CMOS 8-BIT MICROCONTROLLER

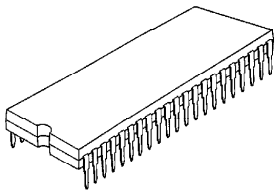
TMP87CK43N, TMP87CM43N

The 87CK43/M43 is the high speed and high performance 8-bit single chip microcomputer. This MCU contains CPU core, ROM, RAM, input/output ports, an A/D converter, six multi-function timer/counters, serial bus interface, and two clock generators on a chip.

PART No.	ROM	RAM	PACKAGE	OTP MCU
TMP87CK43N	24K bytes	1K bytes	SDIP42-P-600-1.78	TMP87PM43N
TMP87CM43N	32K bytes			

FEATURES

SDIP42-P-600-1.78



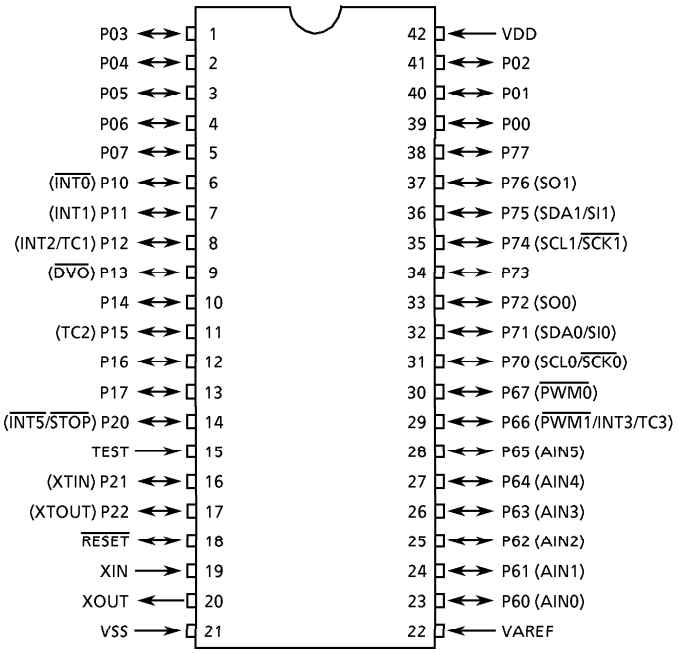
TMP87CK43N
TMP87CM43N
TMP87PM43N

- ◆8-bit single chip microcomputer TLCS-870 Series
- ◆Instruction execution time : 0.5 μs (at 8 MHz), 122 μs (at 32.768kHz)
- ◆412 basic instructions
  - Multiplication and Division (8bits × 8bits , 16bits ÷ 8bits)
  - Bit manipulations (Set/Clear/Complement/Load/Store/Test/Exclusive or)
  - 16-bit data operations
  - 1-byte jump/subroutine-call (Short relative jump / Vector call)
- ◆13 interrupt sources (External : 5, Internal : 8)
  - All sources have independent latches each, and nested interrupt control is available.
  - 3 edge-selectable external interrupts with noise reject
  - High-speed task switching by register bank changeover
- ◆5 Input/Output ports (35 pins)
- ◆Two 16-bit Timer/Counters
  - Timer, Event counter modes
- ◆Two 8-bit Timer/Counters
  - Timer, Event counter, Capture (Pulse width/duty measurement) modes
- ◆Time Base Timer (Interrupt frequency : 1 Hz to 16 kHz)
- ◆Divider output function (frequency : 1 kHz to 8 kHz)
- ◆Watchdog Timer
  - Interrupt source/reset output (programmable)
- ◆**Serial Bus Interface**
  - I²C-Bus / 8-bit SIO modes
  - Selectable two I/O channels
- ◆**8-bit** successive approximate type **A/D converter** with sample and hold
  - 6 analog inputs
  - Conversion time : 23 μs at 8MHz
- ◆PWM outputs
  - 14-bit PWM output (1 channel)
  - 7-bit PWM output (1 channel)
- ◆Dual clock operation
  - Single/Dual-clock mode (option)
- ◆Five Power saving operating modes
  - STOP mode : Oscillation stops. Battery/Capacitor back-up. Port output hold/High-impedance.
  - SLOW mode : Low power consumption operation using low-frequency clock (32.768 kHz).
  - IDLE1 mode : CPU stops, and Peripherals operate using high-frequency clock. Release by interrupts.
  - IDLE2 mode : CPU stops, and Peripherals operate using high- and low-frequency clocks. Release by interrupts.
  - SLEEP mode : CPU stops, and Peripherals operate using low-frequency clock. Release by interrupts.
- ◆Wide operating voltage : 4.5 to 5.5 V at 8 MHz, 2.7 to 5.5 V at 4.19 MHz/32.768 kHz
- ◆Emulation Pod : BM87CM43N0A

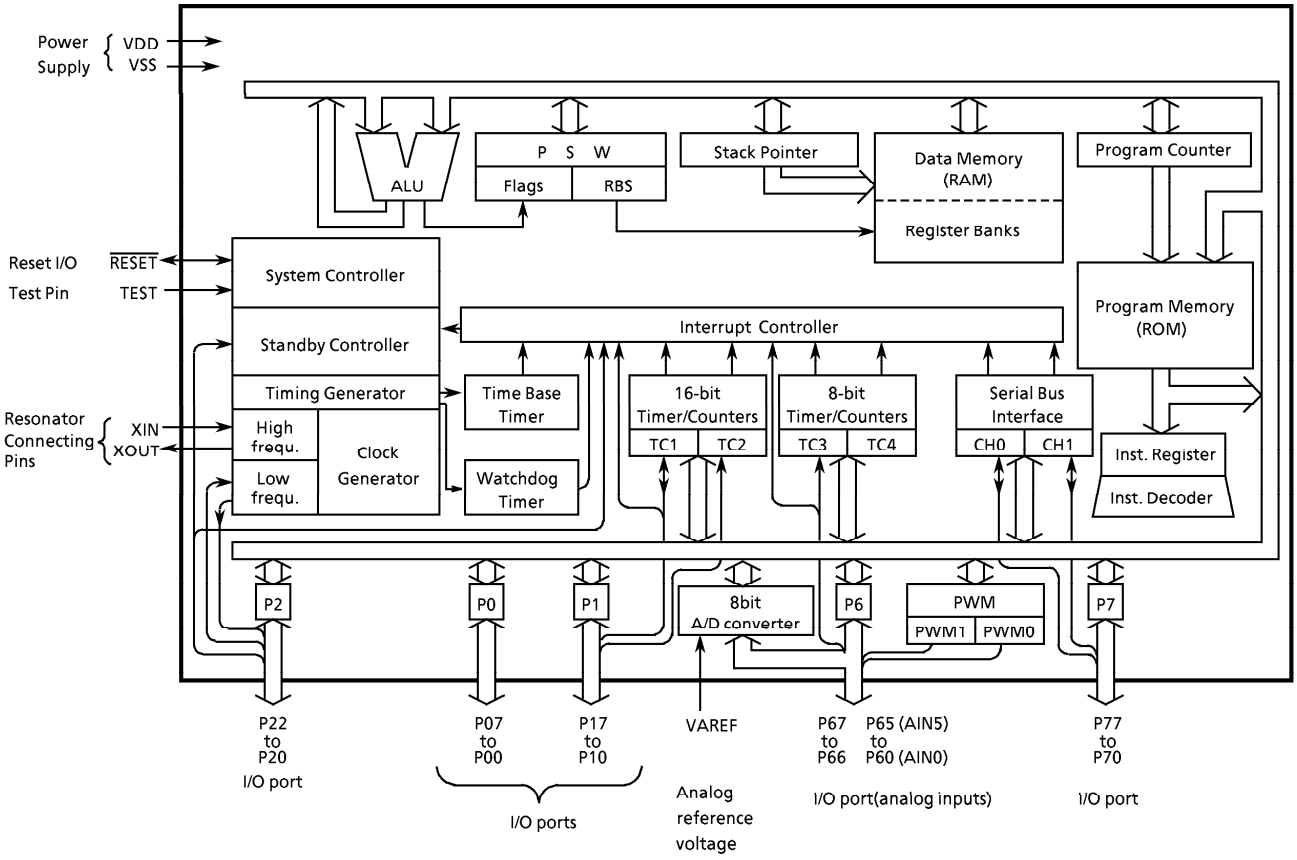
Purchase of TOSHIBA I² C components conveys a license under the Philips I² C Patent Rights to use these components in an I² C system, provided that the system conforms to the I² C Standard Specification as defined by Philips.

PIN ASSIGNMENTS (TOP VIEW)

SDIP42-P-600-1.78



BLOCK DIAGRAM



PIN FUNCTION

PIN NAME	Input / Output	FUNCTION	
P07 to P00	I/O	Two 8-bit programmable input/output ports (tri-state). Each bit of these ports can be individually configured as an input or an output under software control. When used as an external interrupt input or a timer/counter input, these ports must be as inputs. When used as a divider output, the data and input/output control latch must be set to "1".	
P17, P16, P14	I/O		
P15 (TC2)	I/O (Input)		Timer/Counter 2 input
P13 ($\overline{DVO}$)	I/O (Output)		Divider output
P12 (INT2 / TC1)	I/O (Input/Input)		External interrupt input 2 or Timer/Counter 1 input
P11 (INT1)	I/O (Input)		External interrupt input 1
P10 ($\overline{INT0}$)			External interrupt input 0
P22 (XTOUT)	I/O (Output)	3-bit input/output port with latch. When used as an input port, the latch must be set to "1".	Resonator connecting pins (32.768 kHz). For inputting external clock, XTIN is used and XTOUT is opened.
P21 (XTIN)	I/O (Input)		
P20 ($\overline{INT5}$ / $\overline{STOP}$)			External interrupt input 5 or STOP mode release signal input
P67 ($\overline{PWM0}$)	I/O (Output)	8-bit programmable input/output port. Each bit of the port can be individually configured as an input or an output under software control. When used as a PWM output, the data and input/output control latch must be set to "1".	14-bit PWM output
P66 ($\overline{PWM1}$/ INT3/TC3)	I/O (Output/ Input/Input)		7-bit PWM output, External interrupt input 3 or Timer/Counter 3 input
P65 (AIN5) to P60 (AIN0)	I/O (Input)		A/D converter analog inputs
P77	I/O	8-bit programmable input/output port. Each bit of the port can be individually configured as an input or an output under software control. When used as output or I/O port of the Serial Bus Interface, the data and input/output control latch must be set to "1".	
P76 (SO1)	I/O (Output)		Serial data output 1 (SIO)
P75 (SDA1/SI1)	I/O (I or O/Input)		Serial data input/output 1 (I^2C-BUS) / Serial data input 1 (SIO)
P74 (SCL1/$\overline{SCK1}$)	I/O (I or O)		Serial clock input/output 1 (I^2C-BUS/SIO)
P73	I/O		
P72 (SO0)	I/O (Output)		Serial data output 0 (SIO)
P71 (SDA0/SI0)	I/O (I or O/Input)		Serial data input/output 0 (I^2C-BUS) / Serial data input 0 (SIO)
P70 (SCL0/$\overline{SCK0}$)	I/O (I or O)		Serial clock input/output 0 (I^2C-BUS/SIO)
XIN, XOUT	Input, Output	Resonator connecting pins for high-frequency clock. For inputting external clock, XIN is used and XOUT is opened.	
$\overline{RESET}$	I/O	Reset signal input or watchdog timer output/address-trap-reset output/system-clock-reset output.	
TEST	Input	Test pin for out-going test. Be tied to low.	
VDD, VSS	Power Supply	+ 5 V, 0 V (GND)	
VAREF		Analog reference voltage input	

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 (ROM), the data memory (RAM), and the reset circuit.

1.1 Memory Address Map

The TLCS-870 Series is capable of addressing 64K bytes of memory. Figure 1-1 shows the memory address maps of the 87CK43/M43. In the 87CK43/M43, 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 first 128 bytes of the RAM address space.

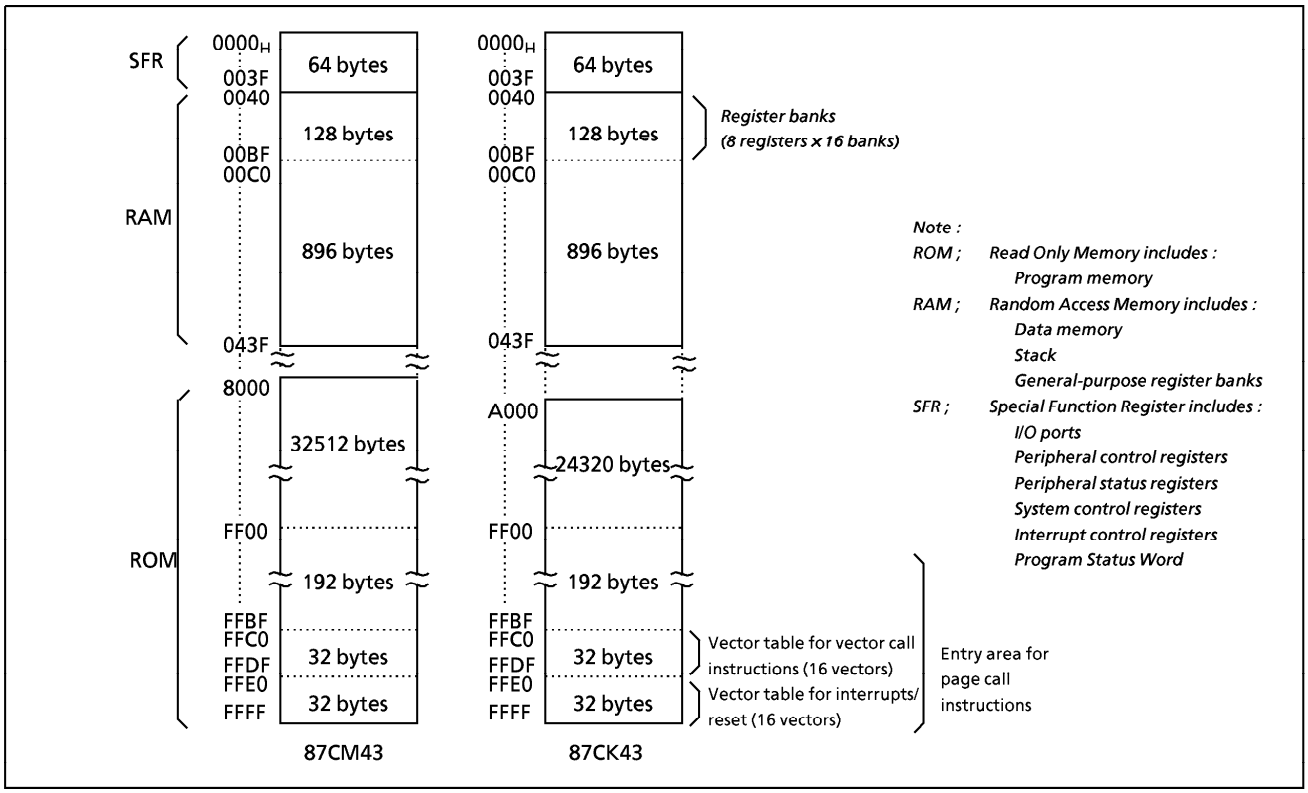


Figure 1-1. Memory Address Maps

1.2 Program Memory (ROM)

The 87CK43 has a 24K bytes (addresses $A000_H$-$FFFF_H$), the 87CM43 has a 32K bytes (addresses 8000_H-$FFFF_H$) of program memory (mask programmed ROM).
Addresses $FF00_H$-$FFFF_H$ in the program memory can also be used for special purposes.

(1) **Interrupt** / **Reset** vector table (addresses $FFE0_H$-$FFFF_H$)
This table consists of a reset vector and 15 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 $FFC0_H$-$FFDF_H$)
This table stores call vectors (subroutine entry address, 2 bytes/vector) for the vector call instructions [CALLV n]. 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 $FF00_H$-$FFFF_H$) for **page call** instructions
This is the subroutine entry address area for the page call instructions [CALLP n]. Addresses $FF00_H$-$FFBF_H$ are normally used because address $FFC0_H$-$FFFF_H$ 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.

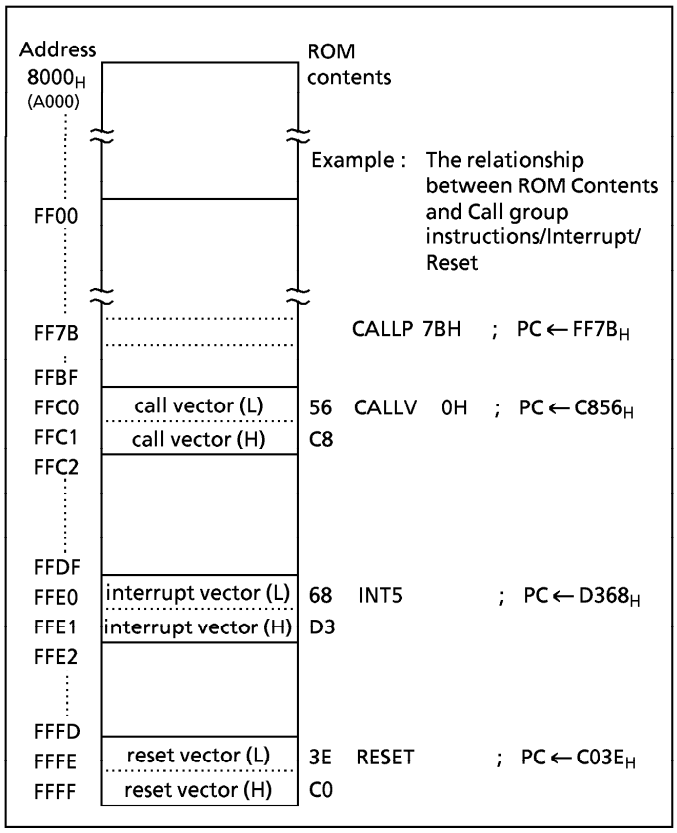


Figure 1-2. Program Memory Map

① 5-bit PC-relative jump [JRS cc, $ + 2 + d]

E8C4H: JRS T, $ + 2 + 08H

When JF = 1, the jump is made to $E8CE_H$, 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 $E8C4_H + 2 = E8C6_H$.)

② 8-bit PC-relative jump [JR cc, $ + 2 + d]

E8C4H : JR Z, $ + 2 + 80H

When ZF = 1, the jump is made to $E846_H$, which is $FF80_H$ (– 128) added to the current contents of the PC.

③ 16-bit absolute jump [JP a]

E8C4H : JP 0C235H

An unconditional jump is made to address $C235_H$. The absolute jump instruction can jump anywhere within the entire 64K-bytes space.

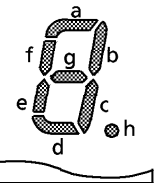
In the TLCS-870 Series, the same instruction used to access the data memory (e.g. [LD A, (HL)]) is also used to read out fixed data (ROM data) stored in the program memory. The register-offset PC-relative addressing (PC + A) instructions can also be used, and the code conversion, table look-up and n-way 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 (HL ≧ 8000_H for 87CM43):

```
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 P7 after executing the following program:

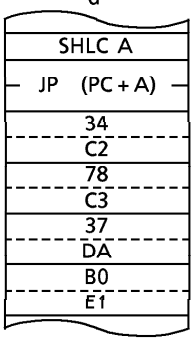
```
        ADD     A, TABLE – $ – 4     ; P7 ←ROM (TABLE + A)
        LD      (P7), (PC + A)
        JRS     T, SNEXT             ; Jump to SNEXT
TABLE : DB      0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0D8H, 80H, 98H
SNEXT :
```



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

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

```
SHLC    A          ; if A = 00H then PC←C234H
JP      (PC + A)     if A = 01H then PC←C378H
                     if A = 02H then PC←DA37H
                     if A = 03H then PC←E1B0H
DW      0C234H, 0C378H, 0DA37H, 0E1B0H
```



Note: DW is a word data definition instruction. A 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 user defined reset vector stored in the vector table (addresses $FFFF_H$ and $FFFE_H$) is loaded into the PC ; therefore, program execution is possible from any desired address. For example, when $C0_H$ and $3E_H$ are stored at addresses $FFFF_H$ and $FFFE_H$, respectively, the execution starts from address $C03E_H$ 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 $C123_H$ is being executed, the PC contains $C125_H$.

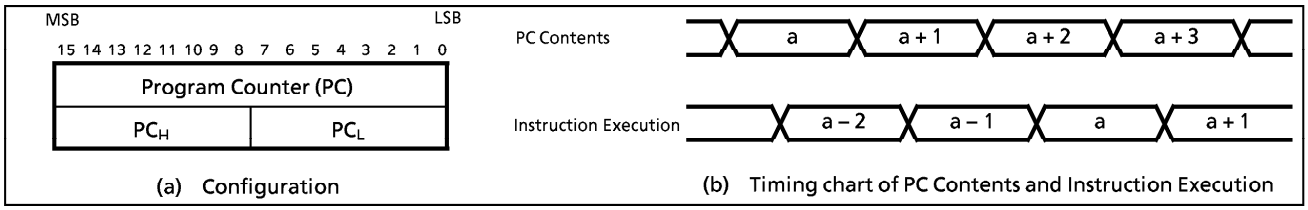


Figure 1-3. Program Counter

1.4 Data Memory (RAM)

The 87CK43/M43 has a 1K bytes (addresses 0040_H-$043F_H$) of data memory (static RAM). Figure 1-4 shows the data memory map.

Addresses 0000_H-$00FF_H$ are used as a direct addressing area to enhance instructions which utilize this addressing mode; therefore, addresses 0040_H-$00FF_H$ in the data memory can also be used for user flags or user counters. General-purpose register banks (8 registers × 16 banks) are also assigned to the 128 bytes of addresses 0040_H-$00BF_H$. 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. The stack depth is limited only by the free data memory size. For more details on the stack, see section "1.7 Stack and Stack Pointer".

The TLCS-870 Series cannot execute programs placed in the data memory. When the program counter indicates a data memory address, a bus error occurs and an address-trap-reset applies.

The $\overline{RESET}$ pin goes low during the address-trap-reset.

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)         ; (00F5H) ← (00F5H) + 1
        AND     (00F5H), 0FH    ; (00F5H) ← (00F5H)∧0FH
```

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

Note : The general-purpuse registers are mapped in the RAM ; therefore, do not clear RAM at the current bank addresses.

Example : Clears RAM to "00_H" except the bank 0:

```
        LD      HL, 0048H       ; Sets start address to HL register pair
        LD      A, H            ; Sets initial data (00H) to A register
        LD      BC, 03F7H         ; Sets number of byte to BC register pair
SRAMCLR: LD     (HL+), A
        DEC     BC
        JRS     F, SRAMCLR
```

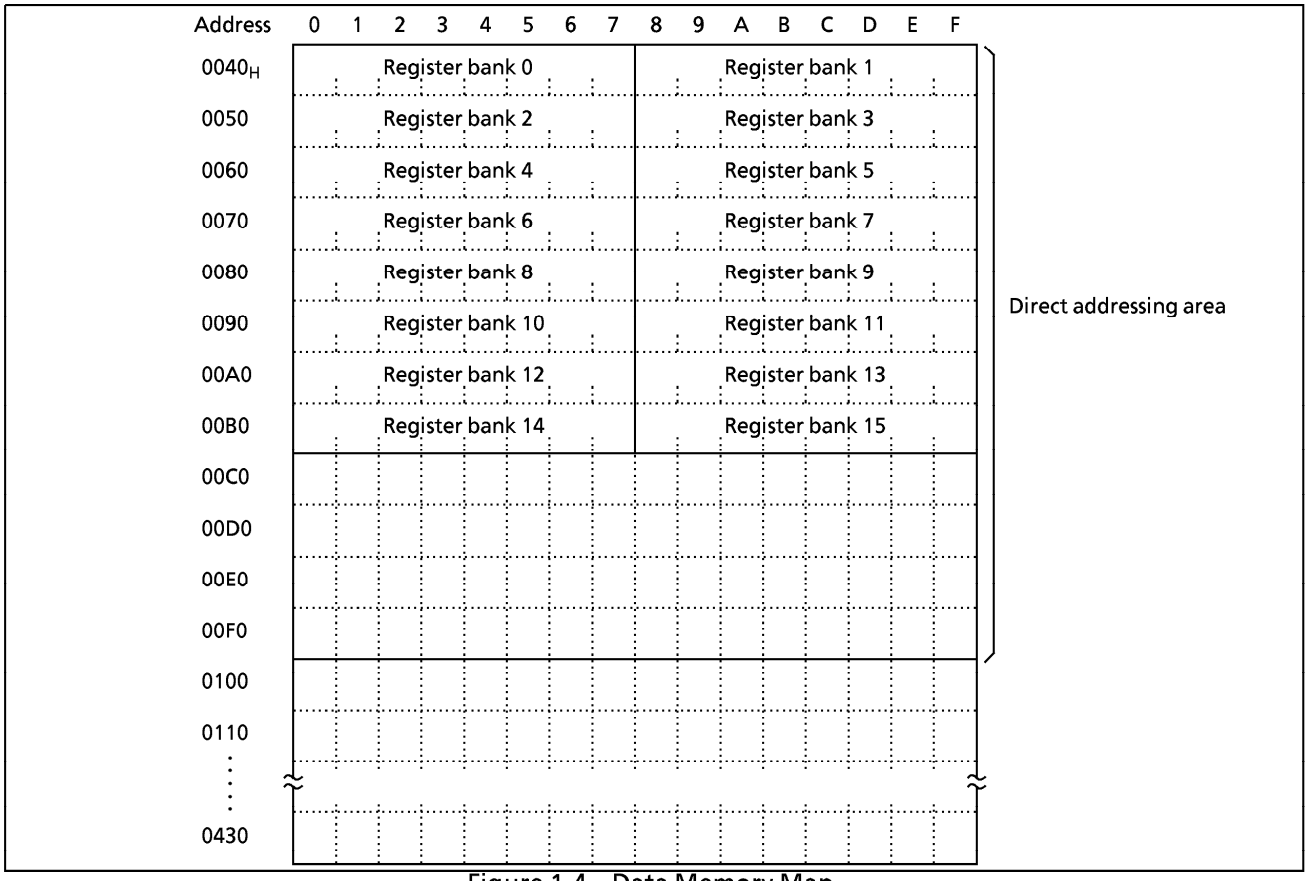


Figure 1-4. Data Memory Map

1.5 General-purpose Register Banks

General-purpose registers are mapped into addresses 0040_H-$00BF_H$ in the data memory as shown in Figure 1-4. 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.

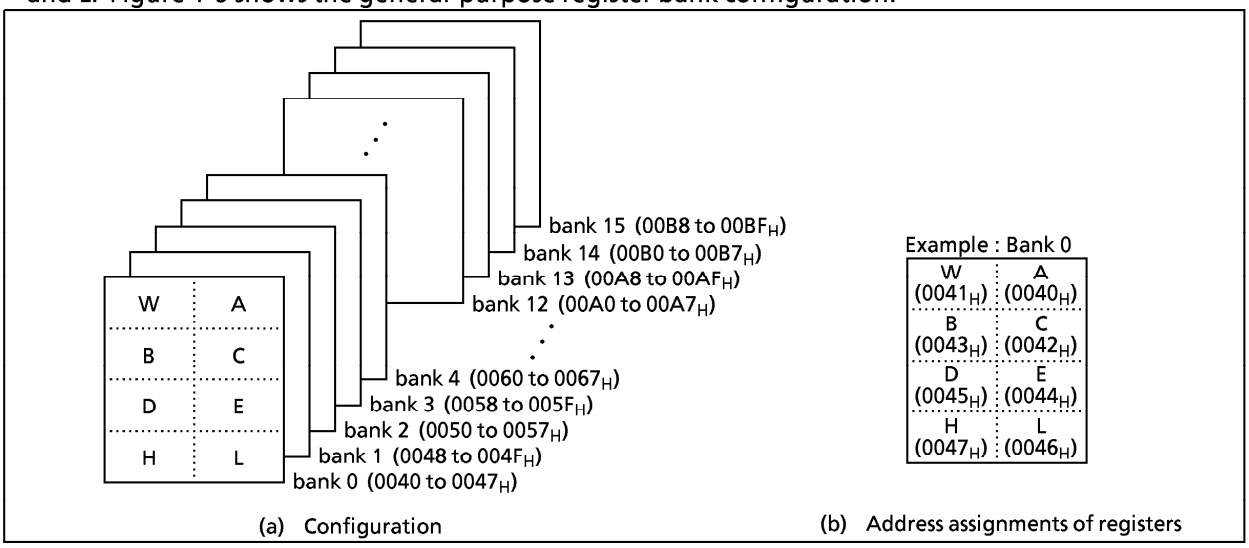


Figure 1-5. General-purpose Register Banks

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 E contents, and stores the result into E.
```

(2) **HL, DE**

The HL and DE specify a memory address. The HL register pair functions as data pointer (HL) / index register (HL + d) / base register (HL + C), and the DE register pair function as a data pointer (DE). The 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)    ; Decrements HL. Then loads the memory contents at the address
                                     specified by new HL into A.
```

The 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, n          ; Sets (number of bytes to transfer) – 1 to B
            LD      HL, DSTA      ; Sets destination address to HL
            LD      DE, SRCA      ; Sets source address to DE
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 counters, 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 [DIV gg, C].

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

Example 2 :  Unsigned integer 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 register 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] and [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 ← PSW (A3-0 ← RBS, A7-4 ← Flags)
```

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 PSW is automatically saved onto the stack. The bank used before the interrupt was accepted is restored automatically 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 restoring)
```

1.6 Program Status Word (PSW)

The program status word (PSW) consists of a register bank selector (RBS) and four 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 (e. g. [LD A, (003FH)], [LD (003FH), A], 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.
[PUSH PSW] and [POP PSW] are 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".

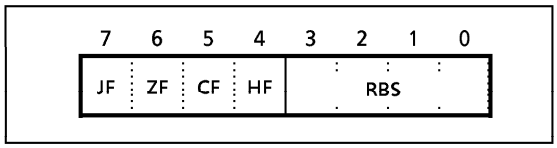


Figure 1-6. PSW (Flags, RBS) Configuration

1.6.2 Flags

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 [SET, CLR, and CPL], the ZF is set to "1" if the contents of the specified bit is "0"; otherwise the ZF is cleared to "0".
This flag is set to "1" when the upper 8 bits of the product are 00_H during the multiplication instruction [MUL], and when 00_H for the remainder during the division instruction [DIV]; otherwise it is cleared to "0".

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

The CF is set to "1" when a carry out of the MSB (most significant bit) of the result occurred during addition or when a borrow into the MSB of the result 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 (quotient-overflow error); otherwise it is cleared. The CF is also affected during the shift/rotate instructions [SHLC, SHRC, ROLC, and RORC]. 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/complement are possible with the CF manipulation instructions.

Example1 : Bit manipulation

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

Example2 : Arithmetic right shift

```
LD      CF, A . 7            ; A ← A / 2
RORC    A
```

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

The HF is set to "1" when a carry occurred between bits 3 and 4 of the operation result during an 8-bit addition, or when a borrow occurred from bit 4 into bit 3 of the result during an 8-bit subtraction; otherwise the HF is cleared to "0". 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)**

Zero or carry information is set to the JF after operation (e. g. INC, ADD, CMP, TEST).
The JF provides the jump condition for conditional jump instructions [JRS T/F, $ + 2 + d], [JR T/F, $ + 2 + d] (T or F is a condition code). Jump is performed if the JF is "1" for a true condition (T), or the JF is "0" for a false condition (F).
The JF is set to "1" after executing the load/exchange/swap/nibble rotate/jump instruction, so that [JRS T, $ + 2 + d] and [JR T, $ + 2 + d] can be regarded as an unconditional jump instruction.

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 become 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	Acc. 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	Acc. 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 and 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 n] / [CALLV n], the contents of the PC (the return address) is saved; on an interrupt acceptance, the contents of the PC and the PSW are saved (the PSW is pushed first, followed by PC_H and PC_L). Therefore, a subroutine call occupies two bytes on the stack; an interrupt occupies three bytes.
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 PC_L is popped first, followed by PC_H and PSW).
The stack can be located anywhere within the data memory space except the register bank area, therefore the stack depth is limited only by the free data memory size.

1.7.2 Stack Pointer (SP)

The stack pointer (SP) is a 16-bit register containing the address of the next free locations on the stack.
The SP is post-decremented when a subroutine call or a push instruction is executed, or when an interrupt is accepted; and the SP is pre-incremented when a return or a pop instruction is executed. Figure 1-8 shows the stacking order.

