TOSHIBA

8 Bit Microcontroller
TLCS-870/C Series

TMP86CH06AUG

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

Differences between the TMP86CH06AUG and the TMP86CH06U

The following table shows the function and specification differences between the TMP86CH06AUG and its previous version TMP86CH06U/UG. Please be careful about these differences if you have switched from the TMP86CH06U to the TMP86CH06AUG. Also, there is correction in data sheet of the TMP86CH06AUG.

Differences in characteristics

Item	TMP86CH06AUG	TMP86CH06U
CGCR register	Not disclosed (Reserved)	CGCR register 0030H
Clock gear	Not to be set Writing is prohibited, and reading yields an undefined value.	To be set CGCR<FCGCK>
DV1CK	Do not set.	To be set CGCR<DV1CK>
DVCK	Do not set.	To be set CGCR<DVCK>
Output level (ALE pin only)	High 2.2 V Low Vdd/2 CL = 50 pF	High 2.2 V Low 0.8V CL = 50 pF
AC characteristic (tACH)	1.5 t –35 ns (min) 58 ns	1.5 t –25 ns (min) 11 ns
AC characteristic (tCA)	0.5 t –32 ns (min) 0 ns	0.5 t –20 ns (min) 11 ns
Operating temperature (Topr)	1.8 V ≤ V_{DD} < 2.0 V : Topr = - 20 to 85 °C 2.0 V ≤ V_{DD} ≤ 5.5 V : Topr = - 40 to 85 °C	1.8 V ≤ V_{DD} ≤ 5.5 V : Topr = - 40 to 85 °C

Correction in Data sheet

Item	TMP86CH06AUG	TMP86CH06U/UG
Guaranteed VDD in external bus mode	Added "VDD = 4.5 V to 5.5 V" to AC Characteristics	Lack of condition

The following name of control registers or bits is different from TMP86CH06U. However, the contents of them are the same as TMP86CH06U. Therefore, the same software can be used for TMP86CH06AUG, if you have switched from the TMP86CH06U to the TMP86CH06AUG.

Name of control registers or bits

Name of register or control bit	TMP86CH06AUG	TMP86CH06U
Bit 7 in INTSEL	IL8ER	INTRX0ER
Bit 3 in INTSEL	IL12ER	INTOC0ER
Bit 2 in INTSEL	IL13ER	INTRX1ER
Bit 1 in INTSEL	IL14ER	INTTXDER
Bit 0 in INTSEL	IL15ER	INT5ER
UART0 control register 1	UART0CR1	UART0CRA
UART0 control register 2	UART0CR2	UART0CRB
UART1 control register 1	UART1CR1	UART1CRA
UART1 control register 2	UART1CR2	UART1CRB
SIO data buffer	SIOBRx (x=0 to 7)	SIODBR

Revision History

Date	Revision	
2006/2/23	1	First Release
2006/8/30	2	Contents Revised

Table of Contents

Differences between the TMP86CH06AUG and the TMP86CH06U

TMP86CH06AUG

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 7
2.1.1 Memory Address Map 7
2.1.2 Program Memory (MaskROM) 8
2.1.3 Data Memory (RAM) 8
2.1.4 External Memory Interfaces 8
2.1.4.1 Controlling
2.1.4.2 External Bus Controller
2.1.4.3 Wait Controller
2.1.5 System Clock Controller 12
2.1.5.1 Clock Generator
2.1.5.2 Timing Generator
2.1.5.3 Operation Mode Control Circuit
2.1.5.4 Operating Mode Control
2.1.6 Reset Circuit 34
2.1.6.1 External Reset Input
2.1.6.2 Address trap reset
2.1.6.3 Watchdog timer reset
2.1.6.4 System clock reset

3. Interrupt Control Circuit

3.1 Interrupt latches (IL15 to IL2) 37
3.2 Interrupt enable register (EIR) 38
3.2.1 Interrupt master enable flag (IMF) 38
3.2.2 Individual interrupt enable flags (EF15 to EF4) 38
3.3 Interrupt Source Selector (INTSEL) 41
3.4 Interrupt Sequence 41
3.4.1 Interrupt acceptance processing is packaged as follows. 41
3.4.2 Saving/restoring general-purpose registers 42
3.4.2.1 Using PUSH and POP instructions
3.4.2.2 Using data transfer instructions
3.4.3 Interrupt return 44
3.5 Software Interrupt (INTSW) 45
3.5.1 Address error detection 45
3.5.2 Debugging 45

3.6 Undefined Instruction Interrupt (INTUNDEF) 45
3.7 Address Trap Interrupt (INTATRAP) 45
3.8 External Interrupts 45

4. Special Function Register (SFR)

4.1 SFR 49

5. I/O Ports

5.1 Port P0 52
5.2 Port P1 53
5.3 Port P2 56
5.4 Port P3 57
5.5 Port P4 58

6. Watchdog Timer (WDT)

6.1 Watchdog Timer Configuration 61
6.2 Watchdog Timer Control 62
6.2.1 Malfunction Detection Methods Using the Watchdog Timer 62
6.2.2 Watchdog Timer Enable 63
6.2.3 Watchdog Timer Disable 64
6.2.4 Watchdog Timer Interrupt (INTWDT) 64
6.2.5 Watchdog Timer Reset 65
6.3 Address Trap 66
6.3.1 Selection of Address Trap in Internal RAM (ATAS) 66
6.3.2 Selection of Operation at Address Trap (ATOUT) 66
6.3.3 Address Trap Interrupt (INTATRAP) 66
6.3.4 Address Trap Reset 67

7. Time Base Timer (TBT)

7.1 Time Base Timer 69
7.1.1 Configuration 69
7.1.2 Control 69
7.1.3 Function 70
7.2 Divider Output (DVO) 71
7.2.1 Configuration 71
7.2.2 Control 71

8. Extended Timer-Counter (ETC0)

8.1 Configuration 73
8.2 Controlling 74
8.3 Source Clock 76
8.3.1 Free Running Timer Mode 76
8.3.2 Event-counter Mode 77
8.4 Capturing input, Output comparing 79
8.4.1 Capturing input 79

8.4.2 Output Comparing 81
8.5 Interrupting 84

9. 8-Bit TimerCounter (TC0, TC1)

9.1 Configuration 85
9.2 TimerCounter Control 86
9.3 Function 92
9.3.1 8-Bit Timer Mode (TC0 and 1) 92
9.3.2 8-Bit Event Counter Mode (TC0, 1) 93
9.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC0, 1) 93
9.3.4 8-Bit Pulse Width Modulation (PWM) Output Mode (TC0, 1) 96
9.3.5 16-Bit Timer Mode (TC0 and 1) 98
9.3.6 16-Bit Event Counter Mode (TC0 and 1) 99
9.3.7 16-Bit Pulse Width Modulation (PWM) Output Mode (TC0 and 1) 99
9.3.8 16-Bit Programmable Pulse Generate (PPG) Output Mode (TC0 and 1) 102
9.3.9 Warm-Up Counter Mode 104
9.3.9.1 Low-Frequency Warm-up Counter Mode (NORMAL1 → NORMAL2 → SLOW2 → SLOW1)
9.3.9.2 High-Frequency Warm-Up Counter Mode (SLOW1 → SLOW2 → NORMAL2 → NORMAL1)

10. Synchronous Serial Interface (SIO)

10.1 Configuration 107
10.2 Control 108
10.3 Serial clock 109
10.3.1 Clock source 109
10.3.1.1 Internal clock
10.3.1.2 External clock
10.3.2 Shift edge 111
10.3.2.1 Leading edge
10.3.2.2 Trailing edge
10.4 Number of bits to transfer 111
10.5 Number of words to transfer 111
10.6 Transfer Mode 112
10.6.1 4-bit and 8-bit transfer modes 112
10.6.2 4-bit and 8-bit receive modes 114
10.6.3 8-bit transfer / receive mode 115

11. Asynchronous Serial interface (UART0)

11.1 Configuration 117
11.2 Control 118
11.3 Transfer Data Format 120
11.4 Transfer Rate 121
11.5 Data Sampling Method 121
11.6 STOP Bit Length 122
11.7 Parity 122
11.8 Transmit/Receive Operation 122
11.8.1 Data Transmit Operation 122
11.8.2 Data Receive Operation 122
11.9 Status Flag 123
11.9.1 Parity Error 123
11.9.2 Framing Error 123

11.9.3 Overrun Error 123
11.9.4 Receive Data Buffer Full 124
11.9.5 Transmit Data Buffer Empty 124
11.9.6 Transmit End Flag 125

12. Asynchronous Serial interface (UART1)

12.1 Configuration 127
12.2 Control 128
12.3 Transfer Data Format 130
12.4 Transfer Rate 131
12.5 Data Sampling Method 131
12.6 STOP Bit Length 132
12.7 Parity 132
12.8 Transmit/Receive Operation 132
12.8.1 Data Transmit Operation 132
12.8.2 Data Receive Operation 132
12.9 Status Flag 133
12.9.1 Parity Error 133
12.9.2 Framing Error 133
12.9.3 Overrun Error 133
12.9.4 Receive Data Buffer Full 134
12.9.5 Transmit Data Buffer Empty 134
12.9.6 Transmit End Flag 135

13. Input/Output Circuitry

13.1 Control Pins 137
13.2 Input/Output Ports 138

14. Electrical Characteristics

14.1 Absolute Maximum Rating 139
14.2 Recommended Operating Conditions 140
14.3 DC Characteristics 141
14.4 AC Characteristics 142
14.4.1 CLOCK 142
14.4.2 External Memory Interface (Multiplexed Bus) 142
14.5 Recommended Oscillating Conditions 144
Note 3: 144
14.6 Handling Precaution 144

15. Package Dimension

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

TMP86CH06AUG

Product No.	ROM (MaskROM)	RAM	Package	OTP MCU	Emulation Chip
TMP86CH06AUG	16384 bytes	512 bytes	P-LQFP44-1010-0.80B	TMP86PH06UG	TMP86C906XB

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. 21interrupt sources (External : 6 Internal : 15)
3. Input / Output ports (35 pins)

   Large current output: 8pins (Typ. 20mA), LED direct drive
4. Watchdog Timer
5. Prescaler
   - Time base timer
   - Divider output function
6. Extended 16bit Timer-Counter
   - External input

     Compare output

     Capture input
7. 8-bit timer counter : 2 ch
   - Timer, Event counter, Programmable divider output (PDO),

     Pulse width modulation (PWM) output,

     Programmable pulse generation (PPG) modes

8. 8-bit SIO/UART0: 1 ch
9. 8-bit UART : 1 ch
10. Clock operation

    Single clock mode

    Dual clock mode
11. 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.
12. 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

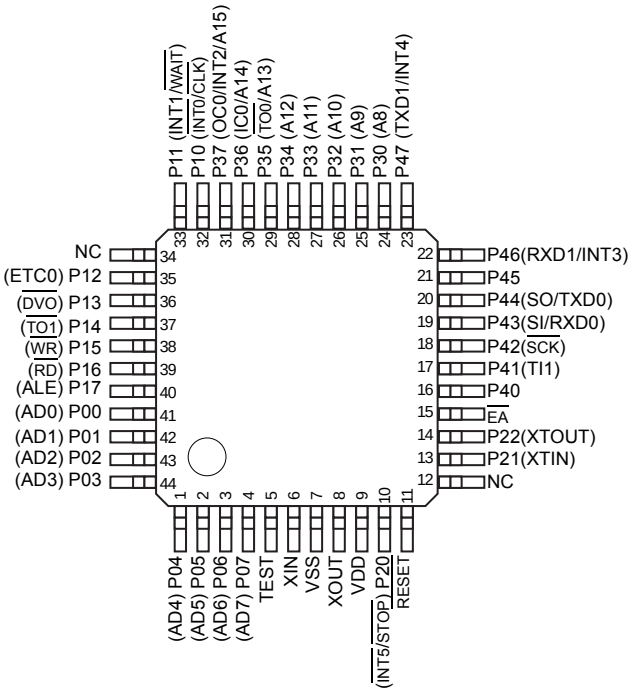


Figure 1-1 Pin Assignment

1.3 Block Diagram

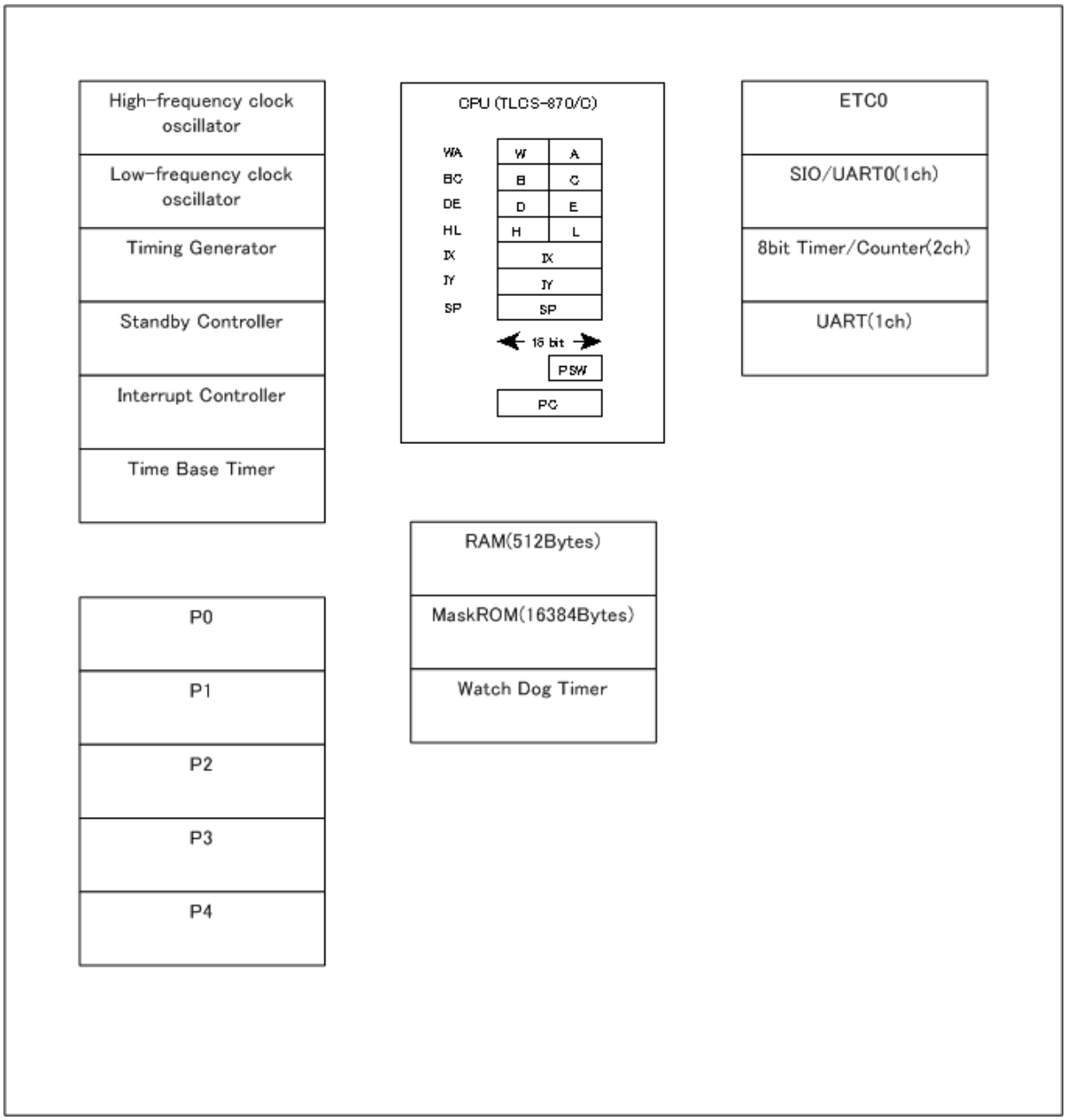


Figure 1-2 Block Diagram

1.4 Pin Names and Functions

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

Pin Name	Pin Number	Input/Output	Functions
P07 AD7	4	IO IO	PORT07 Address/Data bus 7
P06 AD6	3	IO IO	PORT06 Address/Data bus 6
P05 AD5	2	IO IO	PORT05 Address/Data bus 5
P04 AD4	1	IO I	PORT04 Address/Data bus 4
P03 AD3	44	IO IO	PORT03 Address/Data bus 3
P02 AD2	43	IO IO	PORT02 Address/Data bus 2
P01 AD1	42	IO IO	PORT01 Address/Data bus 1
P00 AD0	41	IO IO	PORT00 Address/Data bus 0
P17 ALE	40	IO O	PORT17 Address Latch Enable(The negative edge of ALE supplies an address latch timing on AD0 to AD7 for External Memory)
P16 $\overline{RD}$	39	IO O	PORT16 Read(Generates strobe signal to read data from External Memory)
P15 $\overline{WR}$	38	IO O	PORT15 Write(Generates strobe signal to write data on External Memory)
P14 $\overline{TO1}$	37	IO O	PORT14 TO1 output
P13 $\overline{DVO}$	36	IO O	PORT13 Divider Output
P12 ETC0	35	IO I	PORT12 Extended Timer-Counter
P11 INT1 $\overline{WAIT}$	33	IO I I	PORT11 External interrupt 1 input Wait(Wait request from external Memory)
P10 $\overline{INT0}$ $\overline{CLK}$	32	IO I O	PORT10 External interrupt 0 input Clock(Clock output)
P22 XTOUT	14	IO O	PORT22 Resonator connecting pins(32.768kHz) for inputting external clock
P21 XTIN	13	IO I	PORT21 Resonator connecting pins(32.768kHz) for inputting external clock
P20 $\overline{STOP}$ $\overline{INT5}$	10	IO I I	PORT20 STOP mode release signal input External interrupt 5 input

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

Pin Name	Pin Number	Input/Output	Functions
P37 OC0 INT2 A15	31	IO O I I	PORT37 Output Compare 0 for Extended Timere External interrupt 2 input Address bus 15
P36 IC0 A14	30	IO I I	PORT36 Capture Input 0 for Extended Timer Address bus 14
P35 $\overline{TO0}$ A13	29	IO O I	PORT35 TO0 output Address bus 13
P34 A12	28	IO I	PORT34 Address bus 12
P33 A11	27	IO I	PORT33 Address bus 11
P32 A10	26	IO I	PORT32 Address bus 10
P31 A9	25	IO I	PORT31 Address bus 9
P30 A8	24	IO I	PORT30 Address bus 8
P47 TXD1 INT4	23	IO O I	PORT47 UART data output 1 External interrupt 4 input
P46 RXD1 INT3	22	IO I I	PORT46 UART data input 1 External interrupt 3 input
P45	21	IO	PORT45
P44 SO TXD0	20	IO O O	PORT44 Serial Data Output UART data output 0
P43 SI RXD0	19	IO I I	PORT43 Serial Data Input UART data input 0
P42 $\overline{SCK}$	18	IO IO	PORT42 Serial Clock I/O
P41 TI1	17	IO I	PORT41 TI1 input
P40	16	IO	PORT40
XIN	6	I	Resonator connecting pins for high-frequency clock
XOUT	8	O	Resonator connecting pins for high-frequency clock
$\overline{RESET}$	11	IO	ÉäÉZÉbÉgì¸èoóÕ
TEST	5	I	Test pin for out-going test (Be fixed to low)
VDD	11	IO	+5V
VSS	7	I	0(GND)
NC	34	I	Non Connection

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, the external memory interface, and the reset circuit.

2.1.1 Memory Address Map

The TMP86CH06AUG memory consists of 3 blocks: ROM, RAM, and SFR (Special function register). They are all mapped in 64-Kbyte address space. Figure 2-1 shows the TMP86CH06AUG memory address map. The general-purpose registers are not assigned to the RAM address space.

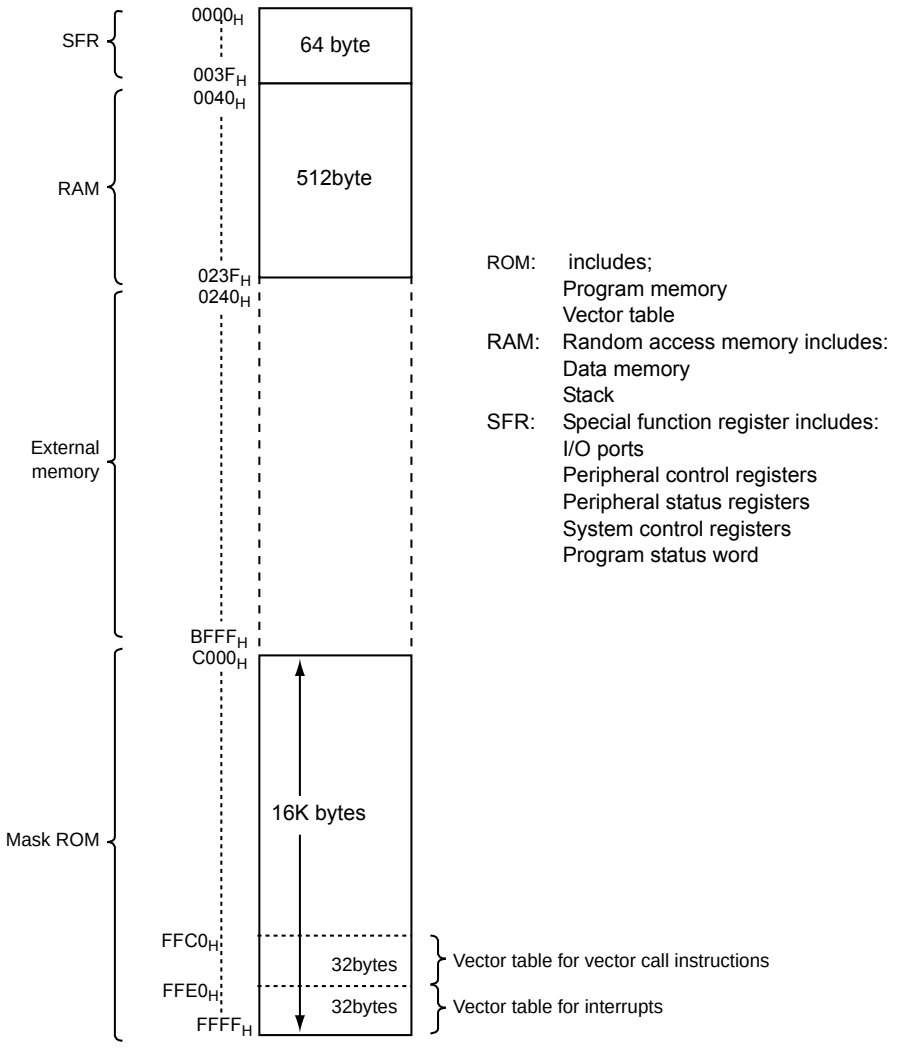


Figure 2-1 Memory Address Map

2.1.2 Program Memory (MaskROM)

The TMP86CH06AUG has a 16 K × 8 bits (Address C000H to FFFFH) of program memory (Mask ROM). Of the 64-Kbyte memory space, the space excluding the SFR and internal RAM areas can be used as external program memory space. When the $\overline{EA}$ pin is connected to GND, this space becomes external memory. A program code placed on the internal RAM can be excutable when a certain procedure is executed (See " 2.1.6.2 Address trap reset ").

2.1.3 Data Memory (RAM)

Data memory consists of internal data memory (Internal ROM or RAM). The TMP86CH06AUG has 512 bytes (Address 0040H to 023FH) of internal RAM. The first 192 bytes (0040H to 00FFH) of the internal RAM are located in the direct area; instructions with shorten operations are available against such an area.

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

Example :Clears RAM to “00H”.

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

2.1.4 External Memory Interfaces

The TMP86CH06AUG can be connected with the external memory through address bus, a data bus, and the control bus. It is available up to 64 Kbytes of data memory area except the area where the internal RAM and SFR are located. Data bus and the lower bits of address bus are multiplexed.

- The operation mode is selected from the following three by setting $\overline{EA}$ terminal and control register EXPCR.

  1) Single chip mode: ($\overline{EA}$ = 1, EXPCR<RDOE, WROE> = “0, 0”)

  The TMP86CH06AUGoperates within internal memory (ROM, RAM and SFR).

  Each terminal for connecting external memory is available for input/output port.

  2) Expansion mode: ($\overline{EA}$ = 1, EXPCR<either or both RDOE, WROE> = “1”)

  After releasing reset, the TMP86CH06AUG is starts to operate within internal memory (ROM, RAM and SFR).

  If either or both RDOE and WROE are enabled, terminals for external memory start to function as ALE and AD7 to 0.

  3) ROM-less mode: ($\overline{EA}$ = 0)

  After releasing reset, the TMP86CH06AUG operates with external memory instead of internal ROM. The internal RAM and SFR are still be used.

- Wait Control

  The operation mode is selected from the following three: Non wait, 1-cycle wait or (1+n)-cycle wait mode.

- Internal ROM security function

The TMP86CH06AUG can keep CPU from reading the data on internal ROM, while its read instruction is located in external memory.

Note: The transfer destination data of the vector table (located from FFC0H to FFFFH) is not protected.

2.1.4.1 Controlling

The external memory interfaces are controlled by expansion control register (EXPCR) and wait control register (WAITCR).

EXPCR directly manages $\overline{RD}$, $\overline{WR}$ and address bus output. And, assigning terminals to $\overline{RD}$ or $\overline{WR}$ subsequently provides AD7 to 0 and ALE. In order to utilize external memory, port directions whether to be input or output is determined regardless of each bit on control register. After reset, EXPCR is a register which can be written only once. Therefore, the second and the following write operation cannot modify the data on EXPCR.

External Access Control

EXPCR (0031H)

7	6	5	4	3	2	1	0	
ABUSEN			–	RDOE	WROE	–	–	(Initial value: 111* 11**) $\overline{EA}$ ← "0" (Initial value: 000* 00**) $\overline{EA}$ ← "1"

ABUSEN	Upper bits on address bus output enable [A15 to A8]	000: input/output port 001: A9 to A8 output 010: A10 to A8 output 011: A11 to A8 output 100: A12 to A8 output 101: A13 to A8 output 110: A14 to A8 output 111: A15 to A8 output Other terminals on P3 port keep their input output mode.	WR 1 time
RDOE	$\overline{RD}$ strobe output enable	0: $\overline{RD}$ strobe output Disable 1: $\overline{RD}$ strobe output Enable	Write only
WROE	WR strobe output enable	0: $\overline{WR}$ strobe output Disable 1: $\overline{WR}$ strobe output Enable	Write only

Note 1: Once either RDOE or WROE or both are enabled to "1", the following terminals are utilized for external memory interface; therefore they cannot be used for general-purpose input/output ports.
- AD7 to 0 (P07 to P00)
- ALE (P17)
- $\overline{RD}$ (P16: if RDOE="1") or $\overline{WR}$ (P15:if WROE = "1")
- CLK (P10)

Note 2: EXPCR is a write only register and must not be used with any of the read-modify-write instructions.

WAITCR manages security for internal ROM, number of waiting cycles and waiting clock output.

When control waiting cycles, WAITCR selects waiting mode from three: no-wait, 1-state wait or (1+n)-state wait.

WAITCR selects output mode from the following: High output by controlling waiting clock output when an external bus is used, $\overline{CLK}$ output during waiting cycles, $\overline{CLK}$ output at all times. When the internal ROM is disabled while programs on the external memory and the internal RAM are operating, data in C000H to FFBFH on the ROM cannot be read under software control. WAITCR is a register which can be written only once. Therefore, the second and the following write operation cannot modify this bit.

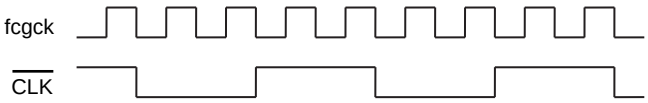
WAIT Control Register

WAITCR (0032H)

7	6	5	4	3	2	1	0	
WAIT		–	"0"	CLKV		–	RDMEM	(Initial value: 01*0 11*1) $\overline{EA}$ ← "0" (Initial value: 00*0 00*1) $\overline{EA}$ ← "1"

WAIT	Control waiting cycles	00: Wait Disable 01: (Fixed) 1-cycle wait Enable 10: (1+n)-cycle wait Enable ($\overline{WAIT}$ pin) 11: Wait Disable	Write only
CLKV	$\overline{CLK}$ output mode	0* : "H"output ($\overline{CLK}$ output disable) 10: $\overline{CLK}$ output enable during waiting cycles 11: always $\overline{CLK}$ output Enable	R/W
RDMEM	security for internal ROM reading out	0: Read out Disable 1: Read out Enable	WR 1 time

Note 1: The timing chart during CLK output enable is described



Note 2: P11 terminal is assigned for $\overline{WAIT}$ input when WAIT is "10"; in the condition, it cannot be used for general-purpose input/output port.
Note 3: WAITCR include a write only register and must not be used with any of the read-modify-write instructions.
Note 4: Make sure to set "0" on bit 4.
Note 5: When external memory is used, the P10 pin cannot be used as an external interrupt input ($\overline{INT0}$) or an I/O port as $\overline{CLK}$ is output from this pin.

2.1.4.2 External Bus Controller

The TMP86CH06AUG is interfaced to an external memory via following buses.

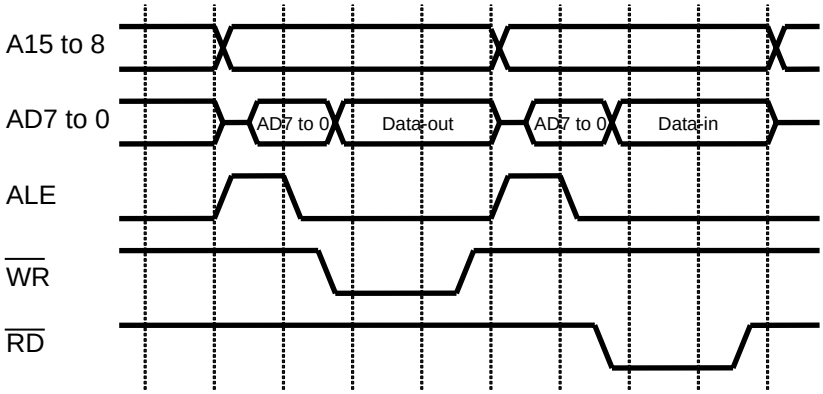
(1) Address/Databus: A15 to 8, AD7 to 0

There are 16-bit output address bus and 8-bit bidirectional databus. The lower 8 bits for address bus and 8 bits for databus are multiplexed. All of the terminals, AD15 to 8 and AD7 to 0, are used also for port. These ports functions as address/databus by setting register EXPCR. Upper bits of address bus can be output either partially or totally.

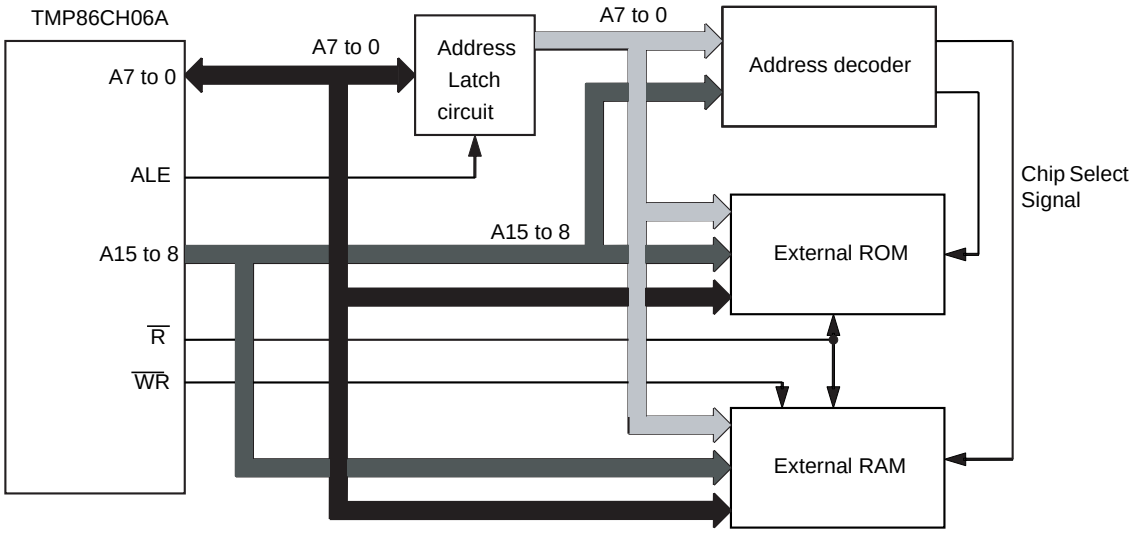
(2) Control signals: $\overline{RD}$, $\overline{WR}$, ALE

$\overline{RD}$ pulse indicates for reading, $\overline{WR}$ pulse indicates for writing, and ALE pulse indicates for address latch enable. These terminals are used also for timing for port, and these ports operate as such control signals. $\overline{RD}$ becomes "L" level on external memory read cycle, and WR becomes "L" level on external memory write cycle. Both RD and WR keep "H" level on both dummy cycle and internal memory read/write cycle.

(a) Timing chart on external memory interface



(b) Example for connecting external ROM/RAM

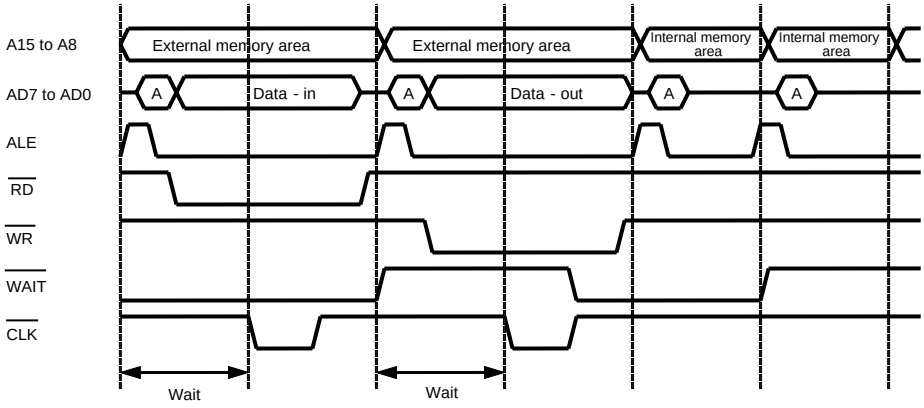


2.1.4.3 Wait Controller

The number of wait cycles is selected from three: no wait, fixed 1-cycle wait and (1+n)-cycle wait.

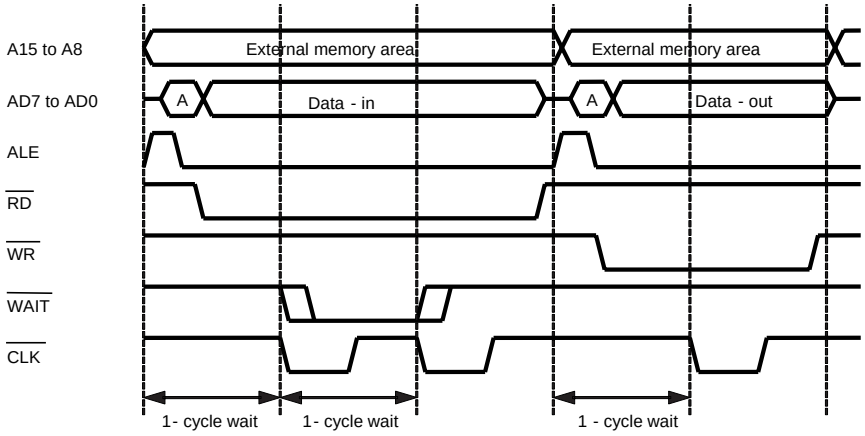
(1) Fixed 1-cycle wait

1-state wait operation is executed regardless of $\overline{\text{WAIT}}$ terminal condition. The following diagram indicates the wait cycle, under WAIT = “01” and /CLK outputs during waiting cycles.



(2) (1+n)-cycle wait

Continuous wait operation is executed as far as "L" input is detected from WAIT terminal, after 1-cycle wait operation is executed. The following diagram indicates the wait cycle, under WAIT="10" and CLK outputs during waiting cycles.



2.1.5 System Clock Controller

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

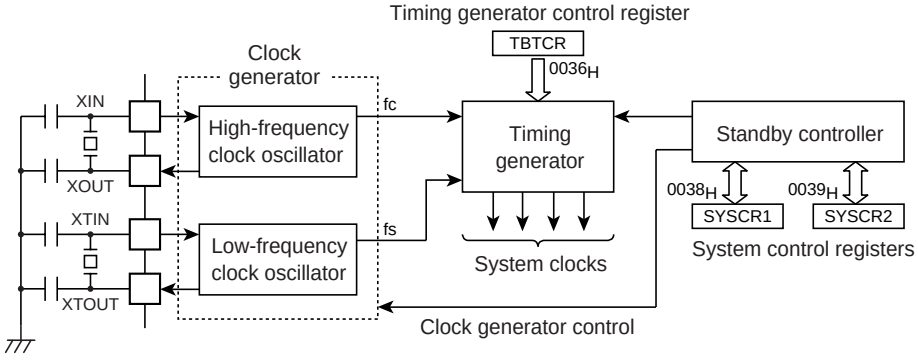


Figure 2-2 System Colck Control

2.1.5.1 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) clocks 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.

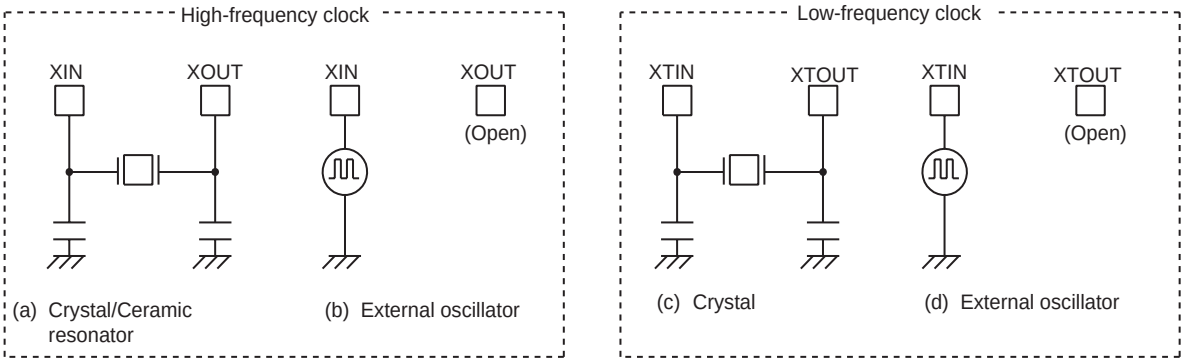


Figure 2-3 Examples of Resonator Connection

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.

