TOSHIBA

# 8 Bit Microcontroller
## TLCS-870/C Series

# TMP86CS49UG

TOSHIBA CORPORATION

The information contained herein is subject to change without notice. 021023 _ D

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. 021023_A

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. 021023_B

The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations. 060106_Q

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. 021023_C

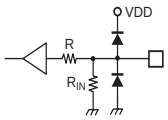
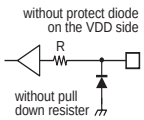
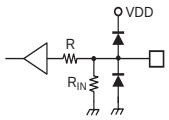
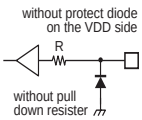
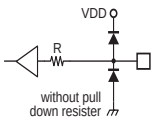
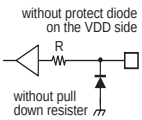
The products described in this document may include products subject to the foreign exchange and foreign trade laws. 021023_F

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. 030619_S

© 2007 TOSHIBA CORPORATION
All Rights Reserved

## Differences among Products

### Differences in Functions

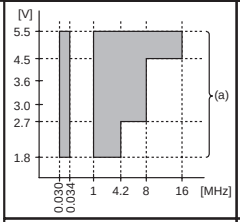
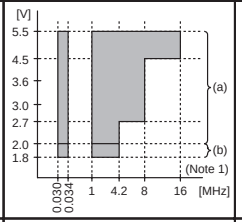
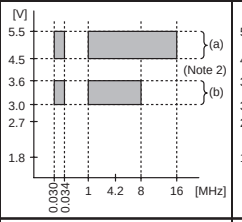
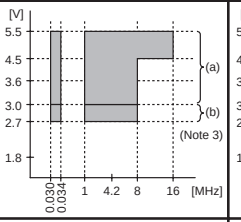
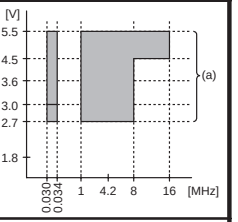
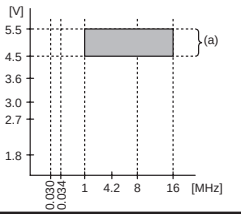
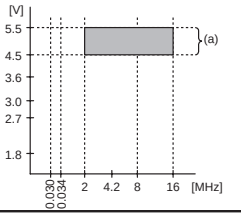
| | 86CH49 | 86CM49 | 86PM49 | 86CS49 | 86FS49 | 86FS49A | 86FS49B |
|---|---|---|---|---|---|---|---|
| ROM | 16 Kbytes (Mask) | 32 Kbytes (Mask) | 32 Kbytes (OTP) | 60 Kbytes (Mask) | 60 Kbytes (Flash) | | |
| RAM | 512 bytes | 1 Kbyte | 1 Kbyte | 2 Kbytes | 2 Kbytes | | |
| DBR(note1) | 128 bytes (Flash control register not contained) | | | | 128 bytes (Flash control register contained) | | |
| I/O | 56 pins | | | | | | |
| High-current port | 13 pins (sink open drain) | | | | | | |
| Interrupt | External: 5 interrupts, Internal: 19 interrupts | | | | | | |
| Timer/counter | 16-bit: 2 channels<br>8-bit: 4 channels | | | | | | |
| UART | 2 channels | | | | | | |
| SIO | 2 channels | | | | | | |
| $I^2C$ | 1 channel | | | | | | |
| Key-on wake-up | 4 channels | | | | | | |
| 10-bit AD converter (note2) | 16 channels | | | | | | |
| Flash Security | N.A. | | | | Read protect | | Read/Write protect |
| Structurer of TEST pin |  |  |  |  |  |  | |
| Emulation chip | TMP86C949XB | | | | | | |
| Package | QFP64-P-1414-0.80A | LQFP64-P-1010-0.50D<br>SDIP64-P-750-1.78 | | QFP64-P-1414-0.80A<br>LQFP64-P-1010-0.50D<br>— | | | |

Note 1: The products with Flash memory (86FS49, 86FS49A, 86FS49B) contain the Flash control register (FLSCR) at 0FFFH in the DBR area. The products with mask ROM or OTP and the emulation chip do not have the FLSCR register. In these devices, therefore, a program that accesses the FLSCR register cannot function properly (executes differently as in the case of a Flash product).

Note 2: In this data sheet,the following pin names and register names have been changed from the data sheet of the old edition. Although the names have been changed, their functions remain the same.

| | OLD name | NEW name |
|---|---|---|
| AD Converter<br>analog input pin name | P60(AIN00)<br>P61(AIN01)<br>P62(AIN02)<br>P63(AIN03)<br>P64(AIN04)<br>P65(AIN05)<br>P66(AIN06)<br>P67(AIN07)<br>P70(AIN10)<br>P71(AIN11)<br>P72(AIN12)<br>P73(AIN13)<br>P74(AIN14)<br>P75(AIN15)<br>P76(AIN16)<br>P77(AIN17) | P60(AIN0)<br>P61(AIN1)<br>P62(AIN2)<br>P63(AIN3)<br>P64(AIN4)<br>P65(AIN5)<br>P66(AIN6)<br>P67(AIN7)<br>P70(AIN8)<br>P71(AIN9)<br>P72(AIN10)<br>P73(AIN11)<br>P74(AIN12)<br>P75(AIN13)<br>P76(AIN14)<br>P77(AIN15) |
| ADCCR1 register <SAIN><br>function name | 0000:AIN00<br>0001:AIN01<br>0010:AIN02<br>0011:AIN03<br>0100:AIN04<br>0101:AIN05<br>0110:AIN06<br>0111:AIN07<br>1000:AIN10<br>1001:AIN11<br>1010:AIN12<br>1011:AIN13<br>1100:AIN14<br>1101:AIN15<br>1110:AIN16<br>1111:AIN17 | 0000:AIN0<br>0001:AIN1<br>0010:AIN2<br>0011:AIN3<br>0100:AIN4<br>0101:AIN5<br>0110:AIN6<br>0111:AIN7<br>1000:AIN8<br>1001:AIN9<br>1010:AIN10<br>1011:AIN11<br>1100:AIN12<br>1101:AIN13<br>1110:AIN14<br>1111:AIN15 |

Differences in Electrical Characteristics

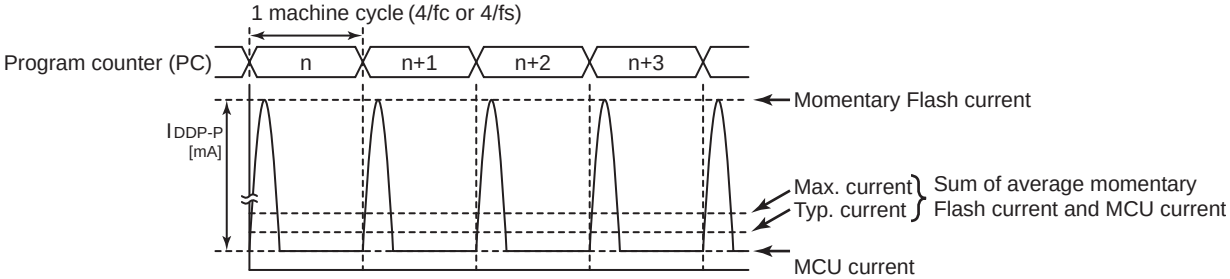
| | | 86CH49<br>86CM49<br>86PM49 | 86CS49 | 86FS49 | 86FS49A | 86FS49B |
|---|---|---|---|---|---|---|
| Operating condition (MCU mode) | Read / Fetch |  |  |  |  |  |
| | | (a) 1.8V to 5.5V (-40 to 85°C) | (a) 2.0V to 5.5V (-40 to 85°C)<br>(b) 1.8V to 2.0V (-20 to 85°C) | (a) 4.5V to 5.5V (-40 to 85°C)<br>(b) 3.0V to 3.6V (-40 to 85°C) | (a) 3.0V to 5.5V (-40 to 85°C)<br>(b) 2.7V to 3.0V (-20 to 85°C) | (a) 2.7V to 5.5V (-40 to 85°C) |
| | Erase / Program | - | - |  | | |
| | | - | - | (a) 4.5V to 5.5V (-10 to 40°C) | | |
| Operating condition (Serial PROM mode) | | - | - |  | | |
| | | - | - | (a) 4.5V to 5.5V (-10 to 40°C) | | |
| Operating Current | | Operating current varies with each product. For details, refer to the datasheet (electrical characteristics) of each product. | | | | |
| | | | | | (Note 4) | |

Note 1: With the 86CS49, the operating temperature (Topr) is -20 °C to 85 °C when the supply voltage VDD is less than 2.0 V.

Note 2: With the 86FS49, the supply voltage VDD is specified as two separate ranges. While the MCU is operating, do not change the supply voltage from range (a) to range (b) or from range (b) to range (a).

Note 3: With the 86FS49A, the operating temperature (Topr) is -20 °C to 85 °C when the supply voltage VDD is less than 3.0 V.

Note 4: With the 86FS49A/B, 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 Flash memory momentarily, as shown in Figure. In this case, the supply current IDD (in NORMAL1, NORMAL2 and SLOW1 modes) is defined as the sum of the average peak current and MCU current.




Intermittent Operation of Flash Memory

# Revision History

| Date | Revision | |
|---|---|---|
| 2006/4/28 | 1 | First Release |
| 2006/10/4 | 2 | Contents Revised |
| 2007/2/2 | 3 | Periodical updating.No change in contents. |
| 2007/2/2 | 4 | Periodical updating.No change in contents. |
| 2007/6/27 | 5 | Contents Revised |

# Table of Contents


Differences among Products

TMP86CS49UG

1.1 Features . . . . . 1
1.2 Pin Assignment . . . . . 3
1.3 Block Diagram . . . . . 4
1.4 Pin Names and Functions . . . . . 5

2. Operational Description

2.1 CPU Core Functions . . . . . 9
2.1.1 Memory Address Map . . . . . 9
2.1.2 Program Memory (MaskROM) . . . . . 9
2.1.3 Data Memory (RAM) . . . . . 9
2.2 System Clock Controller . . . . . 10
2.2.1 Clock Generator . . . . . 10
2.2.2 Timing Generator . . . . . 12
2.2.2.1 Configuration of timing generator
2.2.2.2 Machine cycle
2.2.3 Operation Mode Control Circuit . . . . . 13
2.2.3.1 Single-clock mode
2.2.3.2 Dual-clock mode
2.2.3.3 STOP mode
2.2.4 Operating Mode Control . . . . . 18
2.2.4.1 STOP mode
2.2.4.2 IDLE1/2 mode and SLEEP1/2 mode
2.2.4.3 IDLE0 and SLEEP0 modes (IDLE0, SLEEP0)
2.2.4.4 SLOW mode
2.3 Reset Circuit . . . . . 31
2.3.1 External Reset Input . . . . . 31
2.3.2 Address trap reset . . . . . 32
2.3.3 Watchdog timer reset . . . . . 32
2.3.4 System clock reset . . . . . 32

3. Interrupt Control Circuit

3.1 Interrupt latches (IL23 to IL2) . . . . . 35
3.2 Interrupt enable register (EIR) . . . . . 36
3.2.1 Interrupt master enable flag (IMF) . . . . . 36
3.2.2 Individual interrupt enable flags (EF23 to EF4) . . . . . 37
Note 3: . . . . . 38
3.3 Interrupt Sequence . . . . . 39
3.3.1 Interrupt acceptance processing is packaged as follows. . . . . 39
3.3.2 Saving/restoring general-purpose registers . . . . . 40
3.3.2.1 Using PUSH and POP instructions

3.3.2.2 Using data transfer instructions
3.3.3 Interrupt return ... 41
3.4 Software Interrupt (INTSW) ... 42
3.4.1 Address error detection ... 42
3.4.2 Debugging ... 42
3.5 Undefined Instruction Interrupt (INTUNDEF) ... 42
3.6 Address Trap Interrupt (INTATRAP) ... 42
3.7 External Interrupts ... 43

4. Special Function Register (SFR)

4.1 SFR ... 45
4.2 DBR ... 47

5. I/O Ports

5.1 Port P0 (P07 to P00) ... 50
5.2 Port P1 (P17 to P10) ... 52
5.3 Port P2 (P22 to P20) ... 54
5.4 Port P3 (P37 to P30) (Large Current Port) ... 55
5.5 Port P4 (P47 to P40) ... 56
5.6 Port P5 (P54 to P50) (Large Current Port) ... 58
5.7 Port P6 (P67 to P60) ... 59
5.8 Port P7 (P77 to P70) ... 62

6. Watchdog Timer (WDT)

6.1 Watchdog Timer Configuration ... 65
6.2 Watchdog Timer Control ... 66
6.2.1 Malfunction Detection Methods Using the Watchdog Timer ... 66
6.2.2 Watchdog Timer Enable ... 67
6.2.3 Watchdog Timer Disable ... 68
6.2.4 Watchdog Timer Interrupt (INTWDT) ... 68
6.2.5 Watchdog Timer Reset ... 69
6.3 Address Trap ... 70
6.3.1 Selection of Address Trap in Internal RAM (ATAS) ... 70
6.3.2 Selection of Operation at Address Trap (ATOUT) ... 70
6.3.3 Address Trap Interrupt (INTATRAP) ... 70
6.3.4 Address Trap Reset ... 71

7. Time Base Timer (TBT)

7.1 Time Base Timer ... 73
7.1.1 Configuration ... 73
7.1.2 Control ... 73
7.1.3 Function ... 74
7.2 Divider Output (DVO) ... 75
7.2.1 Configuration ... 75
7.2.2 Control ... 75

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


8.1 Configuration . . . . . 77
8.2 TimerCounter Control . . . . . 78
8.3 Function . . . . . 80
8.3.1 Timer mode . . . . . 80
8.3.2 External Trigger Timer Mode . . . . . 82
8.3.3 Event Counter Mode . . . . . 84
8.3.4 Window Mode . . . . . 85
8.3.5 Pulse Width Measurement Mode . . . . . 86
8.3.6 Programmable Pulse Generate (PPG) Output Mode . . . . . 89


## 9. 16-Bit Timer/Counter2 (TC2)


9.1 Configuration . . . . . 93
9.2 Control . . . . . 94
9.3 Function . . . . . 95
9.3.1 Timer mode . . . . . 95
9.3.2 Event counter mode . . . . . 97
9.3.3 Window mode . . . . . 97


## 10. 8-Bit TimerCounter (TC3, TC4)


10.1 Configuration . . . . . 99
10.2 TimerCounter Control . . . . . 100
10.3 Function . . . . . 105
10.3.1 8-Bit Timer Mode (TC3 and 4) . . . . . 105
10.3.2 8-Bit Event Counter Mode (TC3, 4) . . . . . 106
10.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC3, 4) . . . . . 106
10.3.4 8-Bit Pulse Width Modulation (PWM) Output Mode (TC3, 4) . . . . . 109
10.3.5 16-Bit Timer Mode (TC3 and 4) . . . . . 111
10.3.6 16-Bit Event Counter Mode (TC3 and 4) . . . . . 112
10.3.7 16-Bit Pulse Width Modulation (PWM) Output Mode (TC3 and 4) . . . . . 112
10.3.8 16-Bit Programmable Pulse Generate (PPG) Output Mode (TC3 and 4) . . . . . 115
10.3.9 Warm-Up Counter Mode . . . . . 117
10.3.9.1 Low-Frequency Warm-up Counter Mode (NORMAL1 → NORMAL2 → SLOW2 → SLOW1)
10.3.9.2 High-Frequency Warm-Up Counter Mode (SLOW1 → SLOW2 → NORMAL2 → NORMAL1)


## 11. 8-Bit TimerCounter (TC5, TC6)


11.1 Configuration . . . . . 119
11.2 TimerCounter Control . . . . . 120
11.3 Function . . . . . 125
11.3.1 8-Bit Timer Mode (TC5 and 6) . . . . . 125
11.3.2 8-Bit Event Counter Mode (TC5, 6) . . . . . 126
11.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC5, 6) . . . . . 126
11.3.4 8-Bit Pulse Width Modulation (PWM) Output Mode (TC5, 6) . . . . . 129
11.3.5 16-Bit Timer Mode (TC5 and 6) . . . . . 131
11.3.6 16-Bit Event Counter Mode (TC5 and 6) . . . . . 132
11.3.7 16-Bit Pulse Width Modulation (PWM) Output Mode (TC5 and 6) . . . . . 132
11.3.8 16-Bit Programmable Pulse Generate (PPG) Output Mode (TC5 and 6) . . . . . 135
11.3.9 Warm-Up Counter Mode . . . . . 137
11.3.9.1 Low-Frequency Warm-up Counter Mode

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

## 12. Asynchronous Serial interface (UART1 )

12.1 Configuration . . . . . 139
12.2 Control . . . . . 140
12.3 Transfer Data Format . . . . . 142
12.4 Transfer Rate. . . . . 143
12.5 Data Sampling Method . . . . . 143
12.6 STOP Bit Length . . . . . 144
12.7 Parity . . . . . 144
12.8 Transmit/Receive Operation . . . . . 144
12.8.1 Data Transmit Operation ..... 144
12.8.2 Data Receive Operation ..... 144
12.9 Status Flag . . . . . 145
12.9.1 Parity Error ..... 145
12.9.2 Framing Error ..... 145
12.9.3 Overrun Error ..... 145
12.9.4 Receive Data Buffer Full ..... 146
12.9.5 Transmit Data Buffer Empty ..... 146
12.9.6 Transmit End Flag ..... 147

## 13. Asynchronous Serial interface (UART2 )

13.1 Configuration . . . . . 149
13.2 Control . . . . . 150
13.3 Transfer Data Format . . . . . 152
13.4 Transfer Rate. . . . . 153
13.5 Data Sampling Method . . . . . 153
13.6 STOP Bit Length . . . . . 154
13.7 Parity . . . . . 154
13.8 Transmit/Receive Operation . . . . . 154
13.8.1 Data Transmit Operation ..... 154
13.8.2 Data Receive Operation ..... 154
13.9 Status Flag . . . . . 155
13.9.1 Parity Error ..... 155
13.9.2 Framing Error ..... 155
13.9.3 Overrun Error ..... 155
13.9.4 Receive Data Buffer Full ..... 156
13.9.5 Transmit Data Buffer Empty ..... 156
13.9.6 Transmit End Flag ..... 157

## 14. Synchronous Serial Interface (SIO1)

14.1 Configuration . . . . . 159
14.2 Control . . . . . 160
14.3 Function . . . . . 162
14.3.1 Serial clock ..... 162
14.3.1.1 Clock source
14.3.1.2 Shift edge
14.3.2 Transfer bit direction ..... 164
14.3.2.1 Transmit mode

14.3.2.2 Receive mode
14.3.2.3 Transmit/receive mode
14.3.3 Transfer modes ... 165
14.3.3.1 Transmit mode
14.3.3.2 Receive mode
14.3.3.3 Transmit/receive mode

## 15. Synchronous Serial Interface (SIO2)

15.1 Configuration ... 177
15.2 Control ... 178
15.3 Function ... 180
15.3.1 Serial clock ... 180
15.3.1.1 Clock source
15.3.1.2 Shift edge
15.3.2 Transfer bit direction ... 182
15.3.2.1 Transmit mode
15.3.2.2 Receive mode
15.3.2.3 Transmit/receive mode
15.3.3 Transfer modes ... 183
15.3.3.1 Transmit mode
15.3.3.2 Receive mode
15.3.3.3 Transmit/receive mode

## 16. Serial Bus Interface(I2C Bus) Ver.-D (SBI)

16.1 Configuration ... 195
16.2 Control ... 195
16.3 Software Reset ... 195
16.4 The Data Format in the I2C Bus Mode ... 196
16.5 I2C Bus Control ... 197
16.5.1 Acknowledgement mode specification ... 199
16.5.1.1 Acknowledgment mode (ACK = “1”)
16.5.1.2 Non-acknowledgment mode (ACK = “0”)
16.5.2 Number of transfer bits ... 200
16.5.3 Serial clock ... 200
16.5.3.1 Clock source
16.5.3.2 Clock synchronization
16.5.4 Slave address and address recognition mode specification ... 201
16.5.5 Master/slave selection ... 201
16.5.6 Transmitter/receiver selection ... 201
16.5.7 Start/stop condition generation ... 202
16.5.8 Interrupt service request and cancel ... 202
16.5.9 Setting of I2C bus mode ... 203
16.5.10 Arbitration lost detection monitor ... 203
16.5.11 Slave address match detection monitor ... 204
16.5.12 GENERAL CALL detection monitor ... 204
16.5.13 Last received bit monitor ... 204
16.6 Data Transfer of I2C Bus ... 205
16.6.1 Device initialization ... 205
16.6.2 Start condition and slave address generation ... 205
16.6.3 1-word data transfer ... 205
16.6.3.1 When the MST is “1” (Master mode)
16.6.3.2 When the MST is “0” (Slave mode)
16.6.4 Stop condition generation ... 208
16.6.5 Restart ... 209

## 17. 10-bit AD Converter (ADC)

17.1 Configuration ... 211

17.2 Register configuration . . . . . 212
17.3 Function . . . . . 215
17.3.1 Software Start Mode . . . . . 215
17.3.2 Repeat Mode . . . . . 215
17.3.3 Register Setting . . . . . 216
17.4 STOP/SLOW Modes during AD Conversion . . . . . 217
17.5 Analog Input Voltage and AD Conversion Result . . . . . 218
17.6 Precautions about AD Converter . . . . . 219
17.6.1 Restrictions for AD Conversion interrupt (INTADC) usage . . . . . 219
17.6.2 Analog input pin voltage range . . . . . 219
17.6.3 Analog input shared pins . . . . . 219
17.6.4 Noise Countermeasure . . . . . 219

18. Key-on Wakeup (KWU)

18.1 Configuration . . . . . 221
18.2 Control . . . . . 221
18.3 Function . . . . . 221

19. Input/Output Circuit

19.1 Control pins . . . . . 223
19.2 Input/Output Ports . . . . . 224

20. Electrical Characteristics

20.1 Absolute Maximum Ratings . . . . . 227
20.2 Recommended Operating Conditions . . . . . 228
20.3 DC Characteristics . . . . . 229
20.4 AD Characteristics . . . . . 230
20.5 AC Characteristics . . . . . 232
20.6 Recommended Oscillating Conditions . . . . . 233
20.7 Handling Precaution . . . . . 234

21. Package Dimensions

This is a technical document that describes the operating functions and electrical specifications of the 8-bit microcontroller series TLCS-870/C (LSI).

CMOS 8-Bit Microcontroller

# TMP86CS49UG

| Product No. | ROM (MaskROM) | RAM | Package | FLASH MCU | Emulation Chip |
|---|---|---|---|---|---|
| TMP86CS49UG | 61440 bytes | 2048 bytes | LQFP64-P-1010-0.50D | TMP86FS49AUG | TMP86C949XB |

## 1.1 Features

1. 8-bit single chip microcomputer TLCS-870/C series
   - Instruction execution time :

     0.25 μs (at 16 MHz)

     122 μs (at 32.768 kHz)
   - 132 types & 731 basic instructions
2. 24interrupt sources (External : 5 Internal : 19)
3. Input / Output ports (56 pins)

   Large current output: 13pins (Typ. 20mA), LED direct drive
4. Watchdog Timer
5. Prescaler
   - Time base timer
   - Divider output function
6. 16-bit timer counter: 1 ch
   - Timer, External trigger, Window, Pulse width measurement,

     Event counter, Programmable pulse generate (PPG) modes
7. 16-bit timer counter: 1 ch
   - Timer, Event counter, Window modes
8. 8-bit timer counter : 4 ch
   - Timer, Event counter, Programmable divider output (PDO),

     Pulse width modulation (PWM) output,

060116EBP

- The information contained herein is subject to change without notice. 021023_D
- 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. 021023_A
- 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 malfunctionor 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. 021023_B
- The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations. 060106_Q
- 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. 021023_C
- The products described in this document are subject to the foreign exchange and foreign trade laws. 021023_E
- 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. 030619_S

Programmable pulse generation (PPG) modes

9. 8-bit UART : 2 ch
10. High-Speed SIO: 2ch
11. Serial Bus Interface($I^2C$ Bus): 1ch
12. 10-bit successive approximation type AD converter
    - Analog input: 16 ch
13. Key-on wakeup : 4 ch
14. Clock operation

    Single clock mode

    Dual clock mode
15. 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 interruputs(CPU restarts).

    IDLE2 mode: CPU stops and peripherals operate using high and low frequency clock. Release by interruputs. (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 interruput.(CPU restarts).

    SLEEP2 mode: CPU stops and peripherals operate using high and low frequency clock. Release by interruput.
16. Wide operation voltage:

    4.5 V to 5.5 V at 16MHz /32.768 kHz

    2.7 V to 5.5 V at 8 MHz /32.768 kHz

    1.8 V to 5.5 V at 4.2MHz /32.768 kHz

## 1.2 Pin Assignment

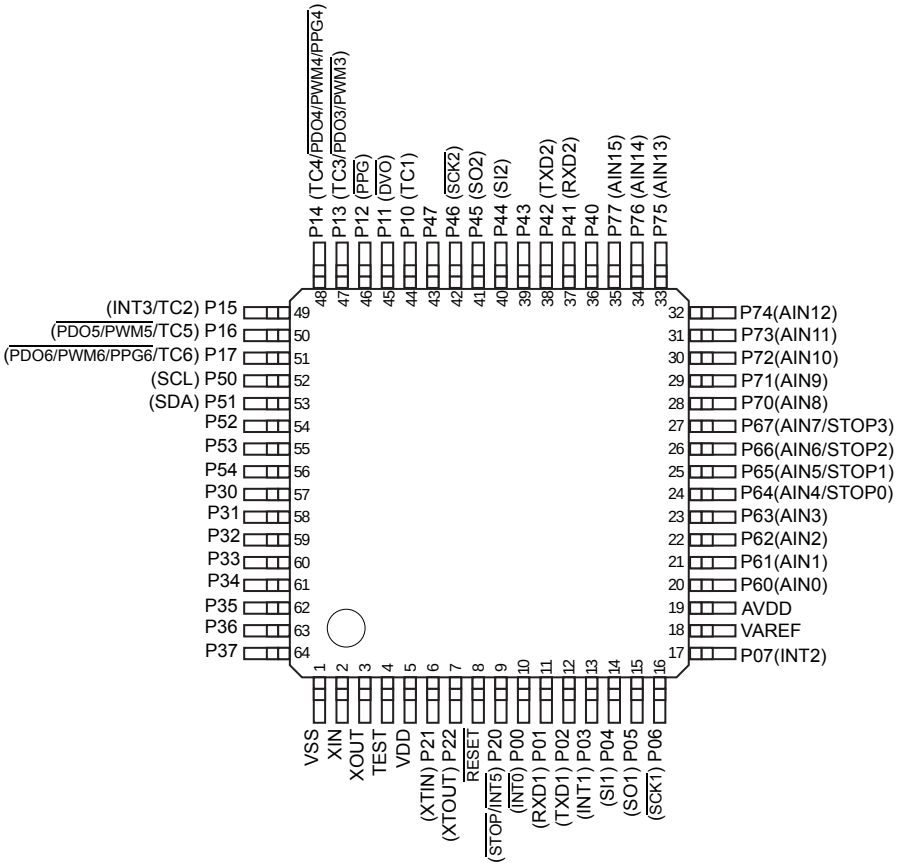



Figure 1-1 Pin Assignment

## 1.3 Block Diagram

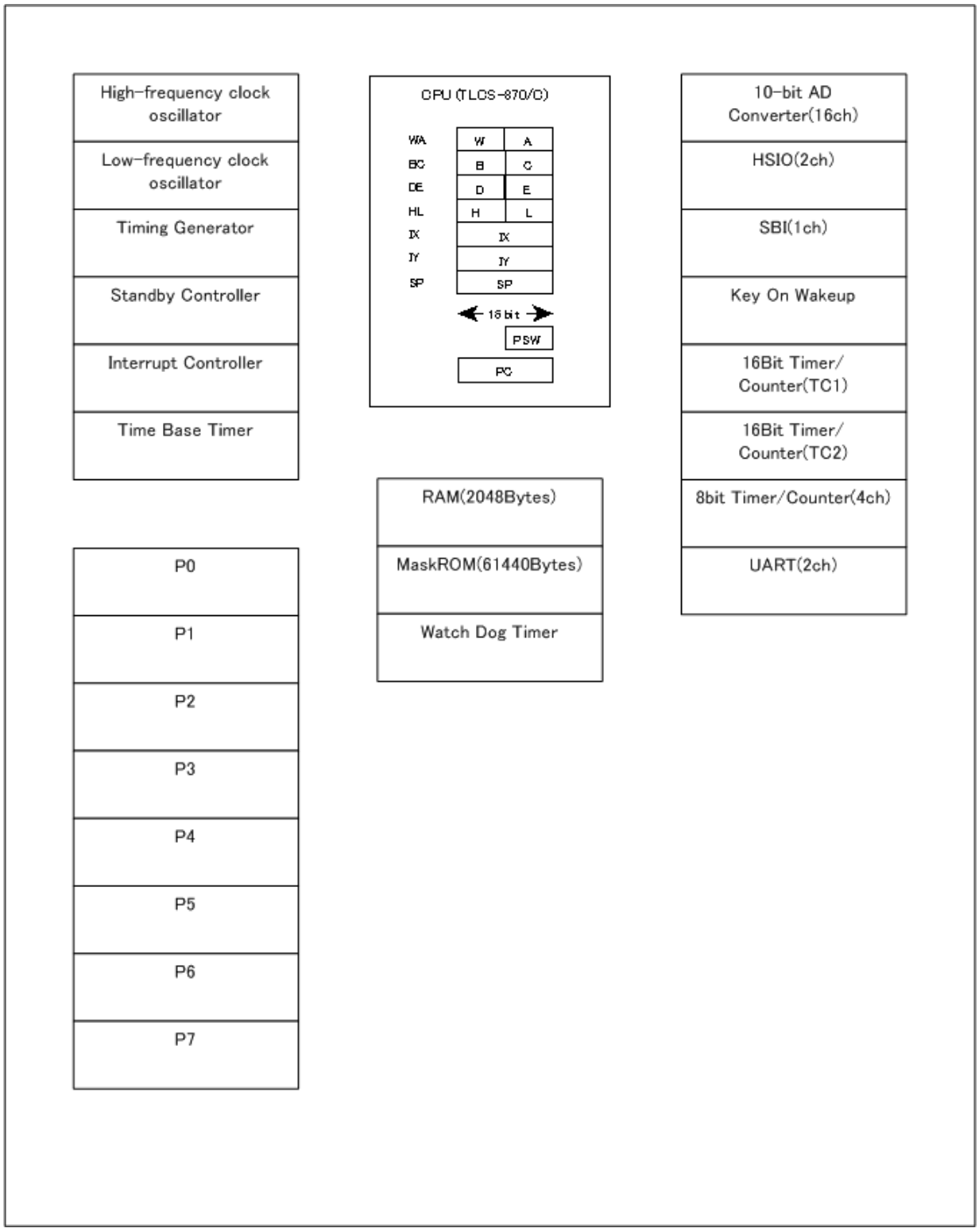



Figure 1-2 Block Diagram

## 1.4 Pin Names and Functions

Table 1-1 Pin Names and Functions(1/3)

| Pin Name | Pin Number | Input/Output | Functions |
|---|---|---|---|
| P07<br>INT2 | 17 | IO<br>I | PORT07<br>External interrupt 2 input |
| P06<br>$\overline{SCK1}$ | 16 | IO<br>IO | PORT06<br>Serial clock input/output 1 |
| P05<br>SO1 | 15 | IO<br>O | PORT05<br>Serial data output 1 |
| P04<br>SI1 | 14 | IO<br>I | PORT04<br>Serial data input 1 |
| P03<br>INT1 | 13 | IO<br>I | PORT03<br>External interrupt 1 input |
| P02<br>TXD1 | 12 | IO<br>O | PORT02<br>UART data output 1 |
| P01<br>RXD1 | 11 | IO<br>I | PORT01<br>UART data input 1 |
| P00<br>$\overline{INT0}$ | 10 | IO<br>I | PORT00<br>External interrupt 0 input |
| P17<br>TC6<br>$\overline{PDO6/PWM6/PPG6}$ | 51 | IO<br>I<br>O | PORT17<br>TC6 input<br>PDO6/PWM6/PPG6 output |
| P16<br>TC5<br>$\overline{PDO5/PWM5}$ | 50 | IO<br>I<br>O | PORT16<br>TC5 input<br>PDO5/PWM5 output |
| P15<br>TC2<br>INT3 | 49 | IO<br>I<br>I | PORT15<br>TC2 input<br>External interrupt 3 input |
| P14<br>TC4<br>$\overline{PDO4/PWM4/PPG4}$ | 48 | IO<br>I<br>O | PORT14<br>TC4 input<br>PDO4/PWM4/PPG4 output |
| P13<br>TC3<br>$\overline{PDO3/PWM3}$ | 47 | IO<br>I<br>O | PORT13<br>TC3 input<br>PDO3/PWM3 output |
| P12<br>$\overline{PPG}$ | 46 | IO<br>O | PORT12<br>PPG output |
| P11<br>$\overline{DVO}$ | 45 | IO<br>O | PORT11<br>Divider Output |
| P10<br>TC1 | 44 | IO<br>I | PORT10<br>TC1 input |
| P22<br>XTOUT | 7 | IO<br>O | PORT22<br>Resonator connecting pins(32.768kHz) for inputting external clock |
| P21<br>XTIN | 6 | IO<br>I | PORT21<br>Resonator connecting pins(32.768kHz) for inputting external clock |
| P20<br>$\overline{INT5}$<br>$\overline{STOP}$ | 9 | IO<br>I<br>I | PORT20<br>External interrupt 5 input<br>STOP mode release signal input |

Table 1-1 Pin Names and Functions(2/3)

| Pin Name | Pin Number | Input/Output | Functions |
|---|---|---|---|
| P37 | 64 | IO | PORT37 |
| P36 | 63 | IO | PORT36 |
| P35 | 62 | IO | PORT35 |
| P34 | 61 | IO | PORT34 |
| P33 | 60 | IO | PORT33 |
| P32 | 59 | IO | PORT32 |
| P31 | 58 | IO | PORT31 |
| P30 | 57 | IO | PORT30 |
| P47 | 43 | IO | PORT47 |
| P46<br>$\overline{SCK2}$ | 42 | IO<br>IO | PORT46<br>Serial clock input/output 2 |
| P45<br>SO2 | 41 | IO<br>O | PORT45<br>Serial data output 2 |
| P44<br>SI2 | 40 | IO<br>I | PORT44<br>Serial data input 2 |
| P43 | 39 | IO | PORT43 |
| P42<br>TXD2 | 38 | IO<br>O | PORT42<br>UART data output 2 |
| P41<br>RXD2 | 37 | IO<br>I | PORT41<br>UART data input 2 |
| P40 | 36 | IO | PORT40 |
| P54 | 56 | IO | PORT54 |
| P53 | 55 | IO | PORT53 |
| P52 | 54 | IO | PORT52 |
| P51<br>SDA | 53 | IO<br>IO | PORT51<br>I2C bus data |
| P50<br>SCL | 52 | IO<br>IO | PORT50<br>I2C bus clock |
| P67<br>AIN7<br>STOP3 | 27 | IO<br>I<br>I | PORT67<br>Analog Input7<br>STOP3 input |
| P66<br>AIN6<br>STOP2 | 26 | IO<br>I<br>I | PORT66<br>Analog Input6<br>STOP2 input |
| P65<br>AIN5<br>STOP1 | 25 | IO<br>I<br>I | PORT65<br>Analog Input5<br>STOP1 input |
| P64<br>AIN4<br>STOP0 | 24 | IO<br>I<br>I | PORT64<br>Analog Input4<br>STOP0 input |
| P63<br>AIN3 | 23 | IO<br>I | PORT63<br>Analog Input3 |
| P62<br>AIN2 | 22 | IO<br>I | PORT62<br>Analog Input2 |

Table 1-1 Pin Names and Functions(3/3)

| Pin Name | Pin Number | Input/Output | Functions |
|---|---|---|---|
| P61<br>AIN1 | 21 | IO<br>I | PORT61<br>Analog Input1 |
| P60<br>AIN0 | 20 | IO<br>I | PORT60<br>Analog Input0 |
| P77<br>AIN15 | 35 | IO<br>I | PORT77<br>Analog Input15 |
| P76<br>AIN14 | 34 | IO<br>I | PORT76<br>Analog Input14 |
| P75<br>AIN13 | 33 | IO<br>I | PORT75<br>Analog Input13 |
| P74<br>AIN12 | 32 | IO<br>I | PORT74<br>Analog Input12 |
| P73<br>AIN11 | 31 | IO<br>I | PORT73<br>Analog Input11 |
| P72<br>AIN10 | 30 | IO<br>I | PORT72<br>Analog Input10 |
| P71<br>AIN9 | 29 | IO<br>I | PORT71<br>Analog Input9 |
| P70<br>AIN8 | 28 | IO<br>I | PORT70<br>Analog Input8 |
| XIN | 2 | I | Resonator connecting pins for high-frequency clock |
| XOUT | 3 | O | Resonator connecting pins for high-frequency clock |
| $\overline{\text{RESET}}$ | 8 | I | Reset signal |
| TEST | 4 | I | Test pin for out-going test. Normally, be fixed to low. |
| VAREF | 18 | I | Analog Base Voltage Input Pin for A/D Conversion |
| AVDD | 19 | I | Analog Power Supply |
| VDD | 5 | I | +5V |
| VSS | 1 | I | 0(GND) |

# 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 TMP86CS49UG memory is composed MaskROM, RAM, DBR(Data buffer register) and SFR(Special function register). They are all mapped in 64-Kbyte address space. Figure 2-1 shows the TMP86CS49UG memory address map.

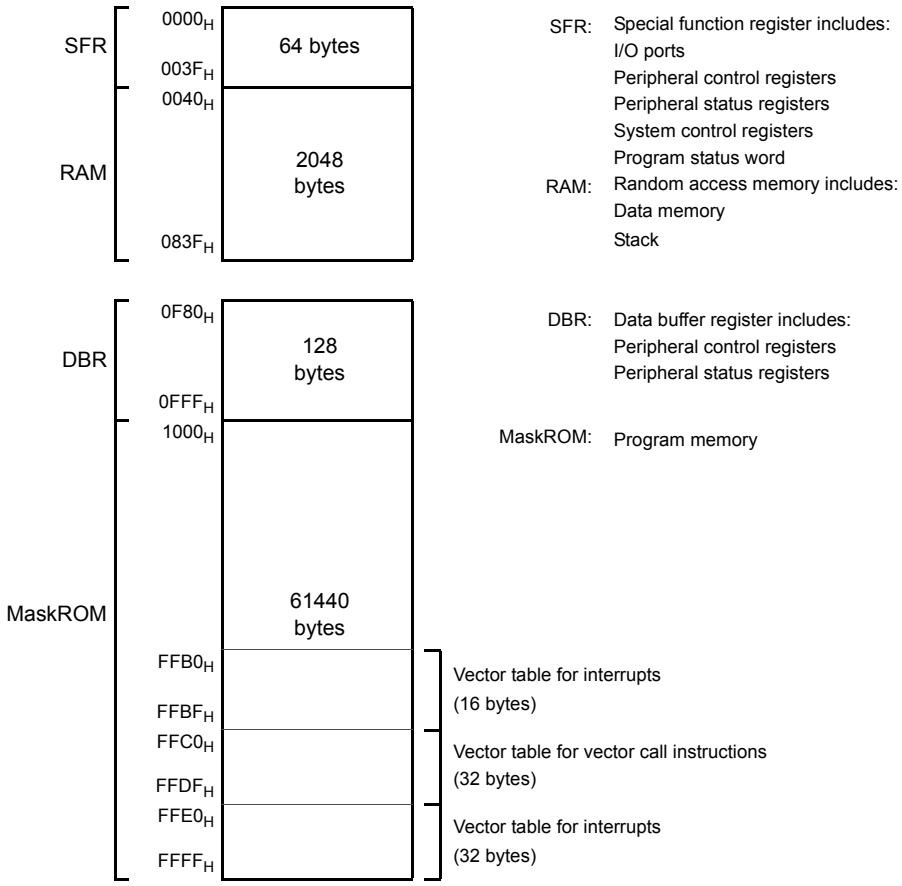



Figure 2-1 Memory Address Map

### 2.1.2 Program Memory (MaskROM)

The TMP86CS49UG has a 61440 bytes (Address 1000H to FFFFH) of program memory (MaskROM ).

### 2.1.3 Data Memory (RAM)

The TMP86CS49UG has 2048 bytes (Address 0040H to 083FH) 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”. (TMP86CS49UG)

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

## 2.2 System Clock Controller

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

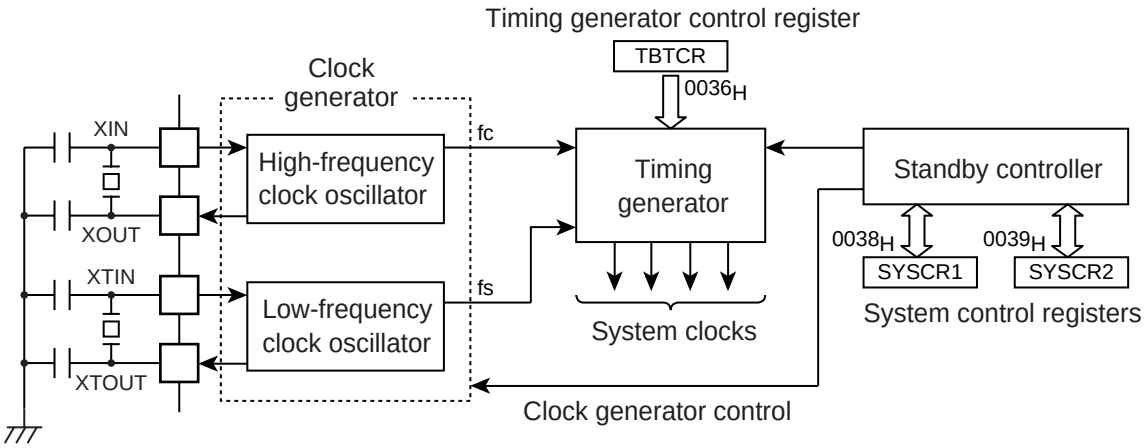



Figure 2-2 System Colck Control

### 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.

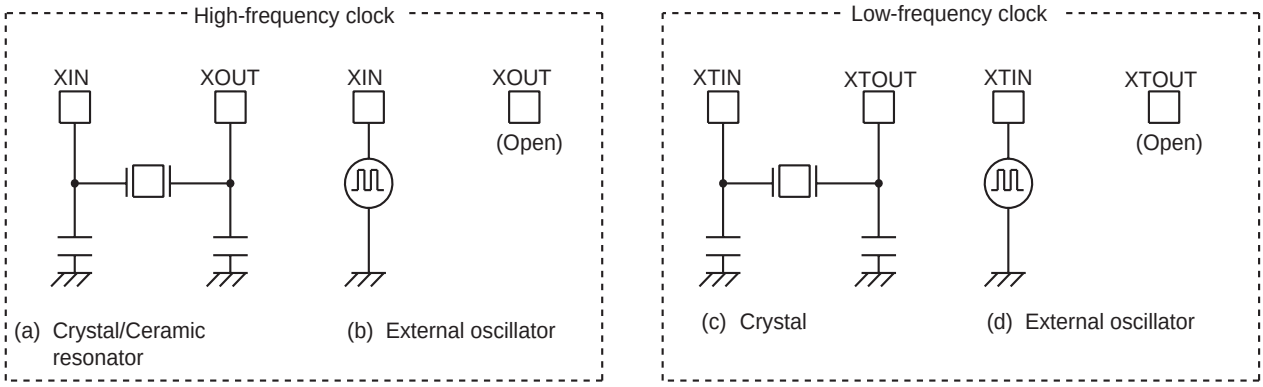



Figure 2-3 Examples of Resonator Connection

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 (fc or fs). The timing generator provides the following functions.

1. Generation of main system clock
2. Generation of divider output ($\overline{DVO}$) 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, SYSCR2<SYSCK> and TBTCR<DV7CK>, that is shown in Figure 2-4. As reset and STOP mode started/canceled, the prescaler and the divider are cleared to "0".

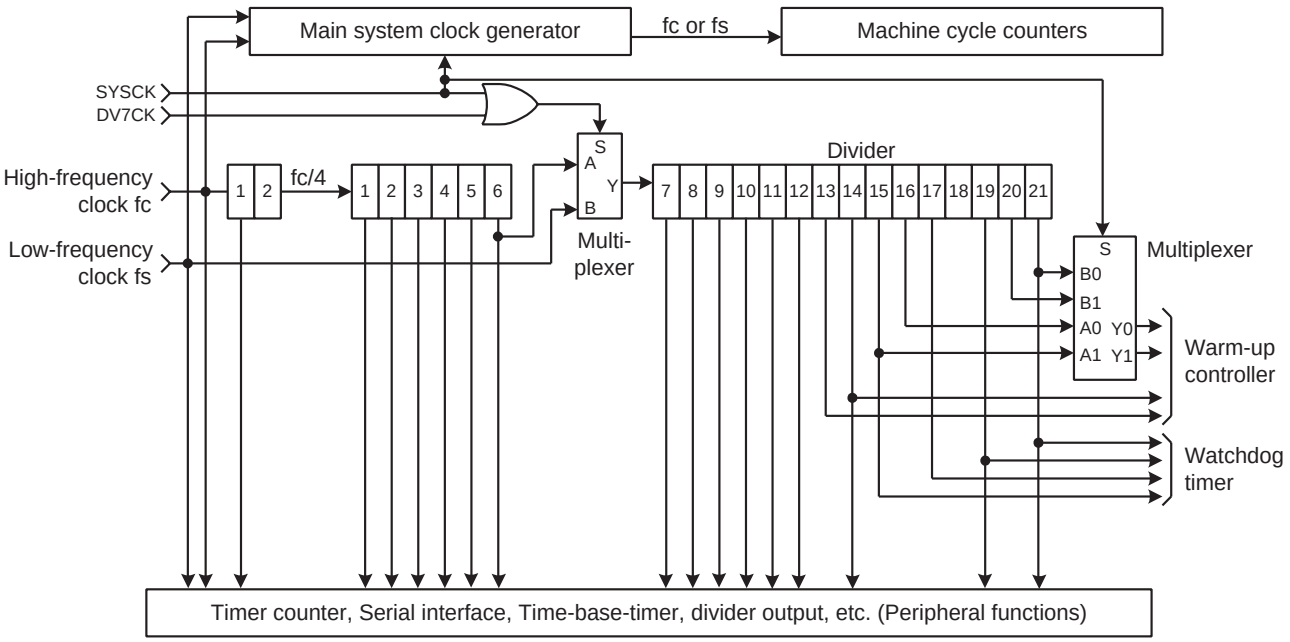



Figure 2-4 Configuration of Timing Generator

Timing Generator Control Register

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

| | | | |
|---|---|---|---|
| DV7CK | Selection of input to the 7th stage of the divider | 0: $fc/2^8$ [Hz]<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 an "machine cycle". There are a total of 10 different types of instructions for the TLCS-870/C Series: Ranging from 1-cycle instructions which require one machine cycle for execution to 10-cycle instructions which require 10 machine cycles for execution. A machine cycle consists of 4 states (S0 to S3), and each state consists of one main system clock.

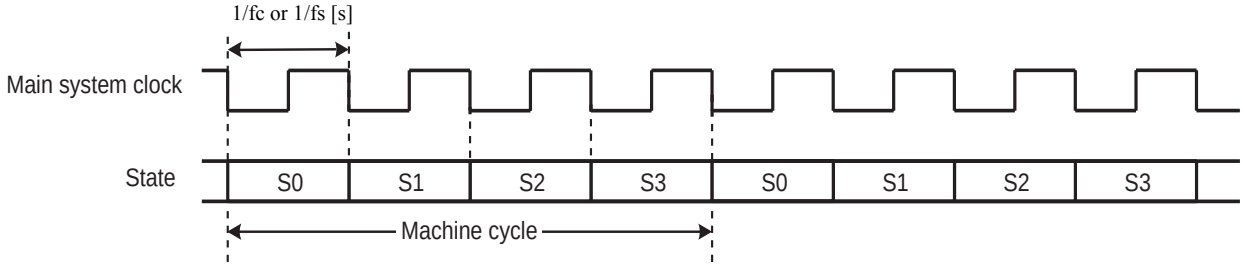



Figure 2-5 Machine Cycle

### 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 TMP86CS49UG 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”, EF7 (TBT interrupt individual enable flag) = “1”, and TBTCR<TBTEN> = “1”, interrupt processing is performed. When IDLE0 mode is entered while TBTCR<TBTEN> = “1”, the INTTBT interrupt latch is set after returning to NORMAL1 mode.

### 2.2.3.2 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/fc [s] in the NORMAL2 and IDLE2 modes, and 4/fs [s] (122 μs at fs = 32.768 kHz) in the SLOW and SLEEP modes.

The TLCS-870/C is placed in the 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>.

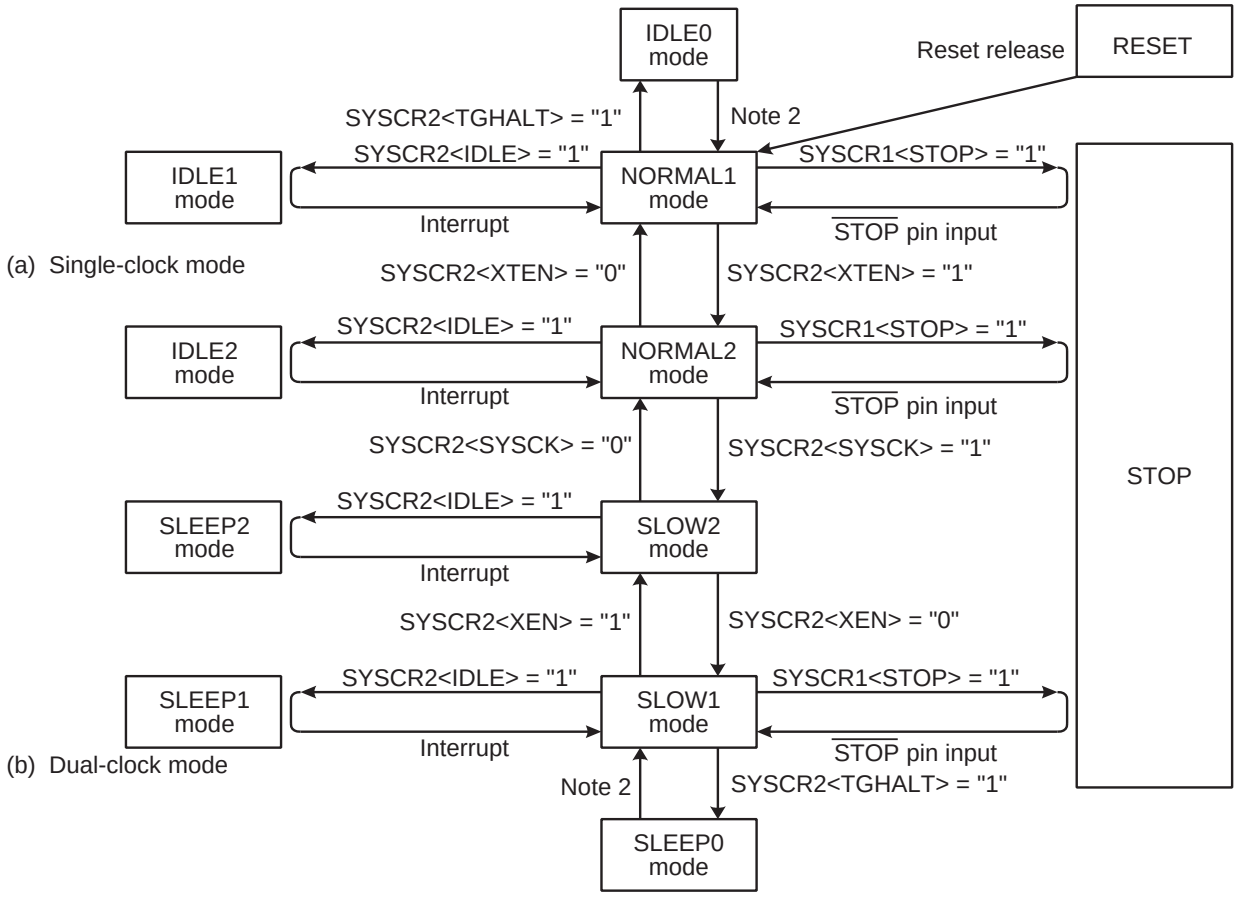
When SLEEP0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

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”, EF7 (TBT interrupt individual enable flag) = “1”, and TBTCR<TBTEN> = “1”, interrupt processing is performed. When SLEEP0 mode is entered while TBTCR<TBTEN> = “1”, the INTTBT interrupt latch is set after returning to SLOW1 mode.

### 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<TBTCK> setting.

Figure 2-6 Operating Mode Transition Diagram

Table 2-1 Operating Mode and Conditions

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

## System Control Register 1

SYSCR1 (0038H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| STOP | RELM | RETM | OUTEN | WUT | | | |

(Initial value: 0000 00**)

| | | | | | |
|---|---|---|---|---|---|
| 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 |
| | | 00 | $3 \times 2^{16}$/fc | $3 \times 2^{13}$/fs | |
| | | 01 | $2^{16}$/fc | $2^{13}$/fs | |
| | | 10 | $3 \times 2^{14}$/fc | $3 \times 2^{6}$/fs | |
| | | 11 | $2^{14}$/fc | $2^{6}$/fs | |

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

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

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

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

Note 5: As the hardware becomes STOP mode under OUTEN = “0”, input value is fixed to “0”; therefore it may cause 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 $\overline{\text{STOP}}$ 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 (0039H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| 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.<br>(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<TBTCK>.

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

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

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

## 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 (STOP3 to STOP0) 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 (STOP3 to STOP0) 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 (STOP3 to STOP0). 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 STOP3 to STOP0 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 STOP3 to STOP0 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 STOP3 to STOP0 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 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
```

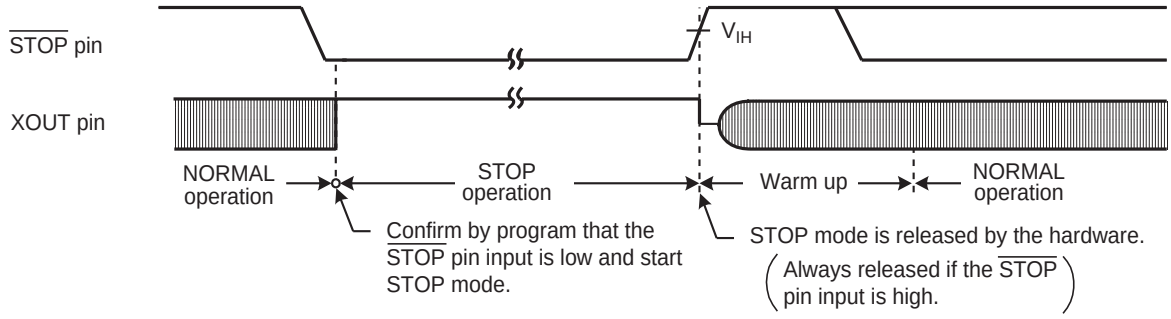



Figure 2-7 Level-sensitive Release Mode

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 $\overline{\text{STOP}}$ pin. In the edge-sensitive release mode, STOP mode is started even when the $\overline{\text{STOP}}$ pin input is high level. Do not use any STOP3 to STOP0 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
```

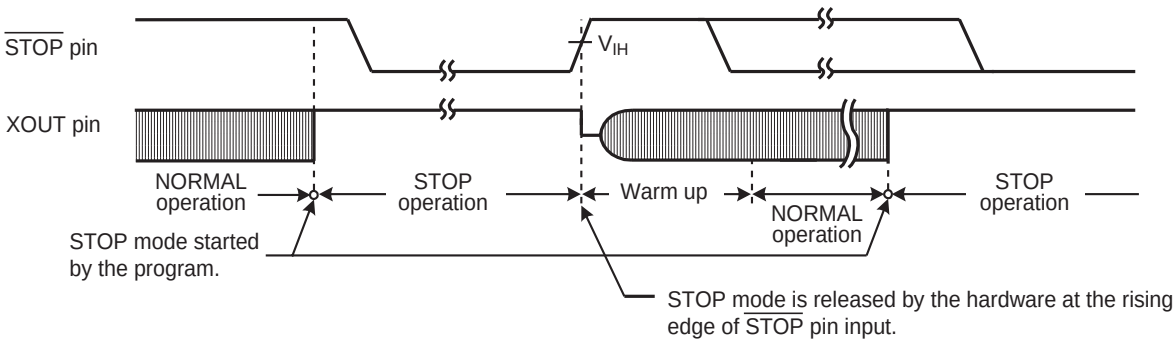



Figure 2-8 Edge-sensitive Release Mode

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 2-2 Warm-up Time Example (at fc = 16.0 MHz, fs = 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.

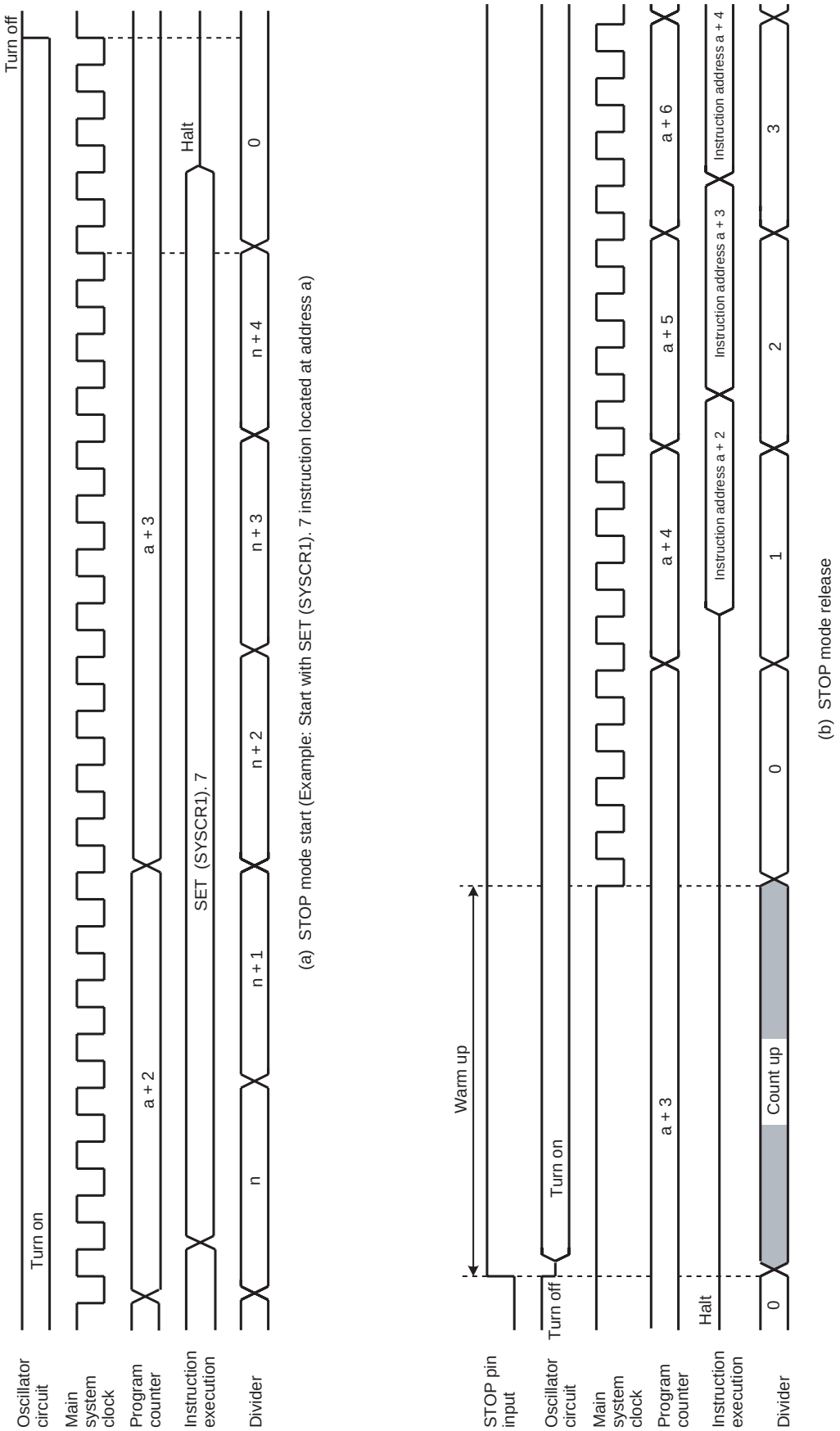



Figure 2-9 STOP Mode Start/Release

### 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.

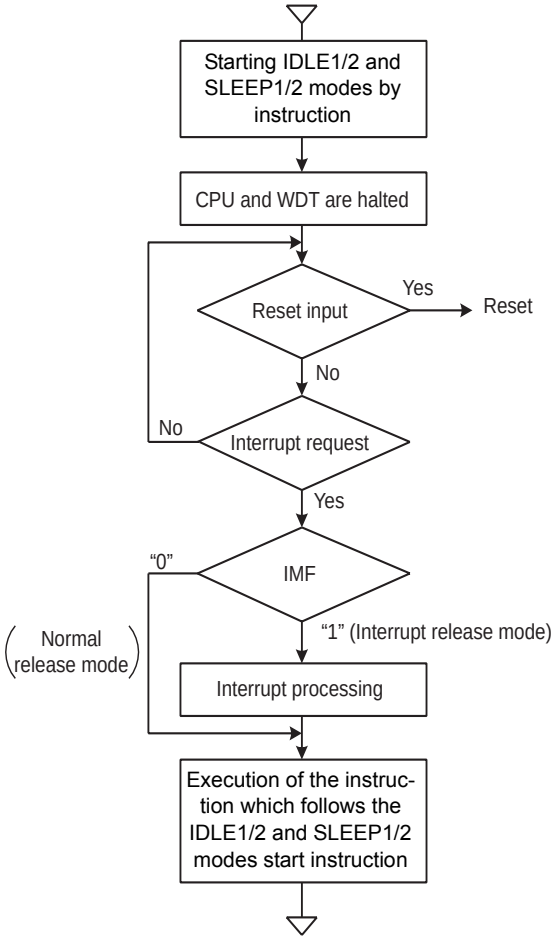



Figure 2-10 IDLE1/2 and SLEEP1/2 Modes

- 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 $\overline{\text{RESET}}$ 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.