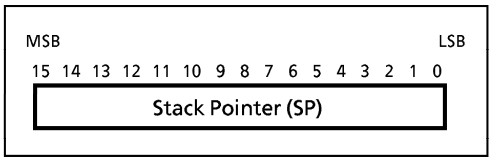


Figure 1-7. Stack Pointer

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, 043FH      ; SP←043F_H
```

Example 2 : To read the SP

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

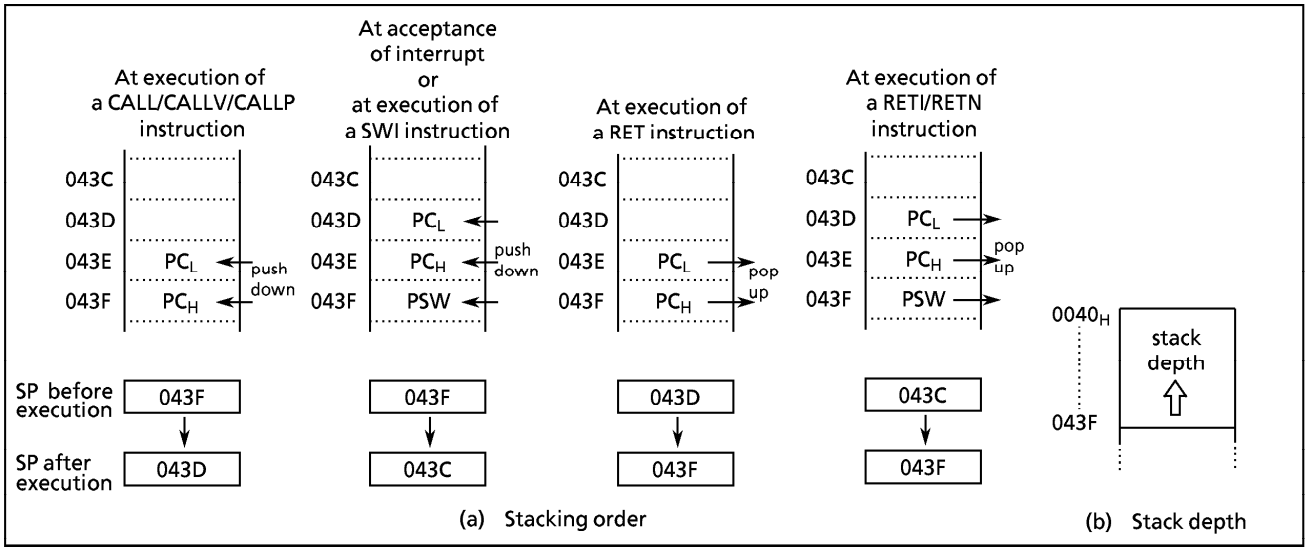


Figure 1-8. Stack

1.8 System Clock Controller

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

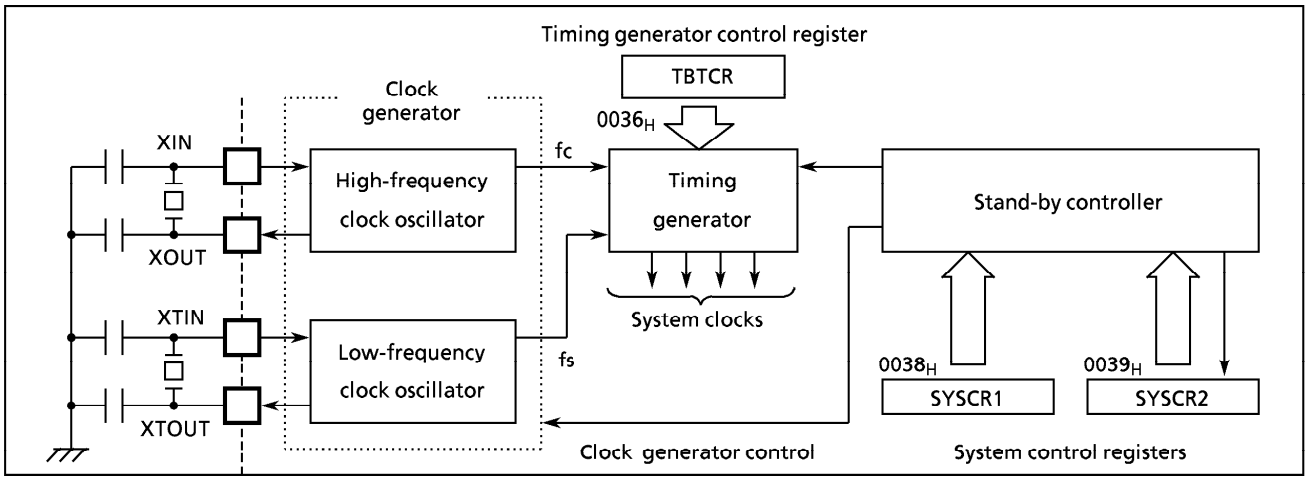


Figure 1-9. System Clock Controller

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. It contains two oscillation circuits: one for the high-frequency clock and one for the low-frequency clock. Power consumption can be reduced by switching of the system clock controller to low-power operation based on the low-frequency clock.

The high-frequency (fc) and low-frequency (fs) clocks can be easily obtained by connecting a resonator between the XIN/XOUT and XTIN/XTOUT pins, respectively. Clock input from an external oscillator is also possible. In this case, external clock is applied to the XIN/XTIN pin with the XOUT/XTOUT pin not connected. The 87CK43/M43 are not provided an RC oscillation.

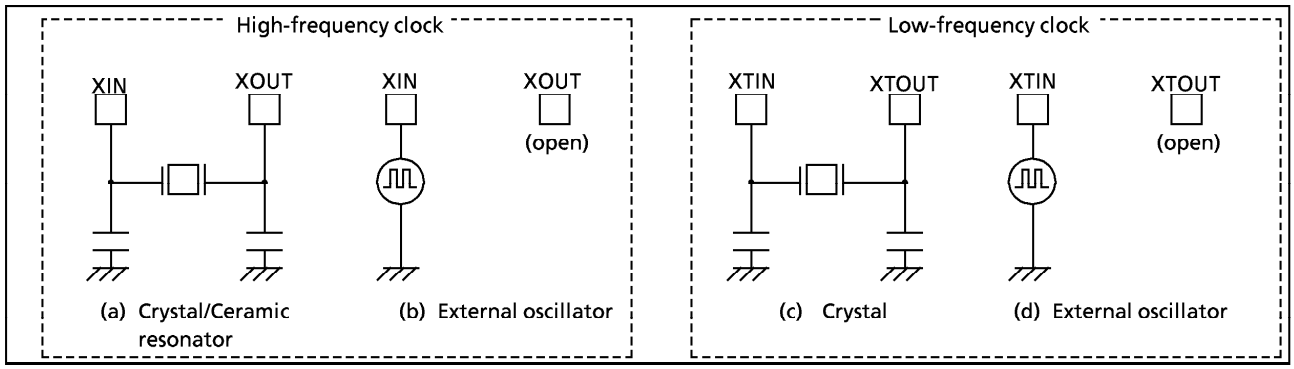
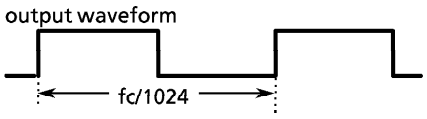


Figure 1-10. Examples of Resonator Connection

Note : Accurate Adjustment of the Oscillation Frequency:
Although no hardware to externally and directly monitor the basic clock pulse is not 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.

Example: To output the high-frequency oscillation frequency adjusting monitor pulse to P13 ($\overline{DVO}$) pin.

```
SFCCHK:  LD   (P1CR), 00001000B   ; Configures port P13 as an output
         SET  (P1).3              ; P13 output latch ← 1
         LD   (TBTCR), 11100000B  ; Enables divider output
         JRS  T, $                ; Loops endless
```



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 ($\overline{DVO}$) 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 TC1 – TC4
⑥ Generation of internal clocks for serial interfaces SIO and I²C-BUS
⑦ Generation of warm-up clocks for releasing STOP mode
⑧ Generation of a clock for releasing reset output

(1) Configuration of Timing Generator

The timing generator consists of a 21-stage divider with a divided-by-4 prescaler, a main system clock generator, and machine cycle counters. An input clock to the 7th stage of the divider depends on the operating mode and DV7CK (bit 4 in TBTCR) shown in Figure 1-11 as follows.
During reset and at releasing STOP mode, the divider is cleared to "0", however, the prescaler is not cleared.

① In the single-clock mode
A divided-by-256 of high-frequency clock ($fc/2^8$) is input to the 7th stage of the divider.

② In the dual-clock mode
During NORMAL2 or IDLE2 mode (SYSCK = 0), an input clock to the 7th stage of the divider can be selected either "$fc/2^8$" or "fs" with DV7CK.
During SLOW or SLEEP mode (SYSCK = 1), fs is automatically input to the 7th stage. To input clock to the 1st stage is stopped ; output from the 1st to 6th stages is also stopped.

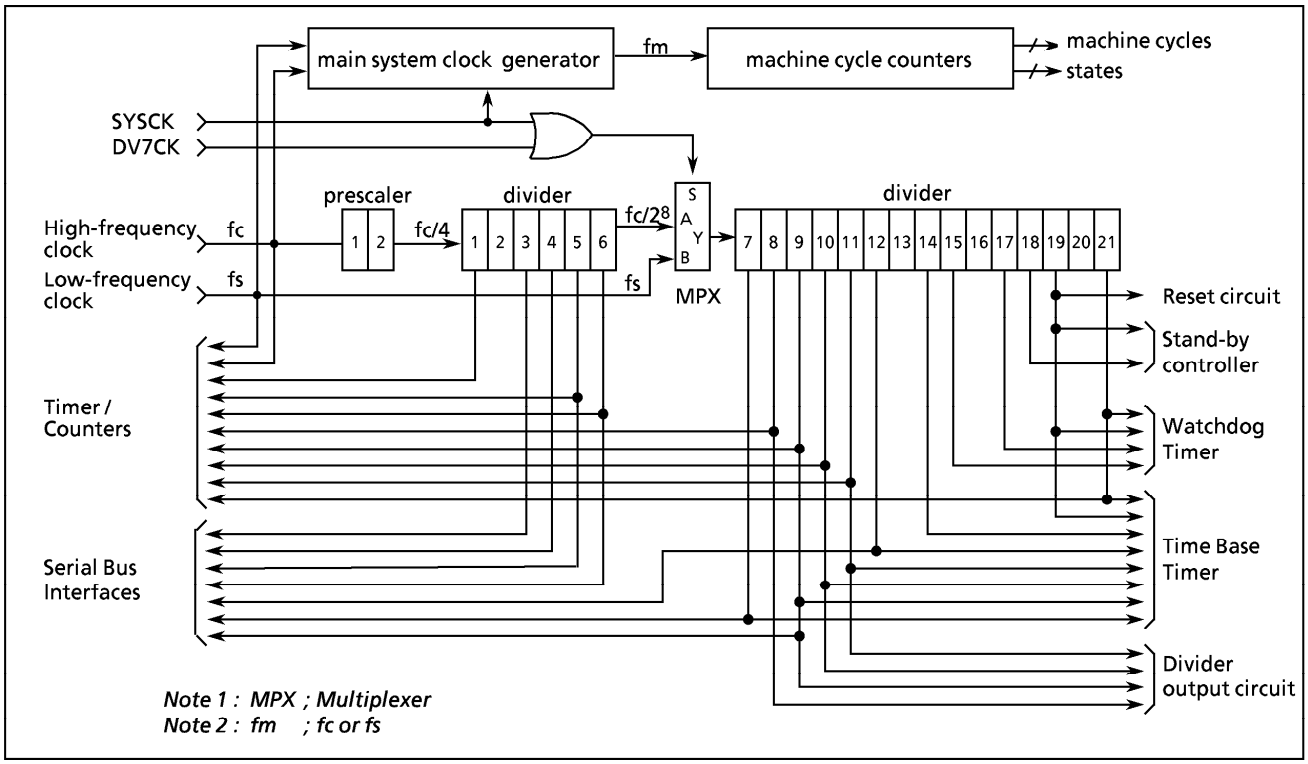


Figure 1-11. Configuration of Timing Generator

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

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

*Note1 : fc ; high-frequency clock [Hz], fs ; low-frequency clock [Hz], * ; don't care*
Note 2 : Do not set DV7CK to "1" in the single-clock mode.
Note 3 : Do not set DV7CK to "1" before low-frequency clock is stable in the dual-clock mode.
Note 4 : TBTCR is a write-only register and must not be used with any of read-modify-write instructions.

Figure 1-12. Timing Generator Control Register

(2) Machine Cycle

Instruction execution and peripheral hardware operation are synchronized with the main system clock. The minimum instruction execution unit is called a "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 - S3), and each state consists of one main system clock.

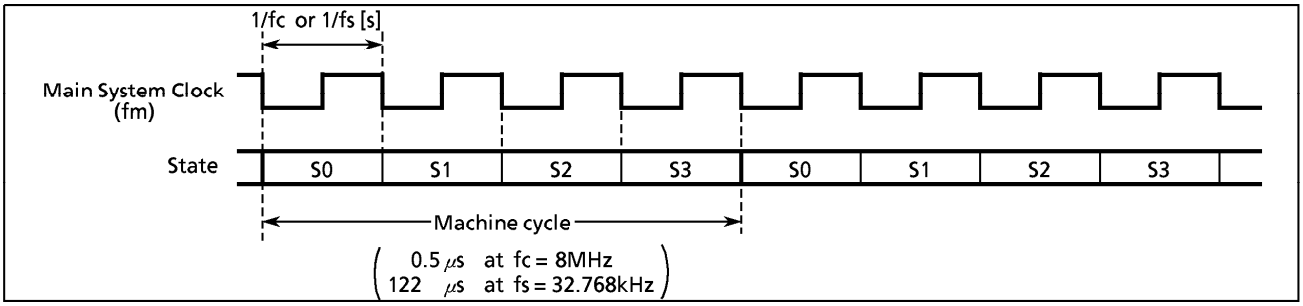


Figure 1-13. Machine Cycle

1.8.3 Stand-by Controller

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

Figure 1-14 shows the operating mode transition diagram and Figure 1-15 shows the system control registers. Either the single-clock or the dual-clock mode can be selected by an option during reset.

(1) **Single-clock mode**

Only the oscillation circuit for the high-frequency clock is used, and P21 (XTIN) and P22 (XTOUT) pins are used as input/output ports. In the single-clock mode, the machine cycle time is 4/fc [s] (0.5 μs at fc = 8 MHz).

① NORMAL1 mode

In this mode, both the CPU core and on-chip peripherals operate using the high-frequency clock. In the case where the single-clock mode has been selected as an option, the 87CK43/M43 are placed in this mode after reset.

② IDLE1 mode

In this mode, the internal oscillation circuit remains active. The CPU and the watchdog timer are halted; however, on-chip peripherals remain active (operate using the high-frequency clock). IDLE1 mode is started by setting IDLE bit in the system control register 2 (SYSCR2), and IDLE1 mode is released to NORMAL1 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 STOP1 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 all output ports can be set to either output hold or high-impedance under software control.

STOP1 mode is started by setting STOP bit in the system control register 1 (SYSCR1), and STOP1 mode is released by an input (either level-sensitive or edge-sensitive can be programmably selected) to the $\overline{STOP}$ pin. After the warming-up period is completed, the execution resumes with the next instruction which follows the STOP mode start instruction.

(2) **Dual-clock mode**

Both high-frequency and low-frequency oscillation circuits are used in this mode. Pins P21 (XTIN) and P22 (XTOUT) cannot be used as input/output ports. The main system clock is obtained from the high-frequency clock in NORMAL2 and IDLE2 modes, and is obtained from the low-frequency clock in SLOW and SLEEP modes. The machine cycle time is 4/fc [s] (0.5 μs at fc = 8 MHz) in NORMAL2 and IDLE2 modes, and 4/fs [s] (122 μs at fs = 32.768 kHz) in SLOW and SLEEP modes. *Note that the 87PM43 is placed in the single-clock mode during reset.* To use the dual-clock mode, the low-frequency oscillator should be turned on by executing [SET (SYSCR2).XTEN] instruction.

① NORMAL2 mode

In this mode, the CPU core operates using the high-frequency clock. On-chip peripherals operate using the high-frequency clock and/or low-frequency clock. In case that the dual-clock mode has been selected by an option, the 87CK43/M43 are placed in this mode after reset.

② SLOW mode

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

Switching back and forth between NORMAL2 and SLOW modes is performed by the system control register 2.

③ IDLE2 mode

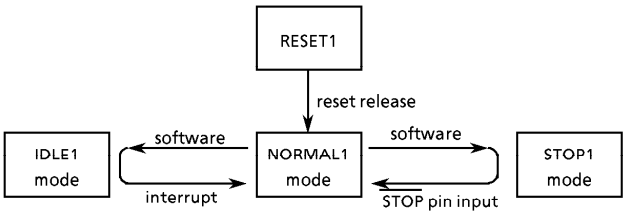
In this mode, the internal oscillation circuits remain active. The CPU and the watchdog timer are halted; however, on-chip peripherals remain active (operate using the high-frequency clock and/or the low-frequency clock). Starting and releasing of IDLE2 mode are the same as for IDLE1 mode, except that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

④ SLEEP mode

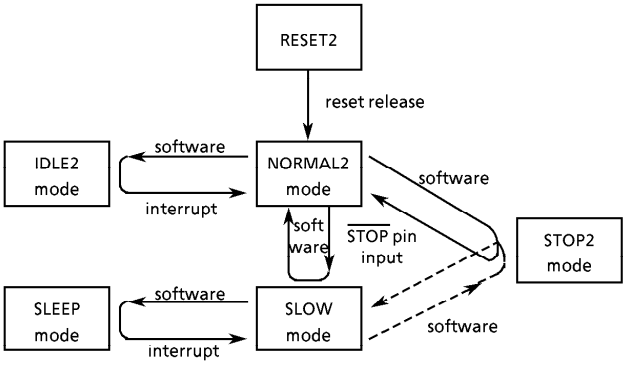
In this mode, the internal oscillation circuit of the low-frequency clock remains active. The CPU, the watchdog timer, and the internal oscillation circuit of the high- frequency clock are halted; however, on-chip peripherals remain active (operate using the low-frequency clock). Starting and releasing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

⑤ STOP2 mode

As in STOP1 mode, all system operations are halted in this mode.



(a) Single-clock mode



(b) Dual-clock mode

Note1 : NORMAL1 and NORMAL2 modes are generically called NORMAL; STOP1 and STOP2 are called STOP; and IDLE1, IDLE2 and SLEEP are called IDLE.

Note2 : There is not RESET2 in the 87PM43.

Operating mode		Frequency		CPU core	On-chip Peripherals	Machine cycle time
		High-frequency	Low-frequency			
Single-Clock	RESET1	turning on oscillation	turning off oscillation	reset	reset	4/fc [s]
	NORMAL1			operate	operate	
	IDLE1			halt		
	STOP1	turning off oscillation			halt	—
Dual-Clock	RESET2	turning on oscillation	turning on oscillation	reset	reset	4/fc [s]
	NORMAL2			High-frequency	operate (High and/or Low)	
	IDLE2			halt		
	SLOW	turning off oscillation		Low-frequency	Low-frequency	4/fs [s]
	SLEEP			halt		
	STOP2		turning off oscillation		halt	—

Figure 1-14. Operating Mode Transition Diagram

System Control Register 1

SYSCR1 (0038_H)

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

(Initial value: 0000 00**)

STOP	STOP mode start	0 : CPU core and peripherals remain active 1 : CPU core and peripherals are halted (start STOP mode)	R/W
RELM	Release method for STOP mode	0 : Edge-sensitive release 1 : Level-sensitive release	
RETM	Operating mode after STOP mode	0 : Return to NORMAL mode 1 : Return to SLOW mode	
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 or 3×2^{13} / fs [s] 01 : 2^{19} / fc or 2^{13} / fs 1* : Reserved	

Note 1 : Always set RETM to "0" when transiting from NORMAL1 mode to STOP1 mode and from NORMAL2 mode to STOP2 mode. Always set RETM to "1" when transiting from SLOW mode to STOP2 mode.

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

Note 3 : fc ; *high-frequency clock* [Hz]
fs ; *low-frequency clock* [Hz]
* ; *don't care*

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

Note5 : If 87CK43/M43 is moved to STOP mode while OUTEN = "0", internal inputs fix "0". Then there is a possibility to set interrupt of falling edge.

System Control Register 2

SYSCR2 (0039_H)

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

(Initial value: 10/100 ****)

XEN	High-frequency oscillator control	0 : Turn off oscillation 1 : Turn on oscillation	R/W
XTEN	Low-frequency oscillator control	0 : Turn off oscillation 1 : Turn on oscillation	
SYSCK	Main system clock select (write)/main system clock monitor (read)	0 : High-frequency clock 1 : Low-frequency clock	
IDLE	IDLE mode start	0 : CPU and watchdog timer remain active 1 : CPU and watchdog timer are stopped (start IDLE mode)	

Note 1 : A reset is applied ($\overline{\text{RESET}}$ pin output goes low) if both XEN and XTEN are cleared to "0".

Note 2 : Do not clear XEN to "0" when SYSCK = 0, and do not clear XTEN to "0" when SYSCK = 1.

*Note 3 : * ; don't care*

Note 4 : Bits 3 - 0 in SYSCR2 are always read in as "1" when a read instruction is executed.

Note 5 : An optional initial value can be selected for XTEN. Always specify when ordering ES (engineering sample). Initial value of XTEN in 87PM43 is "0".

XTEN	operating mode after reset
0	Single-clock mode (NORMAL1)
1	Dual-clock mode (NORMAL2)

Figure 1-15. System Control Registers

1.8.4 Operating Mode Control

(1) **STOP** mode (STOP1, STOP2)

STOP mode is controlled by the system control register 1 (SYSCR1) and the $\overline{STOP}$ pin input. The $\overline{STOP}$ pin is also used both as a port P20 and an $\overline{INT5}$ (external interrupt input 5) pin. STOP mode is started by setting STOP (bit 7 in SYSCR1) to "1". During STOP mode, the following status is maintained.

① Oscillations are turned off, and all internal operations are halted.
② The data memory, registers and port output latches are all held in the status in effect before STOP mode was entered. The port output can be 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 following the instruction which started the STOP mode.

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).

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

In this mode, STOP mode is released by setting the $\overline{STOP}$ 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 $\overline{STOP}$ 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 start STOP mode in the level-sensitive release mode, it is necessary for the program to first confirm that the $\overline{STOP}$ pin input is low. The following method can be used for confirmation:

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

Example : Starting STOP mode with an INT5 interrupt.

```
PINT5 :   TEST   (P2) . 0                  ; To reject noise, the STOP mode does not start if
          JRS    F, SINT5                    port P20 is at high
          LD     (SYSCR1), 01000000B       ; Sets up the level-sensitive release mode.
          SET    (SYSCR1) . 7              ; Starts STOP mode
          LDW    (IL), 1111011101010111B   ; IL11,7,5,3 ← 0 (Clears interrupt latches)
SINT5 :   RETI
```

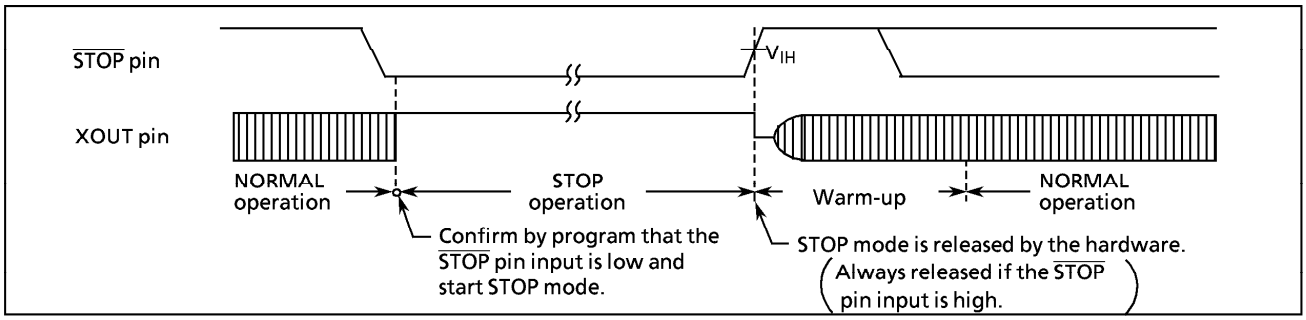


Figure 1-16. Level-sensitive Release Mode

Note 1 : After warming up is started, when $\overline{STOP}$ pin input is changed "L" level, STOP mode is not placed.
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 $\overline{STOP}$ pin input is detected.

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

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

In the edge-sensitive release mode, STOP mode is started even when the $\overline{STOP}$ 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 (disables interrupt service)
SET     (SYSCR1). STOP           ; STOP ← 1 (activates stop mode)
LDW     (IL), 1111011101010111B  ; IL11, 7, 5, 3 ← 0
                                   (clears interrupt latches)
EI                               ; IMF ← 1 (enables interrupt service)
```

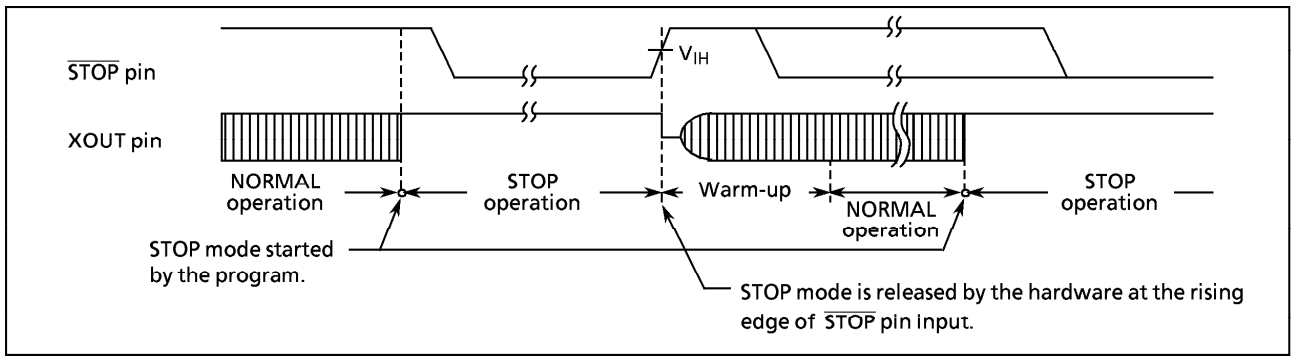


Figure 1-17. Edge-sensitive Release Mode

STOP mode is released by the following sequence:

① When returning to NORMAL2, both the high-frequency and low-frequency clock oscillators are turned on ; when returning to SLOW mode, only the low-frequency clock oscillator is turned on. When returning to NORMAL1, only the high-frequency clock 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. Two different warming-up times can be selected with WUT (bits 2 and 3 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 (e.g. [SET (SYSCR1). 7]). The start is made after the divider of the timing generator is cleared to "0".

Table 1-1. Warming-up Time example

Return to NORMAL mode			Return to SLOW mode	
WUT	At fc = 4.194304 MHz	At fc = 8 MHz	WUT	At fs = 32.768 kHz
3×2^{19} / fc [s]	375 [ms]	196.6 [ms]	3×2^{13} / fs [s]	750 [ms]
2^{19} / fc	125	65.5	2^{13} / fs	250

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 $\overline{RESET}$ pin low, which immediately performs the normal reset operation. 87CK43/M43 restarts NORMAL2 mode (NORMAL1 mode at 87PM43), even if setting is returning to SLOW mode.

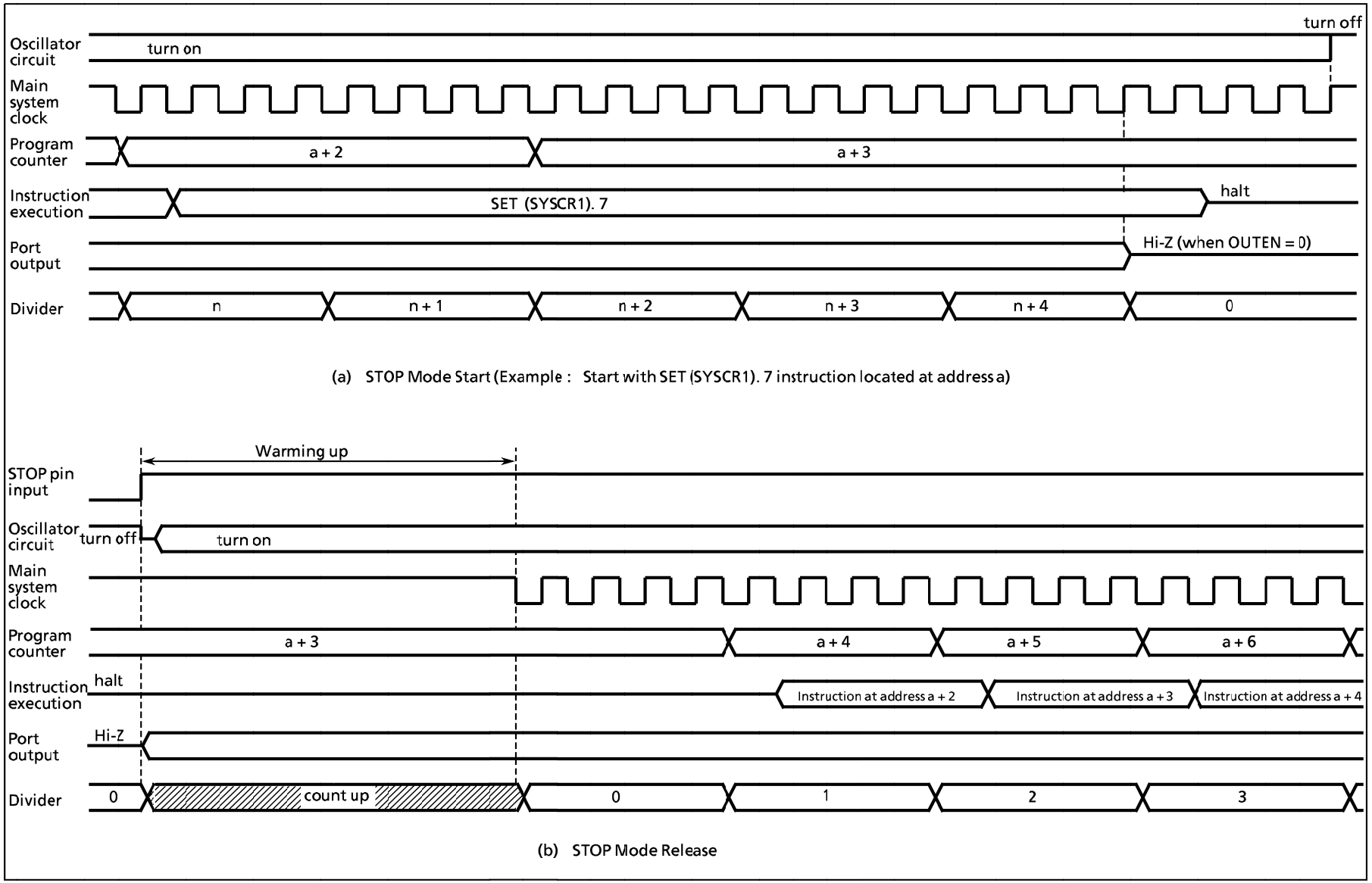


Figure 1-18. STOP Mode Start / Release

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 (IDLE1, IDLE2, SLEEP)

IDLE mode is controlled by the system control register 2 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 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 following the instruction which started IDLE mode.

