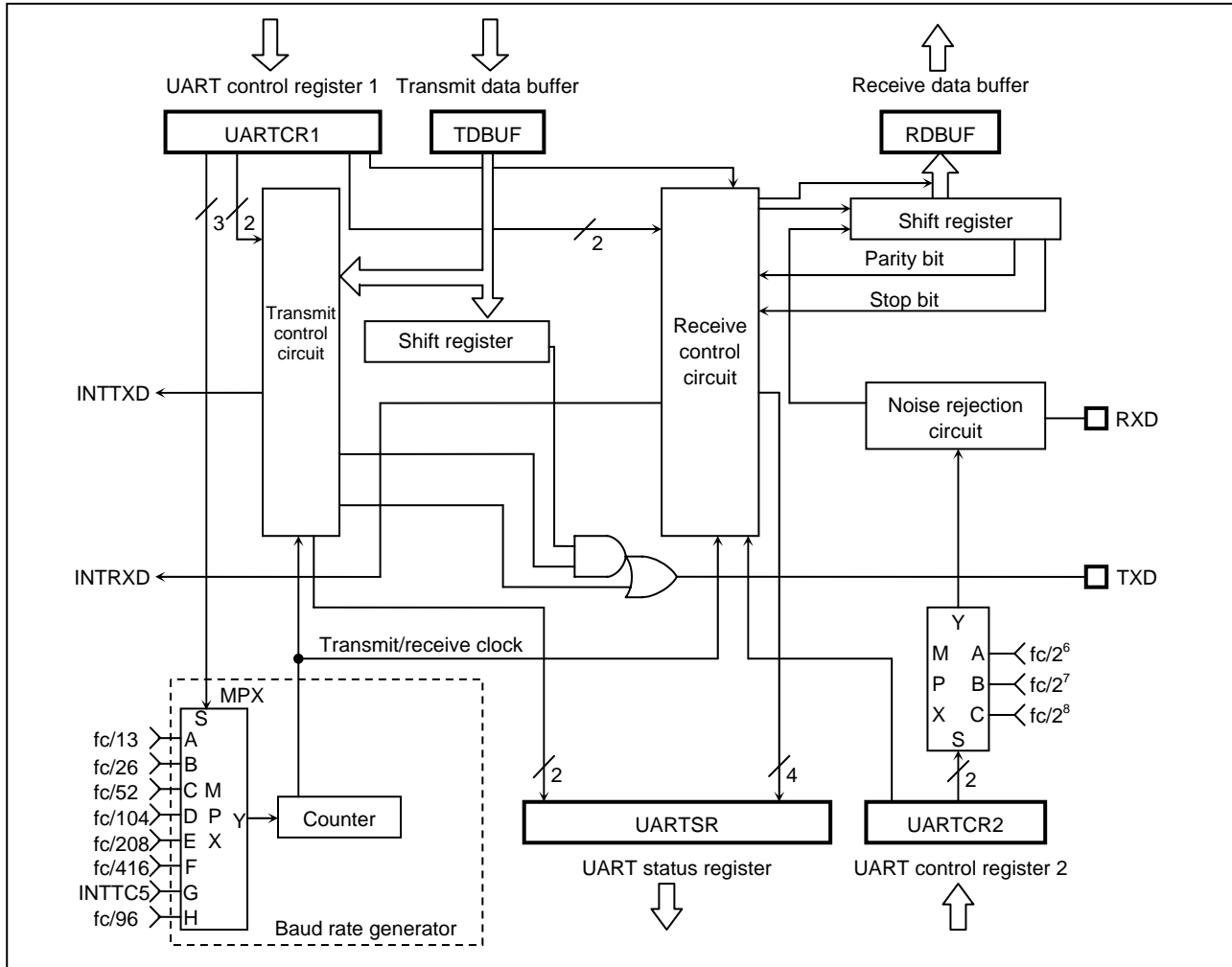
2.11.1 Configuration

Figure 2.11.1 UART

2.11.2 Control

UART is controlled by the UART control registers (UARTCR1, UARTCR2). The operating status can be monitored using the UART status register (UARTSR).
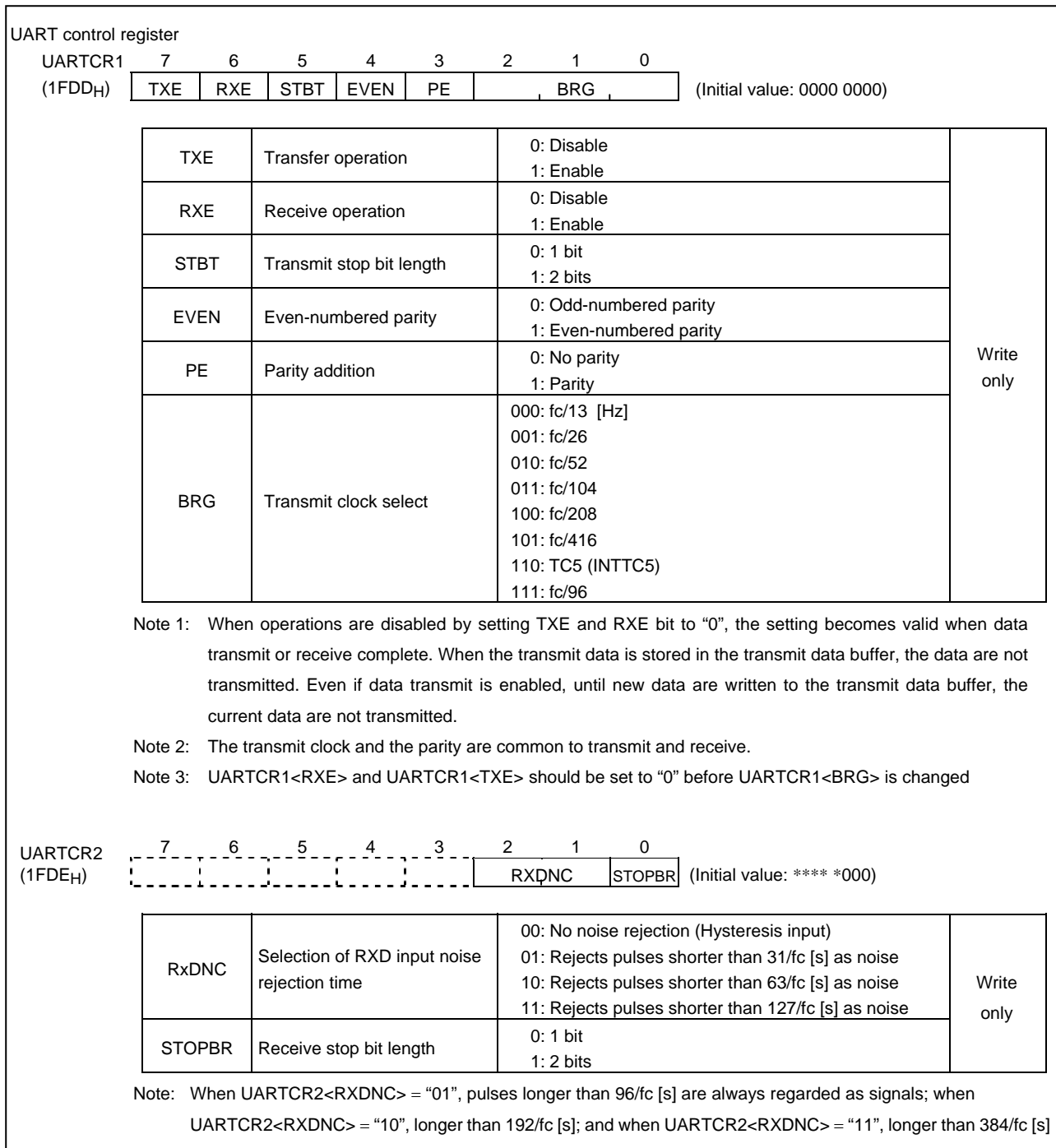
UART control register

UARTCR1
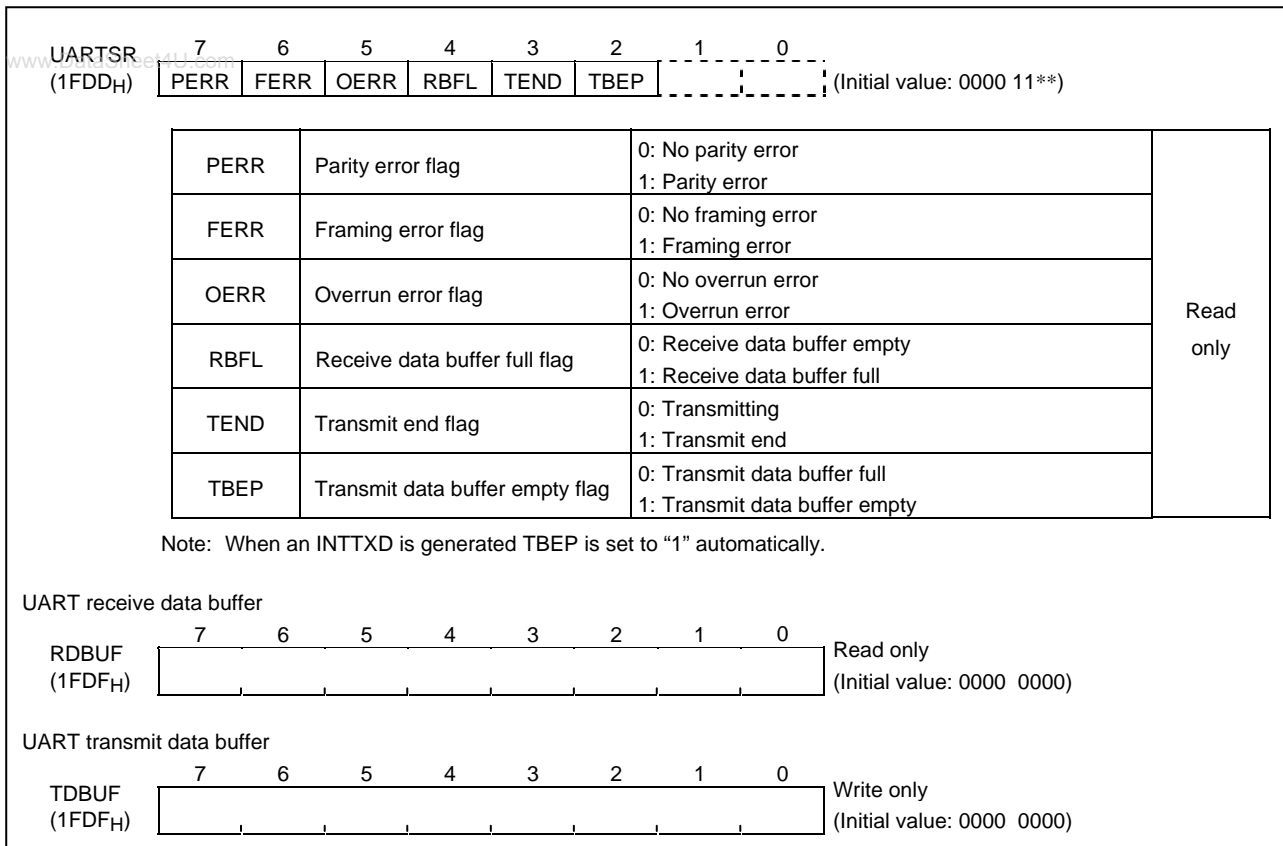(1FDD$_H$)

7	6	5	4	3	2	1	0
TXE	RXE	STBT	EVEN	PE		BRG	

(Initial value: 0000 0000)

TXE	Transfer operation	0: Disable 1: Enable	
RXE	Receive operation	0: Disable 1: Enable	
STBT	Transmit stop bit length	0: 1 bit 1: 2 bits	
EVEN	Even-numbered parity	0: Odd-numbered parity 1: Even-numbered parity	
PE	Parity addition	0: No parity 1: Parity	Write only
BRG	Transmit clock select	000: fc/13 [Hz] 001: fc/26 010: fc/52 011: fc/104 100: fc/208 101: fc/416 110: TC5 (INTTC5) 111: fc/96	

Note 1: When operations are disabled by setting TXE and RXE bit to "0", the setting becomes valid when data transmit or receive complete. When the transmit data is stored in the transmit data buffer, the data are not transmitted. Even if data transmit is enabled, until new data are written to the transmit data buffer, the current data are not transmitted.

Note 2: The transmit clock and the parity are common to transmit and receive.

Note 3: UARTCR1<RXE> and UARTCR1<TXE> should be set to "0" before UARTCR1<BRG> is changed

UARTCR2
(1FDE$_H$)

7	6	5	4	3	2	1	0
					RXDNC		STOPBR

(Initial value: **** *000)

RxDNC	Selection of RXD input noise rejection time	00: No noise rejection (Hysteresis input) 01: Rejects pulses shorter than 31/fc [s] as noise 10: Rejects pulses shorter than 63/fc [s] as noise 11: Rejects pulses shorter than 127/fc [s] as noise	Write only
STOPBR	Receive stop bit length	0: 1 bit 1: 2 bits	

Note: When UARTCR2<RXDNC> = "01", pulses longer than 96/fc [s] are always regarded as signals; when UARTCR2<RXDNC> = "10", longer than 192/fc [s]; and when UARTCR2<RXDNC> = "11", longer than 384/fc [s]

Figure 2.11.2 UART Control Register

UARTSR | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
(1FDD_H) | PERR | FERR | OERR | RBFL | TEND | TBEP | | | (Initial value: 0000 11**)

PERR	Parity error flag	0: No parity error 1: Parity error	
FERR	Framing error flag	0: No framing error 1: Framing error	
OERR	Overrun error flag	0: No overrun error 1: Overrun error	Read only
RBFL	Receive data buffer full flag	0: Receive data buffer empty 1: Receive data buffer full	
TEND	Transmit end flag	0: Transmitting 1: Transmit end	
TBEP	Transmit data buffer empty flag	0: Transmit data buffer full 1: Transmit data buffer empty	

Note: When an INTTXD is generated TBEP is set to "1" automatically.

UART receive data buffer

RDBUF | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Read only
(1FDF_H) | | | | | | | | | (Initial value: 0000 0000)

UART transmit data buffer

TDBUF | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Write only
(1FDF_H) | | | | | | | | | (Initial value: 0000 0000)

Figure 2.11.3 UART Status Register and Data Buffer Registers

2.11.3 Transfer Data Format

In UART, a one-bit start bit (Low level), stop bit (Bit length selectable at high level, by UARTCR1<STBT>), and parity (Select parity in UARTCR1<PE>; even- or odd-numbered parity by UARTCR1<EVEN>) are added to the transfer data. The transfer data formats are shown as follow.

Table 2.11.1 Transfer Data Format

PE	STBT	Frame Length								
		1	2	3	------ 8	9	10	11	12	
0	0	Start	Bit0	Bit1	------ Bit6	Bit7	Stop 1			
0	1	Start	Bit0	Bit1	------ Bit6	Bit7	Stop 1	Stop 2		
1	0	Start	Bit0	Bit1	------ Bit6	Bit7	Parity	Stop 1		
1	1	Start	Bit0	Bit1	------ Bit6	Bit7	Parity	Stop 1	Stop 2	

Note: In order to switch the transmit data format, perform transmit operations in the following sequence except for the initial setting.
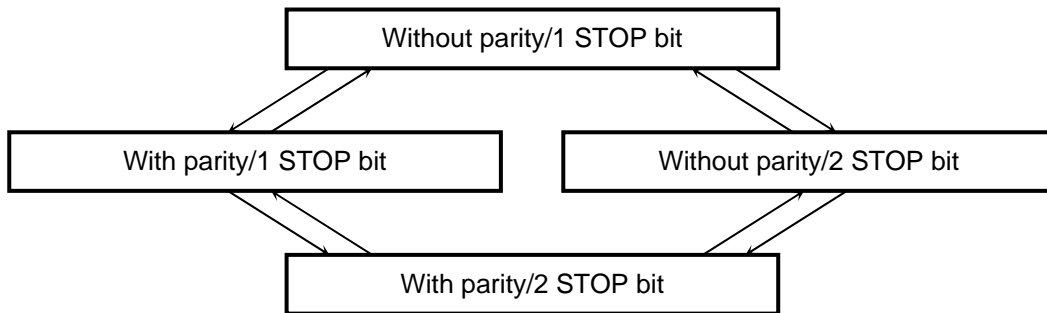
2.11.4 Transfer Rate

The baud rate of UART is set of UARTCR1<BRG>. The example of the baud rate shown as follows.

Table 2.11.2 Transfer Rate

BRG	Source Clock		
	16 MHz	8 MHz	4 MHz
000	76800 [baud]	38400 [baud]	19200 [baud]
001	38400	19200	9600
010	19200	9600	4800
011	9600	4800	2400
100	4800	2400	1200
101	2400	1200	600

When TC5 is used as the UART transfer rate (when UARTCR1<BRG> = "110"), the transfer clock and transfer rate are determined as follows:

$$\text{Transfer clock} = \frac{\text{TC5 source clock}}{\text{TTREG5 set value}}$$

$$\text{Transfer rate} = \frac{\text{Transfer clock}}{16}$$

2.11.5 Data Sampling

The UART receiver keeps sampling input using the clock selected by UARTCR1<BRG> until a start bit is detected in RXD pin input. RT clock starts detecting "L" level of the RXD pin. Once a start bit is detected, the start bit, data bits, stop bit (s), and parity bit are sampled at three times of RT7, RT8, and RT9 during one receiver clock interval (RT clock). (RT0 is the position where the bit supposedly starts). Bit is determined according to majority rule (The data are the same twice or more out of three samplings).
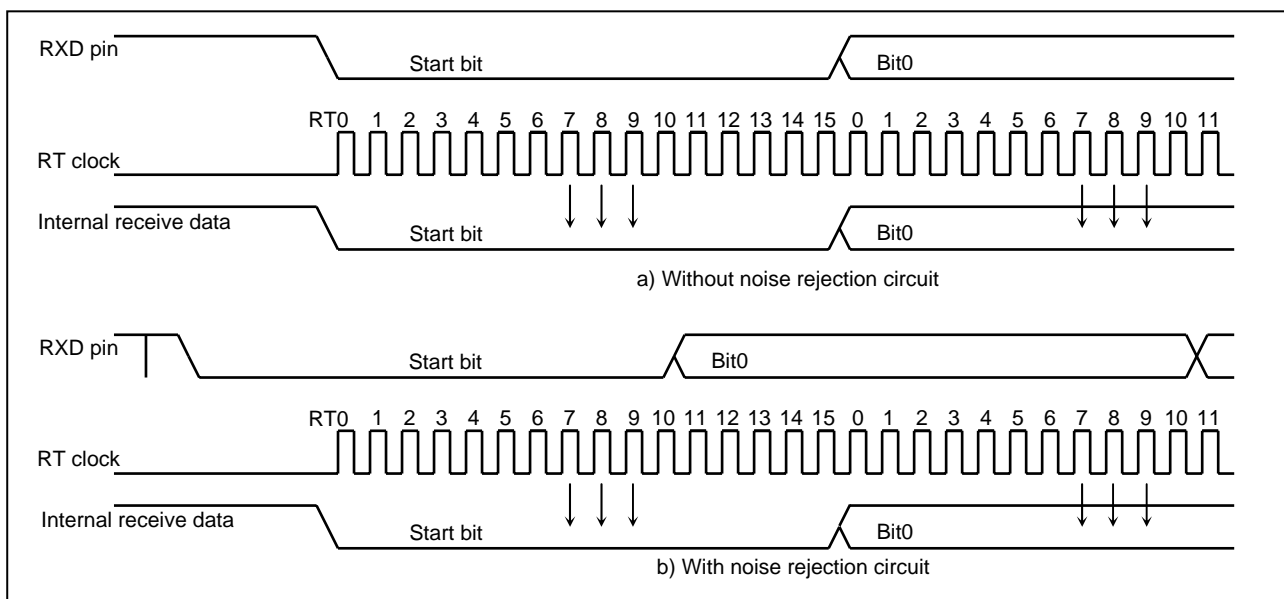
Figure 2.11.4 Data Sampling

2.11.6 STOP Bit Length

Select a transmit stop bit length (1 or 2 bits) by UARTCR1<STBT>.

2.11.7 Parity

Set parity/no parity by UARTCR1<PE>; set parity type (odd- or even-numbered) by UARTCR1<EVEN>.

2.11.8 Transmit/Receive

(1) Data transmit

Set UARTCR1<TXE> to "1". Read UARTSR to check UARTSR<TBEP> = "1", then write data in TDBUF (Transmit data buffer). Writing data in TDBUF zero-clears UARTSR<TBEP>, transfers the data to the transmit shift register and the data are sequentially output from the TXD pin. The data output include a one-bit start bit, stop bits whose number is specified in UARTCR1<STBT> and a parity bit if parity addition is specified. Select the data transfer baud rate using bits 0 to 2 in UARTCR1. When data transmit starts, transmit buffer empty flag UARTSR<TBEP> is set to "1" and an INTTXD interrupt is generated.

While UARTCR1<TXE> = "0" and from when "1" is written to UARTCR1<TXE> to when send data are written to TDBUF, the TXD pin is fixed at high level. When transmitting data, first read UARTSR, then write data in TDBUF. Otherwise, UARTSR<TBEP> is not zero-cleared and transmit does not start.

(2) Data receive

Set UARTCR1<RXE> to "1". When data are received via the RXD pin, the receive data are transferred to RDBUF (Receive data buffer). At this time, the data transmitted include a start bit and stop bit (s) and a parity bit if parity addition is specified. When stop bit (s) are received, data only are extracted and transferred to RDBUF (Receive data buffer). Then the receive buffer full flag UARTSR<RBFL> is set and an INTRXD interrupt is generated. Select the data transfer baud rate using bits 0 to 2 in UARTCR1.

If an overrun error (OERR) occurs when data are received, the data are not transferred to RDBUF (Receive data buffer) but discarded; data in the RDBUF are not affected.

Note: When a receive operation is disabled by setting UARTCR1<RXE> bit to "0", the setting becomes valid when data receive is completed. However, if a framing error occurs in data receive, the receive-disabling setting may not become valid. if a framing error occurs, be sure to perform a re-receive operation.

2.11.9 Status Flag/Interrupt Signal

(1) Parity error

When parity determined using the receive data bits differs from the received parity bit, the parity error flag UARTSR<PERR> is set to "1". The UARTSR<PERR> is cleared to "0" when the RDBUF is read after reading the UARTSR.
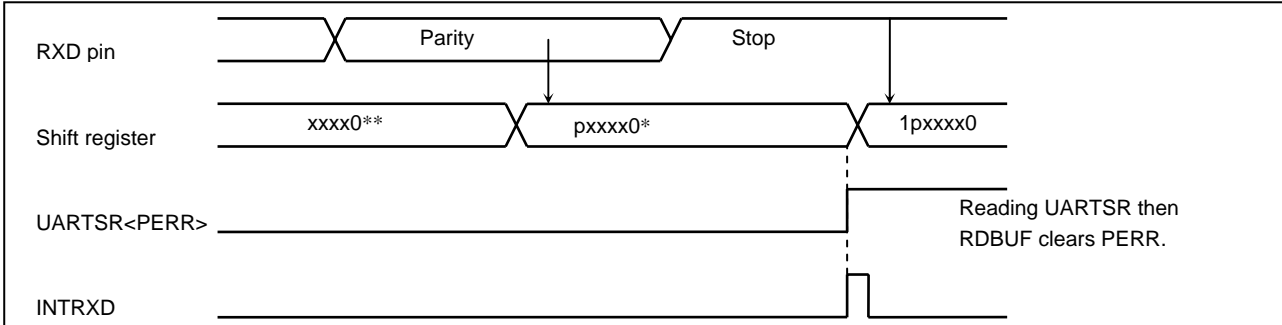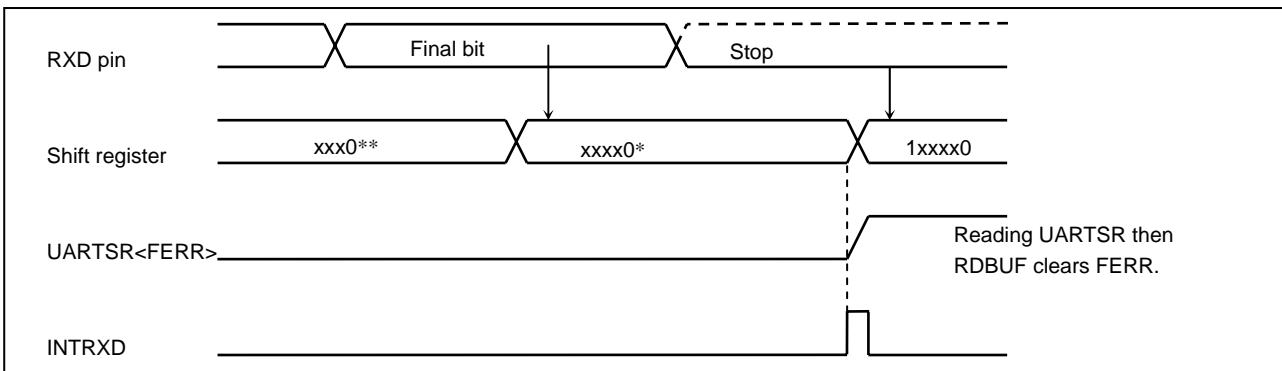

Figure 2.11.5 Generation of Parity Error

(2) Framing error

When "0" is sampled as the stop bit in the receive data, framing error flag UARTSR<FERR> is set to "1". The UARTSR<FERR> is cleared to "0" when the RDBUF is read after reading the UARTSR.

Figure 2.11.6 Generation of Framing Error

(3) Overrun error

When all bits in the next data are received while unread data are still in RDBUF, overrun error flag UARTSR<OERR> is set to "1". In this case, the receive data is discarded; data in RDBUF are not affected. The UARTSR<OERR> is cleared to "0" when the RDBUF is read after reading the UARTSR.
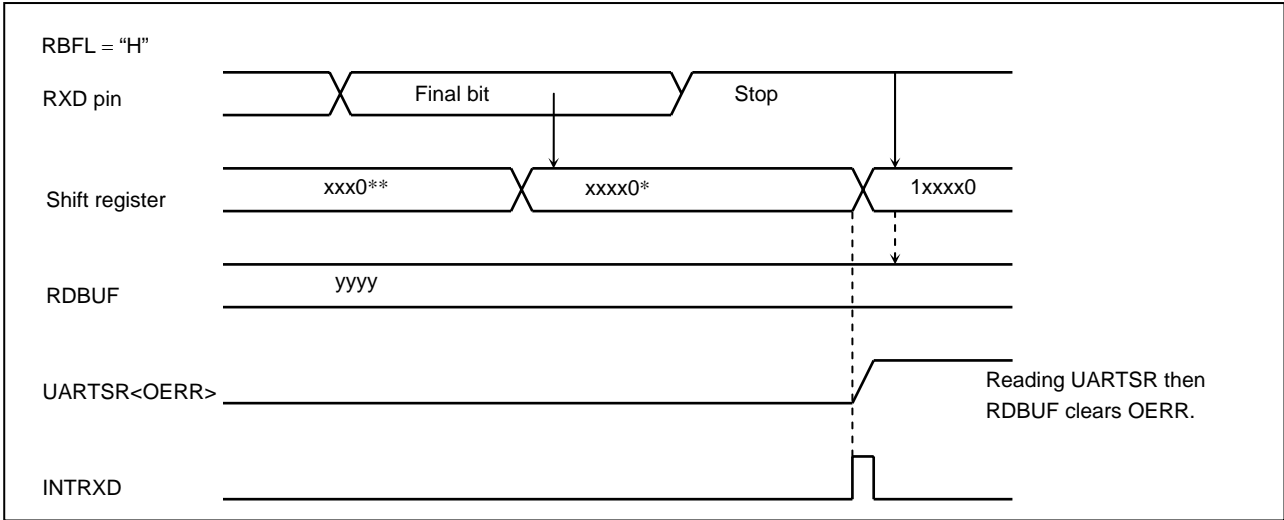

Figure 2.11.7 Generation of Overrun Error

(4) Receive data buffer full

Loading the received data in RDBUF sets receive data buffer full flag UARTSR<RBFL>. The UARTSR<RBFL> is cleared to "0" when the RDBUF is read after reading the UARTSR.
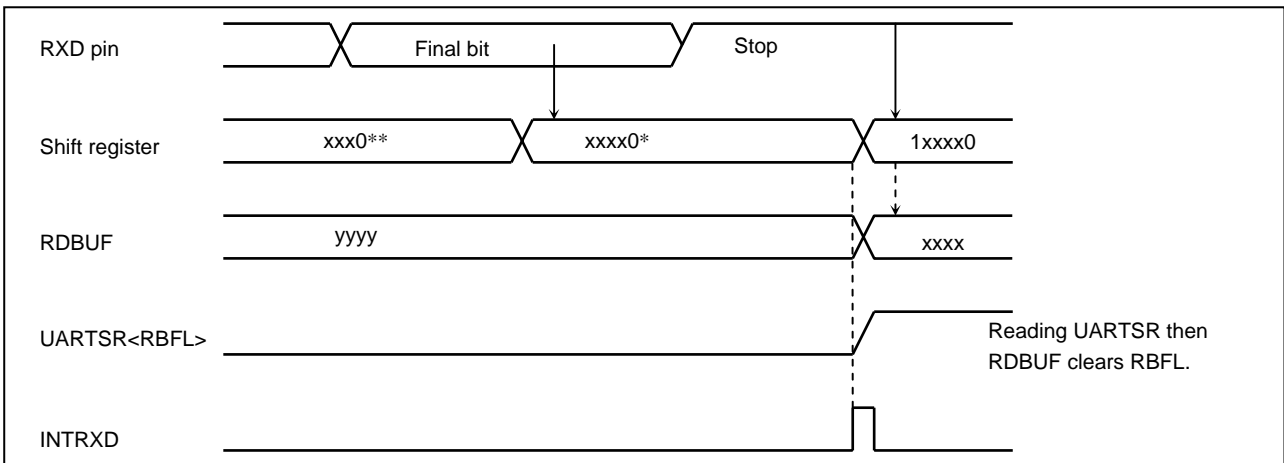
Figure 2.11.8 Generation of Receive Buffer Full

(5) Transmit data buffer empty

When no data is in the transmit buffer TDBUF, UARTSR<TBEP> is set to "1", that is, when data in TDBUF are transferred to the transmit shift register and data transmit starts, transmit data buffer empty flag UARTSR<TBEP> is set to "1". The UARTSR<TBEP> is cleared to "0" when the TDBUF is written after reading the UARTSR.
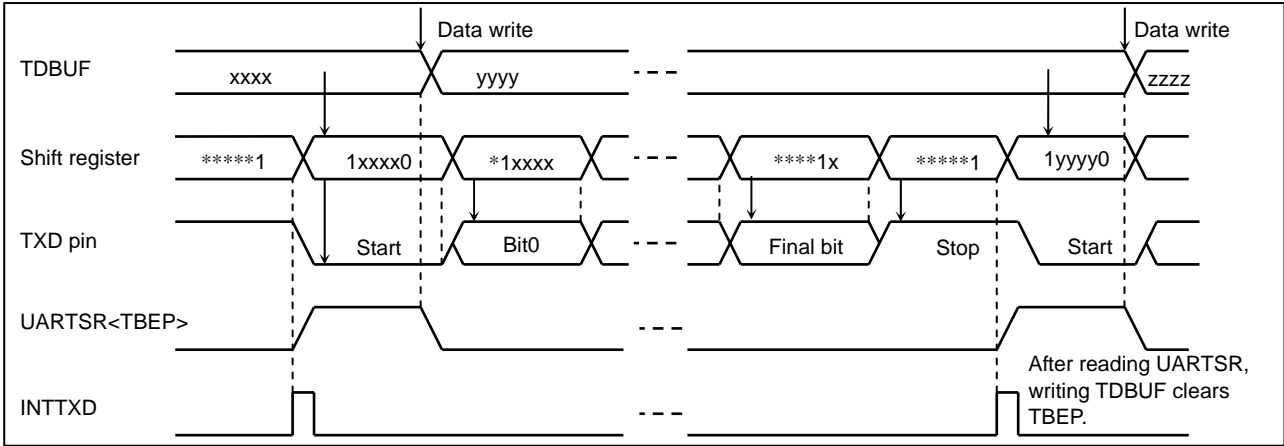

Figure 2.11.9 Generation of Transmit Buffer Empty

(6) Transmit end flag

When data are transmitted and no data is in TDBUF (UARTSR<TBEP> = "1"), transmit end flag UARTSR<TEND> is set to "1". The UARTSR<TEND> is cleared to "0" the data transmit is stated after writing the TDBUF.
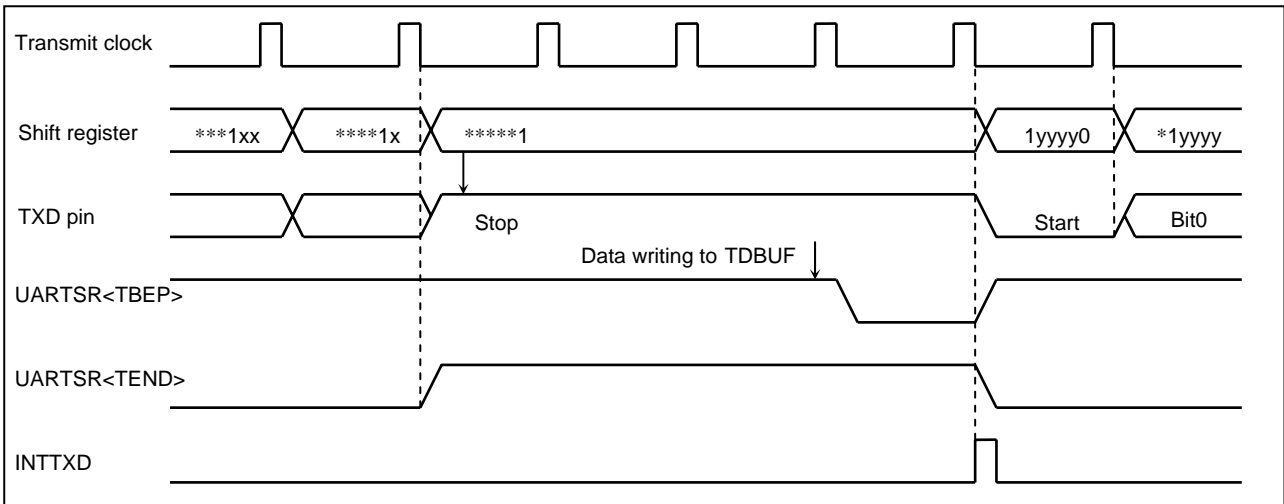

Figure 2.11.10 Generation of Transmit Buffer Empty

2.12 Serial Bus Interface (SBI-ver. D)

The TMP86FM48 has a 1-channel serial bus interface which employs an I²C bus (A bus system by Philips).

The serial interface is connected to external devices through P51 (SDA) and P50 (SCL). The serial bus interface pins are also used for the P5 port. When used for serial bus interface pins, set the P5 output latches of these pins to "1". When not used as serial bus interface pins, the P5 port is used as a normal I/O port.

Note 1: When P5 is used as serial bus interface pins, P50 and P51 should be set as a sink open drain output by clearing P5OUTCR to "0".

Note 2: The serial bus interface can be used only in NORMAL1/2 and IDLE1/2 mode. It can not be used in IDLE0, SLOW1/2 and SLEEP0/1/2 mode.

Note 3: The I²C of TMP86FM48 can be used only in the Standard mode of I²C. The fast mode and the high-speed mode can not be used.
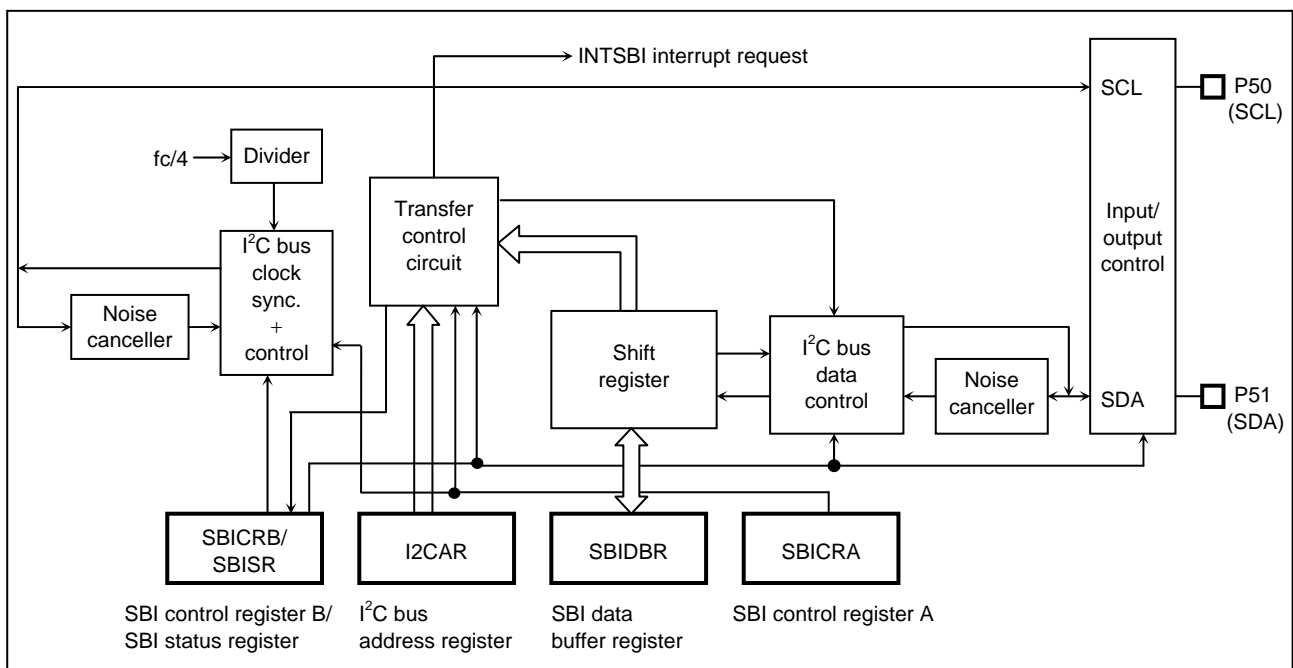
2.12.1 Configuration

Figure 2.12.1 Serial Bus Interface (SBI)

2.12.2 Control

The following registers are used for control the serial bus interface and monitor the operation status.

- Serial bus interface control register A (SBICRA)
- Serial bus interface control register B (SBICRB)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register (SBISR)

2.12.3 Software Reset

A serial bus interface circuit has a software reset function, when a serial bus interface circuit is locked by an external noise, etc.

To reset the serial bus interface circuit, write "01", "10" into the SWRST (Bit1, 0 in SBICRB).