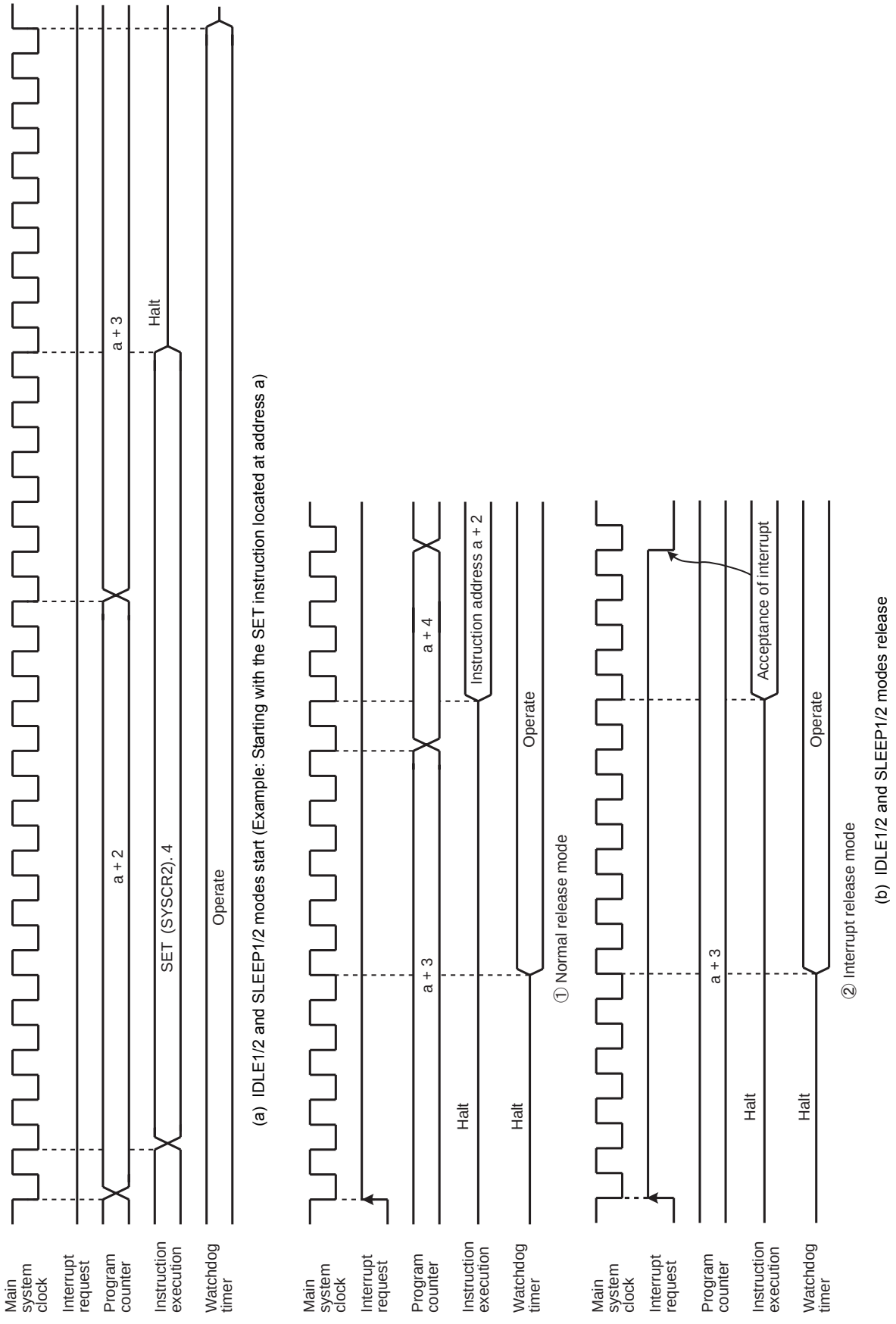



Figure 2-11 IDLE1/2 and SLEEP1/2 Modes Start/Release

### 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 (TBTCR). 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.

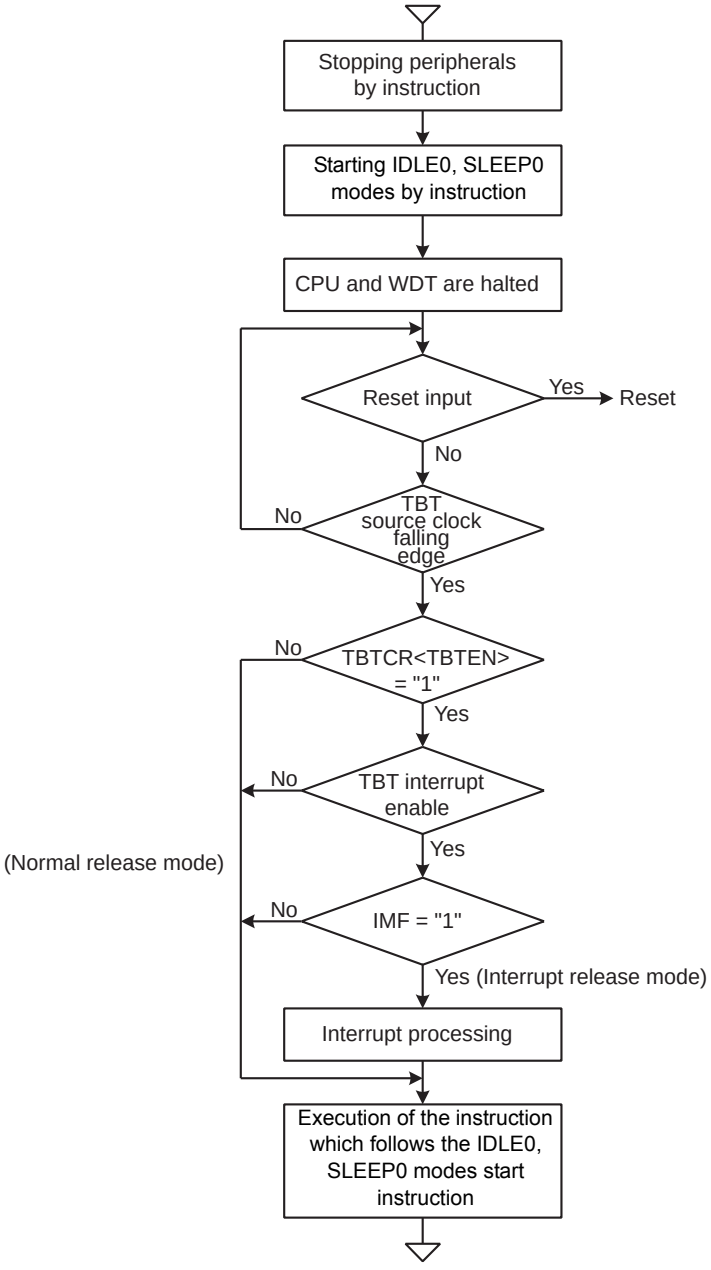



Figure 2-12 IDLE0 and SLEEP0 Modes

- 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 $\overline{\text{RESET}}$ 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•EF7•TBTCR<TBTEN> = "0")

IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK>. After the falling edge is detected, the program operation is resumed from the instruction following the IDLE0 and SLEEP0 modes start instruction. 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•EF7•TBTCR<TBTEN> = "1")

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

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

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

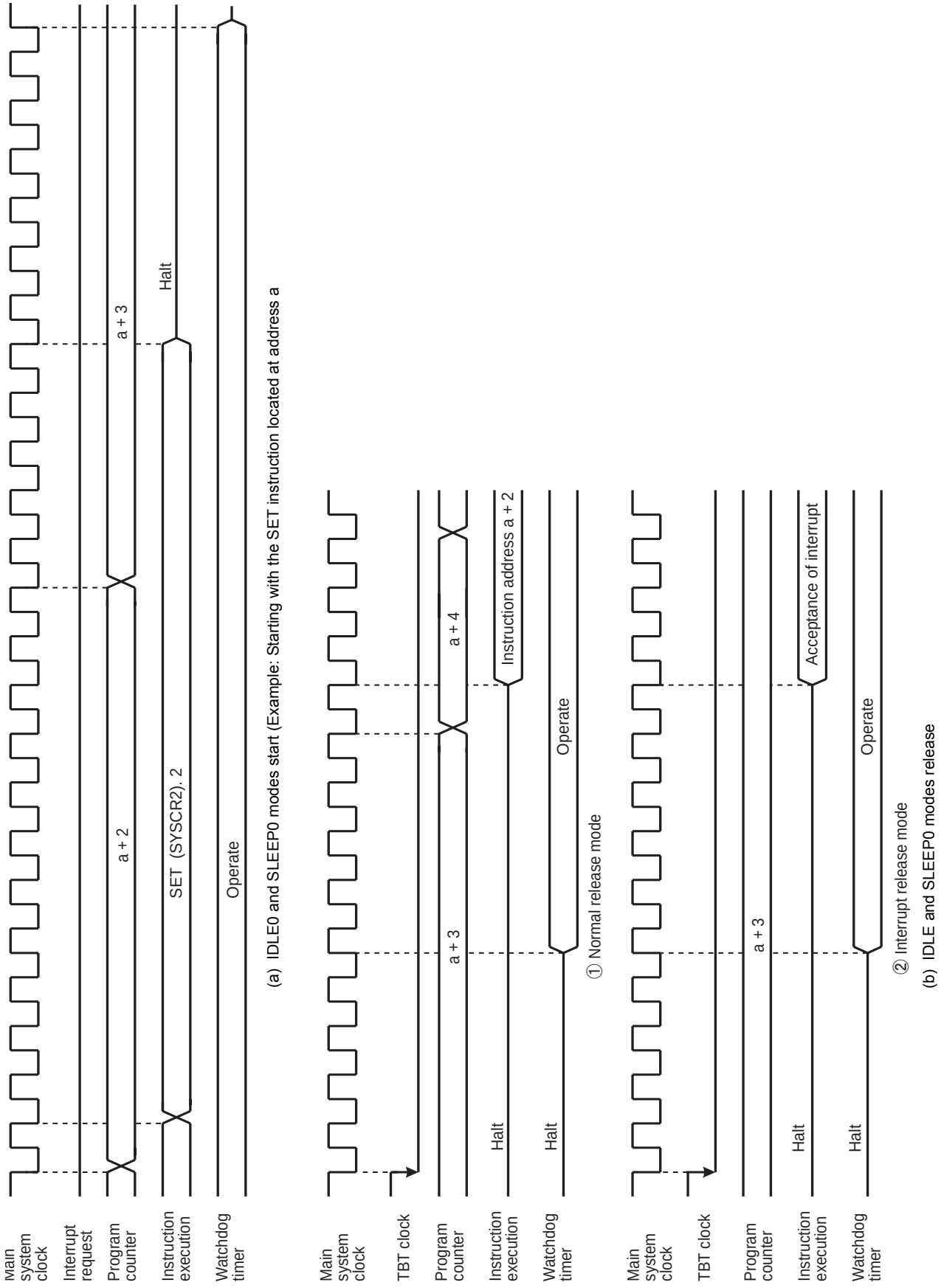



Figure 2-13 IDLE0 and SLEEP0 Modes Start/Release

### 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)
```

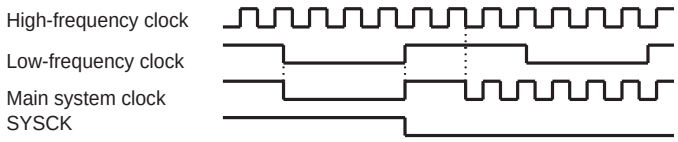
Example 2 :Switching to the SLOW1 mode after low-frequency clock has stabilized.

```
        SET     (SYSCR2). 6     ; SYSCR2<XTEN> ← 1
        LD      (TC5CR), 43H    ; Sets mode for TC6, 5 (16-bit mode, fs for source)
        LD      (TC6CR), 05H    ; Sets warming-up counter mode
        LDW     (TTREG5), 8000H ; Sets warm-up time (Depend on oscillator accompanied)
        DI                      ; IMF ← 0
        SET     (EIRE). 2       ; Enables INTTC6
        EI                      ; IMF ← 1
        SET     (TC6CR). 3      ; Starts TC6, 5
         :
PINTTC6: CLR    (TC6CR). 3      ; Stops TC6, 5
        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
         :
VINTTC6: DW     PINTTC6         ; INTTC6 vector table
```

(2) Switching from SLOW1 mode to NORMAL2 mode

First, set SYSCR2<XEN> to turn on the high-frequency oscillation. When time for stabilization (Warm up) has been taken by the timer/counter (TC6,TC5), clear SYSCR2<SYSCK> to switch the main system clock to the high-frequency clock.
SLOW mode can also be released by inputting low level on the $\overline{\text{RESET}}$ pin. After releasing reset, the operation mode is started from NORMAL1 mode.

Note: After SYSCK is cleared to "0", executing the instructions is continiued 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      (TC5CR), 63H        ; Sets mode for TC6, 5 (16-bit mode, fc for source)
                LD      (TC6CR), 05H        ; Sets warming-up counter mode
                LD      (TTREG6), 0F8H      ; Sets warm-up time
                DI                          ; IMF ← 0
                SET     (EIRE). 2           ; Enables INTTC6
                EI                          ; IMF ← 1
                SET     (TC6CR). 3          ; Starts TC6, 5
                 :
PINTTC6:        CLR     (TC6CR). 3          ; Stops TC6, 5
                CLR     (SYSCR2). 5         ; SYSCR2<SYSCK> ← 0
                                              (Switches the main system clock to the high-frequency clock)
                RETI
                 :
VINTTC6:        DW      PINTTC6             ; INTTC6 vector table
```

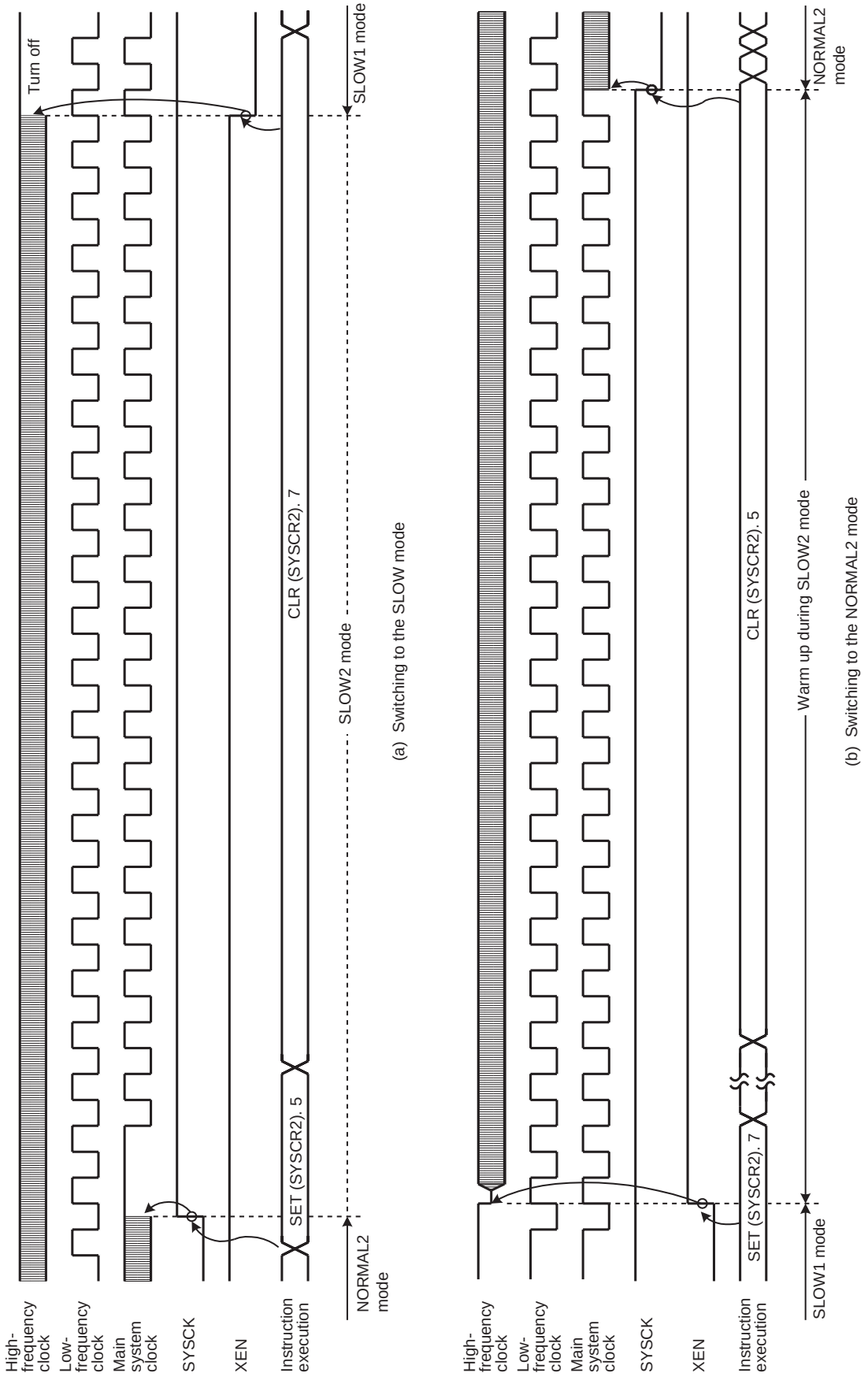



Figure 2-14 Switching between the NORMAL2 and SLOW Modes

## 2.3 Reset Circuit

The TMP86CS49UG 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/fc[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/fc[s] (1.5μs at 16.0 MHz) when power is turned on.

Table 2-3 shows on-chip hardware initialization by reset action.

Table 2-3 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/fc [s]) with the power supply voltage within the operating voltage range and oscillation stable, a reset is applied and the internal state is initialized.

When 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.

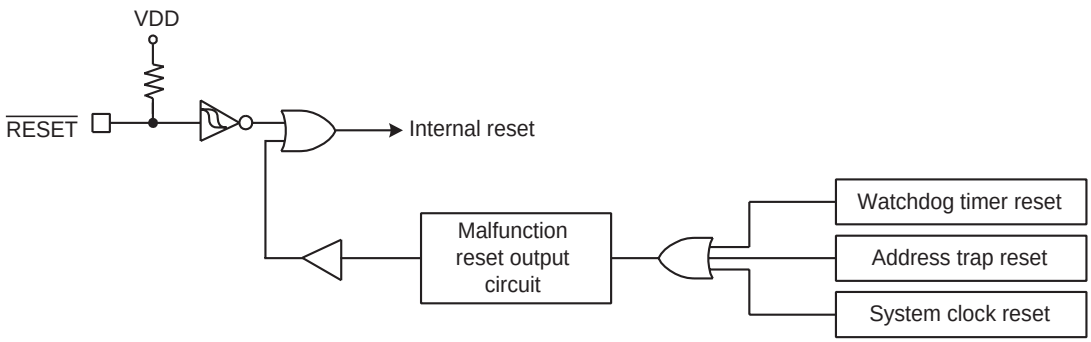
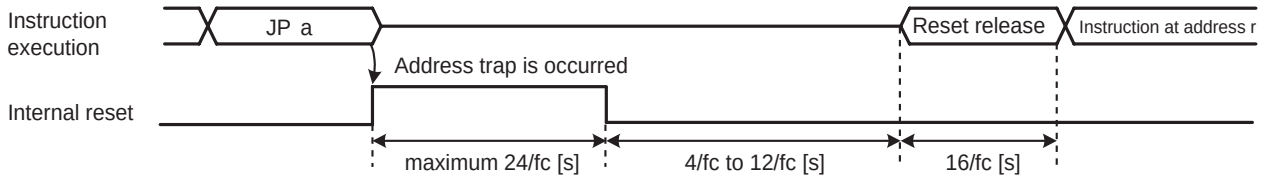



Figure 2-15 Reset Circuit

### 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/fc[s] (1.5μs 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, DBR 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 Section “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/fc (1.5 μs at 16.0 MHz).

# 3. Interrupt Control Circuit

The TMP86CS49UG has a total of 24 interrupt sources excluding reset. 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 | $\overline{\text{INT0}}$ | IMF• EF4 = 1, INT0EN = 1 | IL4 | FFF6 | 5 |
| Internal | INTTC1 | IMF• EF5 = 1 | IL5 | FFF4 | 6 |
| External | INT1 | IMF• EF6 = 1 | IL6 | FFF2 | 7 |
| Internal | INTTBT | IMF• EF7 = 1 | IL7 | FFF0 | 8 |
| External | INT2 | IMF• EF8 = 1 | IL8 | FFEE | 9 |
| Internal | INTTC4 | IMF• EF9 = 1 | IL9 | FFEC | 10 |
| Internal | INTTC3 | IMF• EF10 = 1 | IL10 | FFEA | 11 |
| Internal | INTSBI | IMF• EF11 = 1 | IL11 | FFE8 | 12 |
| External | INT3 | IMF• EF12 = 1 | IL12 | FFE6 | 13 |
| Internal | INTSIO1 | IMF• EF13 = 1 | IL13 | FFE4 | 14 |
| Internal | INTSIO2 | IMF• EF14 = 1 | IL14 | FFE2 | 15 |
| Internal | INTADC | IMF• EF15 = 1 | IL15 | FFE0 | 16 |
| Internal | INTRXD1 | IMF• EF16 = 1 | IL16 | FFBE | 17 |
| Internal | INTTXD1 | IMF• EF17 = 1 | IL17 | FFBC | 18 |
| Internal | INTTC6 | IMF• EF18 = 1 | IL18 | FFBA | 19 |
| Internal | INTTC5 | IMF• EF19 = 1 | IL19 | FFB8 | 20 |
| Internal | INTRXD2 | IMF• EF20 = 1 | IL20 | FFB6 | 21 |
| Internal | INTTXD2 | IMF• EF21 = 1 | IL21 | FFB4 | 22 |
| Internal | INTTC2 | IMF• EF22 = 1 | IL22 | FFB2 | 23 |
| External | $\overline{\text{INT5}}$ | IMF• EF23 = 1 | IL23 | FFB0 | 24 |

Note 1: To use the address trap interrupt (INTATRAP), clear WDTCR1<ATOUT> to "0" (It is set for the "reset request" after reset is cancelled). For details, see "Address Trap".

Note 2: To use the watchdog timer interrupt (INTWDT), clear WDTCR1<WDTOUT> to "0" (It is set for the "Reset request" after reset is released). For details, see "Watchdog Timer".

Note 3: If an INTADC interrupt request is generated while an interrupt with priority lower than the interrupt latch IL15 (INTADC) is being accepted, the INTADC interrupt latch may be cleared without the INTADC interrupt being processed. For details, refer to the corresponding notes in the chapter on the AD converter.

## 3.1 Interrupt latches (IL23 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 002EH, 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 latchess

```
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 002CH, 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 (EF23 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 (EF23 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

(Initial value: 00000000 000000**)

ILH,ILL (003DH, 003CH)

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| IL15 | IL14 | IL13 | IL12 | IL11 | IL10 | IL9 | IL8 | IL7 | IL6 | IL5 | IL4 | IL3 | IL2 | | |

ILH (003DH) ILL (003CH)

(Initial value: 00000000)

ILE (002EH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IL23 | IL22 | IL21 | IL20 | IL19 | IL18 | IL17 | IL16 |

ILE (002EH)

| | | at RD | at WR | |
|---|---|---|---|---|
| IL23 to IL2 | Interrupt latches | 0: No interrupt request<br>1: Interrupt request | 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

(Initial value: 00000000 0000***0)

EIRH,EIRL (003BH, 003AH)

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| EF15 | EF14 | EF13 | EF12 | EF11 | EF10 | EF9 | EF8 | EF7 | EF6 | EF5 | EF4 | | | | IMF |

EIRH (003BH) EIRL (003AH)

(Initial value: 00000000)

EIRE (002CH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EF23 | EF22 | EF21 | EF20 | EF19 | EF18 | EF17 | EF16 |

EIRE (002CH)

| | | | |
|---|---|---|---|
| EF23 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".

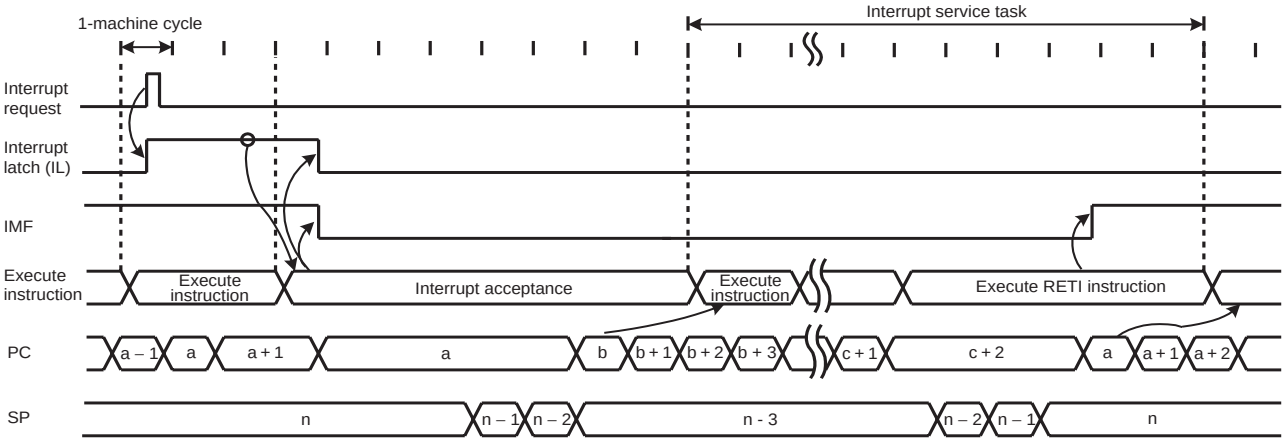
## 3.3 Interrupt Sequence

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.

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

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

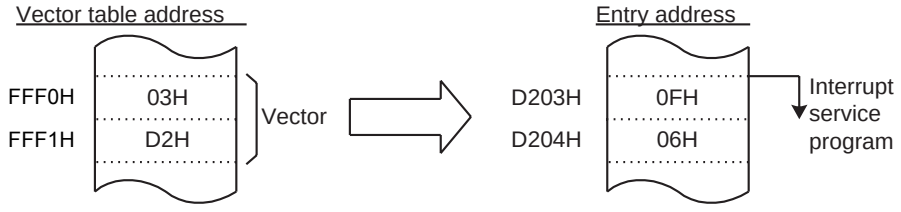



Figure 3-2 Vector table address,Entry address

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.

## 3.3.2 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.

### 3.3.2.1 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
```

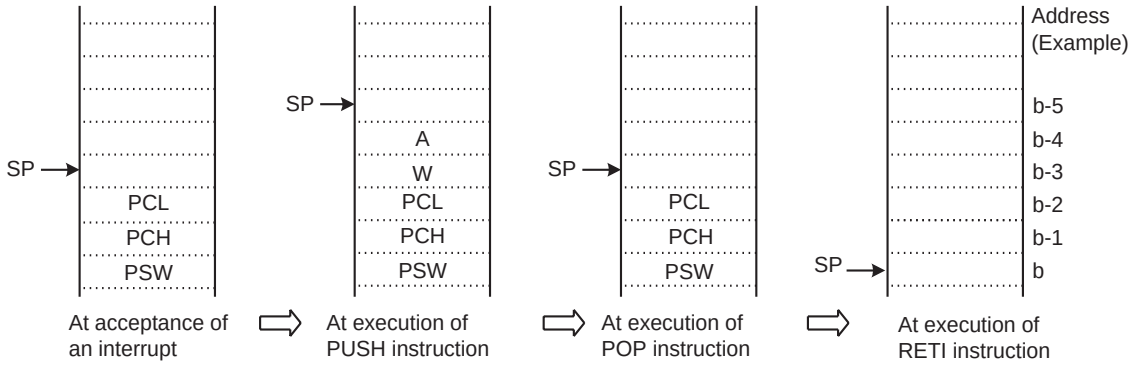


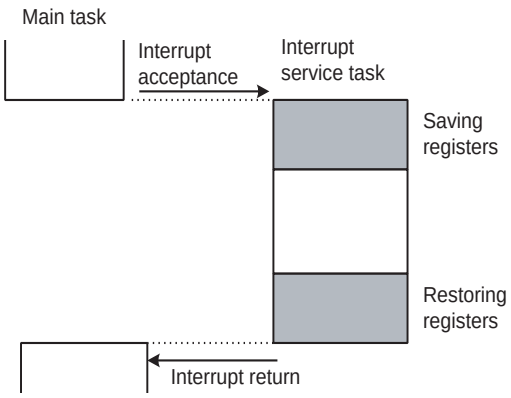

Figure 3-3 Save/store register using PUSH and POP instructions

### 3.3.2.2 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 3-4 Saving/Restoring General-purpose Registers under Interrupt Processing

### 3.3.3 Interrupt return

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

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

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.

## 3.4 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.

### 3.4.1 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.

### 3.4.2 Debugging

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

## 3.5 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.

## 3.6 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).

## 3.7 External Interrupts

The TMP86CS49UG 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 to INT3. The $\overline{\text{INT0}}$/P00 pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.

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

| Source | Pin | Enable Conditions | Release Edge | Digital Noise Reject |
|---|---|---|---|---|
| INT0 | $\overline{\text{INT0}}$ | IMF • FE4 • 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 • FE6 = 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. |
| INT2 | INT2 | IMF • FE8 = 1 | Falling edge<br>or<br>Rising edge | 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. |
| INT3 | INT3 | IMF • FE12 = 1 | Falling edge<br>or<br>Rising edge | 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 • FE23 = 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 INT0EN = "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

| EINTCR | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (0037H) | INT1NC | INT0EN | - | - | INT3ES | INT2ES | INT1ES | | (Initial value: 00** 000*) |

| | | | |
|---|---|---|---|
| INT1NC | Noise reject time select | 0: Pulses of less than 63/fc [s] are eliminated as noise<br>1: Pulses of less than 15/fc [s] are eliminated as noise | R/W |
| INT0EN | P00/$\overline{\text{INT0}}$ pin configuration | 0: P00 input/output port<br>1: $\overline{\text{INT0}}$ pin (Port P00 should be set to an input mode) | R/W |
| INT3 ES | INT3 edge select | 0: Rising edge<br>1: Falling edge | R/W |
| INT2 ES | INT2 edge select | 0: Rising edge<br>1: Falling edge | R/W |
| INT1 ES | INT1 edge select | 0: Rising edge<br>1: Falling edge | R/W |

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

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

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

# 4. Special Function Register (SFR)

The TMP86CS49UG 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 TMP86CS49UG.

## 4.1 SFR

| Address | Read | Write |
|---|---|---|
| 0000H | P0DR | |
| 0001H | P1DR | |
| 0002H | P2DR | |
| 0003H | P3DR | |
| 0004H | P4DR | |
| 0005H | P5DR | |
| 0006H | P6DR | |
| 0007H | P7DR | |
| 0008H | P0OUTCR | |
| 0009H | P1CR | |
| 000AH | P4OUTCR | |
| 000BH | P0PRD | - |
| 000CH | P2PRD | - |
| 000DH | P3PRD | - |
| 000EH | P4PRD | - |
| 000FH | P5PRD | - |
| 0010H | TC1DRAL | |
| 0011H | TC1DRAH | |
| 0012H | TC1DRBL | |
| 0013H | TC1DRBH | |
| 0014H | TTREG3 | |
| 0015H | TTREG4 | |
| 0016H | TTREG5 | |
| 0017H | TTREG6 | |
| 0018H | PWREG3 | |
| 0019H | PWREG4 | |
| 001AH | PWREG5 | |
| 001BH | PWREG6 | |
| 001CH | ADCCR1 | |
| 001DH | ADCCR2 | |
| 001EH | ADCDR2 | - |
| 001FH | ADCDR1 | - |
| 0020H | SIO1CR | |
| 0021H | SIO1SR | - |
| 0022H | SIO1RDB | SIO1TDB |
| 0023H | TC2CR | |
| 0024H | TC2DRL | |
| 0025H | TC2DRH | |

| Address | Read | Write |
|---|---|---|
| 0026H | TC1CR | |
| 0027H | TC3CR | |
| 0028H | TC4CR | |
| 0029H | TC5CR | |
| 002AH | TC6CR | |
| 002BH | SIO2RDB | SIO2TDB |
| 002CH | EIRE | |
| 002DH | Reserved | |
| 002EH | ILE | |
| 002FH | Reserved | |
| 0030H | Reserved | |
| 0031H | SIO2CR | |
| 0032H | SIO2SR | - |
| 0033H | Reserved | |
| 0034H | - | WDTCR1 |
| 0035H | - | WDTCR2 |
| 0036H | TBTCR | |
| 0037H | EINTCR | |
| 0038H | SYSCR1 | |
| 0039H | SYSCR2 | |
| 003AH | EIRL | |
| 003BH | EIRH | |
| 003CH | ILL | |
| 003DH | ILH | |
| 003EH | Reserved | |
| 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 | Read | Write |
|---|---|---|
| 0F80H | Reserved | |
| 0F81H | Reserved | |
| 0F82H | Reserved | |
| 0F83H | Reserved | |
| 0F84H | Reserved | |
| 0F85H | Reserved | |
| 0F86H | Reserved | |
| 0F87H | Reserved | |
| 0F88H | Reserved | |
| 0F89H | Reserved | |
| 0F8AH | Reserved | |
| 0F8BH | Reserved | |
| 0F8CH | Reserved | |
| 0F8DH | Reserved | |
| 0F8EH | Reserved | |
| 0F8FH | Reserved | |
| 0F90H | SBISRA | SBICRA |
| 0F91H | SBIDBR | |
| 0F92H | - | I2CAR |
| 0F93H | SBISRB | SBICRB |
| 0F94H | Reserved | |
| 0F95H | UART1SR | UART1CR1 |
| 0F96H | - | UART1CR2 |
| 0F97H | RD1BUF | TD1BUF |
| 0F98H | UART2SR | UART2CR1 |
| 0F99H | - | UART2CR2 |
| 0F9AH | RD2BUF | TD2BUF |
| 0F9BH | P6CR1 | |
| 0F9CH | P6CR2 | |
| 0F9DH | P7CR1 | |
| 0F9EH | P7CR2 | |
| 0F9FH | - | STOPCR |

| Address | Read | Write |
|---|---|---|
| 0FA0H | Reserved | |
| : : | : : | |
| 0FBFH | Reserved | |

| Address | Read | Write |
|---|---|---|
| 0FC0H | Reserved | |
| : : | : : | |
| 0FDFH | Reserved | |

| Address | Read | Write |
|---|---|---|
| 0FE0H | Reserved | |
| : : | : : | |
| 0FFFH | Reserved | |

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

Note 2: – ; Cannot be accessed.

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

# 5. I/O Ports

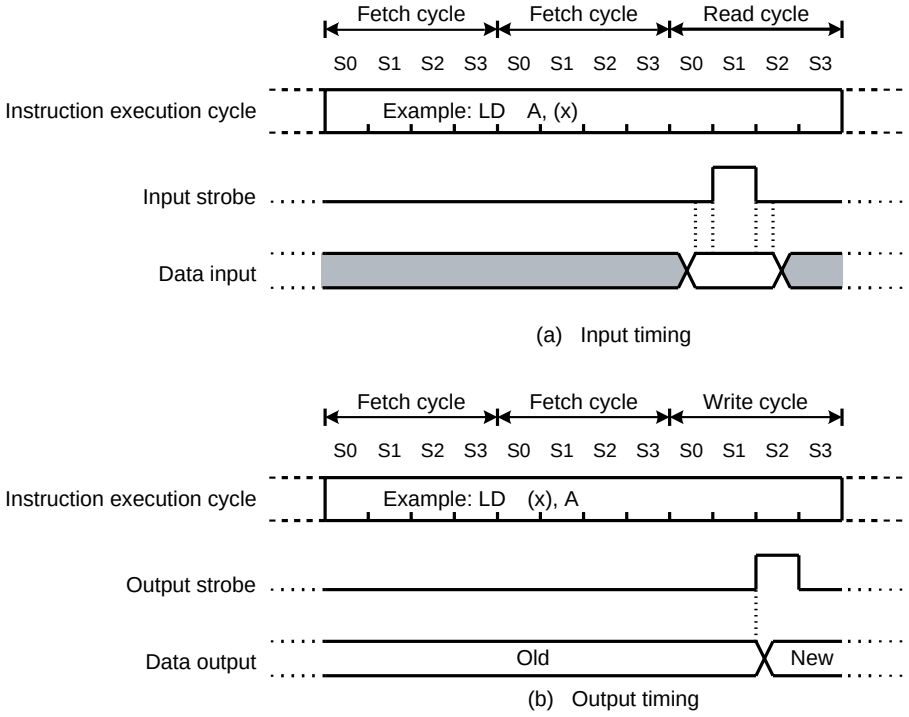
The TMP86CS49UG has 8 parallel input/output ports (56 pins) as follows.

| | Primary Function | Secondary Functions |
|---|---|---|
| Port P0 | 8-bit I/O port | External interrupt, serial interface input/output, UART input/output. |
| Port P1 | 8-bit I/O port | External interrupt, timer counter input/output, divider output. |
| Port P2 | 3-bit I/O port | Low-frequency resonator connections, external interrupt input, STOP mode release signal input. |
| Port P3 | 8-bit I/O port | |
| Port P4 | 8-bit I/O port | Serial interface input/output and UART input/output. |
| Port P5 | 5-bit I/O port | Serial bus interface input/output. |
| Port P6 | 8-bit I/O port | Analog input and key-on wakeup input. |
| Port P7 | 8-bit I/O port | Analog input. |

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 Port P0 (P07 to P00)

Port P0 is an 8-bit input/output port.

Port P0 is also used as an external interrupt input, a serial interface input/output and an UART input/output.

When used as an input port, an external interrupt input, a serial interface input/output and an UART input/output, the corresponding output latch (P0DR) should be set to "1".

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

It can be selected whether output circuit of P0 port is a C-MOS output or a sink open drain individually, by setting P0OUTCR. When a corresponding bit of P0OUTCR is "0". the output circuit is selected to a sink open drain and when a corresponding bit of P0OUTCR is "1", the output circuit is selected to a C-MOS output.

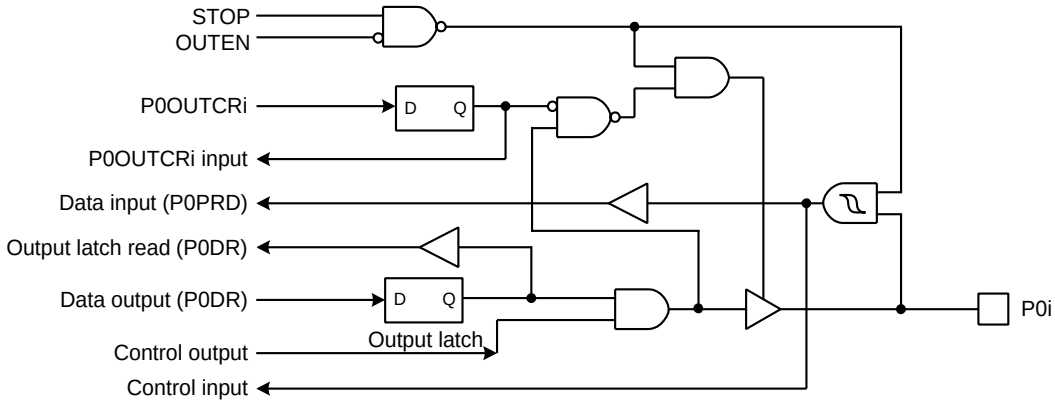
When used as an input port, an external interrupt input, a serial interface input and an UART input, the corresponding output control (P0OUTCR) should be set to "0" after P0DR is set to "1".

P0 port output latch (P0DR) and P0 port terminal input (P0PRD) are located on their respective address.

When read the output latch data, the P0DR should be read. When read the terminal input data, the P0PRD register should be read.

Table 5-1 Register Programming for Multi-function Ports (P07 to P00)

| Function | Programmed Value | |
|---|---|---|
| | P0DR | P0OUTCR |
| Port input, external input, serial interface input or UART input | “1” | “0” |
| Port “0” output | “0” | Programming for each applications |
| Port “1” output, serial interface output or UART output | “1” | |




Note: i = 7 to 0

Figure 5-2 Port 0 and P0OUTCR

P0DR (0000H) R/W

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P07 INT2 | P06 $\overline{SCK1}$ | P05 SO1 | P04 SI1 | P03 INT1 | P02 TXD1 | P01 RXD1 | P00 $\overline{INT0}$ |

(Initial value: 1111 1111)

P0OUTCR (0008H)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P0OUTCR | Port P0 output circuit control (Set for each bit individually) | 0: Sink open-drain output<br>1: C-MOS output | R/W |

P0PRD (000BH) Read only

| P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 |
|---|---|---|---|---|---|---|---|

## 5.2 Port P1 (P17 to P10)

Port P1 is an 8-bit input/output port which can be configured as an input or output in one-bit unit.

Port P1 is also used as a timer/counter input/output, an external interrupt input and a divider output.

Input/output mode is specified by the P1 control register (P1CR).

During reset, the P1CR is initialized to "0" and port P1 becomes an input mode. And the P1DR is initialized to "0".

When used as an input port, a timer/counter input and an external interrupt input, the corresponding bit of P1CR should be set to "0".

When used as an output port, the corresponding bit of P1CR should be set to "1".

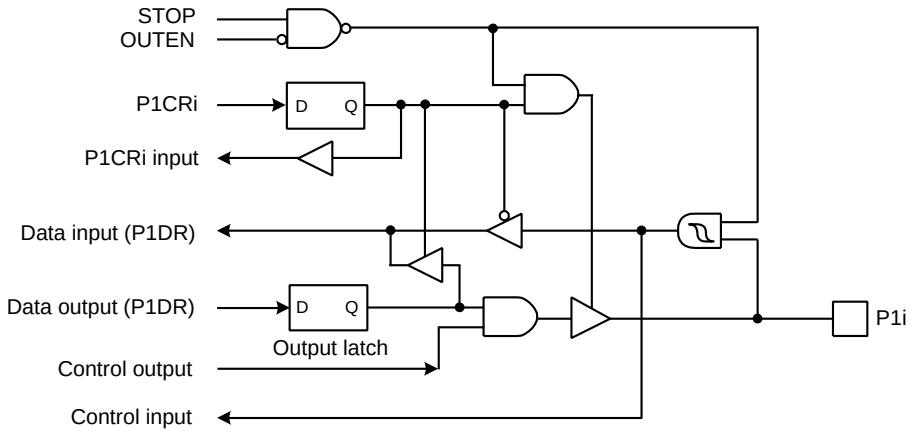
When used as a timer/counter output and a divider output, P1DR is set to "1" beforehand and the corresponding bit of P1CR should be set to "1".

When P1CR is "1", the content of the corresponding output latch is read by reading P1DR.

Table 5-2 Register Programming for Multi-function Ports

| Function | Programmed Value | |
|---|---|---|
| | P1DR | P1CR |
| Port input, timer/counter input or external interrupt input | * | “0” |
| Port “0” output | “0” | “1” |
| Port “1” output, a timer output or a divider output | “1” | “1” |

Note: Asterisk (*) indicates “1” or “0” either of which can be selected.




Note: i = 7 to 0

Figure 5-3 Port 1 and P1CR

Note: The port set to an input mode reads the terminal input data. Therefore, when the input and output modes are used together, the content of the output latch which is specified as input mode might be changed by executing a bit Manipulation instruction.

P1DR (0001H) R/W

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P17<br>TC6<br>$\overline{\text{PWM6}}$<br>$\overline{\text{PDO6}}$<br>$\overline{\text{PPG6}}$ | P16<br>TC5<br>$\overline{\text{PWM5}}$<br>$\overline{\text{PDO5}}$ | P15<br>TC2<br>INT3 | P14<br>TC4<br>$\overline{\text{PWM4}}$<br>$\overline{\text{PDO4}}$<br>$\overline{\text{PPG4}}$ | P13<br>TC3<br>$\overline{\text{PWM3}}$<br>$\overline{\text{PDO3}}$ | P12<br>$\overline{\text{PPG}}$ | P11<br>$\overline{\text{DVO}}$ | P10<br>TC1 |

(Initial value: 0000 0000)

P1CR (0009H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P1CR | I/O control for port P1 (Specified for each bit) | 0: Input mode<br>1: Output mode | R/W |

## 5.3 Port P2 (P22 to P20)

Port P2 is a 3-bit input/output port.

It is also used as an external interrupt, a STOP mode release signal input, and low-frequency crystal oscillator connection pins. When used as an input port or a secondary function pins, respective output latch (P2DR) should be set to "1".

During reset, the P2DR is initialized to "1".

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

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

P2 port output latch (P2DR) and P2 port terminal input (P2PRD) are located on their respective address.

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

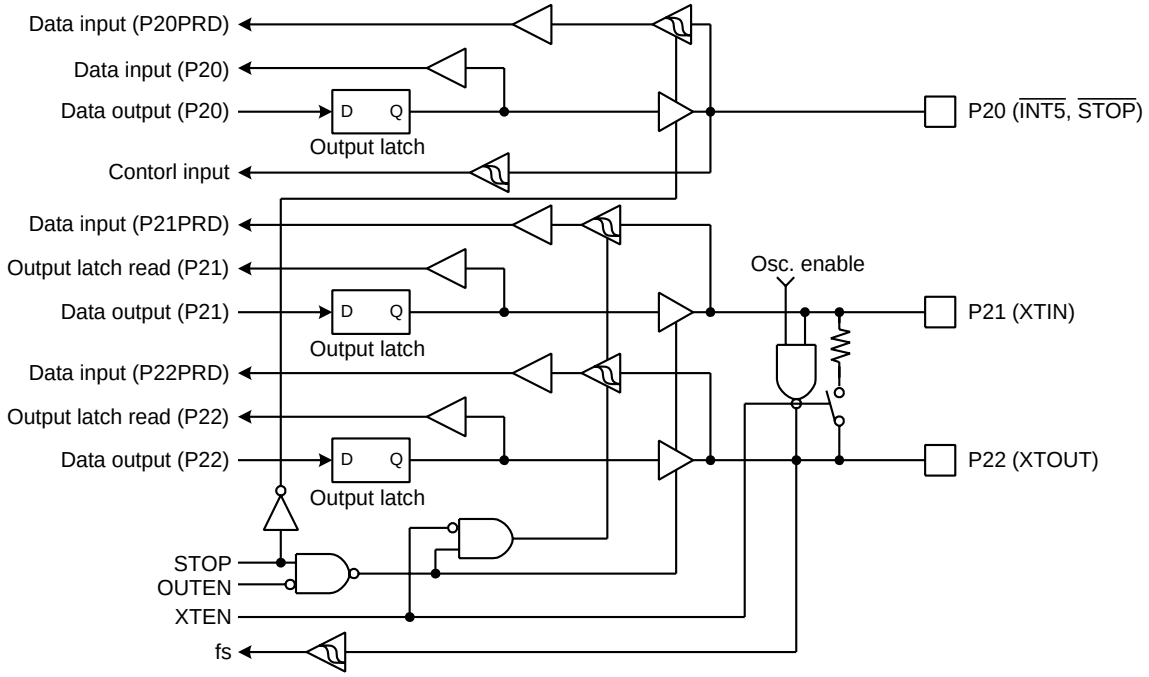



Figure 5-4 Port 2

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

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

## 5.4 Port P3 (P37 to P30) (Large Current Port)

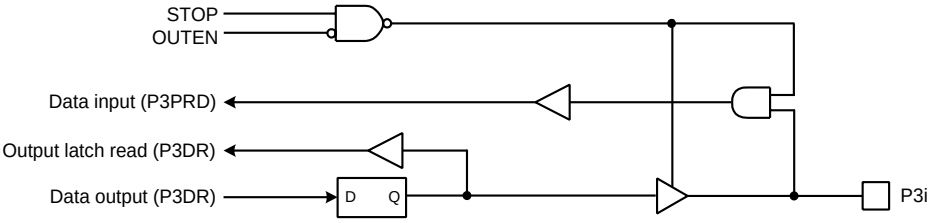
Port P3 is an 8-bit input/output port.

When used as an input port, the corresponding output latch (P3DR) should be set to "1".

During reset, the P3DR is initialized to "1".

P3 port output latch (P3DR) and P3 port terminal input (P3PRD) are located on their respective address.

When read the output latch data, the P3DR should be read. When read the terminal input data, the P3PRD register should be read.




Note: i = 7 to 0

Figure 5-5 Port 3

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| P3DR (0003H) R/W | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 | (Initial value: 1111 1111) |
| P3PRD (000DH) Read only | P37 | P36 | P35 | P34 | P33 | P32 | P31 | P30 | |

## 5.5 Port P4 (P47 to P40)

Port P4 is an 8-bit input/output port.

Port P4 is also used as a serial interface input/output and an UART input/output.

When used as an input port, a serial interface input/output and an UART input/output, the corresponding output latch (P4DR) should be set to "1".

During reset, the P4DR is initialized to "1", and the P4OUTCR is initialized to "0".

It can be selected whether output circuit of P4 port is a C-MOS output or a sink open drain individually, by setting P4OUTCR. When a corresponding bit of P4OUTCR is "0". the output circuit is selected to a sink open drain and when a corresponding bit of P4OUTCR is "1", the output circuit is selected to a C-MOS output.

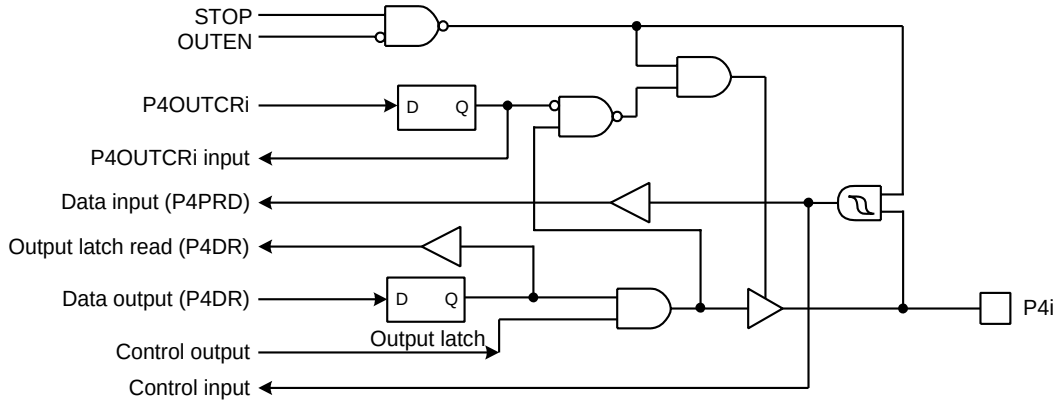
When used as an input port, a serial interface input and an UART input, the corresponding output control (P4OUTCR) should be set to "0" after P4DR is set to "1".

P4 port output latch (P4DR) and P4 port terminal input (P4PRD) are located on their respective address.

When read the output latch data, the P4DR should be read. When read the terminal input data, the P4PRD register should be read.

Table 5-3 Register Programming for Multi-function Ports (P47 to P40)

| Function | Programmed Value | |
|---|---|---|
| | P4DR | P4OUTCR |
| Port input UART input or serial interface input | “1” | “0” |
| Port “0” output | “0” | Programming for each applica-tions |
| Port “1” output UART output or serial interface output | “1” | |




Note: i = 7 to 0

Figure 5-6 Port 4

P4DR (0004H) R/W

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P47 | P46 $\overline{SCK2}$ | P45 SO2 | P44 SI2 | P43 | P42 TXD2 | P41 RXD2 | P40 |

(Initial value: 1111 1111)

P4OUTCR (000AH)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P4OUTCR | Port P4 output circuit control (Set for each bit individually) | 0: Sink open-drain output<br>1: C-MOS output | R/W |

P4PRD (000EH) Read only

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| P47 | P46 | P45 | P44 | P43 | P42 | P41 | P40 |

## 5.6 Port P5 (P54 to P50) (Large Current Port)

Port P5 is an 5-bit input/output port.

Port P5 is also used as an $I^2C$ Bus input/output.

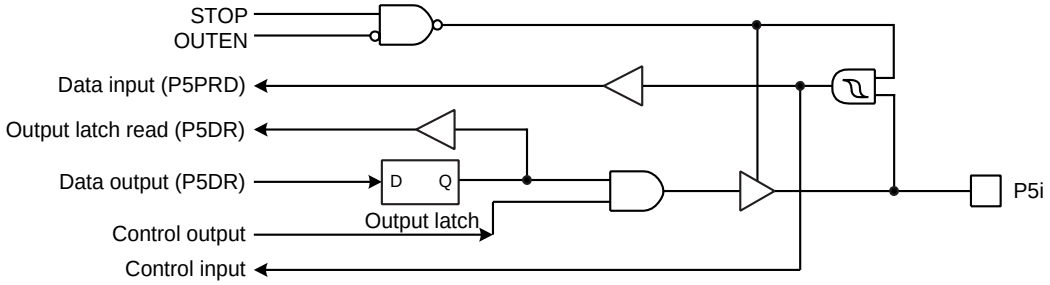
When used as an input port and $I^2C$ Bus input/output, the corresponding output latch (P5DR) should be set to "1".

During reset, the P5DR is initialized to "1".

P5 port output latch (P5DR) and P5 port terminal input (P5PRD) are located on their respective address.

When read the output latch data, the P5DR should be read. When read the terminal input data, the P5PRD register should be read.

If a read instruction is executed for port P5, read data of bit 7 to 5 are unstable.




Note: i = 4 to 0

Figure 5-7 Port 5

P5DR (0005H) R/W

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | P54 | P53 | P52 | P51 SDA | P50 SCL |

(Initial value: ***1 1111)

P5PRD (000FH) Read only

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | | | P54 | P53 | P52 | P51 | P50 |

## 5.7 Port P6 (P67 to P60)

Port P6 is an 8-bit input/output port which can be configured as an input or output in one-bit unit.

Port P6 is also used as an analog input and key-on wakeup input.

Input/output mode is specified by the P6 control register (P6CR1) and P6 input control register (P6CR2).

During reset, the P6CR1 is initialized to "0" the P6CR2 is initialized to "1" and port P6 becomes an input mode. And the P6DR is initialized to "0".

When used as an output port, the corresponding bit of P6CR1 should be set to "1".

When used as an input port , the corresponding bit of P6CR1 should be set to "0" and then, the corresponding bit of P6CR2 should be set to "1".

When used as a key-on wakeup input , the corresponding bit of P6CR1 should be set to "0" and then, the corresponding bit of STOPkEN should be set to "1".

When used as an analog input, the corresponding bit of P6CR1 should be set to "0" and then, the corresponding bit of P6CR2 should be set to "0".

When P6CR1 is "1", the content of the corresponding output latch is read by reading P6DR.

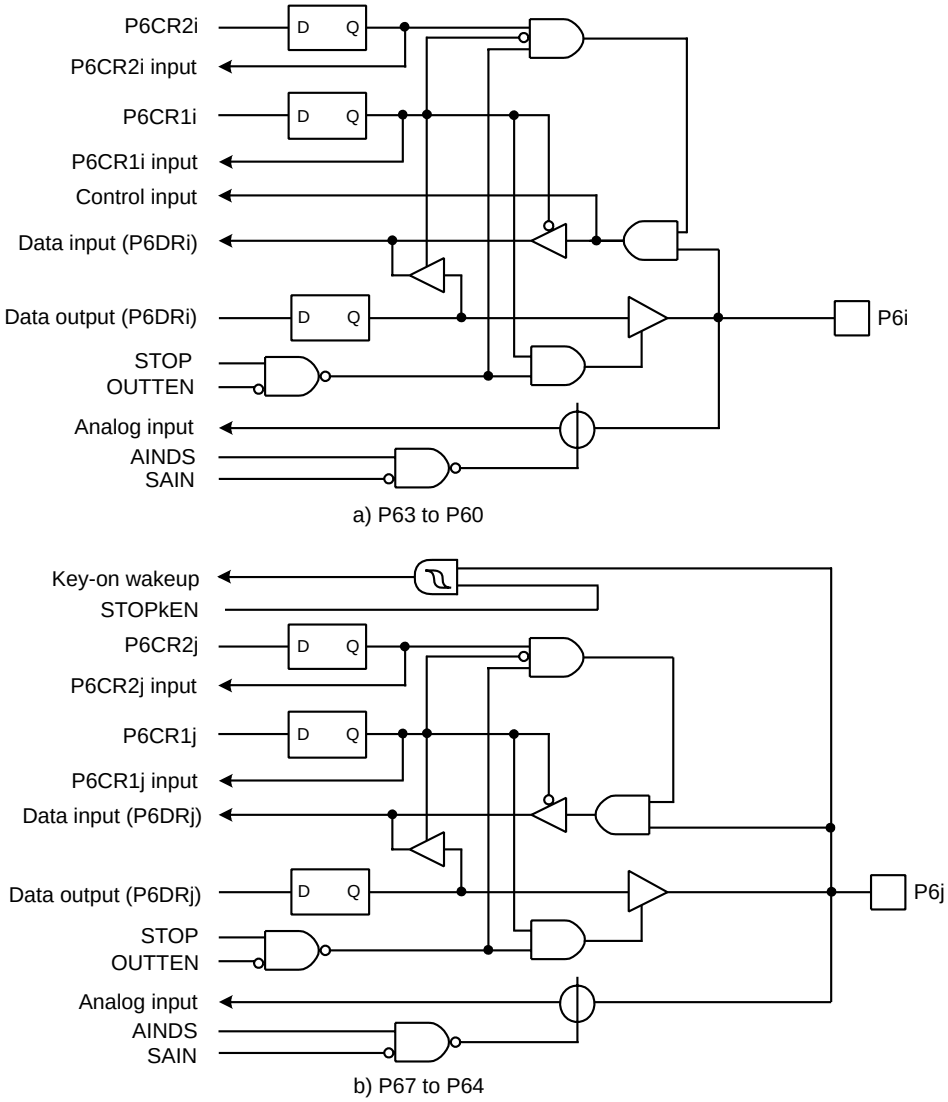
Table 5-4 Register Programming for Multi-function Ports

| Function | Programmed Value | | | |
|---|---|---|---|---|
| | P6DR | P6CR1 | P6CR2 | STOPkEN |
| Port input | * | “0” | “1” | * |
| Key-on wakeup input | * | "0" | * | "1" |
| Analog input | * | “0” | “0” | * |
| Port “0” output | “0” | “1” | * | * |
| Port “1” output | “1” | “1” | * | * |

Note: Asterisk (*) indicates “1” or “0” either of which can be selected.

Table 5-5 Values Read from P6DR and Register Programming

| Conditions | | Values Read from P6DR |
|---|---|---|
| P6CR1 | P6CR2 | |
| “0” | “0” | “0” |
| “0” | “1” | Terminal input data |
| “1” | “0” | Output latch contents |
| | “1” | |




Note 1: i = 3 to 0, j = 7 to 4, k = 3 to 0
Note 2: STOP is bit7 in SYSCR1.
Note 3: SAIN is AD input select signal.
Note 4: STOPkEN is input select signal in a key-on wakeup.

Figure 5-8 Port 6, P6CR1 and P6CR2

P6DR (0006H) R/W

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P67 AIN7 STOP3 | P66 AIN6 STOP2 | P65 AIN5 STOP1 | P64 AIN4 STOP0 | P63 AIN3 | P62 AIN2 | P61 AIN1 | P60 AIN0 |

(Initial value: 0000 0000)

P6CR1 (0F9BH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P6CR1 | I/O control for port P6 (Specified for each bit) | 0: Input mode<br>1: Output mode | R/W |

P6CR2 (0F9CH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 1111 1111)

| | | | |
|---|---|---|---|
| P6CR2 | P6 port input control (Specified for each bit) | 0: Analog input<br>1: Port input | R/W |

Note 1: The port placed in input mode reads the pin input state. Therefore, when the input and output modes are used together, the output latch contents for the port in input mode might be changed by executing a bit manipulation instruction.

Note 2: When used as an analog inport, be sure to clear the corresponding bit of P6CR2 to disable the port input.

Note 3: Do not set the output mode (P6CR1 = “1”) for the pin used as an analog input pin.

Note 4: Pins not used for analog input can be used as I/O ports. During AD conversion, output instructions should not be executed to keep a precision. In addition, a variable signal should not be input to a port adjacent to the analog input during AD conversion.

## 5.8 Port P7 (P77 to P70)

Port P7 is an 8-bit input/output port which can be configured as an input or output in one-bit unit.

Port P7 is also used as an analog input.

Input/output mode is specified by the P7 control register (P7CR1) and P7 input control register (P7CR2).

During reset, the P7CR1 is initialized to "0" the P7CR2 is initialized to "1" and port P7 becomes an input mode. And the P7DR is initialized to "0".

When used as an output port, the corresponding bit of P7CR1 should be set to "1".

When used as an input port, the corresponding bit of P7CR1 should be set to "0" and then, the corresponding bit of P7CR2 should be set to "1".

When used as an analog input, the corresponding bit of P7CR1 should be set to "0" and then, the corresponding bit of P7CR2 should be set to "0".

When P7CR1 is "1", the content of the corresponding output latch is read by reading P7DR.

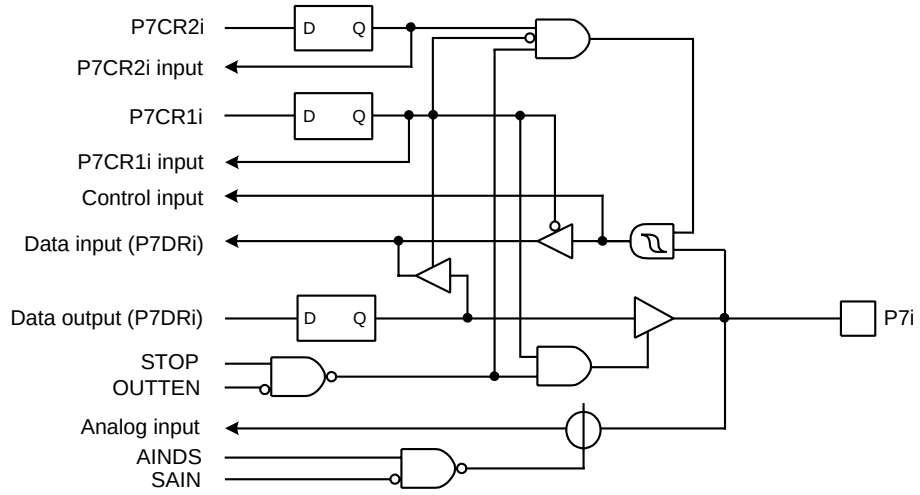
Table 5-6 Register Programming for Multi-function Ports

| Function | Programmed Value | | |
|---|---|---|---|
| | P7DR | P7CR1 | P7CR2 |
| Port input external interrupt input or key-on wakeup input | * | “0” | “1” |
| Analog input | * | “0” | “0” |
| Port “0” output | “0” | “1” | * |
| Port “1” output | “1” | “1” | * |

Note: Asterisk (*) indicates “1” or “0” either of which can be selected.

