

CMOS 8-BIT MICROCONTROLLER

TMP87CC20F , TMP87CH20F , TMP87CK20AF , TMP87CM20AF

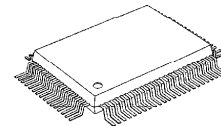
The 87CC20/H20/K20A/M20A are high-speed and high-performance 8-bit single chip microcomputers. These MCU contain a LCD driver, multi-function timer/counters and a serial interface on a chip.

PART No.	ROM	RAM	PACKAGE	OTP MCU
TMP87CC20F	12K × 8-bit	512 × 8-bit	QFP80-P-1420-0.80B	TMP87PH20F
TMP87CH20F	16K × 8-bit			TMP87PM20F
TMP87CK20AF	24K × 8-bit	1K × 8-bit		
TMP87CM20AF	32K × 8-bit			

FEATURES

- ◆ 8-bit single chip microcomputer TLCS-870 Series
- ◆ Instruction execution time : 0.5 μ s (at 8MHz), 122 μ s (at 32.768kHz)
- ◆ 412 basic instructions
 - Multiplication and Division (8bits × 8bits, 16bits ÷ 8bits)
 - Bit manipulations (set/clear/complement/move/test /exclusive or)
 - 16-bit data operations
 - 1-byte jump/subroutine-call (Short relative jump / Vector call)
- ◆ 13 interrupt sources (External : 4, Internal : 9)
 - All sources have independent latches each, and nested interrupt control is available.
 - 2 edge-selectable external interrupts with noise reject
 - High-speed task switching by register bank changeover
- ◆ 7 Input/Output ports (45 pins)
 - High current output : 2 pins (typ. 20mA)
- ◆ 16-bit Timer/Counter
 - Timer, Event counter, Pulse width measurement, Frequency measurement modes
- ◆ Four 8-bit Timer/Counters
 - Timer, Event counter, Capture (Pulse width/duty measurement), PWM output, Programmable divider output modes
- ◆ Time Base Timer (Interrupt frequency : 1Hz to 16kHz)
- ◆ Divider output function (frequency : 1kHz to 8kHz)
- ◆ Watchdog Timer
 - Interrupt source / reset output (programmable)
- ◆ 8-bit Serial Interface
 - With 8 bytes transmit/receive data buffer
 - Internal/external serial clock, and 4/8-bit mode
- ◆ LCD driver/Controller
 - LCD direct drive capability (max. 16-digit display at 1/4 duty LCD).
 - 1/4, 1/3, 1/2 duties or static drive are programmably selectable.
 - With display memory.
- ◆ 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 (32.768kHz).
 - 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 clock. Release by interrupts.
 - SLEEP mode : CPU stops, and peripherals operate using low-frequency clock. Release by interrupts.
- ◆ Wide operating voltage : 2.7 to 6 V at 4.19 MHz / 32.768 kHz, 4.5 to 6 V at 8 MHz / 32.768 kHz (87CC20 / H20, PH20)
2.7 to 5.5 V at 4.19 MHz / 32.768 kHz, 4.5 to 5.5 V at 8 MHz / 32.768 kHz (87CK20A / M20A, PM20)
- ◆ Emulation Pod : BM87CH20F0B

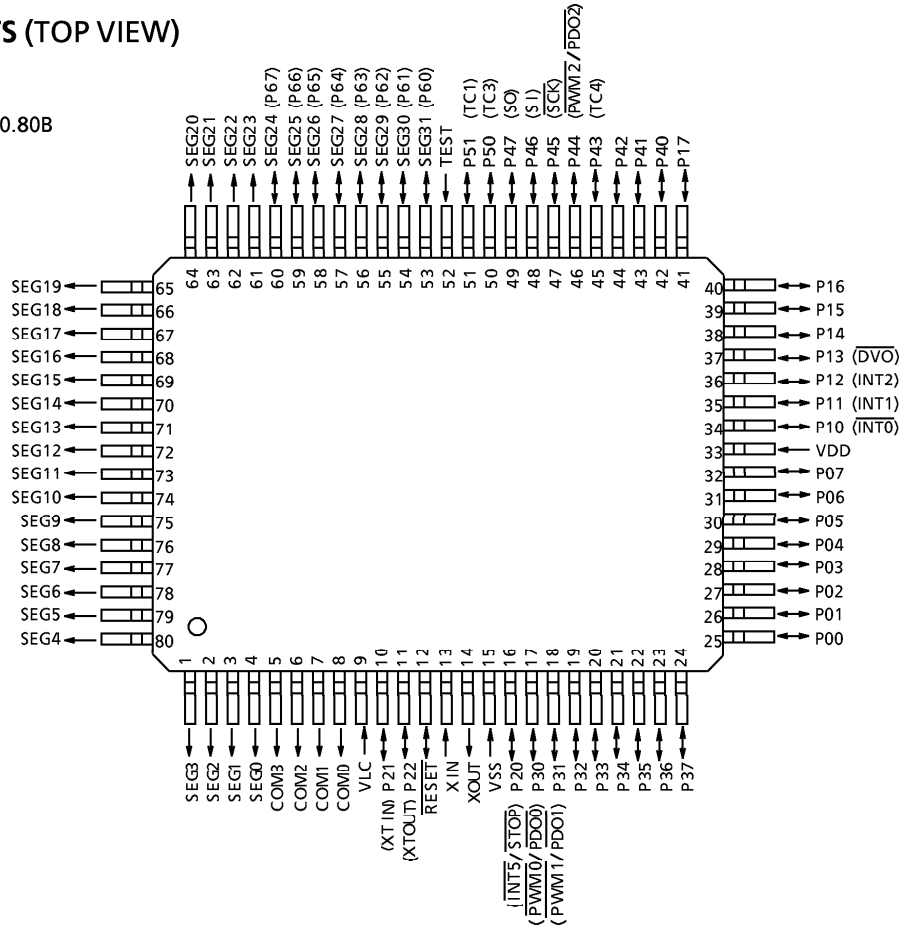
QFP80-P-1420-0.80B



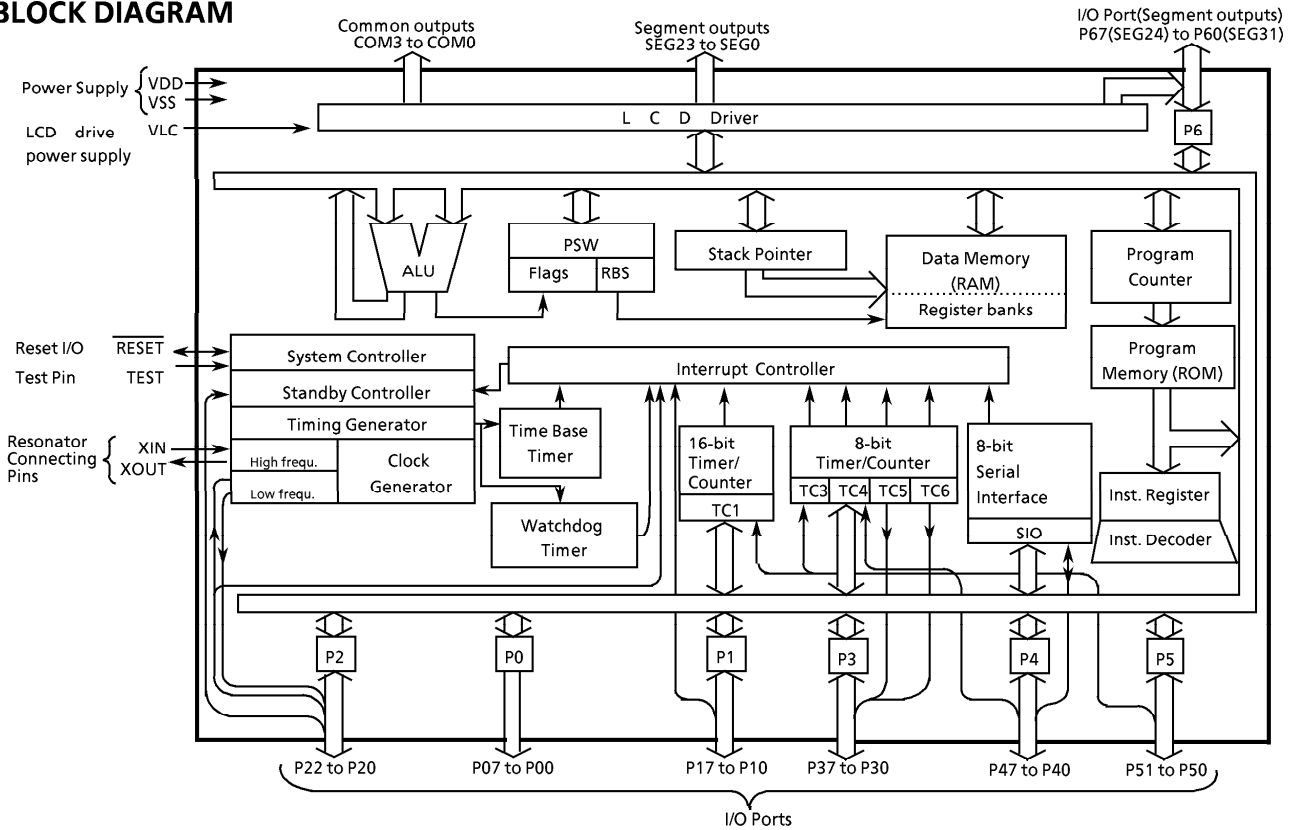
TMP87CC20F
 TMP87CH20F
 TMP87CK20AF
 TMP87CM20AF
 TMP87PH20F
 TMP87PM20F

PIN ASSIGNMENTS (TOP VIEW)

QFP80-P-1420-0.80B



BLOCK DIAGRAM



PIN FUNCTION

PIN NAME	Input / Output	FUNCTION		
P07 to P00	I/O	Two 8-bit programmable input/output ports (tri-state) .		
P17 to P14	I/O	Each bit of these ports can be individually configured as an input or an output under software control. During reset, all bits are configured as inputs. When used as a divider output, the latch must be set to "1".		
P13 (\overline{DVO})	I/O (Output)			Divider output
P12 (INT2)	I/O (Input)			External interrupt input 2
P11 (INT1)				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 (32.768kHz). For inputting external clock, XTIN is used and XTOUT is opened.	
P21 (XTIN)	I/O (Input)		External interrupt input 5 or STOP mode release signal input	
P20 ($\overline{INT5/STOP}$)				
P37 to P32	I/O	8-bit input/output port with latch. When used as an input port, a PWM output, or a PDO output, the latch must be set to "1".		
P31 (PWM1 / PDO1)	I/O (Output)			8-bit PWM1 output or 8-bit PDO1 output
P30 (PWM0 / PDO0)				8-bit PWM0 output or 8-bit PDO0 output
P47 (SO)	I/O (Output)	8-bit input/output port with latch. When used as an input port, a timer/counter input, a PWM output, a PDO output, or a SIO input/output, the latch must be set to "1".		
P46 (SI)	I/O (Input)			SIO serial data input
P45 (\overline{SCK})	I/O (I/O)			SIO serial clock input/output
P44 ($\overline{PWM2 / PDO2}$)	I/O (Output)	8-bit PWM2 output or 8-bit PDO2 output		
P43 (TC4)	I/O (Input)	Timer/Counter 4 input		
P42 to P40	I/O			
P51 (TC1)	I/O (Input)	2-bit input/output port with latch. When used as an input port or a timer/counter input, the latch must be set to "1".		
P50 (TC3)	I/O (Input)	Timer/Counter 3 input		
P67 (SEG24) to P60 (SEG31)	I/O (Output)	8-bit input/output port with latch. When used as an input port, the latch must be set to "1".		
SEG23 to SEG0	Output	LCD Segment outputs		
COM3 to COM0		LCD Common outputs		
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	+ 5V, 0V (GND)		
VLC		LCD drive power supply		

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 87CC20/H20/K20A/M20A.