2.12.4 The Data Format of the I²C Bus

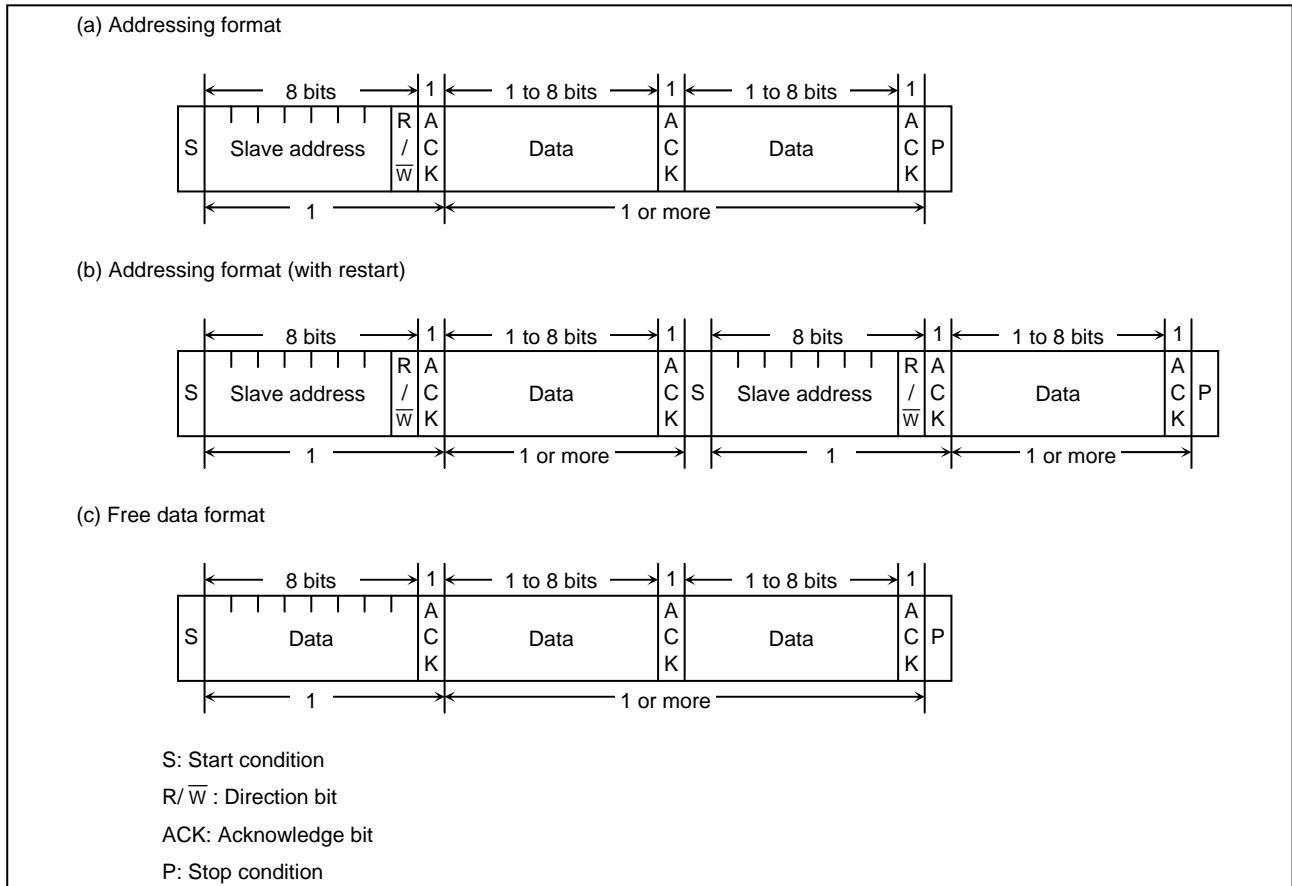The data format of the I²C bus is shown in as below.

Figure 2.12.2 Data Format of I²C Bus

2.12.5 I²C Bus Control

The following registers are used to control the serial bus interface (SBI) and monitor the operation status of the I²C bus.
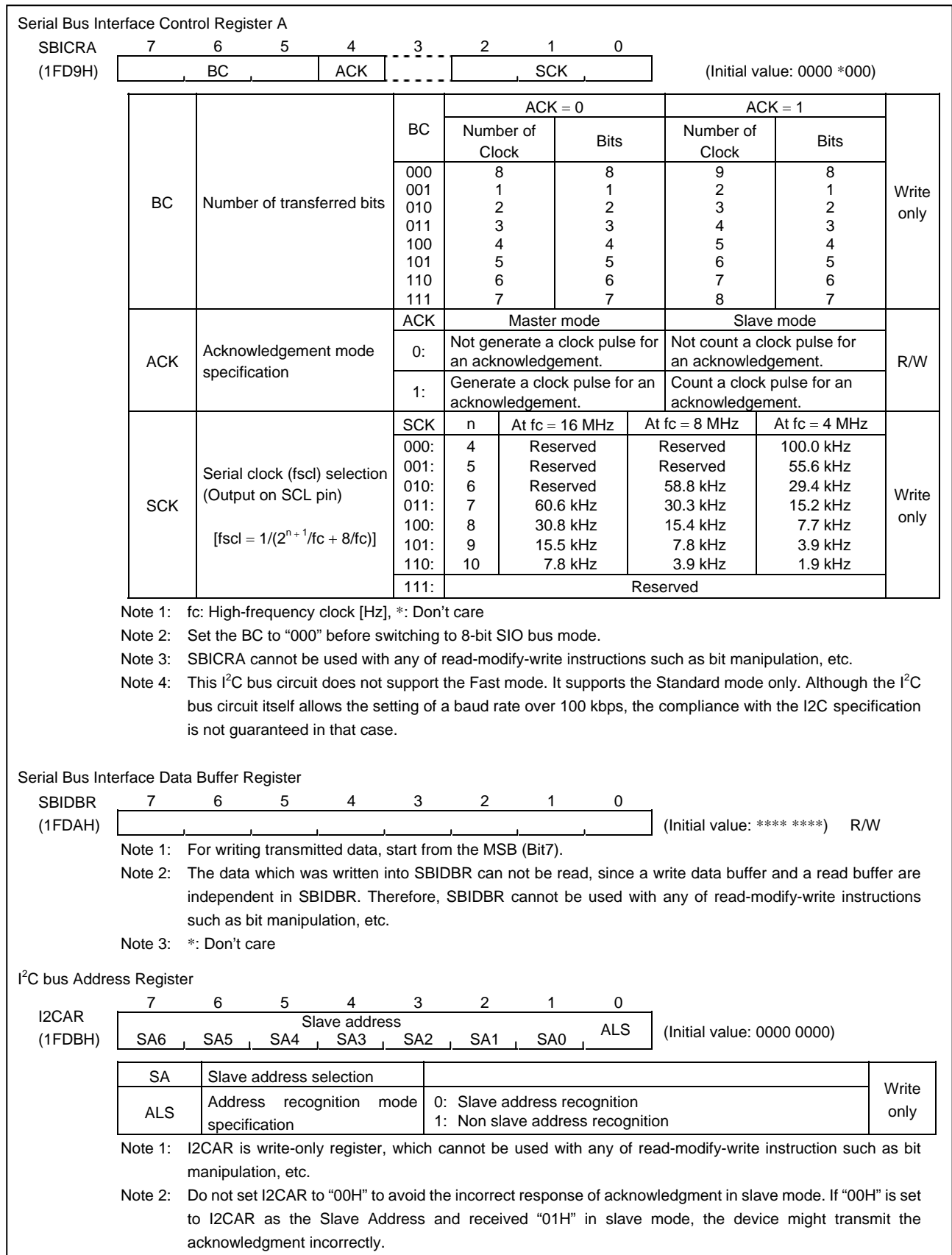

Figure 2.12.3 Serial Bus Interface Control Register A, Serial Bus Interface Data Buffer Register and I²C Bus Address Register

86FM48-105 2007-08-24

Serial Bus Interface Control Register B

SBICRB	7	6	5	4	3	2	1	0	
(1FDCH)	MST	TRX	BB	PIN	SBIM		SWRST1	SWRST0	(Initial value: 0001 0000)

MST	Master/slave selection	0: Slave 1: Master		Write only
TRX	Transmitter/receiver selection	0: Receiver 1: Transmitter		
BB	Start/stop generation	0: Generate a stop condition when MST, TRX and PIN are "1" 1: Generate a start condition when MST, TRX and PIN are "1"		
PIN	Cancel interrupt service request	0: − 1: Cancel interrupt service request		
SBIM	Serial bus interface operating mode selection	00: Port mode (Serial bus interface output disable) 01: Reserved 10: I^2C bus mode 11: Reserved		
SWRST1 SWRST0	Software reset start bit	Software reset starts by first writing "10" and next writing "01"		

Note 1: Switch a mode to port after confirming that the bus is free.

Note 2: Switch a mode to I^2C bus mode after confiming that the port is high level.

Note 3: SBICRB has write-only register and must not be used with any of read-modify-write instructions such as bit manipulation, etc.

Note 4: When the SWRST (Bit1, 0 in SBICRB) is written to "01", "10" in I^2C bus mode, software reset is occurred. In this case, the SBICRA, I2CAR and SBISR registers are initialized and the bits of SBICRB except the SBIM (Bit3, 2 in SBICRB) are also initialized.

Serial Bus Interface Status Register

SBISR	7	6	5	4	3	2	1	0	
(1FDCH)	MST	TRX	BB	PIN	AL	AAS	AD0	LRB	(Initial value: 0001 0000)

MST	Master/slave selection status monitor	0: Slave 1: Master	Read only
TRX	Transmitter/receiver selection status monitor	0: Receiver 1: Transmitter	
BB	Bus status monitor	0: Bus free 1: Bus busy	
PIN	Interrupt service requests status monitor	0: Requesting interrupt service 1: Releasing interrupt service request	
AL	Arbitration lost detection monitor	0: − 1: Arbitration lost detected	
AAS	Slave address match detection monitor	0: Not detect slave address match or "GENERAL CALL" 1: Detect slave address match or "GENERAL CALL"	
AD0	"GENERAL CALL" detection monitor	0: Not detect "GENERAL CALL" 1: Detect "GENERAL CALL"	
LRB	Last received bit monitor	0: Last receive bit is "0" 1: Last receiv bit is "1"	

Figure 2.12.4 Serial Bus Interface Control Register B and Serial Bus Interface Status Register

(1) Acknowledgement mode specification

   a. Acknowledgment mode (ACK = "1")

   To set the device as an acknowledgment mode, the ACK (Bit4 in SBICRA) should be set to "1". When a serial bus interface circuit is a master mode, an additional clock pulse is generated for an acknowledge signal. In a slave mode, a clock is counted for the acknowledge signal.

   In the master transmitter mode, the SDA pin is released in order to receive an acknowledge signal from the receiver during additional clock pulse cycle. In the master receiver mode, the SDA pin is set to low level generation an acknowledge signal during additional clock pulse cycle.

   In a slave mode, when a received slave address matches to a slave address which is set to the I2CAR or when a "GENERAL CALL" is received, the SDA pin is set to low level generating an acknowledge signal. After the matching of slave address or the detection of "GENERAL CALL", in the transmitter, the SDA pin is released in order to receive an acknowledge signal from the receiver during additional clock pulse cycle. In a receiver, the SDA pin is set to low level generation an acknowledge signal during additional clock pulse cycle after the matching of slave address or the detection of "GENERAL CALL"

   The Table 2.12.1 shows the SCL and SDA pins status in acknowledgment mode.

Table 2.12.1 SCL and SDA Pins Status in Acknowledgement Mode

Mode	Pin			Transmitter	Receiver
Master	SCL			An additional clock pulse is generated.	
	SDA			Released in order to receive an acknowledge signal.	Set to low level generating an acknowledge signal
Slave	SCL			A clock is counted for the acknowledge signal.	
	SDA	When slave address matches or a general call is detected		–	Set to low level generating an acknowledge signal.
		After matching of slave address or general call		Released in order to receive an acknowledge signal.	Set to low level generating an acknowledge signal.

   b. Non-acknowledgment mode (ACK = "0")

   To set the device as a non-acknowledgement mode, the ACK should be cleared to "0".

   In the master mode, a clock pulse for an acknowledge signal is not generated.

   In the slave mode, a clock for a acknowledge signal is not counted.

(2) Number of transfer bits

   The BC (Bits7 to 5 in SBICRA) is used to select a number of bits for next transmitting and receiving data.

   Since the BC is cleared to "000" as a start condition, a slave address and direction bit transmissions are always executed in 8 bits. Other than these, the BC retains a specified value.

(3) Serial clock

   a. Clock source

   The SCK (Bits2 to 0 in SBICRA) is used to select a maximum transfer frequency output from the SCL pin in the master mode. Set a communication baud rate that meets the I2C bus specification, such as the shortest pulse width of t_{LOW}, based on the equations shown below.

Four or more machine cycles are required for both high and low levels of pulse width in the external clock which is input from SCL pin.

Note: Since the I^{2}C of TMP86FM48 can not be used as the fast mode and the high-speed mode, do not set SCK as the frequency that is over 100 kHz.

$t_{LOW} = 2^n/fc$

$t_{HIGH} = 2^n/fc + 8/fc$

$fscl = 1/(t_{Low} + t_{HIGH})$

fc: High-frequency clock

SCK (Bits2 to 0 in the SBICRA)	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10

t_{SCKL}, $t_{SCKH} > 4$ tcyc

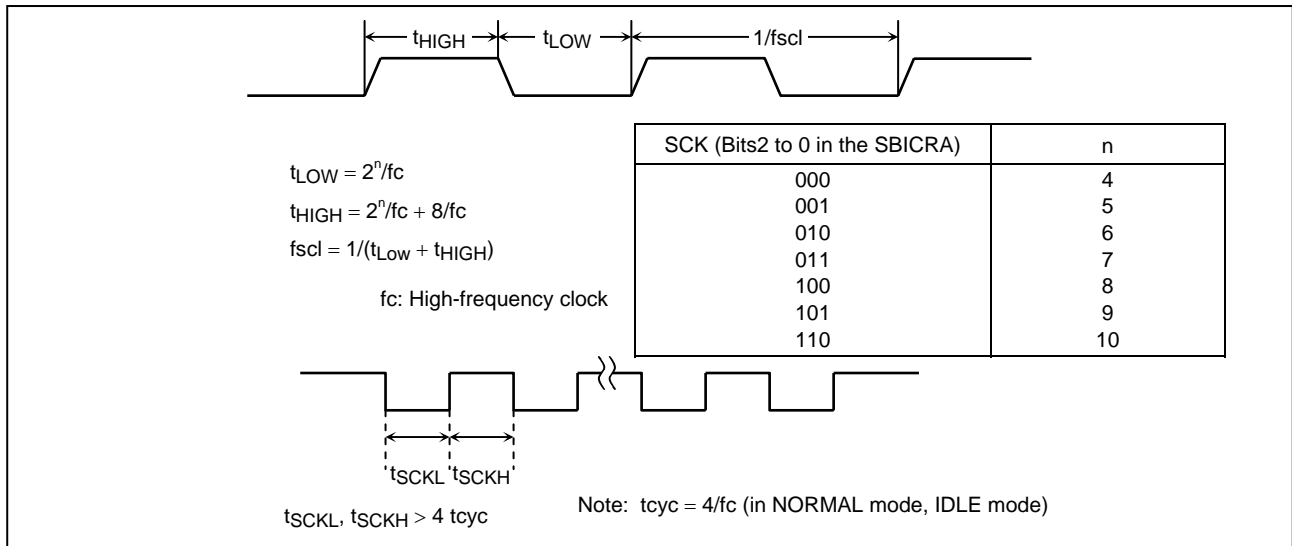Note: tcyc = 4/fc (in NORMAL mode, IDLE mode)

Figure 2.12.5 Clock Source

b. Clock synchronization

In the I^{2}C bus, in order to drive a bus with a wired AND, a master device which pulls down a clock pulse to low will, in the first place, invalidate a clock pulse of another master device which generates a high-level clock pulse.

The serial bus interface circuit has a clock synchronization function. This function ensures normal transfer even if there are two or more masters on the same bus.

The example explains clock synchronization procedures when two masters simultaneously exist on a bus.
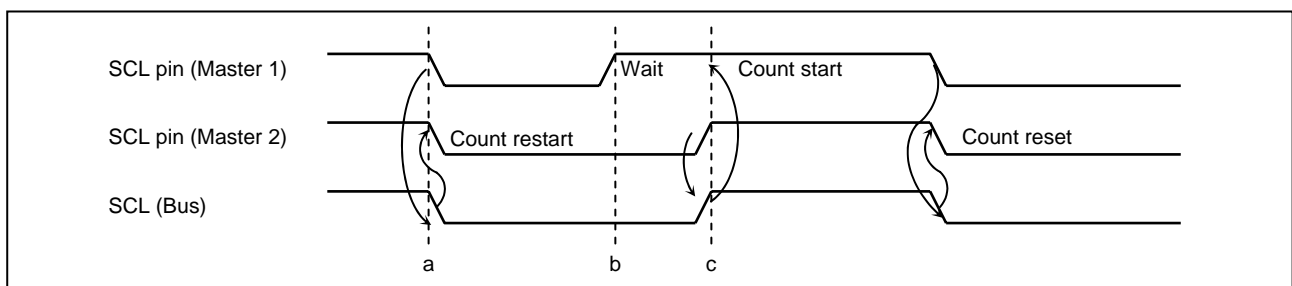
Figure 2.12.6 Clock Synchronization

As Master 1 pulls down the SCL pin to the low level at point "a", the SCL line of the bus becomes the low level. After detecting this situation, Master 2 resets counting a clock pulse in the high level and sets the SCL pin to the low level.

Master 1 finishes counting a clock pulse in the low level at point "b" and sets the SCL pin to the high level. Since Master 2 holds the SCL line of the bus at the low level, Master 1 waits for counting a clock pulse in the high level. After Master 2 sets a clock pulse to the high level at point "c" and detects the SCL line of the bus at the high level, Master 1 starts counting a clock pulse in the high level. Then, the master, which has finished the counting a clock pulse in the high level, pulls

down the SCL pin to the low level.

   The clock pulse on the bus is determined by the master device with the shortest high-level period and the master device with the longest low-level period from among those master devices connected to the bus.

(4) Slave address and address recognition mode specification

   When the serial bus interface circuit is used with an addressing format to recognize the slave address, clear the ALS (Bit0 in I2CAR) to "0", and set the SA (Bits7 to 1 in I2CAR) to the slave address.

   When the serial bus interface circuit is used with a free data format not to recognize the slave address, set the ALS to "1". With a free data format, the slave address and the direction bit are not recognized, and they are processed as data from immediately after start condition.

(5) Master/slave selection

   To set a master device, the MST (Bit7 in SBICRB) should be set to "1". To set a slave device, the MST should be cleared to "0".

   When a stop condition on the bus or an arbitration lost is detected, the MST is cleared to "0" by the hardware.

(6) Transmitter/receiver selection

   To set the device as a transmitter, the TRX (Bit6 in SBICRB) should be set to "1". To set the device as a receiver, the TRX should be cleared to "0". When data with an addressing format is transferred in the slave mode, the TRX is set to "1" by a hardware if the direction bit ($R/\overline{W}$) sent from the master device is "1", and is cleared to "0" by a hardware if the bit is "0". In the master mode, after an acknowledge signal is returned from the slave device, the TRX is cleared to "0" by a hardware if a transmitted direction bit is "1", and is set to "1" by a hardware if it is "0". When an acknowledge signal is not returned, the current condition is maintained.

   When a stop condition on the bus or an arbitration lost is detected, the TRX is cleared to "0" by the hardware. Table 2.12.2 shows TRX changing conditions in each mode and TRX value after changing.

Table 2.12.2 TRX changing conditions in each mode

Mode	Direction Bit	Conditions	TRX after Changing
Slave mode	"0"	A received slave address is the same value set to I2CAR	"0"
	"1"		"1"
Master mode	"0"	ACK signal is returned	"1"
	"1"		"0"

   When a serial bus interface circuit operates in the free data format, a slave address and a direction bit are not recognized. They are handled as data just after generating a start condition. The TRX is not changed by a hardware.

(7) Start/stop condition generation

www.DataSheet4U.com When the BB (Bit5 in SBISR) is "0", a slave address and a direction bit which are set to the SBIDBR are output on a bus after generating a start condition by writing "1" to the MST, TRX, BB and PIN. It is necessary to set transmitted data to the SBIDBR and set ACK to "1" beforehand.
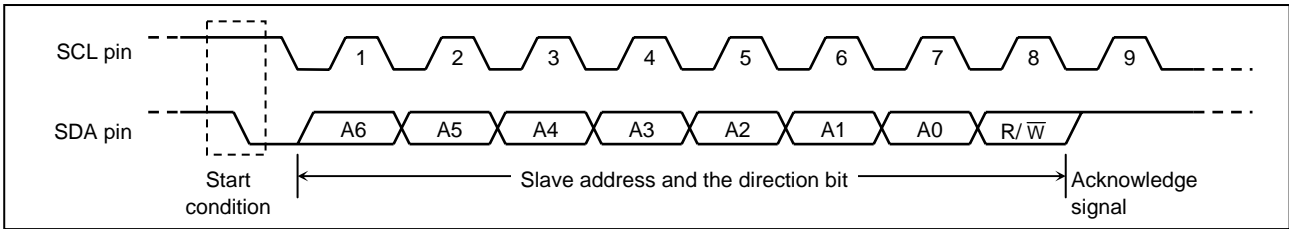

Figure 2.12.7 Start Condition Generation and Slave Address Generation

When the BB is "1", sequence of generating a stop condition is started by writing "1" to the MST, TRX and PIN, and "0" to the BB. Do not modify the contents of MST, TRX, BB and PIN until a stop condition is generated on a bus.
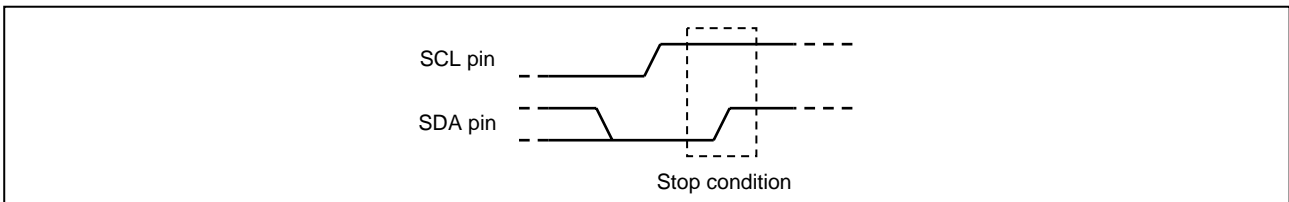

Figure 2.12.8 Stop Condition Generation

When a stop condition is generated and the SCL line on a bus is pulled-down to low level by another device, a stop condition is generated after releasing the SCL line.

The bus condition can be indicated by reading the contents of the BB (Bit5 in SBISR). The BB is set to "1" when a start condition on a bus is detected and is cleared to "0" when a stop condition is detected.

(8) Interrupt service request and cancel

When a serial bus interface circuit is in the master mode and transferring a number of clocks set by the BC and the ACK is complete, a serial bus interface interrupt request (INTSBI) is generated.

In the slave mode, the conditions of generating INTSBI are follows:

- At the end of acknowledge signal when the received slave address matches to the value set by the I2CAR

- At the end of acknowledge signal when a "GENERAL CALL" is received

- At the end of transferring or receiving after matching of slave address or receiving of "GENERAL CALL"

When a serial bus interface interrupt request occurs, the PIN (Bit4 in SBISR) is cleared to "0". During the time that the PIN is "0", the SCL pin is pulled-down to low level.

Either writing data to SBIDBR or reading data 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 (Bit4 in SBICRB) can be set to "1" by the program, the PIN can not be cleared to "0" by the program.

Note: If the arbitration lost occurs, when the slave address does not match, the PIN is not cleared to "0" even though INTSBI is generated.

(9) Setting of I²C bus mode

The SBIM (Bit3 and 2 in SBICRB) is used to set I²C bus mode.

Set the SBIM to "10" in order to set I²C bus mode. Before setting of I²C bus mode, confirm serial bus interface pins in a high level, and then, write "10" to SBIM. And switch a port mode after confirming that a bus is free.

(10) Arbitration lost detection monitor

Since more than one master device can exist simultaneously on a bus, a bus arbitration procedure is implemented in order to guarantee the contents of transferred data.

Data on the SDA line is used for bus arbitration of the I²C bus.

The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on a bus. Master 1 and Master 2 output the same data until point "a". After that, when Master 1 outputs "1" and Master 2 outputs "0", since the SDA line of a bus is wired AND, the SDA line is pulled-down to the low level by Master 2. When the SCL line of a bus is pulled-up at point "b", the slave device reads data on the SDA line, that is data in Master 2. Data transmitted from Master 1 becomes invalid. The state in Master 1 is called "arbitration lost". A master device which loses arbitration releases the SDA pin and the SCL pin in order not to effect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.
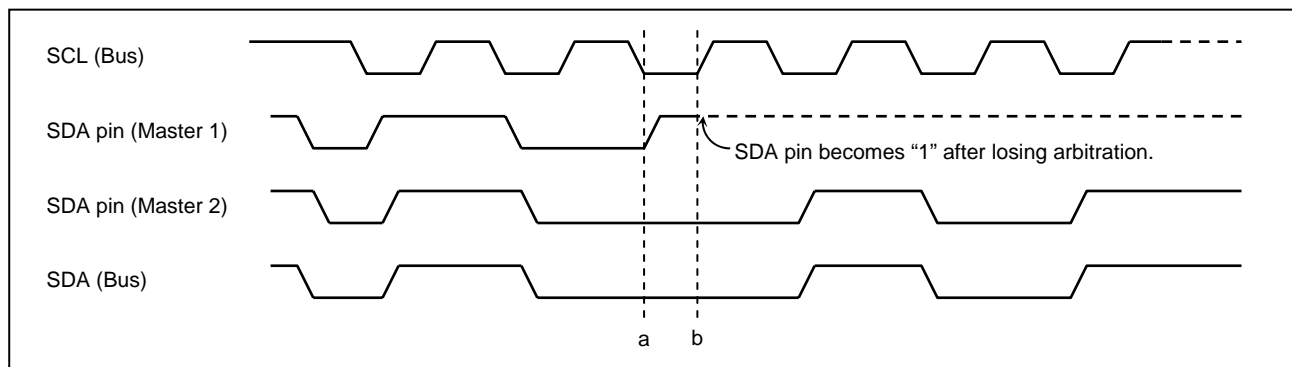
Figure 2.12.9 Arbitration Lost

The serial bus interface circuit compares levels of a SDA line of a bus with its SDA pin at the rising edge of the SCL line. If the levels are unmatched, arbitration is lost and the AL (Bit3 in SBISR) is set to "1".

When the AL is set to "1", the MST and TRX are cleared to "0" and the mode is switched to a slave receiver mode. Thus, the serial bus interface circuit stops output of clock pulses during data transfer after the AL is set to "1".

The AL is cleared to "0" by writing data to the SBIDBR, reading data from the SBIDBR or writing data to the SBICRB.
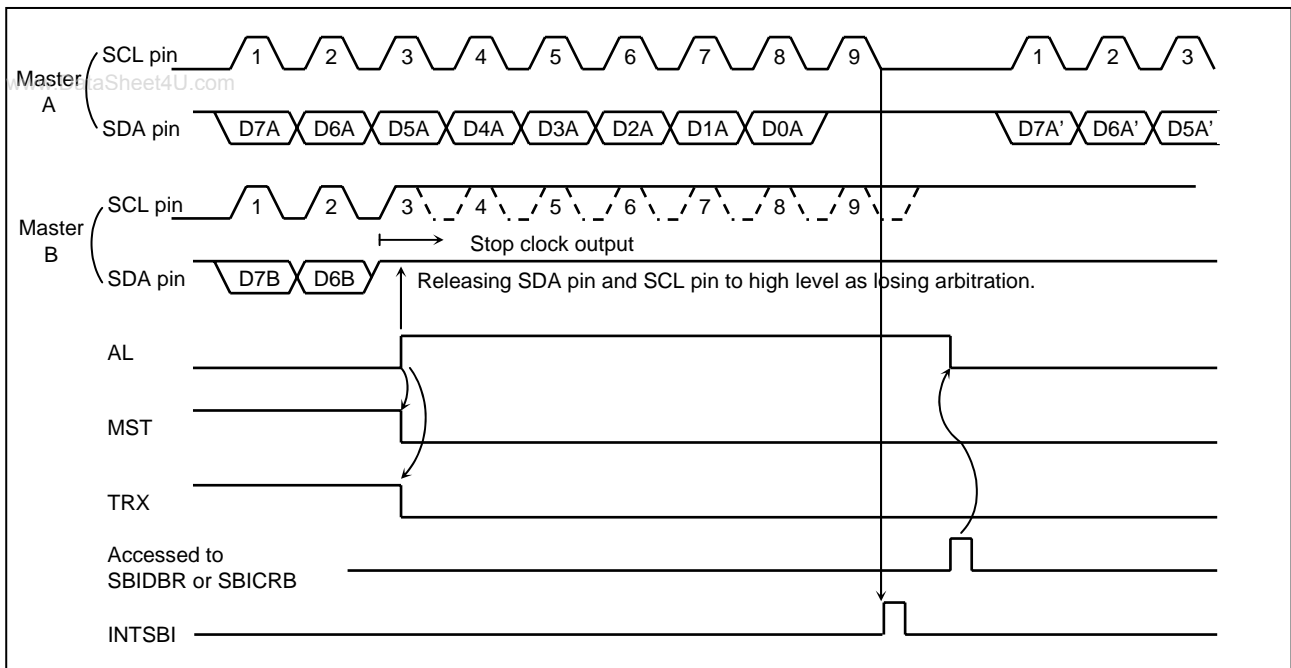
Figure 2.12.10 Example of when a Serial Bus Interface Circuit is a Master B

(11) Slave address match detection monitor

In the slave mode, the AAS (Bit2 in SBISR) is set to "1" when the received data is "GENERAL CALL" or the received data matches the slave address setting by I2CAR with an address recognition mode (ALS = 0).

When a serial bus interface circuit operates in the free data format (ALS = 1), the AAS is set to "1" after receiving the first 1-word of data.

The AAS is cleared to "0" by writing data to the SBIDBR or reading data from the SBIDBR.

(12) GENERAL CALL detection monitor

The AD0 (Bit1 in SBISR) is set to "1" when all 8-bit received data is "0" immediately after a start condition in a slave mode. The AD0 is cleared to "0" when a start or stop condition is detected on a bus.

(13) Last received bit monitor

The SDA value stored at the rising edge of the SCL is set to the LRB (Bit0 in SBISR). In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the LRB.

2.12.6 Data Transfer of I²C Bus

(1) Device initialization

For initialization of device, set the ACK in SBICRA to "1" and the BC to "000". Specify the data length to 8 bits to count clocks for an acknowledge signal. Set a transfer frequency to the SCK in SBICRA.

Next, set the slave address to the SA in I2CAR and clear the ALS to "0" to set an addressing format.

After confirming that the serial bus interface pin is high level, for specifying the default setting to a slave receiver mode, clear "0" to the MST, TRX and BB in SBICRB, set "1" to the PIN, "10" to the SBIM, and "00" to bits SWRST1 and SWRST0.

Note: The initialization of a serial bus interface circuit must be complete within the time from all devices which are connected to a bus have initialized to and device does not generate a start condition. If not, the data can not be received correctly because the other device starts transferring before an end of the initialization of a serial bus interface circuit.

(2) Start condition and slave address generation

Confirm a bus free status (BB = 0).

Set the ACK to "1" and specify a slave address and a direction bit to be transmitted to the SBIDBR.

By writing "1" to the MST, TRX, BB and PIN, the start condition is generated on a bus and then, the slave address and the direction bit which are set to the SBIDBR are output. An INTSBI interrupt request occurs at the 9th falling edge of a 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 is transferred. If data is written to the SBIDBR, data to been outputting may be destroyed.

Note 2: The bus free must be confirmed by software within 98.0 µs (The shortest transmitting time according to the I²C bus standard) after setting of the slave address to be output. Only when the bus free is confirmed, set "1" to the MST, TRX, BB, and PIN to generate the start conditions. If the writing of slave address and setting of MST, TRX, BB and PIN doesn't finish within 98.0 µs, the other masters may start the transferring and the slave address data written in SBIDBR may be broken.
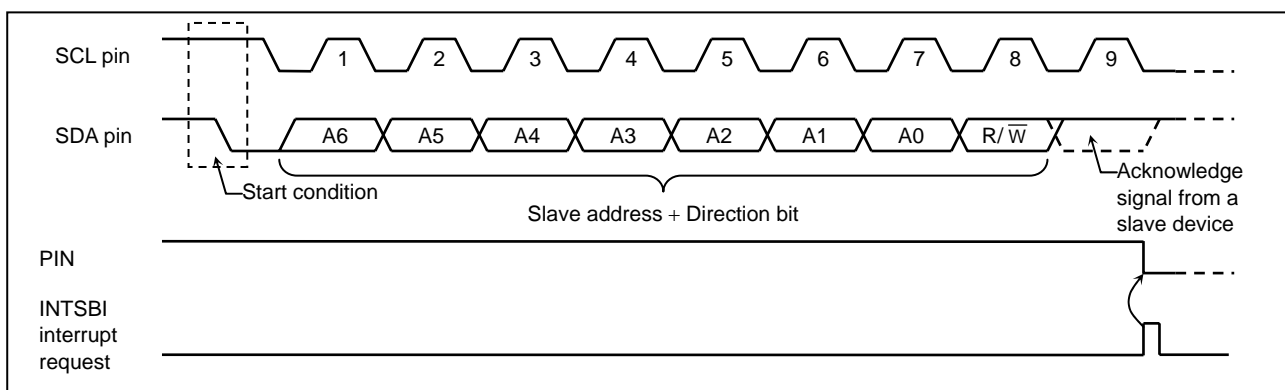
Figure 2.12.11 Start Condition Generation and Slave Address Transfer

(3) 1-word data transfer

Check the MST by the INTSBI interrupt process after an 1-word data transfer is completed, and determine whether the mode is a master or slave.

a. When the MST is "1" (Master mode)

Check the TRX and determine whether the mode is a transmitter or receiver.

1. When the TRX is "1" (Transmitter mode)

Test the LRB. When the LRB is "1", a receiver does not request data. Implement the process to generate a stop condition (Described later) and terminate data transfer.

When the LRB is "0", the receiver requests next data. When the next transmitted data is other than 8 bits, set the BC, set the ACK to "1", and write the transmitted data to the SBIDBR. After writing the data, the PIN becomes "1", a serial clock pulse is generated for transferring a next 1 word of data from the SCL pin, and then the 1 word of data is transmitted. After the data is transmitted, and an INTSBI interrupt request occurs. The PIN become "0" and the SCL pin is set to low level. If the data to be transferred is more than one word in length, repeat the procedure from the LRB test above.
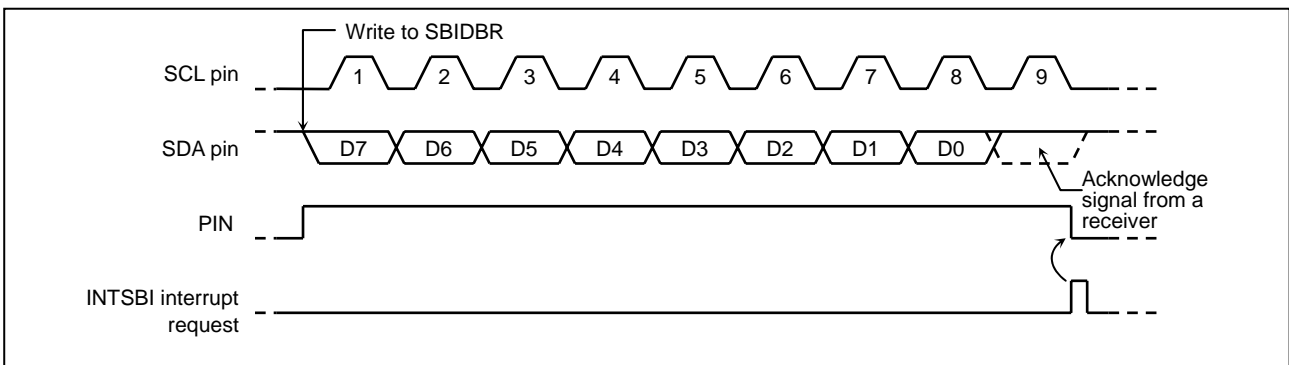
Figure 2.12.12 Example of when BC = "000", ACK = "1"

2. When the TRX is "0" (Receiver mode)

When the next transmitted data is other than of 8 bits, set the BC again. Set the ACK to "1" and read the received data from the SBIDBR (Reading data is undefined immediately after a slave address is sent). After the data is read, the PIN becomes "1". A serial bus interface circuit outputs a serial clock pulse to the SCL to transfer next 1-word of data and sets the SDA pin to "0" at the acknowledge signal timing.

An INTSBI interrupt request occurs and the PIN becomes "0". Then a serial bus interface circuit outputs a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.
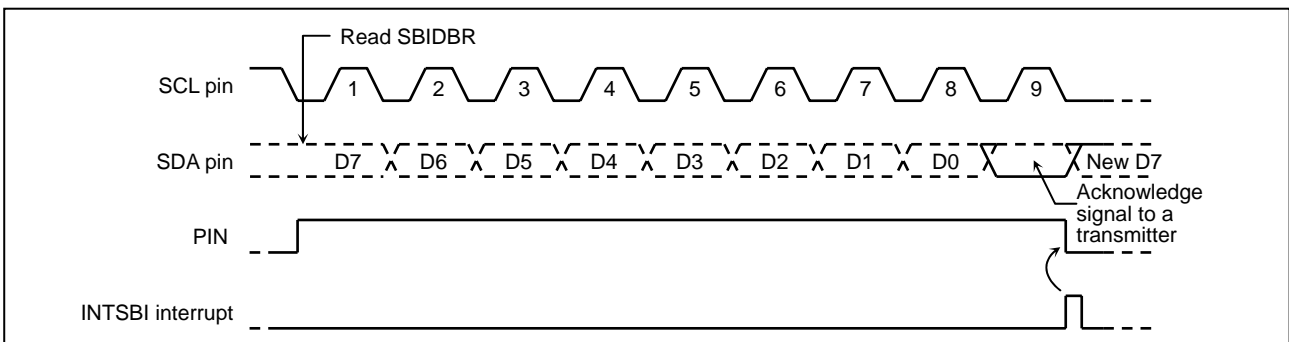
Figure 2.12.13 Example of when BC = "000", ACK = "1"

www.DataSheet4U.com

To make the transmitter terminate transmit, clear the ACK to "0" before reading data which is 1-word before the last data to be received. A serial bus interface circuit does not generate a clock pulse for the acknowledge signal by clearing ACK. In the interrupt routine of end of transmission, when the BC is set to "001" and read the data, PIN is set to "1" and generates a clock pulse for a 1-bit data transfer. In this case, since the master device is a receiver, the SDA line on a bus keeps the high-level. The transmitter receives the high-level signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete.

After 1-bit data is received and an interrupt request has occurred, generates the stop condition to terminate transmit, generate the stop condition to terminate data transfer.
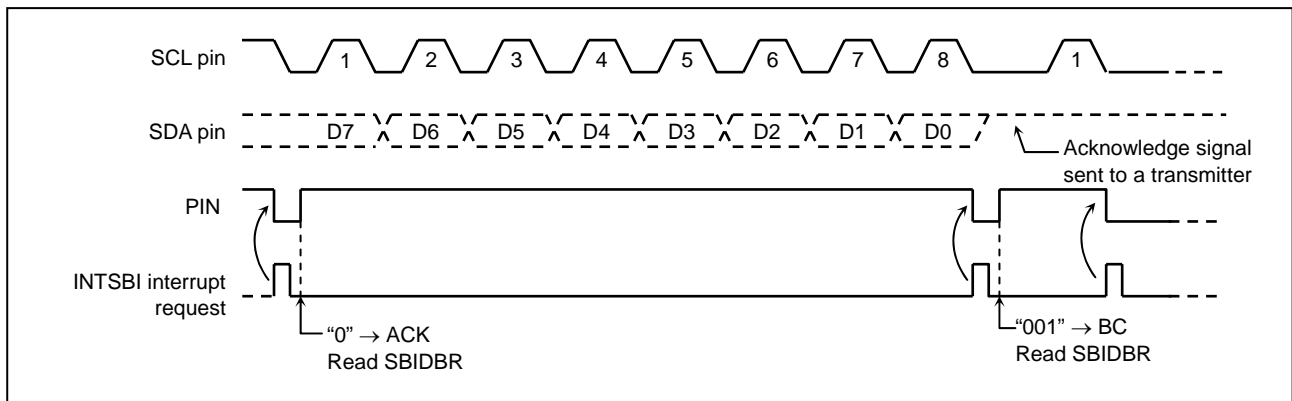
Figure 2.12.14 Termination of Data Transfer in Master Receiver Mode

b.   When the MST is "0" (Slave mode)

In the slave mode, a serial bus interface circuit operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, the conditions of generating INTSBI are follows:

- When the received slave address matches to the value set by the I2CAR
- When a "GENERAL CALL" is received
- At the end of transferring or receiving after matching of slave address or receiving of "GENERAL CALL"

A serial bus interface circuit changes to a slave mode if arbitration is lost in the master mode. And an INTSBI interrupt request occurs when word data transfer terminates after losing arbitration. The behavior of INTSBI and PIN after losing arbitration are shown in Table 2.12.3.

