CMOS 8-bit Microcontroller

TMP87C409BNG, TMP87C409BMG, TMP87C809BNG, TMP87C809BMG

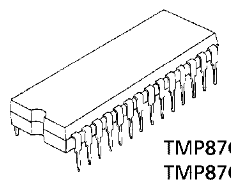
The TMP87C409B/809B are the high speed and high performance 8-bit single chip microcomputers. These MCU contain CPU core, ROM, RAM, input/output ports, three multi-function timer/counters, a 10-bit AD conveter, on a chip. The TMP87C409B/809B provide high current output capability for LED direct drive.

Product No.	ROM	RAM	Package	OTP MCU
TMP87C409BNG	4 K × 8 bits	256 × 8 bits	P-SDIP28-400-1.78	TMP87P809NG
TMP87C409BMG			P-SOP28-450-1.27	TMP87P809MG
TMP87C809BNG	8 K × 8 bits		P-SDIP28-400-1.78	TMP87P809NG
TMP87C809BMG			P-SOP28-450-1.27	TMP87P809MG

Features

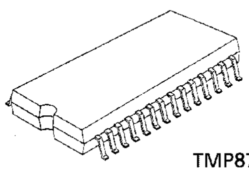
- ◆ 8-bit single chip microcomputer TLCS-870 Series
- ◆ instruction execution time: 0.5 μs (at 8 MHz)
- ◆ 412 basic instruction
  - Multiplication and Division (8 bits × 8 bits, 16 bits ÷ 8 bits)
  - Bit manipulations (Set/Clear/Complement/Load/Store/Test/Exclusive or)
  - 16-bit data operations
  - 1-byte jump/subroutine-call (Short relative jump/Vector call)
- ◆ 11 interrupt sources (External: 4, Internal: 7)
  - All sources have independent latches each, and nested interrupt control is available.
  - 2 edge-selectable external interrupts with noise reject.
  - High-speed task switching by register bank changeover
- ◆ 3 Input/Output ports (22 pins)
  - High current output: 6 pins (Typ. 20 mA)
- ◆ 16-Bit Timer/Counter
  - Timer, Event counter modes

P-SDIP28-400-1.78



TMP87C409BNG
TMP87C809BNG

P-SOP28-450-1.27

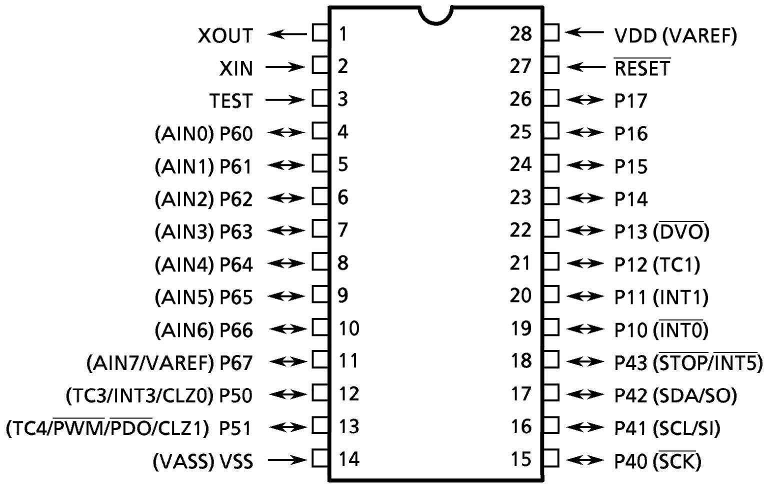


TMP87C409BMG
TMP87C809BMG

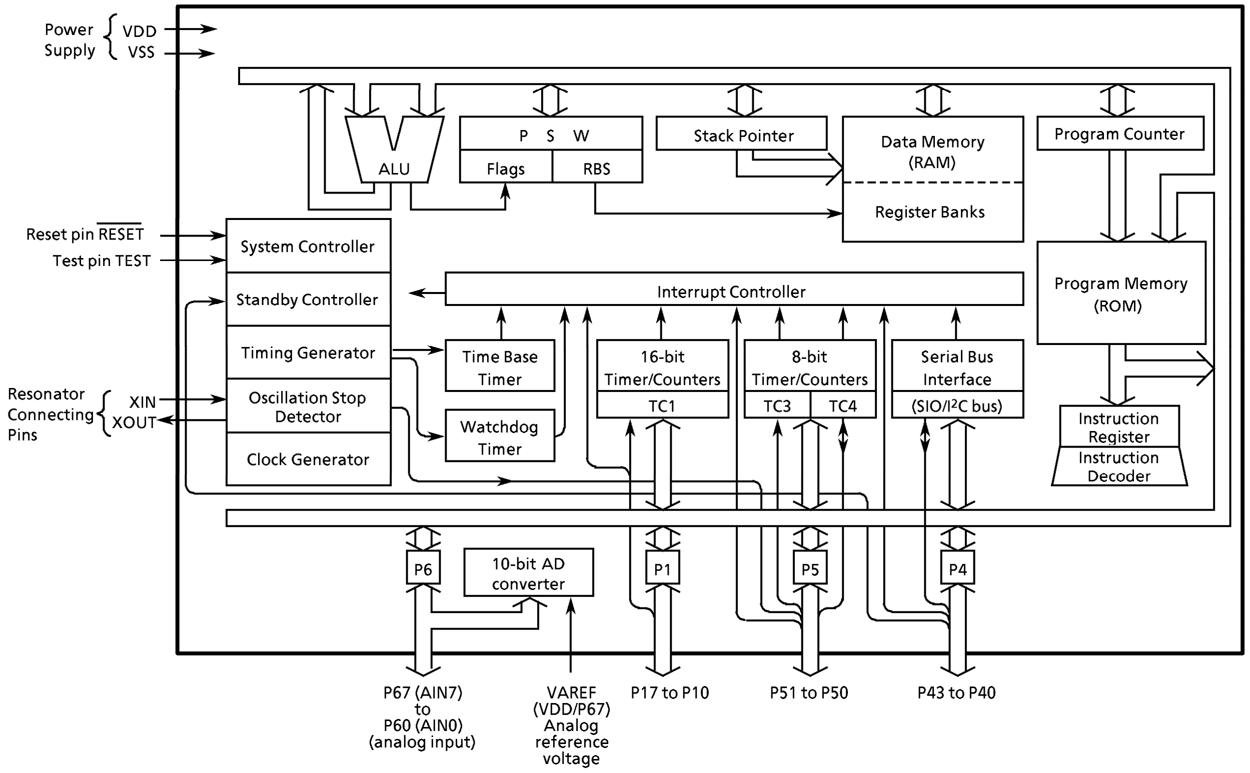
- ◆ Two 8-Bit Timer/Counters
  - ● Timer, Event counter, Capture (Pulse width/duty measurement), PWM output, Programmable divider output modes.
- ◆ Time Base Timer (Interrupt frequency: 1 Hz to 16 kHz)
- ◆ Divider output function (frequency: 1 kHz to 8 kHz)
- ◆ Watch dog Timer
- ◆ Serial bus Interface (SBI-ver. B)
  - ● I^2C bus, 8-bit SIO modes.
- ◆ 10-bit successive approximate type AD converter
  - ● 8 analog inputs
- ◆ Two Oscillation Stop Detector outputs (High-impedance output)
- ◆ Two Power saving operating modes
  - ● STOP mode: Oscillation stops. Battery/Capacitor back-up.
    Port output hold/high-impedance.
  - ● IDLE mode: CPU stops, and Peripherals operate using high-frequency clock. Release by interrupts.
- ◆ Wide operating voltage: 2.2 to 5.5 V at 4.2 MHz/4.5 to 5.5 V at 8 MHz
- ◆ Emulation Pod: BM87C809N0A

Pin Assignments (Top View)

P-SDIP28-400-1.78 / P-SOP28-450-1.27



Block Diagram



Pin Function

Pin Name	Input / Output	Function	
P17 to P14	I/O	8-bit programmable input/output ports (tri-state). Each bit of the port can be individually configured as an input or an output under software control. When used as an external input or a timer counter input, the input mode is configured. When used as an divider output, the latch must be set to “1”.	
P13 ($\overline{\text{DVO}}$)	I/O (Output)		Divider output
P12 (TC1)	I/O (Input)		Timer/Counter 1 input
P11 (INT1)			External interrupt input 1
P10 ($\overline{\text{INT0}}$)			External interrupt input 0
P43 ($\overline{\text{STOP}}$/$\overline{\text{INT5}}$)	I/O (Input/Input)	4-bit input/output port with latch (high current output). When used as an input port, an I^2C input/output or an external interrupt input the latch must be set to “1”.	STOP mode release input/External interrupt 5 input
P42 (SDA/SO)	I/O (I/O/Output)		I^2C bus serial data input/output or SIO serial data output
P41 (SCL/SI)	I/O (I/O/Input)		I^2C bus serial clock input/output or SIO serial data input
P40 ($\overline{\text{SCK}}$)	I/O (I/O)		SIO serial clock input/output
P51 (TC4/$\overline{\text{PWM}}$/$\overline{\text{PDO}}$/CLZ1)	I/O (Input/Output /Output/Output))	2-bit programmable input/output ports (tri-state, high current output). Each bit of the port can be individually configured as an input or an output under software control. When used as a timer counter input or an external interrupt input the input mode is configured. When used as a PWM/PDO output, the latch must be set to “1” and the output mode is configured. When used as a oscillation stop detector output, the output mode is configured.	Timer/counter 4 input or 8-bit PWM output or 8-bit PDO output or oscillation stop detector output 1
P50 (TC3/INT3/CLZ0)	I/O (Input/Input /Output)		Timer/counter 3 input or external interrupt input 3 or oscillation stop detector output 0
P67 (AIN7/VAREF)	I/O (Input/Input)	8-bit programmable input/output ports (tri-state). Each bit of the port can be individually configured as an input or an output under software control. When used as an analog input or an analog reference power supply, the input mode is configured.	AD converter analog input or analog reference power supply
P66 (AIN6) to P60 (AIN0)	I/O (Input)		AD converter analog inputs
XIN, XOUT	Input, Output	Resonator connecting pins for high-frequency clock. For inputting external clock, XIN is used and XOUT is opend.	
$\overline{\text{RESET}}$	Input	Reset signal input.	
TEST	Input	Test pint for out-going test. Be tied to low.	
VDD (VAREF)	Power Supply	+ 5 V	Analog reference power supply
VSS (VASS)		0 V (GND)	Analog reference GND

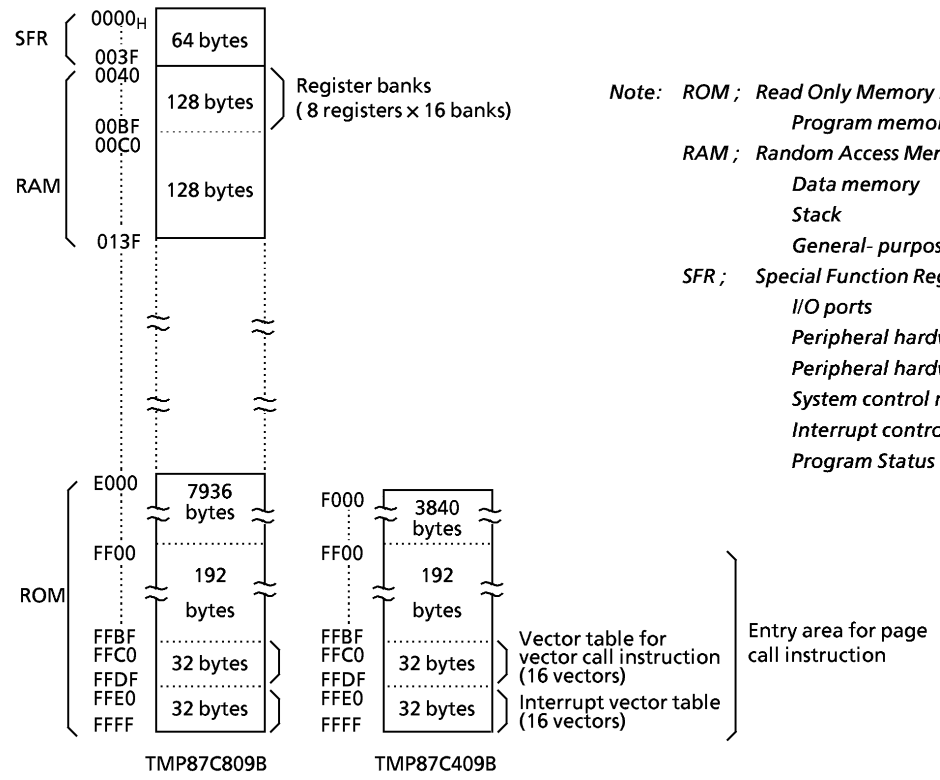
Operational Description

1. CPU Core Functions

The CPU core consists of a CPU, a system clock controller, an interrupt controller, and a watchdog timer. This section provides a description of the CPU core, the program memory, the data memory, and the reset circuit.

1.1 Memory Address Map

The TMP87C409B/809B are capable of addressing 64 Kbytes of memory. Figure 1-1 shows the memory address maps of the TMP87C409B/809B. In the TMP87C409B/809B the memory is organized 3 address spaces (ROM, RAM and SFR). It uses a memory mapped I/O system, and all I/O registers are mapped in the SFR address spaces. There are 16 banks of general-purpose registers. The register banks are also assigned to the RAM address space.



Note: ROM ; Read Only Memory includes:
Program memory
RAM ; Random Access Memory includes:
Data memory
Stack
General- purpose register banks
SFR ; Special Function Register includes:
I/O ports
Peripheral hardware control registers
Peripheral hardware status registers
System control registers
Interrupt control registers
Program Status Word

Figure 1-1. Memory address map

1.2 Program Memory (ROM)

The TMP87C409B has a 4 Kbytes (addresses F000 to FFFF$_H$) , the TMP87C809B has a 8 Kbytes (address E000 to FFFF$_H$) of program memory (mask programmed ROM). Figure 1-2 shows a program memory map. Addresses FF00 to FFFF$_H$ of program memory is also used for a special purpose.

(1) Interrupt vector table (addresses FFE0 to FFFF$_H$)
This table consists of a reset vector and 16 interrupt vectors (2 bytes/vector). These vectors store a reset start address and interrupt service routine entry addresses.

(2) Vector table for vector call instructions (addresses FFC0 to FFDF$_H$)
This table stores call vectors (subroutine entry address, 2 bytes/vector) for the vector call instructions [CALLV a]. There are 16 vectors. The CALLV instruction increases memory efficiency when utilized for frequently used subroutine calls (called from 3 or more locations).

(3) Entry area(addresses FF00 to FFFF$_H$) for page call instructions
This is the subroutine entry address area for the page call instructions [CALLP a]. Addresses FF00 to FFBF$_H$ are normally used because addresses FFC0 to FFFF$_H$ are used for the vector tables.

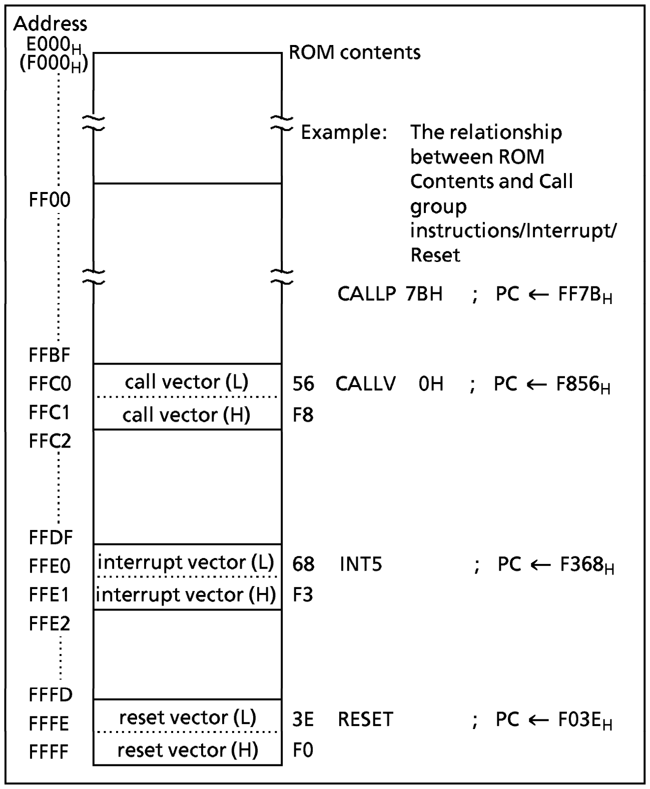
Programs and fixed data are stored in the program memory. The instruction to be executed next is read from the address indicated by the current contents of the program counter (PC). There are relative jump and absolute jump instructions. The concepts of page or bank boundaries are not used in the program memory concerning any jump instruction.

Example: The relationship between the jump instructions and the PC.

① 5-bit PC-relative jump [JRS cc, \$ + 2 + d]
F8C4H: JRS T, \$ + 2 + 08H
When JF = 1, the jump is made to F8CE$_H$, which is 08$_H$ added to the contents of the PC. (The PC contains the address of the instruction being executed + 2; therefore, in this case, the PC contents are F8C4$_H$ + 2 = F8C6$_H$.)

② 8-bit PC-relative jump [JR cc, \$ + 2 + d]
F8C4H: JR Z, \$ + 2 + 80H
When ZF = 1, the jump is made to F846$_H$, which is FF80$_H$ (-128) added to the current contents of the PC.

③ 16-bit absolute jump [JP a]
F8C4H: JP 0F235H
An unconditional jump is made to address F235$_H$. The absolute jump instruction can jump anywhere within the entire 64-Kbytes space.



Address
E000H (F000H)
ROM contents
Example: The relationship between ROM Contents and Call group instructions/Interrupt/ Reset
FF00
CALLP 7BH ; PC ← FF7BH
FFBF
FFC0 call vector (L) 56 CALLV 0H ; PC ← F856H
FFC1 call vector (H) F8
FFC2
FFDF
FFE0 interrupt vector (L) 68 INT5 ; PC ← F368H
FFE1 interrupt vector (H) F3
FFE2
FFFD
FFFE reset vector (L) 3E RESET ; PC ← F03EH
FFFF reset vector (H) F0

Figure 1-2. Program memory map

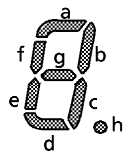
In the TLCS-870 Series, the same instruction used to access the data memory is also used to read out fixed data stored in the program memory. The register offset PC-relative addressing (PC + A) instructions can also be used, and the code conversion, table look-up and n-way multiple jump processing can easily be programmed.

Example 1: Loads the ROM contents at the address specified by the HL register pair contents into the accumulator (TMP87C809B: HL ≧ $E000_H$)

```
        LD      A, (HL)                 ;  A ← ROM (HL)
```

Example 2: Converts BCD to 7-segment code (common anode LED). When A = 05_H, 92_H is output to port P1 after executing the following program.

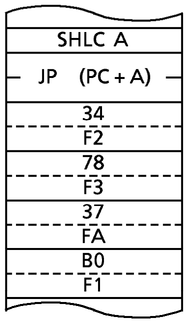
```
        ADD     A, TABLE − $ − 4        ;  P1 ←ROM (TABLE + A)
        LD      (P1), (PC + A)
        JRS     T, SNEXT                ;  Jump to SNEXT
TABLE:  DB      0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0D8H, 80H, 98H
SNEXT:
```



Note: "$" is a header address of ADD instruction. DB is a byte data definition instruction.

Example 3: N-way multiple jump in accordance with the contents of accumulator (0 ≦ A ≦ 3).

```
        SHLC    A                       ;  if A = 00H then PC ← F234H
        JP      (PC + A)                   if A = 01H then PC ← F378H
                                           if A = 02H then PC ← FA37H
                                           if A = 03H then PC ← F1B0H
        DW      0F234H, 0F378H, 0FA37H, 0F1B0H
```



Note: DW is a word data definition instruction. Word = 2 bytes.

1.3 Program Counter (PC)

The program counter (PC) is a 16-bit register which indicates the program memory address where the instruction to be executed next is stored. After reset, the reset vector stored in the vector table (addresses FFFF and $FFFE_H$) is loaded into the PC; therefore, program execution is possible from any desired address. For example, when F0 and $3E_H$ are stored at addresses FFFF and $FFFE_H$, respectively, the execution starts from address $F03E_H$ after reset.

The TLCS-870 Series utilizes pipelined processing (instruction pre-fetch); therefore, the PC always indicates 2 addresses in advance. For example, while a 1-byte instruction stored at address $F123_H$ is being executed, the PC contains $F125_H$.

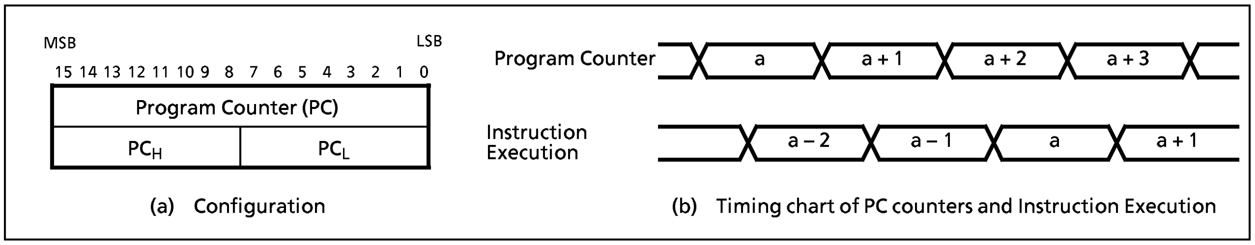


Figure 1-3. Program counter

1.4 Data Memory (RAM)

The TMP87C409B/809B have 256 bytes (addresses 0040 to 013F$_H$) of data memory (static RAM). Figure 1-4 shows the data memory map.

Addresses 0000 to 00FF$_H$ are used as a direct addressing area to enhance instructions which utilize this addressing mode; therefore, addresses 0040 to 00FF$_H$ in the data memory can also be used for user flags or user counters.

Example 1: If bit 2 at data memory address 00C0$_H$ is "1", 00$_H$ is written to data memory at address 00E3$_H$; otherwise, FF$_H$ is written to the data memory at address 00E3$_H$.

```
        TEST  (00C0H).2         ;  if (00C0H)2 = 0 then jump
        JRS   T,SZERO
        CLR   (00E3H)           ;  (00E3H) ← 00H
        JRS   T,SNEXT
SZERO:  LD    (00E3H), 0FFH     ;  (00E3H) ← FFH
SNEXT:
```

Example 2: Increments the contents of data memory at address 00F5$_H$, and clears to 00$_H$ when 10$_H$ is exceeded.

```
        INC   (00F5H)
        AND   (00F5H), 0FH
```

General-purpose register banks (8 registers × 16 banks) are also assigned to the 128 bytes of addresses 0040 to 00BF$_H$. Access as data memory is still possible even when being used for registers. For example, when the contents of the data memory at address 0040$_H$ is read out, the contents of the accumulator in the bank 0 are also read out.

The stack can be located anywhere within the data memory except the register bank area. For more details on the stack, see section "1.7 Stack and Stack Pointer".

With the TLCS-870 Series, programs in data memory cannot be executed. If the program counter indicates a data memory address (addresses 0040 to 013F$_H$), an address-trap-reset is generated due to bus error. (Internal reset is occered)

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 0 except the bank 0

```
            LD    HL, 0048H     ;  Sets start address to HL register pair
            LD    A, H          ;  Sets initial data (A)
            LD    BC, 00F7H     ;  Sets number of byte to BC register pair
SRAMCLR:    LD    (HL + ), A
            DEC   BC
            JRS   F, SRAMCLR
```

Note: "$" is a header address of ADD instruction. The general-purpose registers are mapped in the RAM; therefore, do not clear RAM at the current bank addresses. Clears RAM to 0 except the bank 0.

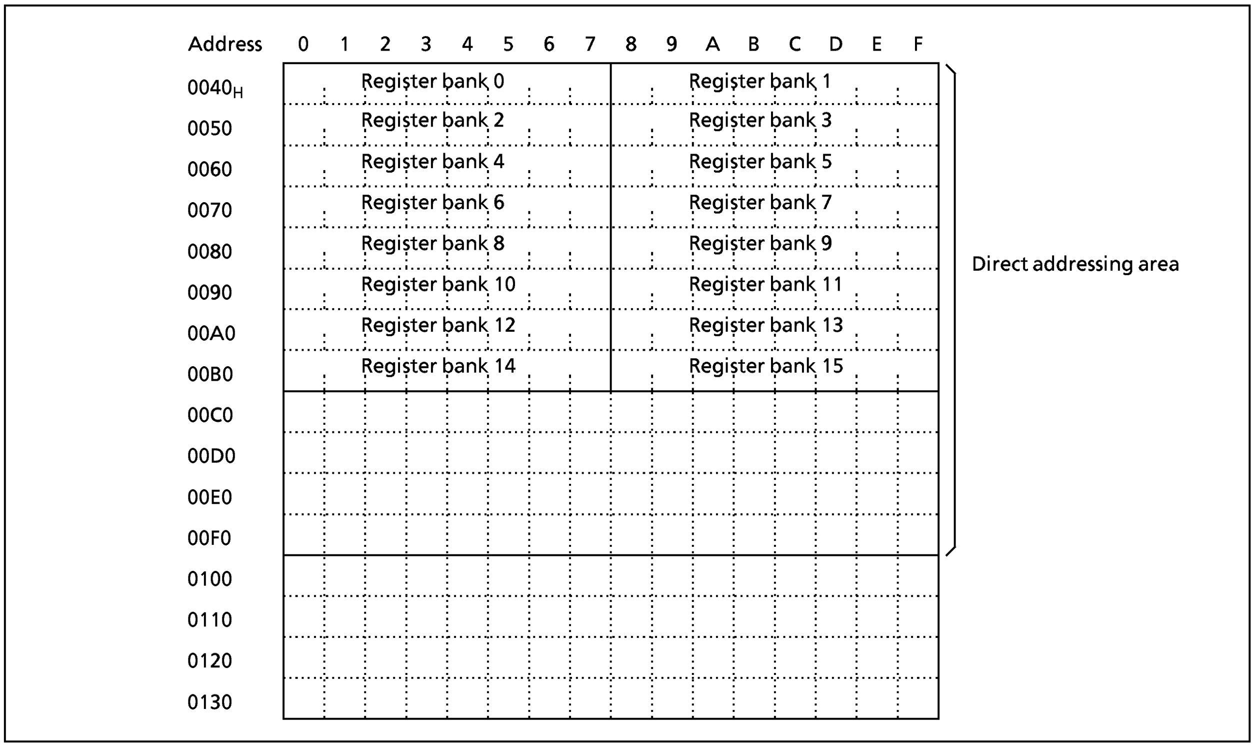


Figure 1-4. Data memory map

1.5 General-purpose Register Banks

General-purpose registers are mapped into addresses 0040 to 00BF$_H$ in the data memory. There are 16 register banks, and each bank contains eight 8-bit registers W, A, B, C, D, E, H, and L. Figure 1-5 shows the general-purpose register bank configuration. The unused register banks can be used as a data memory.

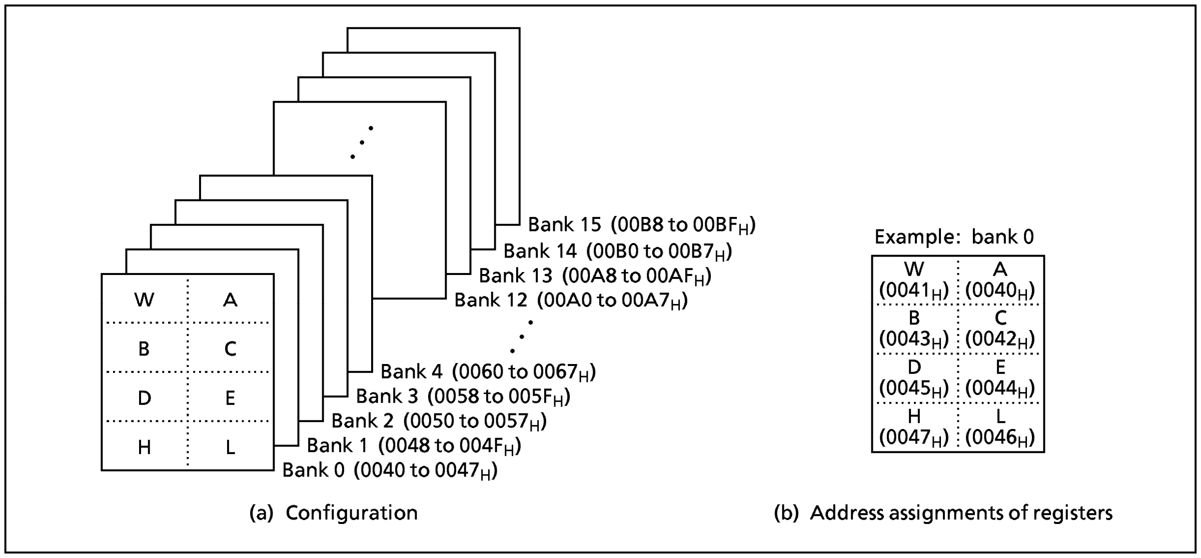


Figure 1-5. General-purpose register banks

In addition to access in 8-bit units, the registers can also be accessed in 16-bit units as the register pairs WA, BC, DE, and HL. Besides its function as a general-purpose register, the register also has the following functions.

(1) A, WA

The A register functions as an 8-bit accumulator and WA the register pair functions as a 16-bit accumulator (W is high byte and A is low byte). Registers other than A can also be used as accumulators for 8-bit operations.

```
Examples: ① ADD  A, B         ;  Adds B contents to A contents and stores the result into A.
          ② SUB  WA, 1234H    ;  Subtracts 1234H from WA contents and stores the result into
                                 WA.
          ③ SUB  E, A         ;  Subtracts A contents from B contents, and stores the result
                                 into E.
```

(2) HL, DE

The HL register functions as a data pointer/index register/base register, and the DE register pair function as a data pointer to specify the memory address.
HL also has an auto-post-increment and auto-pre-decrement functions. This function simplifies multiple digit data processing, software LIFO (last-in first-out) processing, etc.

```
Example 1: ① LD  A, (HL)       ;  Loads the memory contents at the address specified by HL
                                  into A.
           ② LD  A, (HL + 52H) ;  Loads the memory contents at the address specified by the
                                  value obtained by adding 52H to HL contents into A.
           ③ LD  A, (HL + C)   ;  Loads the memory contents at the address specified by the
                                  value obtained by adding the register C contents to HL
                                  contents into A.
           ④ LD  A, (HL + )    ;  Loads the memory contents at the address specified by HL
                                  into A. Then increments HL.
           ⑤ LD  A, ( − HL)    ;  Decrement HL. Then loads the memory contents at the
                                  address specified by new HL into A.
```

TLCS-870 Series can transfer data directly memory to memory, and operate directly between memory data and memory data. This facilitates the programming of block processing.

```
Example 2:   Block transfer
                LD   B, M          ;  m = n − 1 (n: Number of bytes to transfer)
                LD   HL, DSTA      ;  SEts destination address
                LD   DE, SRCA      ;  Sets source address
    SLOOP:      LD   (HL), (DE)    ;  (HL) ← (DE)
                INC  HL            ;  HL ← HL + 1
                INC  DE            ;  DE ← DE + 1
                DEC  B             ;  B ← B − 1
                JRS  F, SLOOP      ;  if B ≧ 0 then loop
```

(3) B, C, BC

Registers B and C can be used as 8-bit buffers or counter, and the BC register pair can be used as a 16-bit buffer or counter. The C register functions as an offset register for register offset index addressing (refer to example 1 ③ above) and as a divisor register for the division instruction.

```
   Example 1:   Repeat processing
                  LD    B, n          ; Sets n as the number of repetitions
     SREPEAT:     processing
                  DEC   B
                  JRS   F, SREPEAT

   Example 2:   Division (16-bit ÷ 8-bit)
                  DIV   WA, C         ; Divides the WA contents by the C contents, places the
                                        quotient in A and the remainder in W.
```

The general-purpose banks are selected by the 4-bit register bank selector (RBS). During reset, the RBS is initialized to "0". The bank selected by the RBS is called the current bank.
Together with the flag, the RBS is assigned to address $003F_H$ in the SFR as the program status word (PSW). There are 3 instructions [LD RBS, n], [PUSH PSW], [POP PSW] to access the PSW. The PSW can be also operated by the memory access instruction.

```
 Example 1:  Incrementing the RBS
                  INC   (003FH)       ; RBS ← RBS + 1

 Example 2:  Reading the RBS
                  LD    A, (003FH)    ; A ← RBS (The flags are simultaneously read in this
                                                instruction.)
```

Highly efficient programming and high speed task switching are possible by using bank changeover to save registers during interrupt and to transfer parameters during subroutine processing. During interrupt, the RBS is automatically saved onto the stack. The bank used before the interrupt is automatically restored by executing an interrupt return instruction [RETI]/[RETN] ; therefore, there is no need for the RBS save/restore software processing.
The TLCS-870 Series supports a maximum of 15 interrupt sources. One bank is assigned to the main program, and one bank can be assigned to each source. Also, to increase the efficiency of data memory usage, assign the same bank to interrupt sources which are not nested.

```
 Example:    Saving/restoring registers during interrupt task using bank changeover.
     PINT1:       LD    RBS, n        ; RBS ← n (Bank changeover)
                  Interrupt processing
                  RETI                ; Maskable interrupt return (Bank automatic restoring)
```

1.6 Program Status Word (PSW)

The program status word (PSW) consists of a register bank selector (RBS) and flags, and the PSW is assigned to address $003F_H$ in the SFR.
The RBS can be read and written using the memory access instruction, however the flags can only be read. When writing to the PSW, the change specified by the instruction is made without writing data to the flags. For example, when the instruction [LD (003FH), 05H] is executed, "5" is written to the RBS and the JF is set to "1", but the other flags are not affected.
During interrupt, PSW is saved to the stack with the program counter. The PSW is restored from the stack by executing return instructions [RETI]/[RETN].
[PUSH PSW] and [POP PSW] are the PSW access instructions.

1.6.1 Register bank selector (RBS)

The register bank selector (RBS) is a 4-bit register used to select general-purpose register banks. For example, when RBS = 2, bank 2 is currently selected. During reset, the RBS is initialized to "0".

