



**THE DATASHEET OF  
TMP86FH09NG**



**TOSHIBA**

8 Bit Microcontroller  
TLCS-870/C Series

**TMP86FH09NG**

November 10, 2005

**TOSHIBA CORPORATION**

The information contained herein is subject to change without notice.

The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others.

TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress.

It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.

In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications.

Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc..

The Toshiba products listed in this document are intended for usage in general electronics applications ( computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).

These Toshiba products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury ("Unintended Usage"). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc.. Unintended Usage of Toshiba products listed in this document shall be made at the customer's own risk.

The products described in this document may include products subject to the foreign exchange and foreign trade laws.

TOSHIBA products should not be embedded to the downstream products which are prohibited to be produced and sold, under any law and regulations.

For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance/Handling Precautions.

## CMOS 8-Bit Microcontroller

## TMP86FH09NG

The TMP86FH09 is a high-speed, high-performance 8-bit microcomputer built around the TLCS-870/C Series core with built-in 16-Kbyte flash memory and it is pin compatible with its mask ROM version, the TMP86CH09/809. Writing programs in the built-in flash memory enables this microcomputer to perform the same operations as the TMP86C809/H09. The built-in flash memory can be rewritten on board (without removing it from the PCB) by a built-in boot program.

| Product No. | ROM(FLASH)  | RAM       | PACKAGE           |
|-------------|-------------|-----------|-------------------|
| TMP86FH09NG | 16384 bytes | 512 bytes | P-SDIP32-400-1.78 |

## 1.1 Features

1. 8-bit single chip microcomputer TLCS-870/C series
  - Instruction execution time :
    - 0.25  $\mu$ s (at 16 MHz)
    - 122  $\mu$ s (at 32.768 kHz)
  - 132 types & 731 basic instructions
2. 18 interrupt sources (External : 5 Internal : 13)
3. Input / Output ports (26pins)
  - Large current output: 8pins (Typ. 20mA)
4. Time Base Timer
5. Watchdog Timer
6. 16-bit timer counter : 1ch
  - Timer, External trigger, Window, Pulse width measurement, Event counter, PPG(Programmable Pulse Generator) output modes
7. 8-bit timer counter : 2ch
  - Timer, Event counter, PWM(Pulse width modulation) output, PDO(Programmable Divider Output) output and PPG(Programmable Pulse Generator) modes
8. Divider output function
9. SEI : 1ch
10. UART : 1ch

This product uses the Super Flash® technology under the licence of Silicon Storage Technology, Inc. Super Flash® is registered trademark of Silicon Storage Technology, Inc.

030619EBP1

- The information contained herein is subject to change without notice.
- The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others.
- TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property. In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc..
- The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.). These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury ( "Unintended Usage" ). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc.. Unintended Usage of TOSHIBA products listed in this document shall be made at the customer's own risk.
- The products described in this document are subject to the foreign exchange and foreign trade laws.
- TOSHIBA products should not be embedded to the downstream products which are prohibited to be produced and sold, under any law and regulations.
- For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance/Handling Precautions.

11. 10-bit successive approximation type AD converter
  - Analog inputs: 6ch
12. Key On Wake Up : 4ch
13. Clock operation
  - Single clock mode
  - Dual clock mode
14. Low power consumption operation
  - STOP mode: Oscillation stops. (Battery/Capacitor back-up.)
  - SLOW1 mode: Low power consumption operation using low-frequency clock.(High-frequency clock stop.)
  - SLOW2 mode: Low power consumption operation using low-frequency clock.(High-frequency clock oscillate.)
  - IDLE0 mode: CPU stops, and only the Time-Based-Timer(TBT) on peripherals operate using high frequency clock. Release by falling edge of the source clock which is set by TBTCR<TBTCK>.
  - IDLE1 mode: CPU stops and peripherals operate using high frequency clock. Release by interrupts(CPU restarts).
  - IDLE2 mode: CPU stops and peripherals operate using high and low frequency clock. Release by interrupts. (CPU restarts).
  - SLEEP0 mode: CPU stops, and only the Time-Based-Timer(TBT) on peripherals operate using low frequency clock.Release by falling edge of the source clock which is set by TBTCR<TBTCK>.
  - SLEEP1 mode: CPU stops, and peripherals operate using low frequency clock. Release by interrupt.(CPU restarts).
  - SLEEP2 mode: CPU stops and peripherals operate using high and low frequency clock. Release by interrupt.

1.2 Pin Assignment

P-SDIP32-400-1.78

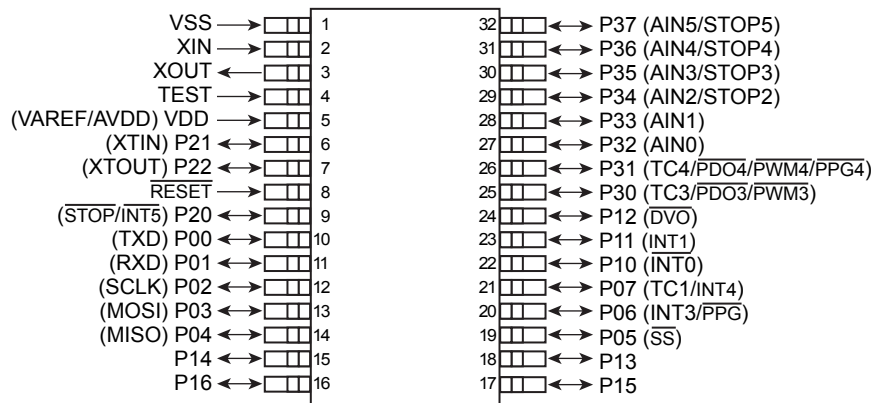
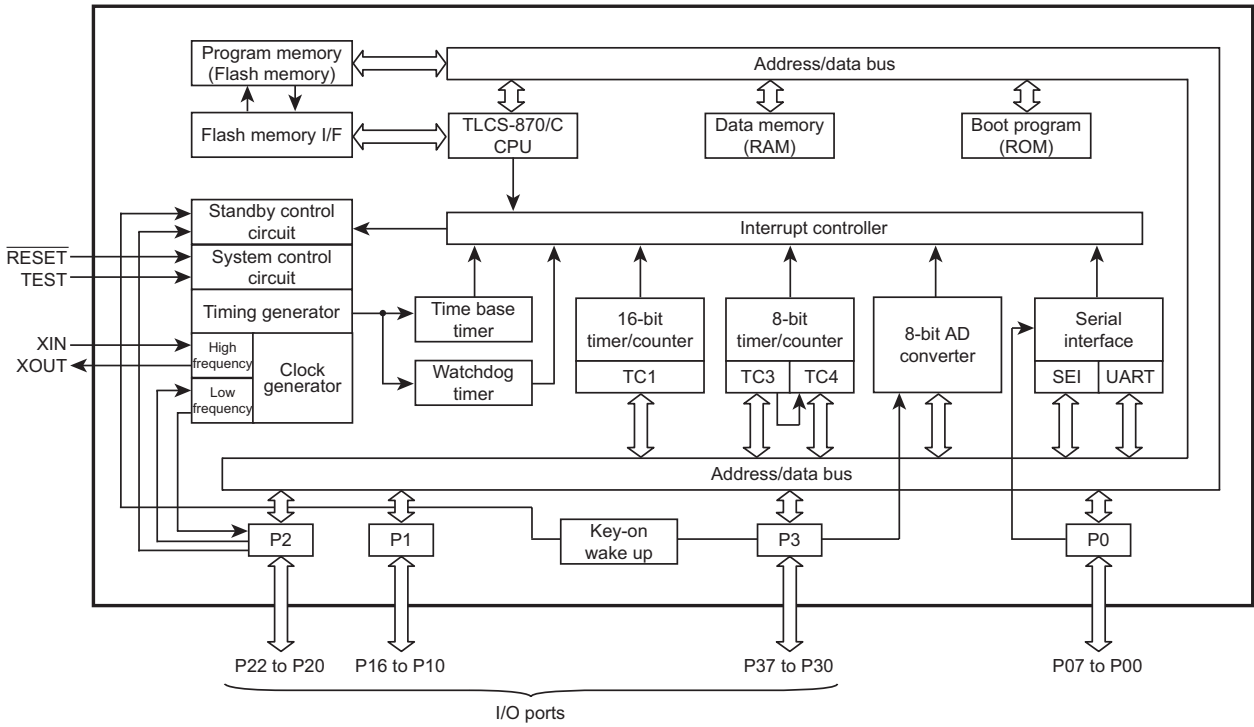


Figure 1-1 Pin Assignment

1.3 Block Diagram



## 2. Operational Description

### 2.1 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, and the reset circuit.

#### 2.1.1 Memory Address Map

The TMP86FH09NG memory is composed Flash, RAM SFR (Special function register) and DBR(Data buffer register). They are all mapped in 64-Kbyte address space. Figure 2-1 shows the TMP86FH09NG memory address map.

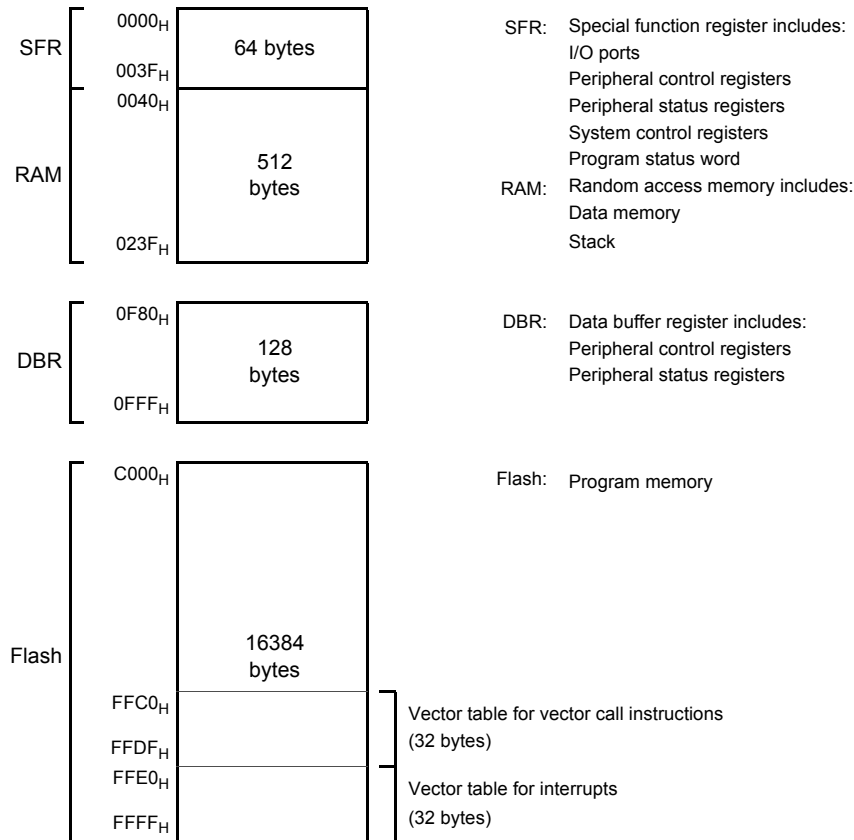


Figure 2-1 Memory Address Map

#### 2.1.2 Program Memory (Flash)

The TMP86FH09NG has a 16384 bytes (Address C000H to FFFFH) of program memory (Flash).

#### 2.1.3 Data Memory (RAM)

The TMP86FH09NG has 512bytes (Address 0040H to 023FH) of internal RAM. The first 192 bytes (0040H to 00FFH) 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”. (TMP86FH09NG)

```

LD      HL, 0040H      ; Start address setup
LD      A, H          ; Initial value (00H) setup
LD      BC, 01FFH
SRAMCLR: LD      (HL), A
INC     HL
DEC     BC
JRS    F, SRAMCLR

```

## 2.2 System Clock Controller

The system clock controller consists of a clock generator, a timing generator, and a standby controller.

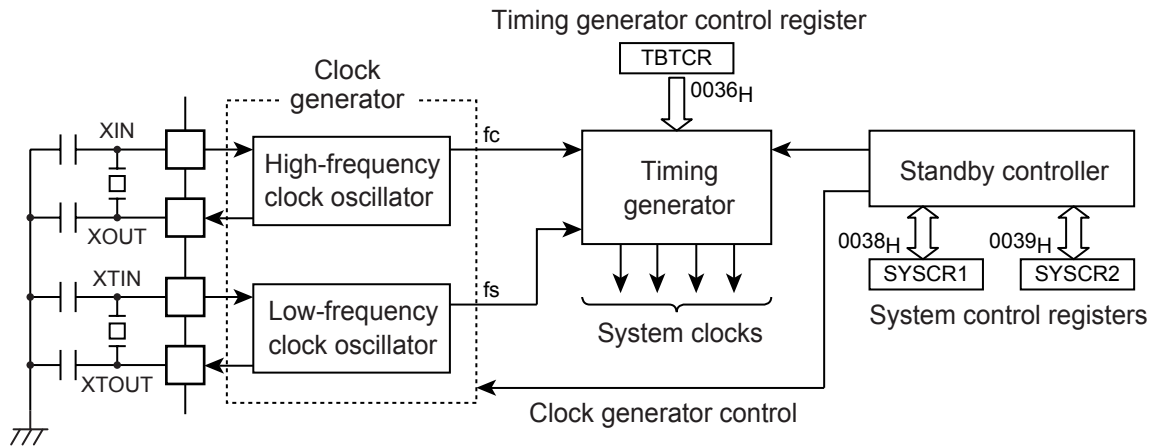


Figure 2-2 System Colck Control

### 2.2.1 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) clock and low-frequency (fs) clock 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.

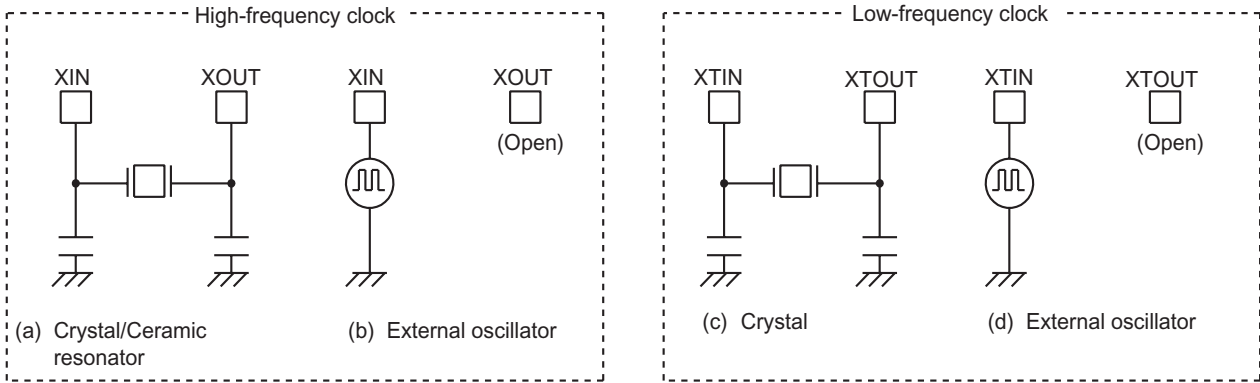


Figure 2-3 Examples of Resonator Connection

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.

## 2.2.2 Timing Generator

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

1. Generation of main system clock
2. Generation of divider output ( $\overline{DV0}$ ) pulses
3. Generation of source clocks for time base timer
4. Generation of source clocks for watchdog timer
5. Generation of internal source clocks for timer/counters
6. Generation of warm-up clocks for releasing STOP mode

### 2.2.2.1 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,  $\text{SYSCK} < \text{SYSCK} >$  and  $\text{TBTCR} < \text{DV7CK} >$ , that is shown in Figure 2-4. As reset and STOP mode started/canceled, the prescaler and the divider are cleared to "0".

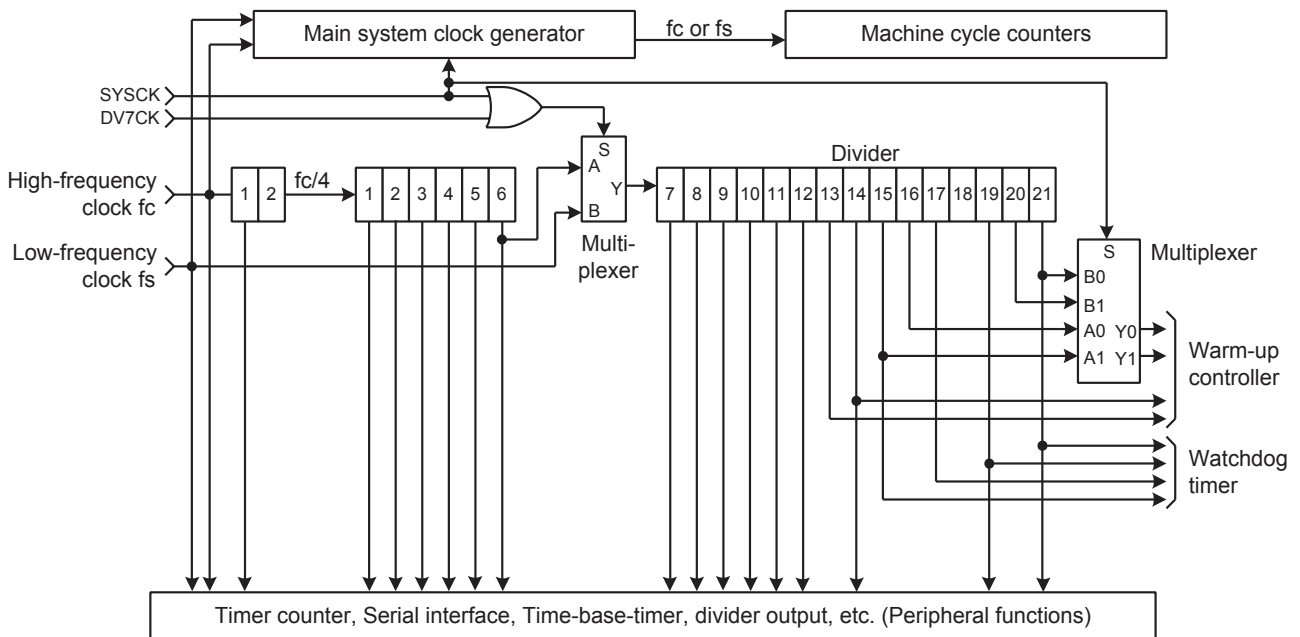


Figure 2-4 Configuration of Timing Generator

Timing Generator Control Register

|                  |         |  |       |                             |         |   |   |     |                            |
|------------------|---------|--|-------|-----------------------------|---------|---|---|-----|----------------------------|
| TBTCR<br>(0036H) | 7       | 6  | 5     | 4                           | 3       | 2 | 1 | 0   | (Initial value: 0000 0000) |
|                  | (DVOEN) | (DVOCK)  | DV7CK | (TBTEN)                     | (TBTCK) |   |   |     |                            |
|                  | DV7CK   | Selection of input to the 7th stage of the divider |       | 0: $fc/2^8$ [Hz]<br>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],  $fs$ : 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 NORMAL1/2 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.

2.2.2.2 Machine cycle

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

The minimum instruction execution unit is called a "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.

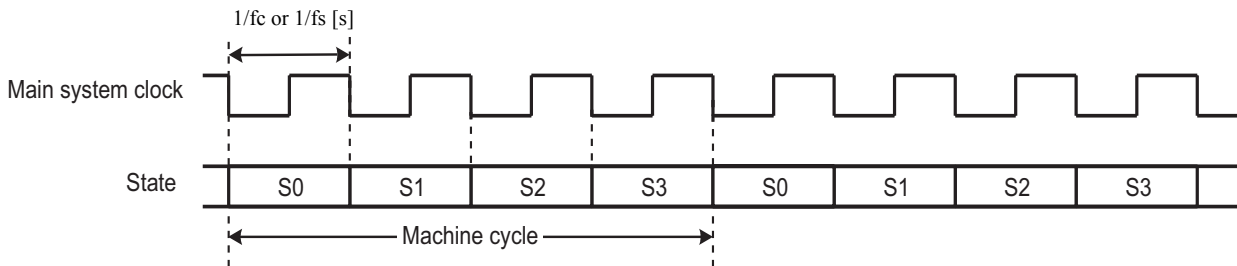


Figure 2-5 Machine Cycle

2.2.3 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 three operating modes: Single clock mode, dual clock mode and STOP mode. These modes are controlled by the system control registers (SYSCR1 and SYSCR2). Figure 2-6 shows the operating mode transition diagram.

2.2.3.1 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].

(1) NORMAL1 mode

In this mode, both the CPU core and on-chip peripherals operate using the high-frequency clock. The TMP86FH09NG is placed in this mode after reset.

## (2) 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> = "1"`, 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.

## (3) IDLE0 mode

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

This mode is enabled by `SYSCR2<TGHALT> = "1"`.

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"`, `EF6` (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.

### 2.2.3.2 Dual-clock mode

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  $4/f_c$  [s] in the NORMAL2 and IDLE2 modes, and  $4/f_s$  [s] (122  $\mu$ s at  $f_s = 32.768$  kHz) in the SLOW and SLEEP modes.

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

## (1) 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.

## (2) 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. As the `SYSCR2<SYSCK>` becomes "1", the hardware changes into SLOW2 mode. As the `SYSCR2<SYSCK>` becomes "0", the hardware changes into NORMAL2 mode. As the `SYSCR2<XEN>` becomes "0", the hardware changes into SLOW1 mode. Do not clear `SYSCR2<XTEN>` to "0" during SLOW2 mode.

## (3) 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 SYSCR2<XEN>. In SLOW1 and SLEEP modes, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

(4) 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.

(5) 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 SLOW1 mode. In SLOW1 and SLEEP1 modes, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

(6) 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.

(7) SLEEP0 mode

In this mode, all the circuit, except oscillator and the timer-base-timer, stops operation. This mode is enabled by setting “1” on bit SYSCR2<TGHALT>.

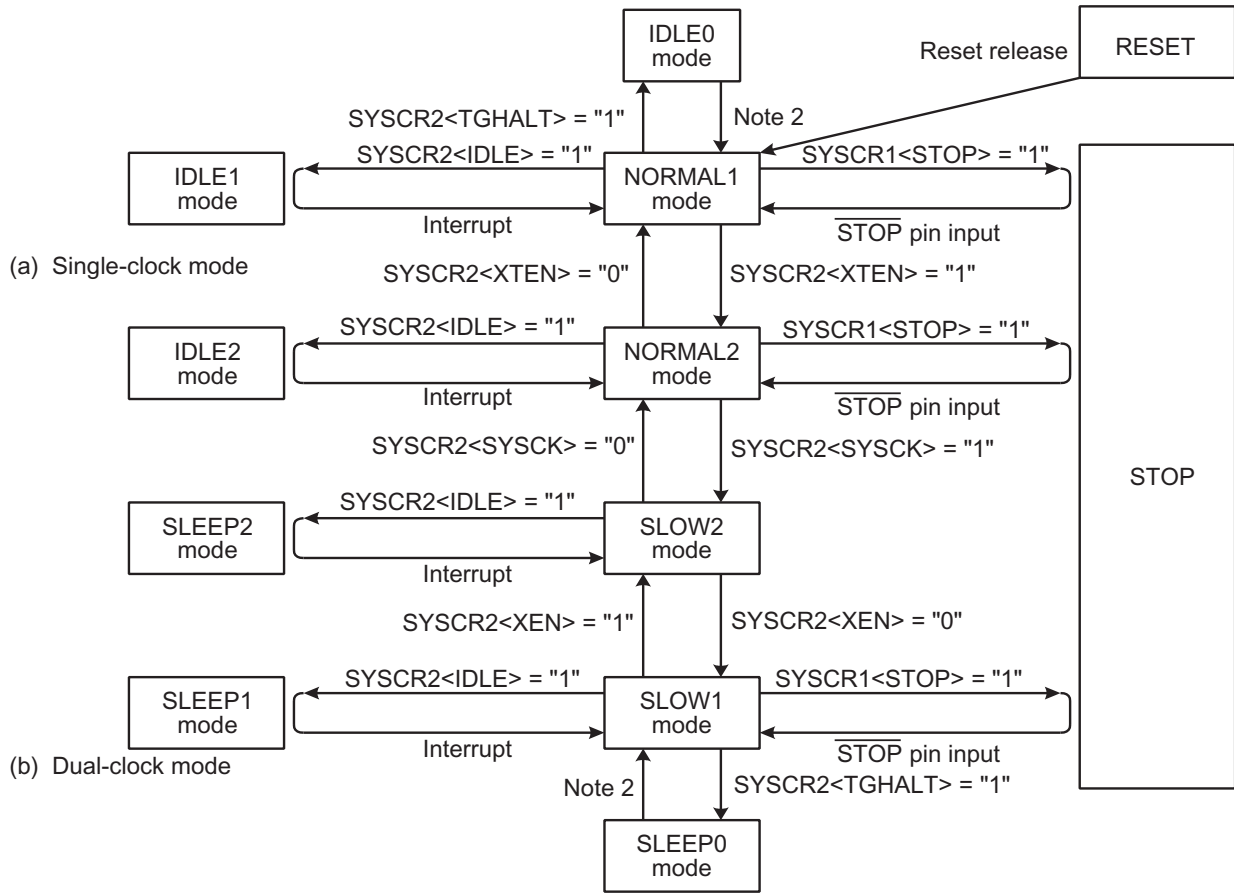
When SLEEP0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

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”, EF6 (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.

### 2.2.3.3 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. After the warm-up period is completed, the execution resumes with the instruction which follows the STOP mode start instruction.



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<TBTCCK> setting.

Figure 2-6 Operating Mode Transition Diagram

| 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   |                |               | Operate                     | Operate | Operate           |                    |
|                | IDLE0   |                |               | Operate                     | Operate | Operate           |                    |
|                | STOP    | Stop           | Halt          | Halt                        | Halt    | -                 |                    |
| Dual clock     | NORMAL2 | Oscillation    | Oscillation   | Operate with high frequency | Operate | Operate           | 4/fc [s]           |
|                | IDLE2   |                |               | Halt                        |         |                   |                    |
|                | SLOW2   |                |               | Operate with low frequency  |         |                   |                    |
|                | SLEEP2  |                |               | Halt                        |         |                   |                    |
|                | SLOW1   | Stop           | Oscillation   | Operate with low frequency  |         |                   | 4/fs [s]           |
|                | SLEEP1  |                |               | Halt                        |         |                   |                    |
|                | SLEEP0  |                |               | Halt                        |         |                   |                    |
|                | STOP    |                |               | Stop                        |         |                   |                    |

**System Control Register 1**

|         |      |      |      |       |     |   |   |   |                            |
|---------|------|------|------|-------|-----|---|---|---|----------------------------|
| SYSCR1  | 7    | 6    | 5    | 4     | 3   | 2 | 1 | 0 |                            |
| (0038H) | STOP | RELM | RETM | OUTEN | WUT |   |   |   | (Initial value: 0000 000*) |

|       |                                     |   |                       |                      |     |
|-------|-------------------------------------|---|-----------------------|----------------------|-----|
| STOP  | STOP mode start                     | 0: CPU core and peripherals remain active<br>1: CPU core and peripherals are halted (Start STOP mode) |                       |                      | R/W |
| RELM  | Release method for STOP mode        | 0: Edge-sensitive release<br>1: Level-sensitive release   |                       |                      | R/W |
| RETM  | Operating mode after STOP mode      | 0: Return to NORMAL1/2 mode<br>1: Return to SLOW1 mode  |                       |                      | R/W |
| OUTEN | Port output during STOP mode        | 0: High impedance<br>1: Output kept   |                       |                      | R/W |
| WUT   | Warm-up time at releasing STOP mode |   | Return to NORMAL mode | Return to SLOW mode  | R/W |
|       |                                     | 000   | $3 \times 2^{16}/fc$  | $3 \times 2^{13}/fs$ |     |
|       |                                     | 010   | $2^{16}/fc$           | $2^{13}/fs$          |     |
|       |                                     | 100   | $3 \times 2^{14}/fc$  | $3 \times 2^6/fs$    |     |
|       |                                     | 110   | $2^{14}/fc$           | $2^6/fs$             |     |

- Note 1: Always set RETM to "0" when transitioning from NORMAL mode to STOP mode. Always set RETM to "1" when transitioning from SLOW mode to STOP mode.
- Note 2: When STOP mode is released with  $\overline{RESET}$  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 external interrupt request on account of falling edge.
- Note 6: When the key-on wakeup is used, RELM should be set to "1".
- Note 7: Port P20 is used as  $\overline{STOP}$  pin. Therefore, when stop mode is started, OUTEN does not affect to P20, and P20 becomes High-Z mode.
- Note 8: The warmig-up time should be set correctly for using oscillator.

**System Control Register 2**

|         |     |      |       |      |   |        |   |   |                            |
|---------|-----|------|-------|------|---|--------|---|---|----------------------------|
| SYSCR2  | 7   | 6    | 5     | 4    | 3 | 2      | 1 | 0 |                            |
| (0039H) | XEN | XTEN | SYSCK | IDLE |   | TGHALT |   |   | (Initial value: 1000 *0**) |

|        |   |  |  |  |     |
|--------|---|--|--|--|-----|
| XEN    | High-frequency oscillator control                                 | 0: Turn off oscillation<br>1: Turn on oscillation  |  |  | R/W |
| XTEN   | Low-frequency oscillator control                                  | 0: Turn off oscillation<br>1: Turn on oscillation  |  |  |     |
| SYSCK  | Main system clock select (Write)/main system clock monitor (Read) | 0: High-frequency clock (NORMAL1/NORMAL2/IDLE1/IDLE2)<br>1: Low-frequency clock (SLOW1/SLOW2/SLEEP1/SLEEP2)                            |  |  |     |
| IDLE   | CPU and watchdog timer control (IDLE1/2 and SLEEP1/2 modes)       | 0: CPU and watchdog timer remain active<br>1: CPU and watchdog timer are stopped (Start IDLE1/2 and SLEEP1/2 modes)                    |  |  | R/W |
| TGHALT | TG control (IDLE0 and SLEEP0 modes)                               | 0: Feeding clock to all peripherals from TG<br>1: Stop feeding clock to peripherals except TBT from TG. (Start IDLE0 and SLEEP0 modes) |  |  |     |

- 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, \*, Don't care
- Note 3: Bits 3, 1 and 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  $TBTCR < TBTCCK >$ .
- 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.

## 2.2.4 Operating Mode Control

### 2.2.4.1 STOP mode

STOP mode is controlled by the system control register 1, the  $\overline{\text{STOP}}$  pin input and key-on wakeup input (STOP5 to 2) 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.

1. Oscillations are turned off, and all internal operations are halted.
2. 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.
3. The prescaler and the divider of the timing generator are cleared to “0”.
4. 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 key-on wakeup input (STOP5 to 2) for releasing STOP mode in edge-sensitive mode.

Note 1: The STOP mode can be released by either the STOP or key-on wakeup pin (STOP5 to 2). However, because the STOP pin is different from the key-on wakeup and can not inhibit the release input, the 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.

#### (1) Level-sensitive release mode (RELM = “1”)

In this mode, STOP mode is released by setting the  $\overline{\text{STOP}}$  pin high or setting the STOP5 to 2 pin input which is enabled by STOPCR. This mode is used for capacitor backup when the main power supply is cut off and long term battery backup.

Even if an instruction for starting STOP mode is executed while  $\overline{\text{STOP}}$  pin input is high or STOP5 to 2 input is low, STOP mode does not start but instead the warm-up sequence starts immediately. 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 or STOP5 to 2 input is high. The following two methods can be used for confirmation.

1. Testing a port.
2. 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  $\overline{\text{STOP}}$  pin input goes low level
           JRS       F, SSTOPH
           DI          ; IMF ← 0
           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
           JRS       F, SINT5           port P20 is at high
           LD        (SYSCR1), 01010000B ; Sets up the level-sensitive release mode.
           DI                          ; IMF ← 0
           SET      (SYSCR1). 7         ; Starts STOP mode

SINT5:      RETI
    
```

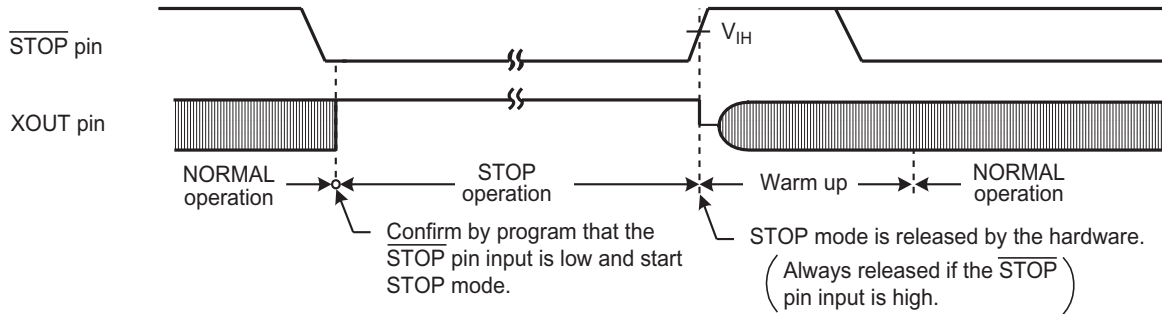


Figure 2-7 Level-sensitive Release Mode

- Note 1: Even if the  $\overline{\text{STOP}}$  pin input is low after warm-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.

(2) 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 STOP pin. In the edge-sensitive release mode, STOP mode is started even when the STOP pin input is high level. Do not use any STOP5 to 2 pin input for releasing STOP mode in edge-sensitive release mode.

Example :Starting STOP mode from NORMAL mode

```

DI          ; IMF ← 0

LD          (SYSCR1), 10010000B ; Starts after specified to the edge-sensitive release mode
    
```

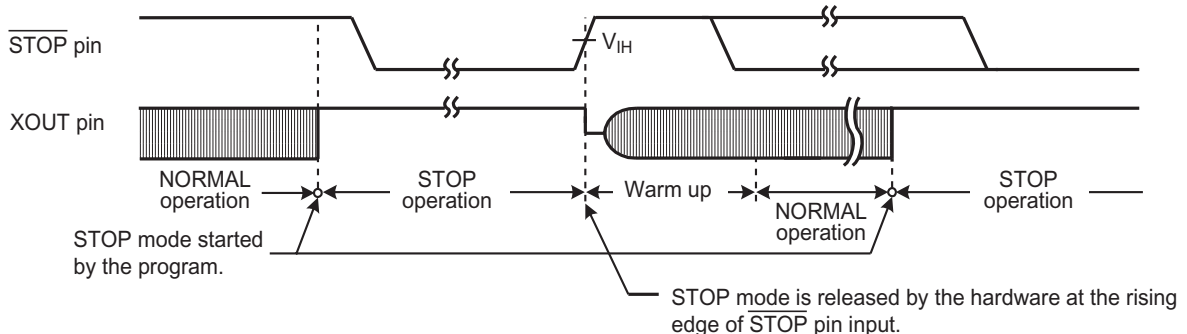


Figure 2-8 Edge-sensitive Release Mode

STOP mode is released by the following sequence.

1. 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.
2. A warm-up period is inserted to allow oscillation time to stabilize. During warm up, all internal operations remain halted. Four different warm-up times can be selected with the SYSCR1<WUT> in accordance with the resonator characteristics.
3. When the warm-up time has elapsed, normal operation resumes with the instruction following the STOP mode start instruction.

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 2-1 Warm-up Time Example (at  $f_c = 16.0$  MHz,  $f_s = 32.768$  kHz)

| WUT | Warm-up Time [ms]     |                     |
|-----|-----------------------|---------------------|
|     | Return to NORMAL Mode | Return to SLOW Mode |
| 00  | 12.288                | 750                 |
| 01  | 4.096                 | 250                 |
| 10  | 3.072                 | 5.85                |
| 11  | 1.024                 | 1.95                |

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 as an approximate value.

Note 2: The CPU wait period for FLASH is shown in parentheses.

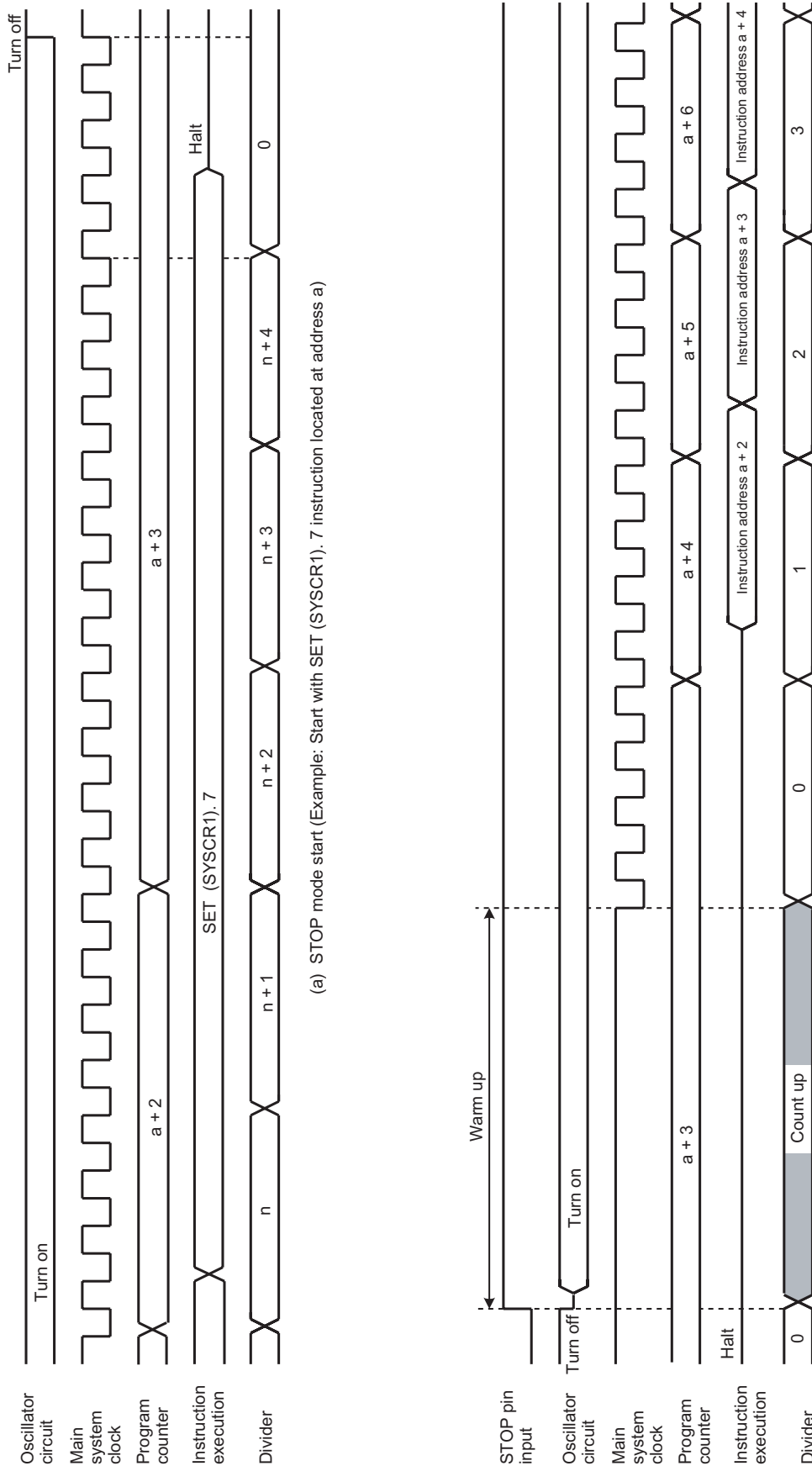


Figure 2-9 STOP Mode Start/Release

### 2.2.4.2 IDLE1/2 mode and 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.

1. Operation of the CPU and watchdog timer (WDT) is halted. On-chip peripherals continue to operate.
2. 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.
3. The program counter holds the address 2 ahead of the instruction which starts these modes.

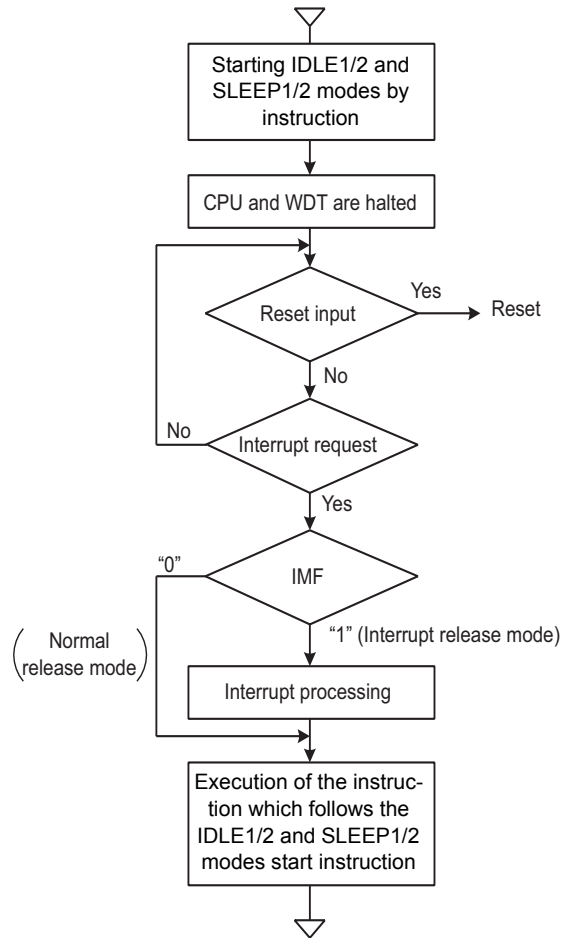


Figure 2-10 IDLE1/2 and SLEEP1/2 Modes

- Start the IDLE1/2 and SLEEP1/2 modes

After IMF is set to "0", set the individual interrupt enable flag (EF) which releases IDLE1/2 and SLEEP1/2 modes. To start IDLE1/2 and SLEEP1/2 modes, 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.

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

(1) 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.

(2) 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) and the interrupt processing is started. 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 modes are started, the watchdog timer interrupt will be processed but IDLE1/2 and SLEEP1/2 modes will not be started.

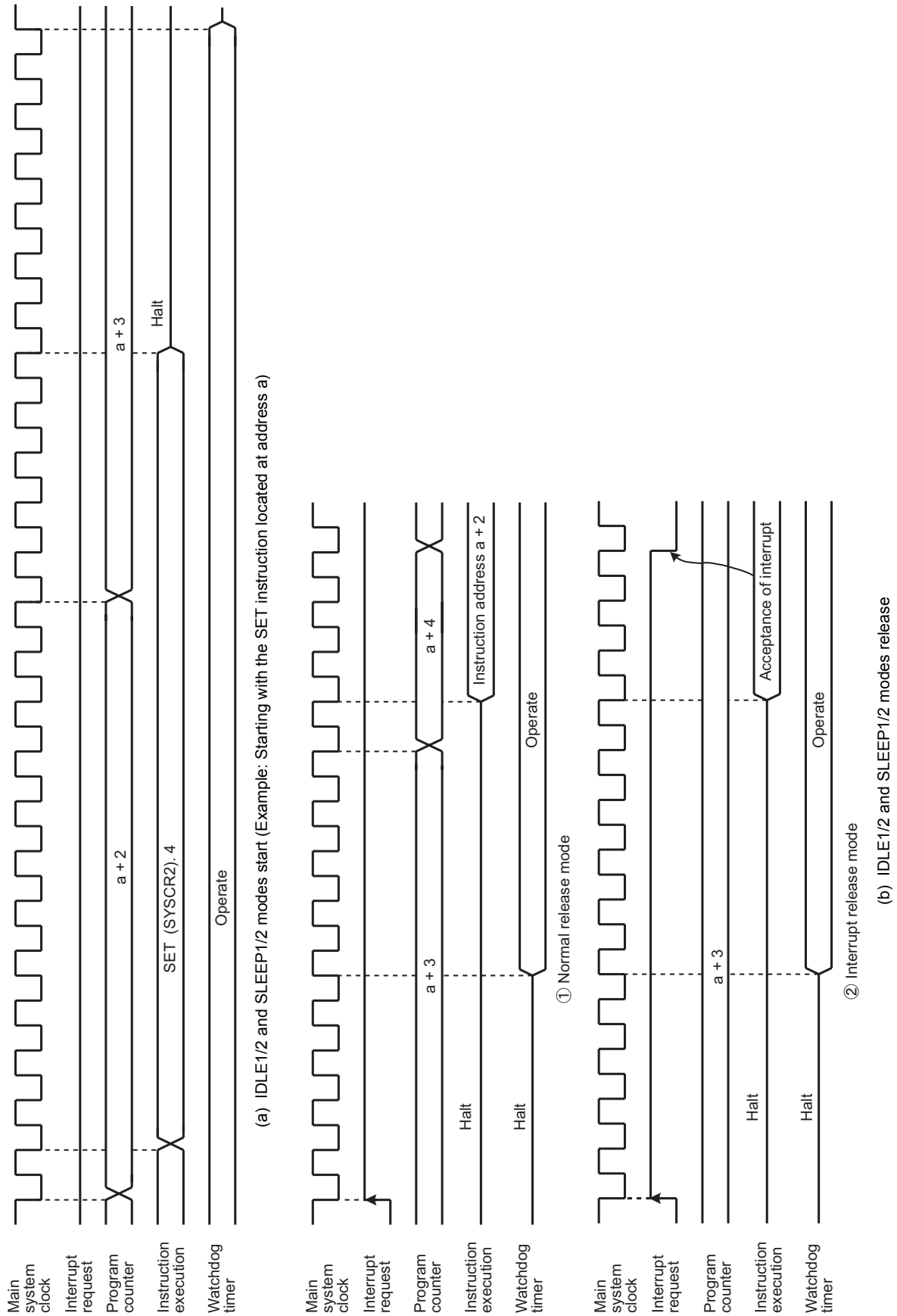


Figure 2-11 IDLE1/2 and SLEEP1/2 Modes Start/Release

2.2.4.3 IDLE0 and SLEEP0 modes (IDLE0, SLEEP0)

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

1. Timing generator stops feeding clock to peripherals except TBT.
2. 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.
3. 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) peripherals.

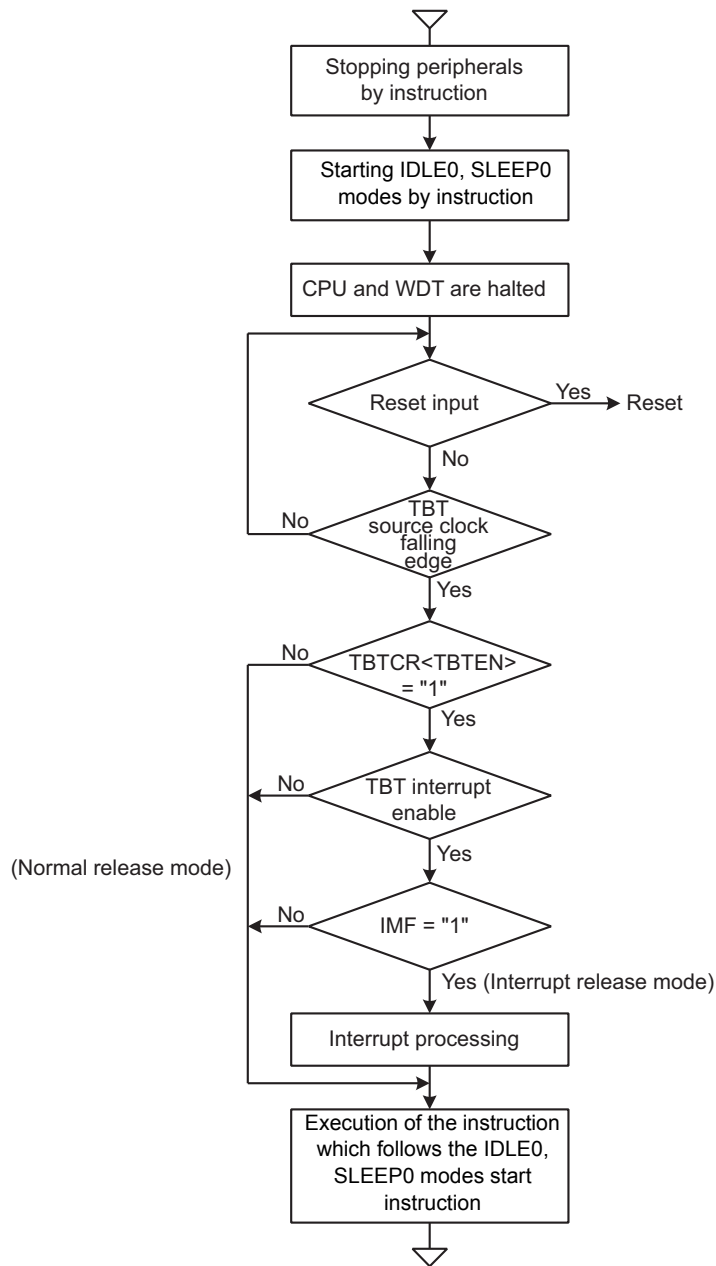


Figure 2-12 IDLE0 and SLEEP0 Modes

- Start the IDLE0 and SLEEP0 modes

Stop (Disable) peripherals such as a timer counter.

To start IDLE0 and SLEEP0 modes, set SYSCR2<TGHALT> to “1”.

- Release the IDLE0 and SLEEP0 modes

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

These modes are selected by interrupt master flag (IMF), the individual interrupt enable flag of TBT 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”.

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

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

- (1) Normal release mode (IMF · EF6 · TBTCR<TBTEN> = “0”)

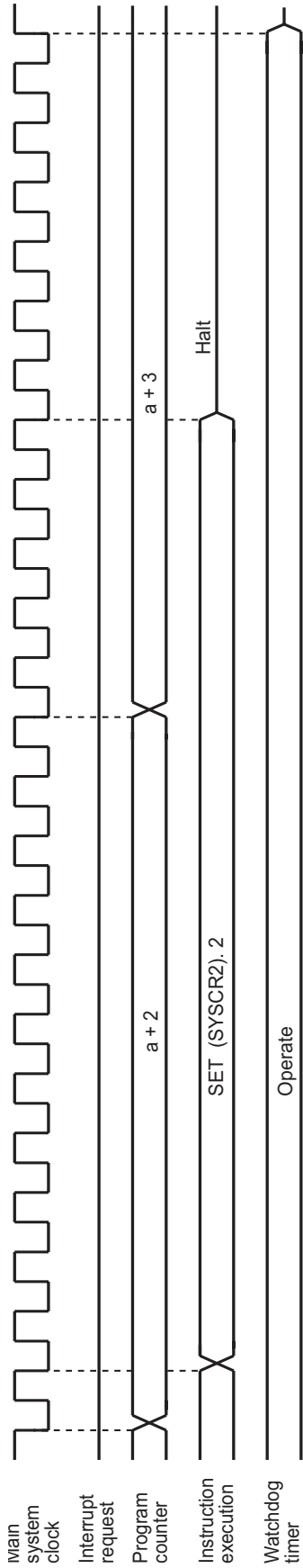
IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK>. After the falling edge is detected, the program operation is resumed from the instruction following the IDLE0 and SLEEP0 modes start instruction. Before starting the IDLE0 or SLEEP0 mode, when the TBTCR<TBTEN> is set to “1”, INTTBT interrupt latch is set to “1”.

- (2) Interrupt release mode (IMF · EF6 · TBTCR<TBTEN> = “1”)

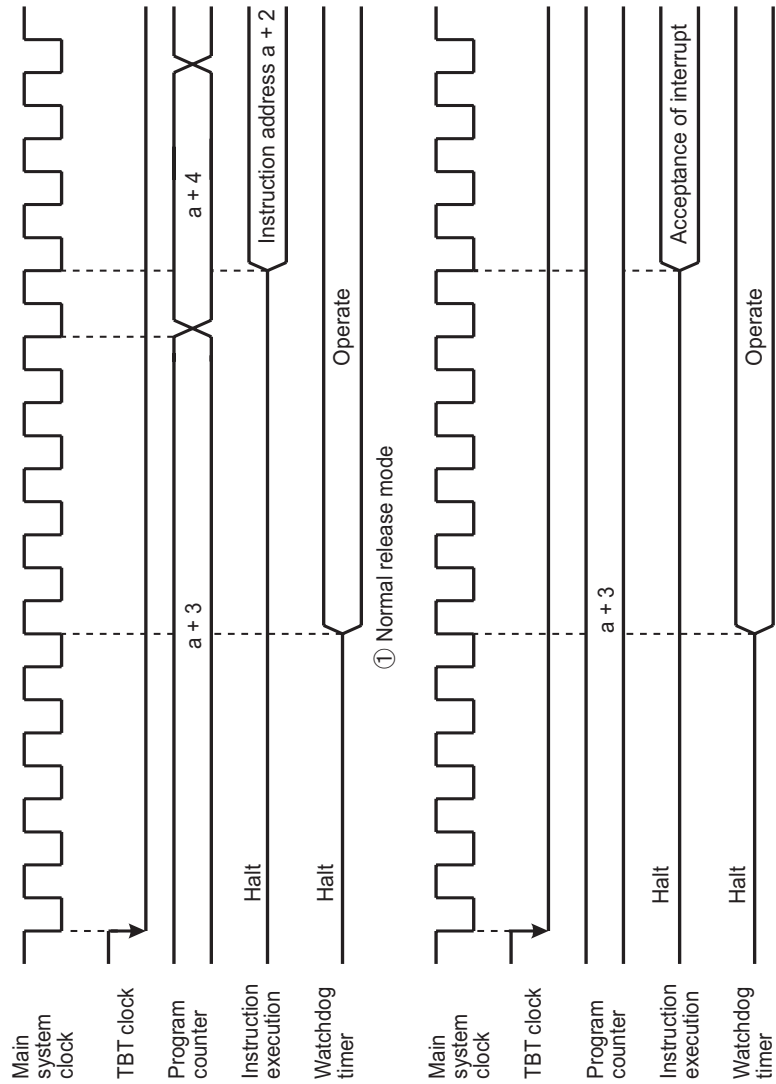
IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK> and INTTBT interrupt processing is started.

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.



(a) IDLE0 and SLEEP0 modes start (Example: Starting with the SET instruction located at address a)



(b) IDLE and SLEEP0 modes release

Figure 2-13 IDLE0 and SLEEP0 Modes Start/Release

### 2.2.4.4 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.

#### (1) 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 can be continued oscillation in order to return to NORMAL2 mode from SLOW mode quickly. Always turn off oscillation of high-frequency clock when switching from SLOW mode to stop mode.

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)

```

Example 2 :Switching to the SLOW1 mode after low-frequency clock has stabilized.

```

SET      (SYSCR2). 6      ; SYSCR2<XTEN> ← 1

LD       (TC3CR), 43H     ; Sets mode for TC4, 3 (16-bit mode, fs for source)

LD       (TC4CR), 05H     ; Sets warming-up counter mode

LDW     (TTREG3), 8000H   ; Sets warm-up time (Depend on oscillator accompanied)

DI      ; IMF ← 0

SET     (EIRH). 3        ; Enables INTTC4

EI      ; IMF ← 1

SET     (TC4CR). 3       ; Starts TC4, 3

:

PINTTC4: CLR      (TC4CR). 3       ; Stops TC4, 3

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

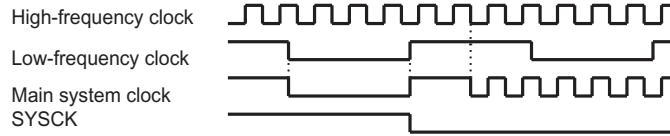
:

VINTTC4: DW       PINTTC4        ; INTTC4 vector table

```

(2) Switching from SLOW1 mode to NORMAL2 mode

Note: 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.