Table 2.12.3 The Behavior of INTSBI and PIN after Losing Arbitration

	When the Arbitration Lost Occurs during Transmission of Slave Address as a Master	When the Arbitration Lost Occurs during Transmission of Data as a Master Transmit Mode
INTSBI	INTSBI is generated at the termination of word data.	
PIN	When the slave address matches the value set by I2CAR, the PIN is cleared to "0" by generating of INTSBI. When the slave address doesn't match the value set by I2CAR, the PIN keeps "1".	PIN keeps "1".

Check the AL (Bit3 in the SBISR), the TRX (Bit6 in the SBISR), the AAS (Bit2 in the SBISR), and the AD0 (Bit1 in the SBISR) and implements processes according to conditions listed in Table 2.12.4.

Table 2.12.4 Operation in the Slave Mode

TRX	AL	AAS	AD0	Conditions	Process
1	1	1	0	A serial bus interface circuit loses arbitration when transmitting a slave address. And receives a slave address of which the value of the direction bit sent from another master is "1".	Set the number of bits in 1 word to the BC and write transmitted data to the SBIDBR.
	0	1	0	In the slave receiver mode, a serial bus interface circuit receives a slave address of which the value of the direction bit sent from the master is "1".	
		0	0	In the slave transmitter mode, 1-word data is transmitted.	Test the LRB. If the LRB is set to "1", set the PIN to "1" since the receiver does not request next data. Then, clear the TRX to "0" to release the bus. If the LRB is set to "0", set the number of bits in 1 word to the BC and write transmitted data to the SBIDBR since the receiver requests next data.
0	1	1	1/0	A serial bus interface circuit loses arbitration when transmitting a slave address. And receives a slave address of which the value of the direction bit sent from another master is "0" or receives a "GENERAL CALL".	Read the SBIDBR for setting the PIN to "1" (Reading dummy data) or write "1" to the PIN.
		0	0	A serial bus interface circuit loses arbitration when transmitting a slave address or data. And terminates transferring word data.	A serial bus interface circuit is changed to slave mode. To clear AL to "0", read the SBIDBR or write the data to SBIDBR.
	0	1	1/0	In the slave receiver mode, a serial bus interface circuit receives a slave address of which the value of the direction bit sent from the master is "0" or receives "GENERAL CALL".	Read the SBIDBR for setting the PIN to "1" (Reading dummy data) or write "1" to the PIN.
		0	1/0	In the slave receiver mode, a serial bus interface circuit terminates receiving of 1-word data.	Set the number of bits in 1-word to the BC and read received data from the SBIDBR.

Note: In the slave mode, if the slave address set in I2CAR is "00000000B", the TRX changes to "1" by receiving the start byte data "00000001B".

(4) Stop condition generation

When the BB is "1", a sequence of generating a stop condition is started by setting "1" to the MST, TRX and PIN, and clear "0" to the BB. Do not modify the contents of the MST, TRX, BB, PIN until a stop condition is generated on a bus.

When a SCL line on a bus is pulled-down by other devices, a serial bus interface circuit generates a stop condition after they release a SCL line.
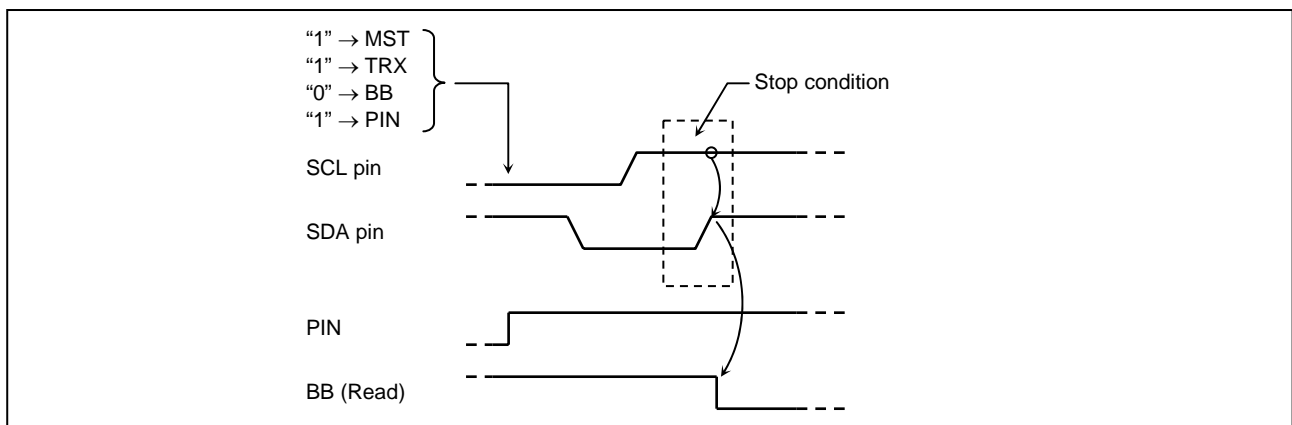
Figure 2.12.15 Stop Condition Generation

(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 a serial bus interface circuit.

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 a bus, a bus is assumed to be in a busy state from other devices. Test the BB until it becomes "0" to check that the SCL pin a serial bus interface circuit is released. Test the LRB until it becomes "1" to check that the SCL line on a bus is not pulled-down to the low level by other devices. After confirming that a 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 a bus is free until the time to generate a start condition.

Note: When restarting after receiving in master receiver mode, because the device doesn't send an acknowledgment as a last data, the level of SCL line can not be confirmed by reading LRB. Therefore, confirm the status of SCL line by reading P5PRD register.
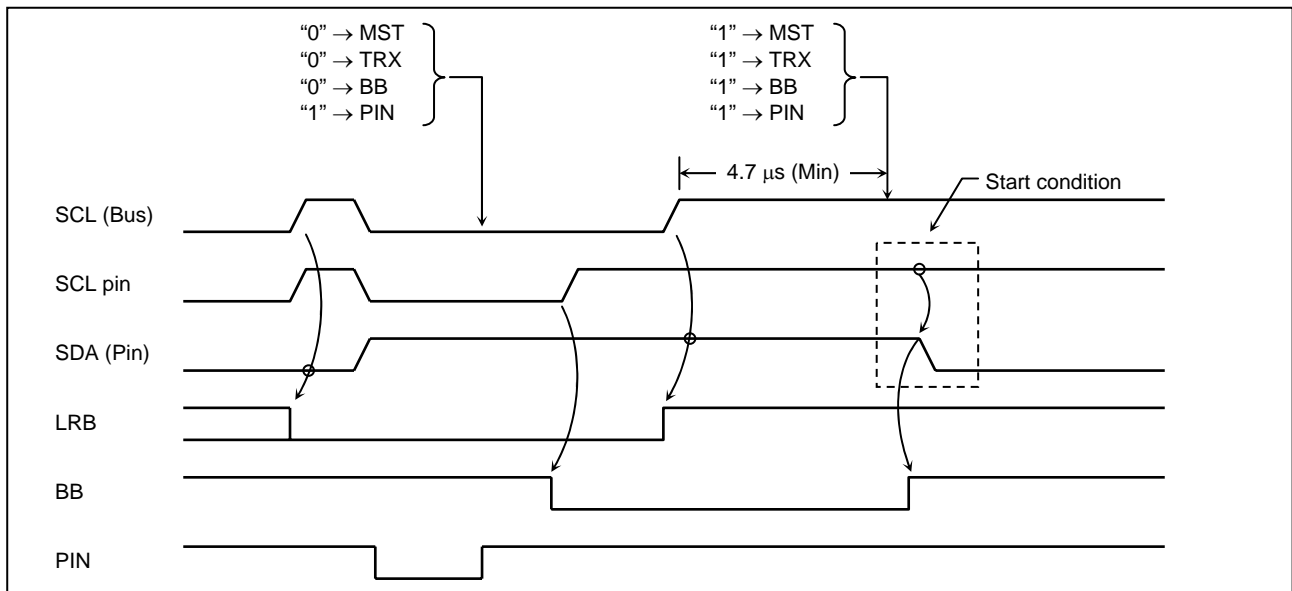
Figure 2.12.16 Timing Diagram when Restarting

2.13   SIO (Synchronous Serial Interface)

The TMP86FM48 contains two channels of SIO (Synchronous serial interface). These serial interfaces connect to an external device via SI1, SI2, SO1, SO2, $\overline{SCK1}$ and $\overline{SCK2}$ pins. The SI1, SI2, SO1, SO2, $\overline{SCK1}$ and $\overline{SCK2}$ pins respectively are shared with P05, P11, P06, P10, P07 and P12. When these pins are used as serial interface, the output latches for each port of P0 and P1 must be set to "1".

Because SIO1 and SIO2 are the same except that the registers and the function pin for each SIO are assigned as different specification, explanation here is made of only SIO1. The registers for SIO1 and SIO2 are listed in Table below.

Table 2.13.1   The Registers for SIO1 and SIO2

	SIO1		SIO2	
	Register	Address	Register	Address
SIO control register	SIO1CR	0017H	SIO2CR	001BH
SIO status register	SIO1SR	0018H	SIO2SR	001CH
SIO receive buffer register	SIO1RDB	0019H	SIO2RDB	001DH
SIO transmit buffer register	SIO1TDB	0019H	SIO2TDB	001DH

2.13.1   Configuration
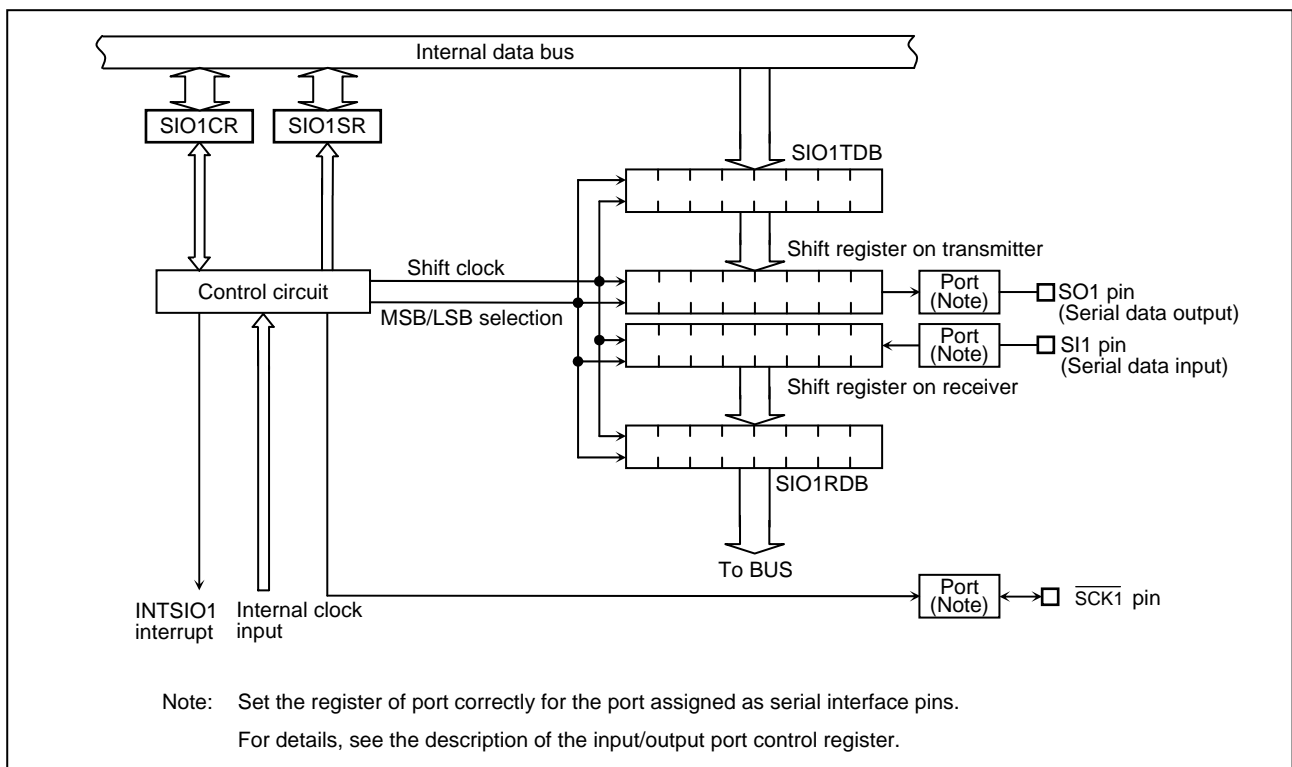
Figure 2.13.1   Synchronous Serial Interface

2.13.2 Control

The SIO is controlled using the serial interface control register (SIO1CR). The operating status of the serial interface can be inspected by reading the status register (SIO1SR).

Serial Interface Control Register 1

SIO1CR (0017H)	7	6	5	4	3	2	1	0	

SIO1CR (0017H): SIOS | SIOINH | SIQM | SIODIR | SCK   (Initial value: 0000 0000)

SIOS	Specify start/stop of transfer	0:	Stop					R/W
		1:	Start					
SIOINH	Forcibly stops transfer (Note 1)	0:	–					
		1:	Forcibly stop (Automatically cleared to "0" after stopping)					
SIOM	Selects transfer mode	00: Transmit mode						
		01: Receive mode						
		10: Transmit/receive mode						
		11: Reserved						
SIODIR	Selects direction of transfer	0:	MSB (Transfer beginning with bit7)					
		1:	LSB (Transfer beginning with bit0)					

			NORMAL 1/2 or IDLE 1/2 mode		SLOW/SLEEP mode
			TBTCR <DV7CK> = "0"	TBTCR <DV7CK> = "1"	
SCK	Selects serial clock	000	$fc/2^{12}$	$fs/2^4$	$fs/2^4$
		001	$fc/2^8$	$fc/2^8$	Reserved
		010	$fc/2^7$	$fc/2^7$	Reserved
		011	$fc/2^6$	$fc/2^6$	Reserved
		100	$fc/2^5$	$fc/2^5$	Reserved
		101	$fc/2^4$	$fc/2^4$	Reserved
		110	$fc/2^3$	$fc/2^3$	Reserved
		111	External clock (input from $\overline{SCK1}$ pin)		

Note 1: When SIO1CR<SIOINH> is set to "1", SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and STOTDB register are initialized.

Note 2: Transfer mode, direction of transfer and serial clock must be select during the transfer is stopping (when SIO1SR<SIOF> = "0").

Note 3: fc: High frequency clock [Hz], fs: Low frequency clock, ∗: Don't care

Figure 2.13.2 Serial Interface Control Register

Serial Interface Status Register

```
           7      6      5      4      3       2      1      0
SIO1SR   SIOF   SEF    TXF    RXF   TXERR  RXERR                    (Initial value: 0010 00**)
(0018H)
```

SIOF	Serial transfer operation status monitor	0: Transfer finished 1: Transfer in progress	Read only
SEF	Number of clocks monitor	0: 8 clocks 1: 1 to 7 clocks	
TXF	Transmit buffer empty flag	0: Data exists in transmit buffer 1: No data exists in transmit buffer	
RXF	Receive buffer full flag	0: No data exists in receive buffer 1: Data exists in receive buffer	
TXERR	Transfer operation error flag	Read 0: − (No error exist) 1: Transmit buffer under run occurs in an external clock mode. Write 0: Clear the flag 1: − (A write of "1" to this bit is ignored)	R/W
RXERR	Receive operation error flag	Read 0: − (No error exist) 1: Receive buffer over run occurs in an external clock mode. Write 0: Clear the flag 1: − (A write of "1" to this bit is ignored)	

Note 1: The operation error flag (TXERR and RXERR) are not automatically cleared by stopping transfer with SIO1CR<SIOS> = "0". Therefore, set these bits to "0" for clearing these error flag. Or set SIO1CR<SIOINH> to "1".

Note 2: *: Don't care

Figure 2.13.3 Serial Interface Status Register

Receive buffer register

```
             7    6    5    4    3    2    1    0    Read only
SIO1RDB                                              (Initial value: 0000 0000)
(0019H)
```

Transmit buffer register

```
             7    6    5    4    3    2    1    0    Write only
SIO1TDB                                              (Initial value: **** ****)
(0019H)
```

Note 1: SIO1TDB is write only register. A bit manipulation should not be performed on the transmit buffer register using a read-modify-write instruction.

Note 2: The SIO1TDB should be written after checking SIO1SR<TXF> = "1". When SIO1SR<TXF> is "0", the writing data can't be transferred to SIO1TDB even if write instruction is executed to SIO1TDB.

Note 3: *: Don't care

Figure 2.13.4 Receive Buffer Register and Transmit Buffer Register

2.13.3 Functional Description

(1) Serial clock

a. Clock source

The serial clock can be selected by using SIO1CR<SCK>. When the serial clock is changed, the writing instruction to SIO1CR<SCK> should be executed while the transfer is stopped (when SIO1SR<SIOF> = "0").

1. Internal clock

Setting the SIO1CR<SCK> to other than "111" outputs the clock (Shown in Table 2.8.2) as serial clock outputs from $\overline{SCK1}$ pin. At the before beginning or finishing of a transfer, $\overline{SCK1}$ pin is kept in high level.

When writing (in the transmit mode) or reading (in the receive mode) data can not 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 completed. The maximum time from releasing the automatic-wait function by reading or writing a data is 1 cycle of the selected serial clock until the serial clock comes out from $\overline{SCK1}$ pin.
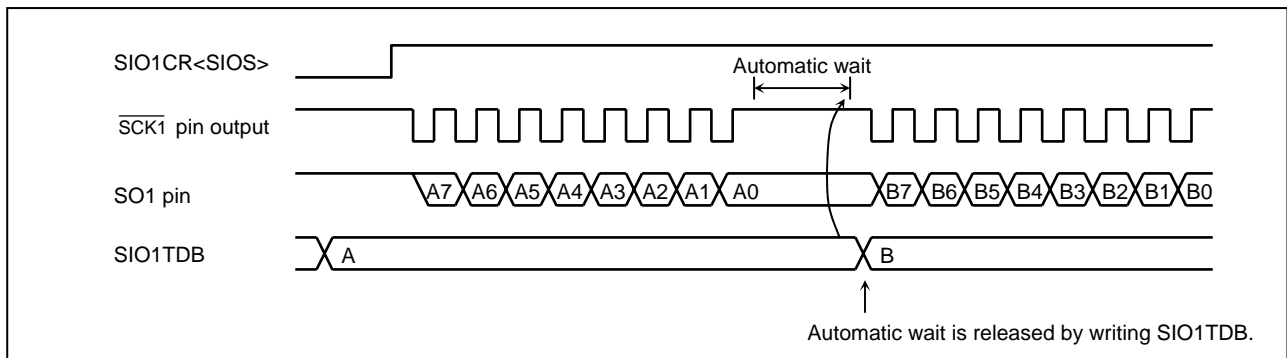
Figure 2.13.5 Automatic-wait Function (Example of transmit mode)

Table 2.13.2 Serial Clock Rate (fc = 16 MHz, fs = 32.768kHz)

| | NORMAL1/2, IDLE1/2 Mode | | | | SLOW1/2, SLEEP1/2 Mode | |
| | TBTCR<DV7CK> = "0" | | TBTCR<DV7CK> = "1" | | | |
	Serial Clock	Baud Rate	Serial Clock	Baud Rate	Serial Clock	Baud Rate
000	$fc/2^{12}$	3.906 kbps	$fs/2^4$	2048 bps	$fs/2^4$	2048 bps
001	$fc/2^8$	62.5 kbps	$fc/2^8$	62.5 kbps	Reserved	–
010	$fc/2^7$	125 kbps	$fc/2^7$	125 kbps	Reserved	–
011	$fc/2^6$	250 kbps	$fc/2^6$	250 kbps	Reserved	–
100	$fc/2^5$	500 kbps	$fc/2^5$	500 kbps	Reserved	–
101	$fc/2^4$	1.00 Mbps	$fc/2^4$	1.00 Mbps	Reserved	–
110	$fc/2^3$	2.00 Mbps	$fc/2^3$	2.00 Mbps	Reserved	–

www.DataSheet4U.com

2. External clock

When an external clock is selected by setting SIO1CR<SCK> to "111", the clock via the $\overline{\text{SCK1}}$ pin from an external source is used as the serial clock.

To ensure shift operation, the serial clock pulse width must be 4/fc or more for both "H" and "L" levels.
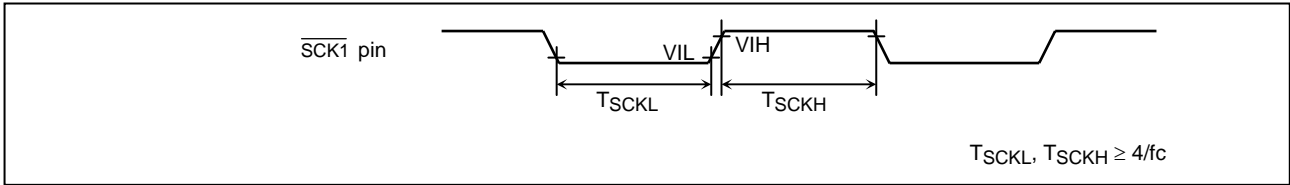
Figure 2.13.6 External Clock

b. Shift edges

The leading edge is used to transmit data, and the trailing edge is used to receive data.

1. Leading edge shift

Data is shifted on leading edges of the serial clock (Falling edges of the $\overline{\text{SCK1}}$ pin input/output).

2. Trailing edge shift

Data is shifted on trailing edges of the serial clock (Rising edges of the $\overline{\text{SCK1}}$ pin input/output).
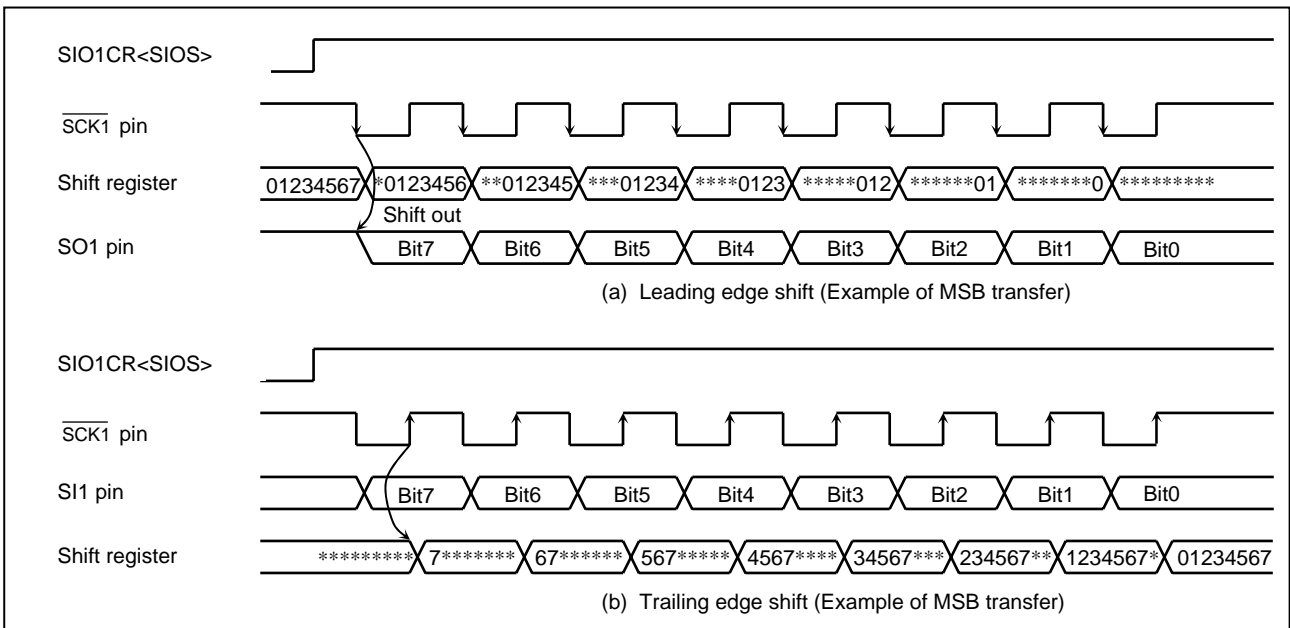
Figure 2.13.7 Shift Edge

(2) Transfer bit direction

Transfer data direction can be selected by using SIO1CR<SIODIR>. The transfer data direction can't be set individually for transmit and receive operations.

When the data direction is changed, the writing instruction to SIO1CR<SIODIR> should be executed while the transfer is stopped (when SIO1SR<SIOF> = "0").
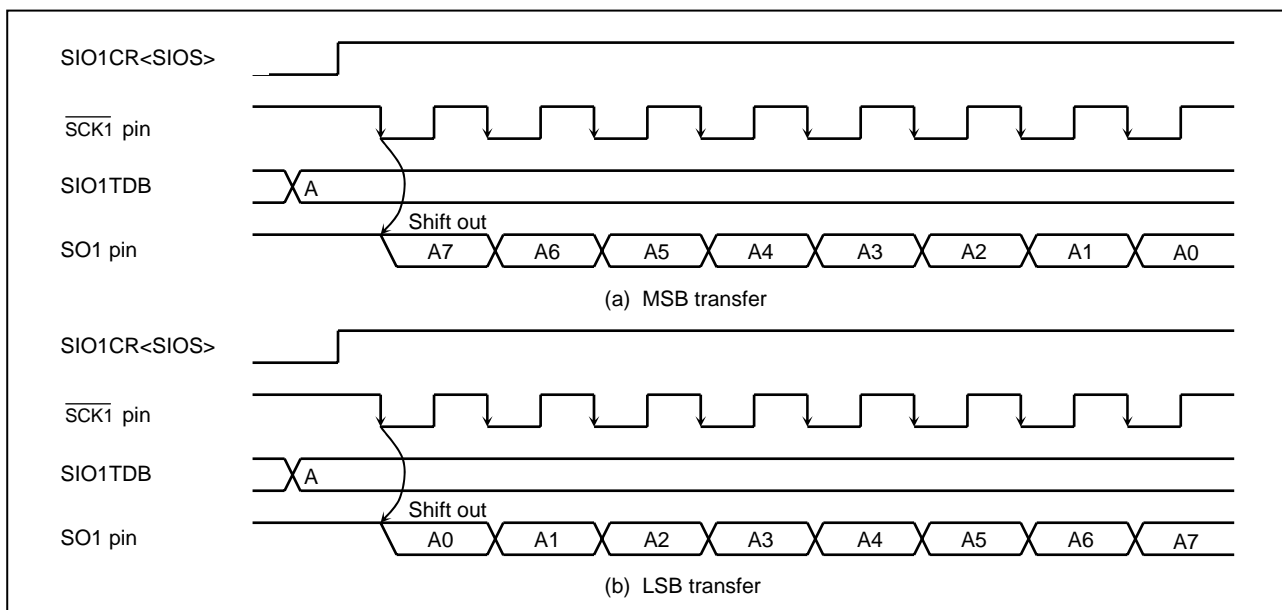
Figure 2.13.8 Transfer Bit Direction (Example of transmit mode)

a. Transmit mode

    1. MSB transmit mode

      MSB transmit mode is selected by setting SIO1CR<SIODIR> to "0", in which case the data is transferred sequentially beginning with the most significant bit (Bit7).

    2. LSB transmit mode

      LSB transmit mode is selected by setting SIO1CR<SIODIR> to "1", in which case the data is transferred sequentially beginning with the least significant bit (Bit0).

b. Receive mode

    1. MSB receive mode

      MSB receive mode is selected by setting SIO1CR<SIODIR> to "0", in which case the data is received sequentially beginning with the most significant bit (Bit7).

    2. LSB receive mode

      LSB receive mode is selected by setting SIO1CR<SIODIR> to "1", in which case the data is received sequentially beginning with the least significant bit (Bit0).

c. Transmit/receive mode

    1. MSB transmit/receive mode

      MSB transmit/receive mode are selected by setting SIO1CR<SIODIR> to "0" in which case the data is transferred sequentially beginning with the most significant bit (Bit7) and the data is received sequentially beginning with the most significant (Bit7).

2. LSB transmit/receive mode

   LSB transmit/receive mode are selected by setting SIO1CR<SIODIR> to "1", in which case the data is transferred sequentially beginning with the least significant bit (Bit0) and the data is received sequentially beginning with the least significant (Bit0).

(3) Transfer modes

Transmit, receive and transmit/receive mode are selected by using SIO1CR<SIOM>.

a. Transmit mode

Transmit mode is selected by writing "00" to SIO1CR<SIOM>.

1. Starting the transmit operation

Transmit mode is selected by setting "00" to SIO1CR<SIOM>. Serial clock is selected by using SIO1CR<SCK>. Transfer direction is selected by using SIO1CR<SIODIR>.

When a transmit data is written to the transmit buffer register (SIO1TDB), SIO1SR<TXF> is cleared to "0".

After SIO1CR<SIOS> is set to "1", SIO1SR<SIOF> is set synchronously to "1" the falling edge of $\overline{SCK1}$ pin.

The data is transferred sequentially starting from SO1 pin with the direction of the bit specified by SBIDIR<SIODIR>, synchronizing with the $\overline{SCK1}$ pin's falling edge.

SIO1SR<SEF> is kept in high level, between the first clock falling edge of $\overline{SCK1}$ pin and eighth clock falling edge.

SIO1SR<TXF> is set to "1" at the rising edge of $\overline{SCK1}$ pin after the data written to the SIO1TDB is transferred to shift register, then the INTSIO1 interrupt request is generated, synchronizing with the next falling edge on $\overline{SCK1}$ pin.

Note 1: In internal clock operation, when SIO1CR<SIOS> is set to "1", transfer mode does not start without writing a transmit data to the transmit buffer register (SIO1TDB).

Note 2: In internal clock operation, when the SIO1CR<SIOS> is set to "1", SIO1TDB is transferred to shift register after maximum 1-cycle of serial clock frequency, then a serial clock is output from $\overline{SCK1}$ pin.

Note 3: In external clock operation, when the falling edge is input from $\overline{SCK1}$ pin after SIO1CR<SIOS> is set to "1", SIO1TDB is transferred to shift register immediately.

2. During the transmit operation

When data is written to SIO1TDB, SIO1SR<TXF> is cleared to "0".

In internal clock operation, in case a next transmit data is not written to SIO1TDB, the serial clock stops to "H" level by an automatic-wait function when all of the bit set in the SIO1TDB has been transmitted. Automatic-wait function is released by writing a transmit data to SIO1TDB. Then, transmit operation is restarted after maximum 1-cycle of serial clock.

When the next data is written to the SIO1TDB before termination of previous 8-bit data with SIO1SR<TXF> = "1", the next data is continuously transferred after transmission of previous data.

In external clock operation, after SIO1SR<TXF> is set to "1", the transmit data must be written to SIO1TDB before the shift operation of the next data begins.

If the transmit data is not written to SIO1TDB, transmit error occurs immediately after shift operation is started. Then, INTSIO1 interrupt request is generated after SIO1SR<TXERR> is set to "1".

www.DataSheet4U.com

3. Stopping the transmit operation

There are two ways for stopping transmits operation.

• The way of clearing SIO1CR<SIOS>.

When SIO1CR<SIOS> is cleared to "0", transmit operation is stopped after all transfer of the data is finished. When transmit operation is finished, SIO1SR<SIOF> is cleared to "0" and SO1 pin is kept in high level.

In external clock operation, SIO1CR<SIOS> must be cleared to "0" before SIO1SR<SEF> is set to "1" by beginning next transfer.

• The way of setting SIO1CR<SIOINH>.

Transmit operation is stopped immediately after SIO1CR<SIOINH> is set to "1". In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

Figure 2.13.9 Example of Internal Clock and MSB Transmit Mode

Figure 2.13.10 Example of External Clock and MSB Transmit Mode

$$4/fc \leq T_{SODH} \leq 8/fc$$
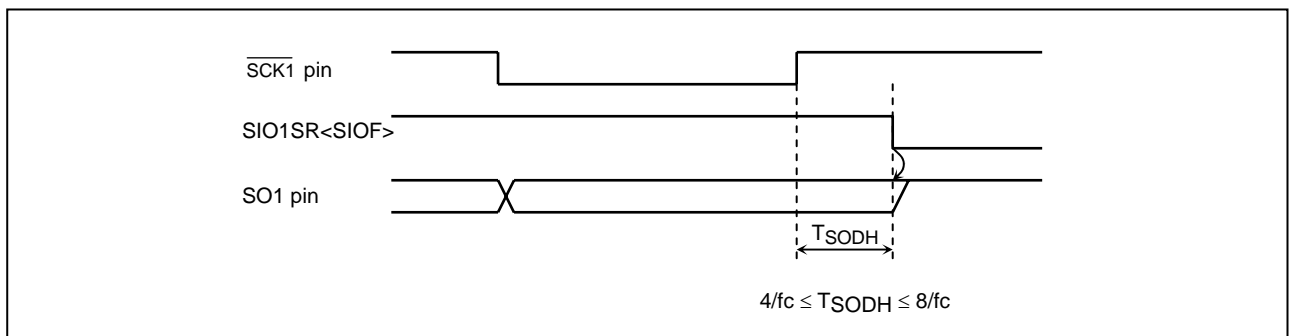
Figure 2.13.11 Hold Time of the End of Transmit Mode

www.DataSheet4U.com

4. Transmit error processing

Transmit errors occur on the following situation.

- Shift operation starts before writing next transmit data to SIO1TDB in external clock operation.

    If transmit errors occur during transmit operation, SIO1SR<TXERR> is set to "1" immediately after starting shift operation. Synchronizing with the next serial clock falling edge, INTSIO1 interrupt request is generated.

    If shift operation starts before writing data to SIO1TDB after SIO1CR <SIOS> is set to "1", SIO1SR<TXERR> is set to "1" immediately after shift operation is started and then INTSIO1 interrupt request is generated.

    SO1 pin is kept in high level when SIO1SR<TXERR> is set to "1". When transmit error occurs, transmit operation must be forcibly stop by writing SIO1CR<SIOINH> to "1". In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.
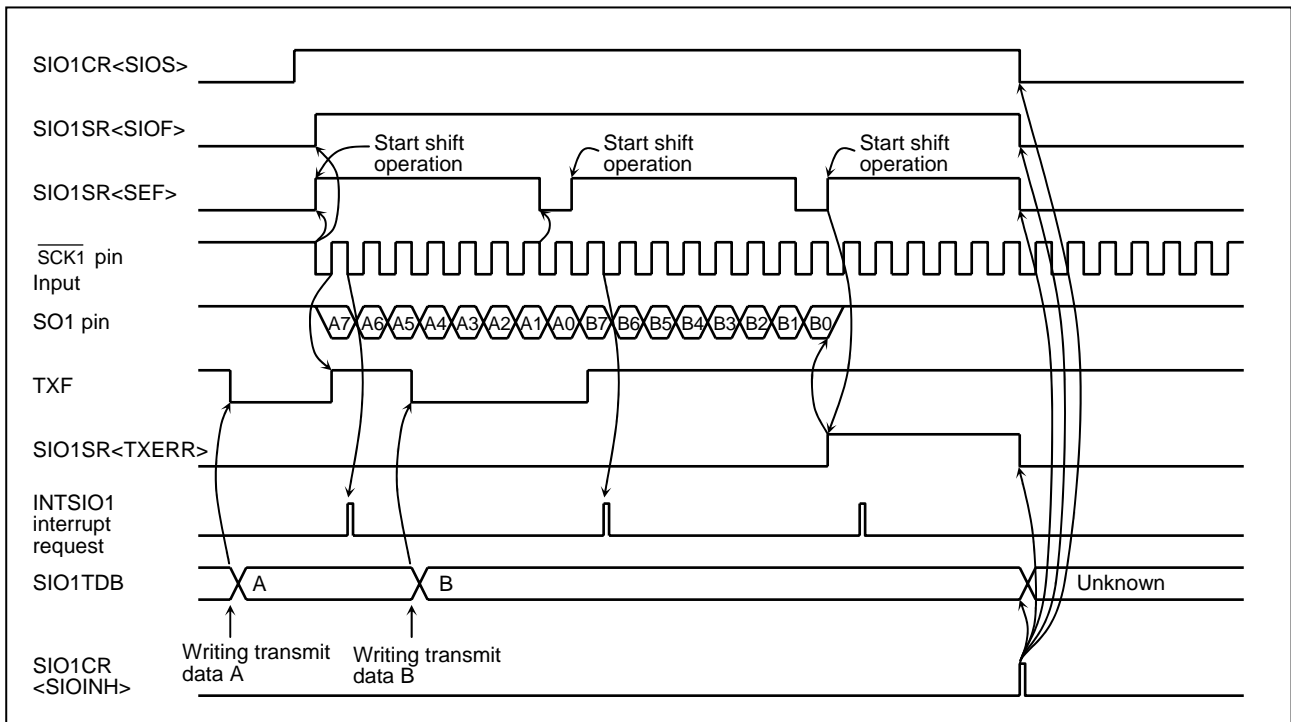
Figure 2.13.12 Example of Transmit Error Processing

b. Receive mode

The receive mode is selected by writing "01" to SIO1CR<SIOM>.

1. Starting the receive operation

Receive mode is selected by setting "01" to SIO1CR<SIOM>. Serial clock is selected by using SIO1CR<SCK>. Transfer direction is selected by using SIO1CR<SIODIR>.

After SIO1CR<SIOS> is set to "1", SIO1SR<SIOF> is set synchronously to "1" the falling edge of $\overline{SCK1}$ pin.

Synchronizing with the $\overline{SCK1}$ pin's rising edge, the data is received sequentially from SI1 pin with the direction of the bit specified by SBIDIR<SIODIR>.

SIO1SR<SEF> is kept in high level, between the first clock falling edge of $\overline{SCK1}$ pin and eighth clock falling edge.

When 8-bit data is received, the data is transferred to SIO1RDB from shift register. INTSIO1 interrupt request is generated and SIO1SR<RXF> is set to "1".

Note: In internal clock operation, when the SIO1CR<SIOS> is set to "1", the serial clock is generated from $\overline{SCK1}$ pin after maximum 1-cycle of serial clock frequency.

2. During the receive operation

The SIO1SR<RXF> is cleared to "0" by reading a data from SIO1RDB.

In the internal clock operation, the serial clock stops to "H" level by an automatic-wait function when the all of the 8-bit data has been received. Automatic-wait function is released by reading a received data from SIO1RDB. Then, receive operation is restarted after maximum 1-cycle of serial clock.

In external clock operation, after SIO1SR<RXF> is set to "1", the received data must be read from SIO1RDB before the next data shift-in operation is finished.

If received data is not read out from SIO1RDB, receive error occurs immediately after shift operation is finished. Then INTSIO interrupt request is generated after SIO1SR<RXERR> is set to "1".

3. Stopping the receive operation

There are two ways for stopping the receive operation.

• The way of clearing SIO1CR<SIOS>.

When SIO1CR<SIOS> is cleared to "0", receive operation is stopped after all of the data is finished to receive. When receive operation is finished, SIO1SR<SIOF> is cleared to "0".

In external clock operation, SIO1CR<SIOS> must be cleared to "0" before SIO1SR<SEF> is set to "1" by starting the next shift operation.

• The way of setting SIO1CR<SIOINH>.