Example : Starting IDLE mode.

```
SET        (SYSCR2) . 4        ;  IDLE←1
```

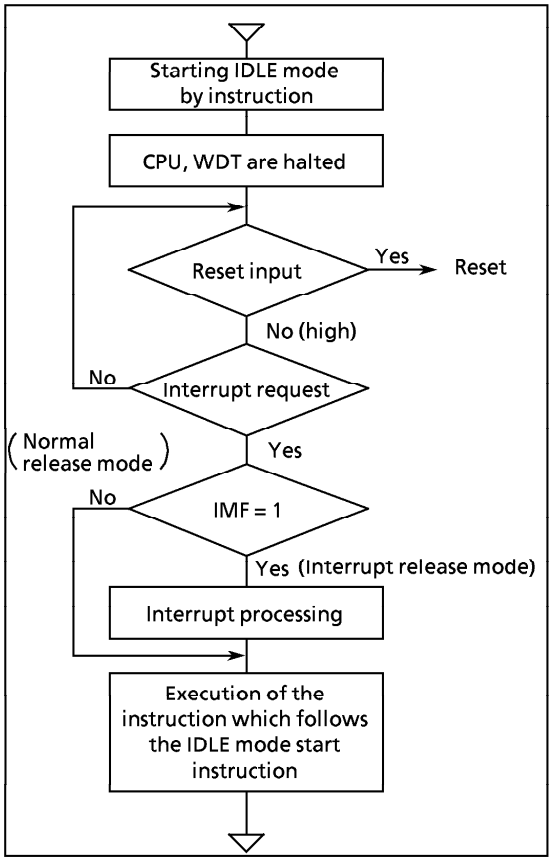


Figure 1-19. IDLE Mode

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 from IDLE1 to NORMAL1, from IDLE2 to NORMAL2, and from SLEEP to SLOW 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 ($\overline{\text{INT0}}$ pin) request. Execution resumes with the instruction following the IDLE mode start instruction (e.g. [SET (SYSCR2).4]). Normally, IL (Interrupt Latch) of interrupt source to release IDLE mode must be cleared by load instructions.

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 ($\overline{\text{INT0}}$ pin) 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. After reset, the 87CK43/M43 are placed in NORMAL 2 mode. (The 87PM43 is placed in NORMAL1 mode.)

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

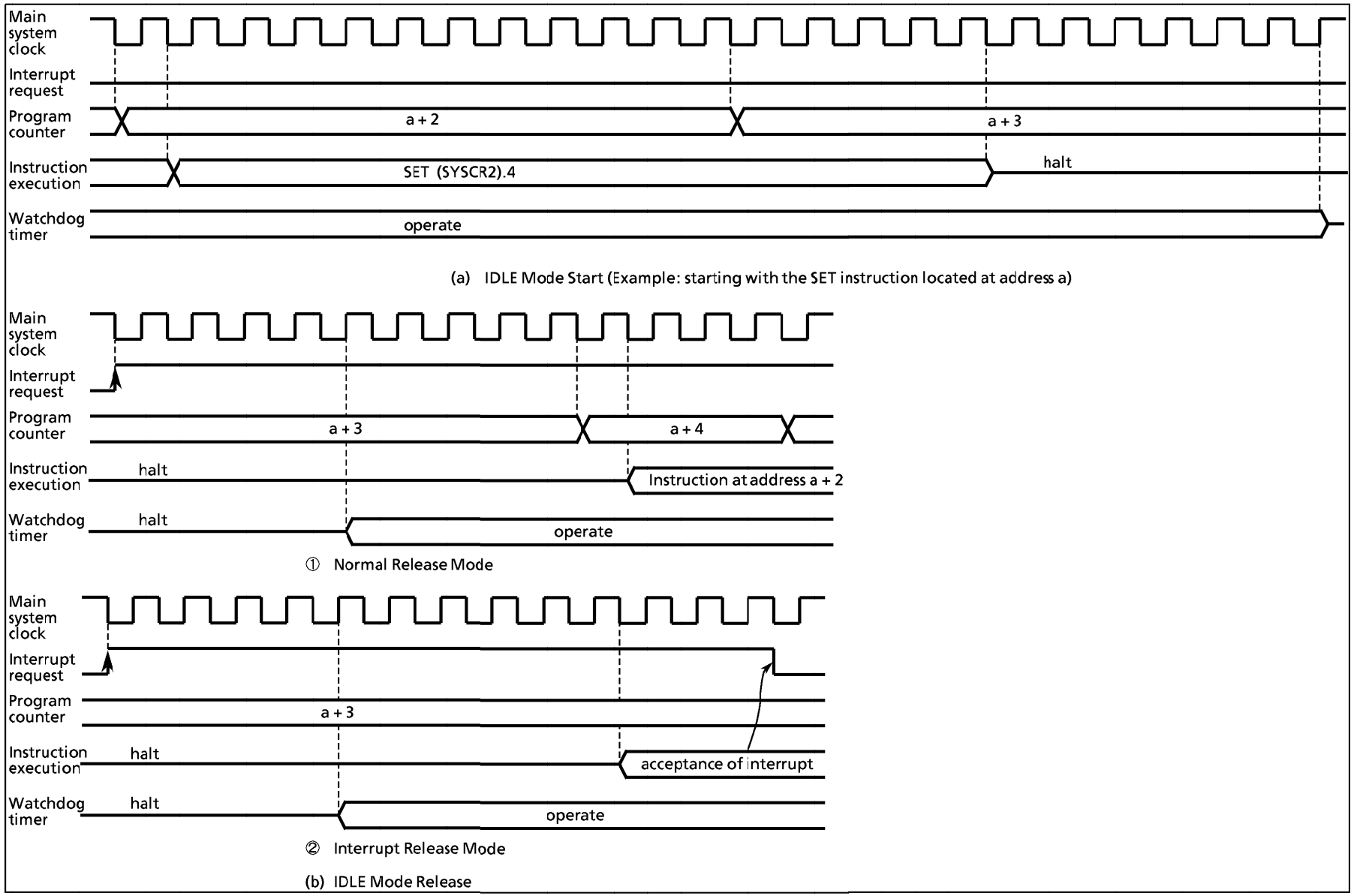


Figure 1-20. IDLE Mode Start/Release

(3) **SLOW** mode

SLOW mode is controlled by the system control register 2 and the timer/counter 2.

a. Switching from NORMAL2 mode to SLOW mode

First, set SYSCK (bit 5 in SYSCR2) to switch the main system clock to the low-frequency clock. Next, clear XEN (bit 7 in SYSCR2) to turn off high-frequency oscillation.

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

*Note : The high frequency clock can be continued oscillation in order to return to **NORMAL2** mode from **SLOW** mode quickly. Always turn off oscillation of high frequency clock when switching from **SLOW** mode to **STOP** mode.*

Example1 : Switching from NORMAL2 mode to SLOW mode.

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

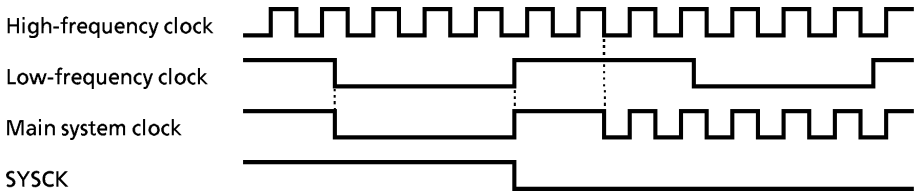
Example2 : Switching to SLOW mode after low-frequency clock oscillation has stabilized.

```
                LD      (TC2CR), 14H     ; Sets TC2 mode
                                           (timer mode, source clock : fs)
                LDW     (TREG2), 8000H   ; Sets warming-up time
                                           (according to Xtal characteristics)
                SET     (EIRH). EF14     ; INTTC2 interrupt enable
                LD      (TC2CR), 34H     ; Starts TC2
                 ⋮
PINTTC2 :       LD      (TC2CR), 10H     ; Stops TC2
                SET     (SYSCR2) . 5     ; SYSCK←1
                CLR     (SYSCR2) . 7     ; XEN←0
                RETI
                 ⋮
VINTTC2 :       DW      PINTTC2          ; INTTC2 vector table
```

b. Switching from SLOW mode to NORMAL2 mode

First, set XEN (bit 7 in SYSCR2) to turn on the high-frequency oscillation. When time for stabilization (warm-up) has been taken by the timer/counter 2 (TC2), clear SYSCK (bit 5 in SYSCR2) to switch the main system clock to the high-frequency clock.

Note1 : After SYSCK is cleared, executing instructions is continued by low-frequency clock while low-frequency clock and high-frequency clock synchronize each other.



Note2 : SLOW mode can also be released by setting the $\overline{\text{RESET}}$ pin low, which immediately performs the reset operation. After reset, the 87CK43/M43 are placed in NORMAL2 mode. (The 87PM43 is placed in NORMAL1 mode.)

Example : Switching from SLOW mode to NORMAL2 mode (fc = 8 MHz, warming-up time is about 7.9 ms).

```
            SET     (SYSCR2) . 7      ; XEN←1      (turns on high-frequency oscillation)
            LD      (TC2CR), 10H      ; Sets TC2 mode
                                        (timer mode, source clock: fc)
            LD      (TREG2 + 1), 0F8H ; Sets the warming-up time
                                        (according to frequency and resonator characteristics)
            SET     (EIRH). EF14      ; INTTC2 interrupt enable
            LD      (TC2CR), 30H      ; Starts TC2
             ⋮
PINTTC2 :   LD      (TC2CR), 10H      ; Stops TC2
            CLR     (SYSCR2) . 5      ; SYSCK←0    (Switches the main system clock to the
                                                   high-frequency clcok)
            RETI
             ⋮
VINTTC2 :   DW      PINTTC2           ; INTTC2 vector table
```

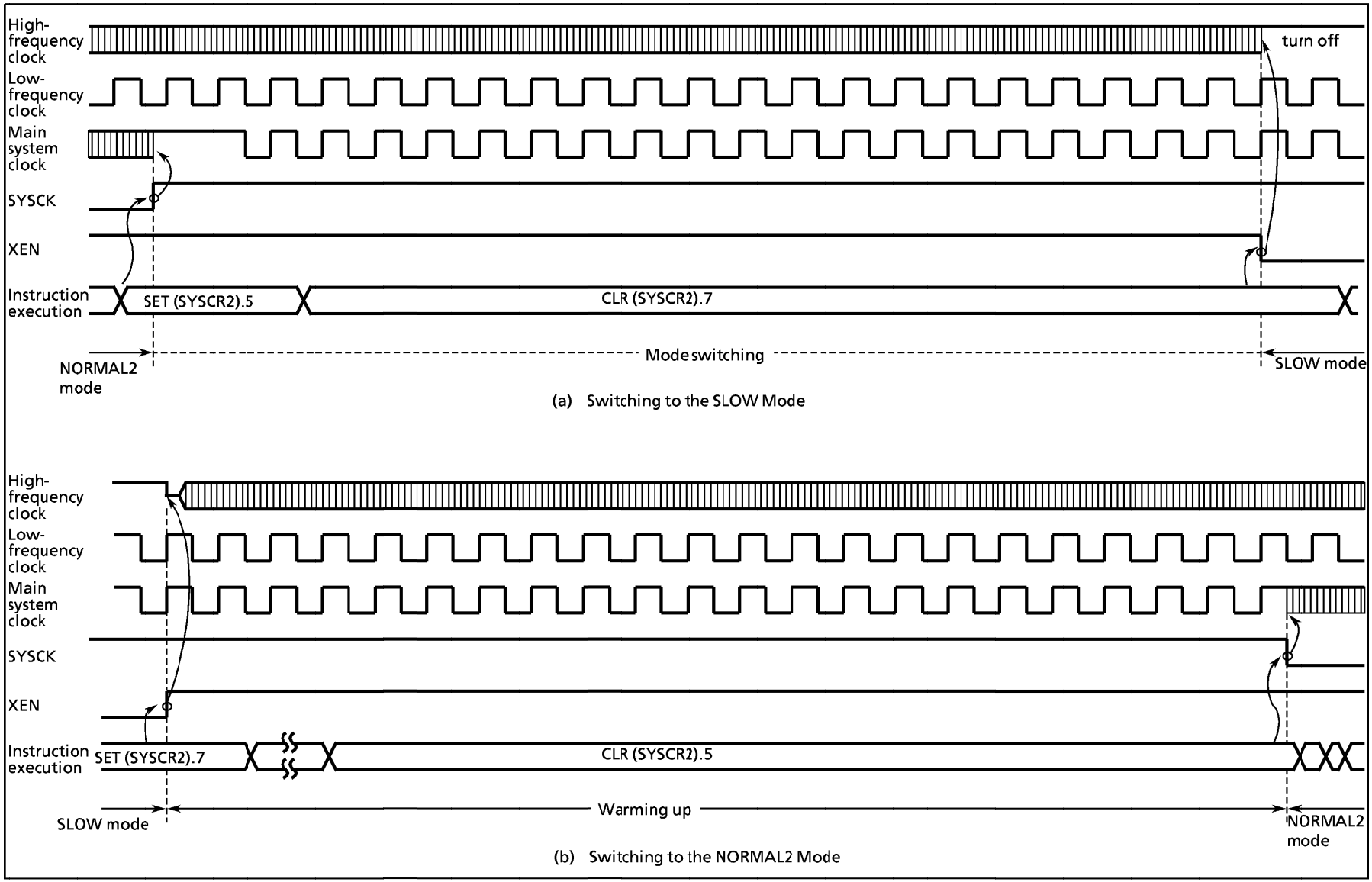


Figure 1-21. Switching between the NORMAL2 and SLOW Modes

1.9 Interrupt Controller

The 87CK43/M43 has a total of 13 interrupt sources: 5 externals and 8 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 1-22 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 TC1 interrupt)	$IMF \cdot EF_4 = 1$	IL_4	$FFF6_H$	4
External	INT1	(External interrupt 2)	$IMF \cdot EF_5 = 1$	IL_5	$FFF4_H$	5
Internal	INTTBT	(Time Base Timer interrupt)	$IMF \cdot EF_6 = 1$	IL_6	$FFF2_H$	6
External	INT2	(External interrupt 2)	$IMF \cdot EF_7 = 1$	IL_7	$FFF0_H$	7
Internal	INTTC3	(8-bit TC3 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 TC4 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
Internal	INTTC2	(16-bit TC2 interrupt)	$IMF \cdot EF_{14} = 1$	IL_{14}	$FFE2_H$	14
External	INT5	(External interrupt 5)	$IMF \cdot EF_{15} = 1$	IL_{15}	$FFE0_H$	Low 15

(1) Interrupt Latches ($IL_{15 \text{ to } 2}$)

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 $003C_H$ and $003D_H$ 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.* Thus, interrupt requests can be cancelled and initialized by the program. Don't clear the IL_2 for a watchdog timer interrupt to "0". Note that interrupt latches cannot be 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), 1111000000111111B     ; IL11 to IL6←0
```

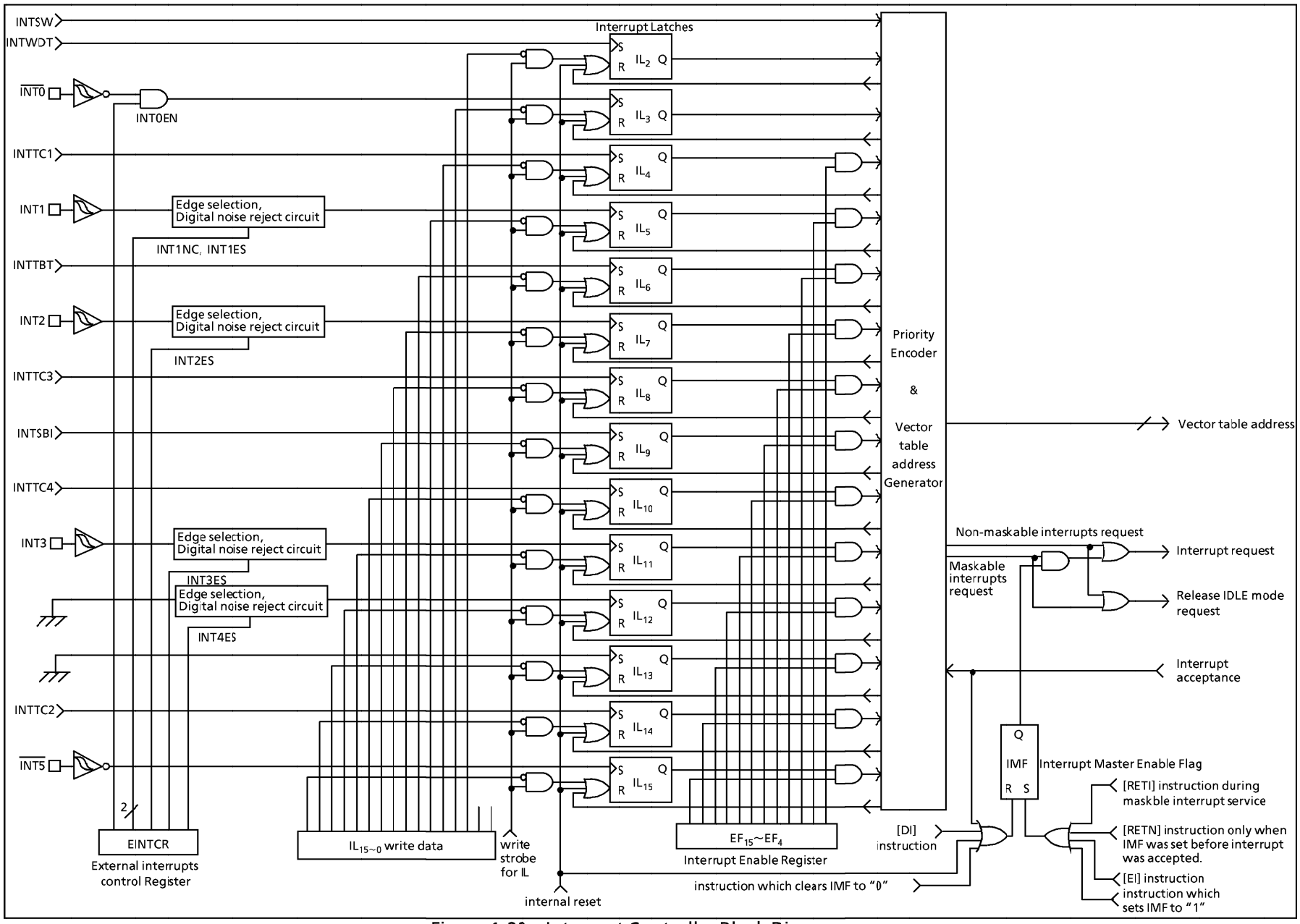


Figure 1-22. Interrupt Controller Block Diagram

Example 2 : Reads interrupt latches

```
LD      WA, (IL)        ; W←IL_H, A←IL_L
```

Example 3: Tests an interrupt latch

```
TEST    (IL).7          ; if IL_7 = 1 then jump
JR      F, SSET
```

(2) **Interrupt Enable Register** (EIR)

The interrupt enable 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. For example, the watchdog timer interrupt is not accepted during the software interrupt service.

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, except for pseudo non-maskable 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 been 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 the IMF remains "0" when cleared by 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. The 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)

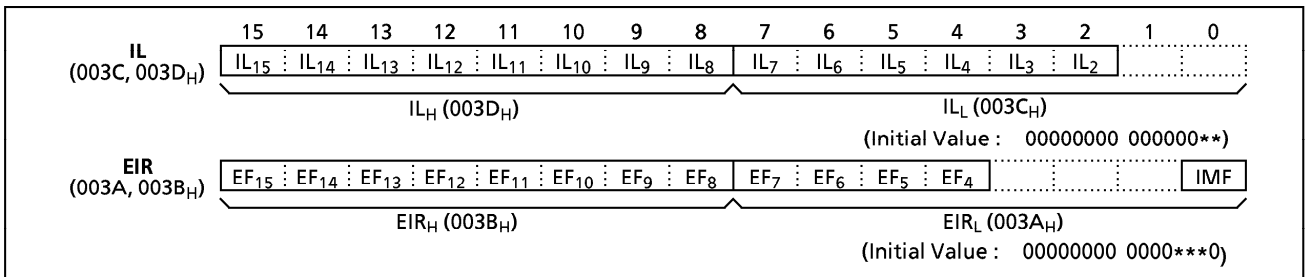
These flags enable and disable the acceptance of individual maskable interrupts, except for an external interrupt 0. Setting the corresponding bit of an individual interrupt enable flag to "1" enables acceptance of an interrupt, setting the bit to "0" disables acceptance.

Example 1 : Sets EF for individual interrupt enable, and sets IMF to "1".

```
LDW     (EIR), 1110100010100001B    ;  EF_15 to EF_13, EF_11, EF_7, EF_5, IMF←1
```

Example 2 : Sets an individual interrupt enable flag to "1".

```
SET     (EIRH).4                    ;  EF_12←1
```



Note1 : Do not use any read-modify-write instruction such as bit manipulation for clearing IL.
Note2 : Do not clear the IL_2 to "0" by an instruction.
Note3 : Do not set IMF to "1" during non-maskable interrupt service programs.

Figure 1-23. 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 (4 μs at fc = 8 MHz in NORMAL mode) 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] (for pseudo non-maskable interrupts).

(1) Interrupt acceptance processing

Interrupt acceptance processing is as follows :

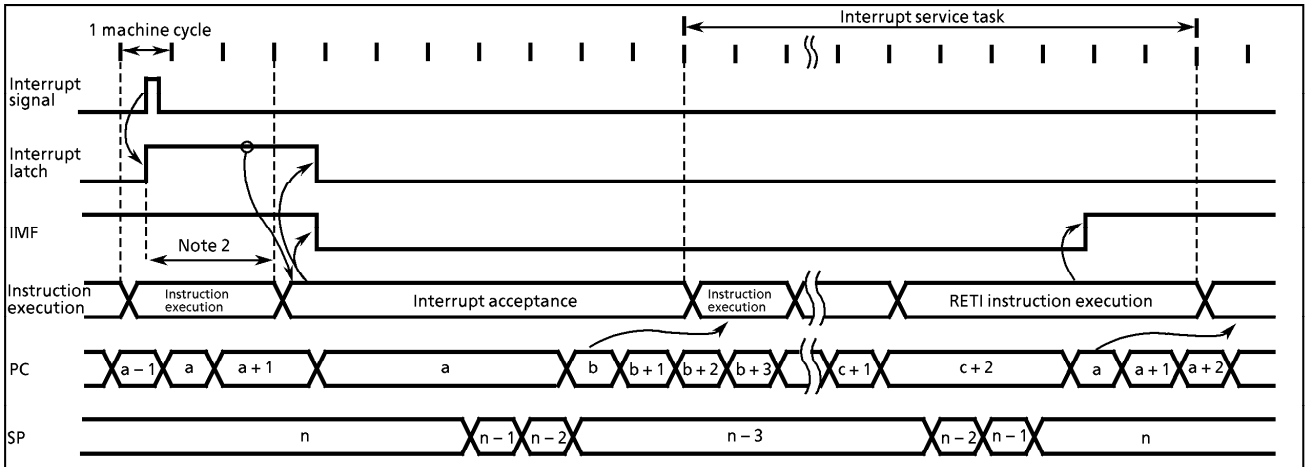
① The interrupt master enable flag (IMF) is cleared to "0" to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.

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

③ The contents of the program counter (return address) and the program status word are saved (pushed) onto the stack. The stack pointer is decremented 3 times.

④ The entry address of the interrupt service program is read from the vector table address, and the entry address is loaded to the program counter.

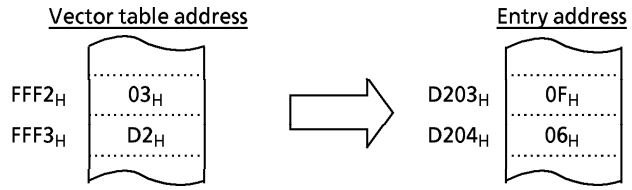
⑤ The instruction stored at the entry address of the interrupt service program is executed.



Note1 : a ; return address, b ; entry address, c ; address when the RETI instruction is stored
Note2 : The maximum response time from when an IL is set until an interrupt acceptance processing starts is 38/fc or 38/fs [s].

Figure 1-24. Timing Chart of Interrupt Acceptance and Interrupt Return Instruction

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



A maskable interrupt is not accepted until the IMF is set to "1" even if a maskable interrupt of higher priority than that of the current 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 of the $\overline{\text{INT0}}$ pin must be disabled with INT0EN in the external interrupt control register (EINTCR) or interrupt processing must be avoided by the program.

Example 1 : Disables an external interrupt 0 using INT0EN:

```
        LD      (EINTCR), 00000000B ;  INT0EN←0
```

Example 2 : Disables the processing of external interrupt 0 under the software control (using bit 0 at address $00F0_H$ 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) Saving/Restoring General Purpose Registers

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 (1μs at 8 MHz)
          [ interrupt processing ]
          RETI                    ;  Restores bank and Returns
```

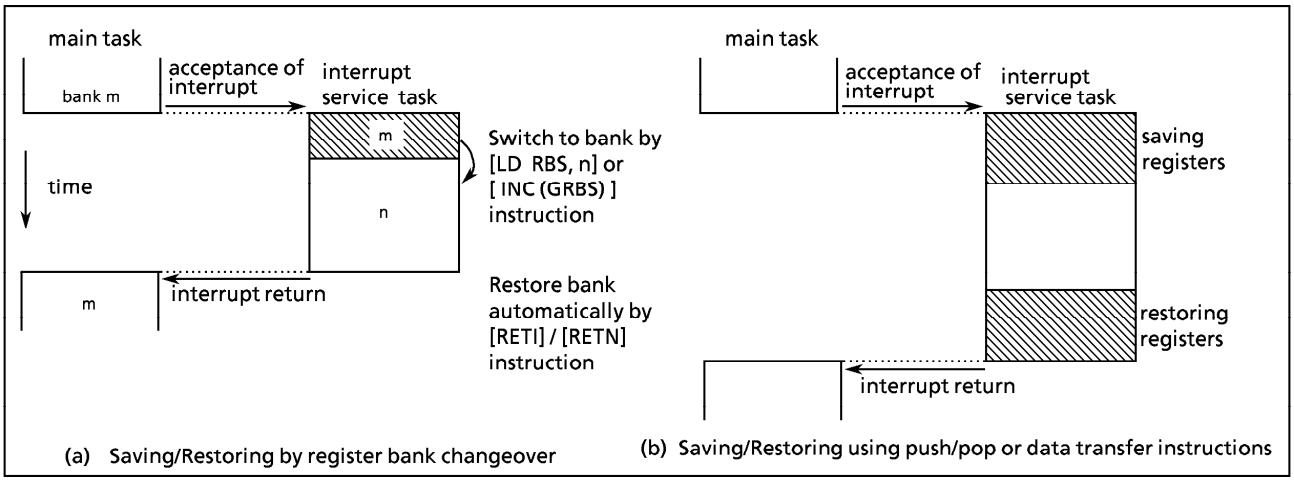
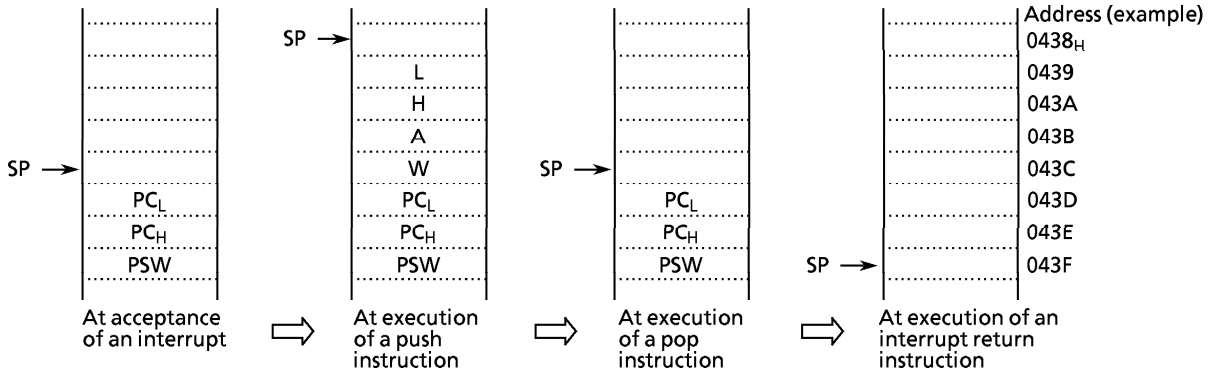


Figure 1-25. Saving/Restoring General-purpose Registers

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
```



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 a register using data transfer instructions

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

(3) The Interrupt Return
The interrupt return instructions [RETI]/[RETN] 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.	① The contents of the program counter and program status word are restored from the stack.
② The stack pointer is incremented 3 times.	② The stack pointer is incremented 3 times.
③ The interrupt master enable flag is set to "1".	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 task 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. Thus, the [SWI] instruction behaves like the [NOP] instruction.

Note : At the development tool, if processing of a non-maskable interrupt is already underway, executing the SWI instruction will generate a software interrupt as a soft ware brake.

Note : At the development tool, if processing of a non-maskable interrupt is already underway, executing the SWI instruction will generate a software interrupt as a soft ware brake.

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

① Address Error Detection

FF_H is read if for some cause such as noise the CPU attempts to fetch an instruction from a non-existent memory address. 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. The address trap reset is generated in case that an instruction is fetched from RAM or SFR areas.

Note : The fetch data from addresses $7F80_H$ to $7FFF_H$ (test ROM area) for 87CK43/M43 is not "FF_H".

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 87CK43/M43 each have five external interrupt inputs ($\overline{INT0}$, INT1, INT2, INT3, and $\overline{INT5}$). Three 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, INT2, INT3.

The $\overline{INT0}$/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 $\overline{INT0}$/P10 pin function selection are performed by the external interrupt control register (EINTCR). When INT0EN = 0, the IL_3 will not be set even if the falling edge of $\overline{INT0}$ pin input is detected.

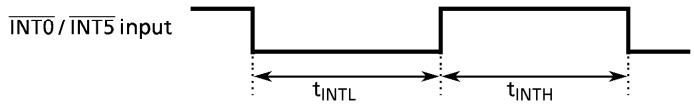
Table 1-3. External Interrupts

Source	Pin	Secondary function pin	Enable conditions	Edge	Digital noise reject
INT0	$\overline{INT0}$	P10	IMF = 1, INT0EN = 1	falling edge	— (hysteresis input)
INT1	INT1	P11	IMF · EF_5 = 1	falling edge or rising edge	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.
INT2	INT2	P12/TC1	IMF · EF_7 = 1		Pulses of less than 7/fc [s] are eliminated as noise. Pulses equal to or more than 24/fc [s] are regarded as signals.
INT3	INT3	P66/TC3/$\overline{PWM1}$	IMF · EF_{11} = 1		
INT5	$\overline{INT5}$	P20/$\overline{STOP}$	IMF · EF_{15} = 1	falling edge	— (hysteresis input)

Note 1 : The noise rejection function is turned off in the SLOW and SLEEP modes. Also, the noise reject times are not constant for pulses input while transiting between operating modes (NORMAL2↔SLOW)

Note 2 : The noise rejection function is also affected for timer/counter input (TC1 and TC3 pins).

Note 3 : The pulse width (both "H" and "L" level) for input to the $\overline{INT0}$ and $\overline{INT5}$ pins must be over 1 machine cycle.



t_{INTL}, t_{INTH} > tcyc (*Note : tcyc = 4/fm [s]*)

Note 4 : If a noiseless signal is input to the external interrupt pin in the NORMAL 1/2 or IDLE 1/2 mode, the maximum time from the edge of input signal until the IL is set is as follows :

① INT1 pin 49/fc [s] (INT1NC = 1) , 193/fc [s] (INT1NC = 0)
② INT2,INT3 pins 25/fc [s]

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 $\overline{INT5}$ (P20/$\overline{STOP}$) which are also used as ports may be set to "1". To specify high-impedance for port output in stop mode, first disable interrupt 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 (disables interrupt service)
SET (SYSCR1). STOP             ;  STOP ← 1 (activates stop mode)
LDW  (IL),1111011101010111B    ;  IL11, 7, 5, 3 ← 0 (clears interrupt latches)
EI                             ;  IMF ← 1 (enables interrupt service)
```

EINTCR (0037_H)

7	6	5	4	3	2	1	0
INT1 NC	INT0 EN			INT3 ES	INT2 ES	INT1 ES	

(Initial value : 00** 000*)

INT1NC	Noise reject time select	0 : Pulses of less than 63/fc [s] are eliminated as noise 1 : Pulses of less than 15/fc [s] are eliminated as noise	Write only
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 INT2 ES INT1 ES	INT3 to INT1 edge select	0 : Rising edge 1 : Falling edge	

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

Note 2 : Edge detection during switching edge selection is invalid.

Note 3 : Do not change EINTCR when IMF = 1. After changing EINTCR, interrupt latches of external interrupt inputs must be cleared to "0" using load instruction.

Note 4 : In order to change of external interrupt input by rewriting the contents of INT2ES and INT3ES during NORMAL1/2 mode, clear interrupt latches of external interrupt inputs (INT2 and INT3) after 8 machine cycles from the time of rewriting. During SLOW mode, 3 machine cycles are required.