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       (TC3CR), 63H     ; Sets mode for TC4, 3 (16-bit mode, fc for source)

LD       (TC4CR), 05H     ; Sets warming-up counter mode

LD       (TTREG4), 0F8H   ; Sets warm-up time

DI       ; IMF ← 0

SET     (EIRH). 3        ; Enables INTTC4

EI       ; IMF ← 1

SET     (TC4CR). 3       ; Starts TC4, 3

:

PINTTC4: CLR      (TC4CR). 3      ; Stops TC4, 3

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

RETI

:

VINTTC4: DW       PINTTC4      ; INTTC4 vector table
    
```

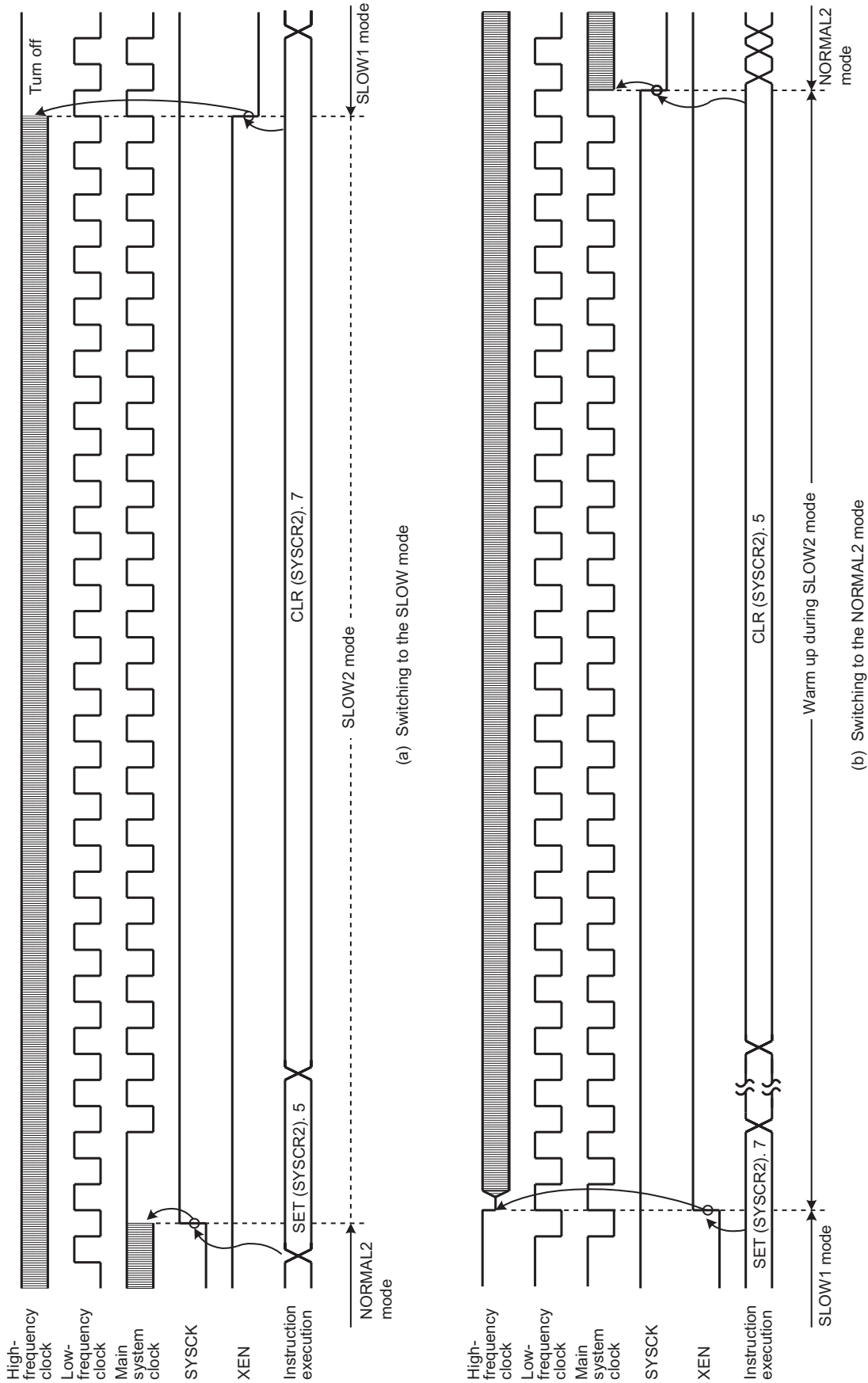


Figure 2-14 Switching between the NORMAL2 and SLOW Modes

## 2.3 Reset Circuit

The TMP86FH09NG has four types of reset generation procedures: An external reset input, an address trap reset, a watchdog timer reset and a system clock reset. Of these reset, the address trap reset, the watchdog timer and the system clock reset are a malfunction reset. When the malfunction reset request is detected, reset occurs during the maximum  $24/f_c[s]$ .

The malfunction reset circuit such as watchdog timer reset, address trap reset and system clock reset is not initialized when power is turned on. Therefore, reset may occur during maximum  $24/f_c[s]$  ( $1.5\mu s$  at 16.0 MHz) when power is turned on.

Table 1-3 shows on-chip hardware initialization by reset action.

Table 2-2 Initializing Internal Status by Reset Action

| On-chip Hardware   | Initial Value   | On-chip Hardware                          | Initial Value                     |
|--|-----------------|---|-----------------------------------|
| Program counter (PC)                                       | (FFFEH)         | 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                   |

### 2.3.1 External Reset Input

The  $\overline{\text{RESET}}$  pin contains a Schmitt trigger (Hysteresis) with an internal pull-up resistor.

When the  $\overline{\text{RESET}}$  pin is held at “L” level for at least 3 machine cycles ( $12/f_c [s]$ ) with the power supply voltage within the operating voltage range and oscillation stable, a reset is applied and the internal state is initialized.

When the  $\overline{\text{RESET}}$  pin input goes high, the reset operation is released and the program execution starts at the vector address stored at addresses FFFEh to FFFFh.

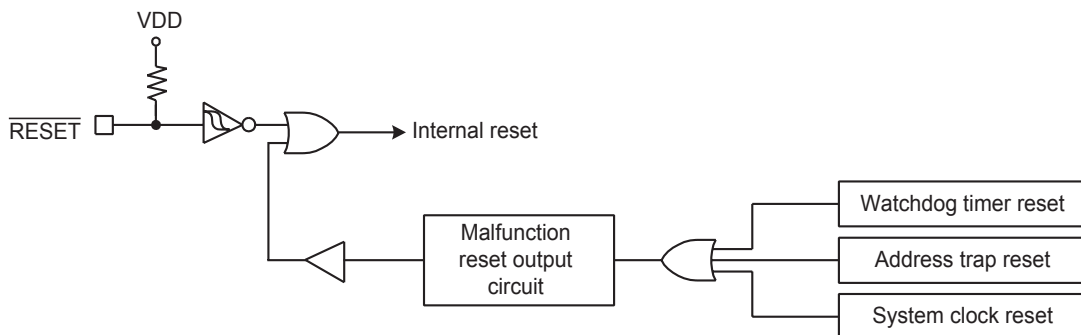
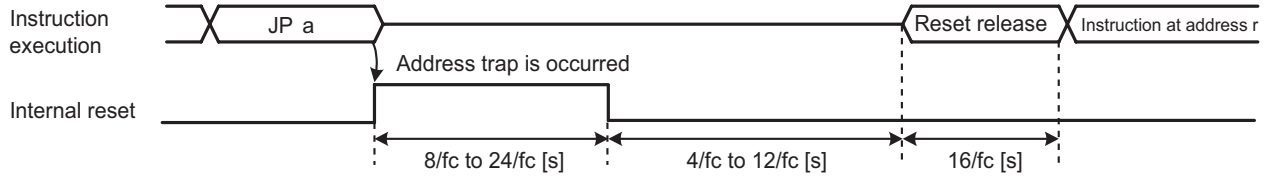


Figure 2-15 Reset Circuit

### 2.3.2 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”), DBR or the SFR area, address trap reset will be generated. The reset time is maximum  $24/f_c$  [s] ( $1.5\mu\text{s}$  at 16.0 MHz).

Note: The operating mode under address trapped is alternative of reset or interrupt. The address trap area is alternative.



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 2-16 Address Trap Reset

### 2.3.3 Watchdog timer reset

Refer to 2.4 “Watchdog Timer”.

### 2.3.4 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”.

The reset time is maximum  $24/f_c$  ( $1.5\mu\text{s}$  at 16.0 MHz).





### 3. Interrupt Control Circuit

The TMP86FH09NG has a total of 17 interrupt sources excluding reset, of which 1 source levels are multiplexed. Interrupts can be nested with priorities. Four of the internal interrupt sources are non-maskable while the rest are maskable.

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.

| Interrupt Factors |   | Enable Condition          | Interrupt Latch | Vector Address | Priority |
|-------------------|---|---------------------------|-----------------|----------------|----------|
| Internal/External | (Reset)   | Non-maskable              | –               | FFFE           | 1        |
| Internal          | INTSWI (Software interrupt)                             | Non-maskable              | –               | FFFC           | 2        |
| Internal          | INTUNDEF (Executed the undefined instruction interrupt) | Non-maskable              | –               | FFFC           | 2        |
| Internal          | INTATRAP (Address trap interrupt)                       | Non-maskable              | IL2             | FFFA           | 2        |
| Internal          | INTWDT (Watchdog timer interrupt)                       | Non-maskable              | IL3             | FFF8           | 2        |
| External          | INT0  | IMF• EF4 = 1, INTOEN = 1  | IL4             | FFF6           | 5        |
| External          | INT1  | IMF• EF5 = 1              | IL5             | FFF4           | 6        |
| Internal          | INTTBT  | IMF• EF6 = 1              | IL6             | FFF2           | 7        |
| Internal          | INTTC1  | IMF• EF7 = 1              | IL7             | FFF0           | 8        |
| Internal          | INTRXD  | IMF• EF8 = 1              | IL8             | FFEE           | 9        |
| Internal          | INTTXD  | IMF• EF9 = 1              | IL9             | FFEC           | 10       |
| Internal          | INTTC3  | IMF• EF10 = 1             | IL10            | FFEA           | 11       |
| Internal          | INTTC4  | IMF• EF11 = 1, IL11ER = 0 | IL11            | FFE8           | 12       |
| External          | INT3  | IMF• EF11 = 1, IL11ER = 1 |                 |                |          |
| Internal          | INTADC  | IMF• EF12 = 1             | IL12            | FFE6           | 13       |
| External          | INTSEI  | IMF• EF13 = 1             | IL13            | FFE4           | 14       |
| External          | INT4  | IMF• EF14 = 1             | IL14            | FFE2           | 15       |
| External          | INT5  | IMF• EF15 = 1             | IL15            | FFE0           | 16       |

Note 1: The INTSEL register is used to select the interrupt source to be enabled for each multiplexed source level (see 3.3 Interrupt Source Selector (INTSEL)).

Note 2: To use the address trap interrupt (INTATRAP), clear WDTTCR1<ATOUT> to “0” (It is set for the “reset request” after reset is cancelled). For details, see “Address Trap”.

Note 3: To use the watchdog timer interrupt (INTWDT), clear WDTTCR1<WDTOUT> to “0” (It is set for the “Reset request” after reset is released). For details, see “Watchdog Timer”.

#### 3.1 Interrupt latches (IL15 to IL2)

An interrupt latch is provided for each interrupt source, except for a software interrupt and an executed the undefined instruction 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. The interrupt latch is cleared to “0” immediately after accepting interrupt. All interrupt latches are initialized to “0” during reset.

The interrupt latches are located on address 003CH and 003DH in SFR area. Each latch can be cleared to “0” individually by instruction. However, IL2 and IL3 should not be cleared to “0” by software. For clearing the interrupt latch, load instruction should be used and then IL2 and IL3 should be set to “1”. If the read-modify-write instructions such as bit manipulation or operation instructions are used, interrupt request would be cleared inadequately if interrupt is requested while such instructions are executed.

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: In main program, before manipulating the interrupt enable flag (EF) or the interrupt latch (IL), be sure to clear IMF to "0" (Disable interrupt by DI instruction). Then set IMF newly again as required after operating on the EF or IL (Enable interrupt by EI instruction)  
In interrupt service routine, because the IMF becomes "0" automatically, clearing IMF need not execute normally on interrupt service routine. However, if using multiple interrupt on interrupt service routine, manipulating EF or IL should be executed before setting IMF="1".

Example 1 :Clears interrupt latches

```
DI                ; IMF ← 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 interrupt latches

```
TEST          (ILL). 7 ; if IL7 = 1 then jump
JR           F, SSET
```

## 3.2 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 003AH and 003BH 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).

### 3.2.1 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: 003AH 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".

### 3.2.2 Individual interrupt enable flags (EF15 to EF4)

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. During reset, all the individual interrupt enable flags (EF15 to EF4) are initialized to "0" and all maskable interrupts are not accepted until they are set to "1".

Note: In main program, before manipulating the interrupt enable flag (EF) or the interrupt latch (IL), be sure to clear IMF to "0" (Disable interrupt by DI instruction). Then set IMF newly again as required after operating on the EF or IL (Enable interrupt by EI instruction)  
In interrupt service routine, because the IMF becomes "0" automatically, clearing IMF need not execute normally on interrupt service routine. However, if using multiple interrupt on interrupt service routine, manipulating EF or IL should be executed before setting IMF="1".

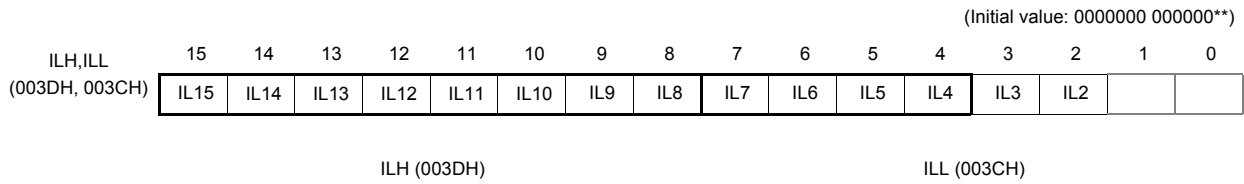
Example 1 :Enables interrupts individually and sets IMF

```
DI ; IMF ← 0
LDW (EIRL), 1110100010100000B ; EF15 to EF13, EF11, EF7, EF5 ← 1
: ; Note: IMF should not be set.
:
EI ; IMF ← 1
```

Example 2 :C compiler description example

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

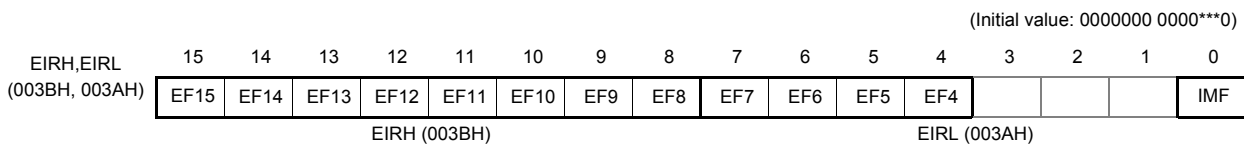
Interrupt Latches



|             |                   |  |  |     |
|-------------|-------------------|--|--|-----|
| IL15 to IL2 | Interrupt latches | at RD<br>0: No interrupt request<br>1: Interrupt request | at WR<br>0: Clears the interrupt request<br>1: (Interrupt latch is not set.) | R/W |
|-------------|-------------------|--|--|-----|

- Note 1: To clear any one of bits IL7 to IL4, be sure to write "1" into IL2 and IL3.
- Note 2: In main program, before manipulating the interrupt enable flag (EF) or the interrupt latch (IL), be sure to clear IMF to "0" (Disable interrupt by DI instruction). Then set IMF newly again as required after operating on the EF or IL (Enable interrupt by EI instruction)  
 In interrupt service routine, because the IMF becomes "0" automatically, clearing IMF need not execute normally on interrupt service routine. However, if using multiple interrupt on interrupt service routine, manipulating EF or IL should be executed before setting IMF="1".
- Note 3: Do not clear IL with read-modify-write instructions such as bit operations.

Interrupt Enable Registers



|             |  |   |     |
|-------------|--|---|-----|
| EF15 to EF4 | Individual-interrupt enable flag<br>(Specified for each bit) | 0: Disables the acceptance of each maskable interrupt.<br>1: Enables the acceptance of each maskable interrupt. | R/W |
| IMF         | Interrupt master enable flag                                 | 0: Disables the acceptance of all maskable interrupts<br>1: Enables the acceptance of all maskable interrupts   |     |

- Note 1: \*: Don't care
- Note 2: Do not set IMF and the interrupt enable flag (EF15 to EF4) to "1" at the same time.
- Note 3: In main program, before manipulating the interrupt enable flag (EF) or the interrupt latch (IL), be sure to clear IMF to "0" (Disable interrupt by DI instruction). Then set IMF newly again as required after operating on the EF or IL (Enable interrupt by EI instruction)  
 In interrupt service routine, because the IMF becomes "0" automatically, clearing IMF need not execute normally on interrupt service routine. However, if using multiple interrupt on interrupt service routine, manipulating EF or IL should be executed before setting IMF="1".

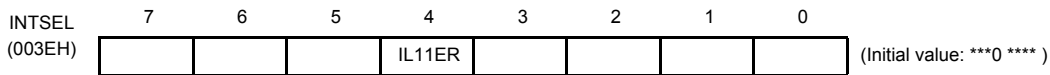
### 3.3 Interrupt Source Selector (INTSEL)

Each interrupt source that shares the interrupt source level with another interrupt source is allowed to enable the interrupt latch only when it is selected in the INTSEL register. The interrupt controller does not hold interrupt requests corresponding to interrupt sources that are not selected in the INTSEL register. Therefore, the INTSEL register must be set appropriately before interrupt requests are generated.

The following interrupt sources share their interrupt source level; the source is selected on the register INTSEL.

1. INTTC4 and INT3 share the interrupt source level whose priority is 12.

#### Interrupt source selector



|        |                        |                      |     |
|--------|------------------------|----------------------|-----|
| IL11ER | Selects INTTC4 or INT3 | 0: INTTC4<br>1: INT3 | R/W |
|--------|------------------------|----------------------|-----|

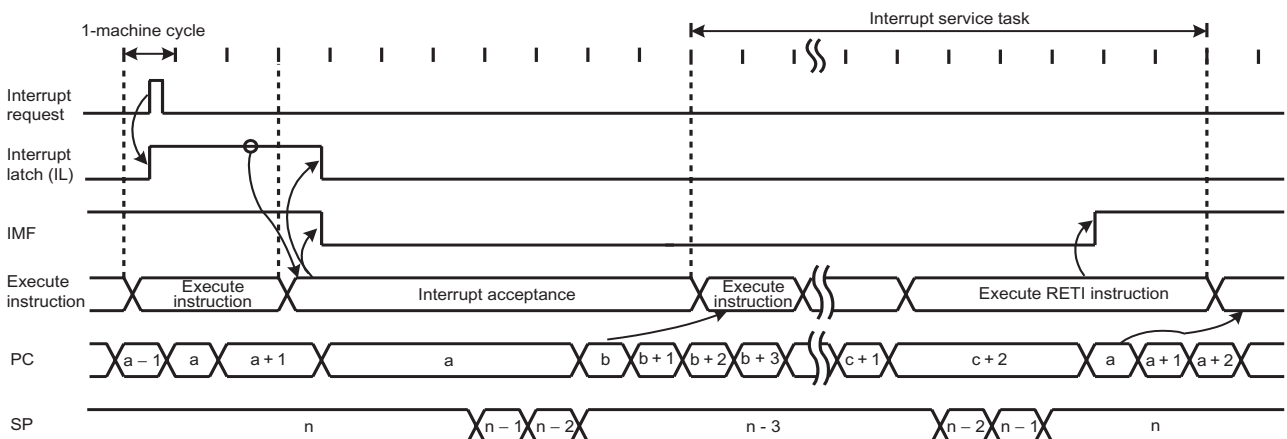
### 3.4 Interrupt Sequence

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 (2 μs @16 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 3-1 shows the timing chart of interrupt acceptance processing.

#### 3.4.1 Interrupt acceptance processing is packaged as follows.

- The interrupt master enable flag (IMF) is cleared to “0” in order to disable the acceptance of any following interrupt.
- The interrupt latch (IL) for the interrupt source accepted is cleared to “0”.
- 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.
- The entry address (Interrupt vector) of the corresponding interrupt service program, loaded on the vector table, is transferred to the program counter.
- 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.

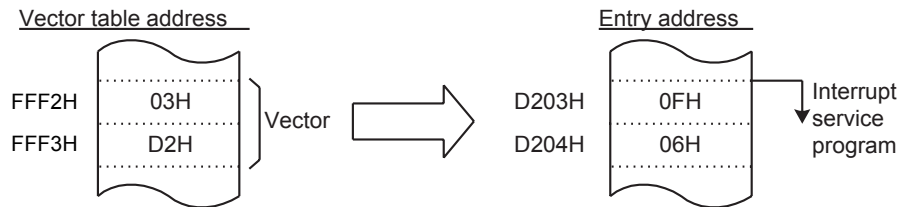


Note 1: a: Return address entry address, b: Entry address, c: Address which RETI instruction is stored

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 3-1 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



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 shorten compared with length between interrupt requests; otherwise the status cannot be recovered as non-maskable interrupt would simply nested.

### 3.4.2 Saving/restoring general-purpose registers

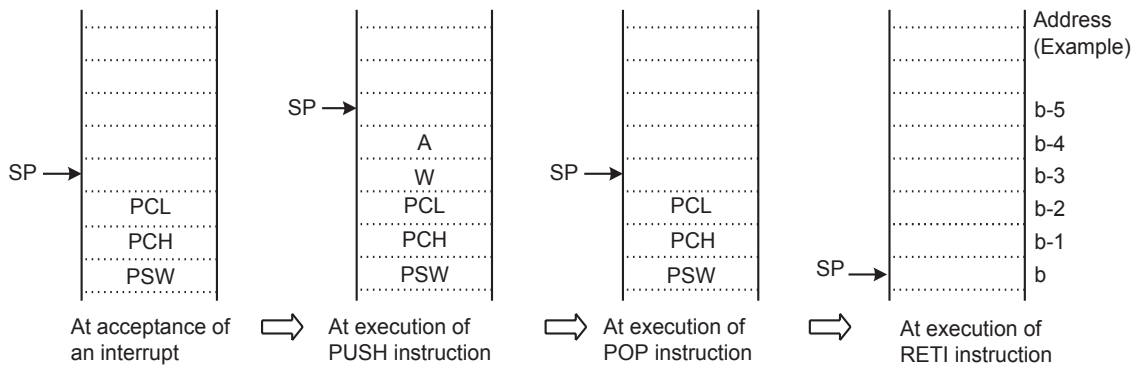
During interrupt acceptance processing, the program counter (PC) and the program status word (PSW, includes IMF) are automatically saved on the stack, but the accumulator and others are not. These registers are saved by software if necessary. When multiple interrupt services are nested, it is also necessary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

#### 3.4.2.1 Using PUSH and POP instructions

If only a specific register is saved or interrupts of the same source are nested, general-purpose registers can be saved/restored using the PUSH/POP instructions.

Example :Save/store register using PUSH and POP instructions

```
PINTxx:    PUSH    WA           ; Save WA register
           (interrupt processing)
           POP     WA           ; Restore WA register
           RETI                ; RETURN
```



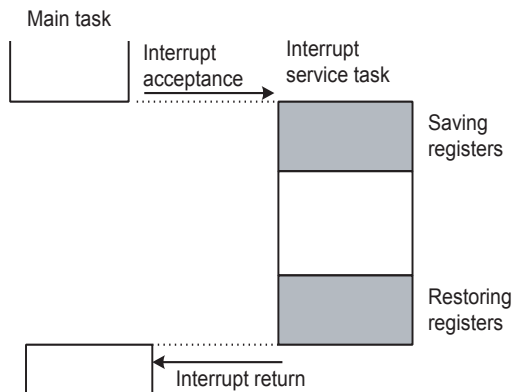
3.4.2.2 Using data transfer instructions

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 data transfer instruction

Figure 3-2 Saving/Restoring General-purpose Registers under Interrupt Processing

3.4.3 Interrupt return

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

- | [RETI]/[RETN] Interrupt Return   |
|--|
| <ol style="list-style-type: none"> <li>1. Program counter (PC) and program status word (PSW, includes IMF) are restored from the stack.</li> <li>2. Stack pointer (SP) is incremented by 3.</li> </ol> |

As for address trap interrupt (INTATRAP), it is required to alter stacked data for program counter (PC) to restarting address, during interrupt service program.

Note: If [RETN] is executed with the above data unaltered, the program returns to the address trap area and INTATRAP occurs again. When interrupt acceptance processing has completed, stacked data for PCL and PCH are located on address (SP + 1) and (SP + 2) respectively.

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
```

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 1: 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).

Note 2: When the interrupt processing time is longer than the interrupt request generation time, the interrupt service task is performed but not the main task.

## 3.5 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.

### 3.5.1 Address error detection

FFH 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 FFH 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 FFH to unused areas of the program memory. Address trap reset is generated in case that an instruction is fetched from RAM, DBR or SFR areas.

### 3.5.2 Debugging

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

### 3.6 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.

### 3.7 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).

### 3.8 External Interrupts

The TMP86FH09NG has 5 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, INT3 and INT4. The  $\overline{\text{INT0}}$ /P10 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  $\overline{\text{INT0}}$ /P10 pin function selection are performed by the external interrupt control register (EINTCR).

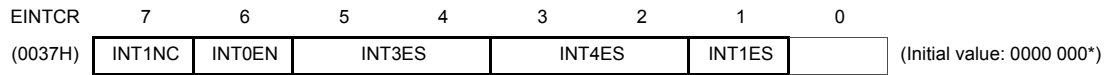
| Source | Pin                      | Enable Conditions                 | Release Edge (level)  | Digital Noise Reject  |
|--------|--------------------------|-----------------------------------|---|---|
| INT0   | $\overline{\text{INT0}}$ | IMF • EF4 • INT0EN=1              | Falling edge  | Pulses 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                     | IMF • EF5 = 1                     | Falling edge<br>or<br>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. |
| INT3   | INT3                     | IMF • EF11 = 1<br>and<br>IL11ER=1 | Falling edge,<br>Rising edge,<br>Falling and Rising edge<br>or<br>H level | Pulses of less than 7/fc [s] are eliminated as noise. Pulses of 25/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.                     |
| INT4   | INT4                     | IMF • EF14 = 1                    | Falling edge,<br>Rising edge,<br>Falling and Rising edge<br>or<br>H level | Pulses of less than 7/fc [s] are eliminated as noise. Pulses of 25/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.                     |
| INT5   | $\overline{\text{INT5}}$ | IMF • EF15 = 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: In NORMAL1/2 or IDLE1/2 mode, if a signal with no noise is input on an external interrupt pin, it takes a maximum of "signal establishment time + 6/fs[s]" from the input signal's edge to set the interrupt latch.

Note 2: When INTOEN = "0", IL4 is not set even if a falling edge is detected on the  $\overline{\text{INT0}}$  pin input.

Note 3: When a pin with more than one function is used as an output and a change occurs in data or input/output status, an interrupt request signal is generated in a pseudo manner. In this case, it is necessary to perform appropriate processing such as disabling the interrupt enable flag.

External Interrupt Control Register



|         |   |  |     |
|---------|---|--|-----|
| INT1NC  | Noise reject time select                        | 0: Pulses of less than $63/f_c$ [s] are eliminated as noise<br>1: Pulses of less than $15/f_c$ [s] are eliminated as noise | R/W |
| INT0EN  | P10/ $\overline{\text{INT0}}$ pin configuration | 0: P10 input/output port<br>1: $\overline{\text{INT0}}$ pin (Port P10 should be set to an input mode)                      | R/W |
| INT3 ES | INT3 edge select                                | 00: Rising edge<br>01: Falling edge<br>10: Rising edge and Falling edge<br>11: H level                                     | R/W |
| INT4 ES | INT4 edge select                                | 00: Rising edge<br>01: Falling edge<br>10: Rising edge and Falling edge<br>11: H level                                     | R/W |
| INT1 ES | INT1 edge select                                | 0: Rising edge<br>1: Falling edge  | R/W |

Note 1:  $f_c$ : 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).

Note 3: The maximum time from modifying INT1NC until a noise reject time is changed is  $2^6/f_c$ .

Note 4: In case  $\overline{\text{RESET}}$  pin is released while the state of INT3 pin keeps "H" level, the external interrupt 3 request is not generated even if the INT3 edge select is specified as "H" level. The rising edge is needed after  $\overline{\text{RESET}}$  pin is released.

Note 5: In case  $\overline{\text{RESET}}$  pin is released while the state of INT4 pin keeps "H" level, the external interrupt 4 request is not generated even if the INT4 edge select is specified as "H" level. The rising edge is needed after  $\overline{\text{RESET}}$  pin is released.



## 4. Special Function Register (SFR)

The TMP86FH09NG adopts the memory mapped I/O system, and all peripheral control and data transfers are performed through the special function register (SFR) or the data buffer register (DBR). The SFR is mapped on address 0000H to 003FH, DBR is mapped on address 0F80H to 0FFFH.

This chapter shows the arrangement of the special function register (SFR) and data buffer register (DBR) for TMP86FH09NG.

### 4.1 SFR

| Address | Read   | Write    |
|---------|--------|----------|
| 0000H   |        | P0DR     |
| 0001H   |        | P1DR     |
| 0002H   |        | P2DR     |
| 0003H   |        | P3DR     |
| 0004H   |        | Reserved |
| 0005H   |        | Reserved |
| 0006H   |        | Reserved |
| 0007H   |        | Reserved |
| 0008H   |        | Reserved |
| 0009H   |        | P1CR     |
| 000AH   |        | P3CR     |
| 000BH   |        | P0OUTCR  |
| 000CH   | P0PRD  | —        |
| 000DH   | P2PRD  | —        |
| 000EH   |        | ADCCR1   |
| 000FH   |        | ADCCR2   |
| 0010H   |        | TC1DRAL  |
| 0011H   |        | TC1DRAH  |
| 0012H   |        | TC1DRBL  |
| 0013H   |        | TC1DRBH  |
| 0014H   |        | TC1CR    |
| 0015H   |        | Reserved |
| 0016H   |        | Reserved |
| 0017H   |        | Reserved |
| 0018H   |        | Reserved |
| 0019H   |        | Reserved |
| 001AH   |        | TC3CR    |
| 001BH   |        | TC4CR    |
| 001CH   |        | TTREG3   |
| 001DH   |        | TTREG4   |
| 001EH   |        | PWREG3   |
| 001FH   |        | PWREG4   |
| 0020H   | ADCDR1 | —        |
| 0021H   | ADCDR2 | —        |
| 0022H   |        | Reserved |
| 0023H   |        | Reserved |
| 0024H   |        | Reserved |
| 0025H   | UARTSR | UARTCR1  |

| Address | Read     | Write   |
|---------|----------|---------|
| 0026H   | —        | UARTCR2 |
| 0027H   | RDBUF    | TDBUF   |
| 0028H   | SESR     | —       |
| 0029H   | SEDR     |         |
| 002AH   | SECR     |         |
| 002BH   | Reserved |         |
| 002CH   | Reserved |         |
| 002DH   | Reserved |         |
| 002EH   | Reserved |         |
| 002FH   | Reserved |         |
| 0030H   | Reserved |         |
| 0031H   | —        | STOPCR  |
| 0032H   | Reserved |         |
| 0033H   | Reserved |         |
| 0034H   | —        | WDTCR1  |
| 0035H   | —        | WDTCR2  |
| 0036H   | TBTCCR   |         |
| 0037H   | EINTCR   |         |
| 0038H   | SYSCR1   |         |
| 0039H   | STSCR2   |         |
| 003AH   | EIRL     |         |
| 003BH   | EIRH     |         |
| 003CH   | ILL      |         |
| 003DH   | ILH      |         |
| 003EH   | INTSEL   |         |
| 003FH   | PSW      |         |

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.).

4.2 DBR

| Address | Bit7 to Bit4 | Bit3 to Bit0 |
|---------|--------------|--------------|
| 0F80H   | Reserved     |              |
| :       | :            |              |
| 0FE8H   | Reserved     |              |
| 0FE9H   | —            | FLSSTB       |
| 0FEAH   | SPCR         |              |
| 0FEBH   | Reserved     |              |
| :       | :            |              |
| 0FFEH   | Reserved     |              |
| 0FFFH   | FLSCR        |              |

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.).



## 5. I/O Ports

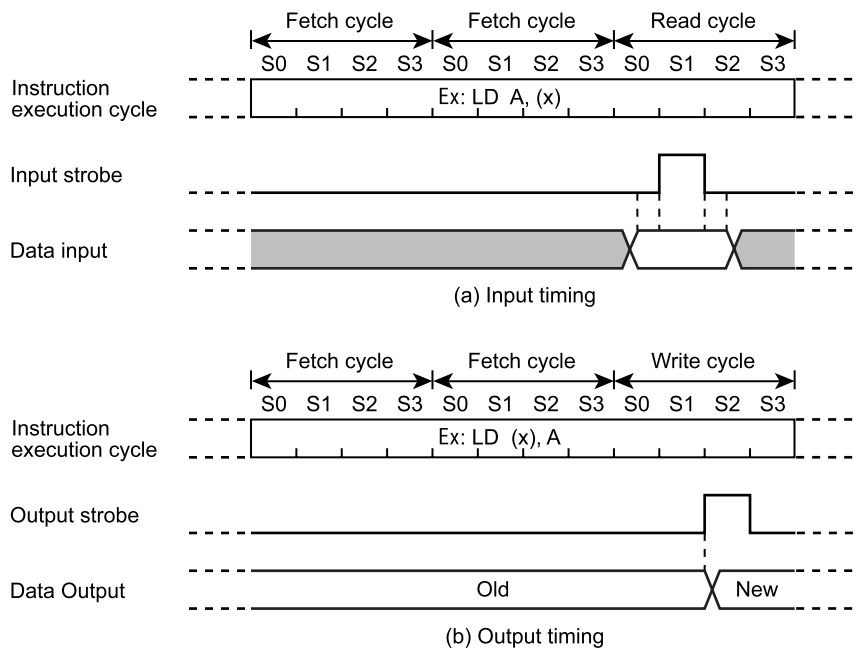
The TMP86FH09NG have 4 parallel input/output ports as follows.

|         | Primary Function | Secondary Functions   |
|---------|------------------|---|
| Port P0 | 8-bit I/O port   | External interrupt input, Timer/Counter input/output, serial interface input/output |
| Port P1 | 7-bit I/O port   | External interrupt input and divider output   |
| Port P2 | 3-bit I/O port   | External interrupt input and STOP mode release signal input                         |
| Port P3 | 8-bit I/O port   | Analog input, STOP mode release signal input and Timer/Counter input/output         |

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 timer before processing. Figure 5-1 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.



Note: The positions of the read and write cycles may vary, depending on the instruction.

Figure 5-1 Input/Output Timing (Example)

## 5.1 P0 (P07 to P00) Port (High Current)

The P0 port is an 8-bit input/output port shared with external interrupt input, SEI serial interface input/output, and UART and 16-bit timer counter input/output. When using this port as an input port or for external interrupt input, SEI serial interface input/output, or UART input/output, set the output latch to 1. When using this port as an output port, the output latch data (P0DR) is output to the P0 port.

When reset, the output latch (P0DR) and the push-pull control register (P0OUTCR) are initialized to 1 and 0, respectively.

The P0 port allows its output circuit to be selected between N-channel open-drain output or push-pull output by the P0OUTCR register.

When using this port as an input port, set the P0OUTCR register's corresponding bit to 0 after setting the P0DR to 1.

The P0 port has independent data input registers. To inspect the output latch status, read the P0DR register. To inspect the pin status, read the P0PRD register.

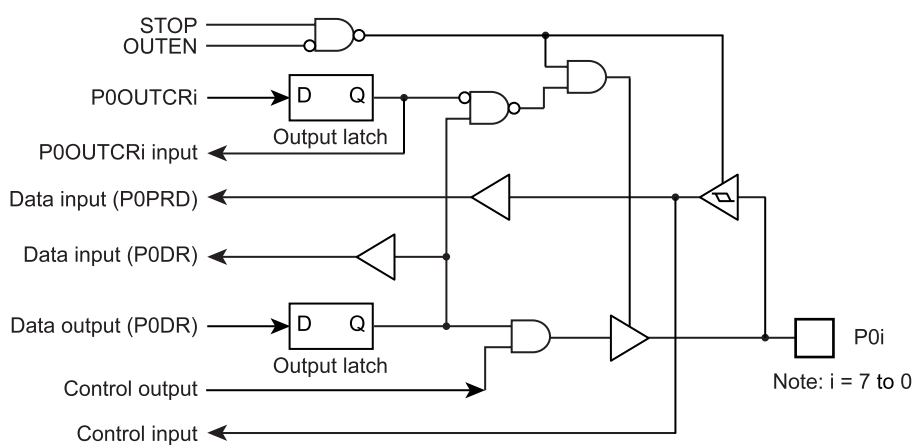


Figure 5-2 P0 Port

|                               | 7                  | 6                  | 5  | 4           | 3           | 2   | 1          | 0          |                            |
|-------------------------------|--------------------|--------------------|--|-------------|-------------|---|------------|------------|----------------------------|
| P0DR<br>(0000H)<br>R/W        | P07<br>TC1<br>INT4 | P06<br>INT3<br>PPG | P05<br>SS  | P04<br>MISO | P03<br>MOSI | P02<br>SCLK                                     | P01<br>RxD | P00<br>TxD | (Initial value: 1111 1111) |
| P0PRD<br>(000CH)<br>Read only | P07                | P06                | P05  | P04         | P03         | P02   | P01        | P00        |                            |
| P0OUTCR<br>(000BH)            | P0OUTCR            |                    | Controls P0 port input/output<br>(specified bitwise) |             |             | 0: Nch open-drain output<br>1: Push-pull output |            |            | R/W                        |

## 5.2 P1 (P16 to P10) Port

The P1 port is a 7-bit input/output port that can be specified for input or output bitwise. The P1 Port Input/output Control Register (P1CR) is used to specify this port for input or output. When reset, the P1CR register is initialized to 0, with the P1 port set for input mode. The P1 port output latch is initialized to 0.

The P1 port is shared with external interrupt input and divider output. When using the P1 port as function pin, set its input pins for input mode. For the output pins, first set their output latches to 1 before setting the pins for output mode.

Note that the P11 pin is an external interrupt input. (When used as an output port, its interrupt latch is set at the rising or falling edge.) The P10 pin can be used as an input/output port or an external interrupt input by selecting its function with the External Interrupt Control Register (INT0EN). When reset, the P10 pin is chosen to be an input port.

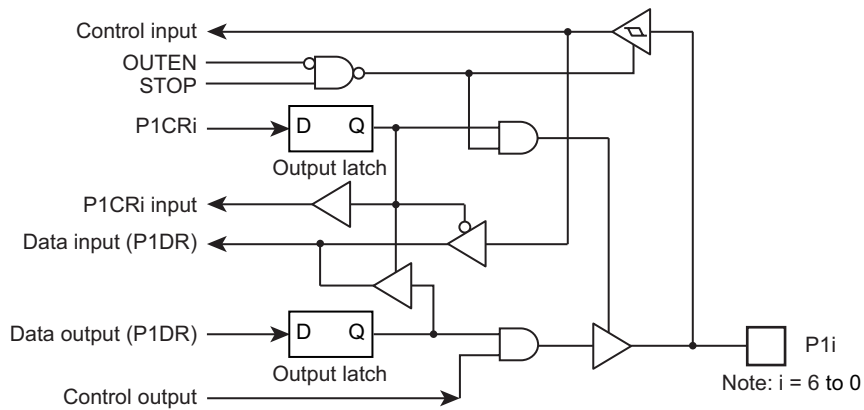


Figure 5-3 P1 Port

|                        |   |     |     |     |     |            |             |             |                            |
|------------------------|---|-----|-----|-----|-----|------------|-------------|-------------|----------------------------|
| P1DR<br>(0001H)<br>R/W | 7 | 6   | 5   | 4   | 3   | 2          | 1           | 0           |                            |
|                        |   | P16 | P15 | P14 | P13 | P12<br>DVO | P11<br>INT1 | P10<br>INT0 | (Initial value: ***0 0000) |
| P1CR<br>(0009H)        | 7 | 6   | 5   | 4   | 3   | 2          | 1           | 0           | (Initial value: ***0 0000) |
|                        |   |     |     |     |     |            |             |             |                            |

|      |   |                                 |     |
|------|---|---------------------------------|-----|
| P1CR | Controls P1 port input/output (specified bitwise) | 0: Input mode<br>1: Output mode | R/W |
|------|---|---------------------------------|-----|

### 5.3 P2 (P22 to P20) Port

The P2 port is a 3-bit input/output port shared with external interrupt input, STOP canceling signal input, and low-frequency resonator connecting pin. When using this port as an input port or function pin, set the output latch to 1. The output latch is initialized to 1 when reset. When operating in dual-clock mode, connect a low-frequency resonator (32.768 kHz) to the P21 (XTIN) and P22 (XTOUT) pins. When operating in single-clock mode, the P21 and P22 pins can be used as ordinary input/output ports. We recommend using the P20 pin for external interrupt input or STOP canceling signal input or as an input port. (When used as an output port, the interrupt latch is set by a falling edge.)

The P2 port has independent data input registers. To inspect the output latch status, read the P2DR register. To inspect the pin status, read the P2PRD register. When the P2DR or P2PRD read instruction is executed for the P2 port, the values read from bits 7 to 3 are indeterminate.

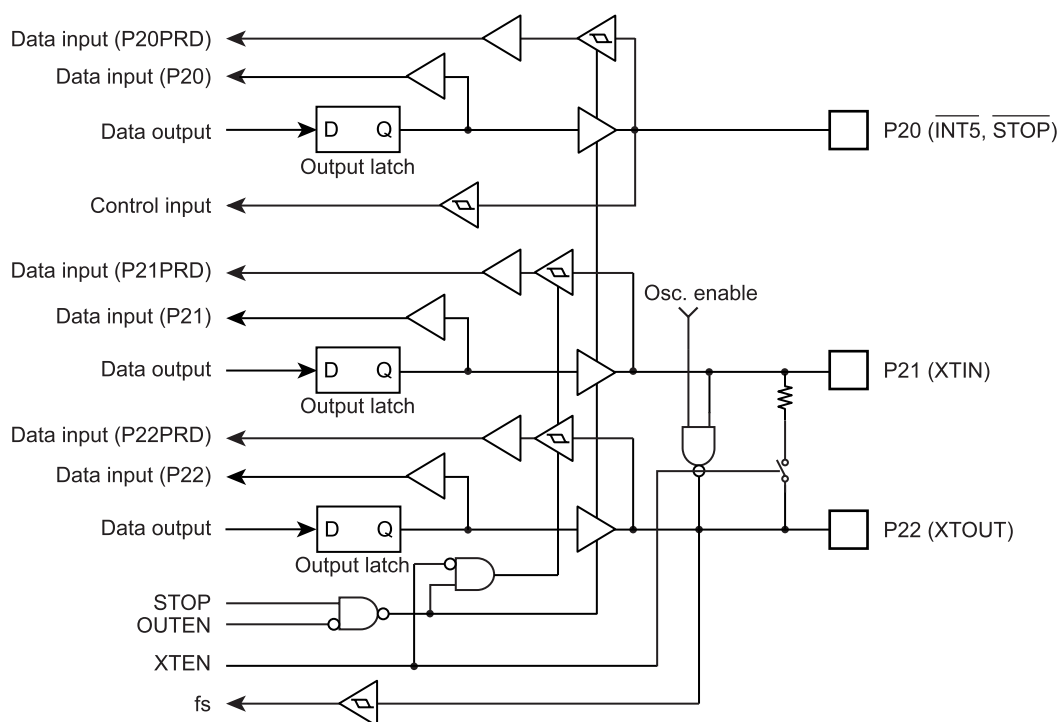


Figure 5-4 P2 Port

|                               |   |   |   |   |   |              |             |                     |                            |
|-------------------------------|---|---|---|---|---|--------------|-------------|---------------------|----------------------------|
|                               | 7 | 6 | 5 | 4 | 3 | 2            | 1           | 0                   |                            |
| P2DR<br>(0002H)<br>R/W        |   |   |   |   |   | P22<br>XTOUT | P21<br>XTIN | P20<br>INT5<br>STOP | (Initial value: **** *111) |
|                               | 7 | 6 | 5 | 4 | 3 | 2            | 1           | 0                   |                            |
| P2PRD<br>(000DH)<br>Read only |   |   |   |   |   | P22          | P21         | P20                 |                            |

Note: The P20 pin is shared with the  $\overline{\text{STOP}}$  pin, so that when in STOP mode, its output goes to a High-Z state regardless of the OUTEN status.

### 5.4 P3 (P37 to P30) Port

The P3 port is an 8-bit input/output port that can be specified for input or output bitwise, and is shared with analog input, key-on wakeup input, and 8-bit timer counter input/output. The P3 Port Input/output Control Register (P3CR) and ADCCR1<AINDS> are used to specify this port for input or output. When reset, the P3CR register and P3DR are cleared to 0, while AINDS is set to 1, so that P37 to P30 function as input port.

When using the P3 port as an input port, set AINDS = 1 while at the same time setting the P3CR register to 0.

When using the P3 port for analog input, set AINDS = 0 and the pins selected with ADCCR1<SAIN > are set for analog input no matter what values are set in the P3DR and P3CR. When using the P3 port as an output port, set the P3CR to 1 and the pin associated with that bit is set for output mode, so that P3DR (output latch data) is output from that pin.

When an input instruction is executed for the P3 port while using the AD converter, the pins selected for analog input read in the P3DR value into the internal circuit and those not selected for analog input read in a 1 or 0 according to the logic level on each pin. Even when an output instruction is executed, no latch data are forwarded to the pins selected for analog input.

Any pins of the P3 port which are not used for analog input can be used as input/output ports. During AD conversion, however, avoid executing output instructions on these ports, because this is necessary to maintain the accuracy of conversion. Also, during AD conversion, take care not to enter a rapidly changing signal to any port adjacent to analog input.

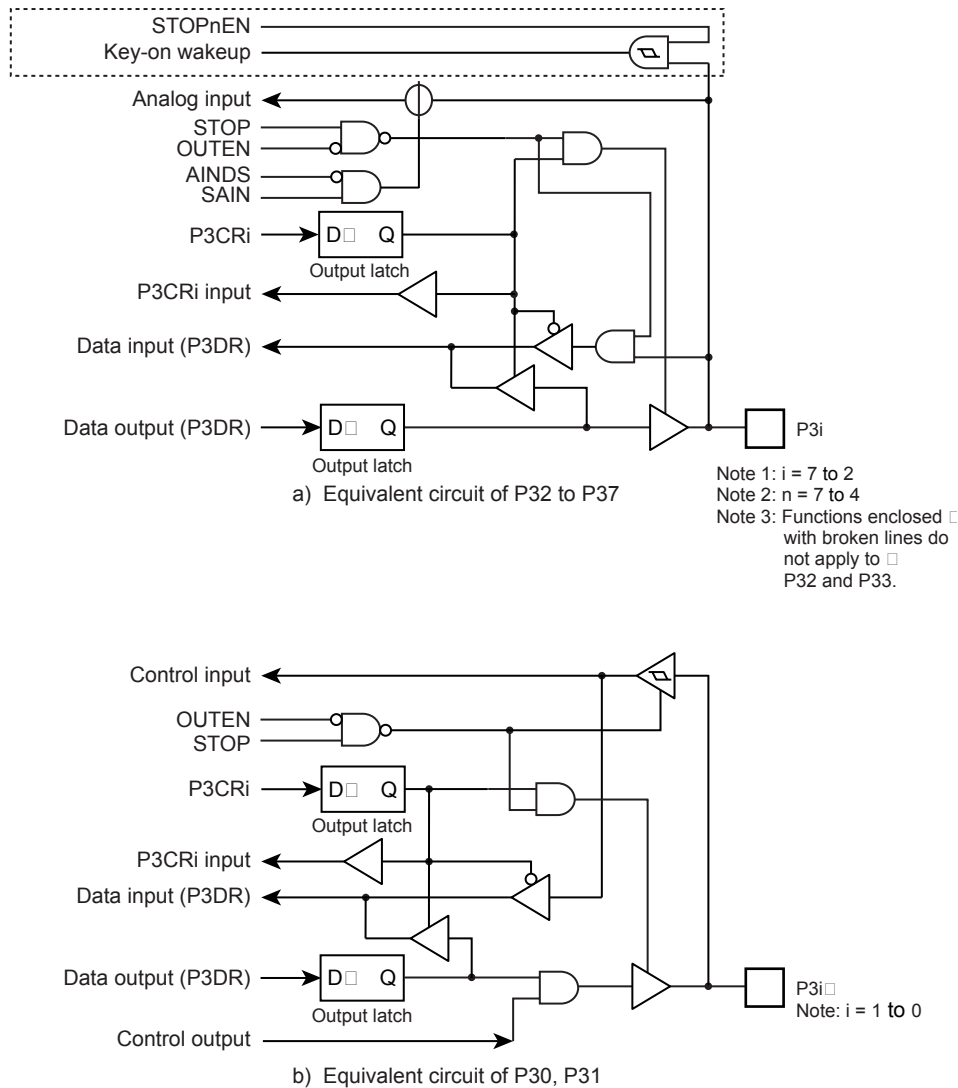


Figure 5-5 P3 Port

|                        |                      |                      |                      |                      |             |             |                                    |                            |                            |
|------------------------|----------------------|----------------------|----------------------|----------------------|-------------|-------------|------------------------------------|----------------------------|----------------------------|
|                        | 7                    | 6                    | 5                    | 4                    | 3           | 2           | 1                                  | 0                          |                            |
| P3DR<br>(0003H)<br>R/W | P37<br>AIN5<br>STOP5 | P36<br>AIN4<br>STOP4 | P35<br>AIN3<br>STOP3 | P34<br>AIN2<br>STOP2 | P33<br>AIN1 | P32<br>AIN0 | P31<br>TC4<br>PD04<br>PWM4<br>PPG4 | P30<br>TC3<br>PD03<br>PWM3 | (Initial value: 0000 0000) |

|                 |   |   |   |   |   |   |   |   |                            |
|-----------------|---|---|---|---|---|---|---|---|----------------------------|
| P3CR<br>(000AH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | (Initial value: 0000 0000) |
|                 |   |   |   |   |   |   |   |   |                            |

|      |   |                                 |     |
|------|---|---------------------------------|-----|
| P3CR | Controls P3 port output (specified bitwise) | 0: Input mode<br>1: Output mode | R/W |
|------|---|---------------------------------|-----|

&lt;P3 Port Input/Output&gt;

|       | Analog Input Mode | Input Mode | Output Mode |
|-------|-------------------|------------|-------------|
| P3CR  | 0                 |            | 1           |
| AINDS | 0                 | 1          |             |
| P3DR  | 0                 | *          |             |

Note 1: When using the port for key-on wakeup input (STOP2 to 5), set the P3CR register's corresponding bits to 0.

Note 2: P30 and P31 are hysteresis inputs. P34 to P37 become hysteresis inputs only during key-on wakeup.

Note 3: Input status on ports set for input mode are read in into the internal circuit. Therefore, when using the ports in a mixture of input and output modes, the contents of the output latches for the ports that are set for input mode may be rewritten by execution of bit manipulating instructions.

## 6. 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).

### 6.1 Time Base Timer

#### 6.1.1 Configuration

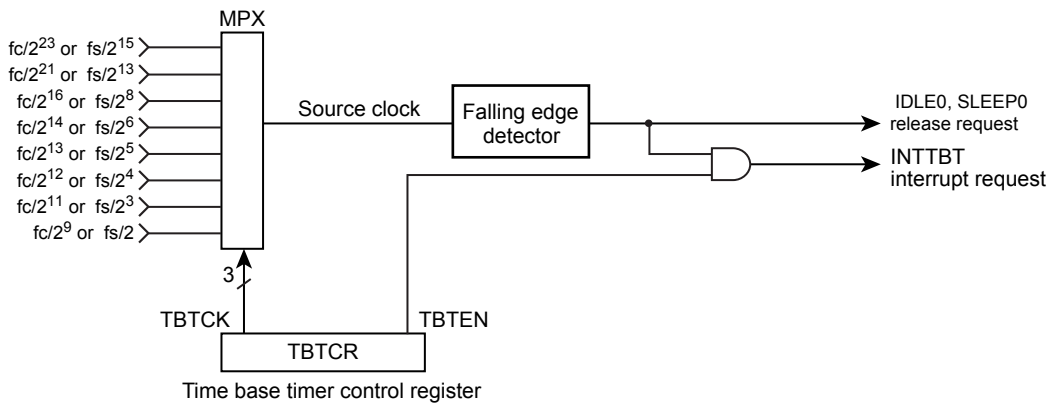


Figure 6-1 Time Base Timer configuration

#### 6.1.2 Control

Time Base Timer is controlled by Time Base Timer control register (TBTCR).

#### Time Base Timer Control Register

|                  |         |         |         |       |      |   |   |   |                            |
|------------------|---------|---------|---------|-------|------|---|---|---|----------------------------|
|                  | 7       | 6       | 5       | 4     | 3    | 2 | 1 | 0 |                            |
| TBTCR<br>(0036H) | (DVOEN) | (DVOCK) | (DV7CK) | TBTEN | TBTC |   |   |   | (Initial Value: 0000 0000) |

| TBTEN | Time Base Timer enable / disable                  | 0: Disable<br>1: Enable |             |                             |     |             |
|-------|---|-------------------------|-------------|-----------------------------|-----|-------------|
| TBTC  | Time Base Timer interrupt Frequency select : [Hz] | NORMAL1/2, IDLE1/2 Mode |             | SLOW1/2<br>SLEEP1/2<br>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 1: fc; High-frequency clock [Hz], fs; Low-frequency clock [Hz], \*; Don't care

Note 2: The interrupt frequency (TBTCK) must be selected with the time base timer disabled (TBTEN="0"). (The interrupt frequency must not be changed with the disable from the enable state.) Both frequency selection and enabling can be performed simultaneously.

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

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

Table 6-1 Time Base Timer Interrupt Frequency ( Example :  $fc = 16.0$  MHz,  $fs = 32.768$  kHz )

| TBTCK | Time Base Timer Interrupt Frequency [Hz] |                         |                        |
|-------|--|-------------------------|------------------------|
|       | NORMAL1/2, IDLE1/2 Mode                  | NORMAL1/2, IDLE1/2 Mode | SLOW1/2, SLEEP1/2 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                   | –                      |

### 6.1.3 Function

An INTTBT ( Time Base Timer Interrupt ) is generated on the first falling edge of source clock ( The divider output of the timing generato which is selected by TBTCK. ) after 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 6-2 ).

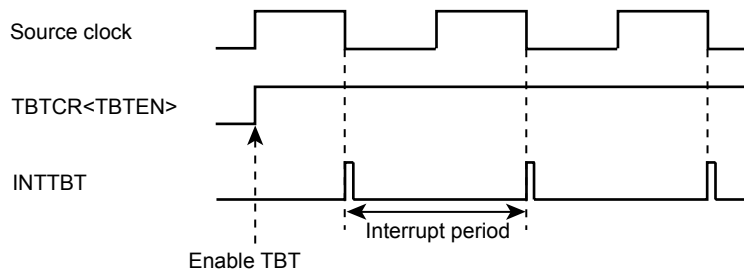


Figure 6-2 Time Base Timer Interrupt

## 6.2 Divider Output ( $\overline{DVO}$ )

Approximately 50% duty pulse can be output using the divider output circuit, which is useful for piezoelectric buzzer drive. Divider output is from  $\overline{DVO}$  pin.

### 6.2.1 Configuration

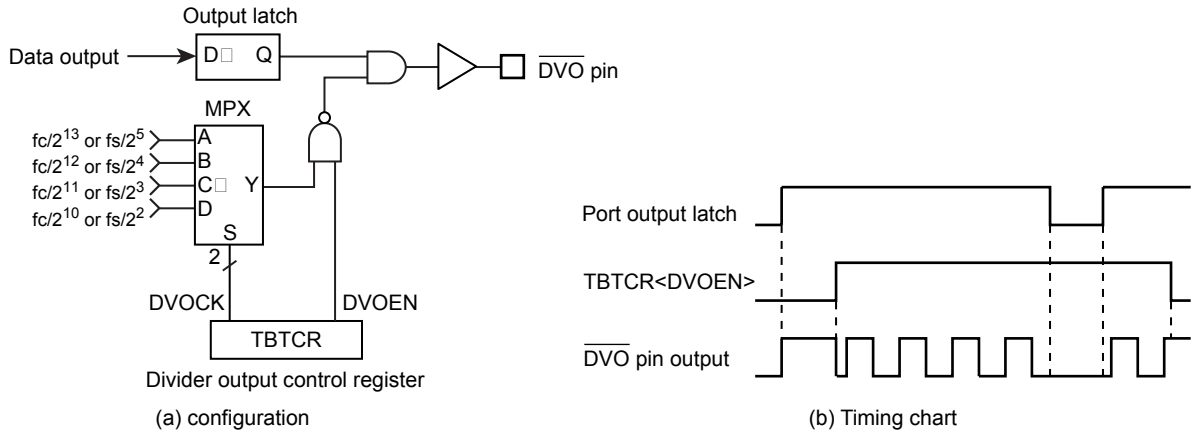
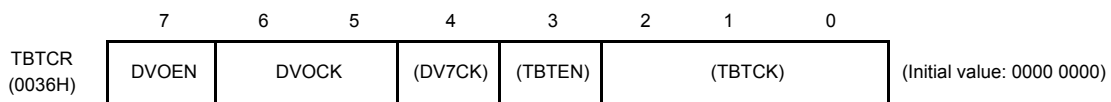


Figure 6-3 Divider Output

### 6.2.2 Control

The Divider Output is controlled by the Time Base Timer Control Register.

#### Time Base Timer Control Register



| DVOEN | Divider output enable / disable                                  | 0: Disable<br>1: Enable   |             |                               | R/W |          |
|-------|--|---------------------------|-------------|-------------------------------|-----|----------|
| DVOCK | Divider Output ( $\overline{DVO}$ )<br>frequency selection: [Hz] | NORMAL 1/2, IDLE 1/2 Mode |             | SLOW 1/2<br>SLEEP 1/2<br>Mode | R/W |          |
|       |  | DV7CK = 0                 | DV7CK = 1   |                               |     |          |
|       |  | 00                        | $fc/2^{13}$ | $fs/2^5$                      |     | $fs/2^5$ |
|       |  | 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: Selection of divider output frequency (DVOCK) must be made while divider output is disabled (DVOEN="0"). Also, in other words, when changing the state of the divider output frequency from enabled (DVOEN="1") to disabled (DVOEN="0"), do not change the setting of the divider output frequency.

Example : 1.95 kHz pulse output (fc = 16.0 MHz)