Table 5-7 Values Read from P7DR and Register Programming

| Conditions | | Values Read from P7DR |
|---|---|---|
| P7CR1 | P7CR2 | |
| “0” | “0” | “0” |
| “0” | “1” | Terminal input data |
| “1” | “0” | Output latch contents |
| | “1” | |




Note 1: i = 7 to 0

Note 2: STOP is bit7 in SYSCR1.

Note 3: SAIN is AD input select signal.

Figure 5-9 Port 7, P7CR1 and P7CR2

P7DR (0007H) R/W

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P77 AIN15 | P76 AIN14 | P75 AIN13 | P74 AIN12 | P73 AIN11 | P72 AIN10 | P71 AIN9 | P70 AIN8 |

(Initial value: 0000 0000)

P7CR1 (0F9DH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 0000 0000)

| | | | |
|---|---|---|---|
| P7CR1 | I/O control for port P7 (Specified for each bit) | 0: Input mode<br>1: Output mode | R/W |

P7CR2 (0F9EH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: 1111 1111)

| | | | |
|---|---|---|---|
| P7CR2 | P7 port input control (Specified for each bit) | 0: Analog input<br>1: Port input, external interrupt input or key-on wakeup input | R/W |

Note 1: The port placed in input mode reads the pin input state. Therefore, when the input and output modes are used together, the output latch contents for the port in input mode might be changed by executing a bit manipulation instruction.

Note 2: When used as an analog inport, be sure to clear the corresponding bit of P7CR2 to disable the port input.

Note 3: Do not set the output mode (P7CR1 = “1”) for the pin used as an analog input pin.

Note 4: Pins not used for analog input can be used as I/O ports. During AD conversion, output instructions should not be executed to keep a precision. In addition, a variable signal should not be input to a port adjacent to the analog input during AD conversion.

# 6. 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.

## 6.1 Watchdog Timer Configuration

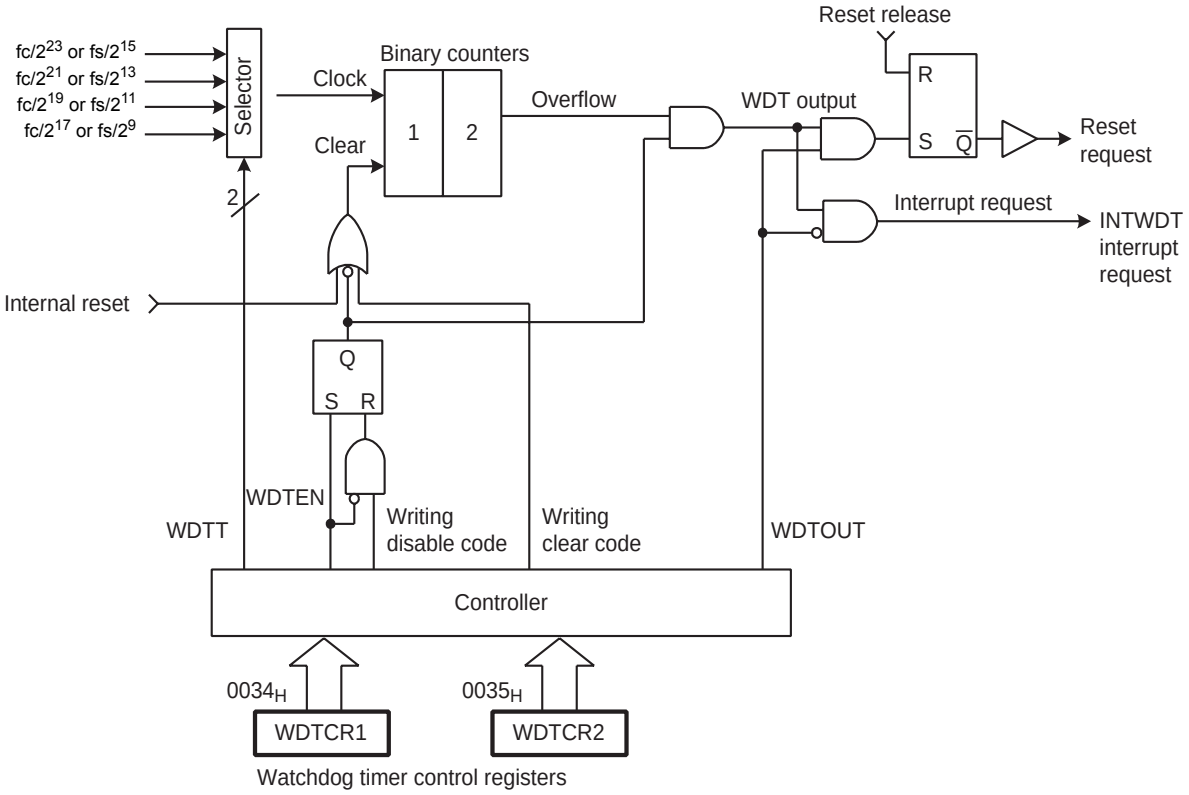



Figure 6-1 Watchdog Timer Configuration

## 6.2 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.

### 6.2.1 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 $2^{21}$/fc [s], and resetting the CPU malfunction detection

```
        LD      (WDTCR2), 4EH       : Clears the binary counters.
        LD      (WDTCR1), 00001101B : WDTT ← 10, WDTOUT ← 1
        LD      (WDTCR2), 4EH       : Clears the binary counters (always clears immediately before and
                                      after changing WDTT).
Within  :
3/4 of  :
WDT
detect  LD      (WDTCR2), 4EH       : Clears the binary counters.
time
        :
Within  :
3/4 of
WDT     LD      (WDTCR2), 4EH       : Clears the binary counters.
detect
time
```

Watchdog Timer Control Register 1

WDTCR1 (0034H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | (ATAS) | (ATOUT) | WDTEN | WDTT | | WDTOUT |

(Initial value: **11 1001)

| | | | | | | |
|---|---|---|---|---|---|---|
| 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] | | NORMAL1/2 mode | | SLOW1/2 mode | Write only |
| | | | DV7CK = 0 | DV7CK = 1 | | |
| | | 00 | $2^{25}$/fc | $2^{17}$/fs | $2^{17}$/fs | |
| | | 01 | $2^{23}$/fc | $2^{15}$/fs | $2^{15}$fs | |
| | | 10 | $2^{21}$fc | $2^{13}$/fs | $2^{13}$fs | |
| | | 11 | $2^{19}$/fc | $2^{11}$/fs | $2^{11}$/fs | |
| WDTOUT | Watchdog timer output select | 0: Interrupt request<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 “6.2.3 Watchdog Timer Disable”.

Watchdog Timer Control Register 2

WDTCR2 (0035H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: **** ****)

| | | | |
|---|---|---|---|
| WDTCR2 | Write<br>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>.

## 6.2.2 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.

### 6.2.3 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 counter
LDW     (WDTCR1), 0B101H    : WDTEN ← 0, WDTCR2 ← Disable code
```

Table 6-1 Watchdog Timer Detection Time (Example: fc = 16.0 MHz, fs = 32.768 kHz)

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

### 6.2.4 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, 083FH           : Sets the stack pointer
LD      (WDTCR1), 00001000B : WDTOUT ← 0
```

### 6.2.5 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 maximum 24/fc [s] (1.5 μs @ fc = 16.0 MHz).

Note: When a watchdog timer reset is generated in the SLOW1 mode, the reset time is maximum 24/fc (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.

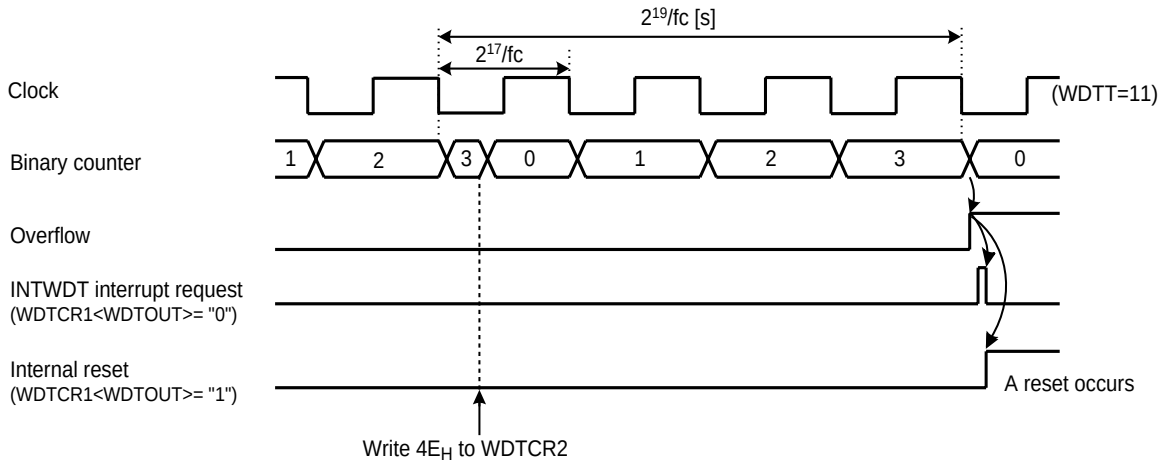



Figure 6-2 Watchdog Timer Interrupt

# 6.3 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 (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 (0035H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |

(Initial value: **** ****)

| | | | |
|---|---|---|---|
| WDTCR2 | Write<br>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 |

## 6.3.1 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>.

## 6.3.2 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>.

## 6.3.3 Address Trap Interrupt (INTATRAP)

While WDTCR1<ATOUT> is "0", 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 (while WDTCR1<ATAS> is "1"), DBR or the SFR area, address trap interrupt (INTATRAP) will be 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 an address trap 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.

### 6.3.4 Address Trap Reset

While WDTCR1<ATOUT> is “1”, 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 (while WDTCR1<ATAS> is “1”), DBR or the SFR area, address trap reset will be generated.

When an address trap reset request is generated, the internal hardware is reset. The reset time is maximum 24/fc [s] (1.5 μs @ fc = 16.0 MHz).

Note: When an address trap reset is generated in the SLOW1 mode, the reset time is maximum 24/fc (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. 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).

## 7.1 Time Base Timer

### 7.1.1 Configuration

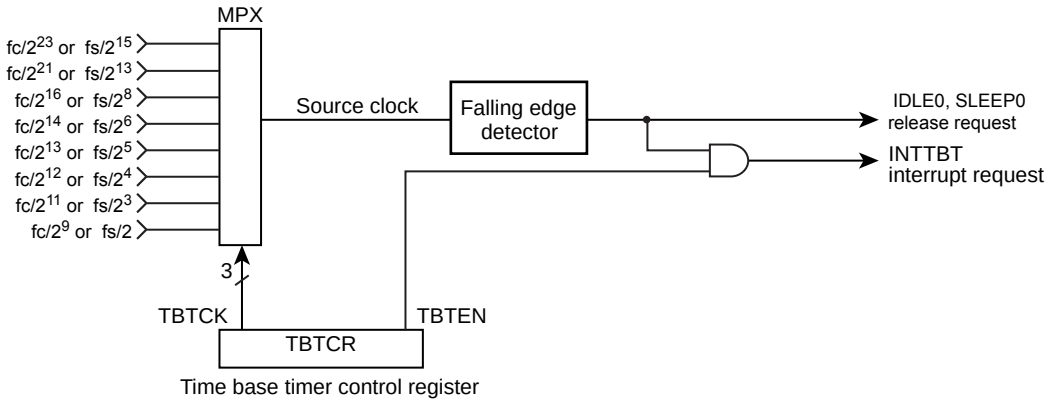



Figure 7-1 Time Base Timer configuration

### 7.1.2 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 (0036H) | (DVOEN) | (DVOCK) | | (DV7CK) | TBTEN | TBTCK | | | (Initial Value: 0000 0000) |

| | | | | | | |
|---|---|---|---|---|---|---|
| TBTEN | Time Base Timer enable / disable | 0: Disable<br>1: Enable | | | | |
| TBTCK | Time Base Timer interrupt Frequency select : [Hz] | | NORMAL1/2, IDLE1/2 Mode | | SLOW1/2 SLEEP1/2 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      (TBTCR) , 00000010B     ; TBTCK ← 010
LD      (TBTCR) , 00001010B     ; TBTEN ← 1
DI                              ; IMF ← 0
SET     (EIRL) . 7
```

Table 7-1 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 | – |

## 7.1.3 Function

An INTTBT ( Time Base Timer Interrupt ) is generated on the first falling edge of source clock ( The divider output of the timing generator 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 7-2 ).

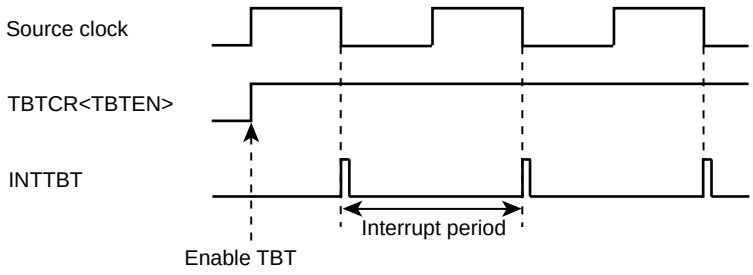



Figure 7-2 Time Base Timer Interrupt

## 7.2 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.

### 7.2.1 Configuration

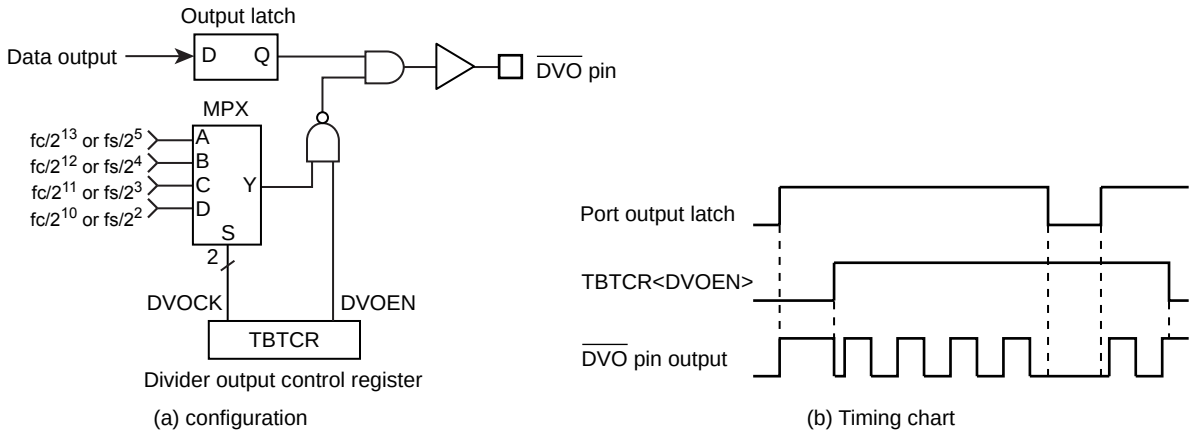



Figure 7-3 Divider Output

### 7.2.2 Control

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

Time Base Timer Control Register

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

| | | | | | |
|---|---|---|---|---|---|
| DVOEN | Divider output enable / disable | 0: Disable<br>1: Enable | | | R/W |
| DVOCK | Divider Output ($\overline{DVO}$) frequency selection: [Hz] | | NORMAL1/2, IDLE1/2 Mode | | SLOW1/2 SLEEP1/2 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 disable(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 7-2 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 |

# 8. 16-Bit TimerCounter 1 (TC1)

## 8.1 Configuration

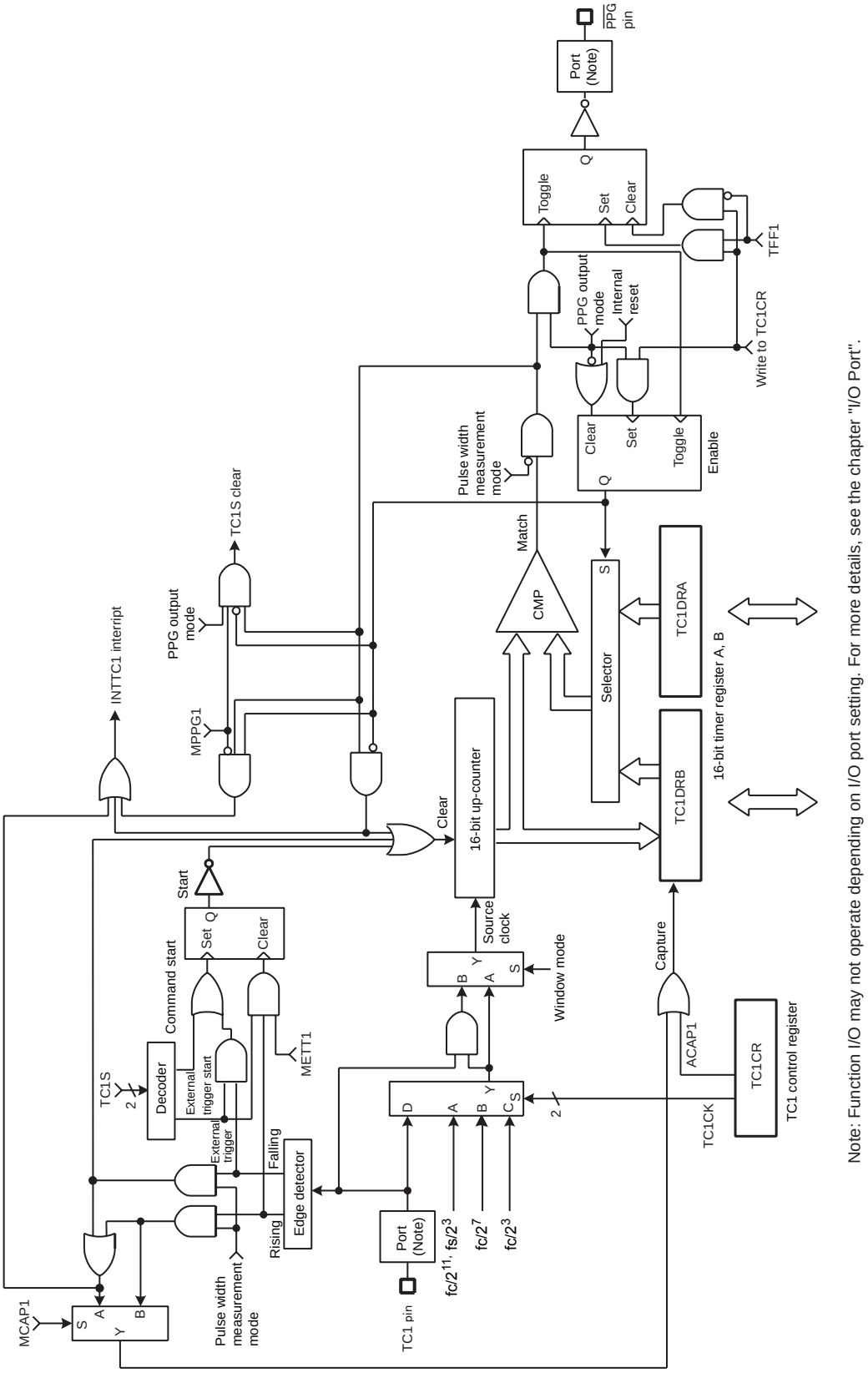



Figure 8-1 TimerCounter 1 (TC1)

## 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

| | 15 14 13 12 11 10 9 8 | 7 6 5 4 3 2 1 0 |
|---|---|---|
| TC1DRA (0011H, 0010H) | TC1DRAH (0011H) | TC1DRAL (0010H) |
| | (Initial value: 1111 1111 1111 1111) | Read/Write |
| TC1DRB (0013H, 0012H) | TC1DRBH (0013H) | TC1DRBL (0012H) |
| | (Initial value: 1111 1111 1111 1111) | Read/Write (Write enabled only in the PPG output mode) |

### TimerCounter 1 Control Register

| | 7 | 6 | 5 4 | 3 2 | 1 0 | |
|---|---|---|---|---|---|---|
| TC1CR (0026H) | TFF1 | ACAP1<br>MCAP1<br>METT1<br>MPPG1 | TC1S | TC1CK | TC1M | Read/Write<br>(Initial value: 0000 0000) |

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| TFF1 | Timer F/F1 control | 0: Clear | | | 1: Set | | | | R/W |
| ACAP1 | Auto capture control | 0:Auto-capture disable | | | 1:Auto-capture enable | | | | R/W |
| 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 | O | O | O | O | O | O | |
| | | 01: Command start | O | – | – | – | – | O | |
| | | 10: Rising edge start (Ex-trigger/Pulse/PPG)<br>Rising edge count (Event)<br>Positive logic count (Window) | – | O | O | O | O | O | |
| | | 11: Falling edge start (Ex-trigger/Pulse/PPG)<br>Falling edge count (Event)<br>Negative logic count (Window) | – | O | O | O | O | O | |
| TC1CK | TC1 source clock select [Hz] | | NORMAL1/2, IDLE1/2 mode | | | | Divider | SLOW, SLEEP 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 TC1CR 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: fc = 16 MHz, fs = 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 fc/$2^{11}$ [Hz] and generating an interrupt 1 second later (fc = 16 MHz, TBTCR<DV7CK> = “0”)

```
LDW     (TC1DRA), 1E84H     ; Sets the timer register (1 s ÷ 2^11/fc = 1E84H)
DI                          ; IMF= “0”
SET     (EIRL). 5           ; 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.

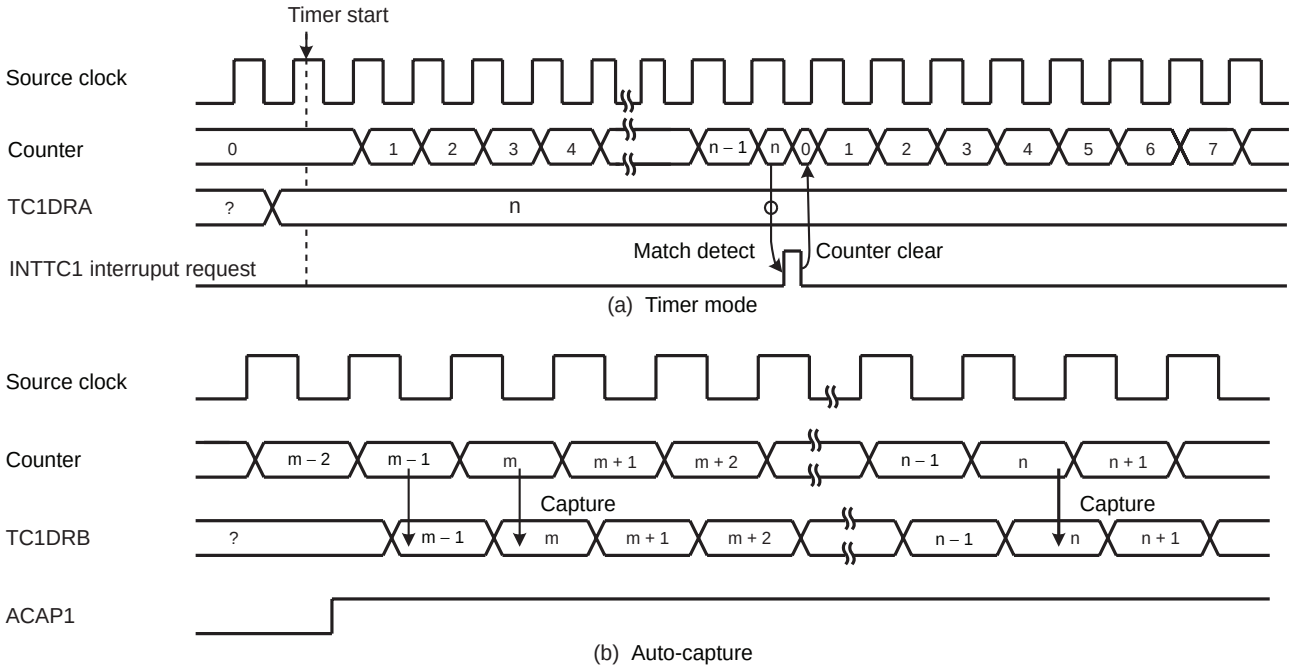



Figure 8-2 Timer Mode Timing Chart

### 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<TC1S>.

- 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.

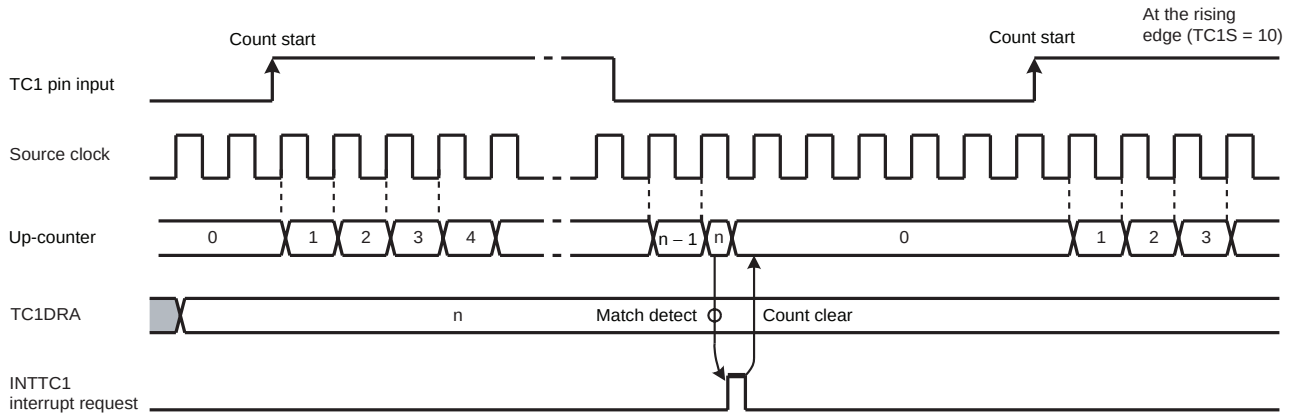
Since the TC1 pin input has the noise rejection, pulses of 4/fc [s] or less are rejected as noise. A pulse width of 12/fc [s] or more is required to ensure edge detection. The rejection circuit is turned off in the SLOW1/2 or SLEEP1/2 mode, but a pulse width of one machine cycle or more is required.

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

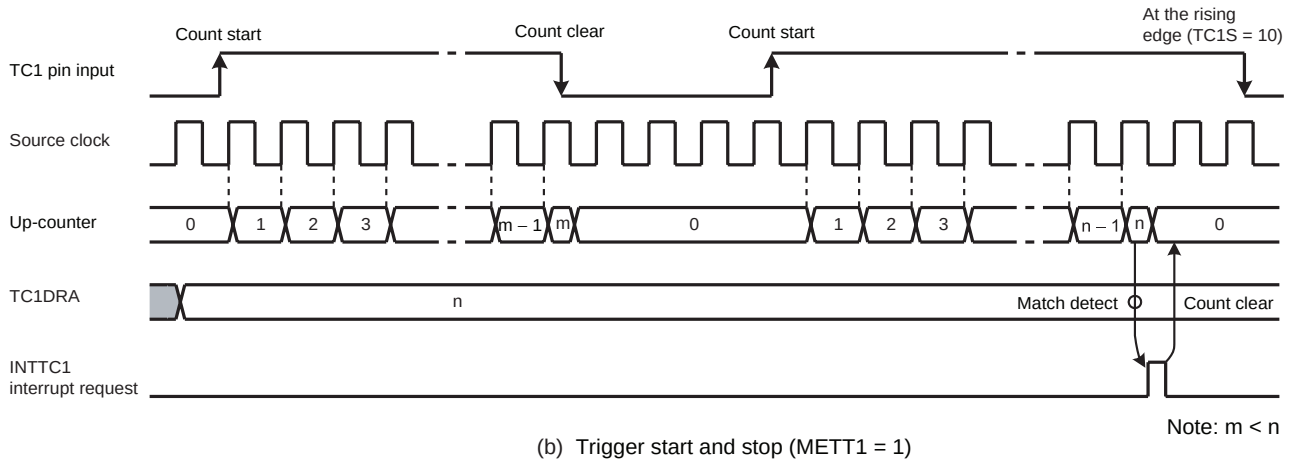
```
LDW     (TC1DRA), 007DH     ; 1ms ÷ 2^7/fc = 7DH
DI                          ; IMF= "0"
SET     (EIRL). 5           ; Enables INTTC1 interrupt
EI                          ; IMF= "1"
LD      (TC1CR), 00000100B  ; Selects the source clock and mode
LD      (TC1CR), 00100100B  ; Starts TC1 external trigger, METT1 = 0
```

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

```
LDW     (TC1DRA), 01F4H     ; 4 ms ÷ 2^7/fc = 1F4H
DI                          ; IMF= "0"
SET     (EIRL). 5           ; Enables INTTC1 interrupt
EI                          ; IMF= "1"
LD      (TC1CR), 00000100B  ; Selects the source clock and mode
LD      (TC1CR), 01110100B  ; Starts TC1 external trigger, METT1 = 1
```




(a) Trigger start (METT1 = 0)




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

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.

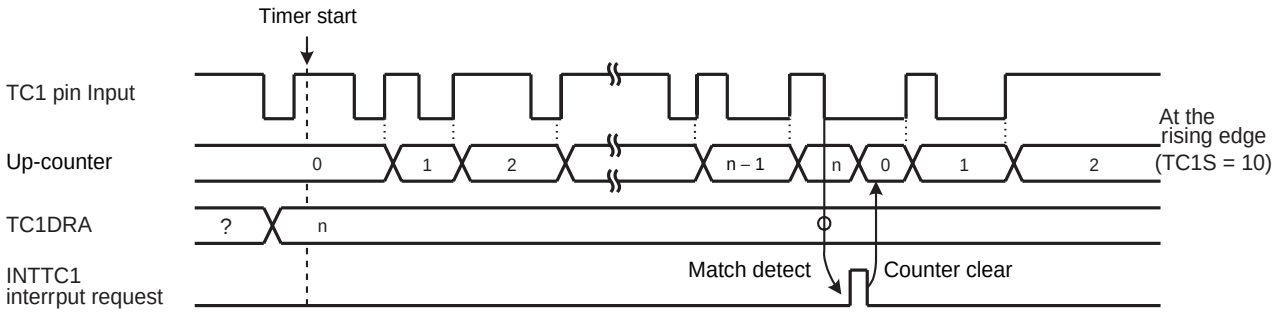



Figure 8-4 Event Counter Mode Timing Chart

Table 8-2 Input Pulse Width to TC1 Pin

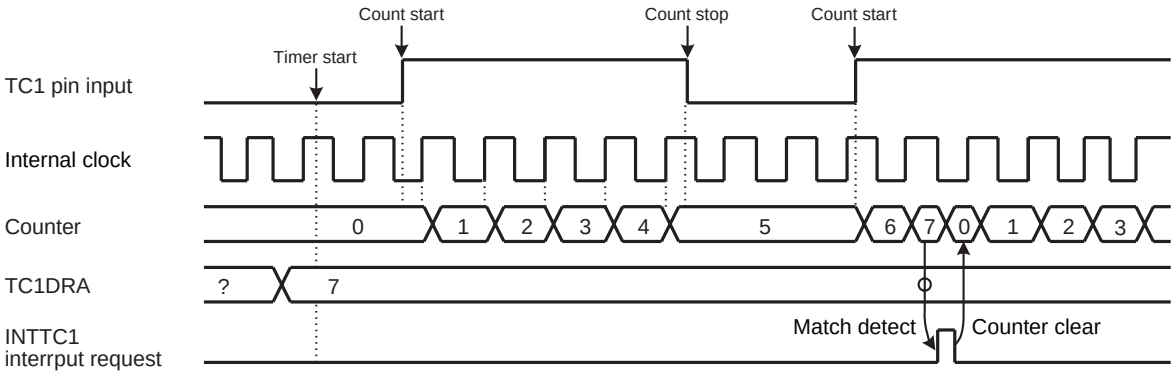
| | Minimum Pulse Width [s] | |
|---|---|---|
| | NORMAL1/2, IDLE1/2 Mode | SLOW1/2, SLEEP1/2 Mode |
| High-going | $2^3$/fc | $2^3$/fs |
| Low-going | $2^3$/fc | $2^3$/fs |

### 8.3.4 Window Mode

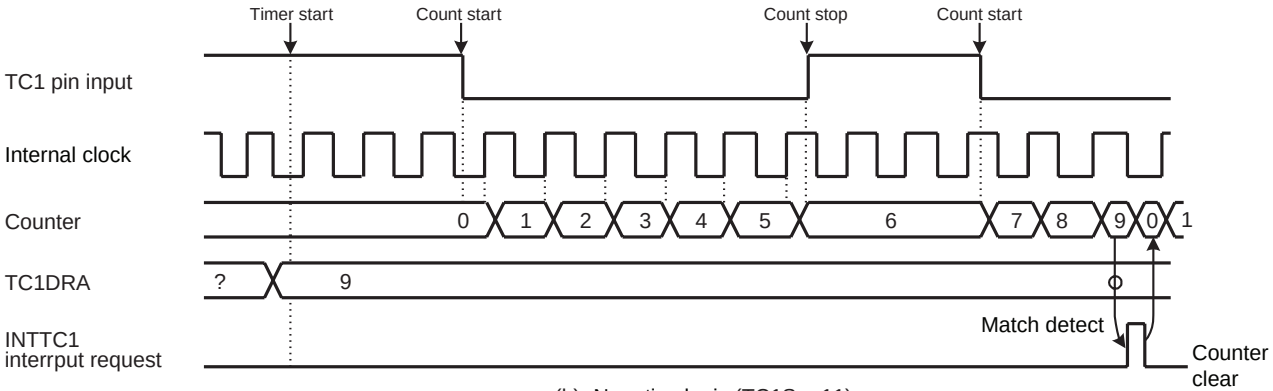
In the window mode, the up-counter counts up at the rising edge of the pulse that is logical ANDed product of the input pulse to the TC1 pin (window pulse) and the internal source clock. Either the positive logic (count up during high-going pulse) or negative logic (count up during low-going pulse) can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




(a) Positive logic (TC1S = 10)




(b) Negative logic (TC1S = 11)

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.

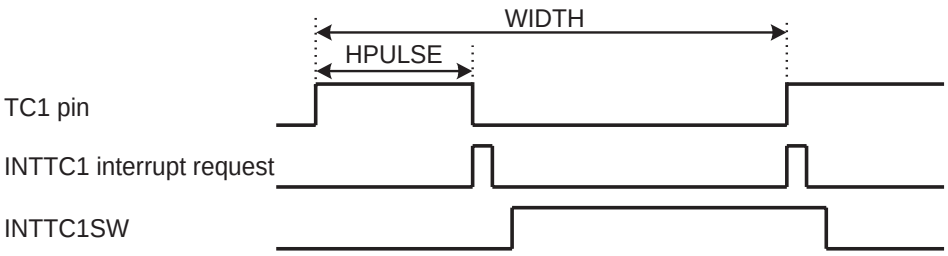
Note 1: The captured value must be read from TC1DRB until the next trigger edge is detected. If not read, the captured value becomes a don't care. It is recommended to use a 16-bit access instruction to read the captured value from TC1DRB.

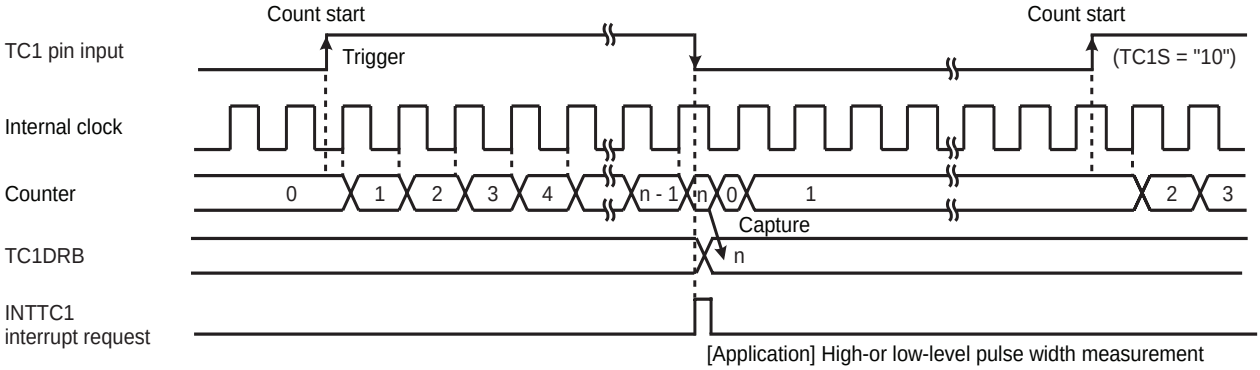
Note 2: For the single-edge capture, the counter after capturing the value stops at "1" until detecting the next edge. Therefore, the second captured value is "1" larger than the captured value immediately after counting starts.

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

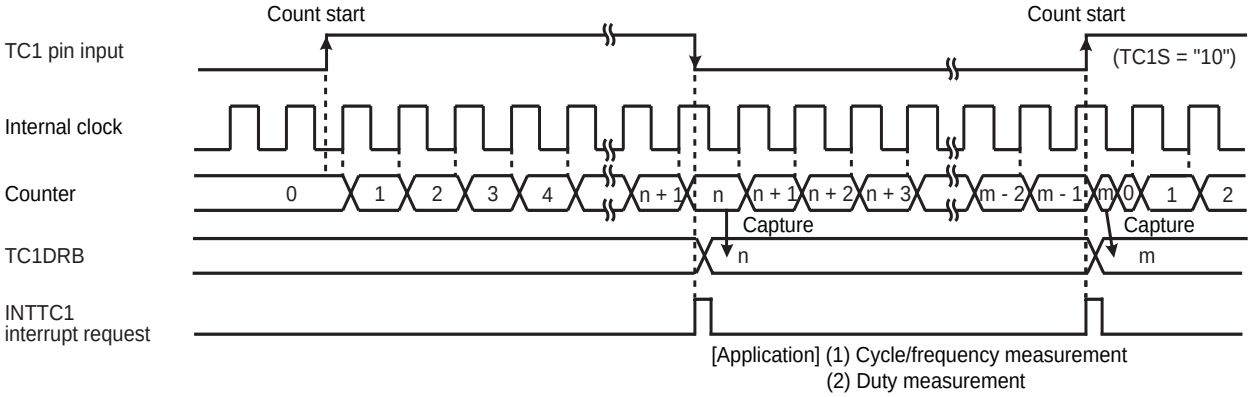
Example :Duty measurement (resolution fc/$2^7$ [Hz])

```
                CLR     (INTTC1SW). 0       ; INTTC1 service switch initial setting
                                              Address set to convert INTTC1SW at each INTTC1
                LD      (TC1CR), 00000110B  ; Sets the TC1 mode and source clock
                DI                          ; IMF= "0"
                SET     (EIRL). 5           ; Enables INTTC1
                EI                          ; IMF= "1"
                LD      (TC1CR), 00100110B  ; Starts TC1 with an external trigger at MCAP1 = 0
                :
PINTTC1:        CPL     (INTTC1SW). 0       ; 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 $\overline{PPG}$ 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 $\overline{PPG}$ 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 $\overline{PPG}$ 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 $\overline{PPG}$ 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 $\overline{PPG}$ 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 $\overline{PPG}$ 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 $\overline{PPG}$ pin, specify TC1CR<TFF1> to "0" to set the high level to the $\overline{PPG}$ pin, and "1" to set the low level to the $\overline{PPG}$ 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 μs and low-going for 200 μs
(fc = 16 MHz)

```
                Setting port
LD      (TC1CR), 10000111B      ; Sets the PPG mode, selects the source clock
LDW     (TC1DRA), 007DH         ; Sets the cycle (1 ms ÷ 2^7/fc ms = 007DH)
LDW     (TC1DRB), 0019H         ; Sets the low-level pulse width (200 μs ÷ 2^7/fc = 0019H)
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 ms ÷ 2^7/fc μs = 007DH)
LDW     (TC1DRB), 0019H         ; Sets the low-level pulse width (200 μs ÷ 2^7/fc = 0019H)
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
```

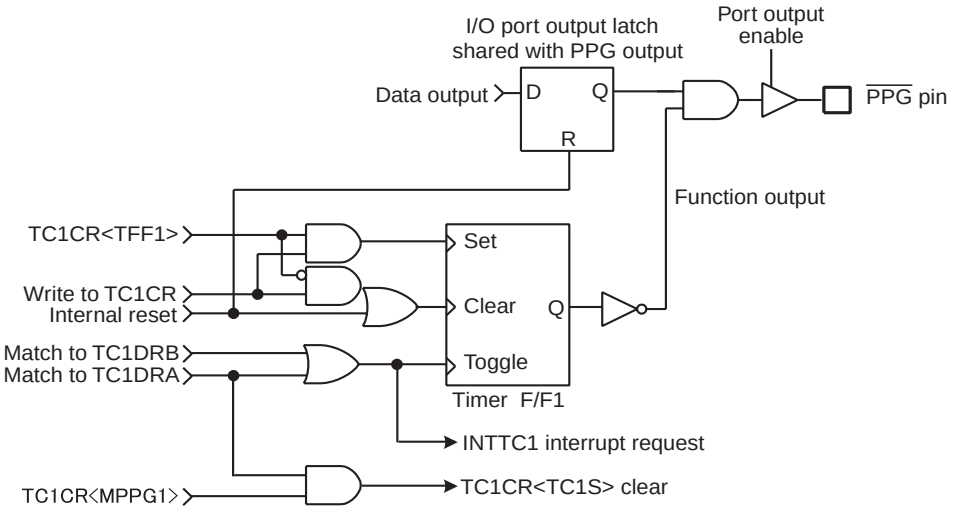



Figure 8-7 $\overline{\text{PPG}}$ Output

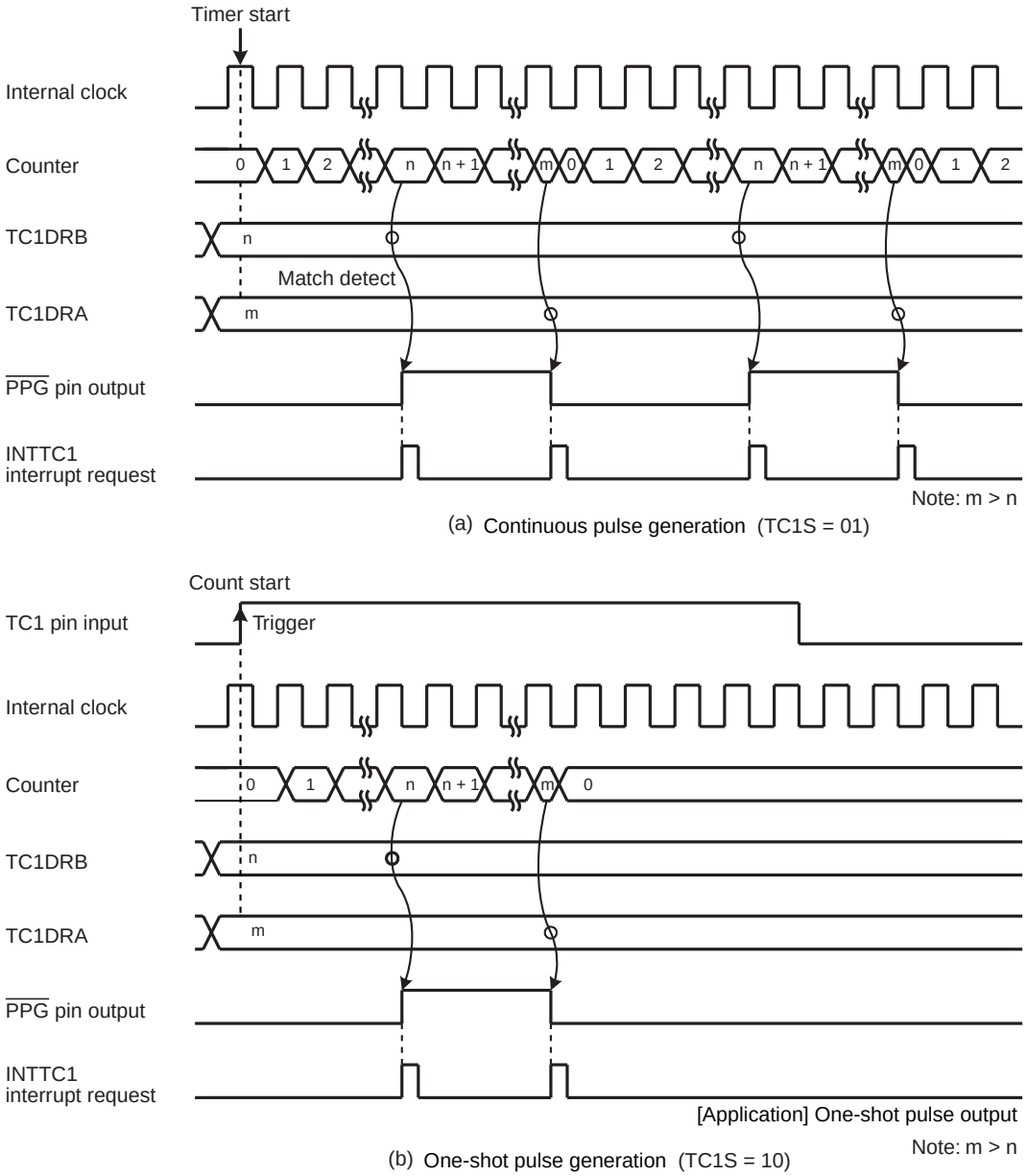
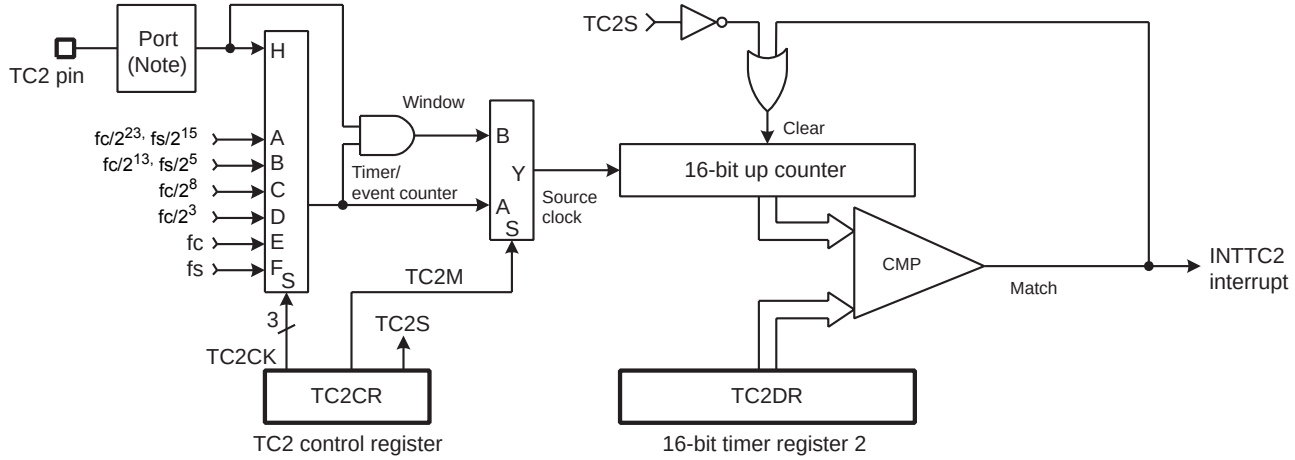



Figure 8-8 PPG Mode Timing Chart

# 9. 16-Bit Timer/Counter2 (TC2)

## 9.1 Configuration




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

Figure 9-1 Timer/Counter2 (TC2)

## 9.2 Control

The timer/counter 2 is controlled by a timer/counter 2 control register (TC2CR) and a 16-bit timer register 2 (TC2DR).

TC2DR (0025H, 0024H)

| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TC2DRH (0025H) | | | | | | | | TC2DRL (0024H) | | | | | | | |
| (Initial value: 1111 1111 1111 1111) | | | | | | | | R/W | | | | | | | |

TC2CR (0023H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | TC2S | TC2CK | | | | TC2M |

(Initial value: **00 00*0)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| TC2S | TC2 start control | 0:Stop and counter clear<br>1:Start | | | | | R/W |
| TC2CK | TC2 source clock select<br>Unit : [Hz] | | NORMAL1/2, IDLE1/2 mode | | Divider | SLOW1/2 mode | SLEEP1/2 mode | R/W |
| | | | DV7CK = 0 | DV7CK = 1 | | | |
| | | 000 | $fc/2^{23}$ | $fs/2^{15}$ | DV21 | $fs/2^{15}$ | $fs/2^{15}$ | |
| | | 001 | $fc/2^{13}$ | $fs/2^{5}$ | DV11 | $fs/2^{5}$ | $fs/2^{5}$ | |
| | | 010 | $fc/2^{8}$ | $fc/2^{8}$ | DV6 | – | – | |
| | | 011 | $fc/2^{3}$ | $fc/2^{3}$ | DV1 | – | – | |
| | | 100 | – | – | – | fc (Note7) | – | |
| | | 101 | fs | fs | – | – | – | |
| | | 110 | Reserved | | | | | |
| | | 111 | External clock (TC2 pin input) | | | | | |
| TC2M | TC2 operating mode select | 0:Timer/event counter mode<br>1:Window mode | | | | | | R/W |

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

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

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

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

Note 5: Values to be loaded to the timer register must satisfy the following condition.
TC2DR > 1 ($TC2DR_{15}$ to $TC2DR_{11}$ > 1 at warm up)

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

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

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

## 9.3 Function

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

And if fc or fs is selected as the source clock in timer mode, when switching the timer mode from SLOW1 to NORMAL2, the timer/counter2 can generate warm-up time until the oscillator is stable.

### 9.3.1 Timer mode

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

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

Table 9-1 Source Clock (Internal clock) for Timer/Counter2 (at fc = 16 MHz, DV7CK=0)

| TC2CK | NORMAL1/2, IDLE1/2 mode | | | | SLOW1/2 mode | | SLEEP1/2 mode | |
|---|---|---|---|---|---|---|---|---|
| | DV7CK = 0 | | DV7CK = 1 | | | | | |
| | Resolution | Maximum Time Setting | Resolution | Maximum Time Setting | Resolution | Maximum Time Setting | Resolution | Maximum Time Setting |
| 000 | 524.29 [ms] | 9.54 [h] | 1 [s] | 18.2 [h] | 1 [s] | 18.2 [h] | 1 [s] | 18.2 [h] |
| 001 | 512.0 [ms] | 33.55 [s] | 0.98 [ms] | 1.07 [min] | 0.98 [ms] | 1.07 [min] | 0.98 [ms] | 1.07 [min] |
| 010 | 16.0 [ms] | 1.05 [s] | 16.0 [ms] | 1.05 [s] | – | – | – | – |
| 011 | 0.5 [ms] | 32.77 [ms] | 0.5 [ms] | 32.77 [ms] | – | – | – | – |
| 100 | – | – | – | – | 62.5 [ns] | – | – | – |
| 101 | 30.52 [ms] | 2 [s] | 30.52 [ms] | 2 [s] | – | – | – | – |

Note: When fc is selected as the source clock in timer mode, it is used at warm-up for switching from SLOW1 mode to NORMAL2 mode.

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

```
LDW     (TC2DR), 061AH      ; Sets TC2DR (25 ms ³ 2^8/fc = 061AH)
DI                          ; IMF= “0”
SET     (EIRE). 6           ; Enables INTTC2 interrupt
EI                          ; IMF= “1”
LD      (TC2CR), 00001000B  ; Source clock / mode select
LD      (TC2CR), 00101000B  ; Starts Timer
```

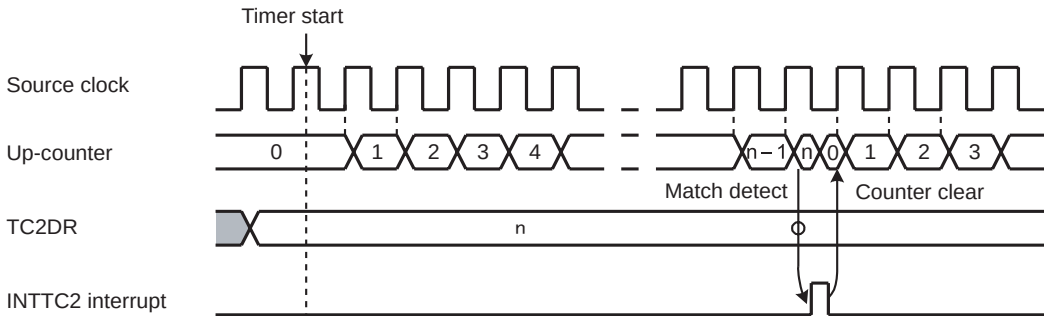



Figure 9-2 Timer Mode Timing Chart

### 9.3.2 Event counter mode

In this mode, events are counted on the rising edge of the TC2 pin input. The contents of TC2DR are compared with the contents of the up counter. If a match is found, an INTTC2 interrupt is generated, and the counter is cleared. Counting up is resumed every the rising edge of the TC2 pin input after the up counter is cleared.

Match detect is executed on the falling edge of the TC2 pin. Therefore, an INTTC2 interrupt is generated at the falling edge after the match of TC2DR and up counter.

The minimum input pulse width of TC2 pin is shown in Table 9-2. Two or more machine cycles are required for both the “H” and “L” levels of the pulse width.

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

```
LDW     (TC2DR), 640            ; Sets TC2DR
DI                              ; IMF= “0”
SET     (EIRE). 6               ;Enables  INTTC2 interrupt
EI                              ; IMF= “1”
LD      (TC2CR), 00011100B      ; TC2 source vclock / mode select
LD      (TC2CR), 00111100B      ; Starts TC2
```

Table 9-2 Timer/Counter 2 External Input Clock Pulse Width

| | Minimum Input Pulse Width [s] | |
|---|---|---|
| | NORMAL1/2, IDLE1/2 mode | SLOW1/2, SLEEP1/2 mode |
| “H” width | $2^3$/fc | $2^3$/fs |
| “L” width | $2^3$/fc | $2^3$/fs |

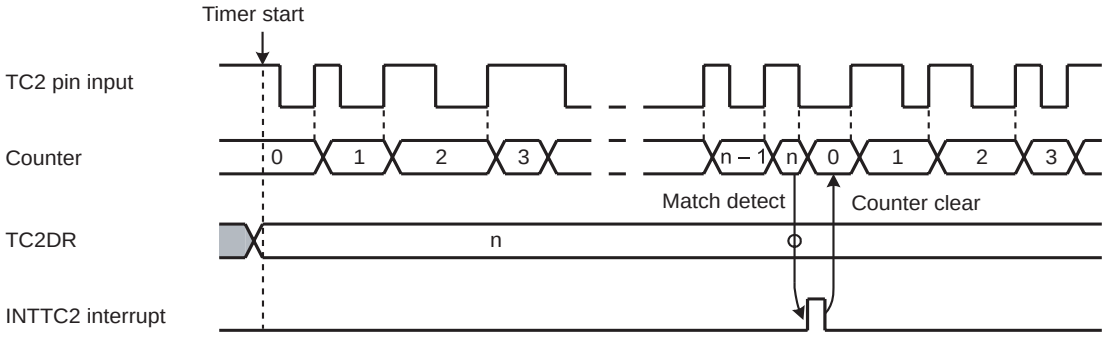



Figure 9-3 Event Counter Mode Timing Chart

### 9.3.3 Window mode

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

The maximum applied frequency (TC2 input) must be considerably slower than the selected internal clock by the TC2CR<TC2CK>.

Note: It is not available window mode in the SLOW/SLEEP mode. Therefore, at the window mode in NORMAL mode, the timer should be halted by setting TC2CR<TC2S> to "0" before the SLOW/SLEEP mode is entered.

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

```
LDW     (TC2DR), 00EAH          ; Sets TC2DR (120 ms ³ 213/fc = 00EAH)
DI                              ; IMF= “0”
SET     (EIRE). 6               ; Enables  INTTC2 interrupt
EI                              ; IMF= “1”
LD      (TC2CR), 00000101B      ; TC2sorce clock / mode select
LD      (TC2CR), 00100101B      ; Starts TC2
```

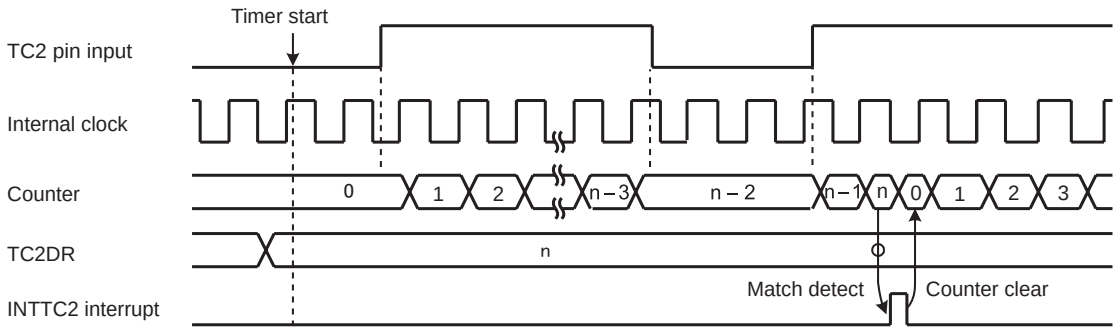



Figure 9-4 Window Mode Timing Chart

# 10. 8-Bit TimerCounter (TC3, TC4)

## 10.1 Configuration

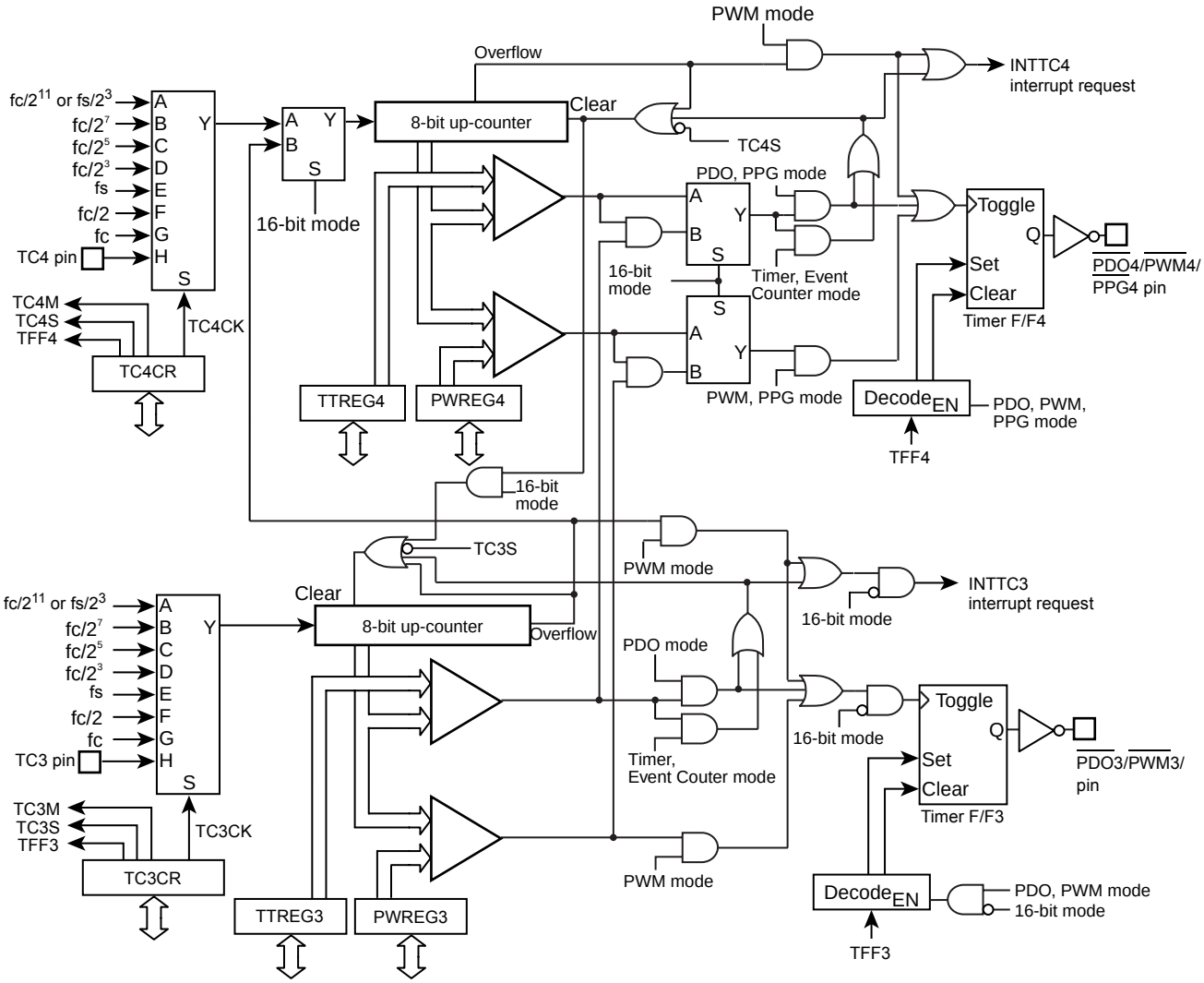



Figure 10-1 8-Bit TimerCounter 3, 4

## 10.2 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

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TTREG3 (0014H) R/W | | | | | | | | | (Initial value: 1111 1111) |
| PWREG3 (0018H) R/W | | | | | | | | | (Initial value: 1111 1111) |

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

| TC3CR (0027H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | TFF3 | TC3CK | | | TC3S | TC3M | | | (Initial value: 0000 0000) |

| | | | | | |
|---|---|---|---|---|---|
| TFF3 | Time F/F3 control | 0: Clear<br>1: Set | | | R/W |
| TC3CK | Operating clock selection [Hz] | | NORMAL1/2, IDLE1/2 mode | SLOW1/2 SLEEP1/2 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 | 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 10-1 and Table 10-2.

Note 7: The timer register settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 10-3.

Note 8: The operating clock fc 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

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TTREG4 (0015H) R/W | | | | | | | | | (Initial value: 1111 1111) |
| PWREG4 (0019H) R/W | | | | | | | | | (Initial value: 1111 1111) |

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

| TC4CR (0028H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | TFF4 | TC4CK | | | TC4S | TC4M | | | (Initial value: 0000 0000) |

| | | | | | |
|---|---|---|---|---|---|
| TFF4 | Timer F/F4 control | 0: Clear<br>1: Set | | | R/W |
| TC4CK | Operating clock selection [Hz] | | NORMAL1/2, IDLE1/2 mode | SLOW1/2 SLEEP1/2 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 TC3 overflow signal regardless of the TC4CK setting.

Note 5: To use the TimerCounter in the 16-bit mode, select the operating mode by programming TC4M, where TC3CR<TC3M> 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 10-1 and Table 10-2.

Note 8: The timer register settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 10-3.

Table 10-1 Operating Mode and Selectable Source Clock (NORMAL1/2 and IDLE1/2 Modes)

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

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: O : Available source clock

Table 10-2 Operating Mode and Selectable Source Clock (SLOW1/2 and SLEEP1/2 Modes)

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

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: O : Available source clock

Table 10-3 Constraints on Register Values Being Compared

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

Note: n = 3 to 4

## 10.3 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.

### 10.3.1 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 (TTREGj) value is detected, an INTTCj 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 $\overline{\text{PDOj}}$, $\overline{\text{PWMj}}$ and $\overline{\text{PPGj}}$ pins may output pulses.

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 the programming. Therefore, if TTREGi is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 3, 4

Table 10-4 Source Clock for TimerCounter 3, 4 (Internal Clock)

| Source Clock | | | Resolution | | Maximum Time Setting | |
|---|---|---|---|---|---|---|
| NORMAL1/2, IDLE1/2 mode | | SLOW1/2, SLEEP1/2 mode | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz | fs = 32.768 kHz |
| DV7CK = 0 | DV7CK = 1 | | | | | |
| $fc/2^{11}$ [Hz] | $fs/2^3$ [Hz] | $fs/2^3$ [Hz] | 128 μs | 244.14 μs | 32.6 ms | 62.3 ms |
| $fc/2^7$ | $fc/2^7$ | – | 8 μs | – | 2.0 ms | – |
| $fc/2^5$ | $fc/2^5$ | – | 2 μs | – | 510 μs | – |
| $fc/2^3$ | $fc/2^3$ | – | 500 ns | – | 127.5 μs | – |

Example :Setting the timer mode with source clock $fc/2^7$ Hz and generating an interrupt 80 μs later
(TimerCounter4, fc = 16.0 MHz)

```
LD      (TTREG4), 0AH          : Sets the timer register (80 μs÷2^7/fc = 0AH).
DI
SET     (EIRH). 1              : Enables INTTC4 interrupt.
EI
LD      (TC4CR), 00010000B     : Sets the operating clock to fc/2^7, and 8-bit timer mode.
LD      (TC4CR), 00011000B     : Starts TC4.
```

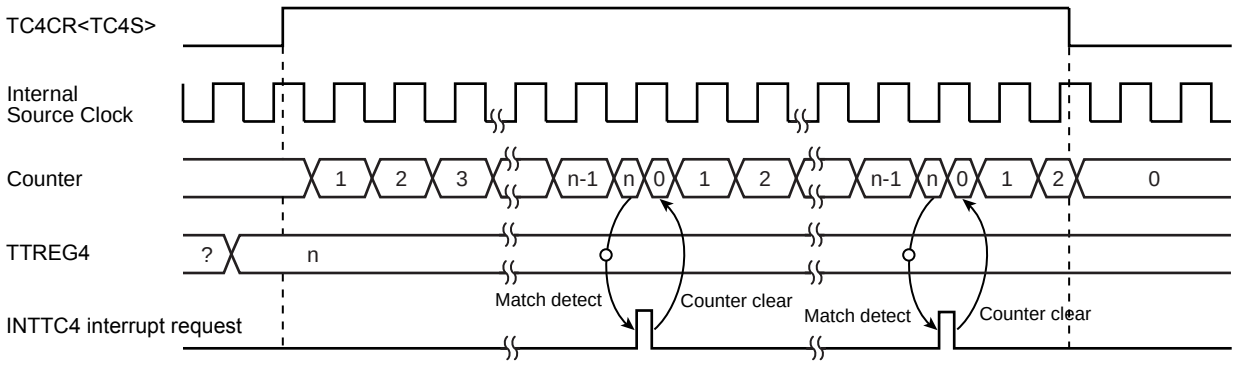



Figure 10-2 8-Bit Timer Mode Timing Chart (TC4)

## 10.3.2 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 INTTCj 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 $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ 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 $\overline{\text{PDOj, PWMj}}$ and $\overline{\text{PPGj}}$ 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 TTREGi is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 3, 4

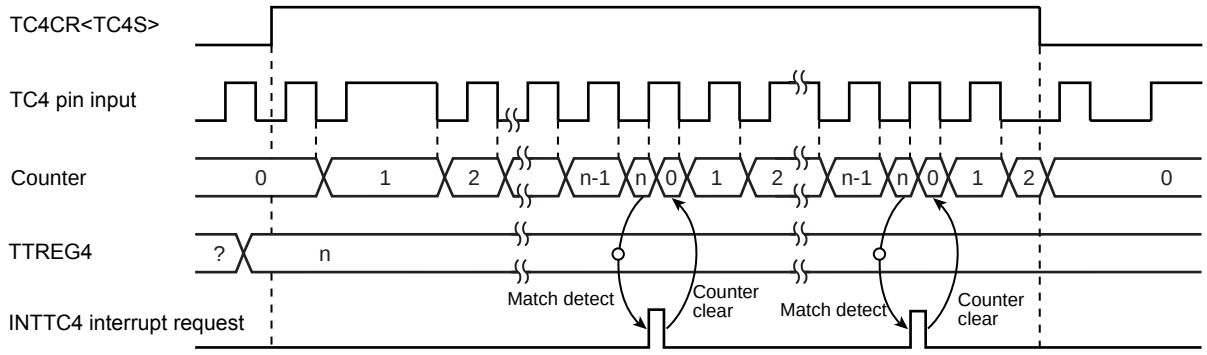



Figure 10-3 8-Bit Event Counter Mode Timing Chart (TC4)

## 10.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC3, 4)

This mode is used to generate a pulse with a 50% duty cycle from the $\overline{\text{PDOj}}$ 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 $\overline{\text{PDOj}}$ 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 $\overline{\text{PDOj}}$ 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 (fc = 16.0 MHz)

```
                Setting port
LD      (TTREG4),  3DH          : 1/1024÷2^7/fc÷2 = 3DH
LD      (TC4CR),   00010001B    : Sets the operating clock to fc/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 $\overline{\text{PDOj}}$ 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 $\overline{\text{PDOj}}$ pin to the high level when the TimerCounter is stopped
CLR (TCjCR).3: Stops the timer.
CLR (TCjCR).7: Sets the $\overline{\text{PDOj}}$ pin to the high level.