2.1.5.2 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

Table 2-1 Divider Output Capability

Divider Output Capability				
DV1	DV2	DV3	DV4	DV5
$fc/2^3$	$fc/2^4$	$fc/2^5$	$fc/2^6$	$fc/2^7$
×	O	O	O	O

Note: When selecting DV1, DV2, DV3, DV4 or DV5 for the operating clock of timer, SIO, etc, check Table 2-1 to see that the divider output is possible (marked with O). If the divider output is impossible (marked with ×), do not select them for the operating clock.

Table 2-2 Input Clock to 7th Stage of the Divider

Single-clock mode	Dual-clock mode		
NORMAL1, IDLE1 mode	NORMAL2, IDLE2 mode (SYSCK=0)		SLOW, SLEEP mode (SYSCK=1)
	DV7CK=0	DV7CK=1	
$fc/2^8$	$fc/2^8$	fs	fs

Note 1: Do not set DV7CK to "1" when single clock mode is selecte.

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

Note 3: In SLOW and SLEEP mode, the input clock to the 1st stage of the divider is stopped; output from the 1st to 6th stages is also stopped.

Note 4: During the warm-up period after STOP mode is released and in IDLE0 mode, the 6th stage of the divider is output to the 7th stage of the divider regardless of the DV7CK setting.

(1) Configuration of timing generator

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

An input clock to the 7th stage of the divider depends on the operating mode, TBTCR<DV7CK> , that is shown in Figure 1-5. As reset and STOP mode started/canceled, the prescaler and the divider are cleared to "0".

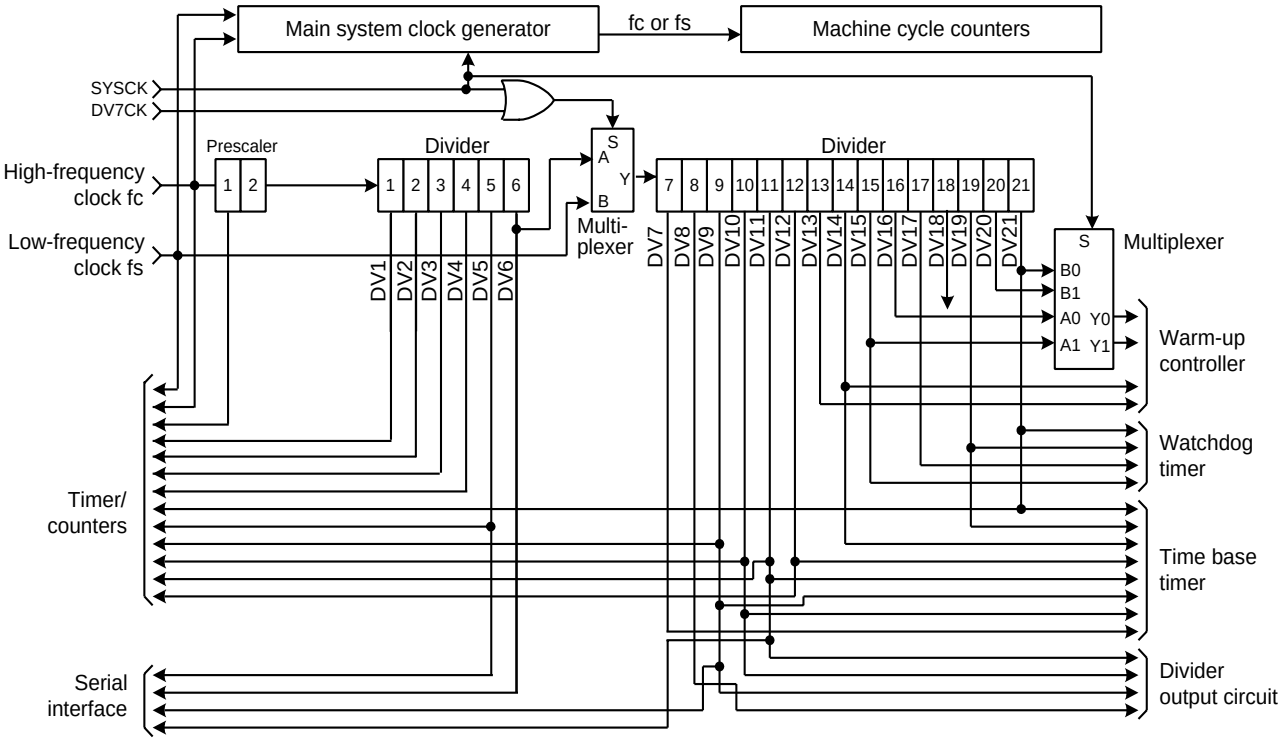


Figure 2-4 Configuration of Timing Generator

Table 2-3 Division Ratio of Divider

	DV7CK=0	DV7CK=1		DV7CK=0	DV7CK=1
DV1	$fc/2^3$	$fc/2^3$	DV12	$fc/2^{14}$	$fs/2^6$
DV2	$fc/2^4$	$fc/2^4$	DV13	$fc/2^{15}$	$fs/2^7$
DV3	$fc/2^5$	$fc/2^5$	DV14	$fc/2^{16}$	$fs/2^8$
DV4	$fc/2^6$	$fc/2^6$	DV15	$fc/2^{17}$	$fs/2^9$
DV5	$fc/2^7$	$fc/2^7$	DV16	$fc/2^{18}$	$fs/2^{10}$
DV6	$fc/2^8$	$fc/2^8$	DV17	$fc/2^{19}$	$fs/2^{11}$
DV7	$fc/2^9$	fs/2	DV18	$fc/2^{20}$	$fs/2^{12}$
DV8	$fc/2^{10}$	$fs/2^2$	DV19	$fc/2^{21}$	$fs/2^{13}$
DV9	$fc/2^{11}$	$fs/2^3$	DV20	$fc/2^{22}$	$fs/2^{14}$
DV10	$fc/2^{12}$	$fs/2^4$	DV21	$fc/2^{23}$	$fs/2^{15}$
DV11	$fc/2^{13}$	$fs/2^5$			

TBTCR (0036H)

7	6	5	4	3	2	1	0
(DVOEN)	(DVOCK)		DV7CK	(TBTEN)	(TBTCK)		

(Initial value: 0**0 0***)

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

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

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

Note 3: fc: High-frequency clock [Hz], 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) 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.

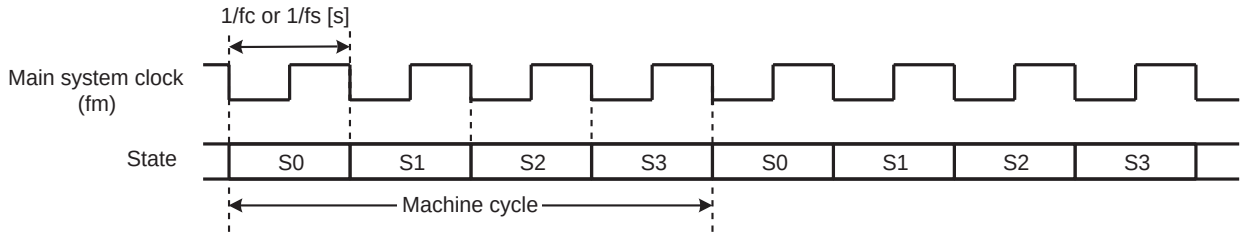


Figure 2-5 Machine Cycle

Table 2-4 Example of Macheine Cycle

Frequency	Machine cycle	
High-frequency clock	fc = 16.0 MHz	0.25 μs
	fc = 12.5MHz	0.32 μs
	fc = 4.2 MHz	0.95 μs
	fc = 1 MHz	4 μs
Low-frequency-clock	fs = 32.8 kHz	122 μs

2.1.5.3 Operation Mode Control Circuit

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

Figure 2-6 shows the operating mode transition diagram.

(1) Single-clock mode

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

(a) NORMAL1 mode

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

The TMP86CH06AUG are placed in this mode after reset.

(b) IDLE1 mode

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

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

(c) IDLE0 mode

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

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

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

When returned from IDLE0 mode, the CPU restarts operating, entering NORMAL1 mode back again. IDLE0 mode is entered and returned regardless of how TBTCR<TBTEN> is set. When IMF = "1", EF6 (TBT interrupt individual enable flag) = "1", and TBTCR<TBTEN> = "1", interrupt processing is performed. When IDLE0 mode is entered while TBTCR<TBTEN> = "1", the INTTBT interrupt latch is set after returning to NORMAL1 mode.

(2) 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.

Note: With the TLCS-870/C Series, no option is provided to select the clock mode after release of reset. When reset is released it becomes the single mode.

(a) NORMAL2 mode

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

(b) SLOW2 mode

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

(c) SLOW1 mode

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

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

(d) IDLE2 mode

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

(e) SLEEP1 mode

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

(f) SLEEP2 mode

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

(g) SLEEP0 mode

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

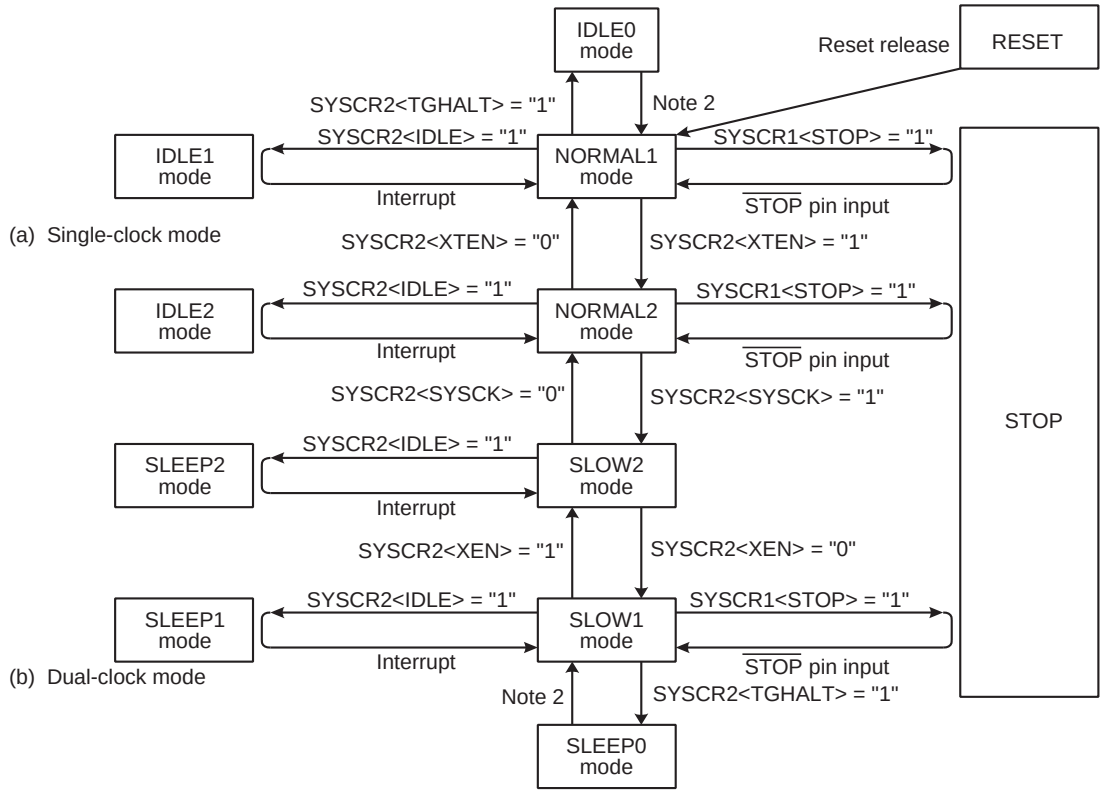
When SLEEP0 mode starts, the CPU stops and the timing generator stops feeding the clock to the peripheral circuits other than TBT. Then, upon detecting the falling edge of the source clock selected with TBTCR<TBTCK>, the timing generator starts feeding the clock to all peripheral circuits.

When returned from SLEEP0 mode, the CPU restarts operating, entering SLOW1 mode back again. SLEEP0 mode is entered and returned regardless of how TBTCR<TBTEN> is set. When IMF = “1”, EF6 (TBT interrupt individual enable flag) = “1”, and TBTCR<TBTEN> = “1”, interrupt processing is performed. When SLEEP0 mode is entered while TBTCR<TBTEN> = “1”, the INTTBT interrupt latch is set after returning to SLOW1 mode.

(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

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 1: CPU core and peripherals are halted (Start STOP mode)			R/W
RELM	Release method for STOP mode	0: Edge-sensitive release 1: Level-sensitive release			
RETM	Operating mode after STOP mode	0: Return to NORMAL1/2 mode 1: Return to SLOW1 mode			
OUTEN	Port output during STOP mode	0: High impedance 1: Output kept			
WUT	Warm-up time at releasing STOP mode		Return to NORMAL mode	Return to SLOW mode	
		00	3×2^{16}/fc	3×2^{13}/fs	
		01	2^{16}/fc	2^{13}/fs	
		10	3×2^{14}/fc	3×2^{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 interrupt request on account of falling edge.

Note 6: When the key-on wakeup is used, the edge realease can not function according to some conditions. It is recommended to set the level realease (RELM = “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 1: Turn on oscillation	R/W
XTEN	Low-frequency oscillator control	0: Turn off oscillation 1: Turn on oscillation	
SYSCK	Main system clock select (Write)/main system clock monitor (Read)	0: High-frequency clock 1: Low-frequency clock	
IDLE	CPU and watchdog timer control (IDLE1/2 and SLEEP1/2 modes)	0: CPU and watchdog timer remain active 1: CPU and watchdog timer are stopped (Start IDLE1/2 and SLEEP1/2 modes)	
TGHALT	TG control (IDLE0 and SLEEP0 modes)	0: Feeding clock to all peripherals from TG 1: Stop feeding clock to peripherals except TBT from TG. (Start IDLE0 and SLEEP0 modes)	

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

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

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, SYSCR2< TGHALT> is automatically cleared to “0”.

Note 8: Before setting SYSCR2< 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.

2.1.5.4 Operating Mode Control

(1) STOP mode

STOP mode is controlled by the system control register 1 and the $\overline{\text{STOP}}$ pin input. 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 SYSCR1<RELM>.

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

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

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

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

1. Testing a port P20.
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    (P2R). 0                ; Wait until the STOP pin input goes low level
        JRS     F, SSTOPH
        DI                              ; IMF←1
        SET     (SYSCR1). 7             ; Starts STOP mode
```

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

```
PINT5:    TEST    (P2R). 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←1
          SET     (SYSCR1). 7           ; Starts STOP mode
SINT5:    RETI
```

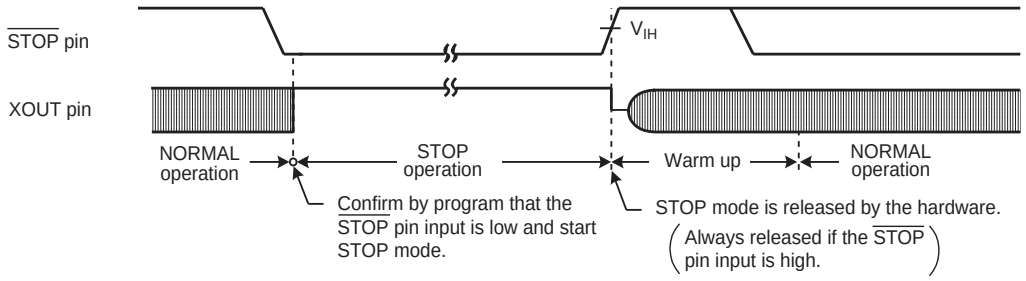


Figure 2-7 Level-sensitive Release Mode

Note 1: Even if the $\overline{\text{STOP}}$ pin input is low after warm up starts, the STOP mode is not restarted.
Note 2: In this case of changing to the level-sensitive mode from the edge-sensitive mode, the release mode is not switched until a rising edge of the $\overline{\text{STOP}}$ pin input is detected.

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

In this mode, STOP mode is released by a rising edge of the $\overline{\text{STOP}}$ pin input. This is used in applications where a relatively short program is executed repeatedly at periodic intervals. This periodic signal (for example, a clock from a low-power consumption oscillator) is input to the $\overline{\text{STOP}}$ pin. In the edge-sensitive release mode, STOP mode is started even when the $\overline{\text{STOP}}$ pin input is high level.

Example :Starting STOP mode from NORMAL mode

```
          DI                            ; IMF←1
          LD      (SYSCR1), 10010000B   ; Starts after specified to the edge-sensitive release mode
```

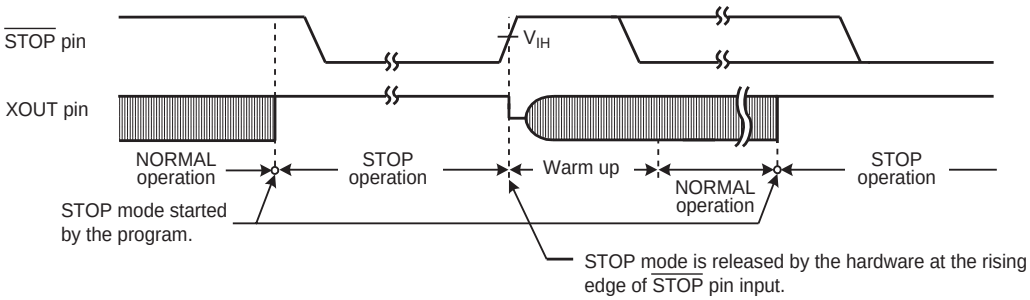


Figure 2-8 Edge-sensitive Release Mode

STOP mode is released by the following sequence.

1. In the dual-clock mode, when returning to NORMAL2 or SLOW2, 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 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. The start is made after the prescaler and the divider of the timing generator are cleared to "0".

Table 2-5 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: 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.

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

Note: 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).

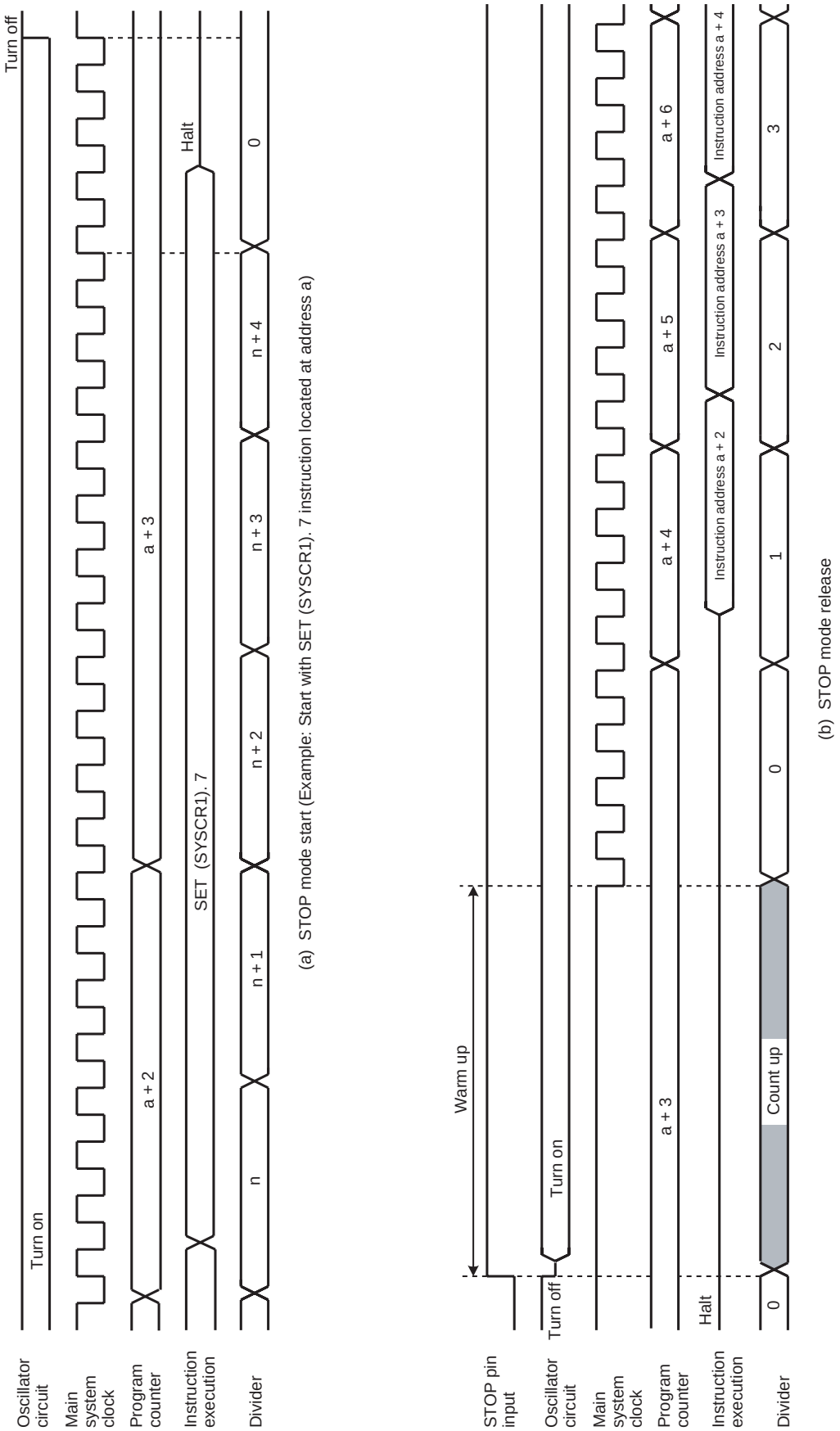


Figure 2-9 STOP Mode Start/Release

(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.

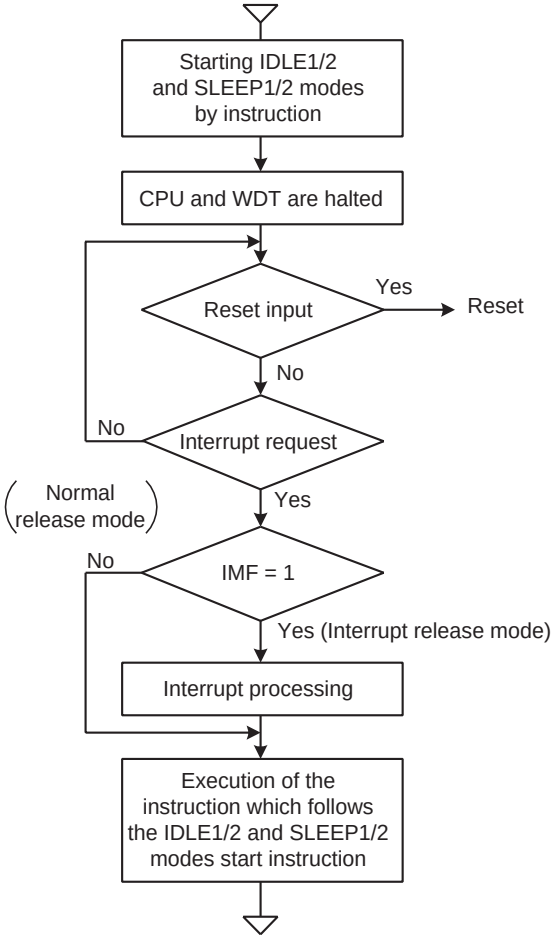


Figure 2-10 IDLE1/2 and SLEEP1/2 Modes

(a) Start the IDLE1/2 and SLEEP1/2 modes

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

(b) 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.

(c) 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.

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

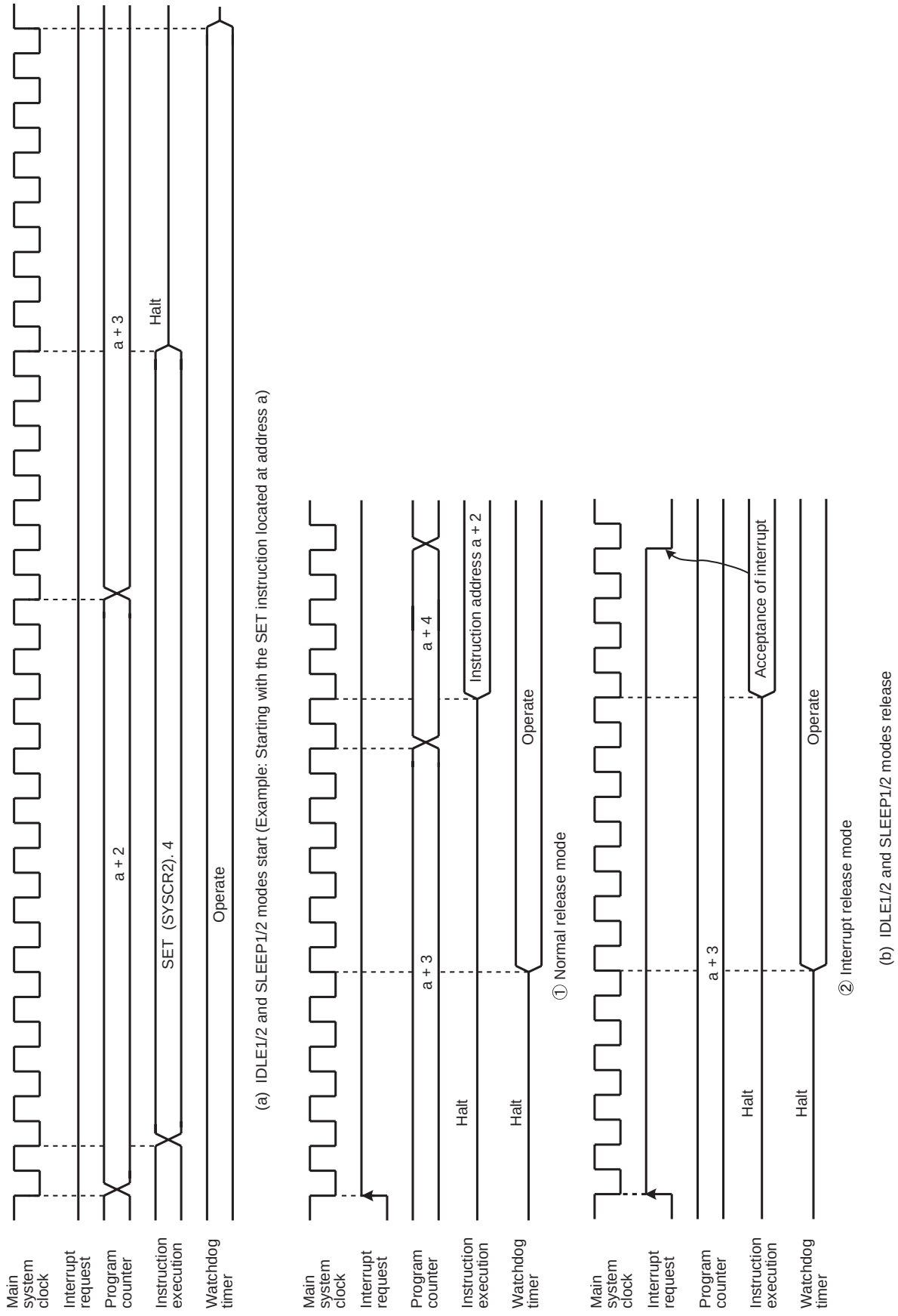


Figure 2-11 IDLE1/2 and SLEEP1/2 Modes Start/Release

(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.

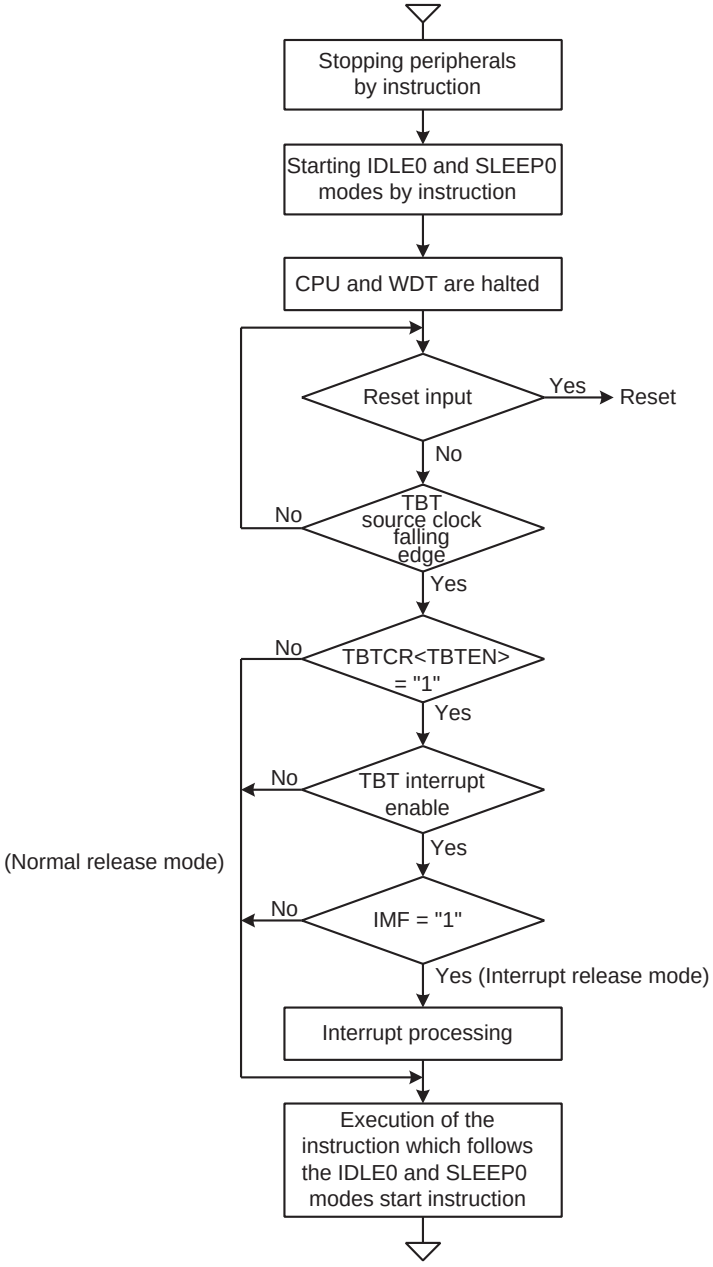


Figure 2-12 IDLE0 and SLEEP0 Modes

(a) Start the IDLE0 and SLEEP0 modes

Stop (Disable) peripherals such as a timer counter.

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

(b) 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).

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.

(c) Normal release mode (IMF • EF6 • ETBTCR<TBTEN> = "0")

IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK> without reference to individual interrupt enable flag (EF). After the falling edge is detected, the program operation is resumed from the instruction following the IDLE0 and SLEEP0 modes start instruction.

(d) Interrupt release mode (IMF • EF6 • ETBTCR<TBTEN> = "1")

IDLE0 and SLEEP0 modes are released by the source clock falling edge, which is setting by the TBTCR<TBTCK> at INTTBT interrupt source enabled with the individual interrupt enable flag (EF) 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.

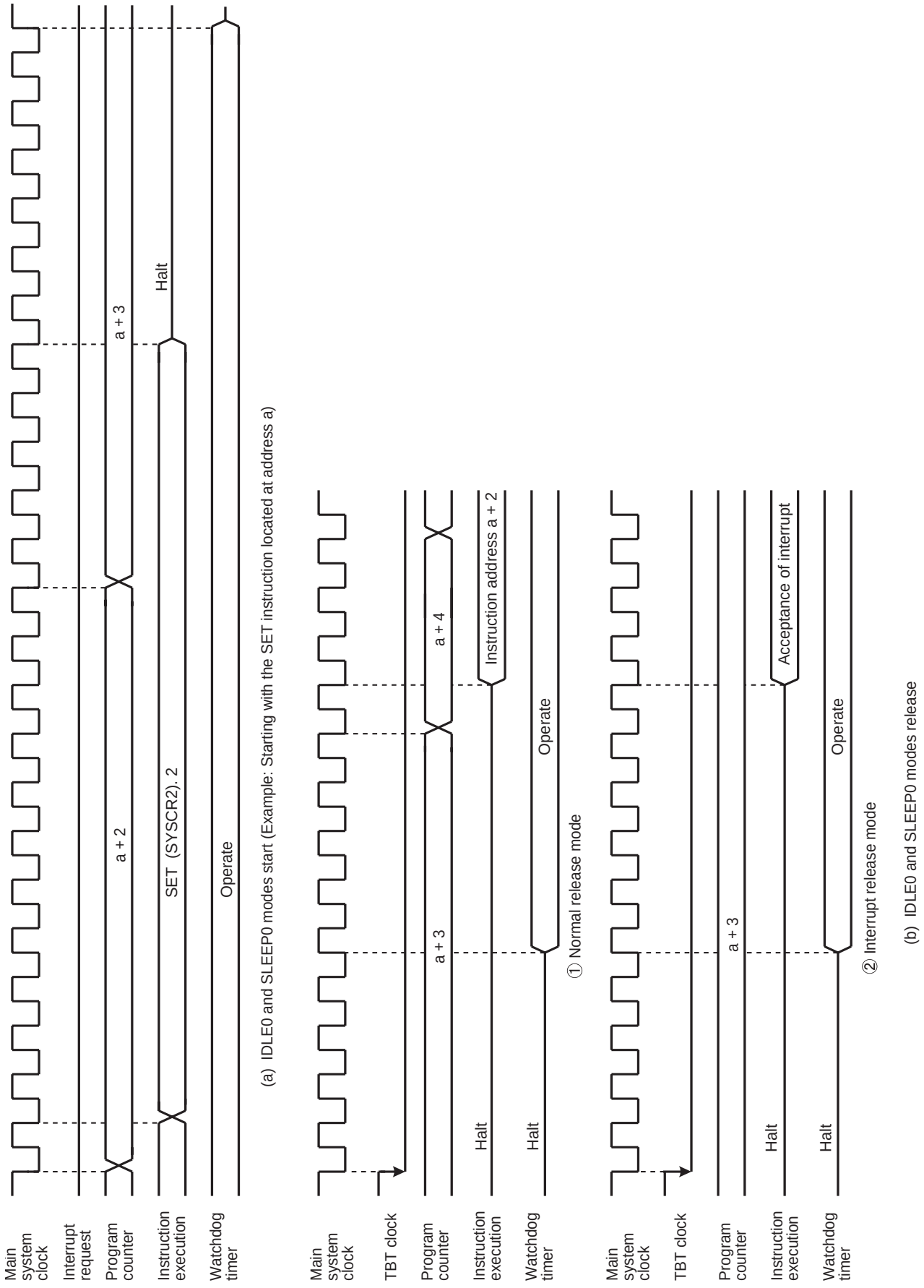


Figure 2-13 IDLE0 and SLEEP0 Modes Start/Release

(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 (TC1, TC0).

(a) Switching from NORMAL2 mode to SLOW1 mode

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

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

Note: The high-frequency clock 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.

When the low-frequency clock oscillation is unstable, wait until oscillation stabilizes before performing the above operations. The timer/counter 1, 0 (TC1, TC0) can conveniently be used to confirm that low-frequency clock oscillation has stabilized.

Example 1 :Switching from NORMAL2 mode to SLOW1 mode.

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

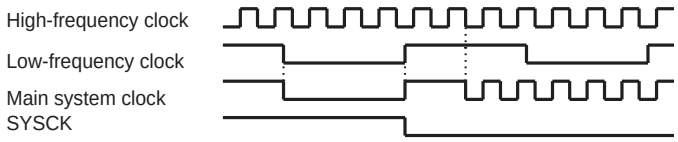
Example 2 :Switching to the SLOW1 mode after low-frequency clock has stabilized.