```
LD      (TBTCR), 00000000B      ; DVOCK ← "00"
LD      (TBTCR), 10000000B      ; DVOEN ← "1"
```

Table 6-2 Divider Output Frequency ( Example : fc = 16.0 MHz, fs = 32.768 kHz )

| DVOCK | Divider Output Frequency [Hz] |           |                        |
|-------|-------------------------------|-----------|------------------------|
|       | NORMAL1/2, IDLE1/2 Mode       |           | SLOW1/2, SLEEP1/2 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                |

## 7. Watchdog Timer (WDT)

The watchdog timer is a fail-safe system to detect rapidly the CPU malfunctions such as endless loops due to spurious noises or the deadlock conditions, and return the CPU to a system recovery routine.

The watchdog timer signal for detecting malfunctions can be programmed only once as “reset request” or “interrupt request”. Upon the reset release, this signal is initialized to “reset request”.

When the watchdog timer is not used to detect malfunctions, it can be used as the timer to provide a periodic interrupt.

Note: Care must be taken in system design since the watchdog timer functions are not be operated completely due to effect of disturbing noise.

### 7.1 Watchdog Timer Configuration

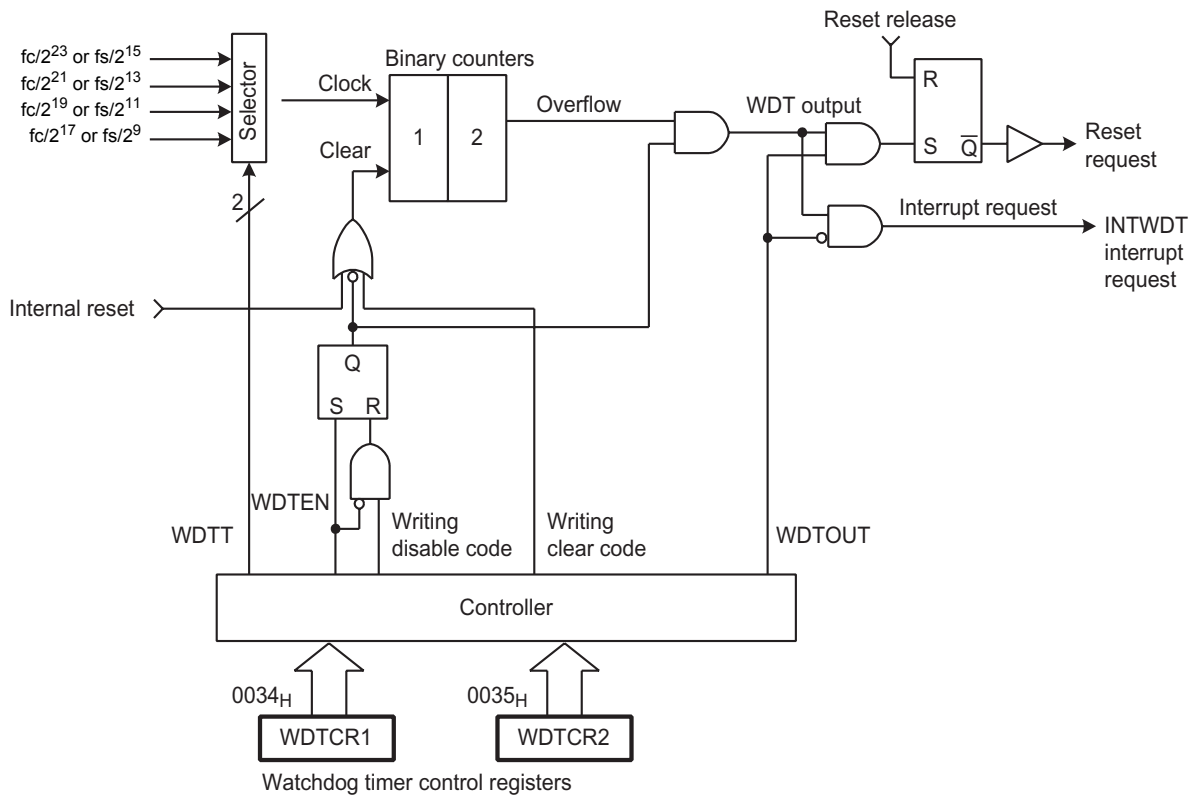


Figure 7-1 Watchdog Timer Configuration

## 7.2 Watchdog Timer Control

The watchdog timer is controlled by the watchdog timer control registers (WDTCR1 and WDTCR2). The watchdog timer is automatically enabled after the reset release.

### 7.2.1 Malfunction Detection Methods Using the Watchdog Timer

The CPU malfunction is detected, as shown below.

1. Set the detection time, select the output, and clear the binary counter.
2. Clear the binary counter repeatedly within the specified detection time.

If the CPU malfunctions such as endless loops or the deadlock conditions occur for some reason, the watchdog timer output is activated by the binary-counter overflow unless the binary counters are cleared. When WDTCR1<WDTOUT> is set to “1” at this time, the reset request is generated and then internal hardware is initialized. When WDTCR1<WDTOUT> is set to “0”, a watchdog timer interrupt (INTWDT) is generated.

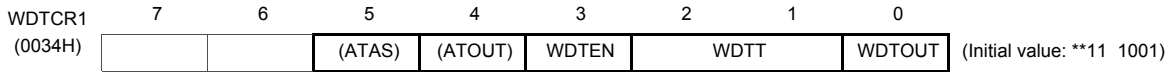
The watchdog timer temporarily stops counting in the STOP mode including the warm-up or IDLE/SLEEP mode, and automatically restarts (continues counting) when the STOP/IDLE/SLEEP mode is inactivated.

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

Example :Setting the watchdog timer detection time to  $2^{21}/f_c$  [s], and resetting the CPU malfunction detection

|                                     |    |                     |  |
|-------------------------------------|----|---------------------|--|
|                                     | LD | (WDTCR2), 4EH       | : Clears the binary counters.  |
|                                     | LD | (WDTCR1), 00001101B | : WDTT ← 10, WDTOUT ← 1  |
| Within 3/4 of WDT<br>detection time | ┌  | LD                  | (WDTCR2), 4EH : Clears the binary counters (always clears immediately before and after changing WDTT). |
|                                     |    | :                   |  |
|                                     |    | :                   |  |
| Within 3/4 of WDT<br>detection time | └  | LD                  | (WDTCR2), 4EH : Clears the binary counters.  |
|                                     |    | :                   |  |
|                                     |    | LD                  | (WDTCR2), 4EH : Clears the binary counters.  |

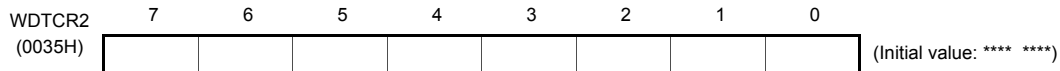
Watchdog Timer Control Register 1



|        |                                   |   |                     |                     |                     |            |
|--------|-----------------------------------|---|---------------------|---------------------|---------------------|------------|
| WDTEN  | Watchdog timer enable/disable     | 0: Disable (Writing the disable code to WDTCR2 is required.)<br>1: Enable | Write only          |                     |                     |            |
| WDTT   | Watchdog timer detection time [s] | 00  | 2 <sup>25</sup> /fc | 2 <sup>17</sup> /fs | 2 <sup>17</sup> /fs | Write only |
|        |                                   | 01  | 2 <sup>23</sup> /fc | 2 <sup>15</sup> /fs | 2 <sup>15</sup> /fs |            |
|        |                                   | 10  | 2 <sup>21</sup> /fc | 2 <sup>13</sup> /fs | 2 <sup>13</sup> /fs |            |
|        |                                   | 11  | 2 <sup>19</sup> /fc | 2 <sup>11</sup> /fs | 2 <sup>11</sup> /fs |            |
|        |                                   | NORMAL1/2 mode  |                     | SLOW1/2 mode        |                     |            |
|        |                                   | DV7CK = 0   | DV7CK = 1           |                     |                     |            |
| WDTOUT | Watchdog timer output select      | 0: Interrupt request<br>1: Reset request                                  |                     |                     |                     | Write only |

- Note 1: After clearing WDTOUT to “0”, the program cannot set it to “1”.
- 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. If WDTCR1 is read, a don't care is read.
- Note 4: To activate the STOP mode, disable the watchdog timer or clear the counter immediately before entering the STOP mode. After clearing the counter, clear the counter again immediately after the STOP mode is inactivated.
- Note 5: To clear WDTEN, set the register in accordance with the procedures shown in “1.2.3 Watchdog Timer Disable”.

Watchdog Timer Control Register 2



|        |                                   |   |            |
|--------|-----------------------------------|---|------------|
| WDTCR2 | Write Watchdog timer control code | 4EH: Clear the watchdog timer binary counter (Clear code)<br>B1H: Disable the watchdog timer (Disable code)<br>D2H: Enable assigning address trap area<br>Others: Invalid | Write only |
|--------|-----------------------------------|---|------------|

- Note 1: The disable code is valid only 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 the clear code 4EH using a cycle shorter than 3/4 of the time set in WDTCR1<WDTT>.

7.2.2 Watchdog Timer Enable

Setting WDTCR1<WDTEN> to “1” enables the watchdog timer. Since WDTCR1<WDTEN> is initialized to “1” during reset, the watchdog timer is enabled automatically after the reset release.

### 7.2.3 Watchdog Timer Disable

To disable the watchdog timer, set the register in accordance with the following procedures. Setting the register in other procedures causes a malfunction of the microcontroller.

1. Set the interrupt master flag (IMF) to “0”.
2. Set WDTCR2 to the clear code (4EH).
3. Set WDTCR1<WDTEN> to “0”.
4. Set WDTCR2 to the disable code (B1H).

Note: While the watchdog timer is disabled, the binary counters of the watchdog timer are cleared.

Example :Disabling the watchdog timer

```
DI                : IMF ← 0
LD                (WDTCR2), 04EH    : Clears the binary coutner
LDW               (WDTCR1), 0B101H  : WDTEN ← 0, WDTCR2 ← Disable code
```

Table 7-1 Watchdog Timer Detection Time (Example: fc = 16.0 MHz, fs = 32.768 kHz)

| TC1CK | NORMAL1/2, IDLE1/2 mode |           | SLOW, SLEEP mode |
|-------|-------------------------|-----------|------------------|
|       | DV7CK = 0               | DV7CK = 1 |                  |
| 00    | 2.097                   | 4         | 244.14           |
| 01    | 524.288 m               | 1         | –                |
| 10    | 131.072 m               | 250 m     | –                |
| 11    | 32.768 m                | 62.5 m    | –                |

### 7.2.4 Watchdog Timer Interrupt (INTWDT)

When WDTCR1<WDTOUT> is cleared to “0”, a watchdog timer interrupt request (INTWDT) is generated by the binary-counter overflow.

A watchdog timer interrupt is the non-maskable interrupt which can be accepted regardless of the interrupt master flag (IMF).

When a watchdog timer interrupt is generated while the other interrupt including a watchdog timer interrupt is already accepted, the new watchdog timer interrupt is processed immediately and the previous interrupt is held pending. Therefore, if watchdog timer interrupts are generated continuously without execution of the RETN instruction, too many levels of nesting may cause a malfunction of the microcontroller.

To generate a watchdog timer interrupt, set the stack pointer before setting WDTCR1<WDTOUT>.

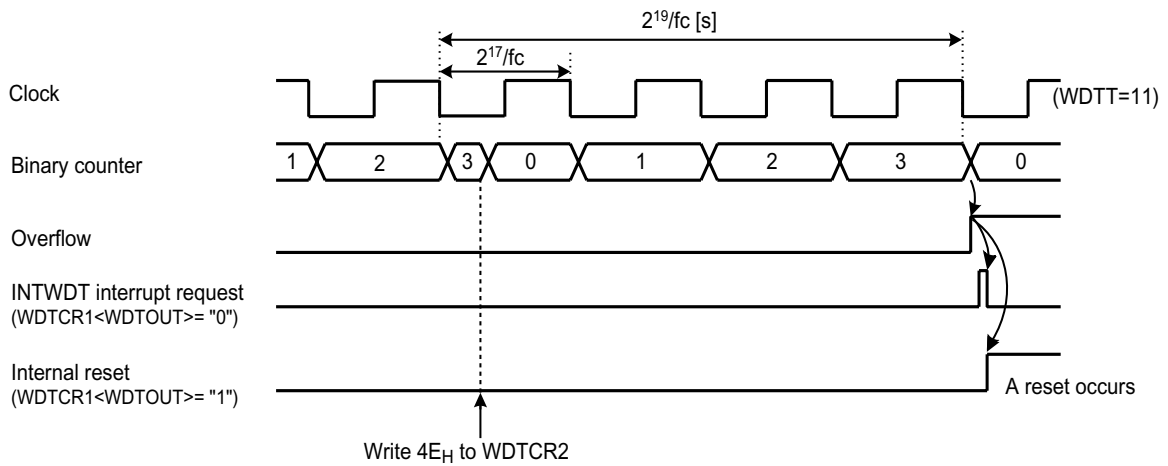
Example :Setting watchdog timer interrupt

```
LD                SP, 013FH        : Sets the stack pointer
LD                (WDTCR1), 00001000B : WDTOUT ← 0
```

### 7.2.5 Watchdog Timer Reset

When a binary-counter overflow occurs while WDTCR1<WDTOUT> is set to “1”, a watchdog timer reset request is generated. When a watchdog timer reset request is generated, the internal hardware is reset. The reset time is  $8/f_c$  to  $24/f_c$  [s] ( $0.5$  to  $1.5 \mu\text{s}$  @  $f_c = 16.0 \text{ MHz}$ ).

Note: When a watchdog timer reset is generated in the SLOW1 mode, the reset time is  $8/f_c$  to  $24/f_c$  (high-frequency clock) since the high-frequency clock oscillator is restarted. However, when crystals have inaccuracies upon start of the high-frequency clock oscillator, the reset time should be considered as an approximate value because it has slight errors.



## 7.3 Address Trap

The Watchdog Timer Control Register 1 and 2 share the addresses with the control registers to generate address traps.

### Watchdog Timer Control Register 1

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

|       |   |  |            |
|-------|---|--|------------|
| ATAS  | Select address trap generation in the internal RAM area | 0: Generate no address trap<br>1: Generate address traps (After setting ATAS to "1", writing the control code D2H to WDTCR2 is required) | Write only |
| ATOUT | Select operation at address trap                        | 0: Interrupt request<br>1: Reset request   |            |

### Watchdog Timer Control Register 2

|                   |   |   |   |   |   |   |   |   |                           |
|-------------------|---|---|---|---|---|---|---|---|---------------------------|
| WDTCR2<br>(0035H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |                           |
|                   |   |   |   |   |   |   |   |   | (Initial value: **** ***) |

|        |  |  |            |
|--------|--|--|------------|
| WDTCR2 | Write Watchdog timer control code and address trap area control code | D2H: Enable address trap area selection (ATRAP control code)<br>4EH: Clear the watchdog timer binary counter (WDT clear code)<br>B1H: Disable the watchdog timer (WDT disable code)<br>Others: Invalid | Write only |
|--------|--|--|------------|

#### 7.3.1 Selection of Address Trap in Internal RAM (ATAS)

WDTCR1<ATAS> specifies whether or not to generate address traps in the internal RAM area. To execute an instruction in the internal RAM area, clear WDTCR1<ATAS> to "0". To enable the WDTCR1<ATAS> setting, set WDTCR1<ATAS> and then write D2H to WDTCR2.

Executing an instruction in the SFR or DBR area generates an address trap unconditionally regardless of the setting in WDTCR1<ATAS>.

#### 7.3.2 Selection of Operation at Address Trap (ATOUT)

When an address trap is generated, either the interrupt request or the reset request can be selected by WDTCR1<ATOUT>.

#### 7.3.3 Address Trap Interrupt

When a binary-counter overflow occurs during WDTCR1<ATOUT> set to "0", an address trap interrupt request (INTADT) is generated.

An address trap interrupt is a non-maskable interrupt which can be accepted regardless of the interrupt master flag (IMF).

When an address trap interrupt is generated while the other interrupt including a watchdog timer interrupt is already accepted, the new address trap is processed immediately and the previous interrupt is held pending. Therefore, if address trap interrupts are generated continuously without execution of the RETN instruction, too many levels of nesting may cause a malfunction of the microcontroller.

To generate address trap interrupts, set the stack pointer beforehand.

### 7.3.4 Address Trap Reset

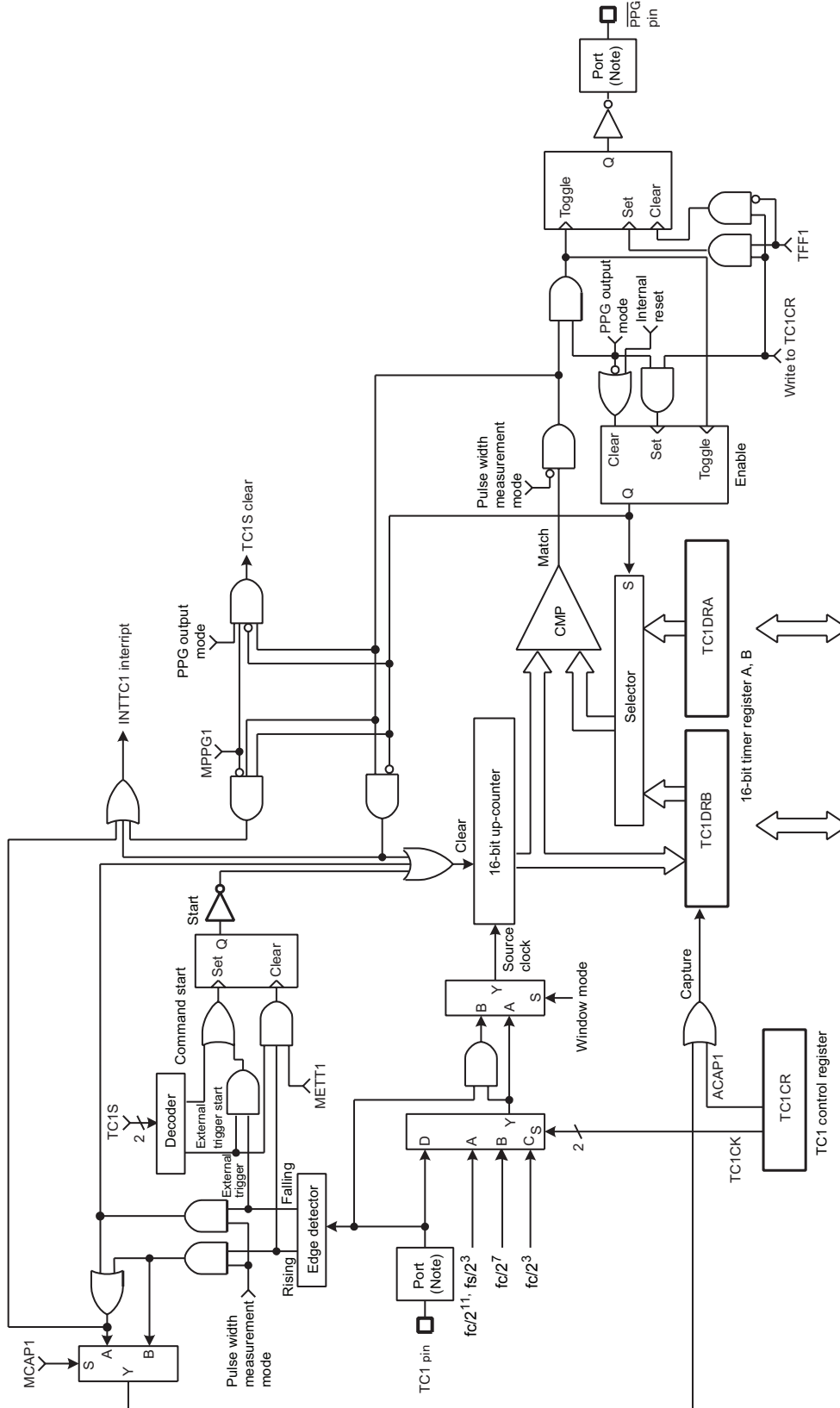
When a binary-counter overflow occurs while WDTCR1<WDTOUT> is set to “1”, an address trap reset request is generated. When an address trap reset request is generated, the internal hardware is reset. The reset time is  $8/f_c$  to  $24/f_c$  [s] ( $0.5$  to  $1.5 \mu\text{s}$  @  $f_c = 16.0 \text{ MHz}$ ).

Note: When an address trap reset is generated in the SLOW1 mode, the reset time is  $8/f_c$  to  $24/f_c$  (high-frequency clock) since the high-frequency clock oscillator is restarted. However, when crystals have inaccuracies upon start of the high-frequency clock oscillator, the reset time should be considered as an approximate value because it has slight errors.



## 8. 16-Bit TimerCounter 1 (TC1)

### 8.1 Configuration



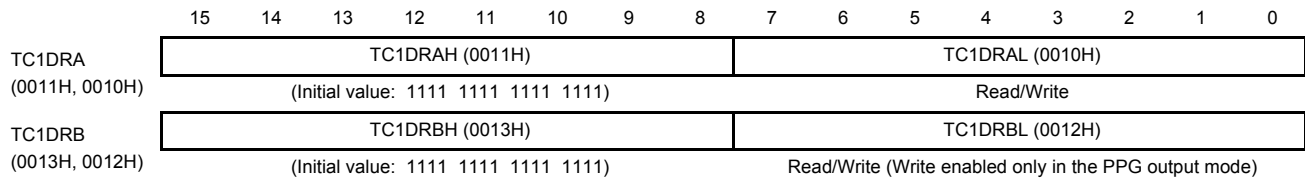
Note: Function I/O may not operate depending on I/O port setting. For more details, see the chapter "I/O Port".

Figure 8-1 TimerCounter 1 (TC1)

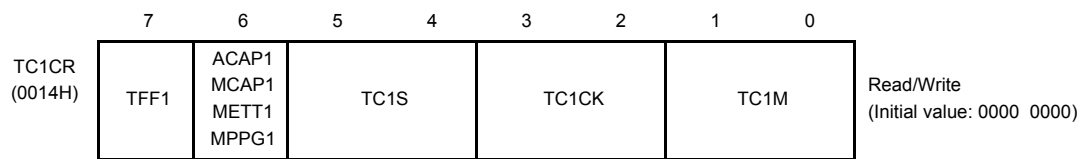
## 8.2 TimerCounter Control

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

### Timer Register



### TimerCounter 1 Control Register



|   |                                      |   |             |                           |       |         |          |                        |     |
|---|--------------------------------------|---|-------------|---------------------------|-------|---------|----------|------------------------|-----|
| TFF1  | Timer F/F1 control                   | 0: Clear  | 1: Set      | R/W                       |       |         |          |                        |     |
| ACAP1   | Auto capture control                 | 0: Auto-capture disable   |             | 1: Auto-capture enable    |       |         |          |                        |     |
| MCAP1   | 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               |       |         |          |                        |     |
| TC1S  | TC1 start control                    |   | Timer       | Extrig-ger                | Event | Win-dow | Pulse    | PPG                    | R/W |
|   |                                      | 00: Stop and counter clear  | 0           | 0                         | 0     | 0       | 0        | 0                      |     |
|   |                                      | 01: Command start   | 0           | -                         | -     | -       | -        | 0                      |     |
|   |                                      | 10: Rising edge start<br>(Ex-trigger/Pulse/PPG)<br>Rising edge count (Event)<br>Positive logic count (Window)   | -           | 0                         | 0     | 0       | 0        | 0                      |     |
| 11: Falling edge start<br>(Ex-trigger/Pulse/PPG)<br>Falling edge count (Event)<br>Negative logic count (Window) | -                                    | 0   | 0           | 0                         | 0     | 0       |          |                        |     |
| TC1CK   | TC1 source clock select [Hz]         | NORMAL1/2, IDLE1/2 mode   |             |                           |       |         | Divider  | SLOW,<br>SLEEP<br>mode | R/W |
|   |                                      | DV7CK = 0   |             | DV7CK = 1                 |       |         |          |                        |     |
|   |                                      | 00  | $fc/2^{11}$ | $fs/2^3$                  |       | DV9     | $fs/2^3$ |                        |     |
|   |                                      | 01  | $fc/2^7$    | $fc/2^7$                  |       | DV5     | -        |                        |     |
| 10  | $fc/2^3$                             | $fc/2^3$  |             | DV1                       | -     |         |          |                        |     |
| 11  | External clock (TC1 pin input)       |   |             |                           |       |         |          |                        |     |
| TC1M  | TC1 operating mode select            | 00: Timer/external trigger timer/event counter mode<br>01: Window mode<br>10: Pulse width measurement mode<br>11: PPG (Programmable pulse generate) output mode |             |                           |       |         |          | R/W                    |     |

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 becomes valid at the rising edge of the first source clock pulse that occurs after the upper byte (TC1DRAH and TC1DRBH) is written. Therefore, write the lower byte and the upper byte in this order (it is recommended to write the register with a 16-bit access instruction). Writing only the lower byte (TC1DRAL and TC1DRBL) does not enable the setting of the timer register.

Note 3: To set the mode, source clock, PPG output control and timer F/F control, write to TC1CR1 during TC1S=00. Set the timer F/F1 control until the first timer start after setting the PPG mode.

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

Note 5: To set the timer registers, the following relationship must be satisfied.

TC1DRA > TC1DRB > 1 (PPG output mode), TC1DRA > 1 (other modes)

Note 6: Set TFF1 to "0" in the mode except PPG output mode.

Note 7: Set TC1DRB after setting TC1M to the PPG output mode.

Note 8: When the STOP mode is entered, the start control (TC1S) is cleared to "00" automatically, and the timer stops. After the STOP mode is exited, set the TC1S to use the timer counter 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.

## 8.3 Function

TimerCounter 1 has six types of operating modes: timer, external trigger timer, event counter, window, pulse width measurement, programmable pulse generator output modes.

### 8.3.1 Timer mode

In the timer mode, the up-counter counts up using the internal clock. When a match between the up-counter and the timer register 1A (TC1DRA) value is detected, an INTTC1 interrupt is generated and the up-counter is cleared. After being cleared, the up-counter restarts counting. Setting TC1CR<ACAP1> to "1" captures the up-counter value into the timer register 1B (TC1DRB) with the 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 8-1 Internal Source Clock for TimerCounter 1 (Example:  $f_c = 16$  MHz,  $f_s = 32.768$  kHz)

| TC1CK | NORMAL1/2, IDLE1/2 mode |                          |                 |                          | SLOW, SLEEP 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 :Setting the timer mode with source clock  $f_c/2^{11}$  [Hz] and generating an interrupt 1 second later ( $f_c = 16$  MHz, TBTCR<DV7CK> = "0")

```
LDW      (TC1DRA), 1E84H      : Sets the timer register ( $1\text{ s} \div 2^{11}/f_c = 1\text{E84H}$ )
DI       : IMF="0"
SET      (EIRL), 7           : Enables INTTC1
EI       : IMF="1"
LD       (TC1CR), 00000000B   : Selects the source clock and mode
LD       (TC1CR), 00010000B   : Starts TC1
```

Example 2 :Auto-capture

```
LD       (TC1CR), 01010000B   : ACAP1 ← 1
:       :
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.

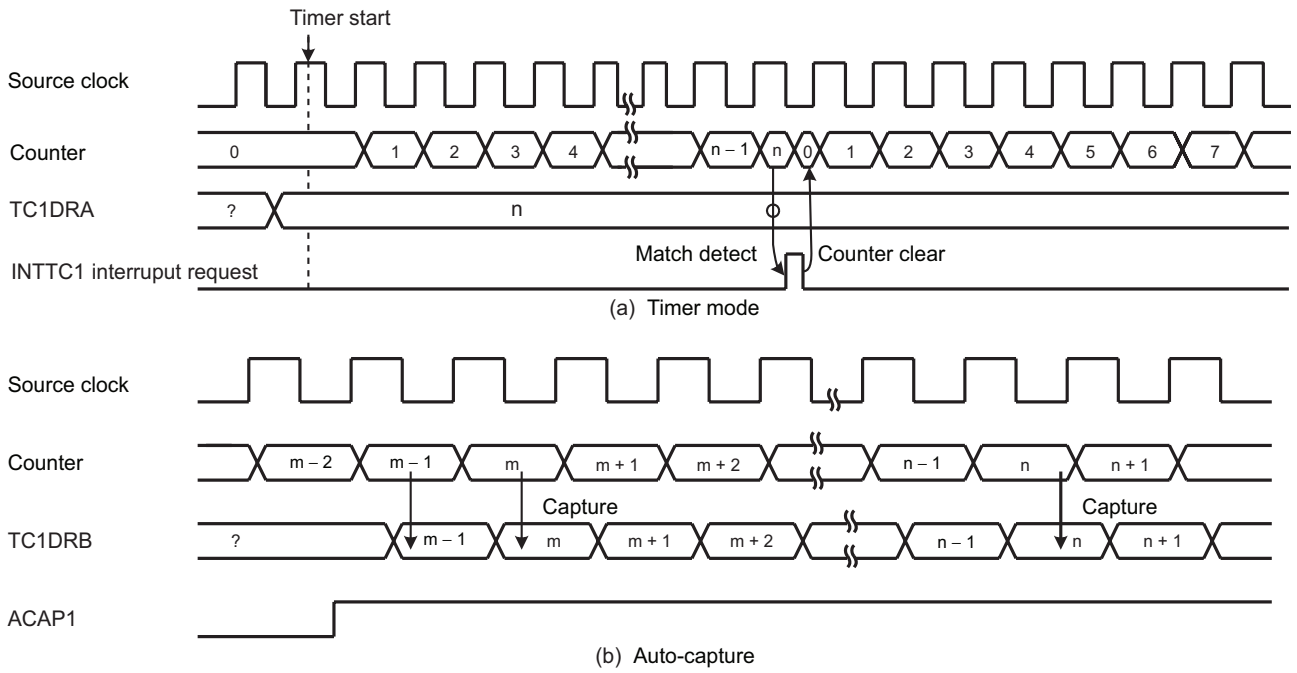


Figure 8-2 Timer Mode Timing Chart

### 8.3.2 External Trigger Timer Mode

In the external trigger timer mode, the up-counter starts counting by the input pulse triggering of the TC1 pin, and counts up at the edge of the internal clock. For the trigger edge used to start counting, either the rising or falling edge is defined in TC1CR<TCIS>.

- When TC1CR<METT1> is set to “1” (trigger start and stop)

When a match between the up-counter and the TC1DRA value is detected after the timer starts, the up-counter is cleared and halted and an INTTC1 interrupt request is generated.

If the edge opposite to trigger edge is detected before detecting a match between the up-counter and the TC1DRA, the up-counter is cleared and halted without generating an interrupt request. Therefore, this mode can be used to detect exceeding the specified pulse by interrupt.

After being halted, the up-counter restarts counting when the trigger edge is detected.

- When TC1CR<METT1> is set to “0” (trigger start)

When a match between the up-counter and the TC1DRA value is detected after the timer starts, the up-counter is cleared and halted and an INTTC1 interrupt request is generated.

The edge opposite to the trigger edge has no effect in count up. The trigger edge for the next counting is ignored if detecting it before detecting a match between the up-counter and the TC1DRA.

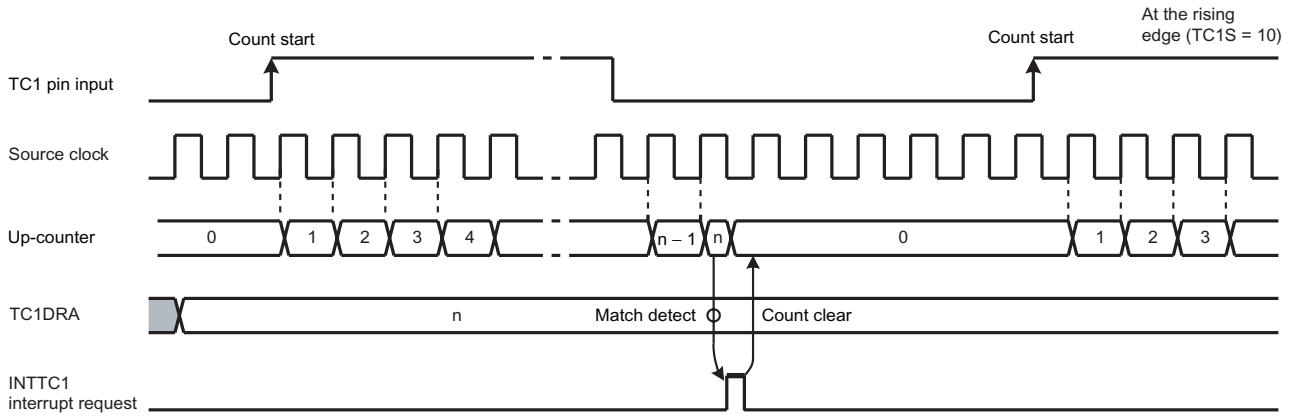
Since the TC1 pin input has the noise rejection, pulses of  $4/f_c$  [s] or less are rejected as noise. A pulse width of  $12/f_c$  [s] or more is required to ensure edge detection. The rejection circuit is turned off in the SLOW1/2 or SLEEP1/2 mode, but a pulse width of one machine cycle or more is required.

Example 1 :Generating an interrupt 1 ms after the rising edge of the input pulse to the TC1 pin  
( $f_c = 16$  MHz)

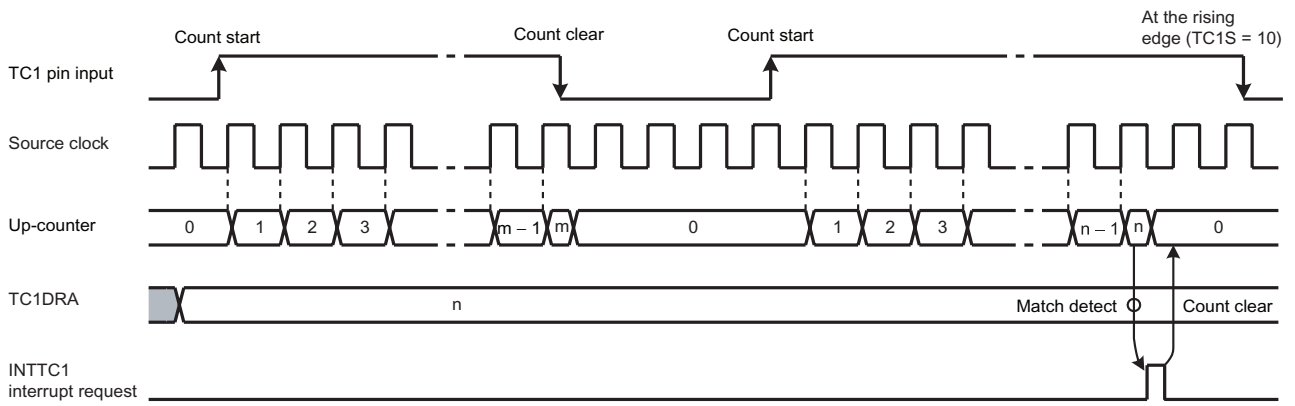
```
LDW      (TC1DRA), 007DH      : 1ms ÷ 27/fc = 7DH
DI       : IMF= "0"
SET      (EIRL). 7           : Enables INTTC1 interrupt
EI       : IMF= "1"
LD       (TC1CR), 00000100B   : Selects the source clock and mode
LD       (TC1CR), 00100100B   : Starts TC1 external trigger, METT1 = 0
```

Example 2 :Generating an interrupt when the low-level pulse with 4 ms or more width is input to the TC1 pin  
( $f_c = 16$  MHz)

```
LDW      (TC1DRA), 01F4H      : 4 ms ÷ 27/fc = 1F4H
DI       : IMF= "0"
SET      (EIRL). 7           : Enables INTTC1 interrupt
EI       : IMF= "1"
LD       (TC1CR), 00000100B   : Selects the source clock and mode
LD       (TC1CR), 01110100B   : Starts TC1 external trigger, METT1 = 0
```



(a) Trigger start (METT1 = 0)



(b) Trigger start and stop (METT1 = 1)

Note:  $m < n$

Figure 8-3 External Trigger Timer Mode Timing Chart

### 8.3.3 Event Counter Mode

In the event counter mode, the up-counter counts up at the edge of the input pulse to the TC1 pin. Either the rising or falling edge of the input pulse is selected as the count up edge in TC1CR<TC1S>.

When a match between the up-counter and the TC1DRA value is detected, an INTTC1 interrupt is generated and the up-counter is cleared. After being cleared, the up-counter restarts counting at each edge of the input pulse to the TC1 pin. Since a match between the up-counter and the value set to TC1DRA is detected at the edge opposite to the selected edge, an INTTC1 interrupt request is generated after a match of the value at the edge opposite to the selected edge.

Two or more machine cycles are required for the low-or high-level pulse input to the TC1 pin.