Receive operation is stopped immediately after SIO1CR<SIOINH> is set to "1". In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.
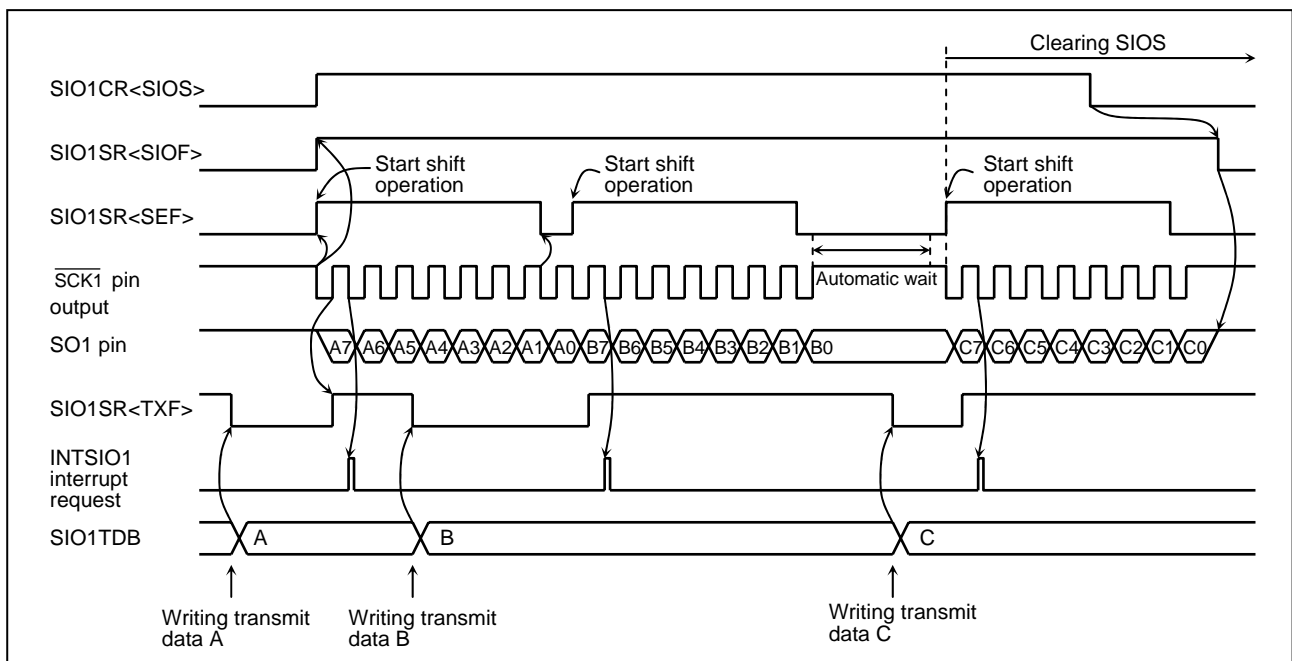
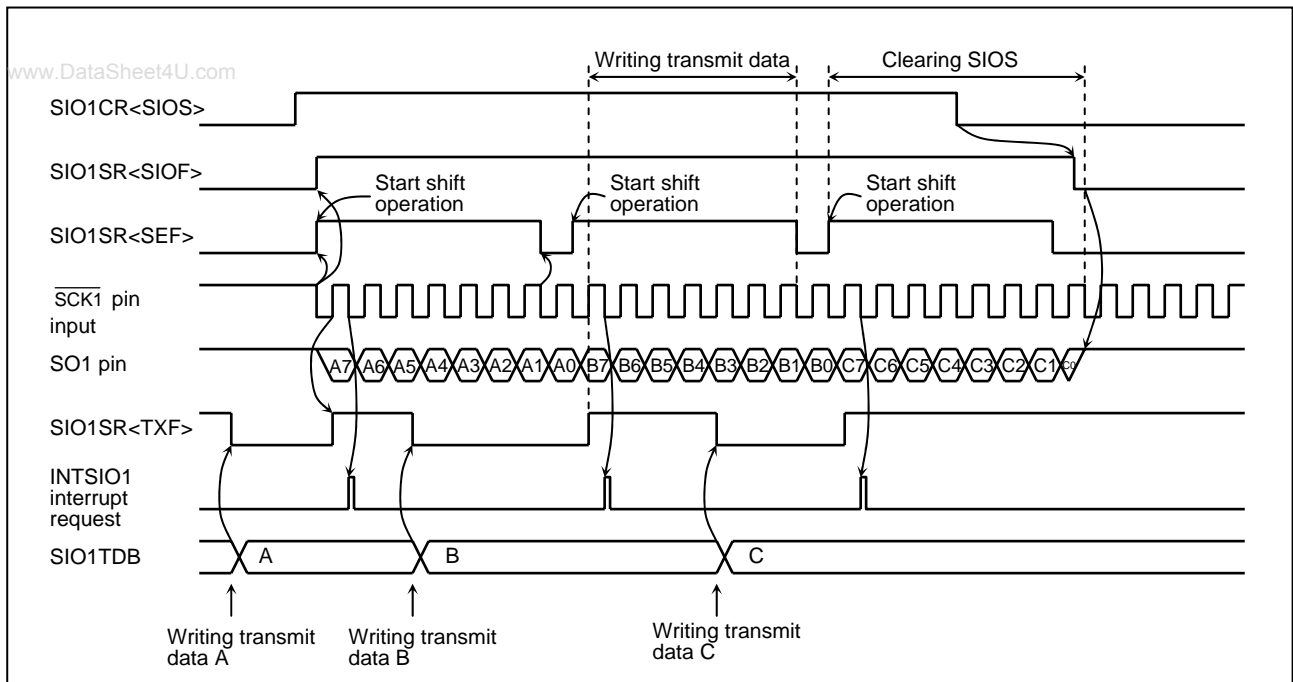Figure 2.13.13 Example of Internal Clock and MSB Receive Mode

Figure 2.13.14 Example of External Clock and MSB Receive Mode

4. Receive error processing

Receive errors occur on the following situation. To protect SIO1RDB and the shift register contents, the received data is ignored while the SIO1SR<RXERR> is "1".

- Shift operation is finished before reading out received data from SIO1RDB at SIO1SR<RXF> is "1" in an external clock operation.

  If receive error occurs, set the SIOCR1<SIOS> to "0" for reading the data that received immediately before error occurence. And read the data from SIO1RDB. Data in shift register (at errors occur) can be read by reading the SIO1RDB again.

  When SIO1SR<RXERR> is cleared to "0" after reading the received data, SIO1SR<RXF> is cleared to "0".

  After clearing SIO1CR<SIOS> to "0", when 8-bit serial clock is input to $\overline{SCK1}$ pin, receive operation is stopped. To restart the receive operation, confirm that SIO1SR<SIOF> is cleared to "0".

  If the receive error occurs, set the SIOCR1<SIOINH> to "1" for stopping the receive operation immediately. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.
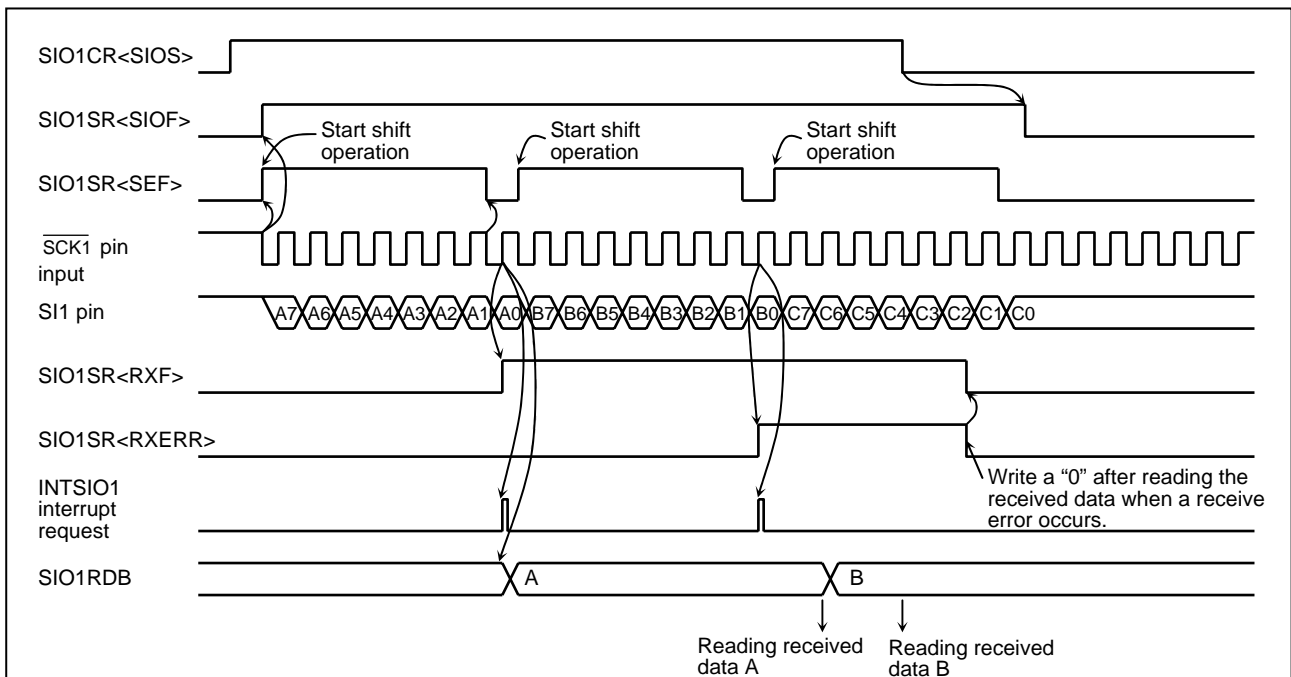
Figure 2.13.15 Example of Receive Error Processing

Note: If receive error is not corrected, an interrupt request does not generate after the error occurs.

c. Transmit/receive mode

The transmit/receive mode are selected by writing "10" to SIO1CR<SIOM>.

1. Starting the transmit/receive operation

Transmit/receive mode is selected by writing "10" to SIO1CR<SIOM>. Serial clock is selected by using SIO1CR<SCK>. Transfer direction is selected by using SIO1CR<SIODIR>.

When a transmit data is written to the transmit buffer register (SIO1TDB), SIO1SR<TXF> is cleared to "0".

After SIO1CR<SIOS> is set to "1", SIO1SR<SIOF> is set synchronously to the falling edge of $\overline{SCK1}$ pin.

The data is transferred sequentially starting from SO1 pin with the direction of the bit specified by SIO1CR<SIODIR>, synchronizing with the $\overline{SCK1}$ pin's falling edge. And receiving operation also starts with the direction of the bit specified by SIO1CR<SIODIR>, synchronizing with the $\overline{SCK1}$ pin's rising edge.

SIO1SR<SEF> is kept in high level between the first clock falling edge of $\overline{SCK1}$ pin and eighth clock falling edge.

SIO1SR<TXF> is set to "1" at the rising edge of $\overline{SCK1}$ pin after the data written to the SIO1TDB is transferred to shift register. When 8-bit data has been received, the received data is transferred to SIO1RDB from shift register, then the INTSIO1 interrupt request occurs, synchronizing with setting SIO1SR<RXF> to "1".

Note 1:   In internal clock operation, when the SIO1CR<SIOS> is set to "1", SIO1TDB is transferred to shift register after maximum 1-cycle of serial clock frequency, then a serial clock is output from $\overline{SCK1}$ pin.

Note 2:   In external clock operation, when the falling edge is input from $\overline{SCK1}$ pin after SIO1CR<SIOS> is set to "1", SIO1TDB is transferred to shift register immediately. When the rising edge is input from $\overline{SCK1}$ pin, receive operation also starts.

2. During the transmit/receive operation

When data is written to SIO1TDB, SIO1SR<TXF> is cleared to "0" and when a data is read from SIO1RDB, SIO1SR<RXF> is cleared to "0".

In internal clock operation, in case of the condition described below, the serial clock stops to "H" level by an automatic-wait function when all of the bit set in the data has been transmitted.

• Next transmit data is not written to SIO1TDB after reading a received data from SIO1RDB

• Received data is not read from SIO1RDB after writing a next transmit data to SIO1TDB

• Neither SIO1TDB nor SIO1RDB is accessed after transmission.

The automatic wait function is released by writing the next transmit data to SIO1TDB after reading the received data from SIO1RDB, or reading the received data from SIO1RDB after writing the next data to SIO1TDB. Then, transmit/receive operation is restarted after maximum 1 cycle of serial clock.

In external clock operation, reading the received data from SIO1RDB and writing the next data to SIO1TDB must be finished before the shift operation of the next data begins.

If the transmit data is not written to SIO1TDB after SIO1SR<TXF> is set to "1", transmit error occurs immediately after shift operation is started. When the transmit error occurred, SIO1SR<TXERR> is set to "1".

If received data is not read out from SIO1RDB before next shift operation starts after setting SIO1SR<RXF> to "1", receive error occurs immediately after shift operation is finished. When the receive error has occurred, SIO1SR<RXERR> is set to "1".

3. Stopping the transmit/receive operation

There are two ways for stopping the transmit/receive operation.

- The way of clearing SIO1CR<SIOS>.

  When SIO1CR<SIOS> is cleared to "0", transmit/receive operation is stopped after all transfer of the data is finished. When transmit/receive operation is finished, SIO1SR<SIOF> is cleared to "0" and SO1 pin is kept in high level.

  In external clock operation, SIO1CR<SIOS> must be cleared to "0" before SIO1SR<SEF> is set to "1" by beginning next transfer.

- The way of setting SIO1CR<SIOINH>.

  Transmit/receive operation is stopped immediately after SIO1CR<SIOINH> is set to "1". In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.
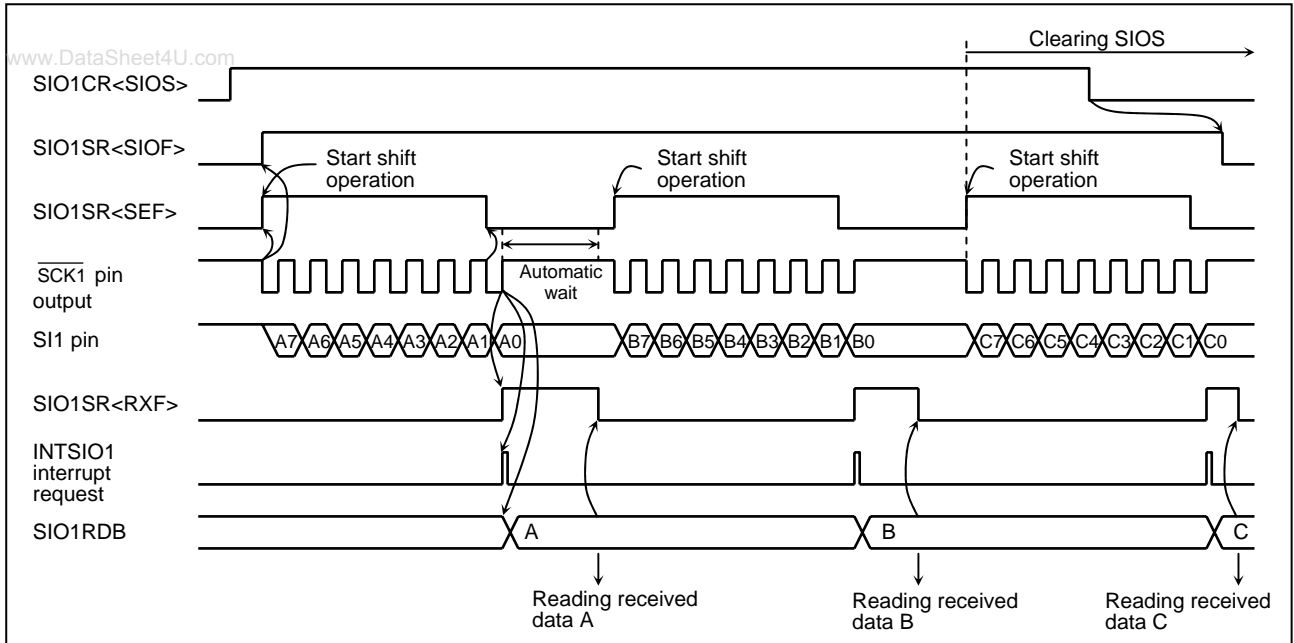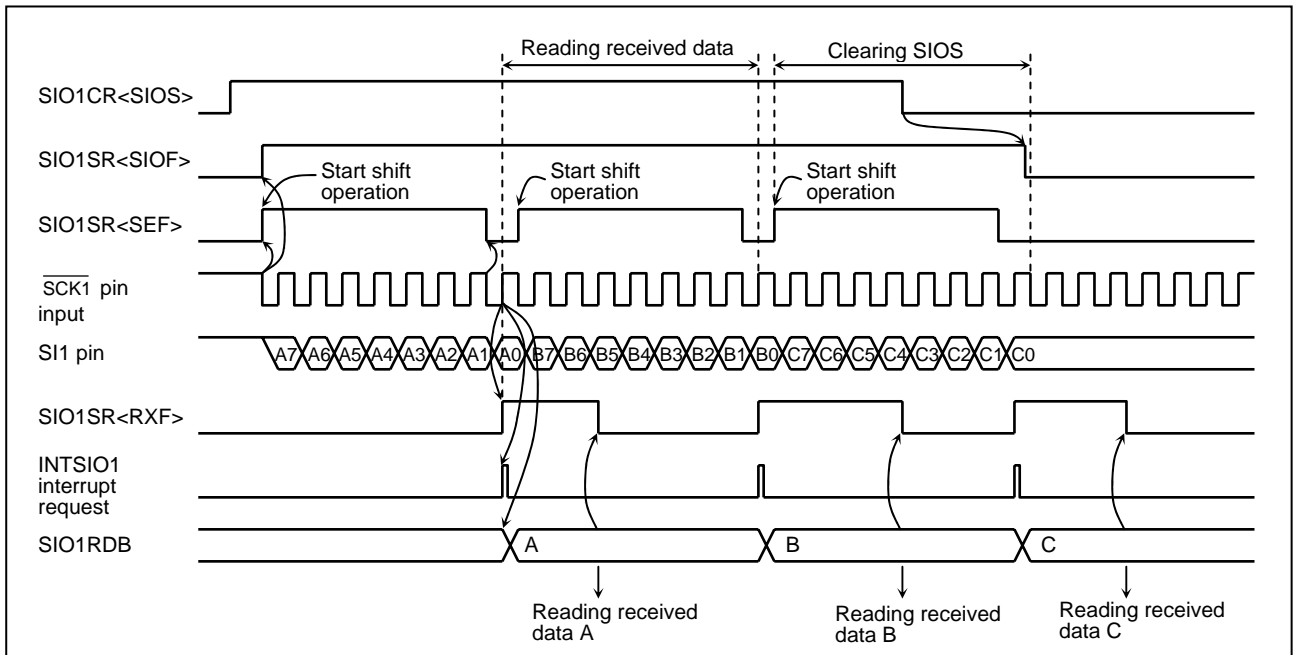

Figure 2.13.16 Example of Internal Clock and MSB Transmit/Receive Mode

Figure 2.13.17 Example of External Clock and MSB Transmit/Receive Mode

4. Transmit/receive error processing

Transmit/receive errors occur on the following situation. Corrective action is different, which errors occur transmits or receives.

Transmit errors

Transmit errors occur on the following situation.

- Shift operation starts before writing next transmit data to SIO1TDB in external clock operation.

    If transmit errors occur during transmit operation, SIO1SR<TXERR> is set to "1" immediately after starting shift operation. And INTSIO1 interrupt request is generated after all of the 8-bit data has been received.

    If shift operation starts before writing data to SIO1TDB after SIO1CR <SIOS> is set to "1", SIO1SR<TXERR> is set immediately after starting shift operation. And INTSIO1 interrupt request is generated after all of the 8-bit data has been received.

    SO1 pin is kept in high level when SIO1SR<TXERR> is set to "1". When transmit error occurs, transmit operation must be forcibly stop by writing SIO1CR<SIOINH> to "1" after the received data is read from SIO1RDB. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.
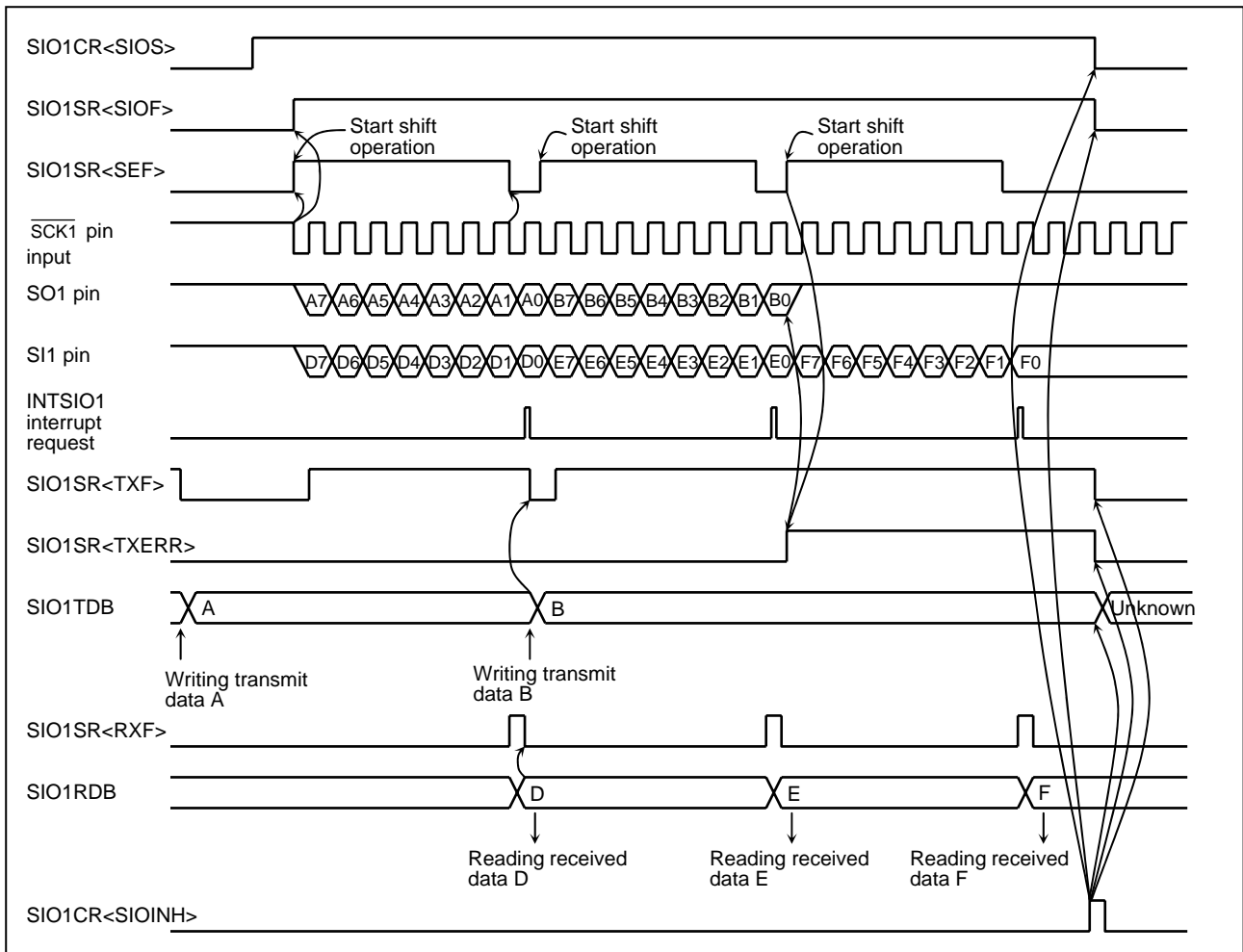
Figure 2.13.18 Example of Transmit/Receive (Transmit) Error Processing

Receive errors

Receive errors occur on the following situation. To protect SIO1RDB and the shift register contents, the received data is ignored while the SIO1SR<RXERR> is "1".

- Shift operation is finished before reading out received data from SIO1RDB at SIO1SR<RXF> is "1" in an external clock operation.

  If receive error occurs, set the SIO1CR<SIOS> to "0" for reading the data that received immediately before error occurence. And read the data from SIO1RDB. Data in shift register (at errors occur) can be read by reading the SIO1RDB again.

  When SIO1SR<RXERR> is cleared to "0" after reading the received data, SIO1SR<RXF> is cleared to "0".

  After clearing SIO1CR<SIOS> to "0", when 8-bit serial clock is input to $\overline{SCK1}$ pin, receive operation is stopped. To restart the receive operation, confirm that SIO1SR<SIOF> is cleared to "0".

  If the received error occurs, set the SIO1CR<SIOINH> to "1" for stopping the receive operation immediately. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.
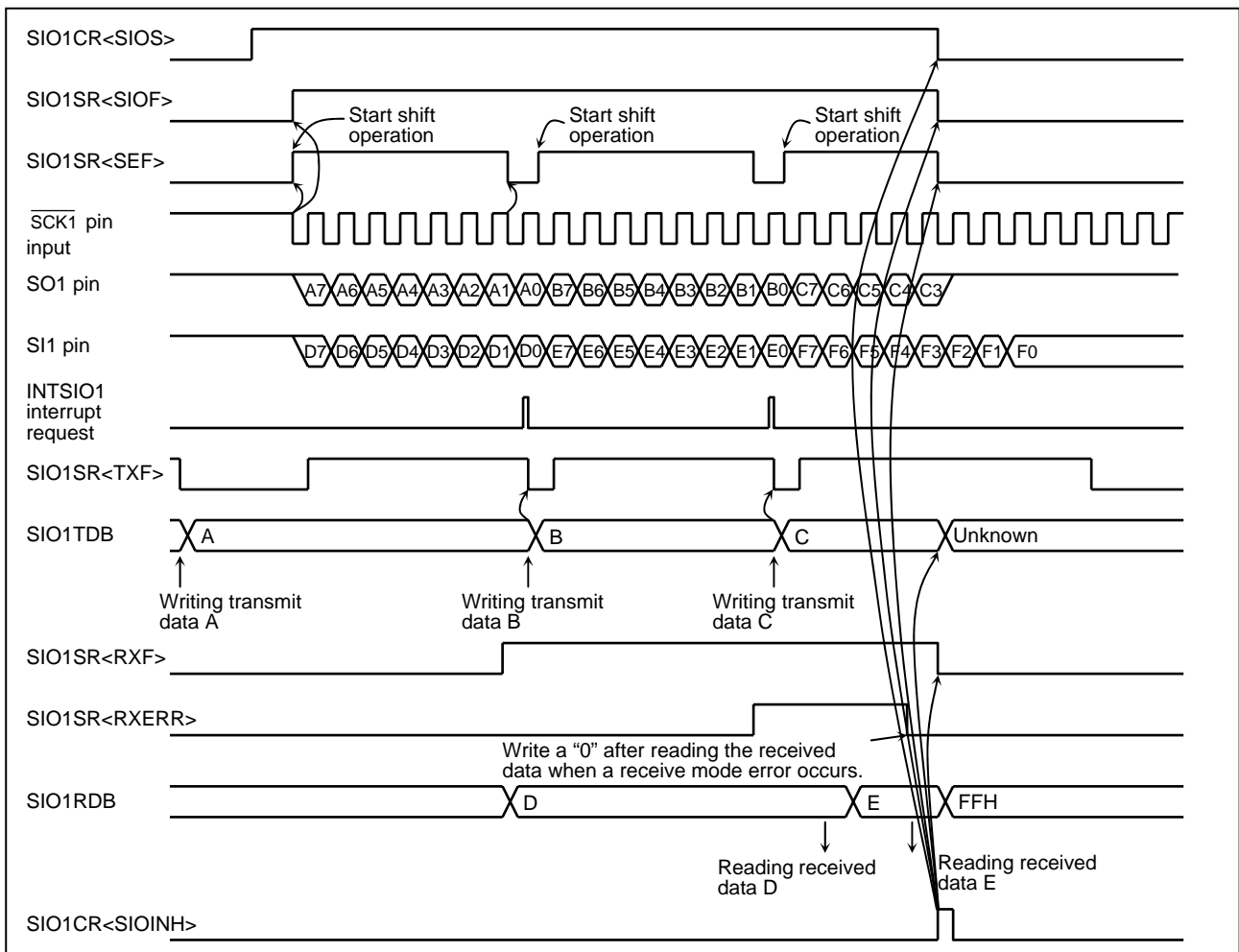
Figure 2.13.19 Example of Transmit/Receive (Receive) Error Processing

Note: If receive error is not corrected, an interrupt request does not generate after the error occurs.
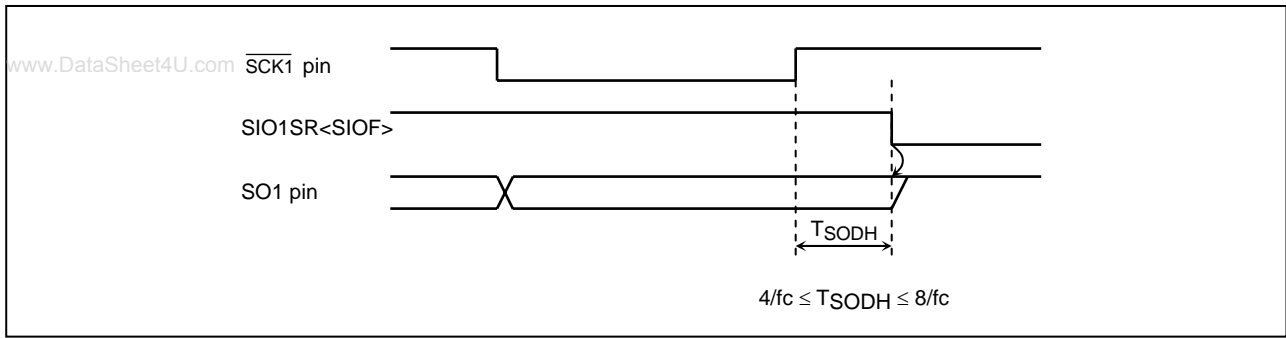
Figure 2.13.20 Hold Time of the End of Transmit/Receive Mode

2.14 Key-On Wake-Up (KWU)

In the TMP86FM48, the STOP mode must be released by not only P20 ($\overline{\text{INT5}} / \overline{\text{STOP}}$) pin but also P64 to P67 pins.

When the STOP mode is released by P64 to P67 pins, the P20 ($\overline{\text{INT5}} / \overline{\text{STOP}}$) pin needs to be used.

2.14.1 Configuration
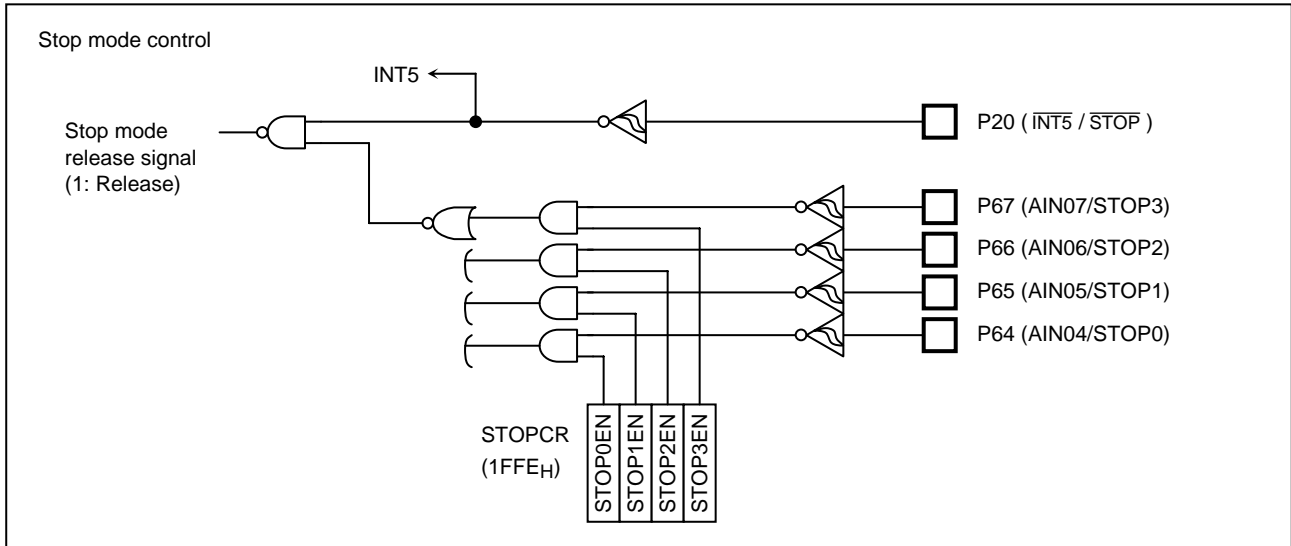
Figure 2.14.1 Key-On Wake-Up Circuit

Note1: $\overline{\text{STOP}}$ pin doesn't have the control register such as STOPCR, so when STOP mode is relesed by STOPx (x: 0 to 3), $\overline{\text{STOP}}$ pin should be used as STOP function.

2.14.2 Control

www.DataSheet4U.c P64 to P67 (STOP0 to STOP3) pin can controlled by key-on wake-up control register (STOPCR). It can be configured as enable/disable in 1-bit unit.

STOP mode can be entered by setting up the system control register1 (SYSCR1), and can be exited by detecting low level of STOP0 to STOP3 pins, which are enabled by STOPCR, for releasing STOP mode (Note 1). Also, because each level of the STOP0 to STOP3 can be confirmed by reading P6DR, check all STOP0 to STOP3 pins that is enabled by STOPCR before the STOP mode is started (Note 2, 3).

Note 1: When the STOP mode is used by edge-sensitive mode (SYSCR1<RELM> = "0"), all bit of STOPCR (STOP3EN to STOP0EN) should be cleared to "0".

Note 2: When the $\overline{STOP}$ pin input is high or STOP0 to STOP3 pin input which is enabled by STOPCR is low, executing an instruction which starts STOP mode will not place in STOP mode but instead will immediately start the release sequence (Warm-up).

Note 3: When confirms the level of STOP0 to STOP3 pin which is enabled by STOPCR, the corresponding bit of P6CR1 should be cleared to "0" before reading P6DR.

Table 2.14.1 Input Edge (Level) of Stop Mode Release

Terminal name	As both terminal	SYSCR1<RELM> = "1"	SYSCR1<RELM> = "0"
		Release edge (Level)	
STOP	P20/$\overline{INT5}$	"H" level (Note2)	Rising edge
STOP0	P64/AIN04	"L" level	Do not use key on
STOP1	P65/AIN05	(Note 2)	wake up function
STOP2	P66/AIN06		(Note 1)
STOP3	P67/AIN07		

Key-On Wake-Up Control Register

STOPCR	7	6	5	4	3	2	1	0	
(1FFE$_H$)	STOP0EN	STOP1EN	STOP2EN	STOP3EN	–	–	–	–	(Initial value: 0000 ****)

			0: Disable	
STOP0EN	Stop mode released by P64 port		1: Enable	
STOP1EN	Stop mode released by P65 port		0: Disable	Write
			1: Enable	only
STOP2EN	Stop mode released by P66 port		0: Disable	
			1: Enable	
STOP3EN	Stop mode released by P67 port		0: Disable	
			1: Enable	

Figure 2.14.2 Key-On Wake-Up Control Register

2.15 10-Bit AD Converter (ADC)

The TMP86FM48 has a 10-bit successive approximation type AD converter.

2.15.1 Configuration

The circuit configuration of the 10-bit AD converter is shown in Figure 2.15.1.

It consists of control registers ADCCR1 and ADCCR2, registers ADCDR1 and ADCDR2, a DA converter, a sample-and-hold circuit, a comparator, and a successive comparison circuit.
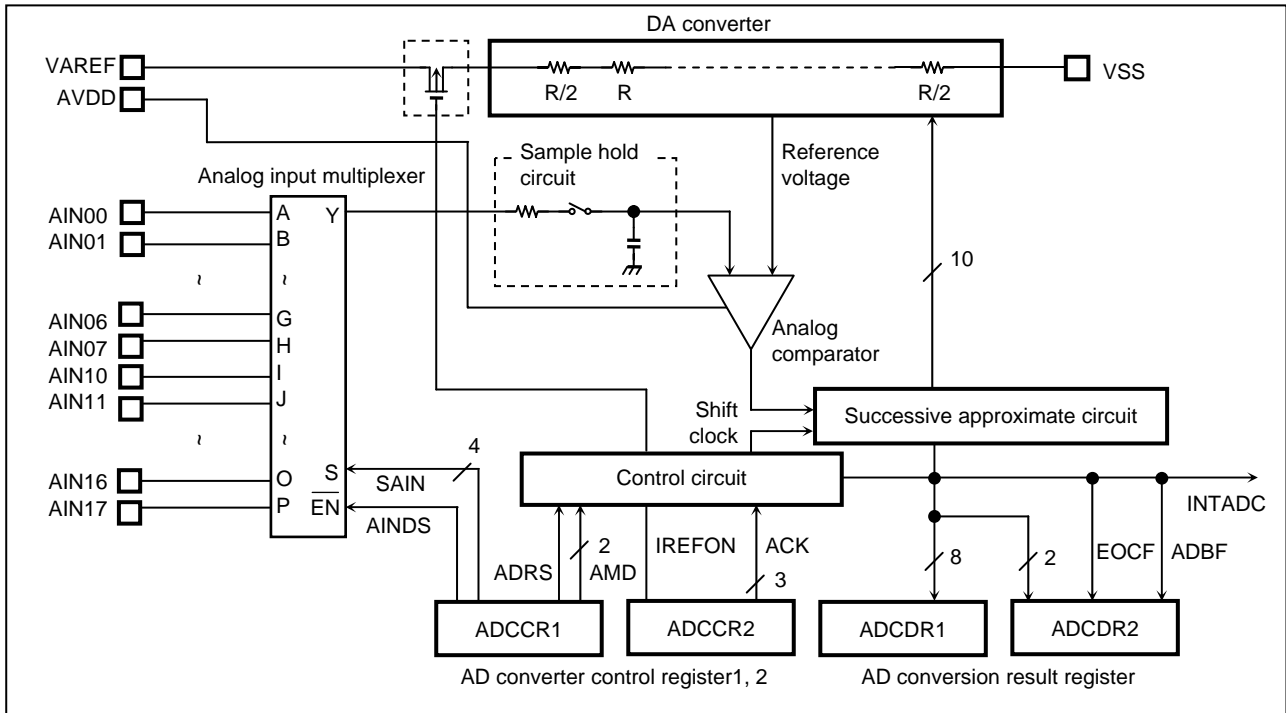
Figure 2.15.1 AD Converter (ADC)

2.15.2 Register Configuration

The AD converter consists of the following four registers:

- AD converter control register 1 (ADCCR1)
- AD converter control register 2 (ADCCR2)
- AD conversion result register 1/2 (ADCDR1/ADCDR2)

(1) AD converter control register 1 (ADCCR1)

This register selects the analog channels and operation mode (Software start or repeat) in which to perform AD conversion and controls the AD converter as it starts operating.

(2) AD converter control register 2 (ADCCR2)

This register selects the AD conversion time and controls the connection of the DA converter (Ladder resistor network).

(3) AD conversion result register (ADCDR1)

This register is used to store the digital value (Bit9 to bit2) after being converted by the AD converter.

(4) AD conversion result register (ADCDR2)

This register is used to store the digital value (Bit1 and bit0) after being converted by the AD converter, and then this register is also used to monitor the operating status of the AD converter.

The AD converter control register configurations are shown in Figure 2.15.2 and Figure 2.15.3.

AD Converter Control Register 1

ADCCR1	7	6	5	4	3	2	1	0	
(000E_H)	ADRS	AMD		AINDS		SAIN			(Initial value: 0001 0000)

ADRS	AD conversion start	0: − 1: Start	R/W
AMD	AD operating mode	00: AD operation disable 01: Software start mode 10: Reserved 11: Repeat mode	
AINDS	Analog input control	0: Analog input enable 1: Analog input disable	
SAIN	Analog input channel select	0000: Selects AIN00 1000: Selects AIN10 0001: Selects AIN01 1001: Selects AIN11 0010: Selects AIN02 1010: Selects AIN12 0011: Selects AIN03 1011: Selects AIN13 0100: Selects AIN04 1100: Selects AIN14 0101: Selects AIN05 1101: Selects AIN15 0110: Selects AIN06 1110: Selects AIN16 0111: Selects AIN07 1111: Selects AIN17	

Note 1:   Select analog input when AD converter stops (ADCDR2<ADBF> = "0").

Note 2:   When the analog input is all use disabling, the AINDS should be set to "1".

Note 3:   During conversion, do not perform output instruction to maintain a precision for all of the pins. And port near to analog input, do not input intense signaling of change.

Note 4:   The ADRS is automatically cleared to "0" after starting conversion.