Note 3: j = 3, 4

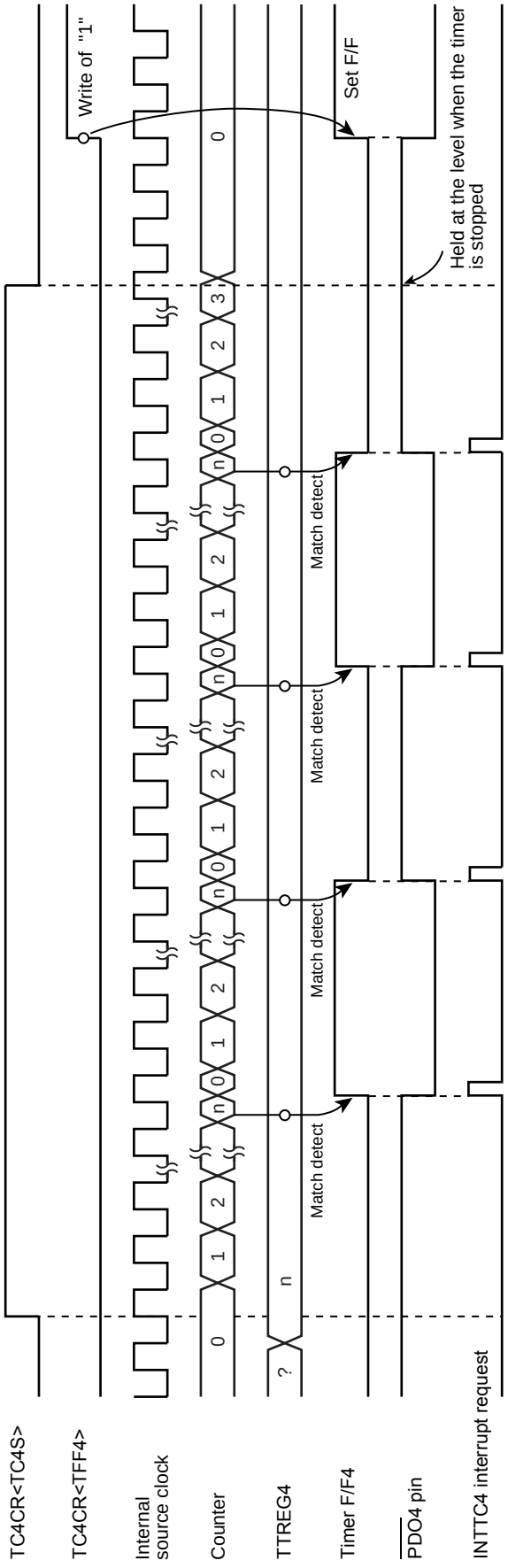



Figure 10-4 8-Bit PDO Mode Timing Chart (TC4)

### 10.3.4 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 10-5 PWM Output Mode

| Source Clock | | | Resolution | | Repeated Cycle | |
|---|---|---|---|---|---|---|
| NORMAL1/2, IDLE1/2 mode | | SLOW1/2, SLEEP1/2 mode | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz | fs = 32.768 kHz |
| DV7CK = 0 | DV7CK = 1 | | | | | |
| $fc/2^{11}$ [Hz] | $fs/2^3$ [Hz] | $fs/2^3$ [Hz] | 128 μs | 244.14 μs | 32.8 ms | 62.5 ms |
| $fc/2^7$ | $fc/2^7$ | – | 8 μs | – | 2.05 ms | – |
| $fc/2^5$ | $fc/2^5$ | – | 2 μs | – | 512 μs | – |
| $fc/2^3$ | $fc/2^3$ | – | 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 | – |

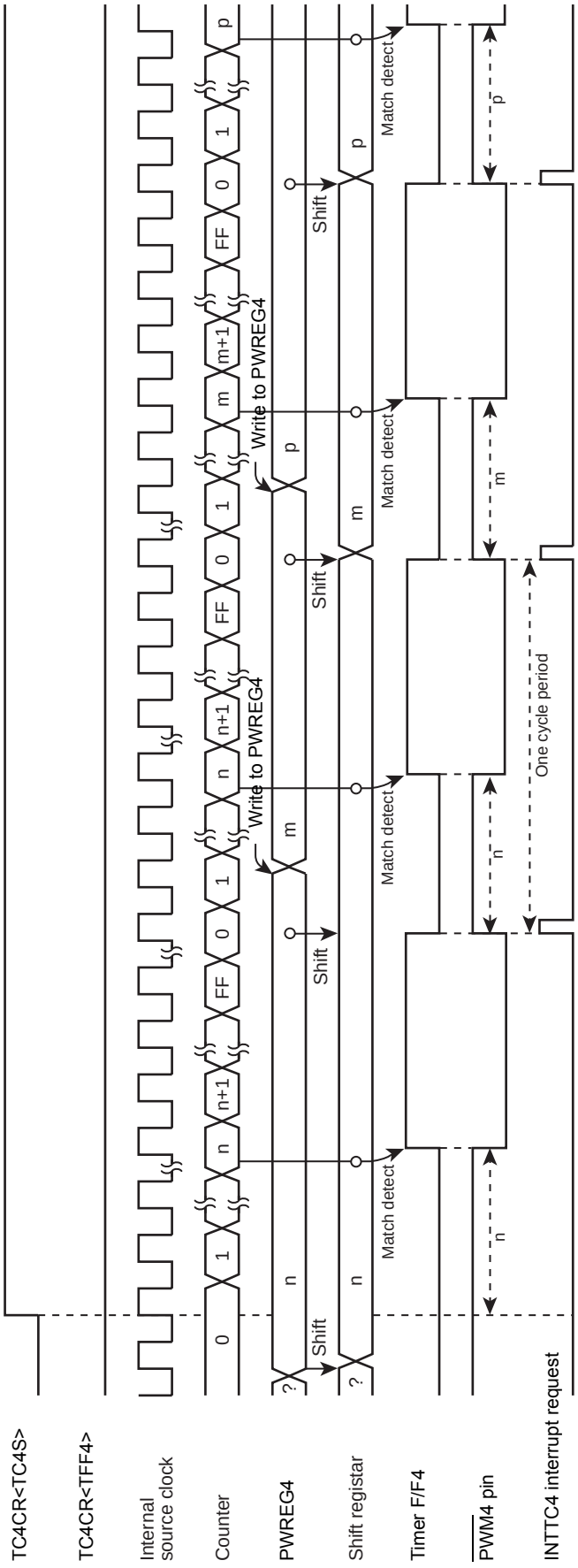



Figure 10-5 8-Bit PWM Mode Timing Chart (TC4)

### 10.3.5 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 lower byte and upper 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 10-6 Source Clock for 16-Bit Timer Mode

| Source Clock | | | Resolution | | Maximum Time Setting | |
|---|---|---|---|---|---|---|
| NORMAL1/2, IDLE1/2 mode | | SLOW1/2, SLEEP1/2 mode | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz | fs = 32.768 kHz |
| DV7CK = 0 | DV7CK = 1 | | | | | |
| $fc/2^{11}$ | $fs/2^3$ | $fs/2^3$ | 128 μs | 244.14 μs | 8.39 s | 16 s |
| $fc/2^7$ | $fc/2^7$ | – | 8 μs | – | 524.3 ms | – |
| $fc/2^5$ | $fc/2^5$ | – | 2 μs | – | 131.1 ms | – |
| $fc/2^3$ | $fc/2^3$ | – | 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÷2^7/fc = 927CH).
DI
SET     (EIRH). 1           : Enables INTTC4 interrupt.
EI
LD      (TC3CR), 13H        :Sets the operating clock to fc/2^7, and 16-bit timer mode
                             (lower byte).
LD      (TC4CR), 04H        : Sets the 16-bit timer mode (upper byte).
LD      (TC4CR), 0CH        : Starts the timer.
```

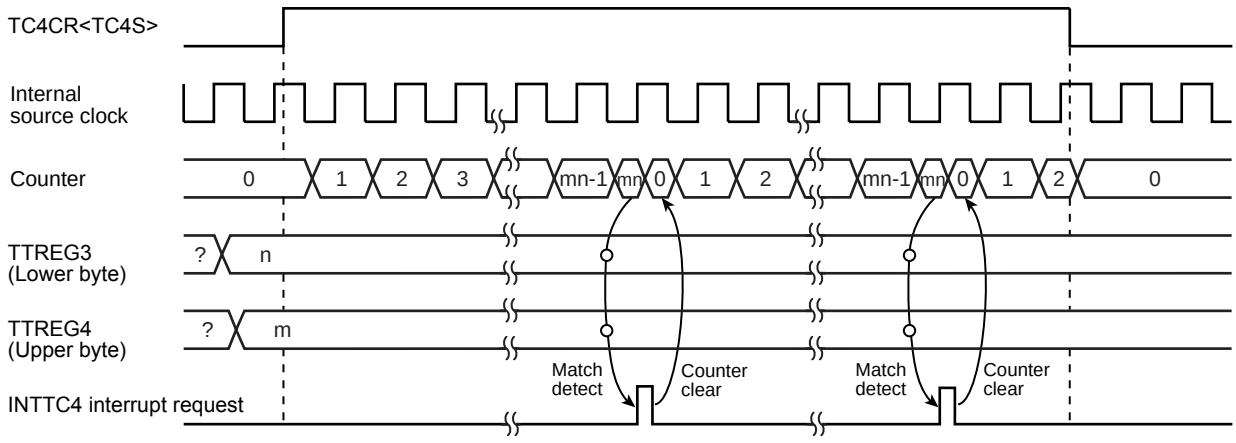



Figure 10-6 16-Bit Timer Mode Timing Chart (TC3 and TC4)

### 10.3.6 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 INTTC4 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 $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ 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

### 10.3.7 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 $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ 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 (PWREG4) 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 $\overline{PWM4}$ 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 fc, fc/2 or fs is selected as the source clock, a pulse is output from the $\overline{PWM4}$ pin during the warm-up period time after exiting the STOP mode.

Table 10-7 16-Bit PWM Output Mode

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

Example :Generating a pulse with 1-ms high-level width and a period of 32.768 ms (fc = 16.0 MHz)

```
        Setting ports
LDW     (PWREG3), 07D0H     : Sets the pulse width.
LD      (TC3CR), 33H        : Sets the operating clock to fc/2^3, and 16-bit PWM output
                              mode (lower byte).
LD      (TC4CR), 056H       : Sets TFF4 to the initial value 0, and 16-bit PWM signal
                              generation mode (upper byte).
LD      (TC4CR), 05EH       : Starts the timer.
```

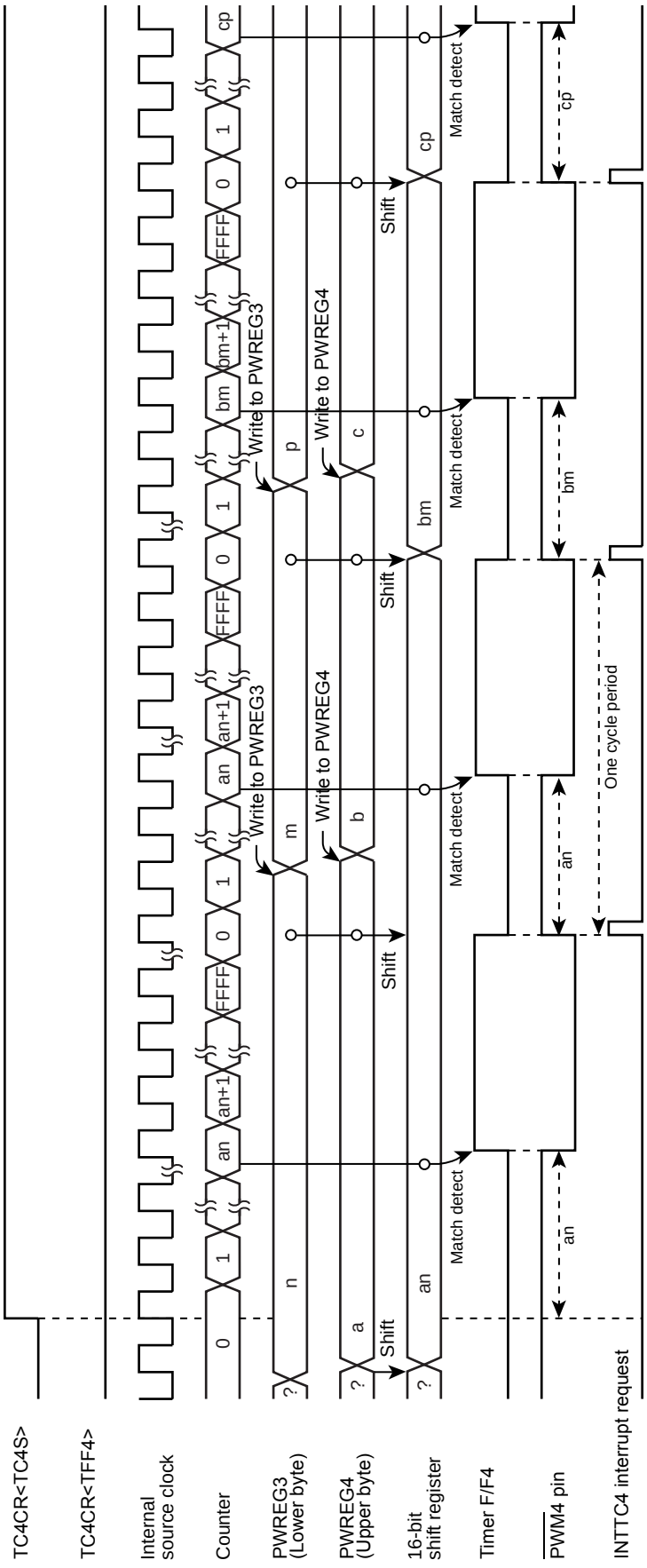



Figure 10-7 16-Bit PWM Mode Timing Chart (TC3 and TC4)

### 10.3.8 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 cascadable 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 NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ 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 $\overline{PPG}4$ 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, and16-bit PPG mode
                              (lower byte).
LD      (TC4CR), 057H       : Sets TFF4 to the initial value 0, and 16-bit
                              PPG 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 $\overline{PPG}4$ 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 $\overline{PPG}4$ pin to the high level when the TimerCounter is stopped
CLR (TC4CR).3: Stops the timer
CLR (TC4CR).7: Sets the $\overline{PPG}4$ pin to the high level

Note 3: i = 3, 4

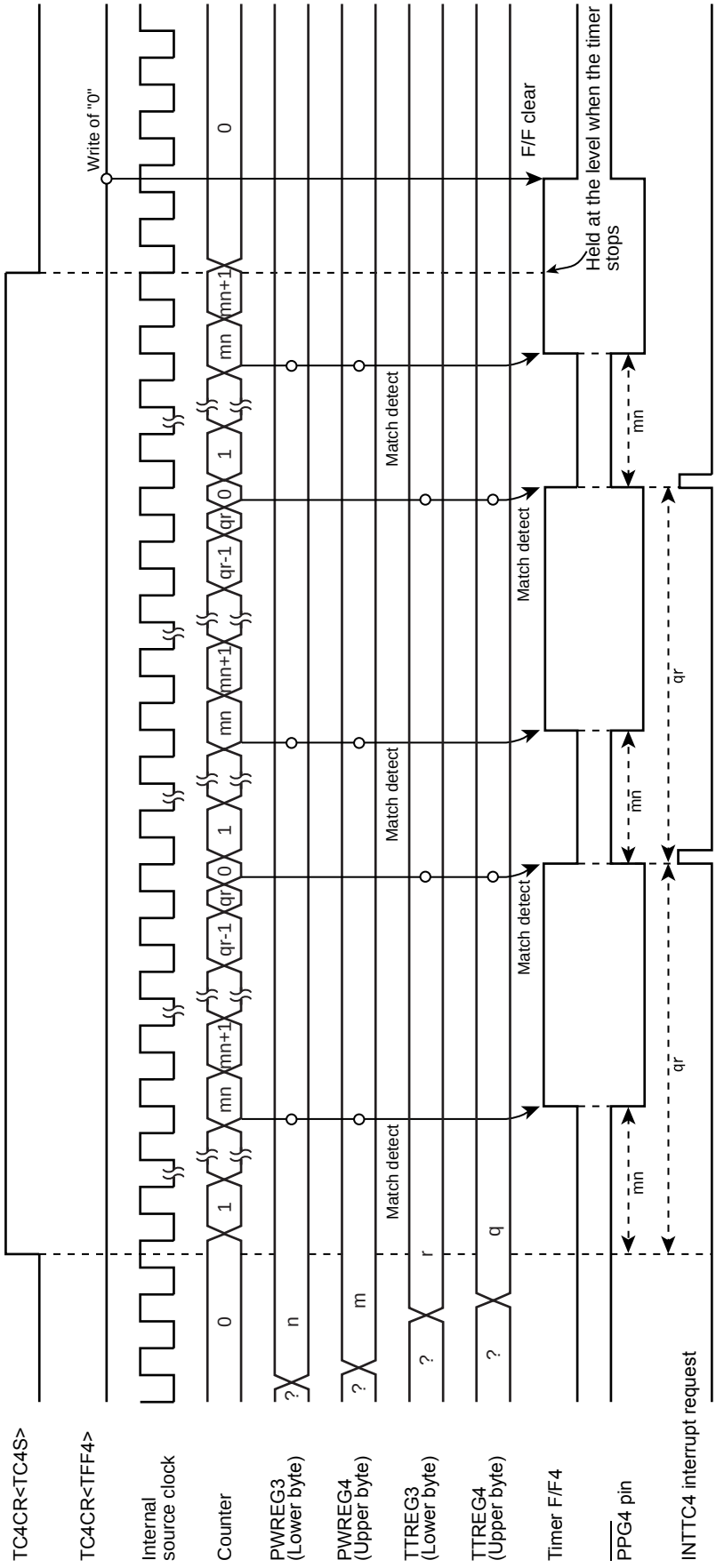



Figure 10-8 16-Bit PPG Mode Timing Chart (TC3 and TC4)

### 10.3.9 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 $\overline{\text{PDOi}}$, $\overline{\text{PWMi}}$ and $\overline{\text{PPGi}}$ 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

#### 10.3.9.1 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<XEN> to 0 to stop the high-frequency clock.

Table 10-8 Setting Time of Low-Frequency Warm-Up Counter Mode (fs = 32.768 kHz)

| Minimum Time Setting (TTREG4, 3 = 0100H) | Maximum Time Setting (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.
                                          (The warm-up time depends on the oscillator characteristic.)
                DI                      : IMF ← 0
                SET     (EIRH). 1       : 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
                                          (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
```

### 10.3.9.2 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 fc 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 10-9 Setting Time in High-Frequency Warm-Up Counter Mode

| Minimum time Setting (TTREG4, 3 = 0100H) | Maximum time Setting (TTREG4, 3 = FF00H) |
|---|---|
| 16 μs | 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 fc, 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). 1           : 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
```

# 11. 8-Bit TimerCounter (TC5, TC6)

## 11.1 Configuration

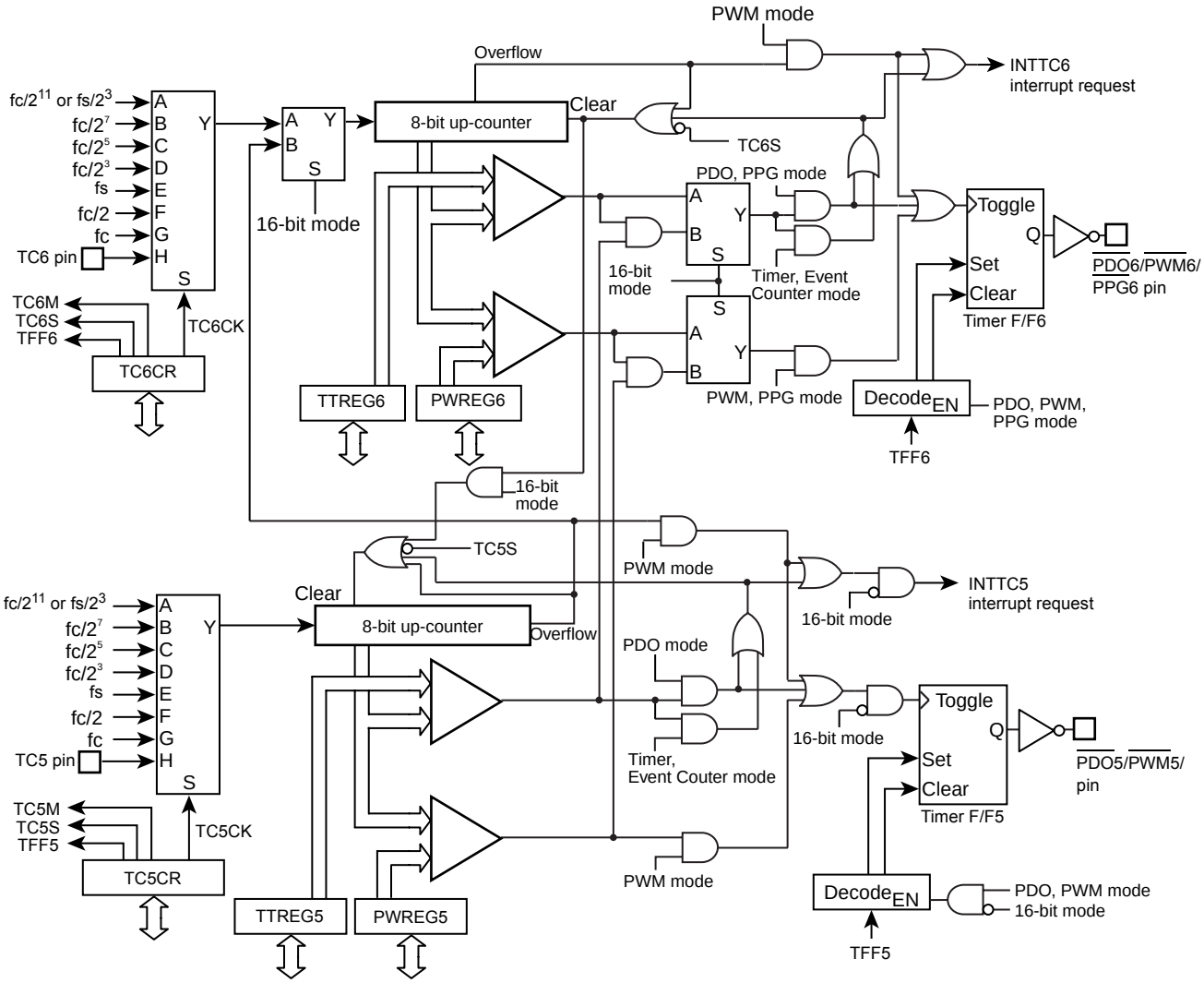



Figure 11-1 8-Bit TimerCounter 5, 6

## 11.2 TimerCounter Control

The TimerCounter 5 is controlled by the TimerCounter 5 control register (TC5CR) and two 8-bit timer registers (TTREG5, PWREG5).

### TimerCounter 5 Timer Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TTREG5 (0016H) R/W | | | | | | | | | (Initial value: 1111 1111) |
| PWREG5 (001AH) R/W | | | | | | | | | (Initial value: 1111 1111) |

Note 1: Do not change the timer register (TTREG5) setting while the timer is running.

Note 2: Do not change the timer register (PWREG5) setting in the operating mode except the 8-bit and 16-bit PWM modes while the timer is running.

### TimerCounter 5 Control Register

| TC5CR (0029H) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | TFF5 | TC5CK | | | TC5S | TC5M | | | (Initial value: 0000 0000) |

| | | | | | |
|---|---|---|---|---|---|
| TFF5 | Time F/F5 control | 0: Clear<br>1: Set | | | R/W |
| TC5CK | Operating clock selection [Hz] | | NORMAL1/2, IDLE1/2 mode | SLOW1/2 SLEEP1/2 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 | fc (Note 8) |
| | | 111 | TC5 pin input | | |
| TC5S | TC5 start control | 0: Operation stop and counter clear<br>1: Operation start | | | R/W |
| TC5M | TC5M 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 TC6M.)<br>1**: Reserved | | | R/W |

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

Note 2: Do not change the TC5M, TC5CK and TFF5 settings while the timer is running.

Note 3: To stop the timer operation (TC5S= 1 → 0), do not change the TC5M, TC5CK and TFF5 settings. To start the timer operation (TC5S= 0 → 1), TC5M, TC5CK and TFF5 can be programmed.

Note 4: To use the TimerCounter in the 16-bit mode, set the operating mode by programming TC6CR<TC6M>, where TC5M must be fixed to 011.

Note 5: To use the TimerCounter in the 16-bit mode, select the source clock by programming TC5CK. Set the timer start control and timer F/F control by programming TC6CR<TC6S> and TC6CR<TFF6>, respectively.

Note 6: The operating clock settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 11-1 and Table 11-2.

Note 7: The timer register settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 11-3.

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

The TimerCounter 6 is controlled by the TimerCounter 6 control register (TC6CR) and two 8-bit timer registers (TTREG6 and PWREG6).

### TimerCounter 6 Timer Register

| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| TTREG6 (0017H) R/W | | | | | | | | | (Initial value: 1111 1111) |
| PWREG6 (001BH) R/W | | | | | | | | | (Initial value: 1111 1111) |

Note 1: Do not change the timer register (TTREG6) setting while the timer is running.

Note 2: Do not change the timer register (PWREG6) setting in the operating mode except the 8-bit and 16-bit PWM modes while the timer is running.

### TimerCounter 6 Control Register

| TC6CR (002AH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | TFF6 | TC6CK | | | TC6S | TC6M | | | (Initial value: 0000 0000) |

| | | | | | |
|---|---|---|---|---|---|
| TFF6 | Timer F/F6 control | 0: Clear<br>1: Set | | | R/W |
| TC6CK | Operating clock selection [Hz] | | NORMAL1/2, IDLE1/2 mode | SLOW1/2 SLEEP1/2 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 | TC6 pin input | | |
| TC6S | TC6 start control | 0: Operation stop and counter clear<br>1: Operation start | | | R/W |
| TC6M | TC6M 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 TC6M, TC6CK and TFF6 settings while the timer is running.

Note 3: To stop the timer operation (TC6S= 1 → 0), do not change the TC6M, TC6CK and TFF6 settings.
To start the timer operation (TC6S= 0 → 1), TC6M, TC6CK and TFF6 can be programmed.

Note 4: When TC6M= 1** (upper byte in the 16-bit mode), the source clock becomes the TC5 overflow signal regardless of the TC6CK setting.

Note 5: To use the TimerCounter in the 16-bit mode, select the operating mode by programming TC6M, where TC5CR<TC5M> must be set to 011.

Note 6: To the TimerCounter in the 16-bit mode, select the source clock by programming TC5CR<TC5CK>. Set the timer start control and timer F/F control by programming TC6S and TFF6, respectively.

Note 7: The operating clock settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 11-1 and Table 11-2.

Note 8: The timer register settings are limited depending on the timer operating mode. For the detailed descriptions, see Table 11-3.

Table 11-1 Operating Mode and Selectable Source Clock (NORMAL1/2 and IDLE1/2 Modes)

| Operating mode | $fc/2^{11}$ or $fs/2^3$ | $fc/2^7$ | $fc/2^5$ | $fc/2^3$ | fs | fc/2 | fc | TC5 pin input | TC6 pin input |
|---|---|---|---|---|---|---|---|---|---|
| 8-bit timer | O | O | O | O | – | – | – | – | – |
| 8-bit event counter | – | – | – | – | – | – | – | O | O |
| 8-bit PDO | O | O | O | O | – | – | – | – | – |
| 8-bit PWM | O | O | O | O | O | O | O | – | – |
| 16-bit timer | O | O | O | O | – | – | – | – | – |
| 16-bit event counter | – | – | – | – | – | – | – | O | – |
| Warm-up counter | – | – | – | – | O | – | – | – | – |
| 16-bit PWM | O | O | O | O | O | O | O | O | – |
| 16-bit PPG | O | O | O | O | – | – | – | O | – |

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 (TC5CK).

Note 2: O : Available source clock

Table 11-2 Operating Mode and Selectable Source Clock (SLOW1/2 and SLEEP1/2 Modes)

| Operating mode | $fc/2^{11}$ or $fs/2^3$ | $fc/2^7$ | $fc/2^5$ | $fc/2^3$ | fs | fc/2 | fc | TC5 pin input | TC6 pin input |
|---|---|---|---|---|---|---|---|---|---|
| 8-bit timer | O | – | – | – | – | – | – | – | – |
| 8-bit event counter | – | – | – | – | – | – | – | O | O |
| 8-bit PDO | O | – | – | – | – | – | – | – | – |
| 8-bit PWM | O | – | – | – | O | – | – | – | – |
| 16-bit timer | O | – | – | – | – | – | – | – | – |
| 16-bit event counter | – | – | – | – | – | – | – | O | – |
| Warm-up counter | – | – | – | – | – | – | O | – | – |
| 16-bit PWM | O | – | – | – | O | – | – | O | – |
| 16-bit PPG | O | – | – | – | – | – | – | O | – |

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 (TC5CK).

Note2: O : Available source clock

Table 11-3 Constraints on Register Values Being Compared

| Operating mode | Register Value |
|---|---|
| 8-bit timer/event counter | 1≤ (TTREGn) ≤255 |
| 8-bit PDO | 1≤ (TTREGn) ≤255 |
| 8-bit PWM | 2≤ (PWREGn) ≤254 |
| 16-bit timer/event counter | 1≤ (TTREG6, 5) ≤65535 |
| Warm-up counter | 256≤ (TTREG6, 5) ≤65535 |
| 16-bit PWM | 2≤ (PWREG6, 5) ≤65534 |
| 16-bit PPG | 1≤ (PWREG6, 5) < (TTREG6, 5) ≤65535<br>and<br>(PWREG6, 5) + 1 < (TTREG6, 5) |

Note: n = 5 to 6

## 11.3 Function

The TimerCounter 5 and 6 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 5 and 6 (TC5, 6) 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.

### 11.3.1 8-Bit Timer Mode (TC5 and 6)

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 (TTREGj) value is detected, an INTTCj 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 $\overline{\text{PDOj}}$, $\overline{\text{PWMj}}$ and $\overline{\text{PPGj}}$ pins may output pulses.

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 the programming. Therefore, if TTREGi is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 5, 6

Table 11-4 Source Clock for TimerCounter 5, 6 (Internal Clock)

| Source Clock | | | Resolution | | Maximum Time Setting | |
|---|---|---|---|---|---|---|
| NORMAL1/2, IDLE1/2 mode | | SLOW1/2, SLEEP1/2 mode | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz | fs = 32.768 kHz |
| DV7CK = 0 | DV7CK = 1 | | | | | |
| $fc/2^{11}$ [Hz] | $fs/2^3$ [Hz] | $fs/2^3$ [Hz] | 128 μs | 244.14 μs | 32.6 ms | 62.3 ms |
| $fc/2^7$ | $fc/2^7$ | – | 8 μs | – | 2.0 ms | – |
| $fc/2^5$ | $fc/2^5$ | – | 2 μs | – | 510 μs | – |
| $fc/2^3$ | $fc/2^3$ | – | 500 ns | – | 127.5 μs | – |

Example :Setting the timer mode with source clock $fc/2^7$ Hz and generating an interrupt 80 μs later (TimerCounter6, fc = 16.0 MHz)

```
LD      (TTREG6), 0AH          : Sets the timer register (80 μs÷2^7/fc = 0AH).
DI
SET     (EIRE). 2              : Enables INTTC6 interrupt.
EI
LD      (TC6CR), 00010000B     : Sets the operating clock to fc/2^7, and 8-bit timer mode.
LD      (TC6CR), 00011000B     : Starts TC6.
```

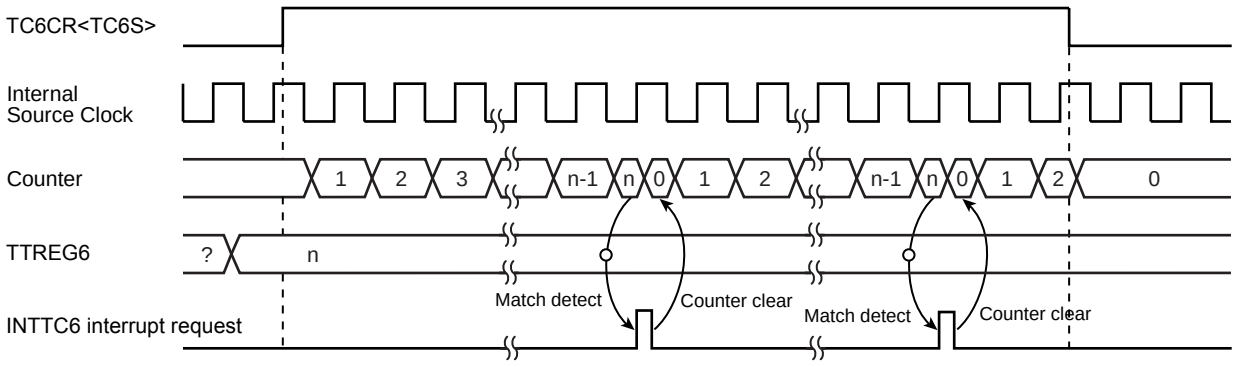



Figure 11-2 8-Bit Timer Mode Timing Chart (TC6)

## 11.3.2 8-Bit Event Counter Mode (TC5, 6)

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 INTTCj 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 $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ 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 $\overline{\text{PDOj, PWMj}}$ and $\overline{\text{PPGj}}$ 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 TTREGi is changed while the timer is running, an expected operation may not be obtained.

Note 3: j = 5, 6

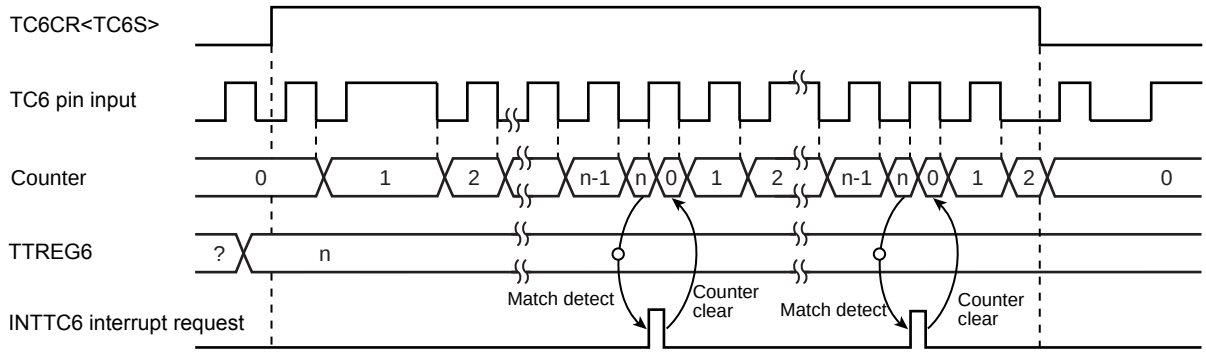



Figure 11-3 8-Bit Event Counter Mode Timing Chart (TC6)

## 11.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC5, 6)

This mode is used to generate a pulse with a 50% duty cycle from the $\overline{\text{PDOj}}$ 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 $\overline{\text{PDOj}}$ 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 $\overline{\text{PDOj}}$ 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 TC6 (fc = 16.0 MHz)

```
                Setting port
LD      (TTREG6),  3DH          : 1/1024÷2^7/fc÷2 = 3DH
LD      (TC6CR),   00010001B    : Sets the operating clock to fc/2^7, and 8-bit PDO mode.
LD      (TC6CR),  00011001B     : Starts TC6.
```

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 $\overline{\text{PDOj}}$ 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 $\overline{\text{PDOj}}$ pin to the high level when the TimerCounter is stopped
CLR (TCjCR).3: Stops the timer.
CLR (TCjCR).7: Sets the $\overline{\text{PDOj}}$ pin to the high level.

Note 3: j = 5, 6

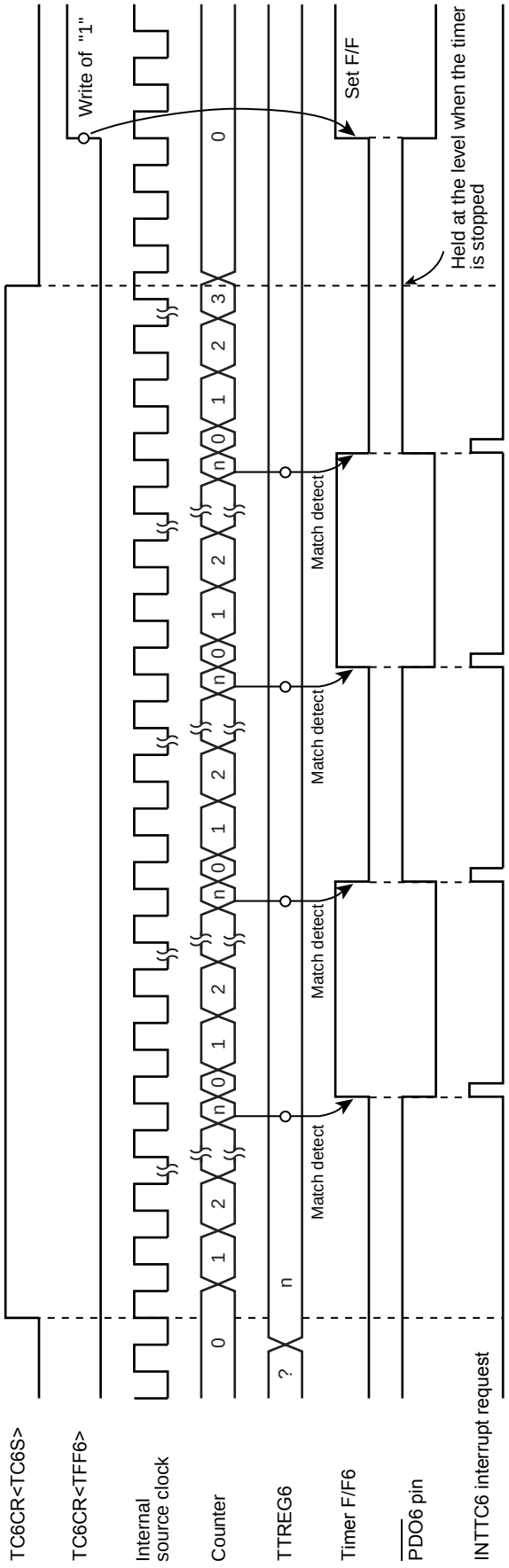



Figure 11-4 8-Bit PDO Mode Timing Chart (TC6)

### 11.3.4 8-Bit Pulse Width Modulation (PWM) Output Mode (TC5, 6)

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 = 5, 6

Table 11-5 PWM Output Mode

| Source Clock | | | Resolution | | Repeated Cycle | |
|---|---|---|---|---|---|---|
| NORMAL1/2, IDLE1/2 mode | | SLOW1/2, SLEEP1/2 mode | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz | fs = 32.768 kHz |
| DV7CK = 0 | DV7CK = 1 | | | | | |
| $fc/2^{11}$ [Hz] | $fs/2^3$ [Hz] | $fs/2^3$ [Hz] | 128 μs | 244.14 μs | 32.8 ms | 62.5 ms |
| $fc/2^7$ | $fc/2^7$ | – | 8 μs | – | 2.05 ms | – |
| $fc/2^5$ | $fc/2^5$ | – | 2 μs | – | 512 μs | – |
| $fc/2^3$ | $fc/2^3$ | – | 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 | – |

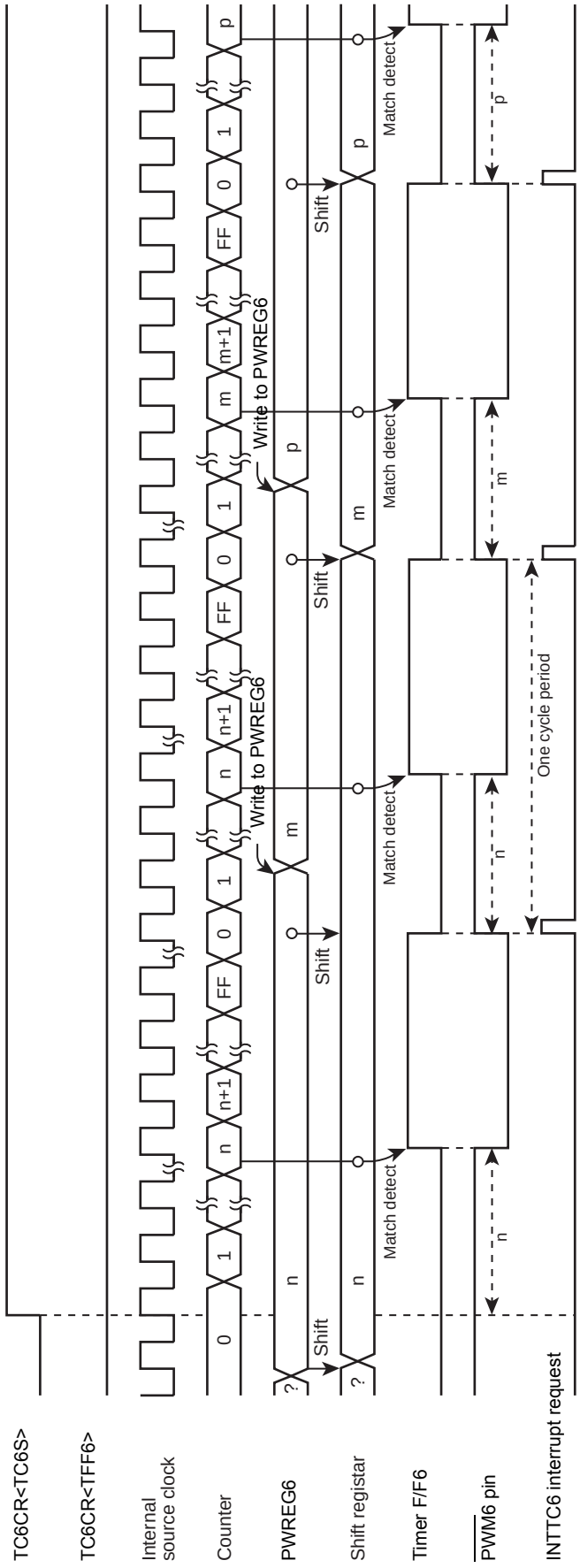



Figure 11-5 8-Bit PWM Mode Timing Chart (TC6)

### 11.3.5 16-Bit Timer Mode (TC5 and 6)

In the timer mode, the up-counter counts up using the internal clock. The TimerCounter 5 and 6 are cascadable to form a 16-bit timer.

When a match between the up-counter and the timer register (TTREG5, TTREG6) value is detected after the timer is started by setting TC6CR<TC6S> to 1, an INTTC6 interrupt is generated and the up-counter is cleared. After being cleared, the up-counter continues counting. Program the lower byte and upper 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 = 5, 6

Table 11-6 Source Clock for 16-Bit Timer Mode

| Source Clock | | | Resolution | | Maximum Time Setting | |
|---|---|---|---|---|---|---|
| NORMAL1/2, IDLE1/2 mode | | SLOW1/2, SLEEP1/2 mode | fc = 16 MHz | fs = 32.768 kHz | fc = 16 MHz | fs = 32.768 kHz |
| DV7CK = 0 | DV7CK = 1 | | | | | |
| $fc/2^{11}$ | $fs/2^3$ | $fs/2^3$ | 128 μs | 244.14 μs | 8.39 s | 16 s |
| $fc/2^7$ | $fc/2^7$ | – | 8 μs | – | 524.3 ms | – |
| $fc/2^5$ | $fc/2^5$ | – | 2 μs | – | 131.1 ms | – |
| $fc/2^3$ | $fc/2^3$ | – | 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     (TTREG5), 927CH     : Sets the timer register (300 ms÷2^7/fc = 927CH).
DI
SET     (EIRE). 2           : Enables INTTC6 interrupt.
EI
LD      (TC5CR), 13H        :Sets the operating clock to fc/2^7, and 16-bit timer mode
                             (lower byte).
LD      (TC6CR), 04H        : Sets the 16-bit timer mode (upper byte).
LD      (TC6CR), 0CH        : Starts the timer.
```

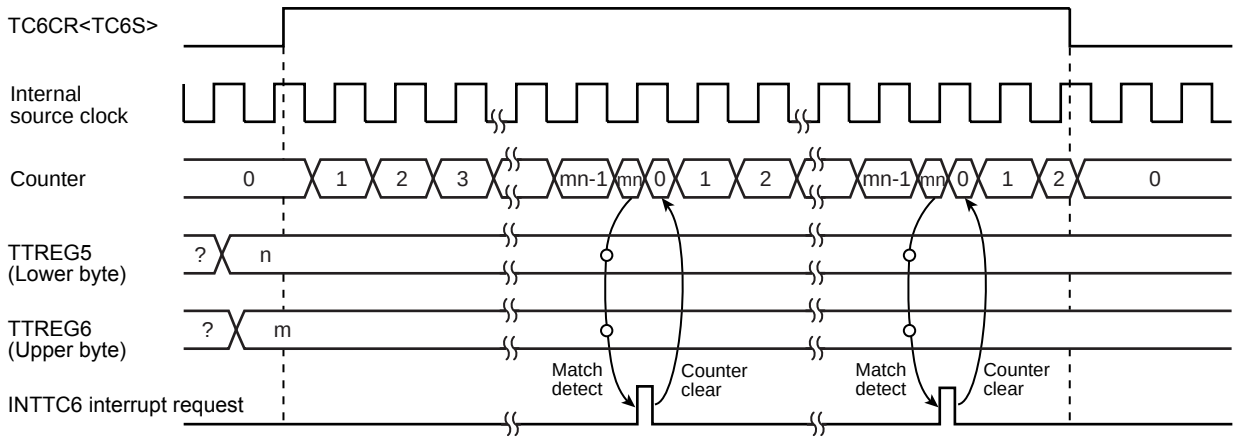



Figure 11-6 16-Bit Timer Mode Timing Chart (TC5 and TC6)

### 11.3.6 16-Bit Event Counter Mode (TC5 and 6)

In the event counter mode, the up-counter counts up at the falling edge to the TC5 pin. The TimerCounter 5 and 6 are cascadable to form a 16-bit event counter.

When a match between the up-counter and the timer register (TTREG5, TTREG6) value is detected after the timer is started by setting TC6CR<TC6S> to 1, an INTTC6 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 TC5 pin. Two machine cycles are required for the low- or high-level pulse input to the TC5 pin.

Therefore, a maximum frequency to be supplied is $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ in the SLOW1/2 or SLEEP1/2 mode. Program the lower byte (TTREG5), and upper byte (TTREG6) 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 = 5, 6

### 11.3.7 16-Bit Pulse Width Modulation (PWM) Output Mode (TC5 and 6)

This mode is used to generate a pulse-width modulated (PWM) signals with up to 16 bits of resolution. The TimerCounter 5 and 6 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 (PWREG5, PWREG6) value is detected, the logic level output from the timer F/F6 is switched to the opposite state. The counter continues counting. The logic level output from the timer F/F6 is switched to the opposite state again by the counter overflow, and the counter is cleared. The INTTC6 interrupt is generated at this time.

Two machine cycles are required for the high- or low-level pulse input to the TC5 pin. Therefore, a maximum frequency to be supplied is $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ to in the SLOW1/2 or SLEEP1/2 mode.

Since the initial value can be set to the timer F/F6 by TC6CR<TFF6>, positive and negative pulses can be generated. Upon reset, the timer F/F6 is cleared to 0.

(The logic level output from the $\overline{PWM}6$ pin is the opposite to the timer F/F6 logic level.)

Since PWREG6 and 5 in the PWM mode are serially connected to the shift register, the values set to PWREG6 and 5 can be changed while the timer is running. The values set to PWREG6 and 5 during a run of the timer are shifted by the INTTCj interrupt request and loaded into PWREG6 and 5. While the timer is stopped, the values are shifted immediately after the programming of PWREG6 and 5. Set the lower byte (PWREG5) and upper byte (PWREG6) in this order to program PWREG6 and 5. (Programming only the lower or upper byte of the register should not be attempted.)

If executing the read instruction to PWREG6 and 5 during PWM output, the values set in the shift register is read, but not the values set in PWREG6 and 5. Therefore, after writing to the PWREG6 and 5, reading data of PWREG6 and 5 is previous value until INTTC6 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 PWREG6 and 5 immediately after the INTTC6 interrupt request is generated (normally in the INTTC6 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 INTTC6 interrupt request is generated.
Note 2: When the timer is stopped during PWM output, the $\overline{PWM}6$ pin holds the output status when the timer is stopped. To change the output status, program TC6CR<TFF6> after the timer is stopped. Do not program TC6CR<TFF6> upon stopping of the timer.
Example: Fixing the $\overline{PWM}6$ pin to the high level when the TimerCounter is stopped

CLR (TC6CR).3: Stops the timer.
CLR (TC6CR).7 : Sets the $\overline{PWM6}$ 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 fc, fc/2 or fs is selected as the source clock, a pulse is output from the $\overline{PWM6}$ pin during the warm-up period time after exiting the STOP mode.

Table 11-7 16-Bit PWM Output Mode

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

Example :Generating a pulse with 1-ms high-level width and a period of 32.768 ms (fc = 16.0 MHz)

```
        Setting ports
LDW     (PWREG5), 07D0H     : Sets the pulse width.
LD      (TC5CR), 33H        : Sets the operating clock to fc/2^3, and 16-bit PWM output
                              mode (lower byte).
LD      (TC6CR), 056H       : Sets TFF6 to the initial value 0, and 16-bit PWM signal
                              generation mode (upper byte).
LD      (TC6CR), 05EH       : Starts the timer.
```

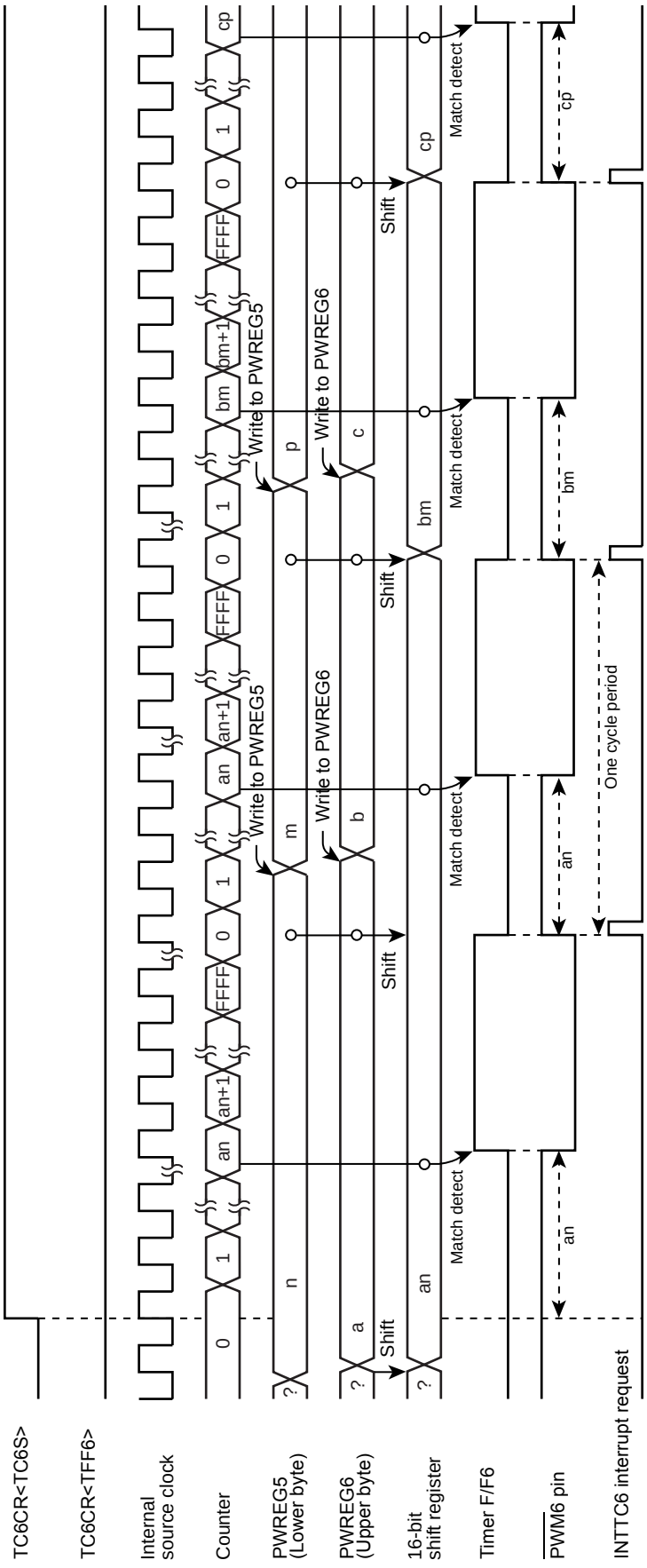



Figure 11-7 16-Bit PWM Mode Timing Chart (TC5 and TC6)

### 11.3.8 16-Bit Programmable Pulse Generate (PPG) Output Mode (TC5 and 6)

This mode is used to generate pulses with up to 16-bits of resolution. The timer counter 5 and 6 are cascadable 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 (PWREG5, PWREG6) value is detected, the logic level output from the timer F/F6 is switched to the opposite state. The counter continues counting. The logic level output from the timer F/F6 is switched to the opposite state again when a match between the up-counter and the timer register (TTREG5, TTREG6) value is detected, and the counter is cleared. The INTTC6 interrupt is generated at this time.

Two machine cycles are required for the high- or low-level pulse input to the TC5 pin. Therefore, a maximum frequency to be supplied is $fc/2^4$ Hz in the NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ to in the SLOW1/2 or SLEEP1/2 mode.

Since the initial value can be set to the timer F/F6 by TC6CR<TFF6>, positive and negative pulses can be generated. Upon reset, the timer F/F6 is cleared to 0.

(The logic level output from the $\overline{PPG}$6 pin is the opposite to the timer F/F6.)