Setting TC1CR<ACAP1> to "1" captures the up-counter value into TC1DRB with the 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.

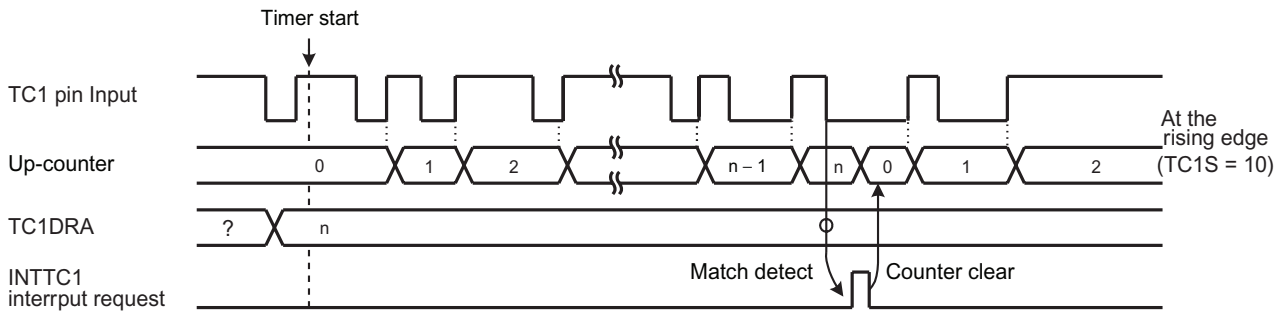


Figure 8-4 Event Counter Mode Timing Chart

Table 8-2 Input Pulse Width to TC1 Pin

|            | Minimum Pulse Width [s] |                        |
|------------|-------------------------|------------------------|
|            | NORMAL1/2, IDLE1/2 Mode | SLOW1/2, SLEEP1/2 Mode |
| High-going | $2^3/f_c$               | $2^3/f_s$              |
| Low-going  | $2^3/f_c$               | $2^3/f_s$              |

8.3.4 Window Mode

In the window mode, the up-counter counts up at the rising edge of the pulse that is logical ANDed product of the input pulse to the TC1 pin (window pulse) and the internal source clock. Either the positive logic (count up during high-going pulse) or negative logic (count up during low-going pulse) can be selected.

When a match between the up-counter and the TC1DRA value is detected, an INTTC1 interrupt is generated and the up-counter is cleared.

Define the window pulse to the frequency which is sufficiently lower than the internal source clock programmed with TC1CR<TC1CK>.

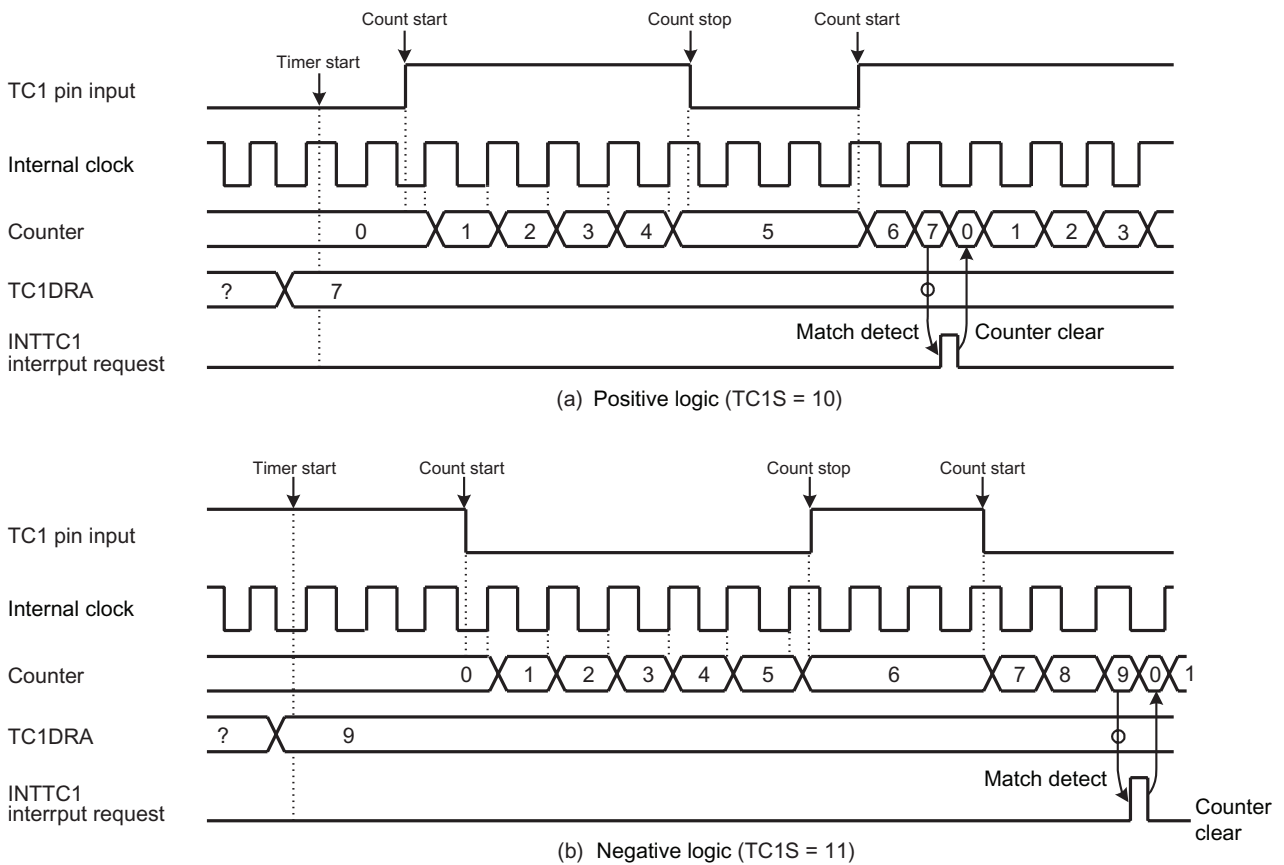


Figure 8-5 Window Mode Timing Chart

### 8.3.5 Pulse Width Measurement Mode

In the pulse width measurement mode, the up-counter starts counting by the input pulse triggering of the TC1 pin, and counts up at the edge of the internal clock. Either the rising or falling edge of the internal clock is selected as the trigger edge in TC1CR<TC1S>. Either the single- or double-edge capture is selected as the trigger edge in TC1CR<MCAP1>.

- When TC1CR<MCAP1> is set to “1” (single-edge capture)

Either high- or low-level input pulse width can be measured. To measure the high-level input pulse width, set the rising edge to TC1CR<TC1S>. To measure the low-level input pulse width, set the falling edge to TC1CR<TC1S>.

When detecting the edge opposite to the trigger edge used to start counting after the timer starts, the up-counter captures the up-counter value into TC1DRB and generates an INTTC1 interrupt request. The up-counter is cleared at this time, and then restarts counting when detecting the trigger edge used to start counting.

- When TC1CR<MCAP1> is set to “0” (double-edge capture)

The cycle starting with either the high- or low-going input pulse can be measured. To measure the cycle starting with the high-going pulse, set the rising edge to TC1CR<TC1S>. To measure the cycle starting with the low-going pulse, set the falling edge to TC1CR<TC1S>.

When detecting the edge opposite to the trigger edge used to start counting after the timer starts, the up-counter captures the up-counter value into TC1DRB and generates an INTTC1 interrupt request. The up-counter continues counting up, and captures the up-counter value into TC1DRB and generates an INTTC1 interrupt request when detecting the trigger edge used to start counting. The up-counter is cleared at this time, and then continues counting.

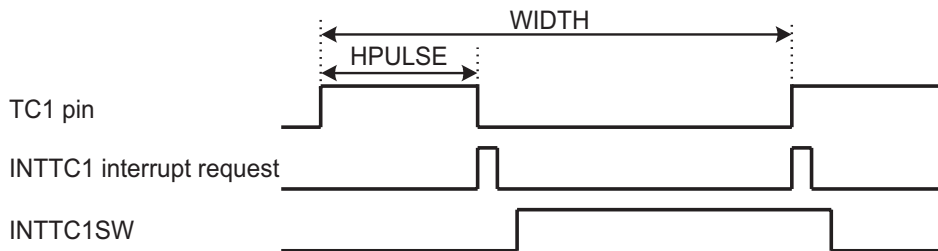
Note 1: The captured value must be read from TC1DRB until the next trigger edge is detected. If not read, the captured value becomes a don't care. It is recommended to use a 16-bit access instruction to read the captured value from TC1DRB.

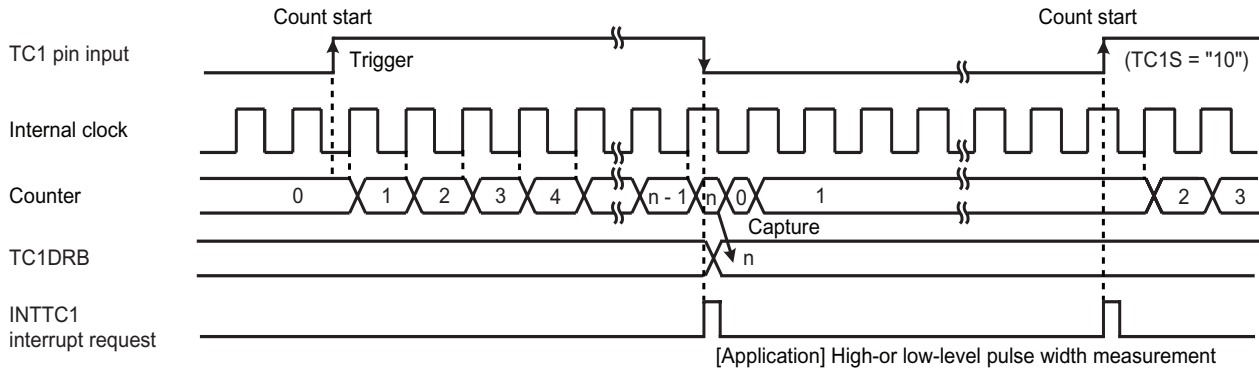
Note 2: For the single-edge capture, the counter after capturing the value stops at “1” until detecting the next edge. Therefore, the second captured value is “1” larger than the captured value immediately after counting starts.

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

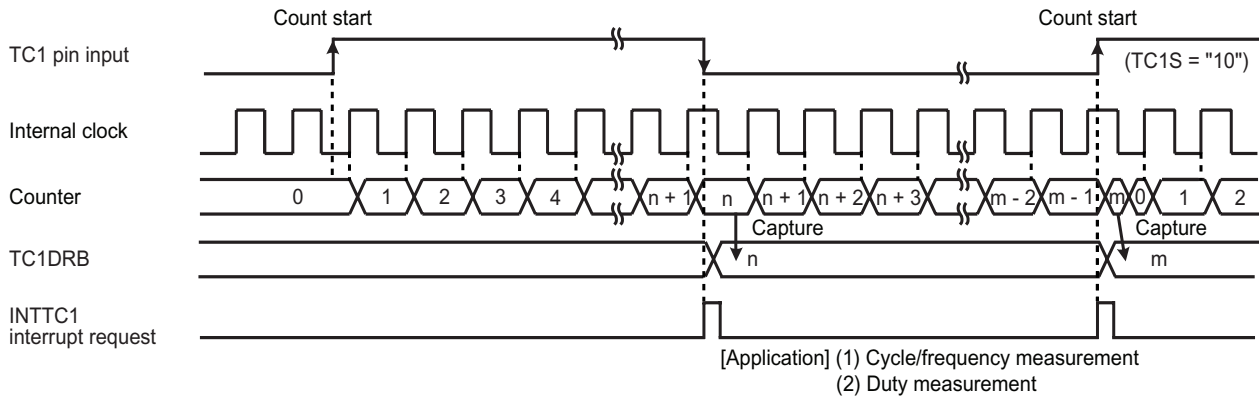
Example :Duty measurement (resolution  $fc/2^7$  [Hz])

|          |      |                    |   |
|----------|------|--------------------|---|
|          | CLR  | (INTTC1SW). 0      | : INTTC1 service switch initial setting<br>Address set to convert INTTC1SW at each INTTC1 |
|          | LD   | (TC1CR), 00000110B | : Sets the TC1 mode and source clock  |
|          | DI   |                    | : IMF= "0"  |
|          | SET  | (EIRL). 7          | : Enables INTTC1  |
|          | EI   |                    | : IMF= "1"  |
|          | LD   | (TC1CR), 00100110B | : Starts TC1 with an external trigger at MCAP1 = 0  |
|          | :    |                    |   |
| PINTTC1: | CPL  | (INTTC1SW). 0      | : INTTC1 interrupt, inverts and tests INTTC1 service switch                               |
|          | JRS  | F, SINTTC1         |   |
|          | LD   | A, (TC1DRBL)       | : Reads TC1DRB (High-level pulse width)   |
|          | LD   | W,(TC1DRBH)        |   |
|          | LD   | (HPULSE), WA       | : Stores high-level pulse width in RAM  |
|          | RETI |                    |   |
| SINTTC1: | LD   | A, (TC1DRBL)       | : Reads TC1DRB (Cycle)  |
|          | LD   | W,(TC1DRBH)        |   |
|          | LD   | (WIDTH), WA        | : Stores cycle in RAM   |
|          | :    |                    |   |
|          | RETI |                    | : Duty calculation  |
|          | :    |                    |   |
| VINTTC1: | DW   | PINTTC1            | : INTTC1 Interrupt vector   |





(a) Single-edge capture (MCAP1 = "1")



(b) Double-edge capture (MCAP1 = "0")

Figure 8-6 Pulse Width Measurement Mode

### 8.3.6 Programmable Pulse Generate (PPG) Output Mode

In the programmable pulse generation (PPG) mode, an arbitrary duty pulse is generated by counting performed in the internal clock. To start the timer, TC1CR<TC1S> specifies either the edge of the input pulse to the TC1 pin or the command start. TC1CR<MPPG1> specifies whether a duty pulse is produced continuously or not (one-shot pulse).

- When TC1CR<MPPG1> is set to “0” (Continuous pulse generation)

When a match between the up-counter and the TC1DRB value is detected after the timer starts, the level of the  $\overline{\text{PPG}}$  pin is inverted and an INTTC1 interrupt request is generated. The up-counter continues counting. When a match between the up-counter and the TC1DRA value is detected, the level of the  $\overline{\text{PPG}}$  pin is inverted and an INTTC1 interrupt request is generated. The up-counter is cleared at this time, and then continues counting and pulse generation.

When TC1S is cleared to “00” during PPG output, the  $\overline{\text{PPG}}$  pin retains the level immediately before the counter stops.

- When TC1CR<MPPG1> is set to “1” (One-shot pulse generation)

When a match between the up-counter and the TC1DRB value is detected after the timer starts, the level of the  $\overline{\text{PPG}}$  pin is inverted and an INTTC1 interrupt request is generated. The up-counter continues counting. When a match between the up-counter and the TC1DRA value is detected, the level of the  $\overline{\text{PPG}}$  pin is inverted and an INTTC1 interrupt request is generated. TC1CR<TC1S> is cleared to “00” automatically at this time, and the timer stops. The pulse generated by PPG retains the same level as that when the timer stops.

Since the output level of the  $\overline{\text{PPG}}$  pin can be set with TC1CR<TFF1> when the timer starts, a positive or negative pulse can be generated. Since the inverted level of the timer F/F1 output level is output to the  $\overline{\text{PPG}}$  pin, specify TC1CR<TFF1> to “0” to set the high level to the  $\overline{\text{PPG}}$  pin, and “1” to set the low level to the  $\overline{\text{PPG}}$  pin. Upon reset, the timer F/F1 is initialized to “0”.

Note 1: To change TC1DRA or TC1DRB during a run of the timer, set a value sufficiently larger than the count value of the counter. Setting a value smaller than the count value of the counter during a run of the timer may generate a pulse different from that specified.

Note 2: Do not change TC1CR<TFF1> during a run of the timer. TC1CR<TFF1> can be set correctly only at initialization (after reset). When the timer stops during PPG, TC1CR<TFF1> can not be set correctly from this point onward if the PPG output has the level which is inverted of the level when the timer starts. (Setting TC1CR<TFF1> specifies the timer F/F1 to the level inverted of the programmed value.) Therefore, the timer F/F1 needs to be initialized to ensure an arbitrary level of the PPG output. To initialize the timer F/F1, change TC1CR<TC1M> to the timer mode (it is not required to start the timer mode), and then set the PPG mode. Set TC1CR<TFF1> at this time.

Note 3: In the PPG mode, the following relationship must be satisfied.  
TC1DRA > TC1DRB

Note 4: Set TC1DRB after changing the mode of TC1M to the PPG mode.

Example :Generating a pulse which is high-going for 800  $\mu$ s and low-going for 200  $\mu$ s  
(fc = 16 MHz)

| Setting port |                    |  |
|--------------|--------------------|--|
| LD           | (TC1CR), 10000111B | : Sets the PPG mode, selects the source clock  |
| LDW          | (TC1DRA), 007DH    | : Sets the cycle ( $1 \text{ ms} \div 2^7 / \text{fc} \text{ ms} = 007\text{DH}$ )         |
| LDW          | (TC1DRB), 0019H    | : Sets the low-level pulse width ( $200 \mu\text{s} \div 2^7 / \text{fc} = 0019\text{H}$ ) |
| LD           | (TC1CR), 10010111B | : Starts the timer   |

Example :After stopping PPG, setting the PPG pin to a high-level to restart PPG  
(fc = 16 MHz)

| Setting port |                    |  |
|--------------|--------------------|--|
| LD           | (TC1CR), 10000111B | : Sets the PPG mode, selects the source clock  |
| LDW          | (TC1DRA), 007DH    | : Sets the cycle ( $1 \text{ ms} \div 2^7 / \text{fc} \mu\text{s} = 007\text{DH}$ )        |
| LDW          | (TC1DRB), 0019H    | : Sets the low-level pulse width ( $200 \mu\text{s} \div 2^7 / \text{fc} = 0019\text{H}$ ) |
| LD           | (TC1CR), 10010111B | : Starts the timer   |
| :            | :                  |  |
| LD           | (TC1CR), 10000111B | : Stops the timer  |
| LD           | (TC1CR), 10000100B | : Sets the timer mode  |
| LD           | (TC1CR), 00000111B | : Sets the PPG mode, TFF1 = 0  |
| LD           | (TC1CR), 00010111B | : Starts the timer   |

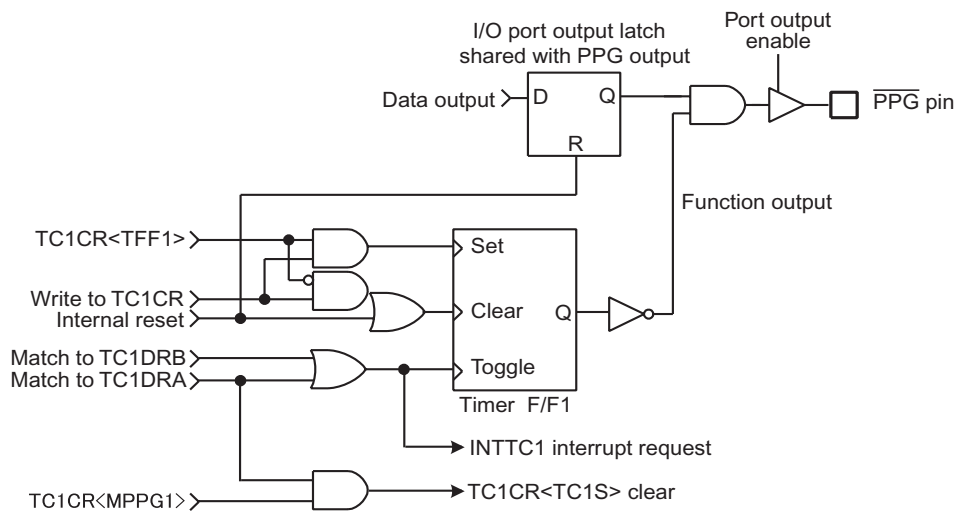
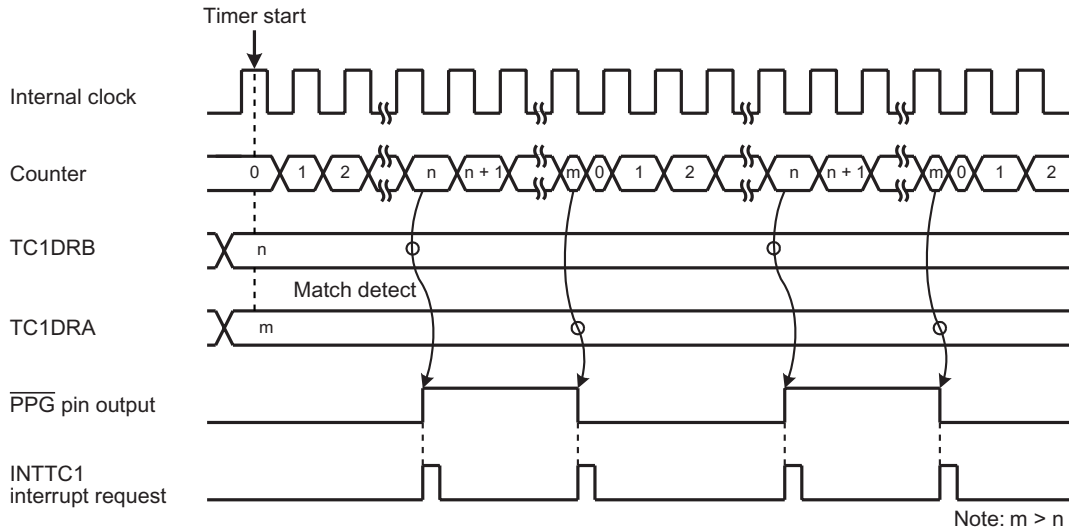
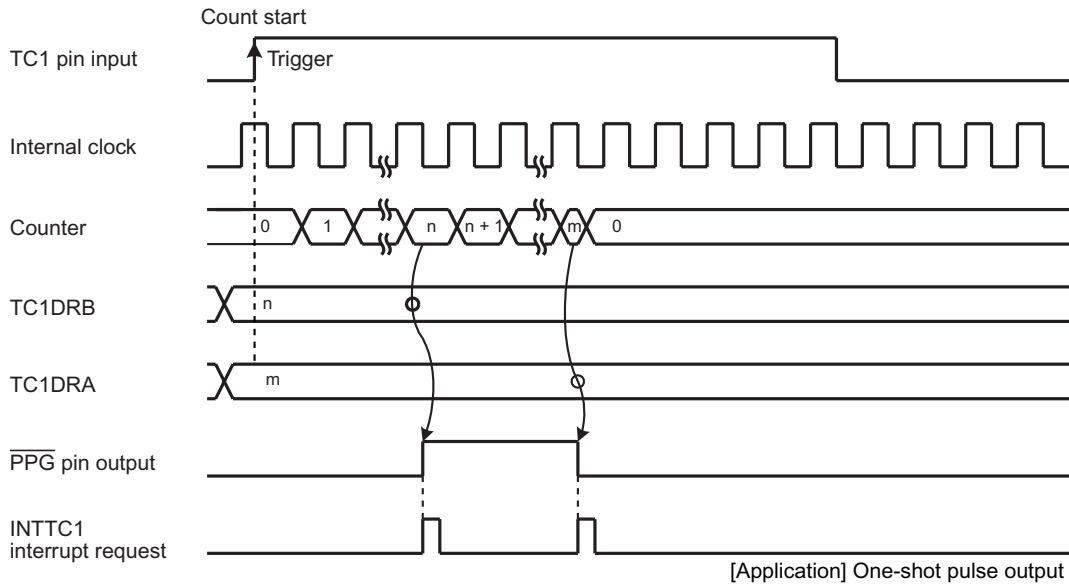


Figure 8-7  $\overline{\text{PPG}}$  Output



(a) Continuous pulse generation (TC1S = 01)



(b) One-shot pulse generation (TC1S = 10)

Figure 8-8 PPG Mode Timing Chart



## 9. 8-Bit TimerCounter (TC3, TC4)

### 9.1 Configuration

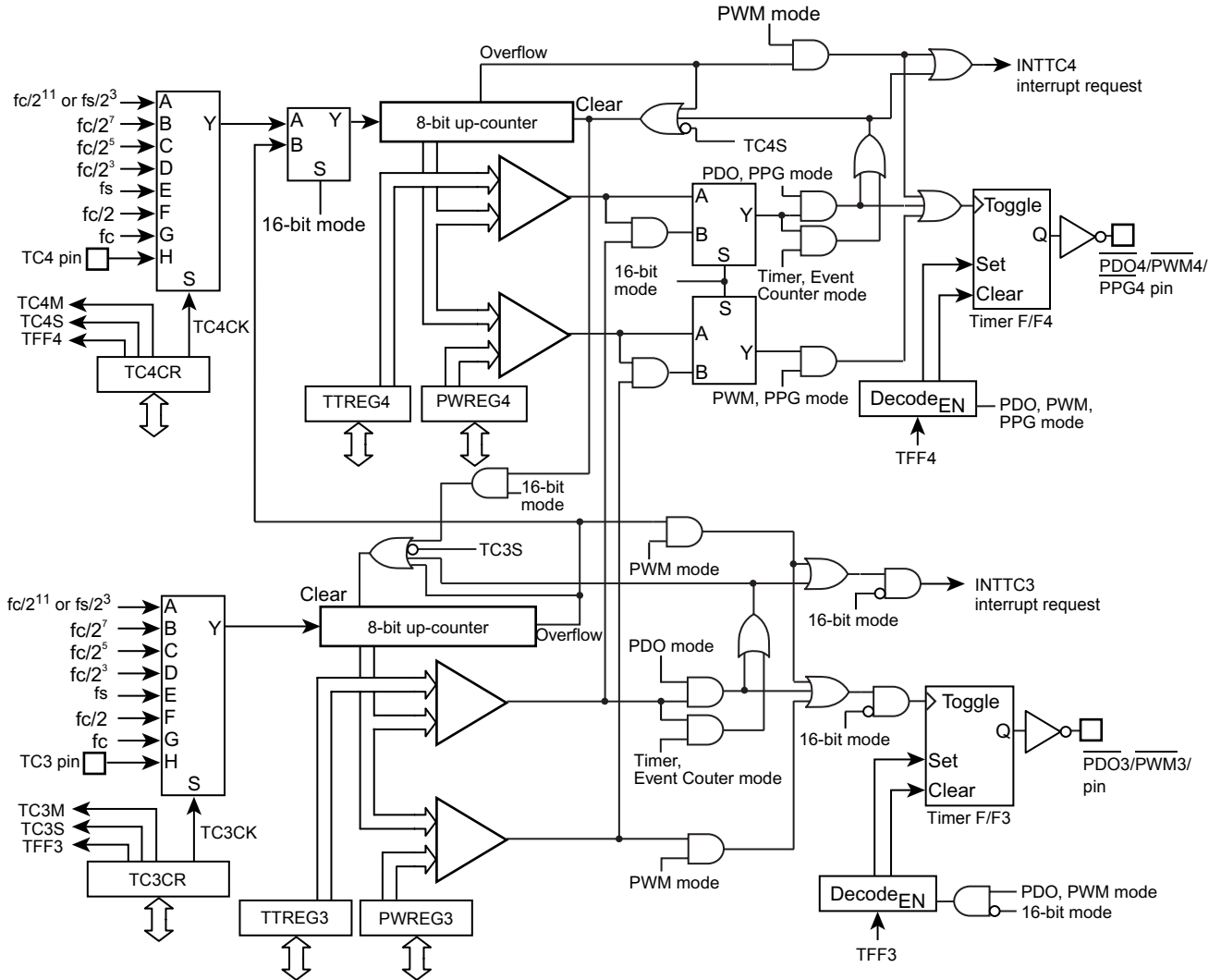
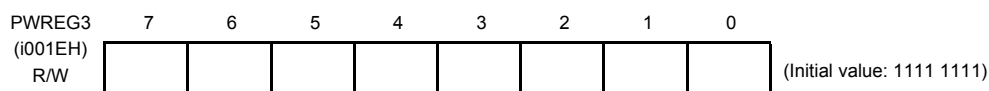
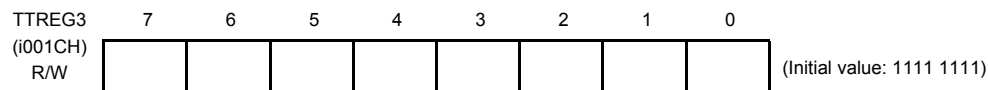


Figure 9-1 8-Bit TimerCounter 3, 4

## 9.2 TimerCounter Control

The TimerCounter 3 is controlled by the TimerCounter 3 control register (TC3CR) and two 8-bit timer registers (TTREG3, PWREG3).

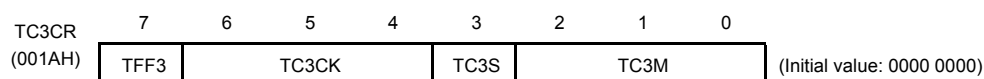
### TimerCounter 3 Timer Register



Note 1: Do not change the timer register (TTREG3) setting while the timer is running.

Note 2: Do not change the timer register (PWREG3) setting in the operating mode except the 8-bit and 16-bit PWM modes while the timer is running.

### TimerCounter 3 Control Register



| TFF3  | Time F/F3 control              | 0: Clear<br>1: Set   |             |                             | R/W |          |
|-------|--------------------------------|--|-------------|-----------------------------|-----|----------|
| TC3CK | Operating clock selection [Hz] | NORMAL1/2, IDLE1/2 mode  |             | SLOW1/2<br>SLEEP1/2<br>mode | R/W |          |
|       |                                | DV7CK = 0  | DV7CK = 1   |                             |     |          |
|       |                                | 000  | $fc/2^{11}$ | $fs/2^3$                    |     | $fs/2^3$ |
|       |                                | 001  | $fc/2^7$    | $fc/2^7$                    |     | A\       |
|       |                                | 010  | $fc/2^5$    | $fc/2^5$                    |     | A\       |
|       |                                | 011  | $fc/2^3$    | $fc/2^3$                    |     | A\       |
|       |                                | 100  | fs          | fs                          |     | fs       |
|       |                                | 101  | $fc/2$      | $fc/2$                      |     | A\       |
| 110   | fc                             | fc   | fc (Note 8) |                             |     |          |
| 111   | TC3 pin input                  |  |             |                             |     |          |
| TC3S  | TC3 start control              | 0: Operation stop and counter clear<br>1: Operation start  |             |                             | R/W |          |
| TC3M  | TC3M operating mode select     | 000: 8-bit timer/event counter mode<br>001: 8-bit programmable divider output (PDO) mode<br>010: 8-bit pulse width modulation (PWM) output mode<br>011: 16-bit mode<br>(Each mode is selectable with TC4M.)<br>1**: Reserved |             |                             | R/W |          |

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

Note 2: Do not change the TC3M, TC3CK and TFF3 settings while the timer is running.

Note 3: To stop the timer operation (TC3S= 1 → 0), do not change the TC3M, TC3CK and TFF3 settings. To start the timer operation (TC3S= 0 → 1), TC3M, TC3CK and TFF3 can be programmed.

Note 4: To use the TimerCounter in the 16-bit mode, set the operating mode by programming TC4CR<TC4M>, where TC3M must be fixed to 011.

Note 5: To use the TimerCounter in the 16-bit mode, select the source clock by programming TC3CK. Set the timer start control and timer F/F control by programming TC4CR<TC4S> and TC4CR<TFF4>, respectively.

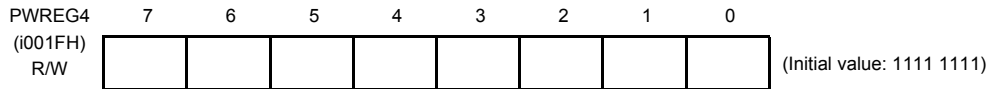
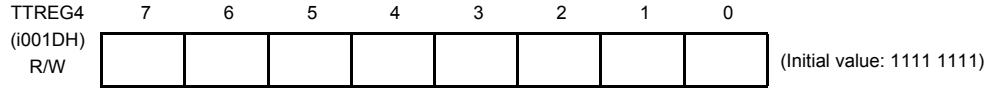
Note 6: The operating clock settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 9-1 and Table 9-2.

Note 7: The timer register settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 9-3.

Note 8: The operating clock  $f_c$  in the SLOW or SLEEP mode can be used only as the high-frequency warm-up mode.

The TimerCounter 4 is controlled by the TimerCounter 4 control register (TC4CR) and two 8-bit timer registers (TTREG4 and PWREG4).

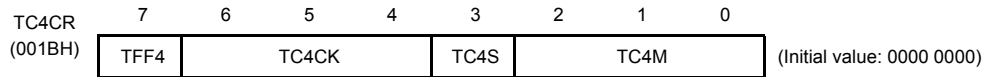
TimerCounter 4 Timer Register



Note 1: Do not change the timer register (TTREG4) setting while the timer is running.

Note 2: Do not change the timer register (PWREG4) setting in the operating mode except the 8-bit and 16-bit PWM modes while the timer is running.

TimerCounter 4 Control Register



| TFF4  | Timer F/F4 control             | 0: Clear<br>1: Set  |             |                             | R/W |          |
|-------|--------------------------------|---|-------------|-----------------------------|-----|----------|
| TC4CK | Operating clock selection [Hz] | NORMAL1/2, IDLE1/2 mode   |             | SLOW1/2<br>SLEEP1/2<br>mode | R/W |          |
|       |                                | 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   | fs          | fs                          |     | fs       |
|       |                                | 101   | $fc/2$      | $fc/2$                      |     | –        |
| 110   | fc                             | fc  | –           |                             |     |          |
| 111   | TC4 pin input                  |   |             |                             |     |          |
| TC4S  | TC4 start control              | 0: Operation stop and counter clear<br>1: Operation start   |             |                             | R/W |          |
| TC4M  | TC4M operating mode select     | 000: 8-bit timer/event counter mode<br>001: 8-bit programmable divider output (PDO) mode<br>010: 8-bit pulse width modulation (PWM) output mode<br>011: Reserved<br>100: 16-bit timer/event counter mode<br>101: Warm-up counter mode<br>110: 16-bit pulse width modulation (PWM) output mode<br>111: 16-bit PPG mode |             |                             | R/W |          |

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

Note 2: Do not change the TC4M, TC4CK and TFF4 settings while the timer is running.

Note 3: To stop the timer operation (TC4S= 1 → 0), do not change the TC4M, TC4CK and TFF4 settings.  
To start the timer operation (TC4S= 0 → 1), TC4M, TC4CK and TFF4 can be programmed.

Note 4: When TC4M= 1\*\* (upper byte in the 16-bit mode), the source clock becomes the TC4 overflow signal regardless of the TC3CK setting.

Note 5: To use the TimerCounter in the 16-bit mode, select the operating mode by programming TC4M, where TC3CR<TC3 M> must be set to 011.

Note 6: To the TimerCounter in the 16-bit mode, select the source clock by programming TC3CR<TC3CK>. Set the timer start control and timer F/F control by programming TC4S and TFF4, respectively.

Note 7: The operating clock settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 9-1 and Table 9-2.

Note 8: The timer register settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 9-3.

Table 9-1 Operating Mode and Selectable Source Clock (NORMAL1/2 and IDLE1/2 Modes)

| Operating mode       | $fc/2^{11}$<br>or<br>$fs/2^3$ | $fc/2^7$ | $fc/2^5$ | $fc/2^3$ | fs | fc/2 | fc | TC3<br>pin input | TC4<br>pin input |
|----------------------|-------------------------------|----------|----------|----------|----|------|----|------------------|------------------|
| 8-bit timer          | 0                             | 0        | 0        | 0        | –  | –    | –  | –                | –                |
| 8-bit event counter  | –                             | –        | –        | –        | –  | –    | –  | 0                | 0                |
| 8-bit PDO            | 0                             | 0        | 0        | 0        | –  | –    | –  | –                | –                |
| 8-bit PWM            | 0                             | 0        | 0        | 0        | 0  | 0    | 0  | –                | –                |
| 16-bit timer         | 0                             | 0        | 0        | 0        | –  | –    | –  | –                | –                |
| 16-bit event counter | –                             | –        | –        | –        | –  | –    | –  | 0                | –                |
| Warm-up counter      | –                             | –        | –        | –        | 0  | –    | –  | –                | –                |
| 16-bit PWM           | 0                             | 0        | 0        | 0        | 0  | 0    | 0  | 0                | –                |
| 16-bit PPG           | 0                             | 0        | 0        | 0        | –  | –    | –  | 0                | –                |

Note 1: For 16-bit operations (16-bit timer/event counter, warm-up counter, 16-bit PWM and 16-bit PPG), set its source clock on lower bit (TC3CK).

Note 2: 0 : Available source clock

Table 9-2 Operating Mode and Selectable Source Clock (SLOW1/2 and SLEEP1/2 Modes)

| Operating mode       | $fc/2^{11}$<br>or<br>$fs/2^3$ | $fc/2^7$ | $fc/2^5$ | $fc/2^3$ | fs | fc/2 | fc | TC3<br>pin input | TC4<br>pin input |
|----------------------|-------------------------------|----------|----------|----------|----|------|----|------------------|------------------|
| 8-bit timer          | 0                             | –        | –        | –        | –  | –    | –  | –                | –                |
| 8-bit event counter  | –                             | –        | –        | –        | –  | –    | –  | 0                | 0                |
| 8-bit PDO            | 0                             | –        | –        | –        | –  | –    | –  | –                | –                |
| 8-bit PWM            | 0                             | –        | –        | –        | 0  | –    | –  | –                | –                |
| 16-bit timer         | 0                             | –        | –        | –        | –  | –    | –  | –                | –                |
| 16-bit event counter | –                             | –        | –        | –        | –  | –    | –  | 0                | –                |
| Warm-up counter      | –                             | –        | –        | –        | –  | –    | 0  | –                | –                |
| 16-bit PWM           | 0                             | –        | –        | –        | 0  | –    | –  | 0                | –                |
| 16-bit PPG           | 0                             | –        | –        | –        | –  | –    | –  | 0                | –                |

Note1: For 16-bit operations (16-bit timer/event counter, warm-up counter, 16-bit PWM and 16-bit PPG), set its source clock on lower bit (TC3CK).

Note2: 0 : Available source clock

Table 9-3 Constraints on Register Values Being Compared

| Operating mode             | Register Value  |
|----------------------------|---|
| 8-bit timer/event counter  | $1 \leq (\text{TTREGn}) \leq 255$   |
| 8-bit PDO                  | $1 \leq (\text{TTREGn}) \leq 255$   |
| 8-bit PWM                  | $2 \leq (\text{PWREGn}) \leq 254$   |
| 16-bit timer/event counter | $1 \leq (\text{TTREG4, 3}) \leq 65535$  |
| Warm-up counter            | $256 \leq (\text{TTREG4, 3}) \leq 65535$  |
| 16-bit PWM                 | $2 \leq (\text{PWREG4, 3}) \leq 65534$  |
| 16-bit PPG                 | $1 \leq (\text{PWREG4, 3}) < (\text{TTREG4, 3}) \leq 65535$<br>and<br>$(\text{PWREG4, 3}) + 1 < (\text{TTREG4, 3})$ |

Note: n = 3 to 4

## 9.3 Function

The TimerCounter 3 and 4 have the 8-bit timer, 8-bit event counter, 8-bit programmable divider output (PDO), 8-bit pulse width modulation (PWM) output modes. The TimerCounter 3 and 4 (TC3, 4) are cascadable to form a 16-bit timer. The 16-bit timer has the operating modes such as the 16-bit timer, 16-bit event counter, warm-up counter, 16-bit pulse width modulation (PWM) output and 16-bit programmable pulse generation (PPG) modes.

### 9.3.1 8-Bit Timer Mode (TC3 and 4)

In the timer mode, the up-counter counts up using the internal clock. When a match between the up-counter and the timer register  $j$  (TTREG $j$ ) value is detected, an INTCC $j$  interrupt is generated and the up-counter is cleared. After being cleared, the up-counter restarts counting.

Note 1: In the timer mode, fix TCjCR<TFFj> to 0. If not fixed, the  $\overline{PDOj}$ ,  $\overline{PWMj}$  and  $\overline{PPGj}$  pins may output pulses.

Note 2: In the timer mode, do not change the TTREG $j$  setting while the timer is running. Since TTREG $j$  is not in the shift register configuration in the timer mode, the new value programmed in TTREG $j$  is in effect immediately after the programming. Therefore, if TTREG $i$  is changed while the timer is running, an expected operation may not be obtained.

Note 3:  $j = 3, 4$

Table 9-4 Source Clock for TimerCounter 3, 4 (Internal Clock)

| Source Clock            |                | SLOW1/2,<br>SLEEP1/2<br>mode | Resolution             |                            | Repeated Cycle         |                            |
|-------------------------|----------------|------------------------------|------------------------|----------------------------|------------------------|----------------------------|
| NORMAL1/2, IDLE1/2 mode |                |                              | $f_c = 16 \text{ MHz}$ | $f_s = 32.768 \text{ kHz}$ | $f_c = 16 \text{ MHz}$ | $f_s = 32.768 \text{ kHz}$ |
| DV7CK = 0               | DV7CK = 1      |                              |                        |                            |                        |                            |
| $f_c/2^{11}$ [Hz]       | $f_s/2^3$ [Hz] | $f_s/2^3$ [Hz]               | 128 $\mu\text{s}$      | 244.14 $\mu\text{s}$       | 32.6 ms                | 62.3 ms                    |
| $f_c/2^7$               | $f_c/2^7$      | –                            | 8 $\mu\text{s}$        | –                          | 2.0 ms                 | –                          |
| $f_c/2^5$               | $f_c/2^5$      | –                            | 2 $\mu\text{s}$        | –                          | 510 $\mu\text{s}$      | –                          |
| $f_c/2^3$               | $f_c/2^3$      | –                            | 500 ns                 | –                          | 127.5 $\mu\text{s}$    | –                          |

Example :Setting the timer mode with source clock  $f_c/2^7$  Hz and generating an interrupt 80  $\mu\text{s}$  later (TimerCounter4,  $f_c = 16.0 \text{ MHz}$ )

```
LD      (TTREG4), 0AH      : Sets the timer register ( $80 \mu\text{s} \div 2^7 / f_c = 0AH$ ).
DI
SET     (EIRH). 3         : Enables INTTC4 interrupt.
EI
LD      (TC4CR), 00010000B : Sets the operating clock to  $f_c/2^7$ , and 8-bit timer mode.
LD      (TC4CR), 00011000B : Starts TC4.
```

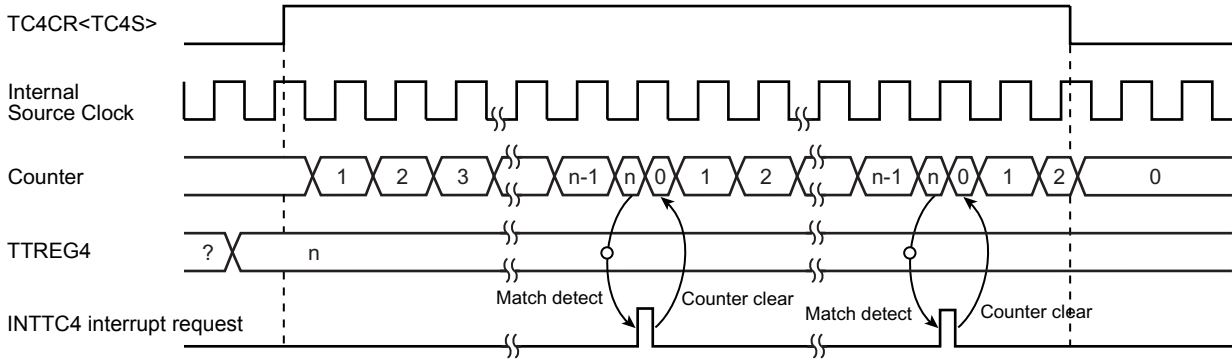


Figure 9-2 8-Bit Timer Mode Timing Chart (TC4)

### 9.3.2 8-Bit Event Counter Mode (TC3, 4)

In the 8-bit event counter mode, the up-counter counts up at the falling edge of the input pulse to the TCj pin. When a match between the up-counter and the TTREGj value is detected, an INTCCj interrupt is generated and the up-counter is cleared. After being cleared, the up-counter restarts counting at the falling edge of the input pulse to the TCj pin. Two machine cycles are required for the low- or high-level pulse input to the TCj pin. Therefore, a maximum frequency to be supplied is  $f_c/2^4$  Hz in the NORMAL1/2 or IDLE1/2 mode, and  $f_s/2^4$  Hz in the SLOW1/2 or SLEEP1/2 mode.

Note 1: In the event counter mode, fix TCjCR<TFFj> to 0. If not fixed, the  $\overline{PDOj}$ ,  $\overline{PWMj}$  and  $\overline{PPGj}$  pins may output pulses.

Note 2: In the event counter mode, do not change the TTREGj setting while the timer is running. Since TTREGj is not in the shift register configuration in the event counter mode, the new value programmed in TTREGj is in effect immediately after the programming. Therefore, if TTREGj is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 3, 4

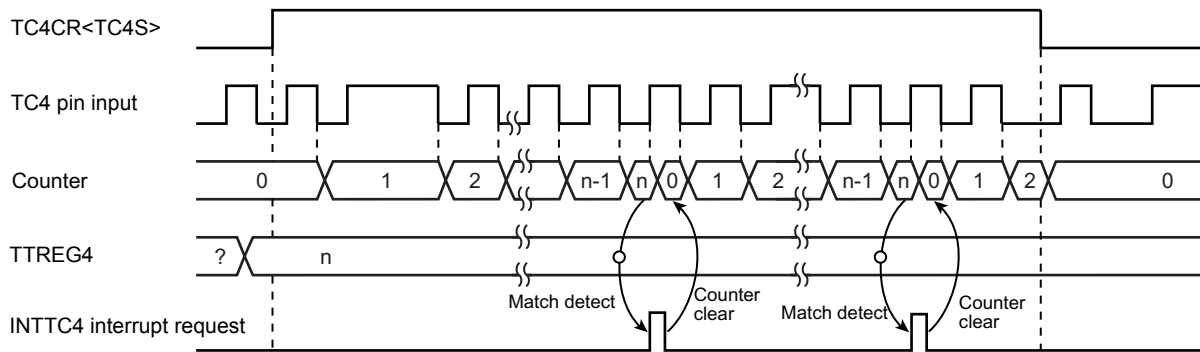


Figure 9-3 8-Bit Event Counter Mode Timing Chart (TC4)

### 9.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC3, 4)

This mode is used to generate a pulse with a 50% duty cycle from the  $\overline{PDOj}$  pin.

In the PDO mode, the up-counter counts up using the internal clock. When a match between the up-counter and the TTREGj value is detected, the logic level output from the  $\overline{PDOj}$  pin is switched to the opposite state and the up-counter is cleared. The INTTCj interrupt request is generated at the time. The logic state opposite to the timer F/Fj logic level is output from the  $\overline{PDOj}$  pin. An arbitrary value can be set to the timer F/Fj by TCjCR<TFFj>. Upon reset, the timer F/Fj value is initialized to 0.

To use the programmable divider output, set the output latch of the I/O port to 1.

Example :Generating 1024 Hz pulse using TC4 ( $f_c = 16.0$  MHz)

| Setting port |                    |   |
|--------------|--------------------|---|
| LD           | (TTREG4), 3DH      | : $1/1024 \div 2^7 / f_c \div 2 = 3DH$                        |
| LD           | (TC4CR), 00010001B | : Sets the operating clock to $f_c/2^7$ , and 8-bit PDO mode. |
| LD           | (TC4CR), 00011001B | : Starts TC4.   |

Note 1: In the programmable divider output mode, do not change the TTREGj setting while the timer is running. Since TTREGj is not in the shift register configuration in the programmable divider output mode, the new value programmed in TTREGj is in effect immediately after programming. Therefore, if TTREGi is changed while the timer is running, an expected operation may not be obtained.

Note 2: When the timer is stopped during PDO output, the  $\overline{PDOj}$  pin holds the output status when the timer is stopped. To change the output status, program TCjCR<TFFj> after the timer is stopped. Do not change the TCjCR<TFFj> setting upon stopping of the timer.

Example: Fixing the  $\overline{PDOj}$  pin to the high level when the TimerCounter is stopped

CLR (TCjCR).3: Stops the timer.

CLR (TCjCR).7: Sets the PDOj pin to the high level.

Note 3: j = 3, 4



### 9.3.4 8-Bit Pulse Width Modulation (PWM) Output Mode (TC3, 4)

This mode is used to generate a pulse-width modulated (PWM) signals with up to 8 bits of resolution. The up-counter counts up using the internal clock.

When a match between the up-counter and the PWREGj value is detected, the logic level output from the timer F/Fj is switched to the opposite state. The counter continues counting. The logic level output from the timer F/Fj is switched to the opposite state again by the up-counter overflow, and the counter is cleared. The INTTCj interrupt request is generated at this time.

Since the initial value can be set to the timer F/Fj by TCjCR<TFFj>, positive and negative pulses can be generated. Upon reset, the timer F/Fj is cleared to 0.

(The logic level output from the  $\overline{\text{PWMj}}$  pin is the opposite to the timer F/Fj logic level.)

Since PWREGj in the PWM mode is serially connected to the shift register, the value set to PWREGj can be changed while the timer is running. The value set to PWREGj during a run of the timer is shifted by the INTTCj interrupt request and loaded into PWREGj. While the timer is stopped, the value is shifted immediately after the programming of PWREGj. If executing the read instruction to PWREGj during PWM output, the value in the shift register is read, but not the value set in PWREGj. Therefore, after writing to PWREGj, the reading data of PWREGj is previous value until INTTCj is generated.

For the pin used for PWM output, the output latch of the I/O port must be set to 1.

Note 1: In the PWM mode, program the timer register PWREGj immediately after the INTTCj interrupt request is generated (normally in the INTTCj interrupt service routine.) If the programming of PWREGj and the interrupt request occur at the same time, an unstable value is shifted, that may result in generation of the pulse different from the programmed value until the next INTTCj interrupt request is generated.

Note 2: When the timer is stopped during PWM output, the  $\overline{\text{PWMj}}$  pin holds the output status when the timer is stopped. To change the output status, program TCjCR<TFFj> after the timer is stopped. Do not change the TCjCR<TFFj> upon stopping of the timer.

Example: Fixing the  $\overline{\text{PWMj}}$  pin to the high level when the TimerCounter is stopped

CLR (TCjCR).3: Stops the timer.