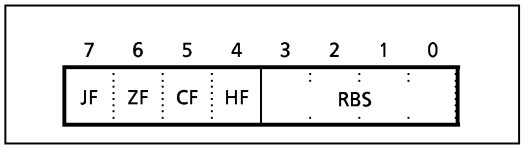


Figure 1-6. PSW (Flags, RBS) configuration

1.6.2 Flags (FLAG)

The flags are configured with the upper 4 bits: a zero flag, a carry flag, a half carry flag and a jump status flag. The flags are set or cleared under conditions specified by the instruction. These flags except the half carry flag are used as jump condition "cc" for conditional jump instructions [JR cc, \$ + 2 + d], [JRS cc, \$ + 2 + d]. After reset, the jump status flag is initialized to "1", other flags are not affected.

(1) **Zero flag (ZF)**

The ZF is set to "1" if the operation result or the transfer data is 00_H (for 8-bit operations and data transfers)/0000_H (for 16-bit operations); otherwise the ZF is cleared to "0".
During the bit manipulation instructions, the ZF is cleared to "0" if the contents of the specified bit is "1". This flag is set to "1" when the upper 8 bits of the product are 00_H during the multiplication instruction, and when 00_H for the remainder during the division instruction; otherwise it is cleared to "0".

(2) **Carry flag (CF)**

The CF is set to "1" when a carry occurred during addition or a borrow occurred during subtraction; otherwise the CF is cleared to "0". During division, this flag is set to "1" when the divisor is 00_H (divided by zero error), or when the quotient is 100_H or higher (quotiont-overflow error). The CF is also affected during the shift/rotate instructions. The data shifted out from a register is set to the CF.
This flag is also a 1-bit register (a boolean accumulator) for the bit manipulation instructions.
Set/clear/invert are possible with the CF manipulation instructions.

Example: Bit manipulation (The result of exclusive-OR between bit 5 content of address 07_H and bit 0 content of address $9A_H$ is written to bit 2 of address 01_H.)

```
LD    CF, (0007H) . 5         ;  (0001H)2 ← (0007H)5 ∀ (009AH)0
XOR   CF, (009AH) . 0
LD    (0001H) . 2, CF
```

(3) **Half carry flag (HF)**

The HF is set to "1" when a carry occurred to bit 4 of the operation result during an 8-bit addition, or when a borrow occurred from bit 4 of the result during an 8-bit subtraction. This flag is useful in the decimal adjustment for BCD operations (adjustments using the [DAA r], or [DAS r] instructions).

Example: BCD operation
(The A becomes 47_H after executing the following program when A = 19_H, B = 28_H.)

```
ADD  A, B        ;  A ← 41H, HF ← 1, CF = 0
DAA  A           ;  A ← 41H + 06H = 47H (decimal-adjust)
```

(4) **Jump status flag (JF)**

The JF is usually set to "1". Zero or carry information is set to the JF after operation. The JF provides the jump condition for conditional jump instructions [JR T/F, $ + 2 = d], [JRS T/F, $ + 2 + d] (T or F is a condition code).

Example: Jump status flag and conditional jump instruction

```
INC  A
JRS  T, SLABLE1   ;  Jump when a carry is caused by the immediately preceding
 ⋮                   operation instruction.
LD   A, (HL)
JRS  T, SLABLE2   ;  JF is set to "1" by the immediately preceding instruction,
 ⋮                   making it an unconditional jump instruction.
```

Example: The accumulator and flags becomes as shown below after executing the following instructions when the WA register pair, the HL register pair, the data memory at address $00C5_H$, the carry flag and the half carry flag contents being "$219A_H$", "$00C5_H$", "$D7_H$", "1", and "0", respectively.

Instruction	Accumulator after execution	Flag after execution			
		JF	ZF	CF	HF
ADDC A, (HL)	72	1	0	1	1
SUBB A, (HL)	C2	1	0	1	0
CMP A, (HL)	9A	0	0	1	0
AND A, (HL)	92	0	0	1	0
LD A, (HL)	D7	1	0	1	0
ADD A, 66H	00	1	1	1	1

Instruction	Accumulator after execution	Flag after execution			
		JF	ZF	CF	HF
INC A	9B	0	0	1	0
ROLC A	35	1	0	1	0
RORC A	CD	0	0	0	0
ADD WA, 0F508H	16A2	1	0	1	0
MUL W, A	13DA	0	0	1	0
SET A.5	BA	1	1	1	0

1.7 Stack, Stack Pointer

1.7.1 Stack

The stack provides the area in which the return address or status, etc. are saved before a jump is performed to the processing routine during the execution of a subroutine call instruction or the acceptance of an interrupt.

On a subroutine call instruction [CALL a] / [CALLP a] / [CALLVn], the return address is saved (the upper byte is pushed first, followed by the lower byte). During software interrupt instruction [SWI] execution or interrupt, the program status word is saved, then the return address is saved.

When returning from the processing routine, executing a subroutine return instruction [RET] restores the contents to the PC from the stack; executing an interrupt return instruction [RETI] / [RETN] restores the contents to the PC and the PSW.

The stack can be located anywhere within the data memory.

1.7.2 Stack pointer (SP)

The stack pointer (SP) is a 16-bit register to point out the first start address on the stack. The SP is post-decrement when a subroutine call or a push instruction is executed, or when an interrupt is accepted; and the SPC is pre-incremented when a return or a pop instruction is executed. The stack deepens to the direction of the lower address. Figure 1-8 shows the change of the stack access and the SP.

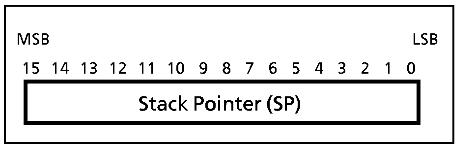


Figure 1-7. Stack pointer

The SP is not initialized hardware-wise but requires initialization by an initialize routine (sets the highest stack address). [LD SP, mn], [LD SP, gg] and [LD gg, SP] are the SP access instructions (mn; 16-bit immediate data, gg; register pair).

Example 1: To initialize the SP

```
        LD    SP, 013FH      ;  SP←013FH
```

Example 2: TO read the SP

```
        LD    HL, SP         ;  HL←SP
```

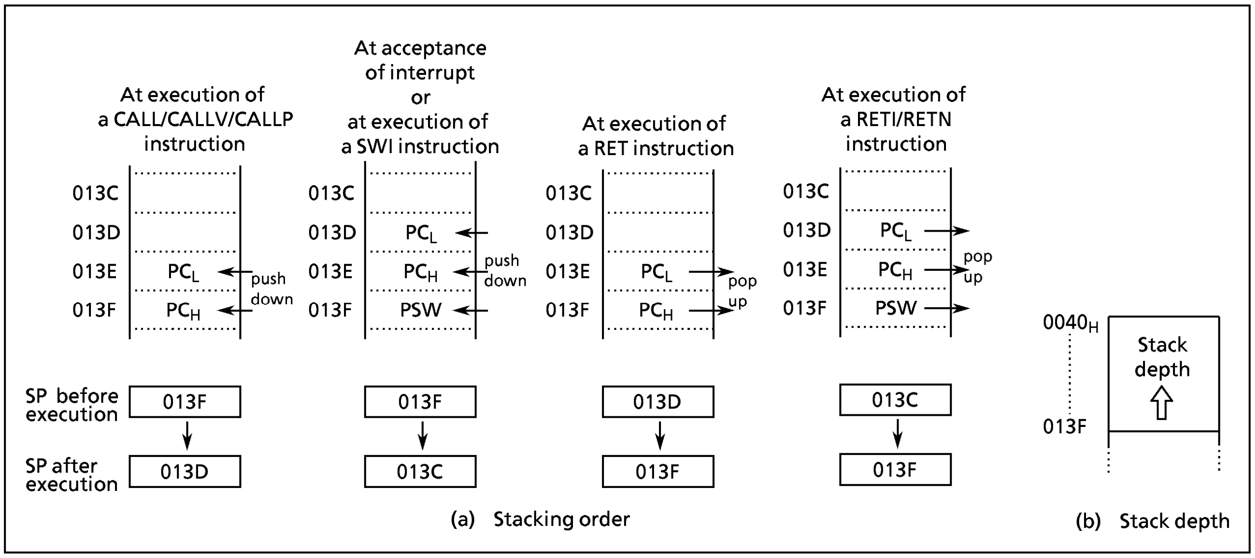


Figure 1-8. Stack

1.8 System Clock Controller

The system clock controller consists of a clock generator, a timing generator, and a stand-by controller.

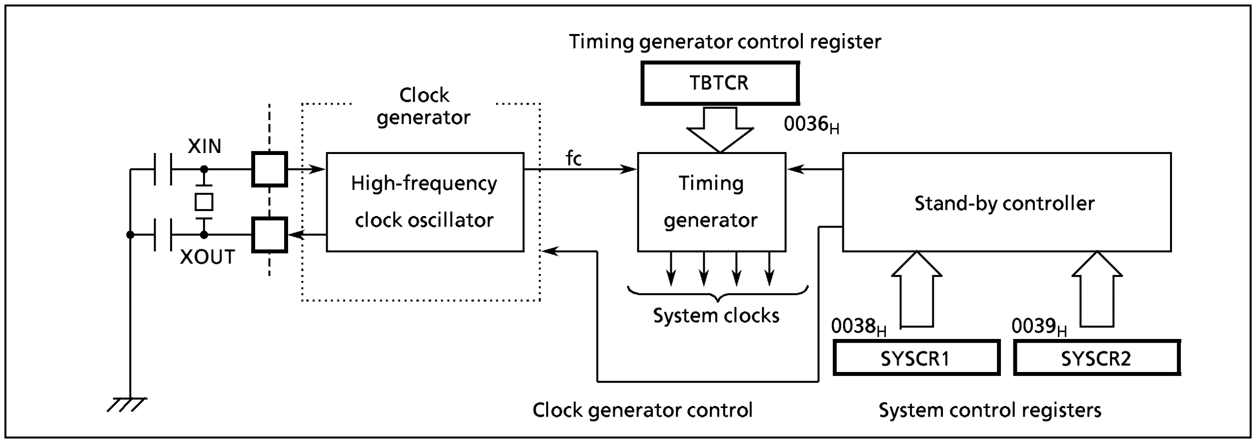


Figure 1-9. System clock controller

1.8.1 Clock generator

The clock generator generates the basic clock which provides the system clocks supplied to the CPU core and peripheral hardware.

The high-frequency (fc) clocks can be easily obtained by connecting a resonator between the XIN/XOUT pins. Clock input from an external oscillator is also possible.

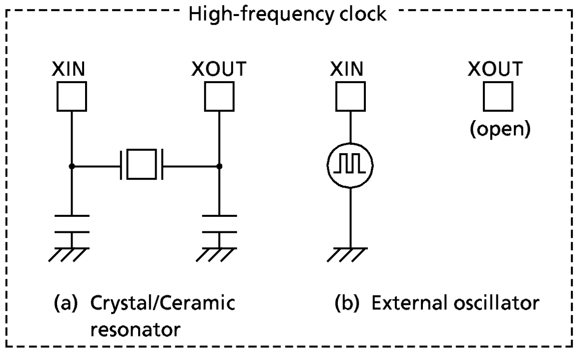


Figure 1-10. Example of resonator connection

Note: Accurate Adjustment of the Oscillation Frequency:
Although no hardware to externally and directly monitor the basic clock pulse is provided, the oscillation frequency can be adjusted by making the program to output fixed frequency pulses to the port while disabling all interrupts and monitoring this pulse. With a system requiring adjustment of the oscillation frequency, the adjusting program must be created beforehand

1.8.2 Timing generator

The timing generator generates from the basic clock the various system clocks supplied to the CPU core and peripheral hardware. The timing generator provides the following functions:

① Generation of main system clock
② Generation of divider output ($\overline{\text{DVO}}$) 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 internal serial clock of serial interface
⑦ Generation of warm-up clocks for releasing STOP mode

(1) Configuration of Timing Generator

The timing generator consists of a 21-stage divider with a divided-by-2 prescaler. During reset and at releasing STOP mode, the divider is cleared to "0", however; the prescaler is not cleared.

Note: Even if the main system clock is changed by the clock gear, the output from the divider is not changed. The peripheral circuit using high-speed divider output (1st output) can not be used when the main system clock slows down.

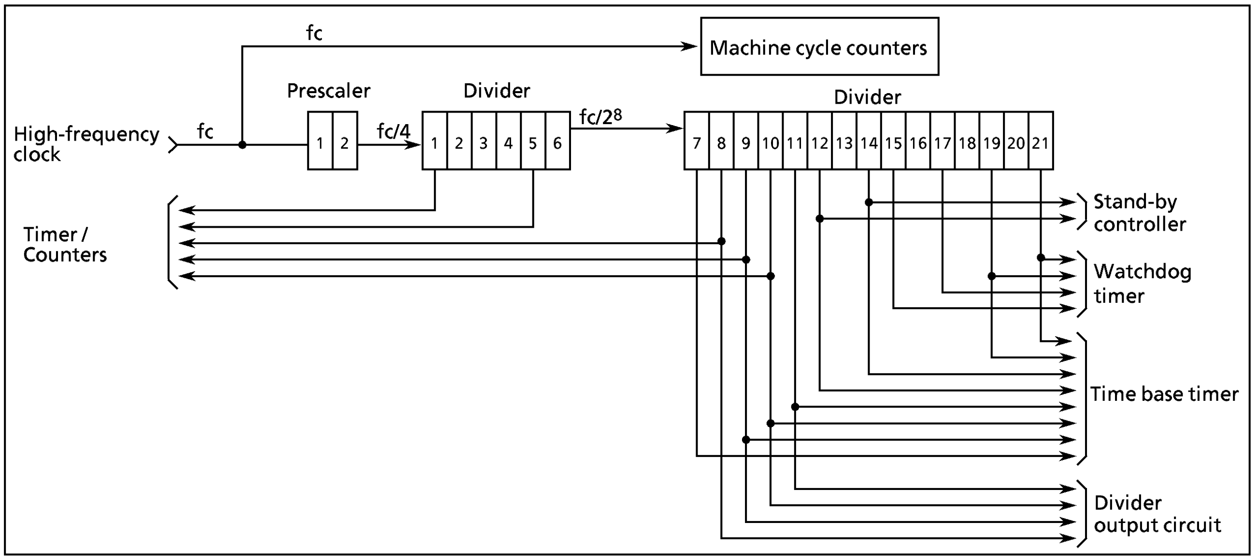


Figure 1-11. Configuration of timing generator

(2) Machine Cycle

Instruction execution and built-in hardware operation are synchronized with the 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 Series: ranging from 1-cycle instructions which require one machine cycle for execution to 10-cycle instructions which require 10 machine cycles for execution.
A machine cycle consists of 4 states (S0 to S3), and each state consists of one main system clock.

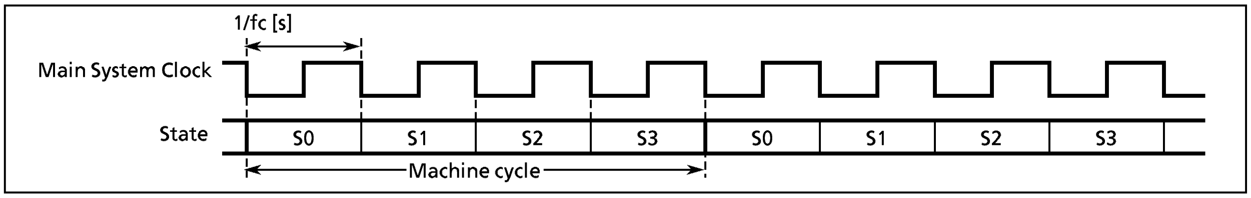


Figure 1-12. Machine cycle

1.8.3 Stand-by controller

The stand-by controller starts and stops the oscillation circuits. These modes are controlled by the system control registers (SYSCR1, SYSCR2). Figure 1-13 shows the operating mode transition diagram and Figure 1-14 shows the system control registers. Either the single-clock or the dual-clock mode can be selected by an option during reset.

(1) **Operation mode**

① NORMAL mode

In this mode, both the CPU core and on-chip peripherals operate. The TMP87C409B/809B are placed in this mode after reset.

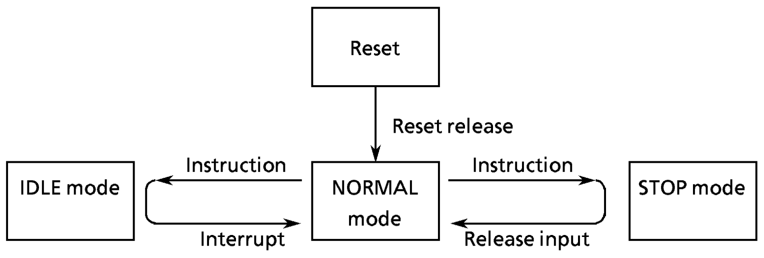
② IDLE mode

In this mode, the CPU and the watchdog timer are halted; however, on-chip peripherals remain active. IDLE mode is started by the system control register 2, and IDLE mode is released to NORMAL mode by an interrupt request from on-chip peripherals or external interrupt inputs. When IMF (interrupt master enable flag) is "1" (interrupt enable), the execution will resume upon acceptance of the interrupt, and the operation will return to normal after the interrupt service is completed. When IMF is "0" (interrupt disable), the execution will resume with the instruction which follows IDLE mode start instruction.

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 the lowest power consumption during this mode. The output status of input output ports can be set to either output hold or high-impedance under software control

STOP mode is started by the system control register 1, and STOP mode is released by STOP input pin (either level-sensitive or edge-sensitive can be selected). After the warming-up period is completed, the execution resumes with the next instruction which follows the STOP mode start instruction.



Single Clock Mode Transition Diagram

Operating mode	Oscillation circuit	CPU core	Peripheral circuit	Machine cycle time
RESET	Turning on oscillation	Reset	Reset	4/fc [s]
NORMAL		Operate	Operate	
IDLE		Halt		
STOP	Turning off oscillation		Halt	—

Figure 1-13. Operating mode transition diagram

System Control Register 1

SYSCR1 (0038_H)

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

(Initial value: 0000 00**)

STOP	STOP mode start	0: CPU core and peripherals remain active 1: CPU core and peripherals are halted	R/W
RELM	Release method for STOP mode	0: $\overline{\text{STOP}}$ pin input rising edge release 1: $\overline{\text{STOP}}$ pin input "H" level release	
OUTEN	Port output control during STOP mode	0: High-impedance 1: Remain unchanged	
WUT	Warming-up time at releasing STOP mode	00: 3×2^{19}/fc 01: 2^{19}/fc 10: 3×2^{12}/fc 11: 2^{12}/fc	

Note 1: Always set bit 5 to "0".
Note 2: Bits 1, 0 in SYSCR1 is read in as undefined value when a read instruction is executed.
Note 3: fc ; High-frequency clock [Hz]
** ; Don't care*
Note 4: When the STOP mode is started by specifying OUTEN = "0", the in internal input of port is fixed to "0" and the interrupt of the falling edge may be set.

System Control Register 2

SYSCR2 (0039_H)

7	6	5	4	3	2	1	0
XEN	0	0	IDLE				

(Initial value: 1000 ****)

XEN	High-frequency oscillator control	0: Turn off oscillation 1: Turn on oscillation	R/W
IDLE	IDLE mode start	0: CPU, WDT operate 1: CPU, WDT halt (IDLE mode start)	

Note 1: An internal reset is applied if XEN is cleared to "0".
Note 2: Always set bit 6, 5 to "0".
*Note 3: WDT ; Watchdog timer, * ; Don't care*
Note 4: Bits 3 to 0 in SYSCR2 are read in as "1" when a read instruction is executed.

Figure 1-14. System control registers 1, 2

1.8.4 Operating mode control

(1) STOP mode (STOP)

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 P43 and an $\overline{\text{INT5}}$ (external interrupt input 5) pin. The STOP mode is started by setting STOP (bit 7 in SYSCR1) to "1". During STOP mode, the following status is maintained.

① High-frequency oscillations are turned off, and all internal operations are halted.

② The data memory (except for DBR), registers, PSW, and port output latches are all held in the status in effect before STOP mode was entered. The port output can select either output hold or high-impedance by setting OUTEN (bit 4 in SYSCR1).

③ The divider of the timing generator is cleared to "0".

④ The program counter holds the address of the instruction after the following instruction which started the STOP mode. [for example, SET (SYSCR1)]

STOP mode includes a level sensitive release mode and an edge-sensitive release mode, either of which can be selected with RELM (bit 6 in SYSCR1).

Note: In STOP Mode changes in external interrupt signals may cause interrupt latches to be set to 1 and interrupt routines to be started immediately after exiting STOP Mode. Therefore be sure to enter STOP Mode only after disabling interrupts. Also, when enabling interrupts after exiting STOP Mode, be sure to first clear interrupt latches for interrupts not to be used.

a. Level-sensitive release mode (RELM = 1)
In this mode, $\overline{\text{STOP}}$ mode is released by setting the STOP pin high. This mode is used for capacitor back-up when the main power supply is cut off and long term battery back-up.
When the $\overline{\text{STOP}}$ pin input is high, executing an instruction which starts the STOP mode will not place in STOP mode but instead will immediately start the release sequence (WARM-up). Thus, to confirm that the $\overline{\text{STOP}}$ pin input is low. The following method can be used for confirmation:
Using an external interrupt input INT5 ($\overline{\text{INT5}}$ is a falling edge-sensitive input).

Example: Starting STOP mode with an INT5 interrupt.

```
PINT5:  TEST  (P4) . 3                 ;  To reject noise, the STOP mode does not start if
        JRS   F, SINT5                    port P43 is at high.
        LD    (SYSCR1), 01000000B      ;  Sets up the level-sensitive release mode.
        SET   (SYSCR1) . 7             ;  Starts STOP mode
        LDW   (IL), 1000011101010111B  ;  IL11, 5, 3 ← 0 (clears interrupt latches)
SINT5:  RETI
```

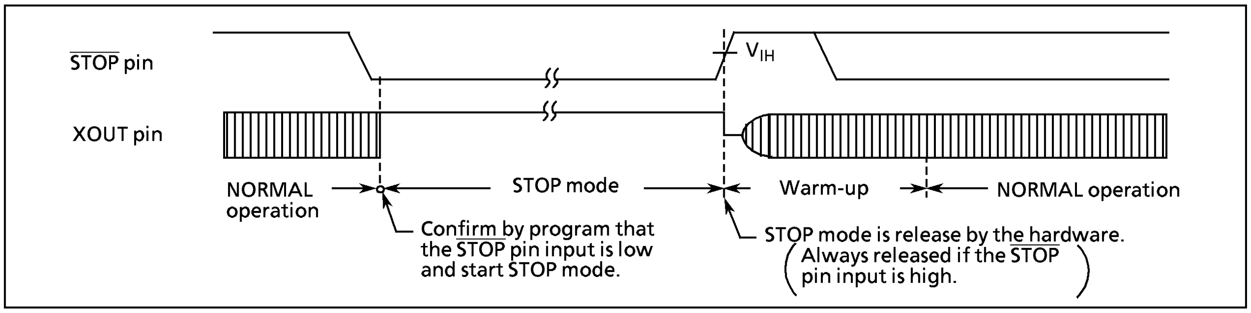

$\overline{\text{STOP}}$ pin
XOUT pin
V_{IH}
NORMAL operation
STOP mode
Warm-up
NORMAL operation
Confirm by program that the $\overline{\text{STOP}}$ pin input is low and start STOP mode.
STOP mode is release by the hardware.
(Always released if the $\overline{\text{STOP}}$ pin input is high.)

Figure 1-15. Level-sensitive release mode

Note 1: After warm-up start, even if $\overline{\text{STOP}}$ pin input is low again, STOP mode does not restart.
Note 2: When changing to the level-sensitive release mode from the edge-sensitive release mode, the release mode is not switched until a rising edge of the $\overline{\text{STOP}}$ pin input is detected.

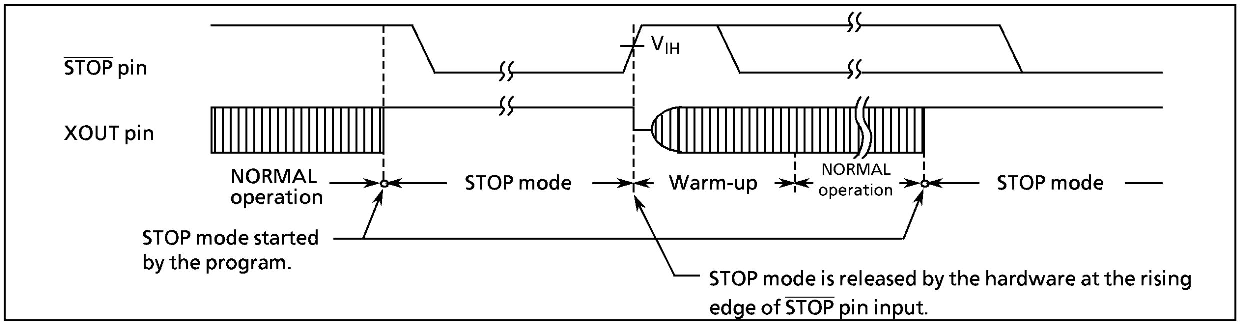
b. Edge-sensitive release mode (RELM = "0")

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 repeatedly executed 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.

Example: Starting STOP mode operation in the edge-sensitive release mode

```
        LD    (SYSCR1), 00000000B       ;  OUTEN ← 0 (specifies high-impedance)
        DI                              ;  IMF ← 0
        SET   (SYSCR1). STOP            ;  STOP ← 1 (activates STOP mode)
        LDW   (IL), 1000011101010111B   ;  IL11, 5, 3 ← 0 (clears interrupt latches)
        EI                              ;  IMF ← 1
```



STOP pin
XOUT pin
V_IH
NORMAL operation
STOP mode
Warm-up
NORMAL operation
STOP mode
STOP mode started by the program.
STOP mode is released by the hardware at the rising edge of STOP pin input.

Figure 1-16. Edge-sensitive release Mode

STOP mode is released by the following sequence:

① The oscillator is turned on.

② A warming-up period is inserted to allow oscillation time to stabilize. During warm-up, all internal operations remain halted. Four different warming-up times can be selected with WUT (bits 3 and 2 in SYSCR1) as determined by the resonator characteristics.

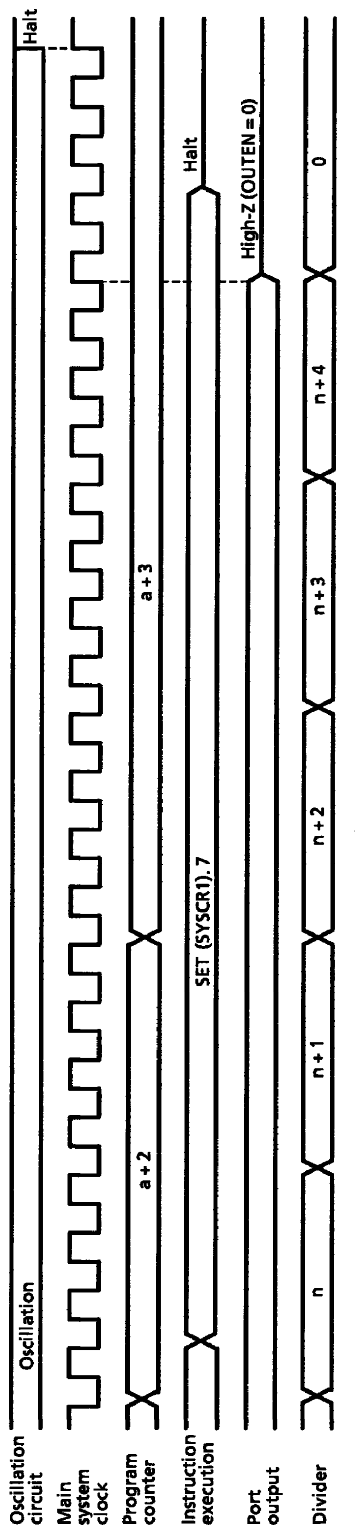
③ When the warming-up time has elapsed, normal operation resumes with the instruction following the STOP mode start instruction. The start is made after the divider of the timing generator is cleared to "0".

Table 1-1. Warming-up time example

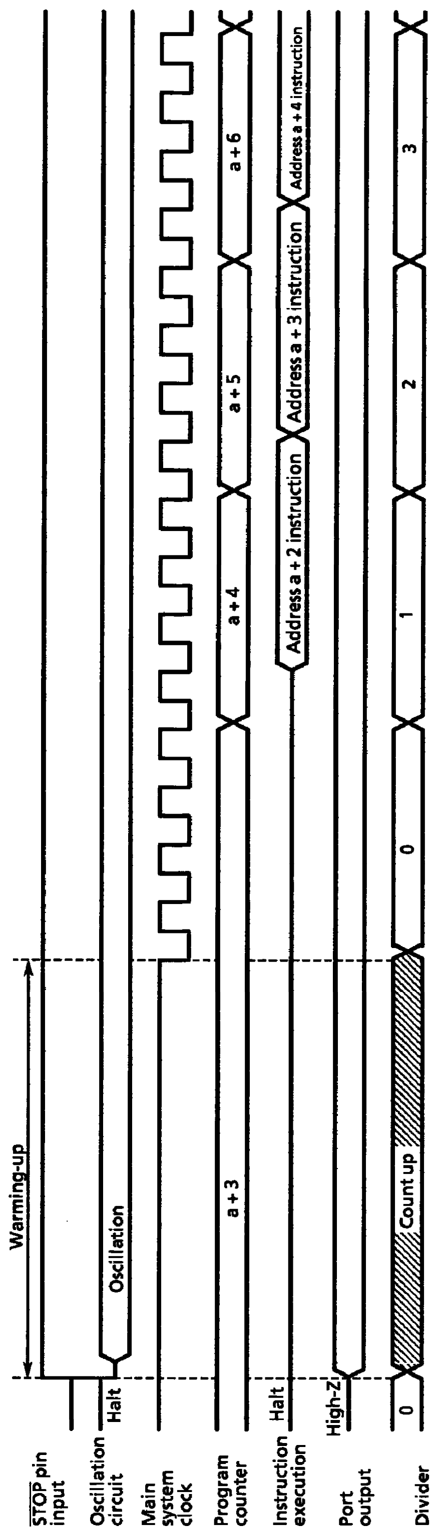
WUT	Warming-up Time at fc = 4.194304 MHz	Warming-up Time at fc = 8 MHz
00	375 ms	196.6 ms
01	125 ms	65.5 ms
10	2.93 ms	1.54 ms
11	976.6 μs	512 μs

Note: The warming-up time is obtained by dividing the basic clock by the divider: therefore, the warming-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 warming-up time must be considered an approximate value.

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



(a) STOP Mode Start (Example : starting with the SET (SYSCR1). 7 instruction located at address 0a)



(b) STOP Mode Release

Figure 1-17. STOP mode start/release

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 high, 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 rate 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).

(2) IDLE mode (IDEL1)

IDLE mode is controlled by the system control register 2 (SYSCR2) and maskable interrupts. The following status is maintained during IDLE mode.

① Operation of the CPU and watchdog timer is halted. On-chip peripherals continue to operate.
② The data memory, CPU registers, PSW, and port output latches are all held in the status in effect before IDLE mode was entered.
③ The program counter holds the address of the instruction after the following instruction which started IDLE mode.

Example: Starting IDLE mode.
SET (SYSCR2) . 4

IDLE mode includes a normal release mode and an interrupt release mode. Selection is made with the interrupt master enable flag (IMF). Releasing the IDLE mode returns to NORMAL.

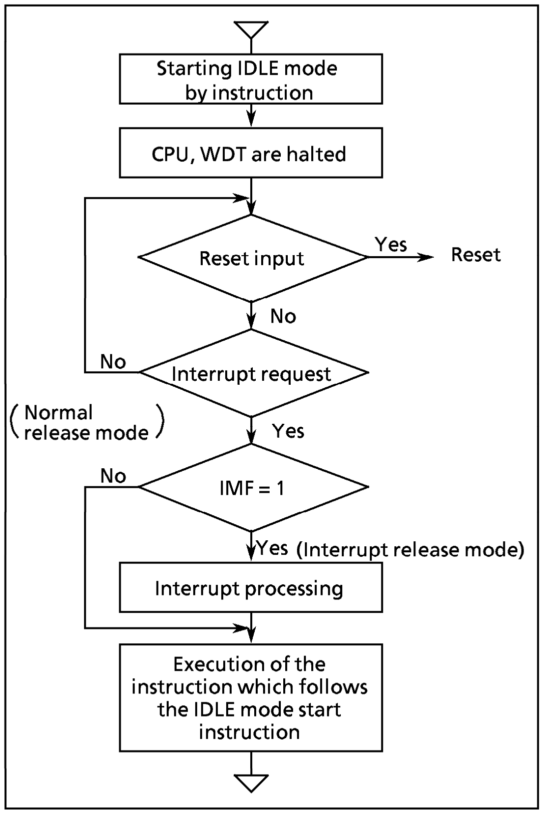
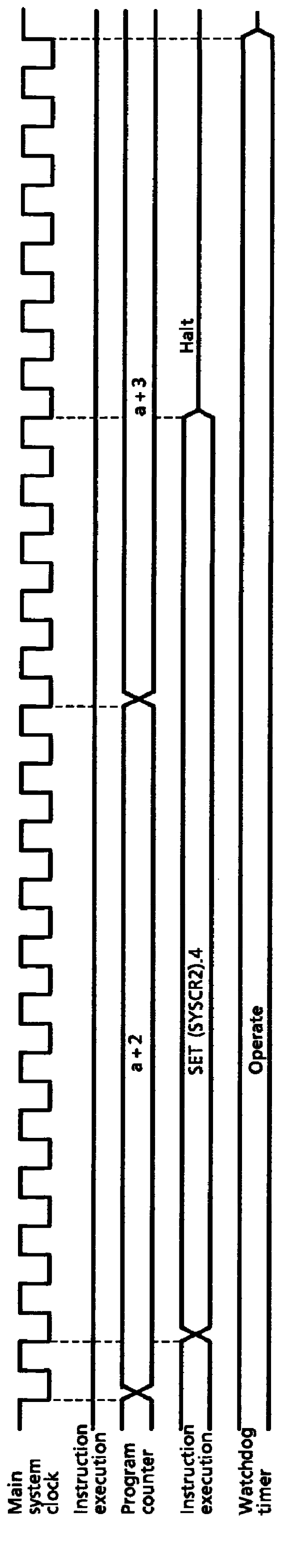

Starting IDLE mode by instruction
CPU, WDT are halted
Reset input
Yes
Reset
No
Interrupt request
No
(Normal release mode)
Yes
IMF = 1
No
Yes (Interrupt release mode)
Interrupt processing
Execution of the instruction which follows the IDLE mode start instruction

Figure 1-18. IDLE mode

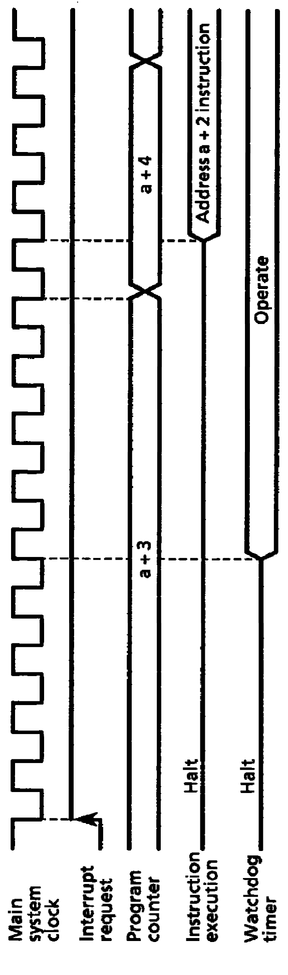
a. Normal release mode (IMF = "0")
IDLE mode is released by any interrupt source enabled by the individual interrupt enable flag (EF) or an external interrupt 0 ($\overline{\text{INT0}}$) request. Execution resumes with the instruction following the IDLE mode start instruction.
The interrupt latches (IL) of the interrupt source used for release is required to be cleared to "0" by load instruction.

b. Interrupt release mode (IMF = "1")
IDLE mode is released and interrupt processing is started by any interrupt source enabled with the individual interrupt enable flag (EF) or an external interrupt 0 ($\overline{\text{INT0}}$) request. After the interrupt is processed, the execution resumes from the instruction following the instruction which started IDLE mode.
IDLE mode can also be released by setting the $\overline{\text{RESET}}$ pin low, which immediately performs the reset operation.