Set the lower byte and upper byte in this order to program the timer register. (TTREG5 → TTREG6, PWREG5 → PWREG6) (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     (PWREG5), 07D0H     : Sets the pulse width.
LDW     (TTREG5), 8002H     : Sets the cycle period.
LD      (TC5CR), 33H        : Sets the operating clock to fc/2^3, and16-bit PPG mode
                              (lower byte).
LD      (TC6CR), 057H       : Sets TFF6 to the initial value 0, and 16-bit
                              PPG mode (upper byte).
LD      (TC6CR), 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{PPG}$6 pin holds the output status when the timer is stopped. To change the output status, program TC6CR<TFF6> after the timer is stopped. Do not change TC6CR<TFF6> upon stopping of the timer.
Example: Fixing the $\overline{PPG}$6 pin to the high level when the TimerCounter is stopped
CLR (TC6CR).3: Stops the timer
CLR (TC6CR).7: Sets the $\overline{PPG}$6 pin to the high level

Note 3: i = 5, 6

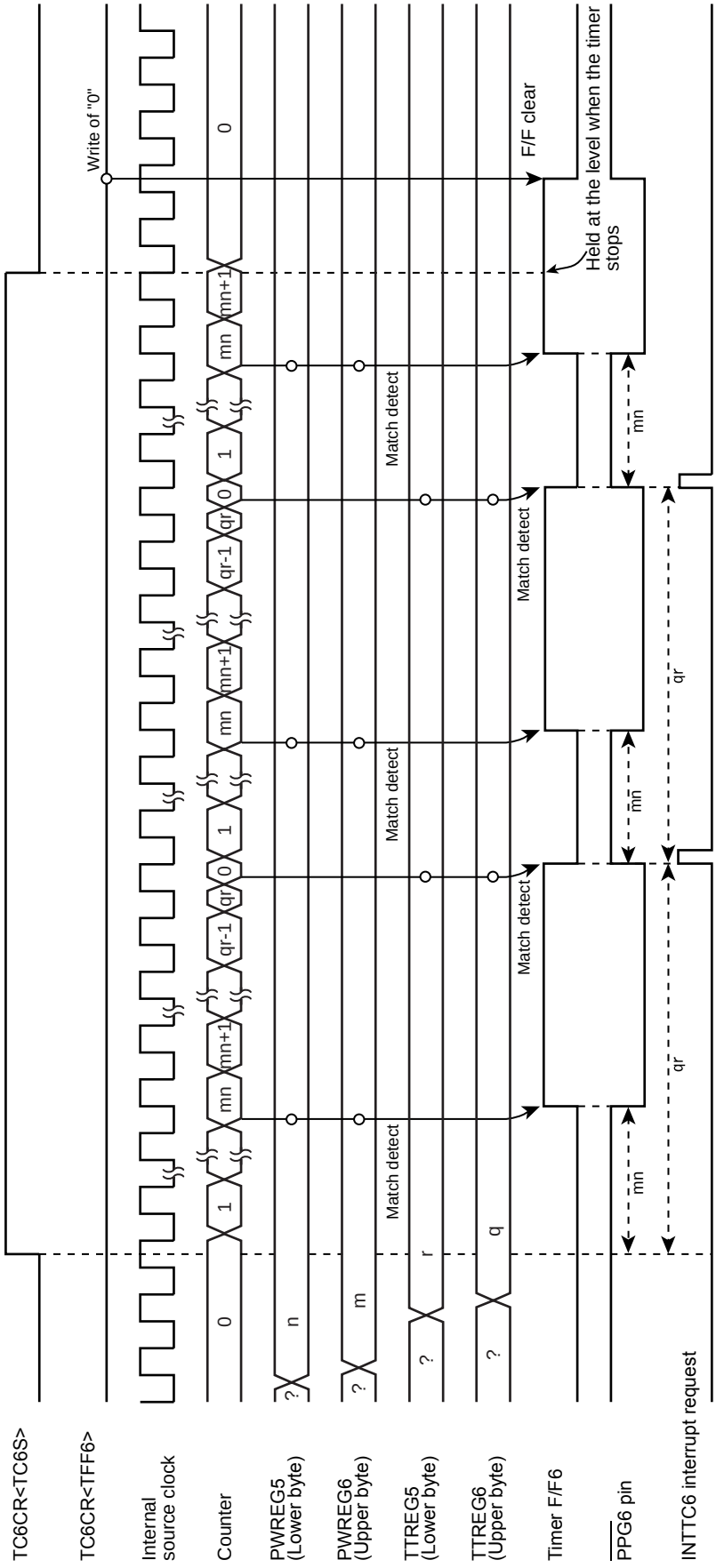



Figure 11-8 16-Bit PPG Mode Timing Chart (TC5 and TC6)

### 11.3.9 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 5 and 6 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 $\overline{\text{PDOi}}$, $\overline{\text{PWMi}}$ and $\overline{\text{PPGi}}$ pins may output pulses.

Note 2: In the warm-up counter mode, only upper 8 bits of the timer register TTREG6 and 5 are used for match detection and lower 8 bits are not used.

Note 3: i = 5, 6

#### 11.3.9.1 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 (TTREG6, 5) value is detected after the timer is started by setting TC6CR<TC6S> to 1, the counter is cleared by generating the INTTC6 interrupt request. After stopping the timer in the INTTC6 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<XEN> to 0 to stop the high-frequency clock.

Table 11-8 Setting Time of Low-Frequency Warm-Up Counter Mode (fs = 32.768 kHz)

| Minimum Time Setting (TTREG6, 5 = 0100H) | Maximum Time Setting (TTREG6, 5 = FF00H) |
|---|---|
| 7.81 ms | 1.99 s |

Example :After checking low-frequency clock oscillation stability with TC6 and 5, switching to the SLOW1 mode

```
                SET     (SYSCR2).6      : SYSCR2<XTEN> ← 1
                LD      (TC5CR), 43H    : Sets TFF5=0, source clock fs, and 16-bit mode.
                LD      (TC6CR), 05H    : Sets TFF6=0, and warm-up counter mode.
                LD      (TTREG5), 8000H : Sets the warm-up time.
                                          (The warm-up time depends on the oscillator characteristic.)
                DI                      : IMF ← 0
                SET     (EIRE). 2       : Enables the INTTC6.
                EI                      : IMF ← 1
                SET     (TC6CR).3       : Starts TC6 and 5.
                  :           :
PINTTC6:        CLR     (TC6CR).3       : Stops TC6 and 5.
                SET     (SYSCR2).5      : SYSCR2<SYSCK> ← 1
                                          (Switches the system clock to the low-frequency clock.)
                CLR     (SYSCR2).7      : SYSCR2<XEN> ← 0 (Stops the high-frequency clock.)
                RETI
                  :           :
VINTTC6:        DW      PINTTC6         : INTTC6 vector table
```

### 11.3.9.2 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 fc 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 (TTREG6, 5) value is detected after the timer is started by setting TC6CR<TC6S> to 1, the counter is cleared by generating the INTTC6 interrupt request. After stopping the timer in the INTTC6 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 11-9 Setting Time in High-Frequency Warm-Up Counter Mode

| Minimum time Setting (TTREG6, 5 = 0100H) | Maximum time Setting (TTREG6, 5 = FF00H) |
|---|---|
| 16 μs | 4.08 ms |

Example :After checking high-frequency clock oscillation stability with TC6 and 5, switching to the NORMAL1 mode

```
                SET     (SYSCR2).7          : SYSCR2<XEN> ← 1
                LD      (TC5CR), 63H        : Sets TFF5=0, source clock fc, and 16-bit mode.
                LD      (TC6CR), 05H        : Sets TFF6=0, and warm-up counter mode.
                LD      (TTREG5), 0F800H    : Sets the warm-up time.
                                              (The warm-up time depends on the oscillator characteristic.)
                DI                          : IMF ← 0
                SET     (EIRE). 2           : Enables the INTTC6.
                EI                          : IMF ← 1
                SET     (TC6CR).3           : Starts the TC6 and 5.
                 :          :
PINTTC6:        CLR     (TC6CR).3           : Stops the TC6 and 5.
                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
                 :          :
VINTTC6:        DW      PINTTC6             : INTTC6 vector table
```

# 12. Asynchronous Serial interface (UART1 )

## 12.1 Configuration

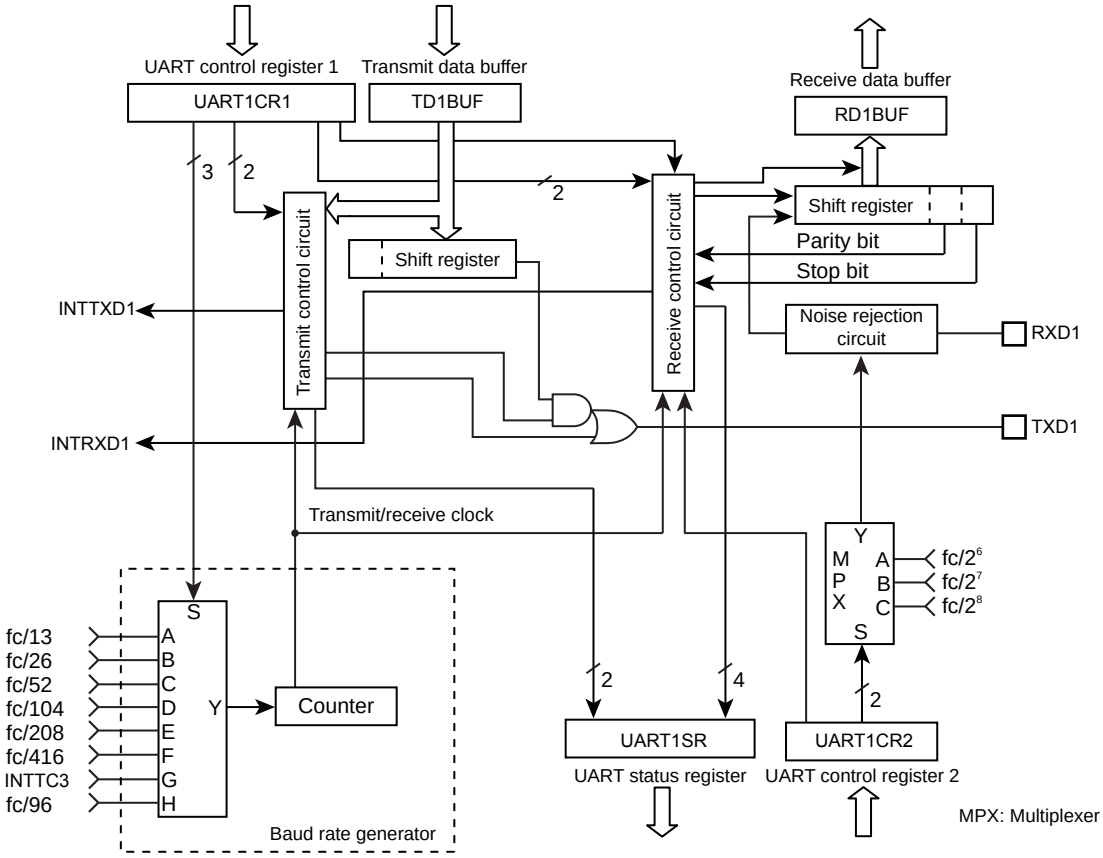



Figure 12-1 UART1 (Asynchronous Serial Interface)

## 12.2 Control

UART1 is controlled by the UART1 Control Registers (UART1CR1, UART1CR2). The operating status can be monitored using the UART status register (UART1SR).

### UART1 Control Register1

UART1CR1 (0F95H)

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

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

Note 3: UART1CR1<RXE> and UART1CR1<TXE> should be set to “0” before UART1CR1<BRG> is changed.

### UART1 Control Register2

UART1CR2 (0F96H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | RXDNC | STOPBR | |

(Initial value: **** *000)

| | | | |
|---|---|---|---|
| RXDNC | Selection of RXD input noise rejection time | 00: No noise rejection (Hysteresis input)<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 UART1CR2<RXDNC> = “01”, pulses longer than 96/fc [s] are always regarded as signals; when UART1CR2<RXDNC> = “10”, longer than 192/fc [s]; and when UART1CR2<RXDNC> = “11”, longer than 384/fc [s].

## UART1 Status Register

UART1SR (0F95H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PERR | FERR | OERR | RBFL | TEND | TBEP | | |

(Initial value: 0000 11**)

| | | | |
|---|---|---|---|
| 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.

## UART1 Receive Data Buffer

RD1BUF (0F97H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Read only

(Initial value: 0000 0000)

## UART1 Transmit Data Buffer

TD1BUF (0F97H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Write only

(Initial value: 0000 0000)

## 12.3 Transfer Data Format

In UART1, an one-bit start bit (Low level), stop bit (Bit length selectable at high level, by UART1CR1<STBT>), and parity (Select parity in UART1CR1<PE>; even- or odd-numbered parity by UART1CR1<EVEN>) are added to the transfer data. The transfer data formats are shown as follows.

| PE | STBT | Frame Length: 1 | 2 | 3 | | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | Start | Bit 0 | Bit 1 | - - - - | Bit 6 | Bit 7 | Stop 1 | | |
| 0 | 1 | Start | Bit 0 | Bit 1 | - - - - | Bit 6 | Bit 7 | Stop 1 | Stop 2 | |
| 1 | 0 | Start | Bit 0 | Bit 1 | - - - - | Bit 6 | Bit 7 | Parity | Stop 1 | |
| 1 | 1 | Start | Bit 0 | Bit 1 | - - - - | Bit 6 | Bit 7 | Parity | Stop 1 | Stop 2 |

Figure 12-2 Transfer Data Format

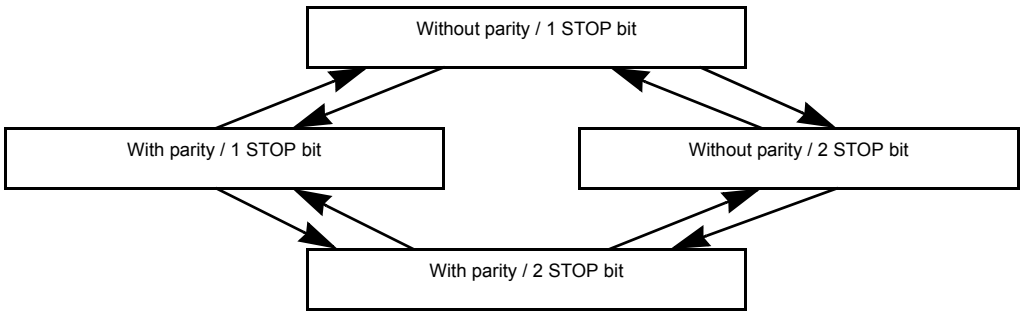



Figure 12-3 Caution on Changing Transfer Data Format

Note: In order to switch the transfer data format, perform transmit operations in the above Figure 12-3 sequence except for the initial setting.

## 12.4 Transfer Rate

The baud rate of UART1 is set of UART1CR1<BRG>. The example of the baud rate are shown as follows.

Table 12-1 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 UART1 transfer rate (when UART1CR1<BRG> = "110"), the transfer clock and transfer rate are determined as follows:

Transfer clock [Hz] = TC3 source clock [Hz] / TTREG3 setting value

Transfer Rate [baud] = Transfer clock [Hz] / 16

## 12.5 Data Sampling Method

The UART1 receiver keeps sampling input using the clock selected by UART1CR1<BRG> until a start bit is detected in RXD1 pin input. RT clock starts detecting "L" level of the RXD1 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).

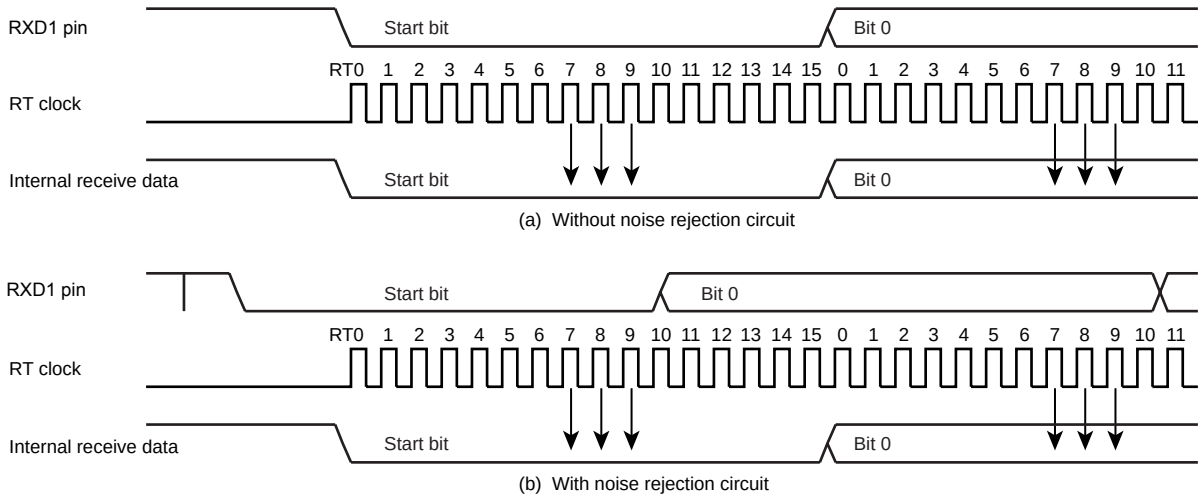



Figure 12-4 Data Sampling Method

## 12.6 STOP Bit Length

Select a transmit stop bit length (1 bit or 2 bits) by UART1CR1<STBT>.

## 12.7 Parity

Set parity / no parity by UART1CR1<PE> and set parity type (Odd- or Even-numbered) by UART1CR1<EVEN>.

## 12.8 Transmit/Receive Operation

### 12.8.1 Data Transmit Operation

Set UART1CR1<TXE> to "1". Read UART1SR to check UART1SR<TBEP> = "1", then write data in TD1BUF (Transmit data buffer). Writing data in TD1BUF zero-clears UART1SR<TBEP>, transfers the data to the transmit shift register and the data are sequentially output from the TXD1 pin. The data output include a one-bit start bit, stop bits whose number is specified in UART1CR1<STBT> and a parity bit if parity addition is specified. Select the data transfer baud rate using UART1CR1<BRG>. When data transmit starts, transmit buffer empty flag UART1SR<TBEP> is set to "1" and an INTTXD1 interrupt is generated.

While UART1CR1<TXE> = "0" and from when "1" is written to UART1CR1<TXE> to when send data are written to TD1BUF, the TXD1 pin is fixed at high level.
When transmitting data, first read UART1SR, then write data in TD1BUF. Otherwise, UART1SR<TBEP> is not zero-cleared and transmit does not start.

### 12.8.2 Data Receive Operation

Set UART1CR1<RXE> to "1". When data are received via the RXD1 pin, the receive data are transferred to RD1BUF (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 RD1BUF (Receive data buffer). Then the receive buffer full flag UART1SR<RBFL> is set and an INTRXD1 interrupt is generated. Select the data transfer baud rate using UART1CR1<BRG>.

If an overrun error (OERR) occurs when data are received, the data are not transferred to RD1BUF (Receive data buffer) but discarded; data in the RD1BUF are not affected.

Note: When a receive operation is disabled by setting UART1CR1<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.

## 12.9 Status Flag

### 12.9.1 Parity Error

When parity determined using the receive data bits differs from the received parity bit, the parity error flag UART1SR<PERR> is set to “1”. The UART1SR<PERR> is cleared to “0” when the RD1BUF is read after reading the UART1SR.

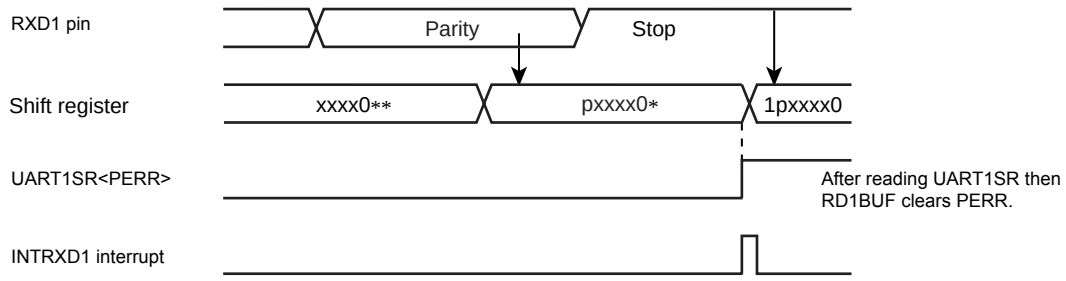



Figure 12-5 Generation of Parity Error

### 12.9.2 Framing Error

When “0” is sampled as the stop bit in the receive data, framing error flag UART1SR<FERR> is set to “1”. The UART1SR<FERR> is cleared to “0” when the RD1BUF is read after reading the UART1SR.

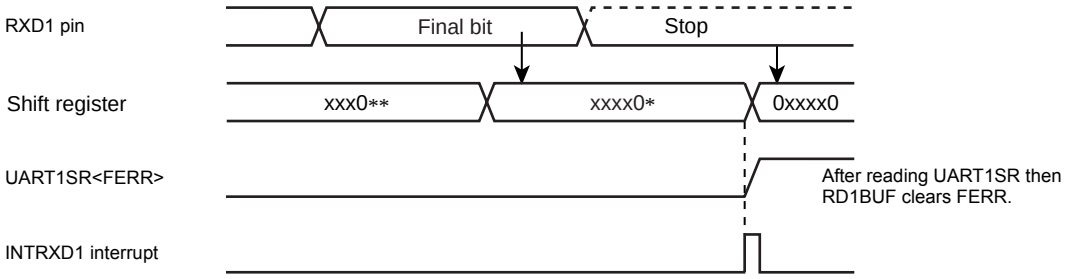



Figure 12-6 Generation of Framing Error

### 12.9.3 Overrun Error

When all bits in the next data are received while unread data are still in RD1BUF, overrun error flag UART1SR<OERR> is set to “1”. In this case, the receive data is discarded; data in RD1BUF are not affected. The UART1SR<OERR> is cleared to “0” when the RD1BUF is read after reading the UART1SR.

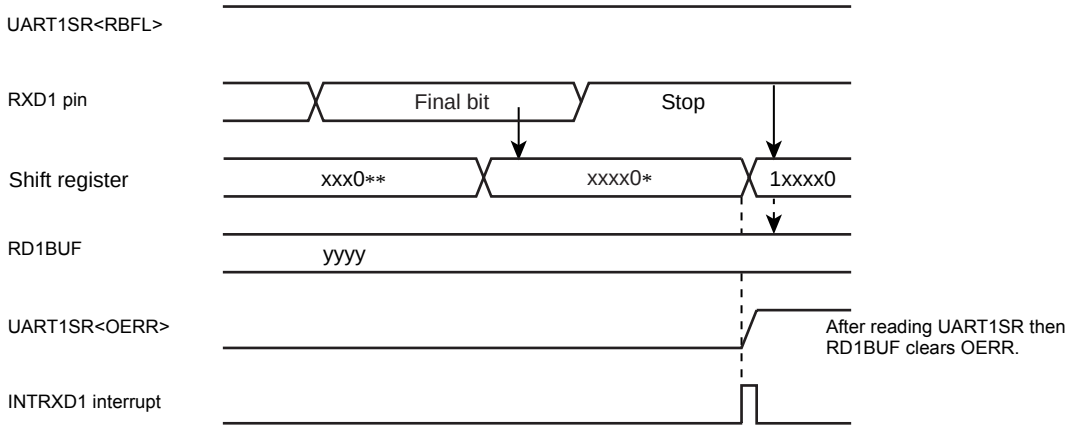



Figure 12-7 Generation of Overrun Error

Note: Receive operations are disabled until the overrun error flag UART1SR<OERR> is cleared.

### 12.9.4 Receive Data Buffer Full

Loading the received data in RD1BUF sets receive data buffer full flag UART1SR<RBFL> to "1". The UART1SR<RBFL> is cleared to “0” when the RD1BUF is read after reading the UART1SR.

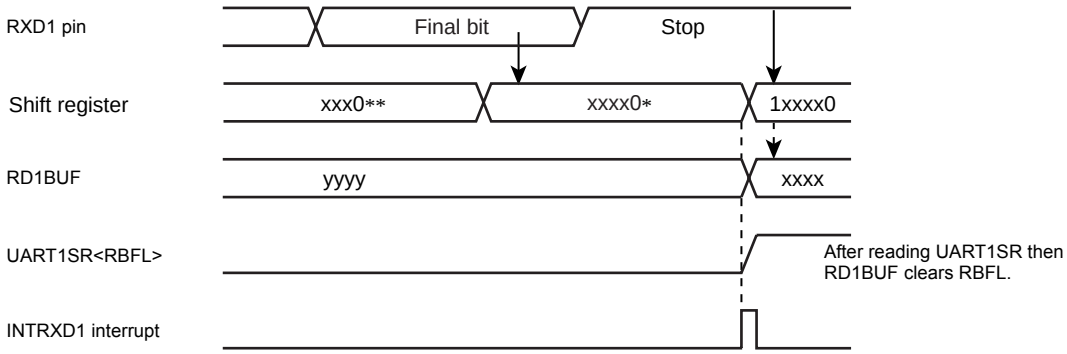



Figure 12-8 Generation of Receive Data Buffer Full

Note: If the overrun error flag UART1SR<OERR> is set during the period between reading the UART1SR and reading the RD1BUF, it cannot be cleared by only reading the RD1BUF. Therefore, after reading the RD1BUF, read the UART1SR again to check whether or not the overrun error flag which should have been cleared still remains set.

### 12.9.5 Transmit Data Buffer Empty

When no data is in the transmit buffer TD1BUF, that is, when data in TD1BUF are transferred to the transmit shift register and data transmit starts, transmit data buffer empty flag UART1SR<TBEP> is set to “1”. The UART1SR<TBEP> is cleared to “0” when the TD1BUF is written after reading the UART1SR.

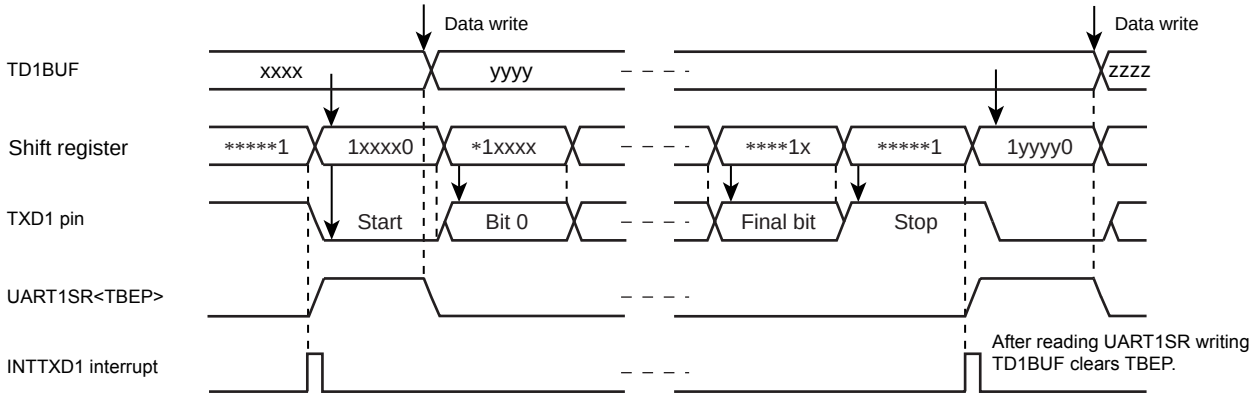



Figure 12-9 Generation of Transmit Data Buffer Empty

### 12.9.6 Transmit End Flag

When data are transmitted and no data is in TD1BUF (UART1SR<TBEP> = "1"), transmit end flag UART1SR<TEND> is set to "1". The UART1SR<TEND> is cleared to "0" when the data transmit is started after writing the TD1BUF.

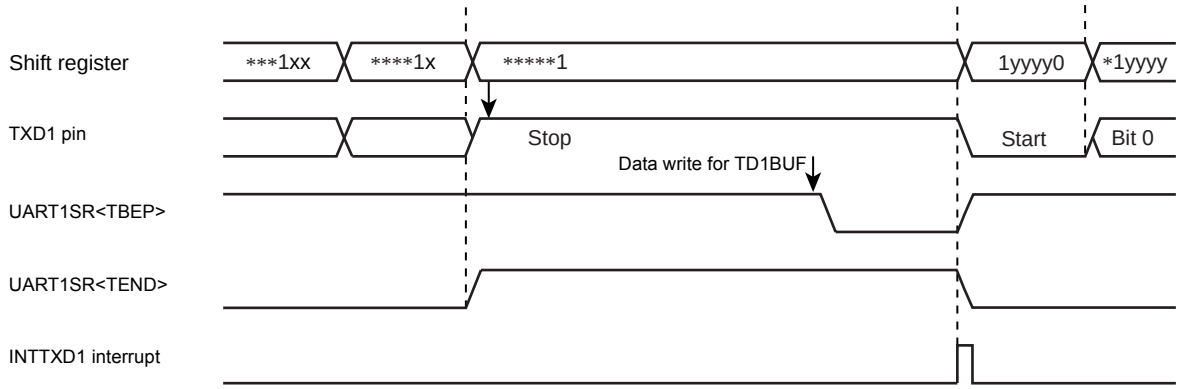



Figure 12-10 Generation of Transmit End Flag and Transmit Data Buffer Empty

# 13. Asynchronous Serial interface (UART2 )

## 13.1 Configuration

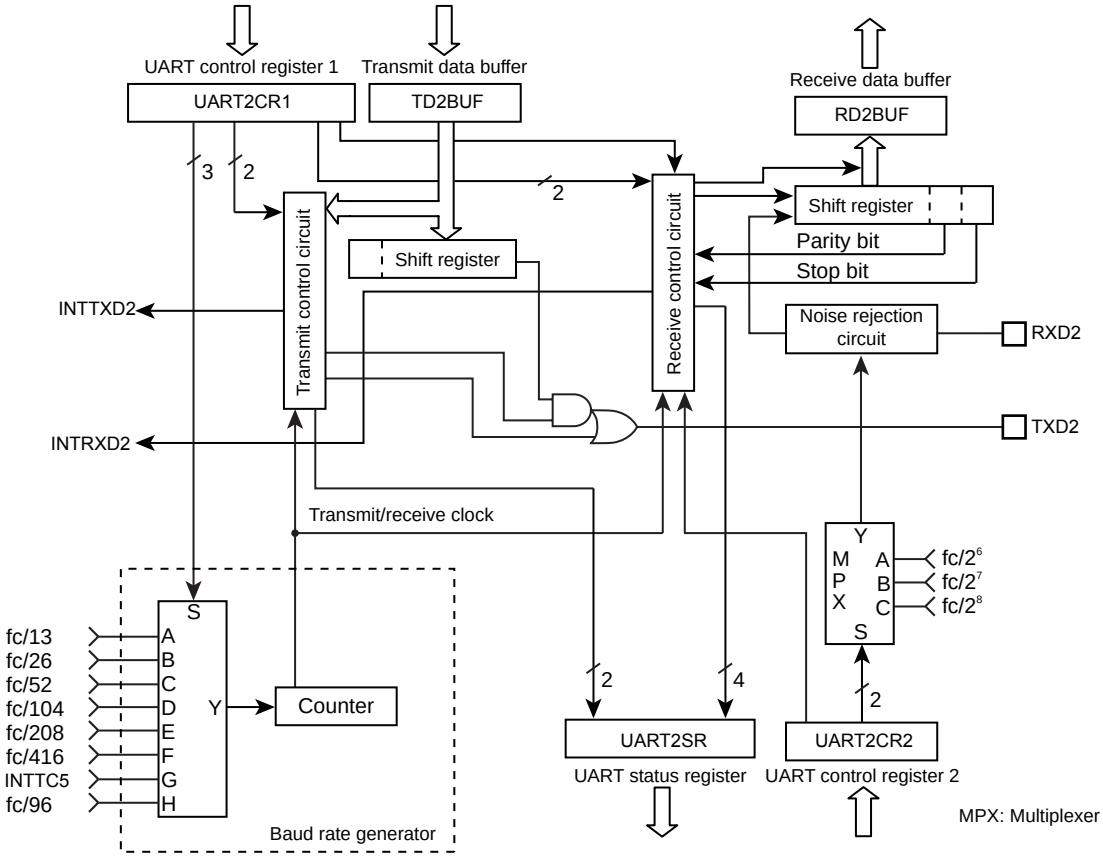



Figure 13-1 UART2 (Asynchronous Serial Interface)

## 13.2 Control

UART2 is controlled by the UART2 Control Registers (UART2CR1, UART2CR2). The operating status can be monitored using the UART status register (UART2SR).

### UART2 Control Register1

UART2CR1 (0F98H)

| 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: TC5 ( Input INTTC5)<br>111: fc/96 | |

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

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

Note 3: UART2CR1<RXE> and UART2CR1<TXE> should be set to “0” before UART2CR1<BRG> is changed.

### UART2 Control Register2

UART2CR2 (0F99H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | RXDNC | STOPBR | |

(Initial value: **** *000)

| | | | |
|---|---|---|---|
| RXDNC | Selection of RXD input noise rejection time | 00: No noise rejection (Hysteresis input)<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 UART2CR2<RXDNC> = “01”, pulses longer than 96/fc [s] are always regarded as signals; when UART2CR2<RXDNC> = “10”, longer than 192/fc [s]; and when UART2CR2<RXDNC> = “11”, longer than 384/fc [s].

## UART2 Status Register

UART2SR (0F98H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PERR | FERR | OERR | RBFL | TEND | TBEP | | |

(Initial value: 0000 11**)

| | | | |
|---|---|---|---|
| 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.

## UART2 Receive Data Buffer

RD2BUF (0F9AH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Read only

(Initial value: 0000 0000)

## UART2 Transmit Data Buffer

TD2BUF (0F9AH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Write only

(Initial value: 0000 0000)

## 13.3 Transfer Data Format

In UART2, an one-bit start bit (Low level), stop bit (Bit length selectable at high level, by UART2CR1<STBT>), and parity (Select parity in UART2CR1<PE>; even- or odd-numbered parity by UART2CR1<EVEN>) are added to the transfer data. The transfer data formats are shown as follows.

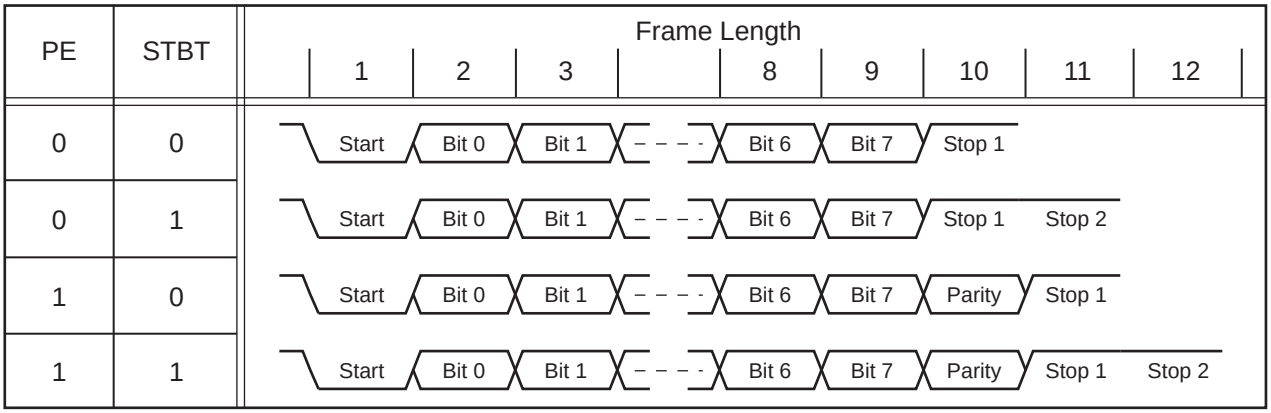



Figure 13-2 Transfer Data Format

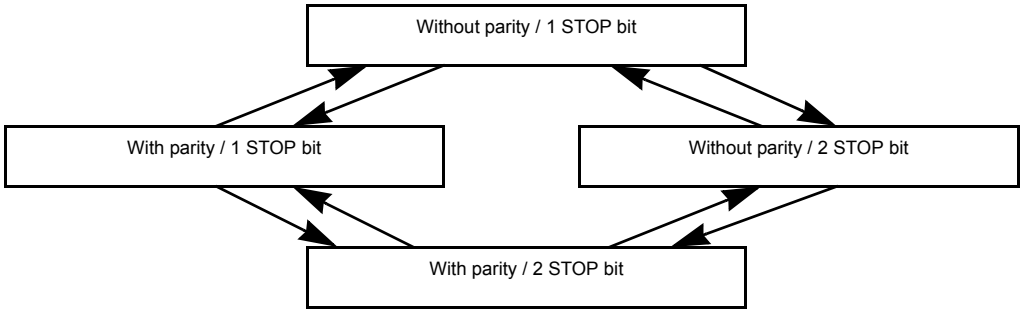



Figure 13-3 Caution on Changing Transfer Data Format

Note: In order to switch the transfer data format, perform transmit operations in the above Figure 13-3 sequence except for the initial setting.

## 13.4 Transfer Rate

The baud rate of UART2 is set of UART2CR1<BRG>. The example of the baud rate are shown as follows.

Table 13-1 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 TC5 is used as the UART2 transfer rate (when UART2CR1<BRG> = "110"), the transfer clock and transfer rate are determined as follows:

Transfer clock [Hz] = TC5 source clock [Hz] / TTREG5 setting value

Transfer Rate [baud] = Transfer clock [Hz] / 16

## 13.5 Data Sampling Method

The UART2 receiver keeps sampling input using the clock selected by UART2CR1<BRG> until a start bit is detected in RXD2 pin input. RT clock starts detecting "L" level of the RXD2 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).

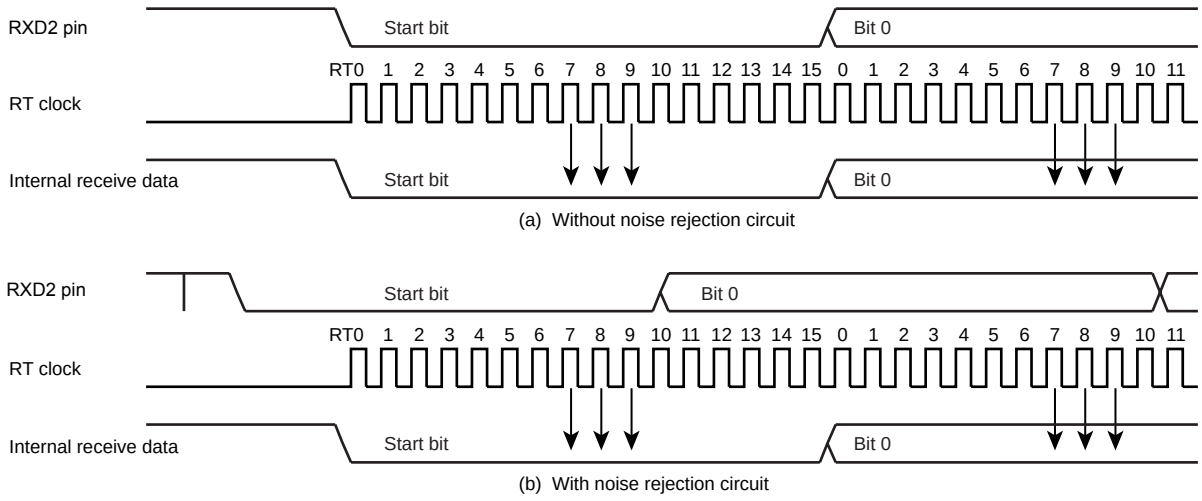



Figure 13-4 Data Sampling Method

## 13.6 STOP Bit Length

Select a transmit stop bit length (1 bit or 2 bits) by UART2CR1<STBT>.

## 13.7 Parity

Set parity / no parity by UART2CR1<PE> and set parity type (Odd- or Even-numbered) by UART2CR1<EVEN>.

## 13.8 Transmit/Receive Operation

### 13.8.1 Data Transmit Operation

Set UART2CR1<TXE> to "1". Read UART2SR to check UART2SR<TBEP> = "1", then write data in TD2BUF (Transmit data buffer). Writing data in TD2BUF zero-clears UART2SR<TBEP>, transfers the data to the transmit shift register and the data are sequentially output from the TXD2 pin. The data output include a one-bit start bit, stop bits whose number is specified in UART2CR1<STBT> and a parity bit if parity addition is specified. Select the data transfer baud rate using UART2CR1<BRG>. When data transmit starts, transmit buffer empty flag UART2SR<TBEP> is set to "1" and an INTTXD2 interrupt is generated.

While UART2CR1<TXE> = "0" and from when "1" is written to UART2CR1<TXE> to when send data are written to TD2BUF, the TXD2 pin is fixed at high level.
When transmitting data, first read UART2SR, then write data in TD2BUF. Otherwise, UART2SR<TBEP> is not zero-cleared and transmit does not start.

### 13.8.2 Data Receive Operation

Set UART2CR1<RXE> to "1". When data are received via the RXD2 pin, the receive data are transferred to RD2BUF (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 RD2BUF (Receive data buffer). Then the receive buffer full flag UART2SR<RBFL> is set and an INTRXD2 interrupt is generated. Select the data transfer baud rate using UART2CR1<BRG>.

If an overrun error (OERR) occurs when data are received, the data are not transferred to RD2BUF (Receive data buffer) but discarded; data in the RD2BUF are not affected.

Note: When a receive operation is disabled by setting UART2CR1<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.

## 13.9 Status Flag

### 13.9.1 Parity Error

When parity determined using the receive data bits differs from the received parity bit, the parity error flag UART2SR<PERR> is set to “1”. The UART2SR<PERR> is cleared to “0” when the RD2BUF is read after reading the UART2SR.

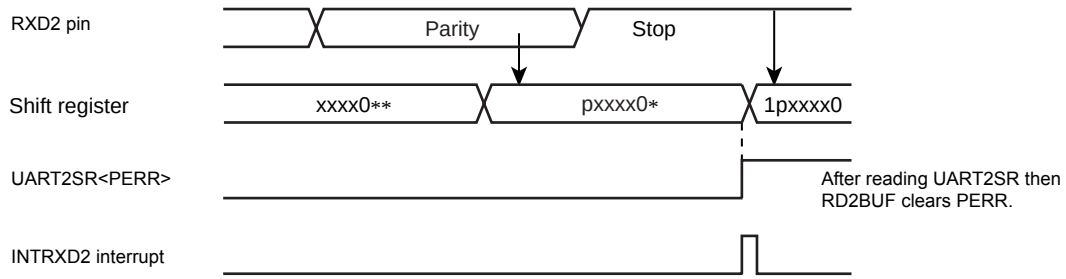



Figure 13-5 Generation of Parity Error

### 13.9.2 Framing Error

When “0” is sampled as the stop bit in the receive data, framing error flag UART2SR<FERR> is set to “1”. The UART2SR<FERR> is cleared to “0” when the RD2BUF is read after reading the UART2SR.

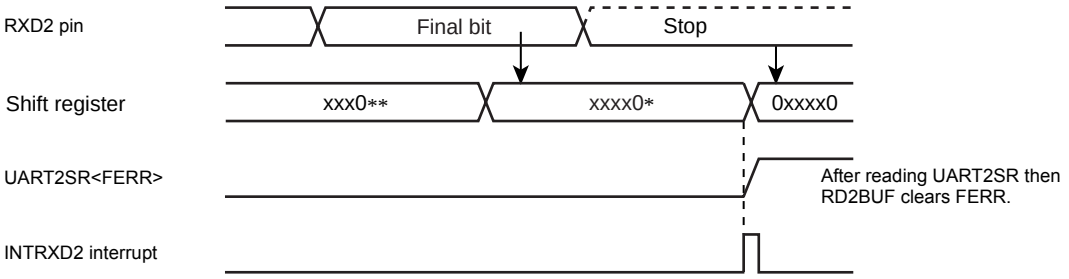



Figure 13-6 Generation of Framing Error

### 13.9.3 Overrun Error

When all bits in the next data are received while unread data are still in RD2BUF, overrun error flag UART2SR<OERR> is set to “1”. In this case, the receive data is discarded; data in RD2BUF are not affected. The UART2SR<OERR> is cleared to “0” when the RD2BUF is read after reading the UART2SR.

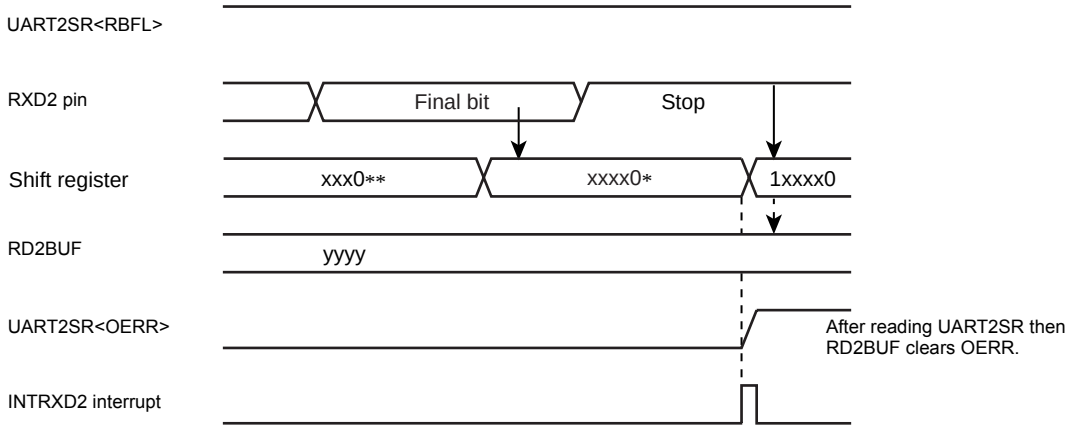



Figure 13-7 Generation of Overrun Error

Note: Receive operations are disabled until the overrun error flag UART2SR<OERR> is cleared.

### 13.9.4 Receive Data Buffer Full

Loading the received data in RD2BUF sets receive data buffer full flag UART2SR<RBFL> to "1". The UART2SR<RBFL> is cleared to “0” when the RD2BUF is read after reading the UART2SR.

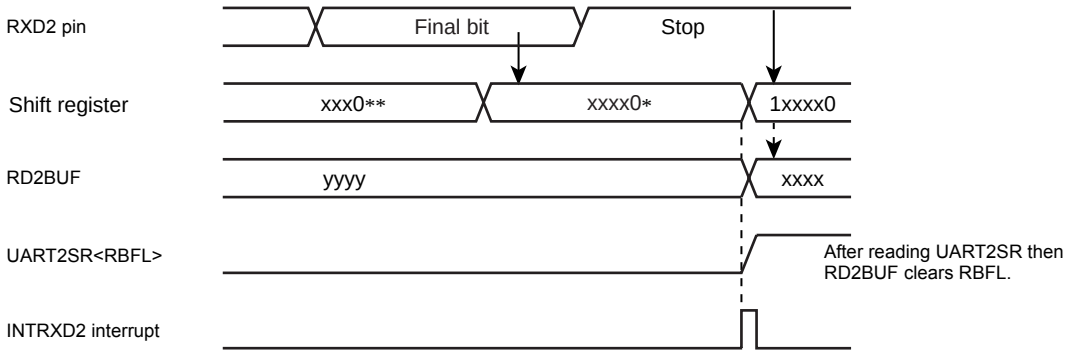



Figure 13-8 Generation of Receive Data Buffer Full

Note: If the overrun error flag UART2SR<OERR> is set during the period between reading the UART2SR and reading the RD2BUF, it cannot be cleared by only reading the RD2BUF. Therefore, after reading the RD2BUF, read the UART2SR again to check whether or not the overrun error flag which should have been cleared still remains set.

### 13.9.5 Transmit Data Buffer Empty

When no data is in the transmit buffer TD2BUF, that is, when data in TD2BUF are transferred to the transmit shift register and data transmit starts, transmit data buffer empty flag UART2SR<TBEP> is set to “1”. The UART2SR<TBEP> is cleared to “0” when the TD2BUF is written after reading the UART2SR.

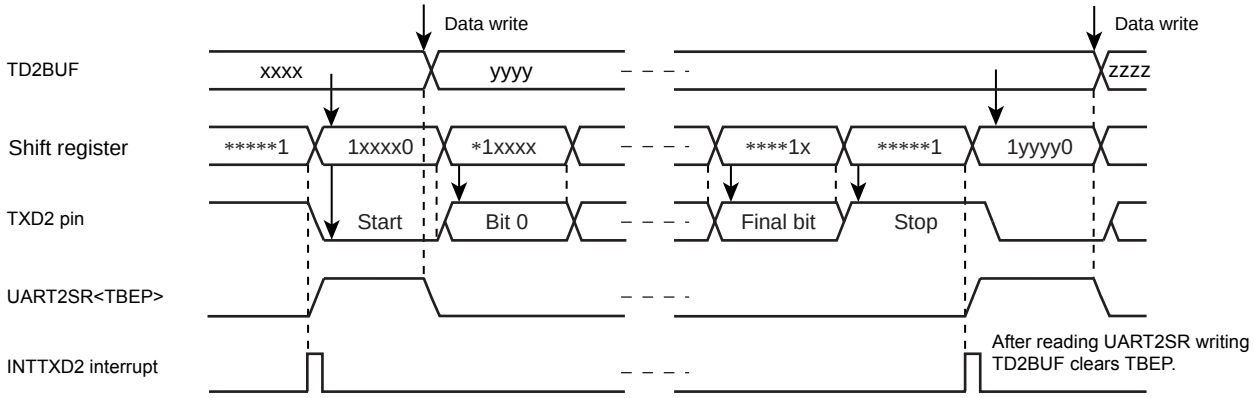



Figure 13-9 Generation of Transmit Data Buffer Empty

### 13.9.6 Transmit End Flag

When data are transmitted and no data is in TD2BUF (UART2SR<TBEP> = "1"), transmit end flag UART2SR<TEND> is set to "1". The UART2SR<TEND> is cleared to "0" when the data transmit is started after writing the TD2BUF.

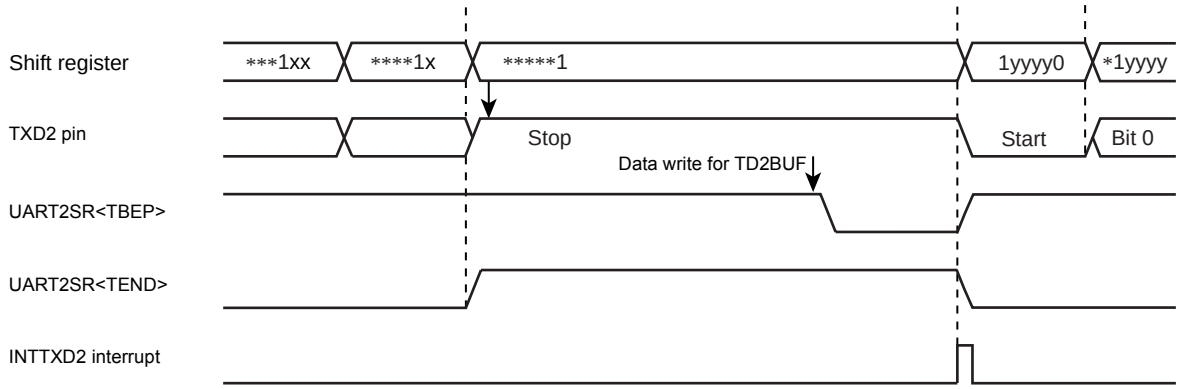


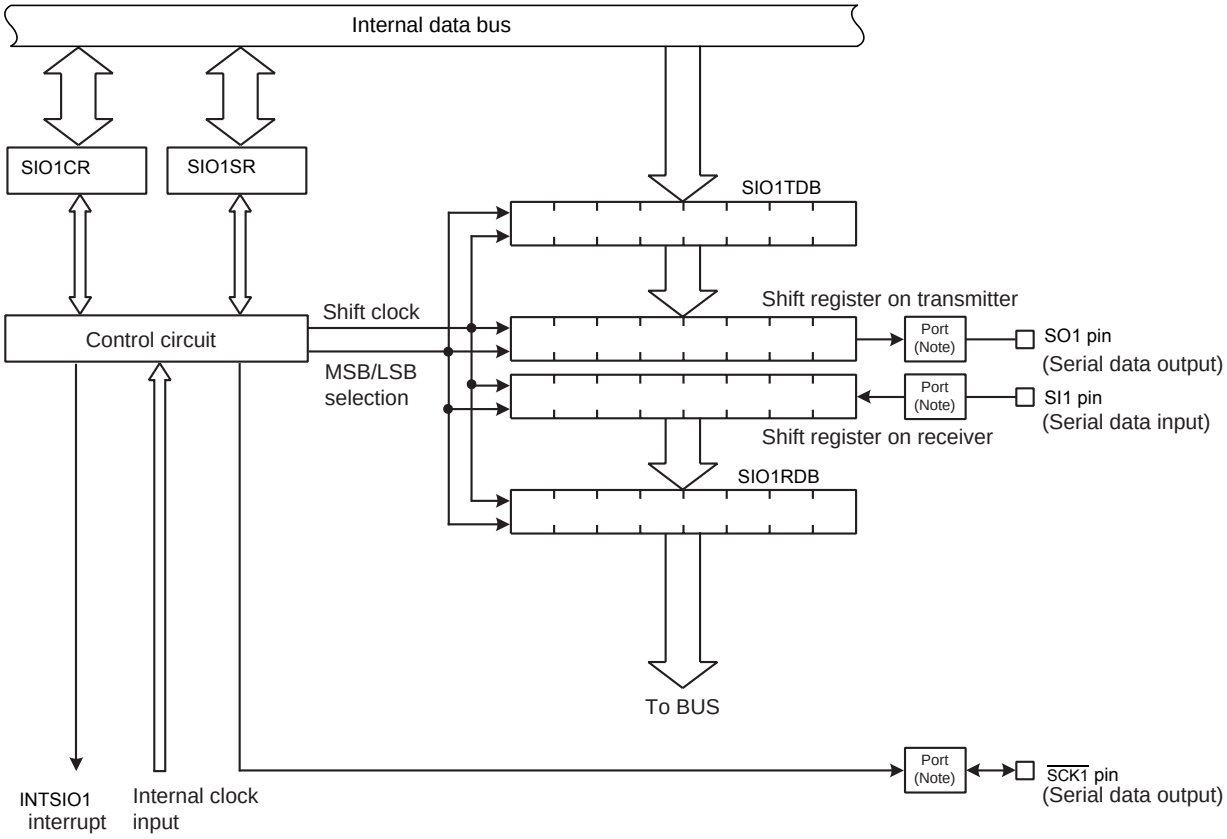

Figure 13-10 Generation of Transmit End Flag and Transmit Data Buffer Empty

# 14. Synchronous Serial Interface (SIO1)

The serial interfaces connect to an external device via SI1, SO1, and $\overline{SCK1}$ pins.

When these pins are used as serial interface, the output latches for each port should be set to "1".

## 14.1 Configuration




Note: Set the register of port correctly for the port assigned as serial interface pins.
For details, see the description of the input/output port control register.

Figure 14-1 Synchronous Serial Interface (SIO)

## 14.2 Control

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

### Serial Interface Control Register

SIO1CR (0020H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SIOS | SIOINH | SIOM | | SIODIR | SCK | | |

(Initial value: 0000 0000)

| | | | | | |
|---|---|---|---|---|---|
| SIOS | Specify start/stop of transfer | 0: Stop<br>1: Start | | | R/W |
| SIOINH | Forcibly stops transfer (Note 1) | 0: –<br>1: Forcibly stop (Automatically cleared to "0" after stopping) | | | |
| SIOM | Selects transfer mode | 00: Transmit mode<br>01: Receive mode<br>10: Transmit/receive mode<br>11: Reserved | | | |
| SIODIR | Selects direction of transfer | 0: MSB (Transfer beginning with bit7)<br>1: LSB (Transfer beginning with bit0) | | | |
| SCK | Selects serial clock | | NORMAL1/2 or IDLE1/2 modes | | SLOW/SLEEP mode |
| | | | TBTCR <DV7CK> = "0" | TBTCR <DV7CK> = "1" | |
| | | 000 | $fc/2^{12}$ | $fs/2^4$ | $fs/2^4$ |
| | | 001 | $fc/2^8$ | $fc/2^8$ | Reserved |
| | | 010 | $fc/2^7$ | $fc/2^7$ | Reserved |
| | | 011 | $fc/2^6$ | $fc/2^6$ | Reserved |
| | | 100 | $fc/2^5$ | $fc/2^5$ | Reserved |
| | | 101 | $fc/2^4$ | $fc/2^4$ | Reserved |
| | | 110 | $fc/2^3$ | $fc/2^3$ | Reserved |
| | | 111 | External clock (Input from $\overline{SCK1}$ pin) | | |

Note 1: When SIO1CR<SIOINH> is set to “1”, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

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

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

### Serial Interface Status Register

SIO1SR (0021H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SIOF | SEF | TXF | RXF | TXERR | RXERR | | |

(Initial value: 0010 00**)

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

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

Note 2: *: Don't care

### Receive buffer register

SIO1RDB (0022H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Read only

(Initial value: 0000 0000)

### Transmit buffer register

SIO1TDB (0022H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Write only

(Initial value: **** ****)

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

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

Note 3: *: Don't care

# 14.3 Function

## 14.3.1 Serial clock

### 14.3.1.1 Clock source

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

(1) Internal clock

Setting the SIO1CR<SCK> to other than "111B" outputs the clock (shown in " Table 14-1 Serial Clock Rate (fc = 16 MHz, fs = 32.768kHz) ") as serial clock outputs from $\overline{SCK1}$ pin. At the before beginning or finishing of a transfer, $\overline{SCK1}$ pin is kept in high level.

When writing (in the transmit mode) or reading (in the receive mode) data can not follow the serial clock rate, an automatic-wait function is executed to stop the serial clock automatically and hold the next shift operation until reading or writing is completed (shown in " Figure 14-2 Automatic-wait Function (Example of transmit mode) "). The maximum time from releasing the automatic-wait function by reading or writing a data is 1 cycle of the selected serial clock until the serial clock comes out from $\overline{SCK1}$ pin.

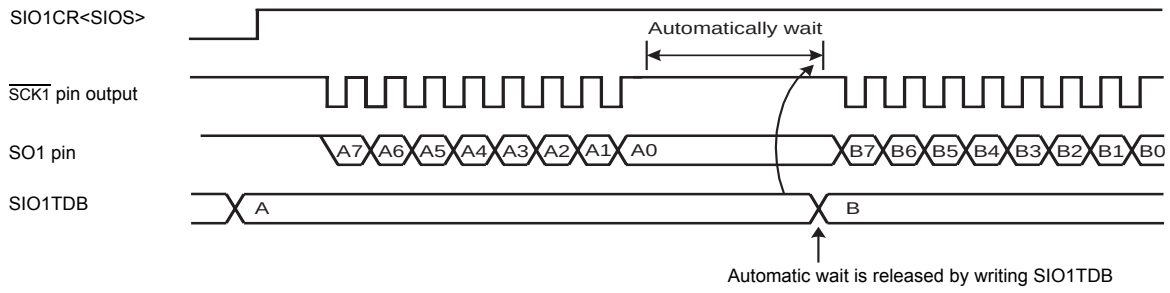



Figure 14-2 Automatic-wait Function (Example of transmit mode)

Table 14-1 Serial Clock Rate (fc = 16 MHz, fs = 32.768kHz)

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

(2) External clock

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

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

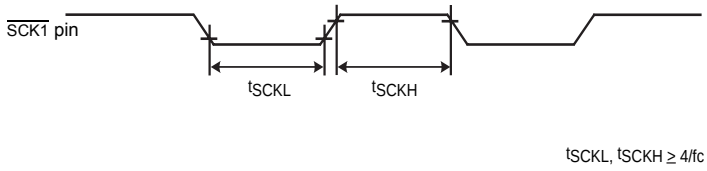



Figure 14-3 External Clock

### 14.3.1.2 Shift edge

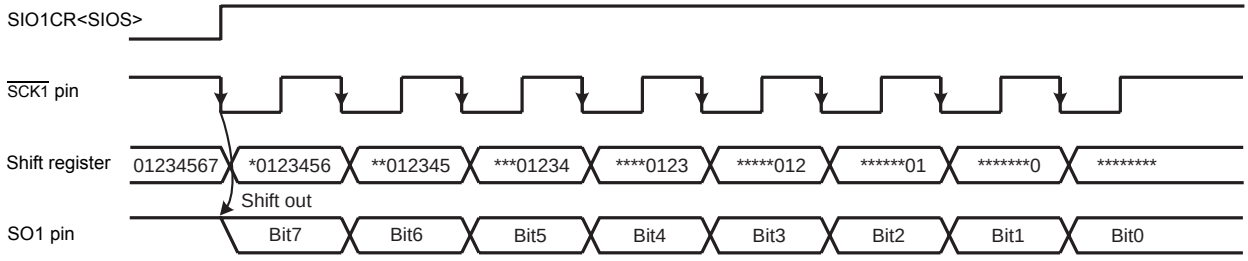
The leading edge is used to transmit data, and the trailing edge is used to receive data.

(1) Leading edge shift

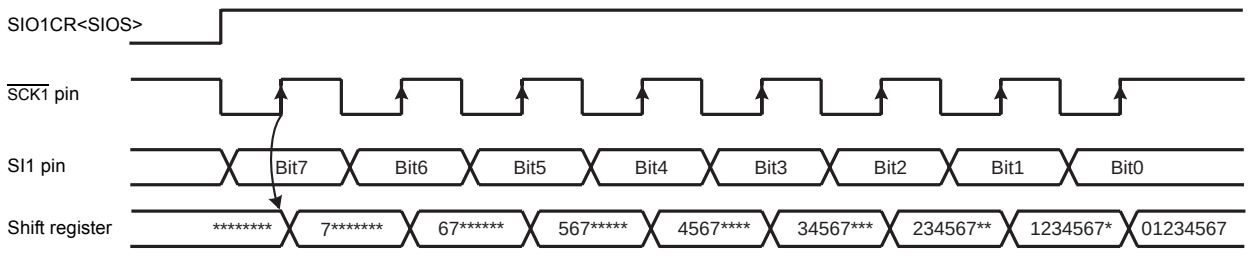
Data is shifted on the leading edge of the serial clock (falling edge of the $\overline{SCK1}$ pin input/output).

(2) Trailing edge shift

Data is shifted on the trailing edge of the serial clock (rising edge of the $\overline{SCK1}$ pin input/output).