CLR (TCjCR).7: Sets the  $\overline{\text{PWMj}}$  pin to the high level.

Note 3: To enter the STOP mode during PWM output, stop the timer and then enter the STOP mode. If the STOP mode is entered without stopping the timer when fc, fc/2 or fs is selected as the source clock, a pulse is output from the  $\overline{\text{PWMj}}$  pin during the warm-up period time after exiting the STOP mode.

Note 4: j = 3, 4

Table 9-5 PWM Output Mode

| Source Clock            |                        | SLOW1/2,<br>SLEEP1/2<br>mode | Resolution  |                 | Repeated Cycle |                 |
|-------------------------|------------------------|------------------------------|-------------|-----------------|----------------|-----------------|
| NORMAL1/2, IDLE1/2 mode | DV7CK = 1              |                              | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz    | fs = 32.768 kHz |
| DV7CK = 0               | DV7CK = 1              | fs/2 <sup>3</sup> [Hz]       |             |                 |                |                 |
| fc/2 <sup>11</sup> [Hz] | fs/2 <sup>3</sup> [Hz] | fs/2 <sup>3</sup> [Hz]       | 128 μs      | 244.14 μs       | 32.8 ms        | 62.5 ms         |
| fc/2 <sup>7</sup>       | fc/2 <sup>7</sup>      | –                            | 8 μs        | –               | 2.05 ms        | –               |
| fc/2 <sup>5</sup>       | fc/2 <sup>5</sup>      | –                            | 2 μs        | –               | 512 μs         | –               |
| fc/2 <sup>3</sup>       | fc/2 <sup>3</sup>      | –                            | 500 ns      | –               | 128 μs         | –               |
| fs                      | fs                     | fs                           | 30.5 μs     | 30.5 μs         | 7.81 ms        | 7.81 ms         |
| fc/2                    | fc/2                   | –                            | 125 ns      | –               | 32 μs          | –               |
| fc                      | fc                     | –                            | 62.5 ns     | –               | 16 μs          | –               |



### 9.3.5 16-Bit Timer Mode (TC3 and 4)

In the timer mode, the up-counter counts up using the internal clock. The TimerCounter 3 and 4 are cascadable to form a 16-bit timer.

When a match between the up-counter and the timer register (TTREG3, TTREG4) value is detected after the timer is started by setting TC4CR<TC4S> to 1, an INTTC4 interrupt is generated and the up-counter is cleared. After being cleared, the up-counter continues counting. Program the upper byte and lower byte in this order in the timer register. (Programming only the upper or lower byte should not be attempted.)

Note 1: In the timer mode, fix TCjCR<TFFj> to 0. If not fixed, the  $\overline{PDOj}$ ,  $\overline{PWMj}$ , and  $\overline{PPGj}$  pins may output a pulse.

Note 2: In the timer mode, do not change the TTREGj setting while the timer is running. Since TTREGj is not in the shift register configuration in the timer mode, the new value programmed in TTREGj is in effect immediately after programming of TTREGj. Therefore, if TTREGj is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 3, 4

Table 9-6 Source Clock for 16-Bit Timer Mode

| Source Clock            |                   | SLOW1/2,<br>SLEEP1/2<br>mode | Resolution |             | Repeated Cycle  |             |
|-------------------------|-------------------|------------------------------|------------|-------------|-----------------|-------------|
| NORMAL1/2, IDLE1/2 mode | DV7CK = 0         |                              | DV7CK = 1  | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz |
| fc/2 <sup>11</sup>      | fs/2 <sup>3</sup> | fs/2 <sup>3</sup>            | 128 μs     | 244.14 μs   | 8.39 s          | 16 s        |
| fc/2 <sup>7</sup>       | fc/2 <sup>7</sup> | –                            | 8 μs       | –           | 524.3 ms        | –           |
| fc/2 <sup>5</sup>       | fc/2 <sup>5</sup> | –                            | 2 μs       | –           | 131.1 ms        | –           |
| fc/2 <sup>3</sup>       | fc/2 <sup>3</sup> | –                            | 500 ns     | –           | 32.8 ms         | –           |

Example :Setting the timer mode with source clock  $fc/2^7$  Hz, and generating an interrupt 300 ms later  
(fc = 16.0 MHz)

```
LDW    (TTREG3), 927CH    : Sets the timer register (300 ms=27/fc = 927CH).
DI
SET    (EIRH), 3         : Enables INTTC4 interrupt.
EI
LD     (TC3CR), 13H      : Sets the operating clock to fc/27, and 16-bit timer mode
                                (lower byte).
LD     (TC3CR), 04H      : Sets the 16-bit timer mode (upper byte).
LD     (TC3CR), 0CH      : Starts the timer.
```

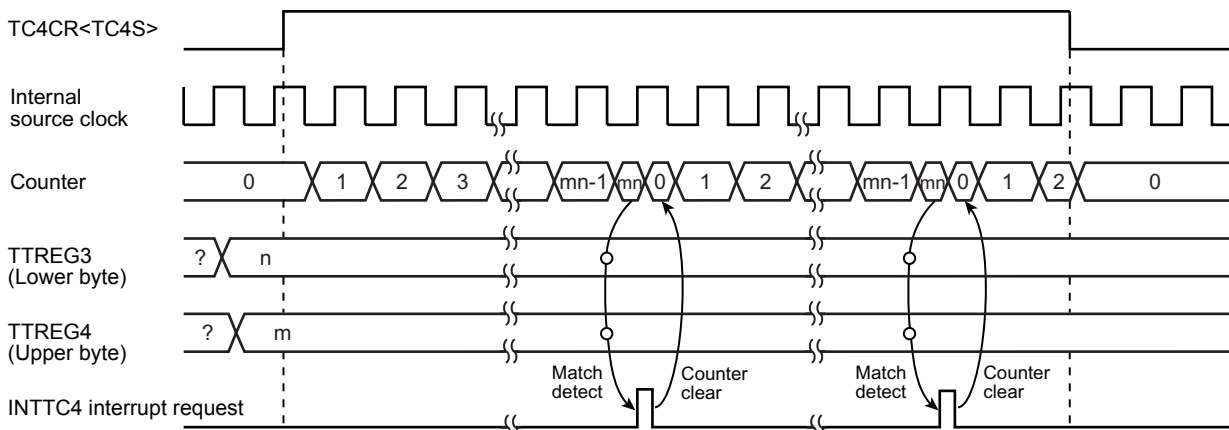


Figure 9-6 16-Bit Timer Mode Timing Chart (TC3 and TC4)

### 9.3.6 16-Bit Event Counter Mode (TC3 and 4)

In the event counter mode, the up-counter counts up at the falling edge to the TC3 pin. The TimerCounter 3 and 4 are cascadable to form a 16-bit event counter.

When a match between the up-counter and the timer register (TTREG3, TTREG4) value is detected after the timer is started by setting TC4CR<TC4S> to 1, an INTCC4 interrupt is generated and the up-counter is cleared.

After being cleared, the up-counter restarts counting at the falling edge of the input pulse to the TC3 pin. Two machine cycles are required for the low- or high-level pulse input to the TC3 pin.

Therefore, a maximum frequency to be supplied is  $f_c/2^4$  Hz in the NORMAL1 or IDLE1 mode, and  $f_s/2^4$  in the SLOW1/2 or SLEEP1/2 mode. Program the lower byte (TTREG3), and upper byte (TTREG4) in this order in the timer register. (Programming only the upper or lower byte should not be attempted.)

Note 1: In the event counter mode, fix TCjCR<TFFj> to 0. If not fixed, the  $\overline{PDOj}$ ,  $\overline{PWMj}$  and  $\overline{PPGj}$  pins may output pulses.

Note 2: In the event counter mode, do not change the TTREGj setting while the timer is running. Since TTREGj is not in the shift register configuration in the event counter mode, the new value programmed in TTREGj is in effect immediately after the programming. Therefore, if TTREGj is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 3, 4

### 9.3.7 16-Bit Pulse Width Modulation (PWM) Output Mode (TC3 and 4)

This mode is used to generate a pulse-width modulated (PWM) signals with up to 16 bits of resolution. The TimerCounter 3 and 4 are cascadable to form the 16-bit PWM signal generator.

The counter counts up using the internal clock or external clock.

When a match between the up-counter and the timer register (PWREG3, PWREG4) value is detected, the logic level output from the timer F/F4 is switched to the opposite state. The counter continues counting. The logic level output from the timer F/F4 is switched to the opposite state again by the counter overflow, and the counter is cleared. The INTTC4 interrupt is generated at this time.

Two machine cycles are required for the high- or low-level pulse input to the TC3 pin. Therefore, a maximum frequency to be supplied is  $f_c/2^4$  Hz in the NORMAL1 or IDLE1 mode, and  $f_s/2^4$  to in the SLOW1/2 or SLEEP1/2 mode.

Since the initial value can be set to the timer F/F4 by TC4CR<TFF4>, positive and negative pulses can be generated. Upon reset, the timer F/F4 is cleared to 0.

(The logic level output from the  $\overline{PWM4}$  pin is the opposite to the timer F/F4 logic level.)

Since PWREG4 and 3 in the PWM mode are serially connected to the shift register, the values set to PWREG4 and 3 can be changed while the timer is running. The values set to PWREG4 and 3 during a run of the timer are shifted by the INTTCj interrupt request and loaded into PWREG4 and 3. While the timer is stopped, the values are shifted immediately after the programming of PWREG4 and 3. Set the lower byte (PWREG3) and upper byte (PWREG3) in this order to program PWREG4 and 3. (Programming only the lower or upper byte of the register should not be attempted.)

If executing the read instruction to PWREG4 and 3 during PWM output, the values set in the shift register is read, but not the values set in PWREG4 and 3. Therefore, after writing to the PWREG4 and 3, reading data of PWREG4 and 3 is previous value until INTTC4 is generated.

For the pin used for PWM output, the output latch of the I/O port must be set to 1.

Note 1: In the PWM mode, program the timer register PWREG4 and 3 immediately after the INTTC4 interrupt request is generated (normally in the INTTC4 interrupt service routine.) If the programming of PWREGj and the interrupt request occur at the same time, an unstable value is shifted, that may result in generation of pulse different from the programmed value until the next INTTC4 interrupt request is generated.

Note 2: When the timer is stopped during PWM output, the  $\overline{PWM4}$  pin holds the output status when the timer is stopped. To change the output status, program TC4CR<TFF4> after the timer is stopped. Do not program TC4CR<TFF4> upon stopping of the timer.

Example: Fixing the  $\overline{PWM4}$  pin to the high level when the TimerCounter is stopped

CLR (TC4CR).3: Stops the timer.  
CLR (TC4CR).7 : Sets the  $\overline{\text{PWM4}}$  pin to the high level.

Note 3: To enter the STOP mode, stop the timer and then enter the STOP mode. If the STOP mode is entered without stopping of the timer when  $f_c$ ,  $f_c/2$  or  $f_s$  is selected as the source clock, a pulse is output from the  $\overline{\text{PWM4}}$  pin during the warm-up period time after exiting the STOP mode.

Table 9-7 16-Bit PWM Output Mode

| Source Clock            |                        | SLOW1/2,<br>SLEEP1/2<br>mode | Resolution             |                            | Repeated Cycle         |                            |
|-------------------------|------------------------|------------------------------|------------------------|----------------------------|------------------------|----------------------------|
| NORMAL1/2, IDLE1/2 mode |                        |                              | $f_c = 16 \text{ MHz}$ | $f_s = 32.768 \text{ kHz}$ | $f_c = 16 \text{ MHz}$ | $f_s = 32.768 \text{ kHz}$ |
| DV7CK = 0               | DV7CK = 1              |                              |                        |                            |                        |                            |
| $f_c/2^{11}$            | $f_s/2^3 \text{ [Hz]}$ | $f_s/2^3 \text{ [Hz]}$       | 128 $\mu\text{s}$      | 244.14 $\mu\text{s}$       | 8.39 s                 | 16 s                       |
| $f_c/2^7$               | $f_c/2^7$              | –                            | 8 $\mu\text{s}$        | –                          | 524.3 ms               | –                          |
| $f_c/2^5$               | $f_c/2^5$              | –                            | 2 $\mu\text{s}$        | –                          | 131.1 ms               | –                          |
| $f_c/2^3$               | $f_c/2^3$              | –                            | 500ns                  | –                          | 32.8 ms                | –                          |
| $f_s$                   | $f_s$                  | $f_s$                        | 30.5 $\mu\text{s}$     | 30.5 $\mu\text{s}$         | 2 s                    | 2 s                        |
| $f_c/2$                 | $f_c/2$                | –                            | 125 ns                 | –                          | 8.2 ms                 | –                          |
| $f_c$                   | $f_c$                  | –                            | 62.5 ns                | –                          | 4.1 ms                 | –                          |

Example :Generating a pulse with 1-ms high-level width and a period of 32.768 ms ( $f_c = 16.0 \text{ MHz}$ )

Setting ports

- LDW (PWREG3), 07D0H : Sets the pulse width.
- LD (TC3CR), 33H : Sets the operating clock to  $f_c/2^3$ , and 16-bit PWM output mode (lower byte).
- LD (TC4CR), 056H : Sets TFF4 to the initial value 1, and 16-bit PWM signal generation mode (upper byte).
- LD (TC4CR), 05EH : Starts the timer.

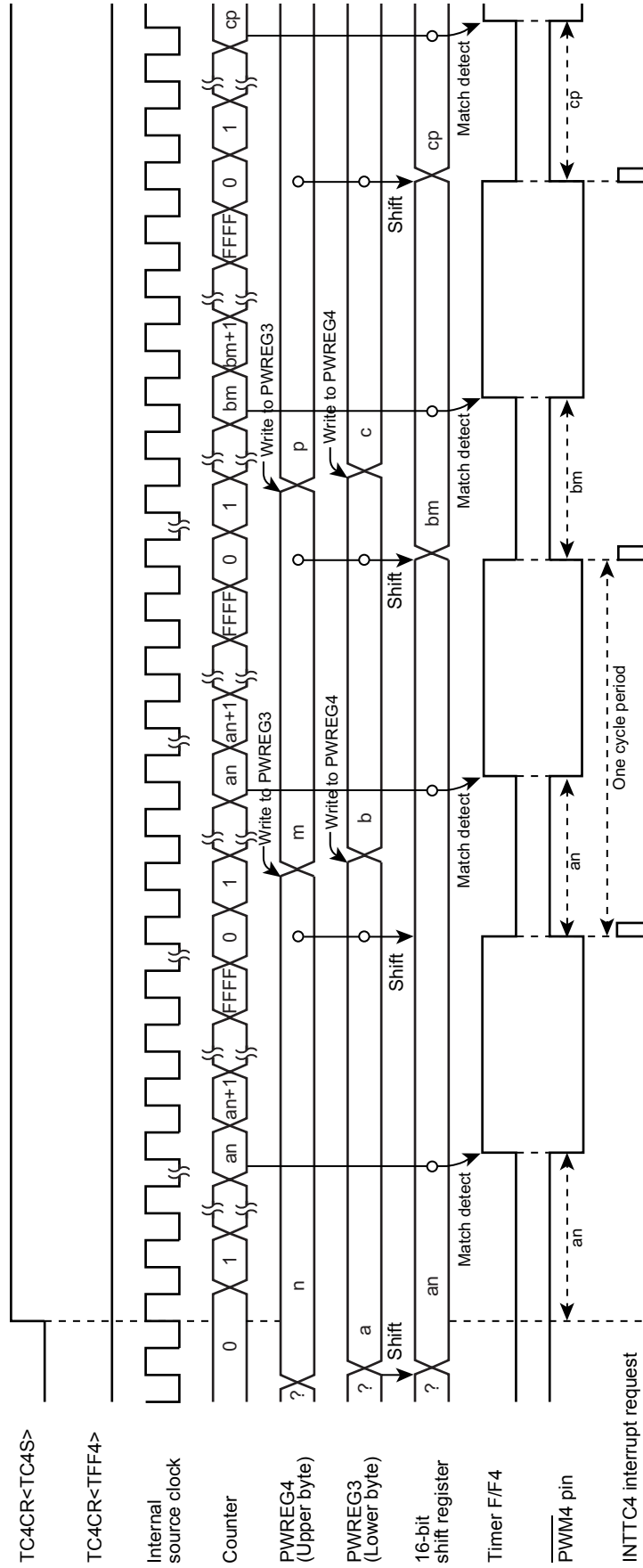


Figure 9-7 16-Bit PWM Mode Timing Chart (TC3 and TC4)

### 9.3.8 16-Bit Programmable Pulse Generate (PPG) Output Mode (TC3 and 4)

This mode is used to generate pulses with up to 16-bits of resolution. The timer counter 3 and 4 are cascaded to enter the 16-bit PPG mode.

The counter counts up using the internal clock or external clock. When a match between the up-counter and the timer register (PWREG3, PWREG4) value is detected, the logic level output from the timer F/F4 is switched to the opposite state. The counter continues counting. The logic level output from the timer F/F4 is switched to the opposite state again when a match between the up-counter and the timer register (TTREG3, TTREG4) value is detected, and the counter is cleared. The INTTC4 interrupt is generated at this time.

Two machine cycles are required for the high- or low-level pulse input to the TC3 pin. Therefore, a maximum frequency to be supplied is  $fc/2^4$  Hz in the NORMAL1 or IDLE1 mode, and  $fc/2^4$  to in the SLOW1/2 or SLEEP1/2 mode.

Since the initial value can be set to the timer F/F4 by TC4CR<TFF4>, positive and negative pulses can be generated. Upon reset, the timer F/F4 is cleared to 0.

(The logic level output from the  $\overline{PPG4}$  pin is the opposite to the timer F/F4.)

Set the lower byte and upper byte in this order to program the timer register. (TTREG3 → TTREG4, PWREG3 → PWREG4) (Programming only the upper or lower byte should not be attempted.)

For PPG output, set the output latch of the I/O port to 1.

Example :Generating a pulse with 1-ms high-level width and a period of 16.385 ms ( $fc = 16.0$  MHz)

| Setting ports |                 |  |
|---------------|-----------------|--|
| LDW           | (PWREG3), 07D0H | : Sets the pulse width.  |
| LDW           | (TTREG3), 8002H | : Sets the cycle period.   |
| LD            | (TC3CR), 33H    | : Sets the operating clock to $fc/2^3$ , and 16-bit PWM mode (lower byte). |
| LD            | (TC4CR), 057H   | : Sets TFF4 to the initial value 1, and 16-bit PWM mode (upper byte).      |
| LD            | (TC4CR), 05FH   | : Starts the timer.  |

Note 1: In the PPG mode, do not change the PWREGi and TTREGi settings while the timer is running. Since PWREGi and TTREGi are not in the shift register configuration in the PPG mode, the new values programmed in PWREGi and TTREGi are in effect immediately after programming PWREGi and TTREGi. Therefore, if PWREGi and TTREGi are changed while the timer is running, an expected operation may not be obtained.

Note 2: When the timer is stopped during PPG output, the  $\overline{PPG4}$  pin holds the output status when the timer is stopped. To change the output status, program TC4CR<TFF4> after the timer is stopped. Do not change TC4CR<TFF4> upon stopping of the timer.

Example: Fixing the  $\overline{PPG4}$  pin to the high level when the TimerCounter is stopped

CLR (TC4CR).3: Stops the timer

CLR (TC4CR).7: Sets the  $\overline{PPG4}$  pin to the high level

Note 3: i = 3, 4



### 9.3.9 Warm-Up Counter Mode

In this mode, the warm-up period time is obtained to assure oscillation stability when the system clocking is switched between the high-frequency and low-frequency. The timer counter 3 and 4 are cascadable to form a 16-bit TimerCounter. The warm-up counter mode has two types of mode; switching from the high-frequency to low-frequency, and vice-versa.

Note 1: In the warm-up counter mode, fix TCiCR<TFFi> to 0. If not fixed, the  $\overline{PD0i}$ ,  $\overline{PWMi}$  and  $\overline{PPGi}$  pins may output pulses.

Note 2: In the warm-up counter mode, only upper 8 bits of the timer register TTREG4 and 3 are used for match detection and lower 8 bits are not used.

Note 3: i = 3, 4

#### 9.3.9.1 Low-Frequency Warm-up Counter Mode (NORMAL1 → NORMAL2 → SLOW2 → SLOW1)

In this mode, the warm-up period time from a stop of the low-frequency clock fs to oscillation stability is obtained. Before starting the timer, set SYSCR2<XTEN> to 1 to oscillate the low-frequency clock. When a match between the up-counter and the timer register (TTREG4, 3) value is detected after the timer is started by setting TC4CR<TC4S> to 1, the counter is cleared by generating the INTTC4 interrupt request. After stopping the timer in the INTTC4 interrupt service routine, set SYSCR2<SYSCK> to 1 to switch the system clock from the high-frequency to low-frequency, and then clear of SYSCR2<XTEN> to 0 to stop the high-frequency clock.

Table 9-8 Setting Time of Low-Frequency Warm-Up Counter Mode (fs = 32.768 kHz)

| Maximum Time Setting<br>(TTREG4, 3 = 0100H) | Maximum Time Setting<br>(TTREG4, 3 = FF00H) |
|---|---|
| 7.81 ms                                     | 1.99 s                                      |

Example :After checking low-frequency clock oscillation stability with TC4 and 3, switching to the SLOW1 mode

|          |      |                 |  |
|----------|------|-----------------|--|
|          | SET  | (SYSCR2).6      | : SYSCR2<XTEN> ← 1   |
|          | LD   | (TC3CR), 43H    | : Sets TFF3=0, source clock fs, and 16-bit mode.   |
|          | LD   | (TC4CR), 05H    | : Sets TFF4=0, and warm-up counter mode.   |
|          | LD   | (TTREG3), 8000H | : Sets the warm-up time.<br>(The warm-up time depends on the oscillator characteristic.) |
|          | DI   |                 | : IMF ← 0  |
|          | SET  | (EIRH). 3       | : Enables the INTTC4.  |
|          | EI   |                 | : IMF ← 1  |
|          | SET  | (TC4CR).3       | : Starts TC4 and 3.  |
|          | :    | :               |  |
| PINTTC4: | CLR  | (TC4CR).3       | : Stops TC4 and 3.   |
|          | SET  | (SYSCR2).5      | : SYSCR2<SYSCK> ← 1<br>(Switches the system clock to the low-frequency clock.)           |
|          | CLR  | (SYSCR2).7      | : SYSCR2<XEN> ← 0 (Stops the high-frequency clock.)                                      |
|          | RETI |                 |  |
|          | :    | :               |  |
| VINTTC4: | DW   | PINTTC4         | : INTTC4 vector table  |

9.3.9.2 High-Frequency Warm-Up Counter Mode  
(SLOW1 → SLOW2 → NORMAL2 → NORMAL1)

In this mode, the warm-up period time from a stop of the high-frequency clock  $f_c$  to the oscillation stability is obtained. Before starting the timer, set SYSCR2<XEN> to 1 to oscillate the high-frequency clock. When a match between the up-counter and the timer register (TTREG4, 3) value is detected after the timer is started by setting TC4CR<TC4S> to 1, the counter is cleared by generating the INTTC4 interrupt request. After stopping the timer in the INTTC4 interrupt service routine, clear SYSCR2<SYSCK> to 0 to switch the system clock from the low-frequency to high-frequency, and then SYSCR2<XTEN> to 0 to stop the low-frequency clock.

Table 9-9 Setting Time in High-Frequency Warm-Up Counter Mode

| Minimum time (TTREG4, 3 = 0100H) | Maximum time (TTREG4, 3 = FF00H) |
|----------------------------------|----------------------------------|
| 16 $\mu$ s                       | 4.08 ms                          |

Example :After checking high-frequency clock oscillation stability with TC4 and 3, switching to the NORMAL1 mode

```

SET      (SYSCR2).7      : SYSCR2<XEN> ← 1
LD       (TC3CR), 63H    : Sets TFF3=0, source clock fs, and 16-bit mode.
LD       (TC4CR), 05H    : Sets TFF4=0, and warm-up counter mode.
LD       (TTREG3), 0F800H : Sets the warm-up time.
                                     (The warm-up time depends on the oscillator characteristic.)

DI       : IMF ← 0
SET      (EIRH). 3      : Enables the INTTC4.
EI       : IMF ← 1
SET      (TC4CR).3      : Starts the TC4 and 3.
:        :
PINTTC4: CLR      (TC4CR).3 : Stops the TC4 and 3.
CLR      (SYSCR2).5      : SYSCR2<SYSCK> ← 0
                                     (Switches the system clock to the high-frequency clock.)
CLR      (SYSCR2).6      : SYSCR2<XTEN> ← 0
                                     (Stops the low-frequency clock.)

RETI
:        :
VINTTC4: DW       PINTTC4 : INTTC4 vector table
    
```



# 10. Asynchronous Serial interface (UART )

## 10.1 Configuration

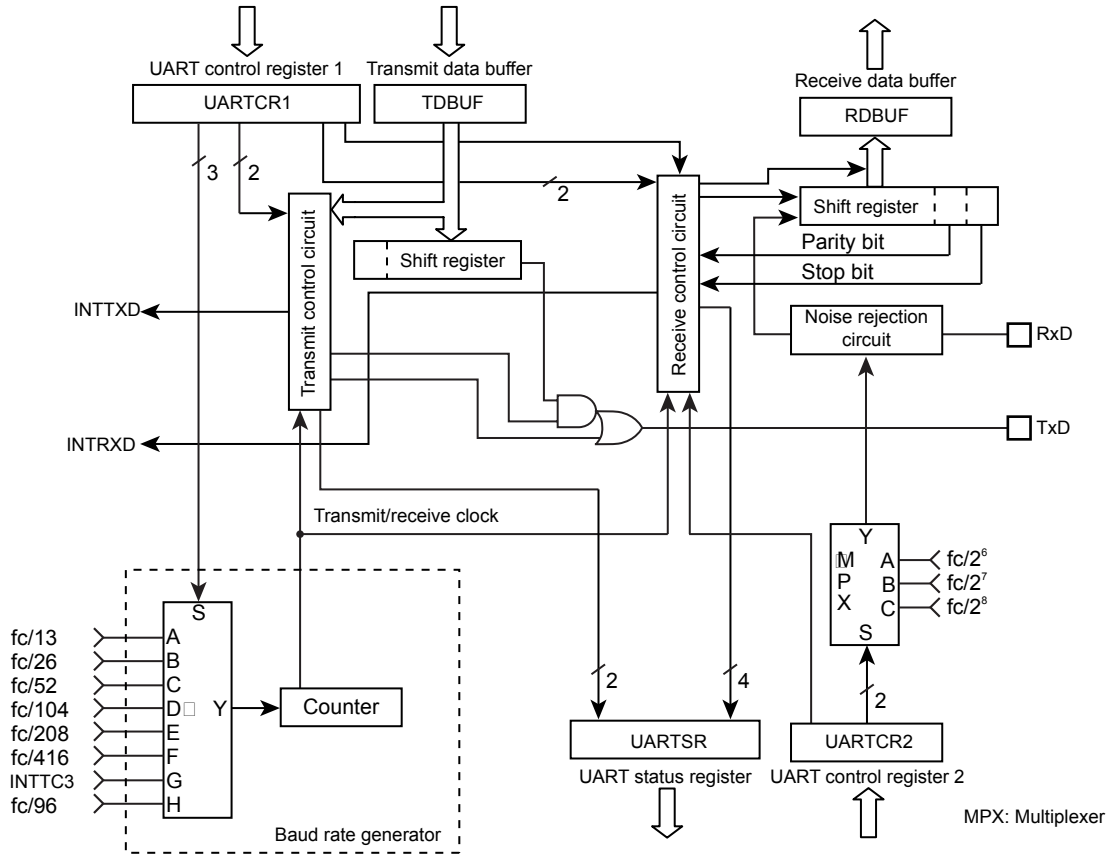


Figure 10-1 UART (Asynchronous Serial Interface)

## 10.2 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 Register1

|                    |     |     |      |      |    |     |   |   |                            |
|--------------------|-----|-----|------|------|----|-----|---|---|----------------------------|
| UARTCR1<br>(0025H) | 7   | 6   | 5    | 4    | 3  | 2   | 1 | 0 |                            |
|                    | TXE | RXE | STBT | EVEN | PE | BRG |   |   | (Initial value: 0000 0000) |

|      |                          |  |            |
|------|--------------------------|--|------------|
| TXE  | Transfer operation       | 0: Disable<br>1: Enable  | Write only |
| RXE  | Receive operation        | 0: Disable<br>1: Enable  |            |
| STBT | Transmit stop bit length | 0: 1 bit<br>1: 2 bits  |            |
| EVEN | Even-numbered parity     | 0: Odd-numbered parity<br>1: Even-numbered parity  |            |
| PE   | Parity addition          | 0: No parity<br>1: Parity  |            |
| BRG  | Transmit clock select    | 000: fc/13 [Hz]<br>001: fc/26<br>010: fc/52<br>011: fc/104<br>100: fc/208<br>101: fc/416<br>110: TC3 ( Input INTTC3)<br>111: fc/96 |            |

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

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

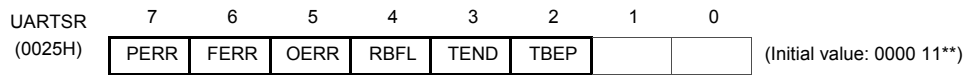
### UART Control Register2

|                    |   |   |   |   |   |       |        |   |                            |
|--------------------|---|---|---|---|---|-------|--------|---|----------------------------|
| UARTCR2<br>(0026H) | 7 | 6 | 5 | 4 | 3 | 2     | 1      | 0 |                            |
|                    |   |   |   |   |   | RXDNC | STOPBR |   | (Initial value: **** *000) |

|        |  |  |            |
|--------|--|--|------------|
| RXDNC  | Selection of RXD input noise rejectio time | 00: No noise rejection (Hysteresis input)<br>01: Rejects pulses shorter than 31/fc [s] as noise<br>10: Rejects pulses shorter than 63/fc [s] as noise<br>11: Rejects pulses shorter than 127/fc [s] as noise | Write only |
| STOPBR | Receive stop bit length                    | 0: 1 bit<br>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].

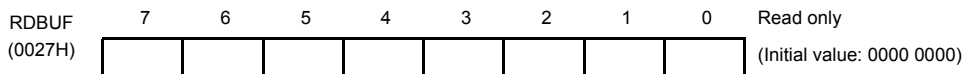
UART Status Register



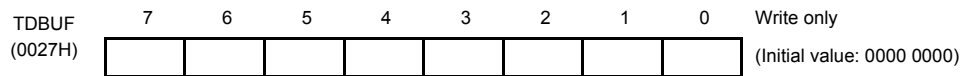
|      |                                 |   |           |
|------|---------------------------------|---|-----------|
| PERR | Parity error flag               | 0: No parity error<br>1: Parity error   | Read only |
| FERR | Framing error flag              | 0: No framing error<br>1: Framing error   |           |
| OERR | Overrun error flag              | 0: No overrun error<br>1: Overrun error   |           |
| RBFL | Receive data buffer full flag   | 0: Receive data buffer empty<br>1: Receive data buffer full                                       |           |
| TEND | Transmit end flag               | 0: On transmitting<br>1: Transmit end   |           |
| TBEP | Transmit data buffer empty flag | 0: Transmit data buffer full (Transmit data writing is finished)<br>1: Transmit data buffer empty |           |

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

UART Receive Data Buffer



UART Transmit Data Buffer



### 10.3 Transfer Data Format

In UART, an 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 follows.

| PE | STBT | Frame Length |   |   |   |   |   |   |   |   |    |    |    |
|----|------|--------------|---|---|---|---|---|---|---|---|----|----|----|
|    |      | 1            | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 0  | 0    |              |   |   |   |   |   |   |   |   |    |    |    |
| 0  | 1    |              |   |   |   |   |   |   |   |   |    |    |    |
| 1  | 0    |              |   |   |   |   |   |   |   |   |    |    |    |
| 1  | 1    |              |   |   |   |   |   |   |   |   |    |    |    |

Figure 10-2 Transfer Data Format

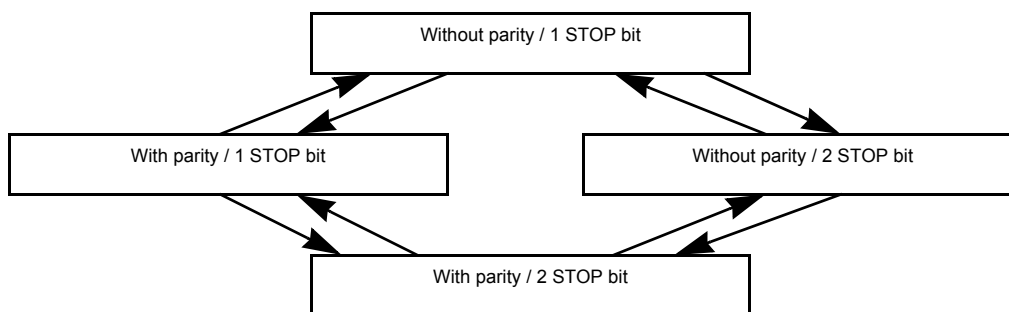


Figure 10-3 Caution on Changing Transfer Data Format

Note: In order to switch the transfer data format, perform transmit operations in the above Figure 10-3 sequence except for the initial setting.

## 10.4 Transfer Rate

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

Table 10-1 Transfer Rate (Example)

| 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 TC3 is used as the UART transfer rate (when UARTCR1<BRG> = “110”), the transfer clock and transfer rate are determined as follows:

$$\text{Transfer clock [Hz]} = \text{TC3 source clock [Hz]} / \text{TTREG3 setting value}$$

$$\text{Transfer Rate [baud]} = \text{Transfer clock [Hz]} / 16$$

## 10.5 Data Sampling Method

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).

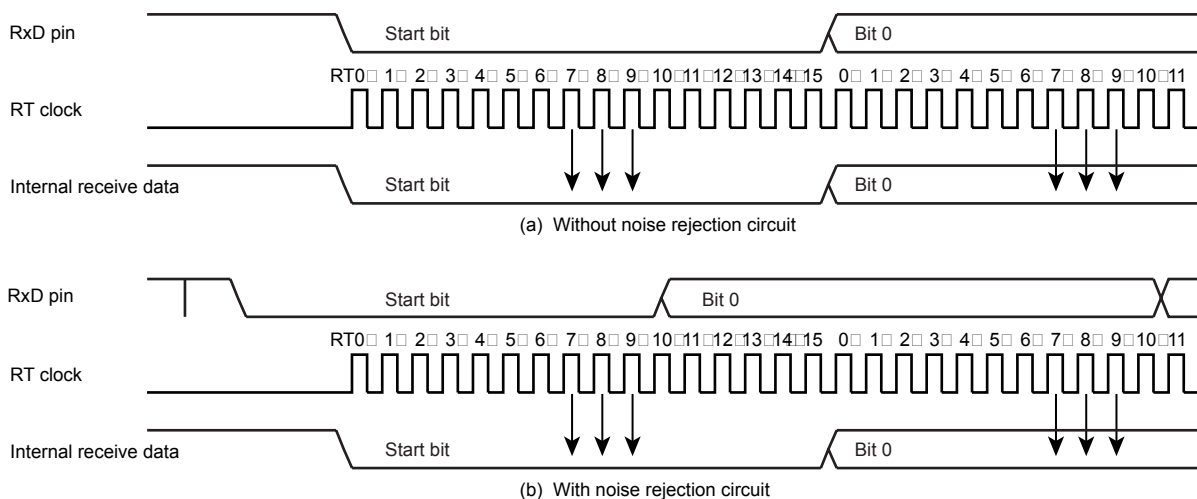


Figure 10-4 Data Sampling Method

## 10.6 STOP Bit Length

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

## 10.7 Parity

Set parity / no parity by UARTCR1<PE> and set parity type (Odd- or Even-numbered) by UARTCR1<EVEN>.

## 10.8 Transmit/Receive Operation

### 10.8.1 Data Transmit Operation

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 UARTCR1<BRG>. 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.

### 10.8.2 Data Receive Operation

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 includes 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 UARTCR1<BRG>.

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.

## 10.9 Status Flag

### 10.9.1 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.

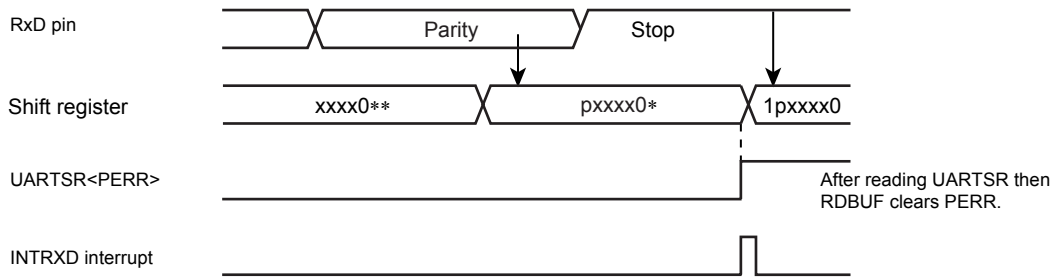


Figure 10-5 Generation of Parity Error

### 10.9.2 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.

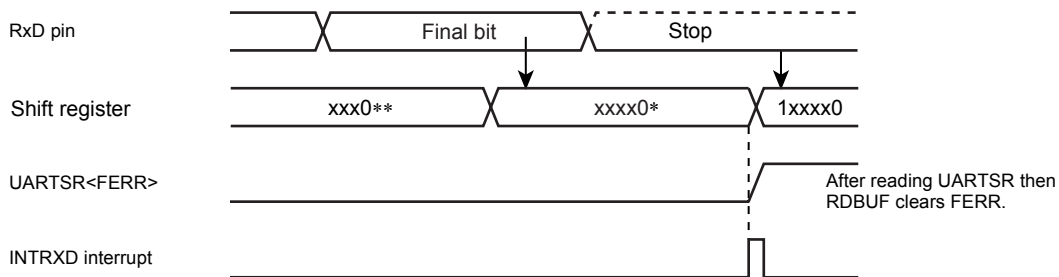


Figure 10-6 Generation of Framing Error

### 10.9.3 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.

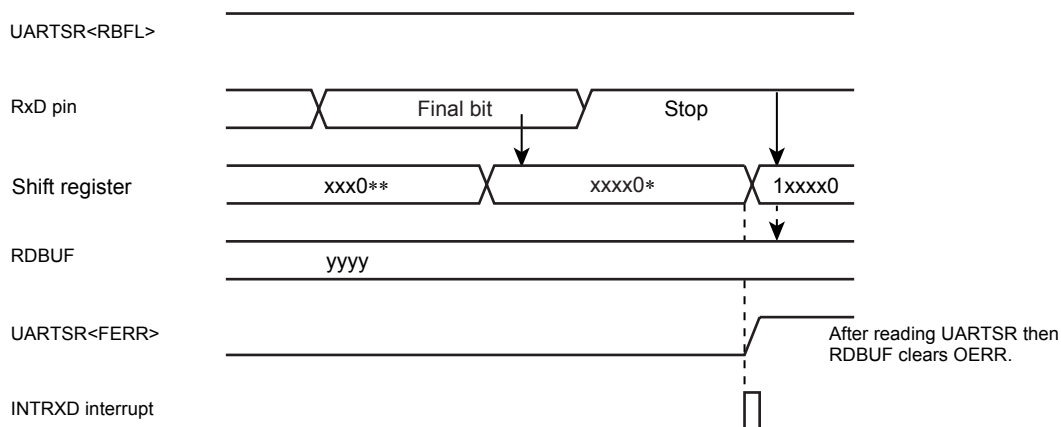


Figure 10-7 Generation of Overrun Error

### 10.9.4 Receive Data Buffer Full

Loading the received data in RDBUF sets receive data buffer full flag UARTSR<RBFL> to "1". The UARTSR<RBFL> is cleared to "0" when the RDBUF is read after reading the UARTSR.

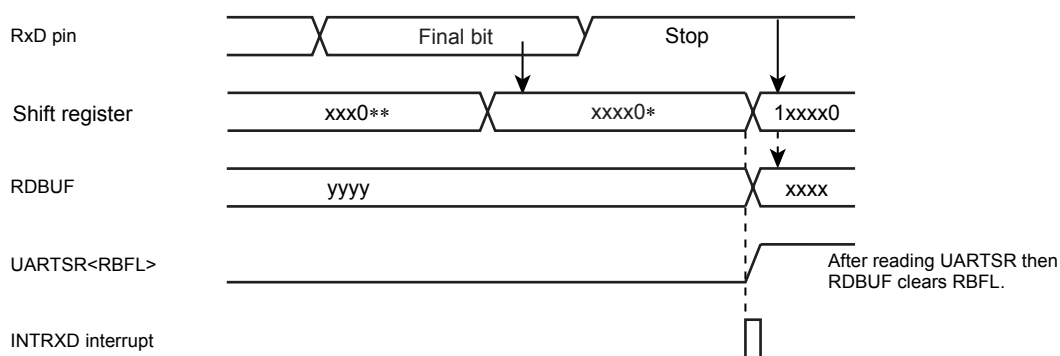


Figure 10-8 Generation of Receive Data Buffer Full

### 10.9.5 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.

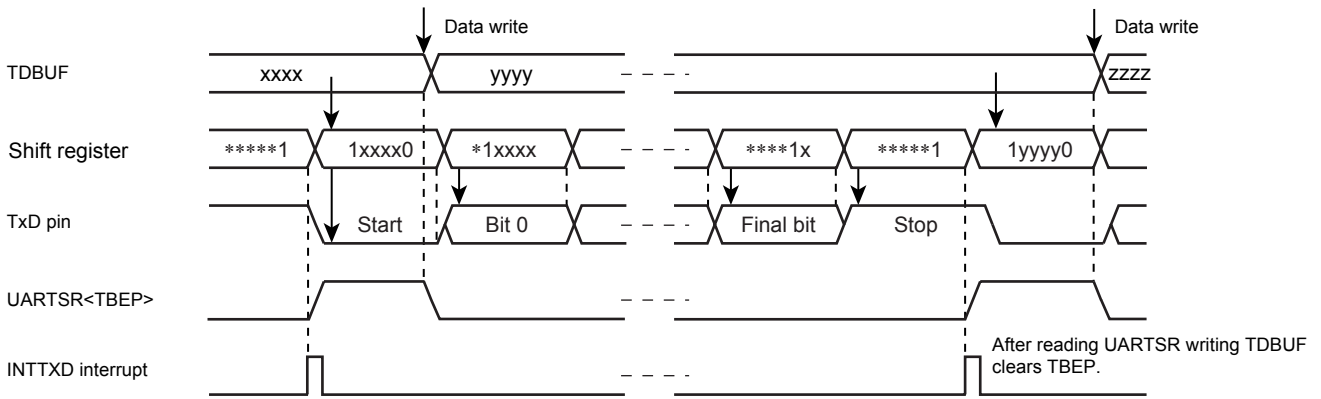


Figure 10-9 Generation of Transmit Data Buffer Empty

### 10.9.6 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” when the data transmit is started after writing the TDBUF.

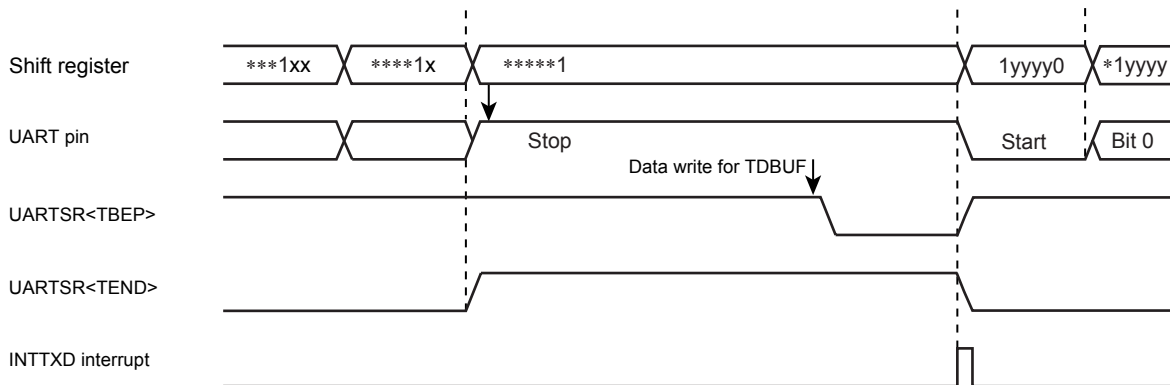


Figure 10-10 Generation of Transmit End Flag and Transmit Data Buffer Empty



# 11. Serial Expansion Interface (SEI)

SEI is one of the serial interfaces incorporated in the TMP86FH09NG. It allows connection to peripheral devices via full-duplex synchronous communication protocols. The TMP86FH09NG contain one channel of SEI.

SEI is connected with an external device through SCLK, MOSI, MISO and the terminal SS. SCLK, MOSI, MISO, and SS pins respectively are shared with P02, P03, P04 and P05. When using these ports as SCLK, MOSI, MISO, or SS pins, set the each Port Output Latch to "1".

## 11.1 Features

- The master outputs the shift clock for only a data transfer period.
- The clock polarity and phase are programmable.
- The data is 8 bits long.
- MSB or LSB-first can be selected.
- The programmable data and clock timing of SEI can be connected to almost all synchronous serial peripheral devices. Refer to "11.4 SEI Transfer Formats".
- The transfer rate can be selected from the following four (master only):  
4 Mbps, 2 Mbps, 1 Mbps, or 250 kbps (when operating at 16 MHz)
- The error detection circuit supports the following functions:
  - a. Write collision detection: When the shift register is accessed for write during transfer
  - b. Overflow detection: When new data is received while the transfer-finished flag is set (slave only)

Note: Mode fault detect function is not supported. Make sure to set SECR<MODE> bit to "1" for disabling the Mode fault detection.

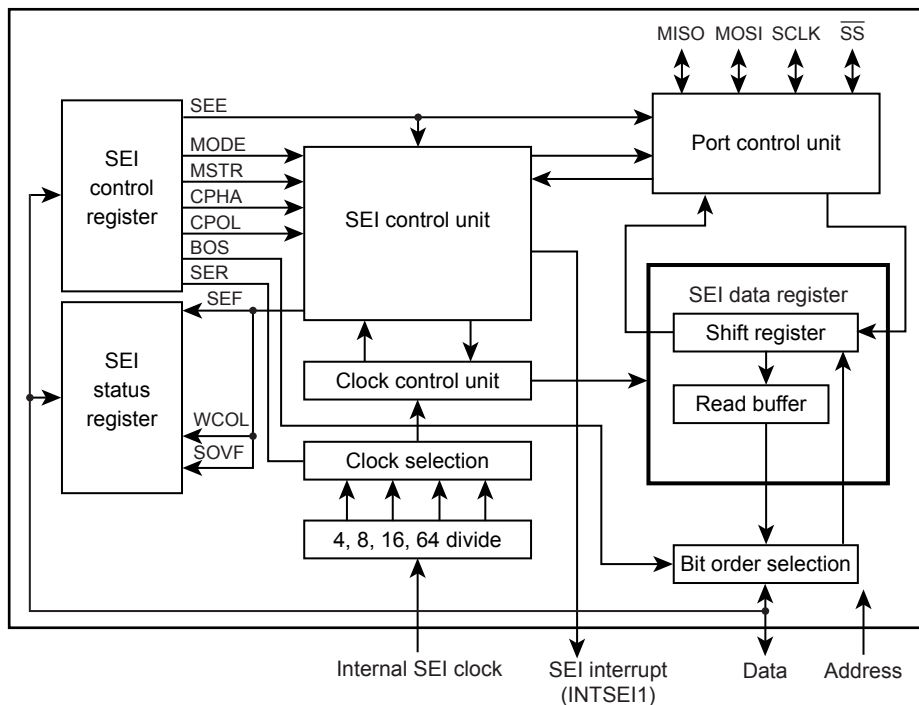


Figure 11-1 SEI (Serial Extended Interface)

## 11.2 SEI Operation

During a SEI transfer, data transmission (serial shift-out) and reception (serial shift-in) are performed simultaneously. The serial clock synchronizes the timing at which information on the two serial data lines are shifted or sampled. Slave device can be selected individually using the slave select pin ( $\overline{SS}$  pin). For unselected slave devices, data on the SEI bus cannot be taken in.

When operating as the master devices, the  $\overline{SS}$  pin can be used to indicate multiple-master bus connection.

### 11.2.1 Controlling SEI clock polarity and phase

The SEI clock allows its phase and polarity to be selected in software from four combinations available by using two bits, CPHA and CPOL (SECR<CPHL,CPOL>).

The clock polarity is set by CPOL to select between active-high or active-low (The transfer format is unaffected).

The clock phase is set by CPHA. The master device and the slave devices to communicate with must have the same clock phase and polarity.

If multiple slave devices with different transfer formats exist on the same bus, the format can be changed to that of the slave device to which to transfer.

Table 11-1 Clock Phase and Polarity

|      |   |
|------|---|
| CPHA | SEI control register (SECR 002AH) bit 2 |
| CPOL | SEI control register (SECR 002AH) bit 3 |

### 11.2.2 SEI data and clock timing

The programmable data and clock timing of SEI allows connection to almost all synchronous serial peripheral devices. Refer to Section "" 11.4 SEI Transfer Formats "".

## 11.3 SEI Pin Functions

The TMP86FH09NG have four input/output pins associated with SEI transfer. The functionality of each pin depends on the SEI device's mode (master or slave).

The SCLK pin, MOSI pin and MISO pin of all SEI devices are connected with the same name pin to each other .

### 11.3.1 SCLK pin

The SCLK pin functions as an output pin when SEI is set for master, or as an input pin when SEI is set for slave.

When SEI is set for master, serial clock is output from the SCLK pin to external devices. After the master starts transfer, eight serial clock pulses are output from the SCLK pin only during transfer.

When SEI is set for slave, the SCLK pin functions as an input pin.

During data transfer between master and slave, device operation is synchronized by the serial clock output from the master.

When the  $\overline{SS}$  pin of the slave device is "H", data is not taken in regardless of whether the serial clock is available.

For both master and slave devices, data is shifted in and out at a rising or falling edge of the serial clock, and is sampled at the opposite edge where the data is stable. The active edge is determined by SEI transfer protocols.

Note: Noise in a slave device's SCLK input may cause the device to operate erratically.

### 11.3.2 MISO/MOSI pins

The MISO and MOSI pins are used for serial data transmission/reception. The status of each pin during master and slave are shown in the table below.