In the TLCS-870 Series, the memory is organized 4 address spaces (ROM, RAM, SFR, and DBR). It uses a memory mapped I/O system, and all I/O registers are mapped in the SFR/DBR 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.

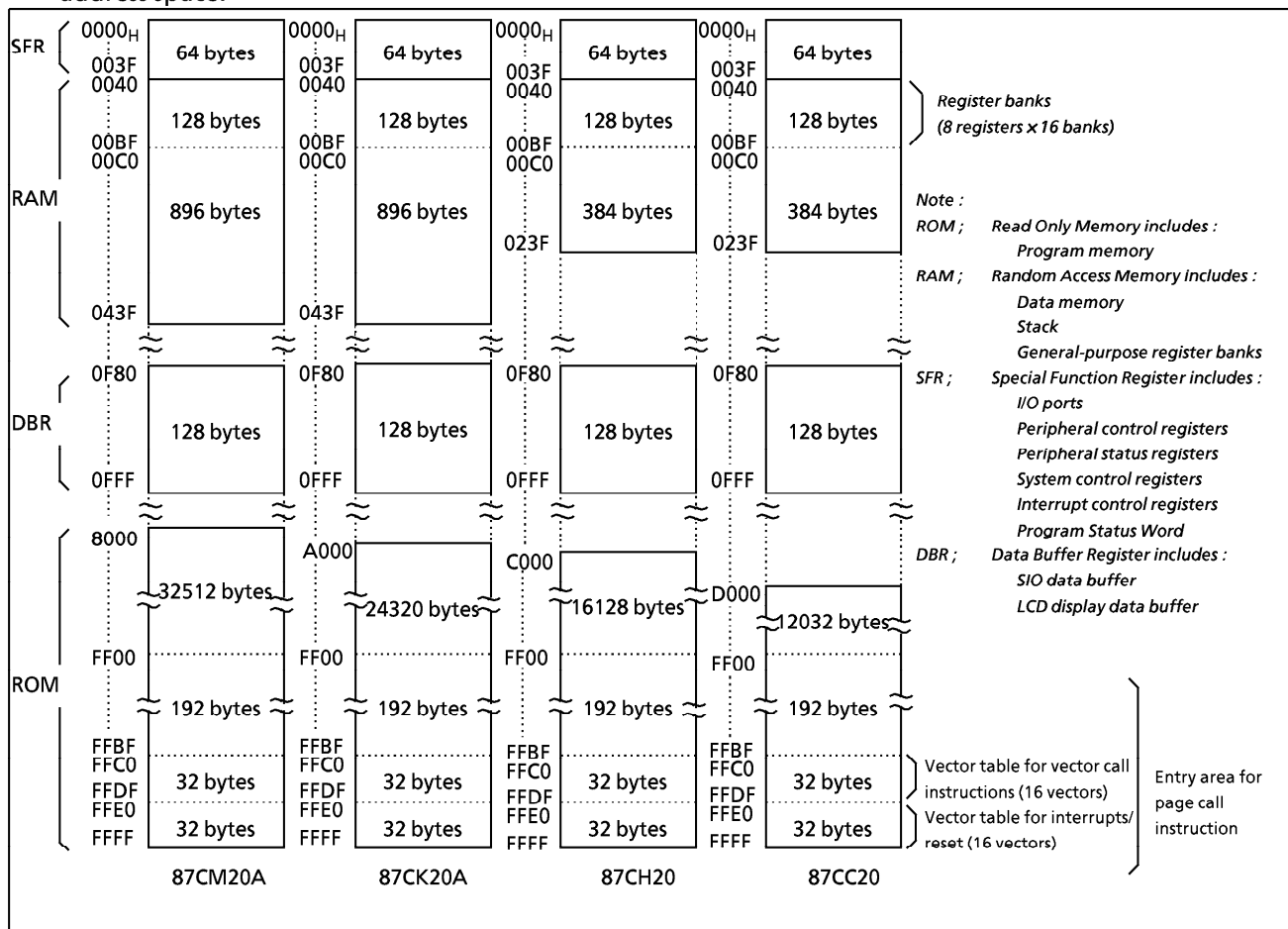


Figure 1-1. Memory Address Maps

1.2 Program Memory (ROM)

The 87CC20 has a 12K x 8-bit (addresses D000H-FFFFH), the 87CH20 has a 16K x 8-bit (addresses C000H-FFFFH), the 87CK20A has a 24K x 8-bit (addresses A000H-FFFFH), and the 87CM20A has a 32K x 8-bit (addresses 8000H-FFFFH) of program memory (mask programmed ROM).

Addresses FF00H-FFFFH in the program memory can also be used for special purposes.

(1) **Interrupt/Reset vector table (addresses FFE0H-FFFFH)**

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 addresses 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.

- 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 current 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-byte space.

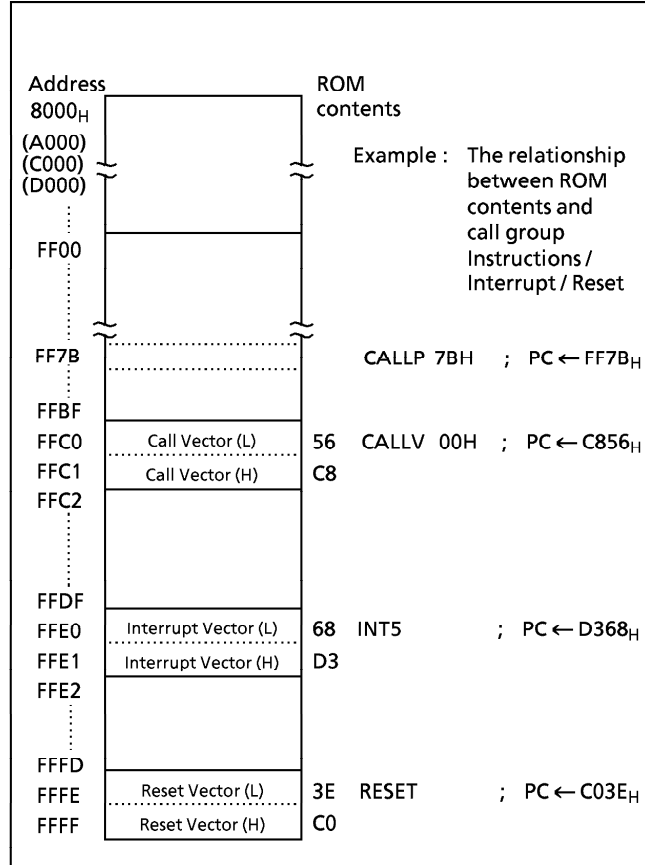


Figure 1-2. Program Memory Map

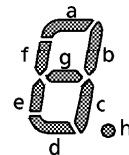
In the TLCS-870 Series, the same instruction used to access the data memory (e.g. [LD A, (HL)]) is also used to read out fixed data (ROM data) stored in the program memory. The register-offset-PC-relative-addressing (PC + A) instructions can also be used, and the code conversion, table look-up and n-way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

Example 1 : Converts BCD to 7-segment code (common anode LED). When A = 05_H, 92_H is output to port P3 after executing the following program:

```

ADD  A, TABLE - $ - 4      ; P3 ← ROM (TABLE + A)
LD   (P3), (PC + A)
JRS  T, SNEXT
TABLE : DB 0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0D8H, 80H, 98H
SNEXT :
    
```

Notes : "\$" is a header address of the ADD instruction.
DB is a byte data definition instruction.



Example 2 : N-way multiple jump in accordance with the contents of accumulator ($0 \leq A \leq 3$):

```

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.

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, 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.

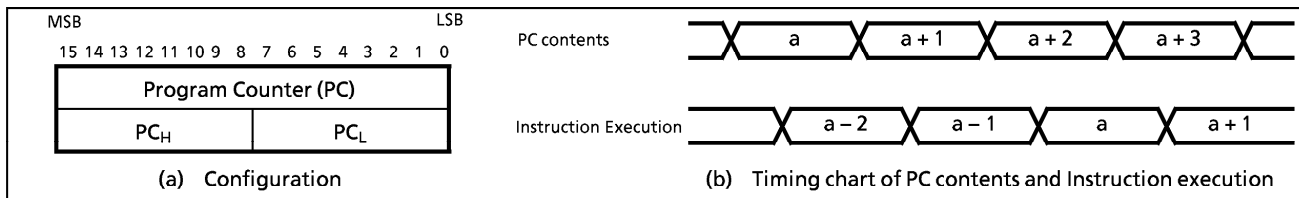


Figure 1-3. Program Counter

1.4 Data Memory (RAM)

The 87CC20/H20 each have 512 × 8-bit (addresses 0040_H-023F_H), the 87CK20A/M20A have 1K × 8-bit (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 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".

With the **TLCS-870 series**, programs in data memory cannot be executed. If the program counter indicates a specific data memory address (addresses 0040_H - 023F_H), an address-trap-reset is generated due to bus error. (Output from the RESET pin goes low.)

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 that general-purpose registers are mapped in the RAM; therefore, *do not clear RAM at the current bank address*.

Example 1 : Clears RAM to "00_H" except the bank 0 (87CC20/CH20, 87PH20)

```

LD    HL, 0048H        ; Sets start address to HL register pair
LD    A, H             ; Sets initial data (00H) to A register
LD    BC, 01F7H       ; Sets number of byte to BC register pair
SRAMCLR: LD    (HL+), A
DEC    BC
JRS   F, SRAMCLR
    
```

Example 2 : Clears RAM to "00_H" except the bank 0 (87CK20A/CM20A, 87PM20)