```
        SET     (SYSCR2). 6     ; SYSCR2<XTEN> ← 1
        LD      (TC0CR), 43H    ; Sets mode for TC1, TC0 (16-bit TC, fs for source)
        LD      (TC1CR), 05H
        LDW     (TTREG0), 8000H ; Sets warm-up time (Depend on oscillator accompanied)
        DI                      ; IMF ← 0
        SET     (EIRL). 7       ; Enables INTTC1
        EI                      ; IMF ← 1
        SET     (TC1CR). 3      ; Starts TC1, 0
         :
PINTTC1: CLR    (TC1CR). 3      ; Stops TC1, 0
        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
         :
VINTTC1: DW     PINTTC1         ; INTTC1 vector table
```

(b) Switching from SLOW1 mode to NORMAL2 mode

First, set SYSCR2<XEN> to turn on the high-frequency oscillation. When time for stabilization (Warm up) has been taken by the timer/counter 1, 0 (TC1, TC0), clear SYSCR2<SYSCK> to switch the main system clock to the high-frequency clock.

Note: After SYSCR2<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.



Note: SLOW mode can also be released by inputting low level on the $\overline{\text{RESET}}$ pin, which immediately performs the reset operation. After reset, TMP86CH06AUG are placed in NORMAL1 mode.

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

```
                SET     (SYSCR2). 7         ; SYSCR2<XEN> ← 1 (Starts high-frequency oscillation)
                LD      (TC0CR), 63H        ; Sets mode for TC1, TC0 (16-bit TC, fc for source)
                LD      (TC1CR), 05H
                LD      (TTREG1), 0F8H      ; Sets warm-up time
                DI                          ; IMF ← 0
                SET     (EIRL). 7           ; Enables INTTC1
                EI                          ; IMF ← 1
                SET     (TC1CR). 3          ; Starts TC1, 0
                 :
PINTTC1:        CLR     (TC1CR). 3          ; Stops TC1, 0
                CLR     (SYSCR2). 5         ; SYSCR2<SYSCK> ← 0
                                              (Switches the main system clock to the high-frequency clock)
                RETI
                 :
VINTTC1:        DW      PINTTC1             ; INTTC1 vector table
```

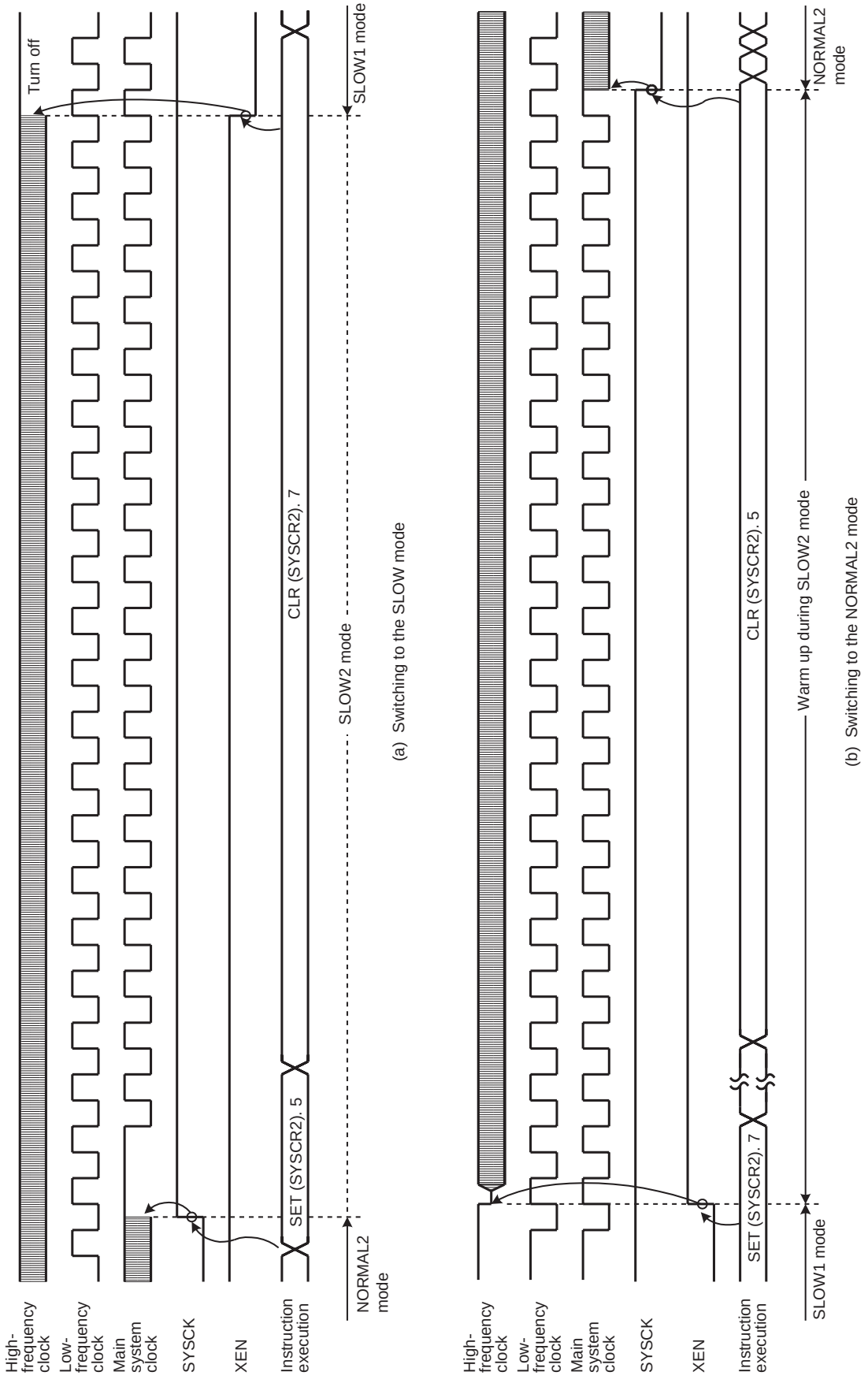


Figure 2-14 Switching between the NORMAL2 and SLOW Modes

2.1.6 Reset Circuit

The TMP86CH06AUG have four types of reset generation procedures: An external reset input, an address trap reset, a watchdog timer reset and a system clock reset. Table 2-6 shows on-chip hardware initialization by reset action.

The malfunction reset circuit such as watchdog timer reset, address trap reset and system clock reset is not initialized when power is turned on. The $\overline{\text{RESET}}$ pin can reset state at the maximum 24/fc [s] (1.5 ms at 16.0 MHz) when power is turned on.

Table 2-6 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

2.1.6.1 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.

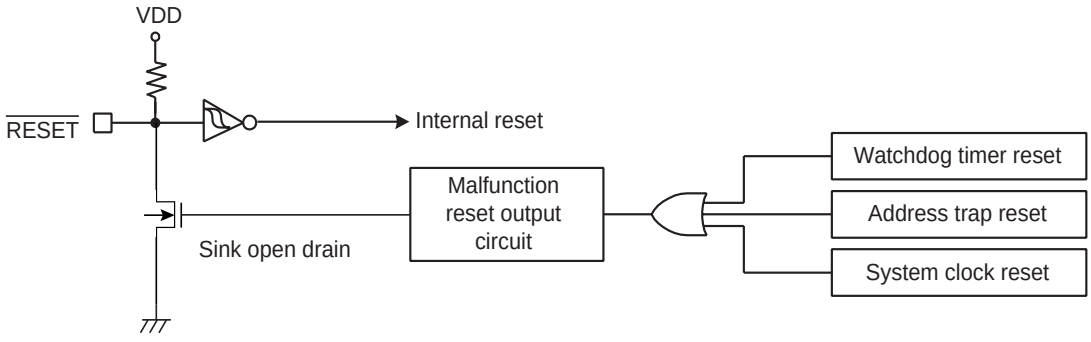
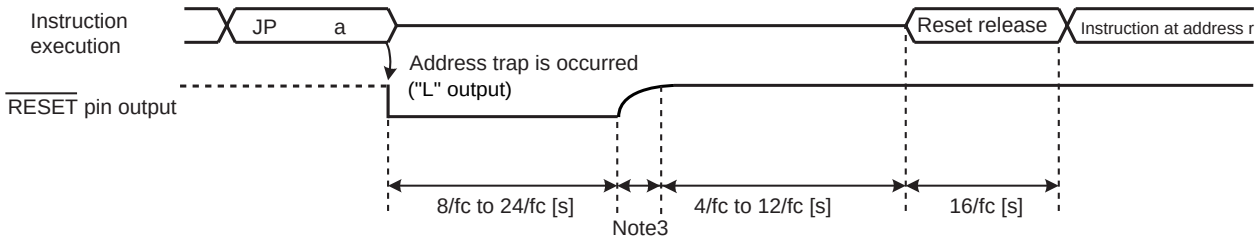


Figure 2-15 Reset Circuit

2.1.6.2 Address trap reset

If the CPU should start looping for some cause such as noise and an attempt be made to fetch an instruction from the on-chip RAM (when WDTCR1<ATAS> is set to “1”) or the SFR area, address trap reset will be generated. The reset time is about 8/fc to 24/fc [s] (0.5 to 1.5 ms at 16.0 MHz).



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

Note 2: During reset release, reset vector “r” is read out, and an instruction at address “r” is fetched and decoded.

Note 3: Varies on account of external condition : Voltage or Capacitance.

Figure 2-16 Address Trap Reset

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

2.1.6.3 Watchdog timer reset

Refer to “Watchdog Timer”.

2.1.6.4 System clock reset

Clearing both SYSCR2<XEN> and SYSCR2<XTEN> to “0”, clearing SYSCR2<XEN> to “0” when SYSCK = “0”, or clearing SYSCR2<XTEN> to “0” when SYSCK = “1” stops system clock, and causes the microcomputer to deadlock. This can be prevented by automatically generating a reset signal whenever XEN = XTEN = “0”, XEN = SYSCK = “0”, or XTEN = “0”/SYSCK = “1” is detected to continue the oscillation. Reset signal output comes from $\overline{\text{RESET}}$ pin. The reset time is about 8/fc to 24/fc [s] (0.5 to 1.5 ms at 16.0 MHz).

3. Interrupt Control Circuit

The TMP86CH06AUG has a total of 21 interrupt sources excluding reset, of which 5 source levels are multiplexed. Interrupts can be nested with priorities. Four of the internal interrupt sources are non-maskable while the rest are maskable.

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

Interrupt Factors		Enable Condition	Interrupt Latch	Vector Address	Priority
Internal/External	(Reset)	Non-maskable	–	FFFE	1
Internal	INTSWI (Software interrupt)	Non-maskable	–	FFFC	2
Internal	INTUNDEF (Executed the undefined instruction interrupt)	Non-maskable	–	FFFC	2
Internal	INTATRAP (Address trap interrupt)	Non-maskable	IL2	FFFA	2
Internal	INTWDT (Watchdog timer interrupt)	Non-maskable	IL3	FFF8	2
External	INT1	IMF• EF5 = 1	IL5	FFF4	6
Internal	INTTBT	IMF• EF6 = 1	IL6	FFF2	7
Internal	INTTC1	IMF• EF7 = 1	IL7	FFF0	8
Internal	INTRXD0	IMF• EF8 = 1, IL8ER = 0	IL8	FFEE	9
Internal	INTSIO	IMF• EF8 = 1, IL8ER = 1			
Internal	INTTXD0	IMF• EF9 = 1	IL9	FFEC	10
Internal	INTET0	IMF• EF10 = 1	IL10	FFEA	11
Internal	INTIC0	IMF• EF11 = 1	IL11	FFE8	12
Internal	INTOC0	IMF• EF12 = 1, IL12ER = 0	IL12	FFE6	13
External	INT2	IMF• EF12 = 1, IL12ER = 1			
External	INT3	IMF• EF13 = 1, IL13ER = 0	IL13	FFE4	14
Internal	INTRXD1	IMF• EF13 = 1, IL13ER = 1			
External	INT4	IMF• EF14 = 1, IL14ER = 0	IL14	FFE2	15
Internal	INTTXD1	IMF• EF14 = 1, IL14ER = 1			
Internal	INTTC0	IMF• EF15 = 1, IL15ER = 0	IL15	FFE0	16
External	INT5	IMF• EF15 = 1, IL15ER = 1			

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

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

Note 3: To use the watchdog timer interrupt (INTWDT), clear WDTCR1<WDTOUT> to "0" (It is set for the "Reset request" after reset is released). For details, see "Watchdog Timer".

3.1 Interrupt latches (IL15 to IL2)

An interrupt latch is provided for each interrupt source, except for a software interrupt and an executed the undefined instruction interrupt. When interrupt request is generated, the latch is set to "1", and the CPU is requested to accept the interrupt if its interrupt is enabled. The interrupt latch is cleared to "0" immediately after accepting interrupt. All interrupt latches are initialized to "0" during reset.

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

Interrupt latches are not set to “1” by an instruction.

Since interrupt latches can be read, the status for interrupt requests can be monitored by software.

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

Example 1 :Clears interrupt latches

```
DI                              ; IMF ← 0
LDW     (ILL), 1110100000111111B ; IL12, IL10 to IL6 ← 0
EI                              ; IMF ← 1
```

Example 2 :Reads interrupt 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 003AH and 003BH in SFR area, and they can be read and written by an instructions (Including read-modify-write instructions such as bit manipulation or operation instructions).

3.2.1 Interrupt master enable flag (IMF)

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

The IMF is located on bit0 in EIRL (Address: 003AH in SFR), and can be read and written by an instruction. The IMF is normally set and cleared by [EI] and [DI] instruction respectively. During reset, the IMF is initialized to “0”.

3.2.2 Individual interrupt enable flags (EF15 to EF4)

Each of these flags enables and disables the acceptance of its maskable interrupt. Setting the corresponding bit of an individual interrupt enable flag to “1” enables acceptance of its interrupt, and setting the bit to “0” disables acceptance. During reset, all the individual interrupt enable flags (EF15 to EF4) are initialized to “0” and all maskable interrupts are not accepted until they are set to “1”.

Note: In main program, before manipulating the interrupt enable flag (EF) or the interrupt latch (IL), be sure to clear IMF to "0" (Disable interrupt by DI instruction). Then set IMF newly again as required after operating on the EF or IL (Enable interrupt by EI instruction)
In interrupt service routine, because the IMF becomes "0" automatically, clearing IMF need not execute nor-

mally 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)

		at RD	at WR	
IL15 to IL2	Interrupt latches	0: No interrupt request 1: Interrupt request	0: Clears the interrupt request 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)

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

Note 1: *: Don't care

Note 2: Do not set IMF and the interrupt enable flag (EF15 to EF4) to "1" at the same time.

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

3.3 Interrupt Source Selector (INTSEL)

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

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

1. INTRXD0 and INTSIO share the interrupt source level whose priority is 9.
2. INTOC0 and INT2 share the interrupt source level whose priority is 13.
3. INT3 and INTRXD1 share the interrupt source level whose priority is 14.
4. INT4 and INTTXD1 share the interrupt source level whose priority is 15.
5. INTTC0 and INT5 share the interrupt source level whose priority is 16.

Interrupt source selector

INTSEL (003EH)

7	6	5	4	3	2	1	0
IL8ER	-	-	-	IL12ER	IL13ER	IL14ER	IL15ER

(Initial value: 0*** 0000)

IL8ER	Selects INTRXD0 or INTSIO	0: INTRXD0 1: INTSIO	R/W
IL12ER	Selects INTOC0 or INT2	0: INTOC0 1: INT2	R/W
IL13ER	Selects INT3 or INTRXD1	0: INT3 1: INTRXD1	R/W
IL14ER	Selects INT4 or INTTXD1	0: INT4 1: INTTXD1	R/W
IL15ER	Selects INTTC0 or INT5	0: INTTC0 1: INT5	R/W

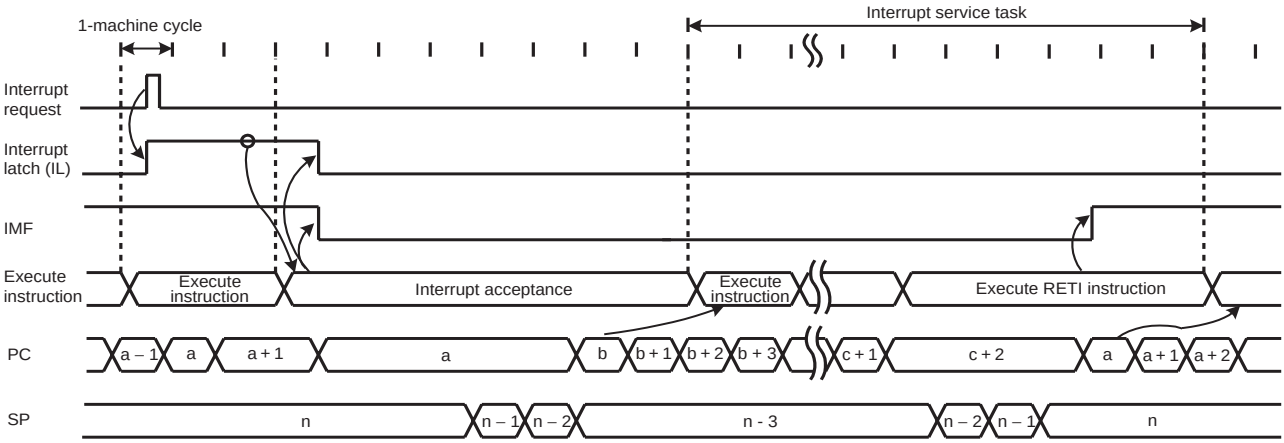
3.4 Interrupt Sequence

An interrupt request, which raised interrupt latch, is held, until interrupt is accepted or interrupt latch is cleared to "0" by resetting or an instruction. Interrupt acceptance sequence requires 8 machine cycles (2 μs @16 MHz) after the completion of the current instruction. The interrupt service task terminates upon execution of an interrupt return instruction [RETI] (for maskable interrupts) or [RETN] (for non-maskable interrupts). Figure 3-1 shows the timing chart of interrupt acceptance processing.

3.4.1 Interrupt acceptance processing is packaged as follows.

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

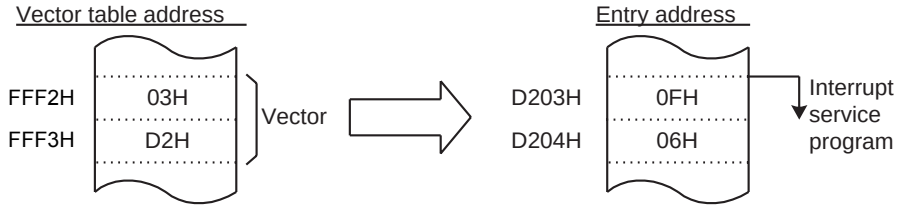


Figure 3-2 Vector table address,Entry address

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.

3.4.2 Saving/restoring general-purpose registers

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

3.4.2.1 Using PUSH and POP instructions

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

Example :Save/store register using PUSH and POP instructions

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

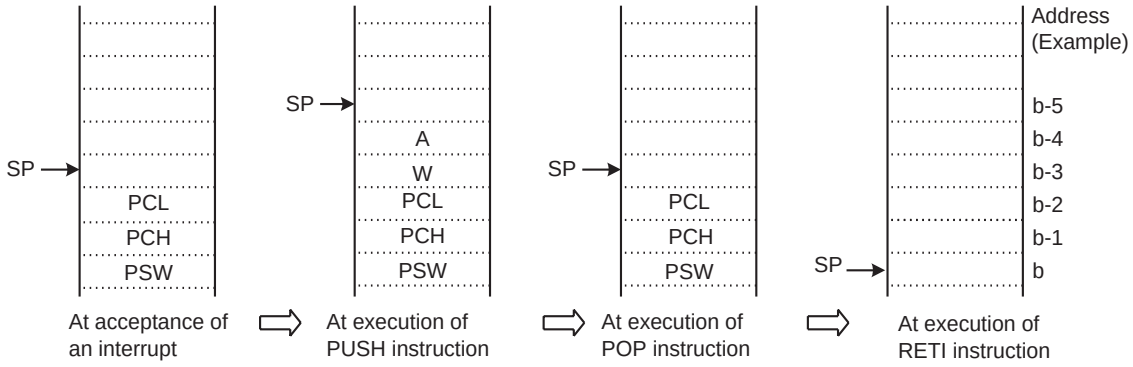


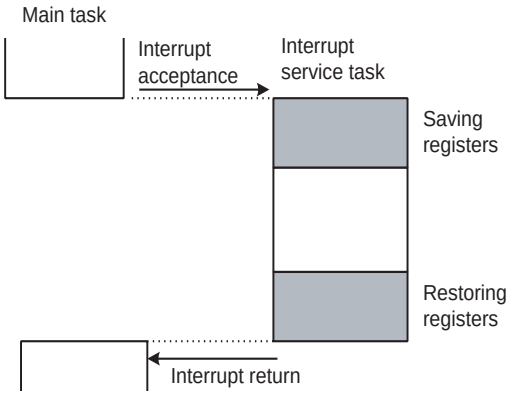
Figure 3-3 Save/store register using PUSH and POP instructions

3.4.2.2 Using data transfer instructions

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

Example :Save/store register using data transfer instructions

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



Saving/Restoring general-purpose registers using PUSH/POP data transfer instruction

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

3.4.3 Interrupt return

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

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

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

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

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

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

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

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

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

Note 1: It is recommended that stack pointer be return to rate before INTATRAP (Increment 3 times), if return interrupt instruction [RETN] is not utilized during interrupt service program under INTATRAP (such as Example 2).

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

3.5 Software Interrupt (INTSW)

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

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

3.5.1 Address error detection

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

3.5.2 Debugging

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

3.6 Undefined Instruction Interrupt (INTUNDEF)

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

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

3.7 Address Trap Interrupt (INTATRAP)

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

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

3.8 External Interrupts

The TMP86CH06AUG has 6 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 INT4. The $\overline{\text{INT0}}$/P10 pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.

Edge selection, noise reject control and $\overline{\text{INT0}}$/P10 pin function selection are performed by the external interrupt control register (EINTCR).

Source	Pin	Enable Conditions	Release Edge (level)	Digital Noise Reject
INT0	$\overline{INT0}$	IMF • EF4 • INT0EN=1	Falling edge	Pulses of less than 2/fc [s] are eliminated as noise. Pulses of 7/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT1	INT1	IMF • EF5 = 1	Falling edge or Rising edge	Pulses of less than 15/fc or 63/fc [s] are eliminated as noise. Pulses of 49/fc or 193/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT2	INT2	IMF • EF12 = 1 and IL12ER=1	Falling edge or 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 • EF13 = 1 and IL13ER=0	Falling edge or 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.
INT4	INT4	IMF • EF14 = 1 and IL14ER=0	Falling edge, Rising edge, Falling and Rising edge or H level	Pulses of less than 7/fc [s] are eliminated as noise. Pulses of 25/fc [s] or more are considered to be signals. In the SLOW or the SLEEP mode, pulses of less than 1/fs [s] are eliminated as noise. Pulses of 3.5/fs [s] or more are considered to be signals.
INT5	$\overline{INT5}$	IMF • EF15 = 1 and IL15ER=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 $\overline{INT0}$ 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	INT4ES	INT3ES	INT2ES	INT1ES		(Initial value: 0000 000*)

INT1NC	Noise reject time select	0: Pulses of less than 63/fc [s] are eliminated as noise 1: Pulses of less than 15/fc [s] are eliminated as noise	R/W
INT0EN	P10/$\overline{\text{INT0}}$ pin configuration	0: P10 input/output port 1: $\overline{\text{INT0}}$ pin (Port P10 should be set to an input mode)	R/W
INT4 ES	INT4 edge select	00: Rising edge 01: Falling edge 10: Rising edge and Falling edge 11: H level	R/W
INT3 ES	INT3 edge select	0: Rising edge 1: Falling edge	R/W
INT2 ES	INT2 edge select	0: Rising edge 1: Falling edge	R/W
INT1 ES	INT1 edge select	0: Rising edge 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.

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

4. Special Function Register (SFR)

The TMP86CH06AUG adopts the memory mapped I/O system, and all peripheral control and data transfers are performed through the special function register (SFR). The SFR is mapped on address 0000H to 003FH.

This chapter shows the arrangement of the special function register (SFR) for TMP86CH06AUG.

4.1 SFR

Address	Read	Write
0000H	P0DR	
0001H	P1DR	
0002H	P2DR	
0003H	P3DR	
0004H	P4DR	
0005H	Reserved	
0006H	Reserved	
0007H	Reserved	
0008H	P0CR	
0009H	P1CR	
000AH	P3CR	
000BH	P4CR	
000CH	P4OED	
000DH	P2R	-
000EH	P4R	-
000FH	Reserved	
0010H	ET0CR	
0011H	ET0MIO	
0012H	ET0RL	
0013H	ET0RH	
0014H	ET0ICAL	-
0015H	ET0ICAH	-
0016H	ET0ICBL	PME
0017H	ET0ICBH	-
0018H	ET0OCRL	
0019H	ET0OCRH	
001AH	UART0SR	UARTCR1
001BH	-	UART0CR2
001CH	RD0BUF	TD0BUF
001DH	RD1BUF	TD1BUF
001EH	UART1SR	UART1CR1
001FH	-	UART1CR2
0020H	TC0CR	
0021H	TC1CR	
0022H	TTREG0	
0023H	TTREG1	
0024H	PWREG0	
0025H	PWREG1	
0026H	-	SIOCR1
0027H	SIOSR	SIOCR2

Address	Read	Write
0028H	SIOBR0	
0029H	SIOBR1	
002AH	SIOBR2	
002BH	SIOBR3	
002CH	SIOBR4	
002DH	SIOBR5	
002EH	SIOBR6	
002FH	SIOBR7	
0030H	Reserved	
0031H	-	EXPCR
0032H	WAITCR	
0033H	Reserved	
0034H	-	WDTCR1
0035H	-	WDTCR2
0036H	TBTCR	
0037H	EINTCR	
0038H	SYSCR1	
0039H	SYSCR2	
003AH	EIRL	
003BH	EIRH	
003CH	ILL	
003DH	ILH	
003EH	INTSEL	
003FH	PSW	

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

Note 2: – ; Cannot be accessed.

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

5. I/O Ports

The TMP86CH06AUG has 5 parallel input/output ports (35 pins) as follows.

1. Port P0;

   8-bit I/O port (utilized also for Address/Data bus)

2. Port P1;

   8-bit I/O port (utilized also for External interrupt input, Timer input/output and External memory management output)

3. Port P2;

   3-bit I/O port (utilized also for Low frequency resonator connections, External interrupt input and STOP mode releasing signal input)

4. Port P3;

   8-bit I/O port (utilized also for Address bus output, Timer input/output and External interrupt input)

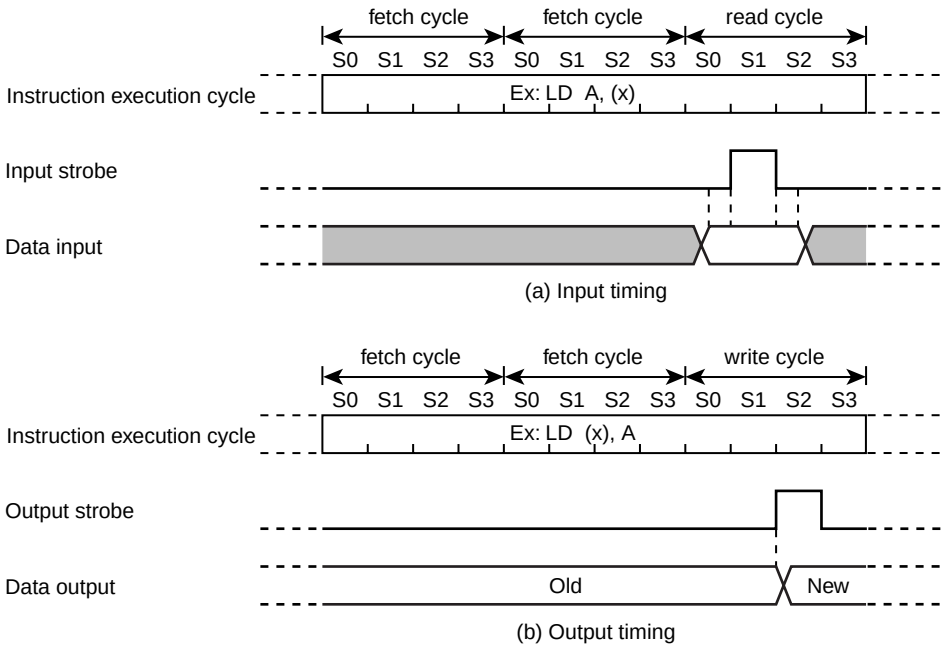
5. Port P4;

   8-bit I/O port (utilized also for Timer input and Serial interface input/output)

Each output port contains a latch, which holds the output data. All input ports do not have latches, so the external input data should be externally held until the input data is read from outside or reading should be performed several timer before processing. Figure 5-1 shows input/output timing examples.

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

Output data changes in the S2 state of the write cycle during execution of the instruction which writes to an I/O port.



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

Figure 5-1 Input/Output Timing (Example)

5.1 Port P0

Port P0 is the 8-bit I/O port that allows selection of input/output on bit basis. Input/output mode is specified on the P0 port input/output control register (P0CR). During reset, all the bits on P0CR are initialized to "0" and P0 becomes input port. Reset operation also initializes all the bits on P0 port output latch (P0DR) to "0".

Besides input/output port, Port P0 functions as multiplexed Address/Data bus (AD7 to 0). Port P0 becomes 8-bit bidirectional Address/Data bus (AD7 to 0) when the CPU accesses to the external memory. When it functions as data bus, the judgment on its input rate is based on TTL level.

Note: Input status is read while the port is input mode. Therefore the contents of output latch, that belongs to the terminal for input, may alter if both input and output are mixed in P0 port.

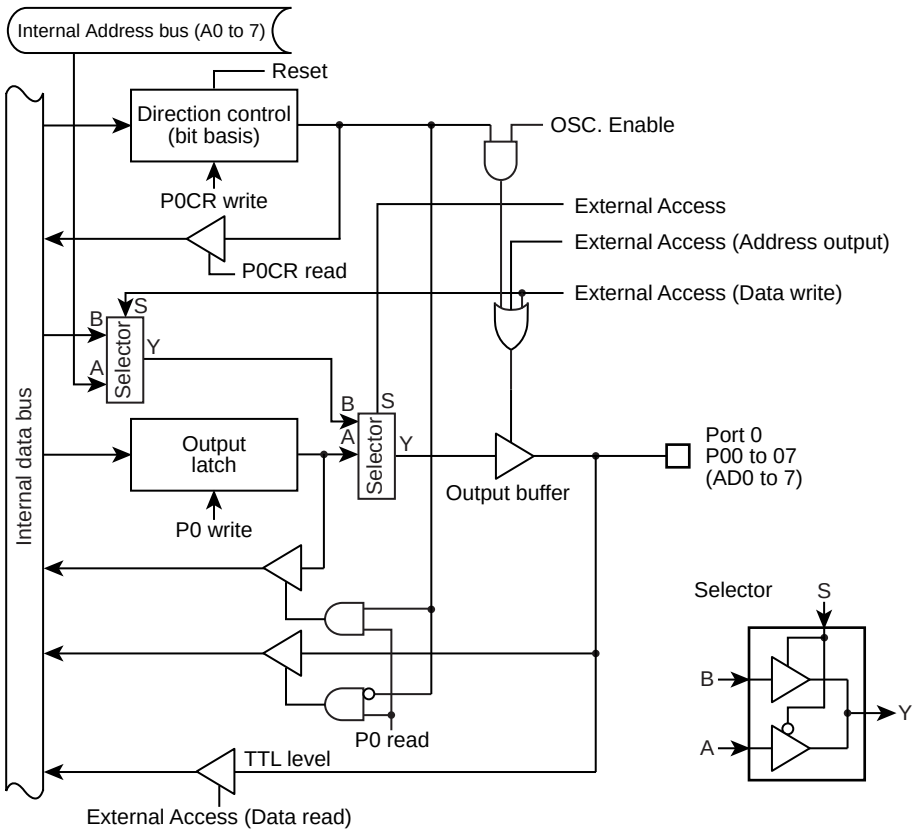


Figure 5-2 Port P0

Port P0

	7	6	5	4	3	2	1	0	
P0DR (0000H)	P07	P06	P05	P04	P03	P02	P01	P00	(Initial value: 0000 0000)

	7	6	5	4	3	2	1	0	
P0CR (0008H)	P0CR7	P0CR6	P0CR5	P0CR4	P0CR3	P0CR2	P0CR1	P0CR0	(Initial value: 0000 0000)

P0CR	I/O control for port P0 (Set for each bit individually)	0: Input mode 1: Output mode	R/W

5.2 Port P1

Port P1 is the 8-bit I/O port that allows selection of input/output on bit basis. Input/output mode is specified on the P1 port input/output control register (P1CR). During reset, all the bits on P1CR are initialized to "0" and P1 becomes input port. Reset operation also initializes all the bits on P1 port output latch (P1DR) to "0".

Besides input/output port, the terminals in Port P1 functions as follows.

P15, P16 and P17 have $\overline{WR}$, $\overline{RD}$ and ALE function respectively. These terminals output signals for $\overline{WR}$, $\overline{RD}$ and ALE when the CPU accesses to the external memory. In order to utilize $\overline{WR}$ and $\overline{RD}$ functions, set WROE and RDOE, located on the EXPCR, respectively. If WROE, RDOE or both are enabled, P17 functions as ALE. If the device is released from reset while $\overline{EA}$ terminal is low, $\overline{CLK}$ signal is uttered every machine cycle.

P10, P11 and P12 have $\overline{INT0}$, INT1, and ETC0 function respectively. In order to utilize these functions, the terminal should be set for input.

Furthermore, pll has the function of $\overline{WAIT}$ while the external memory is begin connected. If the $\overline{WAIT}$ function is not necessary as external memory is utilized, clear the WAIT bits (bit 7, 6) on WAITCR to "00".

P13 and P14 have $\overline{DVO}$ and TO1 function respectively. In order to utilize these functions, the output latch belongs to each terminal should be set to "1" before the terminal is set for output.

Note 1: Input status is read while the port is input mode. Therefore the contents of output latch, that belongs to the terminal for input, may alter if both input and output are mixed in P1 port.

Note 2: When the external memory is used, P10 pin cannot be used as external interrupt pin ($\overline{INT0}$) or input/output port because $\overline{CLK}$ is output from P10.

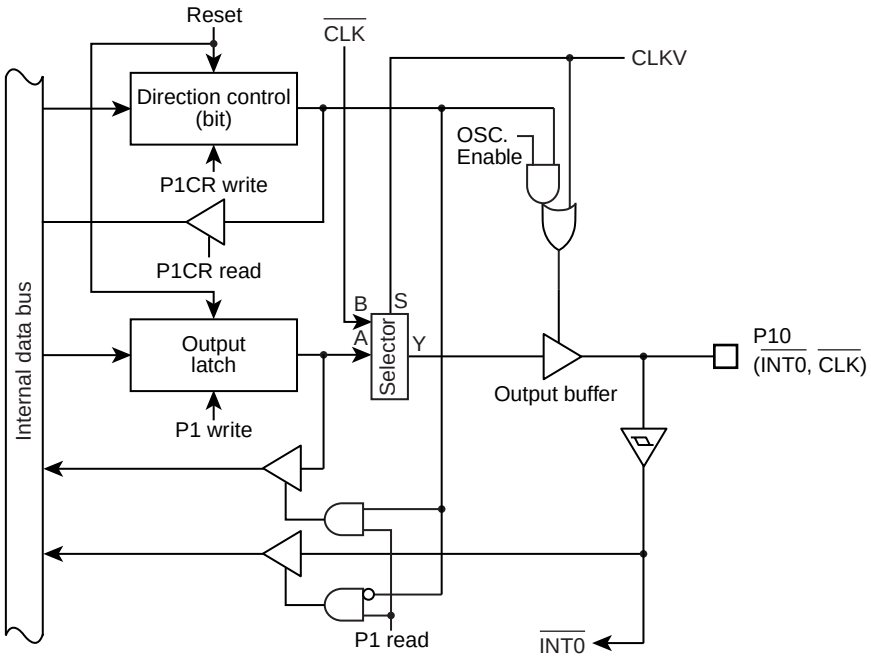


Figure 5-3 P10

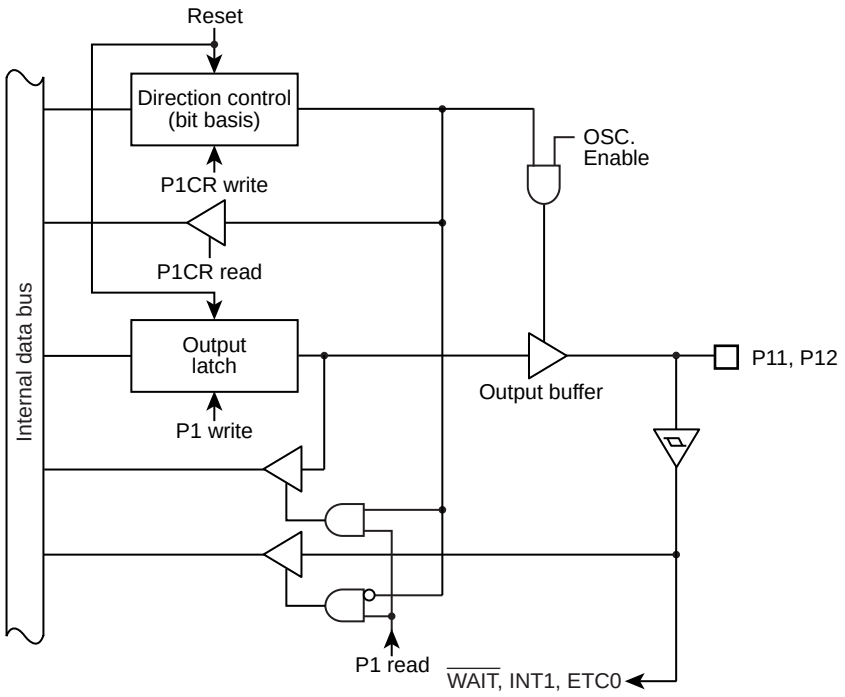


Figure 5-4 P11, P12

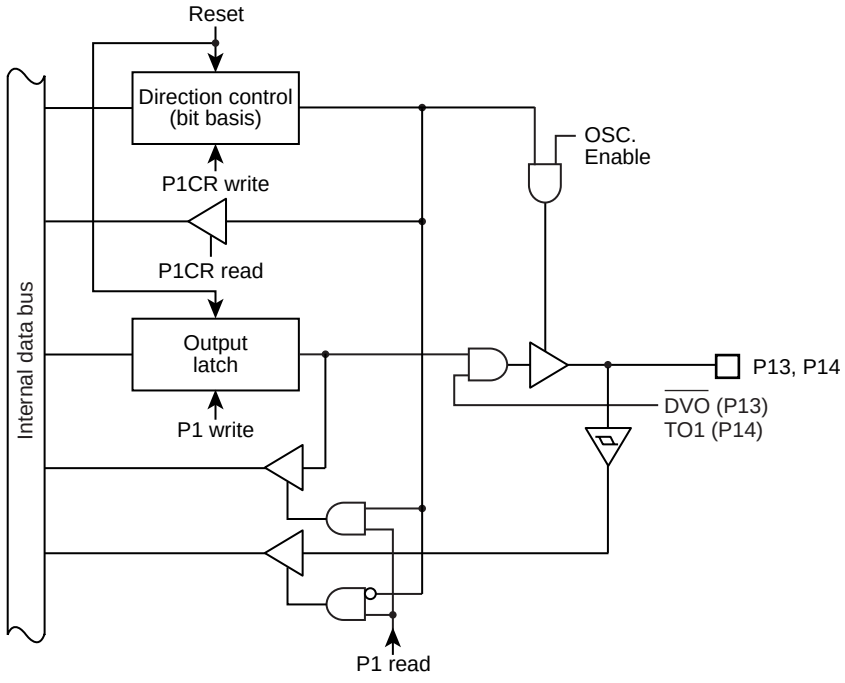


Figure 5-5 P13, P14

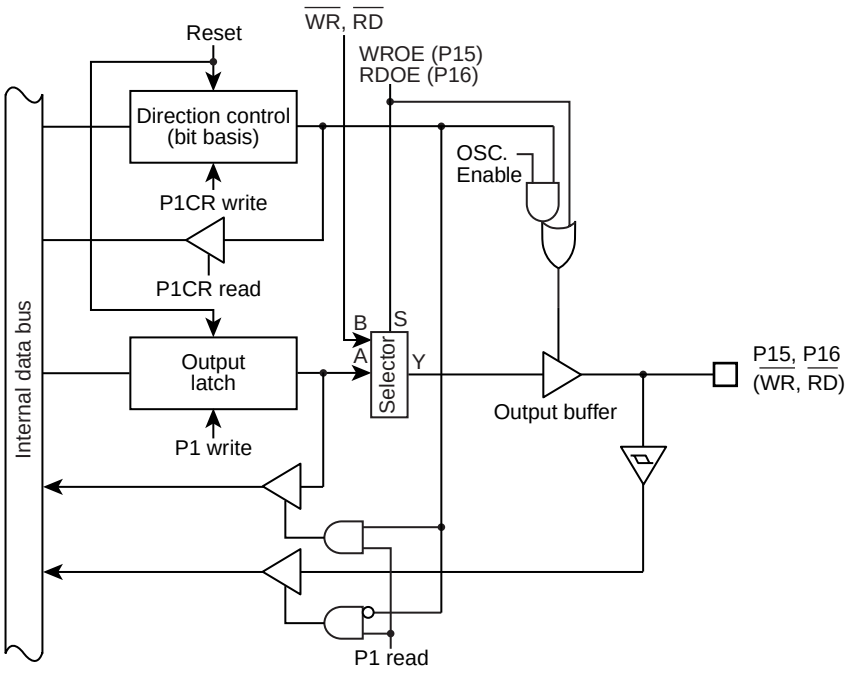


Figure 5-6 P15, P16

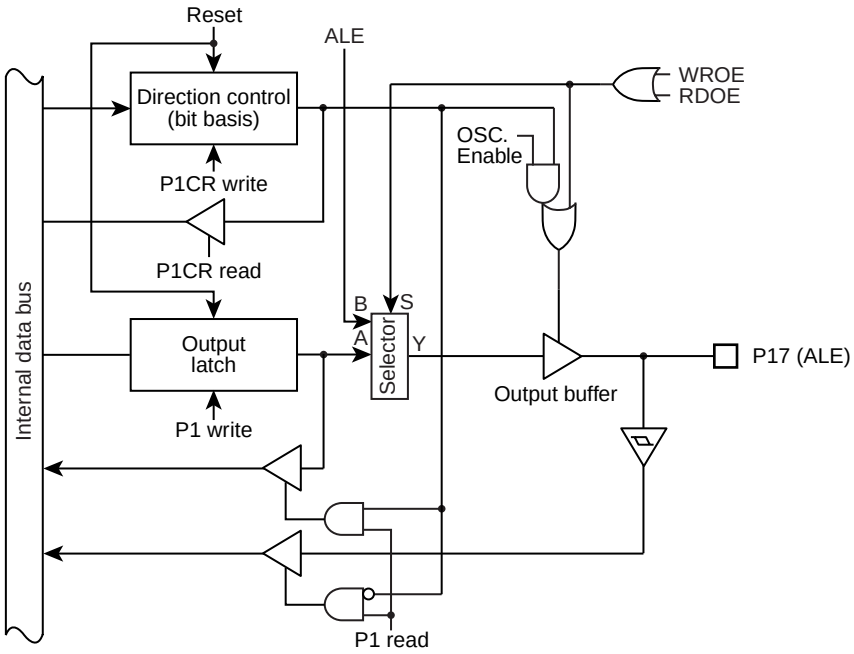


Figure 5-7 P17

Port P1DR and P1CR register

P1DR (0001H)

7	6	5	4	3	2	1	0	
P17	P16	P15	P14	P13	P12	P11	P10	(Initial value: 0000 0000)

P1CR (0009H)

7	6	5	4	3	2	1	0	
P1CR7	P1CR6	P1CR5	P1CR4	P1CR3	P1CR2	P1CR1	P1CR0	(Initial value: 0000 0000)

P1CR	I/O control for port P1 (Set for each bit individually)	0: Input mode 1: Output mode	R/W

5.3 Port P2

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 Xtal connection pins. When they are used as an input port or a secondary function pins, the respective output latch should be set to "1".

A low-frequency Xtal (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.

When a read instruction is executed for port P2, bits 7 to 3 are read as undefined values.

P2 port output latch (P2DR) and P2 port terminal input (P2R) are located on their respective addresses. Therefore, if input and output pins are mixed in P2 port, Read Write Modify instructions do not affect the output latch belongs to the terminal for input.

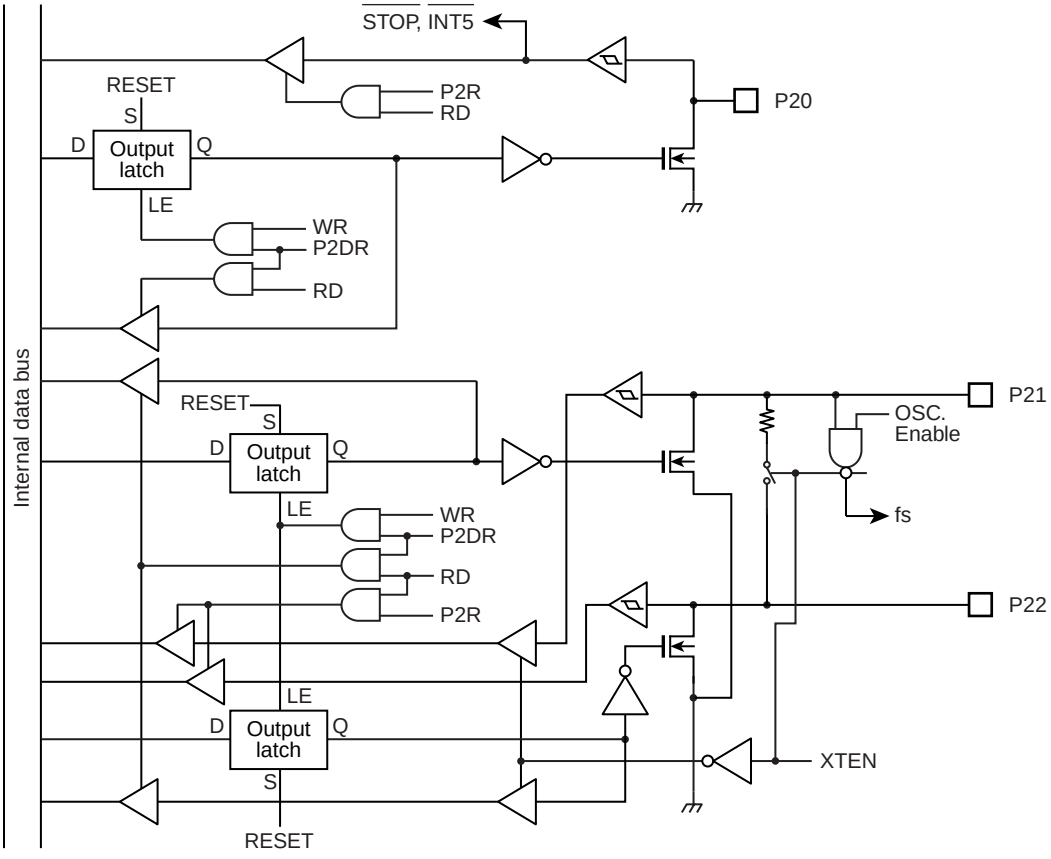


Figure 5-8 Port P2

Port P2

P2DR (0002H)

7	6	5	4	3	2	1	0	
*	*	*	*	*	P22	P21	P20	(Initial value: 1111 1111)

P2R (000DH)

7	6	5	4	3	2	1	0	
*	*	*	*	*	P22IN	P21IN	P20IN	(Initial value: 1111 1---) Read-only

Note 1: 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 2: Bit 7 through bit 3 in P2R contain unstable values.

Note 3: *: Don't care.

5.4 Port P3

Port P3 is the 8-bit I/O port that allows selection of input/output on bit basis. Input/output mode is specified on the P3 port input/output control register (P3CR). During reset, all the bits on P3CR are initialized to "0" and P3 becomes input port. Reset operation also initializes all the bits on P3 port output latch (P3DR) to "0".

Besides input/output port, the terminals in Port P3 functions as follows.

P36 and P37 have IC0 and INT2 function respectively. In order to utilize these functions, the terminal should be set for input.

P35 and P37 have TO0 and OC0 function respectively. In order to utilize these functions, the output latch belongs to each terminal should be set to "1" before the terminal is set for output.

Port P3 also functions as address bus (A15 to 8). Port P3 becomes 8-bit bidirectional address bus (A15 to 8) when the CPU accesses to the external memory. In order to utilize address bus (A15 to 8) function, set ABUSEN, located on the EXPCR.

Note: Input status is read while the port is input mode. Therefore the contents of output latch, that belongs to the terminal for input, may alter if both input and output are mixed in P3 port.

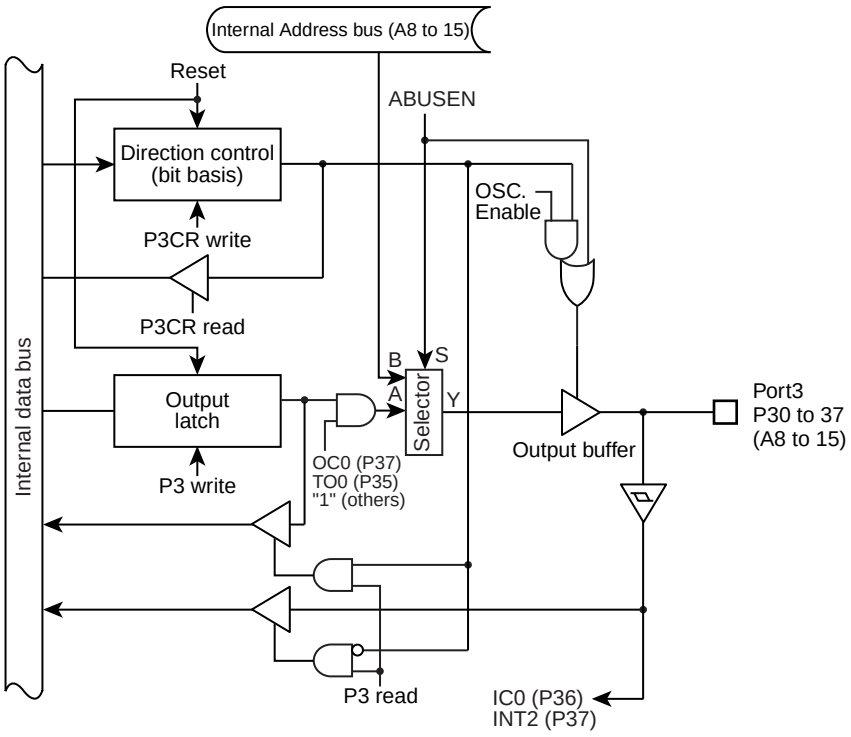


Figure 5-9 Port P3

Port P3

P3DR (0003H)

7	6	5	4	3	2	1	0	
P37	P36	P35	P34	P33	P32	P31	P30	(Initial value: 0000 0000)

P3CR (000AH)

7	6	5	4	3	2	1	0	
P3CR7	P3CR6	P3CR5	P3CR4	P3CR3	P3CR2	P3CR1	P3CR0	(Initial value: 0000 0000)

P3CR	I/O control for port P3 (Set for each bit individually)	0: Input mode 1: Output mode	R/W

5.5 Port P4

Port P4 is the 8-bit I/O port that allows selection of input/output on bit basis. Input/output mode is specified on the P4 port input/output control register (P4CR). During reset, all the bits on P4CR are initialized to “0” and P4 becomes input port. Reset operation also initializes all the bits on P4 port output latch (P4DR) to “1”.

Port P4 has programmable open-drain output function. The data on P4 port open-drain control register (P4ODE) determines whether open-drain output is enabled or disabled on bit basis. During reset, all the bits on P4ODE are initialized to “0”, and under the circumstances P4 becomes CMOS output port if P4CR is set to “1”.

Besides input/output port, the terminals in Port P4 functions as follows.

P40 and P41 have TI0 and TI1 function respectively: both TI0 and TI1 are for 8-bit timers. In order to utilize these functions, the terminal should be set for input.

P46 and P47 have INT3 and INT4 function respectively: both INT3 and INT4 are for interrupts. In order to utilize these functions, the terminal should be set for input.

The bundles, one consists of P42, P43 and P44 and the other consists of P46 and P47, have serial interface function respectively.

P4 output latch (P4DR) and P4 port terminal input (P4R) are located on their respective addresses. Therefore, if input and output pins are mixed in P4 port, Read Write Modify instructions do not affect the output latch belongs to the terminal for input.

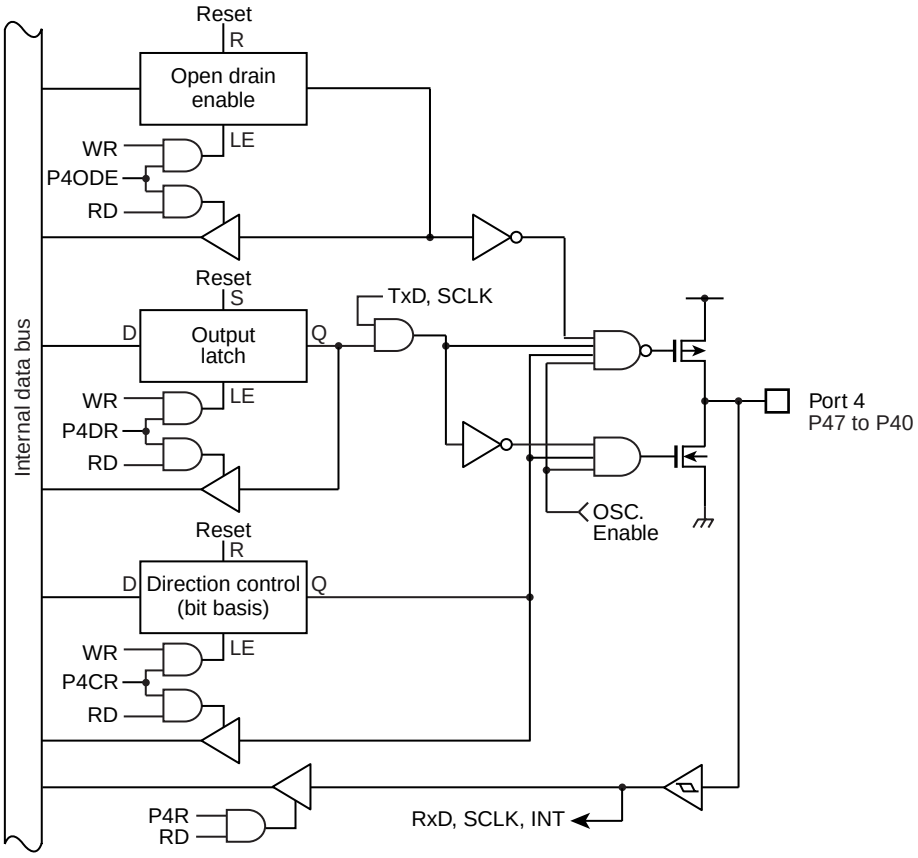


Figure 5-10 Port P4

Port P4

P4DR (0004H)

7	6	5	4	3	2	1	0
P47	P46	P45	P44	P43	P42	P41	P40

(Initial value: 1111 1111)

P4R (000EH)

7	6	5	4	3	2	1	0
P47IN	P46IN	P45IN	P44IN	P43IN	P42IN	P41IN	P40IN

(Initial value: ---- ----)
Read only

P4CR (000BH)

7	6	5	4	3	2	1	0
P4CR7	P4CR6	P4CR5	P4CR4	P4CR3	P4CR2	P4CR1	P4CR0

(Initial value: 0000 0000)

P4CR	I/O control for port P4 (Set for each bit individually)	0: Input mode 1: Output mode	R/W

P4ODE (000CH)

7	6	5	4	3	2	1	0
P4ODE7	P4ODE6	P4ODE5	P4ODE4	P4ODE3	P4ODE2	P4ODE1	P4ODE0

(Initial value: 0000 0000)

P4ODE	I/O control for port P4 (Set for each bit individually)	0: 3-state output mode 1: Nch O.D. output mode	R/W

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

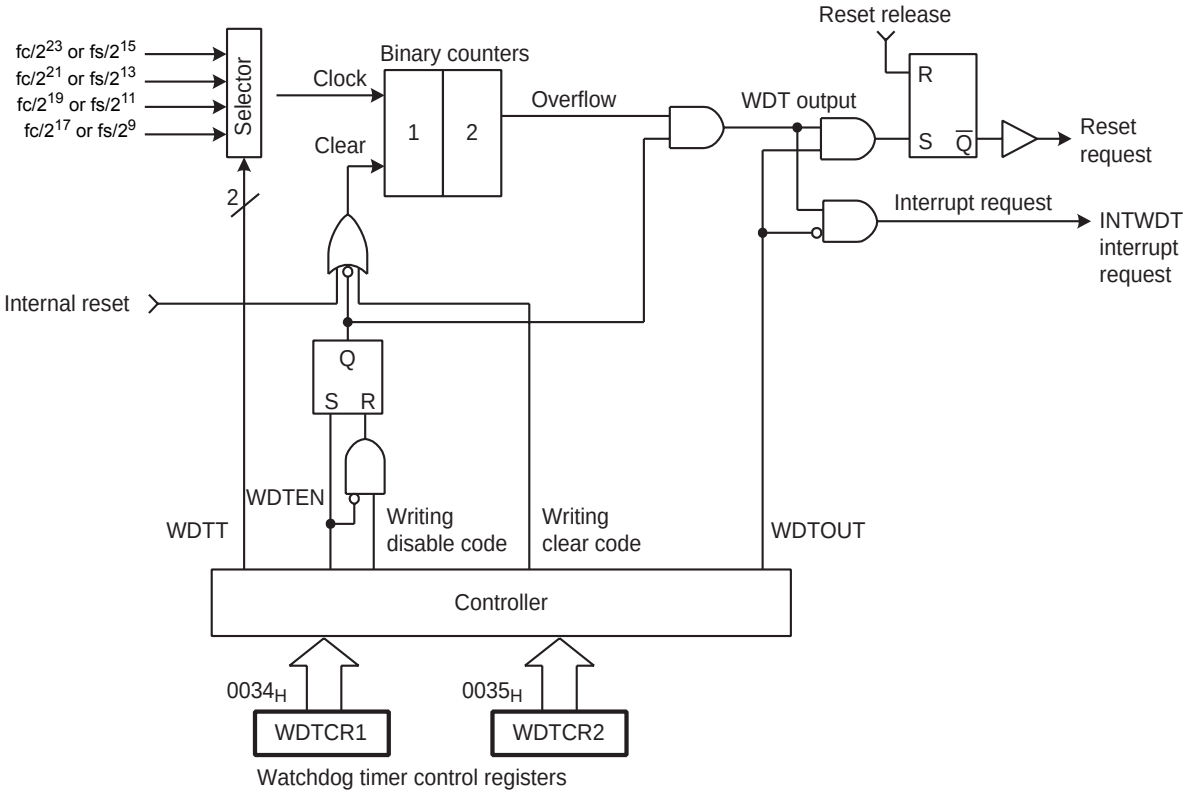


Figure 6-1 Watchdog Timer Configuration

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 the $\overline{\text{RESET}}$ pin outputs a low-level signal, 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
:                            detection time
LD    (WDTCR2), 4EH          : Clears the binary counters.
:                            Within 3/4 of WDT
:                            detection time
LD    (WDTCR2), 4EH          : Clears the binary counters.
```