Table 11-2 MISO/MOSI Pin Status

|        | MISO   | MOSI   |
|--------|--------|--------|
| Master | Input  | Output |
| Slave  | Output | Input  |

Also, the SCLK, MOSI, and MISO pins can be set for open-drain by the each pin's input/output control register (In case P0 Port, Input/output Control Register is P0OUTCR).

### 11.3.3 $\overline{SS}$ pin

The  $\overline{SS}$  pin function differently when the SEI is the master and when it is a slave.

When the SEI is a slave, this pin is used to enable the SEI transmission/reception. When the slave's  $\overline{SS}$  pin is high, the slave device ignores the serial clock from the master and places its MISO output pin in the High-Z state. Nor does it receive data from the MISO pin. When the slave's  $\overline{SS}$  pin is L, the SEI operates as slave.

## 11.4 SEI Transfer Formats

The transfer formats are set using CPHA and CPOL (SECR<CPHA,CPOL>). CPHA allows transfer protocols to be selected between two.

### 11.4.1 CPHA (SECR register bit 2) = 0 format

Figure 11-2 shows a transfer format where CPHA = 0.

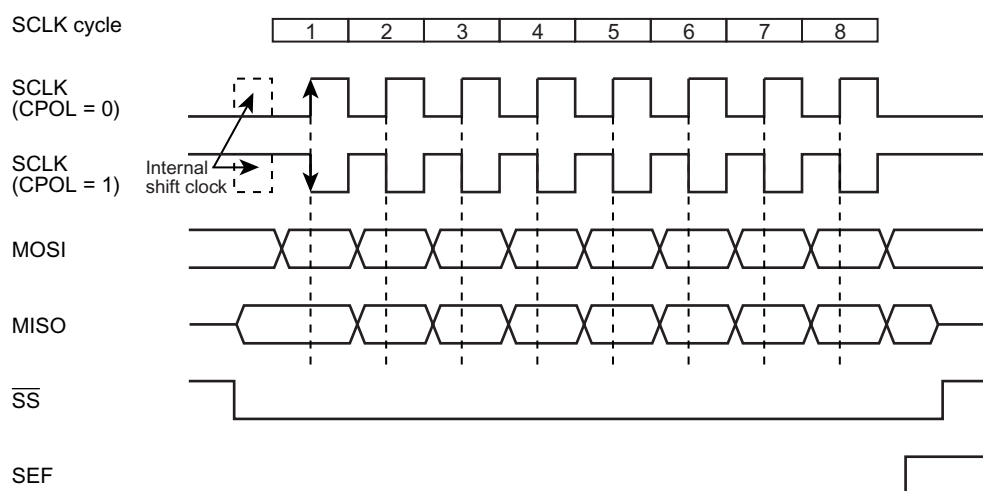


Figure 11-2 Transfer Format where CPHA = 0

Table 11-3 Transfer Format Details where CPHA = 0

|          | SCLK Level when not Communicating (IDLE) | Data Shift                     | Data Sampling                  |
|----------|--|--------------------------------|--------------------------------|
| CPOL = 0 | "L" level                                | Falling edge of transfer clock | Rising edge of transfer clock  |
| CPOL = 1 | "H" level                                | Rising edge of transfer clock  | Falling edge of transfer clock |

- In master mode, transfer is initiated by writing new data to the SEDR register. At this time, the new data changes state on the MOSI pin a half clock period before the shift clock starts pulsing. Use BOS (SECR<BOS>) to select whether the data should be shifted out beginning with the MSB or LSB. The SEF flag (SESR<SEF>) is set after the last shift cycle.
- In slave mode, writing data to the SEDR register is inhibited when the  $\overline{SS}$  pin is "L". A write during this period causes collision of writes, so that the WCOL flag (SESR<WCOL>) is set. Therefore, when writing data to the SEDR (SEI Data Register) after the SEF flag is set upon completion of transfer, make sure the  $\overline{SS}$  pin goes "H" again before writing the next data to the SEDR register.

Note: In slave mode, be careful not to write data while the SEF flag is set and the  $\overline{SS}$  pin remains "L".

11.4.2 CPHA = 1 format

Figure 11-3 shows a transfer format where CPHA = 1.

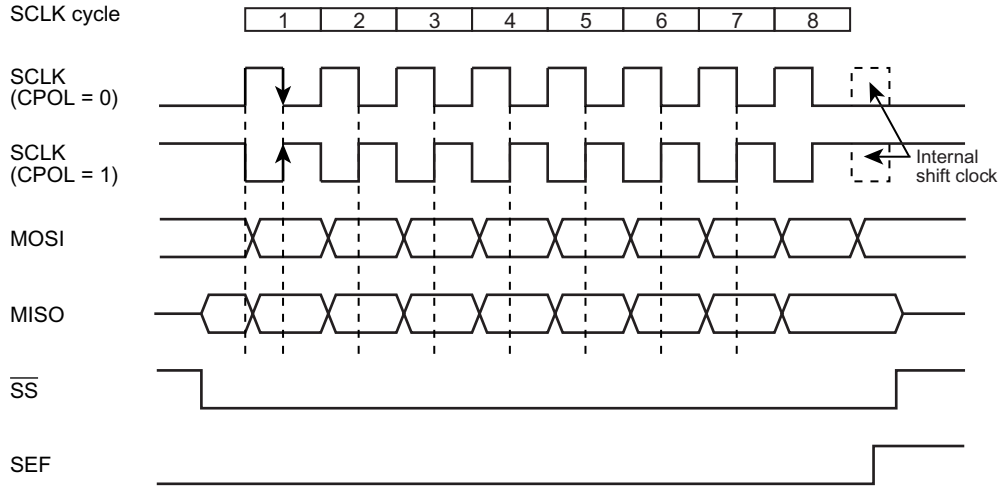


Figure 11-3 Transfer Format where CPHA = 1

Table 11-4 Transfer Format Details where CPHA = 1

|        | SCLK Level when Not Communicating (IDLE) | Data Shift                     | Data Sampling                  |
|--------|--|--------------------------------|--------------------------------|
| CPOL=0 | "L" level                                | Rising edge of transfer clock  | Falling edge of transfer clock |
| CPOL=1 | "H" level                                | Falling edge of transfer clock | Rising edge of transfer clock  |

- In master mode, transfer is initiated by writing new data to the SEDR register. The new data changes state on the MOSI pin at the first edge of the shift clock. Use BOS (SECR<BOS>) to select whether the data should be shifted out beginning with the MSB or LSB.
- In slave mode, unlike in the case of CPHA = 0 format, data can be written to the SEDR (SEI Data Register) regardless of whether the SS pin is "L" or "H".  
In both master and slave modes, the SEF flag (SESR<SEF>) is set after the last shift cycle. Writing data to the SEDR register while data transfer is in progress causes collision of writes. Therefore, wait until the SEF flag is set before writing new data to the SEDR register.

## 11.5 Functional Description

Figure 11-4 shows how the SEI master and slave are connected.

When the master device sends data from its MOSI pin to a slave device's MOSI pin, the slave device returns data from its MISO pin to the master device's MISO pin. This means that data are exchanged between master and slave via full-duplex communication, with data output and input operations synchronized by the same clock signal. After end of transfer, the transmit byte in 8 bit shift register is replaced with the receive byte.

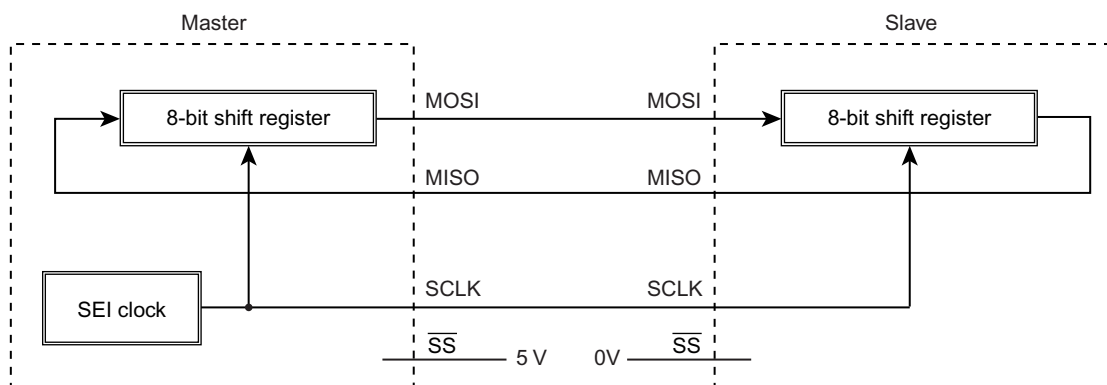


Figure 11-4 Master and Slave Connection in SEI

Figure 11-5 shows an example of SEI system configuration.

The port used as SEI output can be set to be an open-drain output. Therefore, this port can be connected to multiple devices.

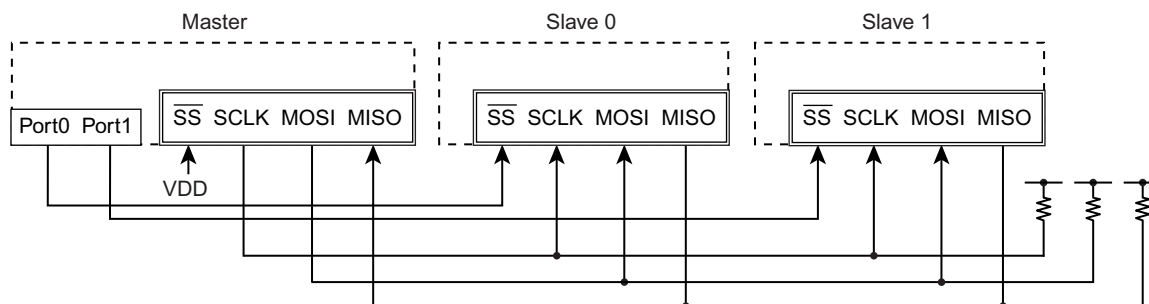


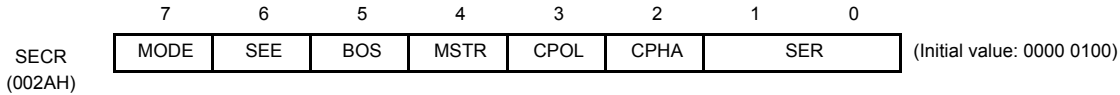
Figure 11-5 Example of SEI System Configuration (1 Master and 2 Slaves)

In the example system configuration, all SCLK pins are interconnected, so are all MOSI and MISO pins. One SEI device is set for master, while all other SEI devices on the SEI bus are set for slave. The master device sends data from its SCLK and MOSI pins to the slave devices' SCLK and MOSI pins. One selected slave device returns data from its MISO pin to the master device's MISO pin.

## 11.6 SEI Registers

The SEI interface has the SEI Control Register (SECR), SEI Status Register (SESR), and SEI Data Register (SEDR) which are used to set up the SEI system and enable/disable SEI operation.

### 11.6.1 SEI Control Register (SECR)



Read-modify-write instruction are prohibited

|      |                                    |   |     |
|------|------------------------------------|---|-----|
| MODE | Mode fault detection <sup>#1</sup> | 0: Enables mode fault detection<br>1: Disables mode fault detection<br>It is available in Master mode only.<br>(Note: Make sure to set <MODE> bit to "1" for disabling Mode fault detection | R/W |
| SEE  | SEI operation <sup>#2</sup>        | 0: Disables SEI operation<br>1: Enables SEI operation   |     |
| BOS  | Bit order selection                | 0: Transmitted beginning with the MSB (bit 7) of SEDR register<br>1: Transmitted beginning with the LSB (bit 0) of SEDR register  |     |
| MSTR | Mode selection <sup>#3</sup>       | 0: Sets SEI for slave<br>1: Sets SEI for master   |     |
| CPOL | Clock polarity                     | 0: Selects active-"H" clock. SCLK remains "L" when IDLE.<br>1: Selects active-"L" clock. SCLK remains "H" when IDLE.  |     |
| CPHA | Clock phase                        | Selects clock phase. For details, refer to Section "SEI Transfer Formats".  |     |
| SER  | Selects SEI transfer rate          | 00: Divide-by-4<br>01: Divide-by-8<br>10: Divide-by-16<br>11: Divide-by-64  |     |

- #1 If mode fault detection is enabled, an interrupt is generated when the MODF flag (SESR<MODF>) is set.
- #2 SEI operation can only be disabled after transfer is completed. Before the SEI can be used, the each Port Control Register and Output Latch Control must be set for the SEI function (In case P0 port, P0OUTCR and PODR).  
When using the SEI as the master, set the SECR<SEE> bit to "1" (to enable SEI operation) and then place transmit data in the SEDR register. This initiates transmission/reception.
- #3 Master/slave settings must be made before enabling SEI operation (This means that the SECR<MSTR> bit must first be set before setting the SECR<SEE> bit to "1").

#### 11.6.1.1 Transfer rate

- (1) Master mode (Transfer rate =  $f_c / \text{Internal clock divide ratio}$  (unit : bps))

The table below shows the relationship between settings of the SER bit and transfer bit rates when the SEI is operating as the master.

Table 11-5 SEI Transfer Rate

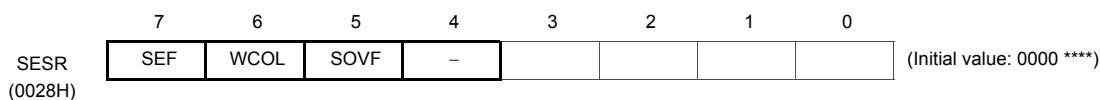
| SER | Internal Clock Divide Ratio of SEI | Transfer Rate when $f_c = 16 \text{ MHz}$ |
|-----|------------------------------------|---|
| 00  | 4                                  | 4 Mbps                                    |
| 01  | 8                                  | 2 Mbps                                    |
| 10  | 16                                 | 1 Mbps                                    |
| 11  | 64                                 | 250 kbps                                  |

(2) Slave mode

When the SEI is operating as a slave, the serial clock is input from the master and the setting of the SER bit has no effect. The maximum transfer rate is  $f_c/4$ .

Note: Take note of the following relationship between the serial clock speed and  $f_c$  on the master side:  
 $15.625 \text{ kbps} < \text{Transfer rate} < f_c/4 \text{ bps}$   
 Example)  $15.625 \text{ kbps} < \text{Transfer rate} < 4 \text{ Mbps}$  ( $f_c = 16 \text{ MHz}$  at  $V_{DD} = 4.5 \text{ to } 5.5 \text{ V}$ )

11.6.2 SEI Status Register (SESR)

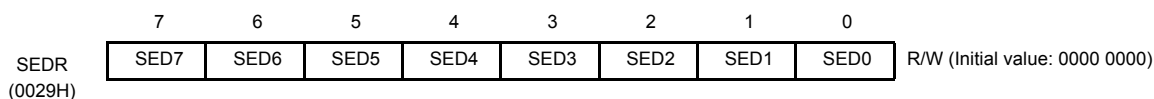


|      |   |   |           |
|------|---|---|-----------|
| SEF  | Transfer-finished flag <sup>#1</sup>      | 0: Transfer in progress<br>1: Transfer completed                          | Read only |
| WCOL | Write collision error flag <sup>#2</sup>  | 0: No write collision error occurred<br>1: Write collision error occurred |           |
| SOVF | Overflow error flag (slave) <sup>#3</sup> | 0: No overflow occurred<br>1: Overflow occurred                           |           |

- #1 The SEF flag is automatically set at completion of transfer. The SEF flag thus set is automatically cleared by reading the SESR register and accessing the SEDR register.
- #2 The WCOL flag is automatically set by a write to the SEDR register while transfer is in progress. Writing to the SEDR register during transfer has no effect. The WCOL flag thus set is automatically cleared by reading the SESR register and accessing the SEDR register. No interrupts are generated for reasons that the WCOL flag is set.
- #3 During master mode:  
This bit does not function; its data when read is "0".  
During slave mode:  
The SOVF flag is automatically set when the device finishes reading the next data while the SEF flag is set. The SOVF flag thus set is automatically cleared by reading the SESR register and accessing the SEDR register. The SOVF flag also is cleared by a switchover to master mode. No interrupts are generated for reasons that the SOVF flag is set.

11.6.3 SEI Data Register (SEDR)

The SEI Data Register (SEDR) is used to send and receive data. When the SEI is set for master, data transfer is initiated by writing to this SEDR register. If the master device needs to write to the SEDR register after transfer began, always check to see by means of an interrupt or by polling that the SEF flag (SESR<SEF>) is set, before writing to the SEDR register.



## 11.7 Interrupt Generation

The SEI for the TMP86FH09NG uses INTSEI1. When the SESR<SEF> changes state from “0” to “1”, respective interrupts is generated.

Table 11-6 SEI Interrupt

|                                   |                             |
|-----------------------------------|-----------------------------|
| SEI interrupt channel 1 (INTSEI1) | Interrupt generated for SEF |
|-----------------------------------|-----------------------------|

## 11.8 SEI System Errors

The SEI has the facility to detect following two system errors.

- Write collision error:

When the SEDR register is accessed for write during transfer.

- Overflow error:

When the new data byte is shift in before the previous data byte is read in slave mode.

### 11.8.1 Write collision error

Collision of writes occurs when an attempt is made to write to the SEDR register while transfer is in progress. Because the SEDR register is not configured as dual-buffers when sending data, a write to the SEDR register directly results in writing to the SEI shift register. Therefore, writing to the SEDR register while transfer is in progress causes a write collision error.

In no case is data transfer stopped in the middle, so that the write data which caused a write collision error will not be written to the shift register. Because slaves cannot control the timing at which the master starts a transfer, collision of writes normally occurs on the slave side.

Write collision errors do not normally occur on the master side because the master has the right to perform a transfer at any time, but in view of SEI logic both the master and slaves have the facility to detect write collision errors.

A write collision error tends to occur on the slave side when the master shifts out data at a speed faster than that at which the slave processes the transferred data. More specifically, a write collision error occurs in cases where the slave transfers a new value to the SEDR register when the master already started a shift cycle for the next byte.

### 11.8.2 Overflow error

The transfer bit rate on the SEI bus is determined by the master. A high bit rate causes a problem that a slave cannot keep abreast with transfer from the master, because the master is shifting out data faster than can be processed by the slave. The SEI module uses the SOVF flag (SESR<SOVF>) to detect that data has overflowed.

The SOVF flag is set in the following cases:

- When the SEI module is set for slave
- When the old data byte remains to be read while a new data byte has been received

When the SOVF flag is set, the SEDR register is overwritten with a new data byte.

Note: Please carefully examine the communication processing routine and communication rate when designing your application system.

## 11.9 Bus Driver Protection

- One method to protect the device against latch-up due to collision of the bus drivers is the use of an open-drain option. This means changing the SEI pins' CMOS outputs to the open-drain type, which is accomplished by setting the SCLK, MOSI, and MISO pins for open-drain individually by using the each Port

Input/output Control Register. In this case, these pins must be provided with pull-up resistors external to the chip.

- When using the SEI pins as CMOS outputs, we recommend connecting them to the bus via resistors in order to protect the device against collision of drivers. However, be sure to select the appropriate resistance value which will not affect actual device operation (Example: 1  $\Omega$  to several k $\Omega$ ).

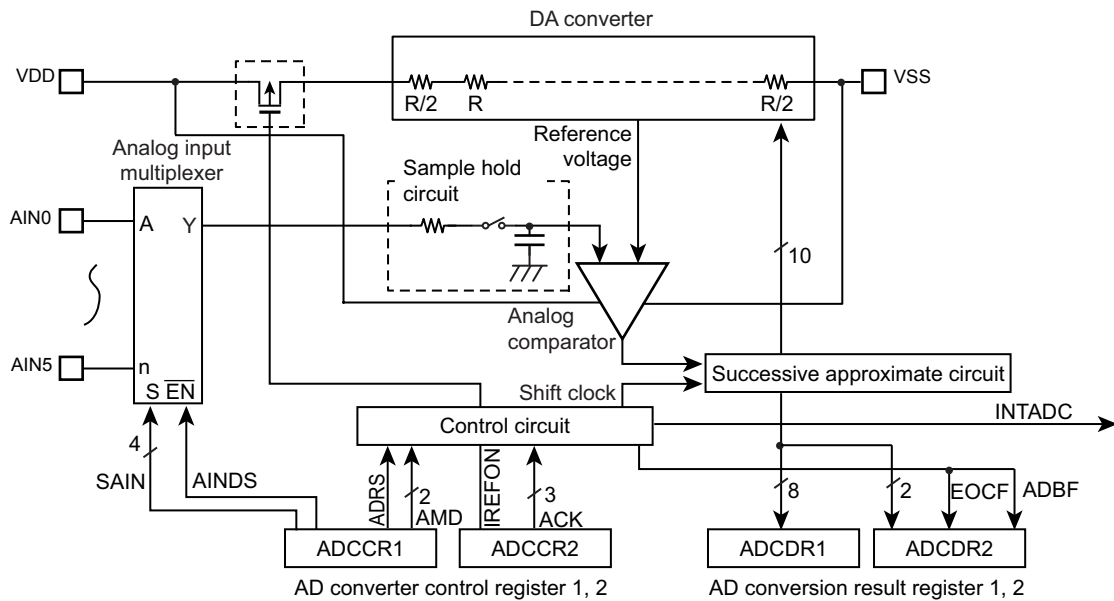
## 12. 10-bit AD Converter (ADC)

The TMP86FH09NG have a 10-bit successive approximation type AD converter.

### 12.1 Configuration

The circuit configuration of the 10-bit AD converter is shown in Figure 12-1.

It consists of control register ADCCR1 and ADCCR2, converted value register ADCSR1 and ADCDR2, a DA converter, a sample-hold circuit, a comparator, and a successive comparison circuit.



Note: Before using AD converter, set appropriate value to I/O port register combining a analog input port. For details, see the section on "I/O ports".

Figure 12-1 10-bit AD Converter

## 12.2 Register configuration

The AD converter consists of the following four registers:

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).

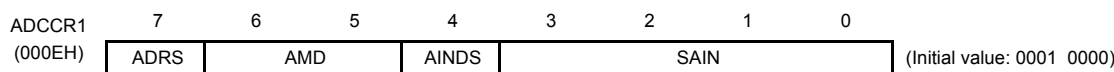
3. AD converted value register 1 (ADCDR1)

This register used to store the digital value after being converted by the AD converter.

4. AD converted value register 2 (ADCDR2)

This register monitors the operating status of the AD converter.

### AD Converter Control Register 1



|       |                             |  |     |
|-------|-----------------------------|--|-----|
| ADRS  | AD conversion start         | 0: -<br>1: AD conversion start   | R/W |
| AMD   | AD operating mode           | 00: AD operation disable<br>01: Software start mode<br>10: Reserved<br>11: Repeat mode   |     |
| AINDS | Analog input control        | 0: Analog input enable<br>1: Analog input disable  |     |
| SAIN  | Analog input channel select | 0000: AIN0<br>0001: AIN1<br>0010: AIN2<br>0011: AIN3<br>0100: AIN4<br>0101: AIN5<br>0110: Reserved<br>0111: Reserved<br>1000: Reserved<br>1001: Reserved<br>1010: Reserved<br>1011: Reserved<br>1100: Reserved<br>1101: Reserved<br>1110: Reserved<br>1111: Reserved |     |

Note 1: Select analog input channel during AD converter stops (ADCCR2<ADBF> = "0").

Note 2: When the analog input channel is all use disabling, the ADCCR1<AINDIS> should be set to "1".

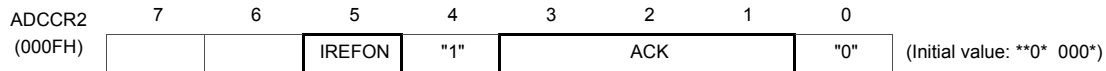
Note 3: During conversion, Do not perform port output instruction to maintain a precision for all of the pins because analog input port use as general input port. And for port near to analog input, Do not input intense signaling of change.

Note 4: The ADCCR1<ADRS> is automatically cleared to "0" after starting conversion.

Note 5: Do not set ADCCR1<ADRS> newly again 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).

Note 6: After STOP mode started, AD converter control register1 (ADCCR1) is all initialized and no data can be written in this register. Therefore, to use AD converter again, set the ADCCR1 newly after returning to NORMAL1 or NORMAL2 mode.

**AD Converter Control Register 2**



|        |   |  |     |
|--------|---|--|-----|
| IREFON | DA converter (Ladder resistor) connection control                                     | 0: Connected only during AD conversion<br>1: Always connected  |     |
| ACK    | AD conversion time select<br>(Refer to the following table about the conversion time) | 000: Reserved<br>001: Reserved<br>010: 78/fc<br>011: 156/fc<br>100: 312/fc<br>101: 624/fc<br>110: 1248/fc<br>111: Reserved | R/W |

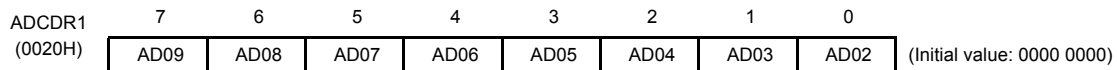
- Note 1: Always set bit0 in ADCCR2 to "0" and set bit4 in ADCCR2 to "1".
- Note 2: When a read instruction for ADCCR2, bit6 to 7 in ADCCR2 read in as undefined data.
- Note 3: After STOP mode started, AD converter control register2 (ADCCR2) is all initialized and no data can be written in this register. Therefore, to use AD converter again, set the ADCCR2 newly after returning to NORMAL1 or NORMAL2 mode.

**Table 12-1 ACK setting and Conversion time**

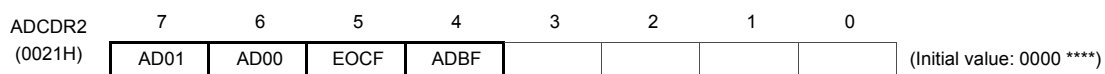
| Condition<br>ACK | Conversion time | 16 MHz  | 8 MHz    | 4 MHz    | 2 MHz    | 10 MHz   | 5 MHz    | 2.5 MHz  |
|------------------|-----------------|---------|----------|----------|----------|----------|----------|----------|
| 000              | Reserved        |         |          |          |          |          |          |          |
| 001              | Reserved        |         |          |          |          |          |          |          |
| 010              | 78/fc           | -       | -        | 19.5 μs  | 39.0 μs  | -        | 15.6 μs  | 31.2 μs  |
| 011              | 156/fc          | -       | 19.5 μs  | 39.0 μs  | 78.0 μs  | 15.6 μs  | 31.2 μs  | 62.4 μs  |
| 100              | 312/fc          | 19.5 μs | 39.0 μs  | 78.0 μs  | 156.0 μs | 31.2 μs  | 62.4 μs  | 124.8 μs |
| 101              | 624/fc          | 39.0 μs | 78.0 μs  | 156.0 μs | -        | 62.4 μs  | 124.8 μs | -        |
| 110              | 1248/fc         | 78.0 μs | 156.0 μs | -        | -        | 124.8 μs | -        | -        |
| 111              | Reserved        |         |          |          |          |          |          |          |

- Note 1: Setting for "-" in the above table are inhibited.    fc: High Frequency oscillation clock [Hz]
- Note 2: Set conversion time setting should be kept more than the following time by Power supply voltage(VDD) .
- VDD = 4.5 to 5.5 V      15.6 μs and more

**AD Converted value Register 1**



**AD Converted value Register 2**



|      |                         |  |           |
|------|-------------------------|--|-----------|
| EOCF | AD conversion end flag  | 0: Before or during conversion<br>1: Conversion completed  |           |
| ADBF | AD conversion BUSY flag | 0: During stop of AD conversion<br>1: During AD conversion | Read only |

Note 1: The ADCDR2<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: The ADCDR2<ADBF> is set to "1" when AD conversion starts, and cleared to "0" when AD conversion finished. It also is cleared upon entering STOP mode .

Note 3: If a read instruction is executed for ADCDR2, read data of bit3 to bit0 are unstable.

## 12.3 Function

### 12.3.1 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 converted value registers (ADCDR1, ADCDR2) and at the same time ADCDR2<EOCF> is set to 1, the AD conversion finished interrupt (INTADC) is generated.

ADRS is automatically cleared after AD conversion has started. Do not set ADCCR1<ADRS> newly again (Restart) 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).

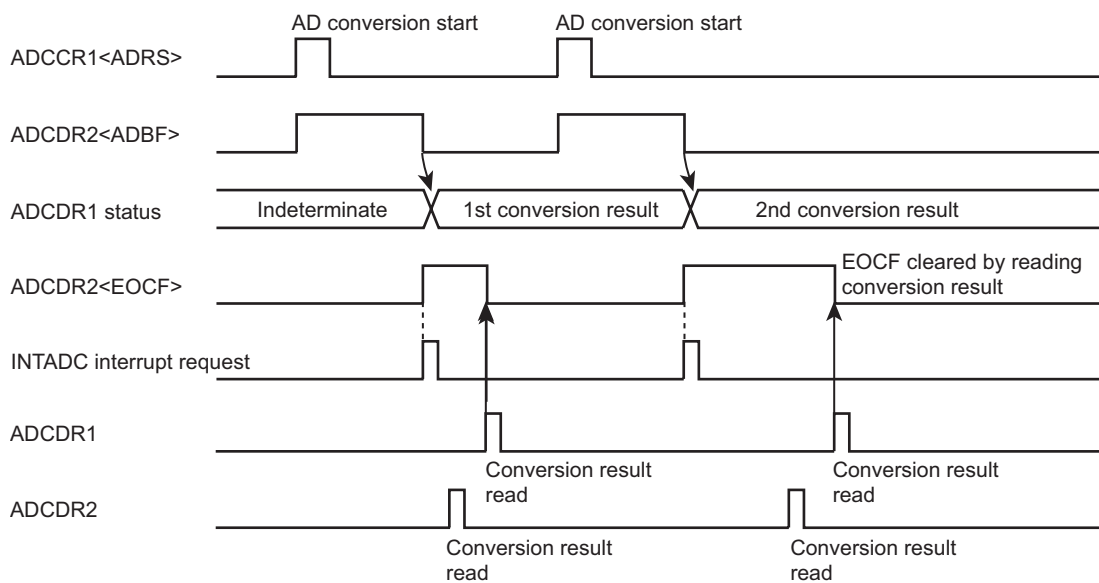


Figure 12-2 Software Start Mode

### 12.3.2 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” (Repeat mode).

After completion of the AD conversion, the conversion result is stored in AD converted value 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 “00” (Disable mode) by writing 0s. The AD convert operation is stopped immediately. The converted value at this time is not stored in the AD converted value register.

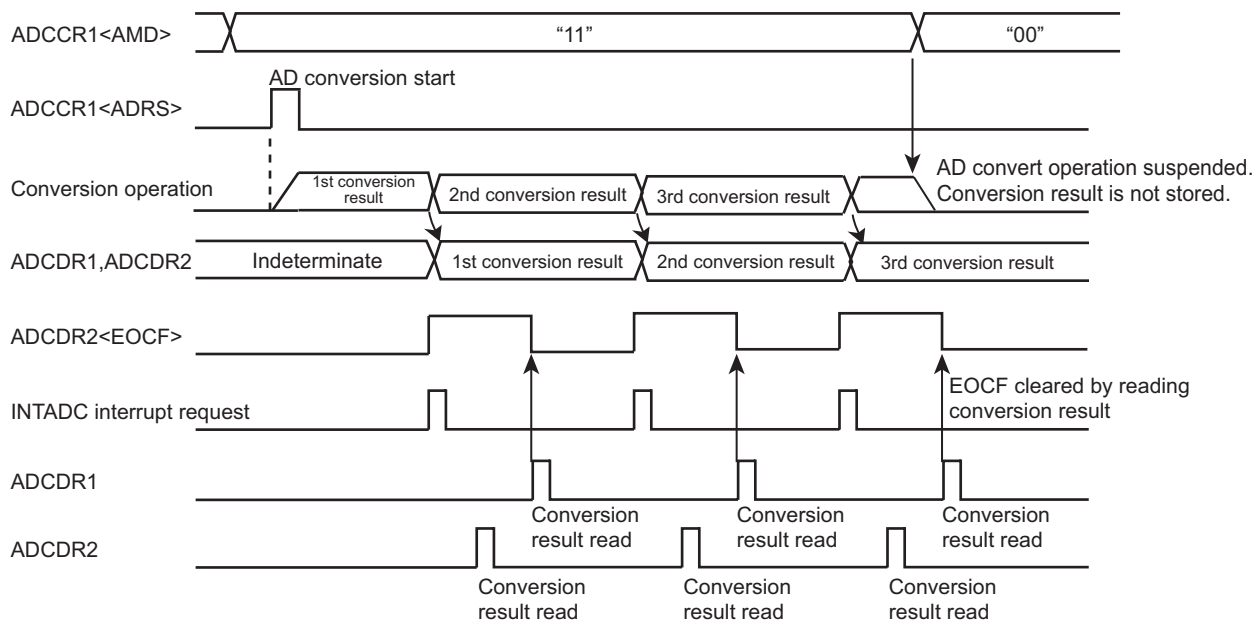


Figure 12-3 Repeat Mode

### 12.3.3 Register Setting

- 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).
- 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 Figure 12-1 and AD converter control register 2.
  - Choose IREFON for DA converter control.
- After setting up (1) and (2) above, set AD conversion start (ADRS) of AD converter control register 1 (ADCCR1) to "1". If software start mode has been selected, AD conversion starts immediately.
- After an elapse of the specified AD conversion time, the AD converted value is stored in AD converted value register 1 (ADCDR1) and the AD conversion finished flag (EOCF) of AD converted value register 2 (ADCDR2) is set to "1", upon which time AD conversion interrupt INTADC is generated.
- 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.

Example :After selecting the conversion time 19.5  $\mu$ s 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 0009EH and store the upper 8 bits in address 0009FH in RAM. The operation mode is software start mode.

```

: (port setting)      :                               ;Set port register appropriately before setting AD
                               ;converter registers.
:                               ;
:                               ;(Refer to section I/O port in details)
LD      (ADCCR1) , 00100011B   ; Select AIN3
LD      (ADCCR2) , 11011000B   ;Select conversion time(312/fc) and operation
                               ;mode
SLOOP : SET      (ADCCR1) . 7    ; ADRS = 1(AD conversion start)
        TEST     (ADCCR2) . 5    ; EOCF= 1 ?
        JRS      T, SLOOP
        LD      A , (ADCDR2)     ; Read result data
        LD      (9EH) , A
        LD      A , (ADCDR1)     ; Read result data
        LD      (9FH), A

```

## 12.4 STOP Modes during AD Conversion

When standby mode (STOP 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 standby mode (STOP mode).) When restored from standby mode (STOP mode), AD conversion is not automatically restarted, so it is necessary to restart AD conversion. Note that since the analog reference voltage is automatically disconnected, there is no possibility of current flowing into the analog reference voltage.

## 12.5 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 12-4.

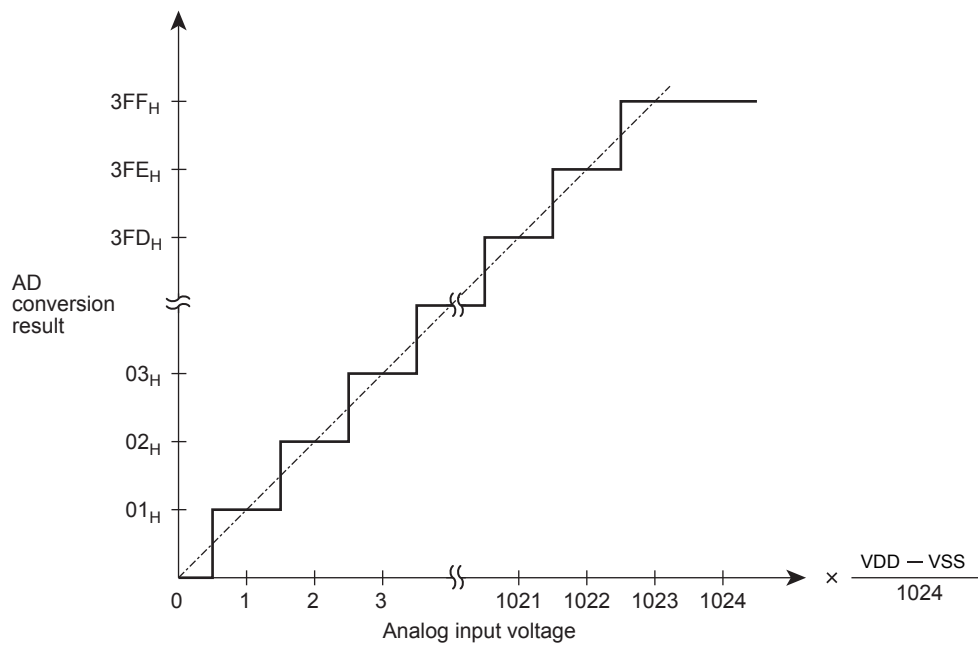


Figure 12-4 Analog Input Voltage and AD Conversion Result (Typ.)

## 12.6 Precautions about AD Converter

### 12.6.1 Analog input pin voltage range

Make sure the analog input pins (AIN0 to AIN5) are used at voltages within VDD to VSS. 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.

### 12.6.2 Analog input shared pins

The analog input pins (AIN0 to AIN5) 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.

### 12.6.3 Noise Countermeasure

The internal equivalent circuit of the analog input pins is shown in Figure 12-5. 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 k $\Omega$  or less. Toshiba also recommends attaching a capacitor external to the chip.

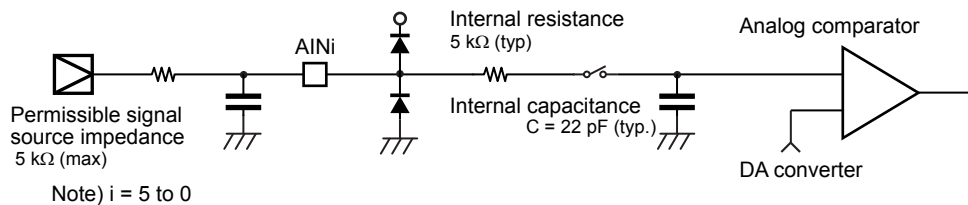


Figure 12-5 Analog Input Equivalent Circuit and Example of Input Pin Processing





## 13.2 Control

The P34 to P37 (STOP2 to STOP5) pins can individually be disabled/enabled using Key-on Wakeup Control Register (STOPCR). Before these pins can be used to place the device out of STOP mode, they must be set for input using the P3 Port Input/Output Register (P3CR), P3Port Output Latch (P3DR), AD Control Register (ADCCR1).

STOP mode can be entered by setting up the System Control Register (SYSCR1), and can be released by detecting the active edge (rising or falling edge) on any STOP2 to STOP5 pins which are available for STOP mode release.

Note: When using Key-on Wakeup function, select level mode ( set SYSCR1<RELM> to "0" or "1" ) for selection of STOP mode release method.

Although P20 pin is shared with  $\overline{\text{INT5}}$  and  $\overline{\text{STOP}}$  pin input, use mainly  $\overline{\text{STOP}}$  pin to release STOP mode. This is because Key-on Wakeup function is comprised of  $\overline{\text{STOP}}$  pin and STOP2 to STOP5 pins as shown in the configuration diagram.

Note 1: When STOP mode release by an edge on STOP pin, follow one of the two methods described below.

- (1) Disable all of STOP2 to 5 pin inputs.
- (2) Fix STOP2 to 5 pin inputs high or low level.

Note 2: When using key-on wakeup (STOP2 to 5 pins) to exit STOP mode, make sure  $\overline{\text{STOP}}$  pin is held high and STOP2 to 5 pin inputs are held high or low level, because STOP mode release signal is created by ORing the  $\overline{\text{STOP}}$  pin input and the STOP2 to 5 pin input together.

### Key-on Wakeup STOP Mode Control Register

|         |       |       |       |       |   |   |   |   |                             |
|---------|-------|-------|-------|-------|---|---|---|---|-----------------------------|
| STOPCR  | 7     | 6     | 5     | 4     | 3 | 2 | 1 | 0 |                             |
| (0031H) | STOP5 | STOP4 | STOP3 | STOP2 |   |   |   |   | (Initial value : 0000 ****) |

|       |                                  |                         |            |
|-------|----------------------------------|-------------------------|------------|
| STOP2 | STOP mode release by P34 (STOP2) | 0: Disable<br>1: Enable | Write only |
| STOP3 | STOP mode release by P35 (STOP3) | 0: Disable<br>1: Enable |            |
| STOP4 | STOP mode release by P36 (STOP4) | 0: Disable<br>1: Enable |            |
| STOP5 | STOP mode release by P37 (STOP5) | 0: Disable<br>1: Enable |            |

|   |  |  |
|---|--|--|
| <Example of STOP mode release><br>The device is placed out of STOP mode in the following maner. |  |  |
|   | P20( $\overline{\text{STOP}}$ )                                | P3x                                      |
| STOP mode release using P3x (STOP2 to 5)  | Level detection mode: Low<br>Edge detection mode: Disable      | Edge detection<br>Rising or falling edge |
| STOP mode release using P20 ( $\overline{\text{STOP}}$ )  | Level detection mode: High<br>Edge detection mode: Rising edge | STOPCR: inhibited                        |

Note: Assertion of the STOP mode release signal is not recognized within three instruction cycles after executing the STOP instruction.

## 14. Flash Memory

TMP86FH09NG has 16384byte flash memory (address: C000H to FFFFH). The write and erase operations to the flash memory are controlled in the following three types of mode.

- MCU mode

The flash memory is accessed by the CPU control in the MCU mode. This mode is used for software bug correction and firmware change after shipment of the device since the write operation to the flash memory is available by retaining the application behavior.

- Serial PROM mode

The flash memory is accessed by the CPU control in the serial PROM mode. Use of the serial interface (UART) enables the flash memory to be controlled by the small number of pins. TMP86FH09NG in the serial PROM mode supports on-board programming which enables users to program flash memory after the microcontroller is mounted on a user board.

- Parallel PROM mode

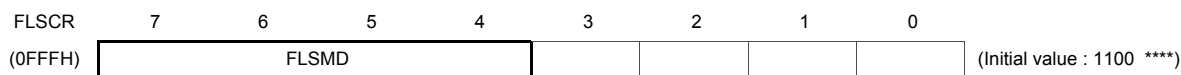
The parallel PROM mode allows the flash memory to be accessed as a stand-alone flash memory by the program writer provided by the third party. High-speed access to the flash memory is available by controlling address and data signals directly. For the support of the program writer, please ask Toshiba sales representative.

In the MCU and serial PROM modes, the flash memory control register (FLSCR) is used for flash memory control. This chapter describes how to access the flash memory using the flash memory control register (FLSCR) in the MCU and serial PROM modes.

## 14.1 Flash Memory Control

The flash memory is controlled via the flash memory control register (FLSCR) and flash memory standby control register (FLSSTB).

### Flash Memory Control Register



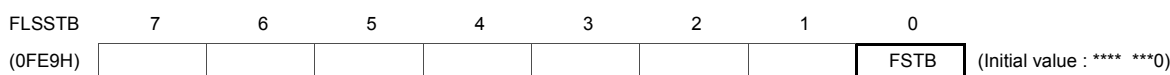
|       |   |   |     |
|-------|---|---|-----|
| FLSMD | Flash memory command sequence execution control | 1100: Disable command sequence execution<br>0011: Enable command sequence execution<br>Others: Reserved | R/W |
|-------|---|---|-----|

Note 1: The command sequence of the flash memory can be executed only when FLSMD="0011B". In other cases, any attempts to execute the command sequence are ineffective.

Note 2: FLSMD must be set to either "1100B" or "0011B".

Note 3: Bits 3 through 0 in FLSCR are always read as don't care.

### Flash Memory Standby Control Register



|      |                              |   |            |
|------|------------------------------|---|------------|
| FSTB | Flash memory standby control | 0: Disable the standby function.<br>1: Enable the standby function. | Write only |
|------|------------------------------|---|------------|

Note 1: When FSTB is set to 1, do not execute the read/write instruction to the flash memory because there is a possibility that the expected data is not read or the program is not operated correctly. If executing the read/write instruction, FSTB is initialized to 0 automatically.

Note 2: If an interrupt is issued when FSTB is set to 1, FSTB is initialized to 0 automatically and then the vector area of the flash memory is read.

Note 3: If the IDLE0/1/2, SLEEP0/1/2 or STOP mode is activated when FSTB is set to 1, FSTB is initialized to 0 automatically. In the IDLE0/1/2, SLEEP0/1/2 or STOP mode, the standby function operates regardless of FSTB setting.

#### 14.1.1 Flash Memory Command Sequence Execution Control (FLSCR<FLSMD>)

The flash memory can be protected from inadvertent write due to program error or microcontroller misoperation. This write protection feature is realized by disabling flash memory command sequence execution via the flash memory control register (write protect). To enable command sequence execution, set FLSCR<FLSMD> to "0011B". To disable command sequence execution, set FLSCR<FLSMD> to "1100B". After reset, FLSCR<FLSMD> is initialized to "1100B" to disable command sequence execution. Normally, FLSCR<FLSMD> should be set to "1100B" except when the flash memory needs to be written or erased.

#### 14.1.2 Flash Memory Standby Control (FLSSTB<FSTB>)

Low power consumption is enabled by cutting off the steady-state current of the flash memory. In the IDLE0/1/2, SLEEP0/1/2 or STOP mode, the steady-state current of the flash memory is cut off automatically.

When the program is executed in the RAM area (without accessing the flash memory) in the NORMAL 1/2 or SLOW1/2 mode, the current can be cut off by the control of the register. To cut off the steady-state current of the flash memory, set FLSSTB<FSTB> to "1" by the control program in the RAM area. The procedures for controlling the FLSSTB register are explained below.

(Steps 1 and 2 are controlled by the program in the flash memory, and steps 3 through 8 are controlled by the write control program executed in the RAM area.)

1. Transfer the control program of the FLSSTB register to the RAM area.
2. Jump to the RAM area.
3. Disable (DI) the interrupt master enable flag (IMF = "0").
4. Set FLSSTB<FSTB> to "1".
5. Execute the user program.
6. Repeat step 5 until the return request to the flash memory is detected.
7. Set FLSSTB<FSTB> to "0".
8. Jump to the flash memory area.

Note 1: The standby function is not operated by setting FLSSTB<FSTB> with the program in the flash memory. You must set FLSSTB<FSTB> by the program in the RAM area.

Note 2: To use the standby function by setting FLSSTB<FSTB> to "1" with the program in the RAM area, FLSSTB<FSTB> must be set to "0" by the program in the RAM area before returning the program control to the flash memory. If the program control is returned to the flash memory with FLSSTB<FSTB> set to "1", the program may misoperate and run out of control.

## 14.2 Command Sequence

The command sequence in the MCU and the serial PROM modes consists of six commands (JEDEC compatible), as shown in Table 14-1. Addresses specified in the command sequence are recognized with the lower 12 bits (excluding BA, SA, and FF7FH used for read protection). The upper 4 bits are used to specify the flash memory area,

Table 14-1 Command Sequence

|   | Command Sequence                | 1st Bus Write Cycle |      | 2nd Bus Write Cycle |      | 3rd Bus Write Cycle |      | 4th Bus Write Cycle |                  | 5th Bus Write Cycle |      | 6th Bus Write Cycle |      |
|---|---------------------------------|---------------------|------|---------------------|------|---------------------|------|---------------------|------------------|---------------------|------|---------------------|------|
|   |                                 | Address             | Data | Address             | Data | Address             | Data | Address             | Data             | Address             | Data | Address             | Data |
| 1 | Byte program                    | 555H                | AAH  | AAAH                | 55H  | 555H                | A0H  | BA<br>(Note 1)      | Data<br>(Note 1) | -                   | -    | -                   | -    |
| 2 | Sector Erase<br>(4-kbyte Erase) | 555H                | AAH  | AAAH                | 55H  | 555H                | 80H  | 555H                | AAH              | AAAH                | 55H  | SA<br>(Note 2)      | 30H  |
| 3 | Chip Erase<br>(All Erase)       | 555H                | AAH  | AAAH                | 55H  | 555H                | 80H  | 555H                | AAH              | AAAH                | 55H  | 555H                | 10H  |
| 4 | Product ID Entry                | 555H                | AAH  | AAAH                | 55H  | 555H                | 90H  | -                   | -                | -                   | -    | -                   | -    |
| 5 | Product ID Exit                 | XXH                 | F0H  | -                   | -    | -                   | -    | -                   | -                | -                   | -    | -                   | -    |
|   | Product ID Exit                 | 555H                | AAH  | AAAH                | 55H  | 555H                | F0H  | -                   | -                | -                   | -    | -                   | -    |
| 6 | Read Protect                    | 555H                | AAH  | AAAH                | 55H  | 555H                | A5H  | FF7FH               | 00H              | -                   | -    | -                   | -    |

Note 1: Set the address and data to be written.

Note 2: The area to be erased is specified with the upper 4 bits of the address.

### 14.2.1 Byte Program

This command writes the flash memory for each byte unit. The addresses and data to be written are specified in the 4th bus write cycle. Each byte can be programmed in a maximum of 40  $\mu$ s. The next command sequence cannot be executed until the write operation is completed. To check the completion of the write operation, perform read operations repeatedly until the same data is read twice from the same address in the flash memory. During the write operation, any consecutive attempts to read from the same address is reversed bit 6 of the data (toggling between 0 and 1).

Note: To rewrite data to Flash memory addresses at which data (including FFH) is already written, make sure to erase the existing data by "sector erase" or "chip erase" before rewriting data.

### 14.2.2 Sector Erase (4-kbyte Erase)

This command erases the flash memory in units of 4 kbytes. The flash memory area to be erased is specified by the upper 4 bits of the 6th bus write cycle address. For example, to erase 4 kbytes from F000H to FFFFH, specify one of the addresses in F000H-FFFFH as the 6th bus write cycle. The sector erase command is effective only in the MCU and serial PROM modes, and it cannot be used in the parallel PROM mode.

A maximum of 30 ms is required to erase 4 kbytes. The next command sequence cannot be executed until the erase operation is completed. To check the completion of the erase operation, perform read operations repeatedly for data polling until the same data is read twice from the same address in the flash memory. During the erase operation, any consecutive attempts to read from the same address is reversed bit 6 of the data (toggling between 0 and 1).

### 14.2.3 Chip Erase (All Erase)

This command erases the entire flash memory in approximately 30 ms. The next command sequence cannot be executed until the erase operation is completed. To check the completion of the erase operation, perform read operations repeatedly for data polling until the same data is read twice from the same address in the flash memory. During the erase operation, any consecutive attempts to read from the same address is reversed bit 6 of the data (toggling between 0 and 1). After the chip is erased, all bytes contain FFH.