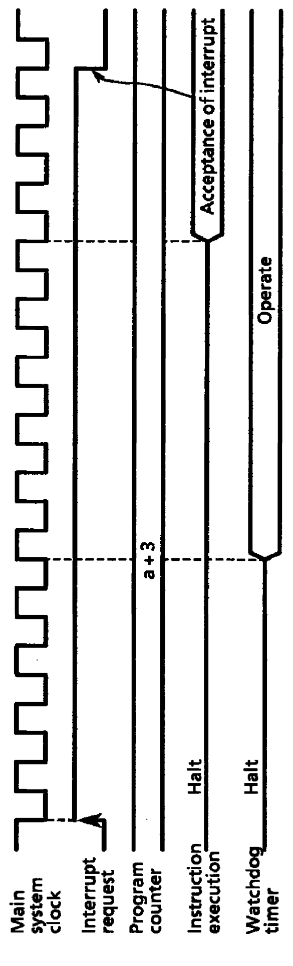
Note: When a watchdog timer interrupt is generated immediately before the IDLE mode is started, the watchdog timer interrupt will be processed by IDLE mode will not be started.



(a) IDLE Modes Start (Example : starting with the SET instruction located at address a)



① Normal Release Mode



② Interrupt Release Mode

(b) IDLE Mode Release

Figure 1-19. IDLE mode start/release

1.9 Interrupt Controller

The TMP87C409B/809B each have a total of 11 interrupt sources: 4 externals and 7 internals. Nested interrupt control with priorities is also possible. Two of the internal sources are pseudo non-maskable interrupts; the remainder are all maskable interrupts.

Interrupt latches (IL) that hold the interrupt requests are provided for interrupt sources. Each interrupt vector is independent. The interrupt latch is set to "1" when an interrupt request is generated and requests the CPU to accept the interrupt. The acceptance of maskable interrupts can be selectively enabled and disabled by the program using the interrupt master enable flag (IMF) and the individual interrupt enable flags (EF). When two or more interrupts are generated simultaneously, the interrupt is accepted in the highest priority order as determined by the hardware. Figure 1-20 shows the interrupt controller.

Table 1-2. Interrupt sources

Interrupt Source			Enable Condition	Interrupt Latch	Vector Table Address	Priority
Internal / External	(Reset)		Non-Maskable	—	$FFFE_H$	High 0
Internal	INTSW	(Software interrupt)	Pseudo non-maskable	—	$FFFC_H$	1
Internal	INTWDT	(Watchdog timer interrupt)		IL_2	$FFFA_H$	2
External	INT0	(External interrupt 0)	IMF = 1, INT0EN = 1	IL_3	$FFF8_H$	3
Internal	INTTC1	(16-bit timer / counter 1 interrupt)	$IMF \cdot EF_4 = 1$	IL_4	$FFF6_H$	4
External	INT1	(External interrupt 1)	$IMF \cdot EF_5 = 1$	IL_5	$FFF4_H$	5
Internal	INTTBT	(Time base timer interrupt)	$IMF \cdot EF_6 = 1$	IL_6	$FFF2_H$	6
Reserved			$IMF \cdot EF_7 = 1$	IL_7	$FFF0_H$	7
Internal	INTTC3	(8-bit timer / counter 3 interrupt)	$IMF \cdot EF_8 = 1$	IL_8	$FFEE_H$	8
Internal	INTSBI	(Serial bus interface interrupt)	$IMF \cdot EF_9 = 1$	IL_9	$FFEC_H$	9
Internal	INTTC4	(8-bit timer / counter 4 interrupt)	$IMF \cdot EF_{10} = 1$	IL_{10}	$FFEA_H$	10
External	INT3	(External interrupt 3)	$IMF \cdot EF_{11} = 1$	IL_{11}	$FFE8_H$	11
Reserved			$IMF \cdot EF_{12} = 1$	IL_{12}	$FFE6_H$	12
Reserved			$IMF \cdot EF_{13} = 1$	IL_{13}	$FFE4_H$	13
Reserved			$IMF \cdot EF_{14} = 1$	IL_{14}	$FFE2_H$	14
External	INT5	(External interrupt 5)	$IMF \cdot EF_{15} = 1$	IL_{15}	$FFE0_H$	Low 15

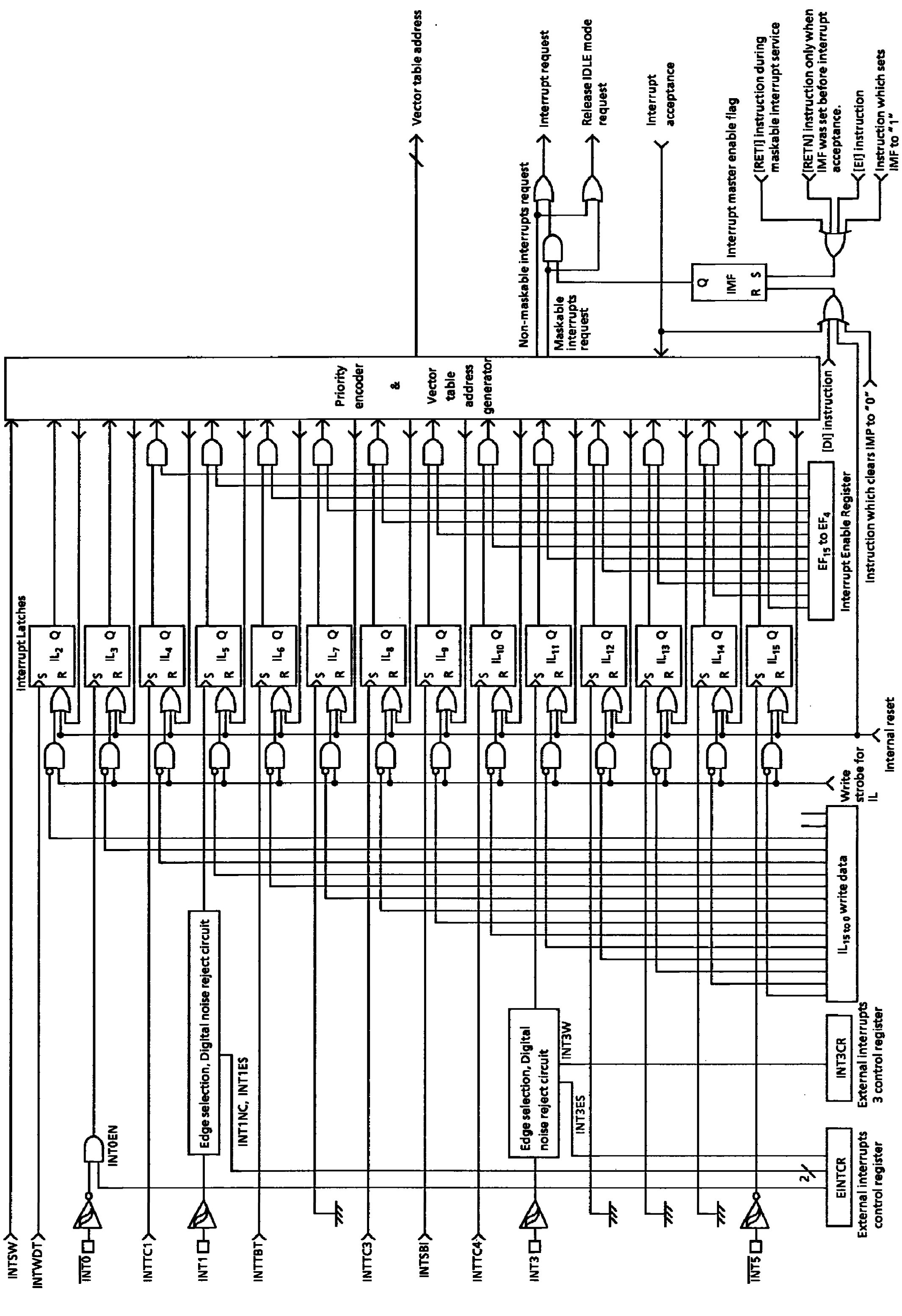

Vector table address
Interrupt request
Release IDLE mode request
Interrupt acceptance
Interrupt master enable flag
[RETI] instruction during maskable interrupt service
[RETN] instruction only when IMF was set before interrupt acceptance.
[EI] instruction
Instruction which sets IMF to "1"
Non-maskable interrupts request
Maskable interrupts request
IMF
Priority encoder & Vector table address generator
[DI] instruction
Instruction which clears IMF to "0"
EF15 to EF4
Interrupt Enable Register
Interrupt Latches
IL2
IL3
IL4
IL5
IL6
IL7
IL8
IL9
IL10
IL11
IL12
IL13
IL14
IL15
Internal reset
Write strobe for IL
IL15 to 0 write data
Edge selection, Digital noise reject circuit
INT1NC, INT1ES
Edge selection, Digital noise reject circuit
INT3W
INT3ES
INT0EN
INT3CR
External interrupts 3 control register
EINTCR
External interrupts control register
INTSW
INTWDT
INT0
INTTC1
INT1
INTTBT
INTTC3
INTSBI
INTTC4
INT3
INT5

Figure 1-20. Interrupt controller block diagram

(1) Interrupt Latches (IL_{15} to IL_2)

Interrupt latches are provided for each source, except for a software interrupt. The latch is set to "1" when an interrupt request is generated, and requests the CPU to accept the interrupt. The latch is cleared to "0" just after the interrupt is accepted. All interrupt latches are initialized to "0" during reset.

The interrupt latches are assigned to addresses 003C and 003DH in the SFR. Each latch can be cleared to "0" individually by an instruction; however, the read-modify-write instruction such as bit manipulation or operation instructions cannot be used (Do not clear IL_2 for a watchdog timer interrupt to "0"). Thus, interrupt requests can be canceled and initialized by the program. Note that interrupt latches cannot be directly set to "1" by any instruction. The contents of interrupt latches can be read out by an instruction. Therefore, testing interrupt requests by software is possible.

Example 1: Clears interrupt latches

```
        LDW   (IL), 1000110000111111B     ; IL9, IL8, IL6
```

Example 2: Reads interrupt latches

```
        LD    WA, (IL)                    ; W ← ILH, A ← ILL
```

Example 3: Tests an interrupt latch

```
        TEST  (IL).6                      ; IL6 = 1 then jump
        JR    F, SSET
```

(2) Interrupt Enable Register (EIR)

The interrupt registers (EIR) enable and disable the acceptance of interrupts except for the pseudo non-maskable interrupts (software and watchdog timer interrupts). Pseudo non-maskable interrupts are accepted regardless of the contents of the EIR; however, the pseudo non-maskable interrupts cannot be nested more than once at the same time.

The EIR consists of an interrupt master enable flag (IMF) and individual interrupt enable flags (EF). These registers are assigned to addresses $003A_H$ and $003B_H$ in the SFR, and can be read and written by an instruction (including read-modify-write instructions such as bit manipulation instructions).

① Interrupt Master Enable Flag (IMF)

The interrupt master enable flag (IMF) enables and disables the acceptance of all interrupts. Clearing this flag to "0" disables the acceptance of all maskable interrupts. Setting to "1" enables the acceptance of interrupts.

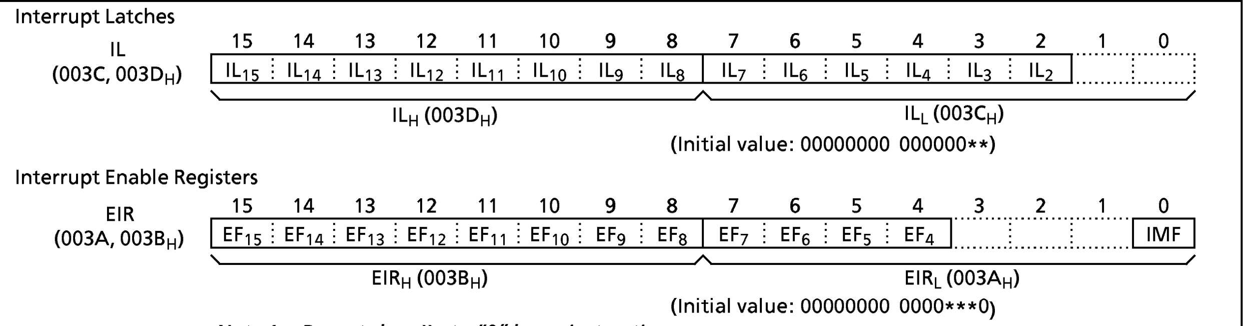
When an interrupt is accepted, this flag is cleared to "0" to temporarily disable the acceptance of maskable interrupts. After execution of the interrupt service program, this flag is set to "1" by the maskable interrupt return instruction [RETI] to again enable the acceptance of interrupts. If an interrupt request has already occurred, interrupt service starts immediately after execution of the [RETI] instruction.

Pseudo non-maskable interrupts are returned by the [RETN] instruction. In this case, the IMF is set to "1" only when pseudo non-maskable interrupt service is started with interrupt acceptance enabled (IMF = 1). Note that IMF remains "0" when cleared in the interrupt service program.

The IMF is assigned to bit 0 at address $003A_H$ in the SFR, and can be read and written by an instruction. IMF is normally set and cleared by the [EI] and [DI] instructions, and the IMF is initialized to "0" during reset.

② Individual Interrupt Enable Flags (EF_{15} to EF_4)
These flags enable and disable the acceptance of individual maskable interrupts, except for an external interrupt 0. Setting the corresponding bit of an individual interrupt enable flag to "1" enables acceptance of an interrupt, setting the bit to "0" disables acceptance.

```
Example 1:  Sets individual interrupt enable flags
                DI                                    ;  IMF ← 0
                LDW        (EIRL),  1110100010100000B ;  EF15 to 13, EF11, EF7, EF5 ← 1
                                                         Note: Do not set IMF
                 ⋮
                EI                                    ;  IMF ← 1
Example 2:  Example of description in C
                unsigned int      __io (3AH)    EIRL  ;  /* 3AH: address for EIRL */
                __DI ( )                              ;
                EIRL = 10100000B                      ;
                 ⋮
                __EI ( )                              ;
```


Interrupt Latches
IL
(003C, 003D$_H$)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
IL_{15} IL_{14} IL_{13} IL_{12} IL_{11} IL_{10} IL_9 IL_8 IL_7 IL_6 IL_5 IL_4 IL_3 IL_2
IL_H (003D$_H$)
IL_L (003C$_H$)
(Initial value: 00000000 000000**)
Interrupt Enable Registers
EIR
(003A, 003B$_H$)
15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
EF_{15} EF_{14} EF_{13} EF_{12} EF_{11} EF_{10} EF_9 EF_8 EF_7 EF_6 EF_5 EF_4 IMF
EIR_H (003B$_H$)
EIR_L (003A$_H$)
(Initial value: 00000000 0000***0)

Note 1: Do not clear IL_2 to "0" by an instruction.
Note 2: Do not clear IL with read-modify-write instructions such as bit operations.
Note 3: Before you set EF, be sure to clear IMF (to disable interrupt).
*Note 4: * ; Don't care*
Note 5: Do not set IMF to 1 simultaneously with EF.
Note 6: Do not set IMF to "1" during non-maskable interrupt service program.

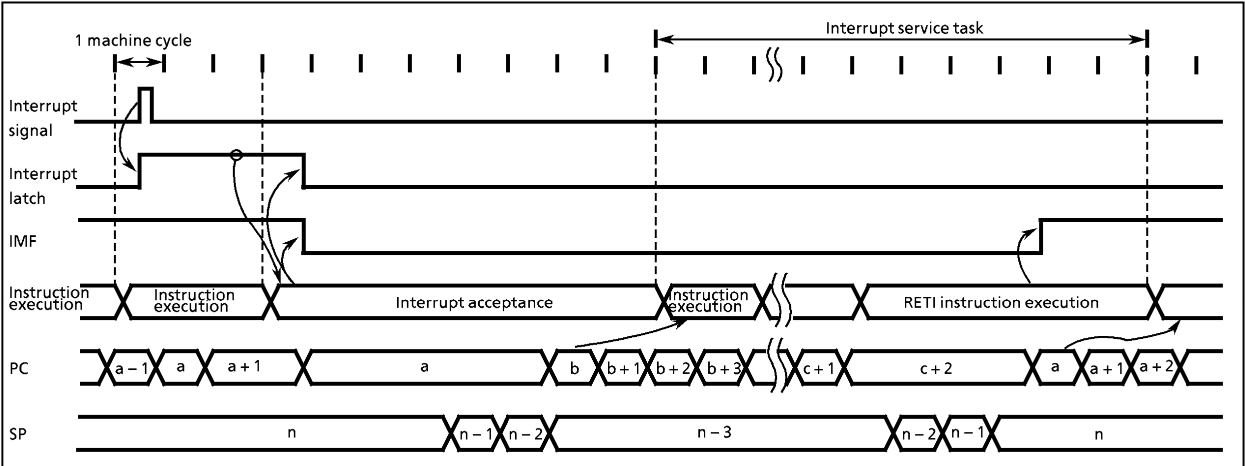
Figure 1-21. Interrupt Latch (IL) and Interrupt Enable Register (EIR)

1.9.1 Interrupt sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to "0" by a reset or an instruction. Interrupt acceptance sequence requires 8 machine cycles after the completion of the current instruction execution. The interrupt service task terminates upon execution of an interrupt return instruction [RETI] (for maskable interrupts or [RETN](pseudo non-maskable interrupts). Figure 1-22 shows the timing chart of interrupt acceptance and interrupt return instruction.

(1) Interrupt acceptance processing

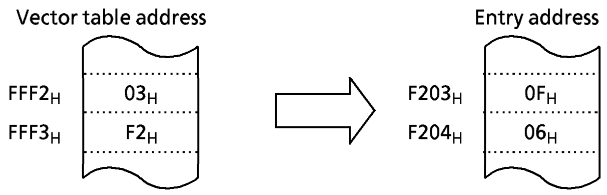
① The interrupt master enable flag (IMF) is cleared to "0" to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.
② The interrupt latch (IL) for the interrupt source accepted is cleared o "0".
③ The contents of the program counter and the program status word are saved (pushed) onto the stack. (pushed down in order of PSW, PC_H, PC_L). The contents of Stack Pointer is decreased by 3.
④ The entry address of the interrupt service program is read from the vector table address corresponding to the interrupt source, and the entry address is loaded to the program counter.
⑤ The instruction stored at the entry address of the interrupt service program is executed.



Note 1: a ; return address b ; entry address c ; address when the RETI instruction is stored
Note 2: The maximum response time from when an IL is set until an interrupt acceptance processing starts is 38/fc [s]. It equals to setting the IL on the first machine cycle in 10 cycles instruction execution.

Figure 1-22. Timing chart of interrupt acceptance and interrupt return instruction

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



A maskable interrupt is not accepted until the IMF is set to "1" even if a maskable interrupt of higher priority than that of the current interrupt interrupt being serviced. When nested interrupt service is necessary, 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. However, an acceptance of external interrupt 0 cannot be disabled by the EF, therefore, if disablement is necessary, either the external interrupt function must be disabled with the external interrupt control register (INT0EN) or interrupt processing must be avoided by the program. (When INT0EN = 0, the interrupt latch IL3 is not set, therefore, the falling edge of the INT0 pin input cannot be detected.)
When INT0EN = 0, the interrupt latch IL3 is not set, therefore, the falling edge of the $\overline{\text{INT0}}$ pin input cannot be detected.

Example 1: Disables an external interrupt 0 using INT0EN:

```
        CLR    (EINTCR). INT0EN      ;  INT0EN ← 0
```

Example 2: Disables the processing of external interrupt 0 under the software control (using bit 0 at address 00F0H as the interrupt processing disable switch):

```
PINT0:  TEST   (00F0H) . 0           ;  Returns without interrupt processing if (00F0H)0 = 1.
        JRS    T, SINT0
        RETI
SINT0:  Interrupt processing
        RETI
         :
VINT0:  DW     PINT0
```

(2) General-purpose register save / restore

During interrupt acceptance processing, the program counter and the program status word are automatically saved on the stack, but not the accumulator and other registers. These registers are saved by the program if necessary. Also, when nesting multiple interrupt services, it is necessary to avoid using the same data memory area for saving registers.
The following method is used to save / restore the general-purpose registers:

① General-purpose register save / restore by register bank changeover:
General-purpose registers can be saved at high-speed by switching to a register bank that is not in use. Normally, bank 0 is used for the main task and banks 1 to 15 are assigned to interrupt service tasks. To increase the efficiency of data memory utilization, the same bank is assigned for interrupt sources which are not nested.
The switched bank is automatically restored by executing an interrupt return instruction [RETI] or [RETN]. Therefore, it is not necessary for a program to save the RBS.

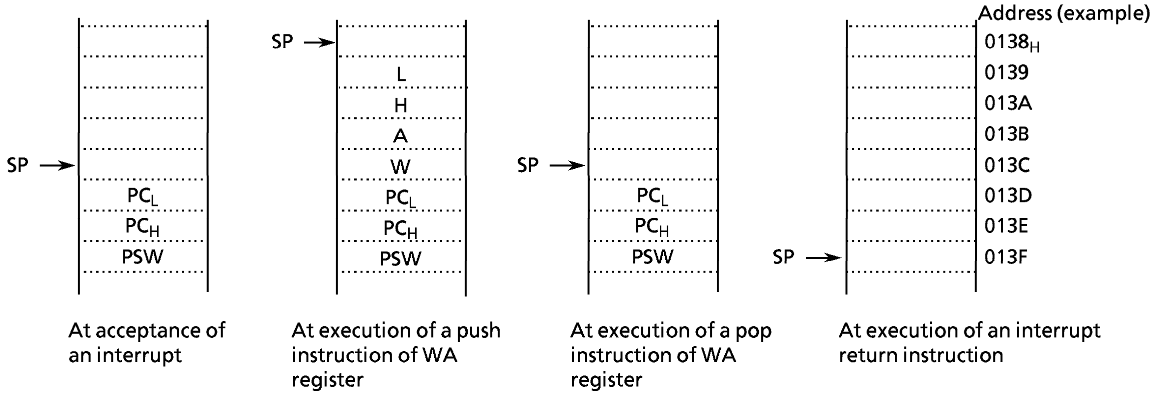
Example: Register Bank Changeover

```
PINTxx:   LD     RBS, n        ;  Switches to bank n (1 μs at 8 MHz)
           Interrupt processing
          RETI                 ;  Restores bank and Returns
```

② General-purpose register save / restore using push and pop instructions:
To save only a specific register, and when the same interrupt source occurs more than once, the general-purpose registers can be saved/restored using push/pop instructions.

Example: Register save using push and pop instructions

```
PINTxx: PUSH    WA          ; Save WA register pair
        PUSH    HL          ; Save HL register pair
        Interrupt processing
        POP     HL          ; Restore HL register pair
        POP     WA          ; Restore WA register pair
        RETI                ; Return
```


Address (example)
0138H
0139
013A
013B
013C
013D
013E
013F
SP
L
H
A
W
PCL
PCH
PSW
At acceptance of an interrupt
At execution of a push instruction of WA register
At execution of a pop instruction of WA register
At execution of an interrupt return instruction

③ General-purpose registers save/restore using data transfer instruction:
Data transfer instructions can be used to save only a specific general-purpose register during processing of a single interrupt.

Example: Saving / Restoring registers by data memory transfer instructions

```
PINTxx: LD      (GSAVA), A  ; Save A register
        Interrupt processing
        LD      A, (GSAVA)  ; Restore A register
        RETI                ; Return
```

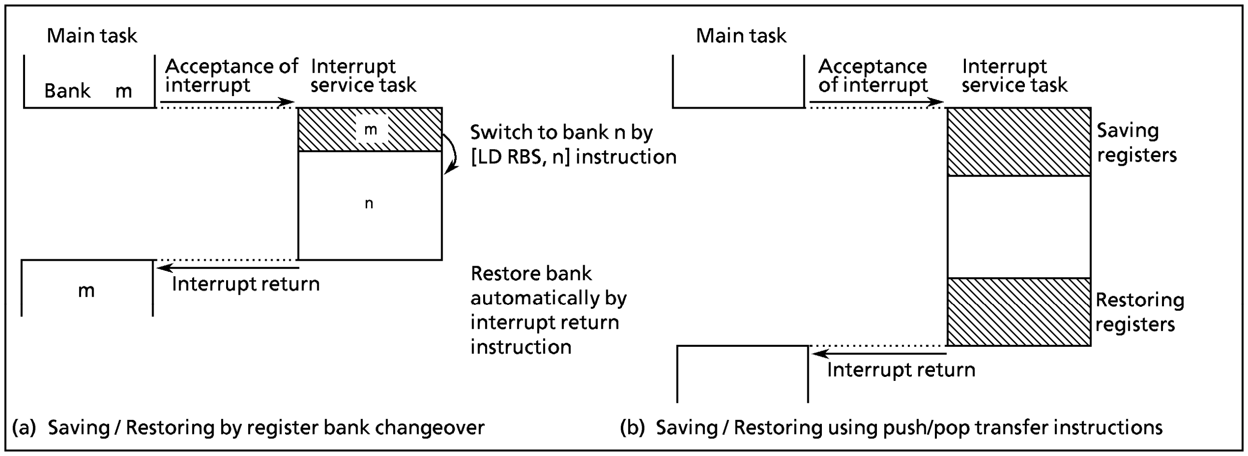

Main task
Bank m
Acceptance of interrupt
Interrupt service task
m
n
Switch to bank n by [LD RBS, n] instruction
m
Interrupt return
Restore bank automatically by interrupt return instruction
(a) Saving / Restoring by register bank changeover
Main task
Acceptance of interrupt
Interrupt service task
Saving registers
Restoring registers
Interrupt return
(b) Saving / Restoring using push/pop transfer instructions

Figure 1-23. Saving / Restoring general-purpose registers

(3) Interrupt return

The interrupt return instructions perform the following operations.

[RETI] Maskable interrupt return	[RETN] Non-maskable interrupt return
① The contents of the program counter and the program status word are restored from the stack. ② The stack pointer is incremented 3 times. ③ The interrupt master enable flag is set to "1".	① The contents of the program counter and program status word are restored from the stack. ② The stack pointer is incremented 3 times. ③ The interrupt master enable flag is set to "1" only when a non-maskable interrupt is accepted in interrupt enable status. However, the interrupt master enable flag remains at "0" when so clear by an interrupt service program.

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

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

1.9.2 Software interrupt (INTSW)

Executing the [SWI] instruction generates a software interrupt and immediately starts interrupt processing (INTSW is highest prioritized interrupt). However, if processing of a non-maskable interrupt is already underway, executing the SWI instruction will not generate a software interrupt but will result in the same operation as the [NOP] instruction.

Note: Software interrupt generates during non-maskable interrupt processing to use SWI instruction for software break in a development tool.

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

① Address Error Detection

FF_H is read if for some cause such as noise the CPU attempts to fetch an instruction from a non-existent memory address. Code FF_H is the SWI instruction, so a software interrupt is generated and an address error is detected. The address error detection range can be further expanded by writing FF_H to unused areas of the program memory. Address-trap-reset is generated for instruction fetch from RAM area (addresses 0040 to $013F_H$) or SFR area (0000 to $003F_H$).

Note: The fetch data from addresses, BF80 to $BFFF_H$ for TMP87C409B/809B and TMP87P809 is not "FF_H", because the outgoing test ROM is contained.

② Debugging

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

1.9.3 External interrupts

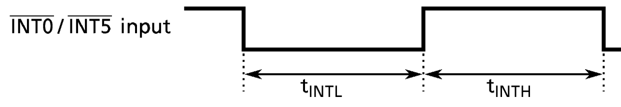
The TMP87C409B/809B each have four external interrupt inputs. Two of these are equipped with digital noise rejection circuits(pulse inputs of less than a certain time are eliminated as noise).
Edge selection is also possible with INT1, INT3 pin. The $\overline{\text{INT0}}$ / P10 pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.
Edge selection, noise rejection control and $\overline{\text{INT0}}$ / P10 pin function selection are performed by the external interrupt control register (EINTCR). Both edge detection of INT3 is controlled by the external interrupt 3 control register (INT3CR).

Table 1-3. External interrupts

Source	Pin	Secondary function pin	Enable conditions	Edge			Digital noise rejection circuit
				rising	falling	both	
INT0	$\overline{\text{INT0}}$	P10	IMF = 1 INT0EN = 1	–	○	–	– (hysteresis input)
INT1	INT1	P11	IMF · EF_5 = 1	INT1ES = 0	INT1ES = 1	–	Pulses of less than 15/fc or 63/fc[s] are eliminated as noise. Pulses equal to or more than 48/fc [s] or 192/fc [s] are regarded as signals.
INT3	INT3	P50 (TC3 / CLZ0)	IMF · EF_{11} = 1 INT3W = 0	INT3ES = 0	INT3ES = 1	–	For falling or rising edge, pulses less than 7/fc [s] are cancelled as noise. Pulses equal to or more then 24/fc [s] are regarded as signals.
			IMF · EF_{11} = 1 INT3W = 1	–	–	INT3W = 1	
INT5	$\overline{\text{INT5}}$	P43 / $\overline{\text{STOP}}$	IMF · EF_{15} = 1	–	○	–	– (hysteresis input)

Note 1: The noise rejection function is also affected to detect the edge of timer / counter input (TC1, TC3 pin).

Note 2: The pulse width (both "H" and "L" level) for input to the $\overline{\text{INT0}}$ and $\overline{\text{INT5}}$ pins must be over 1 machine cycle.



t_{INTL}, t_{INTH} > tcyc
Note: tcyc = 4/fc [s] at NORMAL, IDLE mode

Note 3: If a noiseless signal is input to the external interrupt pin in the NORMAL or IDLE mode, the maximum time from the edge of input signal until the IL is set is as follows:
① INT1 pin 49/fc [s] (at INT1NC = 1), 193/fc [s] (at INT1NC = 0)
② INT3 pin 25/fc [s]

Note 4: When INT0EN = 0, the interrupt latch IL3 is not set even if the falling edge of $\overline{\text{INT0}}$ pin input is detected.

Note 5: When high-impedance is specified for port output in STOP mode, port input is forcibly fixed to low level internally. Thus, interrupt latches of external interrupt inputs except P43 ($\overline{\text{STOP}}$ / $\overline{\text{INT5}}$) which are also used as ports may be set to "1". To specify high-impedance for port output in STOP mode, first disable inteerupt service (IMF = 0), activate stop mode. After releasing stop mode, clear interrupt latches using load instruction, then, enable interrupt service.

Example: Activating stop mode

```
LD     (SYSCR1) , 01000000B          ;  OUTEN ← 0 (specifies high-impedance)
DI                                   ;  IMF ← 0
SET    (SYSCR1) , STOP               ;  STOP ← 1 (activates STOP mode)
LDW    (IL) , 1000011101010111B      ;  IL11, 5, 3 ← 0 (clears interrupt latches)
EI                                   ;  IMF ← 1
```

Note 6: When INT3W = 1, the edge of an interrupt can be detected by reading INT3R and INT3F (bits 6 and 5 in INT3CR).

EINTCR (0037_H)

7	6	5	4	3	2	1	0
INT1 NC	INT0 EN			INT3 ES	TC1 ES	INT1 ES	

(Initial value: 00∗∗ 000∗)

INT1NC	INT1 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.)	
INT3 ES TC1ES INT1 ES	INT3, TC1, INT1 edge select	0: Rising edge 1: Falling edge	

Note: fc ; High-frequency clock [Hz] ∗ ; Don't care

INT3CR (0025_H)

7	6	5	4	3	2	1	0
INT3W	INT3R	INT3F					

(Initial value: 000∗ ∗∗∗∗)

INT3W	INT3 both edge selection	0: Refer to INT3ES 1: Both edge detection	Write-Only
INT3R	INT3 rising edge detection flag	0: Rising edge non-detected 1: Rising edge detected	Read-only
INT3F	INT3 falling edge detection flag	0: Falling edge non-detected 1: Falling edge detected	

Note 1: INT3R and INT3F are effective only in INT3W = 1.
Note 2: INT3R and INT3F are cleared to "0" when reading the INT3CR.

Figure 1-24. External interrupt control register

1.10 Watchdog Timer (WDT)

The watchdog timer rapidly detects the CPU malfunction such as endless looping caused by noise or the like, and resumes the CPU to the normal state.
The watchdog timer signal for detecting malfunction can be selected either an internal reset or a non-maskable interrupt request. However, selection is possible only once after reset.
After reset, the signal is initialised to the reset output.
When the watchdog timer is not being used for malfunction detection, it can be used as a timer to generate an interrupt at fixed intervals.

Note: The functions of the watchdog timer may not be fully realized due to disturbance noise, etc. Therefore, careful consideration is required in the designing stage.