*Note 5 : In order to change an edge of timer counter input by rewriting the contents of INT2ES and INT3ES during NORMAL1/2 mode, rewrite the contents after timer counter is stopped (TC*s = 0), that is, interrupt disable state. Then, clear interrupt laches of external interrupt inputs (INT2 and INT3) after 8 machine cycles from the time of rewriting to change to interrupt enable state. Finally, start timer counter. During SLOW mode, 3 machine cycles are required.*

Example : When changing TC1 pin inputs edge in event counter mode from rising edge to falling edge.

```
                    LD (TC1CR),00001100B    ;  TC1S ← 0 (stops TC1)
                    DI                      ;  IMF ← 0 (disables interrupt service)
                    LD (EINTCR),00000100B   ;  INT2ES ← 1 (change edge selection)
                    NOP
  ↑ 8 machine       ~
    cycles          NOP
  ↓                 LD (ILL),01111111B      ;  IL7 ← 0 (clears interrupt latch)
                    EI                      ;  IMF ← 1 (enables interrupt service)
                    LD (TC1CR),00011100B    ;  TC1S ← 1 (starts TC1)
```

Note 6 : If changing the contents of INT1ES during NORMAL1/2 mode, interrupt latch of external interrupt input INT1 must be cleared after 14 machine cycles (when INT1NC = 1) or 50 machine cycles (when INT1NC = 0) from the time of changing. During SLOW mode, 3 machine cycles are required.

Figure 1-26. 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 a reset output or a non-maskable interrupt request. However, selection is possible only once after reset. At first the reset output is selected.

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

1.10.1 Watchdog Timer Configuration

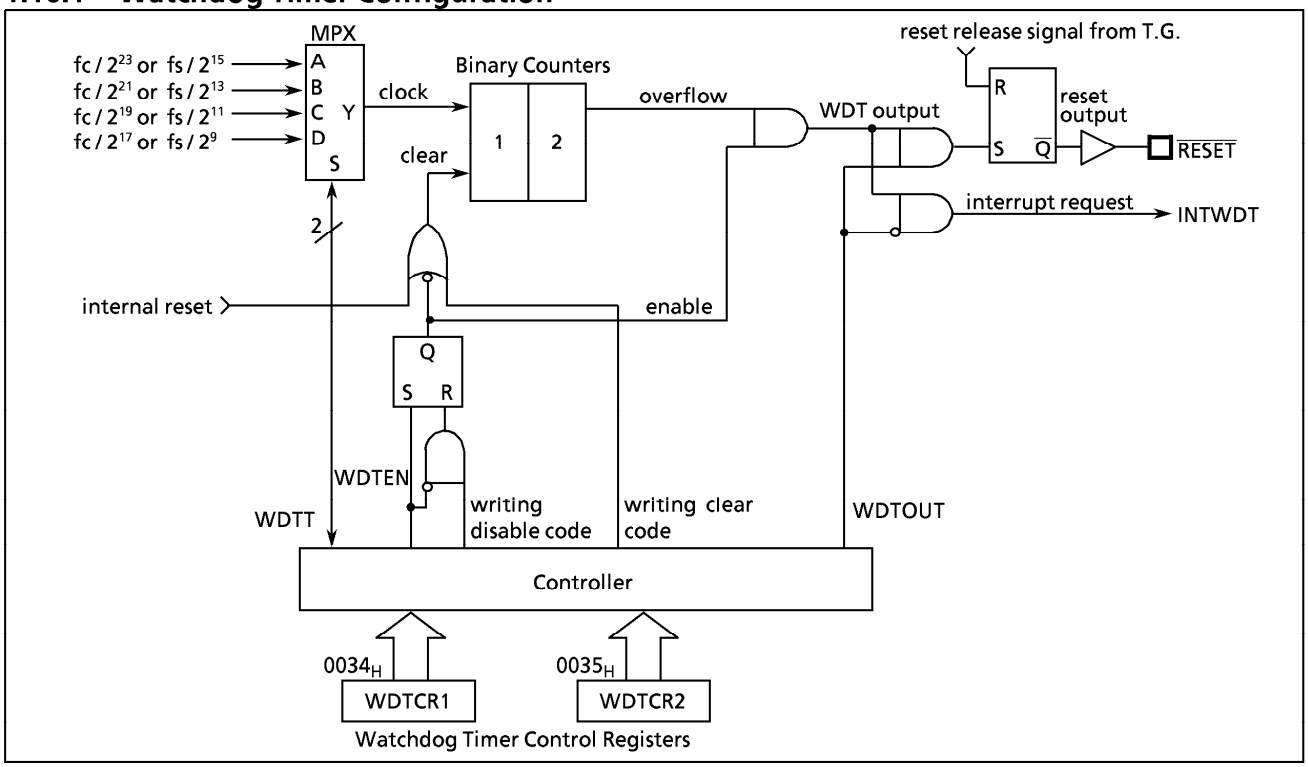


Figure 1-27. Watchdog Timer Configuration

1.10.2 Watchdog Timer Control

Figure 1-28 shows the watchdog timer control registers (WDTCR1, WDTCR2). The watchdog timer is automatically enabled after reset.

(1) Malfunction detection methods using the watchdog timer

The CPU malfunction is detected as follows.

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

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

If the CPU 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, a reset is generated, which drives the $\overline{\text{RESET}}$ pin low to reset the internal hardware and the external circuits. 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 WDT          :                            (always clear immediately after changing WDTT)
detection time      :
                    LD    (WDTCR2), 4EH          ; Clears the binary counters
Within 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 or 2^{17} / fs [s] 01 : 2^{23} / fc or 2^{15} / fs 10 : 2^{21} / fc or 2^{13} / fs 11 : 2^{19} / fc or 2^{11} / fs	
WDTOUT	Watchdog timer output select	0 : Interrupt request 1 : Reset output	

Note 1 : WDTOUT cannot be set to "1" by program after clearing WDTOUT to "0".
*Note 2 : fc ; High-frequency clock [Hz] fs ; Low-frequency clock [Hz] * ; don't care*
Note 3 : WDTCR1 is a write-only register and must not be used with any of read-modify-write instructions.
Note 4 : 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.

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*

Figure 1-28. Watchdog Timer Control Registers

Table 1-4. Watchdog Timer Detection Time

Operating mode			Detection time	
NORMAL1	NORMAL2	SLOW	At fc = 8 MHz	At fs = 32.768 kHz
2^{25} / fc [s]	2^{25} / fc, 2^{17} / fs	2^{17} / fs	4.194 s	4 s
2^{23} / fc	2^{23} / fc, 2^{15} / fs	2^{15} / fs	1.048 ms	1 s
2^{21} / fc	2^{21} / fc, 2^{13} / fs	——	262.1 ms	250 ms
2^{19} / fc	2^{19} / fc, 2^{11} / fs	——	65.5 ms	62.5 ms

(2) Watchdog Timer Enable

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

Example : Enables watchdog timer

```
LD        (WDTCR1), 00001000B        ; WDTEN←1
```

(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". The watchdog timer is halted temporarily in STOP mode (including warm-up) and IDLE mode, and restarts automatically after STOP or IDLE mode is released.

During disabling the watchdog timer, the binary counters are cleared to "0".

Example : Disables watchdog timer

```
LDW     (WDTCR1) , 0B101H        ; WDTEN←0, WDTCR2←disable code
```

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 non-maskable 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, 043FH                ; Sets the stack pointer
LD      (WDTCR1) , 00001000B     ; WDTOUT←0
```

1.10.4 Watchdog Timer Reset

If the watchdog timer output becomes active, a reset is generated, which drives the $\overline{RESET}$ pin (sink open drain output with pull-up registor) low to reset the internal hardware and the external circuits. The reset output time is 2^{20}/fc [s] (131 ms @ fc = 8 MHz). The high-frequency clock oscillator also turns on when a watchdog timer reset is generated in SLOW mode.

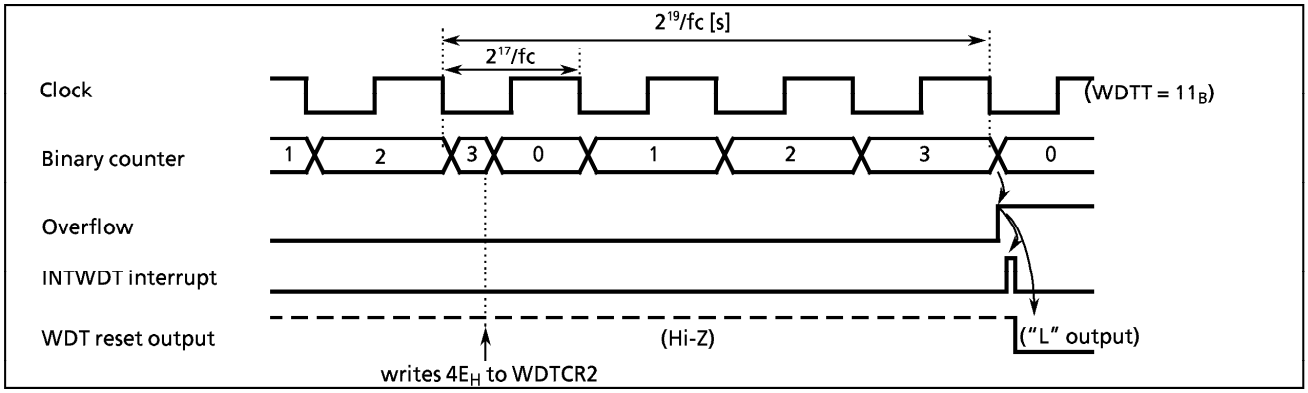


Figure 1-29. Watchdog Timer Interrupt / Reset

1.11 Reset Circuit

The TLCS-870 series has four types of reset generation procedures: an external reset input, an address-trap-reset, a watchdog timer reset and a system-clock-reset. Table 1-5 shows on-chip hardware initialization by reset action.

The internal source reset circuit (watchdog timer reset, address trap reset, and system clock reset) is not initialized when power is turned on. Thus, output from the $\overline{RESET}$ pin may go low (2^{20}/fc [s] 131 ms at 8 MHz) when power is turned on.

Table 1-5. Initializing Internal Status by Reset Action

On-chip Hardware		Initial Value	On-chip Hardware	Initial Value
Program counter	(PC)	$(FFFF_H) \cdot (FFFE_H)$	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 I/O ports	Refer to I/O port circuitry
Interrupt individual enable flags	(EF)	0	Control registers	Refer to each of control register
Interrupt latches	(IL)	0		

1.11.1 External Reset Input

When the $\overline{RESET}$ pin is held at low 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 $\overline{RESET}$ pin input goes high, the reset operation is released and the program execution starts at the vector address stored at addresses $FFFE_H$ - $FFFF_H$.

The $\overline{RESET}$ pin contains a Schmitt trigger (hysteresis) with an internal pull-up resistor. A simple power-on-reset can be applied by connecting an external capacitor and a diode.

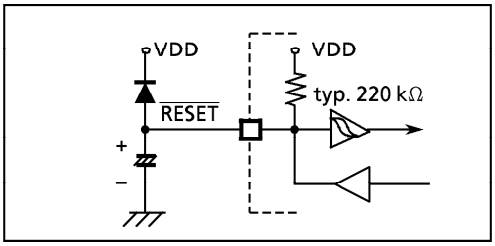


Figure 1-30. Simple Power-on-Reset Circuitry

1.11.2 Address-Trap-Reset

If the CPU malfunction occurs and an attempt is made to fetch an instruction from the RAM or the SFR area, an address-trap-reset will be generated. Then, the $\overline{RESET}$ pin output will go low. The reset time is 2^{20}/fc [s] (131ms at 8MHz).

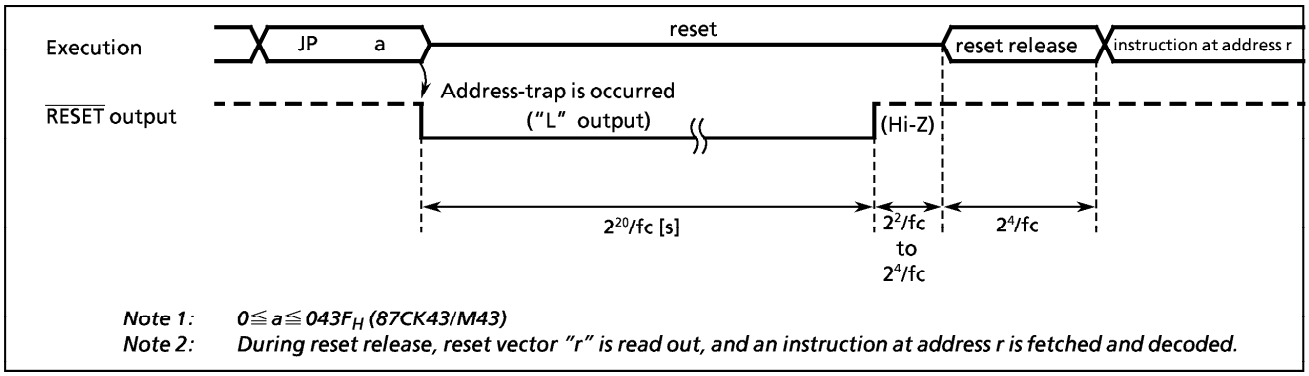


Figure 1-31. Address-Trap-Reset

1.11.3 Watchdog Timer Reset

Refer to Section "1.10 Watchdog Timer".

1.11.4 System-Clock-Reset

Clearing both XEN and XTEN (bits 7 and 6 in SYSCR2) to "0" stops both high-frequency and low-frequency oscillation, and causes the MCU to deadlock. This can be prevented by automatically generating a reset signal whenever XEN = XTEN = 0 is detected to continue the oscillation. Then, the $\overline{RESET}$ pin output goes low from high-impedance. The reset time is 2^{20}/fc [s] (131ms at 8MHz).

2. ON-CHIP PERIPHERALS FUNCTIONS

2.1 Special Function Registers (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_H – $003F_H$.
Figure 2-1 shows the 87CK43/M43 SFRs.

Address	Read	Write
0000_H	P0 port	
01	P1 Port	
02	P2 Port	
03	reserved	
04	reserved	
05	reserved	
06	P6 Port	
07	P7 Port	
08	reserved	
09	reserved	
0A	–	P0CR (P0 I/O control)
0B	–	P1CR (P1 I/O control)
0C	–	P6CR (P6 I/O control)
0D	–	P7CR (P7 I/O control)
0E	ADCCR (A/D converter control)	
0F	ADCDR (A/D conv. result)	–
10	–	$TREG1_L$ (Timer register 1)
11	–	$TREG1_H$
12	reserved	
13	reserved	
14	–	TC1CR (TC1 control)
15	–	TC2CR (TC2 control)
16	–	$TREG2_L$ (Timer register 2)
17	–	$TREG2_H$
18	TREG3A (Timer register 3A)	
19	TREG3B (Timer register 3B)	–
1A	–	TC3CR (TC3 control)
1B	–	TREG4 (Timer register 4)
1C	–	TC4CR (TC4 control)
1D	reserved	
1E	reserved	
1F	reserved	

Address	Read	Write
0020_H		SBICR1 (SBI control1)
21	SBIDBR (SBI data buffer)	
22		I2CAR (I^2C bus address)
23	SBISR (SBI status)	SBICR2 (SBI control2)
24	reserved	
25	PWMSR (PWM status)	PWMCR (PWM control)
26		PWMDBR (PWM data buffer)
27	reserved	
28	reserved	
29	reserved	
2A	reserved	
2B	reserved	
2C	reserved	
2D	reserved	
2E	reserved	
2F	reserved	
30	reserved	
31	reserved	
32	reserved	
33	reserved	
34	–	WDTCR1 (WDT control)
35	–	WDTCR2
36	–	TBTCR (TBT / TG / DVO control)
37	–	EINTCR (External interrupt control)
38	SYSCR1 (System control)	
39	SYSCR2	
3A	EIR_L (Interrupt enable register)	
3B	EIR_H	
3C	IL_L (Interrupt latch)	
3D	IL_H	
3E	reserved	
3F	PSW (Program status word)	RBS (Register bank selector)

Special Function Registers

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, SBIDBR 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 : SBI : Serial Bus Interface
PWM : Pulse Width Modulation

Figure 2-1. SFR

2.2 I/O Ports

The 87CK43/M43 have 5parallel input/output ports (35pins) each as follows:

	Primary Function	Secondary Functions
Port P0	8-bit I/O port	———
Port P1	8-bit I/O port	external interrupt input, timer/counter input, and divider output
Port P2	3-bit I/O port	low-frequency resonator connections, external interrupt input, and STOP mode release signal input
Port P6	8-bit I/O port	analog input, external interrupt input, timer/counter input, pulse width modulation output
Port P7	8-bit I/O port	serial bus interface input/output

Each output port contains a latch, which holds the output data. All input ports do not have latches, so the external input data should 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.

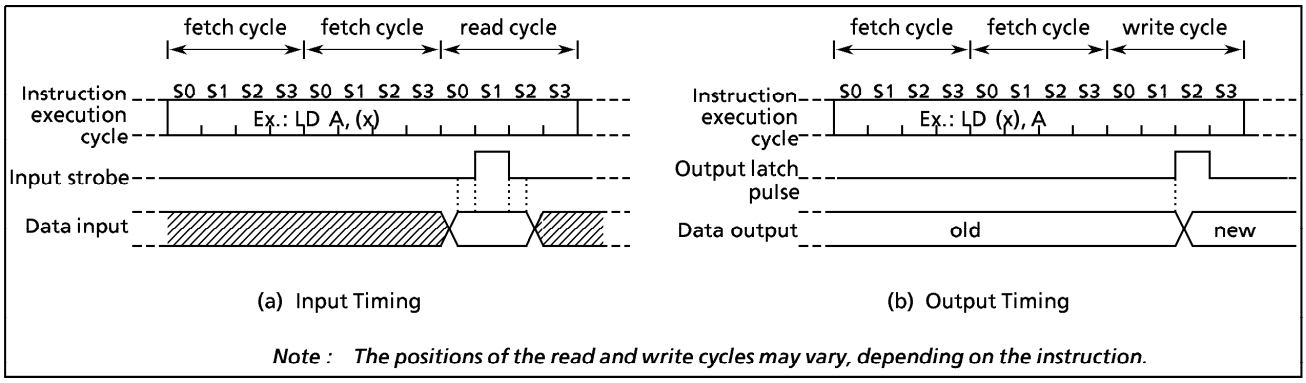


Figure 2-2. Input/Output Timing (Example)

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

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

2.2.1 Port P0 (P07 - P00)

Port P0 is an 8-bit general-purpose input/output port which can be configured as either an input or an output in one-bit unit under software control. Input/output mode is specified by the corresponding bit in the port P0 input/output control register (P0CR). Port P0 is configured as an input if its corresponding P0CR bit is cleared to "0", and as an output if its corresponding P0CR bit is set to "1".
During reset, P0CR is initialized to "0", which configures port P0 as input. The P0 output latches are also initialized to "0". Data is written into the output latch regardless of the P0CR contents. Therefore initial output data should be written into the output latch before setting P0CR.

Note : Input mode port is read the state of input pin.
When input/output mode is used to mixed, the contents of input mode port may be changed by executing bit manipulation instructions.

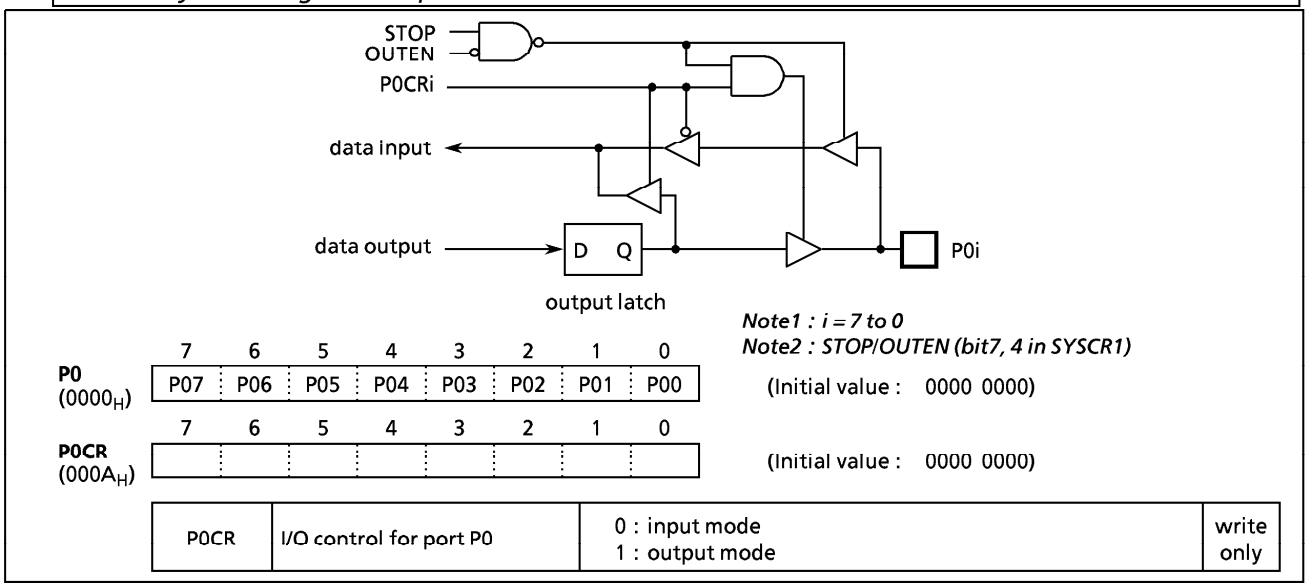


Figure 2-3. Port P0 and P0CR

Example : Setting the upper 4 bits of port P0 as an input port and the lower 4 bits as an output port (Initial output data are 1010_B).

```
LD      (P0), 00001010B     ; Sets initial data to P0 output latches
LD      (P0CR), 00001111B   ; Sets the port P0 input/output mode
```

2.2.2 Port P1 (P17 - P10)

Port P1 is an 8-bit input/output port which can be configured as an input or an output in one-bit unit under software control. Input/output mode is specified by the corresponding bit in the port P1 input/output control register (P1CR). Port P1 is configured as an input if its corresponding P1CR bit is cleared to "0", and as an output if its corresponding P1CR bit is set to "1". During reset, the P1CR is initialized to "0", which configures port P1 as an input . The P1 output latches are also initialized to "0". Data is written into the output latch regardless of P1CR contents. Therfore initial output data should be written into the output latch before setting P1CR. 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 P10 ($\overline{INT0}$) can be configured as either an I/O port or an external interrupt input with INT0EN (bit 6 in EINTCR). During reset, pin P10 ($\overline{INT0}$) is configured as an input port P10.

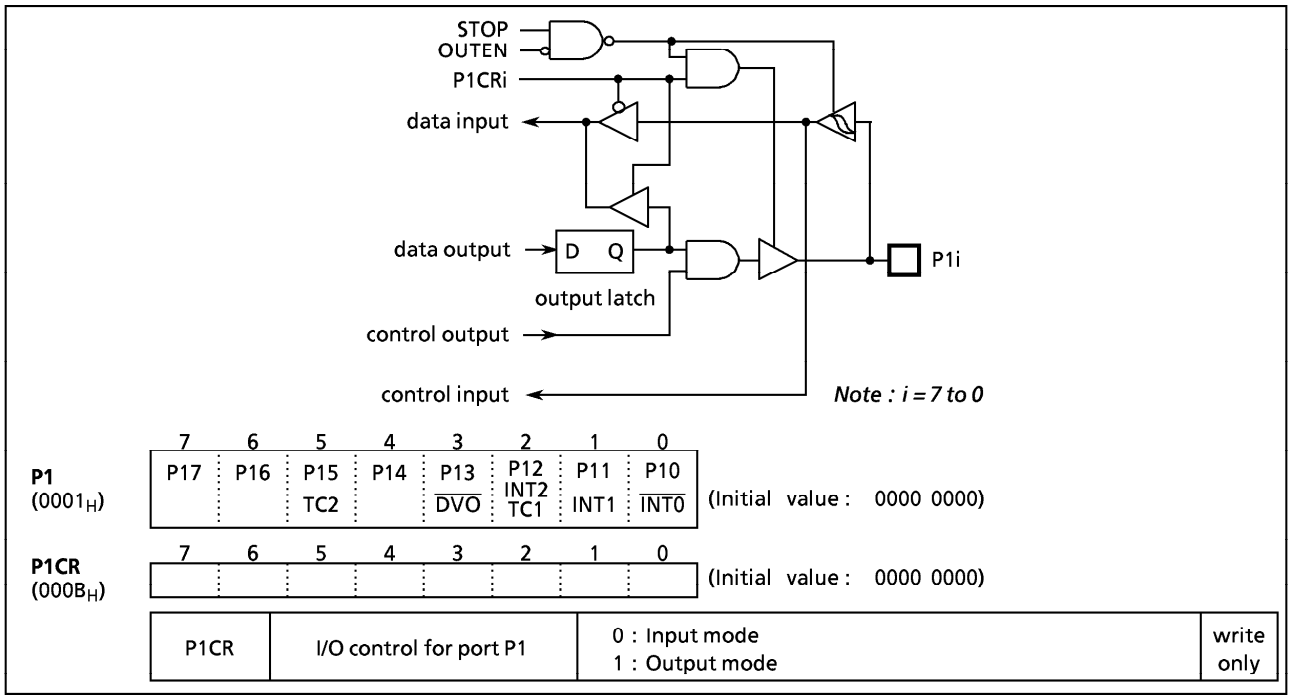


Figure 2-4. Port P1 and P1CR

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
```

Note : Input mode port is read the state of input pin.
When input/output mode is used to mixed, the contents of input mode port may be changed by executing bit manipulation instructions.

2.2.3 Port P2 (P22 - P20)

Port P2 is a 3-bit input/output port. It is also used as an external interrupt input, and STOP mode releases signal input, and low-frequency crystal connection pins. When used as an input port, or a secondary function pin, the output latch should be set to "1". During reset , the output latches are initialized to "1".

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

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

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

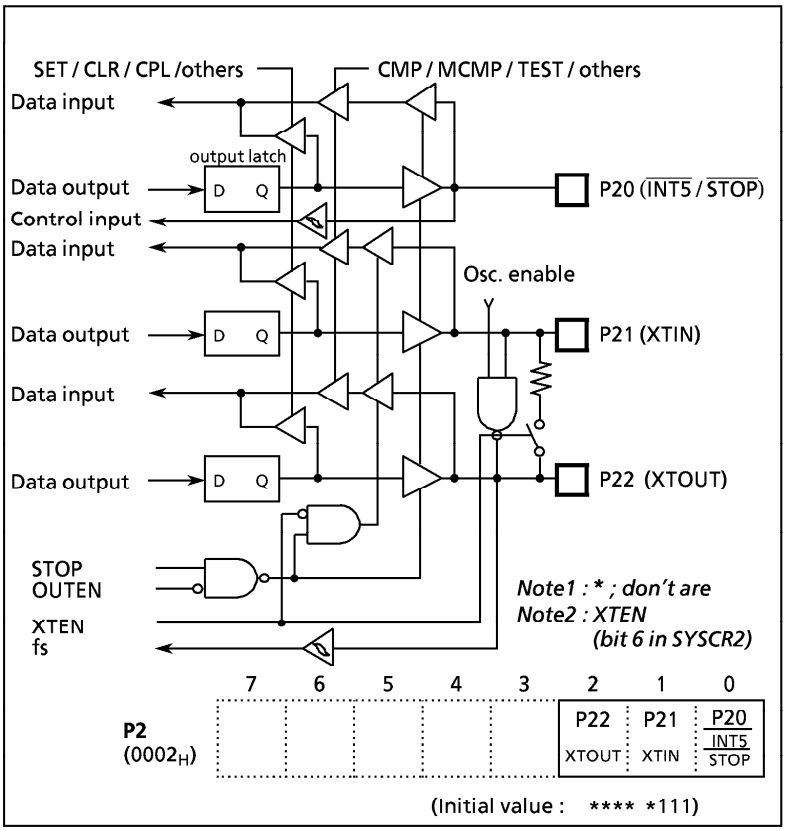


Figure 2-5. Port P2

2.2.4 Port P6 (P67 - P60)

Port P6 is an 8-bit input/output port which can be configured as an input or an output in one-bit unit under software control. Input or output mode is selected by the corresponding bit in the port P6 input/output control register (P6CR).

The P67 is also used as an output for the $\overline{PWM0}$. The P66 is also used as an output for the $\overline{PWM1}$, an external interrupt input, and a timer/counter input. The P65~P60 is also used as an analog input (AIN5~AIN0) for the A/D converter. When used as an analog input, AINDS (bit 4 in the ADCCR) must be cleared to "0" and its corressponding P6CR bit must be set to "1". In this case, unuse pin as analog input is configured as only input port.

During reset, AINDS is initialized to "0" and there bits of P6CR are initialized to "0011 1111B", which configures port P67, P66 as input port and port P65~P60 as analog input. The P6 output latches are initialized to "1". Data is written into the output latch regardless of the P6CR contents. Therefore initial output data should be written into the output latch before setting P6CR.

Note :	*Input mode port is read the state of input pin.* *When input/output mode is used to mixed, the contents of input mode port may be changed by executing bit manipulation instructions.*

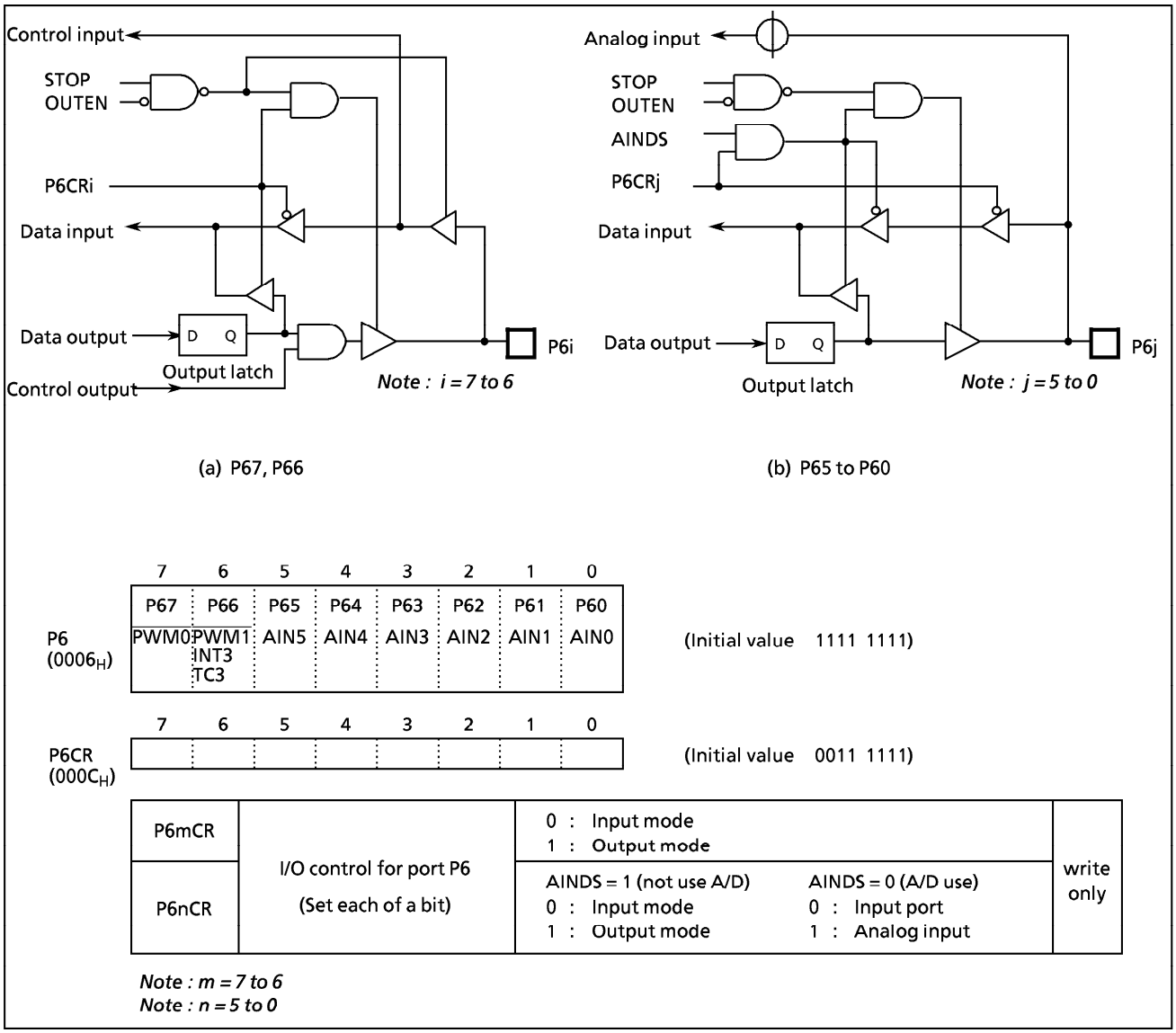


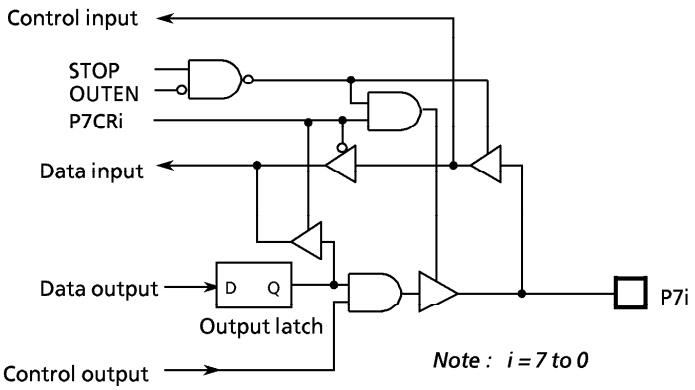
Figure 2-6. Port P6 and P6CR

2.2.5 Port P7 (P77 - P70)

Port P7 is an 8-bit general-purpose input/output port which can be configured as either input or output in one-bit unit under software control. Input/output mode is specified by the corresponding bit in the port P7 input/output control register (P7CR). Port P7 is also used as a input/output for the serial bus interface. For example, port P7 is configured as an input if its corresponding P7CR bit is cleared to "0", and as an output if its corresponding bit is set to "1". During reset, P7CR is initialized to "0", which configures port P7 as input. The output latches are initialized to "1". When used as a serial bus interface, the output latch should be set to "1". Set to output mode the P7CR.