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.) 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 1: Reset request				Write only

Note 1: After clearing WDTOUT to “0”, the program cannot set it to “1”.

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

Note 3: WDTCR1 is a write-only register and must not be used with any of read-modify-write instructions. If WDTCR1 is read, a don’t care is read.

Note 4: To activate the STOP mode, disable the watchdog timer or clear the counter immediately before entering the STOP mode. After clearing the counter, clear the counter again immediately after the STOP mode is inactivated.

Note 5: To clear WDTEN, set the register in accordance with the procedures shown in “1.2.3 Watchdog Timer Disable”.

Watchdog Timer Control Register 2

WDTCR2 (0035H)

7	6	5	4	3	2	1	0

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

WDTCR2	Write Watchdog timer control code	4EH: Clear the watchdog timer binary counter (Clear code) B1H: Disable the watchdog timer (Disable code) D2H: Enable assigning address trap area 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 coutner
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, 023FH           : 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 $\overline{\text{RESET}}$ pin outputs a low-level signal and 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.

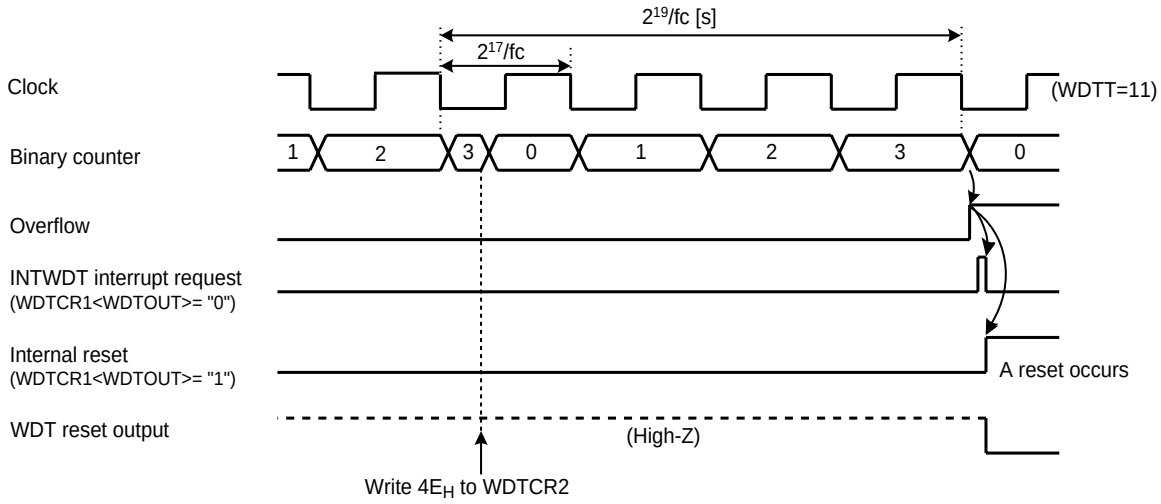


Figure 6-2 Watchdog Timer Interrupt/Reset

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 1: Generate address traps (After setting ATAS to "1", writing the control code D2H to WDTCR2 is reguired)	Write only
ATOUT	Select opertion at address trap	0: Interrupt request 1: Reset request	

Watchdog Timer Control Register 2

WDTCR2 (0035H)	7	6	5	4	3	2	1	0

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

WDTCR2	Write Watchdog timer control code and address trap area control code	D2H: Enable address trap area selection (ATRAP control code) 4EH: Clear the watchdog timer binary counter (WDT clear code) B1H: Disable the watchdog timer (WDT disable code) 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 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") 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 a watchdog timer interrupt is already accepted, the new address trap is processed immediately and the previous interrupt is held pending. Therefore, if address trap interrupts are generated continuously without execution of the RETN instruction, too many levels of nesting may cause a malfunction of the microcontroller.

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

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") or the SFR area, address trap reset will be generated.

When an address trap reset request is generated, the $\overline{\text{RESET}}$ pin outputs a low-level signal and 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

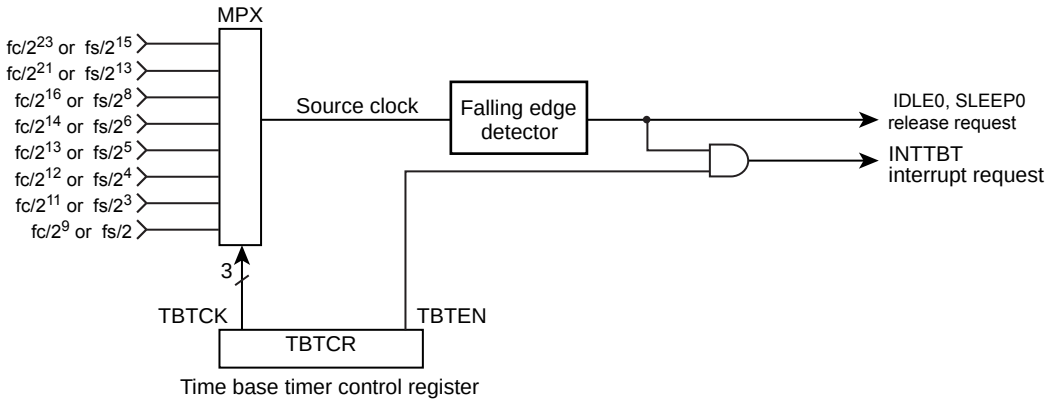


Figure 7-1 Time Base Timer configuration

7.1.2 Control

Time Base Timer is controled 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 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) . 6
```

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 generato which is selected by TBTCK.) after time base timer has been enabled.

The divider is not cleared by the program; therefore, only the first interrupt may be generated ahead of the set interrupt period (Figure 7-2).

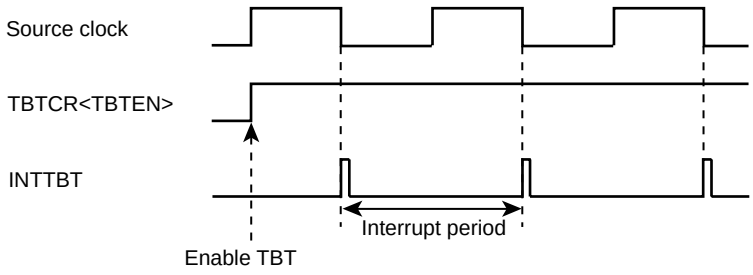


Figure 7-2 Time Base Timer Interrupt

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

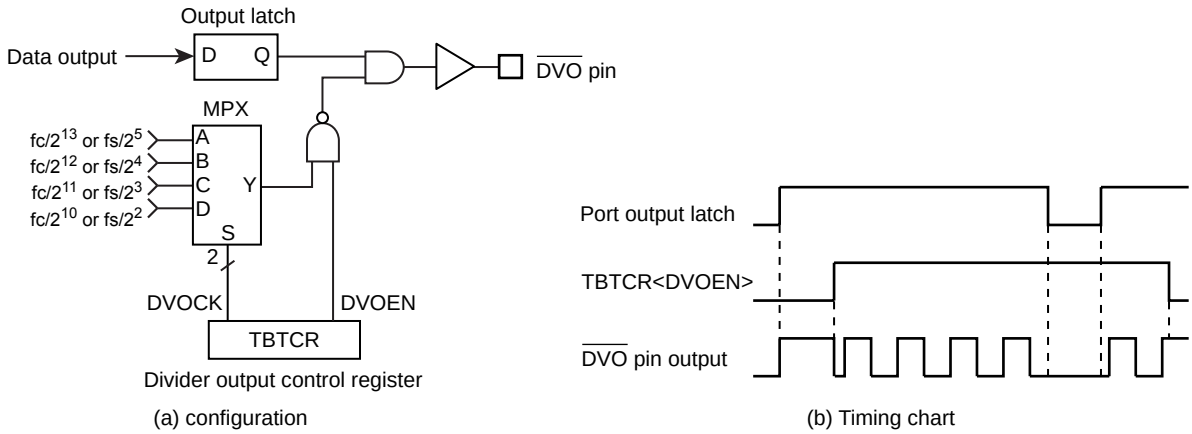


Figure 7-3 Divider Output

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 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. Extended Timer-Counter (ETC0)

The TMP86CH06AUG contains 16-bit extended timer-counter (ETC0), which is accompanied by terminals for capturing input and outputting comparison. The alternative of internal or external input is to be source clock.
The ETC0 pin, IC0 pin and OC0 pin can also be used as the port P12, P36 and P37.

8.1 Configuration

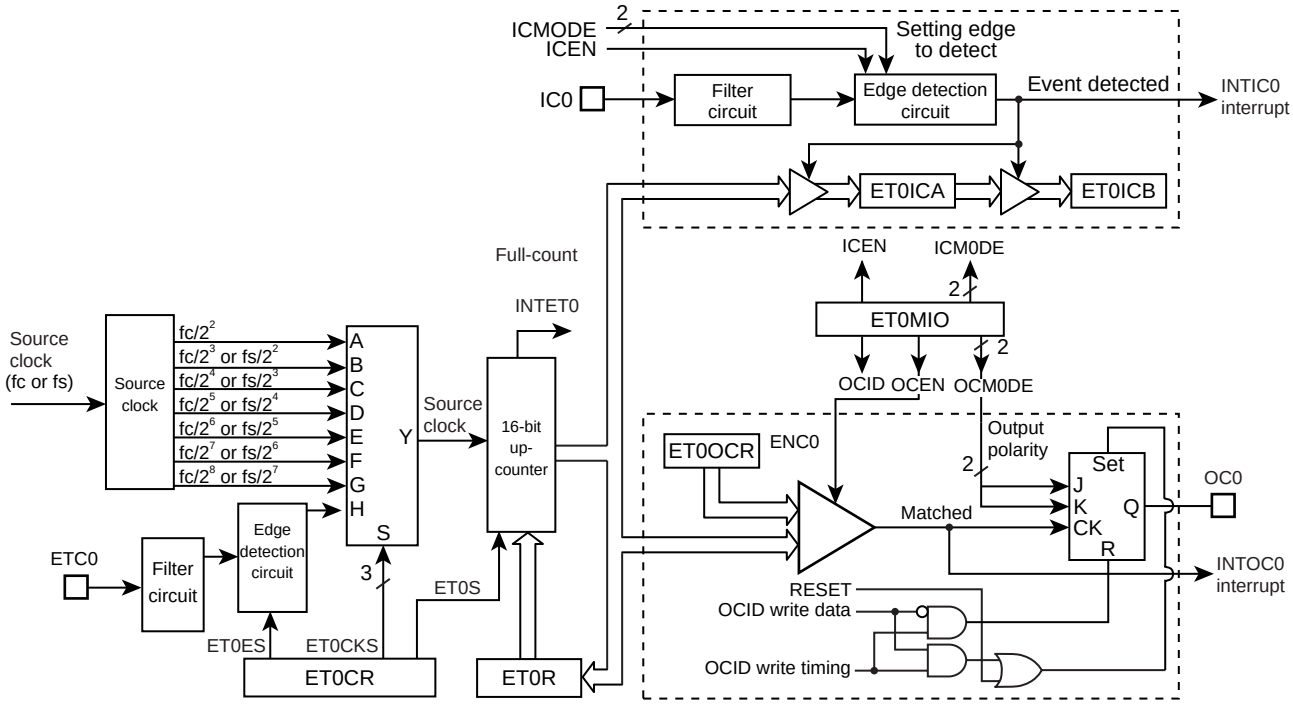


Figure 8-1 Extended Timer-Counter 0 (ETC0)

8.2 Controlling

The extended timer-counter (ETC0) is controlled on the following registers.

Extended Timer-Counter Register

This 16-bit timer register is to set and record the ETC0 counter value. It is able to read while counting.

ET0R (0012, 0013H)	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	ET0RH (0013H)								ET0RL (0012H)							

Read/Write (Initial value: 0000 0000 0000 0000)

Note 1: Data written in the counter, does not reach lower byte of the ETC0 register until data on upper byte of it is set because of buffering. Writing only to lower byte is not allowed.

Note 2: On reading the register, the lower byte read operation latches the timer-counter value. Therefore, always read the lower byte first, and the upper byte last.

Note 3: Writing FFFFH to the ETC0 register generates an immediate interrupt request signal.

Note 4: The lower byte of the ETC0 register and the lower byte of the ETC0 compare register share their buffering register. Therefore, do not execute write instruction against the ETC0 compare register, between writing on lower byte of the ETC0 register and upper byte of it.

External Timer-counter Control Register

Principal mode register for ETC0

ET0CR (0010H)	7	6	5	4	3	2	1	0
	ET0S				ET0ES	ET0CKS		

(Initial value: 0*** 0000)

ET0S	ETC0 start/stop	0: Stop 1: Start			R/W
ET0ES	Count edge selection for event counter mode	0: Rising edge 1: Falling edge			
ET0CKS	Source clock for ETC0		NORMAL1/2, IDLE1/2 mode	SLOW, SLEEP mode	
		000	$fc/2^2$	–	
		001	$fc2^3$	$fs/2^2$	
		010	$fc/2^4$	$fs/2^3$	
		011	$fc/2^5$	$fs/2^4$	
		100	$fc/2^6$	$fs/2^5$	
		101	$fc/2^7$	$fs/2^6$	
		110	$fc/2^8$	$fs/2^7$	
		111	External clock (event counter mode)		

Note 1: *: Don't care

Note 2: Altering source clock(ET0CK) and edge type under event counter mode is allowed only if the extended timer has been stopped. (Do not to change the setting on from enabling status to disabling status.)
They can be set concurrently with enabling (start instruction).

Pulse Measure Enable Register

Mode register for capturing in order to calculate time difference or effective counting

PME (0016H)	7	6	5	4	3	2	1	0
								PME

(Initial value: **** ***0)

PME	edge which causes interrupt request	0: of earliest detection 1: of second earliest detection	Write only

Note 1: If the address, PME locates, is read, the lower bits on ET0ICB is extracted.

Note 2: As writing on PME by software, only the data "1" is accepted. PME is cleared to "0" by hardware when the edge is detected or the device is totally reset.

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

ETC0 Capture/Compare Mode Register

The register for both capturing and comparing

ET0MIO (0011H)

7	6	5	4	3	2	1	0
OCIDEN	ICEN	ICMODE		OCID	OCEN	OCMODE	

(Initial value: 1000 1000)

OCIDEN	Assignment control for OCID to initialize	0: Disable 1: Enable	R/W
ICEN	Capturing channel assignment control	0: Disable (used for port) 1: Enable (used for capturing channel)	
ICMODE	Capturing mode Edge control	00: No detection 01: Detect rising edge 10: Detect falling edge 11: Detect both edges	
OCID	Initial value for comparator (Set to "1" after RESET)	0: Initialize to “0” 1: Initialize to “1” * Valid when OCIDEN = “1”	
OCEN	Comparing channel assign-ment control	0: Disable (used for port) 1: Enable (used for comparing channel)	
OCMODE	Comparator output mode polarity control	00: NOP, output unchanged (steady output) 01: “1” output 10: “0” output 11: TOGGLE output (invert output)	

ETC0 Capture Register A/B

These 16-bit registers record the time when the capture terminal input changed. At maximum, 2 latest data are recorded; after detection, the capture register A holds the latest.

ET0ICA (0014, 0015H)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ET0ICAH (0015H)								ET0ICAL (0014H)							

Read only (Initial value: 0000 0000 0000 0000)

ET0ICB (0016, 0017H)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ET0ICBH (0017H)								ET0ICBL (0016H)							

Read only (Initial value: 0000 0000 0000 0000)

Note: Since capturing is restored by reading the upper byte of the capture register A, read capture register B before reading capture register A. Read capture register A from lower byte to upper byte.

ETC0 Compare Register

The 16-bit register is to set the time for changing output on comparator.

ET0OCR (0018, 0019H)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ET0OCRH (0019H)								ET0OCRL (0018H)							

Read/Write (Initial value: 0000 0000 0000 0000)

Note 1: Data written in the counter, does not reach lower byte of the ETC0 compare register until data on upper byte of it is set because of buffering. Writing only to lower bite is not allowed.

Note 2: The lower byte of the ETC0 register and the lower byte of the ETC0 compare register share their buffering register. Therefore, do not execute write instruction against the ETC0 register, between writing on lower byte of the ETC0 compare register and upper byte of it.

8.3 Source Clock

The source clock for the extended timer-counter (ETC0) has 2 sorts of operating mode: the free-running mode whose source clock is provided from internal clock, and event-counter mode whose source clock is provided from external clock (through terminal ETC0).

Table 8-1 Source Clock and Accuracy for ETC0 (at fc = 8 MHz, fs = 32.768 kHz)

Operating Mode	Source Clock	Edge	Accuracy	
			(fc)	(fs)
Free Running Timer	Internal	Rising	500 ns to 32 μs	122 μs to 3.91 ms
Event Counter	ETCO terminal	Rising or falling	1 μs or more	244 μs or more

8.3.1 Free Running Timer Mode

The internal clock increase the extended timer-counter (ETC0). The interrupt INTET0 is requested when the counter reaches FFFFH, meanwhile the counter is still counting up. The source clock is selected on the bit ETC0CR<ET0CKS>. The counter becomes event-counter mode if ET0CR<ET0CKS> = “111”. In order to generate interrupt after a certain period, subtract a rate from FFFFH and set its result on the ETC0 register (ET0R). The ET0R is cleared if 0000H is written on ETC0R. Both writing and reading against the ET0R are available, while the counter is in motion or halt. By executing read instruction against the ET0R, the current rate on the counter is extracted. Notice that reading and writing against the ET0R should be done from lower byte to upper byte consecutively.

Since the counter extract the whole data for the ET0R as data is written on upper byte, the exclusive writing against lower byte of ET0R is not valid. As for reading, data on the counter is latched on reading against lower byte.

Example 1 :Generate interrupt of ETC0 in 8 s, from source clock of fs/4 (fs = 32.768 kHz)

```
LD      (ET0CR), 00000001B      ; Stop the counter, and assign fs/2^2 to source clock
LD      (ET0R), 0000H           ; Set timer register (8 s ÷ 122 μs = FFFFH)
DI
SET     (EIRH).2                ; Enable interrupt INTET0
EI
SET     (ET0CR).7               ; ETC0 start
```

Example 2 :Generate interrupt of ETC0 in 256 μs, from source clock of fc/8 (fc = 8 MHz)

```
LD      (ET0CR), 00000001B      ; Stop the counter, and assign fc/8 to source clock
LDW     (ET0R), 0FEFFH          ; Set timer register (FFFFH−100H)
DI
SET     (EIRH).2                ; Enable interrupt INTET0
EI
SET     (ET0CR).7               ; ETC0 start
```

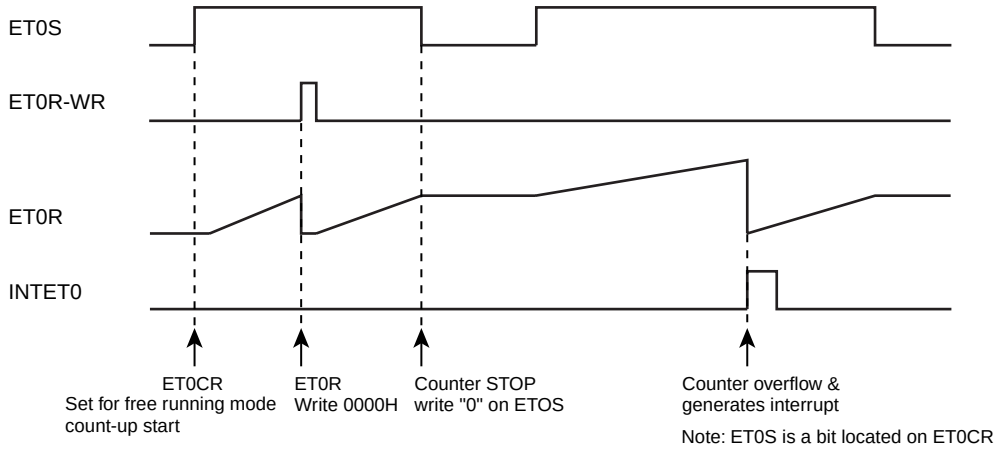


Figure 8-2 Example for Free-running mode operation

Table 8-2 ETC0 Internal clock (fc = 8 MHz, fs = 32.768 kHz)

ET0CKS	NORMAL 1/2, IDLE 1/2 mode		SLOW, SLEEP mode	
	Accuracy	Range	Accuracy	Range
000	500 ns	32.77 ms	–	–
001	1 μs	65.54 ms	122.0 μs	8 s
010	2 μs	131.07 ms	244.1 μs	16 s
011	4 μs	262.14 ms	488.3 μs	32 s
100	8 μs	524.29 ms	976.6 μs	64 s
101	16 μs	1.05 s	1.953 ms	128 s
110	32 μs	2.10 s	3.906 ms	256 s

Note: ET0CKS is a group of bits, which manages the source clock of ETC0.

8.3.2 Event-counter Mode

The edge of ETC0 terminal input increase the extended timer-counter (ETC0). The counter becomes event-counter mode if ET0CR<ET0CKS> = “111”.

In order to utilize event-counter mode, set ETC0 terminal to input mode on P1CR. The type of capturing edge, either rising or falling, can be selected on ET0CR<ET0ES>. The counter operates similar to free running counter mode, except for the source clock: internal clock or terminal input.

Example :Generate interrupt of ETC0 after counting 100 (64H) times of rising edge