```

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
    
```

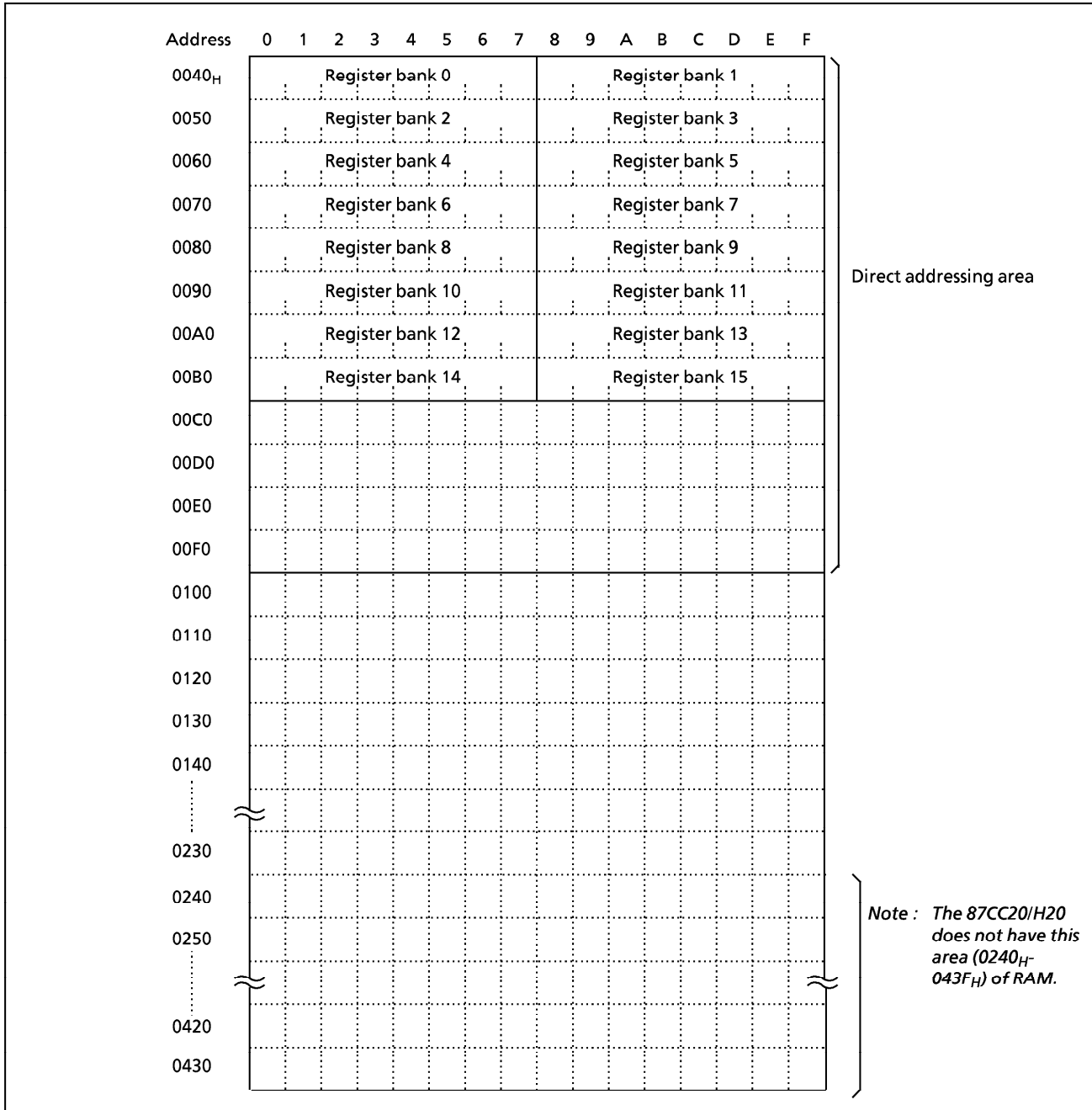


Figure 1-4. Data Memory Map

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.

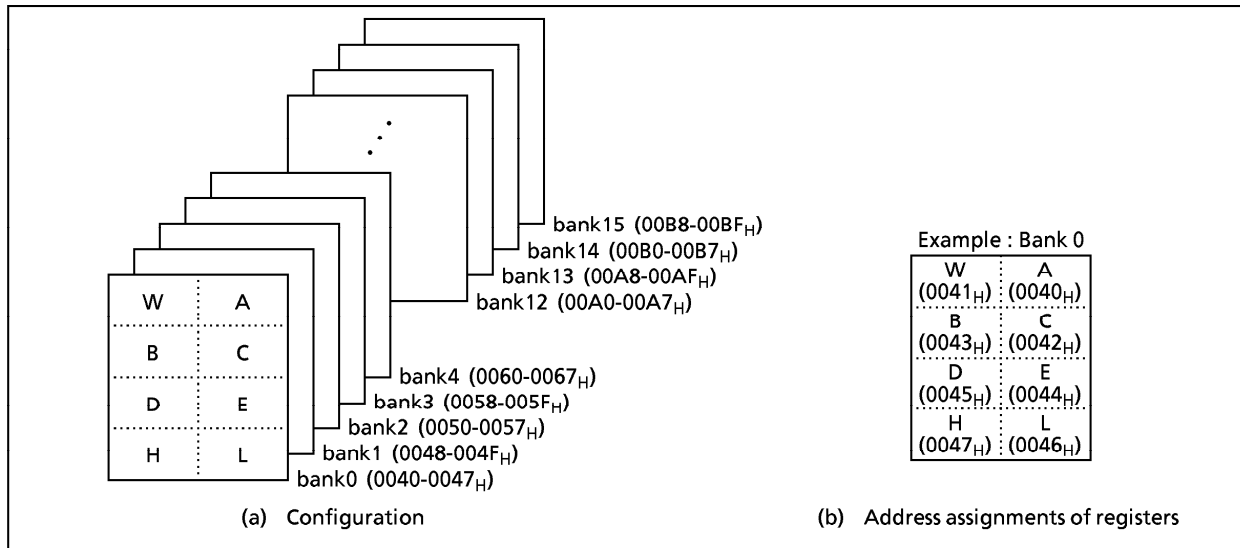


Figure 1-5. General-purpose Register Banks

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 the WA 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.

- Example :
- ① ADD A, B ; Adds B contents to A contents, and stores the result into A.
 - ② SUB WA, 1234H ; Subtracts 1234_H 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 can 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 functions as a data pointer (DE). The HL also has an auto-post-increment and an 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 52_H 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
JRS F, SLOOP
```

(3) B, C, BC

Registers B and C can be used in 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 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 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 flags, the RBS is assigned to address 003FH 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 task, and one bank can be assigned to each interrupt task. 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. The PSW is assigned to address 003FH 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. [LD RBS, n], [PUSH PSW] and [POP PSW] are the PSW access instructions.

1.6.1 Register Bank Selector (RBS)

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

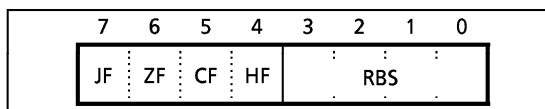


Figure 1-6. PSW (Flags, RBS) Configuration

1.6.2 Flags

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.

cc	Means	Condition
T	True	JF = 1
F	False	JF = 0
Z	Zero	ZF = 1
NZ	Not Zero	ZF = 0
CS	Carry Set	CF = 1
CC	Carry Clear	CF = 0
EQ	Equal	ZF = 1
NE	Not Equal	ZF = 0
LT	Unsigned Less Than	CF = 1
GE	Unsigned Greater Than or Equal	CF = 0
LE	Unsigned Less Than or Equal	(CF∨ZF) = 1
GT	Unsigned Greater Than	(CF∨ZF) = 0

(1) Zero flag (ZF)

The ZF is set to "1" if operation result or transfer data is 00H (for 8-bit operations and data transfers) / 0000H (for 16-bit operations); otherwise it is cleared to "0".

During the bit manipulation instruction [SET, CLR and CPL], it is also set to "1" if the contents of the specified bit is "0"; Otherwise it is cleared to "0". This flag is set to "1" when the upper 8 bits of the product are 00H during the multiplication instruction [MUL], and when 00H 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 00H (divided by zero error), or when the quotient is 100H or higher (Quotient 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 [ADD and ADDC], or when a borrow occurred from bit 4 into bit 3 of the result during an 8-bit subtraction [SUB, SUBB, CMP and MCMP]; 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).

Note that an undefined data is set to the HF during a 16-bit operation.

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 normally 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 : The accumulator and flags become as shown below after executing the following instructions when the WA register pair, the HL register pair, 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	Flags 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	Flags after execution			
		JF	ZF	CF	HF
INC A	9B	0	0	1	0
ROL A	35	1	0	1	0
RORC A	CD	0	0	0	0
SET A.5	BA	1	1	1	0
ADD WA, 0F508H	16A2	1	0	1	0
MUL W, A	13DA	0	0	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 of 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, a push, a software interrupt 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.

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).

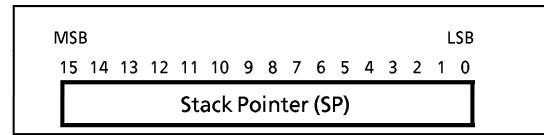


Figure 1-7. Stack Pointer

Example 1 : To initialize the SP

```
LD    SP, 023FH    ; SP←023FH
```

Example 2 : To read the SP

```
LD    HL, SP      ; HL←SP
```

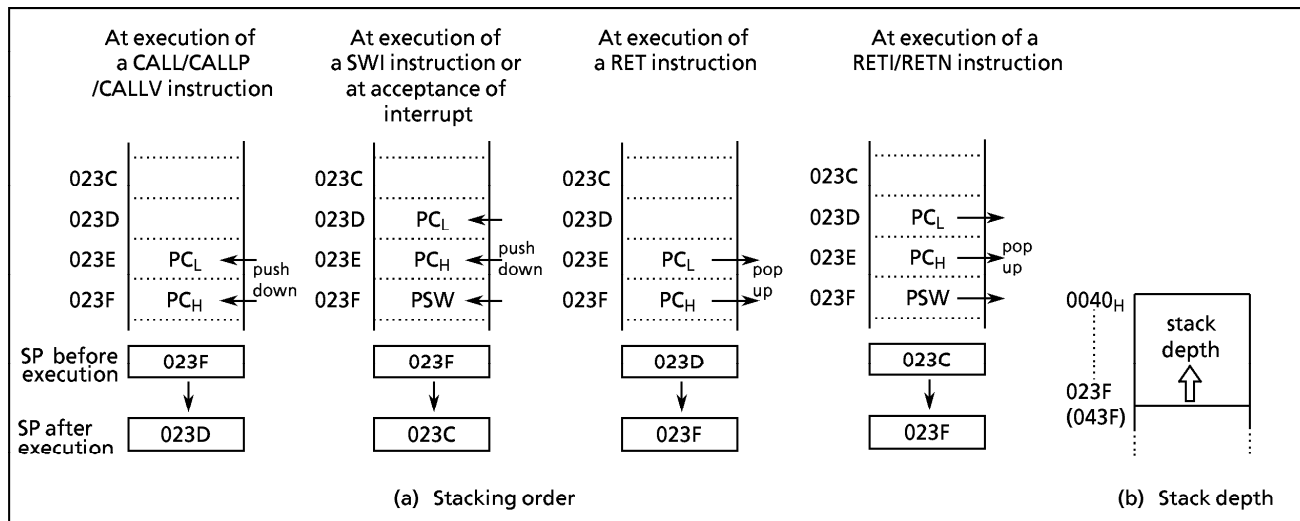


Figure 1-8. Stack

1.8 System Clock Controller

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

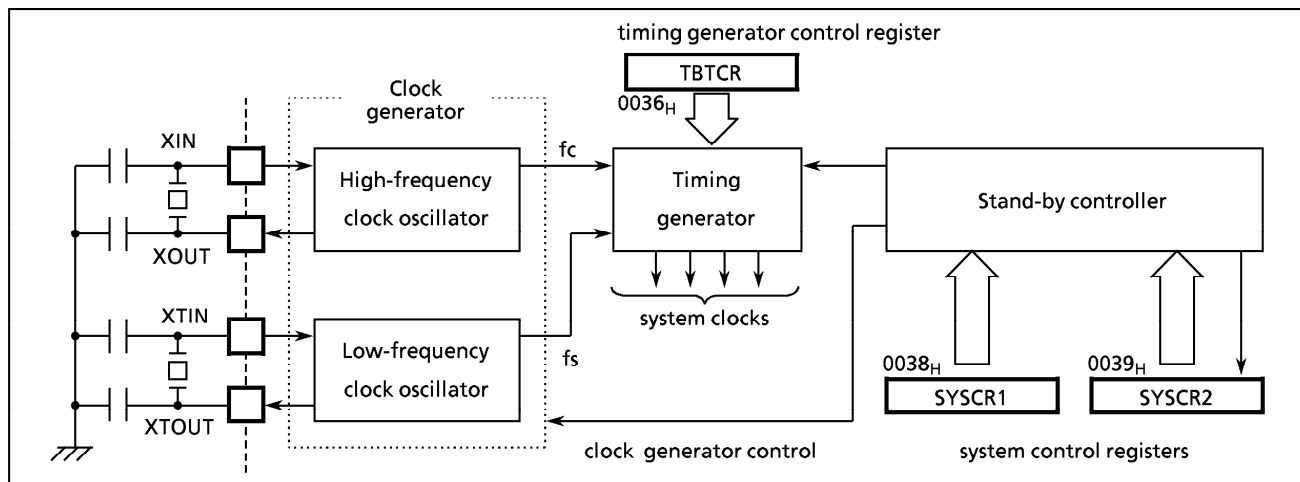


Figure 1-9. System Clock Controller

1.8.1 Clock Generator

The clock generator generates the basic clock which provides the system clocks supplied to the CPU core and on-chip 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 (f_c) and low-frequency (f_s) 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 87CC20/H20/K20A/M20A are not provided an RC oscillation.