(a) Leading edge shift (Example of MSB transfer)




(b) Trailing edge shift (Example of MSB transfer)

Figure 14-4 Shift Edge

### 14.3.2 Transfer bit direction

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

When the data direction is changed, the writing instruction to SIO1CR<SIODIR> should be executed while the transfer is stopped (when SIO1CR<SIOF>= “0”)

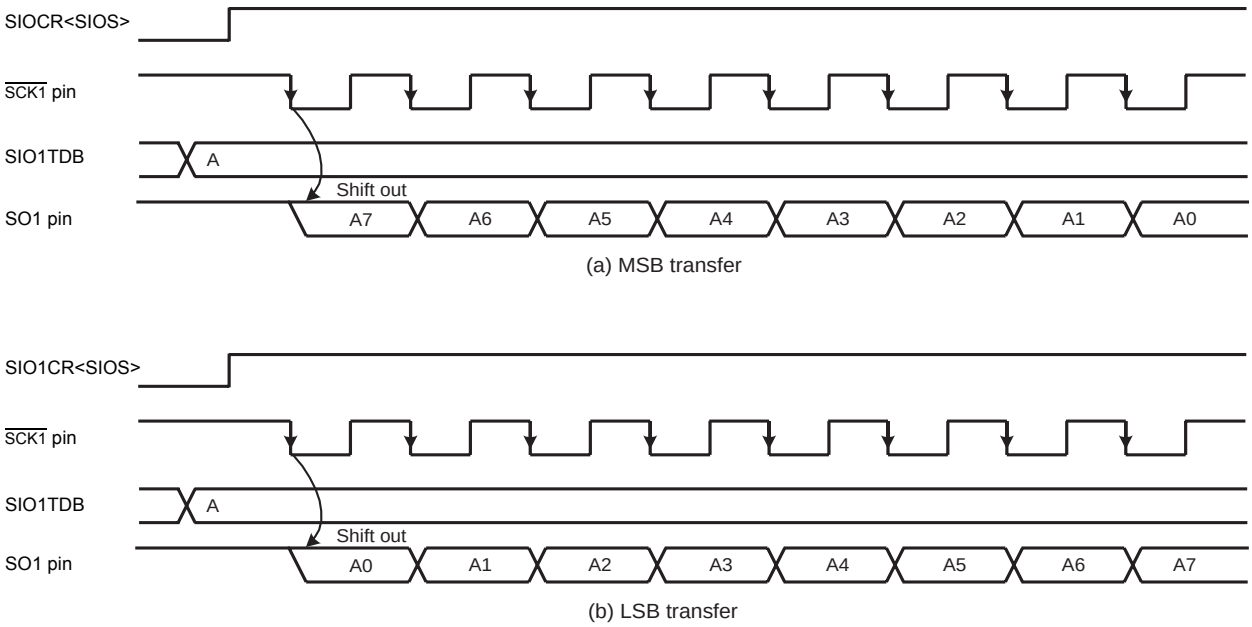



Figure 14-5 Transfer Bit Direction (Example of transmit mode)

#### 14.3.2.1 Transmit mode

##### (1) MSB transmit mode

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

##### (2) LSB transmit mode

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

#### 14.3.2.2 Receive mode

##### (1) MSB receive mode

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

(2) LSB receive mode

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

#### 14.3.2.3 Transmit/receive mode

(1) MSB transmit/receive mode

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

(2) LSB transmit/receive mode

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

### 14.3.3 Transfer modes

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

#### 14.3.3.1 Transmit mode

Transmit mode is selected by writing “00B” to SIO1CR<SIOM>.

(1) Starting the transmit operation

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

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

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

The data is transferred sequentially starting from SO1 pin with the direction of the bit specified by SIO1CR<SIODIR>, synchronizing with the $\overline{\text{SCK1}}$ pin's falling edge.

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

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

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

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

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

(2) During the transmit operation

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

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

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

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

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

(3) Stopping the transmit operation

There are two ways for stopping transmits operation.

- The way of clearing SIO1CR<SIOS>.
  When SIO1CR<SIOS> is cleared to “0”, transmit operation is stopped after all transfer of the data is finished. When transmit operation is finished, SIO1SR<SIOF> is cleared to “0” and SO1 pin is kept in high level.
  In external clock operation, SIO1CR<SIOS> must be cleared to “0” before SIO1SR<SEF> is set to “1” by beginning next transfer.
- The way of setting SIO1CR<SIOINH>.
  Transmit operation is stopped immediately after SIO1CR<SIOINH> is set to “1”. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

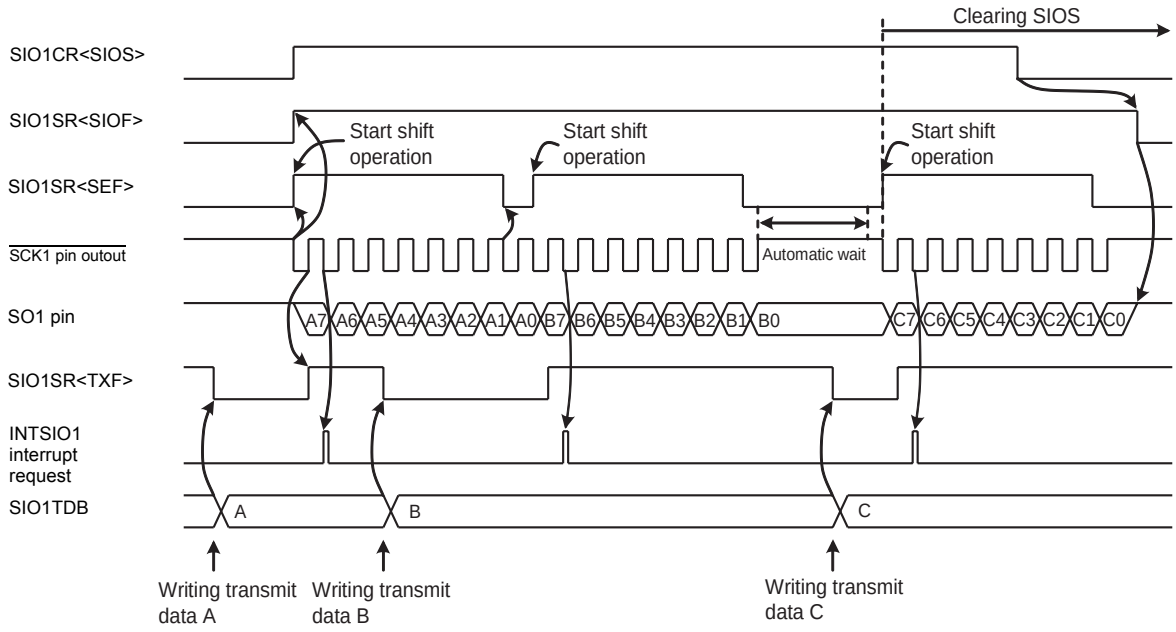



Figure 14-6 Example of Internal Clock and MSB Transmit Mode

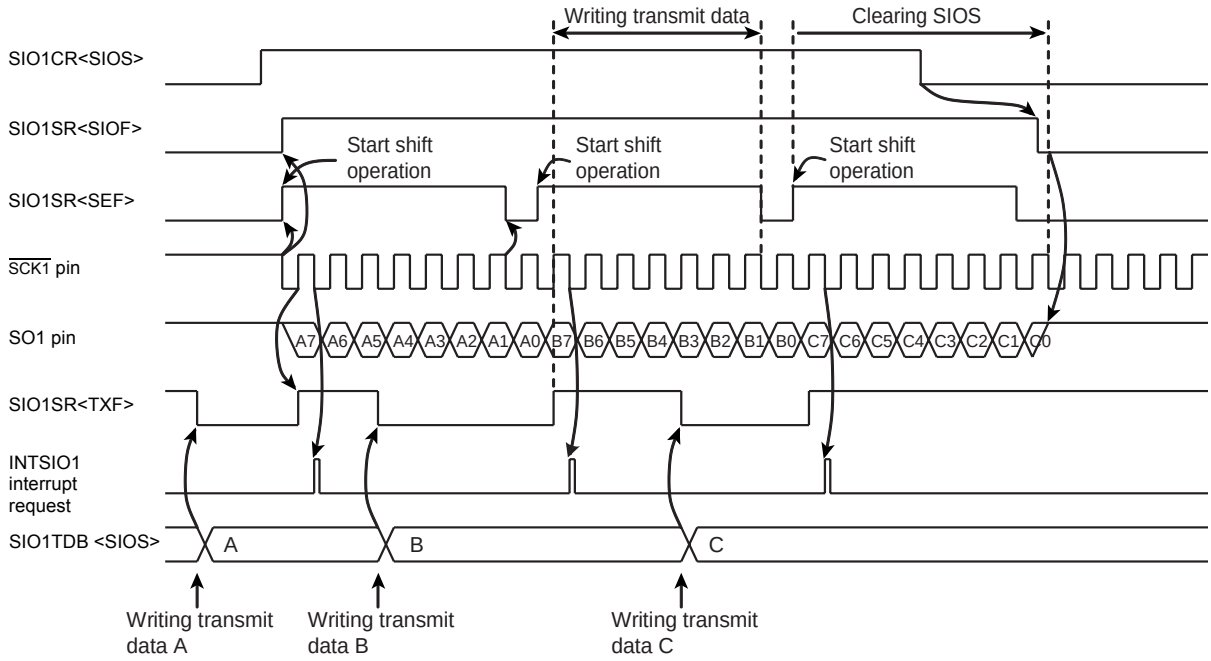



Figure 14-7 Exaple of External Clock and MSB Transmit Mode

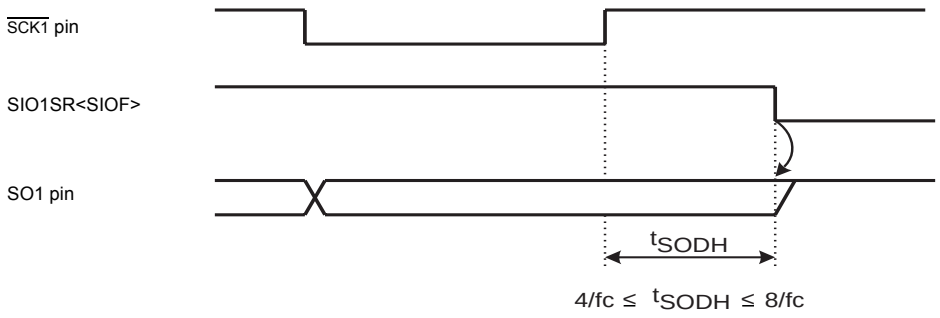



Figure 14-8 Hold Time of the End of Transmit Mode

(4) Transmit error processing

Transmit errors occur on the following situation.

- Shift operation starts before writing next transmit data to SIO1TDB in external clock operation.
  If transmit errors occur during transmit operation, SIO1SR<TXERR> is set to “1” immediately after starting shift operation. Synchronizing with the next serial clock falling edge, INTSIO1 interrupt request is generated.
  If shift operation starts before writing data to SIO1TDB after SIO1CR<SIOS> is set to “1”, SIO1SR<TXERR> is set to “1” immediately after shift operation is started and then INTSIO1 interrupt request is generated.
  SIO1 pin is kept in high level when SIO1SR<TXERR> is set to “1”. When transmit error occurs, transmit operation must be forcibly stop by writing SIO1CR<SIOINH> to “1”. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

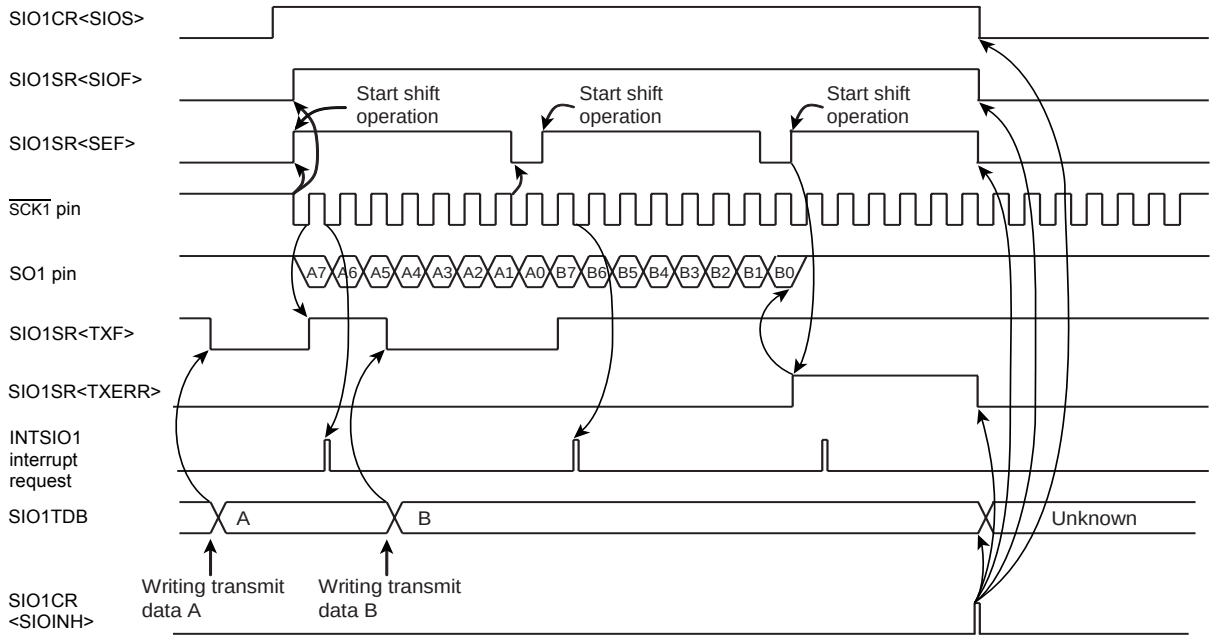



Figure 14-9 Example of Transmit Error Processingme

### 14.3.3.2 Receive mode

The receive mode is selected by writing “01B” to SIO1CR<SIOM>.

#### (1) Starting the receive operation

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

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

Synchronizing with the $\overline{\text{SCK1}}$ pin's rising edge, the data is received sequentially from SI1 pin with the direction of the bit specified by SBI1DIR<SIODIR>.

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

When 8-bit data is received, the data is transferred to SIO1RDB from shift register. INTSIO1 interrupt request is generated and SIO1SR<RXF> is set to “1”

Note: In internal clock operation, when the SIO1CR<SIOS> is set to "1", the serial clock is generated from $\overline{\text{SCK1}}$ pin after maximum 1-cycle of serial clock frequency.

#### (2) During the receive operation

The SIO1SR<RXF> is cleared to “0” by reading a data from SIO1RDB.

In the internal clock operation, the serial clock stops to “H” level by an automatic-wait function when the all of the 8-bit data has been received. Automatic-wait function is released by reading a received data from SIO1RDB. Then, receive operation is restarted after maximum 1-cycle of serial clock.

In external clock operation, after SIO1SR<RXF> is set to “1”, the received data must be read from SIO1RDB, before the next data shift-in operation is finished.

If received data is not read out from SIO1RDB receive error occurs immediately after shift operation is finished. Then INTSIO1 interrupt request is generated after SIO1SR<RXERR> is set to "1".

(3) Stopping the receive operation

There are two ways for stopping the receive operation.

- The way of clearing SIO1CR<SIOS>.
  When SIO1CR<SIOS> is cleared to "0", receive operation is stopped after all of the data is finished to receive. When receive operation is finished, SIO1SR<SIOF> is cleared to "0". In external clock operation, SIO1CR<SIOS> must be cleared to "0" before SIO1SR<SEF> is set to "1" by starting the next shift operation.
- The way of setting SIO1CR<SIOINH>.
  Receive operation is stopped immediately after SIO1CR<SIOINH> is set to "1". In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

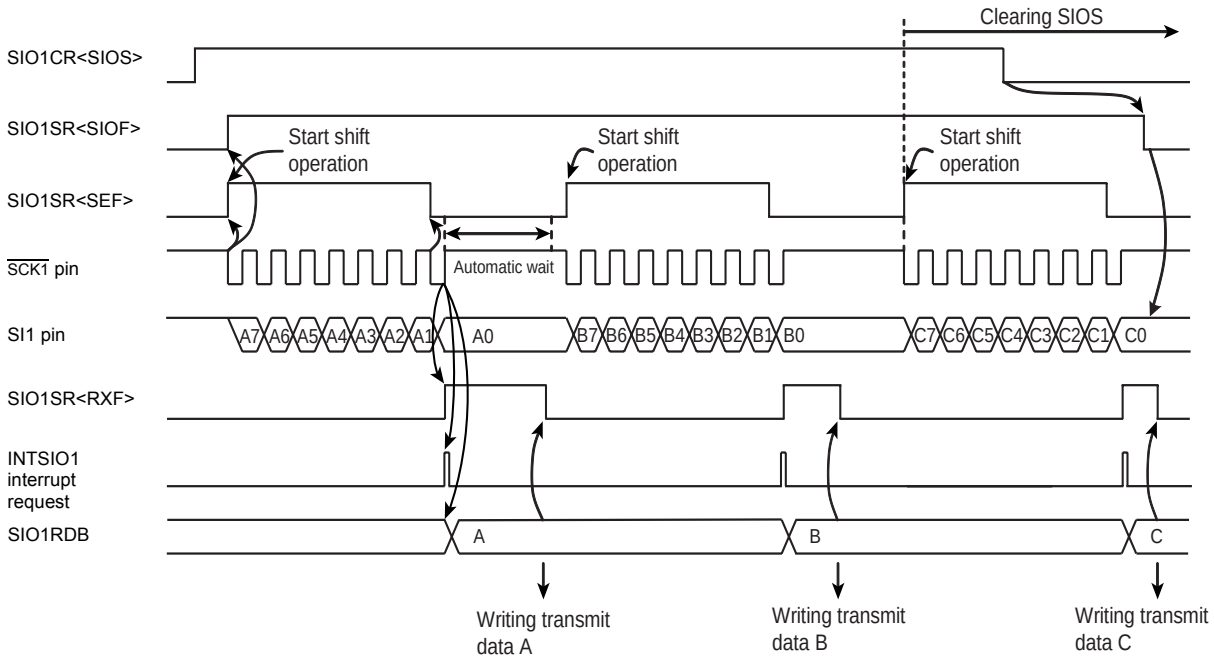



Figure 14-10 Example of Internal Clock and MSB Receive Mode

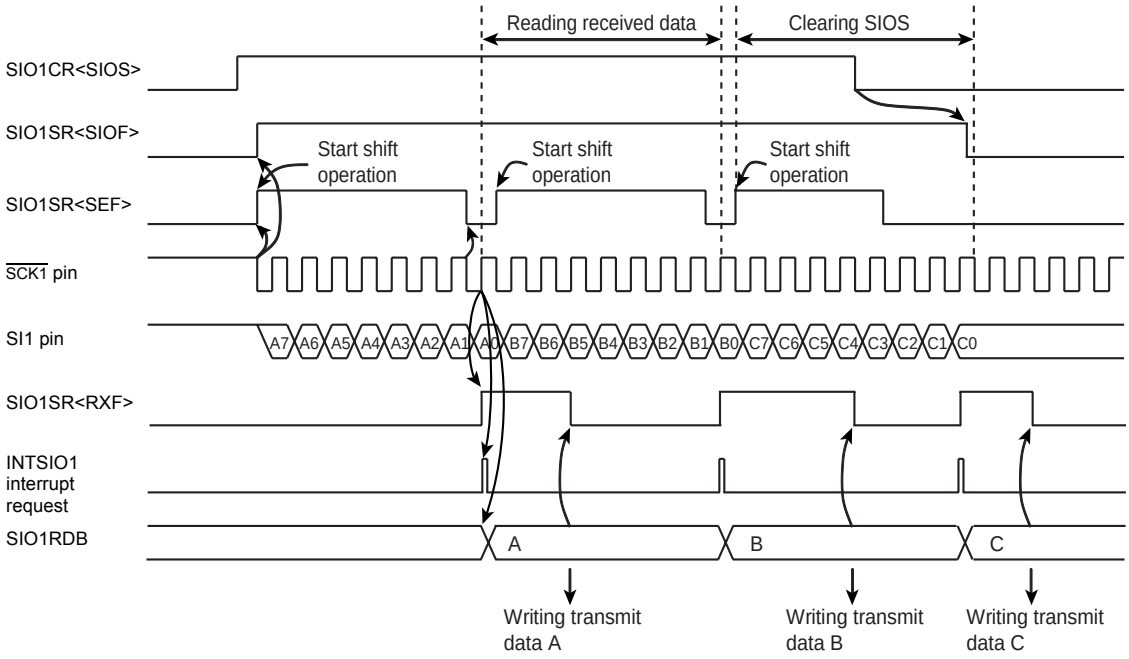



Figure 14-11 Example of External Clock and MSB Receive Mode

(4) Receive error processing

Receive errors occur on the following situation. To protect SIO1RDB and the shift register contents, the received data is ignored while the SIO1SR<RXERR> is "1".

- Shift operation is finished before reading out received data from SIO1RDB at SIO1SR<RXF> is "1" in an external clock operation.
  If receive error occurs, set the SIO1CR<SIOS> to "0" for reading the data that received immediately before error occurence. And read the data from SIO1RDB. Data in shift register (at errors occur) can be read by reading the SIO1RDB again.
  When SIO1SR<RXERR> is cleared to "0" after reading the received data, SIO1SR<RXF> is cleared to "0".
  After clearing SIO1CR<SIOS> to "0", when 8-bit serial clock is input to $\overline{\text{SCK1}}$ pin, receive operation is stopped. To restart the receive operation, confirm that SIO1SR<SIOF> is cleared to "0".
  If the receive error occurs, set the SIO1CR<SIOINH> to "1" for stopping the receive operation immediately. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

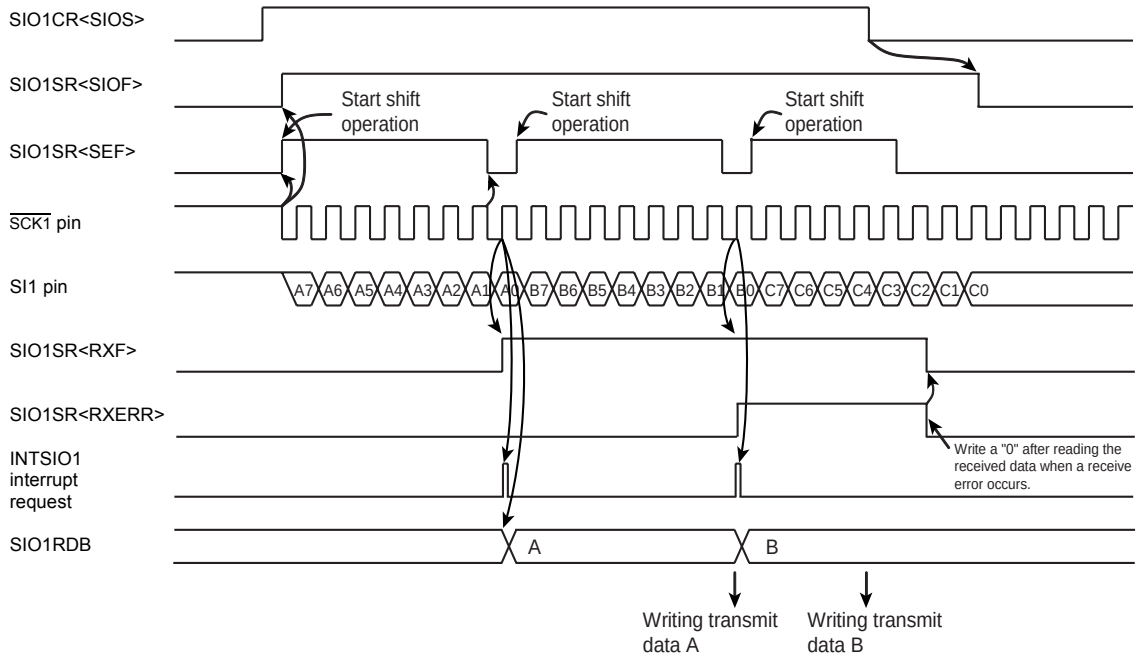



Figure 14-12 Example of Receive Error Processing

Note: If receive error is not corrected, an interrupt request does not generate after the error occurs.

### 14.3.3.3 Transmit/receive mode

The transmit/receive mode are selected by writing “10” to SIO1CR<SIOM>.

#### (1) Starting the transmit/receive operation

Transmit/receive mode is selected by writing “10B” to SIO1CR<SIOM>. Serial clock is selected by using SIO1CR<SCK>. Transfer direction is selected by using SIO1CR<SIODIR>.

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

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

The data is transferred sequentially starting from SO1 pin with the direction of the bit specified by SIO1CR<SIODIR>, synchronizing with the $\overline{\text{SCK1}}$ pin's falling edge. And receiving operation also starts with the direction of the bit specified by SIO1CR<SIODIR>, synchronizing with the $\overline{\text{SCK1}}$ pin's rising edge.

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

SIO1SR<TXF> is set to “1” at the rising edge of $\overline{\text{SCK1}}$ pin after the data written to the SIO1TDB is transferred to shift register. When 8-bit data has been received, the received data is transferred to SIO1RDB from shift register, then the INTSIO1 interrupt request occurs, synchronizing with setting SIO1SR<RXF> to “1”.

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

Note 2: In external clock operation, when the falling edge is input from $\overline{\text{SCK1}}$ pin after SIO1CR<SIOS> is set to "1", SIO1TDB is transferred to shift register immediately. When the rising edge is input from $\overline{\text{SCK1}}$ pin, receive operation also starts.

(2) During the transmit/receive operation

When data is written to SIO1TDB, SIO1SR<TXF> is cleared to “0” and when a data is read from SIO1RDB, SIO1SR<RXF> is cleared to “0”.

In internal clock operation, in case of the condition described below, the serial clock stops to “H” level by an automatic-wait function when all of the bit set in the data has been transmitted.

- Next transmit data is not written to SIO1TDB after reading a received data from SIO1RDB.
- Received data is not read from SIO1RDB after writing a next transmit data to SIO1TDB.
- Neither SIO1TDB nor SIO1RDB is accessed after transmission.

The automatic wait function is released by writing the next transmit data to SIO1TDB after reading the received data from SIO1RDB, or reading the received data from SIO1RDB after writing the next data to SIO1TDB.
Then, transmit/receive operation is restarted after maximum 1 cycle of serial clock.
In external clock operation, reading the received data from SIO1RDB and writing the next data to SIO1TDB must be finished before the shift operation of the next data begins.
If the transmit data is not written to SIO1TDB after SIO1SR<TXF> is set to “1”, transmit error occurs immediately after shift operation is started. When the transmit error occurred, SIO1SR<TXERR> is set to “1”.
If received data is not read out from SIO1RDB before next shift operation starts after setting SIO1SR<RXF> to “1”, receive error occurs immediately after shift operation is finished. When the receive error has occurred, SIO1SR<RXERR> is set to “1”.

(3) Stopping the transmit/receive operation

There are two ways for stopping the transmit/receive operation.

- The way of clearing SIO1CR<SIOS>.
  When SIO1CR<SIOS> is cleared to “0”, transmit/receive operation is stopped after all transfer of the data is finished. When transmit/receive operation is finished, SIO1SR<SIOF> is cleared to “0” and SO1 pin is kept in high level.
  In external clock operation, SIO1CR<SIOS> must be cleared to “0” before SIO1SR<SEF> is set to “1” by beginning next transfer.
- The way of setting SIO1CR<SIOINH>.
  Transmit/receive operation is stopped immediately after SIO1CR<SIOINH> is set to “1”. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

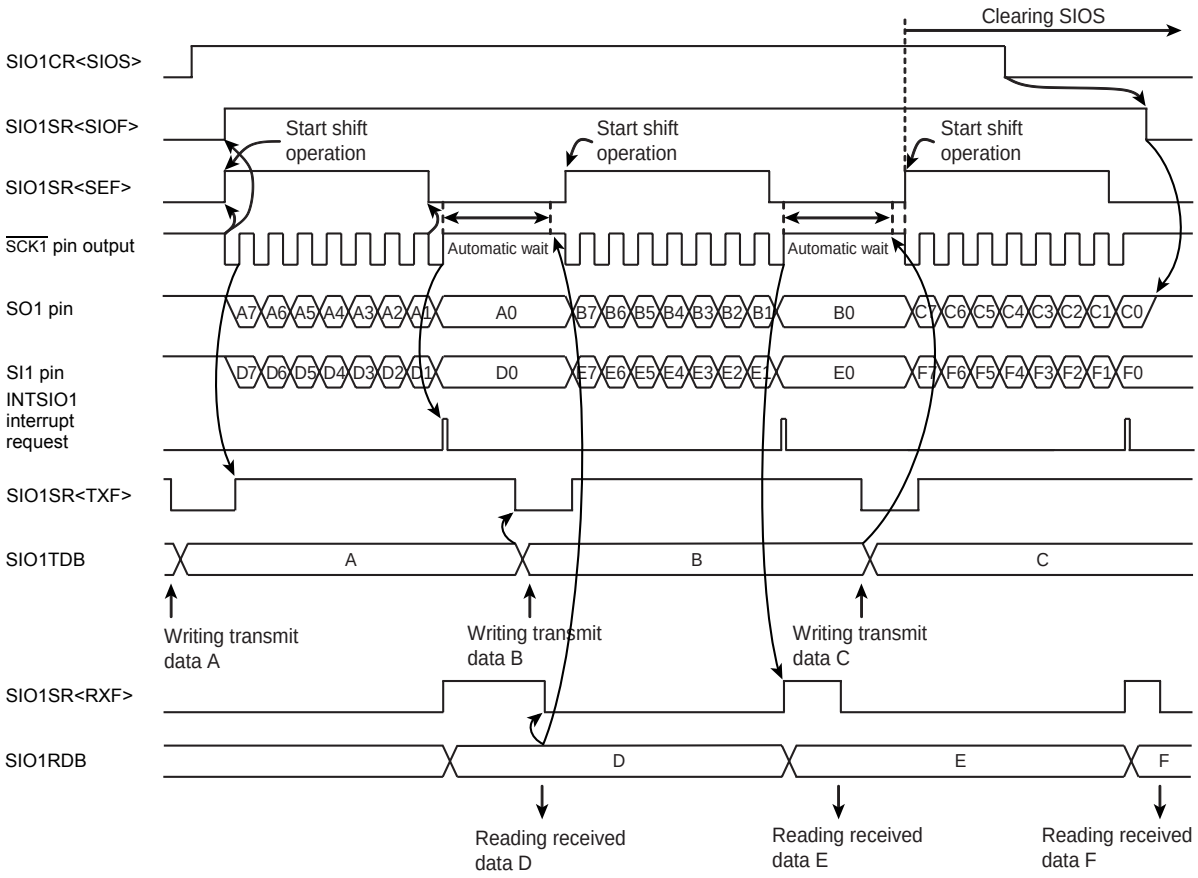



Figure 14-13 Example of Internal Clock and MSB Transmit/Receive Mode

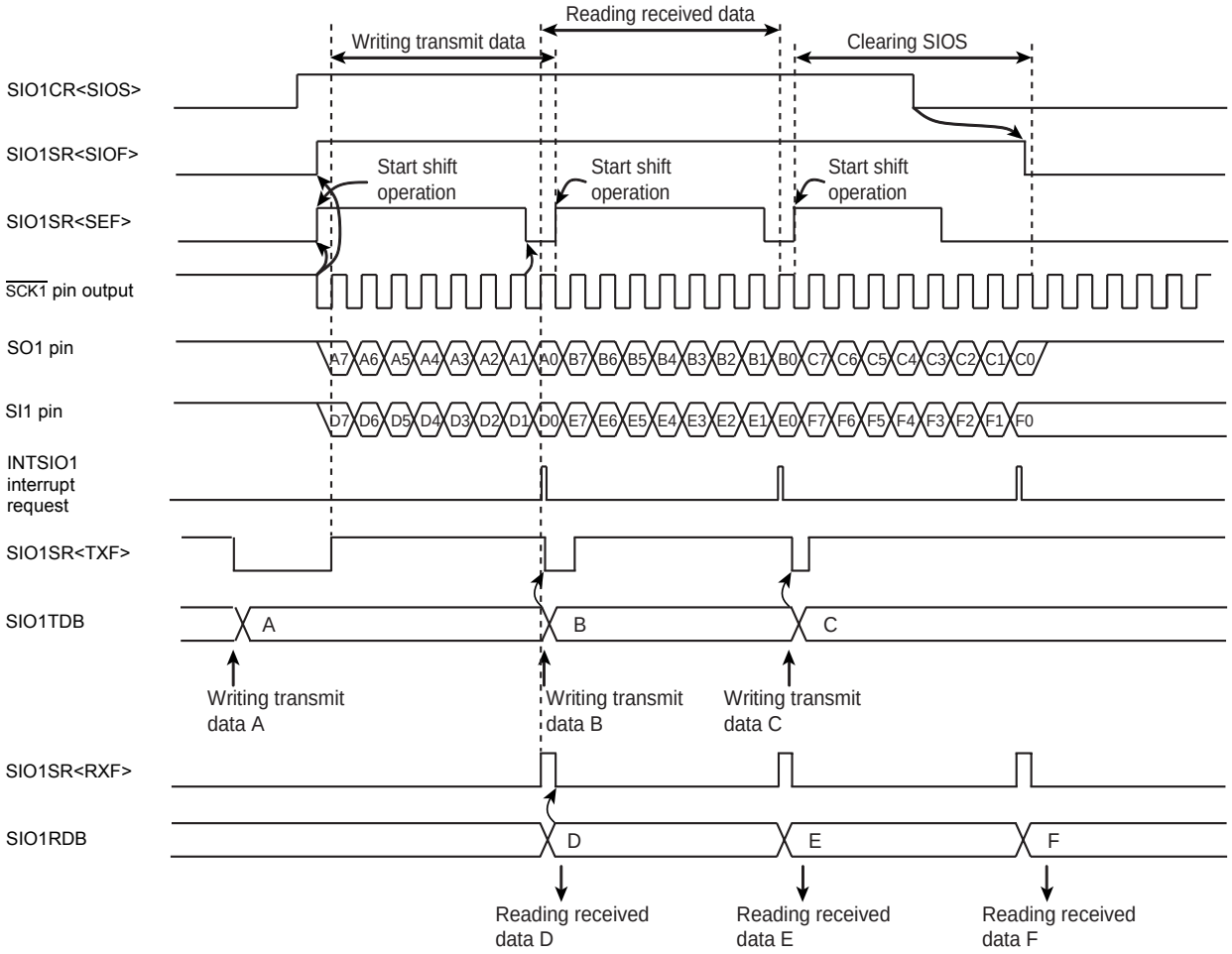



Figure 14-14 Example of External Clock and MSB Transmit/Receive Mode

### (4) Transmit/receive error processing

Transmit/receive errors occur on the following situation. Corrective action is different, which errors occur transmits or receives.

#### (a) Transmit errors

Transmit errors occur on the following situation.

- Shift operation starts before writing next transmit data to SIO1TDB in external clock operation.
  If transmit errors occur during transmit operation, SIO1SR<TXERR> is set to “1” immediately after starting shift operation. And INTSIO1 interrupt request is generated after all of the 8-bit data has been received.
  If shift operation starts before writing data to SIO1TDB after SIO1CR<SIOS> is set to “1”, SIO1SR<TXERR> is set immediately after starting shift operation. And INTSIO1 interrupt request is generated after all of the 8-bit data has been received.
  SO1 pin is kept in high level when SIO1SR<TXERR> is set to “1”. When transmit error occurs, transmit operation must be forcibly stop by writing SIO1CR<SIOINH> to “1” after the received data is read from SIO1RDB. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

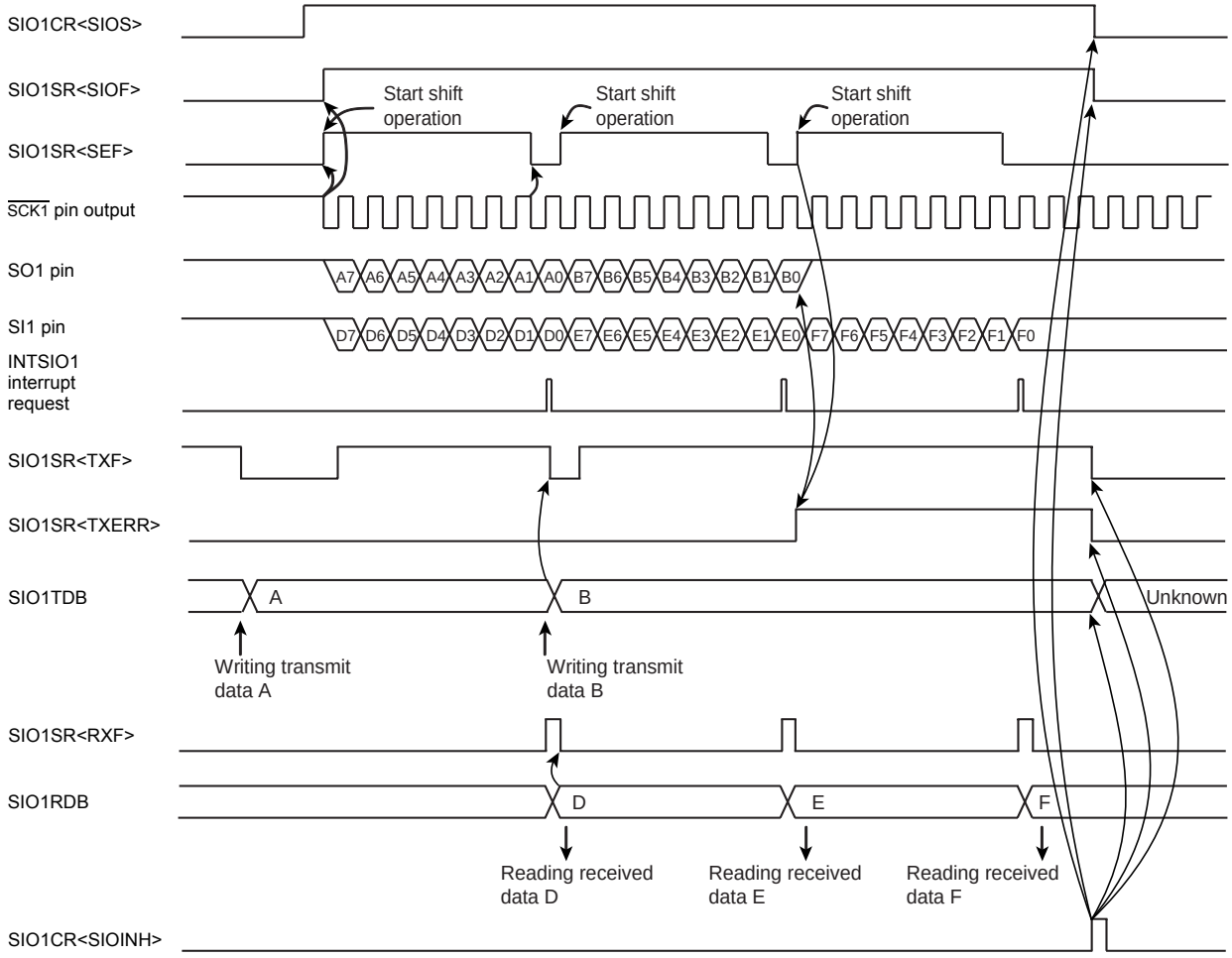



Figure 14-15 Example of Transmit/Receive (Transmit) Error Processing

(b) Receive errors

Receive errors occur on the following situation. To protect SIO1RDB and the shift register contents, the received data is ignored while the SIO1SR<RXERR> is "1".

- Shift operation is finished before reading out received data from SIO1RDB at SIO1SR<RXF> is "1" in an external clock operation.
  If receive error occurs, set the SIO1CR<SIOS> to "0" for reading the data that received immediately before error occurence. And read the data from SIO1RDB. Data in shift register (at errors occur) can be read by reading the SIO1RDB again.
  When SIO1SR<RXERR> is cleared to "0" after reading the received data, SIO1SR<RXF> is cleared to "0".
  After clearing SIO1CR<SIOS> to "0", when 8-bit serial clock is input to $\overline{\text{SCK1}}$ pin, receive operation is stopped. To restart the receive operation, confirm that SIO1SR<SIOF> is cleared to "0".
  If the received error occurs, set the SIO1CR<SIOINH> to "1" for stopping the receive operation immediately. In this case, SIO1CR<SIOS>, SIO1SR register, SIO1RDB register and SIO1TDB register are initialized.

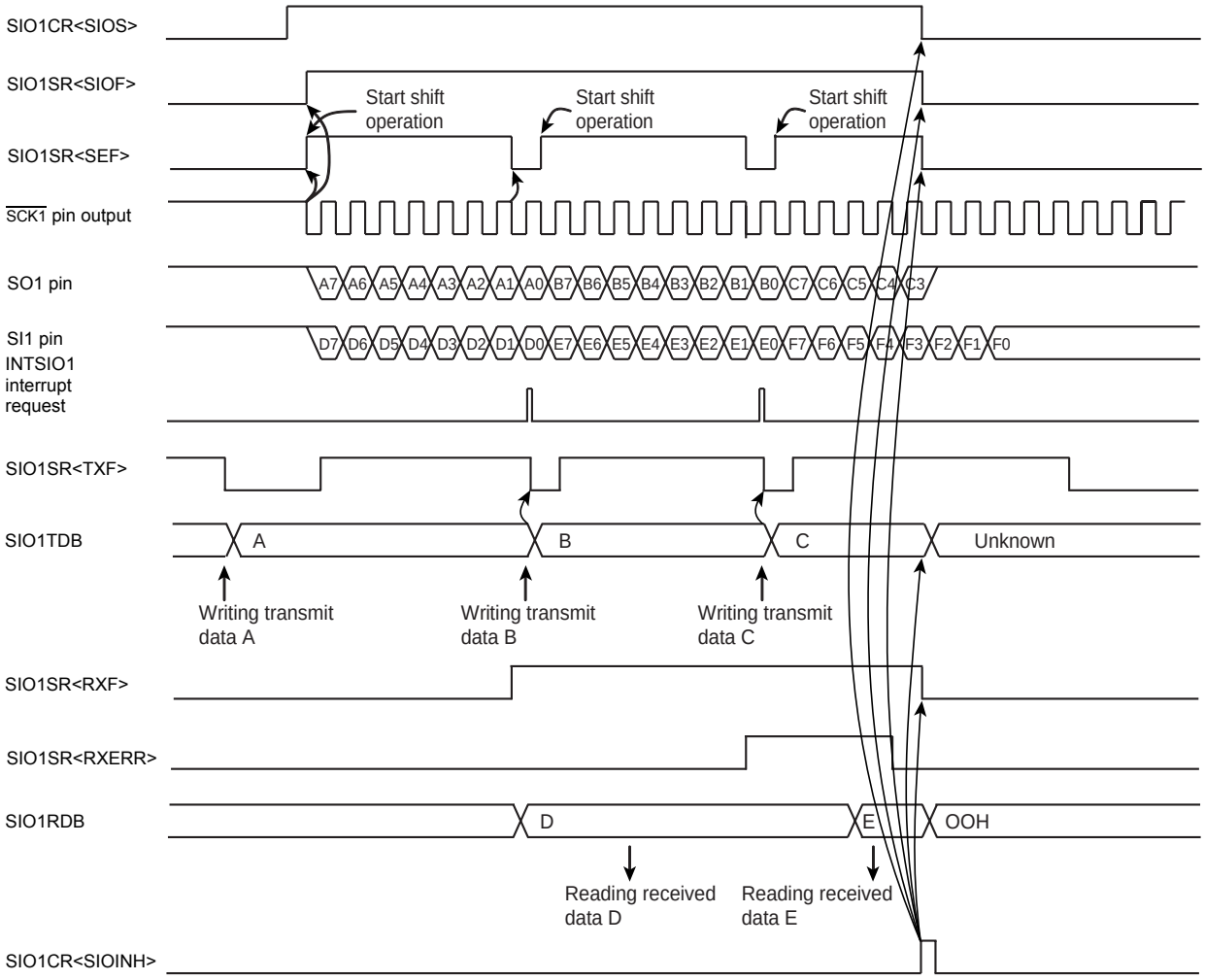



Figure 14-16 Example of Transmit/Receive (Receive) Error Processing

Note: If receive error is not corrected, an interrupt request does not generate after the error occurs.

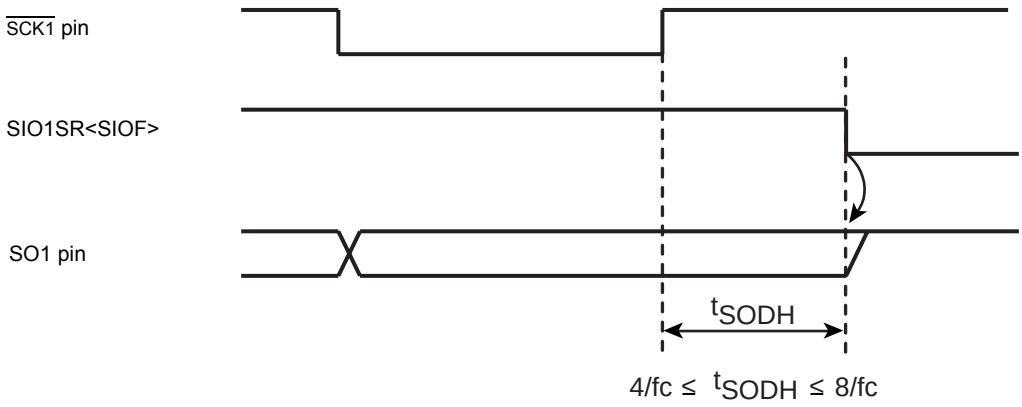


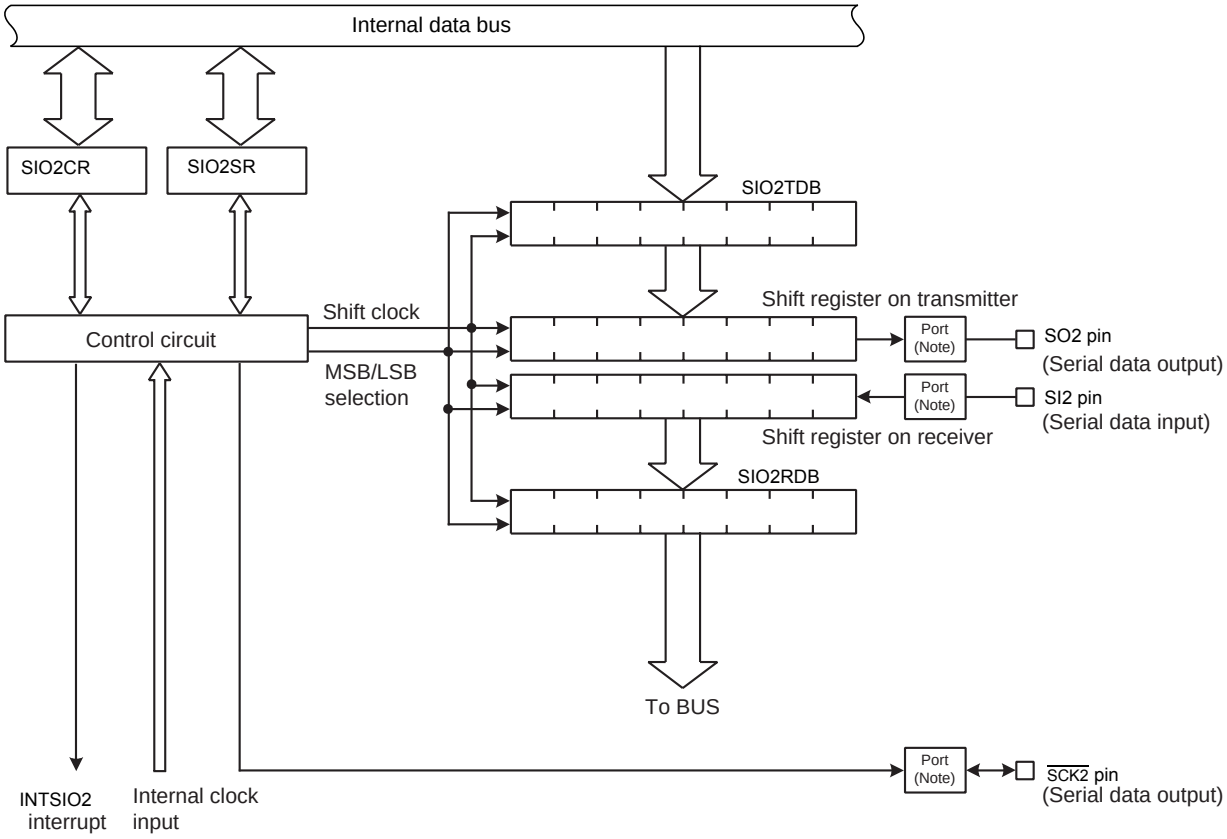

Figure 14-17 Hold Time of the End of Transmit/Receive Mode

# 15. Synchronous Serial Interface (SIO2)

The serial interfaces connect to an external device via SI2, SO2, and $\overline{SCK2}$ pins.

When these pins are used as serial interface, the output latches for each port should be set to "1".

## 15.1 Configuration




Note: Set the register of port correctly for the port assigned as serial interface pins.
For details, see the description of the input/output port control register.

Figure 15-1 Synchronous Serial Interface (SIO)

## 15.2 Control

The SIO is controlled using the serial interface control register (SIO2CR). The operating status of the serial interface can be inspected by reading the status register (SIO2CR).

### Serial Interface Control Register

SIO2CR (0031H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SIOS | SIOINH | SIOM | | SIODIR | SCK | | |

(Initial value: 0000 0000)

| | | | | | |
|---|---|---|---|---|---|
| SIOS | Specify start/stop of transfer | 0: Stop<br>1: Start | | | R/W |
| SIOINH | Forcibly stops transfer (Note 1) | 0: –<br>1: Forcibly stop (Automatically cleared to "0" after stopping) | | | |
| SIOM | Selects transfer mode | 00: Transmit mode<br>01: Receive mode<br>10: Transmit/receive mode<br>11: Reserved | | | |
| SIODIR | Selects direction of transfer | 0: MSB (Transfer beginning with bit7)<br>1: LSB (Transfer beginning with bit0) | | | |
| SCK | Selects serial clock | | NORMAL1/2 or IDLE1/2 modes | | SLOW/SLEEP mode |
| | | | TBTCR <DV7CK> = "0" | TBTCR <DV7CK> = "1" | |
| | | 000 | $fc/2^{12}$ | $fs/2^4$ | $fs/2^4$ |
| | | 001 | $fc/2^8$ | $fc/2^8$ | Reserved |
| | | 010 | $fc/2^7$ | $fc/2^7$ | Reserved |
| | | 011 | $fc/2^6$ | $fc/2^6$ | Reserved |
| | | 100 | $fc/2^5$ | $fc/2^5$ | Reserved |
| | | 101 | $fc/2^4$ | $fc/2^4$ | Reserved |
| | | 110 | $fc/2^3$ | $fc/2^3$ | Reserved |
| | | 111 | External clock (Input from $\overline{SCK2}$ pin) | | |

Note 1: When SIO2CR<SIOINH> is set to “1”, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

Note 2: Transfer mode, direction of transfer and serial clock must be select during the transfer is stopping (when SIO2SR<SIOF> "0").

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

## Serial Interface Status Register

SIO2SR (0032H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SIOF | SEF | TXF | RXF | TXERR | RXERR | | |

(Initial value: 0010 00**)

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

Note 1: The operation error flag (TXERR and RXERR) are not automatically cleared by stopping transfer with SIO2CR<SIOS> "0". Therefore, set these bits to "0" for clearing these error flag. Or set SIO2CR<SIOINH> to "1".

Note 2: *: Don't care

## Receive buffer register

SIO2RDB (002BH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Read only

(Initial value: 0000 0000)

## Transmit buffer register

SIO2TDB (002BH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Write only

(Initial value: **** ****)

Note 1: SIO2TDB is write only register. A bit manipulation should not be performed on the transmit buffer register using a read-modify-write instruction.

Note 2: The SIO2TDB should be written after checking SIO2SR<TXF> "1". When SIO2SR<TXF> is "0", the writing data can't be transferred to SIO2TDB even if write instruction is executed to SIO2TDB .

Note 3: *: Don't care

# 15.3 Function

## 15.3.1 Serial clock

### 15.3.1.1 Clock source

The serial clock can be selected by using SIO2CR<SCK>. When the serial clock is changed, the writing instruction to SIO2CR<SCK> should be executed while the transfer is stopped (when SIO2SR<SIOF> "0")

(1) Internal clock

Setting the SIO2CR<SCK> to other than "111B" outputs the clock (shown in " Table 15-1 Serial Clock Rate (fc = 16 MHz, fs = 32.768kHz) ") as serial clock outputs from $\overline{SCK2}$ pin. At the before beginning or finishing of a transfer, $\overline{SCK2}$ pin is kept in high level.

When writing (in the transmit mode) or reading (in the receive mode) data can not follow the serial clock rate, an automatic-wait function is executed to stop the serial clock automatically and hold the next shift operation until reading or writing is completed (shown in " Figure 15-2 Automatic-wait Function (Example of transmit mode) "). The maximum time from releasing the automatic-wait function by reading or writing a data is 1 cycle of the selected serial clock until the serial clock comes out from $\overline{SCK2}$ pin.

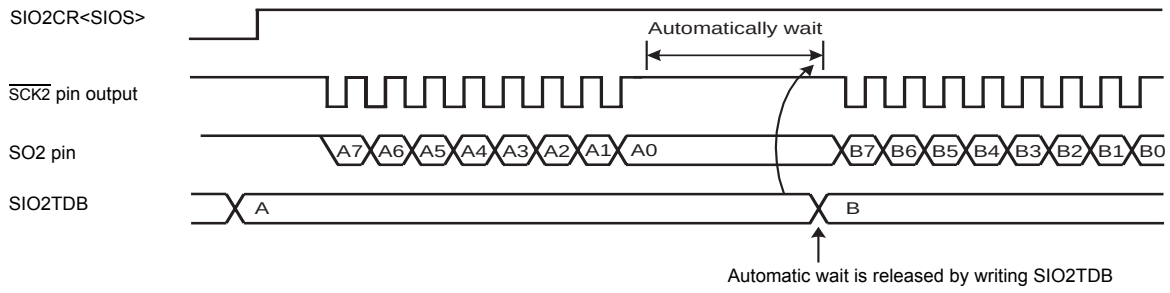



Figure 15-2 Automatic-wait Function (Example of transmit mode)

Table 15-1 Serial Clock Rate (fc = 16 MHz, fs = 32.768kHz)

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

(2) External clock

When an external clock is selected by setting SIO2CR<SCK> to "111B", the clock via the $\overline{SCK2}$ pin from an external source is used as the serial clock.

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

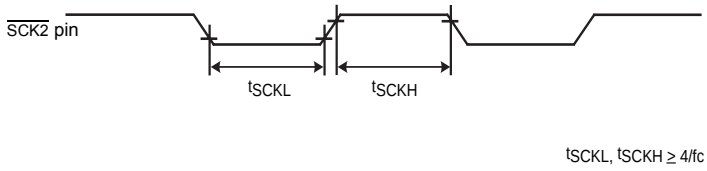



Figure 15-3 External Clock

### 15.3.1.2 Shift edge

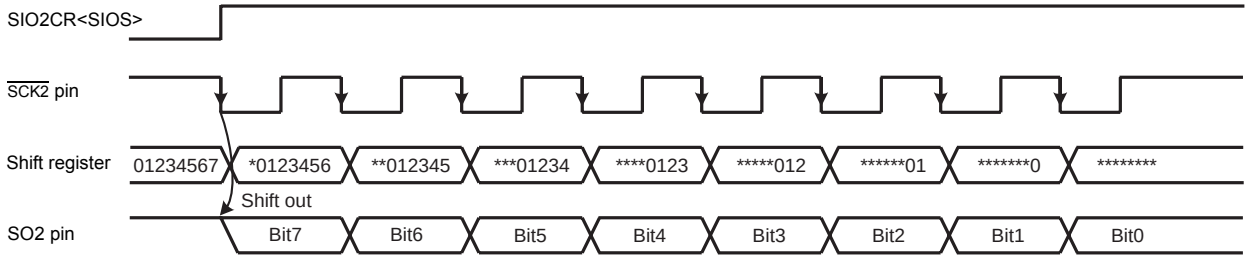
The leading edge is used to transmit data, and the trailing edge is used to receive data.

(1) Leading edge shift

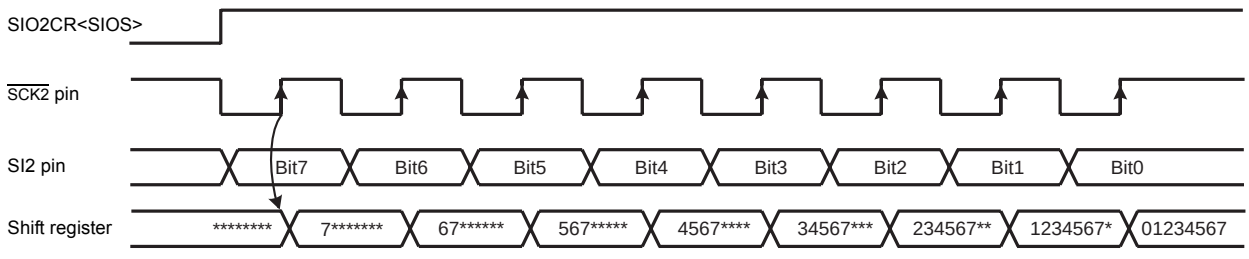
Data is shifted on the leading edge of the serial clock (falling edge of the $\overline{SCK2}$ pin input/output).

(2) Trailing edge shift

Data is shifted on the trailing edge of the serial clock (rising edge of the $\overline{SCK2}$ pin input/output).




(a) Leading edge shift (Example of MSB transfer)




(b) Trailing edge shift (Example of MSB transfer)

Figure 15-4 Shift Edge

## 15.3.2 Transfer bit direction

Transfer data direction can be selected by using SIO2CR<SIODIR>. The transfer data direction can't be set individually for transmit and receive operations.

When the data direction is changed, the writing instruction to SIO2CR<SIODIR> should be executed while the transfer is stopped (when SIO2CR<SIOF>= “0”)

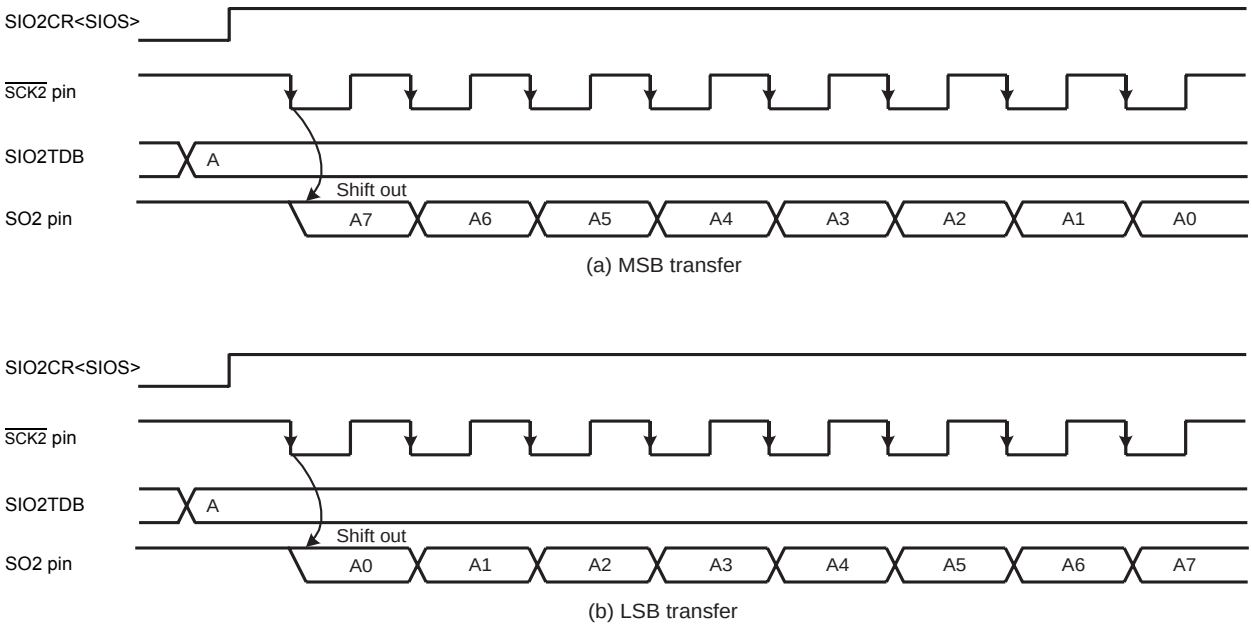



Figure 15-5 Transfer Bit Direction (Example of transmit mode)

### 15.3.2.1 Transmit mode

#### (1) MSB transmit mode

MSB transmit mode is selected by setting SIO2CR<SIODIR> to “0”, in which case the data is transferred sequentially beginning with the most significant bit (Bit7).

#### (2) LSB transmit mode

LSB transmit mode is selected by setting SIO2CR<SIODIR> to “1”, in which case the data is transferred sequentially beginning with the least significant bit (Bit0).

### 15.3.2.2 Receive mode

#### (1) MSB receive mode

MSB receive mode is selected by setting SIO2CR<SIODIR> to “0”, in which case the data is received sequentially beginning with the most significant bit (Bit7).

(2) LSB receive mode

LSB receive mode is selected by setting SIO2CR<SIODIR> to “1”, in which case the data is received sequentially beginning with the least significant bit (Bit0).

#### 15.3.2.3 Transmit/receive mode

(1) MSB transmit/receive mode

MSB transmit/receive mode are selected by setting SIO2CR<SIODIR> to “0” in which case the data is transferred sequentially beginning with the most significant bit (Bit7) and the data is received sequentially beginning with the most significant (Bit7).

(2) LSB transmit/receive mode

LSB transmit/receive mode are selected by setting SIO2CR<SIODIR> to “1”, in which case the data is transferred sequentially beginning with the least significant bit (Bit0) and the data is received sequentially beginning with the least significant (Bit0).

### 15.3.3 Transfer modes

Transmit, receive and transmit/receive mode are selected by using SIO2CR<SIOM>.

#### 15.3.3.1 Transmit mode

Transmit mode is selected by writing “00B” to SIO2CR<SIOM>.

(1) Starting the transmit operation

Transmit mode is selected by setting “00B” to SIO2CR<SIOM>. Serial clock is selected by using SIO2CR<SCK>. Transfer direction is selected by using SIO2CR<SIODIR>.

When a transmit data is written to the transmit buffer register (SIO2TDB), SIO2SR<TXF> is cleared to “0”.

After SIO2CR<SIOS> is set to “1”, SIO2SR<SIOF> is set synchronously to “1” the falling edge of $\overline{\text{SCK2}}$ pin.

The data is transferred sequentially starting from SO2 pin with the direction of the bit specified by SIO2CR<SIODIR>, synchronizing with the $\overline{\text{SCK2}}$ pin's falling edge.

SIO2SR<SEF> is kept in high level, between the first clock falling edge of $\overline{\text{SCK2}}$ pin and eighth clock falling edge.

SIO2SR<TXF> is set to “1” at the rising edge of  pin after the data written to the SIO2TDB is transferred to shift register, then the INTSIO2 interrupt request is generated, synchronizing with the next falling edge on $\overline{\text{SCK2}}$ pin.

Note 1: In internal clock operation, when SIO2CR<SIOS> is set to "1", transfer mode does not start without writing a transmit data to the transmit buffer register (SIO2TDB).

Note 2: In internal clock operation, when the SIO2CR<SIOS> is set to "1", SIO2TDB is transferred to shift register after maximum 1-cycle of serial clock frequency, then a serial clock is output from $\overline{\text{SCK2}}$ pin.