```
CLR     (ET0CR). 7      ; stops the counter
CLR     (P1CR). 2       ; sets P12 (ETC0) for input
LD      (ET0CR), 07H    ; event counter mode, detecting rising edge
LDW     (ET0R), 0FF9BH  ; sets timer register (FFFFH to 0064H)
DI
SET     (EIRH). 2       ; enables interrupt INTET0
EI                      ;
SET     (ET0CR). 7      ; starts the counter
```

Note: Altering source clock and edge type under event counter mode is allowed only if the extended timer has been stopped.

Since ETC0 input terminal has digital noise cancellor, only pulse width of 2 machine cycles or longer is accepted. Pulse width of 1 machine cycle or shorter is neglected. It is unstable to acknowledge pulse width of between 1 and 2, as valid input.

(input pulse width) < (1 machine cycle) → not counted

(1 machine cycle) ≤ (input pulse width) < (2 machine cycles) → counted or not counted (depend on timing)

(2 machine cycle) ≤ (input pulse width) → counted

4/fc = 1 machine cycle (0.5 μs at fc = 8 MHz)

After detecting a rising (falling) edge, it is required to detect falling (rising) edge before detecting another rising (falling) edge.

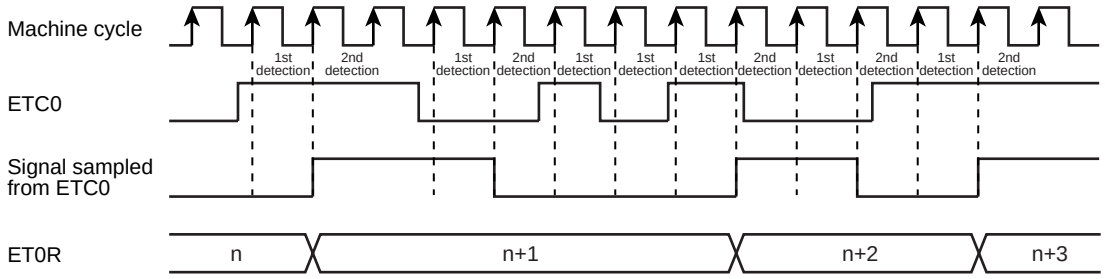


Figure 8-3 Edge Filtering in Event Counter Mode (Rising Edge Detection)

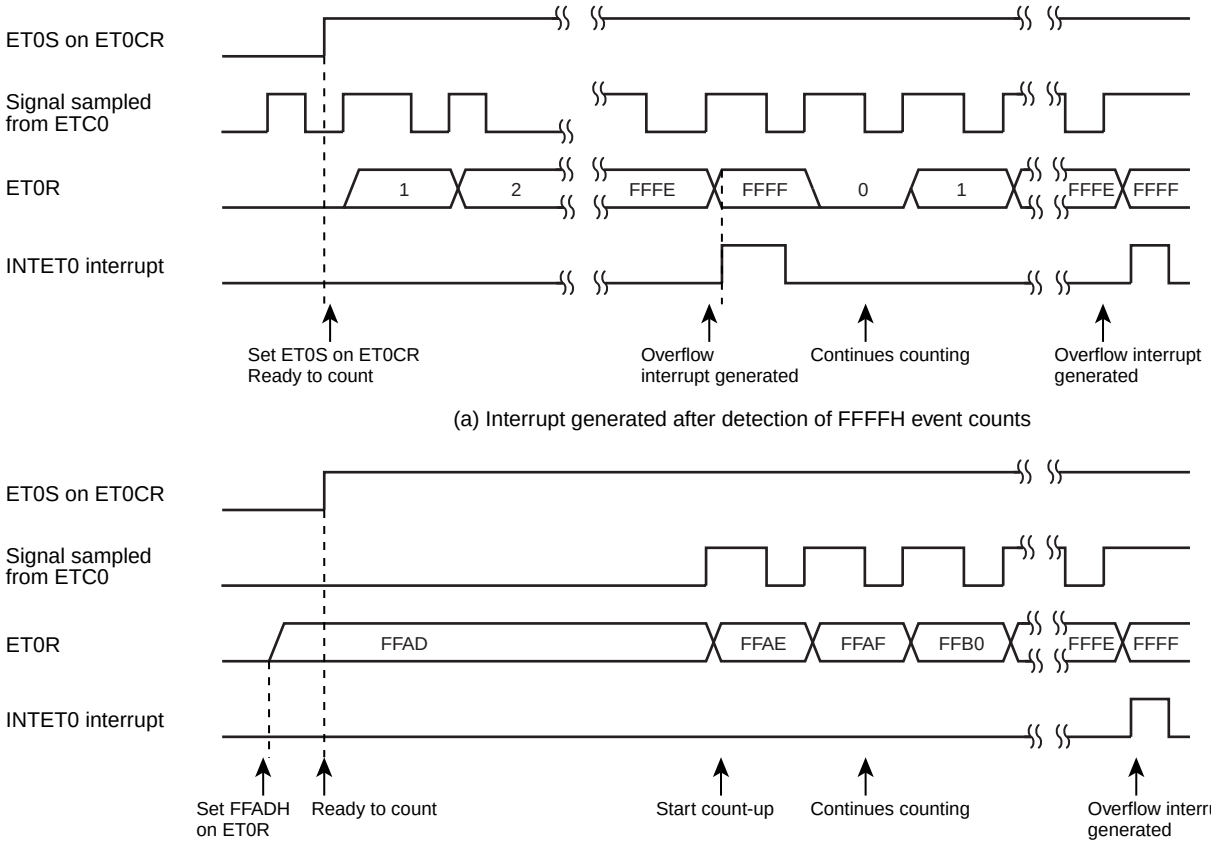


Figure 8-4 Example of Event Counter Mode Operation (ETC0 Rising Edge Detection)

8.4 Capturing input, Output comparing

The extended timer-counter (ETC0) involves terminals for capturing input IC0 (P36) and output comparing OC0 (P37). Both IC0 and OC0 are available for general input/output ports if they are not involved in the extended timer-counter (ETC0).

Table 8-3 Capturing input and Output comparing Functions

	Shared function	Operation	Accuracy	Data register
Capture input	P36	Rising edge Falling edge Both edges	Depend on timer	2
Output compare	P37	"1" output "0" output toggle NOP		1

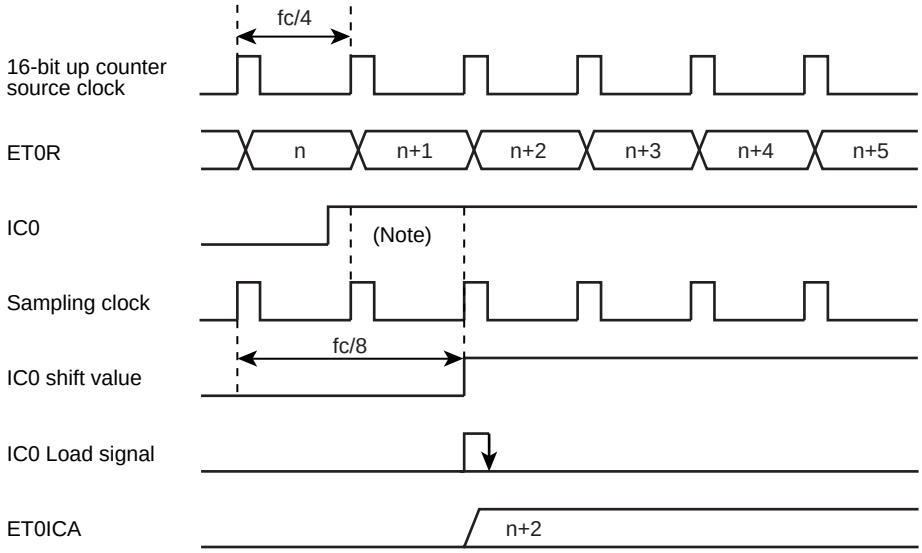
When capturing input function is to be used, assign input mode to P36 (set "0" on bit6 on P3CR). When output comparing function is to be used, assign output mode to P37 (set "1" on bit7 on P3CR) after P37 output latch (bit7 on P3DR) is set to "1".

8.4.1 Capturing input

It is the function to measure matters, such as pulse width, frequency or duty.

At the time the capturing acknowledged input changed, the contemporary rate on the extended timer-counter (ETC0) is loaded on the capture register. The capturing contains a digital noise-cancellor circuit in its block. A stable pulse of fc/8 or longer is required in order to inform the hardware of input, otherwise the pulse would not be acknowledged normally. Since the rate on the extended timer-counter (ETC0) is loaded on the capture register after the capturing samples every fc/4 with sampling clock, there is a lag of fc/4 to fc/8 between terminal input and capture register record. After detecting edge, the following capture operation is prohibited, until the data on the capture register ET0ICA. As for reading ET0ICA, read lower byte first then read upper byte.

The capturing operates normally regardless of overflow of the counter, since the counter is still counting-up after overflow.



Note: Sampling clock for IC0 input is fc/4, regardless of its source clock.

Figure 8-5 Example of Event Counter Mode Operation
(IC0 rising edge detection, Source clock of fc/4)

Example 1 :Capturing IC0 on rising edge with ETC0

```
        CLR     (P3CR).6            ; Assigns input mode to P36
        LD      (ET0MIO), 0101****B ; Enables capturing on rising edge
```

Example 2 :Capturing positive pulse width IC0, with ETC0 (without using ET0ICB)

```
        CLR     (P3CR). 6           ; Assigns input mode to P36
        DI
        SET     (EIRH). 3           ; Enables individual interrupt enable flag for INTIC0
        EI
        LD      (ET0MIO), 0111****B ; Enables capturing on both edges
LP:     TEST    (ILH). 3            ; Waits until IC0 rises (IL11 rises)
        JR      T, LP
        DI
        LD      (ILH), F7H          ; Clears IL11
        EI
        LD      A, (ET0ICAL)        ; Reads ET0ICA
        LD      W, (ET0ICAH)
LP2:    TEST    (ILH). 3            ; Waits until IC0 falls (IL11 rises)
        JR      T, LP2
        SUB     WA, (ET0ICA)        ; Calculates pulse width and loads it on HL register
        LD      HL, 0000H
        SUB     HL, WA
        DI
        LD      (ILH), F7H          ; Clears IL11
        EI
```

If the data “1” is set on register PME located on the address same as ET0ICBL, the interrupt INTIC0 is requested as the second valid edge is detected. As this interrupt is requested, the first detected time (loaded on ET0ICB) and the second (loaded on ET0ICA) can be read consecutively. PME is cleared to “0” by hardware, after the edge is detected. Therefore, after the interrupt INTIC0 is requested, it is required to set “1” on PME again in order to continue extracting the 2-word data on capture register. As the second detected time (loaded on ET0ICA) is read, the capture operation is enabled again.

As for extracting the 2-word data on capture register, read the first detected time (loaded on ET0ICB) before reading the second (loaded on ET0ICA). In order to extract the 2-word data normally, the data “1” should be set on PME before the first edge detection, if the capture operation has already been enabled.

Example 3 :Calculate (Second risen time) - (First risen time) and then calculate (Fourth risen time) - (Third risen time), as for IC0 input

```
        CLR     (P3CR).6            ; Assigns IC0 to P36
        SET     (PME).0             ; Rises PME (interrupt after second edge detection)
        DI
        SET     (EIRH).3            ; Enables individual interrupt enable flag for INTIC0
        LD      (ET0MIO), 0101----B ; Enables capturing on rising edge
        EI
LP:     JR      LP
IC0:    SET     (ET0ICBL).0         ; Raises PME (interrupt after second edge detection)
        LD      HL, (ET0ICBL)       ; Reads the first (third) detected time
        LD      BC, (ET0ICAL)       ; Reads the second (fourth) detected time,
                                    ; Enables capturing again
        SUB     BC, HL              ; Calculates time
        RETI
```

8.4.2 Output Comparing

The output level alters at the time stated. The output mode is selected on the bits ET0MIO<OCMODE> whether to be output “1”, output “0”, invert or NOP.

The initial value that the comparator outputs is set on bit ET0MIO<OCID>. Once the comparator is enabled, namely during comparison, bit ET0MIO<OCIDEN> should be cleared to “0” for fear that other than data matching alter OC0 output.

The time to alter output is set on the ETC0 compare register (ETC0OCR). This register should be consecutively written from the lower byte to the upper. As the upper byte is set, the 2-byte data is loaded on the ETC0OCR effectively. Writing only to the lower byte is prohibited.

As the rate of the timer-counter and comparator matches, the stated data depends on ET0MIO<OCMODE> is output on OC0, and the interrupt INTOC0 is requested.

If OCMODE has been set for NOP, the terminal keeps its value and only the interrupt request is required. If OCMODE has been “11”, the terminal inverts its output value.

Note: ET0MIO<OCMODE> set to "11", the comparator output OC0 value is toggled. While the timer-counter is not in motion (ET0CR<ET0S> = “0”), the comparator output is fixed "1".

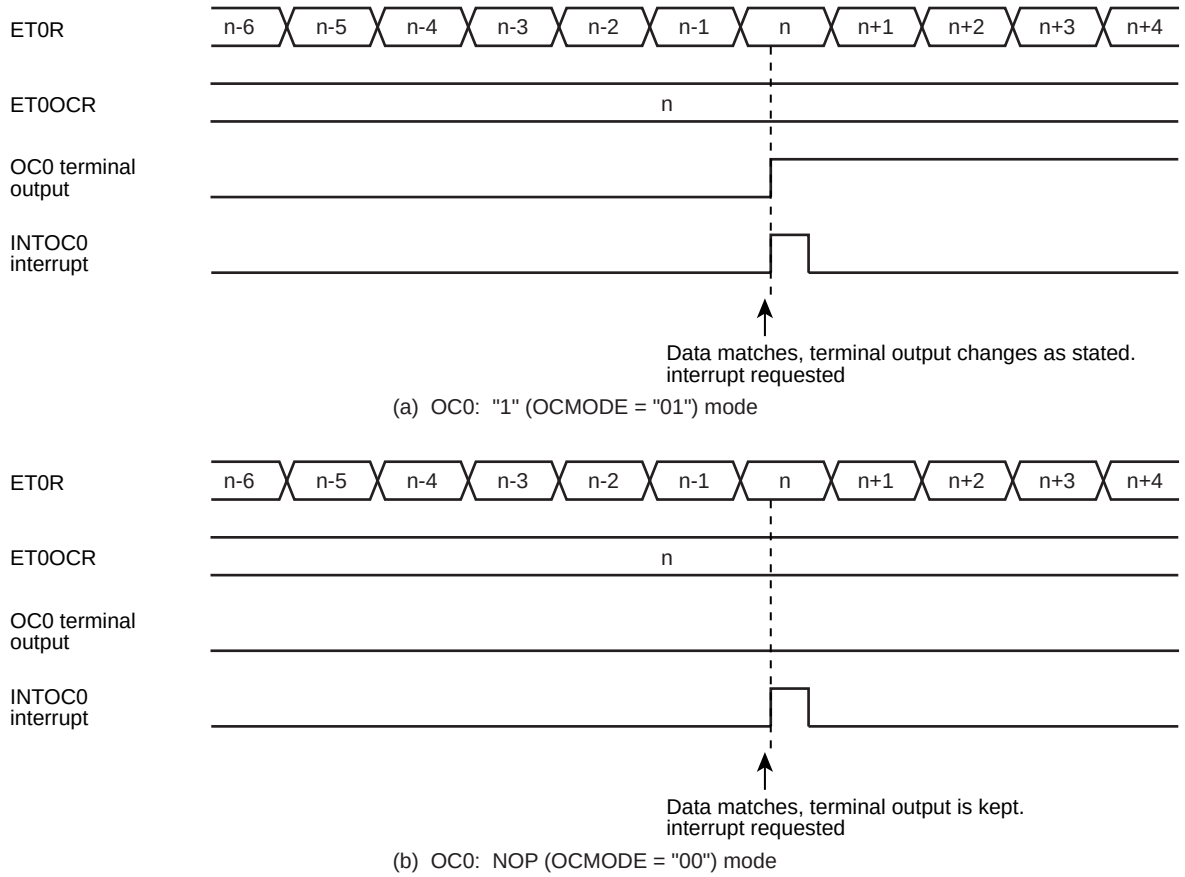


Figure 8-6 Examples for comparing operation

Example 1 :Change OC0 output from “0” to “1” when ET0R = 0C80H. (“1” output mode)

```
SET     (P3DR).7
SET     (P3CR).7            ; assigns OC0 output mode to P37
LDW     (ET0OCRL), 0C80H    ; sets time for changing output
LD      (ET0MIO), 1---0101B ; initializes OC0 to “0” and assign “1” output mode to it
CLR     (ET0MIO).7          ; prohibits OCID revising
```

Example 2 : Change OC0 output from “1” to “0” when ET0R = 0C80H. (toggle output mode)

```
SET     (P3DR).7
SET     (P3CR).7            ; assigns OC0 output mode to P37
LDW     (ET0OCRL), 0C80H    ; sets time for changing output
LD      (ET0MIO), 1---1111B ; initializes OC0 to “1” and assigns toggle output mode to it
CLR     (ET0MIO).7          ; prohibits OCID revising
```

Example 3 :Output the pulse with “0” for 1ms and “1” for 2ms, through OC0 terminal. (at fc = 8 [MHz])

```
                LD      (ET0CR), 01H            ; selects fc/8 (1 μs) for source clock
                LDW     (ET0R), 0000H           ; clears the counter
                LD      BC, 03E8H               ; substitutes “0” width (1 ms) for BC
                LD      DE, 07D0H               ; substitutes “1” width (2 ms) for DE
                LD      WA, 03E8H               ; substitutes “initial value” width (1 ms) for WA
                SET     (P3DR).7
                SET     (P3CR).7                ; assigns OC0 output mode to P37
                LD      (ET0OCRL), A
                LD      (ET0OCRH), W            ; sets time for changing output
                LD      (ET0MIO), 1---0111B     ; initializes OC0 to “0” and assigns toggle output mode to it
                CLR     (ET0MIO).7              ; prohibits OCID revising
                SET     (ET0CR).7               ; ETC0 starts counting
OUT0:
                ADD     WA, DE                  ; calculates time to change output from “1” to “0”
LP:
                TEST    (ILH).4                 ; waits for changing output from “0” to “1” (sets IL12)
                JR      T, LP
                DI
                LD      (ILH), 0EFH             ; clears IL12
                EI
                LD      (ET0OCRL), A
                LD      (ET0OCRH), W            ; sets time for changing output from “1” to “0”
OUT1:
                ADD     WA, BC                  ; calculates time to change output from “0” to “1”
LP1:
                TEST    (ILH).4                 ; waits for changing output from “1” to “0” (sets IL12)
                JR      T, LP1
                DI
                LD      (ILH), 0EFH             ; clears IL12
                EI
                LD      (ET0OCRL), A
                LD      (ET0OCRH), W            ; sets time for changing output from “0” to “1”
                JR      OUT0
```

Example 4 :Generate only interrupt request when ET0R = 0C80H: without OC0 terminal output.

```
                LDW     (ET0OCRL), 0C80H        ; sets time for interrupt request
                LD      (ET0MIO),1---1100B      ; initializes OC0 to “1” and assigns NOP output mode to it
                CLR     (ET0MIO).7              ; prohibits OCID revising
```

8.5 Interrupting

There are 3 sorts of interrupt requesting: for extended timer (INTET0), for capture input (INTIC0) and for output compare (INTOC0).

1. INTET0

   The interrupt requesting is generated as the counter reaches FFFFH; meanwhile the counter is still counting up.

2. INTIC0

   Interrupt requesting is triggered by capture input pin (IC0). The interrupt is requested when the timer value at edge detection is loaded on the capture register.

3. INTOC0

   Interrupt requesting is triggered by comparator. The interrupt is requested when the rate of the timer-counter and comparator matches.

9. 8-Bit TimerCounter (TC0, TC1)

9.1 Configuration

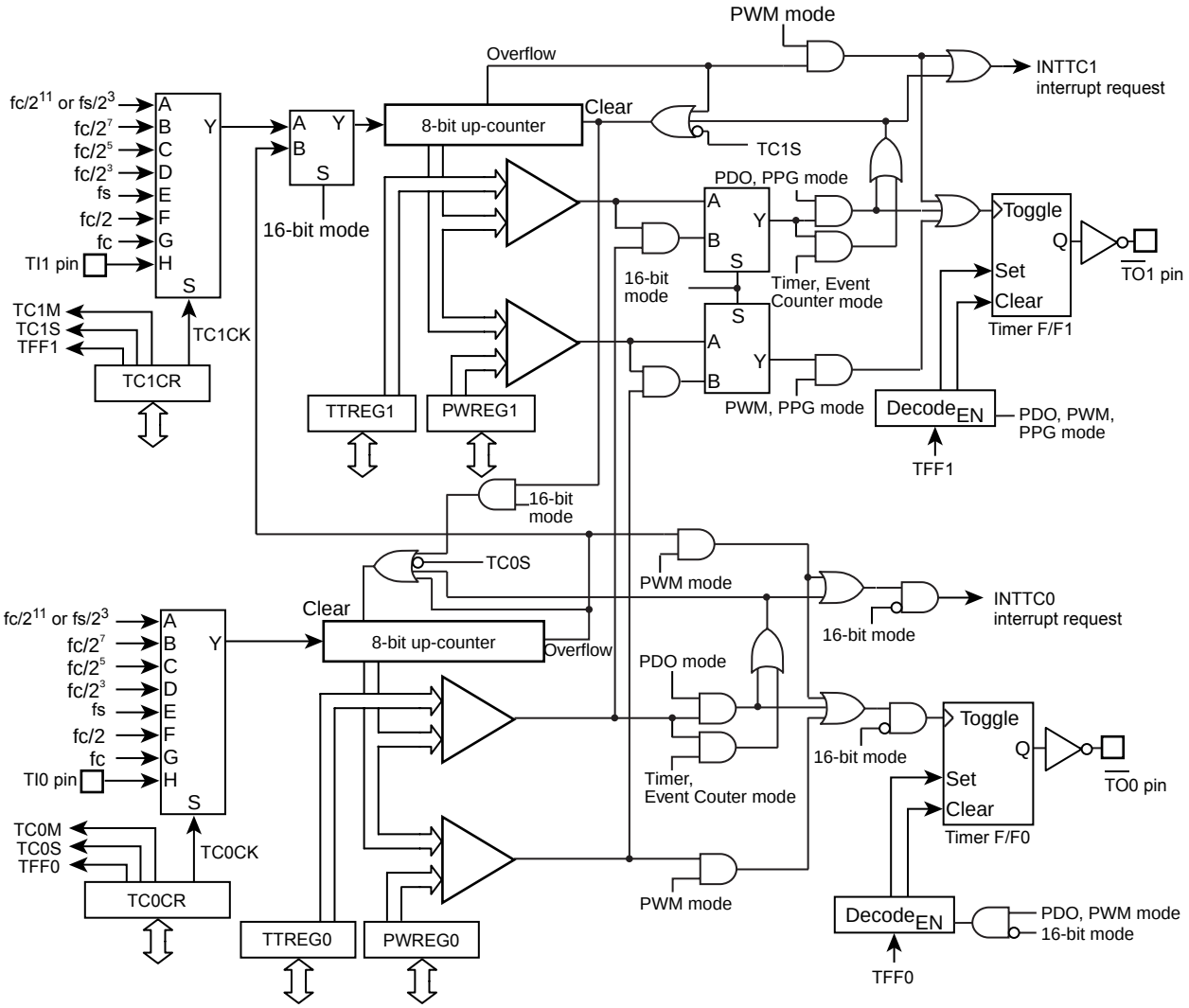


Figure 9-1 8-Bit TimerCouter 0, 1

9.2 TimerCounter Control

The TimerCounter 0 is controlled by the TimerCounter 0 control register (TC0CR) and two 8-bit timer registers (TTREG0, PWREG0).

TimerCounter 0 Timer Register

	7	6	5	4	3	2	1	0	
TTREG0 (0022H) R/W									(Initial value: 1111 1111)
PWREG0 (0024H) R/W									(Initial value: 1111 1111)

Note 1: Do not change the timer register (TTREG0) setting while the timer is running.

Note 2: Do not change the timer register (PWREG0) setting in the operating mode except the 8-bit and 16-bit PWM modes while the timer is running.

TimerCounter 0 Control Register

TC0CR (0020H)	7	6	5	4	3	2	1	0	
	TFF0	TC0CK			TC0S	TC0M			(Initial value: 0000 0000)

TFF0	Time F/F0 control	0: Clear 1: Set			R/W
TC0CK	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(Note 9)	fs(Note 9)	fs
		101	fc/2	fc/2	–
		110	fc	fc	fc (Note 8)
		111	TI0 pin input		
TC0S	TC0 start control	0: Operation stop and counter clear 1: Operation start			R/W
TC0M	TC0M operating mode select	000: 8-bit timer/event counter mode 001: 8-bit programmable divider output (PDO) mode 010: 8-bit pulse width modulation (PWM) output mode 011: 16-bit mode (Each mode is selectable with TC1M.) 1**: Reserved			R/W

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

Note 2: Do not change the TC0M, TC0CK and TFF0 settings while the timer is running.

Note 3: To stop the timer operation (TC0S= 1 → 0), do not change the TC0M, TC0CK and TFF0 settings. To start the timer operation (TC0S= 0 → 1), TC0M, TC0CK and TFF0 can be programmed.

Note 4: To use the TimerCounter in the 16-bit mode, set the operating mode by programming TC1CR<TC1M>, where TC0M must be fixed to 011.

Note 5: To use the TimerCounter in the 16-bit mode, select the source clock by programming TC0CK. Set the timer start control and timer F/F control by programming TC1CR<TC1S> and TC1CR<TFF1>, respectively.

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

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

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

Note 9: When used as NORMAL1 and IDLE1 modes (low frequency is disabled), "fs" can not be used as source clock.

The TimerCounter 1 is controlled by the TimerCounter 1 control register (TC1CR) and two 8-bit timer registers (TTREG1 and PWREG1).

TimerCounter 1 Timer Register

	7	6	5	4	3	2	1	0	
TTREG1 (0023H) R/W									(Initial value: 1111 1111)
PWREG1 (0025H) R/W									(Initial value: 1111 1111)

Note 1: Do not change the timer register (TTREG1) setting while the timer is running.

Note 2: Do not change the timer register (PWREG1) setting in the operating mode except the 8-bit and 16-bit PWM modes while the timer is running.

TimerCounter 1 Control Register

TC1CR (0021H)	7	6	5	4	3	2	1	0	
	TFF1	TC1CK			TC1S	TC1M			(Initial value: 0000 0000)

TFF1	Timer F/F1 control	0: Clear 1: Set			R/W
TC1CK	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(Note 10)	fs(Note 10)	fs
		101	fc/2	fc/2	–
		110	fc	fc	–
		111	TI1 pin input		
TC1S	TC1 start control	0: Operation stop and counter clear 1: Operation start			R/W
TC1M	TC1M operating mode select	000: 8-bit timer/event counter mode 001: 8-bit programmable divider output (PDO) mode 010: 8-bit pulse width modulation (PWM) output mode 011: Reserved 100: 16-bit timer/event counter mode 101: Warm-up counter mode 110: 16-bit pulse width modulation (PWM) output mode 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 TC1M, TC1CK and TFF1 settings while the timer is running.

Note 3: To stop the timer operation (TC1S= 1 → 0), do not change the TC1M, TC1CK and TFF1 settings.
To start the timer operation (TC1S= 0 → 1), TC1M, TC1CK and TFF1 can be programmed.

Note 4: When TC1M= 1** (upper byte in the 16-bit mode), the source clock becomes the TC1 overflow signal regardless of the TC0CK setting.

Note 5: To use the TimerCounter in the 16-bit mode, select the operating mode by programming TC1M, where TC0CR<TC0 M> must be set to 011.

Note 6: To the TimerCounter in the 16-bit mode, select the source clock by programming TC0CR<TC0CK>. Set the timer start control and timer F/F control by programming TC1S and TFF1, respectively.

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

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

Note 9: When used as NORMAL1 and IDLE1 modes (low frequency is disabled), "fs" can not be used as source clock. However, it can be used as source clock in "Warm-up counter mode " for low frequency.

Table 9-1 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	TI0 pin input	TI1 pin input
8-bit timer	O	O	O	O	O	–	–	–	–
8-bit event counter	–	–	–	–	–	–	–	O	O
8-bit PDO	O	O	O	O	O	–	–	O	O
8-bit PWM	O	O	O	O	O	O	O	O	O
16-bit timer	O	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	–	–	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 (TC0CK).

Note 2: O : Available source clock

Table 9-2 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	TI0 pin input	TI1 pin input
8-bit timer	O	–	–	–	–	–	–	–	–
8-bit event counter	–	–	–	–	–	–	–	O	O
8-bit PDO	O	–	–	–	–	–	–	O	O
8-bit PWM	O	–	–	–	O	–	–	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 (TC0CK).

Note2: O : Available source clock

Table 9-3 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≤ (TTREG1, 0) ≤65535
Warm-up counter	256≤ (TTREG1, 0) ≤65535
16-bit PWM	2≤ (PWREG1, 0) ≤65534
16-bit PPG	1≤ (PWREG1, 0) < (TTREG1, 0) ≤65535 and (PWREG1, 0) + 1 < (TTREG1, 0)

Note: n = 0 to 1

9.3 Function

The TimerCounter 0 and 1 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 0 and 1 (TC0, 1) are cascadable to form a 16-bit timer. The 16-bit timer has the operating modes such as the 16-bit timer, 16-bit event counter, warm-up counter, 16-bit pulse width modulation (PWM) output and 16-bit programmable pulse generation (PPG) modes.

9.3.1 8-Bit Timer Mode (TC0 and 1)

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 $\overline{TOj}$ 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 = 0, 1

Table 9-4 Source Clock for TimerCounter 0, 1 (Internal Clock)

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	32.6 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	–
fs	fs	fs	–	30.5 μs	–	7.78 ms

Example :Setting the timer mode with source clock $fc/2^7$ Hz and generating an interrupt 80 μs later
(TimerCounter1, fc = 16.0 MHz)

```
LD      (TTREG1), 0AH          : Sets the timer register (80 μs÷2^7/fc = 0AH).
DI
SET     (EIRL). 7              : Enables INTTC1 interrupt.
EI
LD      (TC1CR), 00010000B     : Sets the operating cock to fc/2^7, and 8-bit timer mode.
LD      (TC1CR), 00011000B     : Starts TC1.
```

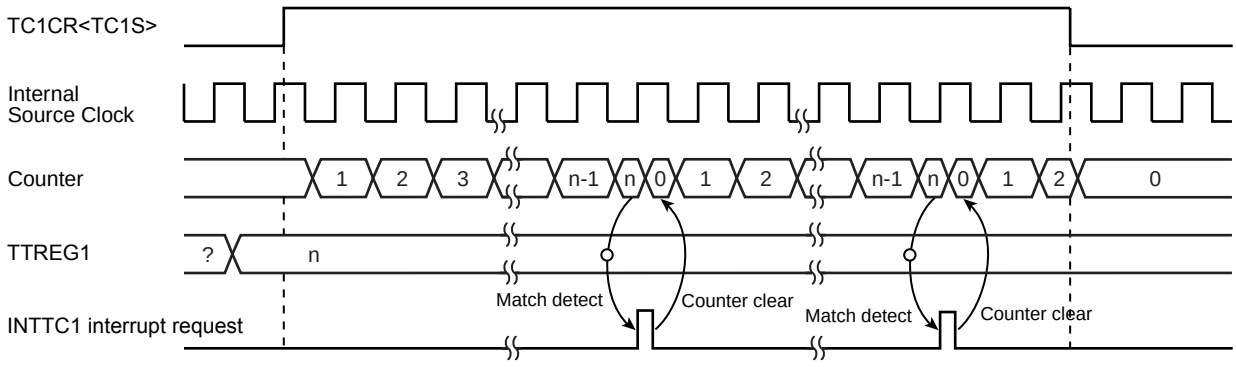


Figure 9-2 8-Bit Timer Mode Timing Chart (TC1)

9.3.2 8-Bit Event Counter Mode (TC0, 1)

In the 8-bit event counter mode, the up-counter counts up at the falling edge of the input pulse to the TIj 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 TIj pin. Two machine cycles are required for the low- or high-level pulse input to the TIj 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 $\overline{TOj}$ 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 = 0, 1

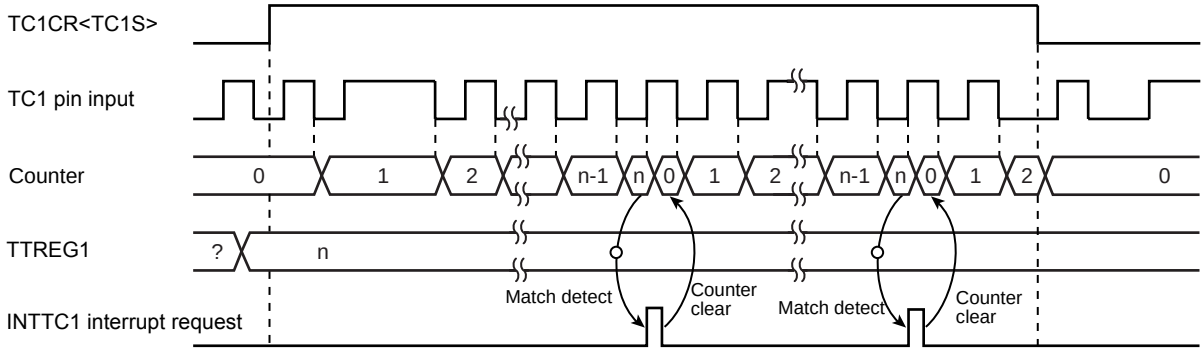


Figure 9-3 8-Bit Event Counter Mode Timing Chart (TC1)

9.3.3 8-Bit Programmable Divider Output (PDO) Mode (TC0, 1)

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

In the PDO mode, the up-counter counts up using the internal clock or external clock. When a match between the up-counter and the TTREGj value is detected, the logic level output from the $\overline{TOj}$ 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 $\overline{TOj}$ 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.

When the TIj pin input is selected as source clock (TC0CK="111"), two machine cycles are required for the low- or high-level pulse input to the TIj 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.

Example :Generating 1024 Hz pulse using TC1 (fc = 16.0 MHz)