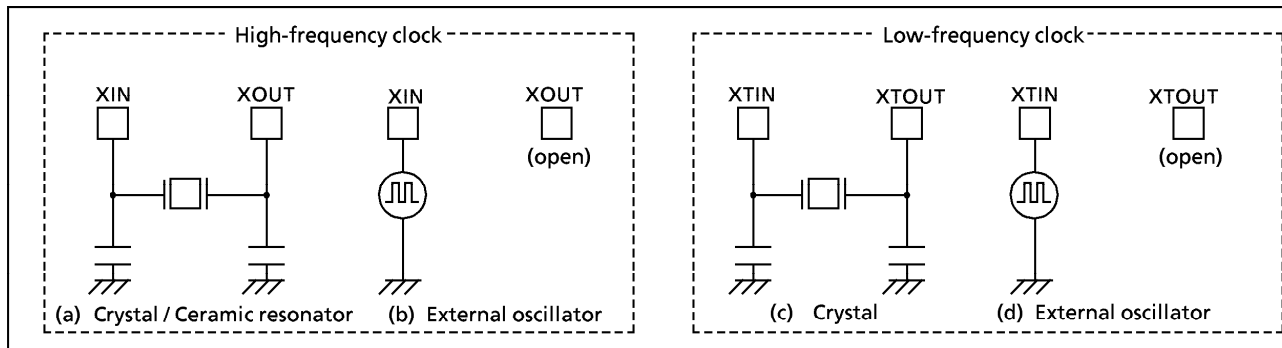
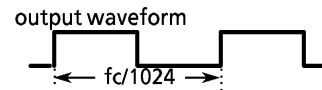


Figure 1-10. Examples of Resonator Connection

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 providing a 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 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 on-chip 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, TC3 – TC6)
- ⑥ Generation of internal serial clocks for serial interface (SIO)
- ⑦ Generation of warm-up clocks for releasing STOP mode
- ⑧ Generation of clocks for releasing reset output
- ⑨ Generation of base clocks for LCD driver/controller

(1) Configuration of timing generator

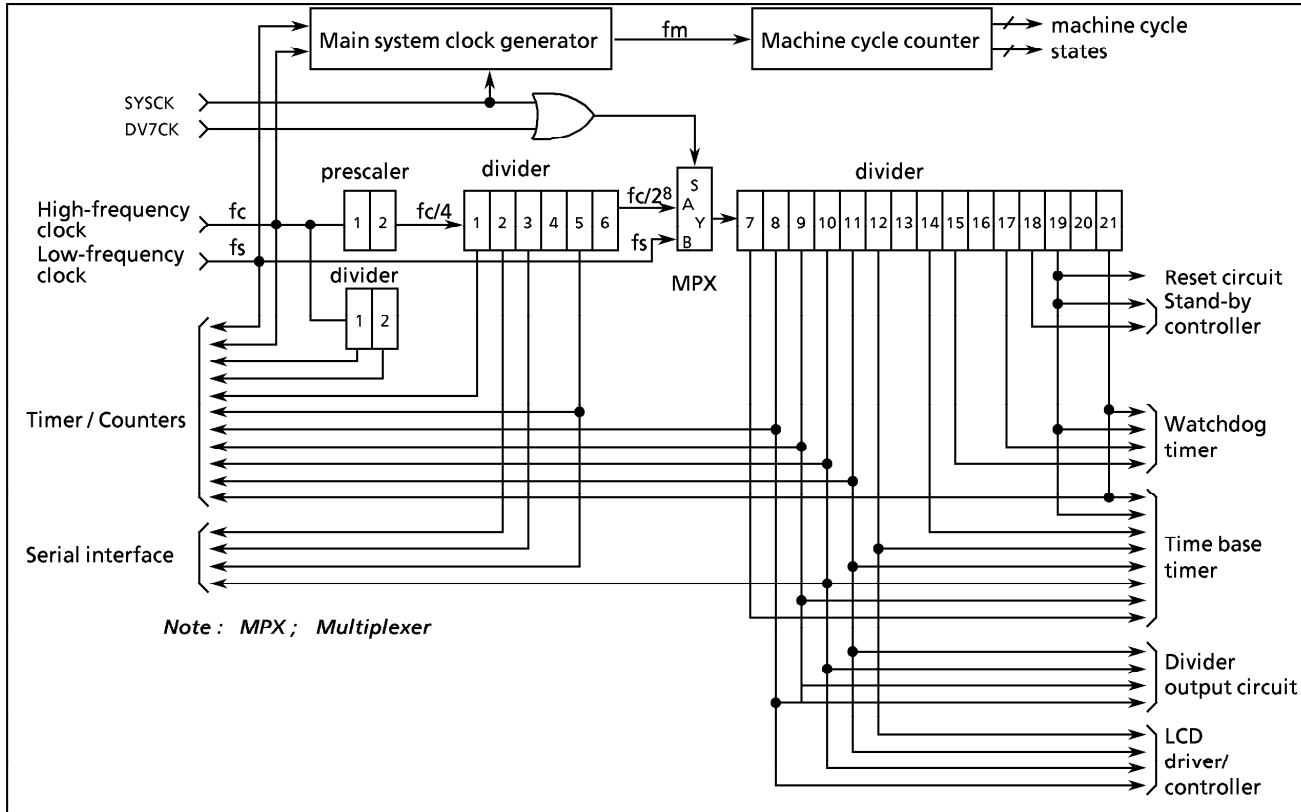


Figure 1-11. Configuration of Timing Generator

The timing generator consists of a 21-stage divider with a divided-by-4 prescaler, and 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 upon 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 ($fc/28$) 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 " $fc/28$ " 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.

TBTCR (0036 _H)	7	6	5	4	3	2	1	0	(Initial value: 0**0 0***)
	(DVOEN)	(DVQCK)	DV7CK	(TBTEN)	(TBTCK)				
	DV7CK	Selection of input clock to the 7th stage of the divider				0 : $fc/28$ [Hz] 1 : fs		write only	

Note1 : fc ; high-frequency clock [Hz], fs ; low-frequency clock [Hz], * ; don't care
 Note2 : Do not set DV7CK to "1" in the single-clock mode.
 Note3 : Do not set DV7CK to "1" before low-frequency clock is stable in the dual-clock mode.
 Note4 : 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 on-chip 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.

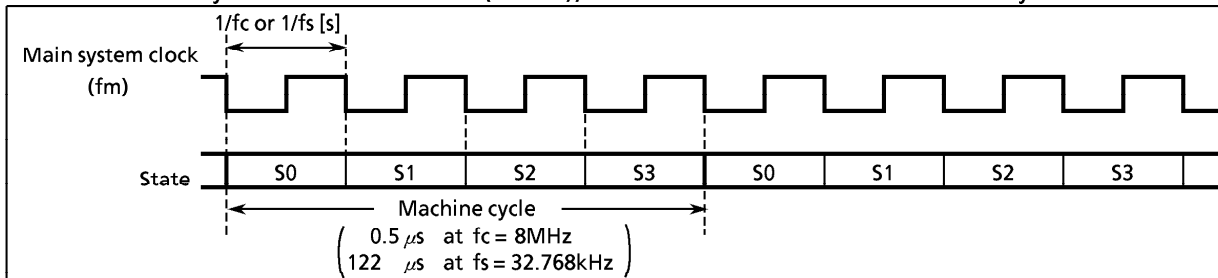


Figure 1-13. Machine Cycle

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 a mask option during reset.

(1) Single-clock mode

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

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 87CC20/H20/K20A/M20A 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 the 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 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 the 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 input (either level-sensitive or edge-sensitive can be programmably selected) to the $\overline{\text{STOP}}$ 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 the 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 $4/f_c$ [s] ($0.5\mu\text{s}$ at $f_c = 8\text{MHz}$) in NORMAL2/IDLE2 modes, and $4/f_s$ [s] ($122\mu\text{s}$ at $f_c = 32.768\text{kHz}$) in SLOW/SLEEP modes.

Note that the 87PH20/M20 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 is operated using the high-frequency clock. The on-chip peripherals are operated using the high-frequency clock and / or low-frequency clock. In case that the dual-clock mode has been selected as an option, the 87CC20/H20/K20A/M20A 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 are operated 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

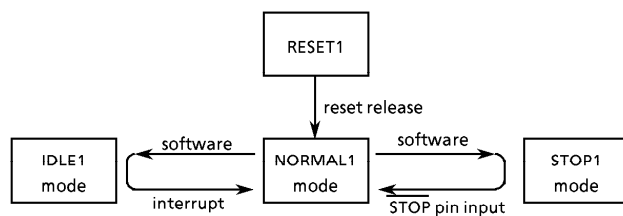
In this mode, the internal oscillation circuits remain active. The CPU and the watchdog timer are halted; however, on-chip peripherals operate using the high-frequency clock and / or low-frequency clock. Starting and releasing of IDLE2 mode are the same as for IDLE1 mode, except that operation returns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

④ SLEEP mode

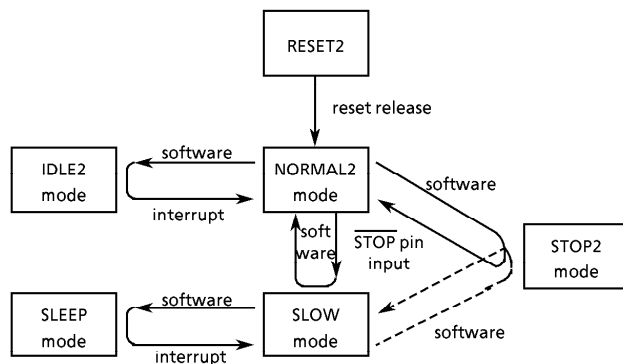
In this mode, the internal oscillation circuit of the low-frequency clock remains active. The CPU, the watchdog timer, and the internal oscillation circuit of the high-frequency clock are halted; however, on-chip peripherals operate using the low-frequency clock. Starting and releasing of SLEEP mode 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

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

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		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	halt