1.10.1 Watchdog timer configuration

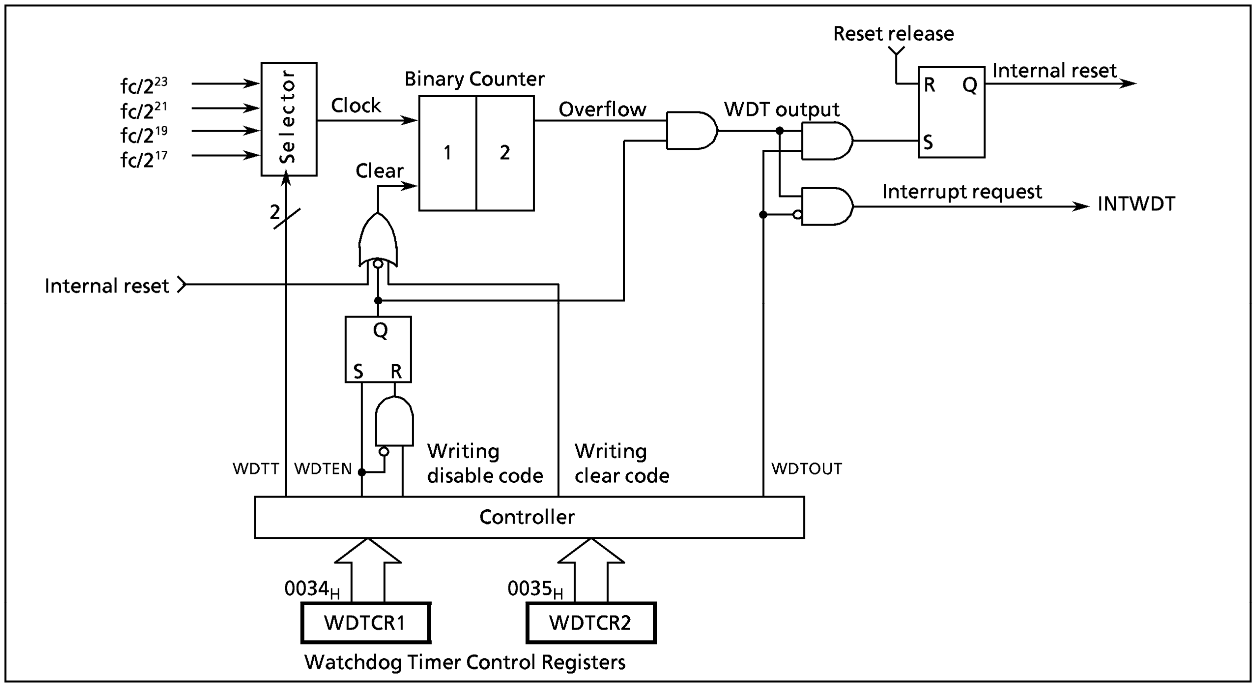


Figure 1-25. Watchdog timer configuration

1.10.2 Watchdog timer control

Figure 1-26 shows the watchdog timer control registers. The watchdog timer is automatically enabled after reset.

(1) Malfunction detection methods using the watchdog timer

The CPU malfunction is detected as follows.

① Setting the detection time, selecting output, and clearing the binary counter.

② Repeatedly clearing the binary counter within the setting detection time.

Note: The binary counter is cleared asynchronously with the source clock. Therefore, the detection time may be reduced to 3/4 of the set time depending on the timing at which the counter is cleared.

If the CPU malfunction occurs for any cause, the watchdog timer output will become active at the rising of an overflow from the binary counters unless the binary counters are cleared. At this time, when WDTOUT = 1, an internal reset is generated. When WDTOUT = 0, a watchdog timer interrupt(INTWDT) is generated.

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

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

```
                        LD     (WDTCR2), 4EH          ;  Clears the binary counters
                        LD     (WDTCR1), 00001101B    ;  WDTT←10, WDTOUT←1
                     ┌ LD     (WDTCR2), 4EH          ;  Clears the binary counters
Within 3/4 of        │  :                                (always clear immediately after changing WDTT)
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 (0034_H)

7	6	5	4	3	2	1	0
				WDT EN	WDTT		WDT OUT

(Initial value: **** 1001)

WDTEN	Watchdog timer enable / disable	0: Disable (It is necessary to write the disable code to WDTCR2.) 1: Enable	Write-only
WDTT	Watchdog timer detection time	00: 2^{25}/fc [s] 01: 2^{23}/fc 10: 2^{21}/fc 11: 2^{19}/fc	
WDTOUT	Watchdog timer output select	0: Interrupt request 1: Internal reset	

Note 1: WDTOUT cannot be set to "1" by program after clearing WDTOUT to "0".
*Note 2: fc ; High-frequency clock[Hz] * ; Don't care*
Note 3: WDTCR1 is a write-only register and must not be used with any of read-modify-write instructions.
Note 4: Disable the watchdog timer or clear the counter just before switching to STOP mode. When the counter is cleared just before switching to STOP mode, clear the counter again subsequently to releasing STOP mode.
Note 5: When WDTOUT = 1, internal reset time is 12/fc [s].

Watchdog Timer Control Register 2

WDTCR2 (0035_H)

7	6	5	4	3	2	1	0

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

WDTCR2	Watchdog timer control code write register	$4E_H$: Watchdog timer binary counter clear (clear code) $B1_H$: Watchdog timer disable(disable code) Others: Invalid	Write-only

Note 1: The disable code is invalid unless written when WDTEN = 0.
*Note 2: * ; Don't care*
Note 3: Since WDTCR2 is a write-only register, read-modify-write instructions (e.g., bit manipulating instructions such as SET or CLR and arithmetic instructions such as AND or OR) cannot be used for read / write to this register.
Note 4: To clear binary counter doesn't initialize the source clock, therefore, it is recommended to clear binary counter within 3/4 of the detection period.

Figure 1-26. Watchdog timer control registers

(2) Watchdog Timer Enable

The watchdog timer is enabled by setting WDTEN (bit 3 in WDTCR1). WDTEN is initialized to "1" during reset, so the watchdog timer operates immediately after reset is released.

(3) Watchdog Timer Disable

The watchdog timer is disabled by writing the disable code ($B1_H$) to WDTCR2 after clearing WDTEN (bit 3 in WDTCR1) to "0". The watchdog timer is not disabled if this procedure is reversed and the disable code is written to WDTCR2 before WDTEN is cleared to "0". During disabling the watchdog timer , the binary counters are cleared to "0".

```
Example:  Disables watchdog timer
                    LDW   (WDTCR1) , 0B101H      ;   WDTEN←0, WDTCR2←disable code
```

Table 1-4. Watchdog timer detection time

Detection time [s]	
WDTT	fc = 8 MHz
00	4.194
01	1.048
10	262.1 m
11	65.5 m

1.10.3 Watchdog timer interrupt (INTWDT)

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

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

```
Example:     Watchdog timer interrupt setting up.
                    LD    SP, 013FH                ;   Sets the stack pointer
                    LD    (WDTCR1) , 00001000B     ;   WDTOUT←0
```

1.10.4 Watchdog timer reset

If the watchdog timer output becomes active, an internal reset is generated, which reset the internal hardware. The internal reset time is 12/fc [s] (1.5 μs at 8 MHz). When an internal reset is generated, the $\overline{\text{RESET}}$ pin remains "H" level.

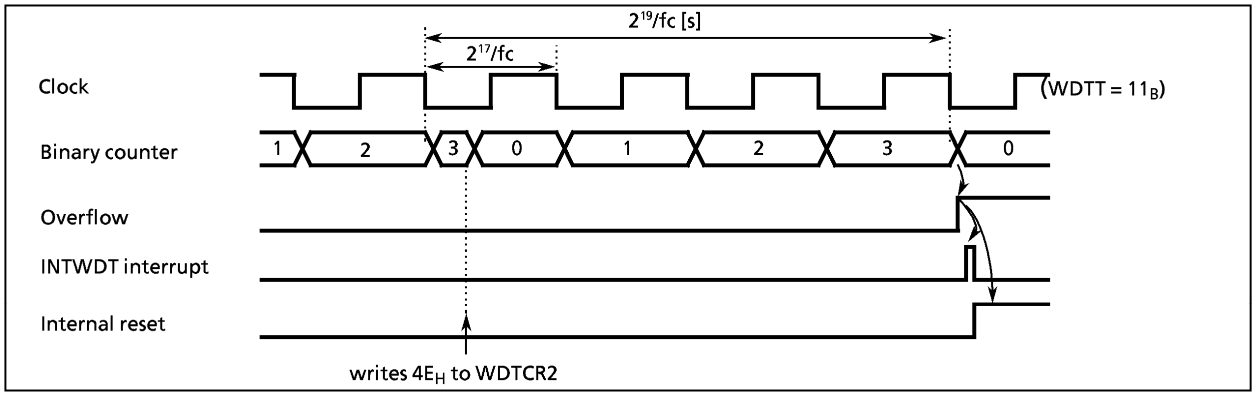


Figure 1-27. Watchdog timer interrupt / reset

Note: Adequate care must be given when designing systems so as to eliminate disturbing noise. Otherwise the Watchdog Timer may not exhibit its full functionality.

1.11 Reset Circuit

TMP87C409B/809B each have four types of reset generation procedures: an external reset input, an address trap reset, a watchdog timer reset and a system clock reset.
Figure 1-28 shows Reset Circuit, Table 1-5 shows on-chip hardware initialization by reset action.

Table 1-5. On-chip hardware initialization by reset action

On-chip Hardware		Initial Value	On-chip Hardware	Initial Value
Program counter	(PC)	$(FFFF_H) \cdot (FFFE_H)$	Prescaler and Divider of Timing generator	0
Register bank selector	(RBS)	0	Watchdog timer	Enable
Jump status flag	(JF)	1		
Interrupt master enable flag	(IMF)	0	Output latches of input/output port	Refer to I/O port circuitry
Interrupt individual enable flags	(EF)	0		
Interrupt latches	(IL)	0	Control register	Refer to control registers

1.11.1 External reset input

The $\overline{\text{RESET}}$ pin contains a hysteresis input 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 "H" level, the reset operation is released and the program execution starts at the vector address stored at addresses FFFE to $FFFF_H$.

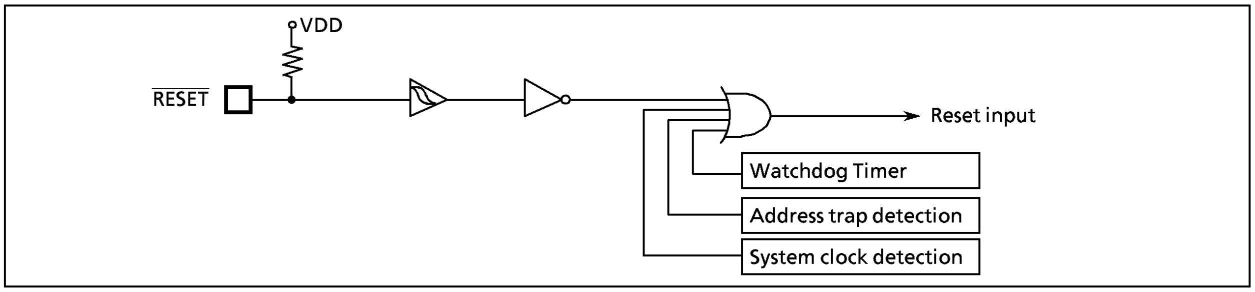
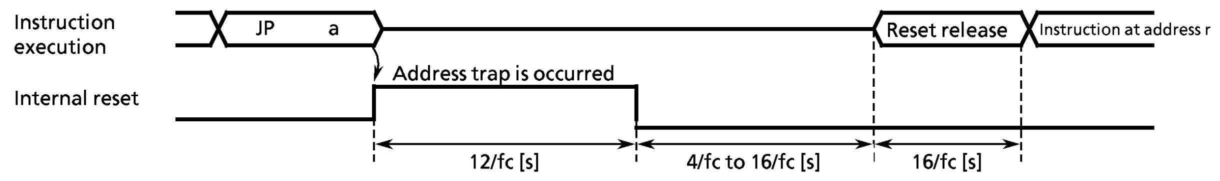


Figure 1-28. Reset circuit

1.11.2 Address-trap-reset

An address-trap-reset is one of fail-safe function that detects CPU malfunction such as endless looping caused by noise or the like. If the CPU attempts to fetch an instruction from RAM or SFR, an internal reset will be generated. The reset time is 12/fc (1.5 μs at 8 MHz).



Note 1: a is an address in on-chip RAM or SFR.
Note 2: During reset release, reset vector "r" is read out, and an instruction at address r is fetched and decoded.

Figure 1-29. Address trap reset

Note: *If an undefined command or a SWI command in the address one before the address trap area is executed, then an address trap reset is applied immediately after the reception of the SWI interrupt is completed.*

1.11.3 Watchdog timer reset

Refer to Section "1.10 Watchdog Timer".

1.11.4 System clock reset

Clearing XEN to "0" stops a system clock, and causes CPU to deadlock. This can be prevented by automatically generating a reset signal whenever XEN = 0 is detected to continue the oscillation. Then internal reset is generated. The reset time is 12/fc [s] (1.5 μs at 8 MHz).

2. On-Chip Peripherals Functions

2.1 Special Function Register (SFR)

The TLCS-870 Series uses the memory mapped I/O system, and all peripheral control and data transfers are performed through the special function registers (SFR).
The SFR are mapped to addresses 0000 to 003F$_H$. Figure 2-1 shows the TMP87C409B/809B SFR.

Address	Read	Write	Address	Read	Write
0000$_H$	reserved		0020$_H$	–	SBICR1 (SBI control 1)
01	P1 port		21	SBIDBR (SBI data buffer)	
02	reserved		22	–	I2AR (I^{2}C bus address)
03	reserved		23	SBISR (SBI status)	SBICR2 (SBI control 2)
04	P4 port		24	reserved	
05	P5 port		25	INT3CR (INT3 control)	
06	P6 port		26	–	P5CR (P5 port I/O control)
07	reserved		27	reserved	
08	reserved		28	–	CLZCR (CLZ control)
09	reserved		29	ADCDRL (AD converter result)	–
0A	reserved		2A	ADCDRH (AD converter result)	–
0B	–	P1CR (P1 port I/O control)	2B	reserved	
0C	–	P6CR (P6 port I/O control)	2C	reserved	
0D	reserved		2D	reserved	
0E	ADCCR (AD converter control)		2E	reserved	
0F	reserved		2F	–	SELREF (VAREF select)
10	–	TREG1L (Timer register 1)	30	reserved	
11	–	TREG1H (Timer register 1)	31	reserved	
12	reserved		32	reserved	
13	reserved		33	–	
14	–	TC1CR (TC1 control)	34	–	WDTCR1 (Watchdog timer
15	reserved		35	–	WDTCR2 control)
16	reserved		36	–	TBTCR (TBT / TG / DVO control)
17	reserved		37	EINTCR (External interrupt control)	
18	TREG3A (Timer register 3A)		38	SYSCR1 (System control)	
19	TREG3B (Timer register 3B)	–	39	SYSCR2 (System control)	
1A	–	TC3CR (TC3 control)	3A	EIR$_L$ (Interrupt enable register)	
1B	–	TREG4 (Timer register 4)	3B	EIR$_H$ (Interrupt enable register)	
1C	–	TC4CR (TC4 control)	3C	IL$_L$ (Interrupt latch)	
1D	reserved		3D	IL$_H$ (Interrupt latch)	
1E	reserved		3E	reserved	
1F	reserved		3F	PSW	RBS (Register bank selector)

(a) Special Function Register

Note 1: Do not access reserved areas by the program.
Note 2: – ; cannot be accessed.
Note 3: When defining address 003F$_H$ with assembler symbols, use GPSW and GRBS.
Note 4: 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.)
Note 5: PSW ; Program Status Word

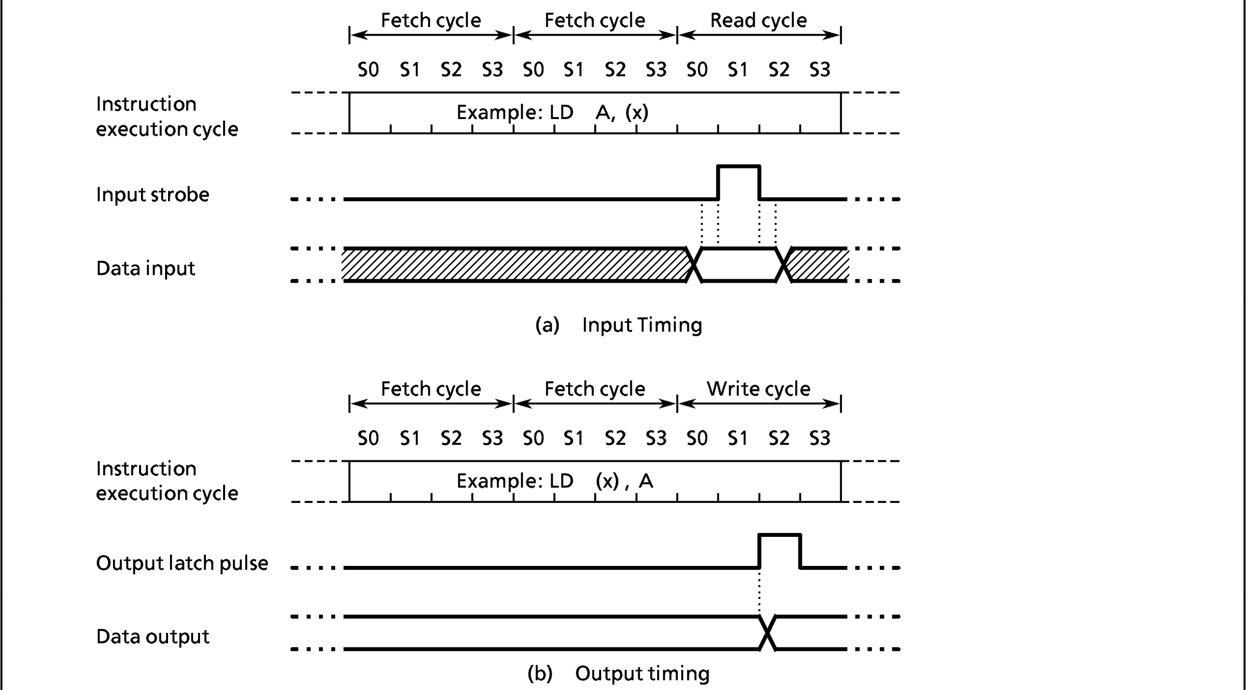
Figure 2-1. SFR

2.2 I/O Ports

The TMP87C409B/809B have 4 ports, 22 pin input/output ports.

①	P1 port	;	8-bit I/O port	(external interrupt input, timer/counter input/output, and divider output)
②	P4 port	;	4-bit I/O port	(serial bus interface, external interrupt input)
③	P5 port	;	2-bit I/O port	(timer/counter input/output, external interrupt input, oscillation stop detector output)
④	P6 port	;	8-bit I/O port	(analog input, analog reference power supply)

Each output port contains a latch, which holds the output data. All input ports do not have latches, so the external input data should either be held externally until read or reading should be performed several times before processing. Figure 2-2 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 can not be recognized from outside, so that transient input such as chattering must be processed by the program. Output data output 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 an write cycles may vary, depending on the instruction.

Figure 2-2. Input/Output timing (Example)

When reading an I/O port except programmable I/O ports, whether the pin input data or the output latch contents are read depends on the instructions, as shown below:

(1) Instructions that read the output latch contents

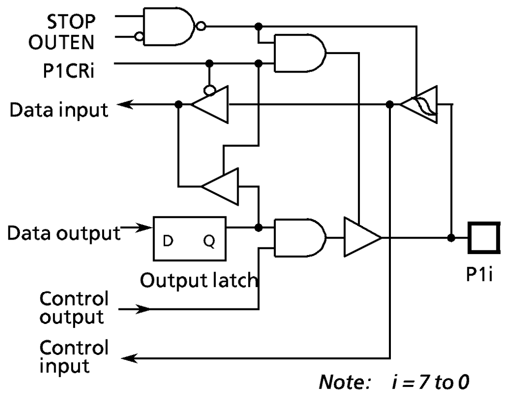
①	XCH r, (src)	⑤	LD (pp).b,CF
②	SET / CLR / CPL (src).b	⑥	ADD / ADDC / SUB / SUBB / AND / OR / XOR (src), n
③	SET / CLR / CPL (pp).g	⑦	(src) side of ADD/ADDC/SUB/SUBB/AND/OR/XOR (src), (HL)
④	LD (src).b, CF		

(2) Instructions that read the pin input data
Instructions other than the above (1) and (HL) side of ADD/ADDC/SUB/SUBB/AND/OR/XOR (scr), (HL)

2.2.1 Port P1 (P17 to P10)

Port P1 is an 8-bit input/output port which can be configured as an input or an output in on-bit unit. Input/output mode is specified by the port P1 input/output control register (P1CR). During reset, the P1CR is initialized to "0", which configures port P1 as an input. The P1 output latches are also initialized to "0".

Port P1 is also used as an external interrupt input, a timer/counter input, and a divider output. When used as secondary function pin, the input pins should be set to the input mode, and the output pins should be set to the output mode and beforehand the output latch should be set to "1". It is recommended that pins P11 and P12 should be used as external interrupt inputs, timer/counter input, or input ports. The interrupt latch is set at the rising or falling edge of the output when used as output ports. Pin 10 can be configured as either an input/output ports with INT0EN or an external interrupt input. During reset, pin P10 is configured as an input port.



Note: i = 7 to 0

P1 (0001_H)

7	6	5	4	3	2	1	0
P17	P16	P15	P14	P13 $\overline{\text{DVO}}$	P12 TC1	P11 INT1	P10 $\overline{\text{INT0}}$

(Initial value: 0000 0000)

P1CR ($000B_H$)

7	6	5	4	3	2	1	0

(Initial value: 0000 0000)

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

Note 1: Ports set to the input mode read the pin states. When input pin and output pin exist in port P1 together, the contents of the output latch of ports set to the input mode may be rewritten by executing the bit manipulation instructions. Pins set to the output mode read a value of the output latch.

Note 2: The P1CR is a write-only register. It can not be operated by the read-modify instruction (Bit manipulation instructions of SET, CLR, etc. and Arithmetic instructions of AND, OR, etc.)

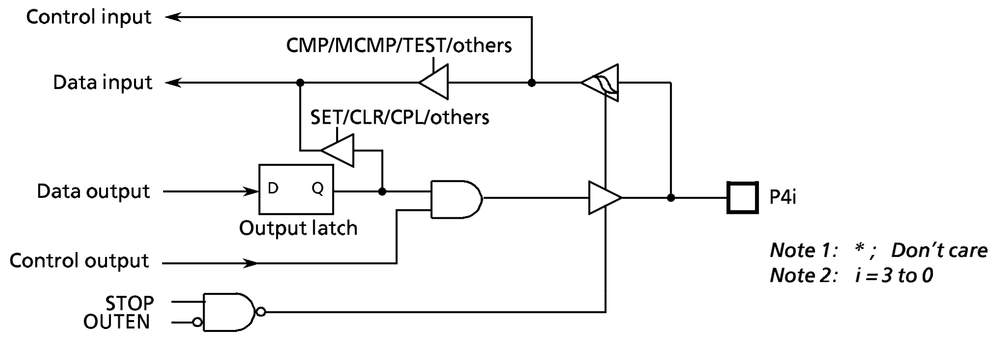
Figure 2-3. Port 1 and P1CR

Example: Sets P17, P16 and P14 as output ports, P13 and P11 as input ports, and the others as function pins. Internal output data is "1" for the P17 and P14 pins, and "0" for the P16 pin.

```
LD    (EINTCR), 01000000B      ;  INT0EN←1
LD    (P1), 10111111B          ;  P17←1, P14←1, P16←0
LD    (P1CR), 11010000B
```

2.2.2 Port P4 (P43 to P40)

Port P4 is a 4-bit input/output port, and is also used as a serial bus interface input/output, an external interrupt input and a STOP mode release signal input. High current output is available so LEDs can be driven directly.
When used as an input port or a secondary function pin, the output latch should be set to "1".
The output latches are initialized to "1" during reset.
Bits 7 to 4 are read in as "1" when read instruction is executed for the port P4.



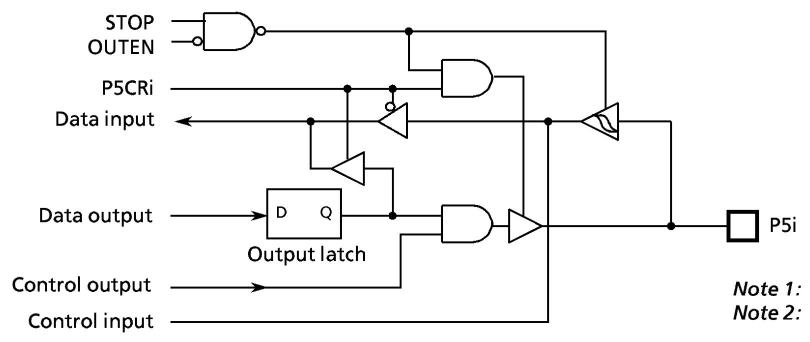
*Note 1: * ; Don't care*
Note 2: i = 3 to 0

	7	6	5	4	3	2	1	0	
P4 (0004_H)	–	–	–	–	P43 $\overline{\text{STOP}}$ $\overline{\text{INT5}}$	P42 SDA SO	P41 SCL SI	P40 $\overline{\text{SCK}}$	(Initial value: **** 1111)

Figure 2-4. Port P4

2.2.3 Port P5 (P51 to P50)

Port P5 is a 2-bit general-purpose input/output port which can be configured as an input or an output in one-bit unit. High current output is available so LEDs can be driven directly.
Input/output mode is specified by the port P5 input/output control register (P5CR). During reset, the P5CR is initialized to “0”, which configures port P5 as an input. The P5 output latches are also initialized to “0”.
Port P5 is also used as a timer/counter input, an external interrupt input and a oscillation stop detector output. When used as secondary function pin, the input pins should be set to the input mode, and the output pins should be set to the output mode. When used as timer/counter input, the output latch should be set to “1”.



*Note 1: * ; Don’t care*
Note 2: i = 1 to 0

P5 (0005_H) (Initial value: **** **00)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	P51 TC4 $\overline{\text{PWM}}$ $\overline{\text{PDO}}$ CLZ1	P50 TC3 INT3 CLZ0

P5CR (0026_H) (Initial value: **** **00)

7	6	5	4	3	2	1	0
–	–	–	–	–	–		

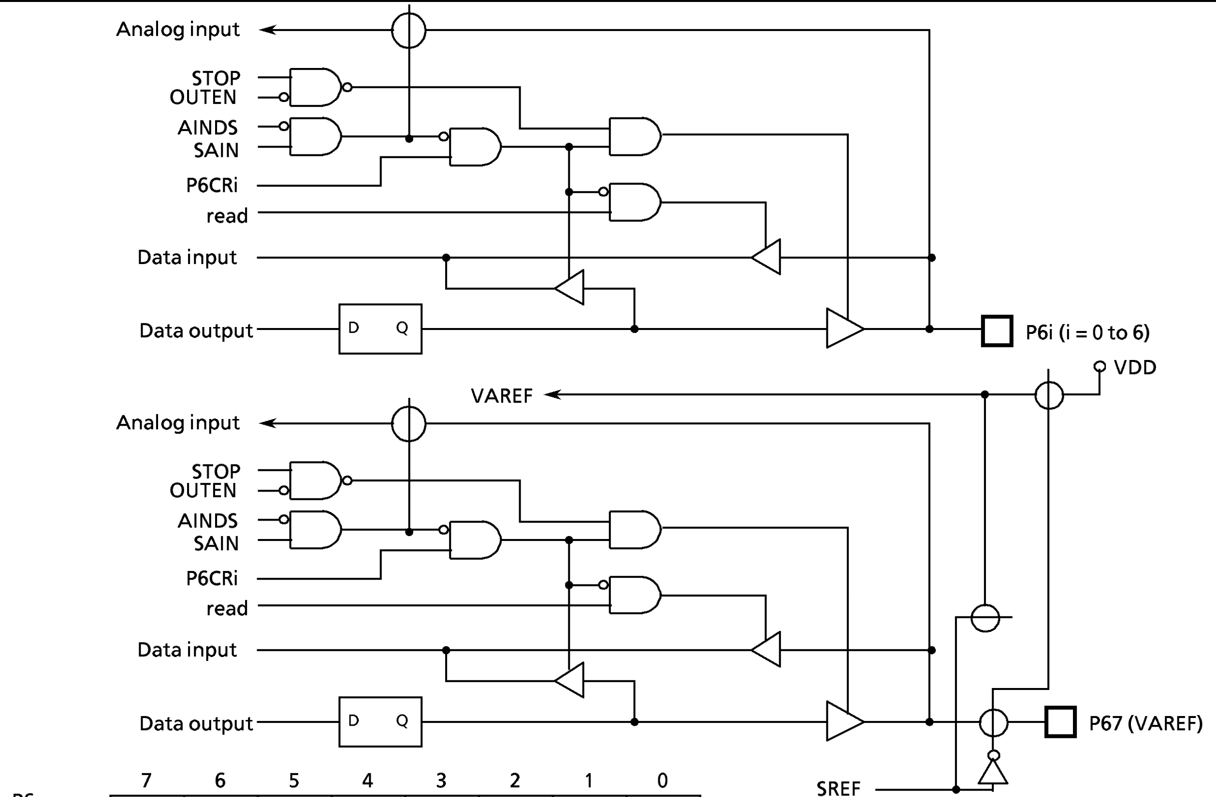
P5CR	I/O control for port P5 (Set for each bit individually)	0: Input mode 1: Output mode	Write-only

Note 1: Ports set to the input mode read the pin states. When input pin and output pin exist in port P5 together, the contents of the output latch of ports set to the input mode may be rewritten by executing the bit manipulation instructions. Pins set to the output mode read a value of the output latch.
Note 2: The P5CR is a write-only register. It can not be operated by the read-modify instruction (Bit manipulation instructions of SET, CLR, etc. and Arithmetic instructions of AND, OR, etc.)

Figure 2-5. Port P5 and P5CR

2.2.4 Port P6 (P67 to P60)

Port P6 is an 8-bit general-purpose input/output port which can be configured as an input or an output in one-bit unit. Port P6 is also used as analog inputs and P67 can be configured as an analog reference power supply pin (VAREF) by setting SREF to “1” (bit 7 in SELREF). Input / output mode is specified by port P6 input/output control register (P6CR) and AINDS (bit 4 in ADCCR). During reset, P6CR is set to “0”, AINDS is set to “1”, and port P6 is in input mode. During reset, the output latches of port P6 is initialized to “0”. P6CR is write-only register. When port P6 is not used as analog input, it can be used input / output port. However output instructions must not be performed to maintain the precision at using AD converter. When the input instruction is executed to port P6 during using AD converter, “0” is read into the pin that selects analog input, and “1” or “0” is read into the pin that does not select analog input, depending on the input level of pins.



P6 (0006$_H$)

7	6	5	4	3	2	1	0
P67 AIN7 VAREF	P66 AIN6	P65 AIN5	P64 AIN4	P63 AIN3	P62 AIN2	P61 AIN1	P60 AIN0

(Initial value: 0000 0000)

P6CR (000C$_H$)

7	6	5	4	3	2	1	0

(Initial value: 0000 0000)

P6CR	I/O control for port P6 (Set for each bit individually)	0: Input mode 1: Output mode	Write-only

Note: Used analog input pins must be configured as input mode.

P67/VAREF select register

SELREF (002F$_H$)

7	6	5	4	3	2	1	0
SREF							

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

SREF	0: Internal VDD 1: External level P67	Write-only

Note 1: Ports set to the input mode read the pin states. When input pin and output pin exist in port P6 together, the contents of the output latch of ports set to the input mode may be rewritten by executing the bit manipulation instructions. Pins set to the output mode read a value of the output latch.

Note 2: The P6CR and SELREF are a write-only registers. They can not be operated by the read-modify instruction (Bit manipulation instructions of SET, CLR, etc. and Arithmetic instructions of AND, OR, etc.)

Figure 2-6. Port P6, P6CR and SELREF

Example: The lower 4-bit of Port P6 is set to an output port and the others are set to an input port.

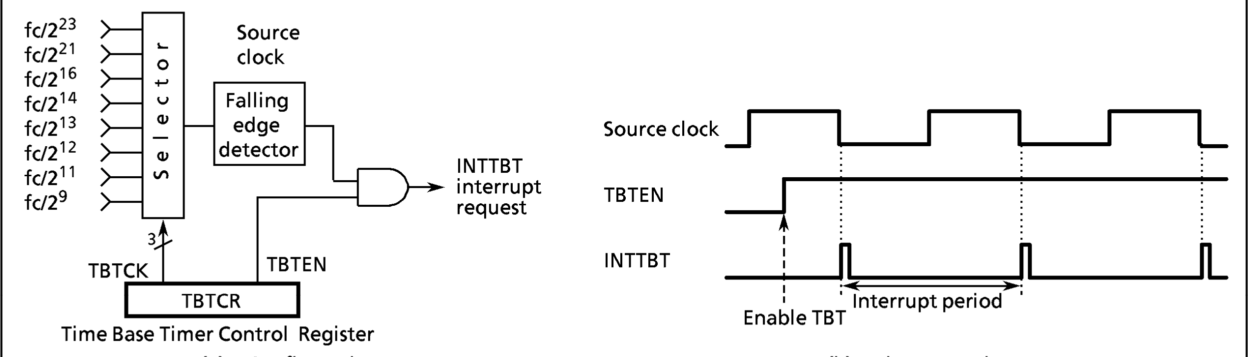
```
LD      (P6CR), 0FH        ;    P6CR←00001111
```

2.3 Time Base Timer (TBT)

The time-base timer is used to generate the base time for key scan and dynamic display processing. For this purpose, it generates a time-base timer interrupt (INTTBT) at predetermined intervals.

This interrupt is generated beginning with the first falling edge of the source clock (the timing generator's divider output selected by TBTCK) after the time-base timer is enabled. Note that since the divider cannot be cleared by a program, the first interrupt only may occur earlier than the set interrupt period. (See Figure 2-7 (b))

When selecting the interrupt frequency, make sure the time-base timer is disabled. (Do not change the selected interrupt frequency when disabling the active timer either.) However, you can select the interrupt frequency simultaneously when enabling the timer.



(a) Configuration (b) Time Base Timer Interrupt

Figure 2-7. Time base timer

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