### 14.2.4 Product ID Entry

This command activates the Product ID mode. In the Product ID mode, the vendor ID, the flash ID, and the read protection status can be read from the flash memory.

Table 14-2 Values To Be Read in the Product ID Mode

| Address | Meaning                | Read Value   |
|---------|------------------------|--|
| F000H   | Vendor ID              | 98H  |
| F001H   | Flash macro ID         | 41H  |
| F002H   | Flash size             | 0EH: 60 kbytes<br>0BH: 48 kbytes<br>07H: 32 kbytes<br>05H: 24 kbytes<br>03H: 16 kbytes<br>01H: 8 kbytes<br>00H: 4 kbytes |
| FF7FH   | Read protection status | FFH: Read protection disabled<br>Other than FFH: Read protection enabled   |

Note: The value at address F002H (flash size) depends on the size of flash memory incorporated in each product. For example, if the product has 60-kbyte flash memory, "0EH" is read from address F002H.

### 14.2.5 Product ID Exit

This command is used to exit the Product ID mode.

### 14.2.6 Read Protect

This command enables the read protection setting in the flash memory. When the read protection is enabled, the flash memory cannot be read in the parallel PROM mode. In the serial PROM mode, the flash write and RAM loader commands cannot be executed.

To disable the read protection setting, it is necessary to execute the chip erase command sequence. Whether or not the read protection is enabled can be checked by reading FF7FH in the Product ID mode. For details, see Table 14-2.

It takes a maximum of 40  $\mu$ s to set read protection in the flash memory. The next command sequence cannot be executed until this operation is completed. To check the completion of the read protect operation, perform read operations repeatedly for data polling until the same data is read twice from the same address in the flash memory. During the read protect operation, any attempts to read from the same address is reversed bit 6 of the data (toggling between 0 and 1).

### 14.3 Toggle Bit (D6)

After the byte program, chip erase, and read protect command sequence is executed, any consecutive attempts to read from the same address is reversed bit 6 (D6) of the data (toggling between 0 and 1) until the operation is completed. Therefore, this toggle bit provides a software mechanism to check the completion of each operation. Usually perform read operations repeatedly for data polling until the same data is read twice from the same address in the flash memory. After the byte program, chip erase, or read protect command sequence is executed, the initial read of the toggle bit always produces a "1".

## 14.4 Access to the Flash Memory Area

When the write, erase and read protections are set in the flash memory, read and fetch operations cannot be performed in the entire flash memory area. Therefore, to perform these operations in the entire flash memory area, access to the flash memory area by the control program in the BOOTROM or RAM area. (The flash memory program cannot write to the flash memory.) The serial PROM or MCU mode is used to run the control program in the BOOTROM or RAM area.

Note 1: The flash memory can be written or read for each byte unit. Erase operations can be performed either in the entire area or in units of 4 kbytes, whereas read operations can be performed by an one transfer instruction. However, the command sequence method is adopted for write and erase operations, requiring several-byte transfer instructions for each operation.

Note 2: To rewrite data to Flash memory addresses at which data (including FFH) is already written, make sure to erase the existing data by "sector erase" or "chip erase" before rewriting data.

### 14.4.1 Flash Memory Control in the Serial PROM Mode

The serial PROM mode is used to access to the flash memory by the control program provided in the BOOTROM area. Since almost of all operations relating to access to the flash memory can be controlled simply by the communication data of the serial interface (UART), these functions are transparent to the user. For the details of the serial PROM mode, see "Serial PROM Mode."

To access to the flash memory by using peripheral functions in the serial PROM mode, run the RAM loader command to execute the control program in the RAM area. The procedures to execute the control program in the RAM area is shown in "14.4.1.1 How to write to the flash memory by executing the control program in the RAM area (in the RAM loader mode within the serial PROM mode)".

#### 14.4.1.1 How to write to the flash memory by executing the control program in the RAM area (in the RAM loader mode within the serial PROM mode)

(Steps 1 and 2 are controlled by the BOOTROM, and steps 3 through 9 are controlled by the control program executed in the RAM area.)

1. Transfer the write control program to the RAM area in the RAM loader mode.
2. Jump to the RAM area.
3. Disable (DI) the interrupt master enable flag (IMF←"0").
4. Set FLSCR<FLSMD> to "0011B" (to enable command sequence execution).
5. Execute the erase command sequence.
6. Read the same flash memory address twice.  
(Repeat step 6 until the same data is read by two consecutive reads operations.)
7. Execute the write command sequence.
8. Read the same flash memory address twice.  
(Repeat step 8 until the same data is read by two consecutive reads operations.)
9. Set FLSCR<FLSMD> to "1100B" (to disable command sequence execution).

Note 1: Before writing to the flash memory in the RAM area, disable interrupts by setting the interrupt master enable flag (IMF) to "0". Usually disable interrupts by executing the DI instruction at the head of the write control program in the RAM area.

Note 2: Since the watchdog timer is disabled by the BOOTROM in the RAM loader mode, it is not required to disable the watchdog timer by the RAM loader program.

Example :After chip erasure, the program in the RAM area writes data 3FH to address F000H.

```
DI                                     : Disable interrupts (IMF←"0")
LD      (FLSCR),0011_1000B           : Enable command sequence execution.
LD      IX,0F555H
LD      IY,0FAAAH
LD      HL,0F000H

; #### Flash Memory Chip erase Process ####
LD      (IX),0AAH                    : 1st bus write cycle
LD      (IY),55H                     : 2nd bus write cycle
LD      (IX),80H                     : 3rd bus write cycle
LD      (IX),0AAH                    : 4th bus write cycle
LD      (IY),55H                     : 5th bus write cycle
LD      (IX),10H                     : 6th bus write cycle
sLOOP1: LD      W,(IX)
      CMP      W,(IX)
      JR      NZ,sLOOP1              : Loop until the same value is read.

; #### Flash Memory Write Process ####
LD      (IX),0AAH                    : 1st bus write cycle
LD      (IY),55H                     : 2nd bus write cycle
LD      (IX),0A0H                    : 3rd bus write cycle
LD      (HL),3FH                     : 4th bus write cycle, (F000H)=3FH
sLOOP2: LD      W,(HL)
      CMP      W,(HL)
      JR      NZ,sLOOP2              : Loop until the same value is read.
LD      (FLSCR),1100_1000B           : Disable command sequence execution.
sLOOP3: JP      sLOOP3
```

## 14.4.2 Flash Memory Control in the MCU mode

In the MCU mode, write operations are performed by executing the control program in the RAM area. Before execution of the control program, copy the control program into the RAM area or obtain it from the external using the communication pin. The procedures to execute the control program in the RAM area in the MCU mode are described below.

### 14.4.2.1 How to write to the flash memory by executing a user write control program in the RAM area (in the MCU mode)

(Steps 1 and 2 are controlled by the program in the flash memory, and steps 3 through 11 are controlled by the control program in the RAM area.)

1. Transfer the write control program to the RAM area.
2. Jump to the RAM area.
3. Disable (DI) the interrupt master enable flag (IMF←"0").
4. Disable the watchdog timer, if it is used.
5. Set FLSCR<FLSMD> to "0011B" (to enable command sequence execution).
6. Execute the erase command sequence.
7. Read the same flash memory address twice.  
(Repeat step 7 until the same data is read by two consecutive read operations.)
8. Execute the write command sequence.
9. Read the same flash memory address twice.  
(Repeat step 9 until the same data is read by two consecutive read operations.)
10. Set FLSCR<FLSMD> to "1100B" (to disable command sequence execution).
11. Jump to the flash memory area.

Note 1: Before writing to the flash memory in the RAM area, disable interrupts by setting the interrupt master enable flag (IMF) to "0". Usually disable interrupts by executing the DI instruction at the head of the write control program in the RAM area.

Note 2: When writing to the flash memory, do not intentionally use non-maskable interrupts (the watchdog timer must be disabled if it is used). If a non-maskable interrupt occurs while the flash memory is being written, unexpected data is read from the flash memory (interrupt vector), resulting in malfunction of the microcontroller.

Example :After sector erasure (E000H-EFFFH), the program in the RAM area writes data 3FH to address E000H.

```
DI                : Disable interrupts (IMF←"0")
LD      (WDTCR2),4EH  : Clear the WDT binary counter.
LDW     (WDTCR1),0B101H : Disable the WDT.
LD      (FLSCR),0011_1000B : Enable command sequence execution.
LD      IX,0F555H
LD      IY,0FAAAH
LD      HL,0E000H

; ##### Flash Memory Sector Erase Process #####
LD      (IX),0AAH      : 1st bus write cycle
LD      (IY),55H      : 2nd bus write cycle
LD      (IX),80H      : 3rd bus write cycle
LD      (IX),0AAH     : 4th bus write cycle
LD      (IY),55H      : 5th bus write cycle
LD      (HL),30H      : 6th bus write cycle
sLOOP1: LD      W,(IX)
        CMP     W,(IX)
        JR      NZ,sLOOP1      : Loop until the same value is read.

; ##### Flash Memory Write Process #####
LD      (IX),0AAH     : 1st bus write cycle
LD      (IY),55H     : 2nd bus write cycle
LD      (IX),0A0H    : 3rd bus write cycle
LD      (HL),3FH     : 4th bus write cycle, (1000H)=3FH
sLOOP2: LD      W,(HL)
        CMP     W,(HL)
        JR      NZ,sLOOP2     : Loop until the same value is read.
LD      (FLSCR),1100_1000B : Disable command sequence execution.
JP      XXXXH       : Jump to the flash memory area.
```

Example :This write control program reads data from address F000H and stores it to 98H in the RAM area.

```
LD      A,(0F000H)   : Read data from address F000H.
LD      (98H),A     : Store data to address 98H.
```

# 15. Serial PROM Mode

## 15.1 Outline

The TMP86FH09NG has a 2048 byte BOOTROM (Mask ROM) for programming to flash memory. The BOOTROM is available in the serial PROM mode, and controlled by TEST, BOOT and RESET pins. Communication is performed via UART. The serial PROM mode has seven types of operating mode: Flash memory writing, RAM loader, Flash memory SUM output, Product ID code output, Flash memory status output, Flash memory erasing and Flash memory read protection setting. Memory address mapping in the serial PROM mode differs from that in the MCU mode. Figure 15-1 shows memory address mapping in the serial PROM mode.

Table 15-1 Operating Range in the Serial PROM Mode

| Parameter             | Min  | Max  | Unit |
|-----------------------|------|------|------|
| Power supply          | 4.75 | 5.25 | V    |
| High frequency (Note) | 2    | 16   | MHz  |

Note: Though included in above operating range, some of high frequencies are not supported in the serial PROM mode. For details, refer to "Table 15-5".

## 15.2 Memory Mapping

The Figure 15-1 shows memory mapping in the Serial PROM mode and MCU mode.

In the serial PROM mode, the BOOTROM (Mask ROM) is mapped in addresses from 7800H to 7FFFH.

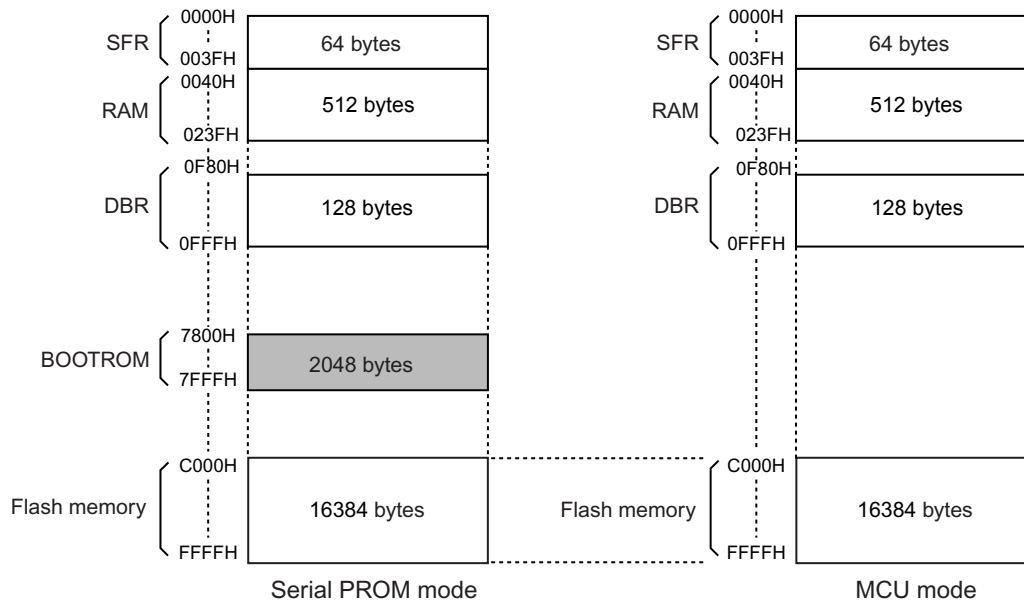



Figure 15-1 Memory Address Maps

## 15.3 Serial PROM Mode Setting

### 15.3.1 Serial PROM Mode Control Pins

To execute on-board programming, activate the serial PROM mode. Table 15-2 shows pin setting to activate the serial PROM mode.

Table 15-2 Serial PROM Mode Setting

| Pin                           | Setting   |
|-------------------------------|---|
| TEST pin                      | High  |
| BOOT/RXD pin                  | High  |
| $\overline{\text{RESET}}$ pin |  |

Note: The BOOT pin is shared with the UART communication pin (RXD pin) in the serial PROM mode. This pin is used as UART communication pin after activating serial PROM mode

### 15.3.2 Pin Function

In the serial PROM mode, TXD (P03) and RXD (P02) are used as a serial interface pin.

Table 15-3 Pin Function in the Serial PROM Mode

| Pin Name<br>(Serial PROM Mode) | Input/<br>Output | Function  | Pin Name<br>(MCU Mode)    |
|--------------------------------|------------------|---|---------------------------|
| TXD                            | Output           | Serial data output  | P03                       |
| BOOT/RXD                       | Input/Input      | Serial PROM mode control/Serial data input  | P02                       |
| $\overline{\text{RESET}}$      | Input            | Serial PROM mode control  | $\overline{\text{RESET}}$ |
| TEST                           | Input            | Fixed to high   | TEST                      |
| VDD                            | Power supply     | 4.75 to 5.25 V  |                           |
| VSS                            | Power supply     | 0 V   |                           |
| I/O ports except<br>P03, P02   | I/O              | These ports are in the high-impedance state in the serial PROM mode. The input level is fixed to the port inputs with a hardware feature to prevent overlap current. (The port inputs are invalid.) To make the port inputs valid, set the pin of the SPCR register to "1" by the RAM loader control program. |                           |
| XIN                            | Input            | Self-oscillate with an oscillator.  | (Note 2)                  |
| XOUT                           | Output           |   |                           |

Note 1: During on-board programming with other parts mounted on a user board, be careful no to affect these communication control pins.

Note 2: Operating range of high frequency in serial PROM mode is 2 MHz to 16 MHz.

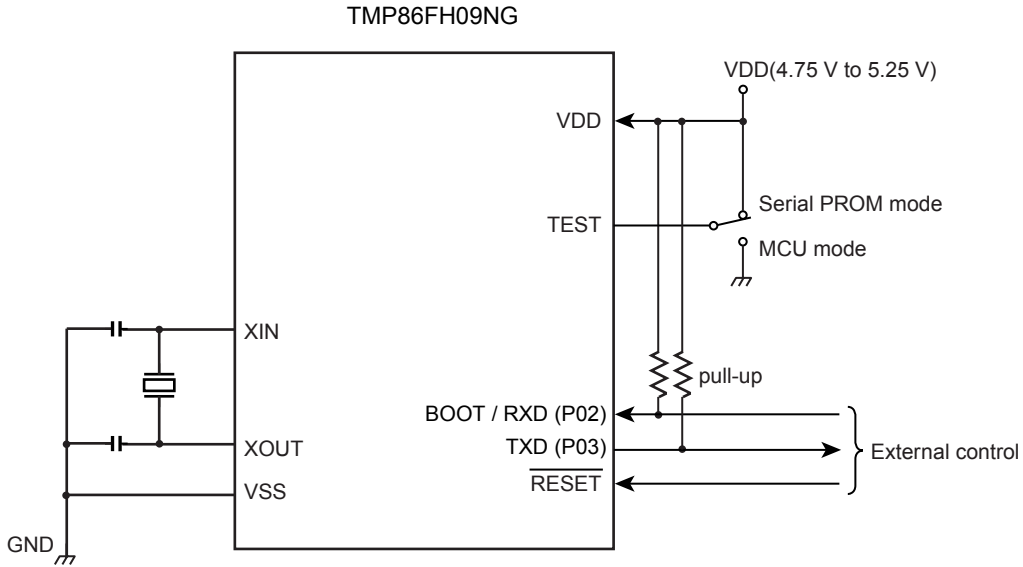


Figure 15-2 Serial PROM Mode Pin Setting

Note 1: For connection of other pins, refer to " Table 15-3 Pin Function in the Serial PROM Mode ".

### 15.3.3 Example Connection for On-Board Writing

Figure 15-3 shows an example connection to perform on-board writing.

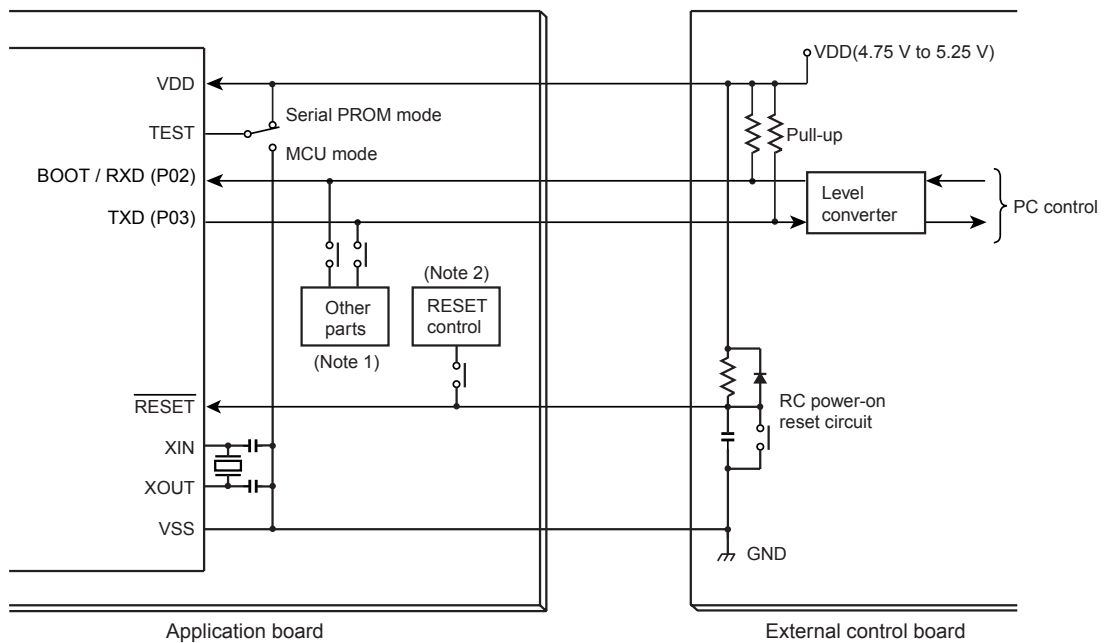


Figure 15-3 Example Connection for On-Board Writing

Note 1: When other parts on the application board effect the UART communication in the serial PROM mode, isolate these pins by a jumper or switch.

Note 2: When the reset control circuit on the application board effects activation of the serial PROM mode, isolate the pin by a jumper or switch.

Note 3: For connection of other pins, refer to " Table 15-3 Pin Function in the Serial PROM Mode ".

### 15.3.4 Activating the Serial PROM Mode

The following is a procedure to activate the serial PROM mode. " Figure 15-4 Serial PROM Mode Timing " shows a serial PROM mode timing.

1. Supply power to the VDD pin.
2. Set the  $\overline{\text{RESET}}$  pin to low.
3. Set the TEST pin and BOOT/RXD pins to high.
4. Wait until the power supply and clock oscillation stabilize.
5. Set the  $\overline{\text{RESET}}$  pin to high.
6. Input the matching data (5AH) to the BOOT/RXD pin after setup sequence. For details of the setup timing, refer to " 15.16 UART Timing ".

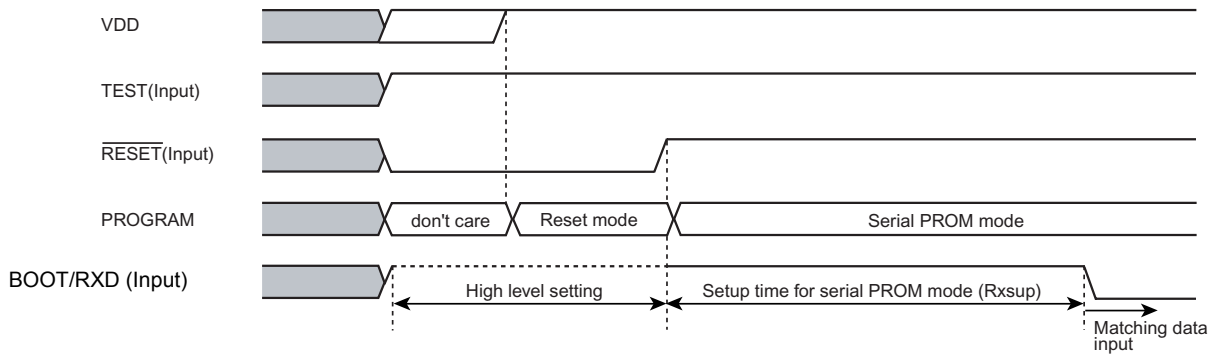


Figure 15-4 Serial PROM Mode Timing

## 15.4 Interface Specifications for UART

The following shows the UART communication format used in the serial PROM mode.

To perform on-board programming, the communication format of the write controller must also be set in the same manner.

The default baud rate is 9600 bps regardless of operating frequency of the microcontroller. The baud rate can be modified by transmitting the baud rate modification data shown in Table 1-4 to TMP86FH09NG. The Table 15-5 shows an operating frequency and baud rate. The frequencies which are not described in Table 15-5 can not be used.

- Baud rate (Default): 9600 bps
- Data length: 8 bits
- Parity addition: None
- Stop bit: 1 bit

Table 15-4 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 15-5 Operating Frequency and Baud Rate in the 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)        | Rating (MHz)   | Baud rate (bps) | (%)   | (bps) | (%)   | (bps) | (%)   | (bps) | (%)   | (bps) | (%)   | (bps) | (%)   | (bps) | (%)   |       |
| 1        | 2                           | 1.91 to 2.10   | -               | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | 9615  | +0.16 |
| 2        | 4                           | 3.82 to 4.19   | -               | -     | -     | -     | -     | -     | -     | -     | 31250 | 0.00  | 19231 | +0.16 | 9615  | +0.16 |       |
|          | 4.19                        | 3.82 to 4.19   | -               | -     | -     | -     | -     | -     | -     | -     | 32734 | +4.75 | 20144 | +4.92 | 10072 | +4.92 |       |
| 3        | 4.9152                      | 4.70 to 5.16   | -               | -     | -     | -     | -     | -     | 38400 | 0.00  | -     | -     | 19200 | 0.00  | 9600  | 0.00  |       |
|          | 5                           | 4.70 to 5.16   | -               | -     | -     | -     | -     | -     | 39063 | +1.73 | -     | -     | 19531 | +1.73 | 9766  | +1.73 |       |
| 4        | 6                           | 5.87 to 6.45   | -               | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | 9375  | -2.34 |       |
|          | 6.144                       | 5.87 to 6.45   | -               | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | -     | 9600  | 0.00  |       |
| 5        | 7.3728                      | 7.05 to 7.74   | -               | -     | -     | -     | 57600 | 0.00  | -     | -     | -     | -     | 19200 | 0.00  | 9600  | 0.00  |       |
| 6        | 8                           | 7.64 to 8.39   | -               | -     | 62500 | 0.00  | -     | -     | 38462 | +0.16 | 31250 | 0.00  | 19231 | +0.16 | 9615  | +0.16 |       |
| 7        | 9.8304                      | 9.40 to 10.32  | 76800           | 0.00  | -     | -     | -     | -     | 38400 | 0.00  | -     | -     | 19200 | 0.00  | 9600  | 0.00  |       |
|          | 10                          | 9.40 to 10.32  | 78125           | +1.73 | -     | -     | -     | -     | 39063 | +1.73 | -     | -     | 19531 | +1.73 | 9766  | +1.73 |       |
| 8        | 12                          | 11.75 to 12.90 | -               | -     | -     | -     | 57692 | +0.16 | -     | -     | 31250 | 0.00  | 18750 | -2.34 | 9375  | -2.34 |       |
|          | 12.288                      | 11.75 to 12.90 | -               | -     | -     | -     | 59077 | +2.56 | -     | -     | 32000 | +2.40 | 19200 | 0.00  | 9600  | 0.00  |       |
|          | 12.5                        | 11.75 to 12.90 | -               | -     | 60096 | -3.85 | 60096 | +4.33 | -     | -     | 30048 | -3.85 | 19531 | +1.73 | 9766  | +1.73 |       |
| 9        | 14.7456                     | 14.10 to 15.48 | -               | -     | -     | -     | 57600 | 0.00  | 38400 | 0.00  | -     | -     | 19200 | 0.00  | 9600  | 0.00  |       |
| 10       | 16                          | 15.27 to 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 "Rating" show frequencies available in the serial PROM mode. Though the frequency is supported in the serial PROM mode, the serial PROM mode may not be activated correctly due to the frequency difference in the external controller (such as personal computer) and oscillator, and load capacitance of communication pins.

Note 2: It is recommended that the total frequency difference is within  $\pm 3\%$  so that auto detection is performed correctly by the reference frequency.

Note 3: The external controller must transmit the matching data (5AH) repeatedly till the auto detection of baud rate is performed. This number indicates the number of times the matching data is transmitted for each frequency.

## 15.5 Operation Command

The eight commands shown in Table 15-6 are used in the serial PROM mode. After reset release, the TMP86FH09NG waits for the matching data (5AH).

Table 15-6 Operation Command in the Serial PROM Mode

| Command Data | Operating Mode                       | Description   |
|--------------|--------------------------------------|---|
| 5AH          | Setup                                | Matching data. Execute this command after releasing the reset.  |
| F0H          | Flash memory erasing                 | Erases the flash memory area (address C000H to FFFFH).  |
| 30H          | Flash memory writing                 | Writes to the flash memory area (address C000H to FFFFH).   |
| 60H          | RAM loader                           | Writes to the specified RAM area (address 0050H to 023FH).  |
| 90H          | Flash memory SUM output              | Outputs the 2-byte checksum upper byte and lower byte in this order for the entire area of the flash memory (address C000H to FFFFH). |
| C0H          | Product ID code output               | Outputs the product ID code (13-byte data).   |
| C3H          | Flash memory status output           | Outputs the status code (7-byte data) such as the read protection condition.  |
| FAH          | Flash memory read protection setting | Enables the read protection.  |

## 15.6 Operation Mode

The serial PROM mode has seven types of modes, that are (1) Flash memory erasing, (2) Flash memory writing, (3) RAM loader, (4) Flash memory SUM output, (5) Product ID code output, (6) Flash memory status output and (7) Flash memory read protection setting modes. Description of each mode is shown below.

### 1. Flash memory erasing mode

The flash memory is erased by the chip erase (erasing an entire flash area) or sector erase (erasing sectors in 4-kbyte units). The erased area is filled with FFH. When the read protection is enabled, the sector erase in the flash erasing mode can not be performed. To disable the read protection, perform the chip erase. Before erasing the flash memory, TMP86FH09NG checks the passwords except a blank product. If the password is not matched, the flash memory erasing mode is not activated.

### 2. Flash memory writing mode

Data is written to the specified flash memory address for each byte unit. The external controller must transmit the write data in the Intel Hex format (Binary). If no error is encountered till the end record, TMP86FH09NG calculates the checksum for the entire flash memory area (C000H to FFFFH), and returns the obtained result to the external controller. When the read protection is enabled, the flash memory writing mode is not activated. In this case, perform the chip erase command beforehand in the flash memory erasing mode. Before activating the flash memory writing mode, TMP86FH09NG checks the password except a blank product. If the password is not matched, flash memory writing mode is not activated.

### 3. RAM loader mode

The RAM loader transfers the data in Intel Hex format sent from the external controller to the internal RAM. When the transfer is completed normally, the RAM loader calculates the checksum. After transmitting the results, the RAM loader jumps to the RAM address specified with the first data record in order to execute the user program. When the read protection is enabled, the RAM loader mode is not activated. In this case, perform the chip erase beforehand in the flash memory erasing mode. Before activating the RAM loader mode, TMP86FH09NG checks the password except a blank product. If the password is not matched, flash RAM loader mode is not activated.

### 4. Flash memory SUM output mode

The checksum is calculated for the entire flash memory area (C000H to FFFFH), and the result is returned to the external controller. Since the BOOTROM does not support the operation command to read the flash memory, use this checksum to identify programs when managing revisions of application programs.

### 5. Product ID code output

The code used to identify the product is output. The code to be output consists of 13-byte data, which includes the information indicating the area of the ROM incorporated in the product. The external controller reads this code, and recognizes the product to write.

(In the case of TMP86FH09NG, the addresses from C000H to FFFFH become the ROM area.)

6. Flash memory status output mode  
The status of the area from FFE0H to FFFFH, and the read protection condition are output as 7-byte code. The external controller reads this code to recognize the flash memory status.
7. Flash memory read protection setting mode  
This mode disables reading the flash memory data in parallel PROM mode. In the serial PROM mode, the flash memory writing and RAM loader modes are disabled. To disable the flash memory read protection, perform the chip erase in the flash memory erasing mode.

15.6.1 Flash Memory Erasing Mode (Operating command: F0H)

Table 15-7 shows the flash memory erasing mode.

Table 15-7 Flash Memory Erasing Mode

|          | Transfer Byte               | Transfer Data from the External Controller to TMP86FH09NG  | Baud Rate                                | Transfer Data from TMP86FH09NG to the External Controller                                    |
|----------|-----------------------------|--|--|--|
| BOOT ROM | 1st byte<br>2nd byte        | Matching data (5AH)<br>-                                   | 9600 bps<br>9600 bps                     | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: No data transmitted |
|          | 3rd byte<br>4th byte        | Baud rate change data (Table 15-4)<br>-                    | 9600 bps<br>9600 bps                     | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)                         |
|          | 5th byte<br>6th byte        | Operation command data (F0H)<br>-                          | Modified baud rate<br>Modified baud rate | -<br>OK: Echo back data (F0H)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)                   |
|          | 7th byte<br>8th byte        | Password count storage address bit 15 to 08 (Note 4, 5)    | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 9th byte<br>10th byte       | Password count storage address bit 07 to 00 (Note 4, 5)    | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 11th byte<br>12th byte      | Password comparison start address bit 15 to 08 (Note 4, 5) | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 13th byte<br>14th byte      | Password comparison start address bit 07 to 00 (Note 4, 5) | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 15th byte<br>:<br>m'th byte | Password string (Note 4, 5)<br>-                           | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | n'th - 2 byte               | Erase area specification (Note 2)                          | Modified baud rate                       | -  |
|          | n'th - 1 byte               | -  | Modified baud rate                       | OK: Checksum (Upper byte) (Note 3)<br>Error: Nothing transmitted                             |
|          | n'th byte                   | -  | Modified baud rate                       | OK: Checksum (Lower byte) (Note 3)<br>Error: Nothing transmitted                             |
|          | n'th + 1 byte               | (Wait for the next operation command data)                 | Modified baud rate                       | -  |

Note 1: "xxH × 3" indicates that the device enters the halt condition after transmitting 3 bytes of xxh.

Note 2: Refer to " 15.13 Specifying the Erasure Area ".

Note 3: Refer to " 15.8 Checksum (SUM) ".

Note 4: Refer to " 15.10 Passwords ".

Note 5: Do not transmit the password string for a blank product.

Note 6: When a password error occurs, TMP86FH09NG stops UART communication and enters the halt mode. Therefore, when a password error occurs, initialize TMP86FH09NG by the RESET pin and reactivate the serial PROM mode.

Note 7: If an error occurs during transfer of a password address or a password string, TMP86FH09NG stops UART communication and enters the halt condition. Therefore, when a password error occurs, initialize TMP86FH09NG by the RESET pin and reactivate the serial PROM mode.

Description of the flash memory erasing mode

1. The 1st through 4th bytes of the transmitted and received data contain the same data as in the flash memory writing mode.

2. The 5th byte of the received data contains the command data in the flash memory erasing mode (F0H).
3. When the 5th byte of the received data contains the operation command data shown in Table 15-6, the device echoes back the value which is the same data in the 6th byte position of the received data (in this case, F0H). If the 5th byte of the received data does not contain the operation command data, the device enters the halt condition after sending 3 bytes of the operation command error code (63H).
4. The 7th through m'th bytes of the transmitted and received data contain the same data as in the flash memory writing mode. In the case of a blank product, do not transmit a password string. (Do not transmit a dummy password string.)
5. The n'th - 2 byte contains the erasure area specification data. The upper 4 bits and lower 4 bits specify the start address and end address of the erasure area, respectively. For the detailed description, see "1.13 Specifying the Erasure Area".
6. The n'th - 1 byte and n'th byte contain the upper and lower bytes of the checksum, respectively. For how to calculate the checksum, refer to "1.8 Checksum (SUM)". Checksum is calculated unless a receiving error or Intel Hex format error occurs. After sending the end record, the external controller judges whether the transmission is completed correctly by receiving the checksum sent by the device.
7. After sending the checksum, the device waits for the next operation command data.

15.6.2 Flash Memory Writing Mode (Operation command: 30H)

Table 15-8 shows flash memory writing mode process.

Table 15-8 Flash Memory Writing Mode Process

|          | Transfer Byte                       | Transfer Data from External Controller to TMP86FH09NG   | Baud Rate                                | Transfer Data from TMP86FH09NG to External Controller  |
|----------|-------------------------------------|---|--|--|
| BOOT ROM | 1st byte<br>2nd byte                | Matching data (5Ah)<br>-                                | 9600 bps<br>9600 bps                     | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: Nothing transmitted |
|          | 3rd byte<br>4th byte                | Baud rate modification data (See Table 15-4)<br>-       | 9600 bps<br>9600 bps                     | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)                         |
|          | 5th byte<br>6th byte                | Operation command data (30H)<br>-                       | Modified baud rate<br>Modified baud rate | -<br>OK: Echo back data (30H)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)                   |
|          | 7th byte<br>8th byte                | Password count storage address bit 15 to 08 (Note 4)    | Modified baud rate                       | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 9th byte<br>10th byte               | Password count storage address bit 07 to 00 (Note 4)    | Modified baud rate                       | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 11th byte<br>12th byte              | Password comparison start address bit 15 to 08 (Note 4) | Modified baud rate                       | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 13th byte<br>14th byte              | Password comparison start address bit 07 to 00 (Note 4) | Modified baud rate                       | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 15th byte<br>:<br>m'th byte         | Password string (Note 5)<br>-                           | Modified baud rate                       | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | m'th + 1 byte<br>:<br>n'th - 2 byte | Intel Hex format (binary) (Note 2)                      | Modified baud rate                       | -<br>-   |
|          | n'th - 1 byte                       | -   | Modified baud rate                       | OK: SUM (Upper byte) (Note 3)<br>Error: Nothing transmitted                                  |
|          | n'th byte                           | -   | Modified baud rate                       | OK: SUM (Lower byte) (Note 3)<br>Error: Nothing transmitted                                  |
|          | n'th + 1 byte                       | (Wait state for the next operation command data)        | Modified baud rate                       | -  |

Note 1: "xxH × 3" indicates that the device enters the halt condition after sending 3 bytes of xxH. For details, refer to " 15.7 Error Code ".

Note 2: Refer to " 15.9 Intel Hex Format (Binary) ".

Note 3: Refer to " 15.8 Checksum (SUM) ".

Note 4: Refer to " 15.10 Passwords ".

Note 5: If addresses from FFE0H to FFFFH are filled with "FFH", the passwords are not compared because the device is considered as a blank product. Transmitting a password string is not required. Even in the case of a blank product, it is required to specify the password count storage address and the password comparison start address. Transmit these data from the external controller. If a password error occurs due to incorrect password count storage address or password comparison start address, TMP86FH09NG stops UART communication and enters the halt condition. Therefore, when a password error occurs, initialize TMP86FH09NG by the  $\overline{\text{RESET}}$  pin and reactivate the serial ROM mode.

Note 6: If the read protection is enabled or a password error occurs, TMP86FH09NG stops UART communication and enters the halt condition. In this case, initialize TMP86FH09NG by the  $\overline{\text{RESET}}$  pin and reactivate the serial ROM mode.

Note 7: If an error occurs during the reception of a password address or a password string, TMP86FH09NG stops UART communication and enters the halt condition. In this case, initialize TMP86FH09NG by the  $\overline{\text{RESET}}$  pin and reactivate the serial PROM mode.

Description of the flash memory writing mode

1. The 1st byte of the received data contains the matching data. When the serial PROM mode is activated, TMP86FH09NG (hereafter called device), waits to receive the matching data (5AH). Upon reception of the matching data, the device automatically adjusts the UART's initial baud rate to 9600 bps.
2. When receiving the matching data (5AH), the device transmits an echo back data (5AH) as the second byte data to the external controller. If the device can not recognize the matching data, it does not transmit the echo back data and waits for the matching data again with automatic baud rate adjustment. Therefore, the external controller should transmit the matching data repeatedly till the device transmits an echo back data. The transmission repetition count varies depending on the frequency of device. For details, refer to Table 15-5.
3. The 3rd byte of the received data contains the baud rate modification data. The five types of baud rate modification data shown in Table 15-4 are available. Even if baud rate is not modified, the external controller should transmit the initial baud rate data (28H: 9600 bps).
4. Only when the 3rd byte of the received data contains the baud rate modification data corresponding to the device's operating frequency, the device echoes back data the value which is the same data in the 4th byte position of the received data. After the echo back data is transmitted, baud rate modification becomes effective. If the 3rd byte of the received data does not contain the baud rate modification data, the device enters the halts condition after sending 3 bytes of baud rate modification error code (62H).
5. The 5th byte of the received data contains the command data (30H) to write the flash memory.
6. When the 5th byte of the received data contains the operation command data shown in Table 1-6, the device echoes back the value which is the same data in the 6th byte position of the received data (in this case, 30H). If the 5th byte of the received data does not contain the operation command data, the device enters the halt condition after sending 3 bytes of the operation command error code (63H).
7. The 7th byte contains the data for 15 to 8 bits of the password count storage address. When the data received with the 7th byte has no receiving error, the device does not send any data. If a receiving error or password error occurs, the device does not send any data and enters the halt condition.
8. The 9th byte contains the data for 7 to 0 bits of the password count storage address. When the data received with the 9th byte has no receiving error, the device does not send any data. If a receiving error or password error occurs, the device does not send any data and enters the halt condition.
9. The 11th byte contains the data for 15 to 8 bits of the password comparison start address. When the data received with the 11th byte has no receiving error, the device does not send any data. If a receiving error or password error occurs, the device does not send any data and enters the halt condition.
10. The 13th byte contains the data for 7 to 0 bits of the password comparison start address. When the data received with the 13th byte has no receiving error, the device does not send any data. If a receiving error or password error occurs, the device does not send any data and enters the halt condition.
11. The 15th through m'th bytes contain the password data. The number of passwords becomes the data (N) stored in the password count storage address. The external password data is compared with N-byte data from the address specified by the password comparison start address. The external controller should send N-byte password data to the device. If the passwords do not match, the device enters the halt condition without returning an error code to the external controller. If the addresses from FFE0H to FFFFH are filled with "FFH", the passwords are not compared because the device is considered as a blank product.
12. The m'th + 1 through n'th - 2 bytes of the received data contain the binary data in the Intel Hex format. No received data is echoed back to the external controller. After receiving the start mark (3AH for ":" in the Intel Hex format), the device starts data record reception. Therefore, the received data except 3AH is ignored until the start mark is received. After receiving the start mark, the device receives the data record, that consists of data length, address, record type, write data and checksum. Since the device starts checksum calculation after receiving an end record, the external controller should wait for the checksum after sending the end record. If a receiving error or Intel Hex format error occurs, the device enters the halts condition without returning an error code to the external controller.
13. The n'th - 1 and n'th bytes contain the checksum upper and lower bytes. For details on how to calculate the SUM, refer to " 15.8 Checksum (SUM) ". The checksum is calculated only when the end record is detected and no receiving error or Intel Hex format error occurs. After sending the end

record, the external controller judges whether the transmission is completed correctly by receiving the checksum sent by the device.

14. After transmitting the checksum, the device waits for the next operation command data.

Note 1: Do not write only the address from FFE0H to FFFFH when all flash memory data is the same. If only these area are written, the subsequent operation can not be executed due to password error.

Note 2: To rewrite data to Flash memory addresses at which data (including FFH) is already written, make sure to erase the existing data by "sector erase" or "chip erase" before rewriting data.

### 15.6.3 RAM Loader Mode (Operation Command: 60H)

Table 15-9 shows RAM loader mode process.

Table 15-9 RAM Loader Mode Process

|          | Transfer Bytes                      | Transfer Data from External Controller to TMP86FH09NG   | Baud Rate   | Transfer Data from TMP86FH09NG to External Controller  |
|----------|-------------------------------------|---|---|--|
| BOOT ROM | 1st byte<br>2nd byte                | Matching data (5AH)<br>-                                | 9600 bps<br>9600 bps  | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: Nothing transmitted |
|          | 3rd byte<br>4th byte                | Baud rate modification data (See Table 15-4)<br>-       | 9600 bps<br>9600 bps  | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)                         |
|          | 5th byte<br>6th byte                | Operation command data (60H)<br>-                       | Modified baud rate<br>Modified baud rate  | -<br>OK: Echo back data (60H)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)                   |
|          | 7th byte<br>8th byte                | Password count storage address bit 15 to 08 (Note 4)    | Modified baud rate  | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 9th byte<br>10th byte               | Password count storage address bit 07 to 00 (Note 4)    | Modified baud rate  | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 11th byte<br>12th byte              | Password comparison start address bit 15 to 08 (Note 4) | Modified baud rate  | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 13th byte<br>14th byte              | Password comparison start address bit 07 to 00 (Note 4) | Modified baud rate  | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 15th byte<br>:<br>m'th byte         | Password string (Note 5)<br>-                           | Modified baud rate  | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | m'th + 1 byte<br>:<br>n'th - 2 byte | Intel Hex format (Binary) (Note 2)                      | Modified baud rate<br>Modified baud rate  | -<br>-   |
|          | n'th - 1 byte                       | -   | Modified baud rate  | OK: SUM (Upper byte) (Note 3)<br>Error: Nothing transmitted                                  |
|          | n'th byte                           | -   | Modified baud rate  | OK: SUM (Lower byte) (Note 3)<br>Error: Nothing transmitted                                  |
|          | RAM                                 | -   | The program jumps to the start address of RAM in which the first transferred data is written. |  |

Note 1: "xxH × 3" indicates that the device enters the halt condition after sending 3 bytes of xxH. For details, refer to " 15.7 Error Code ".

Note 2: Refer to " 15.9 Intel Hex Format (Binary) ".

Note 3: Refer to " 15.8 Checksum (SUM) ".

Note 4: Refer to " 15.10 Passwords ".

Note 5: If addresses from FFE0H to FFFFH are filled with "FFH", the passwords are not compared because the device is considered as a blank product. Transmitting a password string is not required. Even in the case of a blank product, it is required to specify the password count storage address and the password comparison start address. Transmit these data from the external controller. If a password error occurs due to incorrect password count storage address or password comparison start address, TMP86FH09NG stops UART communication and enters the halt condition. Therefore, when a password error occurs, initialize TMP86FH09NG by the  $\overline{\text{RESET}}$  pin and reactivate the serial ROM mode.

Note 6: After transmitting a password string, the external controller must not transmit only an end record. If receiving an end record after a password string, the device may not operate correctly.

Note 7: If the read protection is enabled or a password error occurs, TMP86FH09NG stops UART communication and enters the halt condition. In this case, initialize TMP86FH09NG by the RESET pin and reactivate the serial PROM mode.

Note 8: If an error occurs during the reception of a password address or a password string, TMP86FH09NG stops UART communication and enters the halt condition. In this case, initialize TMP86FH09NG by the RESET pin and reactivate the serial PROM mode.

#### Description of RAM loader mode

1. The 1st through 4th bytes of the transmitted and received data contains the same data as in the flash memory writing mode.
2. In the 5th byte of the received data contains the RAM loader command data (60H).
3. When the 5th byte of the received data contains the operation command data shown in Table 1-6, the device echoes back the value which is the same data in the 6th byte position (in this case, 60H). If the 5th byte does not contain the operation command data, the device enters the halt condition after sending 3 bytes of operation command error code (63H).
4. The 7th through m'th bytes of the transmitted and received data contain the same data as in the flash memory writing mode.
5. The m'th + 1 through n'th - 2 bytes of the received data contain the binary data in the Intel Hex format. No received data is echoed back to the external controller. After receiving the start mark (3AH for ":" in the Intel Hex format), the device starts data record reception. Therefore, the received data except 3AH is ignored until the start mark is received. After receiving the start mark, the device receives the data record, that consists of data length, address, record type, write data and checksum. The writing data of the data record is written into RAM specified by address. Since the device starts checksum calculation after receiving an end record, the external controller should wait for the checksum after sending the end record. If a receiving error or Intel Hex format error occurs, the device enters the halts condition without returning an error code to the external controller.
6. The n'th - 1 and n'th bytes contain the checksum upper and lower bytes. For details on how to calculate the SUM, refer to " 15.8 Checksum (SUM) ". The checksum is calculated only when the end record is detected and no receiving error or Intel Hex format error occurs. After sending the end record, the external controller judges whether the transmission is completed correctly by receiving the checksum sent by the device.
7. After transmitting the checksum to the external controller, the boot program jumps to the RAM address that is specified by the first received data record.

Note 1: To rewrite data to Flash memory addresses at which data (including FFH) is already written, make sure to erase the existing data by "sector erase" or "chip erase" before rewriting data.

### 15.6.4 Flash Memory SUM Output Mode (Operation Command: 90H)

Table 15-10 shows flash memory SUM output mode process.

Table 15-10 Flash Memory SUM Output Process

|          | Transfer Bytes       | Transfer Data from External Controller to TMP86FH09NG | Baud Rate                                | Transfer Data from TMP86FH09NG to External Controller  |
|----------|----------------------|---|--|--|
| BOOT ROM | 1st byte<br>2nd byte | Matching data (5AH)<br>-                              | 9600 bps<br>9600 bps                     | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: Nothing transmitted |
|          | 3rd byte<br>4th byte | Baud rate modification data (See Table 15-4)<br>-     | 9600 bps<br>9600 bps                     | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)                         |
|          | 5th byte<br>6th byte | Operation command data (90H)<br>-                     | Modified baud rate<br>Modified baud rate | -<br>OK: Echo back data (90H)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)                   |
|          | 7th byte             | -   | Modified baud rate                       | OK: SUM (Upper byte) (Note 2)<br>Error: Nothing transmitted                                  |
|          | 8th byte             | -   | Modified baud rate                       | OK: SUM (Lower byte) (Note 2)<br>Error: Nothing transmitted                                  |
|          | 9th byte             | (Wait for the next operation command data)            |  | Modified baud rate   |

Note 1: "xxH × 3" indicates that the device enters the halt condition after sending 3 bytes of xxH. For details, refer to " 15.7 Error Code ".

Note 2: Refer to " 15.8 Checksum (SUM) ".

#### Description of the flash memory SUM output mode

1. The 1st through 4th bytes of the transmitted and received data contains the same data as in the flash memory writing mode.
2. The 5th byte of the received data contains the command data in the flash memory SUM output mode (90H).
3. When the 5th byte of the received data contains the operation command data shown in Table 1-6, the device echoes back the value which is the same data in the 6th byte position of the received data (in this case, 90H). If the 5th byte of the received data does not contain the operation command data, the device enters the halt condition after transmitting 3 bytes of operation command error code (63H).
4. The 7th and the 8th bytes contain the upper and lower bits of the checksum, respectively. For how to calculate the checksum, refer to " 15.8 Checksum (SUM) ".
5. After sending the checksum, the device waits for the next operation command data.

15.6.5 Product ID Code Output Mode (Operation Command: C0H)

Table 15-11 shows product ID code output mode process.

Table 15-11 Product ID Code Output Process

|          | Transfer Bytes       | Transfer Data from External Controller to TMP86FH09NG | Baud Rate                                | Transfer Data from TMP86FH09NG to External Controller  |  |
|----------|----------------------|---|--|--|--|
| BOOT ROM | 1st byte<br>2nd byte | Matching data (5AH)<br>-                              | 9600 bps<br>9600 bps                     | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: Nothing transmitted |  |
|          | 3rd byte<br>4th byte | Baud rate modification data (See Table 15-4)<br>-     | 9600 bps<br>9600 bps                     | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)                         |  |
|          | 5th byte<br>6th byte | Operation command data (C0H)<br>-                     | Modified baud rate<br>Modified baud rate | -<br>OK: Echo back data (C0H)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)                   |  |
|          | 7th byte             |   | Modified baud rate                       | 3AH  | Start mark   |
|          | 8th byte             |   | Modified baud rate                       | 0AH  | The number of transfer data (from 9th to 18th bytes) |
|          | 9th byte             |   | Modified baud rate                       | 02H  | Length of address (2 bytes)                          |
|          | 10th byte            |   | Modified baud rate                       | 1DH  | Reserved data  |
|          | 11th byte            |   | Modified baud rate                       | 00H  | Reserved data  |
|          | 12th byte            |   | Modified baud rate                       | 00H  | Reserved data  |
|          | 13th byte            |   | Modified baud rate                       | 00H  | Reserved data  |
|          | 14th byte            |   | Modified baud rate                       | 01H  | ROM block count (1 block)                            |
|          | 15th byte            |   | Modified baud rate                       | C0H  | First address of ROM (Upper byte)                    |
|          | 16th byte            |   | Modified baud rate                       | 00H  | First address of ROM (Lower byte)                    |
|          | 17th byte            |   | Modified baud rate                       | FFH  | End address of ROM (Upper byte)                      |
|          | 18th byte            |   | Modified baud rate                       | FFH  | End address of ROM (Lower byte)                      |
|          | 19th byte            |   | Modified baud rate                       | 22H  | Checksum of transferred data (9th through 18th byte) |
|          | 20th byte            | (Wait for the next operation command data)            | Modified baud rate                       | -  |  |

Note: "xxH × 3" indicates that the device enters the halt condition after sending 3 bytes of xxH. For details, refer to " 15.7 Error Code ".