Data is written into the output latch regardless of the P7CR contents. Therefore initial output latch before setting P7CR.

Note :	*Input mode port is read the state of input pin.* *When input/output mode is used to mixed, the contents of input mode port may be changed by executing bit manipulation instructions.*



P7 (0007_H)

7	6	5	4	3	2	1	0
P77	P76	P75	P74	P73	P72	P71	P70
	SO1	SDA1 SI1	SCL1 $\overline{SCK1}$		SO0	SDA0 SI0	SCL0 $\overline{SCK0}$

(Initial value 1111 1111)

P7CR ($000D_H$)

7	6	5	4	3	2	1	0

(Initial value 0000 0000)

P7CR	I/O control for port P7	0 : Input mode 1 : Output mode	write only

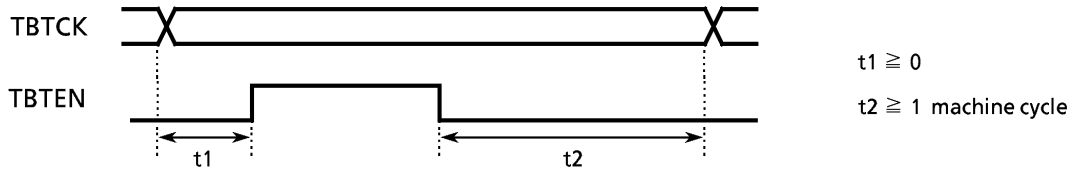
Figure 2-7. Port and P7CR

2.3 Time Base Timer (TBT)

The time base timer generates time base for key scanning, dynamic displaying, etc. It also provides a time base timer interrupt (INTTBT). The time base timer is controlled by the control register (TBTCR) shown in Figure 2-9.

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

The interrupt frequency (TBTCK) must be selected with the time base timer disabled (both frequency selection and enabling can be performed simultaneously).



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

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

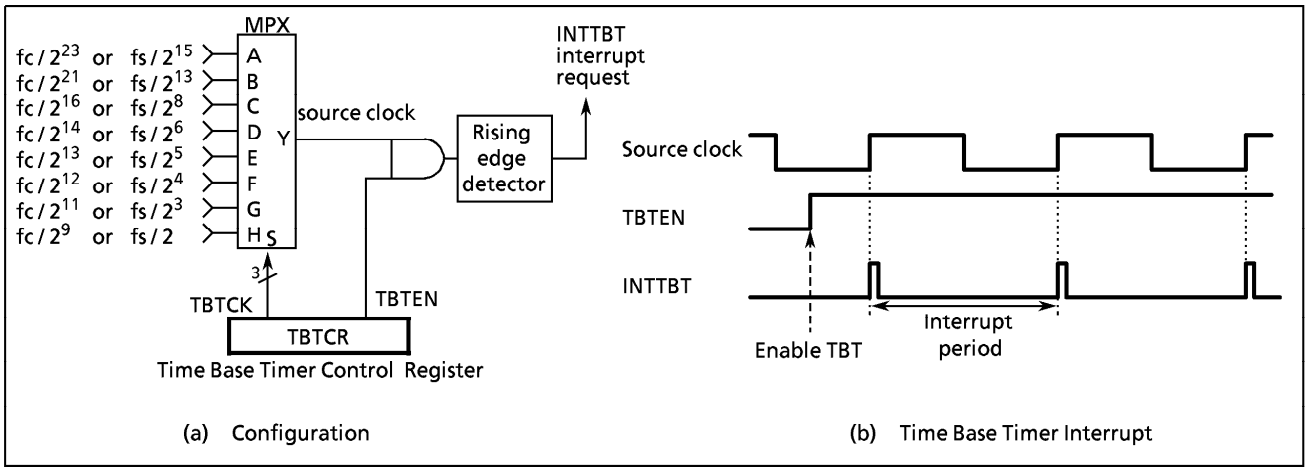


Figure 2-8. Time Base Timer

TBTCR (0036_H)

7	6	5	4	3	2	1	0
(DVOEN)	(DVOCK)		(DV7CK)	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} or fs / 2^{15} [Hz] 001 : fc / 2^{21} or fs / 2^{13} 010 : fc / 2^{16} or fs / 2^{8} 011 : fc / 2^{14} or fs / 2^{6} 100 : fc / 2^{13} or fs / 2^{5} 101 : fc / 2^{12} or fs / 2^{4} 110 : fc / 2^{11} or fs / 2^{3} 111 : fc / 2^{9} or fs / 2	

*Note 1 : fc ; High-frequency clock [Hz], fs ; Low-frequency clock [Hz], * ; don't care*
Note 2 : TBTCR is a write-only register and must not be used with any of read-modify-write instructions.

Figure 2-9. Time Base Timer and Divider Output Control Register

Table 2-1. Time Base Timer Interrupt Frequency

TBTCK	NORMAL1/2, IDLE1/2 mode		SLOW, SLEEP mode	Interrupt Frequency	
	DV7CK = 0	DV7CK = 1		At fc = 8 MHz	At fs = 32.768 kHz
000	fc / 2^{23}	fs / 2^{15}	fs / 2^{15}	0.95 Hz	1 Hz
001	fc / 2^{21}	fs / 2^{13}	fs / 2^{13}	3.81	4
010	fc / 2^{16}	fs / 2^{8}	–	122.07	128
011	fc / 2^{14}	fs / 2^{6}	–	488.28	512
100	fc / 2^{13}	fs / 2^{5}	–	976.56	1024
101	fc / 2^{12}	fs / 2^{4}	–	1953.12	2048
110	fc / 2^{11}	fs / 2^{3}	–	3906.25	4096
111	fc / 2^{9}	fs / 2	–	15625	16384

2.4 Divider Output ($\overline{DVO}$)

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 ($\overline{DVO}$). The P13 output latch should be set to "1" and then the P13 should be configured as an output mode.
Divider output circuit is controlled by the control register (TBTCR) shown in Figure 2-10.
Note that TBTCR is a write-only register.

TBTCR (0036_H)

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

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

DVOEN	Divider output enable/disable	0 : Disable 1 : Enable	Write only
DVOCK	Divider output ($\overline{DVO}$) frequency selection	00 : $fc/2^{13}$ or $fs/2^5$ [Hz] 01 : $fc/2^{12}$ or $fs/2^4$ 10 : $fc/2^{11}$ or $fs/2^3$ 11 : $fc/2^{10}$ or $fs/2^2$	

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

Figure 2-10. 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

DVOCK	Frequency of Divider Output	At fc = 4.194304 MHz	At fc = 8 MHz	At fs = 32.768 kHz
00	$fc/2^{13}$ or $fs/2^5$	0.512 [kHz]	0.976 [kHz]	1.024 [kHz]
01	$fc/2^{12}$ $fs/2^4$	1.024	1.953	2.048
10	$fc/2^{11}$ $fs/2^3$	2.048	3.906	4.096
11	$fc/2^{10}$ $fs/2^2$	4.096	7.812	8.192

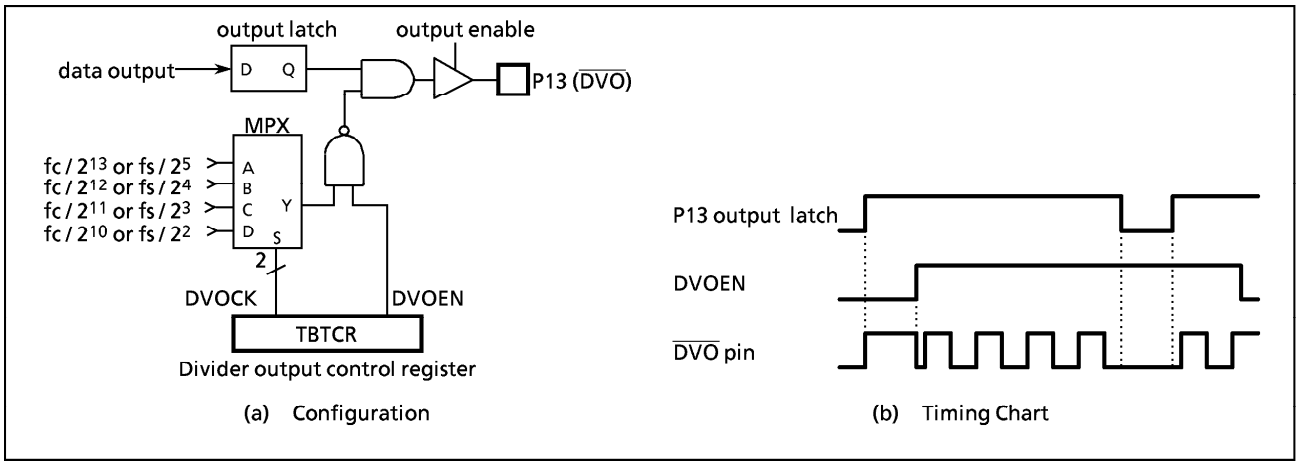


Figure 2-11. Divider Output

2.5 16-bit Timer/Counter 1 (TC1)

2.5.1 Configuration

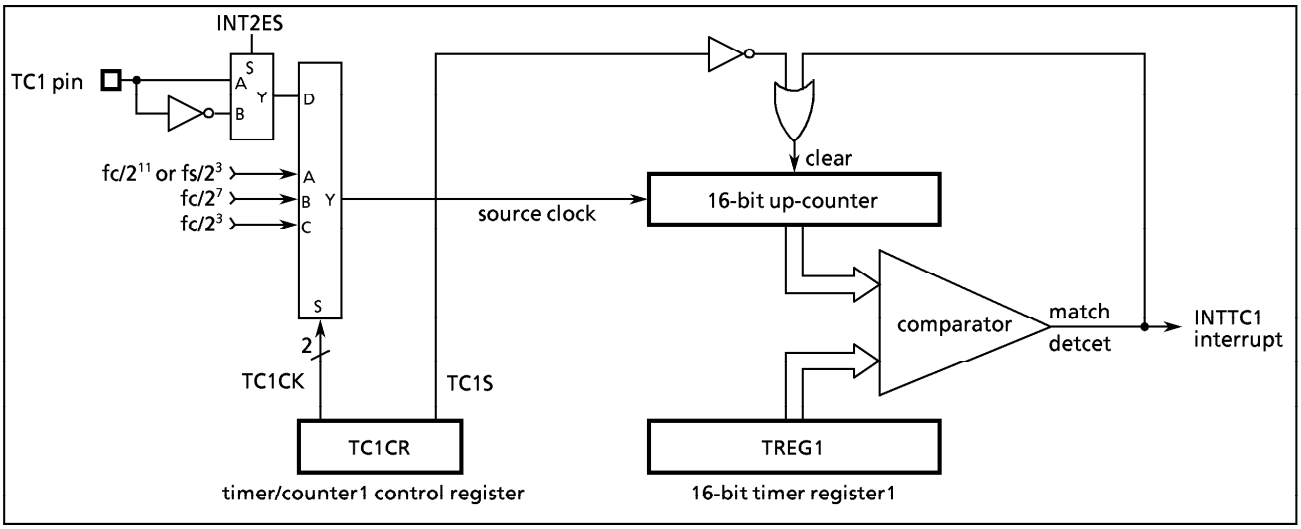


Figure 2-12. Timer/Counter1

2.5.2 Control

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

TREG1 (0010, 0011_H)

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

Write only

TC1CR (0014_H)

"0"	"0"	"0"	TC1S	TC1CK		"0"	"0"

(Initial value ***0 00**)

TC1CK	Timer/counter1 source clock select	00 : internal clock $fc/2^{11}$ or $fs/2^2$ [Hz] 01 : internal clock $fc/2^7$ 10 : internal clock $fc/2^3$ 11 : external clock (TC1 pin input)	Write only
TC1S	Timer/counter1 start control	0 : stop & counter clear 1 : command start	

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

Note 2 : Always write "0" to bit7, 6, 5, 1 and 0 in TC1CR.

Note 3 : Writing to the low-byte of the timer registers ($TREG1_L$), the comparison is inhibited until the high-byte ($TREG1_H$) is written.
After writing to the high-byte, any match during 1 machine cycle (instruction execution cycle) is ignored.

Note 4 : Set the mode, source clock and edge (INT2ES) when TC1 stops (TC1S = 0).

Note 5 : Values to be loaded to timer registers must satisfy the following condition.
TREG1 > 0

Note 6 : TC1CR is a write-only resister, which cannot be accessed by any read-modify-write instruction such as bit operate, etc.

Figure 2-13. Timer Registers and TC1 Control Register

2.5.3 Function

Timer/counter 1 has two operating modes: timer, 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	
NORMAL1/2, IDLE1/2 modes		SLOW, SLEEP modes				
DV7CK = 0	DV7CK = 1		At fc = 8 MHz	At fs = 32.768 kHz	At fc = 8 MHz	At fs = 32.768 kHz
$fc / 2^3$ [Hz]	$fc / 2^3$ [Hz]	–	1 μs	–	65.5 ms	–
$fc / 2^7$	$fc / 2^7$	–	16 μs	–	1.0 s	–
$fc / 2^{11}$	$fs / 2^3$	$fs / 2^3$ [Hz]	256 μs	244.14 μs	16.7 s	16.0 s

Example 1 : Sets the timer mode with source clock $fs/2^3$[Hz] and generates an interrupt 1 [s]. later (at fs = 32.768 kHz).

```
LDW     (TREG1), 1000H          ; Sets the timer register (1s ÷ 2³ / fs = 1000H)
SET     (EIRL), EF4             ; INTTC1 interrupt enable
EI
LD      (TC1CR), 00010000B      ; Starts TC1
```

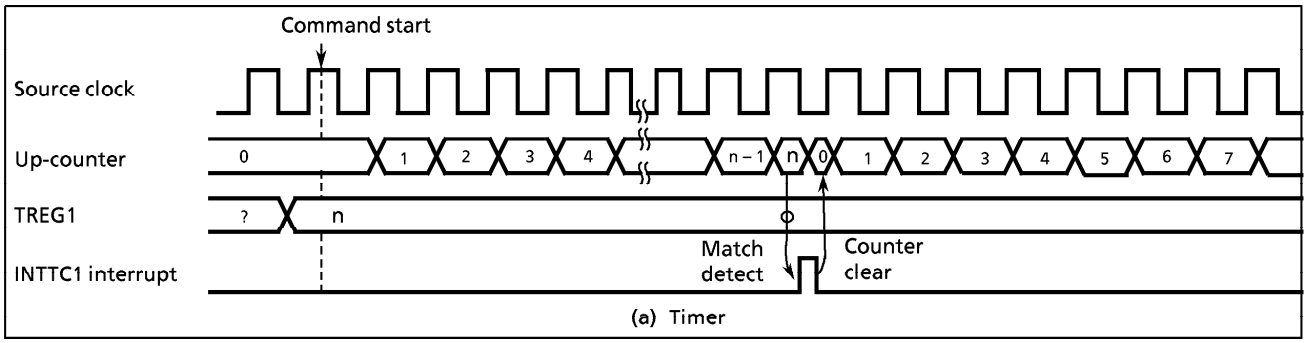


Figure 2-14. Timer Mode Timing Chart

(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 INT2ES in EINTCR. 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. Counting up is resumed every edge of TC1 pin input after the counter is cleared. The maximum applied frequency is fc/2^4 [Hz] in NORMAL1/2 or IDLE1/2 mode and fs/2^4 [Hz] in SLOW or SLEEP mode. Two or more machine cycles are required for both the high and low levels of the pulse width. The contents of up-counter can not be read.

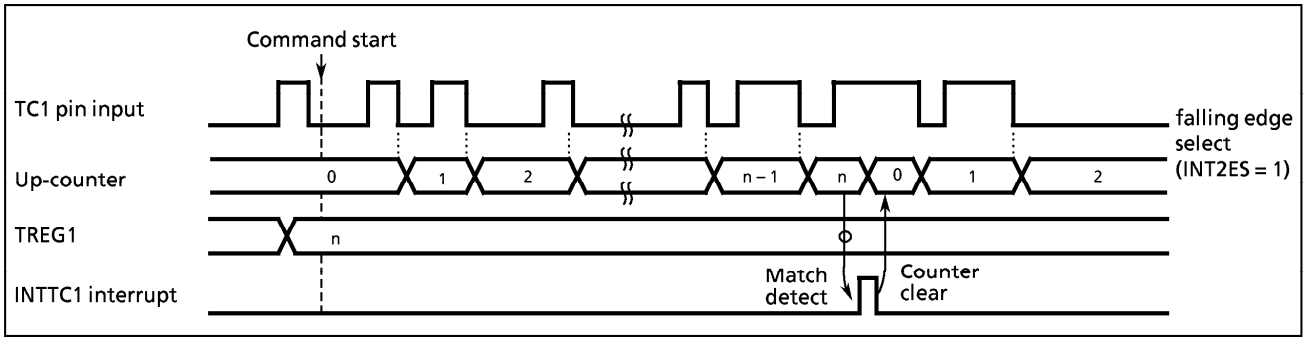


Figure 2-15. Event Counter Mode Timing Chart (INT2ES = 1)

2.6 16-bit Timer/Counter 2 (TC2)

2.6.1 Configuration

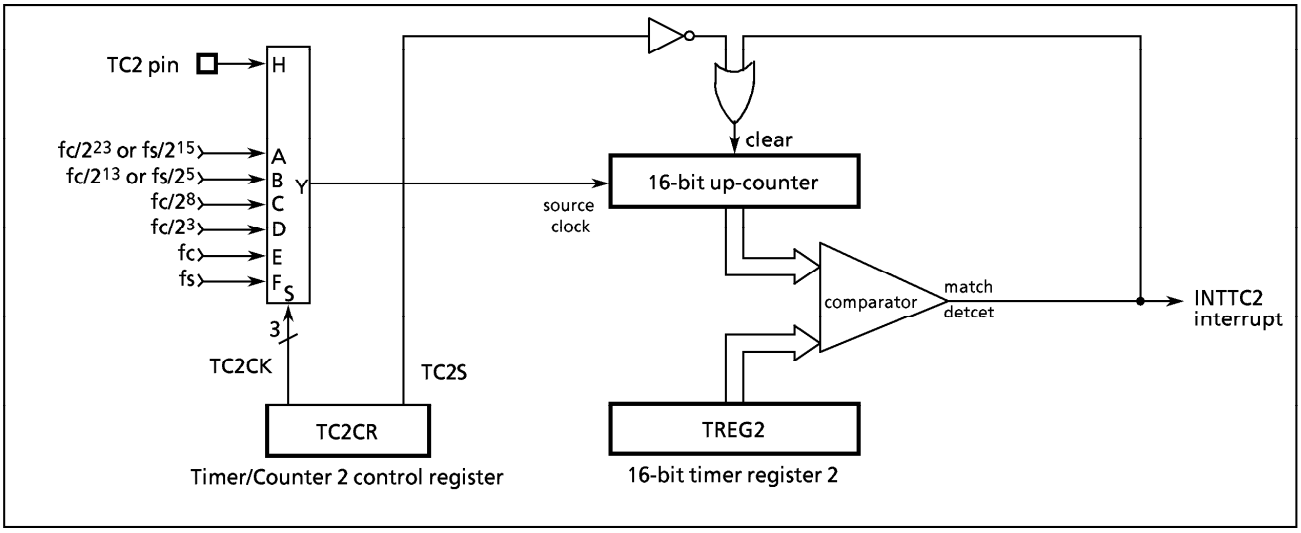


Figure 2-16. Timer/Counter 2 (TC2)

2.6.2 Control

The timer/counter 2 is controlled by a timer/counter 2 control register (TC2CR) and a 16-bit timer register 2 (TREG2). Reset does not affect TREG2.

TREG2 ($0016, 0017_H$)

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
$TREG2_H$ (0017_H)								$TREG2_L$ (0016_H)							

write only

TC2CR (0015_H)

7	6	5	4	3	2	1	0
"0"	"0"	TC2S	TC2CK			"0"	"0"

(Initial value : **00 00**)

TC2CK	Timer/counter 2 source clock select	000 : Internal clock $fc/2^{23}$ or $fs/2^{15}$ [Hz] 001 : 〃 $fc/2^{13}$ or $fs/2^{5}$ 010 : 〃 $fc/2^{8}$ 011 : 〃 $fc/2^{3}$ 100 : 〃 fc (Note 5) 101 : 〃 fs 110 : Reserved 111 : External clock (TC2 pin input)	write only
TC2S	Timer/counter 2 start control	0 : Stop and counter clear 1 : Start	

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

Note 2 : Always write "0" to bit7, 6, 1 and 0 in TC2CR.

Note 3 : When writing to the low-byte of timer register 2 ($TREG2_L$), the comparison is inhibited until the high-byte ($TREG2_H$) is written.
After writing to the high-byte, any match during 1 machine cycle (instruction execution cycle) is ignored.

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

Note 5 : Values to be loaded to the timer register must satisfy the following condition.
TREG2 > 0 ($TREG2_{15\ to\ 11}$>0 when warm-up).

Note 6 : "fc" can be selected as the source clock only in the timer mode during the SLOW mode.

Note 7: TC2CR and TREG2 are write-only registers and must not be used with any of the read-modify-write instructions.

Figure 2-17. Timer Register 2 and TC2 Control Register

2.6.3 Function

The timer/counter 2 has two operating modes: timer, event counter. Also timer/counter 2 is used for warm-up when switching from SLOW mode to NORMAL2 mode.

(1) **Timer** Mode

In this mode, the internal clock is used for counting up. The contents of TREG2 are compared with the contents of up-counter. If a match is found, a timer/ counter 2 interrupt (INTTC2) is generated, and the counter is cleared. Counting up is resumed after the counter is cleared.
Also, when fc is selected as the source clock during SLOW mode, the lower 11 bits of TREG2 are ignored and an INTTC2 interrupt is generated by matching the upper 5 bits. Thus, in this case, only the $TREG2_H$ setting is necessary.

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

Source clock				Resolution		Maximum time setting	
NORMAL1/2, IDLE1/2 mode		SLOW mode	SLEEP mode				
DV7CK = 0	DV7CK = 1			At fc = 8 MHz	At fs = 32.768 kHz	At fc = 8 MHz	At fs = 32.768 kHz
$fc / 2^{23}$[Hz]	$fs / 2^{15}$[Hz]	$fs / 2^{15}$ [Hz]	$fs / 2^{15}$ [Hz]	1.05 s	1 s	19.1 h	18.2 h
$fc / 2^{13}$	$fs / 2^{5}$	$fs / 2^{5}$	$fs / 2^{5}$	1.02 ms	0.98 ms	1.1 min	1 min
$fc / 2^{8}$	$fc / 2^{8}$	–	–	32 μs	–	2.1 s	–
$fc / 2^{3}$	$fc / 2^{3}$	–	–	1 μs	–	65.5 ms	–
–	–	fc (Note)	–	125 ns	–	7.9 ms	–
fs	fs	–	–	–	30.5 μs	–	2 s

Note : "fc" can be used only in the timer mode.

Example : Sets the timer mode with source clock $fc/2^3$ [Hz] and generates an interrupt every 25 ms (at fc = 8 MHz).

```
LDW     (TREG2), 61A8H          ; Sets TREG2 (25 ms ÷ 2^3/fc = 61A8H)
SET     (EIRH), EF14            ; INTTC2 interrupt enable
EI
LD      (TC2CR), 00101100B      ; Starts TC2
```

(2) **Event Counter** Mode

In this mode, events are counted on the rising edge of the TC2 pin input. The contents of TREG2 are compared with the contents of the up-counter. If a match is found, an INTTC2 interrupt is generated, and the counter is cleared. The maximum frequency applied to the TC2 pin is $fc/2^4$ [Hz] in NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ [Hz] in SLOW or SLEEP mode. Two or more machine cycles are required for both the high and low levels of the pulse width.

Example : Sets the event counter mode and generates an INTT2 interrupt 640 counts later.

```
LDW     (TREG2), 640            ; Sets TREG2
SET     (EIRH), EF14            ; INTTC2 interrupt enable
EI
LD      (TC2CR), 00111100B      ; Starts TC2
```

2.7 8-Bit Timer/Counter 3 (TC3)

2.7.1 Configuration

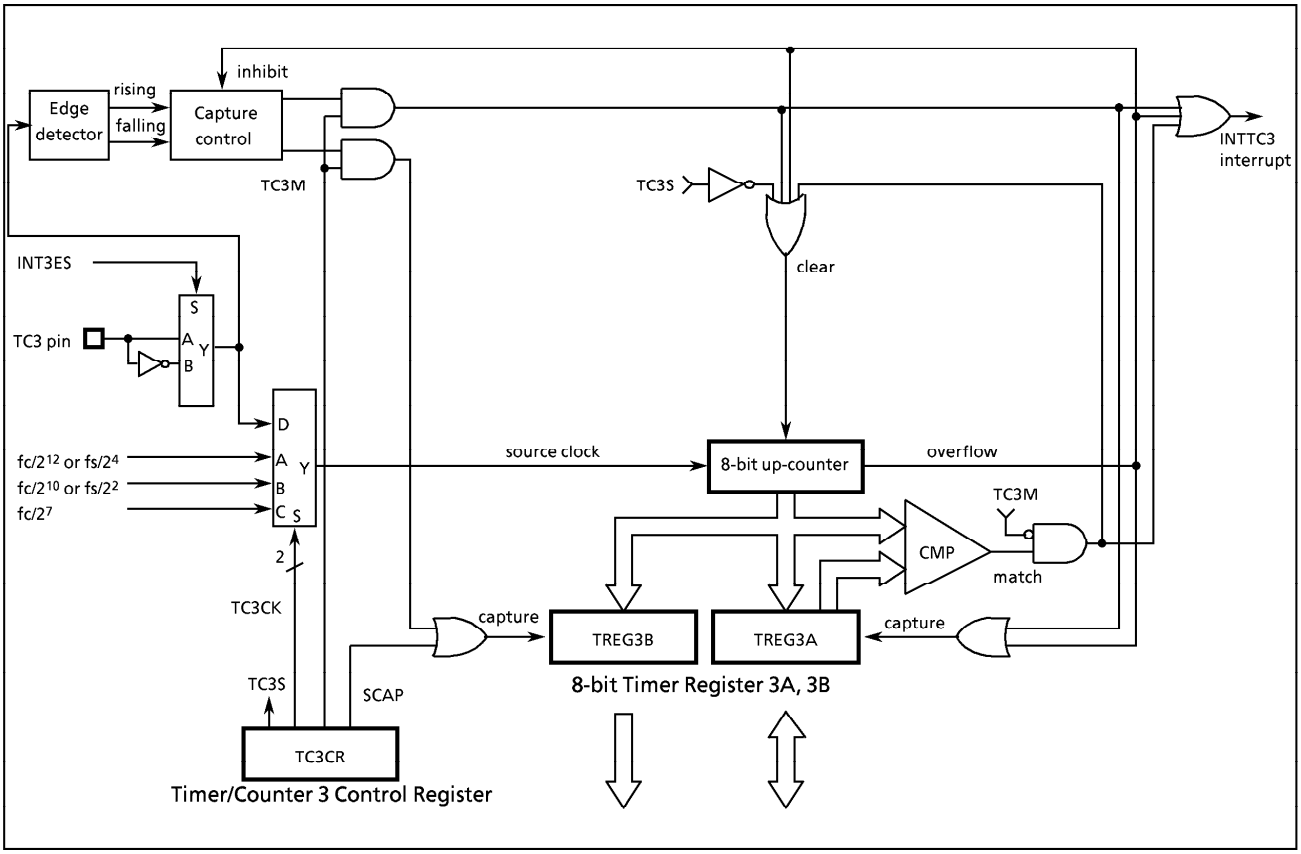


Figure 2-18. Timer/Counter 3

2.7.2 Control

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.

	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 $fc/2^{12}$ or $fs/2^4$ [Hz] 01 : Internal clock $fc/2^{10}$ or $fs/2^2$ 10 : Internal clock $fc/2^7$ 11 : External clock (TC3 pin input)	
TC3S	Timer/counter 3 start select	0 : Stop & clear 1 : Start	
SCAP	Software capture control	0 : – 1 : Software capture	

*Note 1 : fc ; High-frequency clock [Hz] fs ; Low-frequency clock [Hz] * ; don't care*
Note 2 : Set the mode and the source clock when the TC3 stops (TC3S = 0).
Note 3 : Values to be loaded into timer register 3A must satisfy the following condition.
TREG3A > 0 (in the timer/event counter mode)
Note 4 : Software capture can be used in only timer and event counter modes.

Figure 2-19. Timer Register 3A/3B and TC3 Control Register

2.7.3 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.