```
        Setting port
LD      (TTREG1),  3DH          : 1/1024÷2^7/fc÷2 = 3DH
LD      (TC1CR),   00010001B    : Sets the operating clock to fc/2^7, and 8-bit PDO mode.
LD      (TC1CR),  00011001B     : Starts TC1.
```

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

Note 3: j = 0, 1

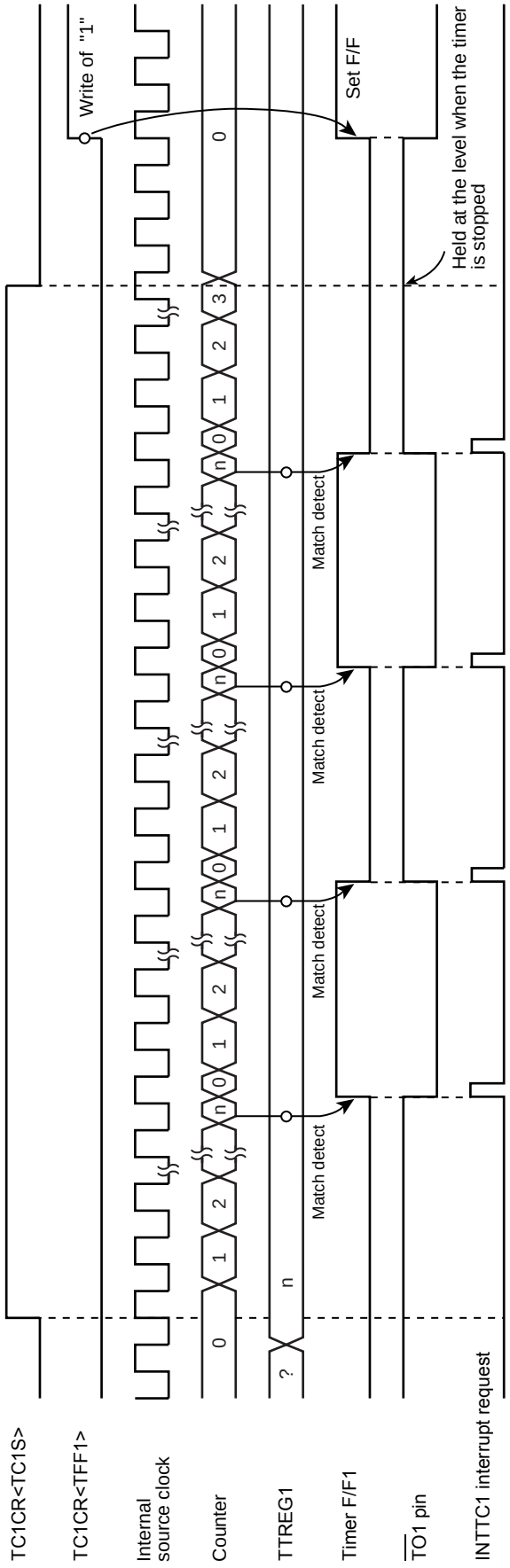


Figure 9-4 8-Bit PDO Mode Timing Chart (TC1)

9.3.4 8-Bit Pulse Width Modulation (PWM) Output Mode (TC0, 1)

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 or external 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 $\overline{TOj}$ 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.

When the TIj pin input is selected as source clock (TC0CK="111"), two machine cycles are required for the low- or high-level pulse input to the TIj 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 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 $\overline{TOj}$ 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 $\overline{TOj}$ pin to the high level when the TimerCounter is stopped
CLR (TCjCR).3: Stops the timer.
CLR (TCjCR).7: Sets the $\overline{TOj}$ 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 $\overline{TOj}$ pin during the warm-up period time after exiting the STOP mode.

Note 4: j = 0, 1

Table 9-5 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	–

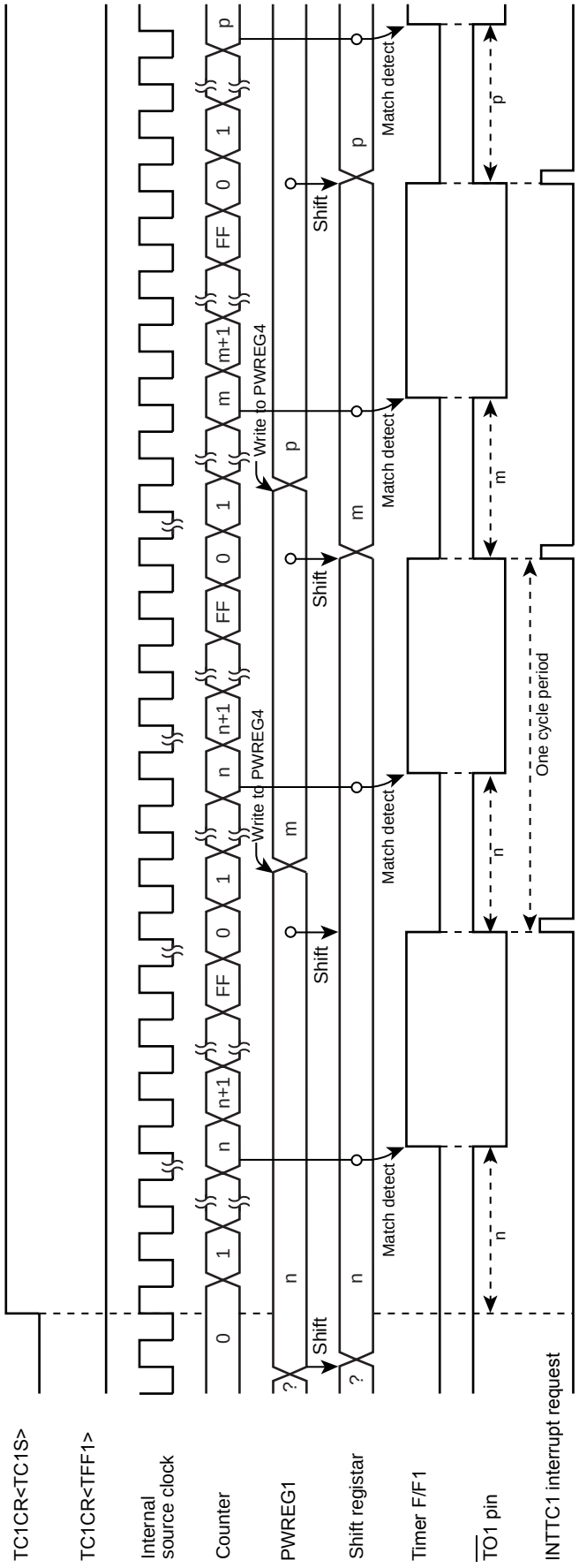


Figure 9-5 8-Bit PWM Mode Timing Chart (TC1)

9.3.5 16-Bit Timer Mode (TC0 and 1)

In the timer mode, the up-counter counts up using the internal clock. The TimerCounter 0 and 1 are cascadable to form a 16-bit timer.

When a match between the up-counter and the timer register (TTREG0, TTREG1) value is detected after the timer is started by setting TC1CR<TC1S> to 1, an INTTC1 interrupt is generated and the up-counter is cleared. After being cleared, the up-counter continues counting. Program the upper byte and lower byte in this order in the timer register. (Programming only the upper or lower byte should not be attempted.)

Note 1: In the timer mode, fix TCjCR<TFFj> to 0. If not fixed, the $\overline{TOj}$ 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 = 0, 1

Table 9-6 Source Clock for 16-Bit Timer 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					
$fc/2^{11}$	$fs/2^3$	$fs/2^3$	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	–
fs	fs	fs	–	30.5 μs	–	2 s

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

```
LDW     (TTREG0), 927CH     : Sets the timer register (300 ms÷2^7/fc = 927CH).
DI
SET     (EIRL). 7           : Enables INTTC1 interrupt.
EI
LD      (TC0CR), 13H        :Sets the operating cock to fc/2^7, and 16-bit timer mode
                             (lower byte).
LD      (TC0CR), 04H        : Sets the 16-bit timer mode (upper byte).
LD      (TC0CR), 0CH        : Starts the timer.
```

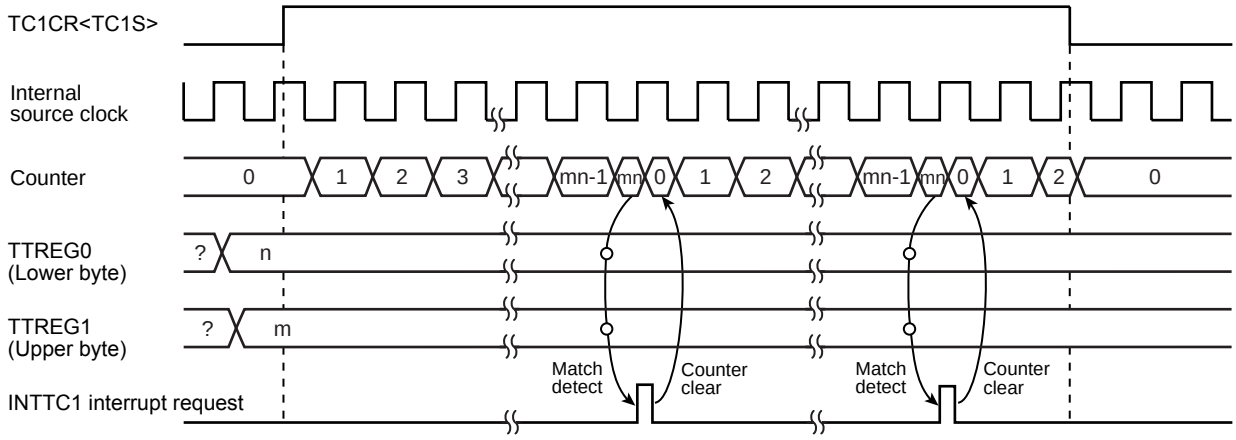


Figure 9-6 16-Bit Timer Mode Timing Chart (TC0 and TC1)

9.3.6 16-Bit Event Counter Mode (TC0 and 1)

In the event counter mode, the up-counter counts up at the falling edge to the TI0 pin. The TimerCounter 0 and 1 are cascadable to form a 16-bit event counter.

When a match between the up-counter and the timer register (TTREG0, TTREG1) value is detected after the timer is started by setting TC1CR<TC1S> to 1, an INTTC1 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 TI0 pin. Two machine cycles are required for the low- or high-level pulse input to the TI0 pin.

Therefore, a maximum frequency to be supplied is $fc/2^4$ Hz in the NORMAL1 or IDLE1 mode, and $fs/2^4$ in the SLOW1/2 or SLEEP1/2 mode. Program the lower byte (TTREG0), and upper byte (TTREG1) 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 $\overline{TOj}$ 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 = 0, 1

9.3.7 16-Bit Pulse Width Modulation (PWM) Output Mode (TC0 and 1)

This mode is used to generate a pulse-width modulated (PWM) signals with up to 16 bits of resolution. The TimerCounter 0 and 1 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 (PWREG0, PWREG1) value is detected, the logic level output from the timer F/F1 is switched to the opposite state. The counter continues counting. The logic level output from the timer F/F1 is switched to the opposite state again by the counter overflow, and the counter is cleared. The INTTC1 interrupt is generated at this time.

When the TI0 pin input is selected as source clock (TC0CK="111"), two machine cycles are required for the high- or low-level pulse input to the TI0 pin. Therefore, a maximum frequency to be supplied is $fc/2^4$ Hz in the NORMAL1 or IDLE1 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/F1 by TC1CR<TFF1>, positive and negative pulses can be generated. Upon reset, the timer F/F1 is cleared to 0.

(The logic level output from the $\overline{TO}$1 pin is the opposite to the timer F/F1 logic level.)

Since PWREG1 and 0 in the PWM mode are serially connected to the shift register, the values set to PWREG1 and 0 can be changed while the timer is running. The values set to PWREG1 and 0 during a run of the timer are shifted by the INTTCj interrupt request and loaded into PWREG1 and 0. While the timer is stopped, the values are shifted immediately after the programming of PWREG1 and 0. Set the lower byte (PWREG0) and upper byte (PWREG0) in this order to program PWREG1 and 0. (Programming only the lower or upper byte of the register should not be attempted.)

If executing the read instruction to PWREG1 and 0 during PWM output, the values set in the shift register is read, but not the values set in PWREG1 and 0. Therefore, after writing to the PWREG1 and 0, reading data of PWREG1 and 0 is previous value until INTTC1 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 PWREG1 and 0 immediately after the INTTC1 interrupt request is generated (normally in the INTTC1 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 INTTC1 interrupt request is generated.

Note 2: When the timer is stopped during PWM output, the $\overline{TO}$1 pin holds the output status when the timer is stopped. To change the output status, program TC1CR<TFF1> after the timer is stopped. Do not program TC1CR<TFF1> upon stopping of the timer.
Example: Fixing the$\overline{TO}$1 pin to the high level when the TimerCounter is stopped
CLR (TC1CR).3: Stops the timer.
CLR (TC1CR).7 : Sets the $\overline{TO}$1 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{TO}$1 pin during the warm-up period time after exiting the STOP mode.

Table 9-7 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	–
$fc/2^3$	$fc/2^3$	–	500ns	–	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     (PWREG0), 07D0H     : Sets the pulse width.
LD      (TC0CR), 33H        : Sets the operating clock to fc/2^3, and 16-bit PWM output
                              mode (lower byte).
LD      (TC1CR), 056H       : Clears TFF1 to the initial value 0, and 16-bit PWM
                              mode (upper byte).
LD      (TC1CR), 05EH       : Starts the timer.
```

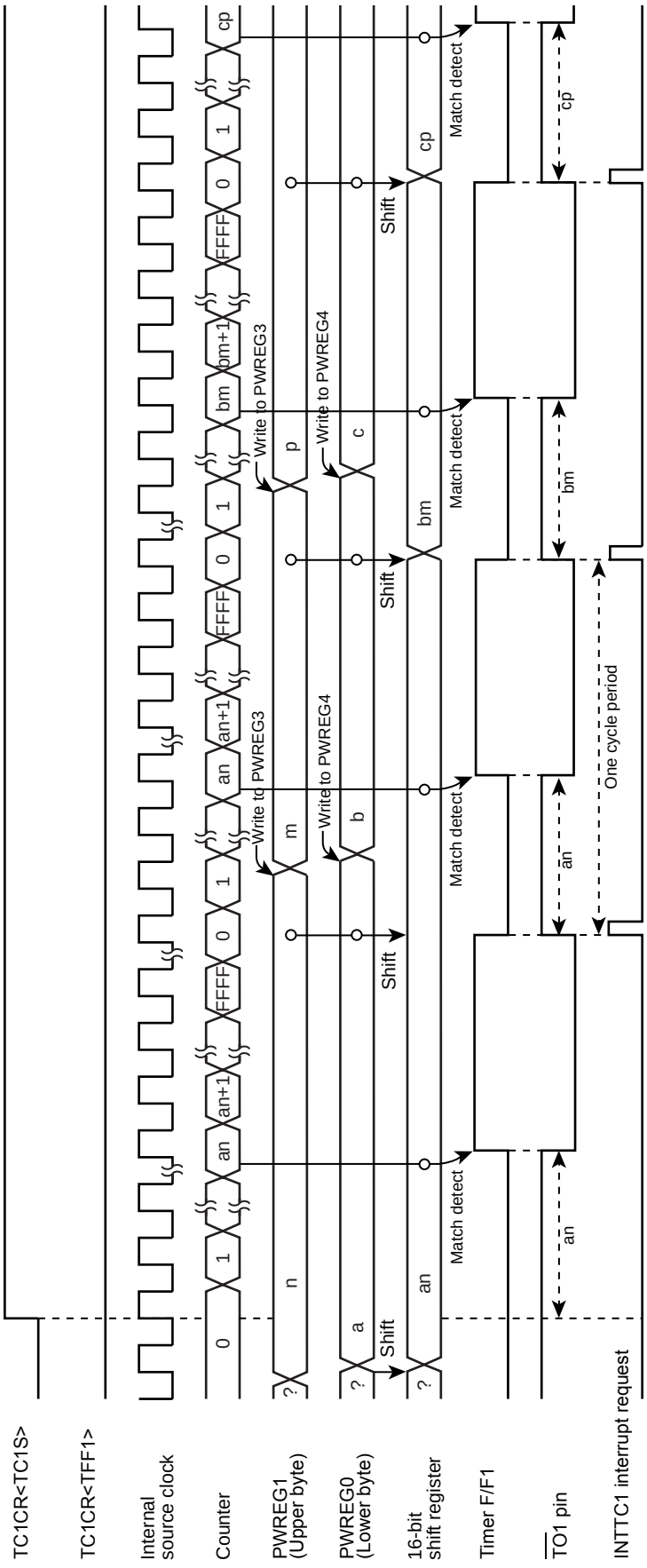


Figure 9-7 16-Bit PWM Mode Timing Chart (TC0 and TC1)

9.3.8 16-Bit Programmable Pulse Generate (PPG) Output Mode (TC0 and 1)

This mode is used to generate pulses with up to 16-bits of resolution. The timer counter 0 and 1 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 (PWREG0, PWREG1) value is detected, the logic level output from the timer F/F1 is switched to the opposite state. The counter continues counting. The logic level output from the timer F/F1 is switched to the opposite state again when a match between the up-counter and the timer register (TTREG0, TTREG1) value is detected, and the counter is cleared. The INTTC1 interrupt is generated at this time.

When the TI0 pin input is selected as source clock (TC0CK="111"), two machine cycles are required for the high- or low-level pulse input to the TI0 pin. Therefore, a maximum frequency to be supplied is $fc/2^4$ Hz in the NORMAL1 or IDLE1 mode, and $fc/2^4$ to in the SLOW1/2 or SLEEP1/2 mode.

Since the initial value can be set to the timer F/F1 by TC1CR<TFF1>, positive and negative pulses can be generated. Upon reset, the timer F/F1 is cleared to 0.

(The logic level output from the $\overline{TO}1$ pin is the opposite to the timer F/F1.)

Set the lower byte and upper byte in this order to program the timer register. (TTREG0 → TTREG1, PWREG0 → PWREG1) (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     (PWREG0), 07D0H     : Sets the pulse width.
LDW     (TTREG0), 8002H     : Sets the cycle period.
LD      (TC0CR), 33H        : Sets the operating clock to fc/2^3, and16-bit PPG mode
                              (lower byte).
LD      (TC1CR), 057H       : Clears TFF1 to the initial value 0, and 16-bit PPG
                              mode (upper byte).
LD      (TC1CR), 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 $\overline{TO}1$ pin holds the output status when the timer is stopped. To change the output status, program TC1CR<TFF1> after the timer is stopped. Do not change TC1CR<TFF1> upon stopping of the timer.
Example: Fixing the $\overline{TO}1$ pin to the high level when the TimerCounter is stopped

```
CLR  (TC1CR).3: Stops the timer
CLR  (TC1CR).7: Sets the TO1 pin to the high level
```

Note 3: i = 0, 1

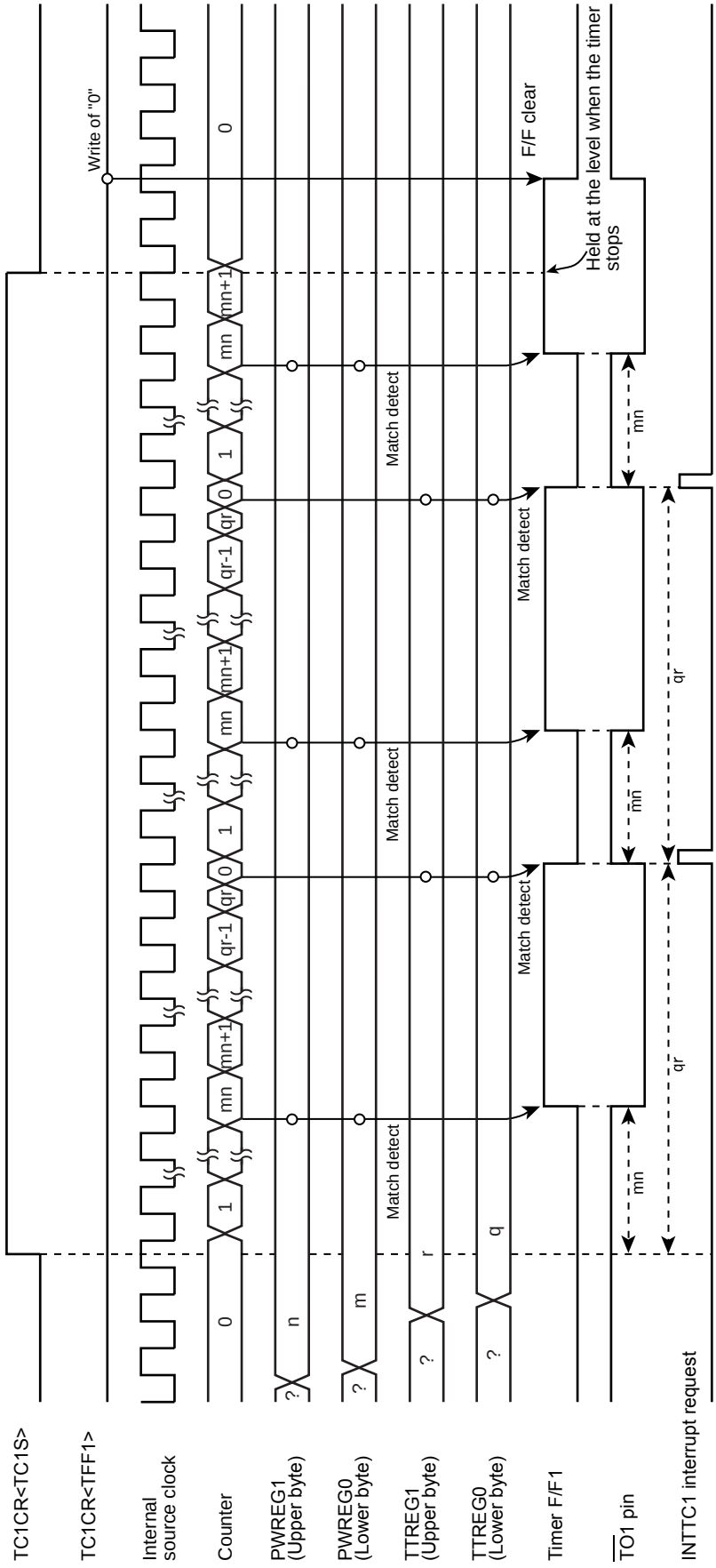


Figure 9-8 16-Bit PPG Mode Timing Chart (TC0 and TC10)

9.3.9 Warm-Up Counter Mode

In this mode, the warm-up period time is obtained to assure oscillation stability when the system clocking is switched between the high-frequency and low-frequency. The timer counter 0 and 1 are cascadable to form a 16-bit TimerCouter. 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 TTREG1 and 0 are used for match detection and lower 8 bits are not used.

Note 3: i = 0, 1

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

In this mode, the warm-up period time from a stop of the low-frequency clock fs to oscillation stability is obtained. Before starting the timer, set SYSCR2<XTEN> to 1 to oscillate the low-frequency clock. When a match between the up-counter and the timer register (TTREG1, 0) value is detected after the timer is started by setting TC1CR<TC1S> to 1, the counter is cleared by generating the INTTC1 interrupt request. After stopping the timer in the INTTC1 interrupt service routine, set SYSCR2<SYSCK> to 1 to switch the system clock from the high-frequency to low-frequency, and then clear of SYSCR2<XTEN> to 0 to stop the high-frequency clock.

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

Maximum Time Setting (TTREG1, 0 = 0100H)	Maximum Time Setting (TTREG1, 0 = FF00H)
7.81 ms	1.99 s

Example :After checking low-frequency clock oscillation stability with TC1 and 0, switching to the SLOW1 mode

```
                SET     (SYSCR2).6          : SYSCR2<XTEN> ← 1
                LD      (TC0CR), 43H        : Sets TFF0=0, source clock fs, and 16-bit mode.
                LD      (TC1CR), 05H        : Sets TFF1=0, and warm-up counter mode.
                LD      (TTREG0), 8000H     : Sets the warm-up time.
                                              (The warm-up time depends on the oscillator characteristic.)
                DI                          : IMF ← 0
                SET     (EIRL). 7           : Enables the INTTC1.
                EI                          : IMF ← 1
                SET     (TC1CR).3           : Starts TC1 and 0.
                  :           :
PINTTC1:        CLR     (TC1CR).3           : Stops TC1 and 0.
                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
                  :           :
VINTTC1:        DW      PINTTC1             : INTTC1 vector table
```

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

In this mode, the warm-up period time from a stop of the high-frequency clock 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 (TTREG1, 0) value is detected after the timer is started by setting TC1CR<TC1S> to 1, the counter is cleared by generating the INTTC1 interrupt request. After stopping the timer in the INTTC1 interrupt service routine, clear SYSCR2<SYSCK> to 0 to switch the system clock from the low-frequency to high-frequency, and then SYSCR2<XTEN> to 0 to stop the low-frequency clock.

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

Minimum time (TTREG1, 0 = 0100H)	Maximum time (TTREG1, 0 = FF00H)
16 μs	4.08 ms

Example :After checking high-frequency clock oscillation stability with TC1 and 0, switching to the NORMAL1 mode

```
                SET     (SYSCR2).7          : SYSCR2<XEN> ← 1
                LD      (TC0CR), 63H        : Sets TFF0=0, source clock fs, and 16-bit mode.
                LD      (TC1CR), 05H        : Sets TFF1=0, and warm-up counter mode.
                LD      (TTREG0), 0F800H    : Sets the warm-up time.
                                              (The warm-up time depends on the oscillator characteristic.)
                DI                          : IMF ← 0
                SET     (EIRL). 7           : Enables the INTTC1.
                EI                          : IMF ← 1
                SET     (TC1CR).3           : Starts the TC1 and 0.
                 :        :
PINTTC1:        CLR     (TC1CR).3           : Stops the TC1 and 0.
                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
                 :        :
VINTTC1:        DW      PINTTC1             : INTTC1 vector table
```

10. Synchronous Serial Interface (SIO)

The TMP86CH06AUG has a clocked-synchronous 8-bit serial interface. Serial interface has an 8-byte transmit and receive data buffer that can automatically and continuously transfer up to 64 bits of data.

Serial interface is connected to outside peripherl devices via SO, SI, SCK port.

10.1 Configuration

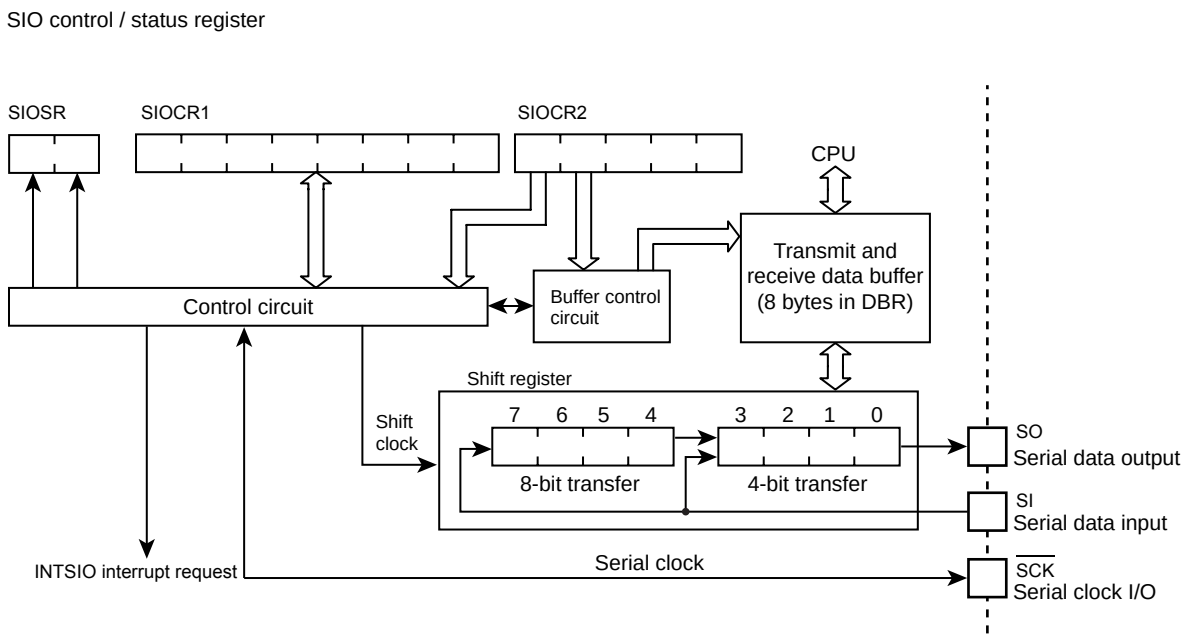


Figure 10-1 Serial Interface

10.2 Control

The serial interface is controlled by SIO control registers (SIOCR1/SIOCR2). The serial interface status can be determined by reading SIO status register (SIOSR).

The transmit and receive data buffer is controlled by the SIOCR2<BUF>. The data buffer is assigned to address 0028H to 002FH for SIO in the DBR area, and can continuously transfer up to 8 words (bytes or nibbles) at one time. When the specified number of words has been transferred, a buffer empty (in the transmit mode) or a buffer full (in the receive mode or transmit/receive mode) interrupt (INTSIO) is generated.

When the internal clock is used as the serial clock in the 8-bit receive mode and the 8-bit transmit/receive mode, a fixed interval wait can be applied to the serial clock for each word transferred. Four different wait times can be selected with SIOCR2<WAIT>.

SIO Control Register 1

SIOCR1	7	6	5	4	3	2	1	0	
(0026H)	SIOS	SIOINH	SIOM			SCK			(Initial value: 0000 0000)

SIOS	Indicate transfer start / stop	0: Stop 1: Start				Write only
SIOINH	Continue / abort transfer	0: Continuously transfer 1: Abort transfer (Automatically cleared after abort)				
SIOM	Transfer mode select	000: 8-bit transmit mode 010: 4-bit transmit mode 100: 8-bit transmit / receive mode 101: 8-bit receive mode 110: 4-bit receive mode Except the above: Reserved				
SCK	Serial clock select		NORMAL1/2, IDLE1/2 mode		SLOW1/2 SLEEP1/2 mode	Write only
			DV7CK = 0	DV7CK = 1		
		000	$fc/2^{13}$	$fs/2^5$	$fs/2^5$	
		001	$fc/2^8$	$fc/2^8$	-	
		010	$fc/2^7$	$fc/2^7$	-	
		011	$fc/2^6$	$fc/2^6$	-	
		100	$fc/2^5$	$fc/2^5$	-	
		101	$fc/2^4$	$fc/2^4$	-	
		110	Reserved			
		111	External clock (Input from SCK pin)			

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

Note 2: Set SIOS to "0" and SIOINH to "1" when setting the transfer mode or serial clock.

Note 3: SIOCR1 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.

SIO Control Register 2

SIOCR2	7	6	5	4	3	2	1	0	
(0027H)				WAIT		BUF			(Initial value: ***0 0000)

WAIT	Wait control	Always sets "00" except 8-bit transmit / receive mode. 00: $T_f = T_D$(Non wait) 01: $T_f = 2T_D$(Wait) 10: $T_f = 4T_D$(Wait) 11: $T_f = 8T_D$ (Wait)	Write only
BUF	Number of transfer words (Buffer address in use)	000: 1 word transfer 0028H 001: 2 words transfer 0028H ~ 0029H 010: 3 words transfer 0028H ~ 002AH 011: 4 words transfer 0028H ~ 002BH 100: 5 words transfer 0028H ~ 002CH 101: 6 words transfer 0028H ~ 002DH 110: 7 words transfer 0028H ~ 002EH 111: 8 words transfer 0028H ~ 002FH	

Note 1: The lower 4 bits of each buffer are used during 4-bit transfers. Zeros (0) are stored to the upper 4bits when receiving.

Note 2: Transmitting starts at the lowest address. Received data are also stored starting from the lowest address to the highest address. (The first buffer address transmitted is 0028H).

Note 3: The value to be loaded to BUF is held after transfer is completed.

Note 4: SIOCR2 must be set when the serial interface is stopped (SIOF = 0).

Note 5: *: Don't care

Note 6: SIOCR2 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.

SIO Status Register

SIOSR	7	6	5	4	3	2	1	0
(0027H)	SIOF	SEF						

SIOF	Serial transfer operating status monitor	0: Transfer terminated 1: Transfer in process	Read only
SEF	Shift operating status monitor	0: Shift operation terminated 1: Shift operation in process	

Note 1: T_f; Frame time, T_D; Data transfer time

Note 2: After SIOS is cleared to "0", SIOF is cleared to "0" at the termination of transfer or the setting of SIOINH to "1".

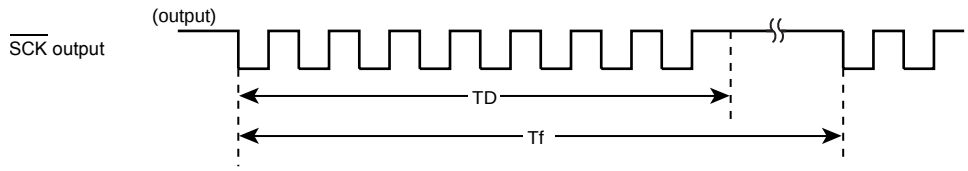


Figure 10-2 Frame time (T_f) and Data transfer time (T_D)

10.3 Serial clock

10.3.1 Clock source

Internal clock or external clock for the source clock is selected by SIOCR1<SCK>.

10.3.1.1 Internal clock

Any of six frequencies can be selected. The serial clock is output to the outside on the SCK pin. The SCK pin goes high when transfer starts.

When data writing (in the transmit mode) or reading (in the receive mode or the transmit/receive mode) cannot keep up with the serial clock rate, there is a wait function that automatically stops the serial clock and holds the next shift operation until the read/write processing is completed.

Table 10-1 Serial Clock Rate

	NORMAL1/2, IDLE1/2 mode				SLOW1/2, SLEEP1/2 mode	
	DV7CK = 0		DV7CK = 1			
SCK	Clock	Baud Rate	Clock	Baud Rate	Clock	Baud Rate
000	$fc/2^{13}$	1.91 Kbps	$fs/2^5$	1024 bps	$fs/2^5$	1024 bps
001	$fc/2^8$	61.04 Kbps	$fc/2^8$	61.04 Kbps	-	-
010	$fc/2^7$	122.07 Kbps	$fc/2^7$	122.07 Kbps	-	-
011	$fc/2^6$	244.14 Kbps	$fc/2^6$	244.14 Kbps	-	-
100	$fc/2^5$	488.28 Kbps	$fc/2^5$	488.28 Kbps	-	-
101	$fc/2^4$	976.56 Kbps	$fc/2^4$	976.56 Kbps	-	-
110	-	-	-	-	-	-
111	External	External	External	External	External	External

Note: 1 Kbit = 1024 bit (fc = 16 MHz, fs = 32.768 kHz)

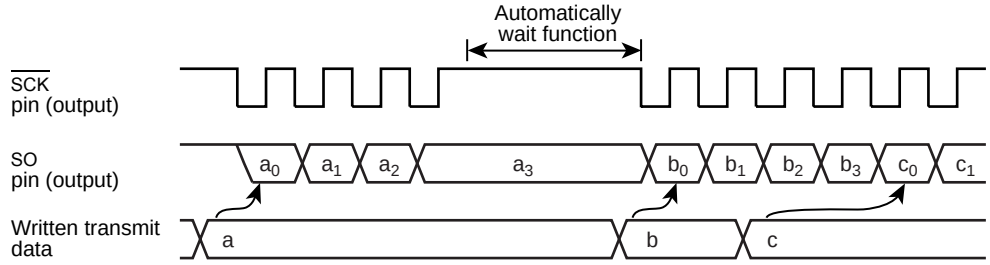


Figure 10-3 Automatic Wait Function (at 4-bit transmit mode)

10.3.1.2 External clock

An external clock connected to the $\overline{SCK}$ pin is used as the serial clock. In this case, output latch of this port should be set to "1". To ensure shifting, a pulse width of at least 4 machine cycles is required. This pulse is needed for the shift operation to execute certainly. Actually, there is necessary processing time for interrupting, writing, and reading. The minimum pulse is determined by setting the mode and the program. Therfore, maximum transfer frequency will be 488.3K bit/sec (at fc=16MHz).

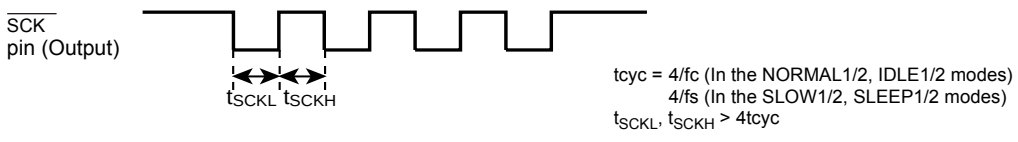


Figure 10-4 External clock pulse width

10.3.2 Shift edge

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

10.3.2.1 Leading edge

Transmitted data are shifted on the leading edge of the serial clock (falling edge of the $\overline{SCK}$ pin input/output).

10.3.2.2 Trailing edge

Received data are shifted on the trailing edge of the serial clock (rising edge of the $\overline{SCK}$ pin input/output).

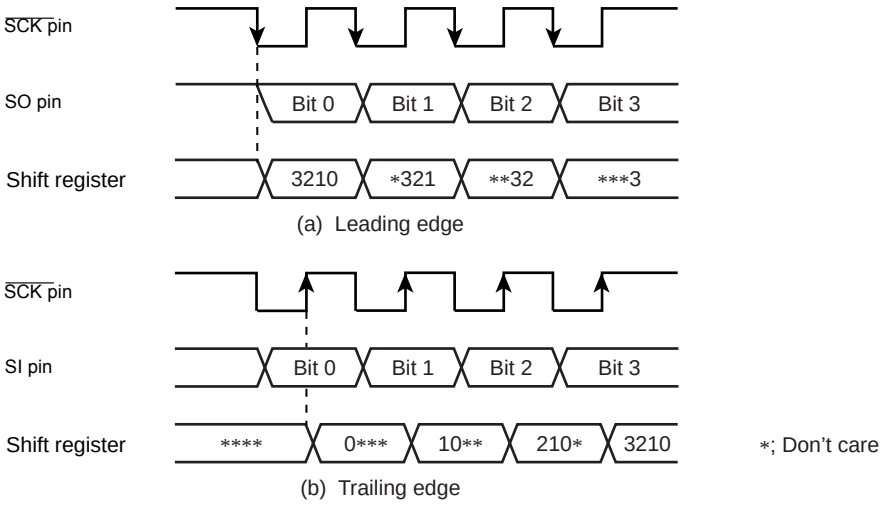


Figure 10-5 Shift edge

10.4 Number of bits to transfer

Either 4-bit or 8-bit serial transfer can be selected. When 4-bit serial transfer is selected, only the lower 4 bits of the transmit/receive data buffer register are used. The upper 4 bits are cleared to "0" when receiving.
The data is transferred in sequence starting at the least significant bit (LSB).

10.5 Number of words to transfer

Up to 8 words consisting of 4 bits of data (4-bit serial transfer) or 8 bits (8-bit serial transfer) of data can be transferred continuously. The number of words to be transferred can be selected by SIOCR2<BUF>.

An INTSIO interrupt is generated when the specified number of words has been transferred. If the number of words is to be changed during transfer, the serial interface must be stopped before making the change. The number of words can be changed during automatic-wait operation of an internal clock. In this case, the serial interface is not required to be stopped.

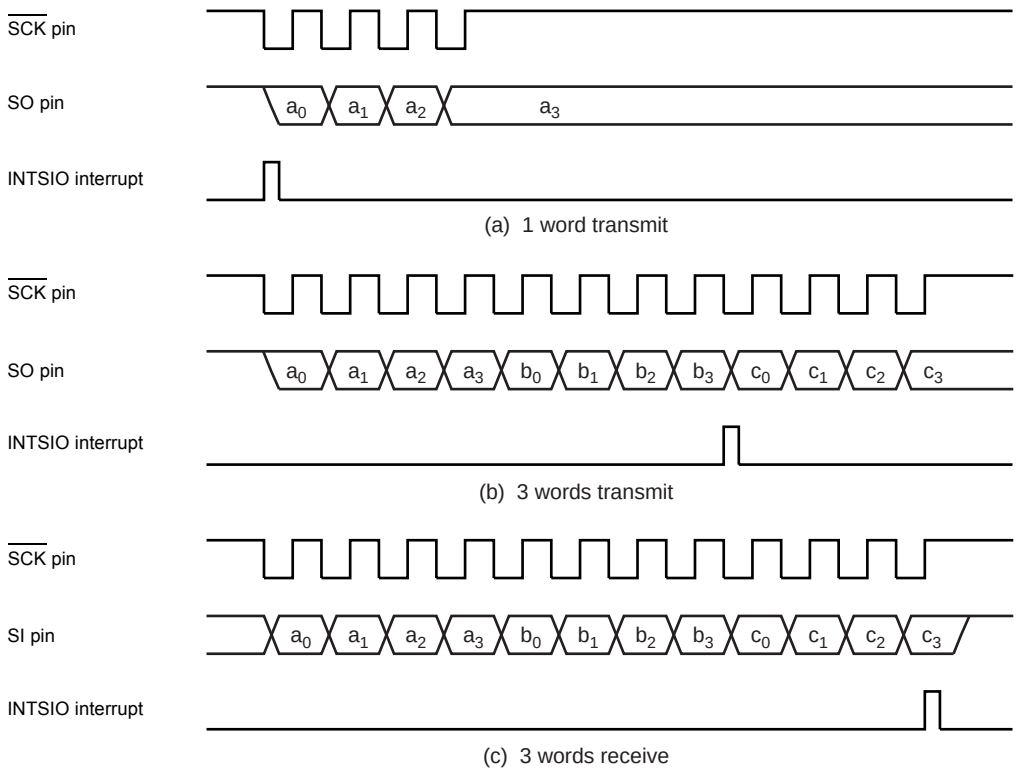


Figure 10-6 Number of words to transfer (Example: 1word = 4bit)

10.6 Transfer Mode

SIOCR1<SIOM> is used to select the transmit, receive, or transmit/receive mode.