Figure 1-14. Operating Mode Transition Diagram

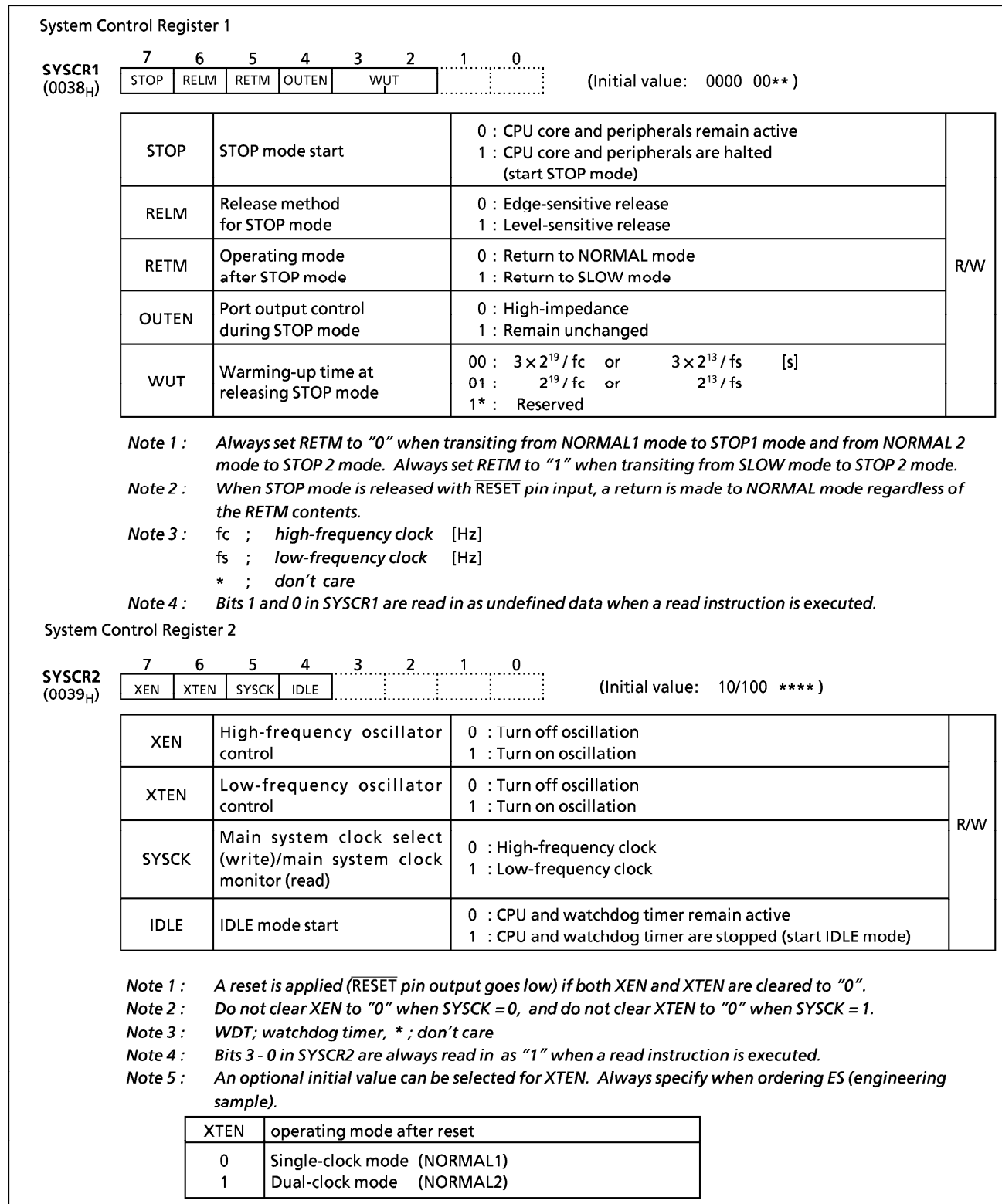


Figure 1-15. System Control Registers

1.8.4 Operating Mode Control

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

STOP mode is controlled by the system control register 1 (SYSCR1) and the $\overline{\text{STOP}}$ pin input. The $\overline{\text{STOP}}$ pin is also used both as a port P20 and an $\overline{\text{INT5}}$ (external interrupt input 5) pin. STOP mode is started by setting 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 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 $\overline{\text{STOP}}$ pin high. This mode is used for capacitor back-up when the main power supply is cut off and for long term battery back-up.

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

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

Example : Starting STOP mode with an INT5 interrupt.

```

PINT5:  TEST  (P2) . 0           ; To reject noise, STOP mode does not start if port P20
        JRS   F, SINT5          is at high
        LD   (SYSCR1), 01010000B ; Sets up the level-sensitive release mode.
        SET  (SYSCR1) . 7       ; Starts STOP mode
SINT5:  RETI
    
```

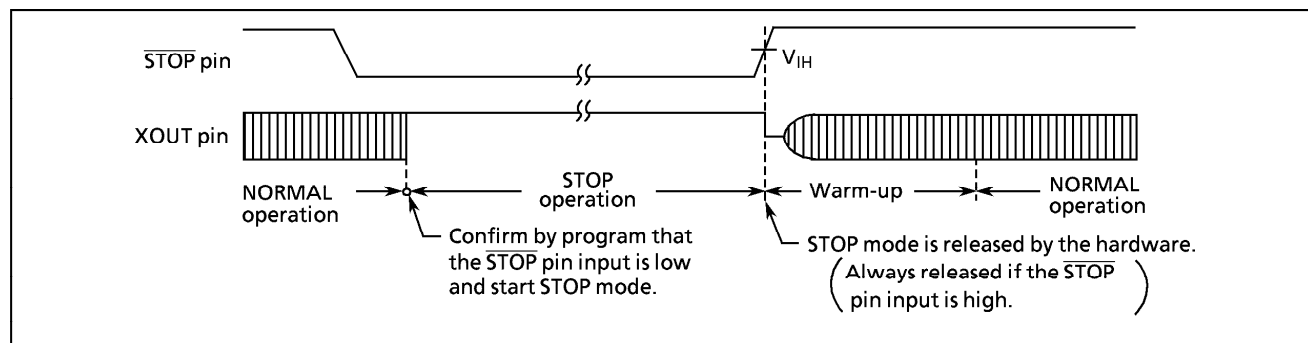


Figure 1-16. Level-sensitive Release Mode

Note : When changing to the level-sensitive 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 the 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, the STOP mode is started even when the \overline{STOP} pin input is "H" level.

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

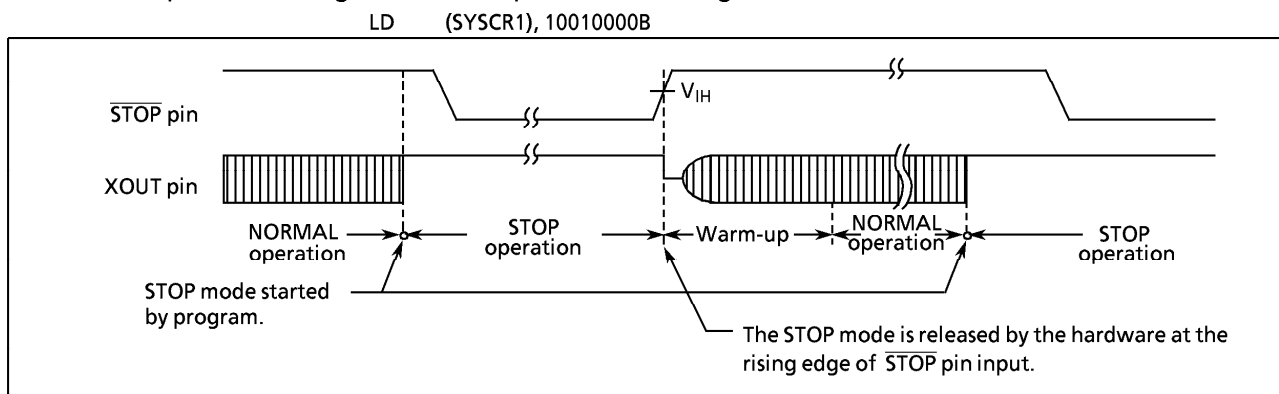


Figure 1-17. Edge-sensitive Release Mode

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
$3 \times 2^{19} / fc$ [s]	375 [ms]	196.6 [ms]	$3 \times 2^{13} / fs$ [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 $\overline{\text{RESET}}$ pin low, which immediately performs the normal reset operation.

*Note : When STOP mode is released with a low hold voltage, the following cautions must be observed.
The power supply voltage must be at the operating voltage level before releasing STOP mode. The $\overline{\text{RESET}}$ pin input must also be 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).*

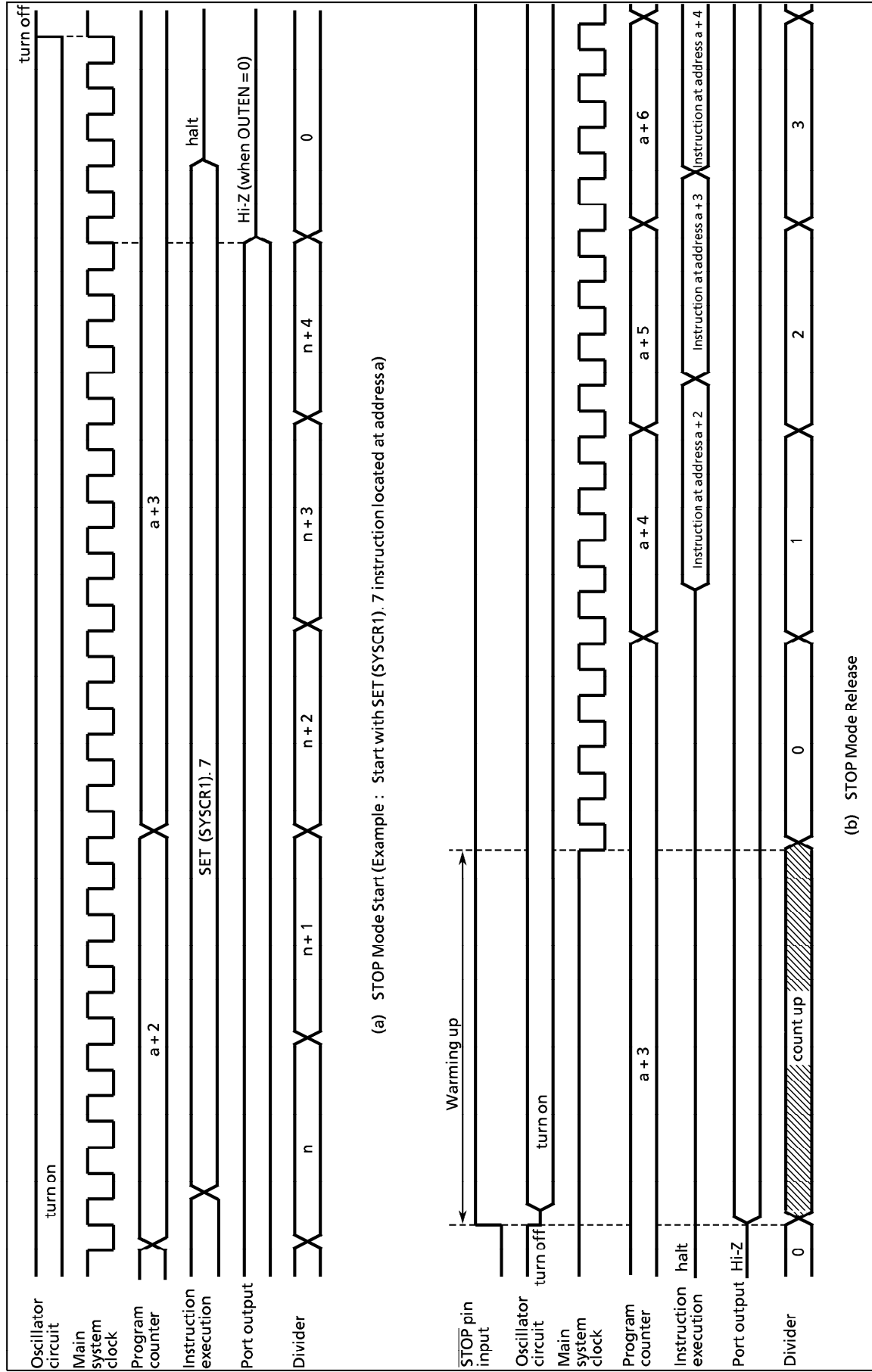


Figure 1-18. STOP Mode Start / Release

(2) IDLE mode (IDLE1, IDLE2, SLEEP)