Source clock			Resolution		Maximum setting time	
NORMAL1/2, IDLE1/2 mode		SLOW, SLEEP mode	At fc = 8 MHz	At fs = 32.768 kHz	At fc = 8 MHz	At fs = 32.768 kHz
DV7CK = 0	DV7CK = 1					
$fc / 2^{12}$ [Hz]	$fs / 2^4$ [Hz]	$fs / 2^4$ [Hz]	512 μs	488.28 μs	131.1 ms	124.5 ms
$fc / 2^{10}$	$fs / 2^2$	$fs / 2^2$	128 μs	122.07 μs	32.6 ms	31.1 ms
$fc / 2^7$	–	–	16 μs	–	4.1 ms	–

Table 2-5. Source Clock (Internal Clock) for Timer Counter 3

(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). 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 NORMAL1/2 or IDLE1/2 mode, and $fs/2^4$ [Hz] in SLOW or SLEEP 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 50 Hz pulses to the TC3 pin.

```
LD  (TREG3A) , 19H          ;  0.5 s ÷ 1 / 50 = 25 = 19H
SET (EIRH), EF11            ;  INTTC3 interrupt enable
EI
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 the counter is cleared 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.

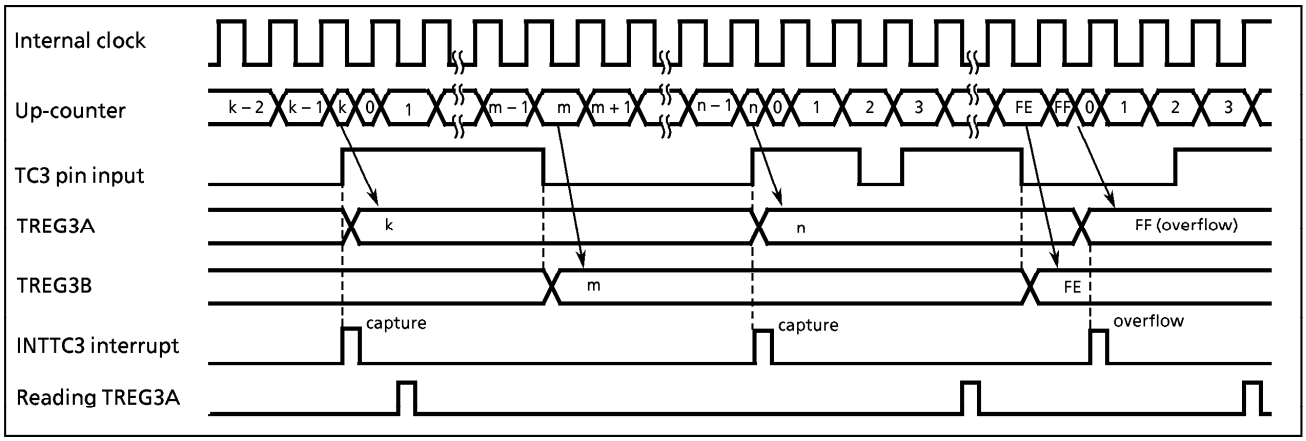


Figure 2-20. Timing Chart for Capture Mode (INT3ES = 0)

2.8 8-bit Timer/Counter (TC4)

2.8.1 Configuration

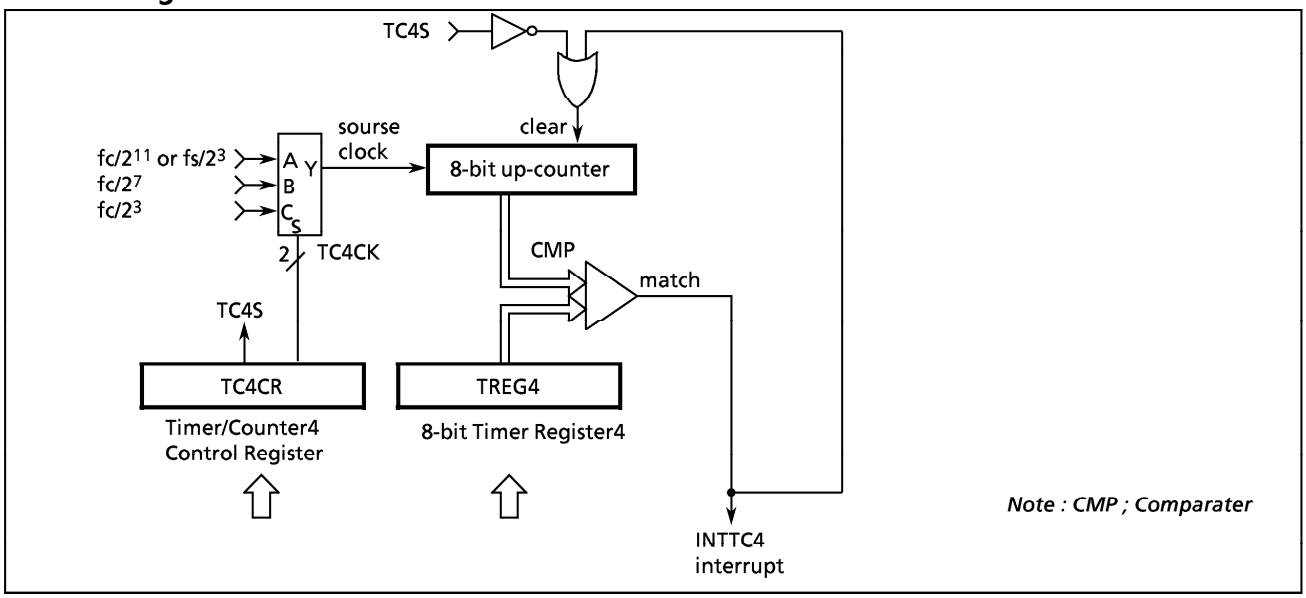


Figure 2-21. Timer/Counter 4

2.8.2 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 ($001C_H$)	"1"	"1"	"0"	TC4S	TC4CK		"0"	"0"	(Initial value : ***0 00**)

TC4CK	TC4 source clock select	00 : Internal clock $fc/2^{11}$ or $fs/2^3$[Hz] 01 : Internal clock $fc/2^7$ 10 : Internal clock $fc/2^3$ 11 : reserved	Write only
TC4S	TC4 start control	0 : Stop & counter clear 1 : Start	

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

Note 2 : Set the operating mode, the source clock selection when the TC4 stops (TC4S = 0).

Note 3 : Always write "0" to bits 0, 1 and 5, "1" to bits 6, 7 in T4CR.

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

TREG4 > 0

Figure 2-22. Timer Register 4 and TC4 Control Register

2.8.3 Function

The timer/counter 4 has one operating mode : timer mode.

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 2-6. Source Clock (Internal Clock) for Timer/Counter 4

Source clock			Resolution		Maximum setting time	
NORMAL1/2, IDLE1/2 mode		SLOW, SLEEP mode	At fc = 8 MHz	At fs = 32.768 kHz	At fc = 8 MHz	At fs = 32.768 kHz
DV7CK = 0	DV7CK = 1					
$fc/2^{11}$ [Hz]	$fs/2^3$ [Hz]	$fs/2^3$ [Hz]	256 μs	244 μs	65.3 ms	62.2 ms
$fc/2^7$	–	–	16 μs		4.1 ms	
$fc/2^3$	–	–	1 μs		256 μs	

2.9 Serial Bus Interface (SBI)

The 87CK43/M43 has a 2-channel serial bus interface which employs a clocked-synchronous 8-bit serial bus interface and an I²C bus (a bus system by Philips).
The serial bus interface pins are also used for the P7 port. When used for serial bus interface pins, set the P7 output latches of these pins to "1", and set to input or output. When not used for serial bus interface pins, the P7 port is used as a normal I/O port.

When the 87CK43/M43 is used as master mode, another devices on same bus must be slave mode. Because the 87CK43/M43 serial bus interface (SBI) does not have arbitration function. The data transfer is carried 9 bits (8bits data and 1bit acknowledge.)

2.9.1 Configuration

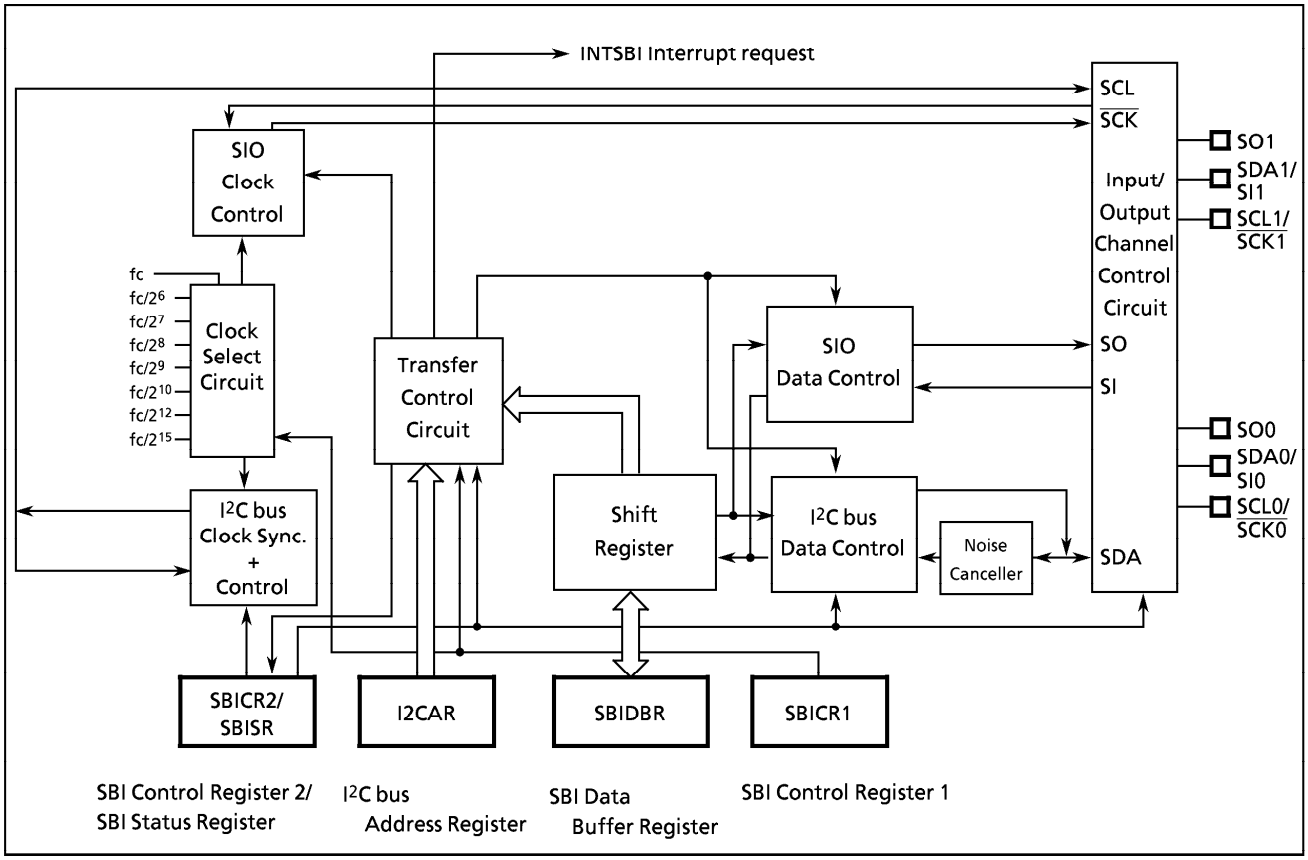


Figure 2-23. Serial Bus Interface (SBI)

2.9.2 Serial Bus Interface (SBI) Control

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

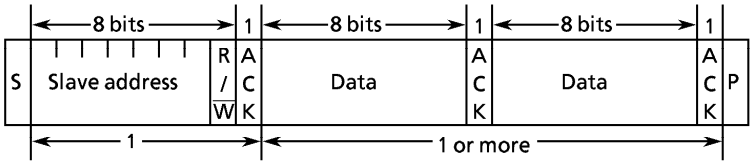
- Serial bus interface control register 1 (SBICR1)
- Serial bus interface control register 2 (SBICR2)
- Serial bus interface data buffer register (SBIDBR)
- I²C bus address register (I2CAR)
- Serial bus interface status register (SBISR)

The above registers differ depending on a mode to be used.
Refer to Section "2.9.4 I²C bus Mode Control" and "2.9.6 Clocked-synchronous 8-bit SIO Mode Control".

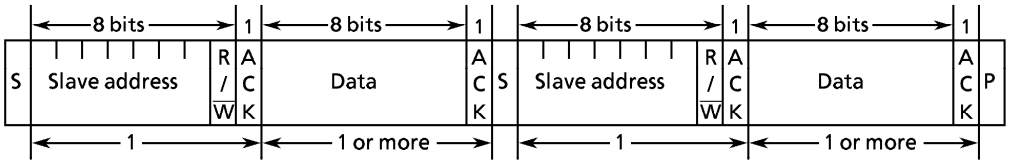
2.9.3 The Data Formats in the I²C bus Mode

The data formats when using the 87CK43/M43 in the I²C bus mode are shown below.

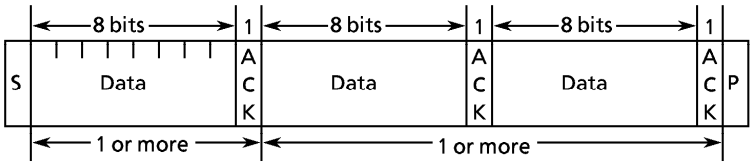
(a) Addressing format



(b) Addressing format (with restart)



(c) Free data format



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

Figure 2-24. The Data Format in the I²C bus Mode

2.9.4 I²C Bus Mode Control

The following registers are used for control and operation status monitoring when using the serial bus interface (SBI) in the I²C bus mode.

Serial Bus Interface Control Register 1

SBICR1 (0020_H)

7	6	5	4	3	2	1	0
"0"	"0"	"0"	ACK	CHS	SCK		

(Initial value ***0 0000)

ACK	Acknowledge mode specification	0 : Acknowledge not returned to transmitter. 1 : Acknowledge returned to transmitter.	Read/ Write
CHS	Input/output channel selection	0 : Channel0 (SCL0, SDA0) 1 : Channel1 (SCL1, SDA1)	Read/ Write
SCK	Serial clock selection	000 : fc/2^6 [Hz] (125 kHz) 001 : fc/2^7 [Hz] (62.5 kHz) 010 : fc/2^8 [Hz] (31.2 kHz) 011 : fc/2^9 [Hz] (15.6 kHz) 100 : fc/2^{10} [Hz] (7.8 kHz) 101 : fc/2^{12} [Hz] (1.9 kHz) 110 : fc/2^{15} [Hz] (244 Hz) 111 : reserved } At fc = 8 MHz Output on SCL pin	Write only

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

Note 2 : Always write "0" to bit 5, 6 and 7 in SBICR1.

Serial Bus Interface Data Buffer Register

SBIDBR (0021_H)

7	6	5	4	3	2	1	0

Read / Write

I²C 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)

ALS	Address recognition mode specification	0 : Slave address recognition 1 : Non slave address recognition	Write only

Figure 2-25. Serial Bus Interface Control Register 1/Serial Bus Interface Data Buffer Register/ I²C bus Address Register in the I²C bus Mode

Serial Bus Interface Control Register 2

SBICR2 (0023_H)

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

(Initial value 00** 00**)

MST	Master/slave selection (Write), Status monitor (Read)	0 : Slave 1 : Master	Read/ Write
TRX	Transmitter/receiver selection (Write), Status monitor (Read)	0 : Receiver 1 : Transmitter	
BB	Start/stop generation (Write), I²C bus status monitor (Read)	0 : Stop condition (Write) , Bus free (Read) 1 : Start condition (Write) , Bus busy (Read)	
PIN	Cancel interrupt service request(Write), Status monitor (Read)	0 : – (Write), Interrupt service requested (Read) 1 : Cancel interrupt service request (Write) , canceled (Read)	
SBIM	Serial bus interface operating mode selection	00 : Port mode (serial bus interface output disable) 01 : SIO mode 10 : I²C bus mode 11 : Reserved	Write only

*Note 1 : * ; don't care*

Note 2 : Switch a mode to port after confirming that the bus is free.

SBISR (0023_H)

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

AAS	Slave address match detection monitor	0 : – 1 : Slave address match or "GENERAL CALL" detected	Read only
AD0	"GENERAL CALL" detection monitor	0 : – 1 : "GENERAL CALL" detected	
LRB	Last received bit monitor	0 : Last received bit "0" 1 : Last received bit "1"	

Figure 2-26. Serial Bus Interface Control Register 2/Serial Bus Interface Status Register in the I²Cbus Mode

(1) **Acknowledge mode specification**

Set the ACK (bit 4 in the SBICR1) to "1" for operation in the acknowledge mode. In the receiver mode during the clock pulse cycle, the SDA pin is set to the low level in order to generate the acknowledge signal. When the ACK is cleared to "0", the SDA pin released high-level in the acknowledge timing.

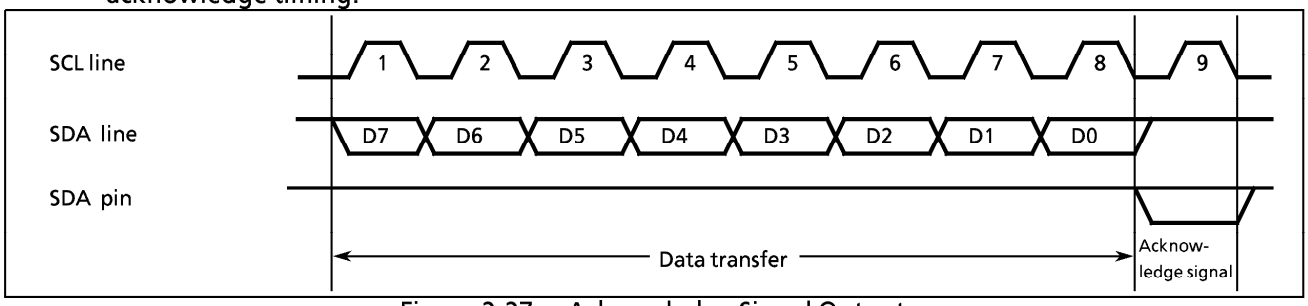


Figure 2-27. Acknowledge Signal Output

(2) **Input/Output channel specification**

Set the input/output to the CHS (bit 3 in the SBICR1)

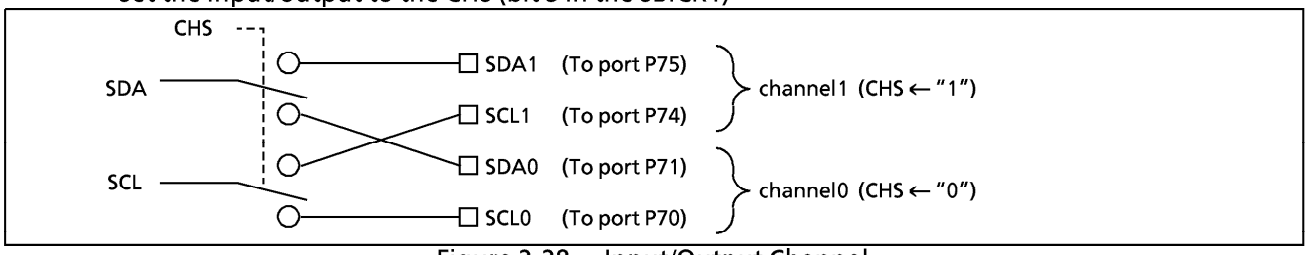


Figure 2-28. Input/Output Channel

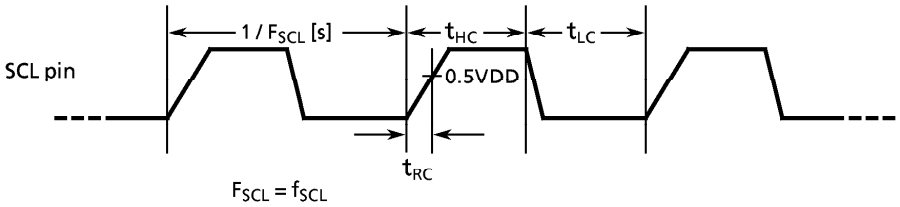
(3) **Serial clock**

a. Clock source

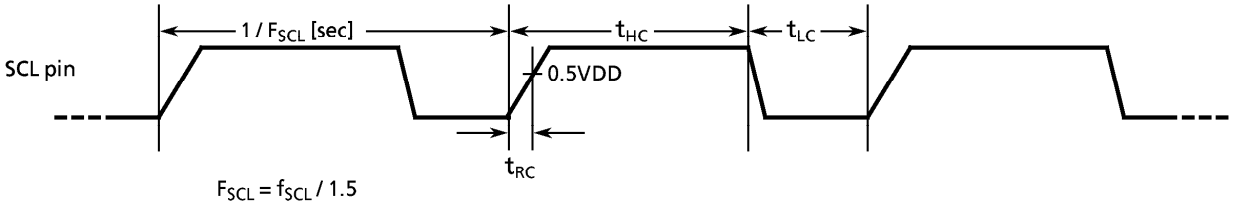
The SCK (bits 2 to 0 in the SBICR1) is used to select a maximum transfer frequency directed from the SCL pin in the master mode. When rising time of the output clock (t_{RC}) is at least 2/fc [s], a high-level time of the output clock (t_{HC}) is t_{SCL}. While the SCL line is fixed to low-level by a slave device, the output clock stops.

The first clock (t_{HC} [s]) after restart is $(t_{SCL}/2) \leqq t_{HC} \leqq t_{SCL}$.

(a) $t_{RC} < (2 / fc)$ [s] 、 $t_{HC} = t_{LC} = t_{SCL} / 2$ [s] ($t_{SCL} = 1 / f_{SCL}$ [s])



(b) $t_{RC} \geqq (2 / fc)$ [s] 、 $t_{HC} = t_{SCL}$ [s] , $t_{LC} = t_{SCL} / 2$ [s]



SCK (bits2 to 0 in the SBICR1)	F_{SCL} (at fc = 8MHz)	
	t_{RC} < 250 [ns]	250 [ns] ≦ t_{RC}
000	125. kHz	83.3 kHz
001	62.5 kHz	41.6 kHz
010	31.2 kHz	20.8 kHz
011	15.6 kHz	10.4 kHz
100	7.8 kHz	5.2 kHz
101	1.95 kHz	1.3 kHz
110	0.24 kHz	0.16 kHz
	Case (a)	Case (b)

Figure 2-29. Serial Clock Output

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

When the 87CK43/M43 is used as a slave device, set the slave address and ALS to the I2CAR. Set "0" to the ALS for the address recognition mode.

(5) **Master/slave selection**

Set the MST (bit 7 in the SBICR2) to "1" for operating the 87CK43/M43 as a master device. Reset the MST for operation as a slave device. The MST is cleared to "0" by the hardware after a stop condition on the bus is detected.

(6) **Transmitter/receiver selection**

Set the TRX (bit 6 in the SBICR2) to "1" for operating the 87CK43/M43 as a transmitter. Reset the TRX for operation as a receiver. When 87CK43/M43 receives a slave address setted in I2CAR or a GENERAL CALL from the master device in the 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, 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 I^2C bus is detected.

(7) **Start/Stop Condition generation**

A start condition and 8-bit of data (a slave address and a direction bit which are set to a data buffer register) are output on a bus by writing "1" to the MST, TRX, and BB when the BB (bit 5 in the SBICR2) is "0".

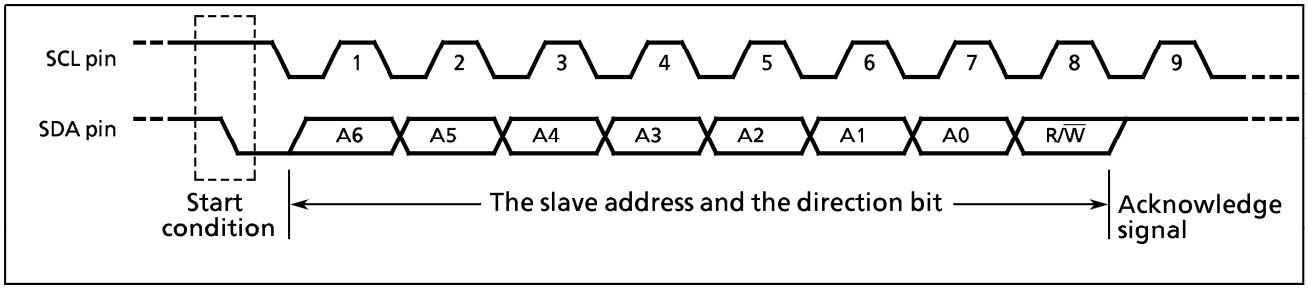


Figure 2-30. Start Condition Generation and Slave Address Generation

A stop condition is output on a bus by writing "1" to the MST and TRX when the BB is "1".

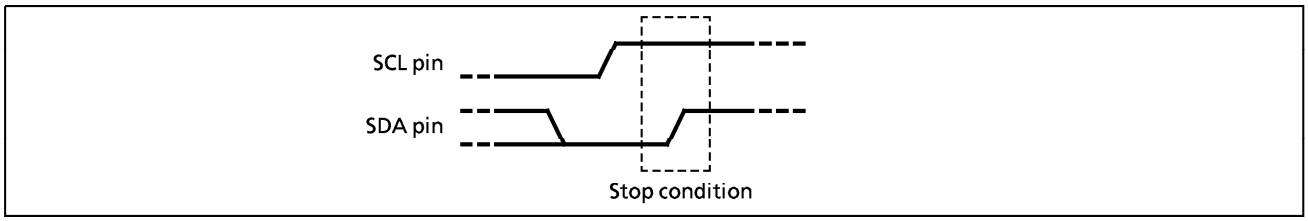


Figure 2-31. Stop Condition Generation

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. In the case of the 87CK43/M43 is a master transmitter, when the MST is set to "0" after a start condition is generated until a stop condition is generated, the value of writing BB is read from BB.

(8) **Cancel interrupt service request**

When a serial bus interface interrupt request (INTSBI) occurs, the PIN (bit 4 in the SBISR) is cleared to "0". During the time that the PIN is "0", the SCL pin is pulled down to the low level.
The PIN is cleared to "0" when 1-byte of data is transmitted or received. 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}.
In the address recognition mode (ALS = 0), the PIN is cleared to "0" when the received slave address is the same as the value set at the I2CAR or when a GENERAL CALL is received (all 8-bit data are "0" after a start condition). Although the PIN (bit 4 in the SBICR2) can be set to "1" by the program, the PIN is not set to "0" when "0" is written.

(9) **Serial bus interface operation mode selection**

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

(10) **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), or when receiving a slave address with the same value that sets a GENERAL CALL or I2CAR. When the ALS is "1", the AAS is set to "1" after receiving the first 1-byte of data. The AAS is reset by either writing/reading data to/from a data buffer register.

(11) **GENERAL CALL detection monitor**

The AD0 (bit 1 in the 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 reset when a start or stop condition is detected on the bus.

(12) **Last received bit monitor**

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

2.9.5 Data Transfer in I^2C bus Mode

(1) **Device Initialization**

Set the ACK, CHS and SCK in the SBICR1. Specify "0" to bits 7 to 5.
Set a slave address and the ALS (ALS = 0 when an addressing format) to the I2CAR.
For specifying the default setting to a slave receiver mode, assign "0" to the MST, TRX, and BB in the SBICR2; "1" to the PIN; "10" to the SBIM; and "0" to bits 0 and 1.

(2) **Start Condition and Slave Address Generation**

Observe 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 writing "1" to the MST, TRX, and BB, the slave address and the direction bit which are set to the SBIDBR and the start condition are output on the bus. A slave device receives these data and pulls down the SDA line of the bus to the low level at the acknowledge signal timing. 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 low level while the PIN is "0". When an interrupt request occurs, the TRX changes by the hardware according to the direction bit only when an acknowledge signal is returned from the slave device.

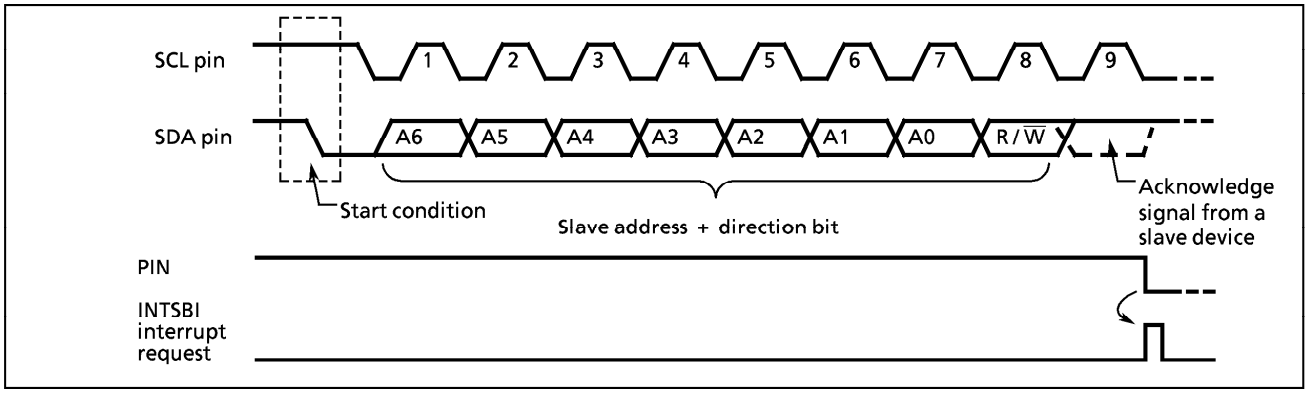


Figure 2-32. Start Condition Generation and Slave Address Transfer

(3) **1-byte Data Transfer**

Test the MST by the INTSBI interrupt process after a 1-byte data transfer is concluded, and determine whether the mode is a master or slave.

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

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

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

Test the LRB. When the LRB is "1", a receiver does not request data. Implement the process to generate a stop condition (described later) and terminate data transfer.
When the LRB is "0", the receiver requests new data. 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-byte of data from the SCL pin, and then the 1-byte data is transmitted from SDA pin. After the data is transmitted, an INTSBI interrupt request occurs. The PIN becomes "0" and the SCL pin is pulled down to the low level. If the data to be transferred is more than one word in length, repeat the procedure from the LRB test above.

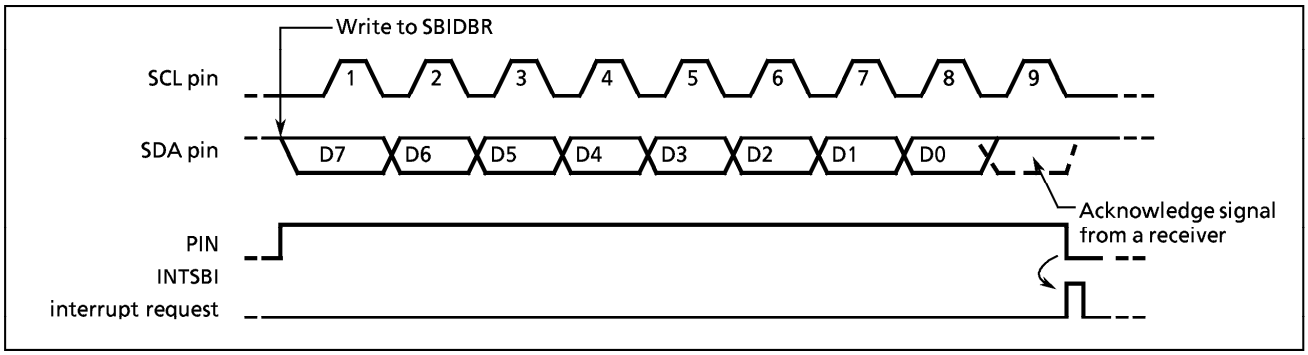


Figure 2-33. 1-byte Data Transfer

② When the TRX is "0" (Receiver mode)
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 87CK43/M43 outputs a serial clock pulse to the SCL pin to transfer new 1-byte of data and sets the SDA pin to "0" at the acknowledge signal timing.
An INTSBI interrupt request then occurs, the PIN becomes "0" and the SCL pin pulled down to the low level. The 87CK43/M43 outputs a clock pulse for 1-byte of data transfer and the acknowledge signal each time that received data is read from the SBIDBR.

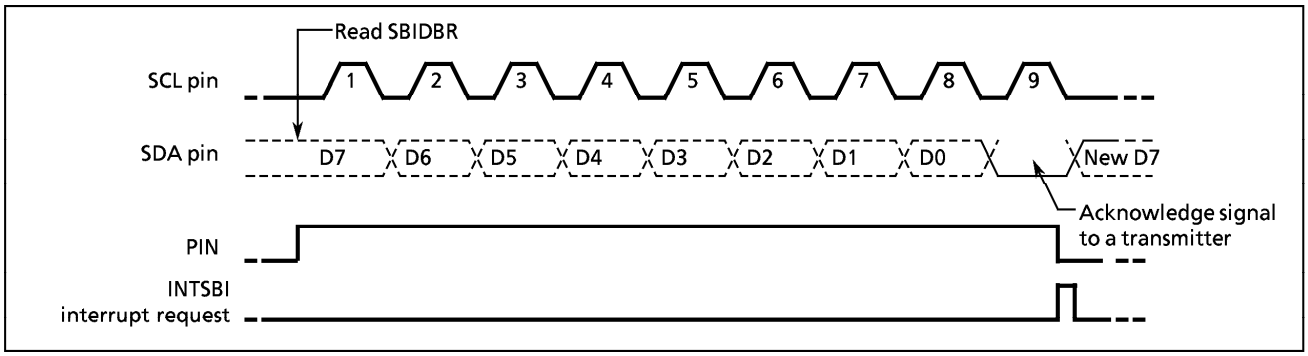


Figure 2-34. 1-byte Data Receive

In order to terminate transmitting data to a transmitter, reset the ACK before reading data which is 1-byte before the last data to be received. The SDA pin released high-level in an acknowledge timing of last received byte. The receiver indicates to the transmitter that data transfer is complete.
After data is received and an interrupt request has occurred, the 87CK43/M43 generates a stop condition and terminates data transfer. When reading data from SBIDBR in the last received byte, the serial clock and acknowledge signal don't outputs because ACK is "0".

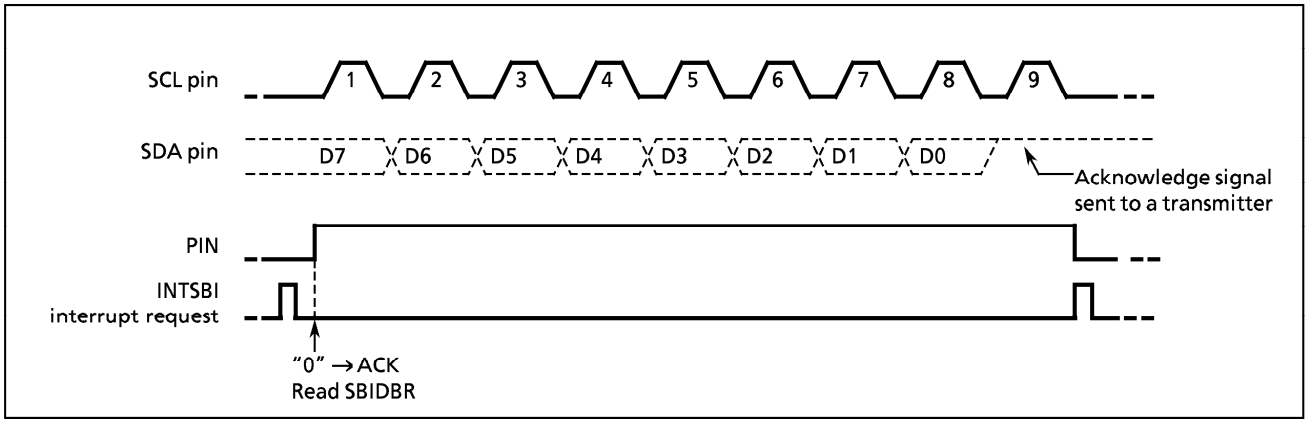


Figure 2-35. Termination of Data Transfer in Master Receiver Mode

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

In the slave mode, an INTSBI interrupt request occurs when the 87CK43/M43 receives 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. When an INTSBI interrupt request occurs, the PIN (bit 4 in the SBICR2) is reset, and the SCL pin is pulled down to the low level. Either reading/writing from/to the SBIDBR or setting the PIN to "1" releases the SCL pin after taking t_{LOW} time.

In the slave mode, the 87CK43/M43 operates either in normal slave mode.

The 87CK43/M43 tests 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	AAS	AD0	Conditions	Process
1	1	0	In the slave receiver mode, the 87CK43/M43 receives a slave address of which the value of the direction bit sent from the master is "1".	Write transmitted data to the SBIDBR.
	0	0	In the slave transmitter mode, 1-byte data is transmitted.	Test the LRB. If the LRB is set to "1", set the PIN to "1" since the receiver does not request further data. Then, reset the TRX to release the bus. If the LRB is set to "0", write transmitted data to the SBIDBR since the receiver requests further data.
0	1	1/0	In the slave receiver mode, the 87CK43/M43 receives a slave address or general CALL of which the value of the direction bit sent from the master is "0".	Read the SBIDBR for setting the PIN to "1" (reading dummy data) or write "1" to the PIN.
	0	1/0	In the slave receiver mode, the 87CK43/M43 terminates receiving of 1-byte data.	Read received data from the SBIDBR.

(4) **Stop Condition Generation**

Writing "1" to the MST, TRX, and PIN, and "0" to the BB generates a stop condition on the bus.,

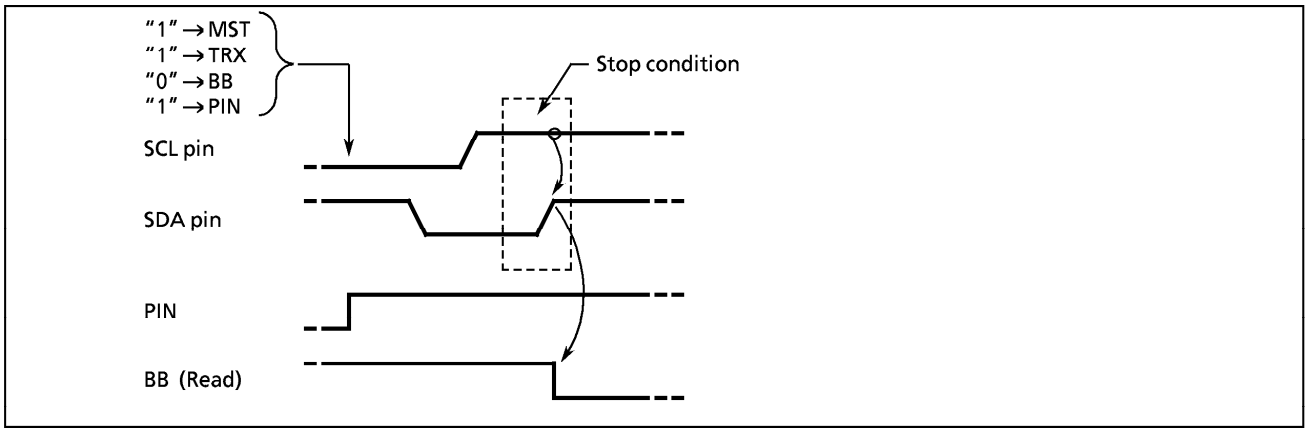


Figure 2-36. Stop Condition Generation

(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 87CK43/M43 is in the master mode.

Specify "0" to the MST, TRX, and BB and "1" to the PIN and release the bus. The SDA pin retains the high level and the SCL pin is released. Since a stop condition is not generated on the bus, the bus is assumed to be in a busy state from other devices. Test the BB until it becomes "1" to check that the SCL pin of the 87CK43/M43 is released. Test the LRB until it becomes "1" to check that the SCL line of the bus is not pulled down to the low level by other devices. After confirming that the bus stays in a free state, generate a start condition with procedure (2).

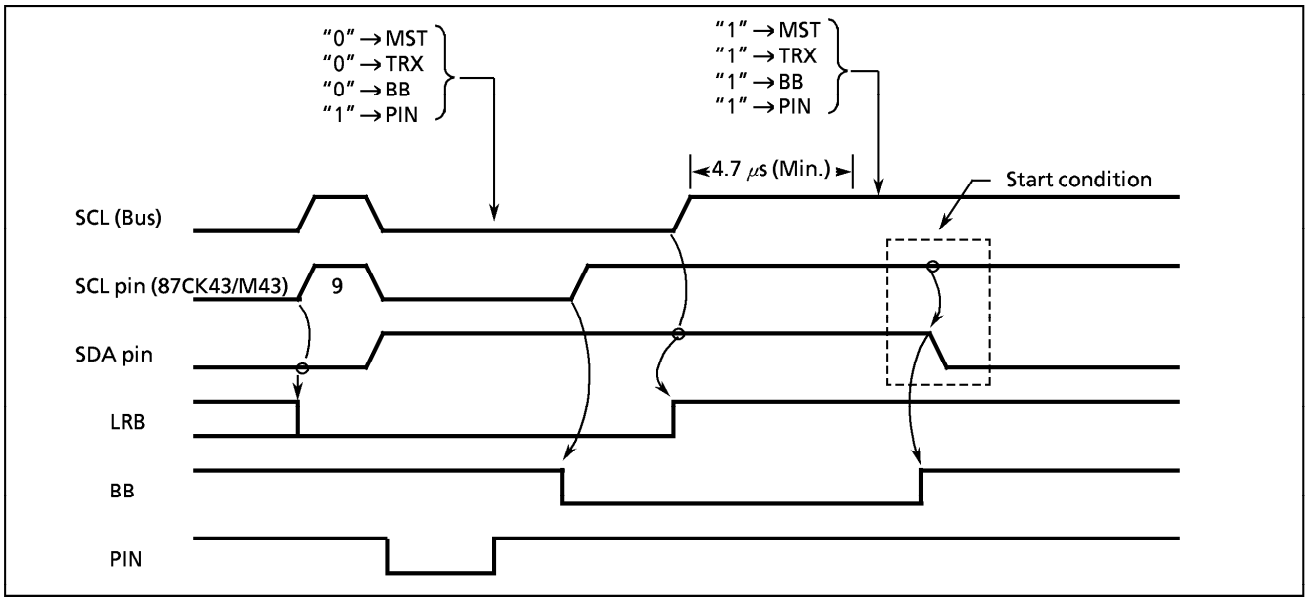


Figure 2-37. Timing Diagram when Restarting the 87CK43/M43

2.9.6 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		CHS	SCK		

(Initial value 0000 0000)

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)	
SIOM	Transfer mode select	00 : 8-bit transmit mode 01 : reserved 10 : 8-bit transmit/receive mode 11 : 8-bit receive mode	
CHS	Input/Output channel selection	0 : Channel 0 ($\overline{SCK0}$, SO0, SI0) 1 : Channel 1 ($\overline{SCK1}$, SO1, SI1)	R/W
SCK	Serial clock select	000 : $fc/2^6$ (125 kHz) 001 : $fc/2^7$ (62.5 kHz) 010 : $fc/2^8$ (31.25 kHz) 011 : $fc/2^9$ (15.62 kHz) 100 : $fc/2^{10}$ (7.81 kHz) 101 : $fc/2^{12}$ (1.95 kHz) 110 : $fc/2^{15}$ (244 kHz) 111 : External clock (input from $\overline{SCK}$ pin) } At fc = 8 MHz (Output on $\overline{SCK}$ pin)	Write only

*Note 1 : fc ; high-frequency clock [Hz], * ; don't care*
Note 2 : Set the SIOS to "0" when setting the transfer mode or serial clock.

Serial Bus Interface Data Buffer Register

SBIDBR (0021_H)

7	6	5	4	3	2	1	0

Read / Write

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.

Serial Bus Interface Status Register

SBISR (0023_H)

7	6	5	4	3	2	1	0
				SIOF	SEF		

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

Figure 2-38. Serial Bus Interface Control Register 1/Serial Bus Interface Data Buffer Register/Serial Bus Interface Control Register 2/Serial Bus Interface Status Register in SIO Mode

(1) Serial Clock

a. Clock source

The SCK (bits 2 to 0 in the 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 high 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.

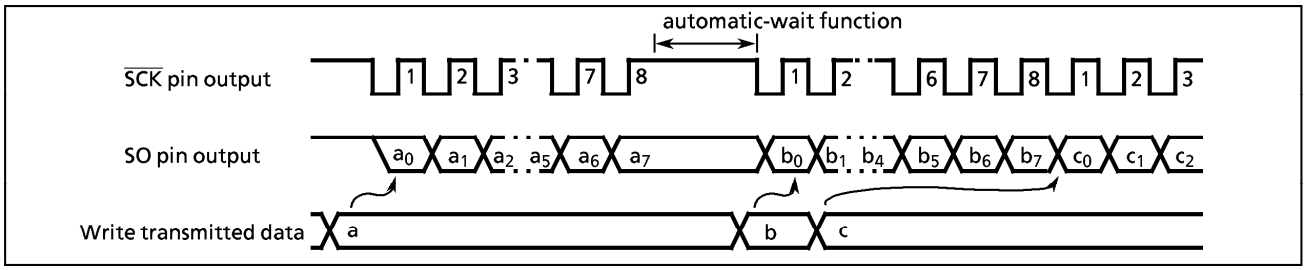


Figure 2-39. Automatic-wait Function

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 longer than 4 machine cyles is required for both high and low levels in the serial clock. The maximum data transfer frequency is 250 kHz (when fc = 8 MHz).

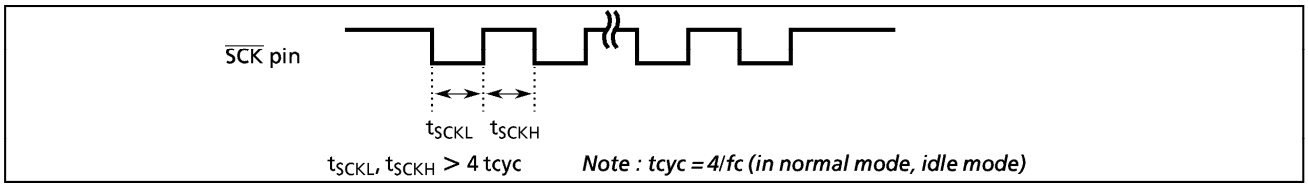


Figure 2-40. External Clock

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 $\overline{SCK}$ pin input/output).

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

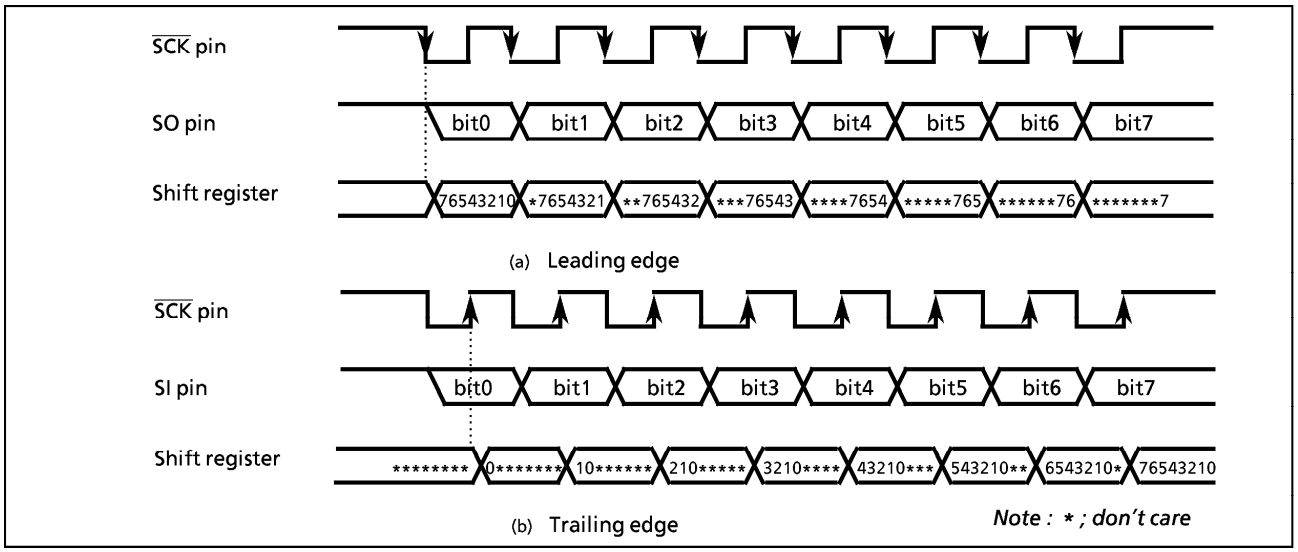


Figure 2-41. Shift Edge

(2) Transfer mode

The SIOM (bits 5 and 4 in the SIO1CR) 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 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 $\overline{SCK}$.
Transmitting data is ended by clearing the SIOS to "0" by the buffer empty interrupt service program or setting the SIOINH to "1". 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 the SIOINH is set, transmitting data stops. The SIOF turns "0".
When the external clock is used, it is also necessary to clear the SIOS to "0" before new data is shifted; otherwise, dummy data is transmitted and operation ends.

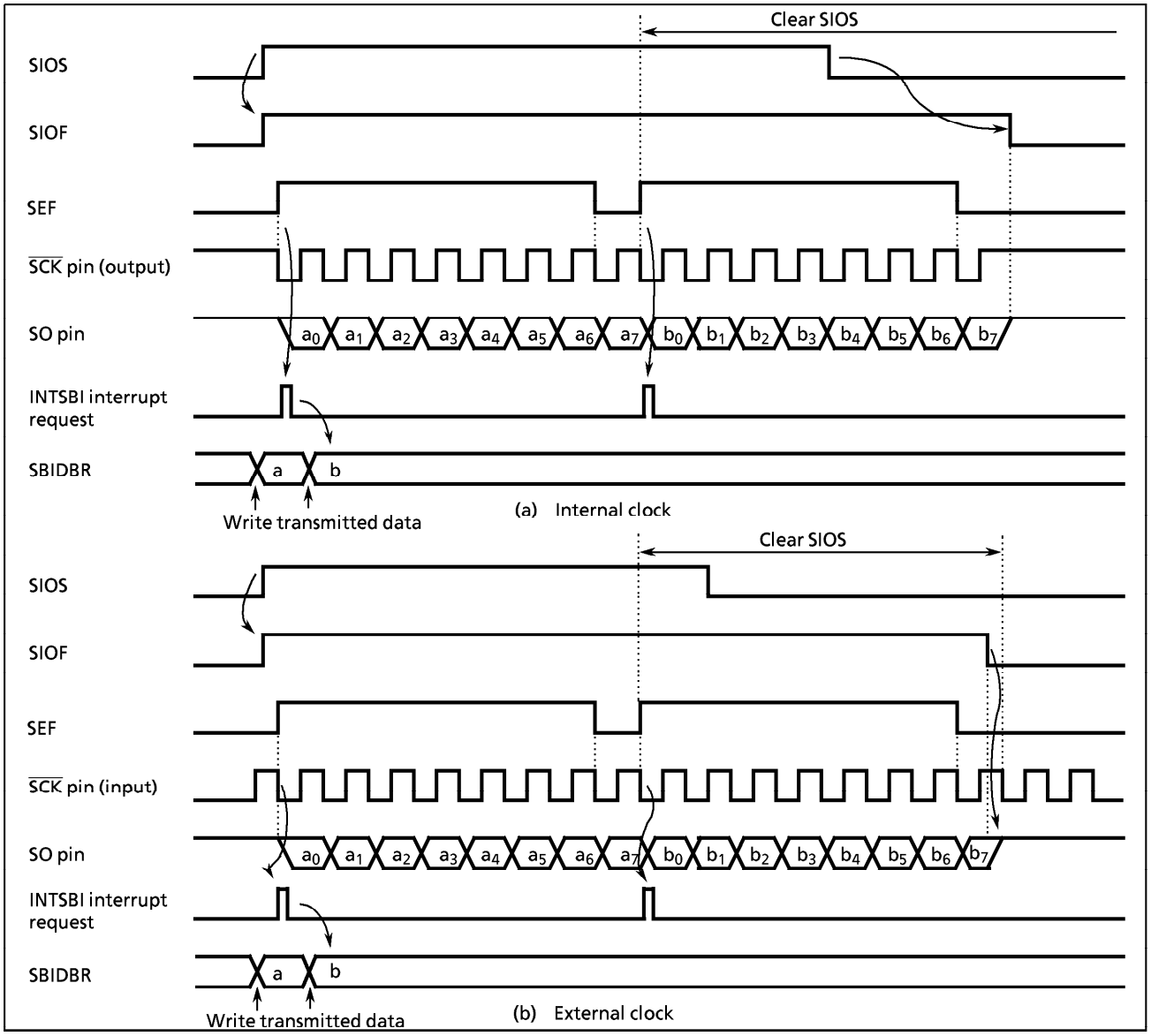


Figure 2-42. Transfer Mode

Example: Program to stop transmitting data (when external clock is used)