```
LD      (TBTCR), 00001010B
SET     (EIRL). 6
```

TBTCR (0036$_H$)

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

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

TBTEN	Time base timer enable/disable	0: Disable 1: Enable	Write-only
TBTCK	Time base timer interrupt frequency select	000: fc/2^{23} [Hz] 001: fc/2^{21} 010: fc/2^{16} 011: fc/2^{14} 100: fc/2^{13} 101: fc/2^{12} 110: fc/2^{11} 111: fc/2^{9}	

*Note 1: fc ; clock [Hz], * ; Don't care*
Note 2: The fourth bit in TBTCR must be to "0".
Note 3: TBTCR are write-only registers, which cannot access any of in read-modify-write instructions such as bit operate, etc.

Figure 2-8. Time base timer control register

Table 2-1. Time base timer interrupt frequency [Hz]

TBTCK	NORMAL, IDLE mode (at fc = 8 MHz)
000	0.95
001	3.81
010	122.07
011	488.28
100	976.56
101	1953.12
110	3906.25
111	15625

2.4 Divider Output (DVO)

A 50% duty pulse can be output using the divider output circuit, which is useful for piezo-electric buzzer drive. Divider output is from pin P13 ($\overline{\text{DVO}}$). The P13 output latch should be set to "1" and then the P13 should be configured as an output mode.

TBTCR (0036_H)

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

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

DVOEN	Divider output enable/disable	0: Disable 1: Enable	Write-only
DVOCK	Divider output ($\overline{\text{DVO}}$ pin) frequency selection	00: $fc/2^{13}$ [Hz] 01: $fc/2^{12}$ 10: $fc/2^{11}$ 11: $fc/2^{10}$	

*Note 1: fc ; High-frequency clock [Hz], * ; Don't care*
Note 2: The divider output frequency should be selected with divider output disabled. (The divider output frequency should not also be changed when divider changed from enabled state to disabled state.)
Note 3: TBTCR are write-only registers, which cannot access any of in read-modify-write instructions such as bit operate, etc.

Figure 2-9. Divider output control register

Example: 1 kHz pulse output (at fc = 8 MHz)

```
SET     (P1).3              ;   P13 output latch ←1
LD (P1CR), 00001000B        ;   Configures P13 as an output mode
LD (TBTCR), 10000000B       ;   DVOEN←1, DVOCK←00
```

Table 2-2. Frequency of divider output [kHz]

DVOCK	At fc = 4 MHz	At fc = 8 MHz
00	0.512	0.976
01	1.024	1.953
10	2.048	3.906
11	4.096	7.812

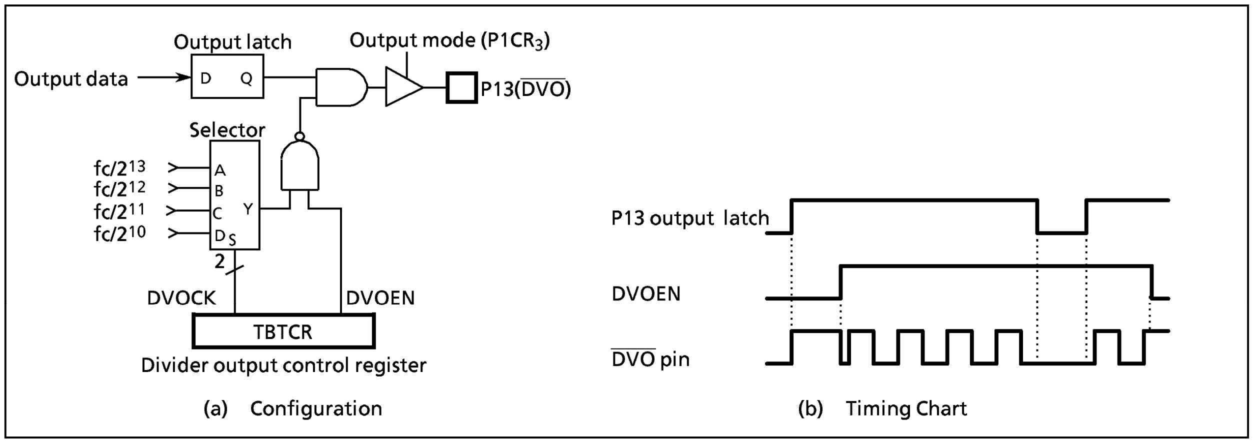


Figure 2-10. Divider output

2.5 16-bit Timer/Counter 1 (TC1)

The TMP87C409B/809B each have a multi-function 16-bit timer/counter (TC1).

2.5.1 Configuration

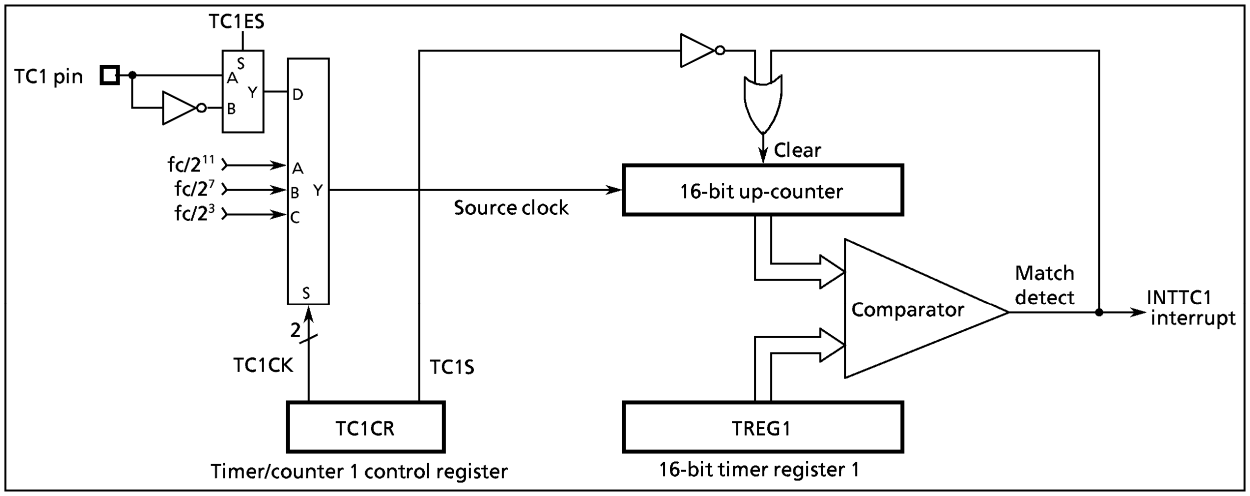


Figure 2-11. Timer/Counter 1

2.5.2 Control

The timer/counter 1 is controlled by a timer/counter 1 control register (TC1CR) and a 16-bit timer register (TREG1). Reset does not affect TREG1.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TREG1 (0010, 0011_H)	$TREG1_H$ (0011_H)								$TREG1_L$ (0010_H)							
									Write-only							

	7	6	5	4	3	2	1	0	
TC1CR (0014_H)	"0"	"0"	"0"	TC1S	TC1CK		"0"	"0"	(Initial value: ***0 0000)

TC1CK	Timer/counter 1 source clock select	00: Internal clock $fc/2^{11}$ 01: Internal clock $fc/2^{7}$ 10: Internal clock $fc/2^{3}$ 11: External clock (TC1 pin input)	Write-only
TC1S	Timer/counter 1 start control	0: Stop and counter clear 1: Command start	

*Note 1: fc; High-frequency clock [Hz], *; Don't care*
Note 2: Always write "0" to bits 0, 1, 5, 6 and 7 in TC1CR.
Note 3: Writing to the low-byte of timer register ($TREG1_L$), the comparison is inhibited until the high-byte ($TREG1_H$) is written.
After writing to the high-byte, the comparison of 1 machine cycle (during instruction execution) is ignored.
Note 4: Set the mode, the source clock and the edge (TC1ES) when the TC1 stops (TC1S = 0).
Note 5: Values to be loaded to the timer register must satisfy the following condition.
$TREG1 > 0$
Note 6: TC1CR and TREG1 are write-only registers, which cannot access any of in read-modify-write instructions such as bit operate, etc.

Figure 2-12. Timer register 1 and TC1 control register

2.5.3 Function

Timer/counter 1 has two operating modes: timer mode and event counter mode.

(1) **Timer** Mode

In this mode, counting up is performed using the internal clock. The contents of TREG1 are compared with the contents of up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared to "0". Counting up resumes after the counter is cleared.

Table 2-3. Timer/counter 1 source clock (Internal clock)

Source clock	Resolution	Maximum time setting
NORMAL, IDLE modes	At fc = 8 MHz	At fc = 8 MHz
$fc/2^3$ [Hz]	1 μs	65.5 ms
$fc/2^7$	16 μs	1.0 s
$fc/2^{11}$	256 μs	16.7 s

Example: Sets the timer mode with source clock $fc/2^{11}$[Hz] and generates an interrupt 1 s. later (at fc = 8 MHz).

```
LD    (TC1CR), 00000000B   ;  Sets the TC1 mode and source clock
LDW   (TREG1), 0F42H       ;  Sets the timer register (1 s ÷ 2^11/fc = F42H)
SET   (EIRL).EF4           ;  Enable INTTC1
EI
LD    (TC1CR), 00010000B   ;  Starts TC1
```

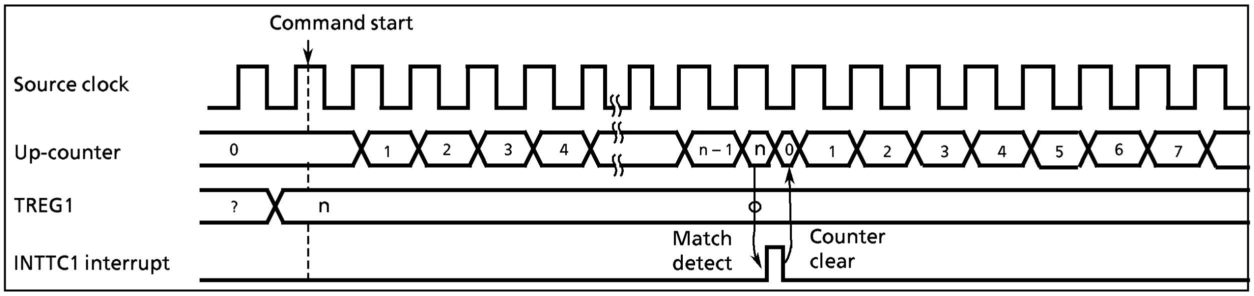


Figure 2-13. Timer mode timing chart

(2) **Event Counter** Mode

In this mode, events are counted on the edge of the TC1 pin input. Either the rising or falling edge can be selected with TC1ES in EINTCR. The contents of TREG1 are compared with the contents of the up-counter. If a match is found, an INTTC1 interrupt is generated, and the counter is cleared. The maximum applied frequency is $fc/2^4$ [Hz] in NORMAL or IDLE mode. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width.

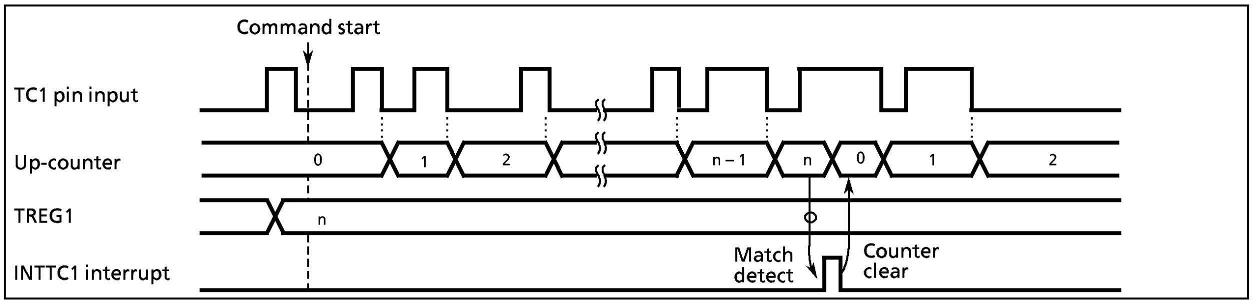


Figure 2-14. Event counter mode timing chart (TC1ES = 1)

2.6 8-Bit Timer/Counter 3 (TC3)

2.6.1 Configuration

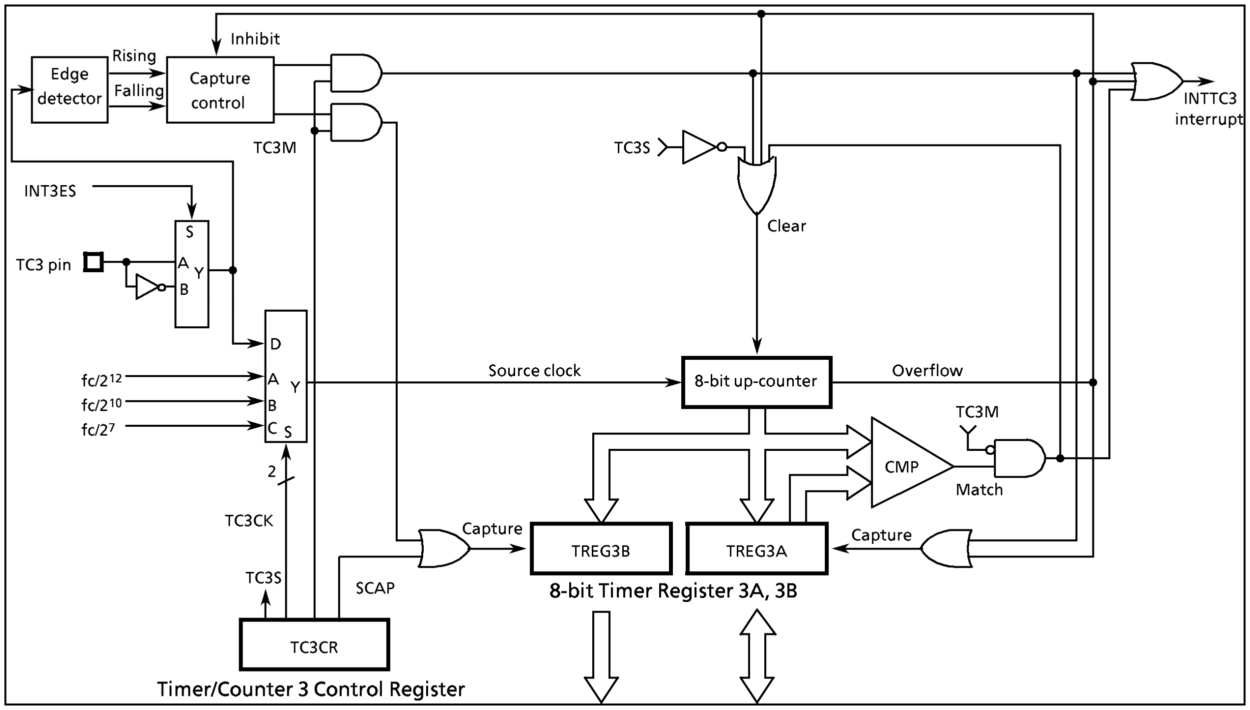


Figure 2-15. Timer/Counter 3

2.6.2 Control

	7	6	5	4	3	2	1	0	
TREG3A (0018_H)									Read/Write
TREG3B (0019_H)									Read-only
TC3CR ($001A_H$)	"0"	SCAP	"0"	TC3S	TC3CK		"0"	TC3M	(Initial value: ∗0∗0 00∗0)

TC3M	Timer/counter 3 operation mode set	0: Timer/event counter 1: Capture	Write-only
TC3CK	Timer/counter 3 source clock select	00: Internal clock $fc/2^{12}$ [Hz] 01: Internal clock $fc/2^{10}$ 10: Internal clock $fc/2^{7}$ 11: External clock (TC3 pin input)	
TC3S	Timer/counter 3 start control	0: Stop & clear 1: Start	
SCAP	Software capture control	0: – 1: Software capture	

Note 1: fc ; High-frequency clock [Hz] ∗ ; Don't care
Note 2: Set the mode, the source clock and the edge selection (INT3ES) when the TC3 stops (TC3S = 0).
Note 3: Values to be loaded into timer register 3A must satisfy the following condition.
TREG3A > 0 (in the timer/event counter mode)
Note 4: Software capture can be used in only timer and event counter modes.
Note 5: TC3CR is a write-only register and must not be used with any of read-modify-write instructions.

Figure 2-16. Timer register 3A/3B and TC3 control register

The timer/counter 3 is controlled by a timer/counter 3 control register (TC3CR) and two 8-bit timer registers (TREG3A and TREG3B). Reset does not affect these timer registers.

2.6.3 Function

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

(1) **Timer** Mode

In this mode, the internal clock is used for counting up. The contents of TREG3A are compared with the contents of up-counter. If a match is found, a timer/counter 3 interrupt (INTTC3) is generated, and the up-counter is cleared. Counting up resumes after the up-counter is cleared. The current contents of up-counter are loaded into TREG3B by setting SCAP (bit 6 in TC3CR) to "1". SCAP is automatically cleared after capturing.

Table 2-4. Source Clock (Internal Clock) for Timer Counter 3

Source clock	Resolution	Maximum setting time
NORMAL, IDLE mode	fc = 8 MHz	fc = 8 MHz
$fc/2^{12}$ [Hz]	512 μs	131.1 ms
$fc/2^{10}$	128 μs	32.6 ms
$fc/2^{7}$	16 μs	4.1 ms

(2) **Event Counter** Mode

In this mode, the TC3 pin input pulses are used for counting up. Either the rising or falling edge can be selected with INT3ES (bit 3 in EINTCR). Both edges can not be used. The contents of TREG3A are compared with the contents of the up-counter. If a match is found, an INTTC3 interrupt is generated and the counter is cleared. The maximum applied frequency is $fc/2^4$ [Hz] in the NORMAL or IDLE mode. Two or more machine cycles are required for both the "H" and "L" levels of the pulse width.
The current contents of up-counter are loaded into TREG3B by setting SCAP (bit 6 in TC3CR) to "1". SCAP is automatically cleared after capturing.

Example: Generates an interrupt every 0.5 s, inputing 50Hz pulses to the TC3 pin.

```
LD    (TC3CR) , 00001100      ;  Sets TC3 mode, source clock
LD    (TREG3A) , 19H          ;  0.5 s ÷ 1/50 = 25 = 19H
LD    (TC3CR) , 00011100B     ;  Start TC3
```

(3) Capture Mode

The pulse width, period and duty of the TC3 pin input are measured in this mode, which can be used in decoding the remote control signals, etc. The counter is free running by the internal clock. On the rising (falling) edge of the TC3 pin input, the current contents of counter is loaded into TREG3A, then the up-counter is cleared and an INTTC3 interrupt is generated. On the falling (rising) edge of the TC3 pin input, the current contents of the counter is loaded into the TREG3B. In this case, counting continues. At the next rising (falling) edge of the TC3 pin input, the current contents of counter are loaded into TREG3A, then the counter is cleared again and an interrupt is generated. If the counter overflows before the edge is detected, FF_H is set to the TREG3A and an overflow interrupt (INTTC3) is generated. During interrupt processing, it can be determined whether or not there is an overflow by checking whether or not the TREG3A value is FF_H. Also, after an interrupt (capture to TREG3A, or overflow detection) is generated, capture and overflow detection are halted until TREG3A has been read out; however, the counter continues.

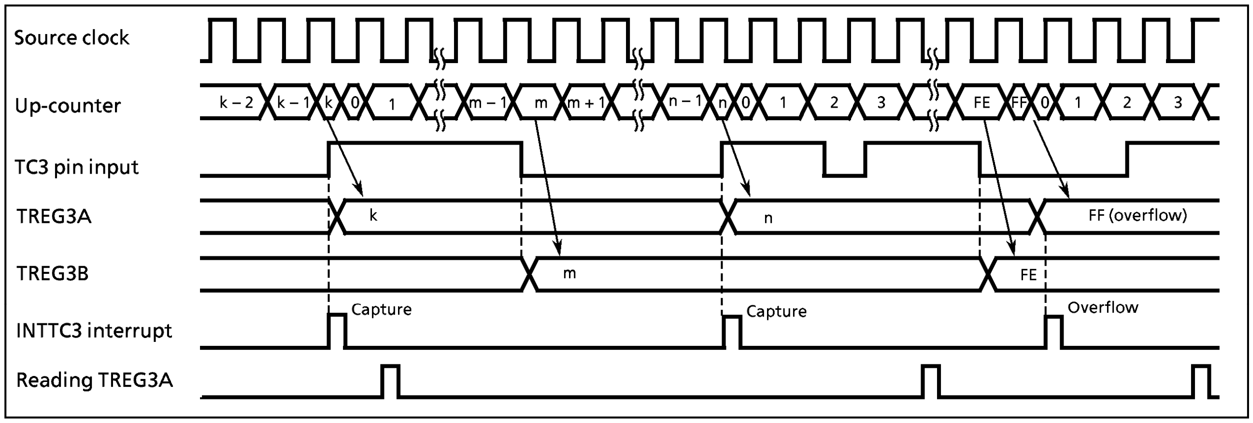


Figure 2-17. Timing chart for capture mode (INT3ES = 0)

2.7 8-bit Timer/Counter (TC4)

2.7.1 Configuration

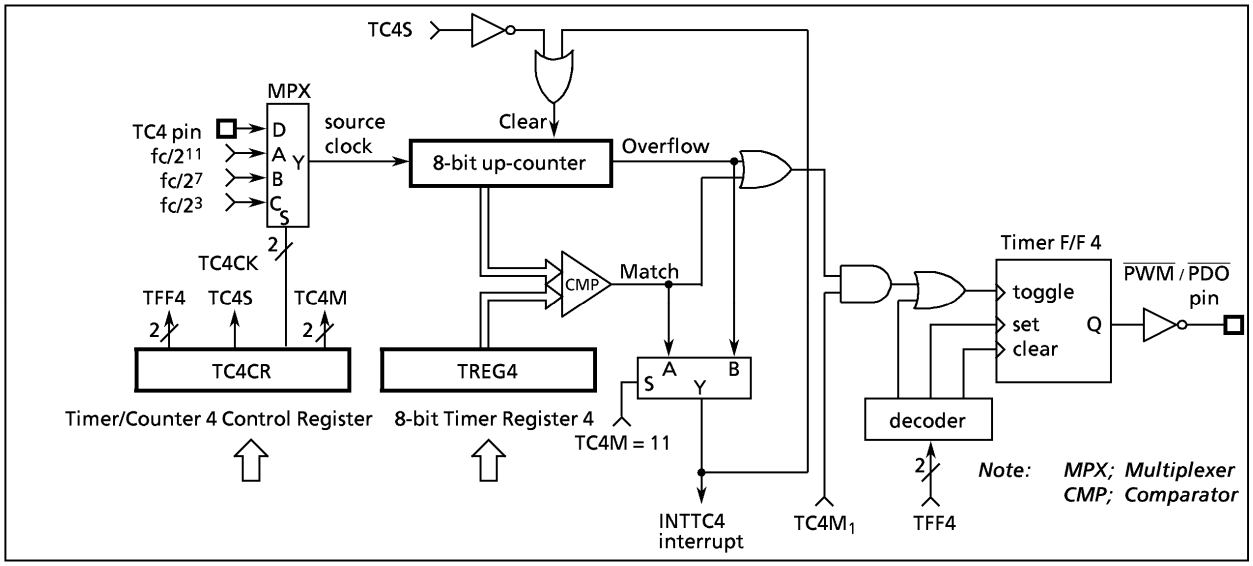


Figure 2-18. Timer/Counter 4

2.7.2 Control

The timer/counter 4 is controlled by a timer/counter 4 control register (TC4CR) and an 8-bit timer register 4 (TREG4). Reset does not affect TREG4.

	7	6	5	4	3	2	1	0	
TREG4 ($001B_H$)									Write-only
TC4CR ($001C_H$)	TFF4		"0"	TC4S	TC4CK		TC4M		(Initial value: 00*0 0000)

TC4M	TC4 operating mode select	00: Timer mode 01: Reserved 10: Programmable divider output (PDO) mode 11: Pulse width modulation (PWM) output mode	Write-only
TC4CK	TC4 source clock select	00: Internal clock fc/2^{11} [Hz] 01: Internal clock fc/2^7 10: Internal clockfc/2^3 11: External clock (TC4 pin input)	
TC4S	TC4 start control	0: Stop & clear 1: Start	
TFF4	Timer F/F 4 control	00: Clear 01: Toggle 10: Set 11: – (Note 3)	

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

Note 2: Set the operating mode, the source clock selection, the timer F/F 4 control and the edge selection (INT4ES) when the TC4 stops (TC4S = 0).

Note 3: TFF4 must be set to "11" in the timer and event counter modes.

Note 4: Values to be loaded to the timer register must satisfy the following condition.
TREG4>0

Note 5: TC4CR and TREG4 are write-only registers and must not be used with any of read-modify-write instructions.

Figure 2-19. Timer register 4 and TC4 control register

2.7.3 Function

The timer/counter 4 has four operating modes: timer, event counter, programmable divider output, and PWM output mode.

(1) **Timer** Mode

In this mode, the internal clock is used for counting up. The contents of TREG4 are compared with the contents of up-counter. If a match is found, a timer/counter 4 interrupt (INTTC4) is generated and the up-counter is cleared to "0". Counting up resumes after the up- counter is cleared.

Table 2-5. Source clock (Internal clock) for timer/counter 4

Source clock	Resolution	Maximum setting time
NORMAL, IDLE mode	fc = 8 MHz	fc = 8 MHz
$fc/2^{11}$ [Hz]	256 μs	65.3 ms
$fc/2^{7}$	16 μs	4.1 ms
$fc/2^{3}$	1 μs	255 μs

(2) **Event Counter** Mode

In this mode, the TC4 pin input (external clock) pulse is used for counting up. The contents of TREG4 are compared with the contents of the up-counter. If a match is found, an INTTC4 interrupt is generated and the counter is cleared. The maximum applied frequency is $fc/2^4$ [Hz] in NORMAL or IDLE mode. Two or more machine cycles are required for both the high and low levels of the pulse width.

(3) **Programmable Divider Output** (PDO) Mode

The internal clock is used for counting up. The contents of TREG4 are compared with the contents of the up-counter. Timer F/F 4 output is toggled and the counter is cleared each time a match is found. Timer F/F 4 output is inverted and output to the $\overline{\text{PDO}}$ (P51) pin. This mode can be used for 50 % duty pulse output. Timer F/F 4 can be initialized by program, and it is initialized to "0" during reset. An INTTC4 interrupt is generated each time the $\overline{\text{PDO}}$ output is toggled.

Example: Output a 1024 Hz pulse (at fc = 4.194304 MHz)