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

- ① Operation of the CPU and watchdog timer is halted. The peripheral hardware continues 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
```

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]).

b. Interrupt release mode (IMF = "1")

IDLE mode is released and interrupt processing is started by any interrupt source enabled by the individual interrupt enable flag (EF) or an external interrupt 0 ($\overline{\text{INT0}}$ pin) request. After the interrupt is processed, 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 87CC20/H20/K20A/M20A are placed in NORMAL mode.

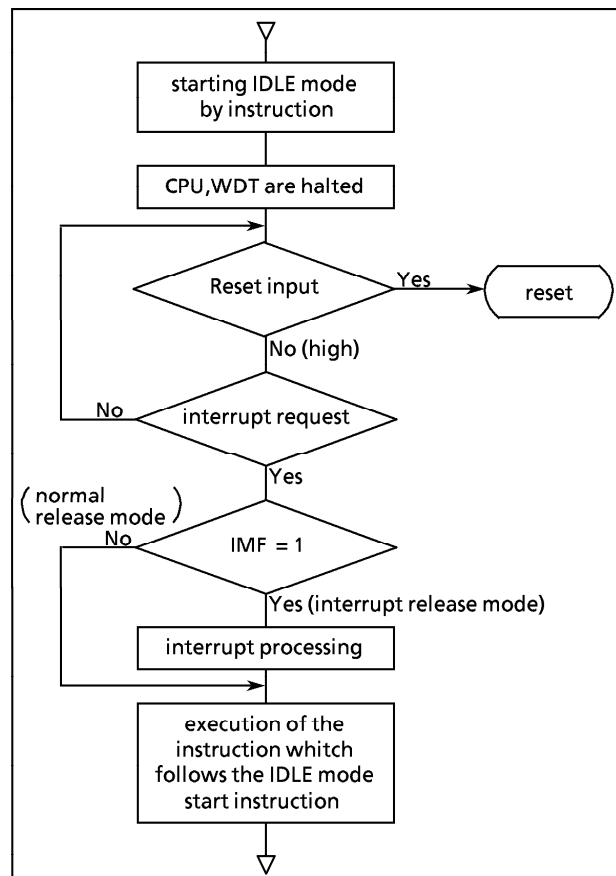


Figure1-19. IDLE Mode

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

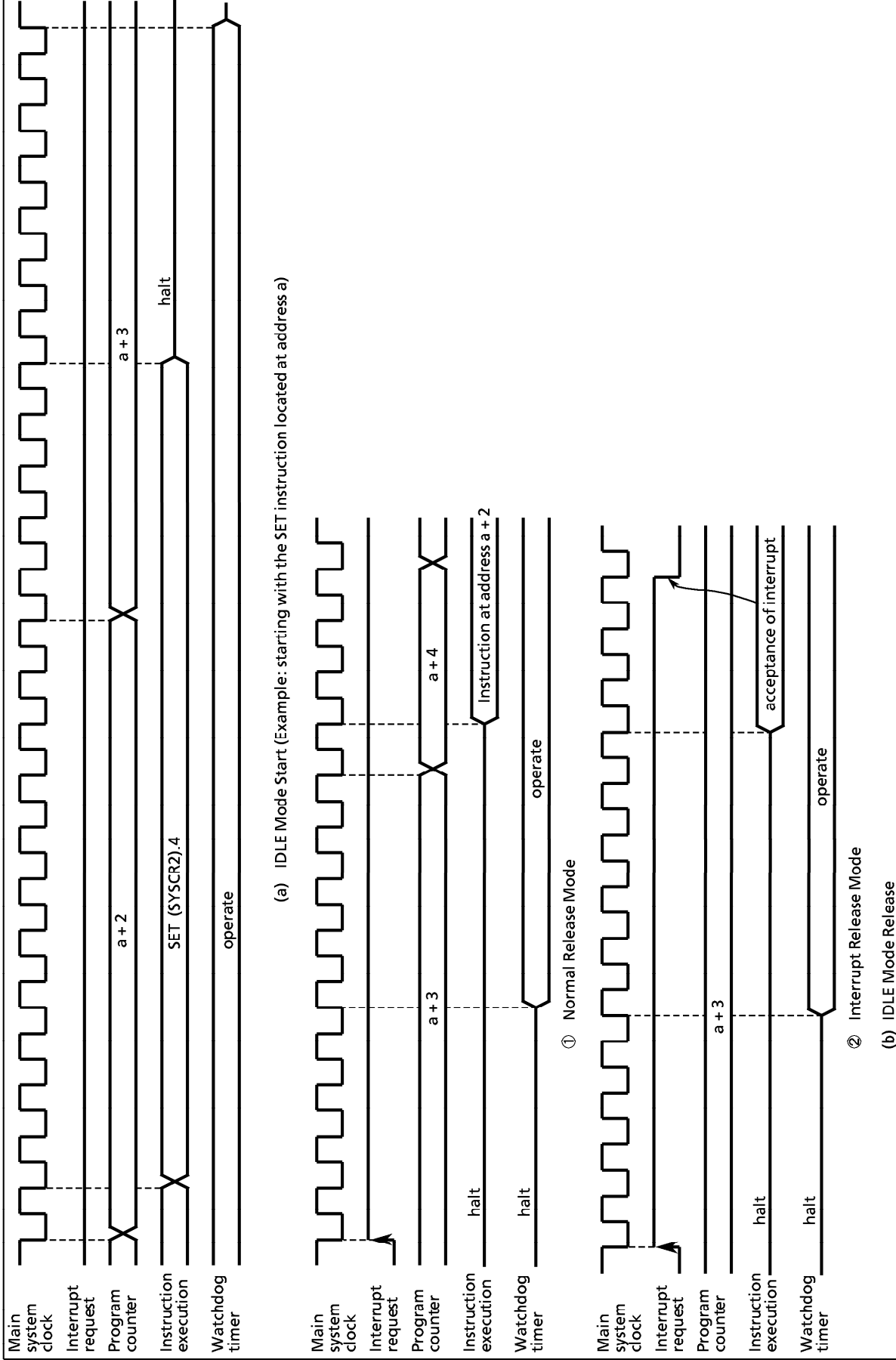


Figure 1-20. IDLE Mode Start/Release

(3) SLOW mode

SLOW mode is controlled by the system control register 2 (SYSCR2) and the timer / counter 1 (TC1).

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 from the high-frequency clock. Next, clear XEN (bit 7 in SYSCR2) to turn off high-frequency oscillation.

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

When the low-frequency clock oscillation is unstable, wait until oscillation stabilizes before performing the above operations. The timer/counter1 (TC1) can conveniently be used to confirm that low-frequency clock oscillation has stabilized.

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     (TC1CR), 04H  ; Sets TC1 mode
                    (timer mode, source clock : fs)
```

```
LDW   (TREG1A), 8000H ; Sets warming-up time
                    (according to Xtal characteristics)
```

```
SET   (EIRH).EF8    ; Enables INTTC1
```

```
LD    (TC1CR), 44H  ; Starts TC1
```

```
⋮
```

```
PINTTC1: LD    (TC1CR), 04H  ; Stops TC1
```

```
SET   (SYSCR2).5    ; SYSCK←1
```

```
CLR   (SYSCR2).7    ; XEN←0
```

```
RETI
```

```
⋮
```

```
VINTTC1: DW    PINTTC1      ; INTTC1 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 1 (TC1), clear SYSCK (bit 5 in SYSCR2) to switch the main system clock to the high-frequency clock.

SLOW mode can be released by setting the $\overline{\text{RESET}}$ pin low, which immediately performs the reset operation. After reset, the 87CC20/H20/K20A/M20A are placed in NORMAL mode.

Example : Switching from SLOW mode to NORMAL2 mode ($f_c = 8\text{MHz}$, warming-up time is 7.9ms).

```

SET      (SYSCR2) . 7      ; XEN←1 (turns on high-frequency oscillation)
LD       (TC1CR), 00H     ; Sets the TC1 mode
                          ; (Timer mode, source clock: fc)
LD       (TREG1A + 1), 0F8H ; Sets the warming up time
                          ; (according to frequency and resonator characteristics).
SET      (EIRH).EF8      ; Enables INTTC1
LD       (TC1CR), 40H     ; Starts TC1
        :
PINTTC1: LD       (TC1CR), 00H ; Stops TC1
        CLR      (SYSCR2) . 5 ; SYSCK←0
                          ; (switches the main system clock to the high-frequency clock)
        RETI
        :
VINTTC1: DW      PINTTC1   ; INTTC1 vector table

```

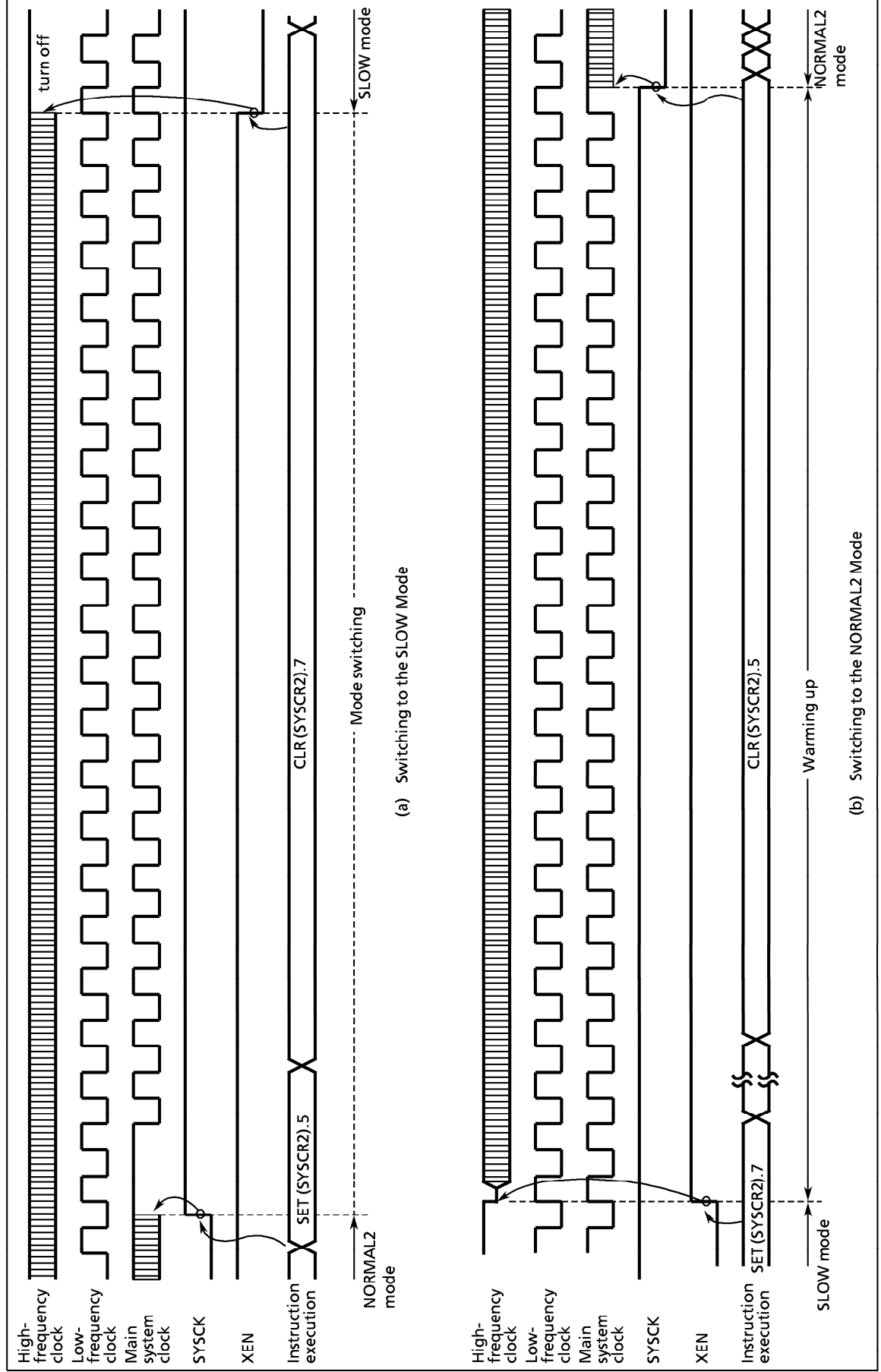


Figure 1-21. Switching between the NORMAL2 and SLOW Modes

1.9 Interrupt Controller

The 87CC20/H20/K20A/M20A each have a total of 13 interrupt sources: 4 externals and 9 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)	non-maskable	IL ₂	FFFA _H	2
External INT0 (External Interrupt 0)	IMF = 1, INT0EN = 1	IL ₃	FFF8 _H	3
reserved	IMF · EF ₄ = 1	IL ₄	FFF6 _H	4
External INT1 (External Interrupt 1)	IMF · EF ₅ = 1	IL ₅	FFF4 _H	5
Internal INTTBT (Time Base Timer Interrupt)	IMF · EF ₆ = 1	IL ₆	FFF2 _H	6
External INT2 (External Interrupt 2)	IMF · EF ₇ = 1	IL ₇	FFF0 _H	7
Internal INTTC1 (16-bit TC1 Interrupt)	IMF · EF ₈ = 1	IL ₈	FFEE _H	8
Internal INTSIO (Serial Interface Interrupt)	IMF · EF ₉ = 1	IL ₉	FFEC _H	9
Internal INTTC4 (8-bit TC4 Interrupt)	IMF · EF ₁₀ = 1	IL ₁₀	FFEA _H	10
Internal INTTC3 (8-bit TC3 Interrupt)	IMF · EF ₁₁ = 1	IL ₁₁	FFE8 _H	11
Internal INTTC5 (8-bit TC5 Interrupt)	IMF · EF ₁₂ = 1	IL ₁₂	FFE6 _H	12
Internal INTTC6 (8-bit TC6 Interrupt)	IMF · EF ₁₃ = 1	IL ₁₃	FFE4 _H	13
reserved	IMF · EF ₁₄ = 1	IL ₁₄	FFE2 _H	14
External INT5 (External Interrupt 5)	IMF · EF ₁₅ = 1	IL ₁₅	FFE0 _H	Low 15

(1) Interrupt Latches (IL_{15~2})

Interrupt latches are provided for each source, except for 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.

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 instructions such as bit manipulation or operation instructions cannot be used (Do not clear the IL₂ for a watchdog timer interrupt to "0")*. Thus, interrupt requests can be cancelled and initialized by the program. 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 : Clear interrupt latches