```
STEST1  :   TEST    (SBISR) . SEF          ; If SEF = 1   then  loop
            JRS     F , STEST1
STEST2  :   TEST    (P3) . 6               ; If SCK = 0   then  loop
            JRS     T , STEST2
            LD      (SBICR1) , 00000111B   ; SIOS ← 0
```

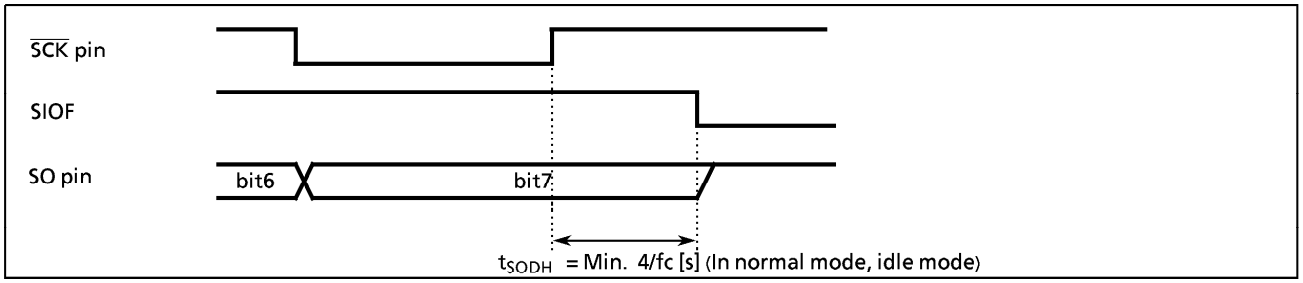


Figure 2-43. Transmitted Data Hold Time at End of Transmit

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.

Receiving data is ended by clearing the SIOS to "0" by the buffer full interrupt service program or setting the SIOINH to "1". 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 the 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.*

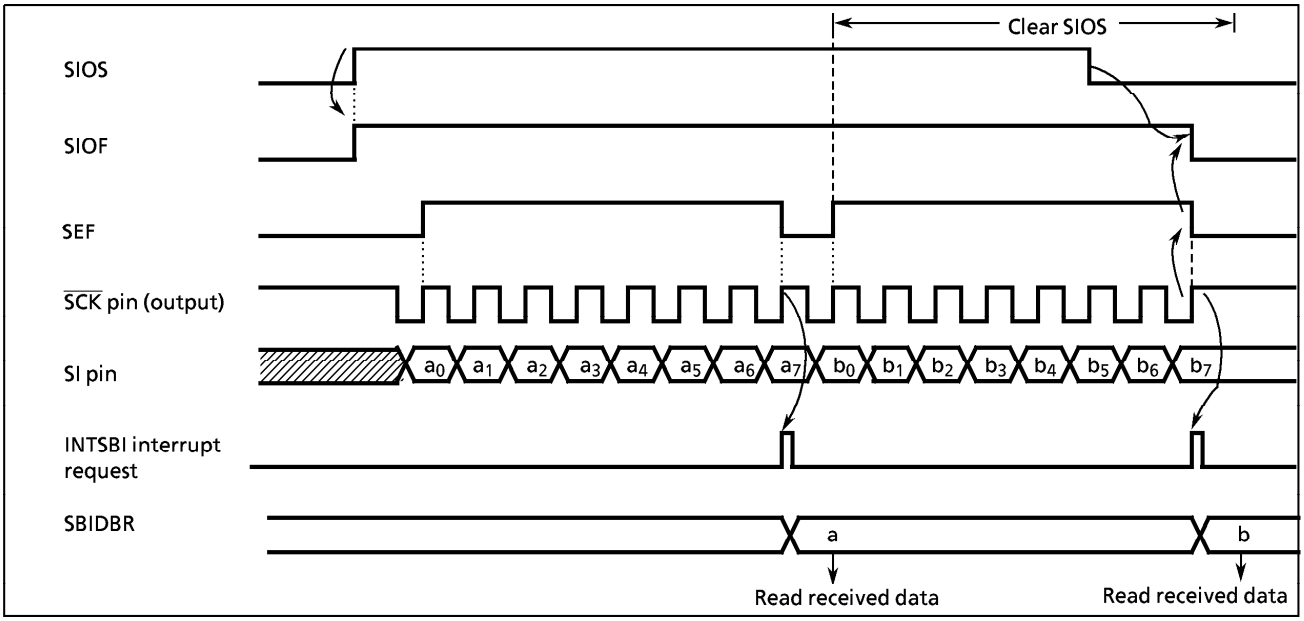


Figure 2-44. Receive Mode (Example : Internal clock)

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 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 $\overline{SCK}$.

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 clearing the SIOS to "0" by the INTSBI interrupt service program or setting the SIOINH to "1". When the SIOS is cleared, received data is transferred to the SBIDBR in complete blocks. The transmit/receive mode ends when the transfer is complete. In order to confirm if data is surely transmitted/received by the program, set the SIOF (bit3 in the SBISR) to be sensed. The SIOF becomes "0" after transmitting/receiving is complete. When the SIOINH is set, transmitting/receiving data stops. The SIOF turns "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.*

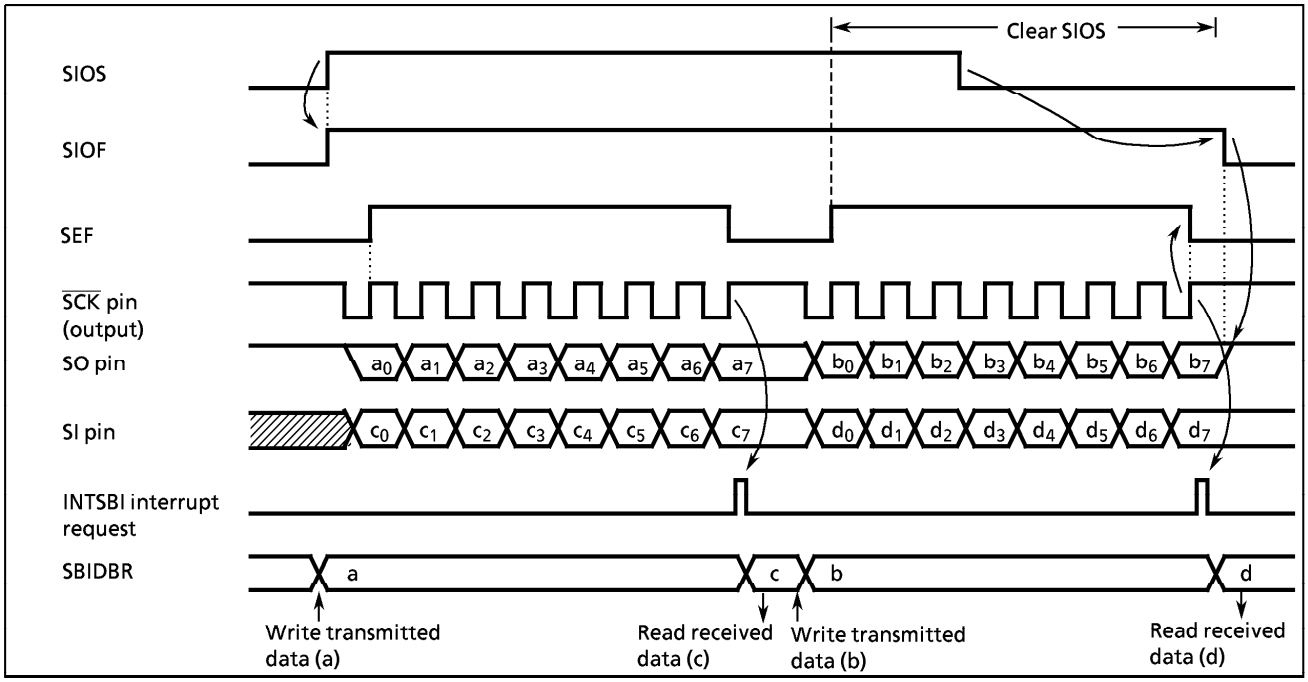


Figure 2-45. Transmit/Receive Mode (Example : Internal clock)

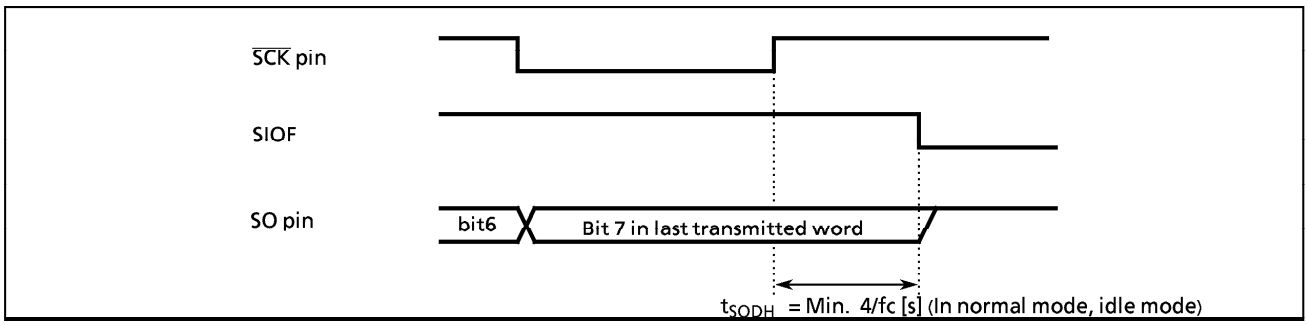


Figure 2-46. Transmitted Data Hold Time at End of Transmit/Receive

2.10 8-bit A/D Converter (ADC)

The 87CK43/M43 each have a 6-channel multiplexed-input 8-bit successive approximate type A/D converter with sample and hold.

2.10.1 Configuration

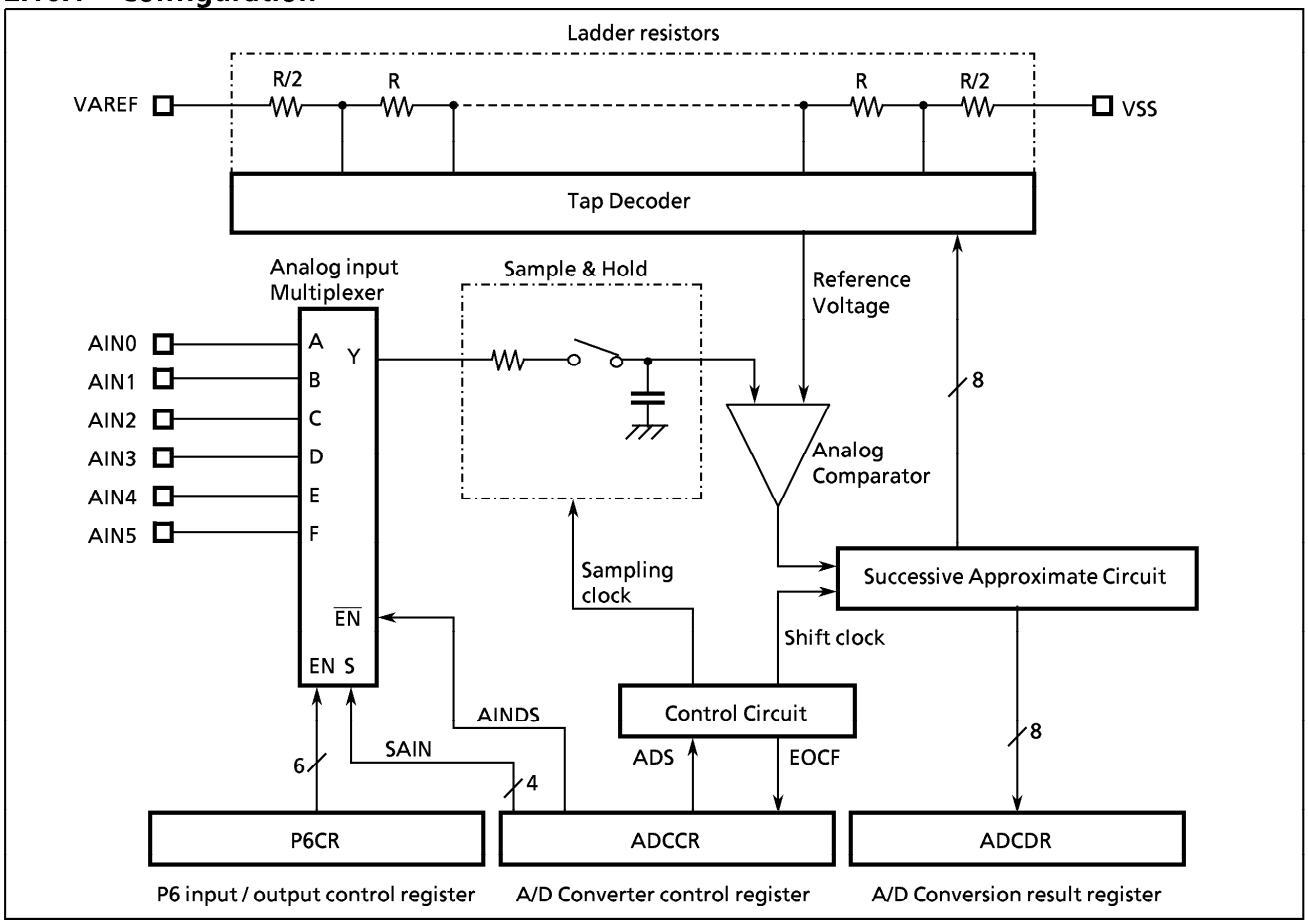


Figure 2-45. A/D Converter

2.10.2 Control

The A/D converter is controlled by an A/D converter control register (ADCCR) and a port P6 input/output control register (P6CR).

A/D Conversion Result Register

ADCDR ($000F_H$)

7	6	5	4	3	2	1	0

Read only

Port P6 Input / Output Control Register

P6CR ($000C_H$)

7	6	5	4	3	2	1	0

(Intial value : 0011 1111)

P6mCR	I/O control for port P6 (Set each of a bit)	0 : input mode 1 : output mode	Note : The P6CR (bit7, 6) does not affect AINDS.	write only
P6nCR		AINDS = 1 (A/D not use) 0 : input mode 1 : output mode	AINDS = 0 (A/D use) 0 : input port 1 : analog input	

Note : m = 7 to 6 n = 5 to 0

Figure 2-46. A/D Converter Result Register and P6CR

ADCCR ($000E_H$)

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

(Initial value : 00*0 0000)

SAIN	Analog input selection	0000 : AIN0 0001 : AIN1 0010 : AIN2 0011 : AIN3 0100 : AIN4 0101 : AIN5 011* : reserved 1 *** : reserved	R/W
AINDS	Analog input control	0 : Enable 1 : Disable	
ADS	A/D conversion start	0 : – 1 : A/D conversion start	
EOCF	End of A/D conversion flag	0 : Under conversion or Before conversion 1 : End of conversion	Read only

*Note 1 : * ; don't care*

Note 2 : Select analog input when A/D converter stops.

Note 3 : The ADS is automatically cleared to "0" after starting conversion.

Note 4 : The EOCF is cleared to "0" when reading the ADCDR.

Note 5 : The EOCF is read-only.

Figure 2-47. A/D Converter Control Register

2.10.3 Operation

Apply analog reference voltage to pins VAREF and VSS.

(1) Start of A/D conversion

First, set the corressponding P6CR bit to "1" for analog input. Clear the AINDS (bit 4 in ADCCR) to "0" and select one of 6 analog input AIN5-AIN0 with the SAIN (bits 3-0 in ADCCR). The pins not setting for analog inputs can be used as input ports, but can not be used as output ports.

A/D conversion is started by setting the ADS (bit 6 in ADCCR) to "1".

Conversion is accomplished in 46 machine cycles (184/fc [s]). In this case, it is 23 μs. (at fc = 8 MHz)

The EOCF (bit 7 in ADCCR) is set to "1" at end of conversion. During A/D conversion, when the ADS is initialized to "1", the A/D conversion is initialized again.

The sampling of analog input voltage is executed into 4 machine cycles after A/D conversion starting.

Note : The circuit of sample and hold is included in a condenser (12pF (typ.)) through a register (5kΩ (typ.)).
Therefore, until 4 machine cycles is over, this condenser must be charged.

(2) Reading of A/D conversion result

After the end of conversion, read the conversion result from the ADCDR.

The EOCF is automatically cleared to "0" when reading the ADCDR. During A/D conversion, when the conversion result is read out, undefined data is read out.

(3) A/D conversion in STOP mode

When the MCU places in the STOP mode during the A/D conversion, the conversion is terminated and the ADCDR contents become indefinite.

However, if the STOP mode is started after the end of conversion (EOCF = 1), the ADCDR contents are held.

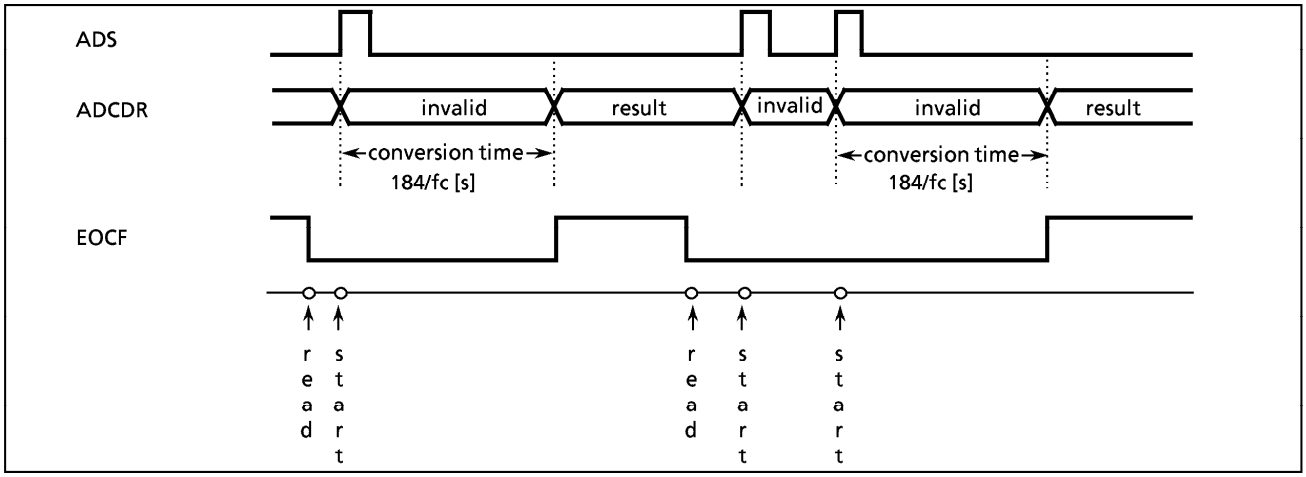


Figure 2-48. A/D Conversion Timing Chart

Example: A/D conversion is started after AIN4 is selected as an analog input channel. Program checks EOCF, and reads the result of A/D conversion, and writes to RAM (address $009E_H$).