Note 3: In external clock operation, when the falling edge is input from $\overline{\text{SCK2}}$ pin after SIO2CR<SIOS> is set to "1", SIO2TDB is transferred to shift register immediately.

(2) During the transmit operation

When data is written to SIO2TDB, SIO2SR<TXF> is cleared to “0”.

In internal clock operation, in case a next transmit data is not written to SIO2TDB, the serial clock stops to “H” level by an automatic-wait function when all of the bit set in the SIO2TDB has been transmitted. Automatic-wait function is released by writing a transmit data to SIO2TDB. Then, transmit operation is restarted after maximum 1-cycle of serial clock.

When the next data is written to the SIO2TDB before termination of previous 8-bit data with SIO2SR<TXF> “1”, the next data is continuously transferred after transmission of previous data.

In external clock operation, after SIO2SR<TXF> is set to “1”, the transmit data must be written to SIO2TDB before the shift operation of the next data begins.

If the transmit data is not written to SIO2TDB, transmit error occurs immediately after shift operation is started. Then, INTSIO2 interrupt request is generated after SIO2SR<TXERR> is set to “1”.

(3) Stopping the transmit operation

There are two ways for stopping transmits operation.

- The way of clearing SIO2CR<SIOS>.
  When SIO2CR<SIOS> is cleared to “0”, transmit operation is stopped after all transfer of the data is finished. When transmit operation is finished, SIO2SR<SIOF> is cleared to “0” and SO2 pin is kept in high level.
  In external clock operation, SIO2CR<SIOS> must be cleared to “0” before SIO2SR<SEF> is set to “1” by beginning next transfer.
- The way of setting SIO2CR<SIOINH>.
  Transmit operation is stopped immediately after SIO2CR<SIOINH> is set to “1”. In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

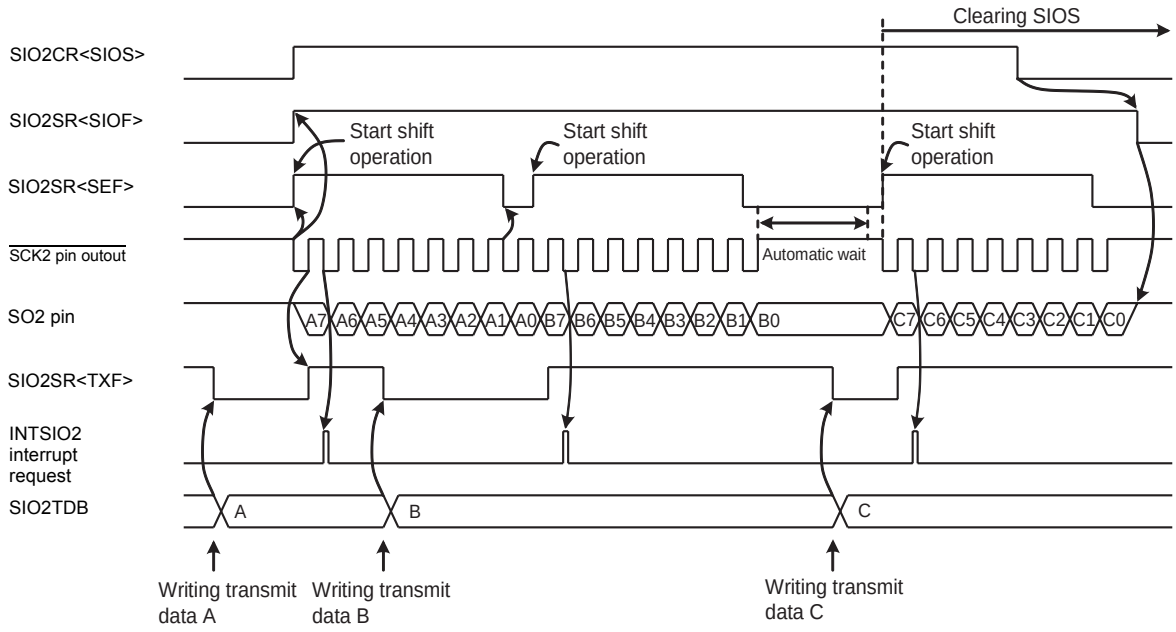



Figure 15-6 Example of Internal Clock and MSB Transmit Mode

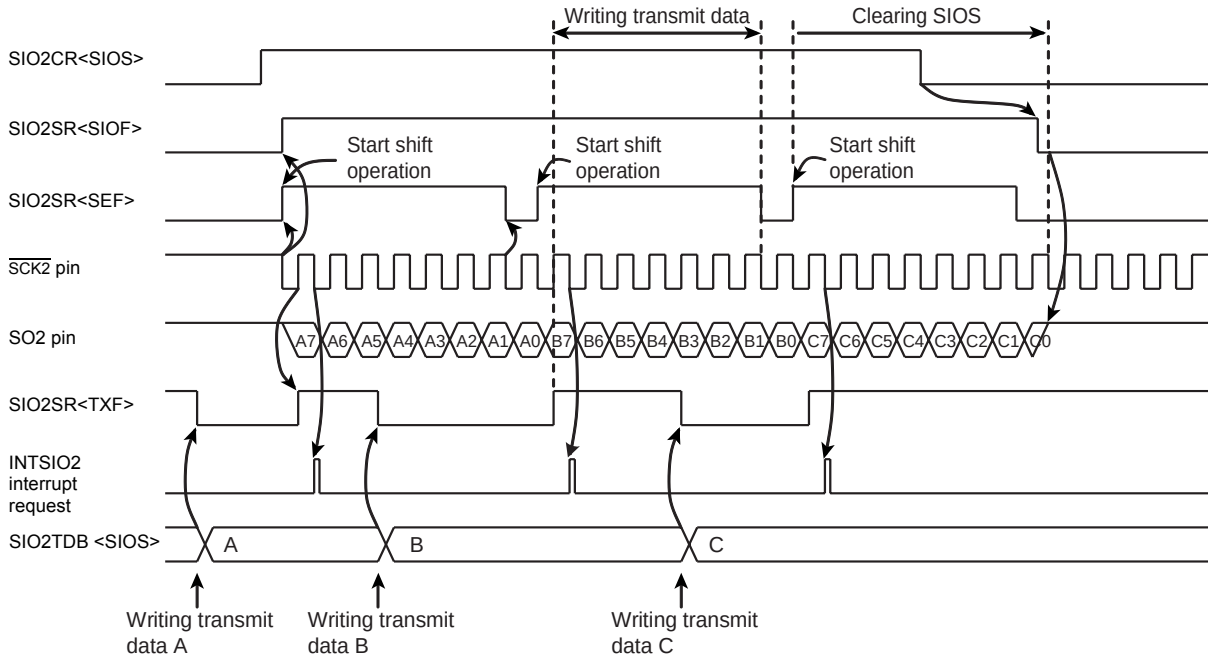



Figure 15-7 Exaple of External Clock and MSB Transmit Mode

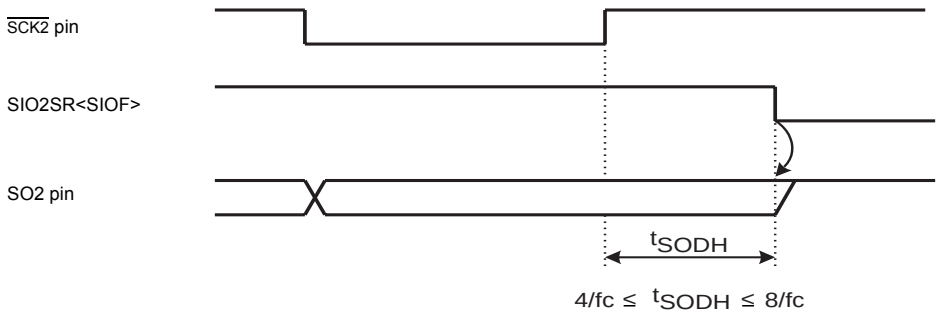



Figure 15-8 Hold Time of the End of Transmit Mode

(4) Transmit error processing

Transmit errors occur on the following situation.

- Shift operation starts before writing next transmit data to SIO2TDB in external clock operation.
  If transmit errors occur during transmit operation, SIO2SR<TXERR> is set to “1” immediately after starting shift operation. Synchronizing with the next serial clock falling edge, INTSIO2 interrupt request is generated.
  If shift operation starts before writing data to SIO2TDB after SIO2CR<SIOS> is set to “1”, SIO2SR<TXERR> is set to “1” immediately after shift operation is started and then INTSIO2 interrupt request is generated.
  SIO2 pin is kept in high level when SIO2SR<TXERR> is set to “1”. When transmit error occurs, transmit operation must be forcibly stop by writing SIO2CR<SIOINH> to “1”. In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

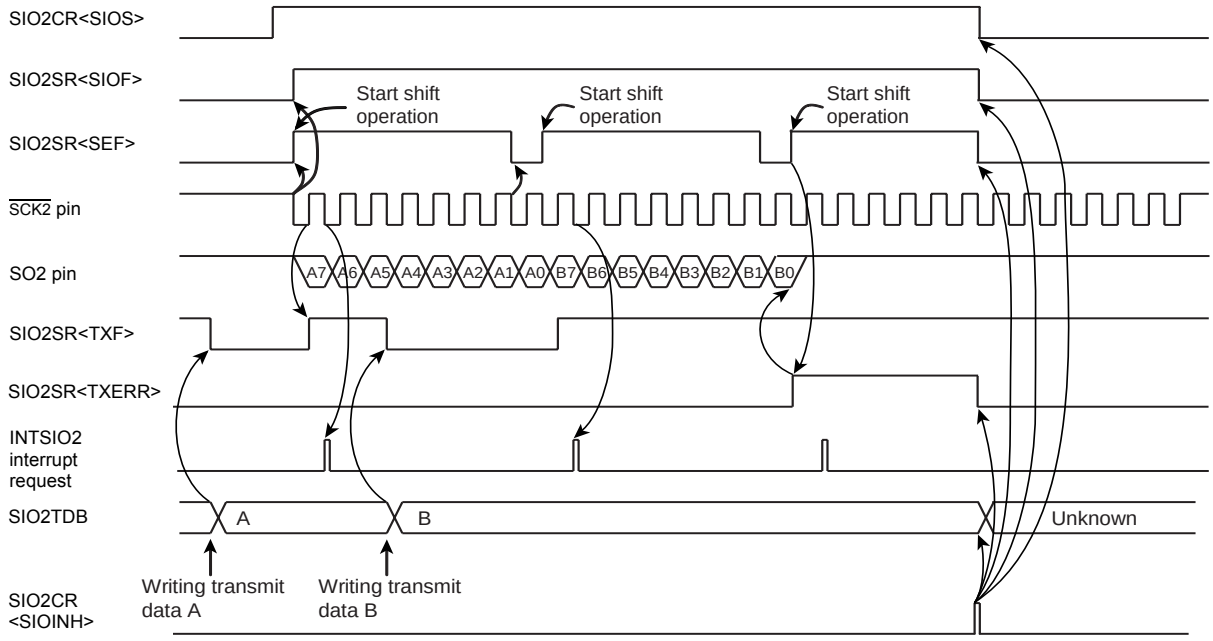



Figure 15-9 Example of Transmit Error Processingme

### 15.3.3.2 Receive mode

The receive mode is selected by writing “01B” to SIO2CR<SIOM>.

#### (1) Starting the receive operation

Receive mode is selected by setting “01” to SIO2CR<SIOM>. Serial clock is selected by using SIO2CR<SCK>. Transfer direction is selected by using SIO2CR<SIODIR>.

After SIO2CR<SIOS> is set to “1”, SIO2SR<SIOF> is set synchronously to “1” the falling edge of $\overline{\text{SCK2}}$ pin.

Synchronizing with the $\overline{\text{SCK2}}$ pin's rising edge, the data is received sequentially from SI2 pin with the direction of the bit specified by SBI2DIR<SIODIR>.

SIO2SR<SEF> is kept in high level, between the first clock falling edge of $\overline{\text{SCK2}}$ pin and eighth clock falling edge.

When 8-bit data is received, the data is transferred to SIO2RDB from shift register. INTSIO2 interrupt request is generated and SIO2SR<RXF> is set to “1”

Note: In internal clock operation, when the SIO2CR<SIOS> is set to "1", the serial clock is generated from $\overline{\text{SCK2}}$ pin after maximum 1-cycle of serial clock frequency.

#### (2) During the receive operation

The SIO2SR<RXF> is cleared to “0” by reading a data from SIO2RDB.

In the internal clock operation, the serial clock stops to “H” level by an automatic-wait function when the all of the 8-bit data has been received. Automatic-wait function is released by reading a received data from SIO2RDB. Then, receive operation is restarted after maximum 1-cycle of serial clock.

In external clock operation, after SIO2SR<RXF> is set to “1”, the received data must be read from SIO2RDB, before the next data shift-in operation is finished.

If received data is not read out from SIO2RDB receive error occurs immediately after shift operation is finished. Then INTSIO2 interrupt request is generated after SIO2SR<RXERR> is set to "1".

(3) Stopping the receive operation

There are two ways for stopping the receive operation.

- The way of clearing SIO2CR<SIOS>.
  When SIO2CR<SIOS> is cleared to "0", receive operation is stopped after all of the data is finished to receive. When receive operation is finished, SIO2SR<SIOF> is cleared to "0". In external clock operation, SIO2CR<SIOS> must be cleared to "0" before SIO2SR<SEF> is set to "1" by starting the next shift operation.
- The way of setting SIO2CR<SIOINH>.
  Receive operation is stopped immediately after SIO2CR<SIOINH> is set to "1". In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

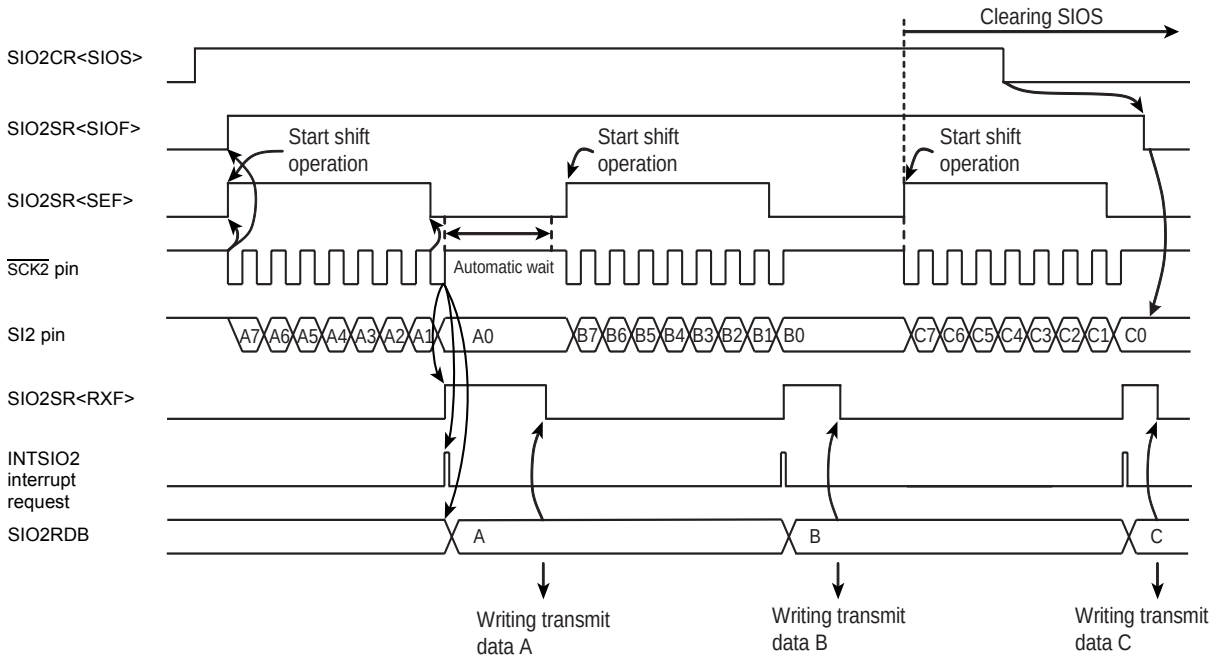



Figure 15-10 Example of Internal Clock and MSB Receive Mode

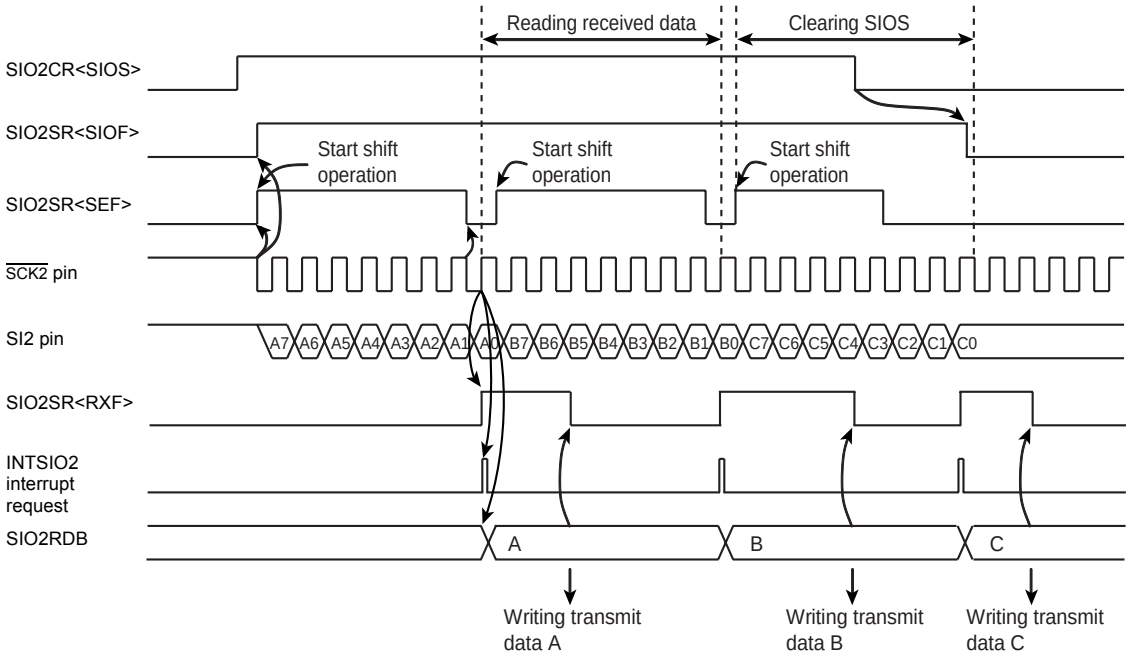



Figure 15-11 Example of External Clock and MSB Receive Mode

(4) Receive error processing

Receive errors occur on the following situation. To protect SIO2RDB and the shift register contents, the received data is ignored while the SIO2SR<RXERR> is "1".

- Shift operation is finished before reading out received data from SIO2RDB at SIO2SR<RXF> is "1" in an external clock operation.
  If receive error occurs, set the SIO2CR<SIOS> to "0" for reading the data that received immediately before error occurence. And read the data from SIO2RDB. Data in shift register (at errors occur) can be read by reading the SIO2RDB again.
  When SIO2SR<RXERR> is cleared to "0" after reading the received data, SIO2SR<RXF> is cleared to "0".
  After clearing SIO2CR<SIOS> to "0", when 8-bit serial clock is input to $\overline{\text{SCK2}}$ pin, receive operation is stopped. To restart the receive operation, confirm that SIO2SR<SIOF> is cleared to "0".
  If the receive error occurs, set the SIO2CR<SIOINH> to "1" for stopping the receive operation immediately. In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

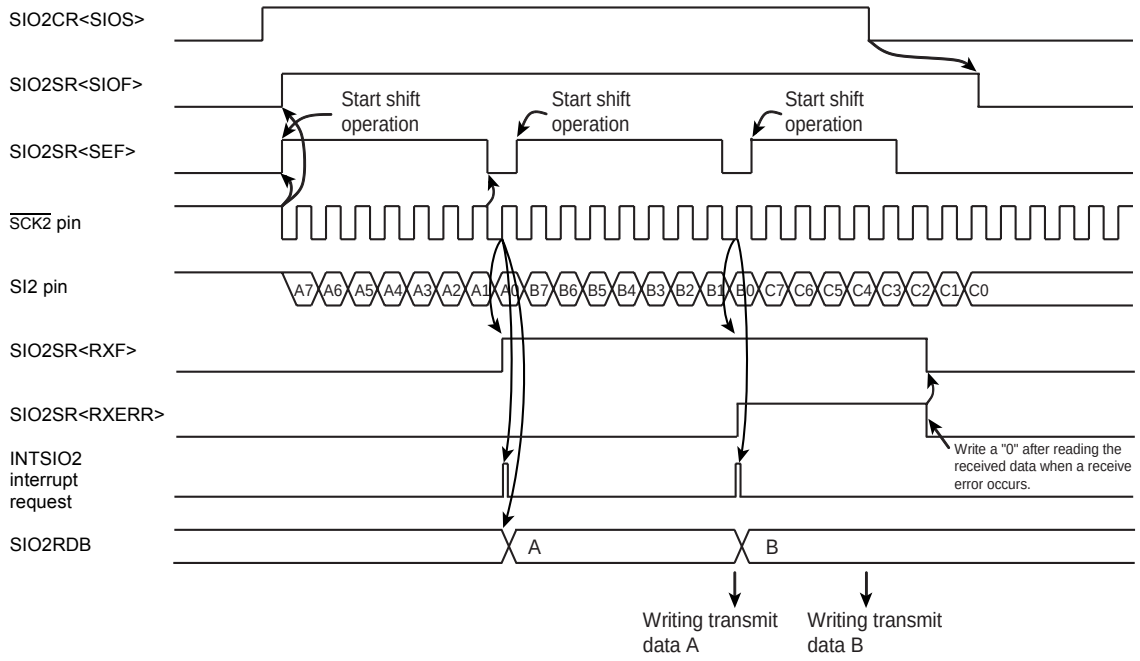



Figure 15-12 Example of Receive Error Processing

Note: If receive error is not corrected, an interrupt request does not generate after the error occurs.

### 15.3.3.3 Transmit/receive mode

The transmit/receive mode are selected by writing “10” to SIO2CR<SIOM>.

#### (1) Starting the transmit/receive operation

Transmit/receive mode is selected by writing “10B” to SIO2CR<SIOM>. Serial clock is selected by using SIO2CR<SCK>. Transfer direction is selected by using SIO2CR<SIODIR>.

When a transmit data is written to the transmit buffer register (SIO2TDB), SIO2SR<TXF> is cleared to “0”.

After SIO2CR<SIOS> is set to “1”, SIO2SR<SIOF> is set synchronously to the falling edge of $\overline{\text{SCK2}}$ pin.

The data is transferred sequentially starting from SO2 pin with the direction of the bit specified by SIO2CR<SIODIR>, synchronizing with the $\overline{\text{SCK2}}$ pin's falling edge. And receiving operation also starts with the direction of the bit specified by SIO2CR<SIODIR>, synchronizing with the $\overline{\text{SCK2}}$ pin's rising edge.

SIO2SR<SEF> is kept in high level between the first clock falling edge of $\overline{\text{SCK2}}$ pin and eighth clock falling edge.

SIO2SR<TXF> is set to “1” at the rising edge of $\overline{\text{SCK2}}$ pin after the data written to the SIO2TDB is transferred to shift register. When 8-bit data has been received, the received data is transferred to SIO2RDB from shift register, then the INTSIO2 interrupt request occurs, synchronizing with setting SIO2SR<RXF> to “1”.

Note 1: In internal clock operation, when the SIO2CR<SIOS> is set to "1", SIO2TDB is transferred to shift register after maximum 1-cycle of serial clock frequency, then a serial clock is output from $\overline{\text{SCK2}}$ pin.

Note 2: In external clock operation, when the falling edge is input from $\overline{\text{SCK2}}$ pin after SIO2CR<SIOS> is set to "1", SIO2TDB is transferred to shift register immediately. When the rising edge is input from $\overline{\text{SCK2}}$ pin, receive operation also starts.

(2) During the transmit/receive operation

When data is written to SIO2TDB, SIO2SR<TXF> is cleared to “0” and when a data is read from SIO2RDB, SIO2SR<RXF> is cleared to “0”.

In internal clock operation, in case of the condition described below, the serial clock stops to “H” level by an automatic-wait function when all of the bit set in the data has been transmitted.

- Next transmit data is not written to SIO2TDB after reading a received data from SIO2RDB.
- Received data is not read from SIO2RDB after writing a next transmit data to SIO2TDB.
- Neither SIO2TDB nor SIO2RDB is accessed after transmission.

The automatic wait function is released by writing the next transmit data to SIO2TDB after reading the received data from SIO2RDB, or reading the received data from SIO2RDB after writing the next data to SIO2TDB.
Then, transmit/receive operation is restarted after maximum 1 cycle of serial clock.
In external clock operation, reading the received data from SIO2RDB and writing the next data to SIO2TDB must be finished before the shift operation of the next data begins.
If the transmit data is not written to SIO2TDB after SIO2SR<TXF> is set to “1”, transmit error occurs immediately after shift operation is started. When the transmit error occurred, SIO2SR<TXERR> is set to “1”.
If received data is not read out from SIO2RDB before next shift operation starts after setting SIO2SR<RXF> to “1”, receive error occurs immediately after shift operation is finished. When the receive error has occurred, SIO2SR<RXERR> is set to “1”.

(3) Stopping the transmit/receive operation

There are two ways for stopping the transmit/receive operation.

- The way of clearing SIO2CR<SIOS>.
  When SIO2CR<SIOS> is cleared to “0”, transmit/receive operation is stopped after all transfer of the data is finished. When transmit/receive operation is finished, SIO2SR<SIOF> is cleared to “0” and SO2 pin is kept in high level.
  In external clock operation, SIO2CR<SIOS> must be cleared to “0” before SIO2SR<SEF> is set to “1” by beginning next transfer.
- The way of setting SIO2CR<SIOINH>.
  Transmit/receive operation is stopped immediately after SIO2CR<SIOINH> is set to “1”. In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

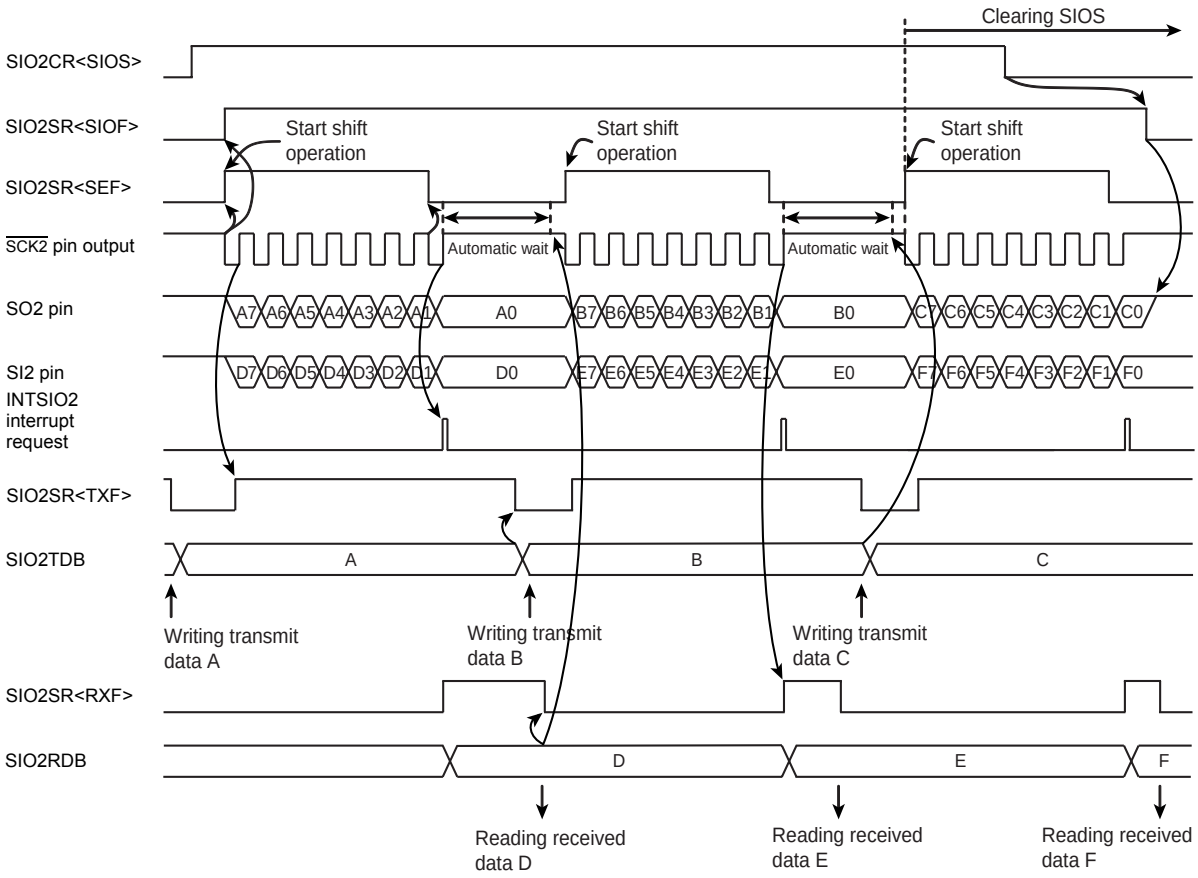



Figure 15-13 Example of Internal Clock and MSB Transmit/Receive Mode

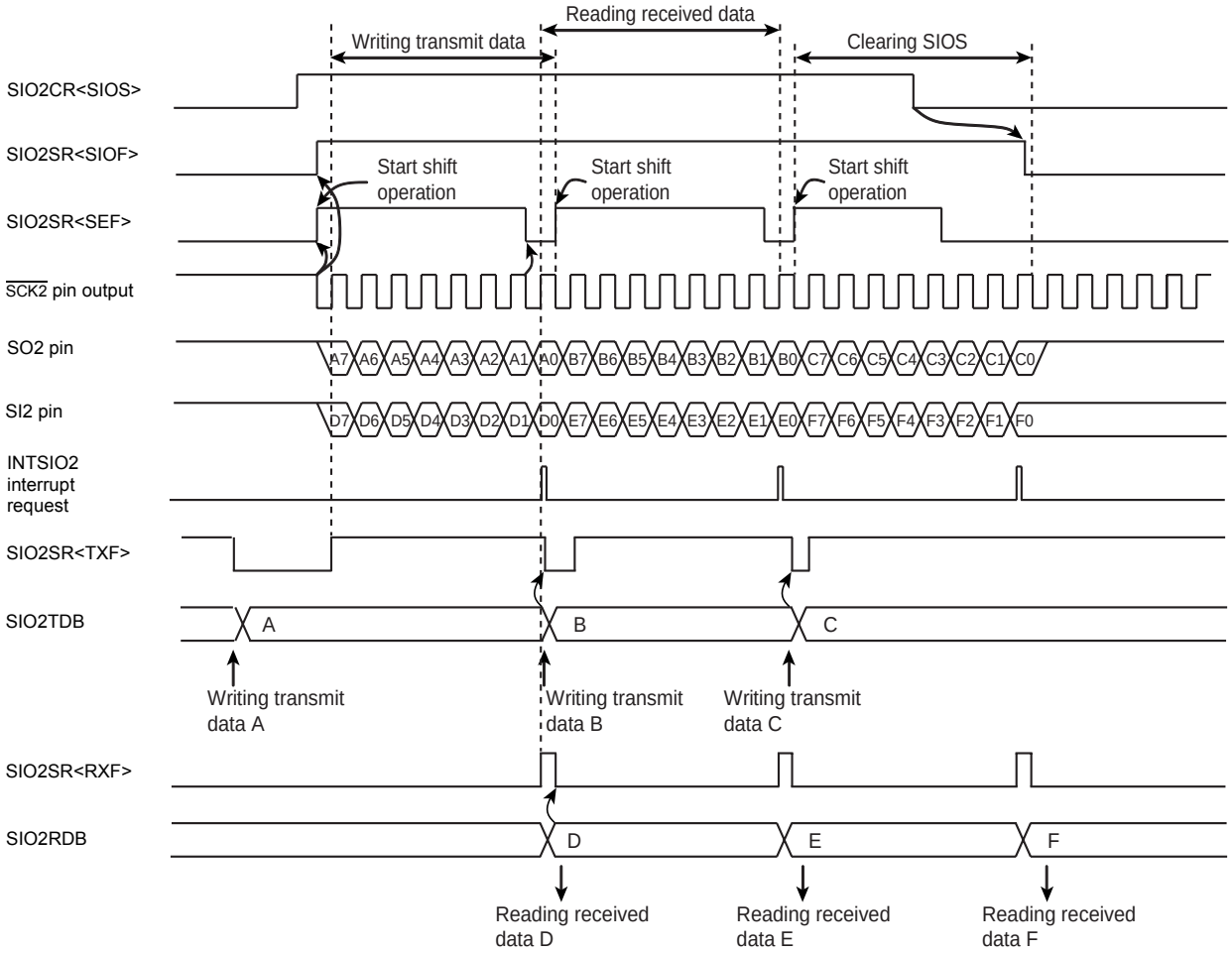



Figure 15-14 Example of External Clock and MSB Transmit/Receive Mode

### (4) Transmit/receive error processing

Transmit/receive errors occur on the following situation. Corrective action is different, which errors occur transmits or receives.

#### (a) Transmit errors

Transmit errors occur on the following situation.

- Shift operation starts before writing next transmit data to SIO2TDB in external clock operation.
  If transmit errors occur during transmit operation, SIO2SR<TXERR> is set to “1” immediately after starting shift operation. And INTSIO2 interrupt request is generated after all of the 8-bit data has been received.
  If shift operation starts before writing data to SIO2TDB after SIO2CR<SIOS> is set to “1”, SIO2SR<TXERR> is set immediately after starting shift operation. And INTSIO2 interrupt request is generated after all of the 8-bit data has been received.
  SO2 pin is kept in high level when SIO2SR<TXERR> is set to “1”. When transmit error occurs, transmit operation must be forcibly stop by writing SIO2CR<SIOINH> to “1” after the received data is read from SIO2RDB. In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

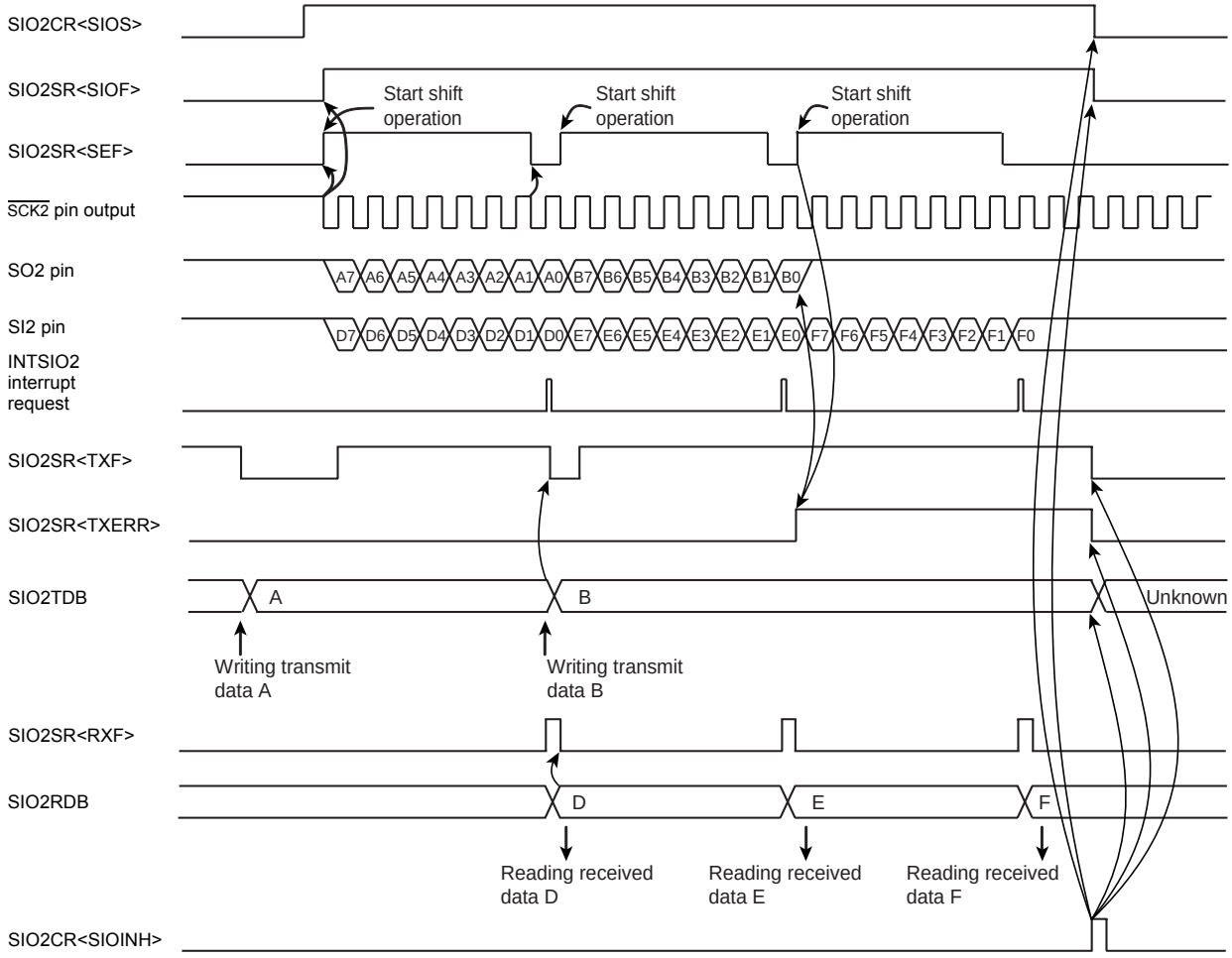



Figure 15-15 Example of Transmit/Receive (Transmit) Error Processing

(b) Receive errors

Receive errors occur on the following situation. To protect SIO2RDB and the shift register contents, the received data is ignored while the SIO2SR<RXERR> is "1".

- Shift operation is finished before reading out received data from SIO2RDB at SIO2SR<RXF> is "1" in an external clock operation.
  If receive error occurs, set the SIO2CR<SIOS> to "0" for reading the data that received immediately before error occurence. And read the data from SIO2RDB. Data in shift register (at errors occur) can be read by reading the SIO2RDB again.
  When SIO2SR<RXERR> is cleared to "0" after reading the received data, SIO2SR<RXF> is cleared to "0".
  After clearing SIO2CR<SIOS> to "0", when 8-bit serial clock is input to $\overline{\text{SCK2}}$ pin, receive operation is stopped. To restart the receive operation, confirm that SIO2SR<SIOF> is cleared to "0".
  If the received error occurs, set the SIO2CR<SIOINH> to "1" for stopping the receive operation immediately. In this case, SIO2CR<SIOS>, SIO2SR register, SIO2RDB register and SIO2TDB register are initialized.

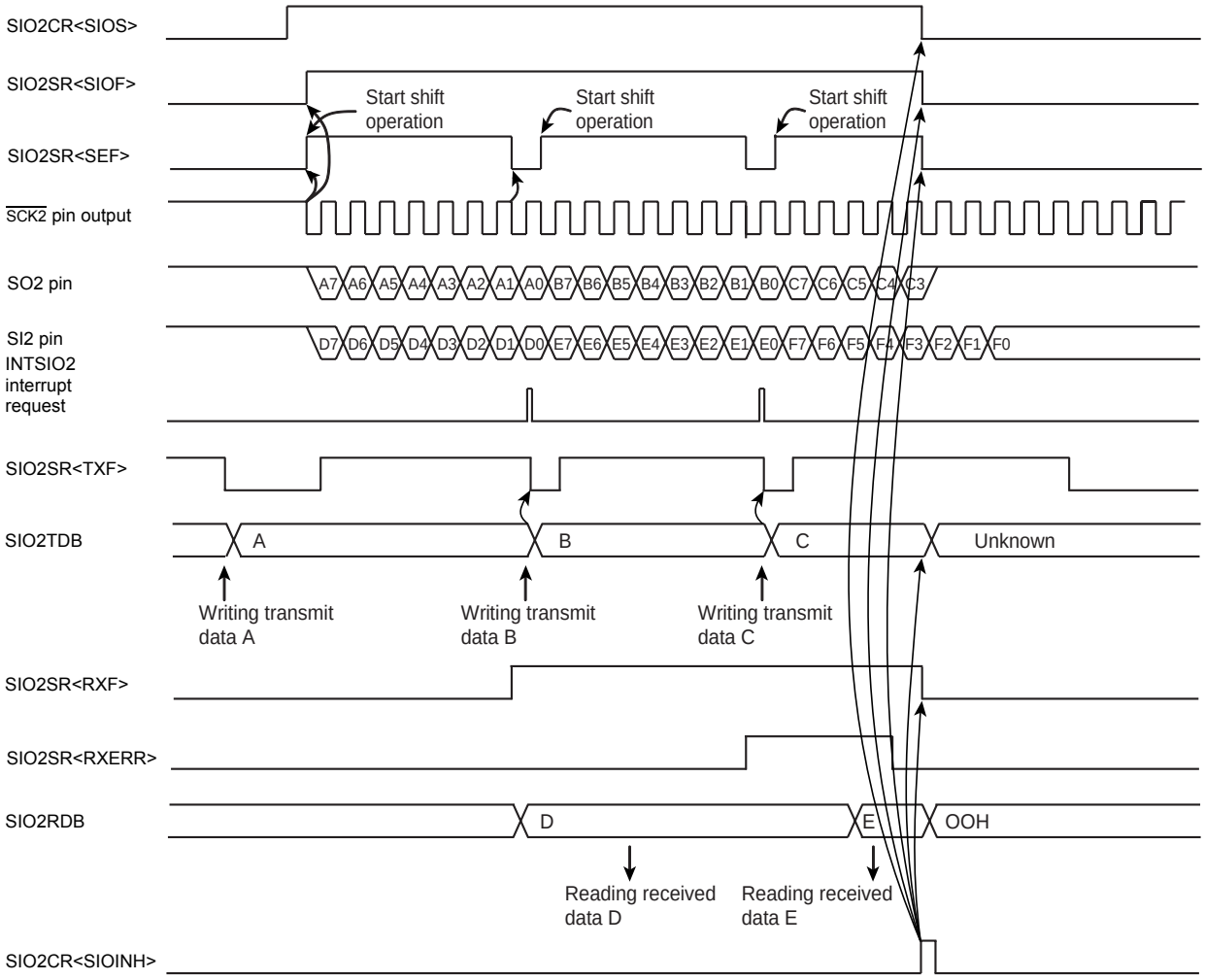



Figure 15-16 Example of Transmit/Receive (Receive) Error Processing

Note: If receive error is not corrected, an interrupt request does not generate after the error occurs.

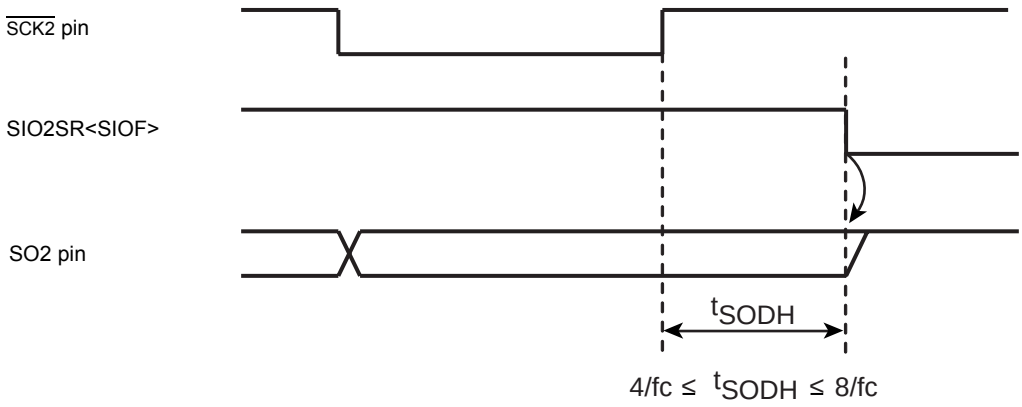



Figure 15-17 Hold Time of the End of Transmit/Receive Mode

# 16. Serial Bus Interface($I^2C$ Bus) Ver.-D (SBI)

The TMP86CS49UG has a serial bus interface which employs an $I^2C$ bus.

The serial interface is connected to an external devices through SDA and SCL.

The serial bus interface pins are also used as the port. When used as serial bus interface pins, set the output latches of these pins to "1". When not used as serial bus interface pins, the port is used as a normal I/O port.

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

Note 2: The serial bus interface can be used only in the Standard mode of $I^2C$. The fast mode and the high-speed mode can not be used.

Note 3: Please refer to the I/O port section about the detail of setting port.

## 16.1 Configuration

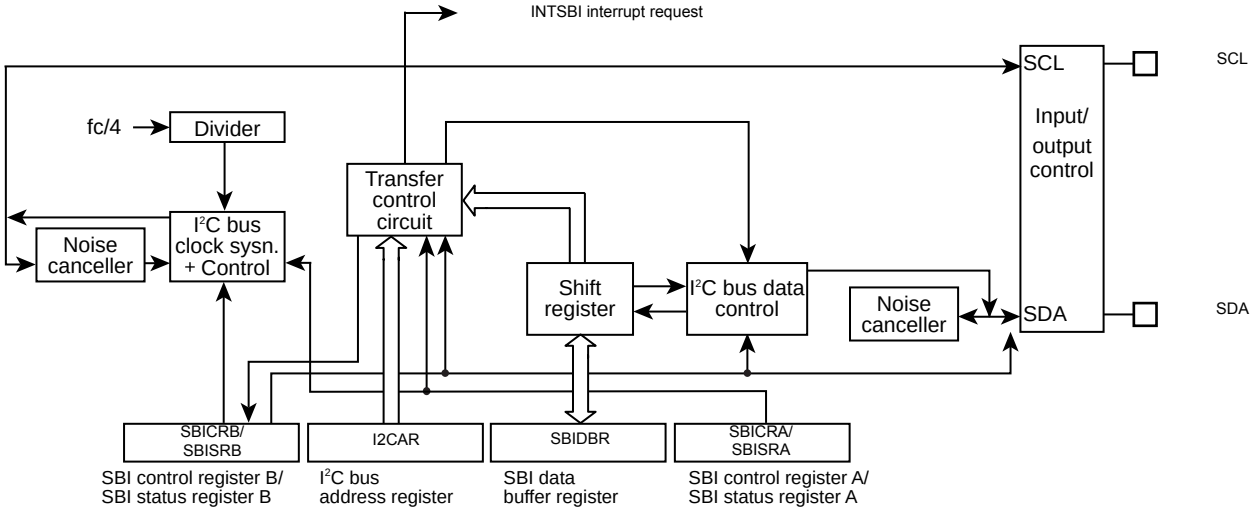



Figure 16-1 Serial Bus Interface (SBI)

## 16.2 Control

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

- Serial bus interface control register A (SBICRA)
- Serial bus interface control register B (SBICRB)
- Serial bus interface data buffer register (SBIDBR)
- $I^2C$ bus address register (I2CAR)
- Serial bus interface status register A (SBISRA)
- Serial bus interface status register B (SBISRB)

## 16.3 Software Reset

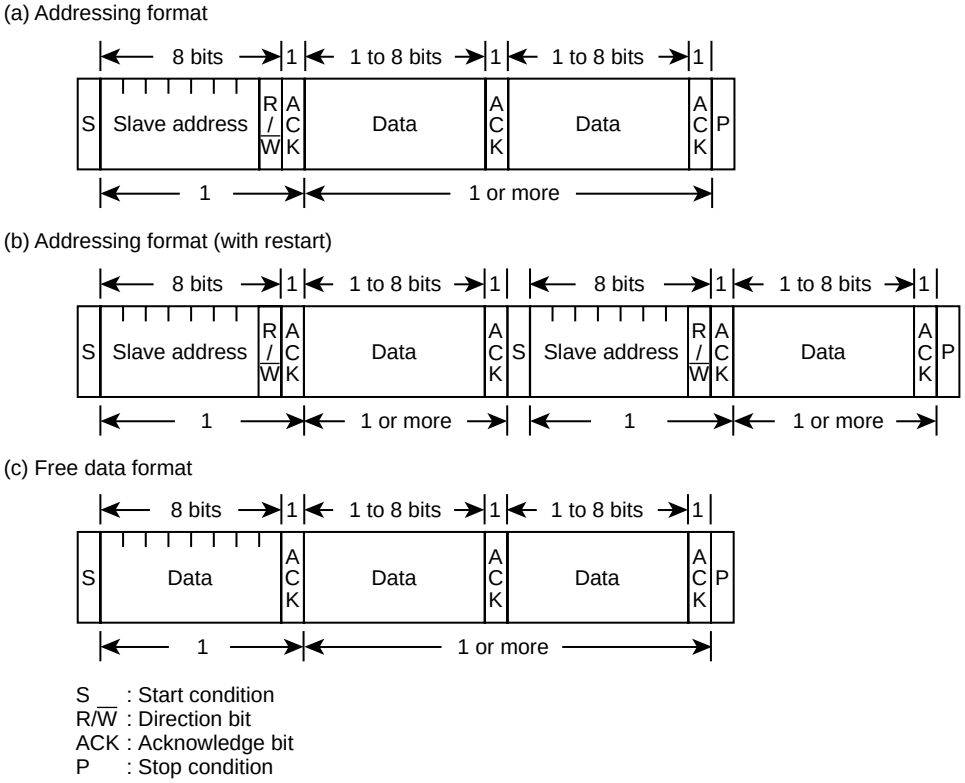
A serial bus interface circuit has a software reset function, when a serial bus interface circuit is locked by an external noise, etc.

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

And a status of software reset canbe read from SWRMON (Bit0 in SBISRA).

# 16.4 The Data Format in the $I^2C$ Bus Mode

The data format of the $I^2C$ bus is shown below.




S : Start condition
R/$\overline{W}$ : Direction bit
ACK : Acknowledge bit
P : Stop condition

Figure 16-2 Data Format in of $I^2C$ Bus

## 16.5 $I^2C$ Bus Control

The following registers are used to control the serial bus interface and monitor the operation status of the $I^2C$ bus.

### Serial Bus Interface Control Register A

SBICRA (0F90H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BC | | | ACK | | SCK | | |

(Initial value: 0000 *000)

| | | BC | ACK = 0 | | ACK = 1 | | |
|---|---|---|---|---|---|---|---|
| | | | Number of Clock | Bits | Number of Clock | Bits | |
| BC | Number of transferred bits | 000: | 8 | 8 | 9 | 8 | Write only |
| | | 001: | 1 | 1 | 2 | 1 | |
| | | 010: | 2 | 2 | 3 | 2 | |
| | | 011: | 3 | 3 | 4 | 3 | |
| | | 100: | 4 | 4 | 5 | 4 | |
| | | 101: | 5 | 5 | 6 | 5 | |
| | | 110: | 6 | 6 | 7 | 6 | |
| | | 111: | 7 | 7 | 8 | 7 | |
| ACK | Acknowledgement mode specification | ACK | Master mode | | Slave mode | | R/W |
| | | 0: | Not generate a clock pulse for an acknowledgement. | | Not count a clock pulse for an acknowledgement. | | |
| | | 1: | Generate a clock pulse for an acknowledgement. | | Count a clock pulse for an acknowledgement. | | |
| SCK | Serial clock (fscl) selection (Output on SCL pin) [fscl = $1/(2^{n+1}/fc + 8/fc)$] | SCK | n | At fc = 16 MHz | At fc = 8 MHz | At fc = 4 MHz | Write only |
| | | 000: | 4 | Reserved | Reserved | 100.0 kHz | |
| | | 001: | 5 | Reserved | Reserved | 55.6 kHz | |
| | | 010: | 6 | Reserved | 58.8 kHz | 29.4 kHz | |
| | | 011: | 7 | 60.6 kHz | 30.3 kHz | 15.2 kHz | |
| | | 100: | 8 | 30.8 kHz | 15.4 kHz | 7.7 kHz | |
| | | 101: | 9 | 15.5 kHz | 7.8 kHz | 3.9 kHz | |
| | | 110: | 10 | 7.8 kHz | 3.9 kHz | 1.9 kHz | |
| | | 111: | Reserved | | | | |

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

Note 2: SBICRA cannot be used with any of read-modify-write instructions such as bit manipulation, etc.

Note 3: Do not set SCK as the frequency that is over 100 kHz.

### Serial Bus Interface Data Buffer Register

SBIDBR (0F91H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

(Initial value: **** ****) R/W

Note 1: For writing transmitted data, start from the MSB (Bit7).

Note 2: The data which was written into SBIDBR can not be read, since a write data buffer and a read buffer are independent in SBIDBR. Therefore, SBIDBR cannot be used with any of read-modify-write instructions such as bit manipulation, etc.

Note 3: *: Don't care

## $I^2C$ bus Address Register

I2CAR (0F92H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| Slave address | | | | | | | ALS | (Initial value: 0000 0000) |
| SA6 | SA5 | SA4 | SA3 | SA2 | SA1 | SA0 | | |

| | | | |
|---|---|---|---|
| SA | Slave address selection | | Write only |
| ALS | Address recognition mode specification | 0: Slave address recognition<br>1: Non slave address recognition | |

Note 1: I2CAR is write-only register, which cannot be used with any of read-modify-write instruction such as bit manipulation, etc.

Note 2: Do not set I2CAR to "00H" to avoid the incorrect response of acknowledgment in slave mode. ( If "00H" is set to I2CAR as the Slave Address and a START Byte "01H" in $I^2C$ bus standard is recived, the device detects slave address match.)

## Serial Bus Interface Control Register B

SBICRB (0F93H)

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

| | | | |
|---|---|---|---|
| MST | Master/slave selection | 0: Slave<br>1: Master | Write only |
| TRX | Transmitter/receiver selection | 0: Receiver<br>1: Transmitter | |
| BB | Start/stop generation | 0: Generate a stop condition when MST, TRX and PIN are "1"<br>1: Generate a start condition when MST, TRX and PIN are "1" | |
| PIN | Cancel interrupt service request | 0: – (Can not clear this bit by a software)<br>1: Cancel interrupt service request | |
| SBIM | Serial bus interface operating mode selection | 00: Port mode (Serial bus interface output disable)<br>01: Reserved<br>10: $I^2C$ bus mode<br>11: Reserved | |
| SWRST1<br>SWRST0 | Software reset start bit | Software reset starts by first writing "10" and next writing "01" | |

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

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

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

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

## Serial Bus Interface Status Register A

SBISRA (0F90H)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | SWRMON | (Initial value: **** ***1) |

| | | | |
|---|---|---|---|
| SWRMON | Software reset monitor | 0: During software reset<br>1: – (Initial value) | Read only |

## Serial Bus Interface Status Register B

SBISRB (0F93H)

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

| | | | |
|---|---|---|---|
| MST | Master/slave selection status monitor | 0: Slave<br>1: Master | Read only |
| TRX | Transmitter/receiver selection status monitor | 0: Receiver<br>1: Transmitter | |
| BB | Bus status monitor | 0: Bus free<br>1: Bus busy | |
| PIN | Interrupt service requests status monitor | 0: Requesting interrupt service<br>1: Releasing interrupt service request | |
| AL | Arbitration lost detection monitor | 0: –<br>1: Arbitration lost detected | |
| AAS | Slave address match detection monitor | 0: -<br>1: Detect slave address match or "GENERAL CALL" | |
| AD0 | "GENERAL CALL" detection monitor | 0: -<br>1: Detect "GENERAL CALL" | |
| LRB | Last received bit monitor | 0: Last receive bit is "0"<br>1: Last receiv bit is "1" | |

## 16.5.1 Acknowledgement mode specification

### 16.5.1.1 Acknowledgment mode (ACK = “1”)

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

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

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

The Table 16-1 shows the SCL and SDA pins status in acknowledgment mode.

Table 16-1 SCL and SDA Pins Status in Acknowledgement Mode

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

### 16.5.1.2 Non-acknowledgment mode (ACK = “0”)

To set the device as a non-acknowledgement mode, the ACK (Bit4 in SBICRA) should be cleared to “0”.

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

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

## 16.5.2 Number of transfer bits

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

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

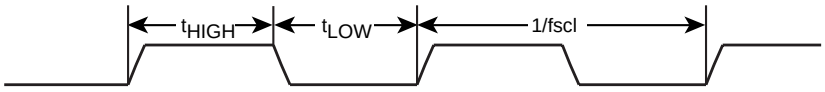
## 16.5.3 Serial clock

### 16.5.3.1 Clock source

The SCK (Bits2 to 0 in SBICRA) is used to select a maximum transfer frequency output from the SCL pin in the master mode.

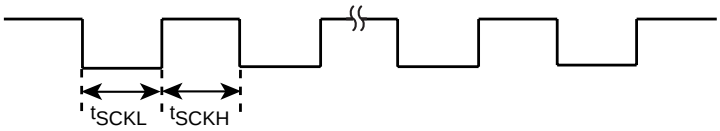
Four or more machine cycles are required for both high and low levels of pulse width in the external clock which is input from SCL pin.

Note: Since the serial bus interface can not be used as the fast mode and the high-speed mode, do not set SCK as the frequency that is over 100 kHz.




$t_{LOW} = 2^n/fc$
$t_{HIGH} = 2^n/fc + 8/fc$
$fscl = 1/(t_{LOW} + t_{HIGH})$

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




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

Note 1: fc = High-frequency clock
Note 2: tcyc = 4/fc (in NORMAL mode, IDLE mode)

Figure 16-3 Clock Source

### 16.5.3.2 Clock synchronization

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

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

The example explains clock synchronization procedures when two masters simultaneously exist on a bus.

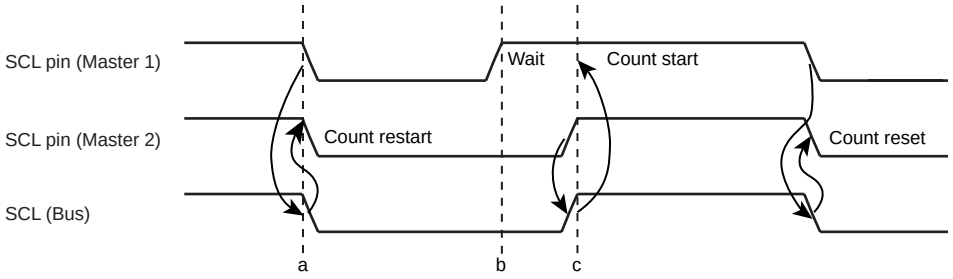



Figure 16-4 Clock Synchronization

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

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

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

### 16.5.4 Slave address and address recognition mode specification

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

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

### 16.5.5 Master/slave selection

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

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

### 16.5.6 Transmitter/receiver selection

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

When a stop condition on the bus or an arbitration lost is detected, the TRX is cleared to “0” by the hardware. " Table 16-2 TRX changing conditions in each mode " shows TRX changing conditions in each mode and TRX value after changing

Table 16-2 TRX changing conditions in each mode

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

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

## 16.5.7 Start/stop condition generation

When the BB (Bit5 in SBISRB) is “0”, a slave address and a direction bit which are set to the SBIDBR are output on a bus after generating a start condition by writing “1” to the MST, TRX, BB and PIN. It is necessary to set ACK to “1” beforehand.

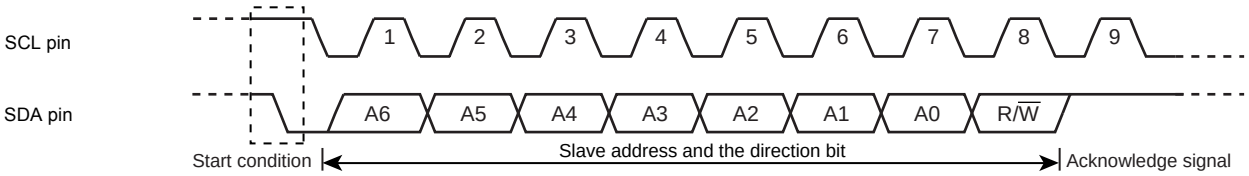



Figure 16-5 Start Condition Generation and Slave Address Generation

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

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

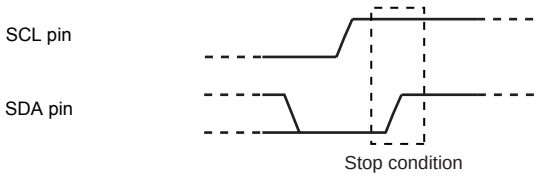



Figure 16-6 Stop Condition Generation

The bus condition can be indicated by reading the contents of the BB (Bit5 in SBISRB). The BB is set to “1” when a start condition on a bus is detected (Bus Busy State) and is cleared to “0” when a stop condition is detected (Bus Free State).

## 16.5.8 Interrupt service request and cancel

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

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

- At the end of acknowledge signal when the received slave address matches to the value set by the I2CAR
- At the end of acknowledge signal when a “GENERAL CALL” is received
- At the end of transferring or receiving after matching of slave address or receiving of “GENERAL CALL”

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

Either writing data to SBIDBR or reading data from the SBIDBR sets the PIN to “1”.

The time from the PIN being set to “1” until the SCL pin is released takes $t_{LOW}$.

Although the PIN (Bit4 in SBICRB) can be set to “1” by the softrware, the PIN can not be cleared to “0” by the softrware.

Note: When the arbitration lost occurs, if the slave address sent from the other master devices is not match, the INTSBI interrupt request is generated. But the PIN is not cleared.

### 16.5.9 Setting of $I^2C$ bus mode

The SBIM (Bit3 and 2 in SBICRB) is used to set $I^2C$ bus mode.

Set the SBIM to “10” in order to set $I^2C$ bus mode. Before setting of $I^2C$ bus mode, confirm serial bus interface pins in a high level, and then, write “10” to SBIM. And switch a port mode after confirming that a bus is free.

### 16.5.10 Arbitration lost detection monitor

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

Data on the SDA line is used for bus arbitration of the $I^2C$ bus.