```
SET   (P5). 1              ;  P51 output latch ← 1
LD    (P5CR), 00000010     ;  Sets P51 (output mode)
LD    (TREG4), 10H         ;  (1/1024 ÷ 2^7/fc) ÷ 2 = 10H
LD    (TC4CR), 00010010B   ;  Starts TC4
```

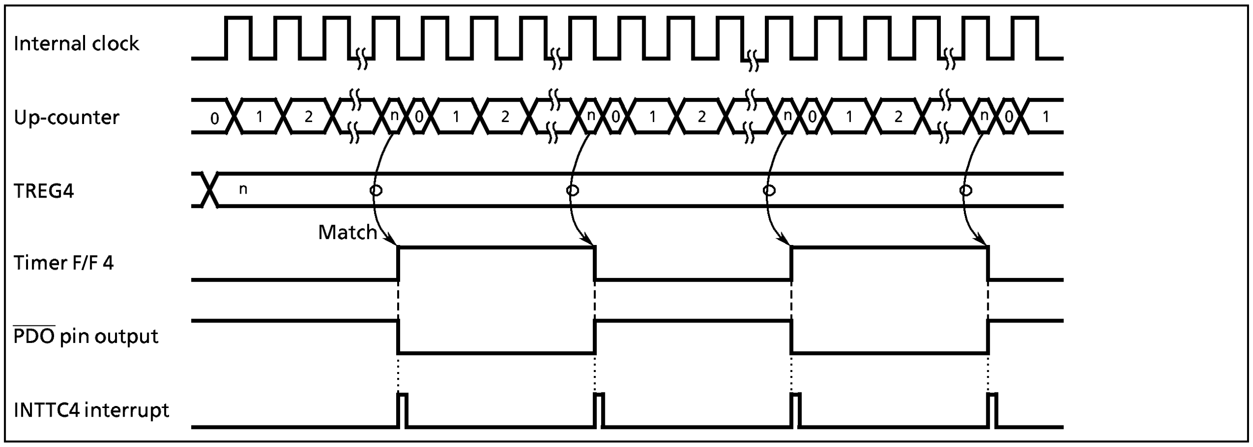


Figure 2-20. Timing chart for PDO mode

(4) **Pulse Width Modulation** (PWM) **Output** Mode

PWM output with a resolution of 8 bits is possible. The internal clock is used for counting up. The contents of TREG4 are compared with the contents of up-counter. If a match is found, the timer F/F 4 output is toggled. The counter continues counting. And, when an overflow occurs, the timer F/F4 output is again toggled and the counter is cleared. Timer F/F 4 output is inverted and output to the $\overline{\text{PWM}}$ (P51) pin. An INTTC4 interrupt is generated when an overflow occurs.

TREG4 is configured a 2-stage shift register and, during output, will not switch until one output cycle is completed even if TREG4 is overwritten; therefore, output can be altered continuously. Also, the first time, TREG4 is shifted by setting TC4S (bit 4 in TC4CR) to "1" after data are loaded to TREG4.

Note: Do not overwrite TREG4 only when an INTTC4 interrupt is generated. Usually, TREG4 is overwritten in the routine of INTTC4 interrupt service.

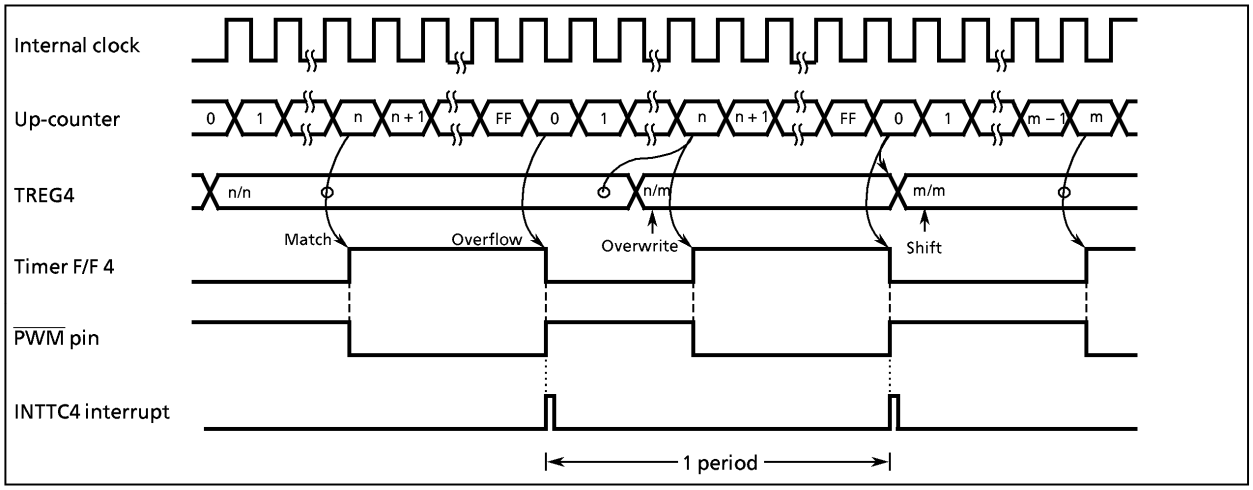


Figure 2-21. Timing chart for PWM mode

Table 2-6. PWM output mode

Source clock	Resolution	Maximum setting time
NORMAL, IDLE mode	fc = 8 MHz	fc = 8 MHz
$fs/2^{11}$ [Hz]	256 μs	65.5 ms
$fc/2^7$	16 μs	4.1 ms
$fc/2^3$	1 μs	256 μs

2.8 Serial Bus Interface (SBI-ver. B)

The TMP87C409B/809B each have a 1-channel serial bus interface which employs a clocked-synchronous 8-bit serial bus interface and an I^2C bus. (a bus system by philips)
The serial bus interface is connected to an external device through P42 (SDA) and P41 (SCL) in the I^2C bus mode; and through P40 ($\overline{SCK}$), P42 (SO), and P41 (SI) in the clocked-synchronous 8-bit SIO mode.
The serial bus interface pins are also used as the P4 port. When used for serial bus interface pins, set the P4 output latches of these pins to "1". When not used as serial bus interface pins, the P4 port is used as a normal I/O port.

2.8.1 Configuration

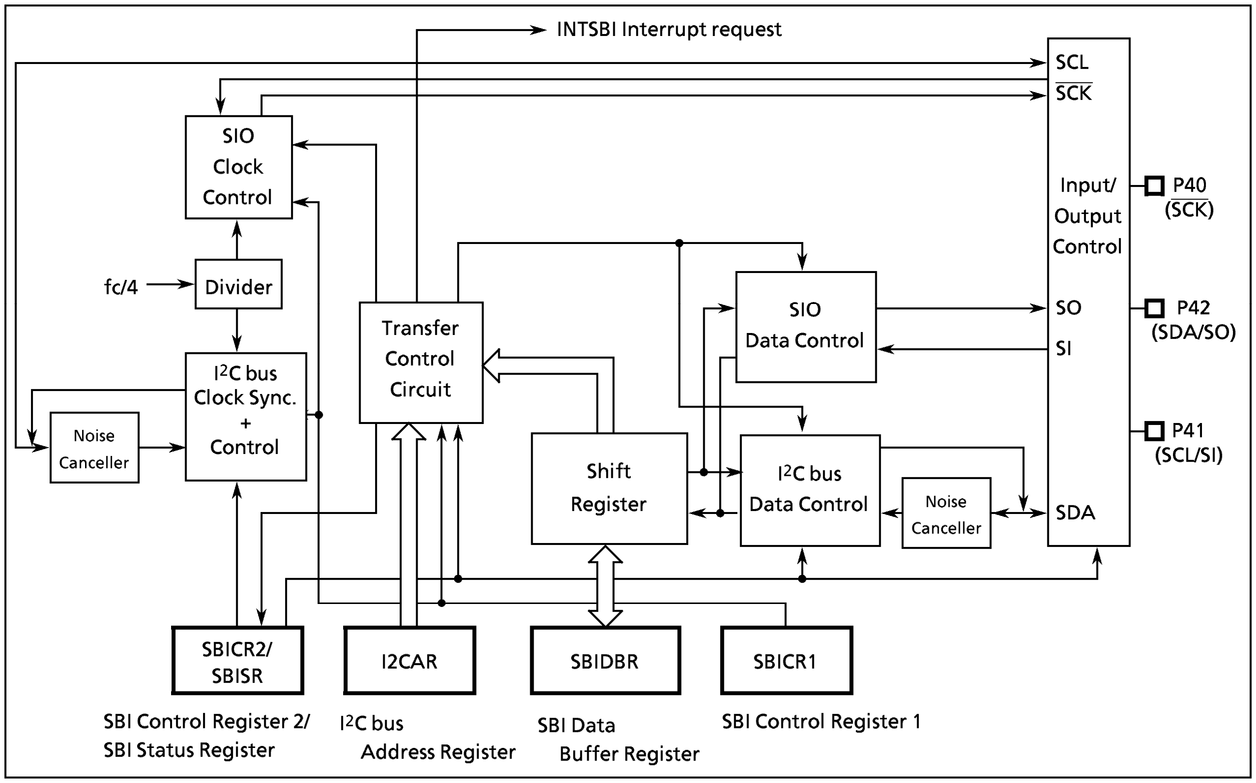


Figure 2-22. Serial bus interface (SBI-ver. B)

2.8.2 Serial bus interface (SBI-ver. B) control

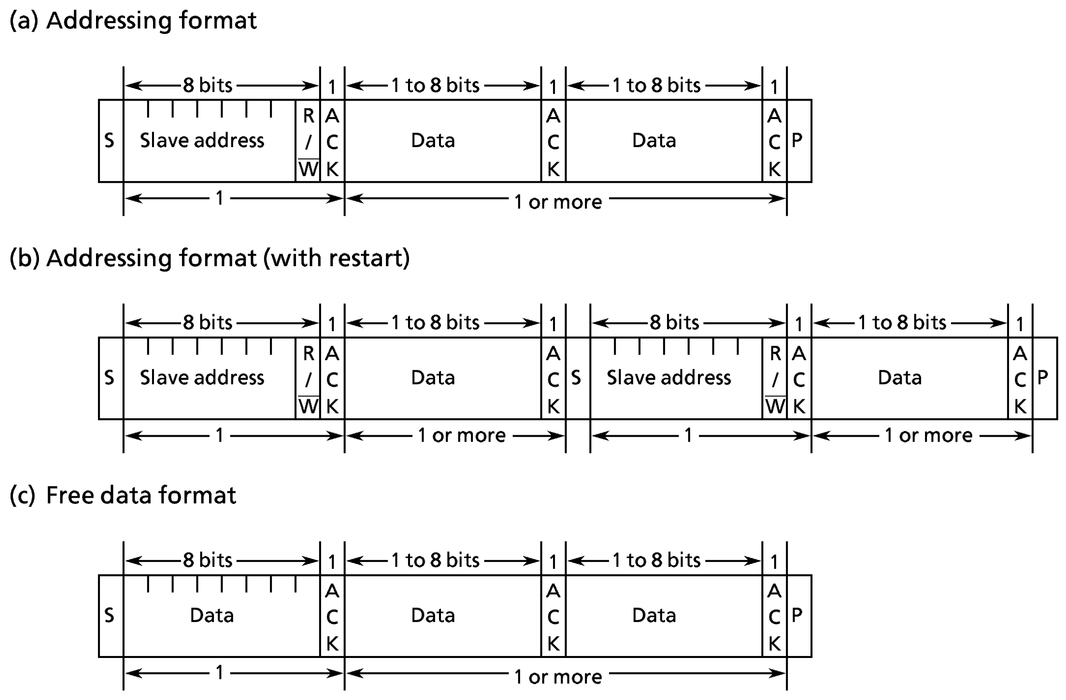
The following registers are used for control and operation status monitoring when using the serial bus interface (SBI-ver. B).

- Serial bus interface control register 1 (SBICR1)
- Serial bus interface control register 2 (SBICR2)
- Serial bus interface data buffer register (SBIDBR)
- I^2C bus address register (I2CAR)
- Serial bus interface status register (SBISR)

The above registers differ depending on a mode to be used.
Refer to Section "2.8.4 I^2C bus Mode Control" and "2.8.6 Clocked-synchronous 8-bit SIO Mode Control".

2.8.3 The data formats in the I^2C bus mode

The data formats in the I^2C bus mode are shown below.



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

Figure 2-23. Data format at I^2C bus mode

2.8.4 I^{2}C bus Mode Control

The following registers are used for control the serial bus interface (SBI-ver. B) and monitor the operation status in the I^{2}C bus mode.

Serial Bus Interface Control Register 1

SBICR1 (0020$_H$)

7	6	5	4	3	2	1	0
BC			ACK	SWRST	SCK		

(Initial value: 0000 *000)

BC	Number of transferred bits	BC	ACK = 0 Number of Clock	ACK = 0 Bits	ACK = 1 Number of Clock	ACK = 1 Bits	Write-only
		000	8	8	9	8	
		001	1	1	2	1	
		010	2	2	3	2	
		011	3	3	4	3	
		100	4	4	5	4	
		101	5	5	6	5	
		110	6	6	7	6	
		111	7	7	8	7	
ACK	Acknowledge mode specification	0: Not generate clock pulse for acknowledge signal (in master mode) / Not count clock pulse for acknowledge signal (in slave mode) 1: Generate clock pulse for acknowledge signal (in master mode) / Count clock pulse for acknowledge signal (in slave mode)					Read/ Write
SWRST	Initiate a internal of SBI	0: – 1: Initialized (Clearing "0" after initialized)					Read/ Write
SCK	Serial clock selection	000: Reserved (Note 5) 001: Reserved (Note 5) 010: 58.8 kHz 011: 30.3 kHz 100: 15.4 kHz 101: 7.75 kHz 110: 3.89 kHz 111: Reserved } at fc = 8 MHz (Output on SCL pin)					Write-only

*Note 1: fc ; high-frequency clock [Hz], * ; Don't care*
Note 2: Set the BC to "000" before switching to a clock-synchronous 8-bit SIO bus mode.
Note 3: SBICR1 has write-only register bits, which cannot access any of in read-modify-write instructions such as bit operate, etc.
*Note 4: * ; Don't care*
Note 5: This I^{2}C bus circuit does not support the Fast mode. It supports the Standard mode only. Although the I^{2}C bus circuit itself allows the setting of a baud rate over 100 kbps, the compliance with the I^{2}C specification is not guaranteed in that case.

Serial Bus Interface Data Buffer Register

SBIDBR (0021$_H$)

7	6	5	4	3	2	1	0

Read / Write (Initial value: 0000 0000)

Note 1: When writing transmitted data, start from the MSB (bit7).
Note 2: Cannot read the data which was written into SBIDBR, since a write data buffer and a read data buffer are independent in SBIDBR. Therefore, cannot access it any of in read-modify-write instructions such as bit operate, etc.
Note 3: The data which was written into SBIDBR is cleared to "0" when INTSBI is generated.

I^{2}C bus Address Register

I2CAR (0022$_H$)

7	6	5	4	3	2	1	0
Slave address							ALS
SA6	SA5	SA4	SA3	SA2	SA1	SA0	

(Initial value: 0000 0000)

SA	TMP87C409B/809B slave address selection		Write-only
ALS	Address recognition mode specification	0: Slave address recognition 1: Non slave address recognition	

Note: I2CAR is a write-only register, which cannot access any of in read-modify-write instructions such as bit operate, etc.

Figure 2-24. Serial bus interface control register 1/serial bus interface data buffer register/ I^{2}C bus address register in the I^{2}C bus mode

Serial Bus Interface Control Register 2

SBICR2 (0023_H) (Initial value: 0001 00**)

7	6	5	4	3	2	1	0
MST	TRX	BB	PIN	SBIM		"0"	"0"

MST	Master/slave selection	0: Slave 1: Master	Write-only
TRX	Transmitter/receiver selection	0: Receiver 1: Transmitter	
BB	Start/stop generation	0: Generate the stop condition when the MST, TRX, and PIN are "1". 1: Generate the start condition when the MST, TRX, and PIN are "1".	
PIN	Cancel interrupt service request	0: – (cannot be cleared to "0") 1: Cancel interrupt service request	
SBIM	Serial bus interface operating mode selection	00: Port mode (serial bus interface output disable) 01: SIO mode 10: I^2C bus mode 11: Reserved	

*Note 1: * ; Don't care*
Note 2: Switch a mode to port mode after confirming that the bus is free.
Note 3: Switch a mode to I2Cbus mode after confirming that input signals via port are "H" level.
Note 4: SBICR2 has write-only register bits, which cannot access any of in read-modify-write instructions such as bit operate, etc.
Note 5: Write "0" to bit 1, 0 in the SBICR2.

Serial Bus Interface Status Register

SBISR (0023_H) (Initial value: 0001 0000)

7	6	5	4	3	2	1	0
MST	TRX	BB	PIN	AL	AAS	AD0	LRB

MST	Master/slave status monitor	0: Slave 1: Master	Read-only
TRX	Transmitter/receiver status monitor	0: Receiver 1: Transmitter	
BB	I^2C bus status monitor	0: Bus free 1: Bus busy	
PIN	Interrupt service request status monitor	0: INTSBI occurs 1: INTSBI does not occur	
AL	Arbitration loss detection monitor	0: Arbitration loss undetected 1: Arbitration loss detected	
AAS	Slave address match detection monitor	0: Slave address unmatch or "GENERAL CALL" undetected 1: Slave address match or "GENERAL CALL" detected	
AD0	"GENERAL CALL" detection monitor	0: "GENERAL CALL" undetected 1: "GENERAL CALL" detected	
LRB	Last received bit monitor	0: Last received bit "0" 1: Last received bit "1"	

Figure 2-25. Serial bus interface control register 2/serial bus interface status register in the I^2C bus mode

(1) **Acknowledge mode specification**

Set the ACK (bit 4 in SBICR1) to "1" for operation in the acknowledge mode. The TMP87C409B/809B generate an additional clock pulse for an acknowledge signal when operating in the master mode. In the transmitter mode during the clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the "L" level in order to generate the acknowledge signal.

Reset the ACK for operation in the non-acknowledge mode. The TMP87C409B/809B don't generate a clock pulse for the acknowledge signal when operating in the master mode.

In the acknowledge mode, the TMP87C409B/809B count a clock pulse for the acknowledge signal when operating in the slave mode. During the clock pulse, when the received slave address is the same as the value set at the I2CAR or when a GENERAL CALL is received, the SDA pin is set to the "L" level in order to generate the acknowledge signal.

In the transmitter mode during the clock pulse cycle after matching the slave addresses or receiving a GENERAL CALL, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the "L" level in order to generate the acknowledge signal.

In non-acknowledge mode, the TMP87C409B/809B don't count a clock pulse for the acknowledge signal when operating in the slave mode.

(2) **Number of transfer bits**

The BC (bits 7 to 5 in the SBICR1) is used to select a number of bits for next transmitting and receiving data.

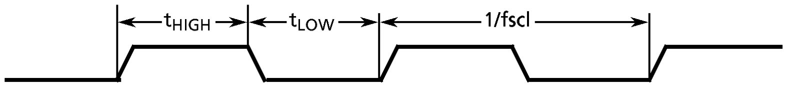
Since the BC is cleared to "000" as a start condition, a slave address and direction bit transmissions are executed in 8 bits. Other than these, the BC retains a specified value.

(3) **Serial clock**

a. Clock source

The SCK (bits 2 to 0 in the SBICR1) is used to select a maximum transfer frequency outputed on the SCL pin in the master mode. Set a communication baud rate that meets the I^{2}C bus specification, such as the shortest pulse width of t_{LOW}, based on the equations shown below.

In both master mode and slave mode, a pulse width of at least 4 machine cycles is require for both "H" and "L" levels.



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

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

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

Note: fc ; high-frequency clock

SCK (bits2 to 0 in the SBICR1)	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10

Figure 2-26. Clock source

b. Clock synchronization

In the I^2C bus mode, in order to drive a bus with a wired-AND, a master device which pulls down a clock line to "L" level, in the first place, invalidate a clock pulse of another master device which generates a "H" level clock pulse. The master device with a "H" level clock pulse needs to detect the situation and implement the following procedure.

The TMP87C409B/809B have a clock synchronization function for normal data transfer even when more than one master exists on a bus.

The example explains clock synchronization procedures when two masters simultaneously exist on a bus.

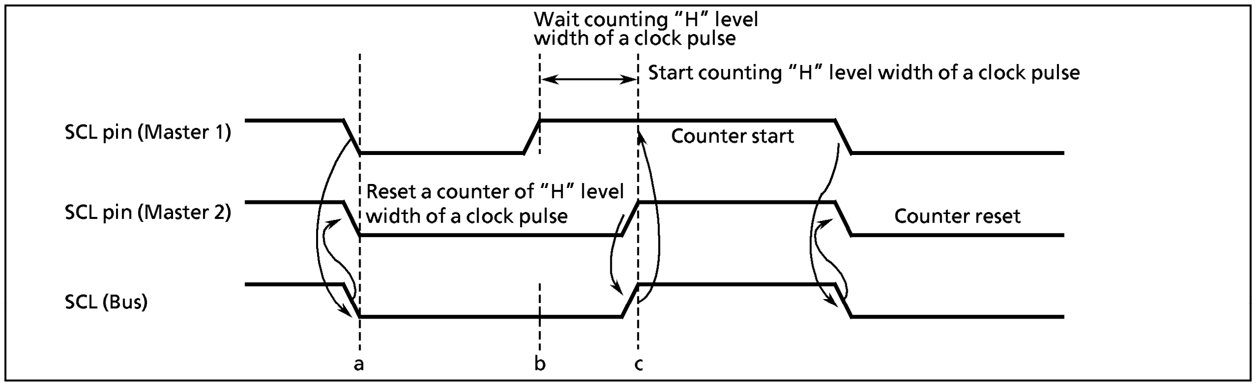

Wait counting "H" level width of a clock pulse
Start counting "H" level width of a clock pulse
SCL pin (Master 1)
Counter start
Reset a counter of "H" level width of a clock pulse
SCL pin (Master 2)
Counter reset
SCL (Bus)
a
b
c

Figure 2-27. Clock synchronization

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

Master 1 finishes counting a clock pulse in the "L" level at point "b" and sets the SCL pin to the "H" level. Since Master 2 holds the SCL line of the bus at the "L" level, Master 1 waits for counting a clock pulse in the "H" level. After Master 2 sets a clock pulse to the "H" level at point "c" and detects the SCL line of the bus at the "H" level, Master 1 starts counting a clock pulse in the "H" level.

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

(4) **Slave address and Address recognition mode specification**

When the serial bus interface circuit is used with an addressing format to recognize the slave address, clear the ALS (bit 0 in I2CAR) to "0", and set the SA (bits 7 to 1 in I2CAR) to the slave address.

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

(5) **Master/slave selection**

Set the MST (bit 7 in the SBICR2) to "1" for operating the TMP87C409B/809B as a masterdevice. Reset the MST to "0" for operation as a slave device. The MST is cleared to "0" by the hardware after a stop condition on a bus is detected or arbitration lost is detected.

(6) **Transmitter / receiver selection**

Set the TRX (bit 6 in SBICR2) to "1" for operating the TMP87C409B/809B as a transmitter. Reset the TRX for operation as a receiver. When data with an addressing format is transferred in the slave mode, the TRX is set to "1" if the direction bit (R/$\overline{W}$) sent from the master device is "1", and is cleared to "0" if the bit is "0". In the master mode, after an acknowledge signal is returned from the slave device with the hardware, the TRX is set to "0" if a transmitted direction bit is "1", and set to "1" if it is "0". When an acknowledge signal is not returned, the current condition is maintained.

The TRX is cleared to "0" by the hardware after a stop condition on the bus is detected or arbitration is lost.

The following table shows TRX changing conditions and TRX value after changing.

Mode	Direction bit	Conditions	TRX after changing
Slave mode	0	When the received slave address is the same as I2CAR	0
	1		1
Master mode	0	When the ACK signal is returned	1
	1		0

When the serial bus interface circuit is used with a free data format, the TRX is not changed by hardware since the slave address and the direction bit are not recognized, and they are processed as data from immediately after start condition.

(7) **Start / stop condition generation**

A start condition and 8-bit data are output on the bus by writing "1" to the MST, TRX and BB when the BB (bit 5 in SBICR2) is "0". It is necessary to set the transmitting data to the data buffer register and "1" to ACK beforehand.

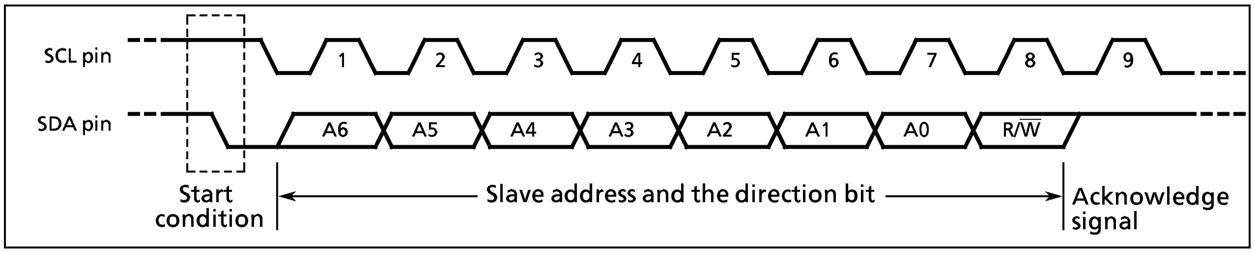


Figure 2-28. Start condition generation and slave address generation

When the BB is "1", a sequence of generating a stop condition is started by writing "1" to the MST, TRX, and PIN, and "0" to the BB.

Do not modify the contents of MST, TRX, BB and PIN until a stop condition is generated on a bus.

Note: When a stop condition is generated and bus SCL line is set to "L" level by the other devices, a stop condition is not started normally. Write "1" to the MST, TRX, and PIN, and "0" to the BB to generate a stop condition after releasing the SLC line.

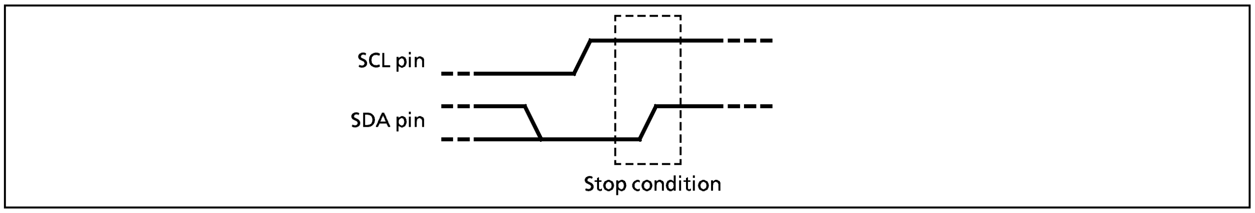


Figure 2-29. Stop condition generation

Note: When a stop condition is generated, a time to rise the SCL line should not exceed $tr = 2^n/fc - 3.5 \times 4/fc$ (s). (n depends on the SCK) If the rising time of the SCL line exceeds the above value, there is a probability that a stop condition is not started normally.

SCK	n	tr (Max, fc = 8 MHz)	tr (Max, fc = 4 MHz)
000	4	0.25 μs	0.50 μs
001	5	2.25 μs	4.50 μs
010	6	6.25 μs	12.5 μs
011	7	14.2 μs	28.5 μs
100	8	30.2 μs	60.5 μs
101	9	62.5 μs	124.5 μs
110	10	126.25 μs	252.5 μs

fc; High-frequency clock [Hz]

The bus condition can be indicated by reading the contents of the BB (bit 5 in the SBISR). The BB is set to "1" when a start condition on a bus is detected,and is cleared to "0" when a stop condition is detected on a bus.

(8) **Interrupt service request cancel**

In the master mode, a serial bus interface interrupt request (INTSBI) occurs after the number of clocks which is specified by the BC and ACK has been transmitted.
In the slave mode, when the received slave address is the same as the value set at the I2CAR, after outputting the acknowledge signal when a GENERAL CALL is received, or when data transfer is complete after matching the slave addresses or receiving a GENERAL CALL, an INTSBI interrupt request occurs.
When a serial bus interface interrupt request occurs, the PIN (bit 4 in SBISR) is cleared to "0". During the time that the PIN is "0", the SCL pin is pulled down to the "L" level.
Either writing / reading data to / from the SBIDBR sets the PIN to "1".
The time from the PIN being set to "1" until the SCL pin is released takes t_{LOW}.
Although the PIN (bit 4 in SBICR2) can be set to "1" by the program, the PIN is not set to "0" when "0" is written.

(9) **Serial bus interface operating mode**

The SBIM (bits 3 , 2 in SBICR2) is used to specify the serial bus interface operation mode. Set the SBIM to "10" after confirming that the serial bus interface pin is set to "H" level when used in the I^{2}C bus mode.
Switch a mode to port after making sure that a bus is free.

(10) **Arbitration lost detection monitor**

Since more than one master device can exist simultaneously on a bus in the I^{2}C bus mode, a bus arbitration procedure is implemented in order to guarantee the contents of transferred data.
Data on the SDA line is used for bus arbitration of the I^{2}C bus.
The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on the bus. Master 1 and Master 2 output the same data until point "a". After Master 1 outputs "1" and Master 2, "0", the SDA line of the bus is wired AND and the SDA line is pulled down to the "L" level by Master 2. When the SCL line of the bus is pulled up at point "b", the slave device reads data on the SDA line, that is, data in Master 2. Data transmitted from Master 1 becomes invalid. The state in Master 1 is called "arbitration lost". A master device which loses arbitration releases the SDA pin and the SCL pin in order not to effect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.

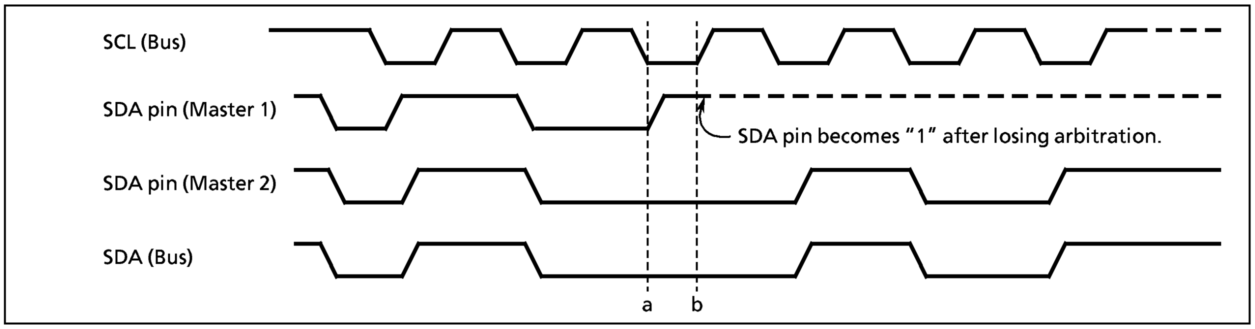


Figure 2-30. Arbitration lost

The TMP87C409B/809B compare levels of the SDA line of the bus with those of the TMP87C409B/809B SDA pin at the rising edge of the SCL line. If the levels are unmatched, arbitration is lost and the AL (bit 3 in SBISR) is set to "1".
When the AL is set to "1", the MST and TRX are reset to "0" and the mode is switched to a slave receiver mode.
The AL is reset to "0" by writing/reading data to / from the SBIDBR or writing data to the SBICR2.

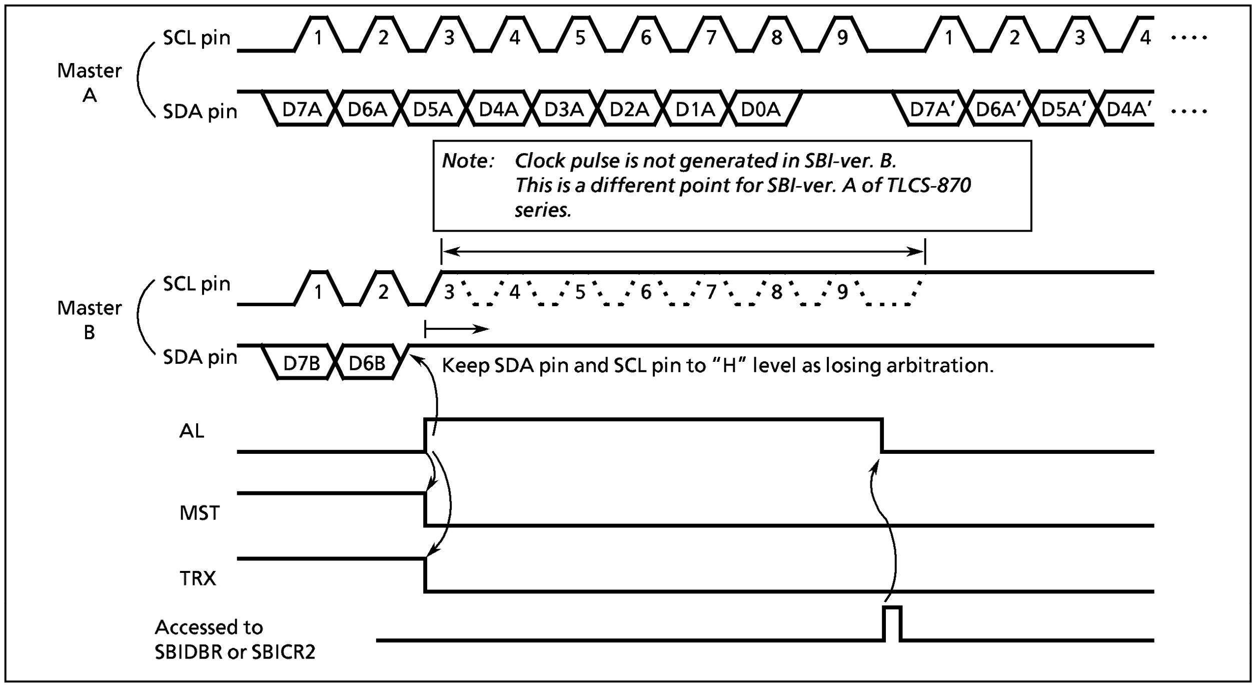

Master A
SCL pin
SDA pin
1 2 3 4 5 6 7 8 9 1 2 3 4 ····
D7A D6A D5A D4A D3A D2A D1A D0A D7A' D6A' D5A' D4A' ····
Note: Clock pulse is not generated in SBI-ver. B. This is a different point for SBI-ver. A of TLCS-870 series.
Master B
SCL pin
SDA pin
1 2 3 4 5 6 7 8 9
D7B D6B
Keep SDA pin and SCL pin to "H" level as losing arbitration.
AL
MST
TRX
Accessed to SBIDBR or SBICR2

Figure 2-31. Example of arbitration lost of when TMP87C409B/809B are a master device B

(11) **Slave address match detection monitor**

The AAS (bit 2 in the SBISR) is set to "1" in the slave mode, in the address recognition mode (ALS = 0), when receiving GENERAL CALL or a slave address with the same value that is set to the I2CAR. When the ALS is "1", the AAS is set to "1" after receiving the first 1-word of data. The AAS is cleared to "0" by after writing/reading data to/from a data buffer register.

(12) **GENERAL CALL detection monitor**
The AD0 (bit 1 in SBISR) is set to "1" in the slave mode, when all 8-bit data received immediately after a start condition are "0". The AD0 is cleared to "0" when a start or stop condition is detected on the bus.

(13) **Last received bit monitor**
The SDA value stored at the rising edge of the SCL line is set to the LRB (bit 0 in SBISR). When the contents of the LRB are read immediately after an INTSBI interrupt request is generated in the acknowledge mode, and ACK signal is read.

(14) **Software reset function**
Software reset function is used to initialize SBI, when SBI is locked by external noise, etc.
SWRST is set to "1", internal reset signal pulse is generated and inputted into SBI circuit.
All command registers and status registers are initialized to an initial value.
SWRST is automatically cleared to "0" after initialize SBI circuit.

2.8.5 Data transfer in I^2C bus mode

(1) **Device Initialization**
First, set the ACK in the SBICR1 to "1", the BC to "000", and the data length to 8-bit to count a clock pulse for the acknowledge signal. In addition, set the transmit frequency to the SCK.
Next, set the slave address to the SA in the I2CAR. Clear the ALS to "0" to set the addressing format.
After confirming that the serial bus interface pin is "H" level, for specifying the default setting to a slave receiver mode, clear "0" to the MST, TRX, and BB in the SBICR2; "1" to the PIN; "10" to the SBIM; and "0" to bits 1 and 0.

Note: To initialize the serial bus interface circuit, a constant period that the start conditions are not generated for any device is required after all devices which are connected to the bus are initialized. Then, the initialization must be completed during the period. If not, other devices may start transmitting data before the serial bus interface circuit has been initialized. Thus, data can not be normally received.

(2) **Start Condition and Slave Address Generation**
Confirm a bus free status (when BB = 0).
Set the ACK to "1" and specify a slave address and a direction bit to be transmitted to the SBIDBR.
When the BB is "0", the start condition are generated and the slave address and the direction bit which are set to the SBIDBR are output on a bus by writing "1" to the MST, TRX, BB, and PIN. An INTSBI interrupt request occurs at the 9th falling edge of the SCL clock cycle, and the PIN is cleared to "0". The SCL pin is pulled down to the "L" level while the PIN is "0". When an interrupt request occurs, the TRX changes by the hardware according to the direction bit only when an acknowledge signal is returned from the slave device.

Note 1: The slave address to be output to the SBIDBR must be set after the bus free is detected by software. If setting of the slave address is executed before detection bus free, the current output data may be corrupted.
Note 2: The bus free must be confirmed by software within 98.0 μs (the shortest transmitting time according to the I2C bus standard) after setting of the slave address to be output. Only when the bus free is confirmed, set "1" to the MST, TRX, BB, and PIN to generate the start conditions. If the start conditions are generated without writing "1" to them, transferring may be executed by other masters between the time when the slave address to be output to the SBIDBR is written and the time when "1" is written to the MST, TRX, BB, and PIN in the SBICR2. Thus, the slave address may be corrupted.

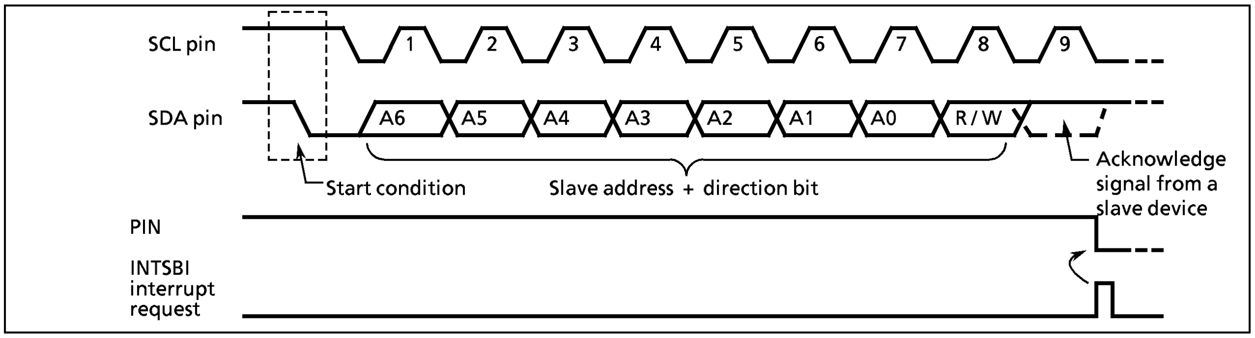


Figure 2-32. Start condition generation and slave address transfer

(3) **1-word Data Transfer**
Check the MST by the INTSBI interrupt process after a 1-word data transfer is completed, and determine whether the mode is a master or slave.

a. When the MST is "1" (Master mode)
Check the TRX and determine whether the mode is a transmitter or receiver.
① When the TRX is "1" (Transmitter mode)
Check the LRB. When the LRB is "1", a receiver does not request data. Implement the process to generate a stop condition (Refer to 2.8.5 (4)) and terminate data transfer.
When the LRB is "0", the receiver requests new data. When the next transmitted data is other than 8 bits, set the BC and write the transmitted data to the SBIDBR. After writing the data, the PIN becomes "1", a serial clock pulse is generated for transferring a new 1-word of data from the SCL pin, and then the 1-word data is transmitted. After the data is transmitted, an INTSBI interrupt request occurs. The PIN becomes "0" and the SCL pin is pulled down to the "L" level. If the data to be transferred is more than one word in length, repeat the procedure from the LRB checking above.

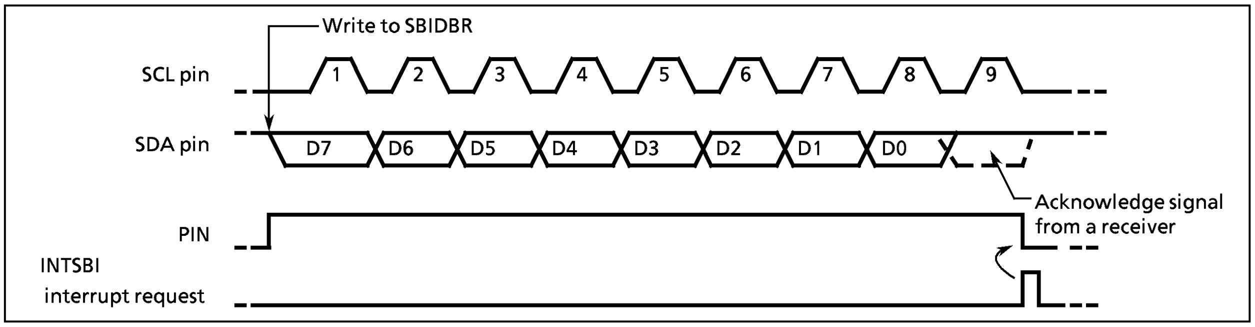


Figure 2-33. Example when BC = "000", ACK = "1" in transmitter mode

② When the TRX is "0" (Receiver mode)
When the next transmitted data is other than 8 bits, set the BC again. Set the ACK to "1" and read the received data from the SBIDBR (data which is read immediately after a slave address is sent is undefined). After the data is read, the PIN becomes "1". The TMP87C409B/809B output a serial clock pulse to the SCL to transfer new 1-word of data and sets the SDA pin to "0" at the acknowledge signal timing.
An INTSBI interrupt request then occurs and the PIN becomes "0". Then the TMP87C409B/809B pull down the SCL pin to the "L" level. The TMP87C409B/809B output a clock pulse for 1-word of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

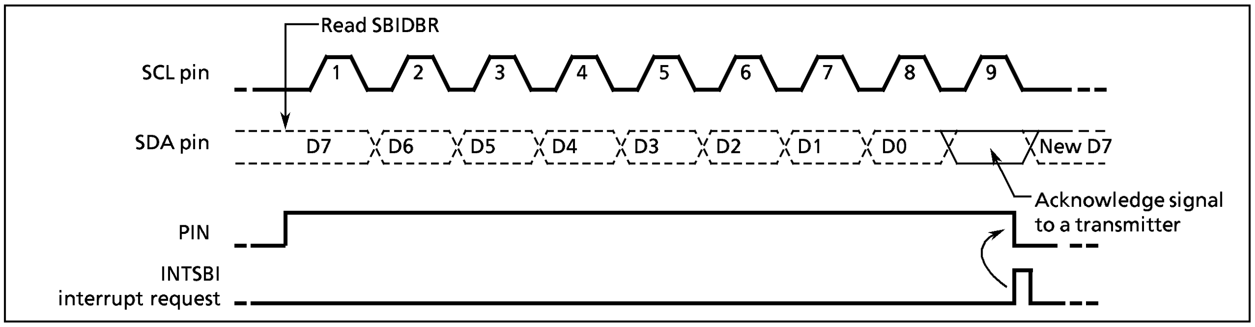


Figure 2-34. Example of when BC = "000", ACK = "1" in receiver mode

In order to terminate transmitting data to a transmitter, clear the ACK to "0" before reading data which is 1 word before the last data to be received. The last data does not generate a clock pulse for the acknowledge signal. After the data is transmitted and an interrupt request has occurred, set the BC to "001" and read the data. The TMP87C409B/809B generate a clock pulse for a 1-bit data transfer. Since the master device is a receiver, the SDA line of the bus keeps the "H" level. The transmitter receives the "H" level signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete.
After 1-bit data is received and an interrupt request has occurred, the TMP87C409B/809B generate a stop condition (Refer to 2.8.5 (4)) and terminates data transfer.

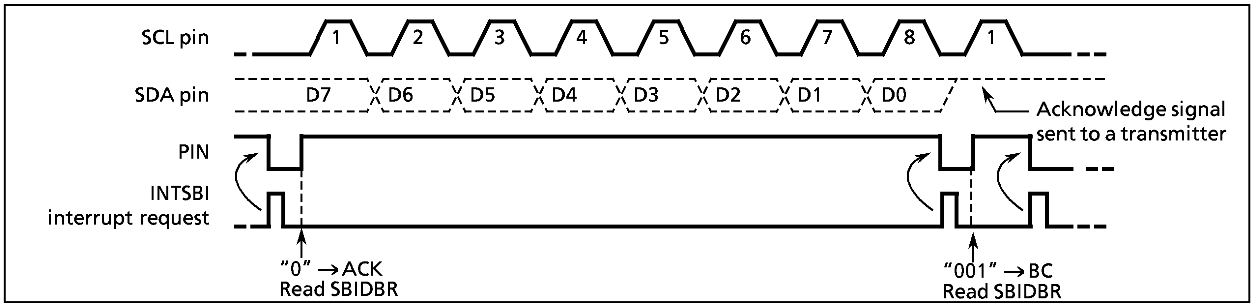


Figure 2-35. Termination of data transfer in master receiver mode

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

In the slave mode, the TMP87C409B/809B operate either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, an INTSBI interrupt request occurs when the TMP87C409B/809B receive a slave address or a GENERAL CALL from the master device, or when a GENERAL CALL is received and data transfer is complete after matching a received slave address. In the master mode, the TMP87C409B/809B operate in a slave mode if it is losing arbitration. An INTSBI interrupt request occurs when word data transfer terminates after losing arbitration. When an INTSBI interrupt request occurs, the PIN (bit 4 in the SBICR2) is reset, and the SCL pin is pulled down to the "L" level. Either reading/writing from/to the SBIDBR or setting the PIN to "1" releases the SCL pin after taking t_{LOW} time.

Check the AL (bit 3 in the SBISR), the TRX (bit 6 in the SBISR), the AAS (bit 2 in the SBISR), and the AD0 (bit 1 in the SBISR) and implements processes according to conditions listed in the next table.

Table 2-7. Operation in the slave mode

TRX	AL	AAS	AD0	Conditions	Process
1	1	1	0	The TMP87C409B/809B loses arbitration when transmitting a slave address and receives a slave address of which the value of the direction bit sent from another master is "1".	Set the number of bits in 1 word to the BC and write transmitted data to the SBIDBR.
	0	1	0	In the slave receiver mode, the TMP87C409B/809B receives a slave address of which the value of the direction bit sent from the master is "1".	
		0	0	In the slave transmitter mode, 1-word data is transmitted.	Check the LRB. If the LRB is set to "1", set the PIN to "1" since the receiver does not request next data. Then, clear the TRX to "0" release the bus. If the LRB is cleared to "0", set the number of bits in a word to the BC and write transmitted data to the SBIDBR since the receiver requests next data.
0	1	1	1/0	The TMP87C409B/809B lose arbitration when transmitting a slave address and receives a slave address or GENERAL CALL of which the value of the direction bit sent from another master is "0".	Read the SBIDBR for setting the PIN to "1" (reading dummy data) or write "1" to the PIN.
		0	0	The TMP87C409B/809B loses arbitration when transmitting a slave address or data and terminates transferring word data.	
	0	1	1/0	In the slave receiver mode, the TMP87C409B/809B receives a slave address or GENERAL CALL of which the value of the direction bit sent from the master is "0".	
		0	1/0	In the slave receiver mode, the TMP87C409B/809B terminates receiving of 1-word data.	Set the number of bits in a word to the BC and read received data from the SBIDBR.

(4) Stop Condition Generation

When BB = "1", writing "1" to the MST, TRX, and PIN and "0" to the BB starts a sequence for outputting a stop condition on the bus. Do not change the contents of the MST, TRX, BB, and PIN until a stop condition is generated on the bus.
When a stop condition is generated and a bus SCL line is set to "L" level by the other devices, a stop condition is not started normally.
Write "1" to the MST, TRX, and PIN, and "0" to the BB to generate a stop condition after releasing the SCL line.

Note: When a stop condition is generated, a time to rise the SCL line should not exceed tr = $2^n/fc - 3.5 \times 4/fc$ (2). (n depends on the SCK.)
If the rising time of the SCL line exceeds the above value, there is a probability that a stop condition is not started normally.

SCK	n	tr (Max., fc = 8 MHz)	tr (Max., fc = 4 MHz)
000	4	0.25 μs	0.50 μs
001	5	2.25 μs	4.50 μs
010	6	6.25 μs	12.5 μs
011	7	14.2 μs	28.5 μs
100	8	30.2 μs	60.5 μs
101	9	62.5 μs	124.5 μs
110	10	126.25 μs	252.5 μs

fc ; High-frequency clock [Hz]

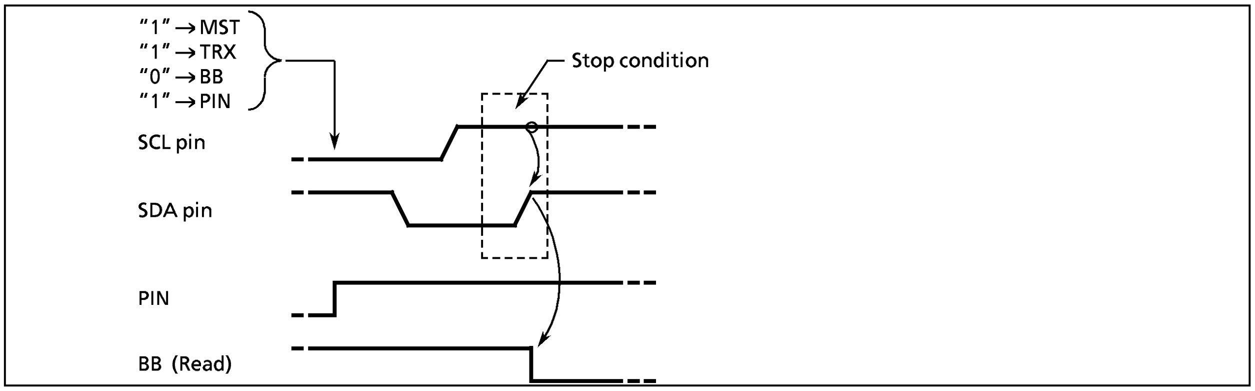


Figure 2-36. Stop condition generation

(5) **Restart**

Restart is used to change the direction of data transfer between a master device and a slave device during transferring data. The following explains how to restart when the TMP87C409B/809B is in the master mode.

Clear "0" to the MST, TRX, and BB and set "1" to the PIN and release the bus. The SDA pin retains the "H" level and the SCL pin is released. Since a stop condition is not generated on a bus, a bus is assumed to be in a busy state from other devices. Check the BB until it becomes "0" to check that the SCL pin of the TMP87C409B/809B is released. Check the LRB until it becomes "1" to check that the SCL line of a bus is not pulled down to the "L" level by other devices. After confirming that a bus stays in a free state, generate a start condition with procedure 2.8.5. (2).

In order to meet setup time when restarting, take at least 4.7 [μs] of waiting time by software from the time of restarting to confirm that the bus is free until the time to generate the start condition.

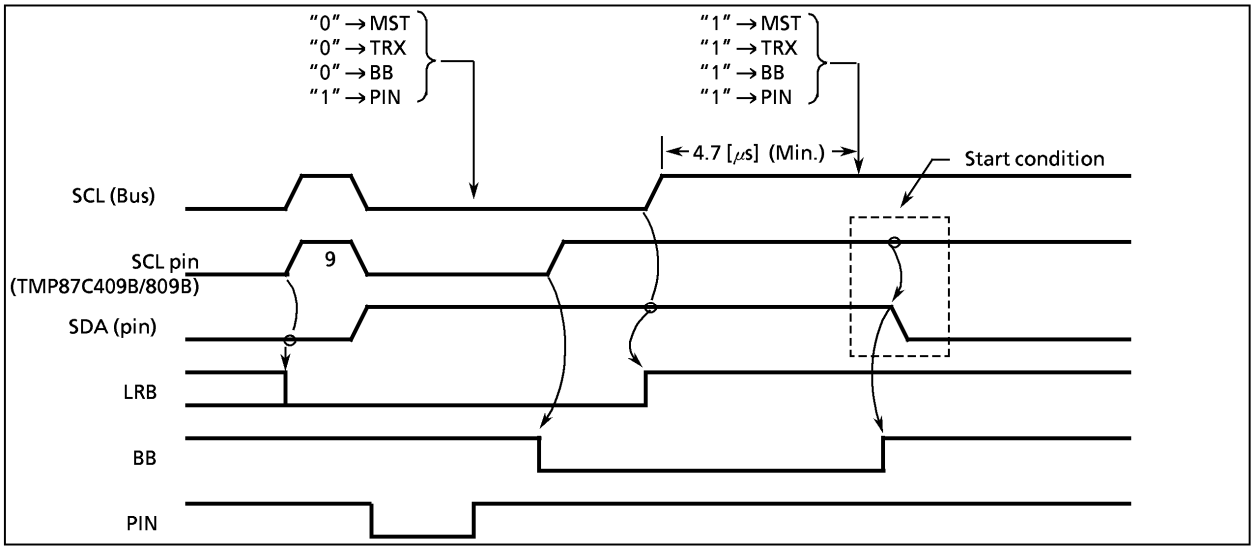


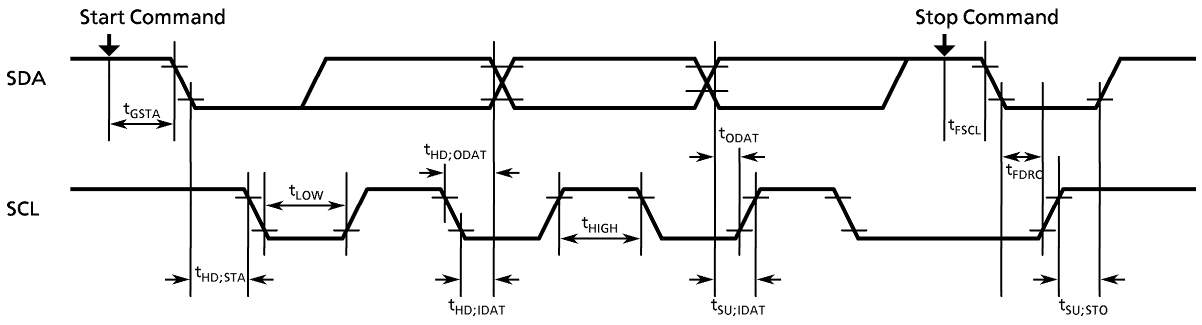
Figure 2-37. Timing diagram when restarting the TMP87C409B/809B

AC Timing for SBI-Ver. B (I^2C bus)

Parameter	Symbol	Min	Typ.	Max	Unit
Hold time (repeated) START condition. After this period, the first clock pulse is generated.	$t_{HD;STA}$	$2^n/fc$	–		s
"H" period of the SCL clock	t_{HIGH}	$2^n/fc + 8/fc$	–	–	s
"L" period of the SCL clock	t_{LOW}	$2^n/fc$	–	–	s
Data hold time (input)	$t_{HD;IDAT}$	0	–	–	ns
Data set-up time (input)	$t_{SU;IDAT}$	250	–	–	ns
Data hold time (output)	$t_{HD;ODAT}$	3/fc	–	7/fc	s
Data output time before rising SCL clock.	t_{ODAT}	–	$2^n/fc - t_{HD;ODAT}$	–	s
Set-up time for STOP condition	$t_{SU;STO}$	$2^n/fc + 4/fc$	–	–	s
The period of generating a start condition when writing START command.	t_{GSTA}	3/fc	–	–	s
The period of falling SCL clock when writing STOP command.	t_{FSCL}	3/fc	–	–	s
The period between falling edge of SDA and rising edge of SCL when generation a STOP condition.	t_{FDRC}	$2^n/fc$	–	–	s

Note: Values those can be applied to "n" in the above table are 4 to 10, and setting values of SCK (bit 2 to 0 of SBICR1) for these values as follows.

SCK (bit 2 to 0 in the SBICR1)	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10



2.8.6 Clocked-synchronous 8-bit SIO mode control

The following registers are used for control and operation status monitoring when using the serial bus interface (SBI) in the clocked-synchronous 8-bit SIO mode.

Serial Bus Interface Control Register 1

SBICR1
(0020_H)

7	6	5	4	3	2	1	0
SIOS	SIO INH	SIOM		"0"	SCK		

(Initial value: 0000 *000)

SIOS	Indicate transfer start/stop	0: Stop 1: Start	Write-only
SIOINH	Continue/abort transfer	0: Continue transfer 1: Abort transfer (automatically cleared after abort)	Write-only
SIOM	Transfer mode select	00: 8-bit transmit mode 01: reserved 10: 8-bit transmit / receive mode 11: 8-bit receive mode	Write-only
SCK	Serial clock select	000: fc/2^5 (250 kHz) 001: fc/2^6 (125 kHz) 010: fc/2^7 (62.5 kHz) 011: fc/2^8 (31.25 kHz) 100: fc/2^9 (15.62 kHz) 101: fc/2^{10} (7.81 kHz) 110: fc/2^{11} (3.90 kHz) (000–110: at fc = 8 MHz; Output on $\overline{\text{SCK}}$ pin) 111: External clock (input from $\overline{\text{SCK}}$ pin)	Write-only

*Note 1: * ; Don't care*
Note 2: Set the SIOS to "0" when setting the transfer mode or serial clock.
Note 3: SBICR1 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.

Serial Bus Interface Data Buffer Register

SBIDBR
(0021_H)

7	6	5	4	3	2	1	0

Read / Write

Note: Cannot read the data which was written into SBIDBR, since a write data buffer and a read buffer are independent in SBIDBR. Therefore, cannot access it any of in read-modify-write instruction such as bit operate, etc.

Serial Bus Interface Control Register 2

SBICR2
(0023_H)

7	6	5	4	3	2	1	0
"0"	"0"	"0"	"1"	SBIM		"0"	"0"

(Initial value: **** 00**)

SBIM	Serial bus interface operation mode selection	00: Port mode (serial bus interface output disable) 01: SIO mode 10: I^2C bus mode 11: reserved	Write-only

*Note 1: * ; Don't care*
Note 2: Switch a mode to port after data transfer is complete.
Note 3: Switch a mode to SIO mode after confirming that input signal via port is "H" level.
Note 4: SBICR2 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.

Serial Bus Interface Status Register

SBISR
(0023_H)

7	6	5	4	3	2	1	0
"1"	"1"	"1"	"1"	SIOF	SEF	"1"	"1"

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	Read-only

Figure 2-38. Control register / data buffer register / status register in SIO mode

(1) Serial clock

a. Clock source

The SCK(bits 2 to 0 in SBICR1) is used to select the following functions.

① Internal clock

In an internal clock mode, any of seven frequencies can be selected. The serial clock is output to the outside on the $\overline{SCK}$ pin. The $\overline{SCK}$ pin becomes a "H" level when data transfer starts. When writing (in the transmit mode) or reading (in the receive mode) data cannot follow the serial clock rate, an automatic-wait function is executed to stop the serial clock automatically and hold the next shift operation until reading or writing is complete.

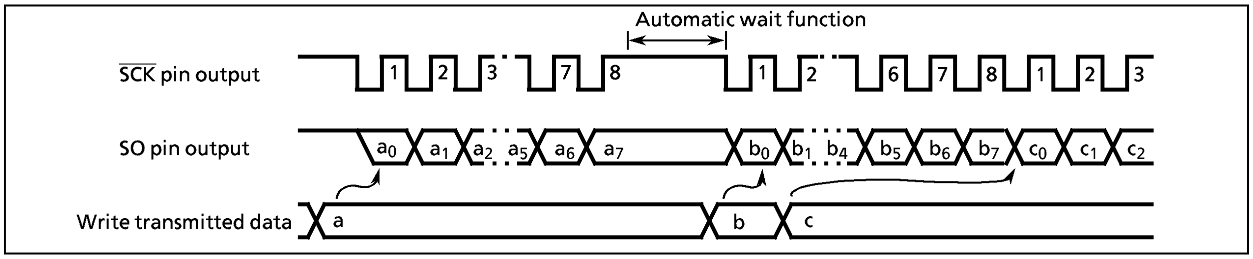

Automatic wait function
$\overline{SCK}$ pin output
SO pin output
Write transmitted data
1 2 3 7 8 1 2 6 7 8 1 2 3
a_0 a_1 a_2 a_5 a_6 a_7 b_0 b_1 b_4 b_5 b_6 b_7 c_0 c_1 c_2
a b c

Figure 2-39. Automatic wait function

② External clock (SCK = "111")

An external clock supplied to the $\overline{SCK}$ pin is used as the serial clock. In order to ensure shift operation, a pulse width of at least 4 machine cycles is required for both high and "L" levels in the serial clock. The maximum data transfer frequency is 250 kHz (fc = 8 MHz).

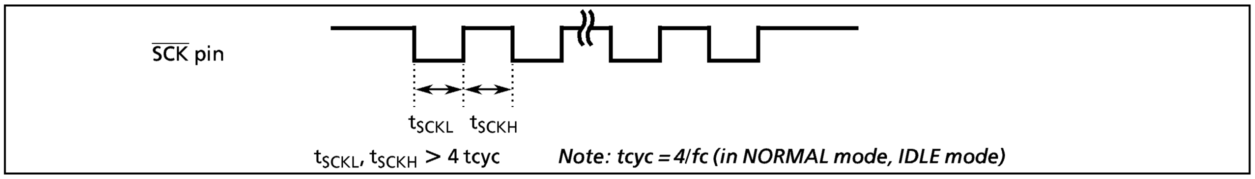

$\overline{SCK}$ pin
t_{SCKL} t_{SCKH}
t_{SCKL}, t_{SCKH} > 4 tcyc
Note: tcyc = 4/fc (in NORMAL mode, IDLE mode)

Figure 2-40. The maximum data transfer frequency in the external clock input

b. Shift edge

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

① Leading edge
Data is shifted on the leading edge of the serial clock (at a falling edge of the $\overline{\text{SCK}}$ pin input/output).

② Trailing edge
Data is shifted on the trailing edge of the serial clock (at a rising edge of the $\overline{\text{SCK}}$ pin input/output).

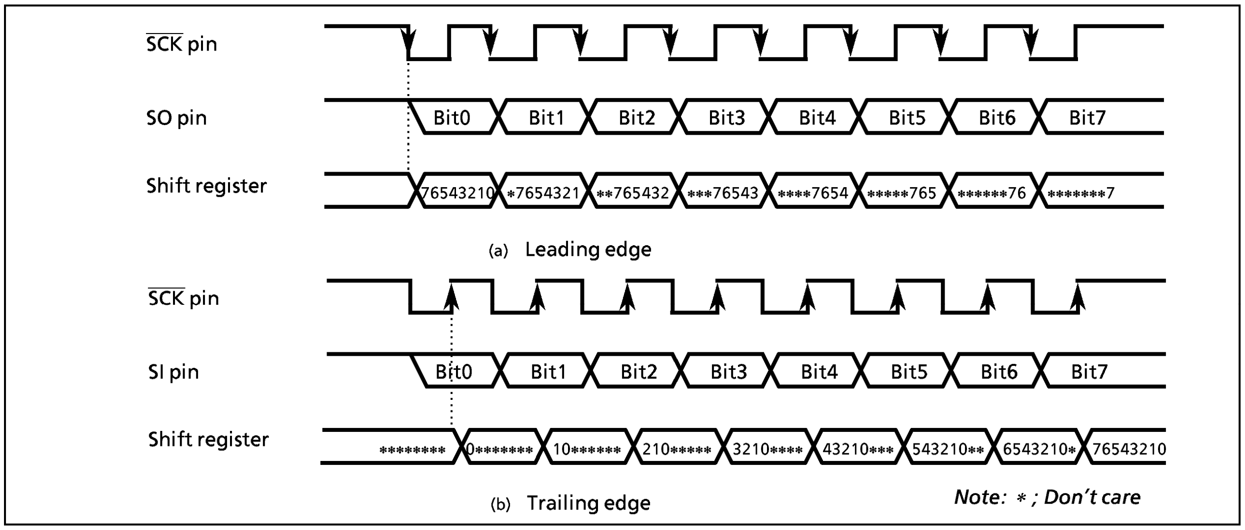


Figure 2-41. Shift edge

(2) Transfer mode

The SIOM (bits 5 and 4 in SBICR) is used to select a transmit, receive, or transmit/receive mode.

a. 8-bit transmit mode

Set a control register to a transmit mode and write data to the SBIDBR.
After the data is written, set the SIOS to "1" to start data transfer. The transmitted data is transferred from the SBIDBR to the shift register and output to the SO pin in synchronous with the serial clock, starting from the least significant bit (LSB). When the data is transferred to the shift register, the SBIDBR becomes empty. The INTSBI (buffer empty) interrupt request is generated to request new data.
When the internal clock is used, the serial clock will stop and automatic-wait function will be initiated if new data is not loaded to the data buffer register after the specified 8-bit data is transmitted. When new data is written, automatic wait function is canceled.
When the external clock is used, data should be written to the SBIDBR before new data is shifted. The transfer speed is determined by the maximum delay time between the time when an interrupt request is generated and the time when data is written to the SBIDBR by the interrupt service program.
When the transmit is started, after the SIOF goes "1" output from the SO pin holds final bit of the last data until falling edge of the $\overline{\text{SCK}}$.
Transmitting data is ended by cleaning SIOS to "0" or setting SIOINH to "1" buffer empty interrupt service program. When the SIOS is cleared, the transmitted mode ends when all data is output. In order to confirm if data is surely transmitted by the program, set the SIOF (bit 3 in the SBISR) to be sensed. The SIOF is cleared to "0" when transmitting is complete. When SIOINH is set, the transmission is immediately ended and SIOF is cleared to "0".

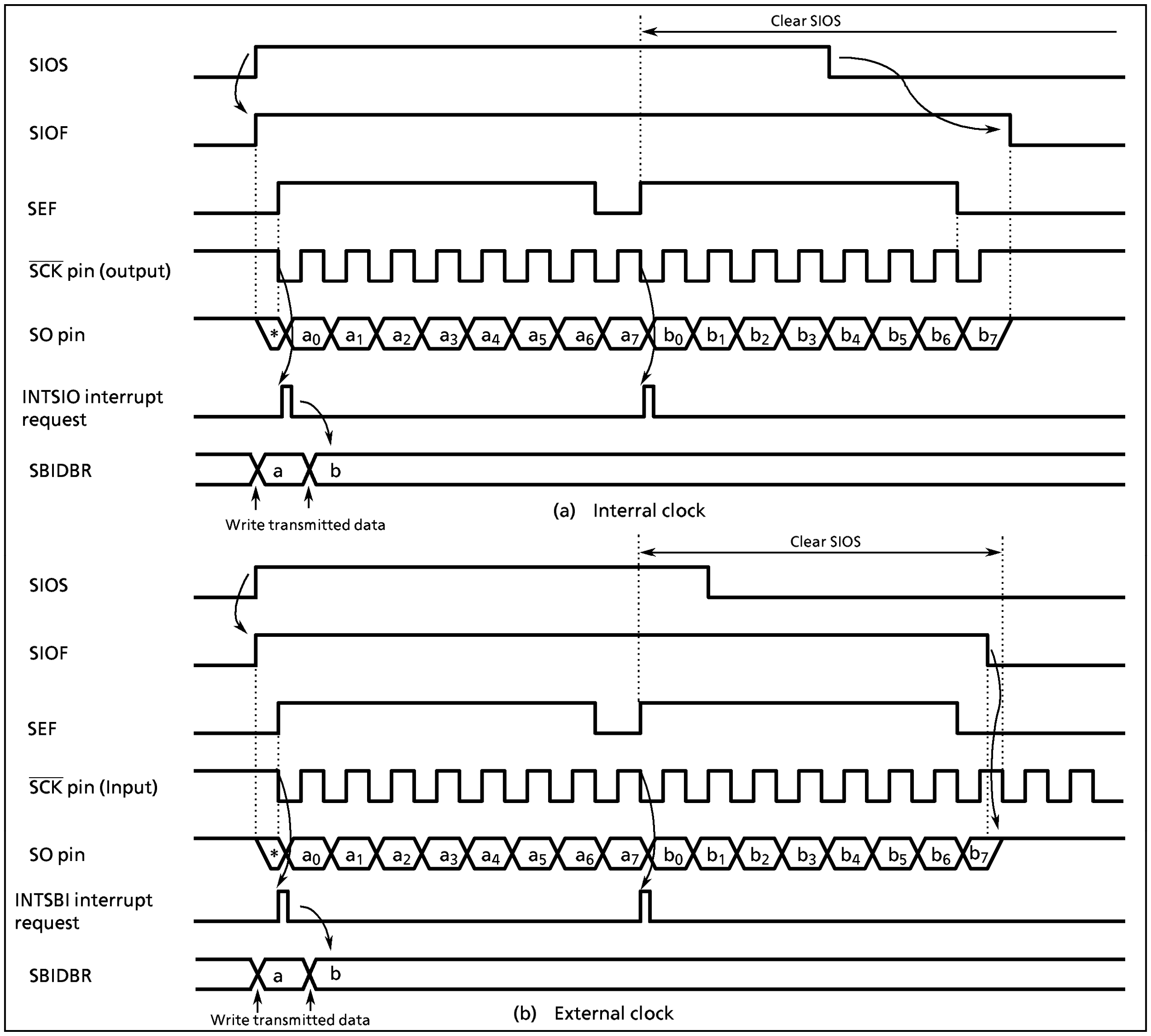


Figure 2-42. Transfer mode

Example: SIO transfer end command (External clock)