Description of Product ID code output mode

1. The 1st through 4th bytes of the transmitted and received data contain the same data as in the flash memory writing mode.
2. The 5th byte of the received data contains the product ID code output mode command data (C0H).
3. When the 5th byte contains the operation command data shown in Table 15-6, the device echoes back the value which is the same data in the 6th byte position of the received data (in this case, C0H). If the 5th byte data does not contain the operation command data, the device enters the halt condition after sending 3 bytes of operation command error code (63H).
4. The 9th through 19th bytes contain the product ID code. For details, refer to " 15.11 Product ID Code ".

5. After sending the checksum, the device waits for the next operation command data.

15.6.6 Flash Memory Status Output Mode (Operation Command: C3H)

Table 15-12 shows Flash memory status output mode process.

Table 15-12 Flash Memory Status Output Mode Process

|          | Transfer Bytes       | Transfer Data from External Controller to TMP86FH09NG | Baud Rate                                | Transfer Data from TMP86FH09NG to External Controller   |                                    |
|----------|----------------------|---|--|---|------------------------------------|
| BOOT ROM | 1st byte<br>2nd byte | Matching data (5AH)<br>-                              | 9600 bps<br>9600 bps                     | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: Nothing transmitted  |                                    |
|          | 3rd byte<br>4th byte | Baud rate modification data (See Table 15-4)<br>-     | 9600 bps<br>9600 bps                     | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)  |                                    |
|          | 5th byte<br>6th byte | Operation command data (C3H)<br>-                     | Modified baud rate<br>Modified baud rate | -<br>OK: Echo back data (C3H)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)  |                                    |
|          | 7th byte             |   | Modified baud rate                       | 3AH   | Start mark                         |
|          | 8th byte             |   | Modified baud rate                       | 04H   | Byte count (from 9th to 12th byte) |
|          | 9th byte             |   | Modified baud rate                       | 00H to 03H  | Status code 1                      |
|          | 10th byte            |   | Modified baud rate                       | 00H   | Reserved data                      |
|          | 11th byte            |   | Modified baud rate                       | 00H   | Reserved data                      |
|          | 12th byte            |   | Modified baud rate                       | 00H   | Reserved data                      |
|          | 13th byte            |   | Modified baud rate                       | Checksum 2's complement for the sum of 9th through 12th bytes<br>9th byte    Checksum<br>00H:    00H<br>01H:    FFH<br>02H:    FEH<br>03H:    FDH |                                    |
|          | 14th byte            | (Wait for the next operation command data)            | Modified baud rate                       | -   |                                    |

Note 1: "xxH × 3" indicates that the device enters the halt condition after sending 3 bytes of xxH. For details, refer to " 15.7 Error Code ".

Note 2: For the details on status code 1, refer to " 15.12 Flash Memory Status Code ".

Description of Flash memory status output mode

1. The 1st through 4th bytes of the transmitted and received data contain the same data as in the Flash memory writing mode.
2. The 5th byte of the received data contains the flash memory status output mode command data (C3H).
3. When the 5th byte contains the operation command data shown in Table 15-6, the device echoes back the value which is the same data in the 6th byte position of the received data (in this case, C3H). If the 5th byte does not contain the operation command data, the device enters the halt condition after sending 3 bytes of operation command error code (63H).
4. The 9th through 13th bytes contain the status code. For details on the status code, refer to " 15.12 Flash Memory Status Code ".
5. After sending the status code, the device waits for the next operation command data.

### 15.6.7 Flash Memory Read Protection Setting Mode (Operation Command: FAH)

Table 15-13 shows Flash memory read protection setting mode process.

Table 15-13 Flash Memory Read Protection Setting Mode Process

|          | Transfer Bytes              | Transfer Data from External Controller to TMP86FH09NG | Baud Rate                                | Transfer Data from TMP86FH09NG to External Controller  |
|----------|-----------------------------|---|--|--|
| BOOT ROM | 1st byte<br>2nd byte        | Matching data (5AH)<br>-                              | 9600 bps<br>9600 bps                     | - (Automatic baud rate adjustment)<br>OK: Echo back data (5AH)<br>Error: Nothing transmitted |
|          | 3rd byte<br>4th byte        | Baud rate modification data (See Table 15-4)<br>-     | 9600 bps<br>9600 bps                     | -<br>OK: Echo back data<br>Error: A1H × 3, A3H × 3, 62H × 3 (Note 1)                         |
|          | 5th byte<br>6th byte        | Operation command data (FAH)<br>-                     | Modified baud rate<br>Modified baud rate | -<br>OK: Echo back data (FAH)<br>Error: A1H × 3, A3H × 3, 63H × 3 (Note 1)                   |
|          | 7th byte<br>8th byte        | Password count storage address 15 to 08 (Note 2)      | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 9th byte<br>10th byte       | Password count storage address 07 to 00 (Note 2)      | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 11th byte<br>12th byte      | Password comparison start address 15 to 08 (Note 2)   | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 13th byte<br>14th byte      | Password comparison start address 07 to 00 (Note 2)   | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | 15th byte<br>:<br>m'th byte | Password string (Note 2)<br>-                         | Modified baud rate<br>Modified baud rate | -<br>OK: Nothing transmitted<br>Error: Nothing transmitted                                   |
|          | n'th byte                   | -   | Modified baud rate                       | OK: FBH (Note 3)<br>Error: Nothing transmitted   |
|          | n'+1th byte                 | (Wait for the next operation command data)            | Modified baud rate                       | -  |

Note 1: "xxH × 3" indicates that the device enters the halt condition after sending 3 bytes of xxH. For details, refer to " 15.7 Error Code ".

Note 2: Refer to " 15.10 Passwords ".

Note 3: If the read protection is enabled for a blank product or a password error occurs for a non-blank product, TMP86FH09NG stops UART communication and enters the halt mode. In this case, initialize TMP86FH09NG by the RESET pin and reactivate the serial PROM mode.

Note 4: If an error occurs during reception of a password address or a password string, TMP86FH09NG stops UART communication and enters the halt mode. In this case, initialize TMP86FH09NG by the RESET pin and reactivate the serial PROM mode.

#### Description of the Flash memory read protection setting mode

1. The 1st through 4th bytes of the transmitted and received data contain the same data as in the Flash memory writing mode.
2. The 5th byte of the received data contains the command data in the flash memory status output mode (FAH).
3. When the 5th byte of the received data contains the operation command data shown in Table 1-6, the device echoes back the value which is the same data in the 6th byte position of the received data (in

this case, FAH). If the 5th byte does not contain the operation command data, the device enters the halt condition after transmitting 3 bytes of operation command error code (63H).

4. The 7th through m'th bytes of the transmitted and received data contain the same data as in the flash memory writing mode.
5. The n'th byte contains the status to be transmitted to the external controller in the case of the successful read protection.

## 15.7 Error Code

When detecting an error, the device transmits the error code to the external controller, as shown in Table 15-14.

Table 15-14 Error Code

| Transmit Data | Meaning of Error Data               |
|---------------|-------------------------------------|
| 62H, 62H, 62H | Baud rate modification error.       |
| 63H, 63H, 63H | Operation command error.            |
| A1H, A1H, A1H | Framing error in the received data. |
| A3H, A3H, A3H | Overrun error in the received data. |

Note: If a password error occurs, TMP86FH09NG does not transmit an error code.

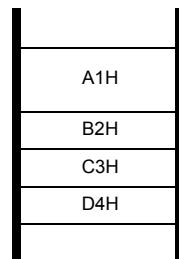
## 15.8 Checksum (SUM)

### 15.8.1 Calculation Method

The checksum (SUM) is calculated with the sum of all bytes, and the obtained result is returned as a word.

The data is read for each byte unit and the calculated result is returned as a word.

Example:



If the data to be calculated consists of the four bytes,  
the checksum of the data is as shown below.

$$\begin{aligned} A1H + B2H + C3H + D4H &= 02EAH \\ \text{SUM (HIGH)} &= 02H \\ \text{SUM (LOW)} &= EAH \end{aligned}$$

The checksum which is transmitted by executing the flash memory write command, RAM loader command, or flash memory SUM output command is calculated in the manner, as shown above.

## 15.8.2 Calculation data

The data used to calculate the checksum is listed in Table 15-15.

Table 15-15 Checksum Calculation Data

| Operating Mode                  | Calculation Data   | Description  |
|---------------------------------|--|--|
| Flash memory writing mode       | Data in the entire area of the flash memory  | Even when a part of the flash memory is written, the checksum of the entire flash memory area (C000H to FFFH) is calculated.<br>The data length, address, record type and checksum in Intel Hex format are not included in the checksum. |
| Flash memory SUM output mode    |  |  |
| RAM loader mode                 | RAM data written in the first received RAM address through the last received RAM address | The length of data, address, record type and checksum in Intel Hex format are not included in the checksum.  |
| Product ID Code Output mode     | 9th through 18th bytes of the transferred data   | For details, refer to " 15.11 Product ID Code ".   |
| Flash Memory Status Output mode | 9th through 12th bytes of the transferred data   | For details, refer to " 15.12 Flash Memory Status Code "   |
| Flash Memory Erasing mode       | All data in the erased area of the flash memory (the whole or part of the flash memory)  | When the sector erase is executed, only the erased area is used to calculate the checksum. In the case of the chip erase, an entire area of the flash memory is used.  |

## 15.9 Intel Hex Format (Binary)

1. After receiving the checksum of a data record, the device waits for the start mark (3AH “:”) of the next data record. After receiving the checksum of a data record, the device ignores the data except 3AH transmitted by the external controller.
2. After transmitting the checksum of end record, the external controller must transmit nothing, and wait for the 2-byte receive data (upper and lower bytes of the checksum).
3. If a receiving error or Intel Hex format error occurs, the device enters the halt condition without returning an error code to the external controller. The Intel Hex format error occurs in the following case:

When the record type is not 00H, 01H, or 02H

When a checksum error occurs

When the data length of an extended record (record type = 02H) is not 02H

When the device receives the data record after receiving an extended record (record type = 02H) with extended address of 1000H or larger.

When the data length of the end record (record type = 01H) is not 00H

## 15.10 Passwords

The consecutive eight or more-byte data in the flash memory area can be specified to the password. TMP86FH09NG compares the data string specified to the password with the password string transmitted from the external controller. The area in which passwords can be specified is located at addresses C000H to FF9FH. The area from FFA0H to FFFFH can not be specified as the passwords area.

If addresses from FFE0H through FFFFH are filled with “FFH”, the passwords are not compared because the product is considered as a blank product. Even in this case, the password count storage addresses and password comparison start address must be specified. Table 15-16 shows the password setting in the blank product and non-blank product.

Table 15-16 Password Setting in the Blank Product and Non-Blank Product

| Password                                    | Blank Product (Note 1)       | Non-Blank Product               |
|---|------------------------------|---------------------------------|
| PNSA<br>(Password count storage address)    | $C000H \leq PNSA \leq FF9FH$ | $C000H \leq PNSA \leq FF9FH$    |
| PCSA<br>(Password comparison start address) | $C000H \leq PCSA \leq FF9FH$ | $C000H \leq PCSA \leq FFA0 - N$ |
| N<br>(Password count)                       | *                            | $8 \leq N$                      |
| Password string setting                     | Not required (Note 5)        | Required (Note 2)               |

Note 1: When addresses from FFE0H through FFFFH are filled with “FFH”, the product is recognized as a blank product.

Note 2: The data including the same consecutive data (three or more bytes) can not be used as a password. (This causes a password error data. TMP86FH09NG transmits no data and enters the halt condition.)

Note 3: \*: Don't care.

Note 4: When the above condition is not met, a password error occurs. If a password error occurs, the device enters the halt condition without returning the error code.

Note 5: In the flash memory writing mode or RAM loader mode, the blank product receives the Intel Hex format data immediately after receiving PCSA without receiving password strings. In this case, the subsequent processing is performed correctly because the blank product ignores the data except the start mark (3AH “:”) as the Intel Hex format data, even if the external controller transmits the dummy password string. However, if the dummy password string contains “3AH”, it is detected as the start mark erroneously. The microcontroller enters the halt mode. If this causes the problem, do not transmit the dummy password strings.

Note 6: In the flash memory erasing mode, the external controller must not transmit the password string for the blank product.

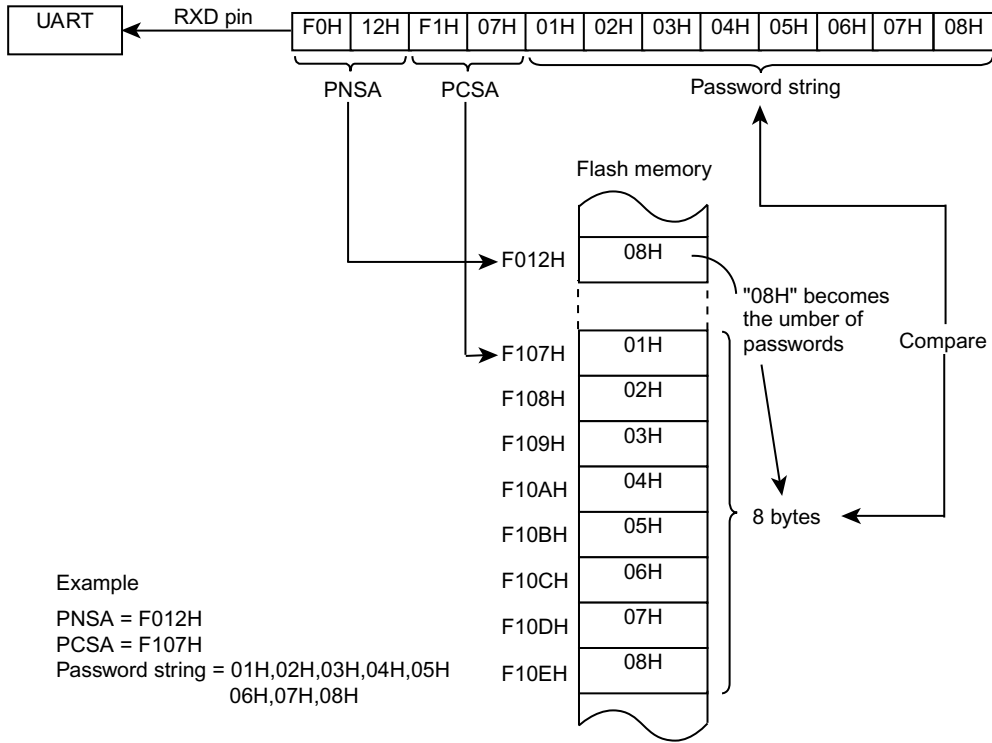


Figure 15-5 Password Comparison

### 15.10.1 Password String

The password string transmitted from the external controller is compared with the specified data in the flash memory. When the password string is not matched to the data in the flash memory, the device enters the halt condition due to the password error.

### 15.10.2 Handling of Password Error

If a password error occurs, the device enters the halt condition. In this case, reset the device to reactivate the serial PROM mode.

### 15.10.3 Password Management during Program Development

If a program is modified many times in the development stage, confusion may arise as to the password. Therefore, it is recommended to use a fixed password in the program development stage.

Example :Specify PNSA to F000H, and the password string to 8 bytes from address F001H (PCSA becomes F001H.)

Password Section code abs = 0F000H

|    |            |                              |
|----|------------|------------------------------|
| DB | 08H        | : PNSA definition            |
| DB | "CODE1234" | : Password string definition |

## 15.11 Product ID Code

The product ID code is the 13-byte data containing the start address and the end address of ROM. Table 15-17 shows the product ID code format.

Table 15-17 Product ID Code Format

| Data | Description   | In the Case of TMP86FH09NG |
|------|---|----------------------------|
| 1st  | Start Mark (3AH)  | 3AH                        |
| 2nd  | The number of transfer data (10 bytes from 3rd to 12th byte)                            | 0AH                        |
| 3rd  | Address length (2 bytes)  | 02H                        |
| 4th  | Reserved data   | 1DH                        |
| 5th  | Reserved data   | 00H                        |
| 6th  | Reserved data   | 00H                        |
| 7th  | Reserved data   | 00H                        |
| 8th  | ROM block count   | 01H                        |
| 9th  | The first address of ROM (Upper byte)   | C0H                        |
| 10th | The first address of ROM (Lower byte)   | 00H                        |
| 11th | The end address of ROM (Upper byte)   | FFH                        |
| 12th | The end address of ROM (Lower byte)   | FFH                        |
| 13th | Checksum of the transferred data (2's compliment for the sum of 3rd through 12th bytes) | 22H                        |

## 15.12 Flash Memory Status Code

The flash memory status code is the 7-byte data including the read protection status and the status of the data from FFE0H to FFFFH. Table 15-18 shows the flash memory status code.

Table 15-18 Flash Memory Status Code

| Data     | Description   | In the Case of TMP86FH09NG   |          |          |     |     |     |     |     |     |     |     |
|----------|---|--|----------|----------|-----|-----|-----|-----|-----|-----|-----|-----|
| 1st      | Start mark  | 3AH  |          |          |     |     |     |     |     |     |     |     |
| 2nd      | Transferred data count (3rd through 6th byte)   | 04H  |          |          |     |     |     |     |     |     |     |     |
| 3rd      | Status code   | 00H to 03H<br>(See figure below)   |          |          |     |     |     |     |     |     |     |     |
| 4th      | Reserved data   | 00H  |          |          |     |     |     |     |     |     |     |     |
| 5th      | Reserved data   | 00H  |          |          |     |     |     |     |     |     |     |     |
| 6th      | Reserved data   | 00H  |          |          |     |     |     |     |     |     |     |     |
| 7th      | Checksum of the transferred data (2's compliment for the sum of 3rd through 6th data) | <table border="0"> <tr> <td>3rd byte</td> <td>checksum</td> </tr> <tr> <td>00H</td> <td>00H</td> </tr> <tr> <td>01H</td> <td>FFH</td> </tr> <tr> <td>02H</td> <td>FEH</td> </tr> <tr> <td>03H</td> <td>FDH</td> </tr> </table> | 3rd byte | checksum | 00H | 00H | 01H | FFH | 02H | FEH | 03H | FDH |
| 3rd byte | checksum  |  |          |          |     |     |     |     |     |     |     |     |
| 00H      | 00H   |  |          |          |     |     |     |     |     |     |     |     |
| 01H      | FFH   |  |          |          |     |     |     |     |     |     |     |     |
| 02H      | FEH   |  |          |          |     |     |     |     |     |     |     |     |
| 03H      | FDH   |  |          |          |     |     |     |     |     |     |     |     |

Status Code 1

|   |   |   |   |   |   |       |       |                            |
|---|---|---|---|---|---|-------|-------|----------------------------|
| 7 | 6 | 5 | 4 | 3 | 2 | 1     | 0     |                            |
|   |   |   |   |   |   | RPENA | BLANK | (Initial Value: 0000 00**) |

|       |                                     |   |
|-------|-------------------------------------|---|
| RPENA | Flash memory read protection status | 0: Read protection is disabled.<br>1: Read protection is enabled.   |
| BLANK | The status from FFE0H to FFFFH.     | 0: All data is FFH in the area from FFE0H to FFFFH.<br>1: The value except FFH is included in the area from FFE0H to FFFFH. |

Some operation commands are limited by the flash memory status code 1. If the read protection is enabled, flash memory writing mode command and RAM loader mode command can not be executed. Erase all flash memory before executing these command.

| RPENA | BLANK | Flash Memory Writing Mode | RAM Loader Mode | Flash memory SUM Output Mode | Product ID Code Output Mode | Flash Memory Status Output Mode | Flash Memory Erasing Mode |              | Read Protection Setting Mode |
|-------|-------|---------------------------|-----------------|------------------------------|-----------------------------|---------------------------------|---------------------------|--------------|------------------------------|
|       |       |                           |                 |                              |                             |                                 | Chip Erase                | Sector Erase |                              |
| 0     | 0     | m                         | m               | m                            | m                           | m                               | m                         |              | ×                            |
| 0     | 1     | Pass                      | Pass            | m                            | m                           | m                               | Pass                      |              | Pass                         |
| 1     | 0     | ×                         | ×               | m                            | m                           | m                               | m                         | ×            | ×                            |
| 1     | 1     | ×                         | ×               | m                            | m                           | m                               | Pass                      | ×            | Pass                         |

Note: m: The command can be executed.

Pass: The command can be executed with a password.

×: The command can not be executed.

(After echoing the command back to the external controller, TMP86FH09NG stops UART communication and enters the halt condition.)

## 15.13 Specifying the Erasure Area

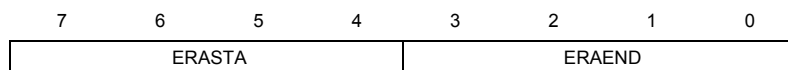
In the flash memory erasing mode, the erasure area of the flash memory is specified by n-2 byte data.

The start address of an erasure area is specified by ERASTA, and the end address is specified by ERAEND.

If ERASTA is equal to or smaller than ERAEND, the sector erase (erasure in 4 kbyte units) is executed. Executing the sector erase while the read protection is enabled results in an infinite loop.

If ERASTA is larger than ERAEND, the chip erase (erasure of an entire flash memory area) is executed and the read protection is disabled. Therefore, execute the chip erase (not sector erase) to disable the read protection.

### Erasure Area Specification Data (n-2 byte data)



|        |                                       |  |
|--------|---------------------------------------|--|
| ERASTA | The start address of the erasure area | 0000: from 0000H<br>0001: from 1000H<br>0010: from 2000H<br>0011: from 3000H<br>0100: from 4000H<br>0101: from 5000H<br>0110: from 6000H<br>0111: from 7000H<br>1000: from 8000H<br>1001: from 9000H<br>1010: from A000H<br>1011: from B000H<br>1100: from C000H<br>1101: from D000H<br>1110: from E000H<br>1111: from F000H |
| ERAEND | The end address of the erasure area   | 0000: to 0FFFH<br>0001: to 1FFFH<br>0010: to 2FFFH<br>0011: to 3FFFH<br>0100: to 4FFFH<br>0101: to 5FFFH<br>0110: to 6FFFH<br>0111: to 7FFFH<br>1000: to 8FFFH<br>1001: to 9FFFH<br>1010: to AFFFH<br>1011: to BFFFH<br>1100: to CFFFH<br>1101: to DFFFH<br>1110: to EFFFH<br>1111: to FFFFH                                 |

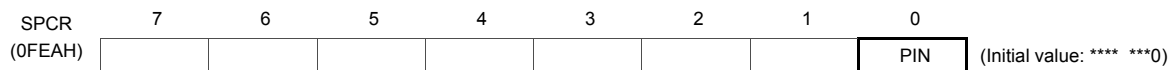
Note: When the sector erase is executed for the area containing no flash cell, TMP86FH09NG stops the UART communication and enters the halt condition.

## 15.14 Port Input Control Register

In the serial PROM mode, the input level is fixed to the all ports except P03 and P02 ports with a hardware feature to prevent overlap current to unused ports. (All port inputs and peripheral function inputs shared with the ports become invalid.) Therefore, to access to the flash memory in the RAM loader mode without UART communication, port inputs must be valid. To make port inputs valid, set the pin of the port input control register (SPCR) to "1".

The SPCR register is not operated in the MCU mode.

Port Input Control Register



|     |  |  |     |
|-----|--|--|-----|
| PIN | Port input control in the serial PROM mode | 0 : Invalid port inputs (The input level is fixed with a hardware feature.)<br>1 : Valid port inputs | R/W |
|-----|--|--|-----|

Note 1: The SPCR register can be read or written only in the serial PROM mode. When the write instruction is executed to the SPCR register in the MCU mode, the port input control can not be performed. When the read instruction is executed for the SPCR register in the MCU mode, read data of bit7 to 1 are unstable.

Note 2: All I/O ports except P03 and P02 ports are controlled by the SPCR register.



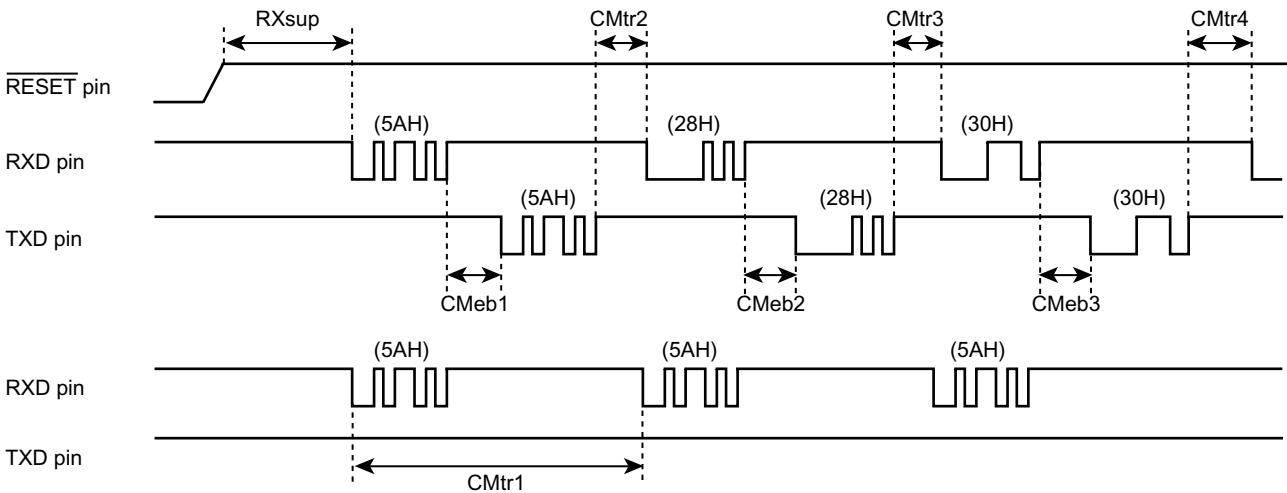
### 15.16UART Timing

Table 15-19 UART Timing-1 (VDD = 4.75 to 5.25 V, fc = 2 to 16 MHz, Topr = -10 to 40°C)

| Parameter  | Symbol | Clock Frequency (fc) | Minimum Required Time |                |
|--|--------|----------------------|-----------------------|----------------|
|  |        |                      | At fc = 2 MHz         | At fc = 16 MHz |
| Time from matching data reception to the echo back               | CMeb1  | Approx. 930          | 465 μs                | 58.1 μs        |
| Time from baud rate modification data reception to the echo back | CMeb2  | Approx. 980          | 490 μs                | 61.3 μs        |
| Time from operation command reception to the echo back           | CMeb3  | Approx. 800          | 400 μs                | 50 μs          |
| Checksum calculation time  | CKsm   | Approx. 7864500      | 3.93 s                | 491.5 μs       |
| Erasure time of an entire flash memory                           | CEall  | -                    | 30 ms                 | 30 ms          |
| Erasure time for a sector of a flash memory (in 4-kbyte units)   | CEsec  | -                    | 15 ms                 | 15 ms          |

Table 15-20 UART Timing-2 (VDD = 4.75 to 5.25 V, fc = 2 to 16 MHz, Topr = -10 to 40°C)

| Parameter   | Symbol | Clock Frequency (fc) | Minimum Required Time |                |
|---|--------|----------------------|-----------------------|----------------|
|   |        |                      | At fc = 2 MHz         | At fc = 16 MHz |
| Time from the reset release to the acceptance of start bit of RXD pin   | RXsup  | 2100                 | 1.05 ms               | 131.3 ms       |
| Matching data transmission interval   | CMtr1  | 28500                | 14.2 ms               | 1.78 ms        |
| Time from the echo back of matching data to the acceptance of baud rate modification data                       | CMtr2  | 380                  | 190 μs                | 23.8 μs        |
| Time from the echo back of baud rate modification data to the acceptance of an operation command                | CMtr3  | 650                  | 325 μs                | 40.6 μs        |
| Time from the echo back of operation command to the acceptance of password count storage addresses (Upper byte) | CMtr4  | 800                  | 400 μs                | 50 μs          |





## 16. Input/Output Circuitry

### 16.1 Control Pins

The input/output circuitries of the TMP86FH09NG control pins are shown below.

| Control Pin               | I/O             | Input/Output Circuitry | Remarks  |
|---------------------------|-----------------|------------------------|--|
| XIN<br>XOUT               | Input<br>Output |                        | Resonator connecting pins<br>$R_f = 1.55 \text{ M}\Omega$ (typ.)<br>$R_o = 0.5 \text{ k}\Omega$ (typ.)   |
| XTIN<br>XTOUT             | Input           |                        | Resonator connecting pins<br>$R_f = 8 \text{ M}\Omega$ (typ.)<br>$R_o = 200 \text{ k}\Omega$ (typ.)      |
| $\overline{\text{RESET}}$ | Input           |                        | Hysteresis input<br>Pull-up resistor<br>$R_{IN} = 220 \text{ k}\Omega$ (typ.)<br>$R = 100 \Omega$ (typ.) |
| TEST                      | Input           |                        | Without Pull-down resistor<br>$R = 100 \Omega$ (typ.)<br>Fix the TEST pin at low-level in MCU mode.      |

## 16.2 Input/Output Ports

| Control Pin | I/O | Input/Output Circuitry  | Remarks  |
|-------------|-----|---|--|
| P0          | I/O | <p>Initial "High-Z"</p> <p>Pch control<br/>Data output<br/>Input from output latch<br/>High-Z control<br/>Pin input</p>   | <p>Sink open drain output<br/>or<br/>Push-Pull output<br/>Hysteresis input<br/>High current output(Nch)<br/>(Programmable port option)</p> |
| P1          | I/O | <p>Initial "High-Z"</p> <p>Data output<br/>Disable<br/>Pin input</p>  | <p>Tri-state I/O<br/>Hysteresis input</p>  |
| P2          | I/O | <p>Initial "High-Z"</p> <p>Data output<br/>Input from output latch<br/>Pin input</p>  | <p>Sink open drain output<br/>Hysteresis input</p>   |
| P3          | I/O | <p>Initial "High-Z"</p> <p>Analog input<br/>Data output<br/>Disable<br/>Key on wake up input<br/>Pin input<br/>P37 to 34</p> <p>Analog input<br/>Data output<br/>Disable<br/>Pin input<br/>P33,32</p> <p>Data output<br/>Disable<br/>Pin input<br/>P31,30</p> | <p>Tri-state I/O<br/>Hysteresis input or<br/>CMOS input</p>  |

Note: Input status on pins set for input mode are read in into the internal circuit. Therefore, when using the ports in a mixture of input and output modes, the contents of the output latches for the ports that are set for input mode may be rewritten by execution of bit manipulating instructions.

## 17. Electrical Characteristics

### 17.1 Absolute Maximum Ratings

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.

(VSS = 0 V)

| Parameter  | Symbol            | Pins             | Ratings                | Unit               |
|--|-------------------|------------------|------------------------|--------------------|
| Supply voltage   | $V_{DD}$          |                  | -0.3 to 6.0            | V                  |
| Input voltage  | $V_{IN}$          |                  | -0.3 to $V_{DD} + 0.3$ | V                  |
| Output voltage   | $V_{OUT1}$        |                  | -0.3 to $V_{DD} + 0.3$ | V                  |
| Output current (Per 1 pin)                                   | $I_{OUT1}$        | P0, P1, P3 ports | -1.8                   | mA                 |
|  | $I_{OUT2}$        | P1, P2, P3 ports | 3.2                    |                    |
|  | $I_{OUT3}$        | P0 ports         | 30                     |                    |
| Output current (Total)                                       | $\Sigma I_{OUT1}$ | P1, P2, P3 ports | 60                     |                    |
|  | $\Sigma I_{OUT2}$ | P0 ports         | 80                     |                    |
| Power dissipation [ $T_{opr} = 85\text{ }^{\circ}\text{C}$ ] | $P_D$             |                  | 300                    | mW                 |
| Soldering temperature (time)                                 | $T_{sld}$         |                  | 260 (10 s)             | $^{\circ}\text{C}$ |
| Storage temperature  | $T_{stg}$         |                  | -55 to 125             |                    |
| Operating temperature  | $T_{opr}$         |                  | -40 to 85              |                    |

## 17.2 Recommended Operating Conditions

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.

### 17.2.1 MCU mode (Flash Programming or erasing)

( $V_{SS} = 0\text{ V}$ ,  $T_{opr} = -10\text{ to }40^{\circ}\text{C}$ )

| Parameter        | Symbol    | Pins                    | Ratings                     | Min                  | Max                  | Unit |
|------------------|-----------|-------------------------|-----------------------------|----------------------|----------------------|------|
| Supply voltage   | $V_{DD}$  |                         | NORMAL1, 2 modes            | 4.75                 | 5.25                 | V    |
| Input high level | $V_{IH1}$ | Except hysteresis input | $V_{DD} \geq 4.75\text{ V}$ | $V_{DD} \times 0.70$ | $V_{DD}$             |      |
|                  | $V_{IH2}$ | Hysteresis input        |                             | $V_{DD} \times 0.75$ |                      |      |
| Input low level  | $V_{IL1}$ | Except hysteresis input | $V_{DD} \geq 4.75\text{ V}$ | 0                    | $V_{DD} \times 0.30$ |      |
|                  | $V_{IL2}$ | Hysteresis input        |                             |                      | $V_{DD} \times 0.25$ |      |
| Clock frequency  | $f_c$     | XIN, XOUT               |                             | 1.0                  | 16.0                 |      |

### 17.2.2 MCU mode (Except Flash Programming or erasing)

( $V_{SS} = 0\text{ V}$ ,  $T_{opr} = -40\text{ to }85^{\circ}\text{C}$ )

| Parameter        | Symbol    | Pins                    | Ratings                               | Min                  | Max                  | Unit |     |
|------------------|-----------|-------------------------|---------------------------------------|----------------------|----------------------|------|-----|
| Supply voltage   | $V_{DD}$  |                         | $f_c = 16\text{ MHz}$                 | 4.5                  | 5.5                  | V    |     |
|                  |           |                         | NORMAL1, 2 modes<br>IDLE0, 1, 2 modes |                      |                      |      |     |
|                  |           |                         | $f_c = 8\text{ MHz}$                  |                      |                      |      |     |
|                  |           |                         | NORMAL1, 2 modes<br>IDLE0, 1, 2 modes |                      |                      |      |     |
|                  |           |                         | $f_s = 32.768\text{ KHz}$             |                      |                      |      |     |
|                  |           |                         | SLOW1, 2 modes<br>SLEEP0, 1, 2 modes  |                      |                      |      |     |
|                  |           |                         | STOP mode                             |                      |                      |      |     |
| Input high level | $V_{IH1}$ | Except hysteresis input | $V_{DD} \geq 4.5\text{ V}$            | $V_{DD} \times 0.70$ | $V_{DD}$             | V    |     |
|                  | $V_{IH2}$ | Hysteresis input        |                                       | $V_{DD} \times 0.75$ |                      |      |     |
| Input low level  | $V_{IL1}$ | Except hysteresis input | $V_{DD} \geq 4.5\text{ V}$            | 0                    | $V_{DD} \times 0.30$ |      |     |
|                  | $V_{IL2}$ | Hysteresis input        |                                       |                      | $V_{DD} \times 0.25$ |      |     |
| Clock frequency  | $f_c$     | XIN, XOUT               | $V_{DD} = 4.5\text{ to }5.5\text{ V}$ | 1.0                  | 8.0                  |      | MHz |
|                  |           |                         |                                       |                      | 16.0                 |      |     |
|                  | $f_s$     | XTIN, XTOUT             | $V_{DD} = 4.5\text{ to }5.5\text{ V}$ | 30.0                 | 34.0                 | kHz  |     |

## 17.2.3 Serial PROM mode

(V<sub>SS</sub> = 0 V, T<sub>opr</sub> = -10 to 40 °C)

| Parameter          | Symbol           | Pins                    | Condition                | Min                    | Max                    | Unit |
|--------------------|------------------|-------------------------|--------------------------|------------------------|------------------------|------|
| Supply voltage     | V <sub>DD</sub>  |                         | NORMAL 1, 2 modes        | 4.75                   | 5.25                   | V    |
| Input high voltage | V <sub>IH1</sub> | Except hysteresis input | V <sub>DD</sub> ≥ 4.75 V | V <sub>DD</sub> × 0.70 | V <sub>DD</sub>        |      |
|                    | V <sub>IH2</sub> | Hysteresis input        |                          | V <sub>DD</sub> × 0.75 |                        |      |
| Input low voltage  | V <sub>IL1</sub> | Except hysteresis input | V <sub>DD</sub> ≥ 4.75 V | 0                      | V <sub>DD</sub> × 0.30 |      |
|                    | V <sub>IL2</sub> | Hysteresis input        |                          | V <sub>DD</sub> × 0.25 |                        |      |
| Clock frequency    | f <sub>c</sub>   | XIN, XOUT               |                          | 2.0                    | 16.0                   | MHz  |

### 17.3 DC Characteristics

( $V_{SS} = 0\text{ V}$ ,  $T_{opr} = -40\text{ to }85\text{ }^{\circ}\text{C}$ )

| Parameter                             | Symbol    | Pins                            | Condition   | Min | Typ.                    | Max     | Unit             |               |       |       |       |
|---------------------------------------|-----------|---------------------------------|---|-----|-------------------------|---------|------------------|---------------|-------|-------|-------|
| Hysteresis voltage                    | $V_{HS}$  | Hysteresis input                |   | –   | 0.9                     | –       | V                |               |       |       |       |
| Input current                         | $I_{IN1}$ | TEST                            | $V_{DD} = 5.5\text{ V}$ , $V_{IN} = 5.5\text{ V}/0\text{ V}$  | –   | –                       | $\pm 2$ | $\mu\text{A}$    |               |       |       |       |
|                                       | $I_{IN2}$ | Sink open drain, tri-state port |   |     |                         |         |                  |               |       |       |       |
|                                       | $I_{IN3}$ | RESET, STOP                     |   |     |                         |         |                  |               |       |       |       |
| Input resistance                      | $R_{IN2}$ | RESET pull-up                   | $V_{DD} = 5.5\text{ V}$ , $V_{IN} = 0\text{ V}$   | –   | 220                     | –       | $\text{k}\Omega$ |               |       |       |       |
| Output leakage current                | $I_{LO1}$ | Sink open drain port            | $V_{DD} = 5.5\text{ V}$ , $V_{OUT} = 5.5\text{ V}$  | –   | –                       | 2       | $\mu\text{A}$    |               |       |       |       |
|                                       | $I_{LO2}$ | Tri-state port                  | $V_{DD} = 5.5\text{ V}$ , $V_{OUT} = 5.5\text{ V}/0\text{ V}$   | –   | –                       | $\pm 2$ |                  |               |       |       |       |
| Output high voltage                   | $V_{OH}$  | Tri-state port                  | $V_{DD} = 4.5\text{ V}$ , $I_{OH} = -0.7\text{ mA}$   | 4.1 | –                       | –       | V                |               |       |       |       |
| Output low voltage                    | $V_{OL}$  | Except XOUT, P3, P5             | $V_{DD} = 4.5\text{ V}$ , $I_{OL} = 1.6\text{ mA}$  | –   | –                       | 0.4     |                  |               |       |       |       |
| Output low curren                     | $I_{OL}$  | High current port (P0 Port)     | $V_{DD} = 4.5\text{ V}$ , $V_{OL} = 1.0\text{ V}$   | –   | 20                      | –       | mA               |               |       |       |       |
| Supply current in NORMAL1, 2 modes    | $I_{DD}$  |                                 | $V_{DD} = 5.5\text{ V}$<br>$V_{IN} = 5.3\text{ V}/0.2\text{ V}$<br>$f_c = 16\text{ MHz}$<br>$f_s = 32.768\text{ kHz}$ | –   | T.B.D                   | T.B.D   | mA               |               |       |       |       |
| Supply current in IDLE 0, 1, 2 modes  |           |                                 |   |     |                         |         |                  | –             | T.B.D | T.B.D |       |
| Supply current in SLOW1 mode          |           |                                 | $V_{DD} = 5.5\text{ V}$<br>$V_{IN} = 5.3\text{ V}/0.2\text{ V}$<br>$f_s = 32.768\text{ kHz}$                          | –   | T.B.D                   | T.B.D   |                  | $\mu\text{A}$ |       |       |       |
| Supply current in SLEEP1 mode         |           |                                 |   |     |                         |         |                  |               | –     | T.B.D | T.B.D |
| Supply current in SLEEP0 mode         |           |                                 |   |     |                         |         |                  |               | –     | T.B.D | T.B.D |
| Supply current in STOP mode           |           |                                 | $V_{DD} = 5.5\text{ V}$<br>$V_{IN} = 5.3\text{ V}/0.2\text{ V}$   | –   | T.B.D                   | T.B.D   |                  |               |       |       |       |
| Peak current for SLOW1 mode (Note5,6) |           |                                 | $I_{DDP-P}$   |     | $V_{DD} = 5.5\text{ V}$ | –       |                  | T.B.D         | –     | mA    |       |

Note 1: Typical values show those at  $T_{opr} = 25^{\circ}\text{C}$  and  $V_{DD} = 5\text{ V}$ .

Note 2: Input current ( $I_{IN3}$ ): The current through pull-up resistor is not included.

Note 3: The supply currents of SLOW2 and SLEEP2 modes are equivalent to those of IDLE0, IDLE1 and IDLE2 modes.

Note 4: When a program is executing in the flash memory or when data is being read from the flash memory, the flash memory operates in an intermittent manner, causing peak currents in the operation current, as shown in Figure 17-1. In this case, the supply current  $I_{DD}$  (in NORMAL1, NORMAL2 and SLOW1 modes) is defined as the sum of the average peak current and MCU current.

Note 5: When designing the power supply, make sure that peak currents can be supplied. In SLOW1 mode, the difference between the peak current and the average current becomes large.

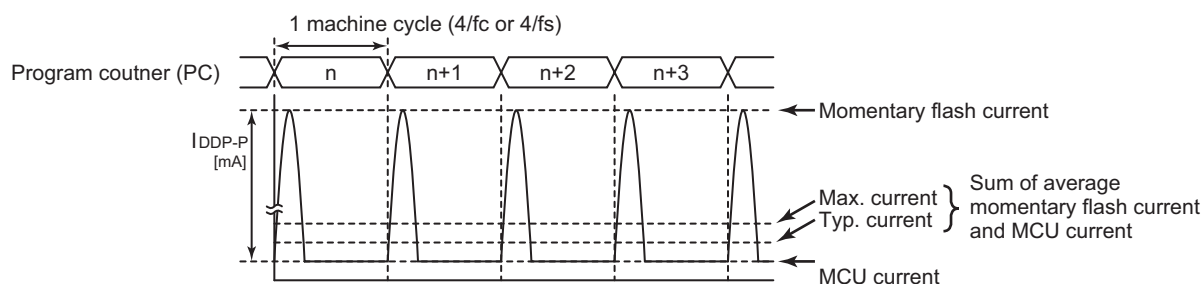


Figure 17-1 Intermittent Operation of Flash Memory

17.4 AD Characteristics

( $V_{SS} = 0.0\text{ V}$ ,  $4.5\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ ,  $T_{opr} = -40\text{ to }85\text{ }^{\circ}\text{C}$ )

| Paramete             | Symbol    | Condition  | Min      | Typ. | Max      | Unit |
|----------------------|-----------|--|----------|------|----------|------|
| Analog input voltage | $V_{AIN}$ |  | $V_{SS}$ | –    | $V_{DD}$ | V    |
| Non linearity error  |           | $V_{DD} = 5.0\text{ V}$<br>$V_{SS} = 0.0\text{ V}$ | –        | –    | T.B.D    | LSB  |
| Zero point error     |           |  | –        | –    | T.B.D    |      |
| Full scale error     |           |  | –        | –    | T.B.D    |      |
| Total error          |           |  | –        | –    | T.B.D    |      |

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 defferent in recommended value by power supply voltage.

Note 3: The voltage to be input on the AIN input pin must not exceed the range between  $V_{DD}$  and  $V_{SS}$ . If a voltage outside this range is input, conversion values will become unstable and conversion values of other channels will also be affected.

## 17.5 AC Characteristics

( $V_{SS} = 0\text{ V}$ ,  $4.5\text{ V} \leq V_{DD} \leq 5.5\text{ V}$ ,  $T_{opr} = -40\text{ to }85^\circ\text{C}$ )

| Parameter                    | Symbol           | Condition  | Min   | Typ.  | Max   | Unit |
|------------------------------|------------------|--|-------|-------|-------|------|
| Machine cycle time           | t <sub>cy</sub>  | NORMAL1, 2 modes   | 0.25  | –     | 4     | μs   |
|                              |                  | IDLE0, 1, 2 modes  |       |       |       |      |
|                              |                  | SLOW1, 2 modes   | 117.6 | –     | 133.3 |      |
|                              |                  | SLEEP0, 1, 2 modes   |       |       |       |      |
| High-level clock pulse width | t <sub>WCH</sub> | For external clock operation (XIN input)<br>f <sub>c</sub> = 16 MHz      | –     | 31.25 | –     | ns   |
| Low-level clock pulse width  | t <sub>WCL</sub> |  |       |       |       |      |
| High-level clock pulse width | t <sub>WSH</sub> | For external clock operation (XTIN input)<br>f <sub>s</sub> = 32.768 kHz | –     | 15.26 | –     | μs   |
| Low-level clock pulse width  | t <sub>WSL</sub> |  |       |       |       |      |

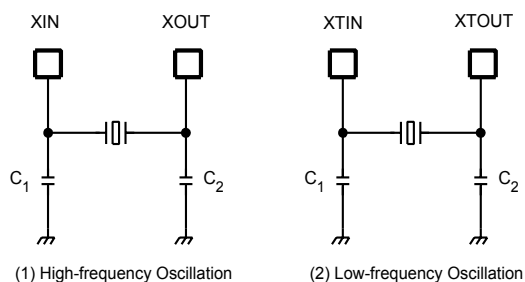
## 17.6 Flash Characteristics

### 17.6.1 Write/Erase Characteristics

( $V_{SS} = 0\text{ V}$ )

| Parameter                                   | Condition  | Min | Typ. | Max. | Unit  |
|---|--|-----|------|------|-------|
| Number of guaranteed writes to flash memory | $V_{SS} = 0\text{ V}$ , $T_{opr} = -10\text{ to }40^\circ\text{C}$ | –   | –    | 100  | Times |

## 17.7 Recommended Oscillating Conditions



Note 1: To ensure stable oscillation, the resonator position, load capacitance, etc. must be appropriate. Because these factors are greatly affected by board patterns, please be sure to evaluate operation on the board on which the device will actually be mounted.

Note 2: 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.

Note 3: The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following URL:  
<http://www.murata.co.jp/search/index.html>

## 17.8 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-63Pb 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 criterion of the above test is as follows:

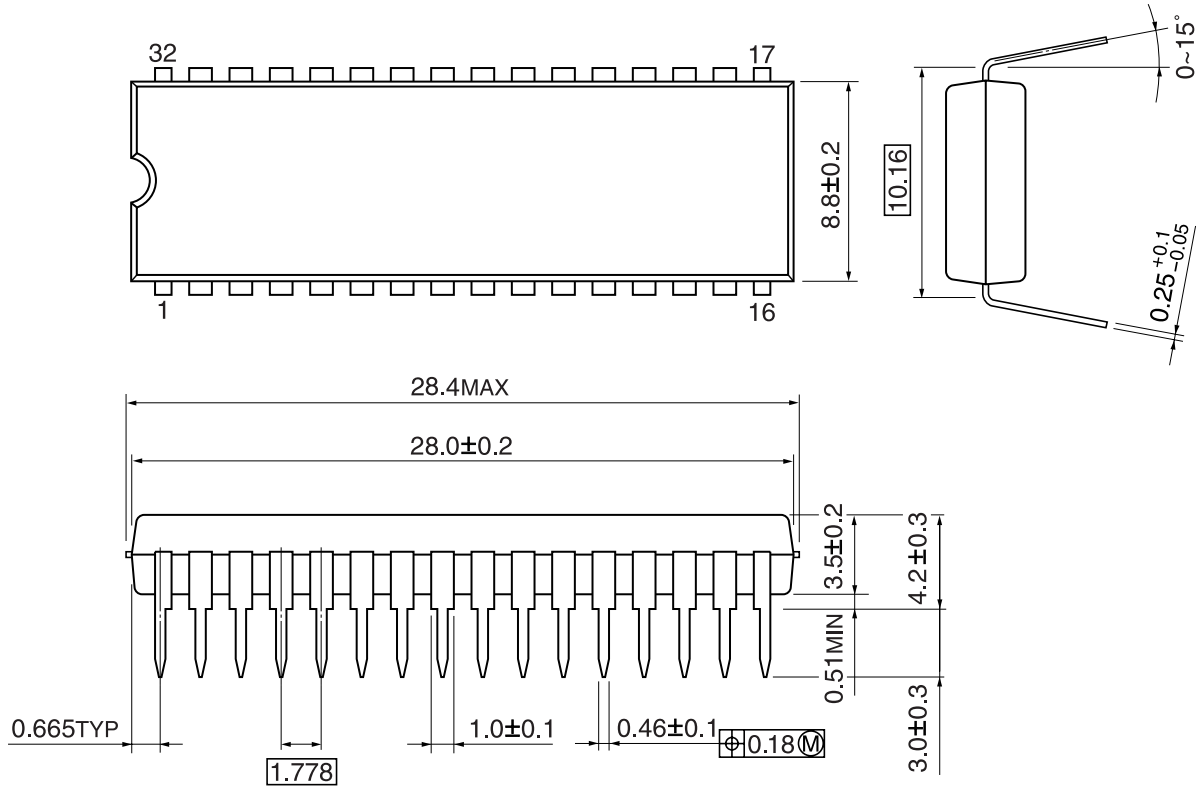
Solderability rate until forming  $\geq 95$  %

- 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.

18. Package Dimension

P-SDIP32-400-1.78

Unit: mm





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.

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- ⊖ [View TMP86FH09NG on WIN SOURCE](#)
- ⊖ [Toshiba Semiconductor and Storage Information](#)

## Optimize Your Supply Chain with WIN SOURCE Solutions

- ✓ Global Sourcing Solution
- ✓ Obsolete Management
- ✓ Cost Control Management
- ✓ Shortage Management
- ✓ Alternative Solution
- ✓ Excess Inventory Management