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

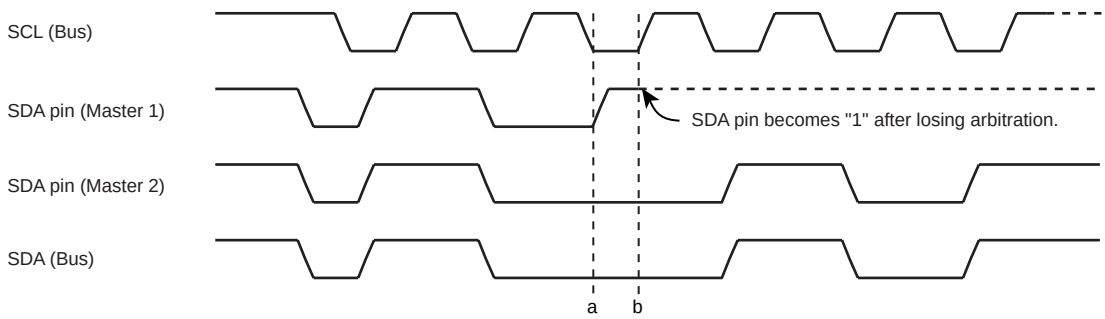



Figure 16-7 Arbitration Lost

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

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

The AL is cleared to “0” by writing data to the SBIDBR, reading data from the SBIDBR or writing data to the SBICRB.

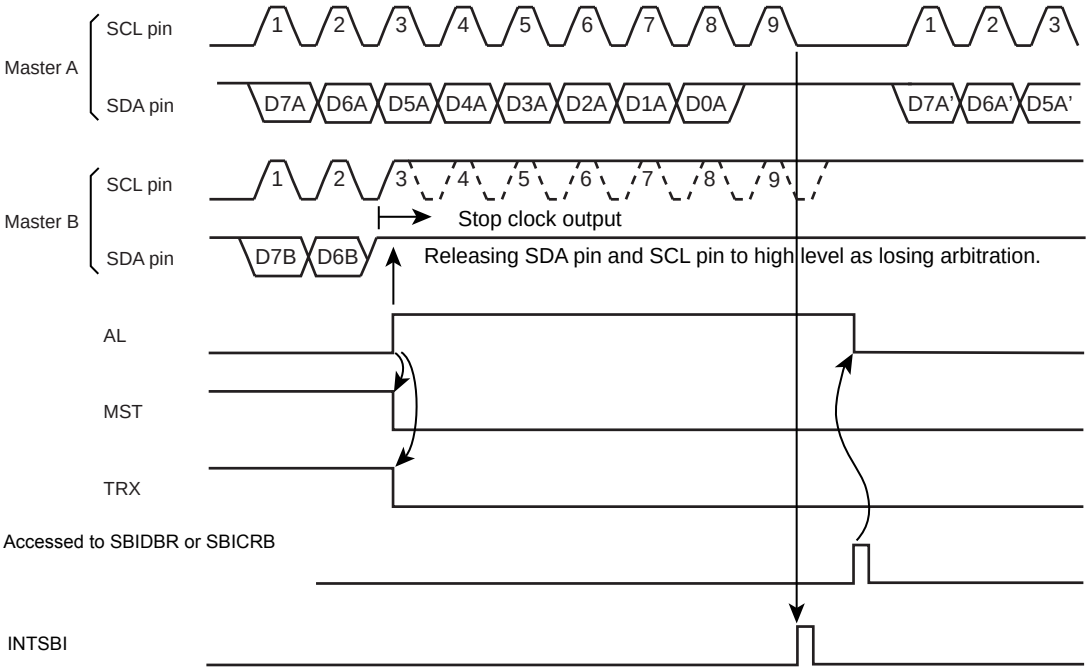



Figure 16-8 Example of when a Serial Bus Interface Circuit is a Master B

## 16.5.11 Slave address match detection monitor

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

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

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

## 16.5.12 GENERAL CALL detection monitor

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

## 16.5.13 Last received bit monitor

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

## 16.6 Data Transfer of I2C Bus

### 16.6.1 Device initialization

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

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

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

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

### 16.6.2 Start condition and slave address generation

Confirm a bus free status (BB = 0).

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

By writing “1” to the MST, TRX, BB and PIN, the start condition is generated on a bus and then, the slave address and the direction bit which are set to the SBIDBR are output. The time from generating the START condition until the falling SCL pin takes $t_{LOW}$.

An INTSBI interrupt request occurs at the 9th falling edge of a SCL clock cycle, and the PIN is cleared to “0”. The SCL pin is pulled-down to the low level while the PIN is “0”. When an interrupt request occurs, the TRX changes by the hardware according to the direction bit only when an acknowledge signal is returned from the slave device.

Note 1: Do not write a slave address to be output to the SBIDBR while data is transferred. If data is written to the SBIDBR, data to been outputting may be destroyed.

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

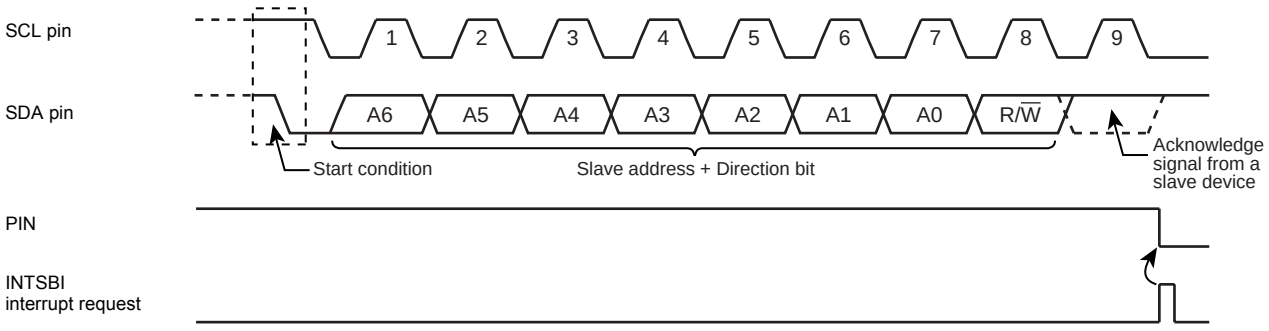



Figure 16-9 Start Condition Generation and Slave Address Transfer

### 16.6.3 1-word data transfer

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

#### 16.6.3.1 When the MST is “1” (Master mode)

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

(1) When the TRX is “1” (Transmitter mode)

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

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

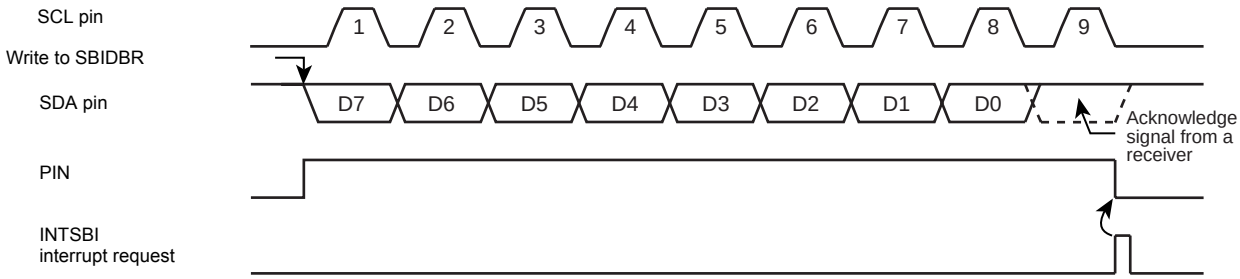



Figure 16-10 Example of when BC = “000”, ACK = “1”

(2) When the TRX is “0” (Receiver mode)

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

An INTSBI interrupt request occurs and the PIN becomes “0”. Then a serial bus interface circuit outputs a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

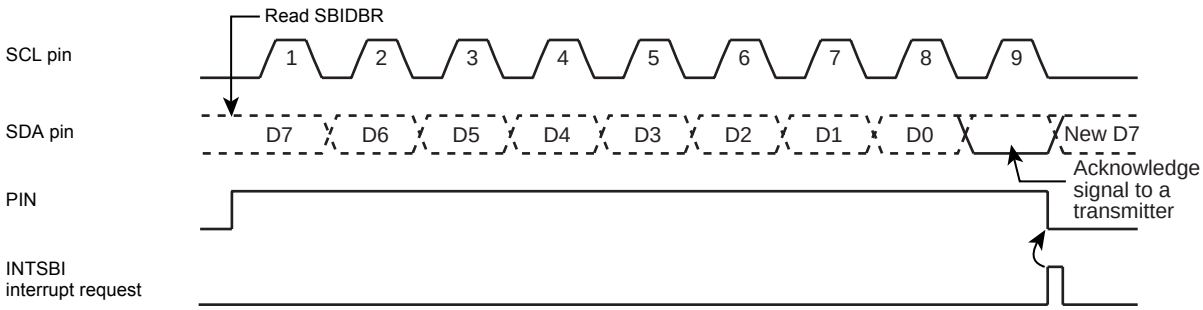



Figure 16-11 Example of when BC = “000”, ACK = “1”

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

After 1-bit data is received and an interrupt request has occurred, generate the stop condition to terminate data transfer.

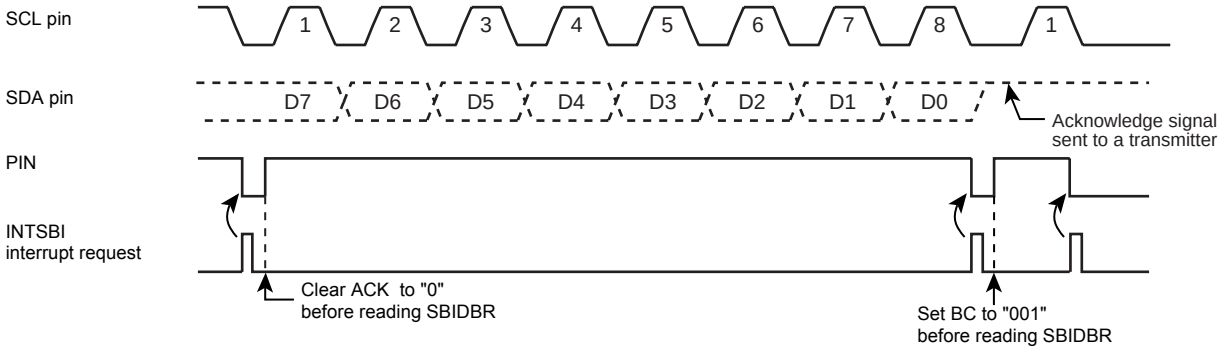



Figure 16-12 Termination of Data Transfer in Master Receiver Mode

### 16.6.3.2 When the MST is “0” (Slave mode)

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

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

- At the end of acknowledge signal when the received slave address matches to the value set by the I2CAR
- At the end of acknowledge signal when a “GENERAL CALL” is received
- At the end of transferring or receiving after matching of slave address or receiving of “GENERAL CALL”

A serial bus interface circuit changes to a slave mode if arbitration is lost in the master mode. And an INTSBI interrupt request occurs when word data transfer terminates after losing arbitration. The behavior of INTSBI interrupt request and PIN after losing arbitration are shown in Table 16-3.

Table 16-3 The Behavior of INTSBI interrupt request and PIN after Losing Arbitration

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

When an INTSBI interrupt request occurs, the PIN (bit 4 in the SBICRB) is reset, and the SCL pin is set to low level. Either reading or writing from or to the SBIDBR or setting the PIN to “1” releases the SCL pin after taking $t_{LOW}$.

Check the AL (Bit3 in the SBISRB), the TRX (Bit6 in the SBISRB), the AAS (Bit2 in the SBISRB), and the AD0 (Bit1 in the SBISRB) and implements processes according to conditions listed in " Table 16-4 Operation in the Slave Mode ".

Table 16-4 Operation in the Slave Mode

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

Note: In the slave mode, if the slave address set in I2CAR is "00H", a START Byte "01H" in $I^2C$ bus standard is recived, the device detects slave address match and the TRX is set to "1".

## 16.6.4 Stop condition generation

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

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

The time from the releasing SCL line until the generating the STOP condition takes $t_{LOW}$.

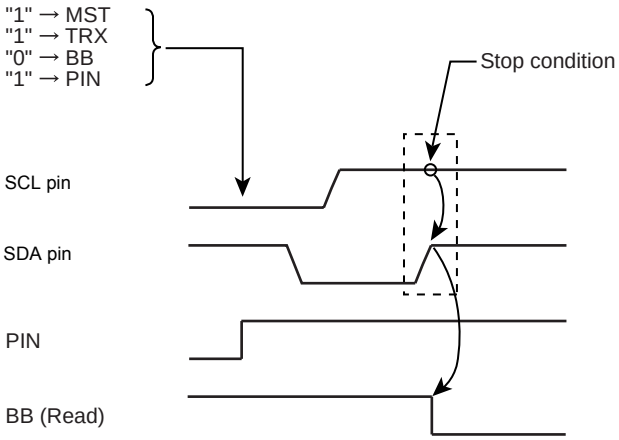



Figure 16-13 Stop Condition Generation

### 16.6.5 Restart

Restart is used to change the direction of data transfer between a master device and a slave device during transferring data. The following explains how to restart a serial bus interface circuit.

Clear “0” to the MST, TRX and BB and set “1” to the PIN. The SDA pin retains the high-level and the SCL pin is released. Since a stop condition is not generated on a bus, a bus is assumed to be in a busy state from other devices. Test the BB until it becomes “0” to check that the SCL pin of a serial bus interface circuit is released. Test the LRB until it becomes “1” to check that the SCL line on a bus is not pulled-down to the low level by other devices. After confirming that a bus stays in a free state, generate a start condition with procedure " 16.6.2 Start condition and slave address generation ".

In order to meet setup time when restarting, take at least 4.7 μs of waiting time by software from the time of restarting to confirm that a bus is free until the time to generate a start condition.

Note: When the master is in the receiver mode, it is necessary to stop the data transmission from the slave devcie before the STOP condtion is generated. To stop the transmission, the master device make the slave device receiving a negative acknowledge. Therefore, the LRB is "1" before generating the Restart and it can not be confirmed that SCL line is not pulled-down by other devices. Please confirm the SCL line state by reading the port.

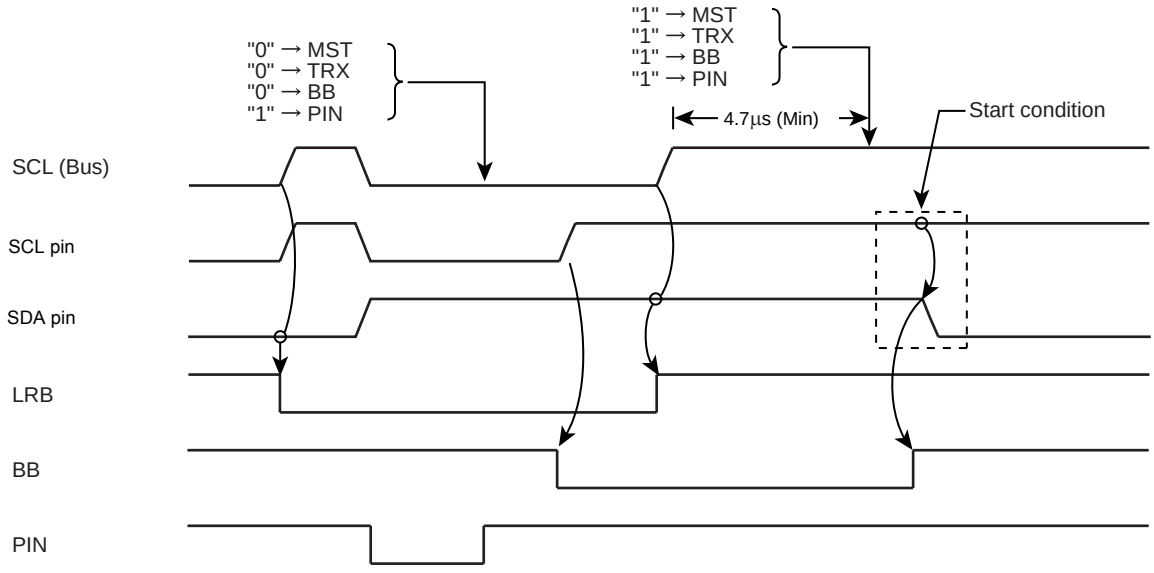



Figure 16-14 Timing Diagram when Restarting

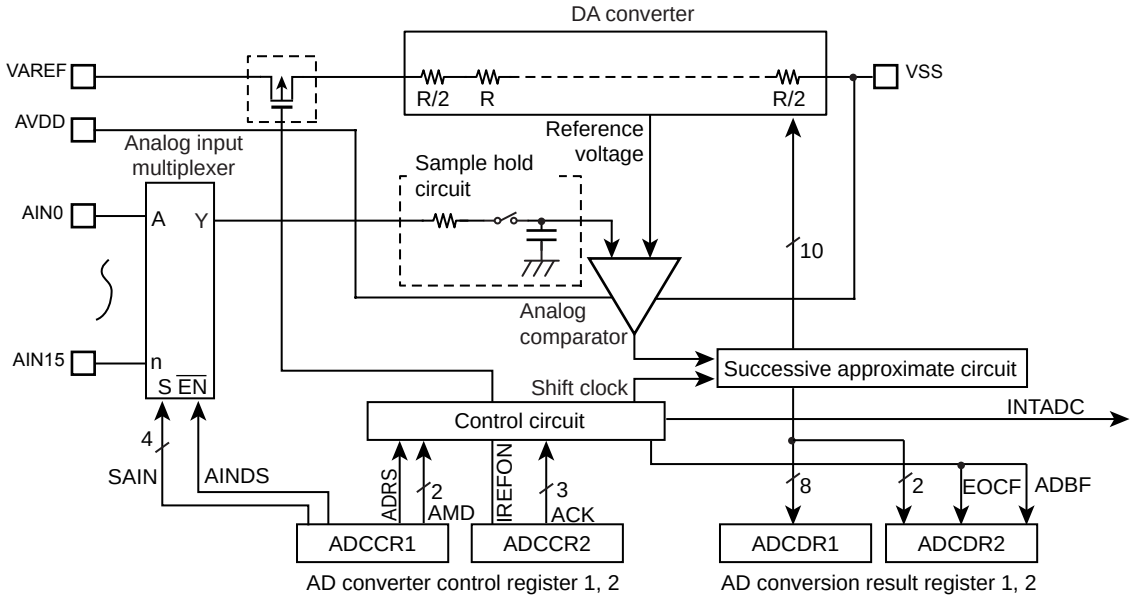
# 17. 10-bit AD Converter (ADC)

The TMP86CS49UG have a 10-bit successive approximation type AD converter.

## 17.1 Configuration

The circuit configuration of the 10-bit AD converter is shown in Figure 17-1.

It consists of control register ADCCR1 and ADCCR2, converted value register ADCDR1 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 conbining a analog input port. For details, see the section on "I/O ports".

Figure 17-1 10-bit AD Converter

## 17.2 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 fter 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

ADCCR1 (001CH)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADRS | AMD | | AINDS | SAIN | | | |

(Initial value: 0001 0000)

| | | | |
|---|---|---|---|
| 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: AIN6<br>0111: AIN7<br>1000: AIN8<br>1001: AIN9<br>1010: AIN10<br>1011: AIN11<br>1100: AIN12<br>1101: AIN13<br>1110: AIN14<br>1111: AIN15 | |

Note 1: Select analog input channel during AD converter stops (ADCDR2<ADBF> = "0").

Note 2: When the analog input channel is all use disabling, the ADCCR1<AINDS> 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 or SLOW/SLEEP mode are started, AD converter control register1 (ADCCR1) is all initialized and no data can be written in this register. Therfore, to use AD converter again, set the ADCCR1 newly after returning to NORMAL1 or NORMAL2 mode.

## AD Converter Control Register 2

| ADCCR2 (001DH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | | | IREFON | "1" | ACK | | | "0" | (Initial value: **0* 000*) |

| | | | |
|---|---|---|---|
| IREFON | DA converter (Ladder resistor) connection control | 0: Connected only during AD conversion<br>1: Always connected | R/W |
| ACK | AD conversion time select<br>(Refer to the following table about the conversion time) | 000: 39/fc<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 | |

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 or SLOW/SLEEP mode are started, AD converter control register2 (ADCCR2) is all initialized and no data can be written in this register. Therfore, to use AD converter again, set the ADCCR2 newly after returning to NORMAL1 or NORMAL2 mode.

### Table 17-1 ACK setting and Conversion time

| Condition / ACK | Conversion time | 16 MHz | 8 MHz | 4 MHz | 2 MHz | 10 MHz | 5 MHz | 2.5 MHz |
|---|---|---|---|---|---|---|---|---|
| 000 | 39/fc | - | - | - | 19.5 μs | - | - | 15.6 μs |
| 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 Analog reference voltage (VAREF) .

- VAREF = 4.5 to 5.5 V 15.6 μs and more
- VAREF = 2.7 to 5.5 V 31.2 μs and more
- VAREF = 1.8 to 5.5 V 124.8 μs and more

## AD Converted value Register 1

| ADCDR1 (001FH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | AD09 | AD08 | AD07 | AD06 | AD05 | AD04 | AD03 | AD02 | (Initial value: 0000 0000) |

## AD Converted value Register 2

| ADCDR2 (001EH) | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| | AD01 | AD00 | EOCF | ADBF | | | | | (Initial value: 0000 ****) |

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

Note 1: The ADCDR2<EOCF> is cleared to "0" when reading the ADCDR1. Therfore, 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 or SLOW mode .

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

## 17.3 Function

### 17.3.1 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).

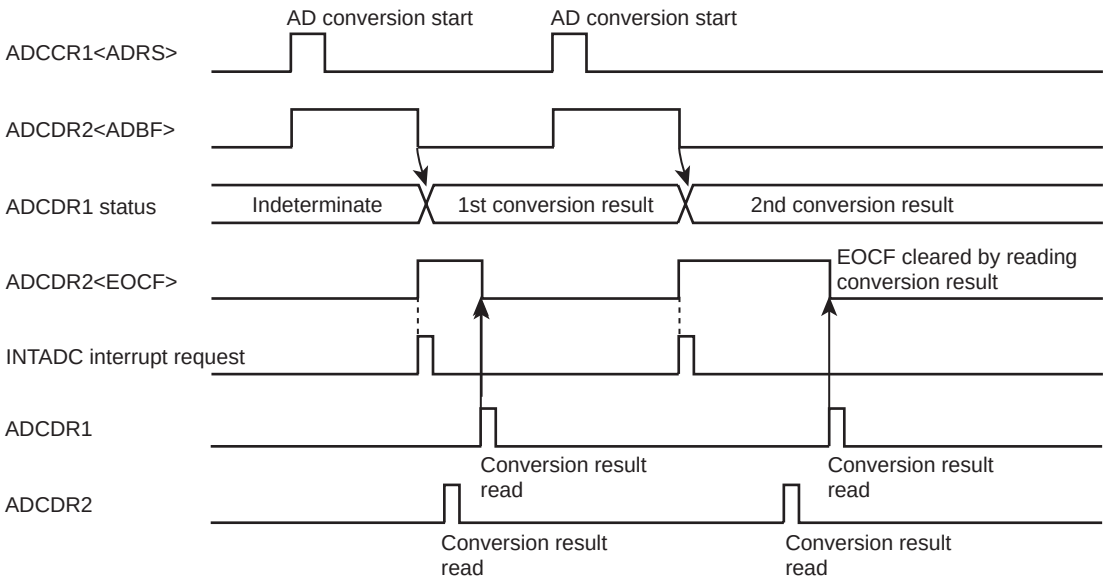



Figure 17-2 Software Start Mode

### 17.3.2 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.

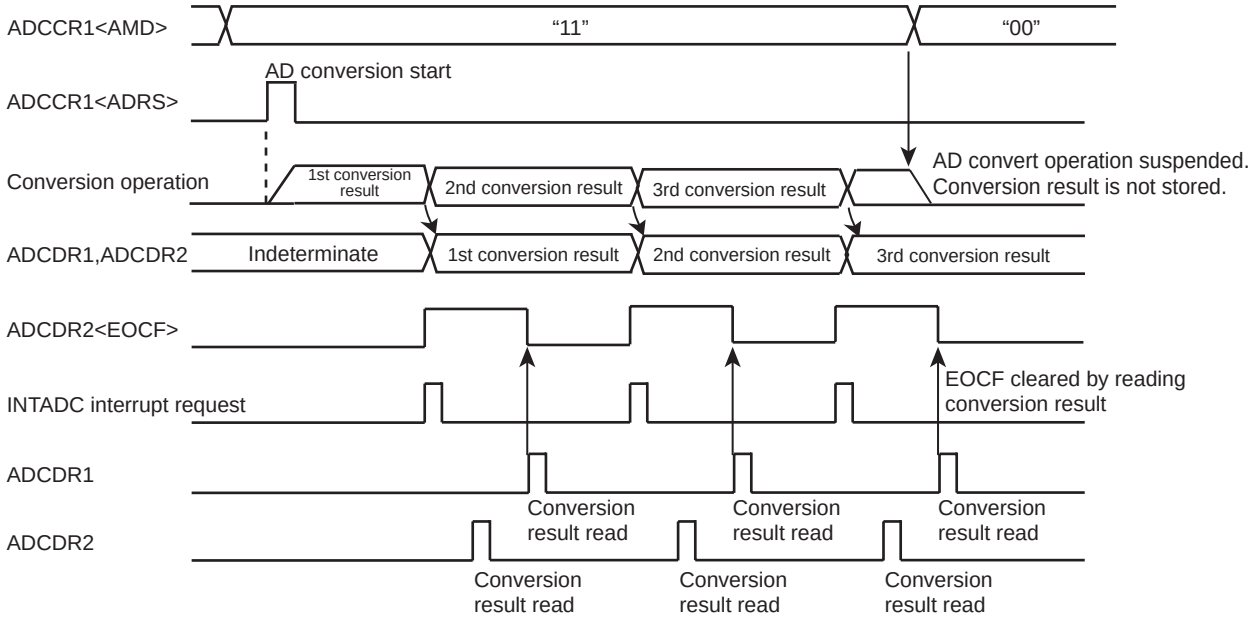



Figure 17-3 Repeat Mode

### 17.3.3 Register Setting

1. Set up the AD converter control register 1 (ADCCR1) as follows:
   - Choose the channel to AD convert using AD input channel select (SAIN).
   - Specify analog input enable for analog input control (AINDS).
   - Specify AMD for the AD converter control operation mode (software or repeat mode).
2. Set up the AD converter control register 2 (ADCCR2) as follows:
   - Set the AD conversion time using AD conversion time (ACK). For details on how to set the conversion time, refer to Figure 17-1 and AD converter control register 2.
   - Choose IREFON for DA converter control.
3. After setting up (1) and (2) above, set AD conversion start (ADRS) of AD converter control register 1 (ADCCR1) to "1". If software start mode has been selected, AD conversion starts immediately.
4. 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.
5. EOCF is cleared to "0" by a read of the conversion result. However, if reconverted before a register read, although EOCF is cleared the previous conversion result is retained until the next conversion is completed.

Example :After selecting the conversion time 19.5 μs at 16 MHz and the analog input channel AIN3 pin, perform AD conversion once. After checking EOCF, read the converted value, store the lower 2 bits in address 0009EH nd store the upper 8 bits in address 0009FH in RAM. The operation mode is software start mode.

```
                : (port setting)    :                   ;Set port register approrriately 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

                SET                 (ADCCR1) . 7        ; ADRS = 1(AD conversion start)
SLOOP :         TEST                (ADCDR2) . 5        ; EOCF= 1 ?
                JRS                 T, SLOOP

                LD                  A , (ADCDR2)        ; Read result data
                LD                  (9EH) , A
                LD                  A , (ADCDR1)        ; Read  result data
                LD                  (9FH), A
```

## 17.4 STOP/SLOW Modes during AD Conversion

When standby mode (STOP or SLOW mode) is entered forcibly during AD conversion, the AD convert operation is suspended and the AD converter is initialized (ADCCR1 and ADCCR2 are initialized to initial value). Also, the conversion result is indeterminate. (Conversion results up to the previous operation are cleared, so be sure to read the conversion results before entering standby mode (STOP or SLOW mode).) When restored from standby mode (STOP or SLOW 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.

## 17.5 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 17-4.

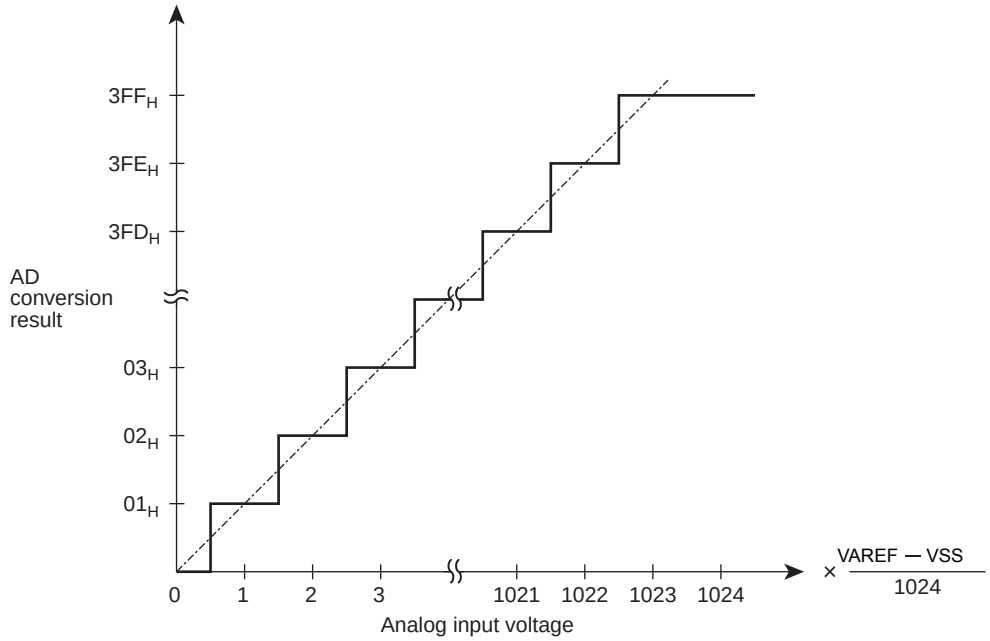



Figure 17-4 Analog Input Voltage and AD Conversion Result (Typ.)

## 17.6 Precautions about AD Converter

### 17.6.1 Restrictions for AD Conversion interrupt (INTADC) usage

When an AD interrupt is used, it may not be processed depending on program composition. For example, if an INTADC interrupt request is generated while an interrupt with priority lower than the interrupt latch IL15 (INTADC) is being accepted, the INTADC interrupt latch may be cleared without the INTADC interrupt being processed.

The completion of AD conversion can be detected by the following methods:

(1) Method not using the AD conversion end interrupt

Whether or not AD conversion is completed can be detected by monitoring the AD conversion end flag (EOCF) by software. This can be done by polling EOCF or monitoring EOCF at regular intervals after start of AD conversion.

(2) Method for detecting AD conversion end while a lower-priority interrupt is being processed

While an interrupt with priority lower than INTADC is being processed, check the AD conversion end flag (EOCF) and interrupt latch IL15. If IL15 = 0 and EOCF = 1, call the AD conversion end interrupt processing routine with consideration given to PUSH/POP operations. At this time, if an interrupt request with priority higher than INTADC has been set, the AD conversion end interrupt processing routine will be executed first against the specified priority. If necessary, we recommend that the AD conversion end interrupt processing routine be called after checking whether or not an interrupt request with priority higher than INTADC has been set.

### 17.6.2 Analog input pin voltage range

Make sure the analog input pins (AIN0 to AIN15) are used at voltages within VAREF 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.

### 17.6.3 Analog input shared pins

The analog input pins (AIN0 to AIN15) 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.

### 17.6.4 Noise Countermeasure

The internal equivalent circuit of the analog input pins is shown in Figure 17-5. 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 kΩ or less. Toshiba also recommends attaching a capacitor external to the chip.

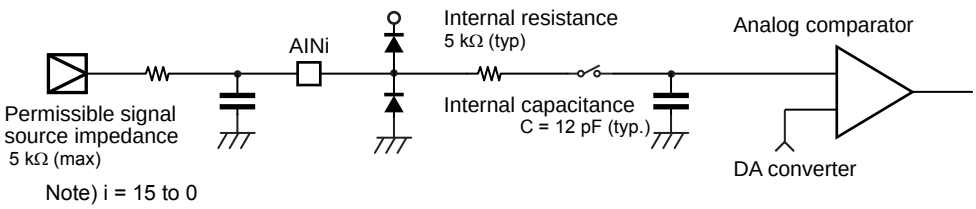



Figure 17-5 Analog Input Equivalent Circuit and Example of Input Pin Processing

# 18. Key-on Wakeup (KWU)

In the TMP86CS49UG, the STOP mode is released by not only P20($\overline{INT5}$/$\overline{STOP}$) pin but also four (STOP0 to STOP3) pins.

When the STOP mode is released by STOP0 to STOP3 pins, the $\overline{STOP}$ pin needs to be used.
In details, refer to the following section " 18.2 Control ".

## 18.1 Configuration

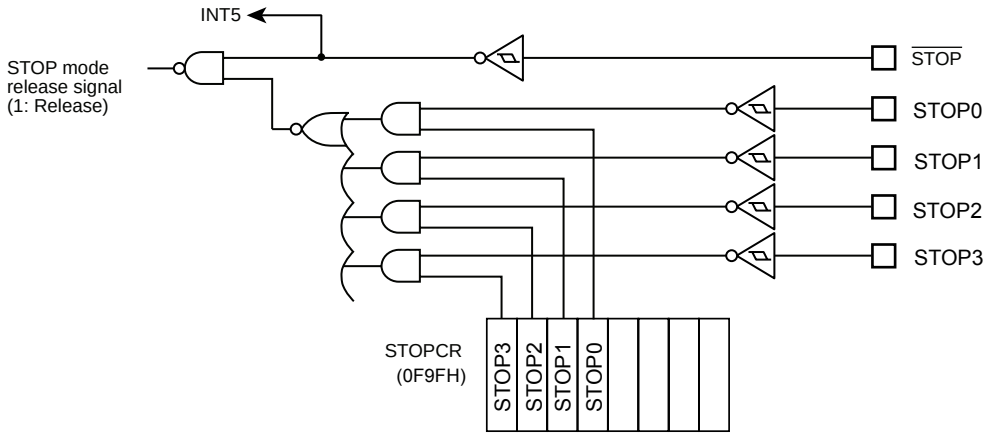



Figure 18-1 Key-on Wakeup Circuit

## 18.2 Control

STOP0 to STOP3 pins can controlled by Key-on Wakeup Control Register (STOPCR). It can be configured as enable/disable in 1-bit unit. When those pins are used for STOP mode release, configure corresponding I/O pins to input mode by I/O port register beforehand.

Key-on Wakeup Control Register

| STOPCR | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
|---|---|---|---|---|---|---|---|---|---|
| (0F9FH) | STOP3 | STOP2 | STOP1 | STOP0 | | | | | (Initial value: 0000 ****) |

| | | | |
|---|---|---|---|
| STOP3 | STOP mode released by STOP3 | 0:Disable<br>1:Enable | Write only |
| STOP2 | STOP mode released by STOP2 | 0:Disable<br>1:Enable | Write only |
| STOP1 | STOP mode released by STOP1 | 0:Disable<br>1:Enable | Write only |
| STOP0 | STOP mode released by STOP0 | 0:Disable<br>1:Enable | Write only |

## 18.3 Function

Stop mode can be entered by setting up the System Control Register (SYSCR1), and can be exited by detecting the "L" level on STOP0 to STOP3 pins, which are enabled by STOPCR, for releasing STOP mode (Note1).

Also, each level of the STOP0 to STOP3 pins can be confirmed by reading corresponding I/O port data register, check all STOP0 to STOP3 pins "H" that is enabled by STOPCR before the STOP mode is started (Note2,3).

Note 1: When the STOP mode released by the edge release mode (SYSCR1<RELM> = “0”), inhibit input from STOP0 to STOP3 pins by Key-on Wakeup Control Register (STOPCR) or must be set "H" level into STOP0 to STOP3 pins that are available input during STOP mode.

Note 2: When the $\overline{\text{STOP}}$ pin input is high or STOP0 to STOP3 pins input which is enabled by STOPCR is low, executing an instruction which starts STOP mode will not place in STOP mode but instead will immediately start the release sequence (Warm up).

Note 3: The input circuit of Key-on Wakeup input and Port input is separated, so each input voltage threshold value is different. Therefore, a value comes from port input before STOP mode start may be different from a value which is detected by Key-on Wakeup input (Figure 18-2).

Note 4: $\overline{\text{STOP}}$ pin doesn’t have the control register such as STOPCR, so when STOP mode is released by STOP0 to STOP3 pins, $\overline{\text{STOP}}$ pin also should be used as STOP mode release function.

Note 5: In STOP mode, Key-on Wakeup pin which is enabled as input mode (for releasing STOP mode) by Key-on Wakeup Control Register (STOPCR) may generate the penetration current, so the said pin must be disabled AD conversion input (analog voltage input).

Note 6: When the STOP mode is released by STOP0 to STOP3 pins, the level of $\overline{\text{STOP}}$ pin should hold "L" level (Figure 18-3).

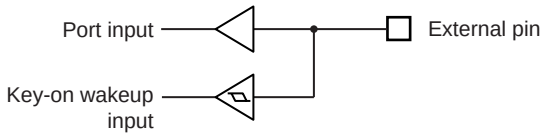



Figure 18-2 Key-on Wakeup Input and Port Input

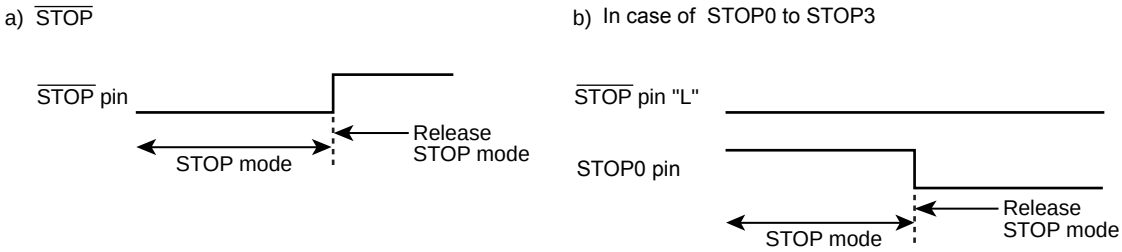



Figure 18-3 Priority of $\overline{\text{STOP}}$ pin and STOP0 to STOP3 pins

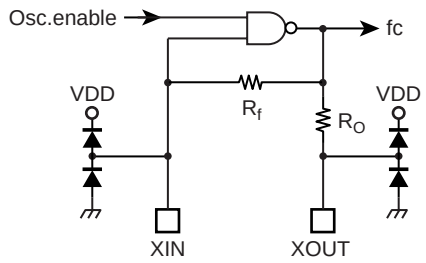
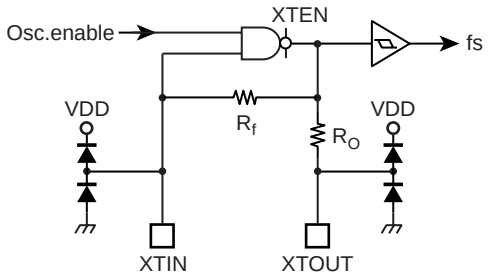
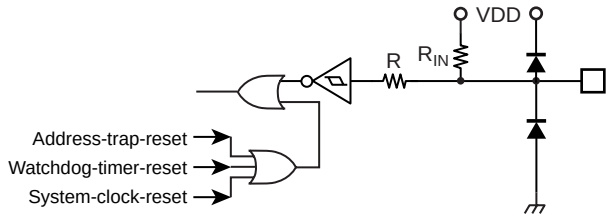
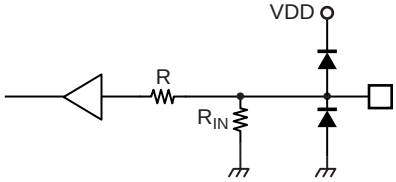
Table 18-1 Release level (edge) of STOP mode

| Pin name | Release level (edge) | |
|---|---|---|
| | SYSCR1<RELM>="1" (Note2) | SYSCR1<RELM>="0" |
| $\overline{\text{STOP}}$ | "H" level | Rising edge |
| STOP0 | "L" level | Don’t use (Note1) |
| STOP1 | "L" level | Don’t use (Note1) |
| STOP2 | "L" level | Don’t use (Note1) |
| STOP3 | "L" level | Don’t use (Note1) |

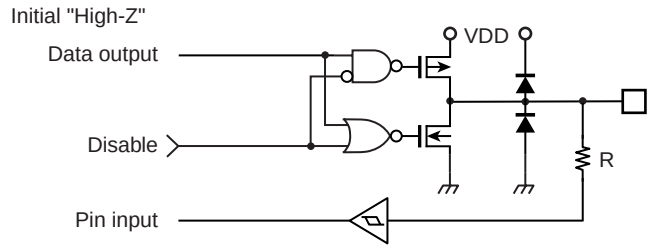
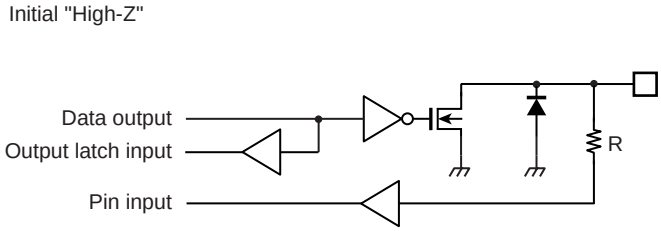
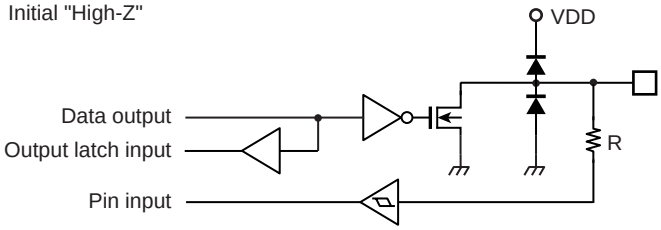
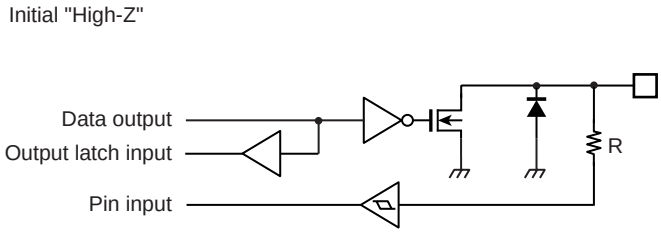
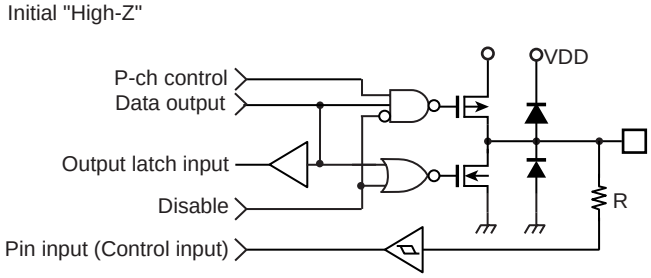
# 19. Input/Output Circuit

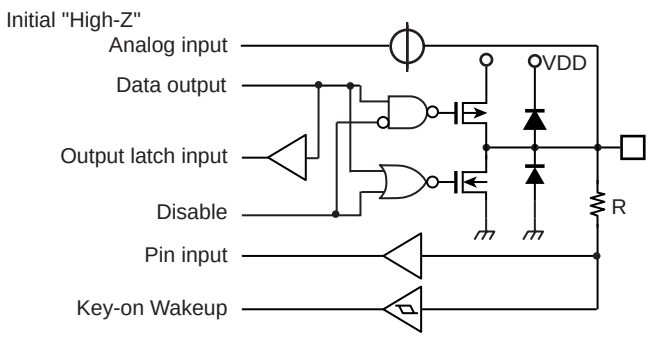
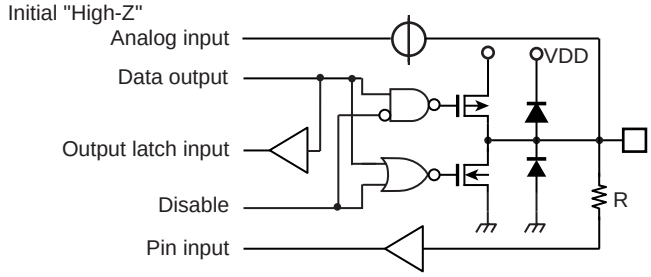
## 19.1 Control pins

The input/output circuitries of the TMP86CS49UG control pins are shown below.

| Control Pin | I/O | Input/Output Circuitry | Remarks |
|---|---|---|---|
| XIN<br>XOUT | Input<br>Output |  | Resonator connecting pins<br>(high frequency)<br>$R_f$ = 1.2 MΩ (typ.)<br>$R_O$ =0.5 kΩ (typ.) |
| XTIN<br>XTOUT | Input<br>Output |  | Resonator connecting pins<br>(Low frequency)<br>$R_f$ = 6 MΩ (typ.)<br>$R_O$ = 220 kΩ (typ.) |
| $\overline{\text{RESET}}$ | Input |  | Hysteresis input<br>Pull-up resistor<br>$R_{IN}$ = 220 kΩ (typ.)<br>R = 100 Ω (typ.) |
| TEST | Input |  | Pull-down resistor<br>$R_{IN}$ = 70 kΩ (typ.)<br>R = 100 Ω (typ.) |

# 19.2 Input/Output Ports

| Port | I/O | Input/Output Circuitry | Remarks |
|---|---|---|---|
| P1 | I/O |  | Tri-state I/O<br>Hysteresis input<br>R = 100 Ω (typ.) |
| P3 | I/O |  | Sink open drain output<br>High current output<br>R = 100 Ω (typ.) |
| P2 | I/O |  | Sink open drain output<br>Hysteresis input<br>R = 100 Ω (typ.) |
| P5 | I/O |  | Sink open drain output<br>High current output<br>Hysteresis input<br>R = 100 Ω (typ.) |
| P0<br>P4 | I/O |  | Sink open drain output or<br>C-MOS output<br>Hysteresis input<br>R = 100 Ω (typ.) |

| Port | I/O | Input/Output Circuitry | Remarks |
|---|---|---|---|
| P67<br>P66<br>P65<br>P64 | I/O | Initial "High-Z"<br> | Tri-state I/O<br>R = 100 Ω (typ.) |
| P63<br>P62<br>P61<br>P60<br>P7 | I/O | Initial "High-Z"<br> | Tri-state I/O<br>R = 100 Ω (typ.) |

# 20. Electrical Characteristics

## 20.1 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 6.5 | 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, P4, P6, P7 ports | -1.8 | mA |
| | $I_{OUT2}$ | P0, P1, P2, P4, P6, P7 ports | 3.2 | |
| | $I_{OUT3}$ | P3, P5 ports | 30 | |
| Output current (Total) | $\Sigma I_{OUT1}$ | P0, P1, P2, P4, P6, P7 ports | 60 | |
| | $\Sigma I_{OUT2}$ | P3, P5 ports | 80 | |
| Power dissipation [Topr = 85 °C] | $P_D$ | | 250 | mW |
| Soldering temperature (time) | Tsld | | 260 (10 s) | °C |
| Storage temperature | Tstg | | -55 to 125 | |
| Operating temperature | Topr | | -40 to 85 | |

## 20.2 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.

($V_{SS}$ = 0 V, Topr = -40 to 85°C)

| Parameter | Symbol | Pins | Ratings | | Min | Max | Unit |
|---|---|---|---|---|---|---|---|
| Supply voltage | $V_{DD}$ | | fc = 16 MHz | NORMAL1, 2 modes<br>IDLE0, 1, 2 modes | 4.5 | 5.5 | V |
| | | | fc = 8 MHz | NORMAL1, 2 modes<br>IDLE0, 1, 2 modes | 2.7 | | |
| | | | fc = 4.2 MHz | NORMAL1, 2 modes<br>IDLE0, 1, 2 modes | 1.8<br>(Note) | | |
| | | | fs = 32.768 KHz | SLOW1, 2 modes<br>SLEEP0, 1, 2 modes | | | |
| | | | STOP mode | | | | |
| Input high level | $V_{IH1}$ | Except hysteresis input | $V_{DD} \geq 4.5$ V | | $V_{DD} \times 0.70$ | $V_{DD}$ | |
| | $V_{IH2}$ | Hysteresis input | | | $V_{DD} \times 0.75$ | | |
| | $V_{IH3}$ | | $V_{DD} < 4.5$ V | | $V_{DD} \times 0.90$ | | |
| Input low level | $V_{IL1}$ | Except hysteresis input | $V_{DD} \geq 4.5$ V | | 0 | $V_{DD} \times 0.30$ | |
| | $V_{IL2}$ | Hysteresis input | | | | $V_{DD} \times 0.25$ | |
| | $V_{IL3}$ | | $V_{DD} < 4.5$ V | | | $V_{DD} \times 0.10$ | |
| Clock frequency | fc | XIN, XOUT | $V_{DD}$ = 1.8 to 5.5V (Note) | | 1.0 | 4.2 | MHz |
| | | | $V_{DD}$ = 2.7 to 5.5V | | | 8.0 | |
| | | | $V_{DD}$ = 4.5 to 5.5 V | | | 16.0 | |
| | fs | XTIN, XTOUT | | | 30.0 | 34.0 | kHz |

Note: When the supply voltage $V_{DD}$ is less than 2.0 V, the operating temperature Topr must be in a range of -20°C to 85°C.

## 20.3 DC Characteristics

($V_{SS}$ = 0 V, Topr = -40 to 85 °C)

| Parameter | Symbol | Pins | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|---|
| Hysteresis voltage | $V_{HS}$ | Hysteresis input | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V/0 V | – | 0.9 | – | V |
| Input current | $I_{IN1}$ | TEST | | – | – | ±2 | μA |
| | $I_{IN2}$ | Sink open drain, tri–state port | | | | | |
| | $I_{IN3}$ | $\overline{RESET}$, $\overline{STOP}$ | | | | | |
| Input resistance | $R_{IN1}$ | TEST pull–down | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V | – | 70 | – | kΩ |
| | $R_{IN2}$ | $\overline{RESET}$ pull–up | $V_{DD}$ = 5.5 V, $V_{IN}$ = 0 V | 100 | 220 | 450 | |
| Output leakage current | $I_{LO1}$ | Sink open drain port | $V_{DD}$ = 5.5 V, $V_{OUT}$ = 5.5 V | – | – | 2 | μA |
| | $I_{LO2}$ | Tri–state port | $V_{DD}$ = 5.5 V, $V_{OUT}$ = 5.5 V/0 V | – | – | ±2 | |
| Output high voltage | $V_{OH}$ | Tri–state port | $V_{DD}$ = 4.5 V, $I_{OH}$ = -0.7 mA | 4.1 | – | – | V |
| Output low voltage | $V_{OL}$ | Except XOUT, P3, P5 | $V_{DD}$ = 4.5 V, $I_{OL}$ = 1.6 mA | – | – | 0.4 | |
| Output low curren | $I_{OL}$ | High current port (P3, P5 Port) | $V_{DD}$ = 4.5 V, $V_{OL}$ = 1.0 V | – | 20 | – | mA |
| Supply current in NORMAL1, 2 modes | $I_{DD}$ | | $V_{DD}$ = 5.5 V<br>$V_{IN}$ = 5.3 V/0.2 V<br>fc = 16 MHz<br>fs = 32.768 kHz | – | 8.5 | 11.5 | mA |
| Supply current in IDLE 0, 1, 2 modes | | | | – | 6.0 | 8.5 | |
| Supply current in SLOW1 mode | | | $V_{DD}$ = 3.0 V<br>$V_{IN}$ = 2.8 V/0.2 V<br>fs = 32.768 kHz | – | 8.5 | 20.0 | μA |
| Supply current in SLEEP1 mode | | | | – | 6.1 | 15.0 | |
| Supply current in SLEEP0 mode | | | | – | 5.0 | 11.0 | |
| Supply current in STOP mode | | | $V_{DD}$ = 5.5 V<br>$V_{IN}$ = 5.3 V/0.2 V | – | 0.5 | 10 | |

Note 1: Typical values show those at Topr = 25°C and $V_{DD}$ = 5 V.

Note 2: Input current ($I_{IN1}$, $I_{IN3}$): The current through pull-up or pull-down resistor is not included.

Note 3: $I_{DD}$ does not include $I_{REF}$.

Note 4: The supply currents of SLOW2 and SLEEP2 modes are equivalent to those of IDLE0, IDLE1 and IDLE2 modes.

# 20.4 AD Characteristics

($V_{SS}$ = 0.0 V, 4.5 V ≤ $V_{DD}$ ≤ 5.5 V, Topr = -40 to 85 °C)

| Paramete | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Analog reference voltage | $V_{AREF}$ | | $A_{VDD}$ - 1.0 | – | $A_{VDD}$ | V |
| Power supply voltage of analog control circuit | $A_{VDD}$ | | $V_{DD}$ | | | |
| Analog reference voltage range (Note 4) | Δ $V_{AREF}$ | | 3.5 | – | – | |
| Analog input voltage | $V_{AIN}$ | | $V_{SS}$ | – | $V_{AREF}$ | |
| Power supply current of analog reference voltage | $I_{REF}$ | $V_{DD}$ = $A_{VDD}$ = $V_{AREF}$ = 5.5 V<br>$V_{SS}$ = 0.0 V | – | 0.6 | 1.0 | mA |
| Non linearity error | | $V_{DD}$ = $A_{VDD}$ = 5.0 V,<br>$V_{SS}$ = 0.0 V<br>$V_{AREF}$ = 5.0 V | – | – | ±2 | LSB |
| Zero point error | | | – | – | ±2 | |
| Full scale error | | | – | – | ±2 | |
| Total error | | | – | – | ±2 | |

($V_{SS}$ = 0 V, 2.7 V ≤ $V_{DD}$ < 4.5 V, Topr = -40 to 85°C)

| Parameter | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Analog reference voltage | $V_{AREF}$ | | $A_{VDD}$ - 1.0 | – | $A_{VDD}$ | V |
| Power supply voltage of analog control circuit | $A_{VDD}$ | | $V_{DD}$ | | | |
| Analog reference voltage range (Note 4) | Δ $V_{AREF}$ | | 2.5 | – | – | |
| Analog input voltage | $V_{AIN}$ | | $V_{SS}$ | – | $V_{AREF}$ | |
| Power supply current of analog reference voltage | $I_{REF}$ | $V_{DD}$ = $A_{VDD}$ = $V_{AREF}$ = 4.5 V<br>$V_{SS}$ = 0.0 V | – | 0.5 | 0.8 | mA |
| Non linearity error | | $V_{DD}$ = $A_{VDD}$ = 2.7 V<br>$V_{SS}$ = 0.0 V<br>$V_{AREF}$ = 2.7 V | – | – | ±2 | LSB |
| Zero point error | | | – | – | ±2 | |
| Full scale error | | | – | – | ±2 | |
| Total error | | | – | – | ±2 | |

($V_{SS}$ = 0 V, 2.0 V ≤ $V_{DD}$ < 2.7 V, Topr = -40 to 85°C) (Note6)

($V_{SS}$ = 0 V, 1.8 V ≤ $V_{DD}$ < 2.0 V, Topr = -10 to 85°C) (Note6)

| Parameter | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Analog reference voltage | $V_{AREF}$ | | $A_{VDD}$ - 0.9 | – | $A_{VDD}$ | V |
| Power supply voltage of analog control circuit | $A_{VDD}$ | | $V_{DD}$ | | | |
| Analog reference voltage range (Note 4) | Δ $V_{AREF}$ | 1.8 V ≤ VDD < 2.0 V | 1.8 | – | – | |
| | | 2.0 V ≤ VDD < 2.7 V | 2.0 | – | – | |
| Analog input voltage | $V_{AIN}$ | | $V_{SS}$ | – | $V_{AREF}$ | |
| Power supply current of analog reference voltage | $I_{REF}$ | $V_{DD}$ = $A_{VDD}$ = $V_{AREF}$ =2.7 V<br>$V_{SS}$ = 0.0 V | – | 0.3 | 0.5 | mA |
| Non linearity error | | $V_{DD}$ = $A_{VDD}$ = 1.8 V<br>$V_{SS}$ = 0.0 V<br>$V_{AREF}$ = 1.8 V | – | – | ±4 | LSB |
| Zero point error | | | – | – | ±4 | |
| Full scale error | | | – | – | ±4 | |
| Total error | | | – | – | ±4 | |

Note 1: The total error includes all errors except a quanitization 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 $V_{AREF}$ 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.

Note 4: Analog reference voltage range: $\Delta V_{AREF} = V_{AREF} - V_{SS}$

Note 5: When AD converter is not used, fix the AVDD and VAREF pin on the $V_{DD}$ level.

Note 6: When AD is used with VDD < 2.0 V, the guaranteed temperature range varies with the operating voltage.

## 20.5 AC Characteristics

($V_{SS}$ = 0 V, 4.5 V ≤ $V_{DD}$ ≤ 5.5 V, Topr = -40 to 85°C)

| Parameter | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Machine cycle time | tcy | 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 | $t_{WCH}$ | For external clock operation (XIN input) fc = 16 MHz | – | 31.25 | – | ns |
| Low-level clock pulse width | $t_{WCL}$ | | | | | |
| High-level clock pulse width | $t_{WSH}$ | For external clock operation (XTIN input) fs = 32.768 kHz | – | 15.26 | – | μs |
| Low-level clock pulse width | $t_{WSL}$ | | | | | |

($V_{SS}$ = 0 V, 2.7 V ≤ $V_{DD}$ < 4.5 V, Topr = -40 to 85°C)

| Paramete | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Machine cycle time | $t_{cy}$ | NORMAL1, 2 modes | 0.5 | – | 4 | μs |
| | | IDLE0, 1, 2 modes | | | | |
| | | SLOW1, 2 modes | 117.6 | – | 133.3 | |
| | | SLEEP0, 1, 2 modes | | | | |
| High-level clock pulse width | $t_{WCH}$ | For external clock operation (XIN input) fc = 8 MHz | – | 62.5 | – | ns |
| Low-level clock pulse width | $t_{WCL}$ | | | | | |
| High-level clock pulse width | $t_{WSH}$ | For external clock operation (XTIN input) fs = 32.768 kHz | – | 15.26 | – | μs |
| Low-level clock pulse width | $t_{WSL}$ | | | | | |

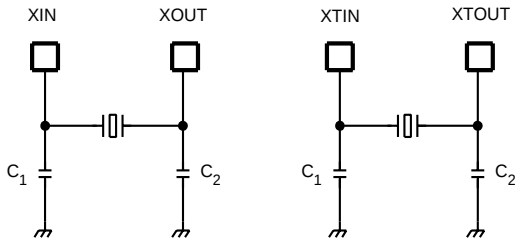
($V_{SS}$ = 0 V, 2.0 V ≤ $V_{DD}$ < 2.7 V, Topr = -40 to 85°C) (Note1)

($V_{SS}$ = 0 V, 1.8 V ≤ $V_{DD}$ < 2.0 V, Topr = -20 to 85°C) (Note1)

| Paramete | Symbol | Condition | Min | Typ. | Max | Unit |
|---|---|---|---|---|---|---|
| Machine cycle time | $t_{cy}$ | NORMAL1, 2 modes | 0.95 | – | 4 | μs |
| | | IDLE0, 1, 2 modes | | | | |
| | | SLOW1, 2 modes | 117.6 | – | 133.3 | |
| | | SLEEP0, 1, 2 modes | | | | |
| High-level clock pulse width | $t_{WCH}$ | For external clock operation (XIN input) fc = 4.2 MHz | – | 119.05 | – | ns |
| Low-level clock pulse width | $t_{WCL}$ | | | | | |
| High-level clock pulse width | $t_{WSH}$ | For external clock operation (XTIN input) fs = 32.768 kHz | – | 15.26 | – | μs |
| Low-level clock pulse width | $t_{WSL}$ | | | | | |

Note 1: When the supply voltage $V_{DD}$ is less than 2.0 V, the operating temperature Topr must be in a range of -20°C to 85°C.

## 20.6 Recommended Oscillating Conditions




(1) High-frequency Oscillation (2) Low-frequency Oscillation

Note 1: A quartz resonator can be used for high-frequency oscillation only when VDD is 2.7 V or above. If VDD is below 2.7 V, use a ceramic resonator.

Note 2: 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 3: 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 4: 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/

## 20.7 Handling Precaution

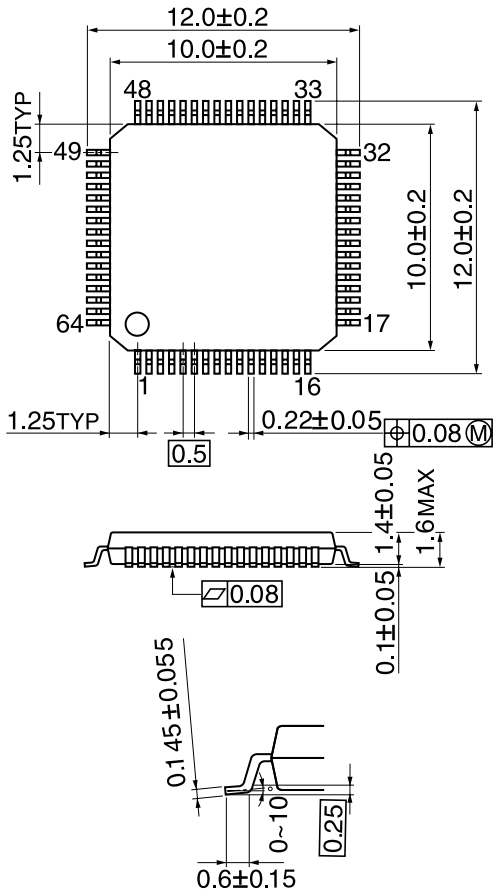
- The solderability test conditions for lead-free products (indicated by the suffix G in product name) are shown below.
  1. When using the Sn-37Pb solder bath
     Solder bath temperature = 230 °C
     Dipping time = 5 seconds
     Number of times = once
     R-type flux used
  2. When using the Sn-3.0Ag-0.5Cu solder bath
     Solder bath temperature = 245 °C
     Dipping time = 5 seconds
     Number of times = once
     R-type flux used

  Note: The pass criteron of the above test is as follows:
  Solderability rate until forming ≥ 95 %
- When using the device (oscillator) in places exposed to high electric fields such as cathode-ray tubes, we recommend electrically shielding the package in order to maintain normal operating condition.

# 21. Package Dimensions

LQFP64-P-1010-0.50D Rev 01

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.