```
LDW (IL), 1110100000111111B ; IL12, IL10~IL6 ← 0
```

Example 2 : Read interrupt latches

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

Example 3 : Tests an interrupt latch

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

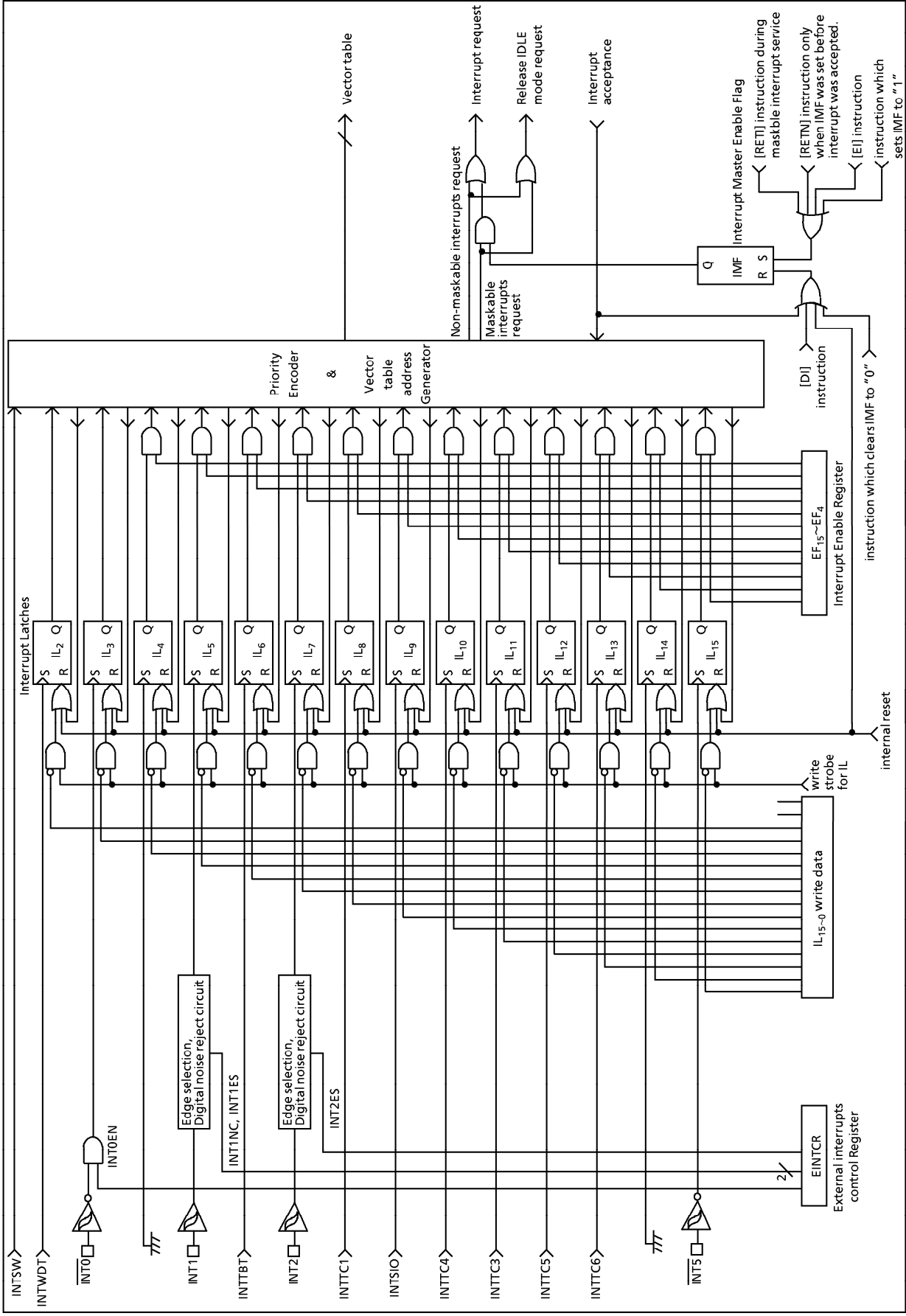


Figure 1-22. Interrupt Controller Block Diagram

(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 same pseudo non-maskable interrupts cannot be used 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). This 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).

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 other 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 processing 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 processing is started with interrupt acceptance enabled (IMF = 1).

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

Note : Do not set IMF to "1" during non-maskable interrupt service programs.

② **Individual interrupt Enable Flags (EF₁₅~EF₅)**

These flags enable and disable the acceptance of individual maskable interrupts, except for 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.

Example1 : Sets EF for individual interrupt enable and Sets IMF to "1"

```
LDW (EIR), 1010100010100001B ; EF15, EF13, EF11, EF7, EF5, IMF←1
```

Example2 : Sets an individual interrupt enable flag to "1"

```
SET (EIRH). 4 ; EF12←1
```

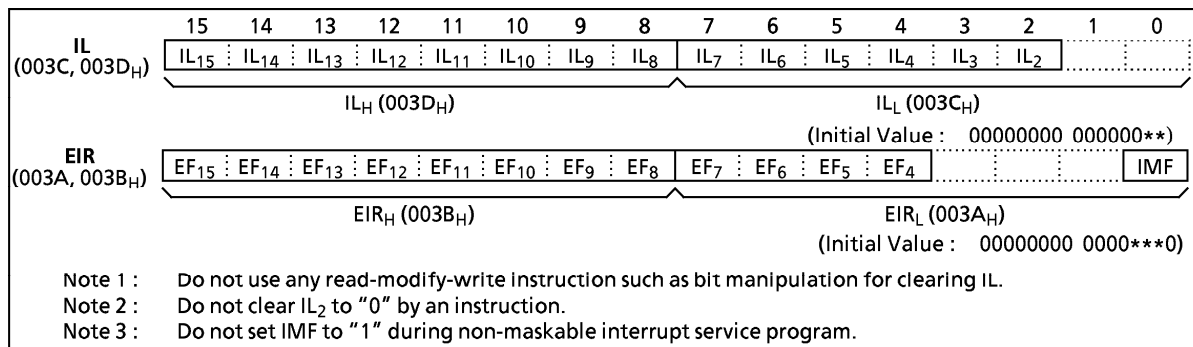


Figure 1-23. Interrupt Latch (IL) , Interrupt Enable Register (EIR)

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 @ fc = 8MHz in NORMAL mode) after the completion of the current instruction execution. The interrupt service task terminates upon execution of the interrupt return instruction [RETI] (for maskable interrupts) or [RETN] (for pseudo non-maskable interrupts).

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) on the stack.
- ④ The entry address of the interrupt service program is read from the vector table address for the interrupt source and load to the program counter.
- ⑤ The instruction stored at the entry address of the interrupt service program is executed.

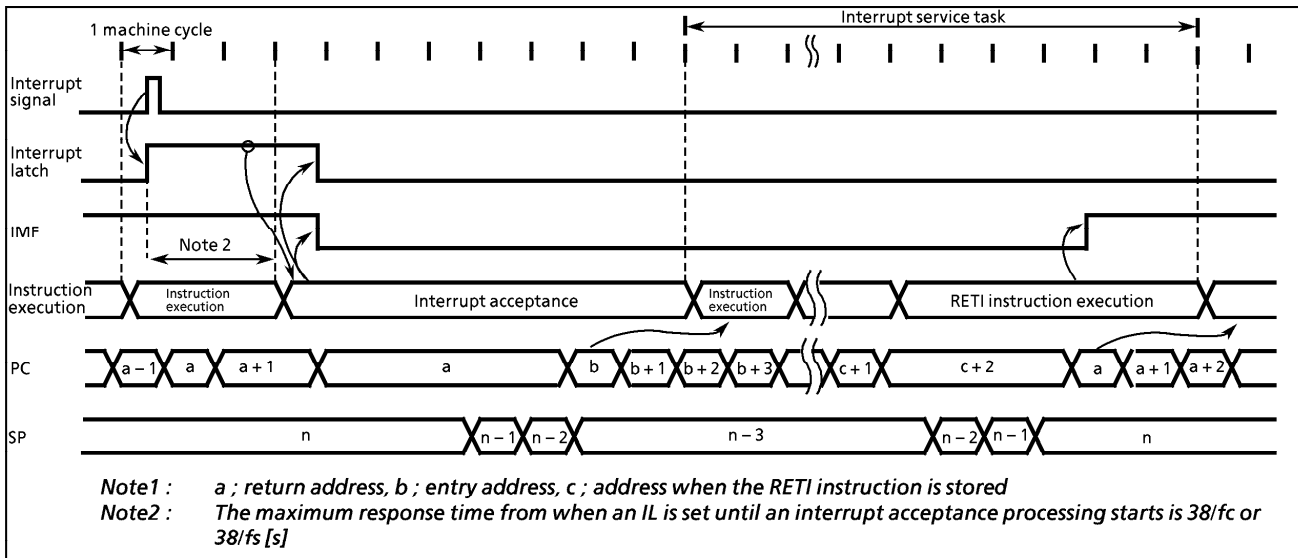
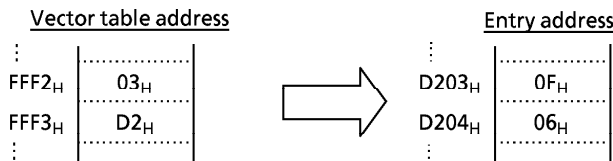


Figure 1-24. Timing Chart of Interrupt Acceptance and Interrupt Return Instruction

Example : Correspondence between vector addresses for INTTBT response processing 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 EF. 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 INT0 pin must be disabled with INT0EN in the external interrupt control register or interrupt processing must be avoided by the program.

Example 1 : Disables an external interrupt 0 using INT0EN