```
STEST1: TEST  (SBISR) . SEF        ;  If SEF = 1  then  loop
        JRS   F , STEST1
STEST2: TEST  (P4) . 0             ;  If SCK = 0 then  loop
        JRS   T , STEST2
        LD    (SBICR1) , 00000111B ;  SIOS ← 0
```

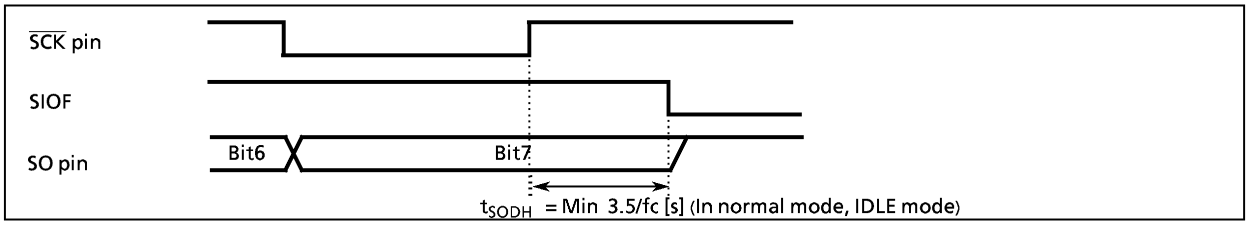


Figure 2-43. Transmitted data hold time at end of transmit

b. 8-bit receive mode

Set a control register to a receive mode and the SIOS to "1" for switching to a receive mode. Data is received from the SI pin to the shift register in synchronous with the serial clock, starting from the least significant bit (LSB). When the 8-bit data is received, the data is transferred from the shift register to the SBIDBR. The INTSBI (buffer full) interrupt request is generated to request of reading the received data. The data is then read from the SBIDBR by the interrupt service program.

When the internal clock is used, the serial clock will stop and automatic-wait function will be initiated until the received data is read from the SBIDBR.

When the external clock is used, since shift operation is synchronized with the clock pulse provided externally, the received data should be read before new data is transferred to the SBIDBR. If the received data is not read, further data to be received is canceled. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when received data is read.

The receiving is ended by clearing SIOS to "0" or setting SIOINH to "1" in buffer full interrupt service program. When SIOINH is set, the receiving is immediately ended and SIOF is cleared to "0". When the SIOS is cleared, received data is transferred to the SBIDBR in complete blocks. The received mode ends when the transfer is complete. In order to confirm if data is surely received by the program, set the SIOF (bit 3 in SBIDBR) to be sensed. The SIOF is cleared to "0" when receiving is complete. After confirming that receiving has ended, the last data is read. When the SIOINH is set, receiving data stops. The SIOF turns "0" (the received data becomes invalid, therefore no need to read it).

Note: When the transfer mode is switched, the SBIDBR contents are lost. In case that the mode needs to be switched, conclude receiving data by clearing the SIOS to "0", read the last data, and then switch the mode.

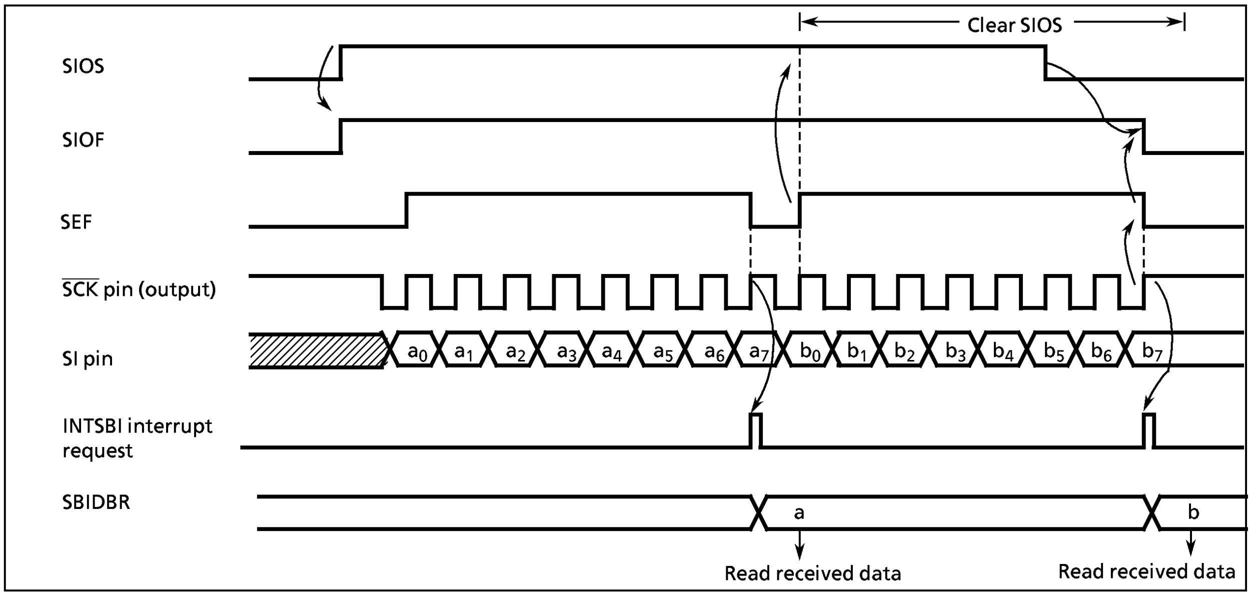


Figure 2-44. Receive mode (Example: Internal clock)

c. 8-bit transmit/receive mode

Set a control register to a transmit/receive mode and write data to the SBIDBR. After the data is written, set the SIOS to "1" to start transmitting/receiving. When transmitting, the data is output from the SO pin on the leading edges in synchronous with the serial clock, starting from the least significant bit (LSB). When receiving, the data is input to the SI pin on the trailing edges of the serial clock. 8-bit data is transferred from the shift register to the SBIDBR, and the INTSBI interrupt request occurs. The interrupt service program reads the received data from the shift register to the SBIDBR, and the INTSBI interrupt request occurs. The interrupt service program reads the received data from the data buffer register and writes data to be transmitted. The SBIDBR is used for both transmitting and receiving. Transmitted data should always be written after received data is read.

When the internal clock is used, automatic-wait function is initiated until received data is read and next data is written.

When the transmit is started, after the SIOF goes "1" output from the SO pin holds final bit of the last data until falling edge of the $\overline{\text{SCK}}$.

When the external clock is used, since the shift operation is synchronized with the external clock, received data is read and transmitted data is written before new shift operation is executed. The maximum transfer speed when the external clock is used is determined by the delay time between the time when an interrupt request is generated and the time when received data is read and transmitted data is written.

Transmitting/receiving data is ended by cleaning the SIOS to "0" or setting SIOINH to "1" in interrupt service program. When the SIOS is cleared, received data is transferred to the SBIDBR in complete blocks. The transmit/receive mode ends when the transfer is completed. In order to confirm if data is surely transmitted/received by the program, set the SIOF (bit 3 in SBISR) to be sensed. The SIOF becomes "0" after transmitting/receiving is complete.

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

*Note: When the transfer mode is switched, the **SBIDBR** contents are lost. In case that the mode needs to be switched, conclude transmitting/receiving data by clearing the **SIOS** to '0", read the last data, and then switch the transfer mode.*

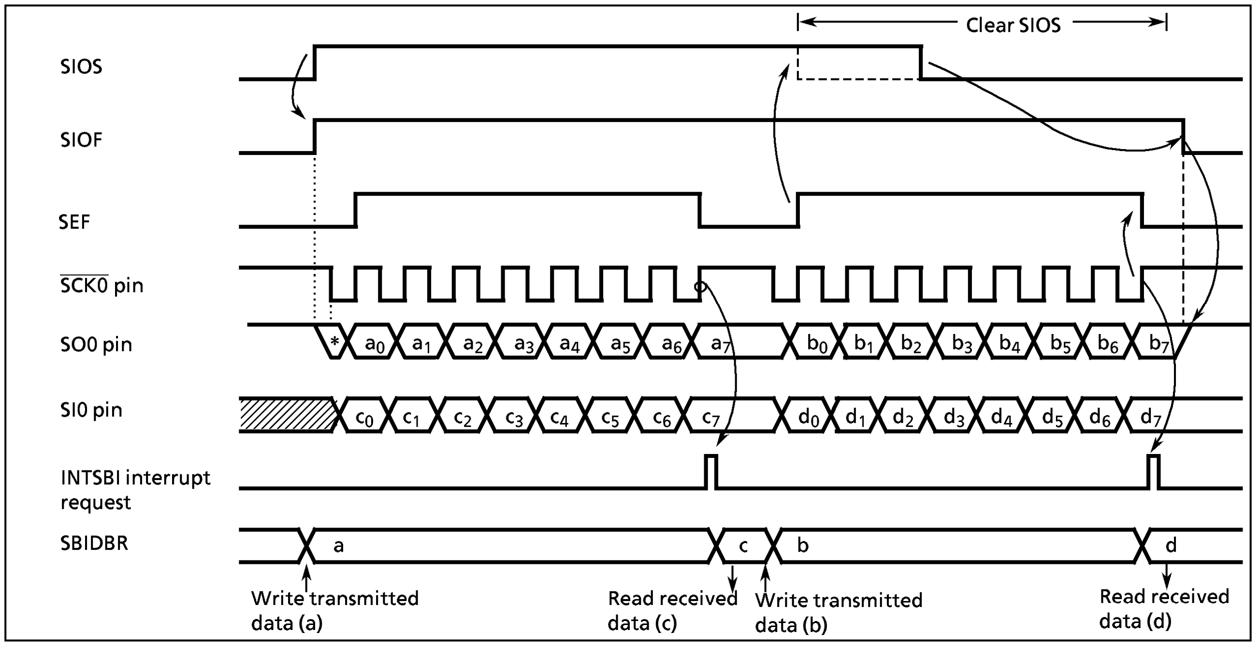

Clear SIOS
SIOS
SIOF
SEF
$\overline{\text{SCK0}}$ pin
SO0 pin
SI0 pin
INTSBI interrupt request
SBIDBR
* a0 a1 a2 a3 a4 a5 a6 a7 b0 b1 b2 b3 b4 b5 b6 b7
c0 c1 c2 c3 c4 c5 c6 c7 d0 d1 d2 d3 d4 d5 d6 d7
a c b d
Write transmitted data (a)
Read received data (c)
Write transmitted data (b)
Read received data (d)

Figure 2-45. Transmit/receive mode (Example: Internal clock)

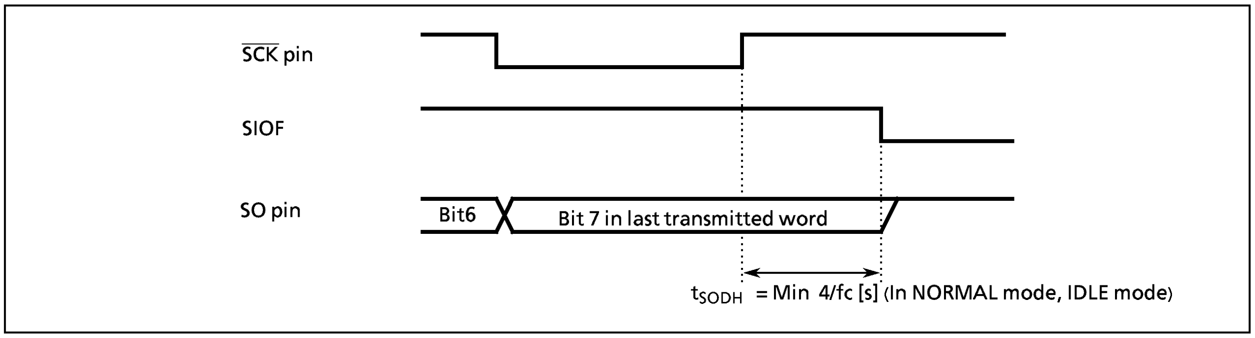

$\overline{SCK}$ pin
SIOF
SO pin
Bit6
Bit 7 in last transmitted word
t_{SODH} = Min 4/fc [s] (In NORMAL mode, IDLE mode)

Figure 2-46. Transmitted data hold time at end of transmit/receive

2.9 Oscillation Stop Detector

The TMP87C409B/809B each have a Oscillation Stop Detector.

If the oscillation stops for any cause, P51 (CLZ1) or P50 (CLZ0) becomes high-impedance.

Note: On the emulator, when Oscillation Stop Detector is enabled, P50 or P51 may become high-impedance by break instruction or single step instruction.
Release of break instruction or single step instruction returns the port in condition before high-impedance.

2.9.1 Configuration

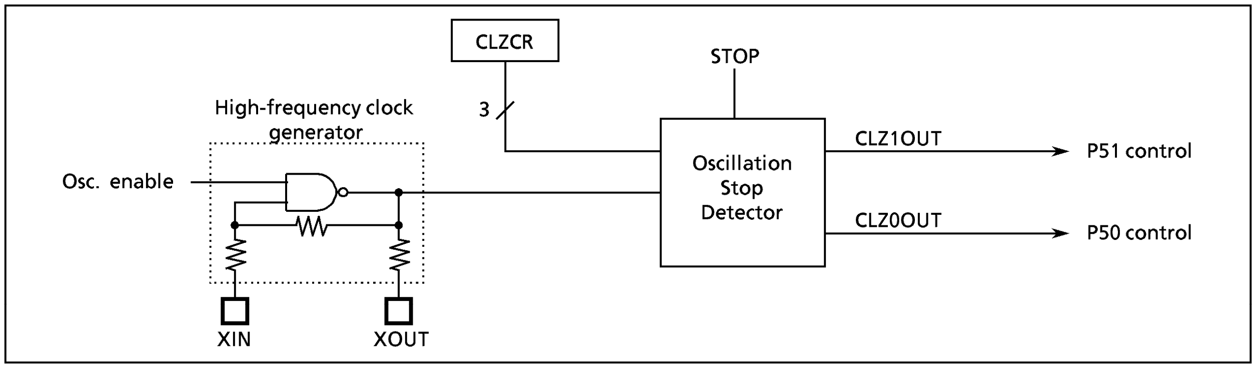


Figure 2-47. Oscillation stop detector

2.9.2 Control

The Oscillation Stop Detector is controlled by register (CLZCR).

Oscillation Stop Detector

CLZCR (0028_H)

7	6	5	4	3	2	1	0
					STOPCR	P51CR	P50CR

(Initial value: **** *000)

STOPCR	Detection output control in STOP mode	0: Enable in STOP mode 1: Disable in STOP mode	Write-only
P51CR	P51 detection output control	0: P51 detection enable 1: P51 detection disable	
P50CR	P50 detection output control	0: P50 detection enable 1: P50 detection disable	

*Note 1: * ; Don't care*
Note 2: CLZCR is write-only register, therefore, it cannot be operated with read modify commands (SET, CLR and other bit operation commands: AND, OR and other multiplication commands).

Figure 2-48. Oscillation stop detector control register

2.9.3 Function

If the oscillation stops for any cause, P51 (CLZ1) or P50 (CLZ0) becomes high-impedance.
When P51 or P50 used as a oscillation stop detector output, the output pins should be set to output mode beforehand the output latch should be set to "1" or "0".
P51/P50 output is selected by P51CR and P50CR (bit 1 and 0 in CLZCR). In STOP mode, the oscillation stop detection can be disabled be setting STOPCR to "1" (bit 2 in CLZCR).

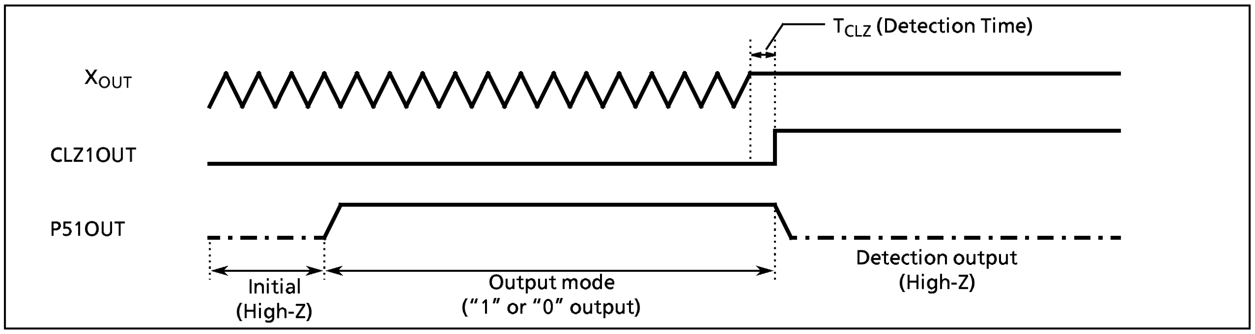

T_{CLZ} (Detection Time)
X_{OUT}
CLZ1OUT
P51OUT
Initial (High-Z)
Output mode ("1" or "0" output)
Detection output (High-Z)

Figure 2-49. Oscillation stop detector timing chart

2.10 10-bit AD Converter (ADC)

The TMP87C409B/809B have a 10-bit successive approximate type AD converter.
Analog reference power supply (VAREF) is automatically cut off in stop mode or analog input disable.

2.10.1 Configuration

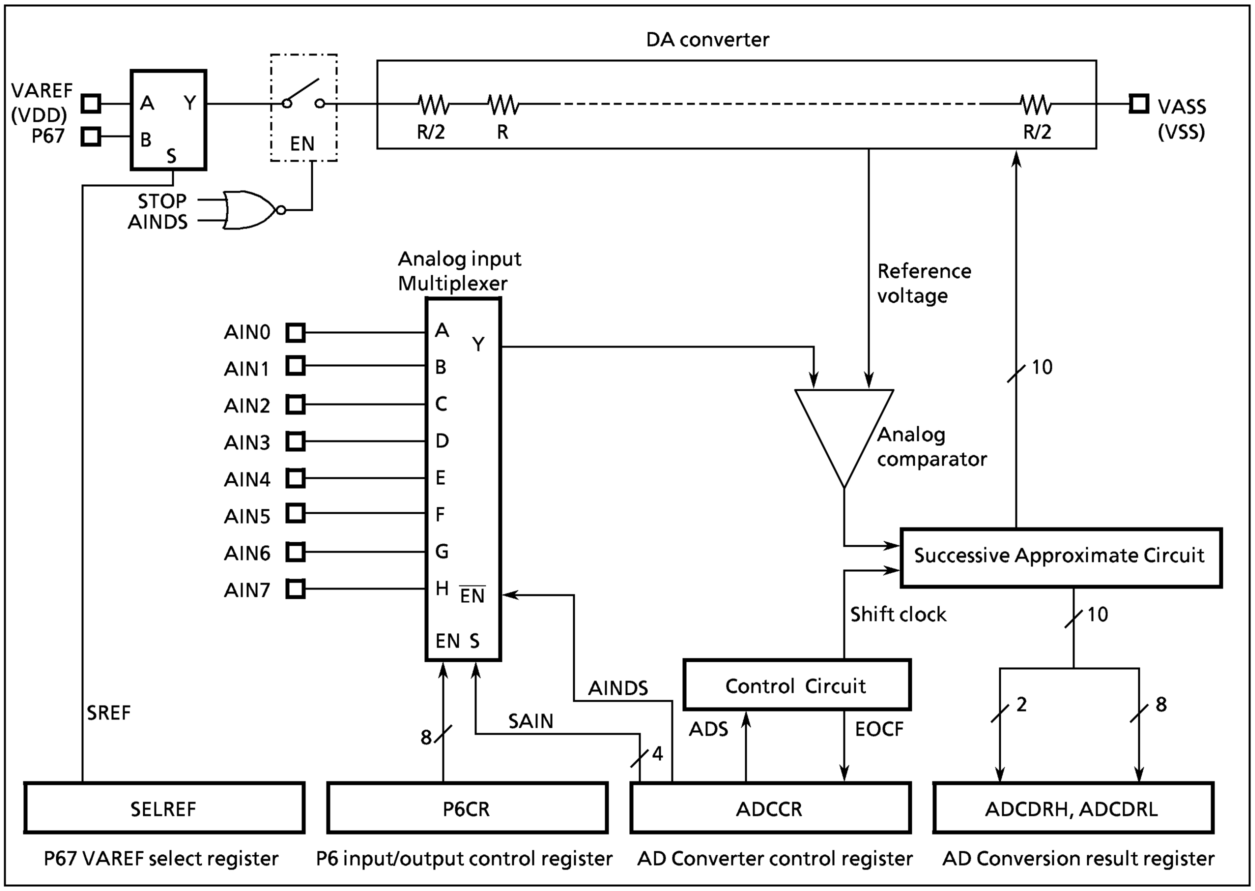


Figure 2-51. AD Converter

2.10.2 Control

The AD converter is controlled by an AD converter control register (ADCCR) and an analog reference power supply select register (SELREF).

Reading EOCF of ADCCR recognizes AD converter operation state, and reading AD conversion result registers (ADCDRH, ADCDRL) recognize AD conversion result.

AD Converter Control Register

ADCCR ($000E_H$)

7	6	5	4	3	2	1	0
EOCF	ADS	ACK		AINDS	SAIN		

(Initial value: 0000 1000)

EOCF	AD conversion end flag	Set to "1" at the end of AD conversion. Cleared when EOCF and $ADCDR_H$ are read successively. 0: Under conversion or Before conversion 1: AD End of conversion	Read-only
ADS	AD conversion start	Automatically cleared to "0" after AD conversion has started. Setting ADS to "1" during AD conversion initializes and newly starts AD conversion. 0: – 1: AD conversion start	R/W
ACK	Conversion time	00: 216/fc: (27 μs at fc = 8 MHz) 01: 384/fc: (48 μs at fc = 8 MHz) 10: 728/fc: (91 μs at fc = 8 MHz) 11: Reserved	
AINDS	Analog input control	0: Enable 1: Disable	
SAIN	Analog input selection	000: AIN0 001: AIN1 010: AIN2 011: AIN3 100: AIN4 101: AIN5 110: AIN6 111: AIN7	

Note 1: Select analog input when AD converter stops.

Note 2: The ADS is automatically cleared to "0" after starting conversion.

Note 3: The EOCF is cleared to "0" when reading the ADCDRL.

Note 4: The EOCF is read-only. The written data is ignored.

VAREF Select Register

SELREF ($002F_H$)

7	6	5	4	3	2	1	0
SREF							

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

SREF	0: Internal VDD 1: External level (P67)	Write-only

*Note 1: * ; Don't care*

Note 2: When used as an analog reference power supply, P67 should be set to the input mode.

Figure 2-51. AD Converter control register and analog reference select register

AD Converter Result Register

ADCDRL (0029_H)

7	6	5	4	3	2	1	0
AD09	AD08	AD07	AD06	AD05	AD04	AD03	AD02

Read-only

ADCDRH ($002A_H$)

7	6	5	4	3	2	1	0
AD01	AD00						

Read-only

Figure 2-52. AD converter result register

2.10.3 Operation

The high side of an analog reference voltage is applied to VAREF pin, and the low side is applied to VASS pin. VAREF can be selected either VDD or P67 by SREF (bit 7 in SELREF). The reference voltage between VAREF and VASS is divided into the voltage corresponding with bits by radar resistance. The reference voltage is compared with an analog input voltage and AD conversion is performed.

Note 1: VASS is the same as VSS.
Note 2: In the TMP87C409B/809B, the analog power supplies (VAREF, VASS) for the AD converter are shared with the digital power supplies (VDD, VSS). For applications that require high accuracy for the AD converter, consider noise reduction on the power supply lines by, for example, lowering the impedance of or inserting a noise filter capacitor into the power supply line.

(1) Start of AD conversion

First, selects one of analog input channels (AIN7 to AIN0) by SAIN(bit 3 to 0 in ADCCR). Clear the AINDS (bit 4 in ADCCR) to "0". The channel used as an analog input is cleared to "0" by P6 input control (P6CR).

Note: The pin that is not used as an analog input can be used as regular input/output pins. During conversion, do not perform output instruction to maintain a precision for all of the pins.

The AD conversion time is programmed into the ACK field (bits 5 and 4) in the ADCCR.
AD conversion is started by setting the ADS (bit 6 in ADCCR) to "1".
AD conversion time is from AD conversion until setting the conversion result to ADCDR. When ACK = 00, conversion is accomplished in 216/fc [s] (54 machine cycles). For example, 27 μs in fc = 8 MHz. The EOCF (bit 7 in ADCCR) is set to "1" at end of conversion.When ADS is set to "1" during AD conversion, conversion is initialized and restarted.Sampling of an analog input voltage is performed in 4 machine cycles after AD conversion start is indicated.

(2) Reading of AD conversion result

After the end of conversion (EOCF = 1), read the conversion result from the ADCDR. The EOCF is automatically cleared to "0" when reading the ADCDR. When the conversion result is read out during AD conversion, the invalid value is read out.

(3) AD conversion in STOP mode

When the MCU places in the STOP mode during the AD conversion, the conversion is terminated and the AD conversion value become indefinite. Thus EOCF is maintained to "0" after returned from the STOP mode. However, if the STOP mode is started after the end of AD conversion (EOCF = "1"), the AD conversion value and EOCF state are held.

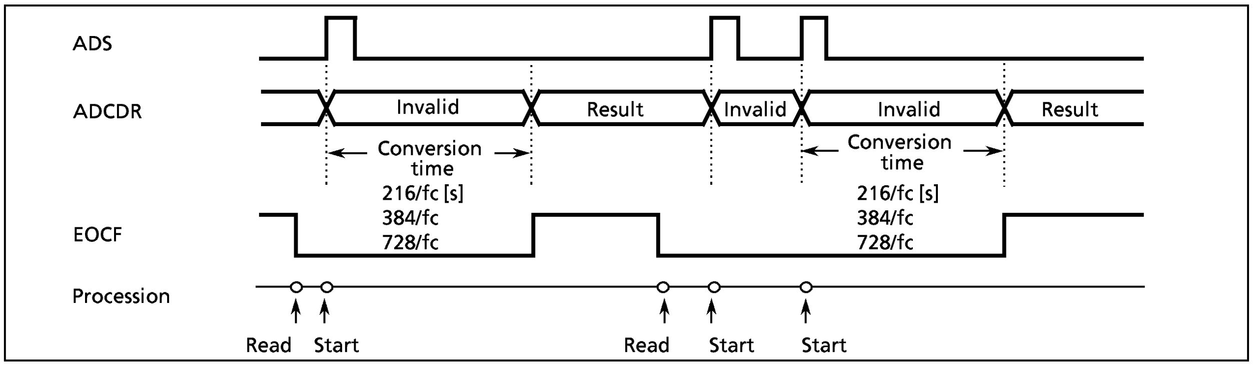


Figure 2-53. AD Conversion Timing chart

Example: After AIN pin 4 is selected as an analog input channel, and conversion time is selected 728/fc [s], AD conversion is started. EOCF is confirmed and the converted result is read out. It is saved to addresses $009E_H$ – $009F_H$ in RAM.