Note 5:   Do not set ADRS newly again during AD conversion. Before setting ADRS newly again, check ADCDR2<EOCF> to see that the conversion is completed or wait until the interrupt signal (INTADC) is generated (e.g., interrupt handling routine).

Note 6:   After STOP or SLOW mode are started, AD converter control register 1 (ADCCR1) is all initialized. Therefore, set the ADCCR1 newly again after exiting these modes.

AD Converter Control Register 2

ADCCR2	7	6	5	4	3	2	1	0	
(000F_H)			IREFON	"1"		ACK		"0"	(Initial value: **00 0000)

IREFON	DA converter (Ladder resistor) connection control	Inputting current to the ladder resistor 0: Connected only during AD conversion 1: Always connected						R/W
ACK	AD conversion time select	ACK	Conversion time	fc = 16 MHz	fc = 8 MHz	fc = 4 MHz	fc = 1 MHz	
		000	39/fc	−	−	−	39.0 μs	
		001	Reserved					
		010	78/fc	−	−	−	78.0 μs	
		011	156/fc	−	−	39.0 μs	156.0 μs	
		100	312/fc	−	39.0 μs	78.0 μs	−	
		101	624/fc	39.0 μs	78.0 μs	156.0 μs	−	
		110	1248/fc	78.0 μs	156.0 μs	−	−	
		111	Reserved					

Note 1:   Settings for "−" in the above table are inhibited.

Note 2:   Set conversion time by analog reference voltage (V_AREF) as follows.

$$V_{AREF} = 2.7 \text{ to } 3.6 \text{ V } (31.2 \text{ μ or more})$$
$$V_{AREF} = 1.8 \text{ to } 3.6 \text{ V } (124.8 \text{ μ or more})$$

Note 3:   Always set bit0 in ADCCR2 to "0" and set bit4 in ADCCR2 to "1".

Note 4:   When a read instruction for ADCCR2, bit6 to 7 in ADCCR2 read in as undefined data.

Note 5:   fc: High-frequency clock [Hz]

Note 6:   After STOP or SLOW mode are started, AD converter control register 2 (ADCCR2) is all initialized. Therefore, set the ADCCR2 newly again after exiting these modes.

Figure 2.15.2 AD Converter Control Register

AD Conversion Result Register

ADCDR1	7	6	5	4	3	2	1	0	
(0027$_H$)	AD09	AD08	AD07	AD06	AD05	AD04	AD03	AD02	(Initial value: 0000 0000)

ADCDR2	7	6	5	4	3	2	1	0	
(0026$_H$)	AD01	AD00	EOCF	ADBF					(Initial value: 0000 ****)

EOCF	AD conversion end flag	0: Before or during conversion 1: Conversion completed	Read only
ADBF	AD conversion busy flag	0: During stop of AD conversion 1: During AD conversion	

Note 1:   The EOCF is cleared to "0" when reading the ADCDR1.

Therefore, the AD conversion result should be read to ADCDR2 more first than ADCDR1.

Note 2:   ADBF is set to "1" when AD conversion starts and cleared to "0" when the AD conversion is finished. It also is cleared upon entering STOP or SLOW mode.

Note 3:   If a read instruction is executed for ADCDR2, read data of bit 3 to 0 are unstable.

Figure 2.15.3 AD Converter Result Register

2.15.3 AD Converter Operation

(1)   Set up the AD converter control register 1 (ADCCR1) as follows:

- Choose the channel to AD convert using AD input channel select (SAIN).

- Specify analog input enable for analog input control (AINDS).

- Specify AMD for the AD converter control operation mode (Software or repeat mode).

(2)   Set up the AD converter control register 2 (ADCCR2) as follows:

- Set the AD conversion time using AD conversion time (ACK). For details on how to set the conversion time, refer to Note 2 for AD converter control register 2.

- Choose IREFON for DA converter control.

(3)   After setting up (1) and (2) above, set AD conversion start (ADRS) of AD converter control register 1 (ADCCR1) to "1".

(4)   After an elapse of the specified AD conversion time, the AD converted value is stored in AD conversion result register 1 (ADCDR1), AD conversion result register (ADCDR2) and then the AD conversion end flag (EOCF) of AD conversion result register 2 (ADCDR2) is set to "1", upon which time AD conversion interrupt INTADC is generated.

(5)   EOCF is cleared to "0" by a read of the conversion result. However, if reconverted before a register read, although EOCF is cleared the previous conversion result is retained until the next conversion is completed.

2.15.4 AD Converter Operation Modes

There are following two AD converter operation modes:

- Software start: AD conversion is performed once by setting AMD to "01B" and ADRS to "1".
- Repeat mode: AD conversion is performed repeatedly by setting AMD to "11B" and ADRS to "1".

(1) Software start mode

After setting ADCCR1<AMD> to "01" (Software start mode), set ADCCR1<ADRS> to "1". AD conversion of the voltage at the analog input pin specified by ADCCR1<SAIN> is thereby started.

After completion of the AD conversion, the conversion result is stored in AD conversion result registers (ADCDR1, ADCDR2) and at the same time ADCDR2<EOCF> is set to "1", the AD conversion finished interrupt (INTADC) is generated.

ADCCR1<ADRS> is automatically cleared to "0" after AD conversion has started. Do not set ADCCR1<ADRS> newly again (Restart) during AD conversion. Before setting ADCCR1<ADRS> newly again, check ADCDR2<EOCF> to see that the conversion is completed or wait until the interrupt signal (INTADC) is generated (e.g., interrupt handling routine).
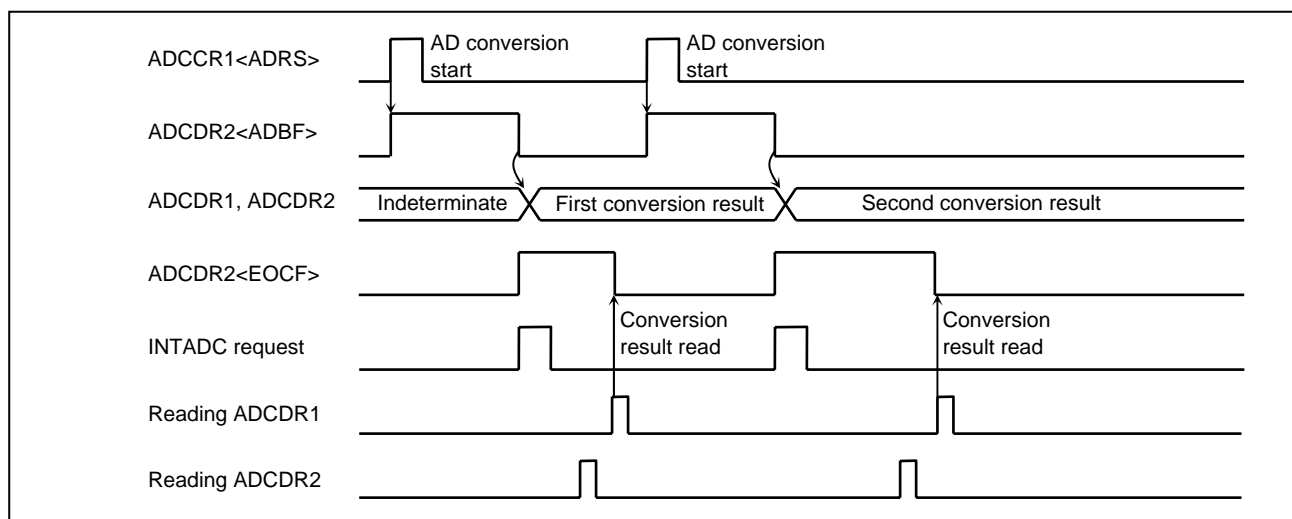
Figure 2.15.4 Operation in Software Start Mode

Example: After selecting the conversion time of 39.0 μs at 16 MHz and the analog input channel AIN3 pin, perform AD conversion once. After checking EOCF, read the converted value, store the lower 2 bits in address 009EH and store the upper 8 bits in address 009F$_H$ on RAM. The operation mode is software start mode.

```
            ; AIN SELECT
            LD      (P6CR1), 00000000B    ;  P6CR1 bit 3 = 0
            LD      (P6CR2), 00000000B    ;  P6CR2 bit 3 = 0
            LD      (ADCCR1), 00100011B   ;  Select AIN3
            LD      (ADCCR2), 11011010B   ;  Select conversion time (624/fc) and
                                             operation mode

            ; AD CONVERT START
            SET     (ADCCR1). 7           ;  ADRS = 1
SLOOP:      TEST    (ADCDR2). 5           ;  EOCF = 1 ?
            JRS     T, SLOOP
            ; RESULT DATA READ
            LD      A, (ADCDR2)
            LD      (9EH), A
            LD      A, (ADCDR1)
            LD      (9FH), A
```

(2) Repeat mode

AD conversion of the voltage at the analog input pin specified by ADCCR1<SAIN> is performed repeatedly. In this mode, AD conversion is started by setting ADCCR1<ADRS> to "1" after setting ADCCR1<AMD> to "11".

After completion of the AD conversion, the conversion result is stored in AD conversion result registers (ADCDR1, ADCDR2) and at the same time ADCDR2<EOCF> is set to "1", the AD conversion finished interrupt (INTADC) is generated.

In repeat mode, each time one AD conversion is completed, the next AD conversion is started. To stop AD conversion, set ADCCR1<AMD> to "00B" (Disable mode). The AD convert operation is stopped immediately. The converted value at this time is not stored in the AD conversion result register.
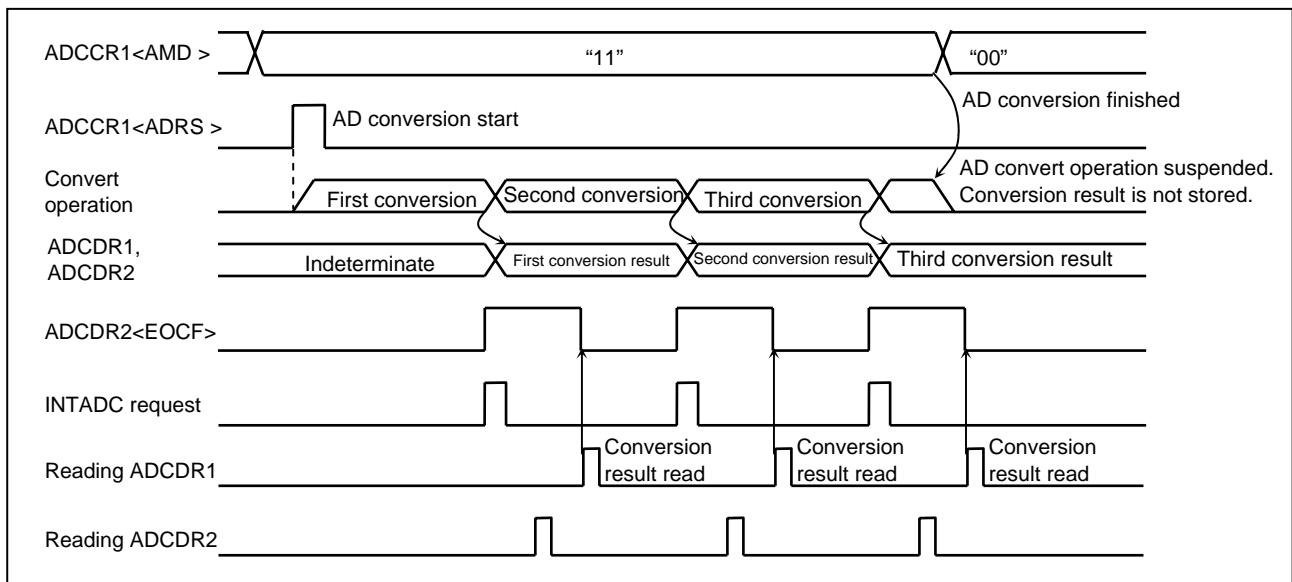
Figure 2.15.5 Operation in Repeat Mode

2.15.5 STOP and SLOW Modes during AD Conversion

When the STOP or SLOW mode is entered forcibly during AD conversion, the AD convert operation is suspended and the AD converter is initialized (ADCCR1 and ADCCR2 are initialized to initial value). Also, the conversion result is indeterminate. (Conversion results up to the previous operation are cleared, so be sure to read the conversion results before entering STOP or SLOW mode.) When released from STOP or SLOW mode, AD conversion is not automatically restarted. Therefore, when the AD converter is used again, it is necessary to restart AD conversion (Set ADCCR1<ADRS> to "1"). Note that since the analog reference voltage is automatically disconnected, there is no possibility of current flowing into the analog reference voltage.

2.15.6 Analog Input Voltage and AD Conversion Result

The analog input voltage is corresponded to the 10-bit digital value converted by the AD as shown in Figure 2.15.6.
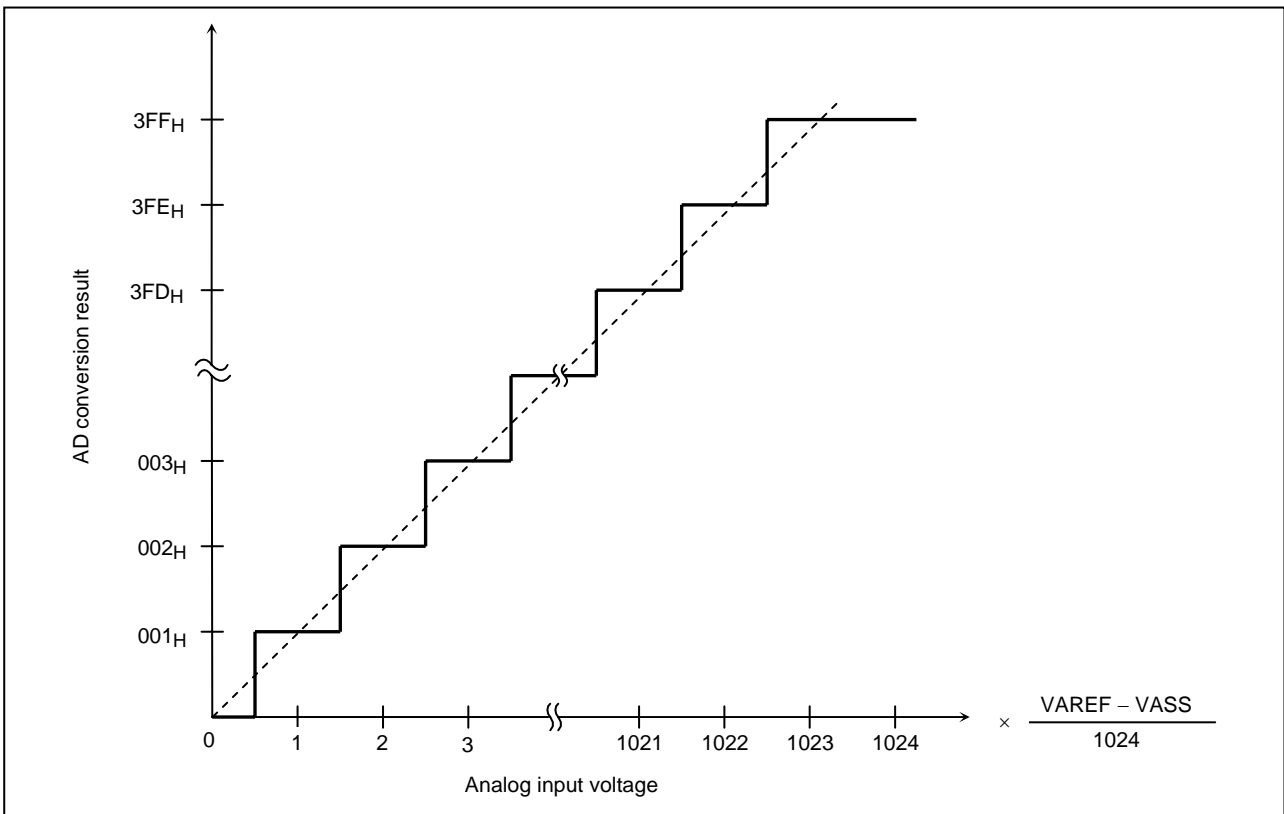
Figure 2.15.6 Analog Input Voltage and AD Conversion Result (typ.)

2.15.7 Precautions about AD Converter

(1) Analog input pin voltage range

Make sure the analog input pins (AIN00 to AIN17) are used at voltages within VSS below VAREF. If any voltage outside this range is applied to one of the analog input pins, the converted value on that pin becomes uncertain. The other analog input pins also are affected by that.

(2) Analog input shared pins

The analog input pins (AIN00 to AIN17) are shared with input/output ports. When using any of the analog inputs to execute AD conversion, do not execute input/output instructions for all other ports. This is necessary to prevent the accuracy of AD conversion from degrading. Not only these analog input shared pins, some other pins may also be affected by noise arising from input/output to and from adjacent pins.

(3) Noise countermeasure

The internal equivalent circuit of the analog input pins is shown in Figure 2.15.7. The higher the output impedance of the analog input source, more easily they are susceptible to noise. Therefore, make sure the output impedance of the signal source in your design is 5 kΩ or less. Toshiba also recommends attaching a capacitor external to the chip.
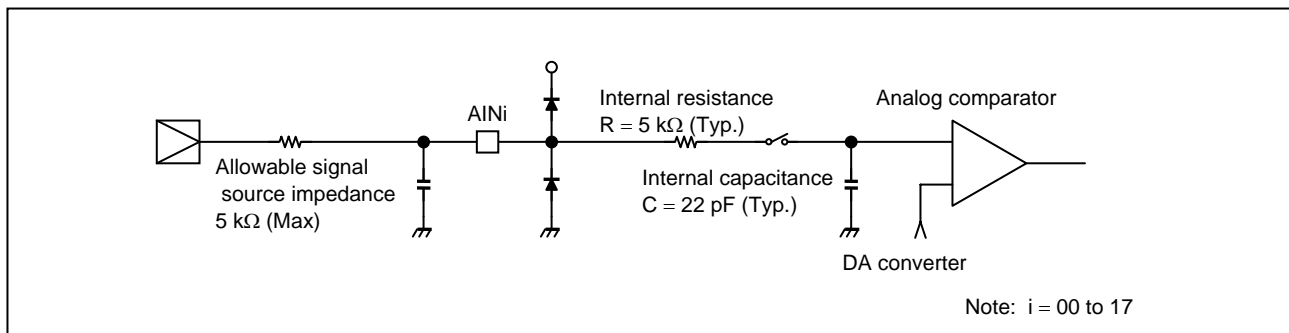
Figure 2.15.7 Analog Input Equivalent Circuit and Example of Input Pin Processing

2.16   FLASH Memory

2.16.1   Outline

The TMP86FM48 incorporates 32768 bytes of FLASH memory (Address 8000H to FFFFH). Of these bytes, 512 bytes (Address 8000H to 81FFH) can be used as data memory. When these 512 bytes (Address 8000H to 81FFH) are used as data memory, the 32256 bytes (Address 8200H to FFFFH) can be used as program memory. The writing to FLASH memory is controlled by FLASH control register (EEPCR), FLASH status register (EEPSR) and FLASH write emulate time control register (EEPEVA).

The FLASH memory of the TMP86FM48 features:

- The FLASH memory is constructed of 512 pages FLASH memory and one page size is 64 bytes (512 pages × 64 bytes = 32768 bytes).

- The TMP86FM48 incorporates a 64-byte temporary data buffer. The data written to FLASH memory is temporarily stored in this data buffer. After 64 bytes data have been written to the temporary data buffer, the writing to FLASH memory automatically starts by page writing (The 64 bytes data are written to specified page of FLASH simultaneously). At the same time, page-by-page erasing occurs automatically. So, it is unnecessary to erase individual pages in advance.

- The FLASH control circuit incorporates an oscillator dedicated to the FLASH. So FLASH writing time is independent of the system clock frequency (fc). In addition, because an FLASH control circuit controls writing time for each FLASH memory cell, the writing time varies in each page (Typically 4 ms per page).

- Controlling the power for the FLASH control circuit (regulator and voltage step-up circuit) achieves low power consumption if the FLASH is not in use (Example: When the program is executed in RAM area).

2.16.2   Conditions for Accessing the FLASH Areas

The conditions for accessing the FLASH areas vary depending on each operation mode. The following tables shows FLASH are access conditions.

Table 2.16.1   FLASH Area Access Conditions

	Area	Operation Mode	
		MCU mode (Note 1)	Serial PROM mode (Note 2)
Data Memory	8000H to 81FFH	Write/read/fetch [Note3] supported	
Program Memory	8200H to FFFFH	Read/fetch only	Write/read/fetch supported

Note 1: "MCU mode" shows NORMAL1/2 and SLOW1/2 modes.

Note 2: "Serial PROM mode" shows the FLASH controlling mode. For details, refer to "2.19 Serial PROM mode".

Note 3: "Fetch" means reading operation of FLASH data as an instruction by CPU.

2.16.3 Differences among Product Series

The specifications of the FLASH product (TMP86FM48) are different from those of the emulation chip (TMP86C948) as listed below. See 2.17.2 "Control" for explanations about the control registers.

	FLASH Product (TMP86FM48)	Emulation Chip (TMP86C948)
Rewriting the EEPCR register<EEPMD, EEPRS, MNPWDW>	It is possible to rewrite the EEPCR register only when the program execution area in use is RAM/BOOT-ROM.	
		In the debugger memory window, it is impossible to rewrite the EEPCR register.
Accessing the EEPEVA register	It is possible only to write- and read-access the EEPEVA register. The writing to this register does not affect the function.	The time required to emulate FLASH writing is put under control.
FLASH write time (The emulation chip is written to emulation memory instead of FLASH)	Typically 4 ms (Independent of the system clock)	The FLASH write time is set up using the EEPEVA register (Dependent on the system clock).
Executing a read instruction/fetch to the 8000H to FFFFH area when EEPSR<BFBUSY> = "1"	If EEPSR<BFBUSY> = "1", executing a read instruction/fetch to the FLASH area causes FFH to be read regardless of what the current ROM data is. Fetching FFH results in a software interrupt occurring.	
		The debugger memory window always displays ROM data.
Executing a write instruction to the 8000H to 81FFH area when EEPCR<EEPMD> = "0011", EEPSR<EWUPEN> = "1" and EEPSR<BFBUSY> = "0". — MCU mode / Serial PROM mode	The EEPSR<BFBUSY> is set to "1" (Write enabled).	
Executing a write instruction to the 8200H to FFFFH area when EEPCR<EEPMD> = "0011", EEPSR<EWUPEN> = "1" and EEPSR<BFBUSY> = "0". — MCU mode	The EEPSR<BFBUSY> stays at "0" (Write disabled).	In the debugger memory window, it is possible to rewrite the 8200H to FFFFH area (The EEPSR<BFBUSY> remains unchanged).
Serial PROM mode	The EEPSR<BFBUSY> is set to "1" (write enabled).	
Data memory (8000H to 81FFH)	512 bytes of FLASH are included in the 8000H to 81FFH area.	512 bytes of emulation memory are included in the 8000H to 81FFH area. (Turning off the power for the emulation chip erases data in the emulation memory.)
BOOT-ROM	2 Kbytes are included in the 3800H to 3FFFH area.	
Operating voltage	VDD = 1.8 to 3.6 V	VDD = 1.8 to 3.3 V

2.16.4 FLASH Memory Configuration

64 consecutive bytes in the FLASH area are treated as one group, which is defined as a page. The TMP86FM48 incorporates a one-page temporary data buffer. Writing data to FLASH is temporarily stored in this 64-byte data buffer. After 64 bytes data have been written to the temporary data buffer, these data are written to specified page of FLASH at a time. However, data can be read from any address byte by byte.

2.16.4.1 Page Configuration

The FLASH area has a page configuration of 64 bytes/page as shown below. The total number of bytes in it is 512 pages × 64 bytes (= 32768 bytes). The writeable area is 8000H to FFFFH in Serial PROM mode.

Note: The program memory (8200H to FFFFH) can be written only in the Serial PROM mode. For details of the Serial PROM mode, refer to "2.19 Serial PROM mode".

Figure 2.16.1 Page Configuration

2.17 Data Memory of FLASH(address 8000H to 81FFH)

The TMP86FM48 incorporates 512 bytes (8000H to 81FFH) of data memory of FLASH, which features:

- In the MCU mode, user-created programs can rewrite the data memory of FLASH in page (64 bytes) units. (It can be used to save application last keys and preset data.)

- In the serial PROM mode, it is possible to perform serial writing to the data memory of FLASH in the same manner as for the program memory. So, initial values can be factory-set in the data memory of FLASH.

- Using support programs incorporated in the BOOT-ROM makes it easy to write to the FLASH.
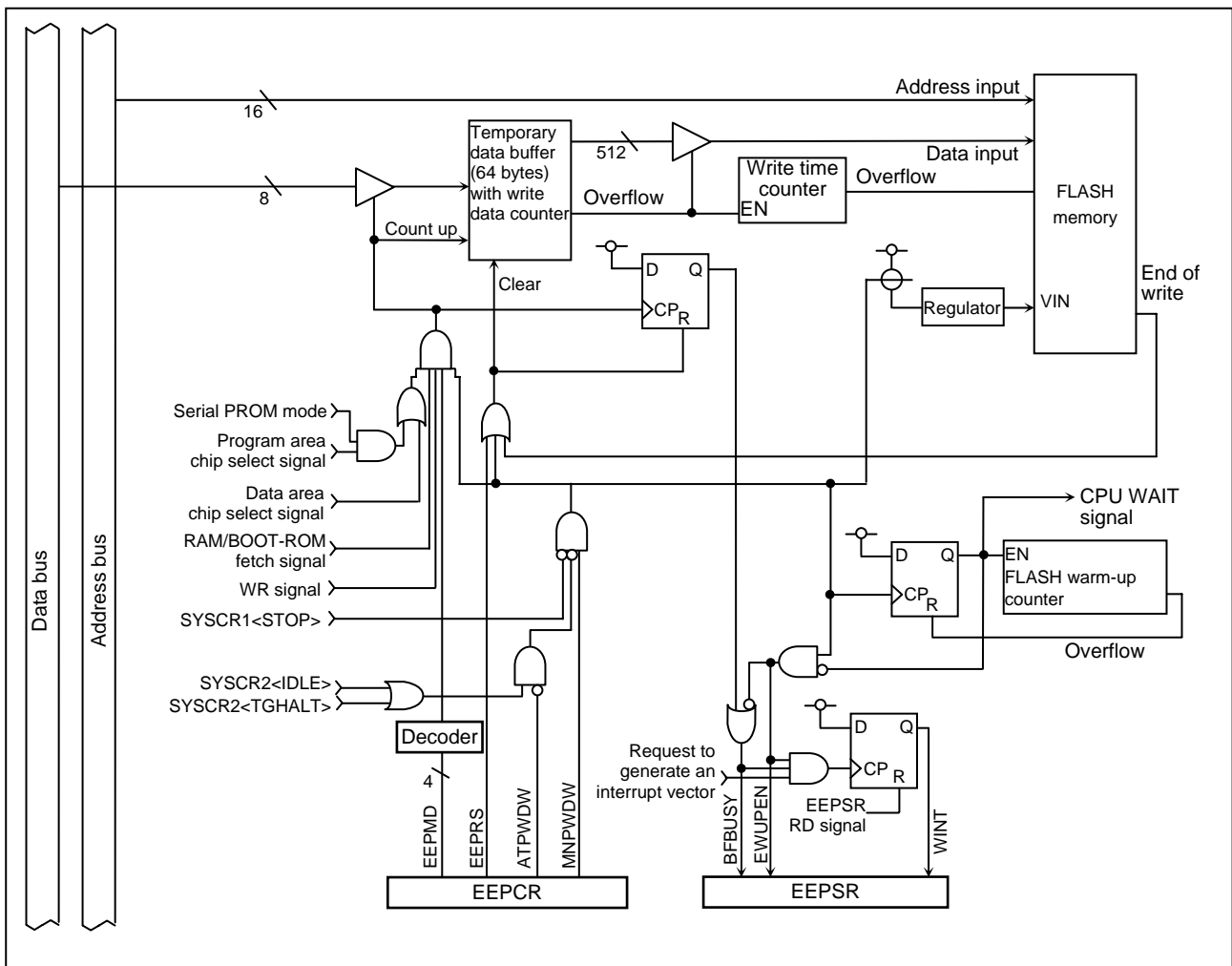
2.17.1 Configuration

Figure 2.17.1 Data Memory

2.17.2 Control

The FLASH is controlled by FLASH control register (EEPCR), FLASH status register (EEPSR) and FLASH write emulate time control register (EEPEVA).
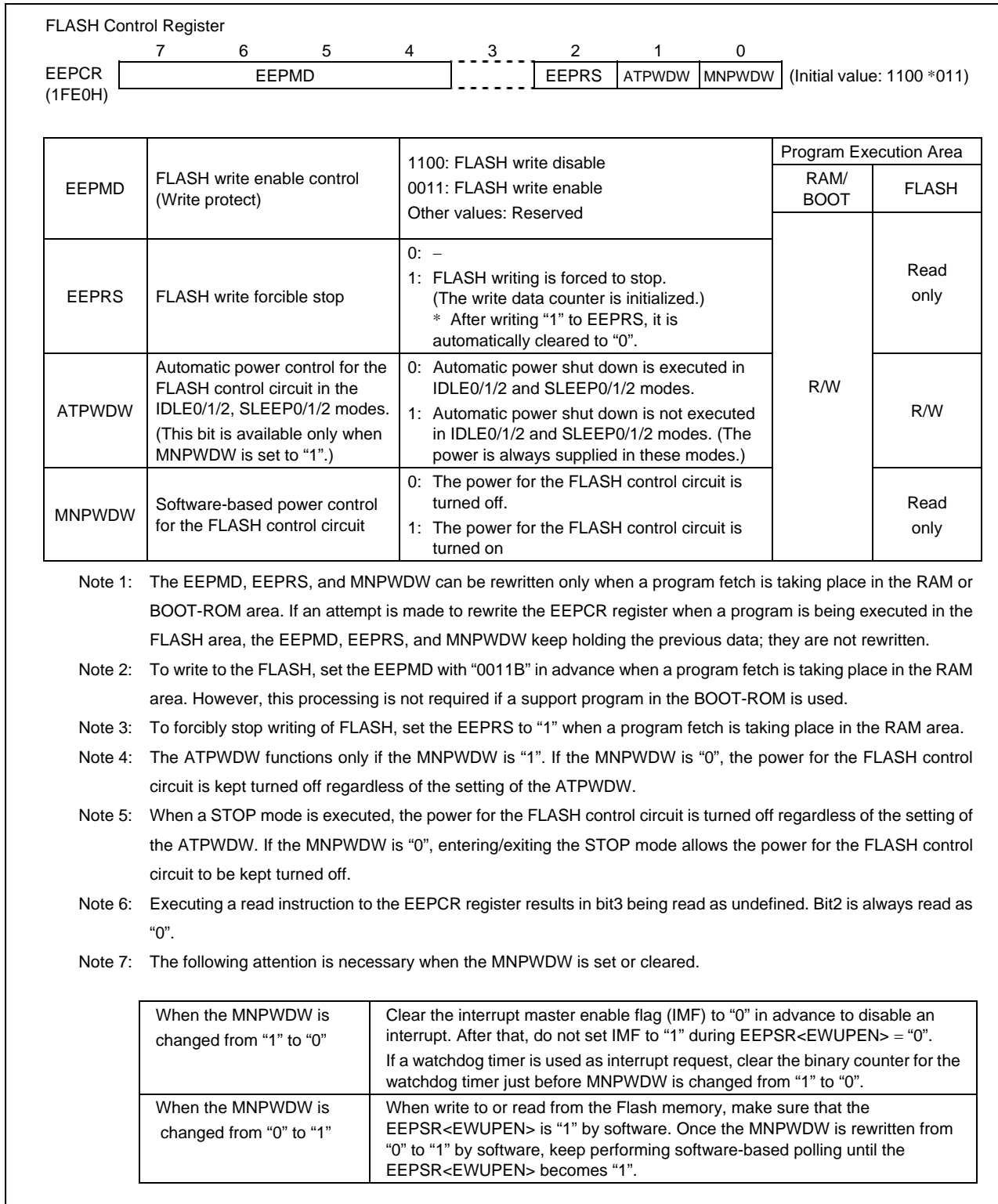
FLASH Control Register

	7	6	5	4	3	2	1	0	
EEPCR (1FE0H)		EEPMD				EEPRS	ATPWDW	MNPWDW	(Initial value: 1100 *011)

			Program Execution Area	
			RAM/ BOOT	FLASH
EEPMD	FLASH write enable control (Write protect)	1100: FLASH write disable 0011: FLASH write enable Other values: Reserved	R/W	Read only
EEPRS	FLASH write forcible stop	0: − 1: FLASH writing is forced to stop. (The write data counter is initialized.) * After writing "1" to EEPRS, it is automatically cleared to "0".		Read only
ATPWDW	Automatic power control for the FLASH control circuit in the IDLE0/1/2, SLEEP0/1/2 modes. (This bit is available only when MNPWDW is set to "1".)	0: Automatic power shut down is executed in IDLE0/1/2 and SLEEP0/1/2 modes. 1: Automatic power shut down is not executed in IDLE0/1/2 and SLEEP0/1/2 modes. (The power is always supplied in these modes.)		R/W
MNPWDW	Software-based power control for the FLASH control circuit	0: The power for the FLASH control circuit is turned off. 1: The power for the FLASH control circuit is turned on		Read only

Note 1: The EEPMD, EEPRS, and MNPWDW can be rewritten only when a program fetch is taking place in the RAM or BOOT-ROM area. If an attempt is made to rewrite the EEPCR register when a program is being executed in the FLASH area, the EEPMD, EEPRS, and MNPWDW keep holding the previous data; they are not rewritten.

Note 2: To write to the FLASH, set the EEPMD with "0011B" in advance when a program fetch is taking place in the RAM area. However, this processing is not required if a support program in the BOOT-ROM is used.

Note 3: To forcibly stop writing of FLASH, set the EEPRS to "1" when a program fetch is taking place in the RAM area.

Note 4: The ATPWDW functions only if the MNPWDW is "1". If the MNPWDW is "0", the power for the FLASH control circuit is kept turned off regardless of the setting of the ATPWDW.

Note 5: When a STOP mode is executed, the power for the FLASH control circuit is turned off regardless of the setting of the ATPWDW. If the MNPWDW is "0", entering/exiting the STOP mode allows the power for the FLASH control circuit to be kept turned off.

Note 6: Executing a read instruction to the EEPCR register results in bit3 being read as undefined. Bit2 is always read as "0".

Note 7: The following attention is necessary when the MNPWDW is set or cleared.

When the MNPWDW is changed from "1" to "0"	Clear the interrupt master enable flag (IMF) to "0" in advance to disable an interrupt. After that, do not set IMF to "1" during EEPSR<EWUPEN> = "0". If a watchdog timer is used as interrupt request, clear the binary counter for the watchdog timer just before MNPWDW is changed from "1" to "0".
When the MNPWDW is changed from "0" to "1"	When write to or read from the Flash memory, make sure that the EEPSR<EWUPEN> is "1" by software. Once the MNPWDW is rewritten from "0" to "1" by software, keep performing software-based polling until the EEPSR<EWUPEN> becomes "1".

Figure 2.17.2 FLASH Control Register

FLASH status register

	7	6	5	4	3	2	1	0	
EEPSR (1FE1H)						WINT	EWUPEN	BFBUSY	(Initial value: **** *010)

			0: Not detected 1: Detected (Interrupt occurred) * WINT is automatically cleared to "0" when read instruction is executed to EEPSR.			Read only
WINT	Interrupt detection during a write to the FLASH					
EWUPEN	FLASH control circuit status monitor	Control circuit status	Operating (power on)		Halt (power off) or warm-up	
		FLASH status	Temporary data buffer empty	Writing	Disable	
			1	1	0	
BFBUSY	FLASH write busy flag		0	1	1	

Note 1: If a nonmaskable interrupt occurs during a write to the FLASH, the WINT is set to "1" and the writing is discontinued, and then warm-up (CPU wait) for the control circuit of Flash memory is executed. (The write data counter is initialized.) If WINT = "1" is detected in the nonmaskable interrupt service routine, a write is not completed successfully. So, it is necessary to try a write again. The content of the page to which a write is taking place may be changed to an unexpected value depending on the timing when the WINT becomes "1".

Note 2: Even if a nonmaskable interrupt occurs during an FLASH warm-up, the CPU stays at a halt until the warm-up is finished.

Note 3: The WINT is automatically cleared to "0" when a read instruction is executed to the EEPSR register.

Note 4: When MNPWDW is changed from "0" to "1", EWUPEN becomes "1" after taking 2^{10}/fc [s] (if SYSCK = "0") or 2^3/fs [s] (if SYSCK = "1"). Before accessing the FLASH, make sure that the EWUPEN is "1" in the RAM area.

Note 5: If the BFBUSY is "1", executing a read instruction or fetch to the FLASH area causes FFH to be read. Fetching FFH results in a software interrupt occurring.

Figure 2.17.3 FLASH Status Register

FLASH Write Emulation Time Control Register (Setting of this register functions only in emulation chip (TMP86C948).)

	7	6	5	4	3	2	1	0	
EEPEVA (1FE2H)						EEPSUCR			(Initial value: **** *000)

			Emulation Chip (TMP86C948)		FLASH Product (TMP86FM48)	
			NORMAL1/2 IDLE1/2 mode	SLOW1/2 SLEEP1/2 mode	All operation modes	
EEPSUCR	Controlling the FLASH write emulation time [s] for emulation chip	000 001 010 011 100 101 110 111	2^{16}/fc 2^{15}/fc 2^{14}/fc 2^{13}/fc 2^{12}/fc Reserved Reserved Reserved	2^{7}/fs 2^{6}/fs 2^{5}/fs 2^{4}/fs 2^{3}/fs Reserved Reserved Reserved	Typ. 4 ms (regardless of register settings and the system clock)	R/W

Note 1: Only in the emulation chip, the EEPSUCR functions. In the FLASH product, it is possible only to write- and read-access the register. It does not actually function. Because the FLASH product incorporates a dedicated oscillator, its write time is independent of the system clock.

Note 2: Executing a read instruction to the EEPEVA register results in bit7 to 3 being read as undefined.

Note 3: The following table lists the write emulation time specified by the setting of the EEPSUCR. Select an appropriate value according to the operating frequency used. The shading indicates recommended settings.

	EEPSUCR Setting	NORMAL1/2 Mode					SLOW1/2 Mode
		fc = 16 MHz	fc = 8 MHz	fc = 4 MHz	fc = 2 MHz	fc = 1 MHz	fs = 32.768 kHz
Write Time [ms]	000	4.10	8.19	16.38	32.77	65.54	3.91
	001	2.05	4.10	8.19	16.38	32.77	1.95
	010	1.02	2.05	4.10	8.19	16.38	0.98
	011	0.51	1.02	2.05	4.10	8.19	0.49
	100	0.26	0.51	1.02	2.05	4.10	0.24

Figure 2.17.4 FLASH Control Register and FLASH Status Register

2.17.3   FLASH Write Enable Control (EEPCR<EEPMD>)

In the FLASH product, the control register can be used to disable a write to the FLASH (Write protect) in order to prevent a write to the FLASH from occurring by mistake because of a program error or microcontroller malfunction. To enable a write to the FLASH, set the EEPCR<EEPMD> with 0011B. To disable a write to the FLASH, set the EEPCR<EEPMD> with 1100B. A reset initializes the EEPCR<EEPMD> to 1100B to disable a write to the FLASH. Usually, set the EEPCR<EEPMD> with 1100B, except when it is necessary to write to the FLASH.

Note: The EEPCR<EEPMD> can be rewritten only when a program is being executed in the RAM area. Executing a write instruction to the EEPCR<EEPMD> in the FLASH area does not change its setting.

2.17.4 FLASH Write Forcible Stop (EEPCR<EEPRS>)

To forcibly stop a write to the FLASH, set the EEPCR<EEPRS> to "1". Setting the EEPCR<EEPRS> to "1" initializes the write data counter of data buffer and forcibly stops a write, and then a warm-up (CPU wait) for the control circuit of Flash memory is executed. After warm-up period, the EEPSR<BFBUSY> is cleared to "0". The warm-up period is 2^{10}/fc (SYSCK = "0") or 2^3/fs (SYSCK = "1"). After this, if writing to FLASH starts again, data is stored as the first byte of the temporary data buffer and sets the EEPSR<BFBUSY> to "1". Therefore, it is necessary to write 64 bytes data to the temporary data buffer.