10.6.1 4-bit and 8-bit transfer modes

In these modes, firstly set the SIO control register to the transmit mode, and then write first transmit data (number of transfer words to be transferred) to the data buffer registers (DBR).

After the data are written, the transmission is started by setting SIOCR1<SIOS> to "1". The data are then output sequentially to the SO pin in synchronous with the serial clock, starting with the least significant bit (LSB). As soon as the LSB has been output, the data are transferred from the data buffer register to the shift register. When the final data bit has been transferred and the data buffer register is empty, an INTSIO (Buffer empty) interrupt is generated to request the next transmitted data.

When the internal clock is used, the serial clock will stop and an automatic-wait will be initiated if the next transmitted data are not loaded to the data buffer register by the time the number of data words specified with the SIOCR2<BUF> has been transmitted. Writing even one word of data cancels the automatic-wait; therefore, when transmitting two or more words, always write the next word before transmission of the previous word is completed.

Note: Automatic waits are also canceled by writing to a DBR not being used as a transmit data buffer register; therefore, during SIO do not use such DBR for other applications. For example, when 3 words are transmitted, do not use the DBR of the remained 5 words.

When an external clock is used, the data must be written to the data buffer register before shifting next data. Thus, the transfer speed is determined by the maximum delay time from the generation of the interrupt request to writing of the data to the data buffer register by the interrupt service program.

The transmission is ended by clearing SIOCR1<SIOS> to "0" or setting SIOCR1<SIOINH> to "1" in buffer empty interrupt service program.

SIOCR1<SIOS> is cleared, the operation will end after all bits of words are transmitted.

That the transmission has ended can be determined from the status of SIOSR<SIOF> because SIOSR<SIOF> is cleared to "0" when a transfer is completed.

When SIOCR1<SIOINH> is set, the transmission is immediately ended and SIOSR<SIOF> is cleared to "0".

When an external clock is used, it is also necessary to clear SIOCR1<SIOS> to "0" before shifting the next data; If SIOCR1<SIOS> is not cleared before shift out, dummy data will be transmitted and the operation will end.

If it is necessary to change the number of words, SIOCR1<SIOS> should be cleared to "0", then SIOCR2<BUF> must be rewritten after confirming that SIOSR<SIOF> has been cleared to "0".

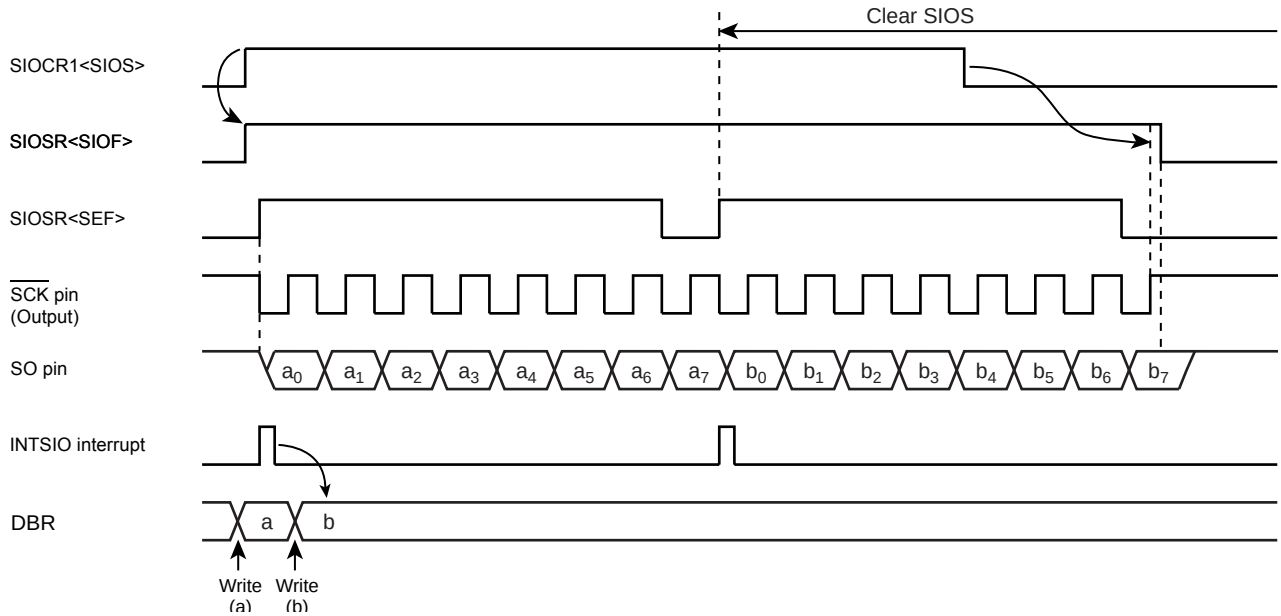


Figure 10-7 Transfer Mode (Example: 8bit, 1word transfer, Internal clock)

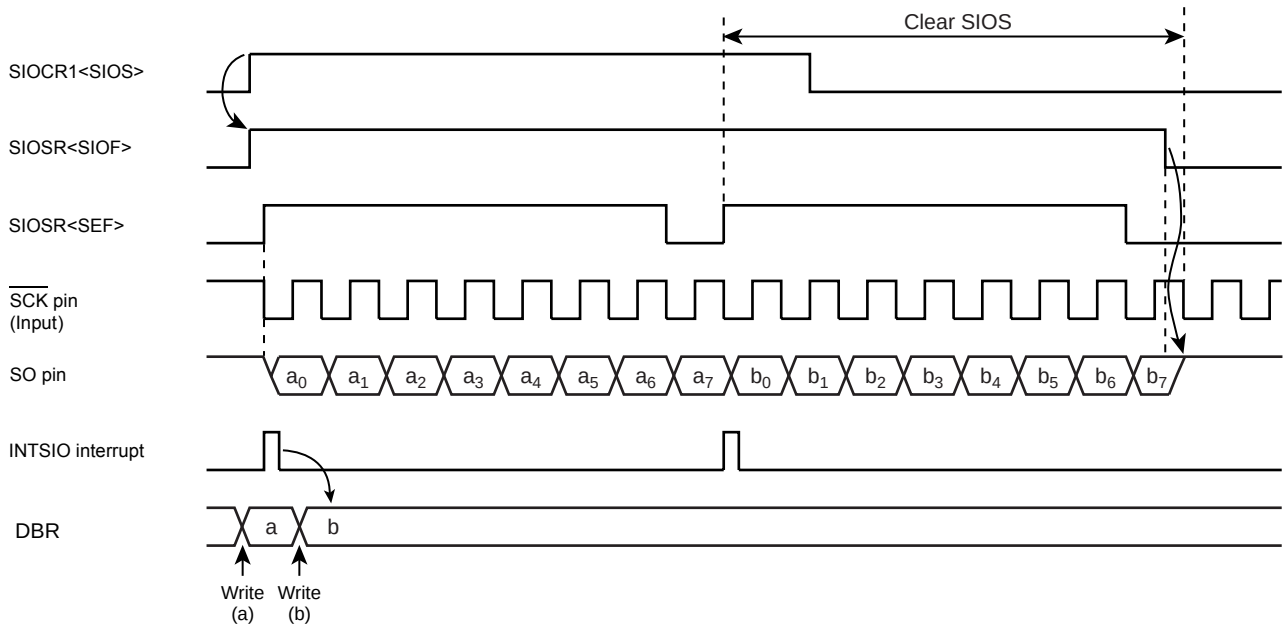


Figure 10-8 Transfer Mode (Example: 8bit, 1word transfer, External clock)

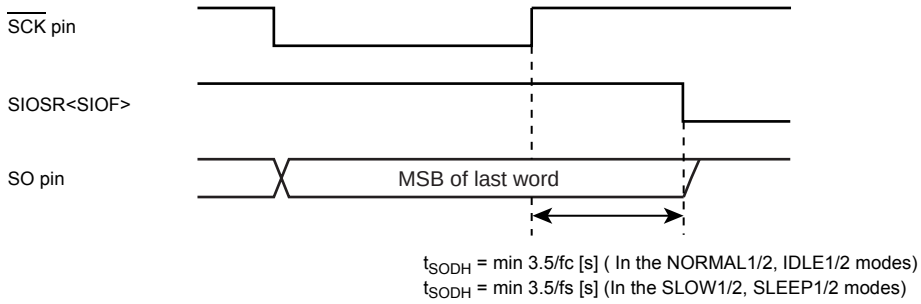


Figure 10-9 Transmiiied Data Hold Time at End of Transfer

10.6.2 4-bit and 8-bit receive modes

After setting the control registers to the receive mode, set SIOCR1<SIOS> to "1" to enable receiving. The data are then transferred to the shift register via the SI pin in synchronous with the serial clock. When one word of data has been received, it is transferred from the shift register to the data buffer register (DBR). When the number of words specified with the SIOCR2<BUF> has been received, an INTSIO (Buffer full) interrupt is generated to request that these data be read out. The data are then read from the data buffer registers by the interrupt service program.

When the internal clock is used, and the previous data are not read from the data buffer register before the next data are received, the serial clock will stop and an automatic-wait will be initiated until the data are read. A wait will not be initiated if even one data word has been read.

Note: Waits are also canceled by reading a DBR not being used as a received data buffer register is read; therefore, during SIO do not use such DBR for other applications.

When an external clock is used, the shift operation is synchronized with the external clock; therefore, the previous data are read before the next data are transferred to the data buffer register. If the previous data have not been read, the next data will not be transferred to the data buffer register and the receiving of any more data will be canceled. When an external clock is used, the maximum transfer speed is determined by the delay between the time when the interrupt request is generated and when the data received have been read.

The receiving is ended by clearing SIOCR1<SIOS> to "0" or setting SIOCR1<SIOINH> to "1" in buffer full interrupt service program.
When SIOCR1<SIOS> is cleared, the current data are transferred to the buffer. After SIOCR1<SIOS> cleared, the receiving is ended at the time that the final bit of the data has been received. That the receiving has ended can be determined from the status of SIOSR<SIOF>. SIOSR<SIOF> is cleared to "0" when the receiving is ended. After confirmed the receiving termination, the final receiving data is read. When SIOCR1<SIOINH> is set, the receiving is immediately ended and SIOSR<SIOF> is cleared to "0". (The received data is ignored, and it is not required to be read out.)

If it is necessary to change the number of words in external clock operation, SIOCR1<SIOS> should be cleared to "0" then SIOCR2<BUF> must be rewritten after confirming that SIOSR<SIOF> has been cleared to "0". If it is necessary to change the number of words in internal clock, during automatic-wait operation which occurs after completion of data receiving, SIOCR2<BUF> must be rewritten before the received data is read out.

Note: The buffer contents are lost when the transfer mode is switched. If it should become necessary to switch the transfer mode, end receiving by clearing SIOCR1<SIOS> to "0", read the last data and then switch the transfer mode.

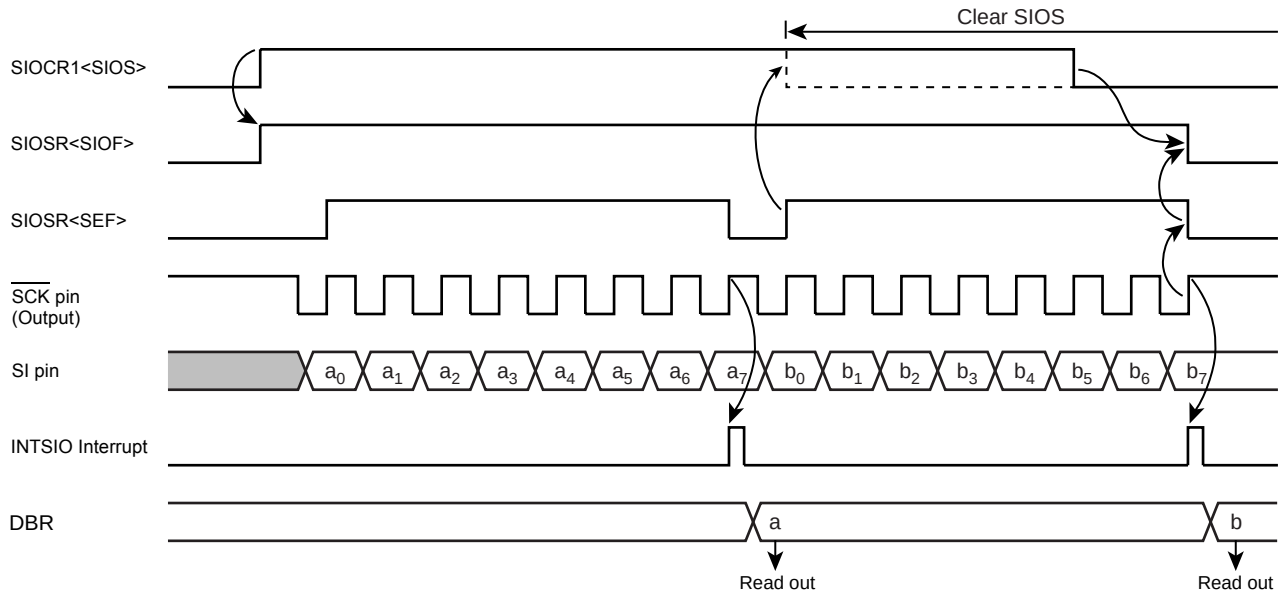


Figure 10-10 Receive Mode (Example: 8bit, 1word transfer, Internal clock)

10.6.3 8-bit transfer / receive mode

After setting the SIO control register to the 8-bit transmit/receive mode, write the data to be transmitted first to the data buffer registers (DBR). After that, enable the transmit/receive by setting SIOCR1<SIOS> to "1". When transmitting, the data are output from the SO pin at leading edges of the serial clock. When receiving, the data are input to the SI pin at the trailing edges of the serial clock. When the all receive is enabled, 8-bit data are transferred from the shift register to the data buffer register. An INTSIO interrupt is generated when the number of data words specified with the SIOCR2<BUF> has been transferred. Usually, read the receive data from the buffer register in the interrupt service. The data buffer register is used for both transmitting and receiving; therefore, always write the data to be transmitted after reading the all received data.

When the internal clock is used, a wait is initiated until the received data are read and the next transfer data are written. A wait will not be initiated if even one transfer data word has been written.

When an external clock is used, the shift operation is synchronized with the external clock; therefore, it is necessary to read the received data and write the data to be transmitted next before starting the next shift operation. When an external clock is used, the transfer speed is determined by the maximum delay between generation of an interrupt request and the received data are read and the data to be transmitted next are written.

The transmit/receive operation is ended by clearing SIOCR1<SIOS> to "0" or setting SIOCR1<SIOINH> to "1" in INTSIO interrupt service program.

When SIOCR1<SIOS> is cleared, the current data are transferred to the buffer. After SIOCR1<SIOS> cleared, the transmitting/receiving is ended at the time that the final bit of the data has been transmitted.
That the transmitting/receiving has ended can be determined from the status of SIOSR<SIOF>. SIOSR<SIOF> is cleared to "0" when the transmitting/receiving is ended.

When SIOCR1<SIOINH> is set, the transmit/receive operation is immediately ended and SIOSR<SIOF> is cleared to "0".

If it is necessary to change the number of words in external clock operation, SIOCR1<SIOS> should be cleared to "0", then SIOCR2<BUF> must be rewritten after confirming that SIOSR<SIOF> has been cleared to "0".

If it is necessary to change the number of words in internal clock, during automatic-wait operation which occurs after completion of transmit/receive operation, SIOCR2<BUF> must be rewritten before reading and writing of the receive/transmit data.

Note: The buffer contents are lost when the transfer mode is switched. If it should become necessary to switch the transfer mode, end receiving by clearing SIOCR1<SIOS> to “0”, read the last data and then switch the transfer mode.

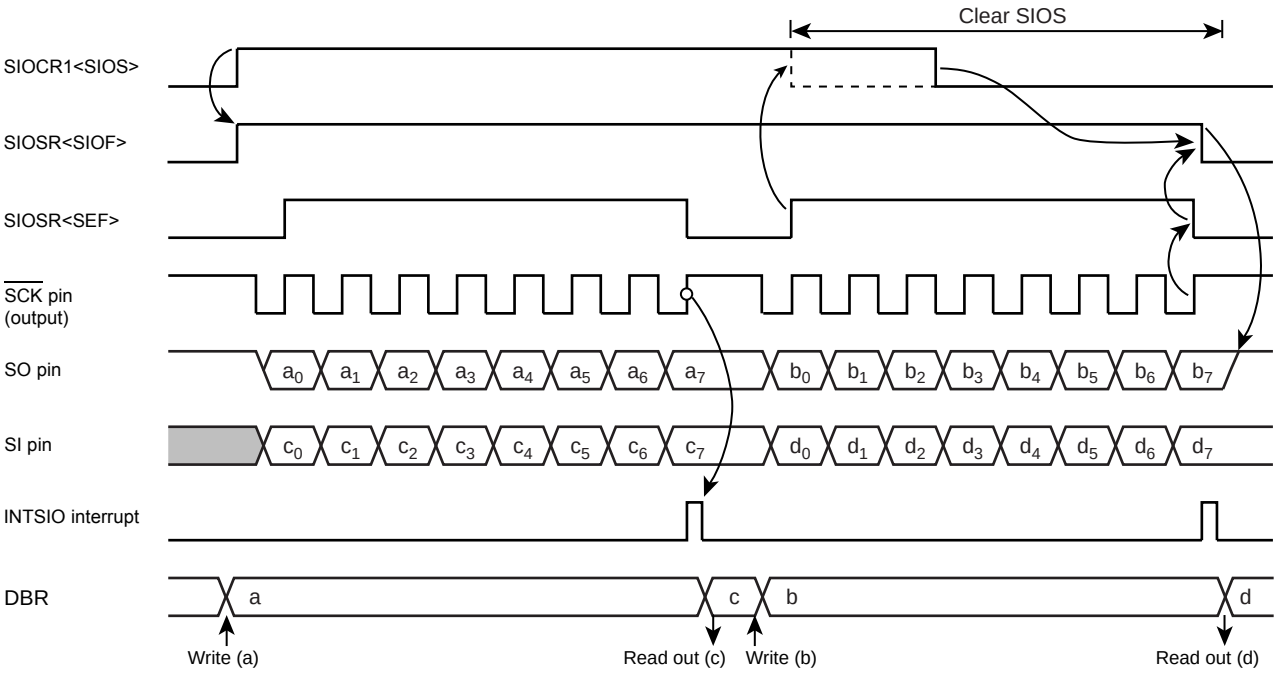


Figure 10-11 Transfer / Receive Mode (Example: 8bit, 1word transfer, Internal clock)

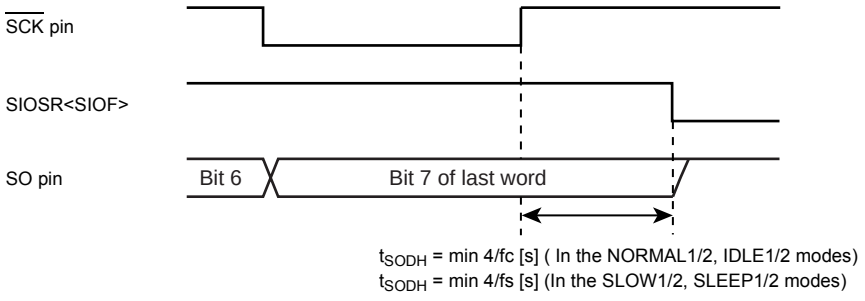


Figure 10-12 Transmitted Data Hold Time at End of Transfer / Receive

11. Asynchronous Serial interface (UART0)

11.1 Configuration

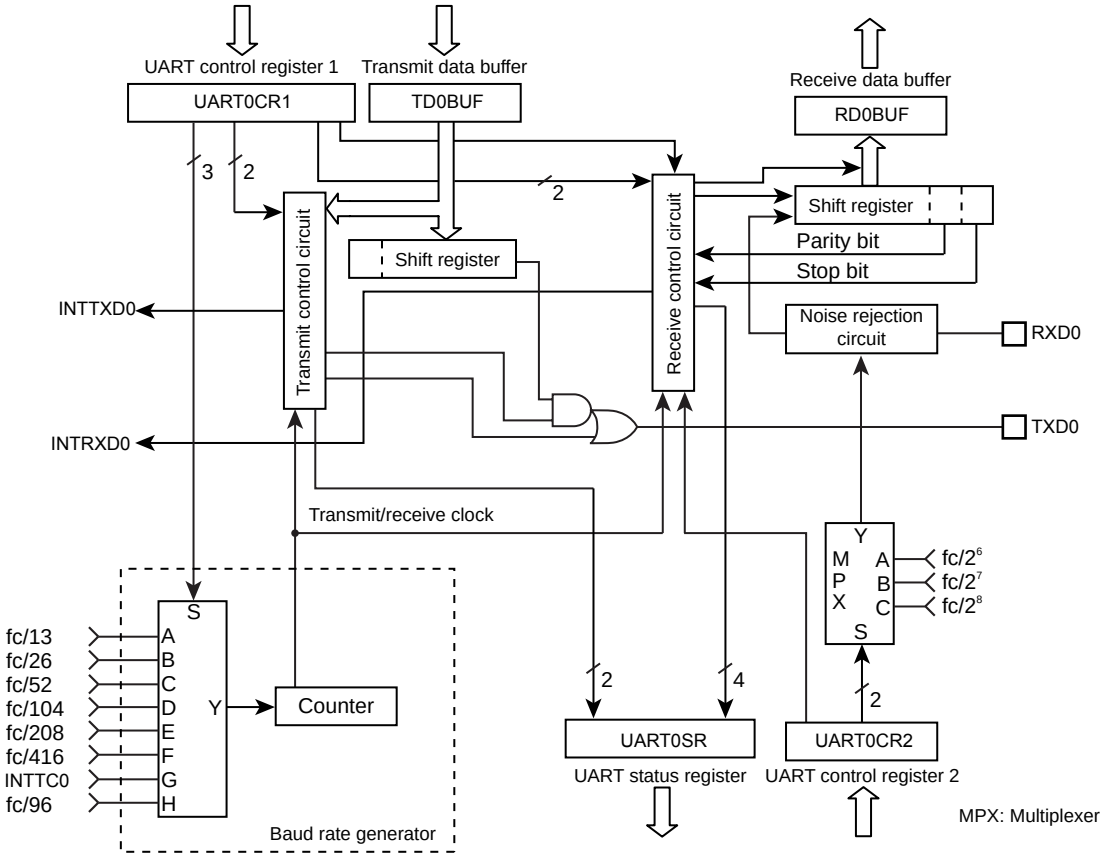


Figure 11-1 UART0 (Asynchronous Serial Interface)

11.2 Control

UART0 is controlled by the UART0 Control Registers (UART0CR1, UART0CR2). The operating status can be monitored using the UART status register (UART0SR).

UART0 Control Register1

UART0CR1 (001AH)

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

(Initial value: 0000 0000)

TXE	Transfer operation	0: Disable 1: Enable	Write only
RXE	Receive operation	0: Disable 1: Enable	
STBT	Transmit stop bit length	0: 1 bit 1: 2 bits	
EVEN	Even-numbered parity	0: Odd-numbered parity 1: Even-numbered parity	
PE	Parity addition	0: No parity 1: Parity	
BRG	Transmit clock select	000: fc/13 [Hz] 001: fc/26 010: fc/52 011: fc/104 100: fc/208 101: fc/416 110: TC0 (Input INTTC0) 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: UART0CR1<RXE> and UART0CR1<TXE> should be set to “0” before UART0CR1<BRG> is changed.

UART0 Control Register2

UART0CR2 (001BH)

7	6	5	4	3	2	1	0
					RXDNC	STOPBR	

(Initial value: **** *000)

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

Note: When UART0CR2<RXDNC> = “01”, pulses longer than 96/fc [s] are always regarded as signals; when UART0CR2<RXDNC> = “10”, longer than 192/fc [s]; and when UART0CR2<RXDNC> = “11”, longer than 384/fc [s].

UART0 Status Register

UART0SR (001AH)

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

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

UART0 Receive Data Buffer

RD0BUF (001CH)

7	6	5	4	3	2	1	0

Read only

(Initial value: 0000 0000)

UART0 Transmit Data Buffer

TD0BUF (001CH)

7	6	5	4	3	2	1	0

Write only

(Initial value: 0000 0000)

11.3 Transfer Data Format

In UART0, an one-bit start bit (Low level), stop bit (Bit length selectable at high level, by UART0CR1<STBT>), and parity (Select parity in UART0CR1<PE>; even- or odd-numbered parity by UART0CR1<EVEN>) are added to the transfer data. The transfer data formats are shown as follows.

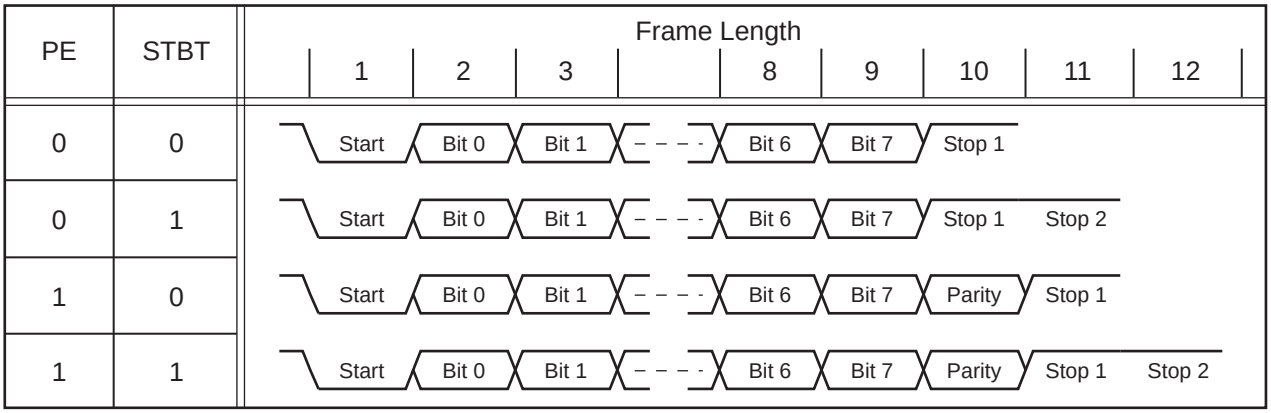


Figure 11-2 Transfer Data Format

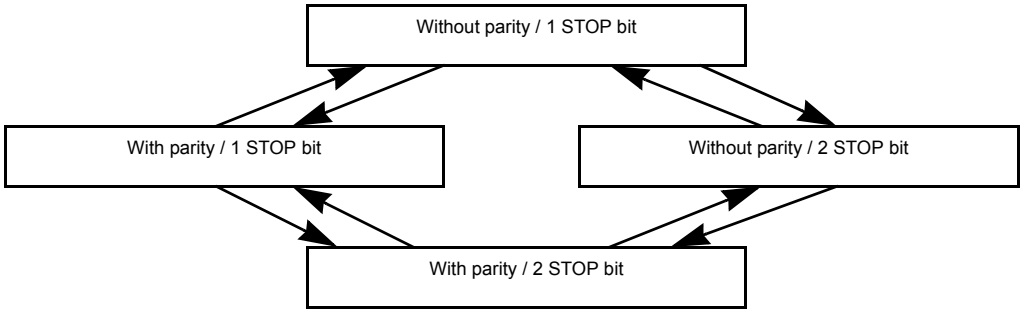


Figure 11-3 Caution on Changing Transfer Data Format

Note: In order to switch the transfer data format, perform transmit operations in the above Figure 11-3 sequence except for the initial setting.

11.4 Transfer Rate

The baud rate of UART0 is set of UART0CR1<BRG>. The example of the baud rate are shown as follows.

Table 11-1 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 TC0 is used as the UART0 transfer rate (when UART0CR1<BRG> = “110”), the transfer clock and transfer rate are determined as follows:

Transfer clock [Hz] = TC0 source clock [Hz] / TTREG0 setting value

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

11.5 Data Sampling Method

The UART0 receiver keeps sampling input using the clock selected by UART0CR1<BRG> until a start bit is detected in RXD0 pin input. RT clock starts detecting “L” level of the RXD0 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).

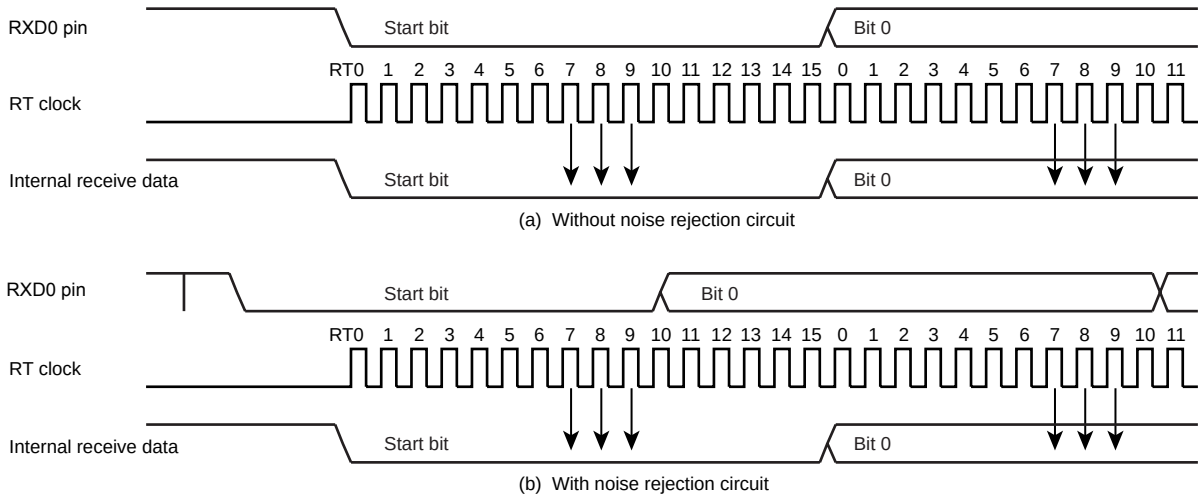


Figure 11-4 Data Sampling Method

11.6 STOP Bit Length

Select a transmit stop bit length (1 bit or 2 bits) by UART0CR1<STBT>.

11.7 Parity

Set parity / no parity by UART0CR1<PE> and set parity type (Odd- or Even-numbered) by UART0CR1<EVEN>.

11.8 Transmit/Receive Operation

11.8.1 Data Transmit Operation

Set UART0CR1<TXE> to "1". Read UART0SR to check UART0SR<TBEP> = "1", then write data in TD0BUF (Transmit data buffer). Writing data in TD0BUF zero-clears UART0SR<TBEP>, transfers the data to the transmit shift register and the data are sequentially output from the TXD0 pin. The data output include a one-bit start bit, stop bits whose number is specified in UART0CR1<STBT> and a parity bit if parity addition is specified. Select the data transfer baud rate using UART0CR1<BRG>. When data transmit starts, transmit buffer empty flag UART0SR<TBEP> is set to "1" and an INTTXD0 interrupt is generated.

While UART0CR1<TXE> = "0" and from when "1" is written to UART0CR1<TXE> to when send data are written to TD0BUF, the TXD0 pin is fixed at high level.
When transmitting data, first read UART0SR, then write data in TD0BUF. Otherwise, UART0SR<TBEP> is not zero-cleared and transmit does not start.

11.8.2 Data Receive Operation

Set UART0CR1<RXE> to "1". When data are received via the RXD0 pin, the receive data are transferred to RD0BUF (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 RD0BUF (Receive data buffer). Then the receive buffer full flag UART0SR<RBFL> is set and an INTRXD0 interrupt is generated. Select the data transfer baud rate using UART0CR1<BRG>.

If an overrun error (OERR) occurs when data are received, the data are not transferred to RD0BUF (Receive data buffer) but discarded; data in the RD0BUF are not affected.

Note: When a receive operation is disabled by setting UART0CR1<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.

11.9 Status Flag

11.9.1 Parity Error

When parity determined using the receive data bits differs from the received parity bit, the parity error flag UART0SR<PERR> is set to “1”. The UART0SR<PERR> is cleared to “0” when the RD0BUF is read after reading the UART0SR.

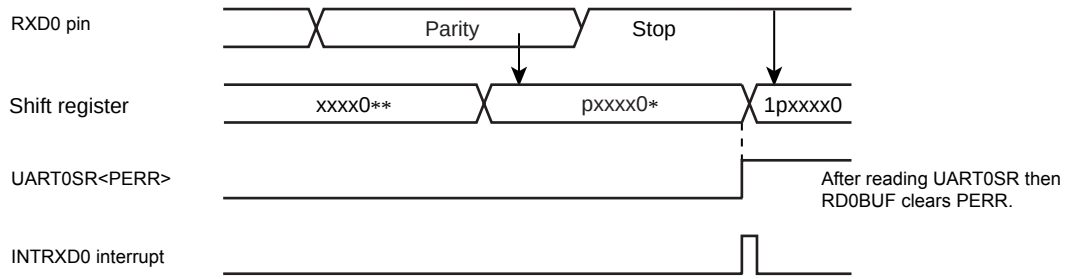


Figure 11-5 Generation of Parity Error

11.9.2 Framing Error

When “0” is sampled as the stop bit in the receive data, framing error flag UART0SR<FERR> is set to “1”. The UART0SR<FERR> is cleared to “0” when the RD0BUF is read after reading the UART0SR.

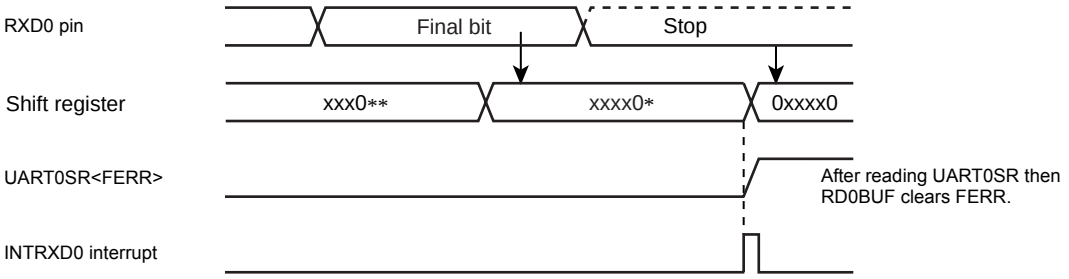


Figure 11-6 Generation of Framing Error

11.9.3 Overrun Error

When all bits in the next data are received while unread data are still in RD0BUF, overrun error flag UART0SR<OERR> is set to “1”. In this case, the receive data is discarded; data in RD0BUF are not affected. The UART0SR<OERR> is cleared to “0” when the RD0BUF is read after reading the UART0SR.

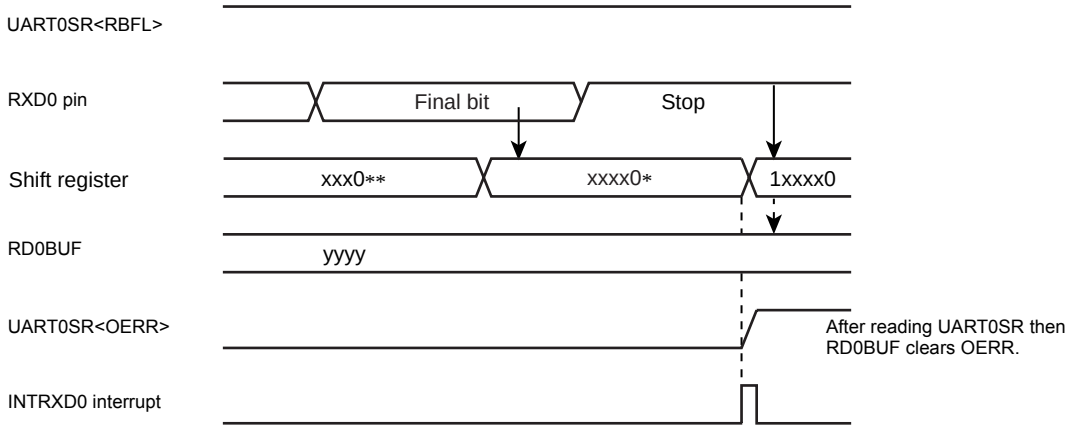


Figure 11-7 Generation of Overrun Error

Note: Receive operations are disabled until the overrun error flag UART0SR<OERR> is cleared.

11.9.4 Receive Data Buffer Full

Loading the received data in RD0BUF sets receive data buffer full flag UART0SR<RBFL> to "1". The UART0SR<RBFL> is cleared to “0” when the RD0BUF is read after reading the UART0SR.

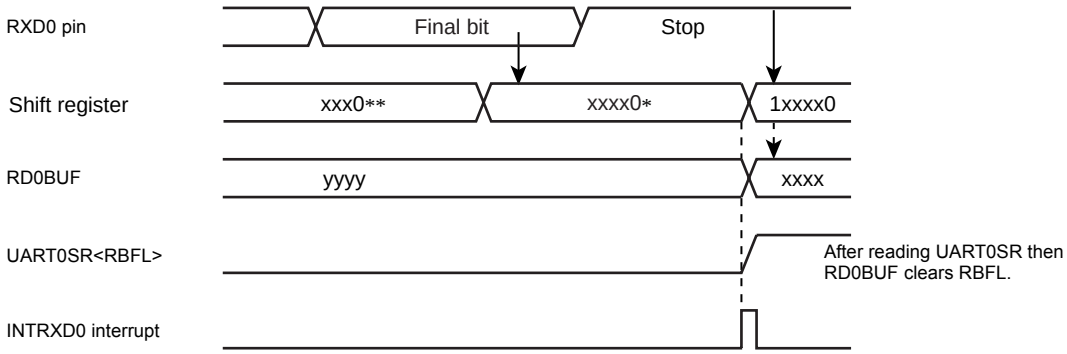


Figure 11-8 Generation of Receive Data Buffer Full

Note: If the overrun error flag UART0SR<OERR> is set during the period between reading the UART0SR and reading the RD0BUF, it cannot be cleared by only reading the RD0BUF. Therefore, after reading the RD0BUF, read the UART0SR again to check whether or not the overrun error flag which should have been cleared still remains set.