```
        ; AIN SELECT
        LD      (ADCCR) , 00000100B     ;   selects AIN4
        ; A/D CONVERT START
        SET     (ADCCR) . 6             ;   ADS = 1
SLOOP   :  TEST (ADCCR) . 7             ;   EOCF = 1 ?
        JRS     T, SLOOP
        ; RESULT DATA READ
        LD      (9EH), (ADCDR)
```

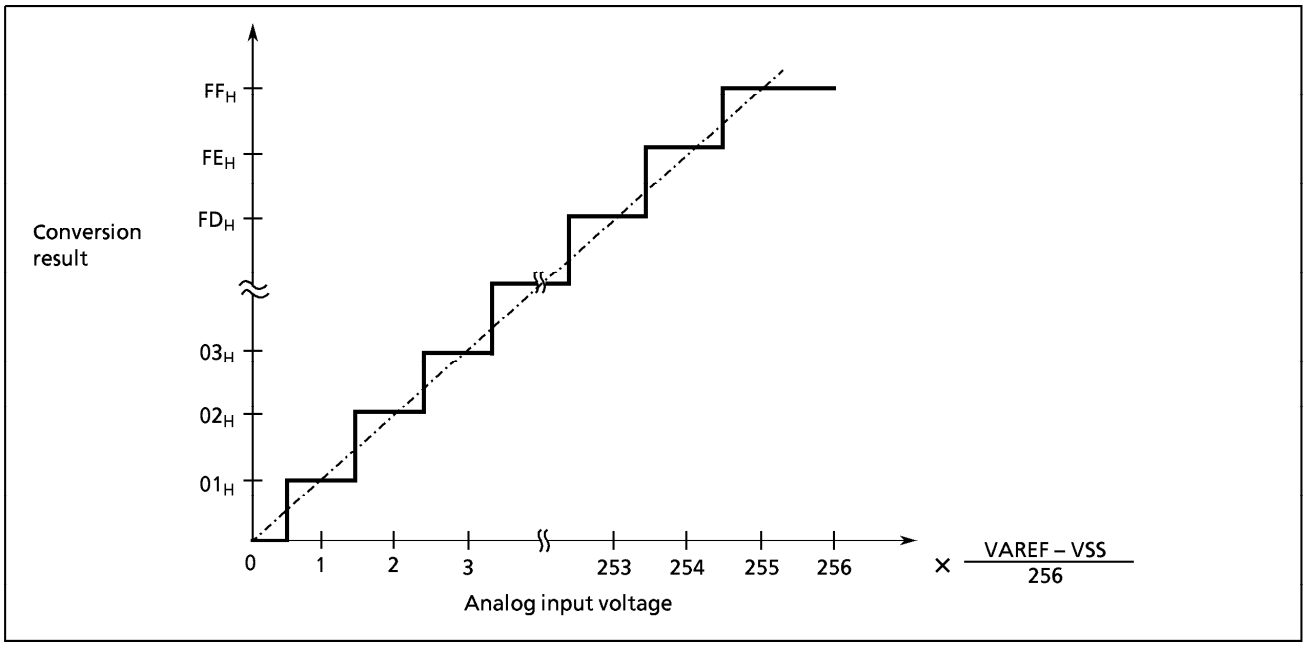


Figure 2-49. Analog Input Voltage vs A/D Conversion Result (typ.)

2.11 Pulse Width Modulation Circuit Output

87CK43/M43 has 2 built-in pulse width modulation (PWM) channels. D/A converter output can easily be obtained by connecting an external low-pass filter. PWM outputs are multiplexed with general purpose I/O ports as; P66 ($\overline{PWM1}$), P67 ($\overline{PWM0}$). When these ports are used PWM outputs,the corresponding bits of P6 output latches should be set to "1".

2.11.1 Configuration

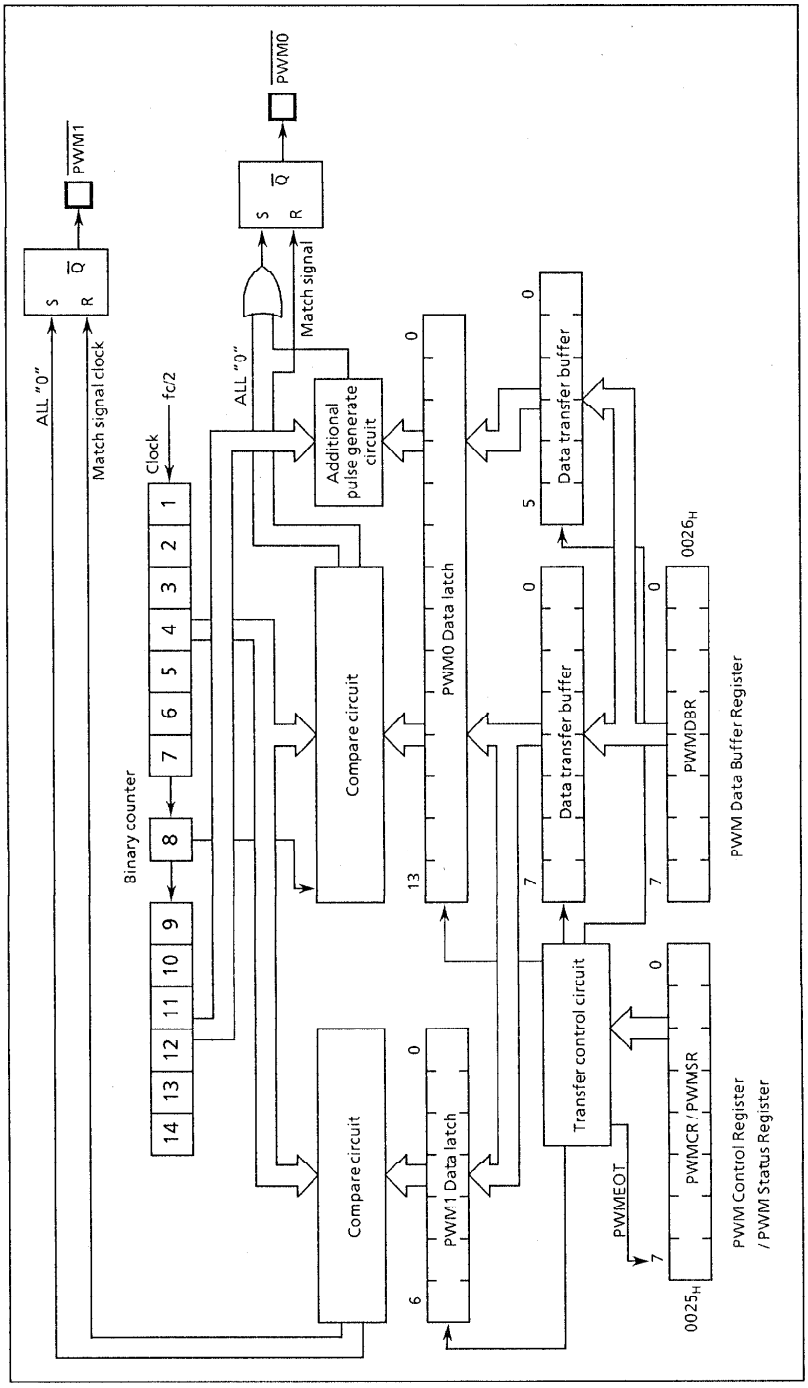


Figure 2-50. Pulse Width Modulation Circuit

2.11.2 PWM Output Wave Form

(1) $\overline{PWM0}$ output

This is 14-bit resolution PWM output and one period is $T_M = 2^{15}/fc$ [s].

The 8 high-order bits of the PWM data latch control the pulse width of the pulse output with a period of T_S ($T_S = T_M/64$), which is the sub-period of the $\overline{PWM0}$. When the 8-bit data are decimal n ($0 \leqq n \leqq 255$), this pulse width becomes $n \times t_0$, where $t_0 = 2/fc$.

The lower 6-bit of 14 bit data are used to control the generation of additional to wide pulse in each T_S period. When the 6-bit data are decimal m ($0 \leqq m \leqq 63$), the additional pulse is generated in each of m periods out of 64 periods contained in a T_M period. The relationship between the 6 bits data and the position of Ts period where the additional pulse is generated is shown in Table 2-8.

Table 2-8. Correspondence between 6 Bits Data and the Additional Pulse Generated Ts Period

Bit position of 6 bits data	Relative position of Ts where the output pulse is generated. (No. i of $T_{S(i)}$ is listed)
Bit 0	32
Bit 1	16, 48
Bit 2	8, 24, 40, 56
Bit 3	4, 12, 20, 28, 36, 44, 52, 60
Bit 4	2, 6, 10, 14, 18, 22, 26, 30, ·····, 58, 62
Bit 5	1, 3, 5, 7, 9, 11, 13, 15, 17, ·····, 59, 61, 63

Note : When the corresponding bit is "1", it is output.

(2) $\overline{PWM1}$ output

These are 7-bit resolution PWM outputs and one period is $T_N = 2^8/fc$ [s]. When the 7bit data are decimal k ($0 \leqq k \leqq 127$), the pulse width becomes $k \times t_0$. The wave form is illustrated in Figure 2-51.

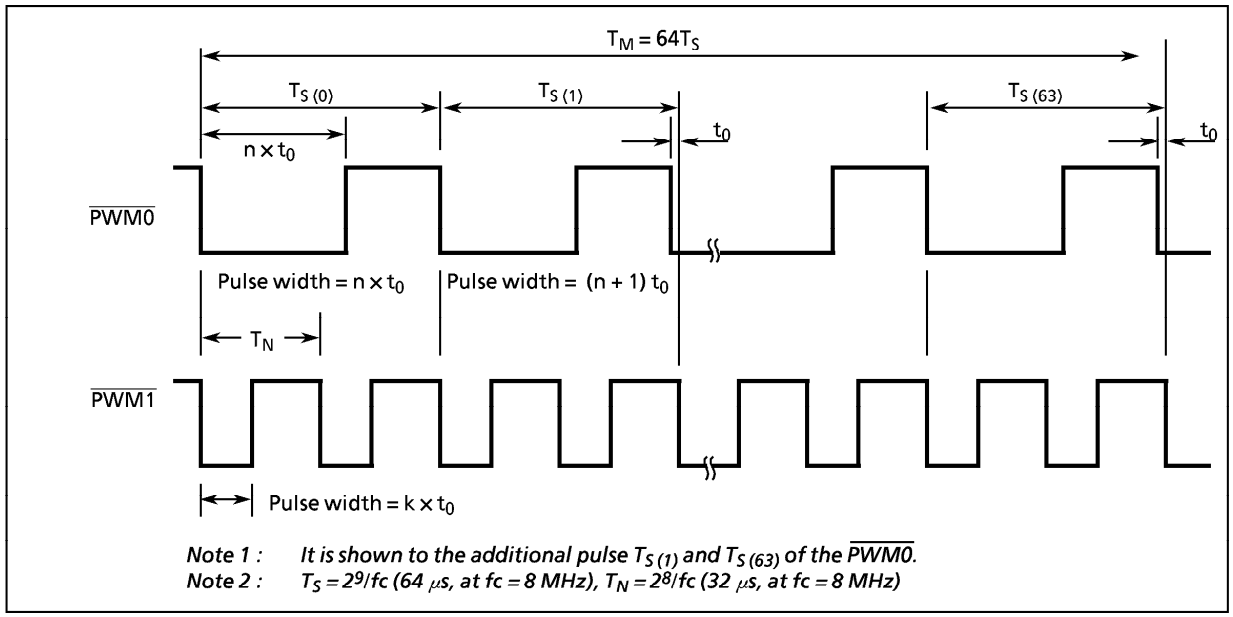


Figure 2-51. PWM Output Wave Form

2.11.3 Control

PWM output is controlled by PWM Control Register (PWMCR) and PWM Data Buffer Register (PWMDBR). The status of transfer PWM data from PWMDBR to PWM data latch is read by PWMEOT of PWM status register (PWMSR).

PWM Control Register

	7	6	5	4	3	2	1	0	
PWMCR (0025H)					PWMDLS				(Initial value **** 0000)

PWMDLS	Selection of PWM data latch and request of data transfer	0000 : Lower 6-bit of PWM0 0001 : 8 High-order bits of PWM0 0010 : PWM1 0011 : reserved 01** : reserved 10** : reserved 1100 : PWM Data Transfer 1101 : reserved 111* : reserved	write only

PWM Status Register

	7	6	5	4	3	2	1	0
PWMSR (0025H)	PWMEOT	"1"	"1"	"1"	"1"	"1"	"1"	"1"

PWMEOT	End of PWM data transfer flag	0 : End of Transfer 1 : Under Transfer	read only

PWM Data Buffer Register

	7	6	5	4	3	2	1	0	
PWMDBR (0026H)									write only

*Note : * : don't care*

Figure 2-52. PWM Control Register / PWM Status Register / PWM Data Buffer Register

(1) Programing of PWM Data

PWM output is controlled by PWM writing the output data to data latches.
For the writing the output data are divided using the PWM Control Register (PWMCR).

1. Write the channel number of PWM data latch to the PWMDLS.
2. Write PWM output data to the PWMDBR.
3. Write "0CH" to the PWMCR.
   When switching of the output data is completed, the PWMEOT becomes "0", indicating that the next data can be written. Do not write PWM data when the PWMEOT is "1" because write errors can occur in this case.

Note : When writing the output data to PWM0 data latch, write "0CH" to the PWMCR after writing of the 14-bits output data is completed.

While the output data are being written to the data latch, the previously written data are being output. The maximum time from the point at which "0CH" is written to the ***PWMDLS*** until PWM output is switched is 2^{15}/fc (4.096 ms, at fc = 8 MHz) for PWM0 output and 2^{9}/fc [s] (64 μs, at fc = 8 MHz) for PWM1 output.

Example : $\overline{\text{PWM0}}$ pin outputs a PWM wave form with a low level of 32 μs width and no additional pulse.
$\overline{\text{PWM1}}$ pin outputs a PWM wave form with a low level of 16 μs width.

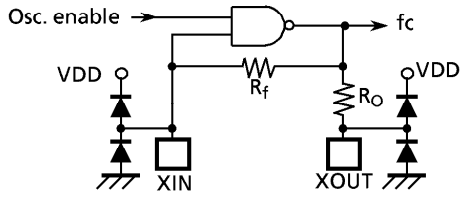
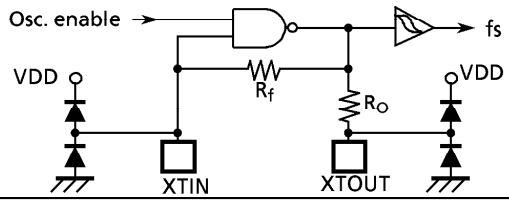
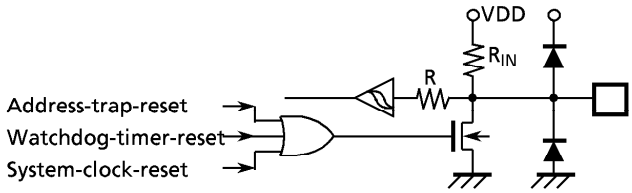
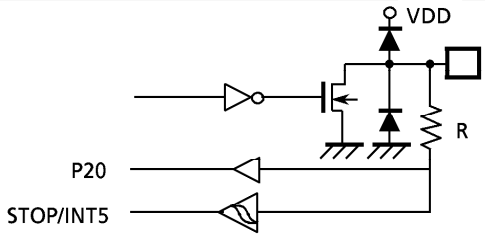
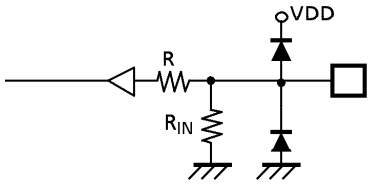
Note : at fc = 8 MHz

```
        LD      (PWMCR), 00H      ; Select lower 6-bit of PWM0
        LD      (PWMDBR), 00H     ; Without the additional pulse
        LD      (PWMCR), 01H      ; Select 8 high-order bits of PWM0
        LD      (PWMDBR), 80H     ; 32 μs ÷ 2/fc = 80H
        LD      (PWMCR), 0CH      ; Request PWM Data Transfer
WAIT0 : TEST    (PWMSR). 7        ; PWMEOT = 0?
        JRS     F, WAIT0
        LD      (PWMCR), 02H      ; Select PWM1
        LD      (PWMDBR), 40H     ; 16 μs ÷ 2/fc = 40H
        LD      (PWMCR), 0CH      ; Request PWM Data Transfer
WAIT1 : TEST    (PWMSR). 7        ; PWMEOT = 0?
        JRS     F, WAIT1
```

INPUT/OUTPUT CIRCUITRY

(1) Control pins

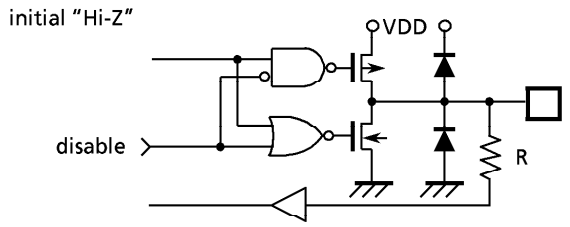
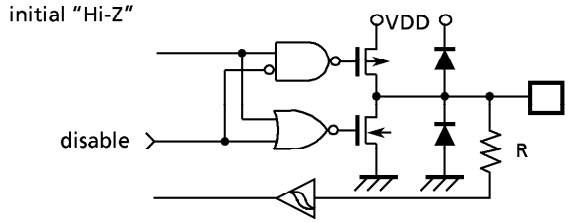
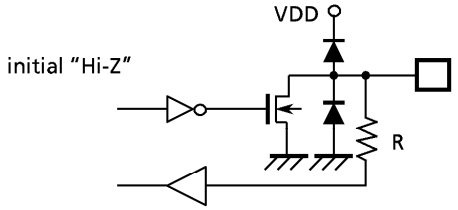
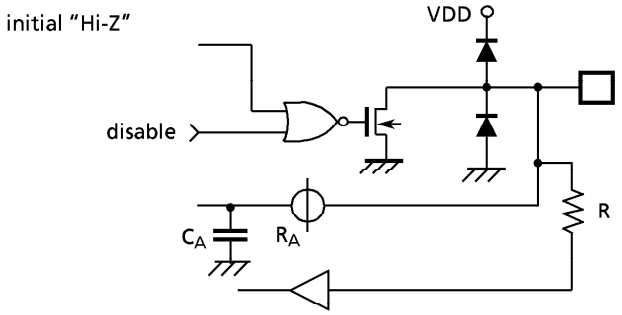
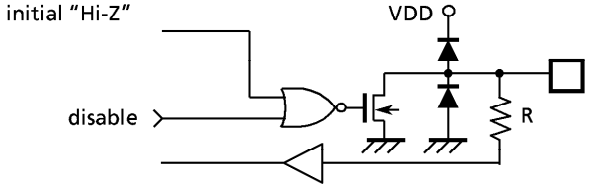
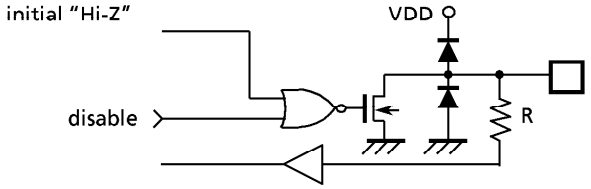
The input/output circuitries of the 87CK43/M43 control pins are shown below, any one of the circuitries can be chosen by a code (NM1 or NM2) as a mask option.

CONTROL PIN	I/O	INPUT/OUTPUT CIRCUITRY and CODE	REMARKS
XIN XOUT	Input Output		Resonator connecting pins (high-frequency) R_f = 1.2 MΩ (typ.) R_O = 1.5 kΩ (typ.)
XTIN XTOUT	Input Output	NM1: Refer to port P2 NM2: 	Resonator connecting pins (low-frequency) R_f = 6 MΩ (typ.) R_O = 220 kΩ (typ.)
$\overline{\text{RESET}}$	I/O		Sink open drain output Hysteresis input Pull-up resistor R_{IN} = 220 kΩ (typ.) R = 1 kΩ (typ.)
$\overline{\text{STOP}}$/$\overline{\text{INT5}}$	Input		Hysteresis input R = 1 kΩ (typ.)
TEST	Input		Pull-down resistor R_{IN} = 70 kΩ (typ.) R = 1 kΩ (typ.)

Note 1 : The 87PM43 does not have a pull-down resistor for TEST pin. Set to "0" TEST pin.
Note 2 : The input/output circuitries of the 87PM43 are the code NM1 type.

(2) Input/Output Ports

The input/output circuitries of the 87CK43/M43 input/output ports are shown below.
A mask option code is only "A".

PORT	I/O	INPUT / OUTPUT CIRCUITRY (CODE A)	REMARKS
P0	I/O		Tri-state I/O R = 1 kΩ (typ.)
P1	I/O		Tri-state I/O Hysteresis input R = 1 kΩ (typ.)
P2	I/O		Sink open drain output R = 1 kΩ
P60 to P65	I/O		Sink open drain output Analog input R = 1 kΩ (typ.) R_A = 5 kΩ (typ.) C_A = 12 pF (typ.)
P66 P67	I/O		Sink open drain output R = 1 kΩ (typ.)
P7	I/O		Sink open drain output R = 1 kΩ (typ.)

ELECTRICAL CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS (V_{SS} = 0 V)

PARAMETER	SYMBOL	PINS	RATINGS	UNIT
Supply Voltage	V_{DD}		– 0.3 to 6.5	V
Input Voltage	V_{IN}		– 0.3 to V_{DD} + 0.3	V
Output Voltage	V_{OUT1}	Ports P0, P1, P21, P22, P60~P65, $\overline{RESET}$, XOUT	– 0.3 to V_{DD} + 0.3	V
	V_{OUT2}	Ports P20, P66, P67, P7	– 0.3 to V_{DD} + 0.3	
Output Current (Per 1 pin)	I_{OUT1}	Ports P0, P1, P2, P6, P7	3.2	mA
Output Current (Total)	ΣI_{OUT1}	Ports P0, P1, P2, P6, P7	120	mA
Power Dissipation [Topr = 70°C]	PD		600	mW
Soldering Temperature (time)	Tsld		260 (10 s)	°C
Storage Temperature	Tstg		– 55 to 125	°C
Operating Temperature	Topr		– 30 to 70	°C

RECOMMENDED OPERATING CONDITIONS (V_{SS} = 0 V, Topr = – 30 to 70 °C)

PARAMETER	SYMBOL	PINS	CONDITIONS		Min.	Max.	UNIT
Supply Voltage	V_{DD}		fc = 8MHz	NORMAL1, 2 mode	4.5	5.5	V
				IDLE1, 2 mode			
			fs = 32.768 kHz	SLOW mode	2.7		
				SLEEP 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	V_{DD} = 4.5 to 5.5 V		2.0	8.0	MHz
	fs	XTIN, XTOUT	V_{DD} = 2.7 to 5.5 V		30.0	34.0	kHz

Note : fc : The condition of power supply voltage is limited to NORMAL1, NORMAL2, IDLE1, and IDLE2 mode.

D.C. CHARACTERISTICS (V_{SS} = 0 V, Topr = – 30 to 70 °C)

PARAMETER	SYMBOL	PINS	CONDITIONS	Min.	Typ.	Max.	UNIT
Hysteresis Voltage	V_{HS}	Hysteresis inputs		–	0.9	–	V
Input Current	I_{IN1}	TEST	V_{DD} = 5.5 V V_{IN} = 5.5 V / 0 V	–	–	± 2	μA
	I_{IN2}	Open drain ports and tri-state ports					
	I_{IN3}	$\overline{RESET}$, $\overline{STOP}$					
Input Resistance	R_{IN2}	$\overline{RESET}$		100	220	450	kΩ
Output Leakage Current	I_{LO}	Open drain ports and tri-state ports	V_{DD} = 5.5 V, V_{OUT} = 5.5 V	–	–	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_{OL}	Except XOUT	V_{DD} = 4.5 V, I_{OL} = 1.6 mA	–	–	0.4	V
Supply Current in NORMAL 1 , 2 mode	I_{DD}		V_{DD} = 5.5 V V_{IN} = 5.3 V / 0 .2 V fc = 8 MHz fs = 32.768 kHz	–	10	16	mA
Supply Current in IDLE 1, 2 mode				–	4.5	6	mA
Supply Current in SLOW mode			V_{DD} = 3.0 V V_{IN} = 2.8 V / 0 .2 V fs = 32.768 kHz	–	30	60	μA
Supply Current in SLEEP mode				–	15	30	μA
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 ; The current through pull-up or pull-down resistor is not included.

A / D CONVERSION 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
Analog Reference Voltage	V_{AREF}	$V_{DD} \geqq$ 4.5V, V_{SS} = 0V	V_{DD} – 1.5	—	V_{DD}	V
Analog Reference Voltage Range	$_{\Delta}V_{AREF}$		3.0	—	—	V
Analog Input Voltage Range	V_{AIN}		V_{SS}	—	V_{AREF}	V
Analog Supply Current	I_{REF}		—	0.5	1.0	mA
Nonlinearity Error		V_{DD} = 5.0V, V_{SS} = 0.000V V_{AREF} = 5.000V	—	—	± 1	LSB
Zero Point Error			—	—	± 1	
Full Scale Error			—	—	± 1	
Total Error			—	—	± 2	

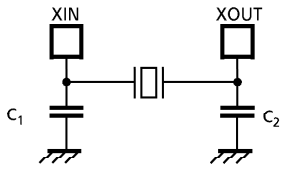
Note : $_{\Delta}V_{AREF} = V_{AREF} - V_{SS}$

A.C. CHARACTERISTICS (V_{SS} = 0 V, V_{DD} = 4.5 to 5.5 V, Topr = – 30 to 70 °C)

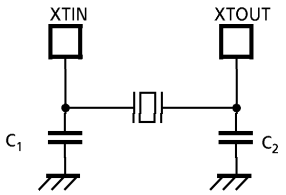
PARAMETER	SYMBOL	CONDITIONS	Min.	Typ.	Max.	UNIT
Machine Cycle Time	t_{cy}	In NORMAL 1, 2 mode	0.5	–	10	μs
		In IDLE 1, 2 mode				
		In SLOW mode	117.6	–	133.3	
		In SLEEP mode				
High Level Clock Pulse Width	t_{WCH}	For external clock operation (XIN input), fc = 8 MHz	62.5	–	–	ns
Low Level Clock Pulse Width	t_{WCL}					
High Level Clock Pulse Width	t_{WSH}	For external clock operation (XTIN input), fs = 32.768 kHz	14.7	–	–	μs
Low Level Clock Pulse Width	t_{WSL}					

RECOMMENDED OSCILLATING CONDITION (V_{SS} = 0 V, V_{DD} = 4.5 to 5.5 V, Topr = – 30 to 70 °C)

PARAMETER	OSCILLATOR	FREQUENCY	RECOMMENDED OSCILLATOR		RECOMMENDED CONDITION	
					C_1	C_2
High-frequency	Ceramic Resonator	8 MHz	KYOCERA	KBR8.0M	30 pF	30 pF
		4 MHz	KYOCERA	KBR4.0MS		
			MURATA	CSA4.00MG		
	Crystal Oscillator	8 MHz	TOYOCOM	210B 8.0000	20 pF	20 pF
		4 MHz	TOYOCOM	204B 4.0000		
Low-frequency	Crystal Oscillator	32.768 kHz	NDK	MX-38T	15 pF	15 pF



(1) High-frequency



(2) Low-frequency

Note: An electrical shield by metal shield plate on the surface of the IC package should be recommendable in order to prevent the device from the high electric fieldstress applied from CRT (Cathode Ray Tube) for continuous reliable operation.