```
         ; AIN SELECT
         LD      (ADCCR) , 00100100B    ;  Selects AIN4, conversion time is 728/fc [s]
         ; AD CONVERT START
         SET     (ADCCR) . 6            ;  ADS = 1
SLOOP:   TEST    (ADCCR) . 7            ;  EOCF = 1 ?
         JRS     T, SLOOP
         ; RESULT DATA READ
         LD      (9EH), (ADCDRH)
         LD      (9FH), (ADCDRL)
```

Figure 2-54 shows the relationship between An analog input voltage and AD converted 10-bit digital value.

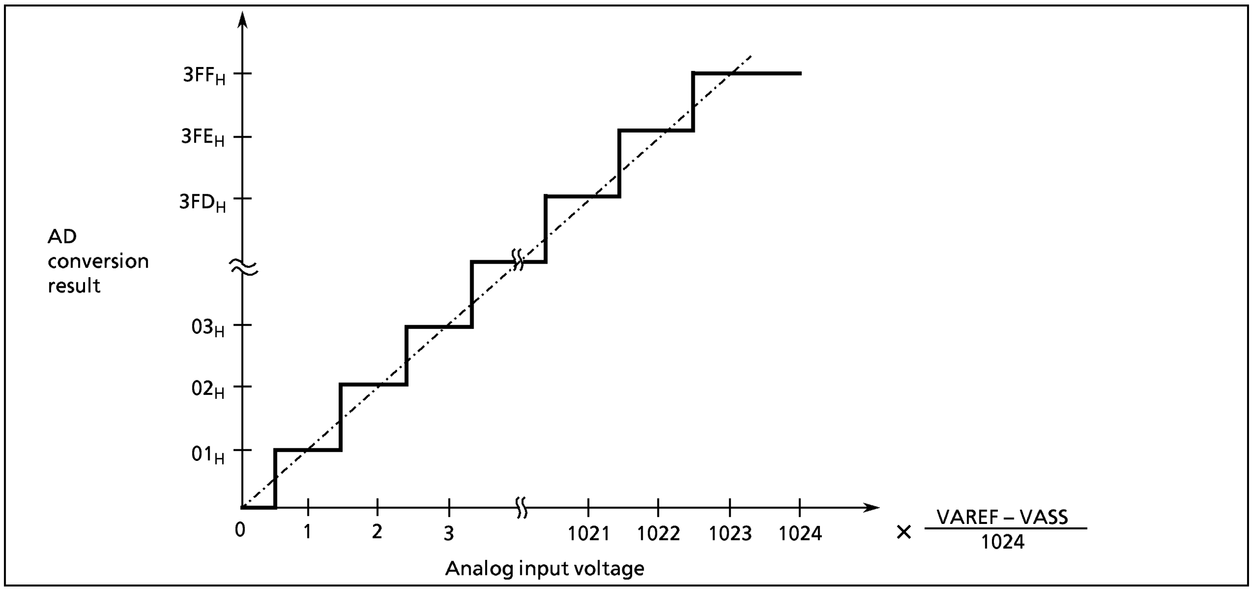


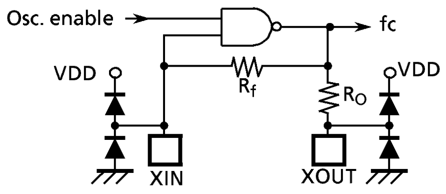
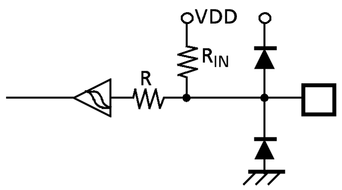
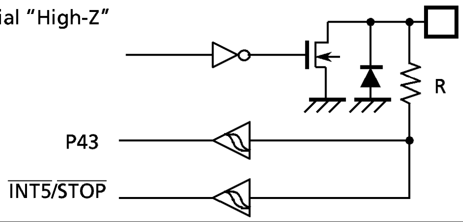
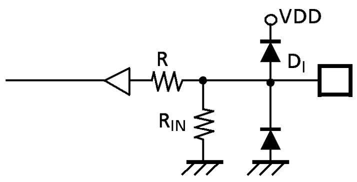
Figure 2-54. Analog input voltage vs. AD conversion result (typ.)

Input/Output Circuitry

When ordering ES from Toshiba, always be sure to specify mask options using a request sheet for the production of Microcontroller Engineering Sample (ES). For details on how to write this specification, refer to Appendix, "Method for Specifying Mask Options in TLCS-870 Series."

(1) Control pins

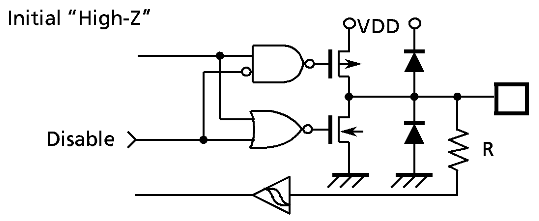
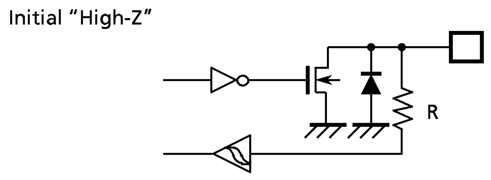
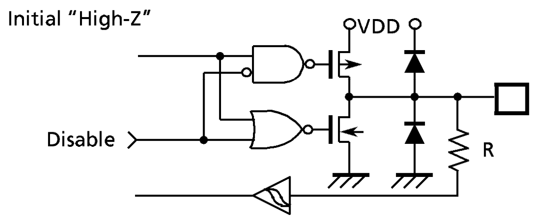
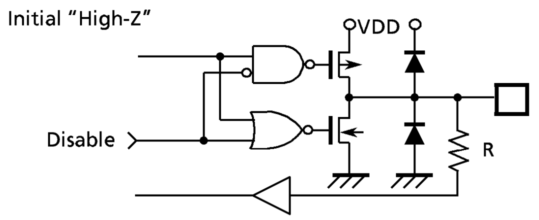
The input/output circuitries of the TMP87C409B/809B control pins are shown below.

Control Pin	I/O	Input/Output Circuitry	Remarks
XIN XOUT	Input Output	 Osc. enable fc VDD R_f R_O VDD XIN XOUT	Resonator connecting pins (high-frequency) R_f = 1.2 MΩ (typ.) R_O = 1.5 kΩ (typ.)
$\overline{\text{RESET}}$	Input		Hysteresis input Pull-up resistor R_{IN} = 220 kΩ (typ.) R = 1 kΩ (typ.)
$\overline{\text{INT5}}$ / $\overline{\text{STOP}}$ (P43)	Input	initial "High-Z"  R P43 $\overline{\text{INT5}}$/$\overline{\text{STOP}}$	Hysteresis input R = 1 kΩ (typ.)
TEST	Input		Pull-down resistor R_{IN} = 70 kΩ R = 1 kΩ

Note: The TMP87P809 does not have a pull-down resistor (R_{IN}) and diode (D_I) for TEST pin. Be sure to fix the TEST pin to low in MCU mode.

(2) Input/Output Ports

The input/output circuitries of the TMP87C409B/809B input/output ports are shown below.
A mask option code is only "A".

Control Pin	I/O	Input/Output Circuitry (CODE A)	Remarks
P1	I/O		Tri-state I/O Hysteresis inut R = 1 kΩ (typ.)
P4	I/O		Sink open drain output Hysteresis input High current output R = 1 kΩ (typ.)
P5	I/O		Tri-state I/O Hysteresis input High current output R = 1 kΩ (typ.)
P6	I/O		Tri-state I/O R = 1 kΩ (typ.)

Electrical Characteristics

Absolute Maximum Ratings

(V_{SS} = 0 V)

Parameter		Symbol	Condition		Ratings	Unit
Supply Voltage		V_{DD}			– 0.3 to 6.5	V
Input Voltage		V_{IN}			– 0.3 to V_{DD} + 0.3	V
Output Voltage		V_{OUT1}	Ports P1, P5, P6, XOUT		– 0.3 to V_{DD} + 0.3	
		V_{OUT2}	Port P4		– 0.3 to 5.5	V
Output Current (Per 1 pin)	IOL	I_{OUT1}	Ports P1, P6		3.2	mA
		I_{OUT2}	Ports P4, P5		30	
	IOH	I_{OUT3}	Ports P1, P5, P6		– 1.8	
Output Current (Total)	IOL	ΣI_{OUT1}	Ports P1, P6		30	mA
		ΣI_{OUT2}	Ports P4, P5		80	
	IOH	ΣI_{OUT3}	Ports P1, P5, P6		30	
Power Dissipation [Topr = 70°C]		PD		SDIP	300	mW
				SOP	180	
Soldering Temperature (time)		Tsld			260 (10 s)	°C
Storage Temperature		Tstg			– 55 to 125	°C
Operating Temperature		Topr			– 30 to 70	°C

Note: The absolute maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded.

Recommended Operating Conditions

(V_{SS} = 0 V, Topr = – 30 to 70°C)

Parameter	Symbol	Pins	Conditions		Min	Max	Unit
Supply Voltage	V_{DD}		fc = 8 MHz	NORMAL mode	4.5	5.5	V
				IDLE mode			
			fc = 4.2 MHz	NORMAL mode	2.2		
				IDLE mode			
				STOP mode	2.0		
Input High Voltage	V_{IH1}	Except hysteresis input	$V_{DD} \geqq 4.5$ V		$V_{DD} \times 0.70$	V_{DD}	V
	V_{IH2}	Hysteresis input			$V_{DD} \times 0.75$		
	V_{IH3}		$V_{DD} < 4.5$ V		$V_{DD} \times 0.90$		
Input Low Voltage	V_{IL1}	Except hysteresis input	$V_{DD} \geqq 4.5$ V		0	$V_{DD} \times 0.30$	V
	V_{IL2}	Hysteresis input				$V_{DD} \times 0.25$	
	V_{IL3}		$V_{DD} < 4.5$ V			$V_{DD} \times 0.10$	
Clock Frequency	fc	XIN, XOUT	VDD = 4.5 to 5.5 V		1.0	8.0	MHz
			V_{DD} = 2.2 V to 5.5 V			4.2	

Note1: 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.
Note2: Clock frequency fc: Supply voltage range is specified in NORMAL mode and IDLE mode.

DC Characteristics

(V_{SS} = 0 V, Topr = – 30 to 70°C)

Parameter	Symbol	Pins	Conditions	Min	Typ.	Max	Unit
Hysteresis Voltage	V_{HS}	Hysteresis input		–	0.9	–	V
Input Current	I_{IN1}	TEST	V_{DD} = 5.5 V V_{IN} = 5.5 V / 0 V	–	–	± 2	μA
	I_{IN2}	Tri-state ports					
	I_{IN3}	$\overline{\text{RESET}}$, $\overline{\text{STOP}}$					
Input Resistance	R_{IN2}	$\overline{\text{RESET}}$		100	220	450	kΩ
Output Leakage Current	I_{LO}	Tri-state ports	V_{DD} = 5.5 V, V_{OUT} = 5.5 V / 0 V	– 2	–	2	μA
Output High Voltage	V_{OH2}	Tri-state ports	V_{DD} = 4.5 V, I_{OH} = – 0.7 mA	4.1	–	–	V
Output Low Voltage	V_{OL1}	Except XOUT, P4 and P5	V_{DD} = 4.5 V, I_{OL} = 1.6 mA	–	–	0.4	
Output Low current	I_{OL3}	P4, P5	V_{DD} = 4.5 V, V_{OL} = 1.0 V	–	20	–	mA
Supply Current in NORMAL mode	I_{DD}		V_{DD} = 5.5 V fc = 8 MHz V_{IN} = 5.3 V / 0.2 V		8	14	mA
Supply Current in IDLE mode					4	6	
Supply Current in NORAML mode			V_{DD} = 3.0 V fc = 4.2 MHz V_{IN} = 2.8V / 0.2 V		2.5	3.5	mA
Supply Current in IDLE mode					1.5	2.0	
Supply Current in STOP mode			V_{DD} = 5.5 V V_{IN} = 5.3 V / 0.2 V		0.5	10	μA

Note 1: Typical values show those at Topr = 25°C, V_{DD} = 5 V.
Note 2: Input Current IIN1, IIN3,: The current through resistor is not included, when the input resistor (pull-up or pull-down) is contained.

AD Conversion Characteristics

(V_{SS} = 0 V, V_{DD} = 2.2 to 5.5 V, Topr = – 30 to 70°C)

Parameter	Symbol	Conditions	Min	Typ.	Max	Unit
Analog Reference Voltage	V_{AREF}		2.2	–	V_{DD}	V
	V_{ASS}		V_{SS}			
Analog Input Voltage range	V_{AIN}		V_{ASS}	–	V_{AREF}	V
Analog Reference Current	I_{REF}	V_{AREF} = 5.5 V, V_{ASS} (V_{SS}) = 0.0 V	–	0.5	1.0	mA
Nonlinearity Error		V_{DD} = 5.0 V V_{AREF} = 5.000 V V_{ASS} (V_{SS}) = 0.000 V or V_{DD} = 2.2 V V_{AREF} = 2.200 V V_{ASS} (V_{SS}) = 0.000 V	–	–	± 2	LSB
Zero Point Error			–	–	± 2	
Full Scale Error			–	–	± 2	
Total Error			–	–	± 4	

Note: Quantizing error is not contained in those errors.

Oscillation Stop Detector Characteristics

(V_{SS} = 0V, Topr = − 30 to 70°C)

Parameter	Symbol	Conditions	Min	Typ.	Max	Unit
Detection time	T_{CLZ}	VDD = 2.2 V to 5.5 V (fc = 2 MHz to 4.2 MHz) VDD = 4.5 V to 5.5 (fc = 8 MHz)	2	20	400	μs

AC Characteristics

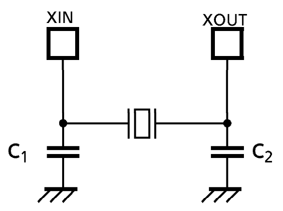
(V_{SS} = 0 V, V_{DD} = 4.5 to 5.5 V, Topr = − 30 to 70°C)

Parameter	Symbol	Conditions	Min	Typ.	Max	Unit
Machine Cycle Time	tcy	In NORMAL mode	0.5	–	4	μs
		In IDLE mode				
High Level Clock Pulse Width	t_{WCH}	For external clock operation	50	–	–	ns
Low Level Clock Pulse Width	t_{WCL}	fc = 8 MHz				

Recommended Oscillating Conditions

(V_{SS} = 0 V, V_{DD} = 2.2 to 5.5 V, Topr = − 30 to 70°C)

Parameter	Oscillator	Oscillation Frequency	Recommended Oscillator
High-frequency Oscillation	Ceramic Resonator	8 MHz (4.5 V to 5.5 V)	MURATA CSTCC8M00G53-R0
			MURATA CSTLS8M00G53-B0
		4 MHz (2.2 V to 5.5 V)	MURATA CSTCR4M00G53-R0
			MURATA CSTLS4M00G53-B0



(1) High-frequency Oscillation

Note 1: When used in high electric field such as a picture tube, the package is recommended to be electrically shielded to maintain a regular operation.

Note 2: The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following URL;http://www.murata.co.jp/search/index.html

Point of note about solderability of lead free products (attach “G” to package name)

Test parameter	Test condition	Note
Solderability	Use of Sn-63Pb solder Bath Solder bath temperature = 230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	Use of Sn-3.0Ag-0.5Cu solder bath Solder bath temperature =245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux (use of lead free)	