After 1 to 63 bytes are saved to the temporary data buffer, if the EEPCR<EEPRS> is set to "1" the specified page of FLASH is not written. (It keeps previous data.)

Note 1: After 64 bytes are written to the temporary data buffer, the setting the EEPCR<EEPRS> to "1" may cause the writing the page of FLASH to an unexpected value.

Note 2: The EEPCR<EEPRS> can be rewritten only when a program is being executed in the RAM area. In the FLASH area, executing a write instruction to the EEPCR<EEPRS> does not affect its setting.

Note 3: During the warm-up period for Flash memory (CPU wait), the peripheral circuits continue operating, but the CPU stays at a halt until the warm-up is finished. Even if an interrupt latch is set to "1" by generating of interrupt request, an interrupt sequence doesn't start till the end of warm-up. If interrupts occur during a warm-up period with IMF = "1", the interrupt sequence which depends on interrupt priority will start after warm-up period.

Note 4: When the EEPCR<EEPRS> is set to "1" with EEPSR<BFBUSY> = "0", a warm-up is not executed.

Note 5: If executed a write or read instruction to the Flash area immediately after setting EEPCR<EEPRS>, insert one or more machine cycle instructions after setting EEPCR<EEPRS>.

Example: Reads the Flash memory data immediately after setting EEPCR<EEPRS> to "1"

```
        LD      HL,8000H
        LD      (EEPCR),3FH        ; Set EEPCR<EEPRS> to "1"
        NOP                        ; NOP
                                     (Do not execute write or read instruction immediately
                                     after setting EEPCR<EEPRS>.)
        LD      A,(HL)             ; Reads the data of address 8000H
                                     (Write or read instruction to the Flash memory)
```
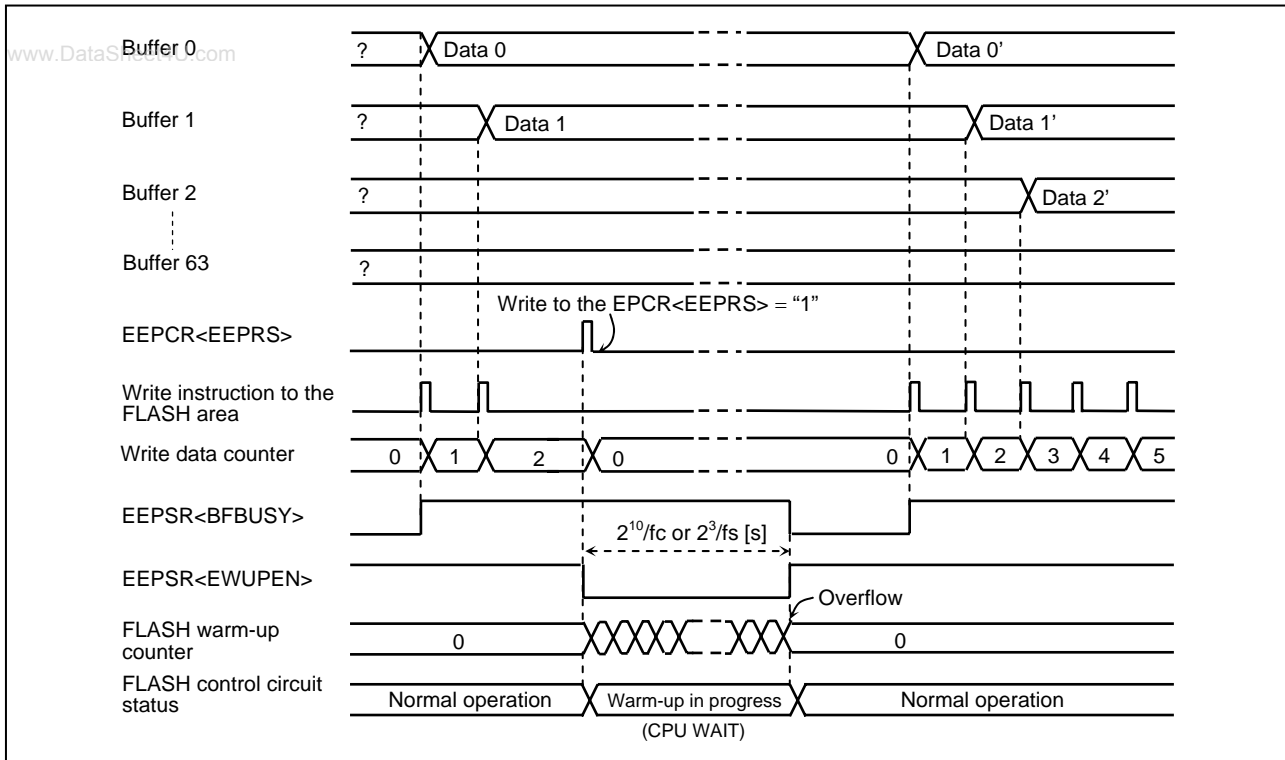
Figure 2.17.5 Write Data Counter Initialization and Write Forcible Stop

2.17.5 Power Control for the FLASH Control Circuit

For the FLASH product, it is possible to turn off the power for FLASH control circuit (such as a regulator) to suppress power consumption if the FLASH area is not accessed. For the emulation chip (TMP86C948), the register setting and the CPU wait functions behave in the same manner as for the FLASH product to maintain compatibility; however, power consumption is not suppressed.

The EEPCR<MNPWDW> and EEPCR<ATPWDW> are used to control the power for the FLASH control circuit. If the power for the FLASH control circuit is turned off according to the setting of these registers, starting to use the circuits again needs to allow warm-up time for the power supply.

Table 2.17.1 Power Supply Warm-up Time (CPU wait) for the FLASH Control Circuit

| NORMAL1/2 IDLE0/1/2 Mode	SLOW1/2 SLEEP0/1/2 Mode	STOP Mode (when EEPCR<MNPWDW> = "1")	
		To Return to a NORMAL Mode	To Return to a SLOW Mode
2^{10}/fc [s] (64 μs @16 MHz)	2^3/fs [s] (244 μs @32.768 kHz)	STOP warm-up time + 2^{10}/fc [s]	STOP warm-up time + 2^3/fs [s]

2.17.5.1 Software-based Power Control for the FLASH Control Circuit (EEPCR<MNPWDW>)

The EEPCR<MNPWDW> is a software-based power control bit for the FLASH control circuit. When a program is being executed in the RAM area, setting this bit enables software-based power control. Clearing the EEPCR<MNPWDW> to "0" immediately turns off the power for the FLASH control circuit. Once the EEPCR<MNPWDW> is switched from "0" to "1", before attempting a read or fetch from the FLASH area, it is necessary to insert a warm up period by software until the power supply is stabilized. In this case, because the CPU wait is not executed, any other instructions except accessing to Flash (write or read) are available. When MNPWDW is changed from "0" to "1", EWUPEN becomes "1" after taking 2^{10}/fc [s] (SYSCK = "0") or 2^3/fs [s] (SYSCK = "1"). Usually software-based polling should be performed until the EEPSR<EWUPEN> becomes "1". An example of setting is given below.

(1) Example of controlling the EEPCR<MNPWDW>

1. Transfer a program for controlling the EEPCR<MNPWDW> to the RAM area.
2. Release an address trap in the RAM area (Setup the WDTCR1 and WDTCR2 registers).
3. Jump to the control program transferred to the RAM area.
4. Clear the interrupt master enable flag (IMF ← "0").
5. Clear the binary counter if the watchdog timer is in use.
6. To turn off the power for the FLASH control circuit, clear the EEPCR<MNPWDW> to "0".
7. Perform CPU processing as required.
8. To access the FLASH area again, set the EEPCR<MNPWDW> to "1".
9. Keep program polling until the EEPSR<EWUPEN> becomes "1".
   (Upon completion of an FLASH warming-up, the EEPSR<EWUPEN> is set to "1". It takes 2^{10}/fc (SYSCK = "0") or 2^3/fs (SYSCK = "1") until EWUPEN becomes "1".)

This procedure enables the FLASH area to be accessed.

If the EEPCR<MNPWDW> is "1", entering a STOP mode forcibly turns off the power for the FLASH control circuit. When the STOP mode is released, a STOP mode oscillation warm-up is carried out, and then the CPU wait period (Warm-up for stabilizing of FLASH power supply circuit) is automatically performed. If the EEPCR<MNPWDW> is "0", entering/exiting the STOP mode keeps the power for the FLASH control circuit turned off.

Note 1: If the EEPSR<EWUPEN> is "0", do not access (Fetch, read, or write) the FLASH area. Executing a read instruction or fetch to the FLASH area causes FFH to be read. Fetching FFH results in a software interrupt occurring.

Note 2: To clear the EEPCR<MNPWDW> to "0", clear the interrupt master enable flag (IMF) to "0" in advance to disable an interrupt. After that, do not set IMF to "1" during EEPSR<EWUPEN> = "0".

Note 3: If the EEPCR<MNPWDW> is "0", generating a nonmaskable interrupt automatically rewrites the MNPWDW to "1" to warm up the FLASH control circuit (CPU wait). That time, the peripheral circuits continue operating, but the CPU stays at a halt until the warm-up is finished.

Note 4: The EEPCR<MNPWDW> can be rewritten only when a program is being executed in the RAM area. In the FLASH area, executing a write instruction to the EEPCR<MNPWDW> does not affect its setting.

Note 5: If a watchdog timer is used as interrupt request, clear the binary counter for the watchdog timer just before MNPWDW is changed from "1" to "0".

Note 6: During the warm-up period with a software polling of EEPSR<EWUPEN>, if a nonmaskable interrupt occurs, the CPU stays at a halt until the warm-up is finished.
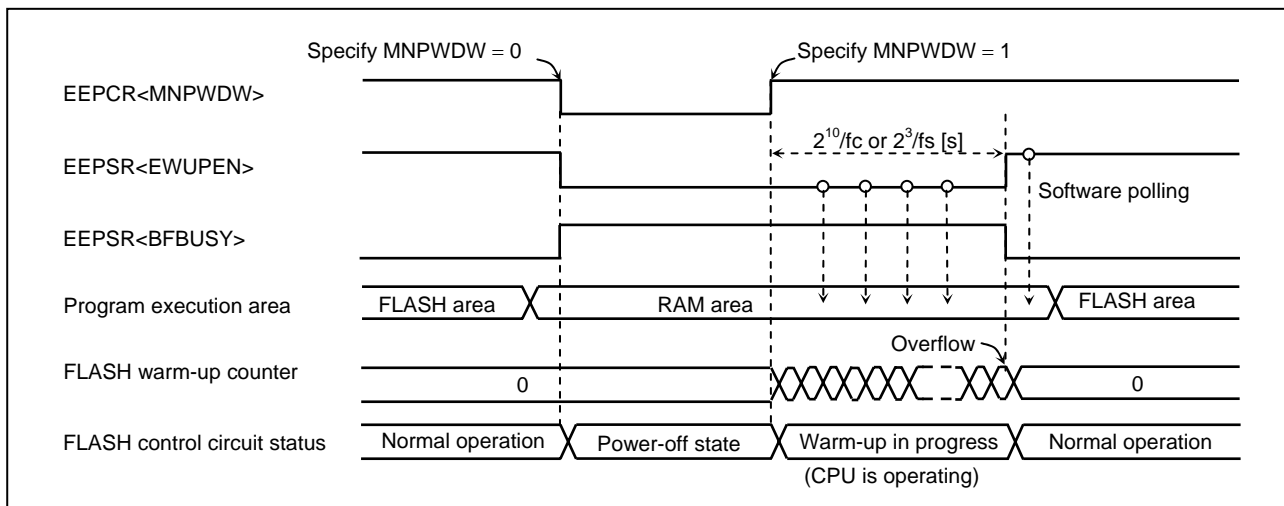
Figure 2.17.6 Software-based Power Control for the FLASH Control Circuit (EEPCR<MNPWDW>)

Example: Performing software-based power control for the FLASH control circuit

```
sRAMAREA:
          DI                              ;   Disable an interrupt (IMF ← "0")
          LD        (WDTCR2),4Eh          ;   Clear the binary counter if the watchdog
                                              timer is in use
          CLR       (EEPCR).0             ;   Clear the EEPCR<MNPWDW> to "0".
          ⋮         ⋮
          SET       (EEPCR).0             ;   Set the EEPCR<MNPWDW> to "1"
sLOOP1:   TEST      (EEPSR).1             ;   Monitor the EEPSR<EWUPEN> register.
          JRS       T,sLOOP1              ;   Jump to sLOOP1 if EEPSR<EWUPEN> =
                                              "0".
          JP        MAIN                  ;   Jump to the FLASH area.
```

Example: Performing software-based power control for the FLASH control circuit
sRAMAREA:

2.17.5.2 Automatic Power Control for the FLASH Control Circuit (EEPCR<ATPWDW>)

The EEPCR<ATPWDW> is an automatic power control bit for the FLASH control circuit. It is possible to suppress power consumption by automatically shutting down the power for the FLASH control circuit when an operation mode is changed to IDLE0/1/2 and SLEEP0/1/2 modes. This bit can be specified regardless of the area in which a program is being executed.

After the EEPCR<ATPWDW> is cleared to "0", entering an operation mode (IDLE0/1/2 or SLEEP0/1/2) where the CPU is at a halt automatically turns off the power for the FLASH control circuit. Once the operation mode is released, the warm-up time (CPU wait) is automatically counted to resume normal processing. The CPU wait period is either $2^{10}/fc$ (SYSCK = "0") or $2^3/fs$ (SYSCK = "1"). If the EEPCR<ATPWDW> is "1", releasing the operation mode does not cause the CPU wait.

If EEPCR<MNPWDW> = "1", executing a STOP mode forcibly turns off the power for the FLASH control circuit regardless of the setting of the EEPCR<ATPWDW>. When the STOP mode is released, a STOP mode oscillation warm-up is carried out, and then an FLASH control circuit warm-up (CPU wait) is automatically performed. If the EEPCR<MNPWDW> is "0", entering/exiting a STOP mode allows the power for the FLASH control circuit to be kept turned off.

Note 1: The EEPCR<ATPWDW> functions only if the EEPCR<MNPWDW> is "1". If the EEPCR<MNPWDW> is "0", the power for the FLASH control circuit is kept turned off when an operation mode is executed or released.

Note 2: During an FLASH warm-up (CPU wait), the peripheral circuits continue operating, but the CPU stays at a halt. Even if an interrupt latch is set under this condition, no interrupt process occurs until the CPU wait is completed. If the IMF is "1" when the interrupt latch is set, interrupt process takes place according to the interrupt priority after the CPU has started operating.
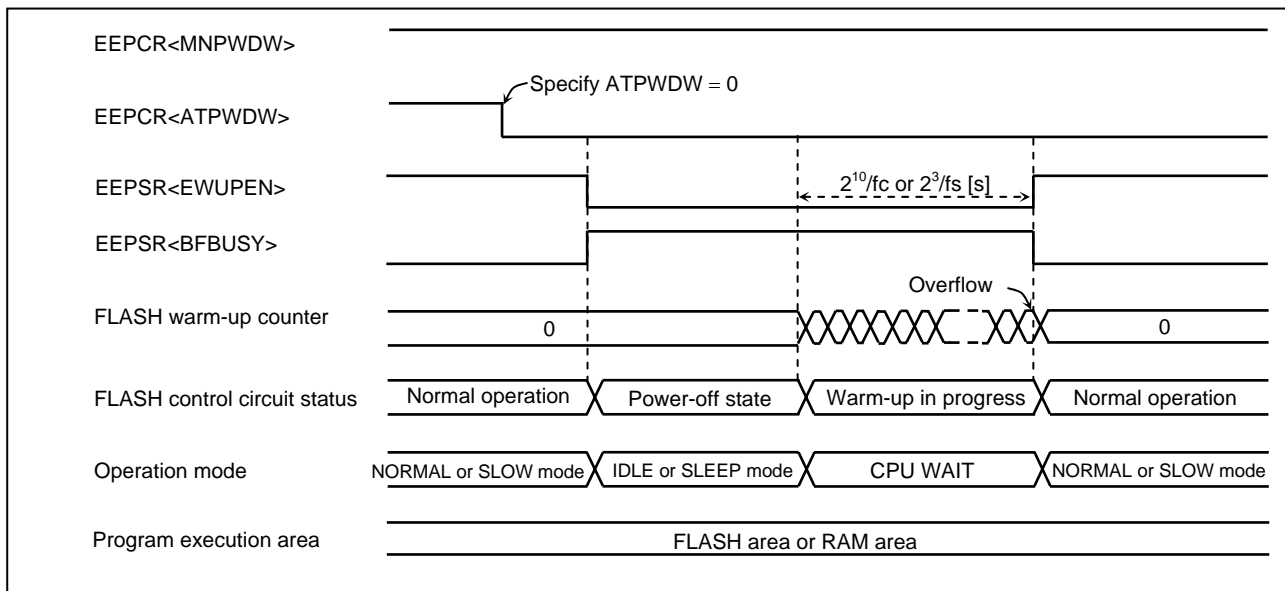
Figure 2.17.7 Automatic Power Control for the FLASH Control Circuit (EEPCR<ATPWDW>)

2.17.6   Accessing the FLASH Data Memory Area

During the writing to the data memory of FLASH area, neither a read nor fetch can be performed for the 8000H to FFFFH area. Therefore, to write the data memory of FLASH, the program being executed should be jumped to RAM area or should be jumped to the support program in BOOT-ROM. For details about the support program in BOOT-ROM, refer to "2.17.6.2 Method of Using Support Programs in the BOOT-ROM".

An LD instruction can be used to read data from the data memory of FLASH area byte by byte. The support program incorporated in the BOOT-ROM can also be used to read data from the data memory of FLASH area.

If a nonmaskable interrupt occurs during a write to the FLASH (EEPSR<BFBUSY> = "1"), the WINT is set to "1" and the writing is discontinued, and then the warm-up (CPU wait) for control circuit of Flash memory is executed (The write data counter is also initialized). If WINT = "1" is detected in the nonmaskable interrupt service routine, a write is not completed successfully. So, it is necessary to try a write again. The warm-up period is $2^{10}/fc$ (SYSCK = "0") or $2^3/fs$ (SYSCK = "1"). After 1 to 63 bytes are saved to the temporary data buffer, if an interrupt generates, the specified page of FLASH is not written. (It keeps previous data.)

Note 1: After 64 bytes are written to the temporary data buffer, the generating of an interrupt may cause the writing the page of FLASH to an unexpected value.

Note 2: During the warm-up period for Flash memory (CPU wait), the peripheral circuits continue operating, but the CPU stays at a halt until the warm-up is finished. Even if an interrupt latch is set to "1" by generating of interrupt request, an interrupt sequence doesn't start till the end of warm-up. If interrupts occur during a warm-up period with IMF = "1", the interrupt sequence which depends on interrupt priority will start after warm-up period.

Note 3: When write the data to Flash memory from RAM area, disable all the nonmaskable interrupt by clearing interrupt master enable flag (IMF) to "0" beforehand. However, in support program of BOOT-ROM, there is no need to clear the IMF because BOOT-ROM already has a DI (Disable Interrupt) instruction.

2.17.6.1 Method of Developing the Control Program in the RAM Area

www.DataSheet4U.com

To develop the program in RAM, the write control program should be stored in FLASH beforehand or should load from external device by using peripheral function (Example: UART, SIO etc). Given below is an example of developing the control program in the RAM area.

(1) Example of developing and writing the control program to the RAM area

1. For the emulation chip, set the EEPEVA register with an optimum time value according to the operating frequency.

2. Transfer the write control program to the RAM area.

3. Release an address trap in the RAM area (Set up the WDTCR1 and WDTCR2 registers).

4. Jump to the RAM area.

5. Monitor the EEPSR<EWUPEN>. If it is "0", set the EEPCR<MNPWDW> to "1", and then start and keep polling until the EEPSR<EWUPEN> becomes "1".

6. Clear the interrupt master enable flag (IMF ← "0").

7. Set the EEPCR with "3BH" (to enable a write to the FLASH).

8. Execute a write instruction for 64 bytes to the FLASH area.

9. Start and keep polling by software until the EEPSR<BFBUSY> becomes "0". (Upon completion of an erase and write to the FLASH cells, the EEPSR<BFBUSY> is set to "1". For the FLASH product, the required write time is typically 4 ms. For the emulation chip, it is the value specified in the EEPEVA register.)

10. Set the EEPCR with "CBH" (to disable a write to the FLASH).

11. Jump to the FLASH area (Main program).

Note: See (2), "Method of specifying an address for a write to the FLASH," for a description about the FLASH address to be specified at step 8 above.

(2) Method of specifying an address for a write to the FLASH

The FLASH page to be written is specified by the 10 high-order bits of the address of the first-byte data. The first-byte data is stored at the first address of the temporary data buffer. If the data to be written is, for example, 8040H, page 1 is selected, and the data is stored at the first address of the temporary data buffer. Even if the 6 low-order bits of the specified address is not 000000B, the first-byte data is always stored at the first address of the data buffer.

Any address can be specified as the second and subsequent address within FLASH area (for the MCU mode, 8000 to 81FFH and, for the serial PROM mode, 8000H to FFFFH). The write data bytes are stored in the temporary data buffer in the sequence they are written, regardless of what address is specified. Usually, the address that is the same as the first-byte is specified for the second and subsequent address. A 16-bit transfer instruction (LDW) can also be used for writing to the temporary data buffer.

Example: Data bytes 00H to 3FH are written to page 1.
(Figure 2.17.8 shows the example of data buffer and pages.)

```
            DI                                  ; Disable an interrupt (IMF ← "0")
            LD      C,00H
            LD      HL,EEPCR                     ; Specify the EEPCR register address.
            LD      IX,8040H                     ; Specify a write address.
            LD      (HL),3BH                     ; Specify the EEPCR
sLOOP1:
            LD      (IX),C                       ; Store data to the temporary data buffer.
                                                   (A write page is selected when the first
                                                   byte is written.)
            INC     C                            ; C = C + 1
            CMP     C,40H                        ; Jump to sLOOP1 if C is not 40H
            JR      NZ,sLOOP1
sLOOP2:
            TEST    (EEPSR).0
            JRS     F,sLOOP2                     ; Jump to sLOOP2 if EEPSR<BFBUSY> =
                                                   "1".
            LD      (HL),0CBH                    ; Specify the EEPCR
```

Note: If the BFBUSY is "1", executing a read instruction or fetch to the FLASH area causes "FFH" to be read. Fetching "FFH" results in a software interrupt occurring.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	00H	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
	10H	11H	12H	13H	14H	15H	16H	Temporary data buffer		19H	1AH	1BH	1CH	1DH	1EH	1FH
	20H	21H	22H	23H	24H	25H	26H			29H	2AH	2BH	2CH	2DH	2EH	2FH
	30H	31H	32H	33H	34H	35H	36H	37H	38H	39H	3AH	3BH	3CH	3DH	3EH	3FH

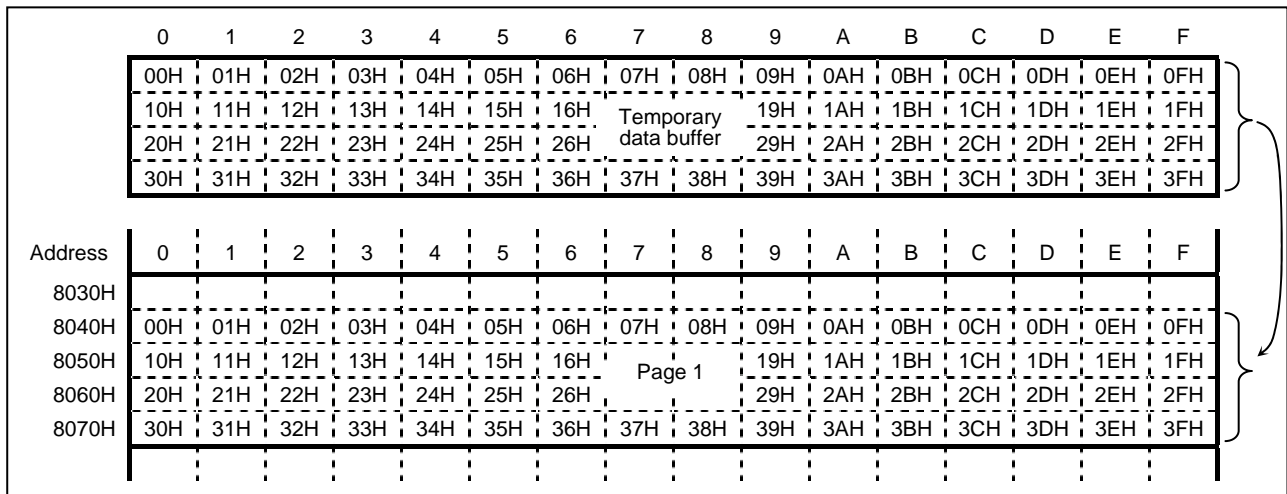| Address	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
8030H																
8040H	00H	01H	02H	03H	04H	05H	06H	07H	08H	09H	0AH	0BH	0CH	0DH	0EH	0FH
8050H	10H	11H	12H	13H	14H	15H	16H	Page 1		19H	1AH	1BH	1CH	1DH	1EH	1FH
8060H	20H	21H	22H	23H	24H	25H	26H			29H	2AH	2BH	2CH	2DH	2EH	2FH
8070H	30H	31H	32H	33H	34H	35H	36H	37H	38H	39H	3AH	3BH	3CH	3DH	3EH	3FH

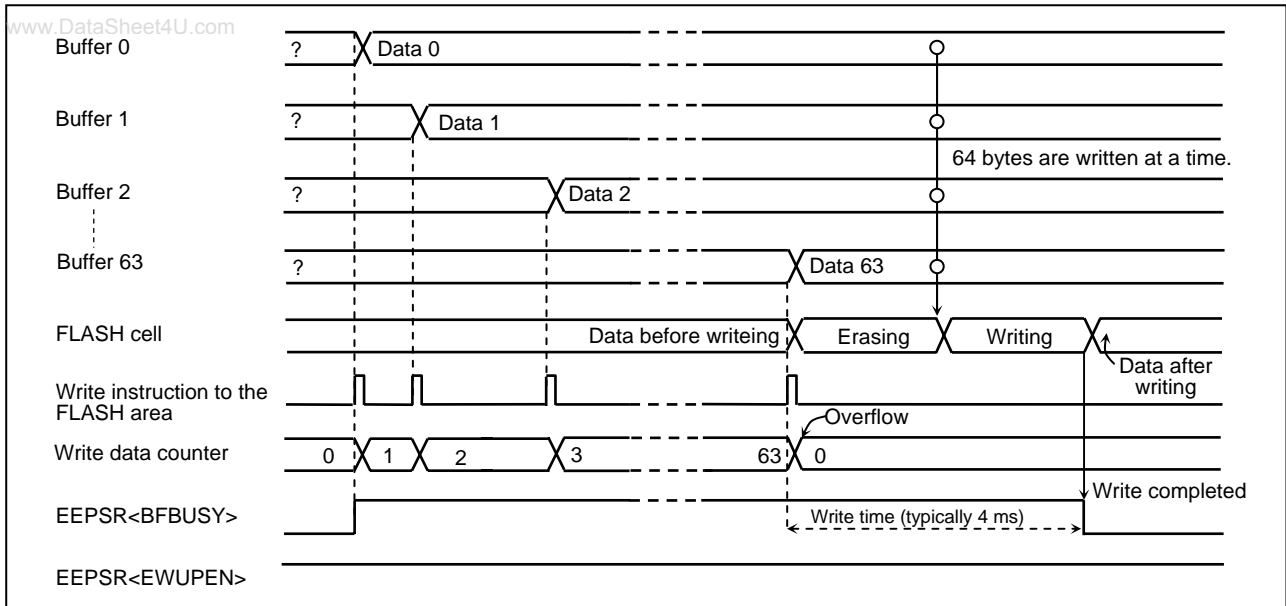Figure 2.17.9 Data Buffer and Write Page (Example)

Figure 2.17.10 Write to the FLASH Data Memory Area (In case of FLASH product)
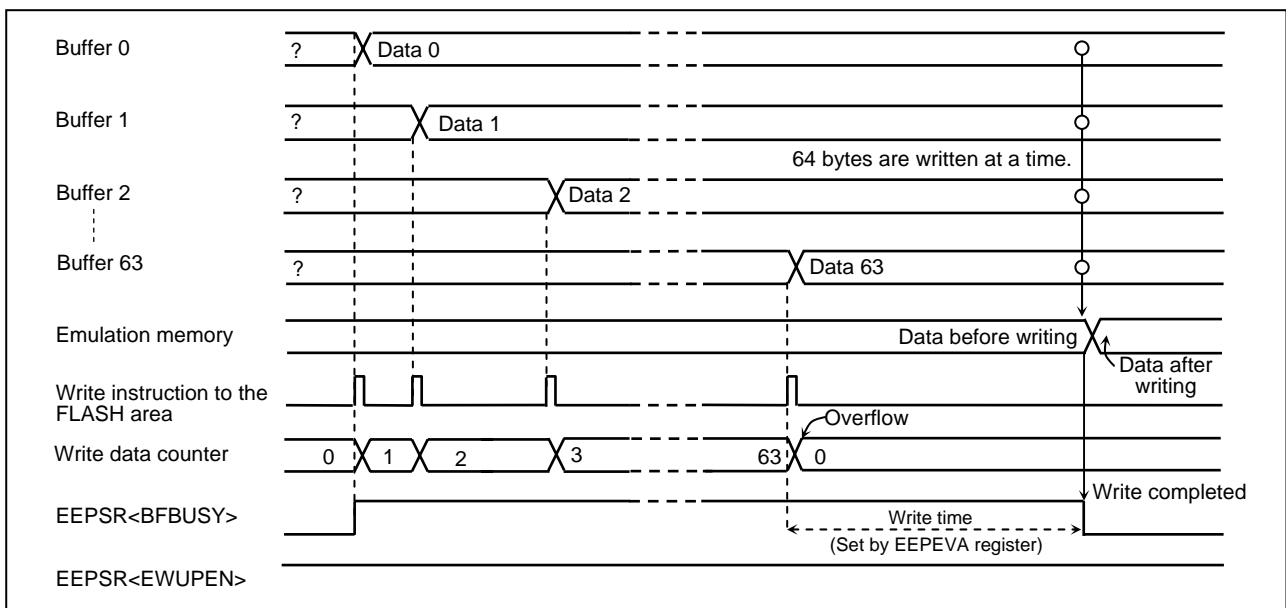

Figure 2.17.11 Write to the FLASH Data Memory Area (In case of emulation chip)

Note 1: The emulation chip is written to emulation memory instead of FLASH cell.

Note 2: In case of emulation chip, the data stacked to data buffer is written to emulation memory just before EEPSR<BFBUSY> is changed from "1" to "0". Therefore, if the writing of FLASH is stopped forcibly after the write data counter becomes overflow, the memory value on the page subjected to a write may be different from FLASH product.

2.17.6.2 Method of Using Support Programs in the BOOT-ROM

www.DataSheet4U.com

The BOOT-ROM of TMP86FM48 has Support Program to simplify writing/reading of FLASH. This program supports three subroutines.

    1. Writing to data FLASH from RAM

    2. Reading from data FLASH to RAM

    3. Reading from program FLASH to RAM

In addition to a program for controlling a write in the serial PROM mode, the BOOT-ROM incorporates support programs for simplifying a write to the data memory of FLASH in the MCU mode. The support programs take the form of a subroutine. After setting general-purpose registers with the necessary data, just execute a CALL instruction for a support program. It enables a write to and a read from the FLASH. There are two subroutines in BOOT-ROM. The Table 2.17.2 shows the function of these subroutines.

Table 2.17.2 Support Program (Subroutines) in BOOT-ROM

Program	CALL address	Function
Support program 1	3E00H	Writing to data memory of FLASH (8000H to 81FFH) from RAM area is available. 64-byte data can be written at a time.
Support program 2	3E2CH	Reading from FLASH memory (8000H to FFFFH) into RAM area is available. 64-byte data can be read at a time.

When using the support program, it is unnecessary to prepare an FLASH write program in advance or develop it in the RAM area.

Support program 1 enables 64 consecutive data bytes to be transferred from the RAM area to any data memory of FLASH page in block. (Only data memory of FLASH is available.) Support program 2 enables data to be transferred from any FLASH memory page (Both data memory of FLASH and program memory are available.) to a specified 64-byte consecutive RAM area in block.

How to use the support programs in the BOOT-ROM is explained below. See (3), "Support program 1," and (4), "Support program 2," for the source code of the support programs.

(1) Example of using support program 1 to write data to the FLASH data area
(Block transfer from the RAM area to the FLASH data area)

    1. For the emulation chip, set the EEPEVA register with the optimum time according to the operating frequency.

    2. Set data in the transfer-source RAM area.

    3. Set the RAM area start address (Transfer source) in the HL register.

    4. Set the FLASH data area start address (Transfer destination) in the DE register.

    5. Set "1FH" in the B register. (Be sure to set 1FH (Half of the number of bytes to be written.))

    6. Clear the binary counter if the watchdog timer is in use.

    7. Execute a CALL instruction to "3E00H".

    8. Data is transferred from the RAM area to the FLASH data area in block. After several milliseconds, program control is returned to the main routine.

www.DataSheet4U.com

Note 1: Steps 1 to 6 above are executed in the FLASH area.

Note 2: Support program 1 rewrites the HL, DE, B, and WA registers. If the existing data in them are necessary, save it in advance.

Note 3: If the EEPCR<MNPWDW> is "0", executing support program 1 rewrites it to "1" before performing a block transfer.

Note 4: Executing a CALL instruction for support program 1 consumes two bytes of stack.

Note 5: If the watchdog timer is in use, be sure to clear the binary counter for it before executing a CALL instruction to 3E00H.

Note 6: Do not specify the address from 8200H to FFFFH as a transfer destination address in MCU mode.

Example) Setting up HL = 0050H, DE = 8100H, and B = 1FH, and executing a CALL instruction for support program 1 (3E00H)
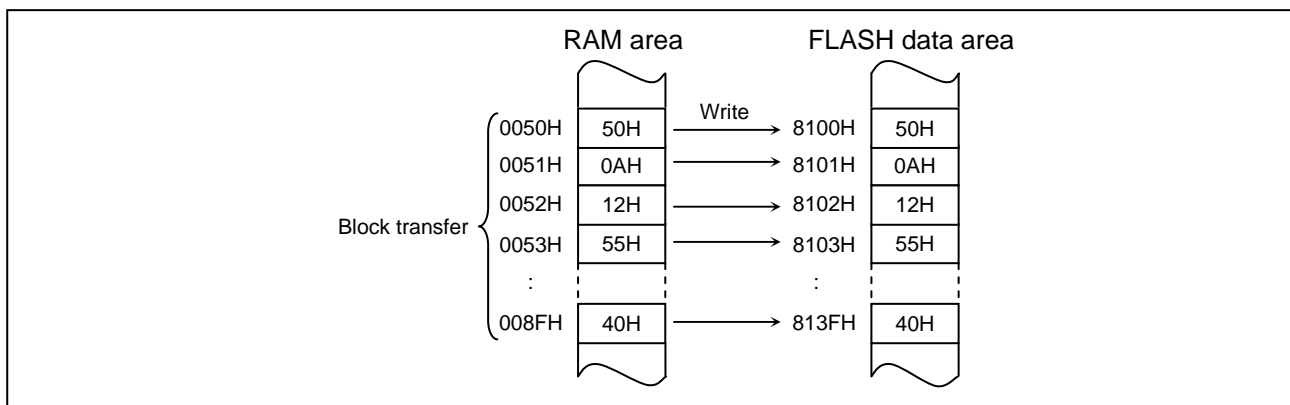
Figure 2.17.12 Example of Using Support Program 1 to Write Data to the FLASH Data Area

(2) Using support program 2 to read data from the FLASH area
(Block transfer from the FLASH area to the RAM area)

1. Set the RAM area start address (Transfer destination) in the HL register.
2. Set the FLASH area start address (Transfer source) in the DE register.
3. Set "1FH" in the B register. (Be sure to set 1FH (Half of the number of bytes to be read.))
4. Execute a CALL instruction to "3E2CH".
5. Data is transferred from the FLASH area to the RAM area in block. Upon completion of processing, program control is returned to the main routine.

Note 1: A LD instruction can be used to read data from the FLASH area in byte units without using support program 2.

Note 2: Steps 1 to 4 above are executed in the FLASH area.

Note 3: Support program 2 rewrites the HL, DE, B, and WA registers. If the existing data in them are necessary, save it in advance.

Note 4: Executing a CALL instruction for support program 2 consumes two bytes of stack.

Example) Setting up HL = 0050H, DE = 8100H, and B = 1FH, and executing a CALL instruction for support program 2 (3E2CH)
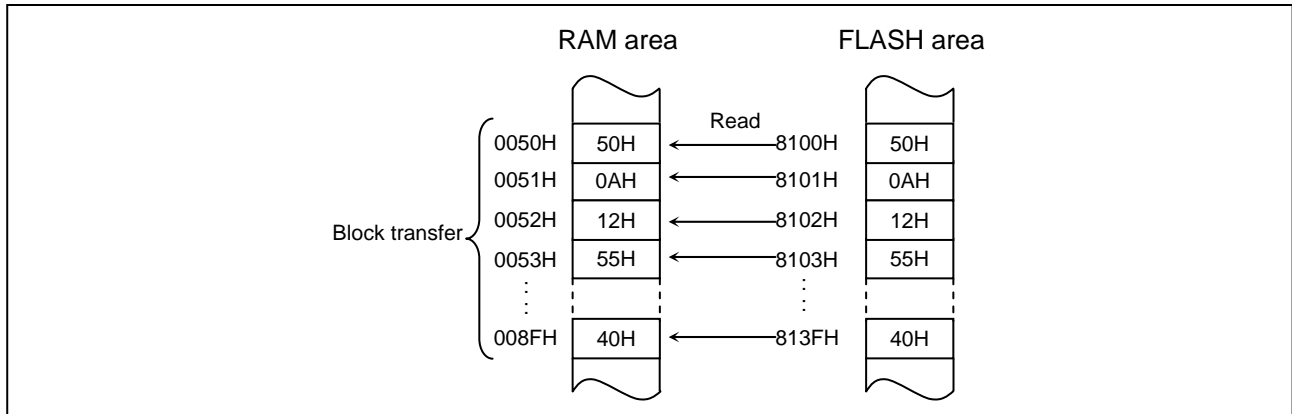
Figure 2.17.13 Example of Using Support Program 2 to Read Data from the FLASH Area

(3) Support program 1 (Block transfer from the RAM area to the FLASH data area)

Shown below is the support program source code for writing data to the FLASH.

```
USER_SUB_WRITE section code abs = 3E00H
sUSER_main1:
            TEST      (EEPSR).1
            JRS       f,sRAM_to_EEP              ;  Jump sRAM_to_EEP if the
                                                    EEPSR<EWUPEN> is "1"
sEEP_warmingup:
            SET       (EEPCR).0                  ;  Set the EEPCR<MNPWDW> to "1".
            TEST      (EEPSR).1                  ;  Wait until a warm-up is completed.
            JRS       t,sEEP_warmingup
sRAM_to_EEP:
            DI                                   ;  Disable an interrupt.
            AND       DE,0FFC0H                  ;  Mask the 6 low-order bits.
            LD        (EEPCR),3BH                ;  Enable a write to the FLASH.
sBFBUSY_loop:
            LD        WA,(HL)                    ;  Read data from the RAM.
            LD        (DE),WA                    ;  Write data to the FLASH.
            INC       HL
            INC       HL
            DEC       B
            JRS       F,sBFBUSY_loop
sEEP_write_end:
            TEST      (EEPSR).0                  ;  Perform polling on the BFBUSY flag.
            JRS       F,sEEP_write_end
            LD        (EEPCR),0CBH               ;  Disable a write to the FLASH.
            RET
```

(4) Support program 2 (Block transfer from the FLASH area to the RAM area)

Shown below is the support program source code for reading data from the FLASH.

```
USER_SUB_READ section code abs = 3E2CH
sUSER_main2:
            AND         DE,0FFC0              ;   Mask the 6 low-order bits.
            LD          (EEPCR),0CBH          ;   Disable a write to the FLASH.
sEEP_read_loop:
            LD          WA,(DE)
            LD          (HL),WA
            INC         HL
            INC         HL
            INC         DE
            INC         DE
            DEC         B
            JRS         F,sEEP_read_loop
            RET
```

2.18 FLASH Program Memory

The TMP86FM48 incorporates 32256 bytes (8200H to FFFFH) of program memory. If the data memory of FLASH is not in use, the TMP86FM48 can be used as an FLASH product with 32768 full bytes. To write data to the program memory (Data memory of FLASH), execute the serial PROM mode.