11.9.5 Transmit Data Buffer Empty

When no data is in the transmit buffer TD0BUF, UART0SR<TBEP> is set to “1”, that is, when data in TD0BUF are transferred to the transmit shift register and data transmit starts, transmit data buffer empty flag UART0SR<TBEP> is set to “1”. The UART0SR<TBEP> is cleared to “0” when the TD0BUF is written after reading the UART0SR.

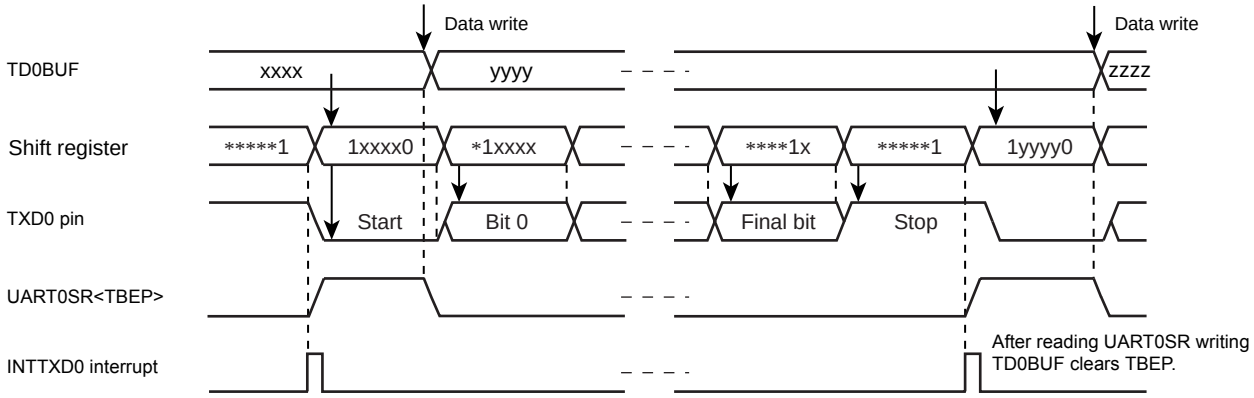


Figure 11-9 Generation of Transmit Data Buffer Empty

11.9.6 Transmit End Flag

When data are transmitted and no data is in TD0BUF (UART0SR<TBEP> = "1"), transmit end flag UART0SR<TEND> is set to "1". The UART0SR<TEND> is cleared to "0" when the data transmit is stated after writing the TD0BUF.

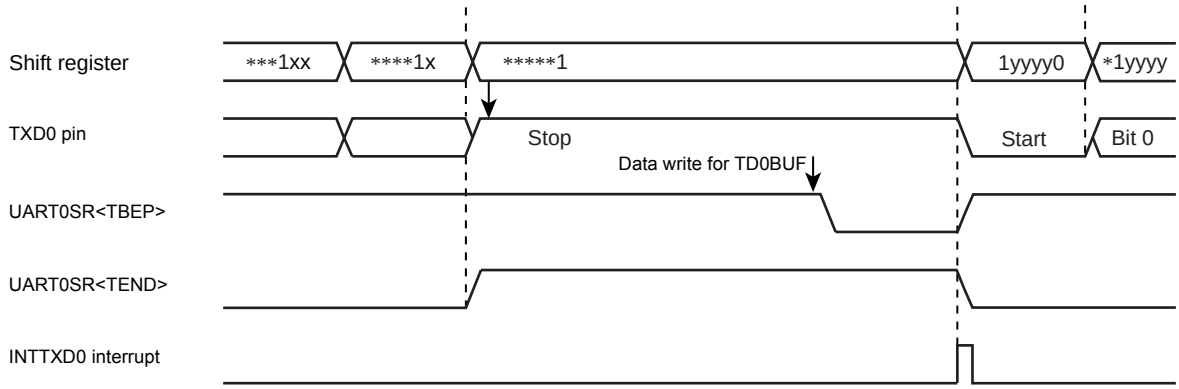


Figure 11-10 Generation of Transmit End Flag and Transmit Data Buffer Empty

12. Asynchronous Serial interface (UART1)

12.1 Configuration

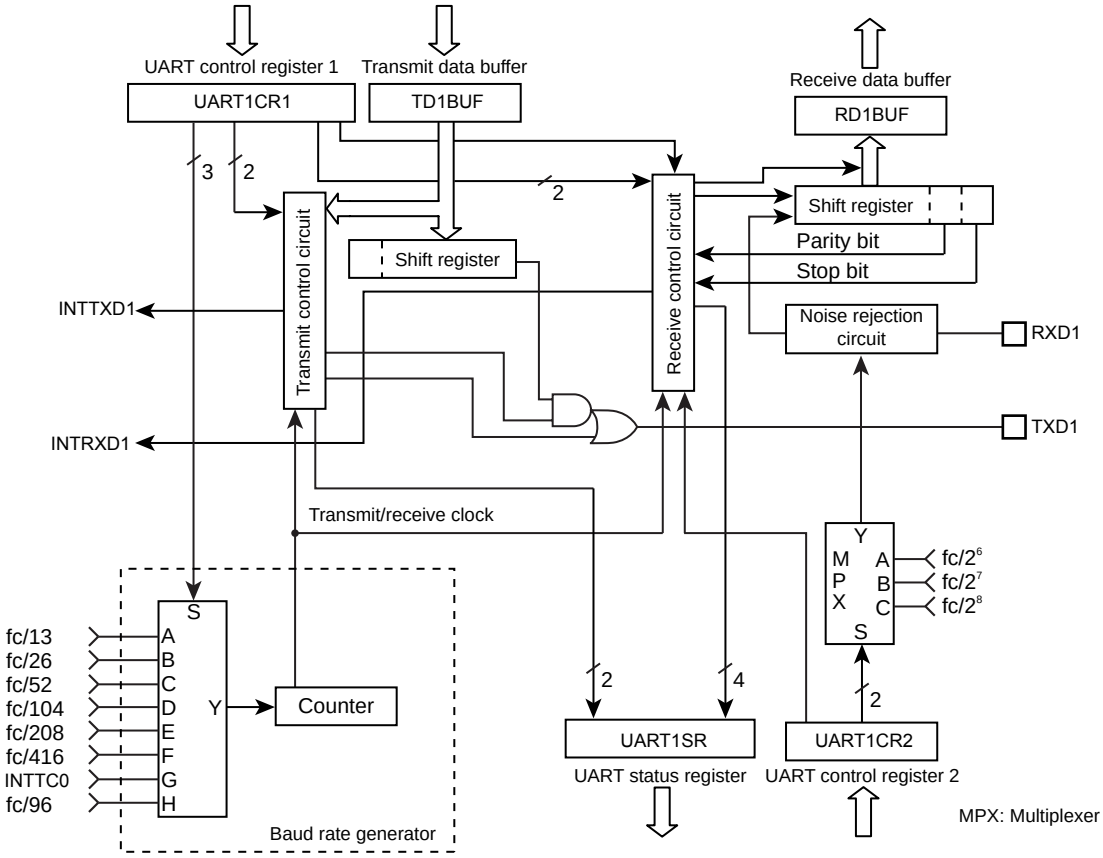


Figure 12-1 UART1 (Asynchronous Serial Interface)

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 (001EH)

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

(Initial value: 0000 0000)

TXE	Transfer operation	0: Disable 1: Enable	Write only
RXE	Receive operation	0: Disable 1: Enable	
STBT	Transmit stop bit length	0: 1 bit 1: 2 bits	
EVEN	Even-numbered parity	0: Odd-numbered parity 1: Even-numbered parity	
PE	Parity addition	0: No parity 1: Parity	
BRG	Transmit clock select	000: fc/13 [Hz] 001: fc/26 010: fc/52 011: fc/104 100: fc/208 101: fc/416 110: TC0 (Input INTTC0) 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 (001FH)

7	6	5	4	3	2	1	0
					RXDNC	STOPBR	

(Initial value: **** *000)

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

Note: When 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 (001EH)

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

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

UART1 Receive Data Buffer

RD1BUF (001DH)

7	6	5	4	3	2	1	0

Read only

(Initial value: 0000 0000)

UART1 Transmit Data Buffer

TD1BUF (001DH)

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

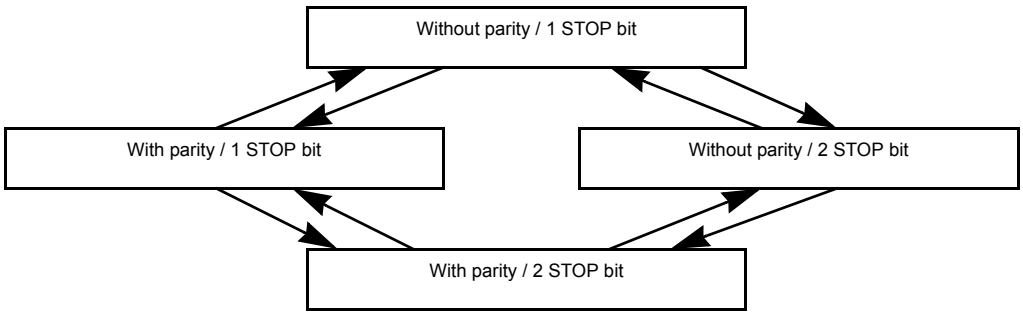


Figure 12-3 Caution on Changing Transfer Data Format

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 TC0 is used as the UART1 transfer rate (when UART1CR1<BRG> = “110”), the transfer clock and transfer rate are determined as follows:

Transfer clock [Hz] = TC0 source clock [Hz] / TTREG0 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).

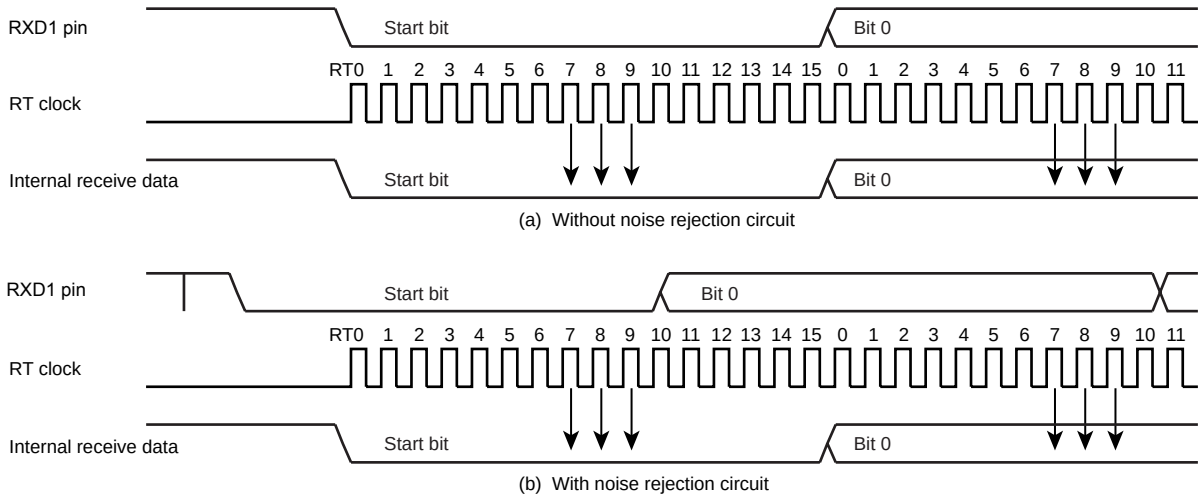


Figure 12-4 Data Sampling Method

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.

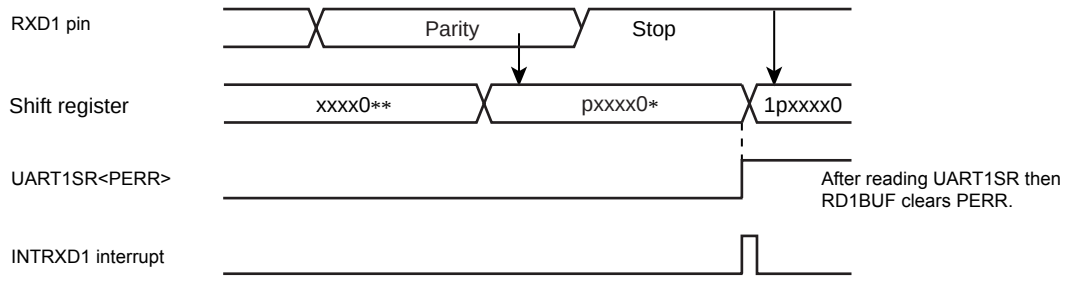


Figure 12-5 Generation of Parity Error

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.

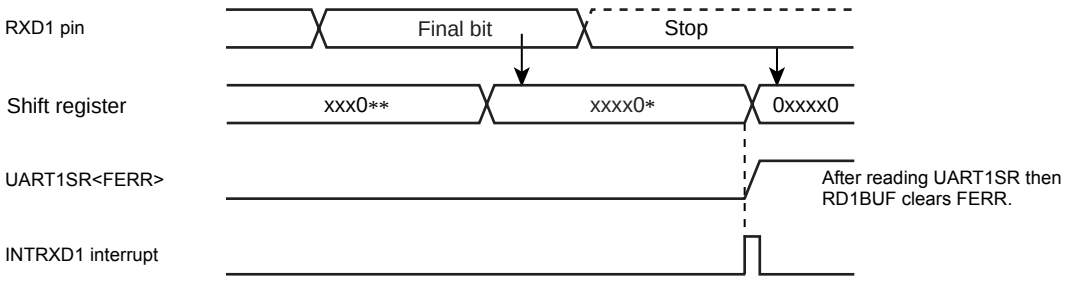


Figure 12-6 Generation of Framing Error

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.

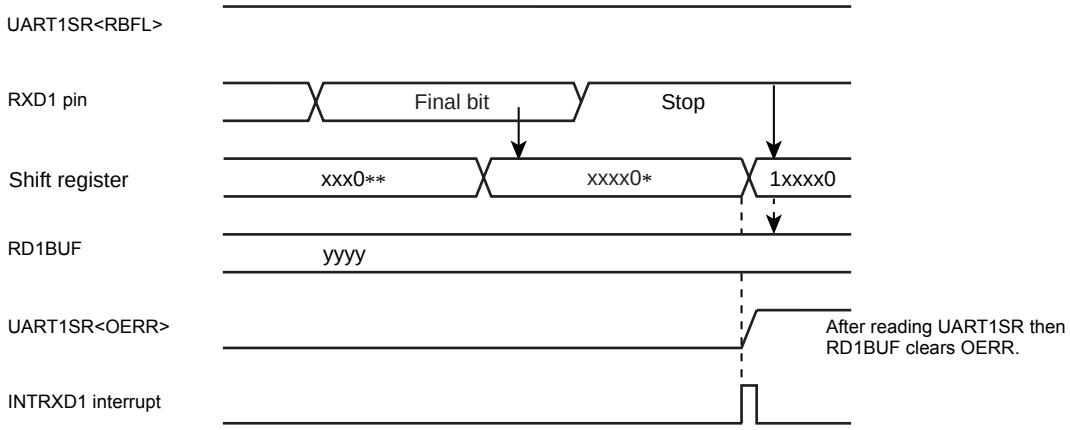


Figure 12-7 Generation of Overrun Error

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.

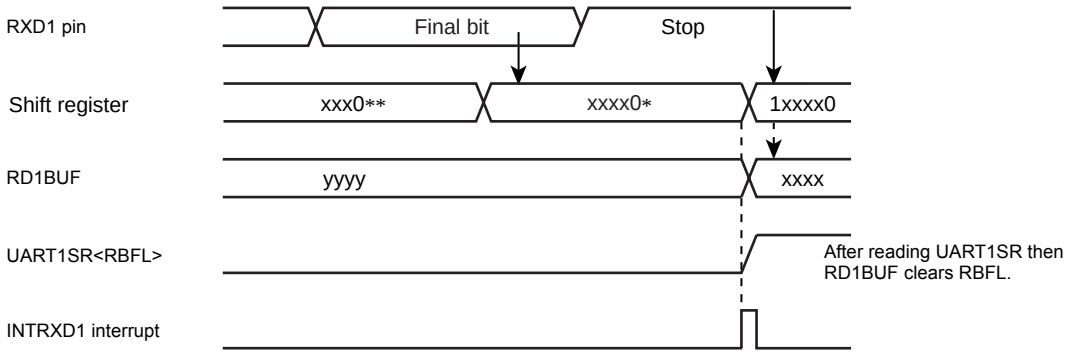


Figure 12-8 Generation of Receive Data Buffer Full

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, UART1SR<TBEP> is set to “1”, 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.

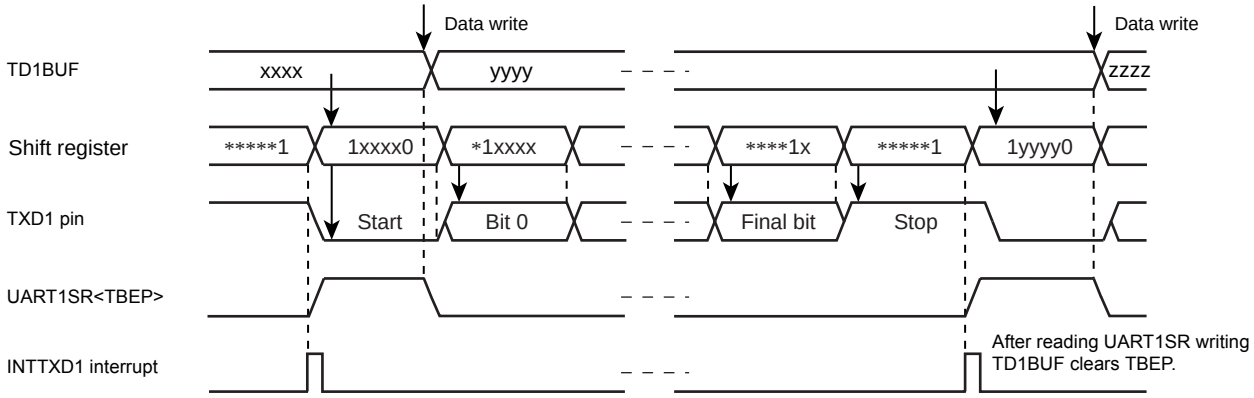


Figure 12-9 Generation of Transmit Data Buffer Empty

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 stated after writing the TD1BUF.

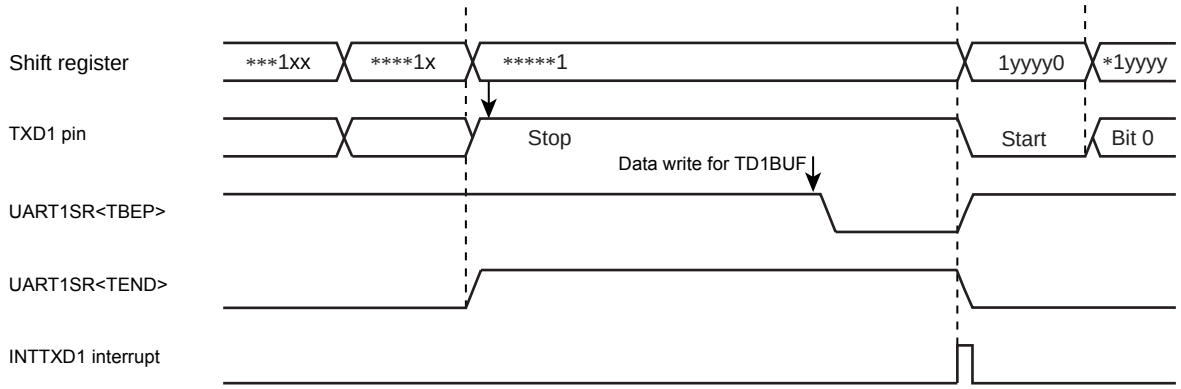
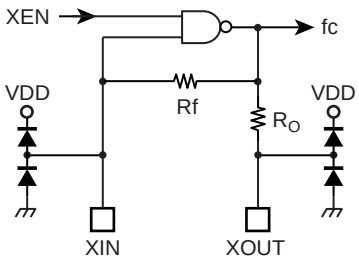
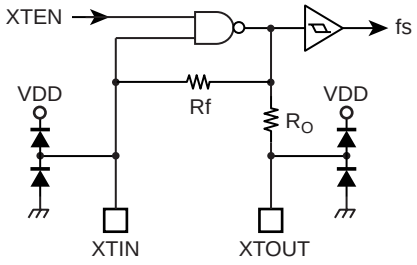
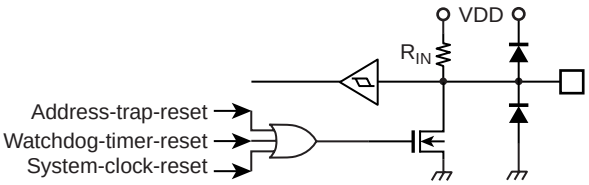
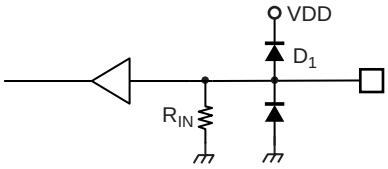
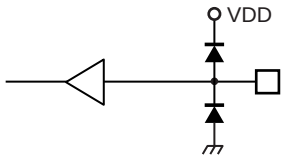


Figure 12-10 Generation of Transmit End Flag and Transmit Data Buffer Empty

13. Input/Output Circuitry

13.1 Control Pins

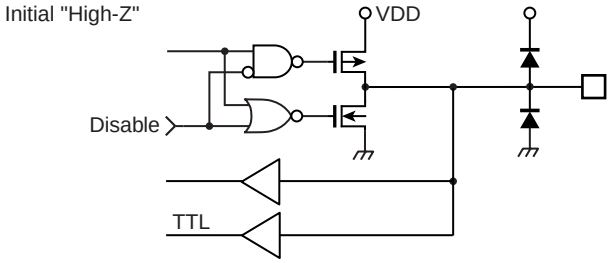
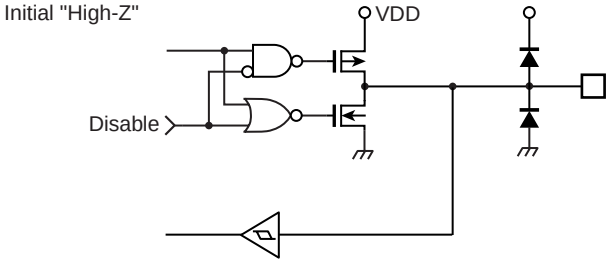
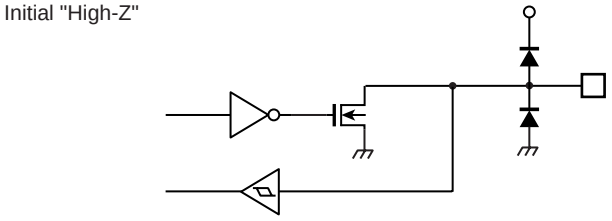
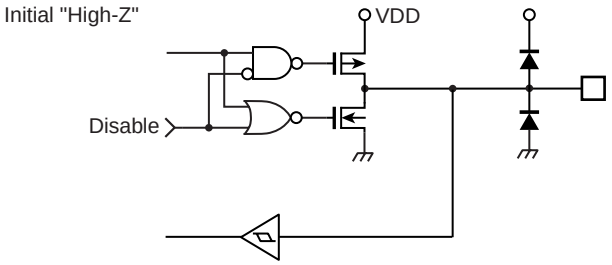
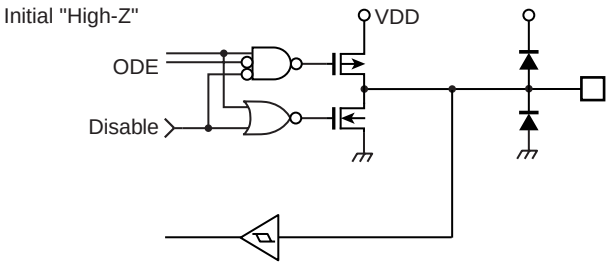
The input/output circuitries of the TMP86CH06AUG control pins are shown below.

Control Pin	I/O	Input/Output Circuitry	Remarks
XIN XOUT	Input Output		Resonator connecting pins (high-frequency) R_f = 1.2 MΩ (typ.) R_O = 1 kΩ (typ.)
XTIN XTOUT	Input Output		Resonator connecting pins (low-frequency) R_f = 6 MΩ (typ.) R_O = 220 kΩ (typ.)
$\overline{\text{RESET}}$	I/O		Sink open drain output Hysteresis input Built in Pull-up resistor R_{IN} = 220 kΩ (typ.)
TEST	Input		Built in Pull-down resistor R_{IN} = 70 kΩ (typ.)
$\overline{\text{EA}}$	Input		

Note: The TMP86PH06 does not have a pull-down resistor (R_{IN}) and a diode (D_1) for TEST pin. Be sure to fix the TEST pin to low level in MCU mode.

13.2 Input/Output Ports

The input/output circuitries of the TMP86CH06AUG input/output ports are shown below.

Port	I/O	Input/Output Circuitry	Remarks
P0	I/O		Tri-state I/O Nch. High-currency output
P1	I/O		Tri-state I/O Hysteresis input
P2	I/O		Sink open drain output Hysteresis input
P3	I/O		Tri-state I/O Hysteresis input
P4	I/O		Tri-state I/O Hysteresis input Programmable Open Drain output

14. Electrical Characteristics

14.1 Absolute Maximum Rating

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.

(V_{SS} = 0 V)

Parameter	Symbol	Pins	Rating	Unit
Supply Voltage	V_{DD}		−0.3 to 6.5	V
Input Voltage	V_{IN}		−0.3 to VDD + 0.3	
Output Voltage	V_{OUT}		−0.3 to VDD + 0.3	
Output Current	I_{OUT1}	P1 to P4	3.2	mA
	I_{OUT2}	P0	30	
Output Current	ΣI_{OUT1}		80	
	ΣI_{OUT2}		120	
Power Dissipation (Topr = 85°C)	PD		350	mW
Soldering Temperature (Time)	Tsld		260 (10 s)	°C
Storage Temperature	Tstg		−55 to 125	
Operating Temperature	Topr		−40 to 85	

14.2 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	Conditions		Min	Max	Unit
Supply Voltage	V_{DD}		fc = 16 MHz	NORMAL1, 2 mode	4.5	5.5	V
				IDLE0, 1, 2 mode			
			fc = 8 MHz	NORMAL1, 2 mode	2.7		
				IDLE0, 1, 2 mode			
			fc = 4.2 MHz	NORMAL1, 2 mode	1.8 (Note2)		
				IDLE0, 1, 2 mode			
			fs = 32.768 kHz	SLOW1, 2 mode			
				SLEEP0, 1, 2 mode			
				STOP mode			
Input High Voltage	V_{IH1}	Except hysteresis and TTL input	$V_{DD} \geq 4.5$ V		$V_{DD} \times 0.70$	V_{DD}	V
	V_{IH2}	Hysteresis input			$V_{DD} \times 0.75$		
	V_{IH3}	Except TTL input	$V_{DD} < 4.5$ V		$V_{DD} \times 0.90$		
	V_{IH4}	TTL input (Data bus)	V_{DD} = 5 V		2.2		
	V_{IH5}		V_{DD} = 1.8 V		$V_{DD} - 0.2$		
Input Low Voltage	V_{IL1}	Except hysteresis and TTL input	$V_{DD} \geq 4.5$ V		0	$V_{DD} \times 0.30$	V
	V_{IL2}	Hysteresis input				$V_{DD} \times 0.25$	
	V_{IL3}	Except TLL input	$V_{DD} < 4.5$ V			$V_{DD} \times 0.10$	
	V_{IL4}	TTL input (Data bus)	V_{DD} = 5 V			0.8	
	V_{IL5}		V_{DD} = 1.8 V			0.2	
Clock Frequency	fc	XIN, XOUT	V_{DD} = 4.5 V to 5.5 V		1.0	16	MHz
			V_{DD} = 2.7 V to 5.5 V			8	
			V_{DD} = 1.8 V to 5.5 V			4.2	
	fs	XTIN, XTOUT			30.0	34.0	kHz

Note 1: Clock Frequency fc; The condition of supply voltage range is the value under NORMAL1/2 and IDLE0/1/2 mode.
Note 2: When the supply voltage is V_{DD}=1.8 to 2.0V, the operating tempreture is Topr= -20 to 85 °C.

14.3 DC Characteristics

(V_{SS} = 0 V, Topr = −40 to 85°C)

Parameter	Symbol	Pins	Conditions	Min	Typ.	Max	Unit
Hysteresis Voltage	V_{HS}	Hysteresis input		–	0.9	–	V
Input Current	I_{IN1}	TEST, $\overline{EA}$	V_{DD} = 5.5 V V_{IN} = 5.5 V/0 V	–	–	±2	μA
	I_{IN2}	Sink Open Drain, Tri-state Port					
	I_{IN3}	$\overline{RESET}$, $\overline{STOP}$					
Input Resistance	R_{IN1}	$\overline{RESET}$		100	220	450	kΩ
	R_{IN2}	TEST		–	70	–	
OSC. Feedback Resistance	Rfx	XIN-XOUT		–	1.2	–	MΩ
	Rfxt	XTIN-XTOUT		–	6	–	
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	
"H" Output Voltage	V_{OH}	Tri-state Port	V_{DD} = 4.5 V, I_{OH} = −0.7 mA	4.1	–	–	V
"L" Output Voltage	V_{OL}	Except P0 and XOUT	V_{DD} = 4.5 V, I_{OL} = 1.6 mA	–	–	0.4	V
"L" Output Current	I_{OL1}	Except P0 and XOUT	V_{DD} = 4.5 V, V_{OL} = 0.4 V	1.6	–	–	mA
	I_{OL2}	P0	V_{DD} = 4.5 V, V_{OL} = 1.0 V	–	20	–	
Supply Current under NORMAL1, 2 mode	I_{DD}		V_{DD} = 5.5 V V_{IN} = 5.3 V/0.2 V fc = 16 MHz fs = 32.768 kHz	–	5.5	7.0	mA
Supply Current under IDLE1, 2 mode				–	2.8	3.5	
Supply Current under NORMAL1, 2 mode			V_{DD} = 5.5 V V_{IN} = 5.3 V/0.2 V fc = 8 MHz fs = 32.768 kHz	–	4.0	5.0	mA
Supply Current under IDLE1, 2 mode				–	2.0	2.5	
Supply Current under SLOW1 mode			V_{DD} = 3.0 V V_{IN} = 2.8 V/0.2 V fs = 32.768 kHz	–	14	25	μA
Supply Current under SLEEP1 mode				–	7.0	15	μA
Supply Current under SLEEP0 mode				–	6.0	15	μA
Supply Current under STOP mode			V_{DD} = 5.5 V V_{IN} = 5.3 V/0.2 V	–	0.5	10	μA

Note 1: Typical values are shown under Topr = 25°C, V_{DD} = 5 V, while conditions are not stated.

Note 2: Input current I_{IN1}, I_{IN3}: The current through pull-up or pull-down resistor is not included.

14.4 AC Characteristics

14.4.1 CLOCK

(V_{SS} = 0 V, V_{DD} = 4.5 to 5.5 V, Topr = –40 to 85°C)

Parameter	Symbol	Conditions	Min	Typ.	Max	Unit
Machine Cycle Time	tcy	NORMAL1, 2 mode	0.25	–	4	μs
		IDLE0, 1, 2 mode				
		SLOW1, 2 mode	117.6	–	133.3	
		SLEEP0, 1, 2 mode				
High Level Clock Pulse Width	t_{WCH}	External clock operation (XIN input) fc = 16 MHz	25	–	–	ns
Low Level Clock Pulse Width	t_{WCL}					
High Level Clock Pulse Width	t_{WSH}	External clock operation (XTIN input) fs = 32.768 kHz	14.7	–	–	μs
Low Level Clock Pulse Width	t_{WSL}					

14.4.2 External Memory Interface (Multiplexed Bus)

(V_{DD} = 4.5 to 5.5)

No.	Symbol	Parameter	Variable		16 MHz		Unit
			Min	Max	Min	Max	
1	t_{AL}	A7 to 0 effective → ALE	0.5t – 15		16		ns
2	t_{LA}	ALE fall → A7 to 0 hold	0.5t – 20		11		ns
3	t_{LL}	ALE pulse width	t – 40		22		ns
4	t_{LC}	ALE fall → $\overline{RD}$, $\overline{WR}$ fall	0.5t – 25		6		ns
5	t_{CL}	$\overline{RD}$, $\overline{WR}$ rise → ALE rise	0.5t – 20		11		ns
6	t_{ACL}	A7 to 0 effective → $\overline{RD}$, $\overline{WR}$ fall	t – 25		37		ns
7	t_{ACH}	A15 to 8 effective → $\overline{RD}$, $\overline{WR}$ fall	1.5t – 35		58		ns
8	t_{CA}	$\overline{RD}$, $\overline{WR}$ rise → A15 to 8 hold	0.5t – 32		0		ns
9	t_{ADL}	A7 to 0 effective → D7 to 0 input		3t – 55		132	ns
10	t_{ADH}	A15 to 8 effective → D7 to 0 input		3.5t – 65		153	ns
11	t_{RD}	$\overline{RD}$ fall → D7 to 0 input		2t – 60		65	ns
12	t_{RR}	$\overline{RD}$ pulse width	2t – 40		85		ns
13	t_{HR}	$\overline{RD}$ rise → D7 to 0 hold	0		0		ns
14	t_{RAE}	$\overline{RD}$ rise → A7 to 0 effective	t – 15		47		ns
15	t_{WW}	$\overline{WR}$ pulse width	2t – 40		85		ns
16	t_{DW}	D7 to 0 effective → $\overline{WR}$ rise	2t – 40		85		ns
17	t_{WD}	$\overline{WR}$ rise → D7 to 0 hold	0.5t – 15		16		ns

Note: t = tcy/4 (t = 62.5 ns at fcgck = 16 MHz)

A.C.Measurement Condition

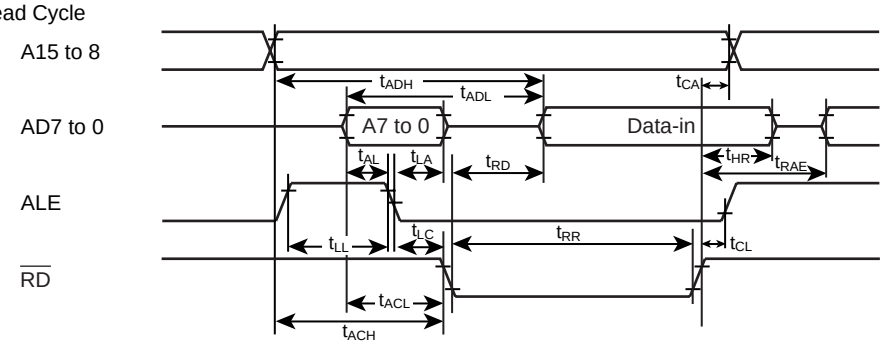
Output Level (ALE only) : High 2.2 V/Low VDD/2, CL = 50 pF

Output Level (except ALE) : High 2.2 V/Low 0.8 V, CL = 50 pF

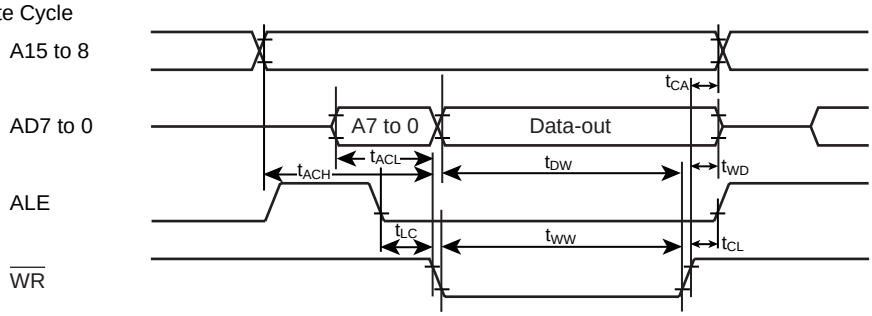
Input Level: High 2.4 V/Low 0.4 V (D7 to D0)

High 0.8 VDD/Low 0.2 VDD (Except D7 to D0)

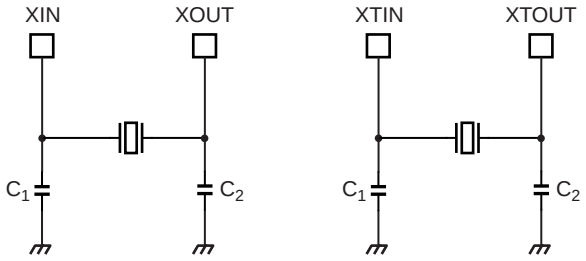
Read Cycle



Write Cycle



14.5 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 V_{DD} is 2.7 V or above. If V_{DD} 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: For the resonators to be used with Toshiba microcontrollers, we recommend ceramic resonators manufactured by Murata Manufacturing Co., Ltd.
For details, please visit the website of Murata at the following URL:
http://www.murata.com

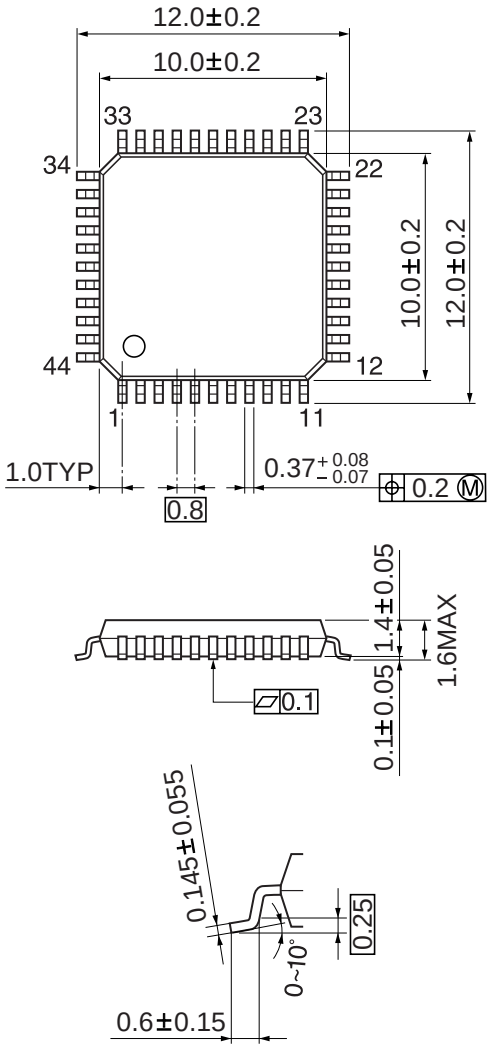
14.6 Handling Precaution

- 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.

15. Package Dimension

P-LQFP44-1010-0.80B 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.