2.18.1 Configuration

The program memory has the same configuration as for the data memory of FLASH. See Section 2.17.1 "Configuration".

2.18.2 Control

The program memory is controlled in the same manner as for the data memory of FLASH. See Section 2.17.2 "Control".

2.18.3 FLASH Write Enable Control (EEPCR<EEPMD>)

The FLASH write enable control register for the program memory behaves in the same manner as for the data memory of FLASH. See Section 2.17.3 "FLASH Write Enable Control (EEPCR<EEPMD>)".

2.18.4 FLASH Write Forcible Stop (EEPCR<EEPRS>)

The FLASH write forcible stop register for the program memory behaves in the same manner as for the data memory of FLASH. See Section 2.17.4 "FLASH Write Forcible Stop (EEPCR<EEPRS>)".

2.18.5 Power Control for the FLASH Control Circuit

The power for the program memory control circuit is controlled in the same manner as for the data memory of FLASH. See Section 2.17.5 "Power Control for the FLASH Control Circuit".

2.18.6 Accessing the FLASH Program Memory Area

Basically, a write to the program memory area is carried out using UART communication after the serial PROM mode is entered. For explanations about what control is performed in the serial PROM mode, see the following descriptions.

2.19 Serial PROM Mode

2.19.1 Outline

The TMP86FM48 has a 2 Kbytes BOOT-ROM for programming to FLASH memory. This BOOT-ROM is a mask ROM that contains a program to write the FLASH memory on-board. The BOOT-ROM is available in a serial PROM mode and it is controlled by BOOT pin and $\overline{\text{RESET}}$ pin, and is communicated via TXD (P06) and RXD (P05) pins. There are four operation modes in a serial PROM mode: FLASH memory writing mode, RAM loader mode, FLASH memory SUM output mode and Product discrimination code output mode. Operating area of serial PROM mode differs from that of MCU mode. The operating area of serial PROM mode shows in Table 2.19.1.

Table 2.19.1 Operating Area of Serial PROM Mode

Parameter	Min	Max	Unit
Operating voltage	2.7	3.6	V
High frequency (Note)	2	16	MHz
Temperature	25 ± 5		°C

Note: Even though included in above operating area, part of frequency can not be supported in serial PROM mode. For details, refer to.

2.19.2 Memory Mapping

The BOOT-ROM is mapped in address 3800_H to $3FFF_H$. The Figure 2.19.1 shows a memory mapping.
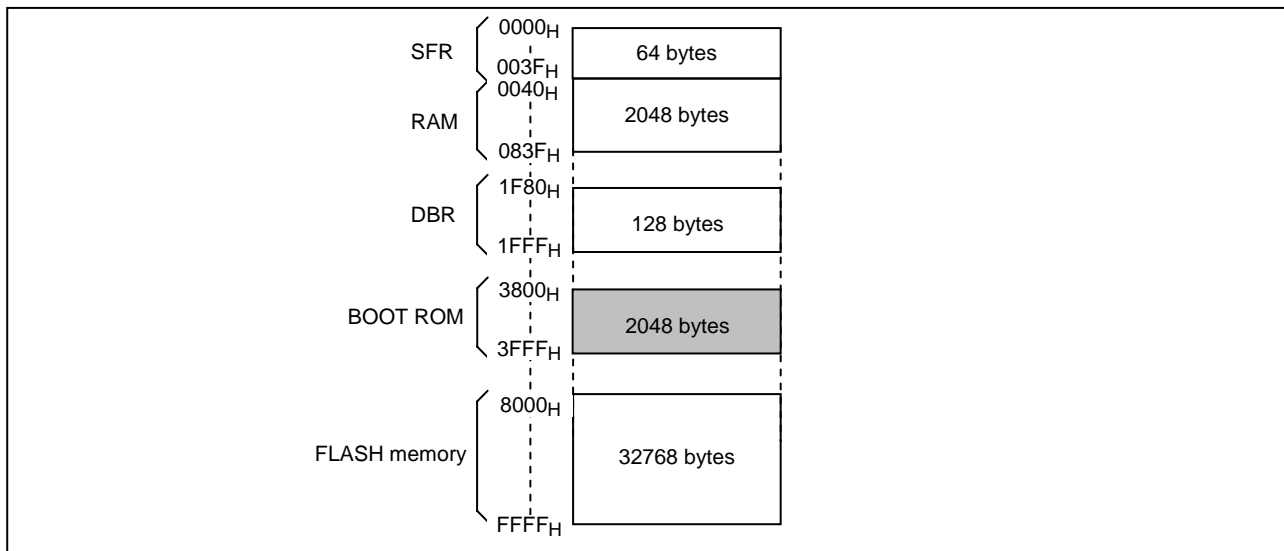
Figure 2.19.1 Memory Address Maps

2.19.3 Serial PROM Mode Setting

2.19.3.1 Serial PROM Mode Control Pins

To execute on-board programming, start the TMP86FM48 in serial PROM mode. Setting of a serial PROM mode is shown in Table 2.19.2.

Table 2.19.2 Serial PROM Mode Setting

Pin	Setting
BOOT pin	High
$\overline{\text{RESET}}$ pin	⌐

2.19.3.2 Pin Function

In the serial PROM mode, TXD (P06) and RXD (P05) pins are used as a serial interface pin.

Pin Name (Serial PROM Mode)	Input/ Output	Function		Pin Name (MCU Mode)
TXD	Output	Serial data output	(Note 1)	P06
RXD	Input	Serial data input		P05
BOOT	Input	Serial PROM mode control		BOOT
RESET	Input	Serial PROM mode control		RESET
TEST	Input	0V		TEST
VDD, AVDD	Power Supply	2.7 V to 3.6 V		
VSS, AVSS/VASS		0V		
VAREF		Open or equal with VDD		
P00 to P04,P07	I/O	Placed in High-Z state during serial PROM mode.		
P10 to P17				
P20 to P22				
P30 to P37				
P50 to P52				
P60 to P67				
P70 to P77				
P80 to P87				
XIN	Input	Resonator connecting pins for high-frequency clock.	(Note 2)	
XOUT	Output	For inputting external clock, XIN is used and XOUT is opened.		

Note 1: When the device is used as on-board writing and other parts are already mounted in place, be careful no to affect these communication control pins.

Note 2: Operating area of high frequency in serial PROM mode is from 2 MHz to 16 MHz.

To set a serial PROM mode, connect device pins as shown in Figure 2.19.2.
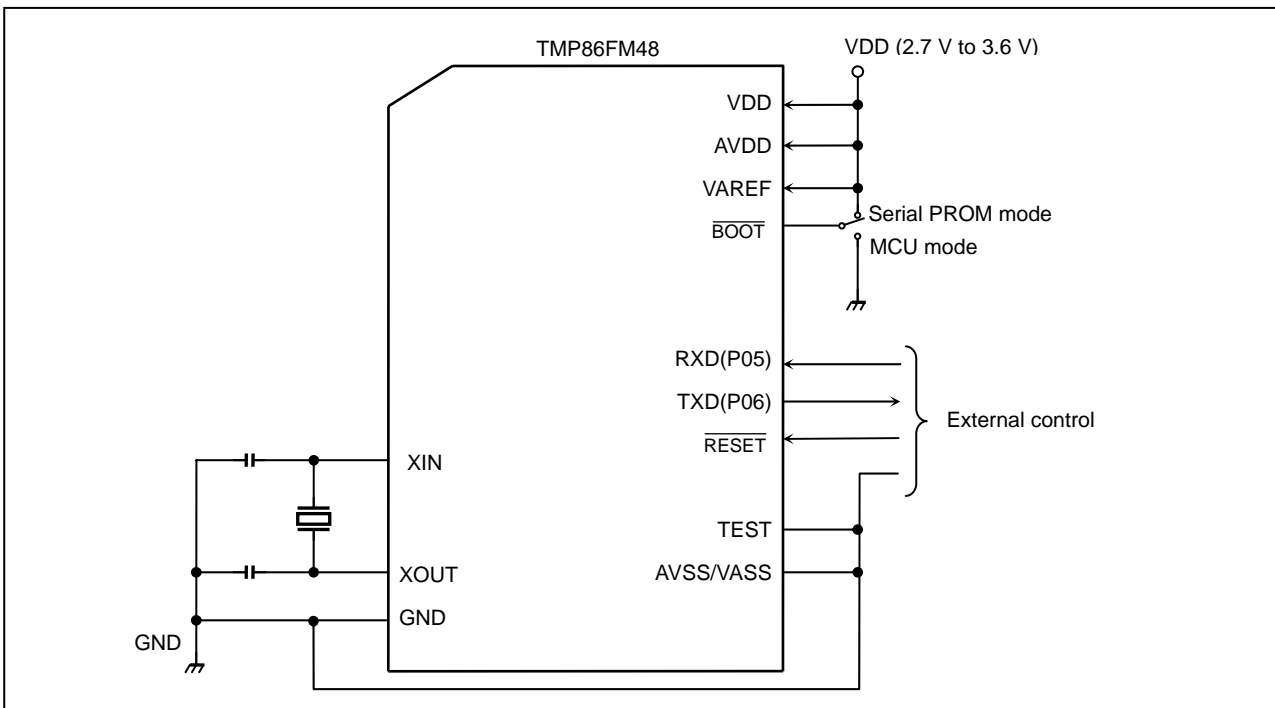
Figure 2.19.2 Serial PROM Mode Port Setting

2.19.3.3 Activating Serial PROM Mode

The following is a procedure of setting of serial PROM mode. Figure 2.19.3 shows a serial PROM mode timing.

(1) Turn on the power to the VDD pin.

(2) Set the $\overline{\text{RESET}}$ and TEST pins to low level.

(3) Set the BOOT pin to high level.

(4) Wait until the power supply and clock sufficiently stabilize.

(5) Release the $\overline{\text{RESET}}$. (Set to high level)

(6) Input a matching data (5AH) to RXD pin after waiting for setup sequence. For details of the setup timing, refer to "2.19.14 UART Timing"
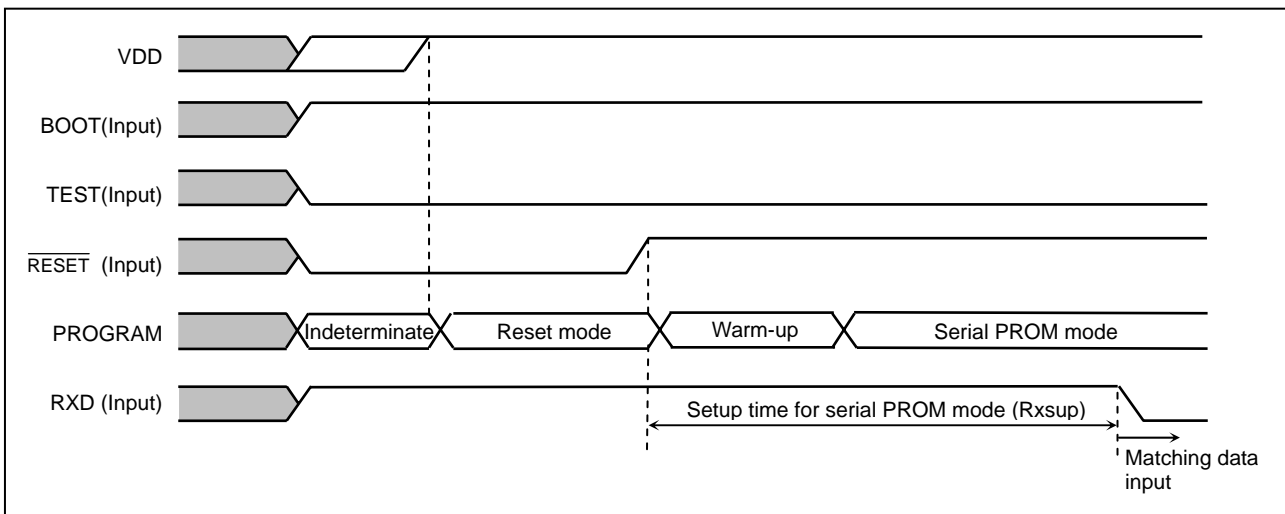
Figure 2.19.3 Serial PROM Mode Timing

2.19.4 Interface Specifications for UART

The following shows the UART communication format used in serial PROM mode.

Before on-board programming can be executed, the communication format on the external controller side must also be set up in the same way as for this product.

Note that although the default baud rate is 9600 bps, it can be changed to other values as shown in Table 2.19.3. The Table 2.19.4 shows an operating frequency and baud rate in serial PROM mode. Except frequency which is not described in Table 2.19.4 can not use in serial PROM mode.

Baud rate (Default): 9600 bps
Data length: 8 bits
Parity addition: None
Stop bit length: 1 bit

Table 2.19.3 Baud Rate Modification Data

Baud rate modification data	04H	05H	06H	07H	0AH	18H	28H
Baud rate (bps)	76800	62500	57600	38400	31250	19200	9600

Table 2.19.4 Operating Frequency and Baud Rate in Serial PROM Mode

(Note 3)	Reference Baud Rate (bps)		76800		62500		57600		38400		31250		19200		9600	
	Baud Rate Modification Data		04H		05H		06H		07H		0AH		18H		28H	
	Ref. Frequency (MHz)	Area (MHz)	Baud rate (bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)	(bps)	(%)
1	2	1.91~2.10	–	–	–	–	–	–	–	–	–	–	–	–	9615	+0.16
2	4	3.82~4.19	–	–	–	–	–	–	–	–	31250	0.00	19231	+0.16	9615	+0.16
	4.19	3.82~4.19	–	–	–	–	–	–	–	–	32734	+4.75	20144	+4.92	10072	+4.92
3	4.9152	4.70~5.16	–	–	–	–	–	–	38400	0.00	–	–	19200	0.00	9600	0.00
	5	4.70~5.16	–	–	–	–	–	–	39063	+1.73	–	–	19531	+1.73	9766	+1.73
4	6	5.87~6.45	–	–	–	–	–	–	-	–	–	–	–	–	9375	–2.34
	6.144	5.87~6.45	–	–	–	–	–	–	-	–	–	–	–	–	9600	0.00
5	7.3728	7.05~7.74	–	–	–	–	57600	0.00	-	–	–	–	19200	0.00	9600	0.00
6	8	7.64~8.39	–	–	62500	0.00	-	–	38462	+0.16	31250	0.00	19231	+0.16	9615	+0.16
7	9.8304	9.40~10.32	76800	0.00	–	–	-	–	38400	0.00	–	–	19200	0.00	9600	0.00
	10	9.40~10.32	78125	+1.73	–	–	-	–	39063	+1.73	–	–	19531	+1.73	9766	+1.73
8	12	11.75~12.90	–	–	–	–	57692	+0.16	-	–	31250	0.00	18750	–2.34	9375	–2.34
	12.288	11.75~12.90	–	–	–	–	59077	+2.56	-	–	32000	+2.40	19200	0.00	9600	0.00
	12.5	11.75~12.90	–	–	60096	–3.85	60096	+4.33	-	–	30048	–3.85	19531	+1.73	9766	+1.73
9	14.7456	14.10~15.48	–	–	–	–	57600	0.00	38400	0.00	–	–	19200	0.00	9600	0.00
10	16	15.27~16.77	76923	+0.16	62500	0.00	–	–	38462	+0.16	31250	0.00	19231	+0.16	9615	+0.16

Note 1: "Ref.Frequency" and "Area" show the high frequency area supported in serial PROM mode. Except the above frequency can not be supported in serial PROM mode even though the high frequency is included in area from 2 MHz to 16 MHz.

Note 2: The total error of frequency must be kept within +/–3% so that the auto-detection of frequency is executed correctly.

Note 3: An external controller should transmit a matching data repeatedly till the TMP86FM48 transmit an echo back data. Above number indicates a transmission number of times of matching data till transmission of echo back data.

2.19.5 Command

There are five commands in serial PROM mode. After reset release, the TMP86FM48 waits a matching data (5AH).

Table 2.19.5 Command in Serial PROM Mode

Command Data	Operation Mode	Remarks
5AH	Setup	Matching data. Always start with this command after reset release.
30H	FLASH memory writing	Writing to area from 8000H to FFFFH is enable.
60H	RAM loader	Writing to area from 0050H to 082FH is enable.
90H	FLASH memory SUM output	The checksum of entire FLASH area (from 8000H to FFFFH) is output in order of the upper byte and the lower byte.
C0H	Product discrimination code output	Product discrimination code, that is expressed by 13 bytes data, is output.

2.19.6 Operation Mode

There are four operating modes in serial PROM mode: FLASH memory writing mode, RAM loader mode, FLASH memory SUM output mode and Product discrimination code output mode. For details about these modes, refer to "(1) FLASH memory writing mode" through "(4) Product discrimination code output mode".

(1) FLASH memory writing mode

The data are written to the specified FLASH memory addresses. The controller should send the write data in the Intel Hex format (Binary). For details of writing data format, refer to "2.19.7 FLASH memory Writing Data Format".

If no errors are encountered till the end record, the SUM of 32 Kbytes of FLASH memory is calculated and the result is returned to the controller.

To execute the FLASH memory writing mode, the TMP86FM48 checks the passwords except a blank product. If the passwords did not match, the program is not executed.

(2) RAM loader mode

The RAM loader transfers the data into the internal RAM that has been sent from the controller in Intel Hex format. When the transfer has terminated normally, the RAM loader calculates the SUM and sends the result to the controller before it starts executing the user program. After sending of SUM, the program jumps to the start address of RAM in which the first transferred data has been written. This RAM loader function provides the user's own way to control on-board programming.

To execute the RAM loader mode, the TMP86FM48 checks the passwords except a blank product. If the passwords did not match, the program is not executed.

(3) FLASH memory SUM output mode

The SUM of 32 Kbytes of FLASH memory is calculated and the result is returned to the controller.

The BOOT ROM does not support the reading function of the FLASH memory. Instead, it has this SUM command to use. By reading the SUM, it is possible to manage Revisions of application programs.

(4) Product discrimination code output mode

The product discrimination code is output as a 13-byte data, that includes the start address and the end address of ROM (In case of TMP86FM48, the start address is 8000H and the end address is FFFFH). Therefore, the controller can recognize the device information by using this function.

2.19.6.1 FLASH Writing Mode (Operation command: 30H)

Table 2.19.6 shows FLASH memory writing mode process.

Table 2.19.6 FLASH Writing Mode Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM48	Baud Rate	Transfer Data from TMP86FM48 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5Ah) –	9600 bps 9600 bps	– (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.19.3) –	9600 bps 9600 bps	– OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (30H) –	Changed new baud rate Changed new baud rate	– OK: Echo back data (30H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte 8th byte	Address 15 to 08 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	9th byte 10th byte	Address 07 to 00 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	11th byte 12th byte	Address 15 to 08 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	13th byte 14th byte	Address 07 to 00 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	15th byte : m'th byte	Password string (Note 5) –	Changed new baud rate Changed new baud rate	– OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	m'th + 1 byte : n'th − 2 byte	Intel Hex format (binary) (Note 2)	Changed new baud rate	–
	n'th − 1 byte	–	Changed new baud rate	OK: SUM (High) (Note 3) Error: Nothing transmitted
	n'th byte	–	Changed new baud rate	OK: SUM (Low) (Note 3) Error: Nothing transmitted
	n'th + 1 byte	(Wait for the next operation) (command data)	Changed new baud rate	–

Note 1: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to "2.19.8 Error Code".

Note 2: Refer to "2.19.10 Intel Hex Format (Binary)".

Note 3: Refer to "2.19.9 Checksum (SUM)".

Note 4: Refer to "2.19.11 Passwords".

Note 5: If all data of addresses from FFE0H to FFFFH are "00H" or "FFH", the passwords comparison is not executed because the device is considered as blank product. However, it is necessary to specify the password count storage addresses and the password comparison start address even though it is a blank product. If a password error occurs, the UART function of TMP86FM48 stops without returning error code to the controller. Therefore, when a password error occurs, the TMP86FM48 should be reset by $\overline{RESET}$ pin input.

Description of FLASH memory writing mode

1. The receive data in the 1st byte is the matching data. When the boot program starts in serial PROM mode, TMP86FM48 (Mentioned as "device" hereafter) waits for the matching data (5AH) to receive. Upon receiving the matching data, it automatically adjusts the UART's initial baud rate to 9,600bps.

2. When the device has received the matching data, the device transmits the data "5AH" as an echo back to the controller. If the device can not receive the matching data, the device does not transmit the echo back data and waits for the matching data again with changing baud rate. Therefore, the controller should send the matching data continuously until the device transmits the echo back data. An external controller should transmit a matching data repeatedly till the device transmit an echo back data. The transmission number of times of matching data varies by the frequency of device. For details, refer to Table 2.19.4.

3. The receive data in the 3rd byte is the baud rate modification data. The seven kinds of baud rate modification data shown in Table 2.19.3 are available. Even if baud rate changing is no need, be sure to send the initial baud rate data (28H: 9,600 bps).

4. When the 3rd byte data is one of the baud rate modification data corresponding to the device's operating frequency, the device sends the echo back data which is the same as received baud rate modification data. Then the baud rate is changed. If the 3rd byte data does not correspond to the baud rate modification data, the device stops UART function after sending 3 bytes of baud rate modification error code: (62H). The changing of baud rate is executed after transmitting the echo back data.

5. The receive data in the 5th byte is the command data (30H) to write the FLASH memory.

6. When the 5th byte is one of the operation command data shown in Table 2.19.5, the device sends the echo back data which is the same as received operation command data (in this case, 30H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).

7. The 7th byte is used as an upper bit (Bit15 to bit8) of the password count storage address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error occurs, the device stops UART function after sending 3 bytes of receiving error code: (A1H or A3H).

8. The 9th byte is used as a lower bit (Bit7 to bit0) of the password count storage address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error occurs, the device stops UART function after sending 3 bytes of receiving error code: (A1H or A3H).

9. The 11th byte is used as an upper bit (Bit15 to bit8) of the password comparison start address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error occurs, the device stops UART function after sending 3 bytes of receiving error code: (A1H or A3H).

10. The 13th byte is used as a lower bit (Bit7 to bit0) of the password comparison start address. When the receiving is executed correctly (No error), the device does not send any data. If the receiving error occurs, the device stops UART function after sending 3 bytes of receiving error code: (A1H or A3H).

11. The 15th through the m'th bytes are the password data. The number of passwords is the data (N) indicated by the password count storage address. The password data are compared for N entries beginning with the password comparison start address. The controller should send N bytes of password data to the device. If the passwords do not match, the device stops UART function without returning error code to the controller. If the data of addresses from FFE0H to FFFFH are all "FFH", the comparison of passwords is not executed because the device is considered as a blank product.

12. The receive data in the m'th + 1 through n'th − 2 byte are received as binary data in Intel Hex format. No received data are echoed back to the controller. The data which is not the start mark (3AH for ":") in Intel Hex format is ignored and does not send an error code to the controller until the device receives the start mark. After receiving the start mark, the device receives the data record, that consists of length of data, address, record type, writing data and checksum. After receiving the checksum of data record, the device waits the start mark data (3AH) again. The data of data record is temporarily stored to RAM and then, is written to specified FLASH memory by page (64 bytes) writing. For details of an organization of FLASH, refer to "2.19.7 Serial PROM Mode". Since after receiving an end record, the device starts to calculate the SUM, the controller should wait the SUM after sending the end record. If receive error or Intel Hex format error occurs, the device stops UART function without returning error code to the controller.

13. The n'th − 1 and the n'th bytes are the SUM value that is sent to the controller in order of the upper byte and the lower byte. For details on how to calculate the SUM, refer to "2.19.9 Checksum (SUM)". The SUM calculation is performed after detecting the end record, but the calculation is not executed when receive error or Intel Hex format error has occurred. The time required to calculate the SUM of the 32 Kbytes of FLASH memory area is approximately 100 ms at fc = 16 MHz. After the SUM calculation, the device sends the SUM data to the controller. After sending the end record, the controller can judge that the transmission has been terminated correctly by receiving the checksum.

14. After sending the SUM, the device waits for the next operation command data.

2.19.6.2 RAM Loader Mode (Operation Command: 60H)

Table 2.19.7 shows RAM loader mode process.

Table 2.19.7 RAM Loader Mode Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM48	Baud Rate	Transfer Data from TMP86FM48 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5AH) −	9600 bps 9600 bps	− (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.19.3) −	9600 bps 9600 bps	− OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (60H) −	Changed new baud rate Changed new baud rate	− OK: Echo back data (60H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte 8th byte	Address 15 to 08 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	− OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	9th byte 10th byte	Address 07 to 00 in which to store Password count (Note 4)	Changed new baud rate Changed new baud rate	− OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	11th byte 12th byte	Address 15 to 08 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	− OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	13th byte 14th byte	Address 07 to 00 in which to start Password comparison (Note 4)	Changed new baud rate Changed new baud rate	− OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	15th byte : m'th byte	Password string (Note 5) −	Changed new baud rate Changed new baud rate	− OK: Nothing transmitted Error: A1H × 3, A3H × 3 (Note 1)
	m'th + 1 byte : n'th − 2 byte	Intel Hex format (Binary) (Note 2)	Changed new baud rate	−
	n'th − 1 byte	−	Changed new baud rate	OK: SUM (High) (Note 3) Error: Nothing transmitted
	n'th byte	−	Changed new baud rate	OK: SUM (Low) (Note 3) Error: Nothing transmitted
RAM	−	The program jumps to the start address of RAM in which the first transferred data has been written.		

Note 1: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to 2.19.8 "Error Code".

Note 2: Refer to 2.19.10 "Intel Hex Format (Binary)".

Note 3: Refer to 2.19.9 "Checksum (SUM)".

Note 4: Refer to 2.19.11 "Passwords".

Note 5: If all data of addresses from FFE0H to FFFFH are "00H" or "FFH", the passwords comparison is not executed because the device is considered as blank product. However, it is necessary to specify the password count storage addresses and the password comparison start address even though it is a blank product. If a password error occurs, the UART function of TMP86FM48 stops without returning error code to the controller. Therefore, when a password error

occurs, the TMP86FM48 should be reset by $\overline{RESET}$ pin input.

Note 6: Do not send only end record after transferring of password string. If the TMP86FM48 receives the end record only after reception of password string, it does not operate correctly.

Note 7: When the FLASH power supply is turned off in user's program by setting EEPCR<MNPWDW>, be sure to disable the watchdog timer (WDT) or to clear the binary counter of WDT immediately before.

Description of RAM loader mode

1. The process of the 1st byte through the 4th byte are the same as FLASH memory writing mode.

2. The receive data in the 5th byte is the RAM loader command data (60H) to write the user's program to RAM.

3. When the 5th byte is one of the operation command data shown in Table 2.19.5, the device sends the echo back data which is the same as received operation command data (in this case, 60H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).

4. The process of the 7th byte through the m'th byte are the same as FLASH memory writing mode.

5. The receive data in the m'th + 1 through n'th – 2byte are received as binary data in Intel Hex format. No received data are echoed back to the controller.
   The data which is not the start mark (3AH for ":") in Intel Hex format is ignored and does not send an error code to the controller until the device receives the start mark. After receiving the start mark, the device receives the data record, that consists of length of data, address, record type, writing data and checksum. After receiving the checksum of data record, the device waits the start mark data (3AH) again. The data of data record is written to specified RAM by the receiving data. Since after receiving an end record, the device starts to calculate the SUM, the controller should wait the SUM after sending the end record. If receive error or Intel Hex format error occurs, the UART function of TMP86FM48 stops without returning error code to the controller.

6. The n'th – 1 and the n'th bytes are the SUM value that is sent to the controller in order of the upper byte and the lower byte. For details on how to calculate the SUM, refer to 2.19.9 "Checksum (SUM)". The SUM calculation is performed after detecting the end record, but the calculation is not executed when receive error or Intel Hex format error has occurred.
   The SUM is calculated by the data written to RAM, but the length of data, address, record type and checksum in Intel Hex format are not included in SUM.

7. The boot program jumps to the first address that is received as data in Intel Hex format after sending the SUM to the controller.

2.19.6.3 FLASH Memory SUM Output Mode (Operation Command: 90H)

Table 2.19.8 shows FLASH memory SUM output mode process.

Table 2.19.8 FLASH Memory SUM Output Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM48	Baud Rate	Transfer Data from TMP86FM48 to External Controller
BOOT ROM	1st byte 2nd byte	Matching data (5AH) −	9600 bps 9600 bps	− (Baud rate auto set) OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte 4th byte	Baud rate modification data (See Table 2.19.3) −	9600 bps 9600 bps	− OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte 6th byte	Operation command data (90H) −	Changed new baud rate Changed new baud rate	− OK: Echo back data (90H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte	−	Changed new baud rate	OK: SUM (High) (Note 2) Error: Nothing transmitted
	8th byte	−	Changed new baud rate	OK: SUM (Low) (Note 2) Error: Nothing transmitted
	9th byte	(Wait for the next operation) (Command data)	Changed new baud rate	−

Note 1: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to "2.19.8 Error Code".

Note 2: Refer to "2.19.9 Checksum (SUM)"

Description of FLASH memory SUM output mode

1. The process of the 1st byte through the 4th byte are the same as FLASH memory writing mode.

2. The receive data in the 5th byte is the FLASH memory SUM command data (90H) to calculate the entire FLASH memory.

3. When the 5th byte is one of the operation command data shown in Table 2.19.5, the device sends the echo back data which is the same as received operation command data (in this case, 90H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).

4. The 7th and the 8th bytes are the SUM value that is sent to the controller in order of the upper byte and the lower byte. For details on how to calculate the SUM, refer to "2.19.9 Checksum (SUM)".

5. After sending the SUM, the device waits for the next operation command data.

2.19.6.4 Product Discrimination Code Output Mode (Operation Command: C0H)

Table 2.19.9 shows product discrimination code output mode process.

Table 2.19.9 Product Discrimination Code Output Process

	Number of Bytes Transferred	Transfer Data from External Controller to TMP86FM48	Baud Rate	Transfer Data from TMP86FM48 to External Controller
BOOT ROM	1st byte	Matching data (5AH)	9600 bps	– (Baud rate auto set)
	2nd byte	–	9600 bps	OK: Echo back data (5AH) Error: Nothing transmitted
	3rd byte	Baud rate modification data (See Table 2.19.3)	9600 bps	–
	4th byte	–	9600 bps	OK: Echo back data Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)
	5th byte	Operation command data (C0H)	Changed new baud rate	–
	6th byte	–	Changed new baud rate	OK: Echo back data (C0H) Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)
	7th byte		Changed new baud rate	3AH — Start mark
	8th byte		Changed new baud rate	0AH — The number of transfer data (from 9th to 18th byte)
	9th byte		Changed new baud rate	02H — Length of address (2 bytes)
	10th byte		Changed new baud rate	00H — Reserved data
	11th byte		Changed new baud rate	00H — Reserved data
	12th byte		Changed new baud rate	00H — Reserved data
	13th byte		Changed new baud rate	00H — Reserved data
	14th byte		Changed new baud rate	01H — The number of ROM block (1 block)
	15th byte		Changed new baud rate	80H — First address of ROM
	16th byte		Changed new baud rate	00H —
	17th byte		Changed new baud rate	FFH — End address of ROM
	18th byte		Changed new baud rate	FFH —
	19th byte		Changed new baud rate	7FH — Checksum of transferred data (from 9th to 18th byte)
	20th byte	(Wait for the next operation) (Command data)	Changed new baud rate	–

Note: "xxH × 3" denotes that operation stops after sending 3 bytes of xxH. For details, refer to "2.19.8 Error Code".

Description of product discrimination code output mode

1. The process of the 1st byte through the 4th byte are the same as FLASH memory writing mode.

2. The receive data in the 5th byte is the product discrimination code output command data (C0H).

3. When the 5th byte is one of the operation command data shown in Table 2.19.5, the device sends the echo back data which is the same as received operation command data (in this case, C0H). If the 5th byte data does not correspond to the operation command data, the device stops UART function after sending 3 bytes of operation command error code: (63H).

4. The 7th and the 19th bytes are the product discrimination code. For details, refer to 2.19.12 "Product Discrimination Code".

5. After sending the SUM, the device waits for the next operation command data.

2.19.7 FLASH memory Writing Data Format

FLASH area of TMP86FM48 consists of 512 pages and one page size is 64 bytes.

Writing to FLASH is executed by page writing. Therefore, it is necessary to send 64 bytes data (for one page) even though only a few bytes data are written. Figure 2.19.4 shows an organization of FLASH area. When the controller sends the writing data to the device, be sure to keep the format described below.

1. The address of data after receiving the FLASH writing command should be the first address of page. For example, in case of page 2, the first address should be 8080H.

2. If the last data's address of data record is not end address of page, the address of the next data record should be the address + 1. For example, if the last data's address is 802FH (Page 0), the address of the next data record should be 8030H (Page 0).

Ex)
:10802000202122232425262728292A2B2C2D2E2FD8   ' 8020H to 802FH data
:108030003031323334353637383930A3B3C3D3E3FC8   ' 8030H to 803FH data

3. The last data's address of data record immediately before sending the end record should be the last address of page. For example, in case of page 1, the last data's address of data record should be 807FH.

Ex)
:10807000303132333435363738393A3B3C3D3E3F88   ' 8070H to 807FH data
:00000001FF   ' End record

Note: Do not write only the addresses from FFE0H to FFFFH when all data of FLASH memory are the same data. If these area are only written, the next operation can not be executed because of password error.
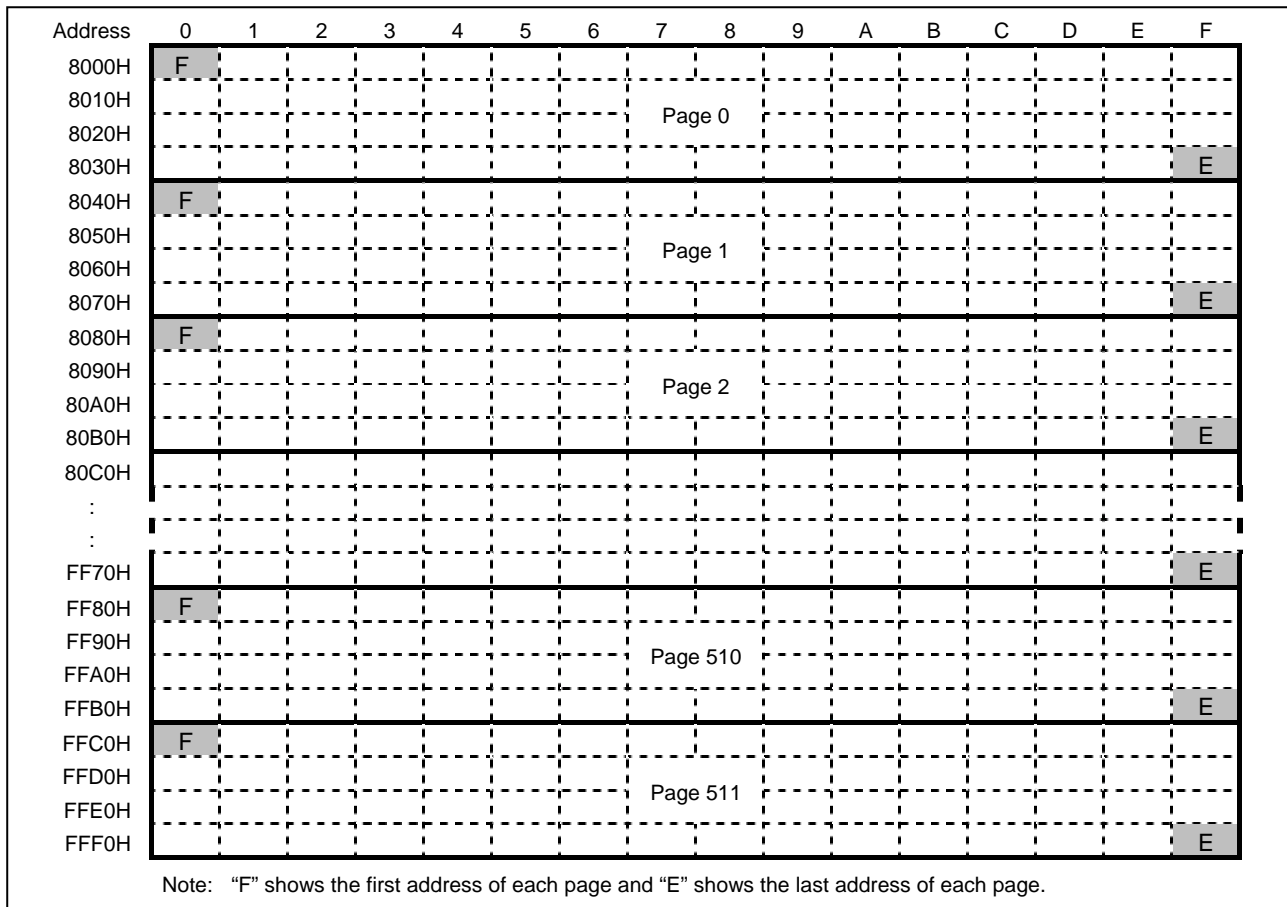

Figure 2.19.4 Organization of FLASH Area

2.19.8 Error Code

When the device detects an error, the error codes are sent to the controller.

Table 2.19.10 Error Code

Transmit Data	Meaning of Transmit Data
62H, 62H, 62H	Baud rate modification error occurred.
63H, 63H, 63H	Operating command error occurred.
A1H, A1H, A1H	Framing error in received data occurred.
A3H, A3H, A3H	Overrun error in received data occurred.

Note1: If password error occurs, the TMP86FM48 doesn't send error codes.

2.19.9 Checksum (SUM)

（1） Calculation method

SUM consists of byte + byte... + byte, the checksum of which is returned in word as the result.

Namely, data is read out in byte and checksum of which is calculated, with the result returned in word.

Example:

A1H
B2H
C3H
D4H

If the data to be calculated consists of the four bytes shown to the left, SUM of the data is

A1H + B2H + C3H + D4H = 02EAH
    SUM (HIGH)   = 02H
    SUM (LOW)   = EAH

The SUM returned when executing the FLASH memory write command, RAM loader command, or FLASH memory SUM command is calculated in the manner shown above.

（2） Calculation data

The data from which SUM is calculated are listed in Table 2.19.11 below.

Table 2.19.11 Checksum Calculation Data

Operating Mode	Calculation Data	Remarks
FLASH memory writing mode	Data in the entire area (32 Kbytes) of FLASH memory	Even when written to part of the FLASH area, data in the entire memory area (32 Kbytes) is calculated.
FLASH memory SUM output mode		The length of data, address, record type and checksum in Intel Hex format are not included in SUM.
RAM loader mode	Data written to RAM	The length of data, address, record type and checksum in Intel Hex format are not included in SUM.
Product Discrimination Code Output mode	Checksum of transferred data (from 9th to 18th byte)	For details, refer to 2.19.12 Product Discrimination Code.

2.19.10   Intel Hex Format (Binary)

1. After receiving the checksum of a record, the device waits for the start mark data (3AH for ":") of the next record. Therefore, the device ignores the data, which does not match the start mark data after receiving the checksum of a record.

2. Make sure that once the controller program has finished sending the checksum of the end record, it does not send anything and waits for two bytes of data to be received (Upper and lower bytes of checksum). This is because after receiving the checksum of the end record, the boot program calculates the checksum and returns the calculated checksum in two bytes to the controller.

3. If a receive error or Intel Hex format error occurs, the UART function of TMP86FM48 stops without returning error code to the controller. In the following cases, an Intel Hex format error occurs:

    • When the record type is not 00H, 01H, or 02H

    • When a SUM error occurred

    • When the data length of an extended record (Type = 02H) is not 02H

    • When the address of an extended record (Type = 02H) is larger than 1000H and after that, receives the data record

    • When the data length of the end record (Type = 01H) is not 00H

2.19.11   Passwords

The eight or more bytes consecutive data in flash memory area can be used as password. In password check, TMP86FM48 compares these data with data which are transmitted from the external controller. The area in which passwords can be specified is located at addresses 8000H to FF9FH. The area from FFA0H to FFFFH can not be specified as passwords area. The device compares the stored passwords with the passwords, which are received from the controller. If all data of addresses from FFE0H to FFFFH are "00H" or "FFH", the passwords comparison is not executed because the device is considered as blank product. It is necessary to specify the password count storage addresses and the password comparison start address even though it is a blank product. Table 2.19.12 shows the password setting in the blank product and non blank product.

Table 2.19.12 Password Setting in the Blank Product and Non Blank Product

Password	Blank Product (Note 1)	Non Blank Product
PNSA (Password count storage addresses)	8000H ≤ PNSA ≤ FF9FH	8000H ≤ PNSA ≤ FF9FH
PCSA (Password comparison start address)	8000H ≤ PCSA ≤ FF9FH	8000H ≤ PCSA ≤ FFA0 − N
N (Password count)	∗	8 ≤ N
Setting of password	No need	Need (Note 2)

Note 1: When all data of addresses from FFE0H to FFFFH area are "00H" or "FFH", the device is judged as blank product.

Note 2: The same three or more bytes consecutive data can not be used as password. When the password includes the same consecutive data (three or more bytes), the password error occurs. If the password error occurred, the UART function of device stops without returning error code.

Note 3: ∗: Don't care.

Note 4: When the password doesn't match the above condition, the password error occurs. If the password error occurred, the UART function of device stops without returning error code.

2.19.11.1 Confirmation Method of the Blank Product and Non Blank Product

The external controller can confirm whether the device is the blank product or not, by transmission of data described below.

(1) Executes FLASH memory writing mode or RAM loader mode.
(2) Transmits the PNSA and PCSA.
(3) Transmits the end record.
(4) In case of the blank product, the device sends checksum of flash memory. In case of the non blank product, the device doesn't send checksum of flash memory but the UART function stops without sending any data.

The external controller can confirm the blank product and non blank product by receiving checksum.

Note: When the UART function stops in non blank product, the TMP86FM48 should be reset by pin reset input for restarting the Serial PROM Mode.

2.19.11.2 Password String

www.DataSheet4U.com A string of passwords in the received data are compared with the data in the FLASH memory. In the following cases, a password error occurs:

- When the received data does not match the data in the FLASH memory
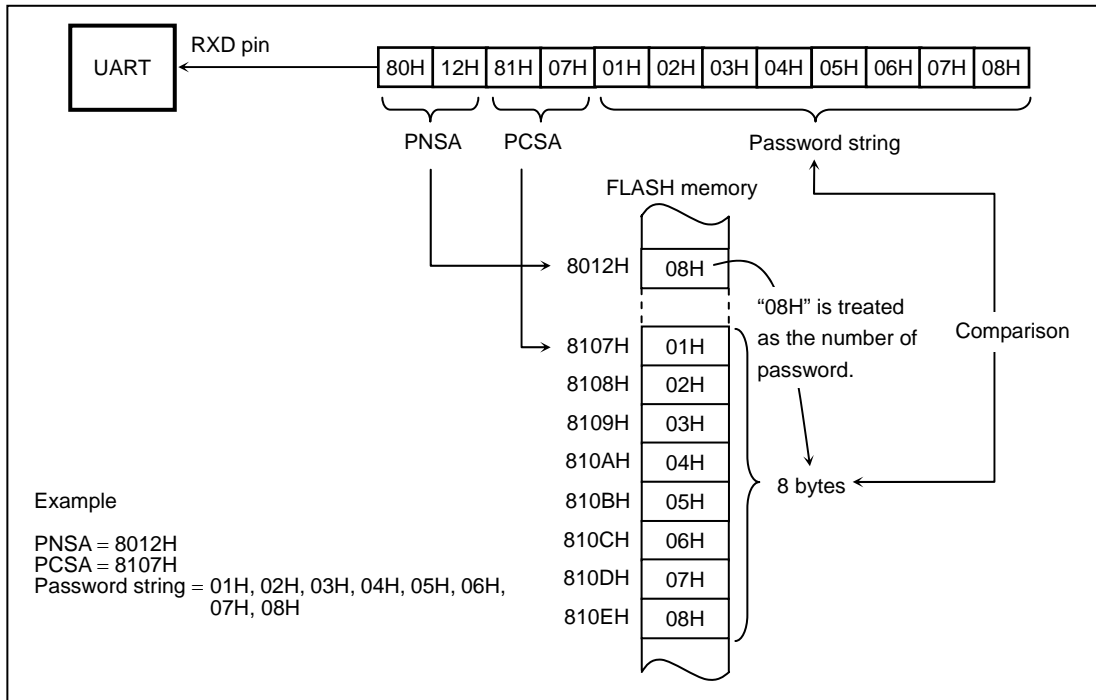
2.19.11.3 Handling of Password Error

If a password error occurs, the UART function of TMP86FM48 stops without returning error code to the controller. Therefore, when a password error occurs, the TMP86FM48 should be reset by $\overline{\text{RESET}}$ pin input.

### 2.19.12　　　Product Discrimination Code

The product discrimination code is a 13-byte data, that includes the start address and the end address of ROM. Table 2.19.13 shows the product discrimination code format.

Table 2.19.13 Product Discrimination Code Format

| Data	The Meaning of Data	In Case of TMP86FM48
1st	Start Mark (3AH)	3AH
2nd	The number of transfer data (from 3rd to 12th byte)	0AH
3rd	Length of address	02H
4th	Reserved data	00H
5th	Reserved data	00H
6th	Reserved data	00H
7th	Reserved data	00H
8th	The number of ROM block	01H
9th	The upper byte of the first address of ROM	80H
10th	The lower byte of the first address of ROM	00H
11th	The upper byte of the end address of ROM	FFH
12th	The lower byte of the end address of ROM	FFH
13th	Checksum of transferred data (from 3rd to 12th byte)	7FH

2.19.13   Flowchart
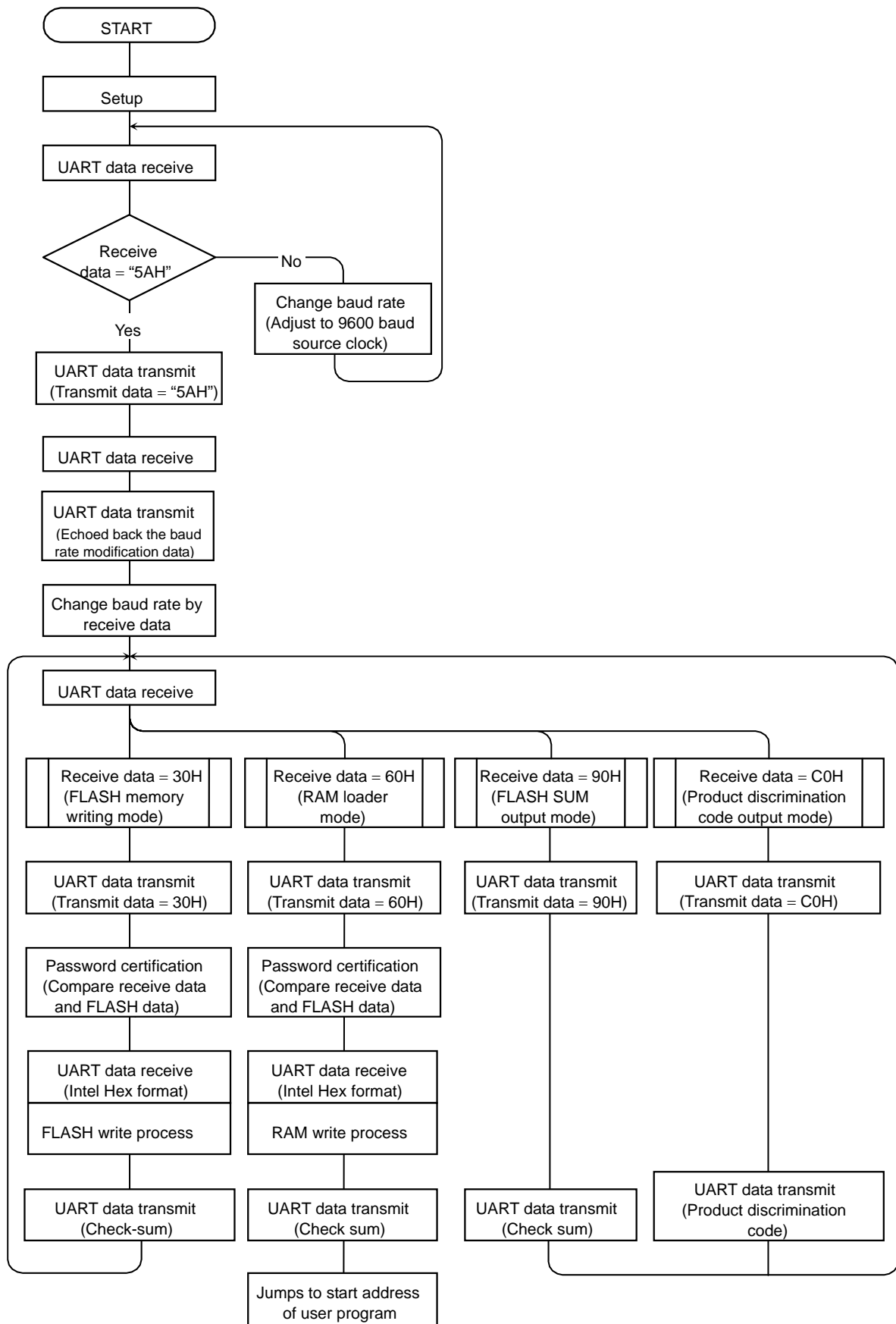
2.19.14   UART Timing

Table 2.19.14 UART Timing-1 (VDD = 2.7 V to 3.6 V, fc = 2 MHz to 16 MHz, Ta = 25°C)

Parameter	Symbol	The Number of Clock (fc)	Required Minimum Time	
			At fc = 2 MHz	At fc = 16 MHz
Time from the reception of a matching data until the output of an echo back	CMeb1	Approx. 600	300 µs	37.5 µs
Time from the reception of a baud rate modification data until the output of an echo back	CMeb2	Approx. 700	350 µs	43.7 µs
Time from the reception of an operation command until the output of an echo back	CMeb3	Approx. 600	300 µs	37.5 µs
Calculation time of checksum	CKsm	Approx. 1573000	786.5 ms	98.3 ms

Table 2.19.15 UART Timing-2 (VDD = 2.7 V to 3.6 V, fc = 2 MHz to 16 MHz, Ta = 25°C)

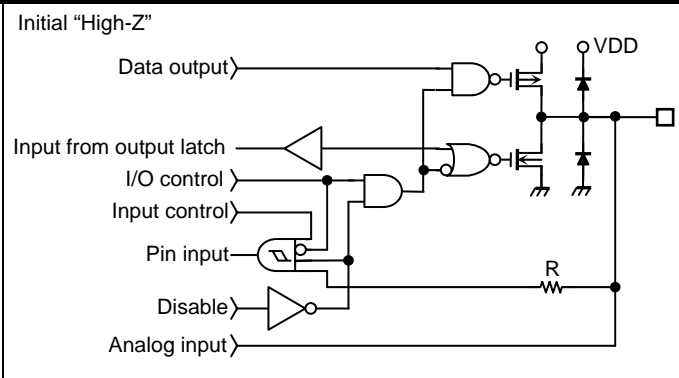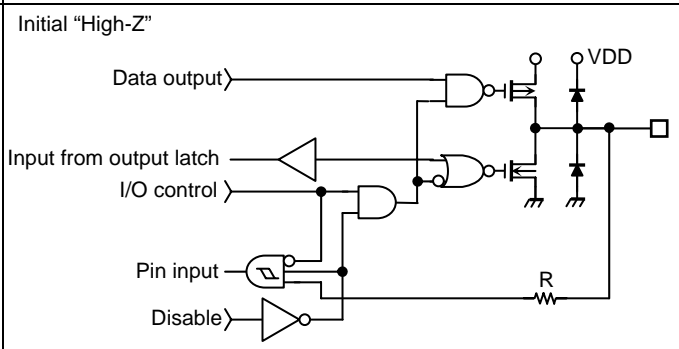| Parameter	Symbol	The Number of Clock (fc)	Required Minimum Time	
			At fc = 2 MHz	At fc = 16 MHz
Time from reset release until acceptance of start bit of RXD pin	RXsup	110000	55 ms	6.9 ms
Time between a matching data and the next matching data	CMtr1	28500	14.3 ms	1.8 ms
Time from the echo back of matching data until the acceptance of baud rate modification data	CMtr2	600	300 µs	37.5 µs
Time from the output of echo back of baud rate modification data until the acceptance of an operation command	CMtr3	750	375 µs	46.9 µs
Time from the output of echo back of operation command until the acceptance of Password count storage addresses	CMtr4	950	475 µs	59.4 µs
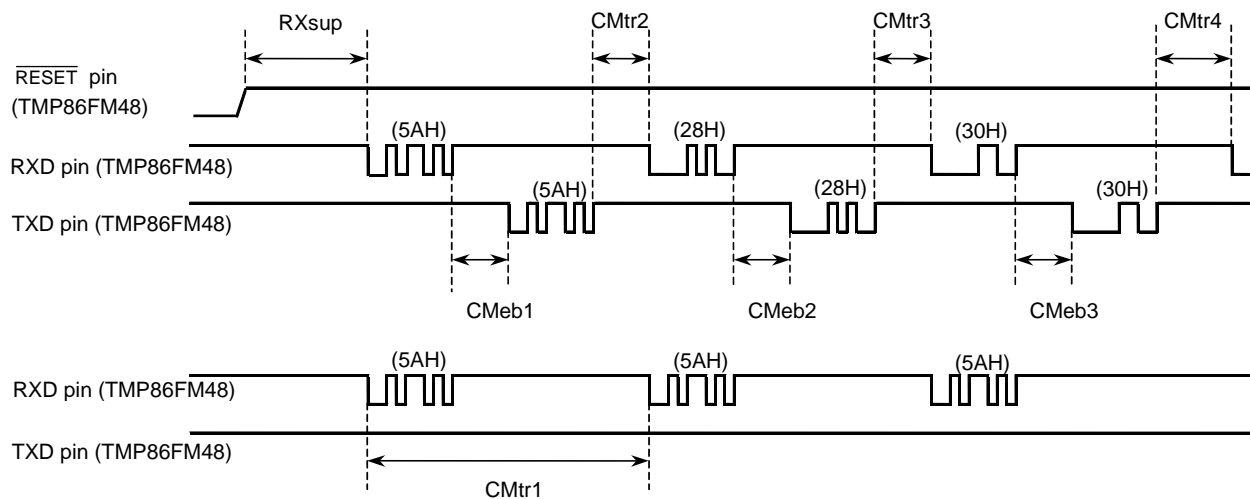
Input/Output Circuitry

(1) Control pins

The input/output circuitries of the TMP86FM48 control pins are shown below.

Control Pin	I/O	Input/Output Circuitry	Remarks
XIN XOUT	Input Output	Osc. enable → fc VDD R_f R_O VDD XIN XOUT	Resonator connecting pins (High frequency) R_f = 3 MΩ (typ.) R_O = 0.5 kΩ (typ.)
XTIN XTOUT	Input Output	NORMAL1 mode: Refer to port P2 NORMAL2 mode: Osc. enable → XTEN fc VDD R R_f R_O VDD XIN XOUT	Resonator connecting pins (Low frequency) R_f = 20 MΩ (typ.) R_O = 220 kΩ (typ.)
$\overline{\text{RESET}}$	Input	VDD R_{IN} R Address-trap-reset → Watchdog-timer → System-clock-reset →	Hysteresis input Pull-up resistor R_{IN} = 220 kΩ (typ.) R = 100 Ω (typ.)
TEST	Input	VDD R R_{IN}	Pull-down resistor R_{IN} = 70 kΩ (typ.) R = 100 Ω (typ.)
BOOT	Input	VDD R R_{IN}	Pull-down resistor R_{IN} = 70 kΩ (typ.) R = 100 Ω (typ.)

(2) Input/output ports

Port	I/O	Input/Output Circuitry	Remarks
P0 P1	I/O	Initial "High-Z" Pch control (Control output) Data output Input from output latch Disable Pin input (Control input) VDD R	Sink open drain output or CMOS output Hysteresis input R = 100 Ω (typ.)
P20	I/O	Initial "High-Z" Pch control Data output Input from output latch Disable Pin input (Control input) VDD R	Sink open drain output or CMOS output Hysteresis input R = 100 Ω (typ.)
P21 P22	I/O	Initial "High-Z" Pull-up resistor Resistor control Data output Input from output latch Disable Pin input VDD R_{IN3} R	Sink open drain output or CMOS output Hysteresis input Programmable pull-up resistor R_{IN3} = 220 kΩ (typ.)
P3	I/O	Initial "High-Z" Pch control Data output Input from output latch Disable Pin input VDD R	Sink open drain or CMOS output Hysteresis input High current output (Nch) R = 100 Ω (typ.)
P5	I/O	Initial "High-Z" Pch control (Control output) Data output Input from output latch Disable Pin input VDD R	Sink open drain output or CMOS output Hysteresis input High current output (Nch) R = 100 Ω (typ.)

Port	I/O	Input/Output Circuitry	Remarks
P6 P7	I/O	Initial "High-Z" Data output Input from output latch I/O control Input control Pin input Disable Analog input VDD R	Tri-state I/O Hysteresis input R = 100 Ω (typ.)
P8	I/O	Initial "High-Z" Data output Input from output latch I/O control Pin input Disable VDD R	Tri-state I/O Hysteresis input R = 100 Ω (typ.)

Electrical Characteristics

Absolute Maximum Ratings $(V_{SS} = 0\ V)$

Parameter	Symbol	Pins	Rating	Unit
Supply voltage	V_{DD}		-0.3 to 4.0	V
Input voltage	V_{IN}		-0.3 to $V_{DD} + 0.3$	
Output voltage	V_{OUT1}		-0.3 to $V_{DD} + 0.3$	
Output current (Per 1 pin)	I_{OUT1}	P0, P1, P20, P3, P5, P6, P7, P8 ports	-2	mA
	I_{OUT2}	P0, P1, P2, P4, P6, P7, P8 ports	2	
	I_{OUT3}	P3, P5 ports	10	
Output current (Total)	ΣI_{OUT1}	P0, P1, P20, P3, P5, P6, P7, P8 ports	-30	
	ΣI_{OUT2}	P0, P1, P2, P4, P6, P7, P8 ports	80	
	ΣI_{OUT3}	P3, P5 ports	30	
Power dissipation [Topr $= 85°C$]	PD		350	mW
Soldering temperature (Time)	Tsld		$260\ (10\ s)$	°C
Storage temperature	Tstg		-55 to 125	
Operating temperature	Topr		-40 to 85	

Note: The absolute 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 absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded.

Recommended Operating Condition-1 (MCU mode)   (V_{SS} = 0 V, Topr = −40 to 85°C)

Parameter	Symbol	Pins	Condition		Min	Max	Unit
Supply voltage	V_{DD}		fc = 16 MHz	NORMAL1, 2 mode	2.7	3.6	V
				IDLE0, 1, 2 mode			
			fc = 8 MHz (In case of connecting the resonator)	NORMAL1, 2 mode	1.8		
				IDLE0, 1, 2 mode			
			fc = 4.2 MHz (In case of external clock input)	NORMAL1, 2 mode			
				IDLE0, 1, 2 mode			
			fs = 32.768 kHz	SLOW1, 2 mode	1.8		
				SLEEP0, 1, 2 mode			
				STOP mode			
Input high level	V_{IH1}	Except Hysteresis input	$V_{DD} \geq 2.7$ V		$V_{DD} \times 0.70$	V_{DD}	
	V_{IH2}	Hysteresis input			$V_{DD} \times 0.75$		
	V_{IH3}		$V_{DD} < 2.7$ V		$V_{DD} \times 0.90$		
Input low level	V_{IL1}	Except Hysteresis input	$V_{DD} \geq 2.7$ V			$V_{DD} \times 0.30$	
	V_{IL2}	Hysteresis input			0	$V_{DD} \times 0.25$	
	V_{IL3}		$V_{DD} < 2.7$ V			$V_{DD} \times 0.10$	
Clock frequency (In case of connecting the resonator)	fc	XIN, XOUT	V_{DD} = 1.8 to 3.6 V		1.0	8.0	MHz
			V_{DD} = 2.7 to 3.6 V			16.0	
	fs	XTIN, XTOUT	V_{DD} = 1.8 to 3.6 V		30.0	34.0	kHz
Clock frequency (In case of external clock input)	fc	XIN, XOUT	V_{DD} = 1.8 to 3.6 V		1.0	4.2	MHz
			V_{DD} = 2.7 to 3.6 V			16.0	
	fs	XTIN, XTOUT	V_{DD} = 1.8 to 3.6 V		30.0	34.0	kHz

Note: The recommended operating conditions for a device are operating conditions under which it can be guaranteed that the device will operate as specified. If the device is used under operating conditions other than the recommended operating conditions (Supply voltage, operating temperature range, specified AC/DC values etc.), malfunction may occur. Thus, when designing products which include this device, ensure that the recommended operating conditions for the device are always adhered to.

Recommended Operating Condition-2 (Serial PROM mode)   (V_{SS} = 0 V, Topr = 25°C ± 5°C)

Parameter	Symbol	Pins	Condition	Min	Max	Unit
Supply voltage	VDD		2 MHz ≤ fc ≤ 16 MHz	2.7	3.6	V
Clock frequency	fc	XIN, XOUT	VDD = 2.7 to 3.6 V	2.0	16.0	MHz

Note: The operating temperature area of serial PROM mode is 25°C ± 5°C and the operating area of high frequency of serial PROM mode is different from MCU mode.

DC Characteristics (V$_{SS}$ = 0 V, Topr = −40 to 85°C)

Parameter	Symbol	Pins			Condition		Min	Typ.	Max	Unit
Hysteresis voltage	V$_{HS}$	Hysteresis input			V$_{DD}$ = 3.3 V		–	0.4	–	V
Input current	I$_{IN1}$	TEST			V$_{DD}$ = 3.6 V, V$_{IN}$ = 0 V		–	–	−5	μA
	I$_{IN2}$	Sink open drain, Tri-state			V$_{DD}$ = 3.6 V, V$_{IN}$ = 3.6 V/0 V		–	–	±5	
	I$_{IN3}$	$\overline{RESET}$			V$_{DD}$ = 3.6 V, V$_{IN}$ = 3.6 V		–	–	+5	
Input resistance	R$_{IN1}$	TEST pull down			V$_{DD}$ = 3.6 V, V$_{IN}$ = 3.6 V		–	70	–	kΩ
	R$_{IN2}$	BOOT pull down			V$_{DD}$ = 3.6 V, V$_{IN}$ = 3.6 V		–	70	–	
	R$_{IN3}$	$\overline{RESET}$ pull up P21, P22 ports			V$_{DD}$ = 3.6 V, V$_{IN}$ = 0 V		100	220	450	
High frequency feedback resistor	R$_{FB}$	XOUT			V$_{DD}$ = 3.6 V		–	3	–	MΩ
Low frequency feedback resistor	R$_{FBT}$	XTOUT			V$_{DD}$ = 3.6 V		–	20	–	
Output leakage current	I$_{LO}$	Sink open drain, Tri-state			V$_{DD}$ = 3.6 V V$_{OUT}$ = 3.4V/0.2 V		–	–	±10	μA
Output high voltage	V$_{OH}$	CMOS, Tri-state			V$_{DD}$ = 3.6 V, I$_{OH}$ = −0.6 mA		3.2	–	–	V
Output low voltage	V$_{OL}$	Except XOUT, P3 and P5 ports			V$_{DD}$ = 3.6 V, I$_{OL}$ = 0.9 mA		–	–	0.4	
Output low current	I$_{OL}$	P3, P5 ports			V$_{DD}$ = 3.6 V, V$_{OL}$ = 1.0 V		–	6	–	mA
Supply current in NORMAL 1, 2 mode	I$_{DD}$	Fetch area	Flash area	V$_{DD}$ = 3.6 V V$_{IN}$ = 3.4 V/0.2 V fc = 16 MHz fs = 32.768 kHz	MNP = "1"	–	5.0	6.0	mA
			RAM area		MNP = "0"	–	3.5	4.8	
Supply current in IDLE 0, 1, 2 mode					MNP•ATP = "1"	–	3.5	4.5	
					MNP•ATP = "0"	–	2.5	3.7	
Supply current in SLOW 1 mode		Fetch area	Flash area	V$_{DD}$ = 3.0 V V$_{IN}$ = 2.8 V/0.2 V fs = 32.768 kHz	MNP = "1"	–	800	1400	μA
			RAM area		MNP = "0"	–	6	20	
Supply current in SLEEP 1 mode					MNP•ATP = "1"	–	800	1400	
					MNP•ATP = "0"	–	5	18	
Supply current in SLEEP 0 mode					MNP•ATP = "1"	–	800	1400	
					MNP•ATP = "0"	–	5	18	
Supply current in STOP mode				V$_{DD}$ = 3.6 V V$_{IN}$ = 3.4 V/0.2 V		–	0.5	10	

Note 1: Typical values show those at Topr = 25°C.

Note 2: Input current (I$_{IN1}$, I$_{IN2}$): The current through pull-up or pull-down resistor is not included.

Note 3: I$_{DD}$ does not include I$_{REF}$ current.

Note 4: The supply currents of SLOW2 and SLEEP2 modes are equivalent to IDLE0, IDLE1, IDLE2.

Note 5: MNP (MNPWDW) shows bit0 in EEPCR register and ATP (ATPWDW) shows bit1 in EEPCR register.

Note 6: "Fetch" means reading operation of FLASH data as an instruction by CPU.

AD Conversion Characteristics (V_{SS} = 0.0 V, 2.7 V ≤ V_{DD} ≤ 3.6 V, Topr = −40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage	V_{AREF}		A_{VDD} − 1.0	−	A_{VDD}	
Power supply voltage of analog control circuit	A_{VDD}			V_{DD}		V
Analog reference voltage range (Note 4)	ΔV_{AREF}		2.5	−	−	
Analog input voltage	V_{AIN}		V_{SS}	−	V_{AREF}	
Power supply current of analog reference voltage	I_{REF}	V_{DD} = A_{VDD} = V_{AREF} = 3.6 V, V_{SS} = 0.0 V	−	0.35	0.61	mA
Non linearity error		V_{DD} = A_{VDD} = 2.7 V, V_{SS} = 0.0 V, V_{AREF} = 2.7 V	−	−	±2	LSB
Zero point error			−	−	±2	
Full scale error			−	−	±2	
Total error			−	−	±2	

(V_{SS} = 0.0 V, 2.0 V ≤ V_{DD} < 2.7 V, Topr = −40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage	V_{AREF}		A_{VDD} − 0.6	−	A_{VDD}	
Power supply voltage of analog control circuit	A_{VDD}			V_{DD}		V
Analog reference voltage range (Note 4)	ΔV_{AREF}		2.0	−	−	
Analog input voltage	V_{AIN}		V_{SS}	−	V_{AREF}	
Power supply current of analog reference voltage	I_{REF}	V_{DD} = A_{VDD} = V_{AREF} = 2.0V, V_{SS} = 0.0 V	−	0.20	0.34	mA
Non linearity error		V_{DD} = A_{VDD} = 2.0 V, V_{SS} = 0.0 V, V_{AREF} = 2.0 V	−	−	±4	LSB
Zero point error			−	−	±4	
Full scale error			−	−	±4	
Total error			−	−	±4	

(V_{SS} = 0.0 V, 1.8 V ≤ V_{DD} < 2.0 V, Topr = −10 to 85°C) (Note 5)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Analog reference voltage	V_{AREF}		A_{VDD} − 0.1	−	A_{VDD}	
Power supply voltage of analog control circuit	A_{VDD}			V_{DD}		V
Analog reference voltage range (Note 4)	ΔV_{AREF}		1.8	−	−	
Analog input voltage	V_{AIN}		V_{SS}	−	V_{AREF}	
Power supply current of analog reference voltage	I_{REF}	V_{DD} = A_{VDD} = V_{AREF} = 1.8 V, V_{SS} = 0.0 V	−	0.18	0.31	mA
Non linearity error		V_{DD} = A_{VDD} = 1.8 V, V_{SS} = 0.0 V, V_{AREF} = 1.8 V	−	−	±4	LSB
Zero point error			−	−	±4	
Full scale error			−	−	±4	
Total error			−	−	±4	

Note 1: The total error includes all errors except a quantization error, and is defined as a maximum deviation from the ideal conversion line.

Note 2: Conversion time is different in recommended value by power supply voltage. About conversion time, please refer to "2.15.2 Register configration".

Note 3: Please use input voltage to AIN input Pin in limit of VAREF − VSS. When voltage of range outside is input, conversion value becomes unsettled and gives affect to other channel conversion value.

Note 4: Analog Reference Voltage Range: ΔVAREF = VAREF − VSS

Note 5: When AD is used with VDD < 2.0 V, the guaranteed temperature range varies with the operating voltage.

Note 6: When AD converter is not used, fix the AVDD pin and VAREFpin on the V_{DD} level.

| AC Characteristics | | (V_{SS} = 0 V, V_{DD} = 2.7 to 3.6 V, Topr = −40 to 85°C) | | | | | |

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Machine cycle time	tcy	NORMAL1, 2 mode	0.25	–	4	μs
		IDLE1, 2 mode				
		SLOW1, 2 mode	117.6	–	133.3	
		SLEEP1, 2 mode				
High Level clock pulse width	twcH	For external clock operation (XIN input), fc = 16 MHz	–	31.25	–	ns
Low level clock pulse width	twcL					
High level clock pulse width	twcH	For external clock operation (XTIN input), fs = 32.768 kHz	–	15.26	–	μs
Low level clock pulse width	twcL					

(V_{SS} = 0 V, V_{DD} = 1.8 to 3.6 V, Topr = −40 to 85°C)

Parameter	Symbol	Condition	Min	Typ.	Max	Unit
Machine cycle time	tcy	NORMAL1, 2 mode	0.5	–	4	μs
		IDLE1, 2 mode				
		SLOW1, 2 mode	117.6	–	133.3	
		SLEEP1, 2 mode				
High level clock pulse width	twcH	For external clock operation (XIN input), fc = 4.2 MHz	–	119.04	–	ns
Low level clock pulse width	twcL					
High level clock pulse width	twcH	For external clock operation (XTIN input), fs = 32.768 kHz	–	15.26	–	μs
Low level clock pulse width	twcL					

| Flash Characteristics | | (V_{SS} = 0 V) | | | | |

Parameter	Condition	Min	Typ.	Max	Unit
Number of guaranteed writes (page writing) to Flash memory in serial PROM mode	V_{DD} = 2.7 to 3.6 V, 2 MHz ≤ fc ≤ 16 MHz (Topr = 25°C ± 5°C)	–	–	10^5	Times
Number of guaranteed writes (page writing) to Flash data memory in MCU mode	V_{DD} = 1.8 to 3.6 V at fc = 8 MHz V_{DD} = 2.7 to 3.6 V at fc = 16 MHz (Topr = −40 to 85°C)	–	–	10^5	
Writing time to Flash data memory for one page (64 bytes) in MCU mode		–	4	6	ms

Recommended Oscillating Conditions

Note 1: An electrical shield by metal shield plate on the surface of IC package is recommended in order to protect the device from the high electric field stress applied from CRT (Cathodic Ray Tube) for continuous reliable operation.

Note 2: The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following http://www.murata.com/

Handling Precaution

- The solderability test conditions for lead-free products (indicated by the suffix G in product name) are shown below.

    1. When using the Sn-37Pb solder bath
       Solder bath temperature = 230°C
       Dipping time = 5 seconds
       Number of times = once
       R-type flux used

    2. When using the Sn-3.0Ag-0.5Cu solder bath
       Solder bath temperature = 245°C
       Dipping time = 5 seconds
       Number of times = once
       R-type flux used

        Note : The pass criteron of the above test is as follows: Solderability rate until forming ≥ 95%
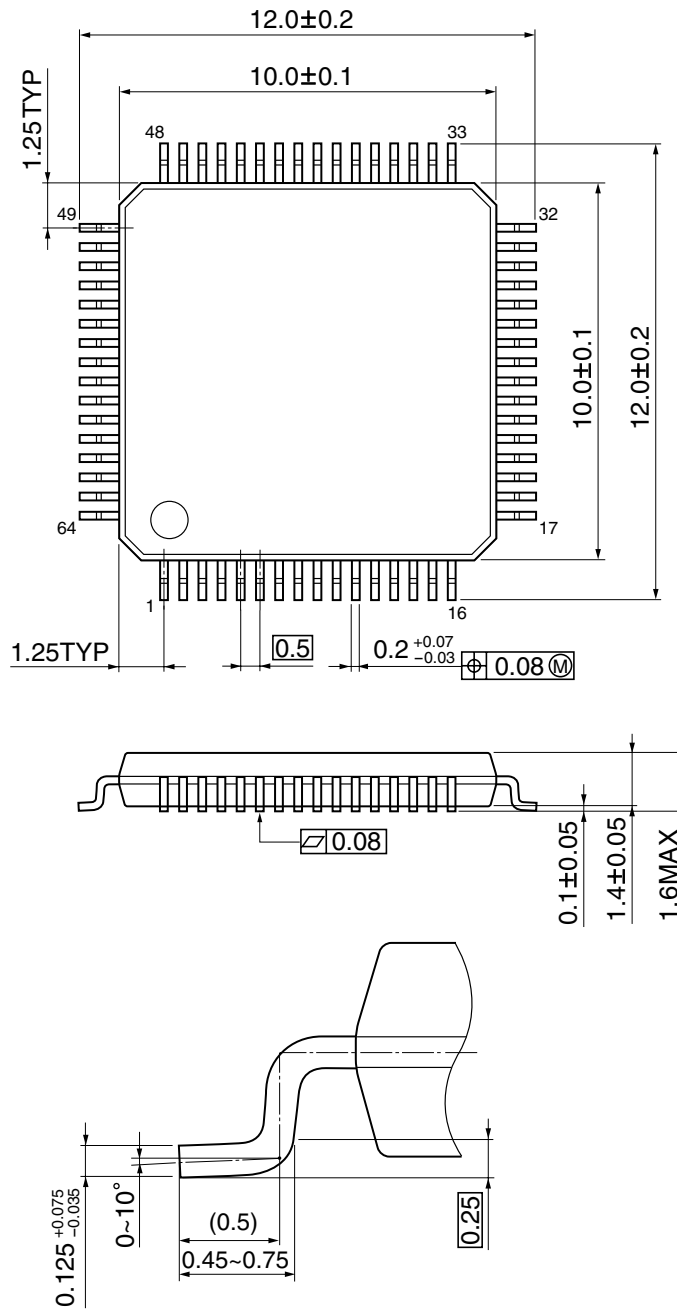
- When using the device (oscillator) in places exposed to high electric fields such as cathode-ray tubes, we recommend electrically shielding the package in order to maintain normal operating condition.

Package Dimensions

LQFP64-P-1010-0.50E

Unit: mm

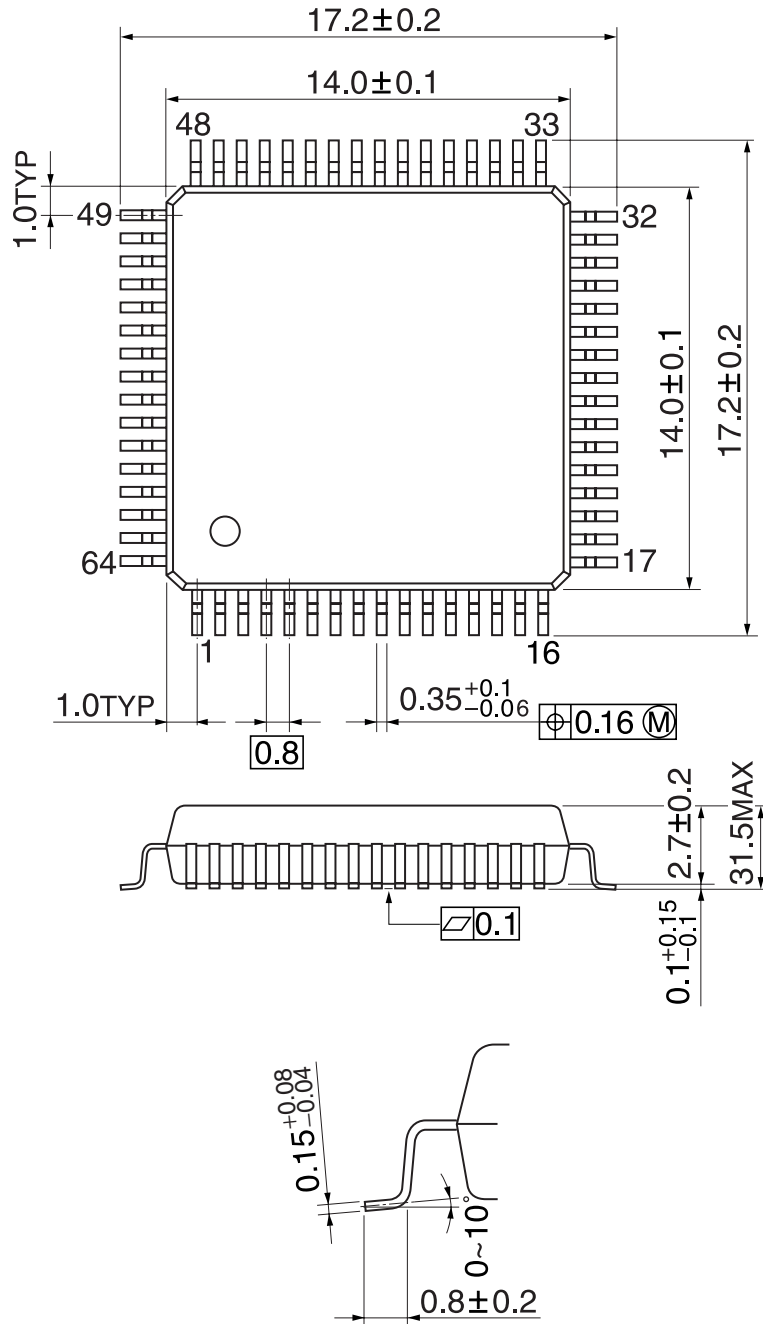QFP64-P-1414-0.80C

Unit: mm

17.2±0.2

14.0±0.1

48   33

1.0TYP

49   32

14.0±0.1

17.2±0.2

64   17

1   16

1.0TYP

0.35 $^{+0.1}_{-0.06}$

0.8

⊕ 0.16 Ⓜ

2.7±0.2

31.5MAX

0.1 $^{+0.15}_{-0.1}$

▱ 0.1

0.15 $^{+0.08}_{-0.04}$

0~10°

0.8±0.2

This is a technical document that describes the operating functions and electrical specifications of the 8-bit microcontroller series TLCS-870/C (LSI).

Toshiba provides a variety of development tools and basic software to enable efficient software development.

These development tools have specifications that support advances in microcomputer hardware (LSI) and can be used extensively. Both the hardware and software are supported continuously with version updates.

The recent advances in CMOS LSI production technology have been phenomenal and microcomputer systems for LSI design are constantly being improved. The products described in this document may also be revised in the future. Be sure to check the latest specifications before using.

Toshiba is developing highly integrated, high-performance microcomputers using advanced MOS production technology and especially well proven CMOS technology.

We are prepared to meet the requests for custom packaging for a variety of application areas.
We are confident that our products can satisfy your application needs now and in the future.