```
LD (EINTCR), 0000000B ; INT0EN←0
```

Example 2 : Disables the processing of an external interrupt 0 under the software control (using bit 0 of address 00F0_H as the interrupt processing disable switch)

```

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

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 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 switching 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 ; Switch to bank n (1μs at 8MHz)
        Interrupt processing
        RETI ; Bank restore and Return
    
```

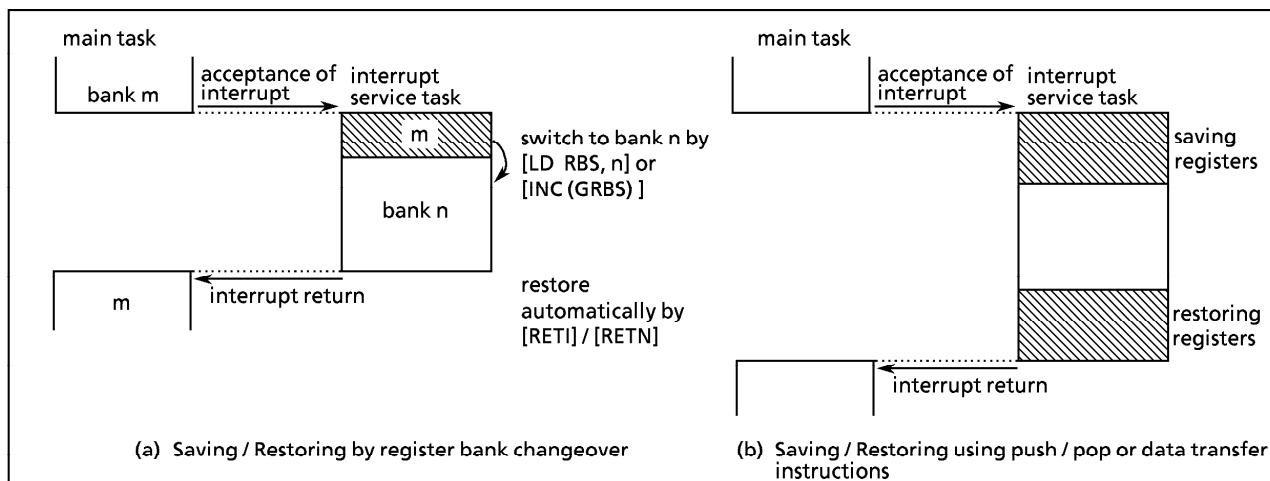


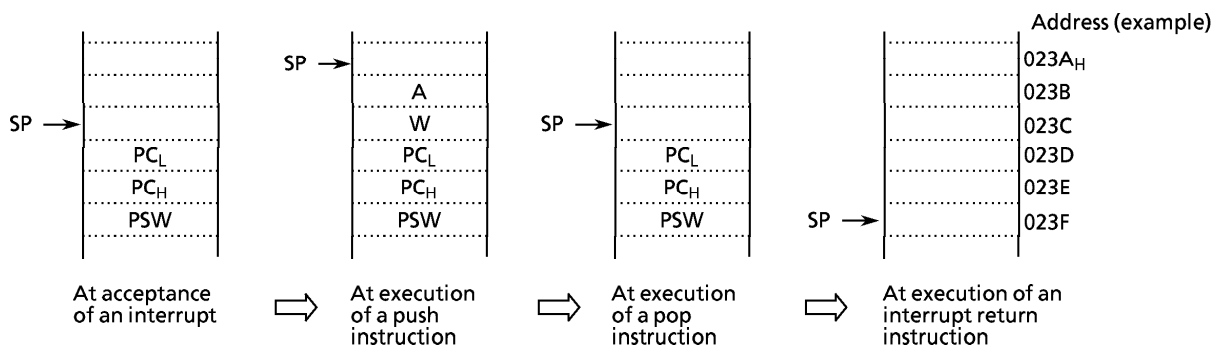
Figure 1-25. Saving / Restoring General-purpose Registers

- ② General-purpose register save / restore using push / 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
        Interrupt processing
        POP WA ; Restore WA register pair
        RETI ; Return
    
```

③ General-purpose registers save / restore using data transfer instructions:
 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
    
```

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 the 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 : If the interrupt processing time is longer than the time required to generate the interrupt request, only the interrupt service program may be executed but not the main task.

1.9.2 External Interrupts

The 87CC20/H20/K20A/M20A each have four external interrupt inputs ($\overline{\text{INT0}}$, INT1, INT2, and $\overline{\text{INT5}}$). Two of these are equipped with digital noise rejection circuits (pulse inputs of less than a certain time are eliminated as noise).

Edge selection is also possible with INT1 and INT2. The $\overline{\text{INT0/P10}}$ pin can be configured as either an external interrupt input pin or an input/output port, and is configured as an input port during reset.

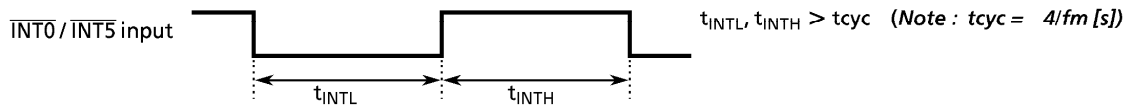
Edge selection, noise rejection control and $\overline{\text{INT0/P10}}$ pin function selection are performed by the external interrupt control register (EINTCR). When $\text{INT0EN} = 0$, the IL_3 will not be set even if the falling edge of $\overline{\text{INT0}}$ pin input is detected.

Table 1-3. External Interrupts

Source	Pin	Secondary function pin	Enable conditions	Edge	Digital noise reject
INT0	$\overline{\text{INT0}}$	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 of equal to or more than 48/fc or 192/fc [s] are regarded as be signals.
INT2	INT2	P12	IMF · EF ₇ = 1		Pulses of less than 7/fc [s] are eliminated as noise. Pulses of equal to or more than 24/fc [s] are regarded as be signals.
INT5	$\overline{\text{INT5}}$	P20 / STOP	IMF · EF ₁₅ = 1	falling edge	— (hysteresis input)

Note 1 : The noise rejection function is turned off in 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 pulse width (both "H" and "L" level) for input to the INT0 and INT5 pins must be over 1 machine cycle.



Note 3 : If a noiseless signal is input to the external interrupt pin in the NORMAL1/2 or IDLE1/2 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 pin 25/fc [s]

Note 4 : 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 $\overline{\text{INT5}}$ (P20/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),1110011101010111B ; IL12,11,7,5,3←0 (clears interrupt latches)
EI ; IMF←1 (enables interrupt service)
```

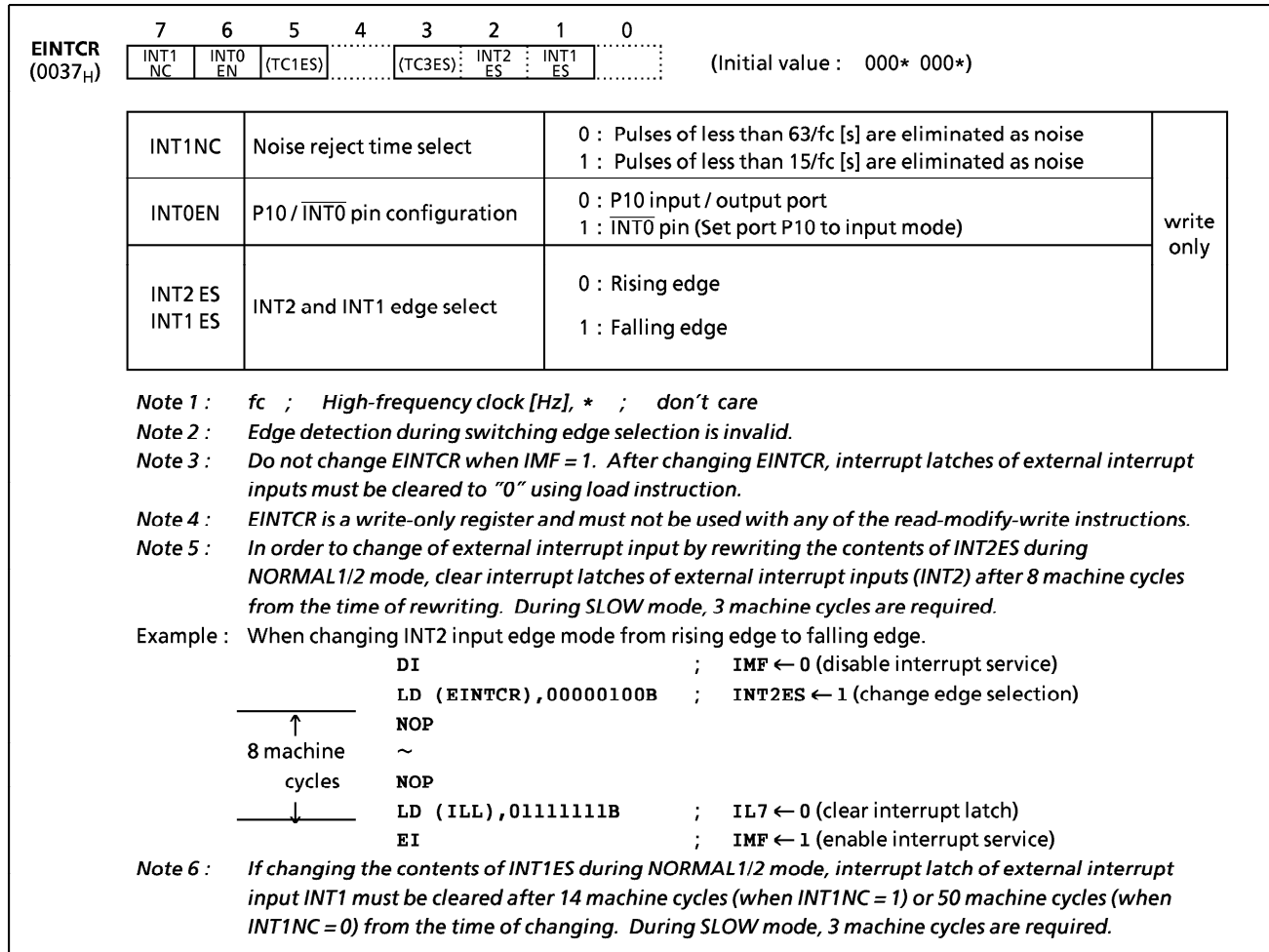


Figure 1-26. External Interrupt Control Register

1.9.3 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.

Use the [SWI] instruction only for detection of the address errors 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. Address-trap reset is generated for instruction fetch from a part of RAM area (addresses 0040_H-023F_H) or SFR area (0000_H-003F_H).

Note : The fetch data from addresses 7F80_H to 7FFF_H (test ROM area) for 87CK20A/M20A, BF80_H to BFFF_H for 87CC20/H20 is not "FF_H".

② Debugging

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

1.10 Watchdog Timer (WDT)

The watchdog timer rapidly detects the CPU malfunction such as endless looping caused by noise or the like, and returns the CPU to the normal state.

The watchdog timer signal for detecting malfunction can be selected either as 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

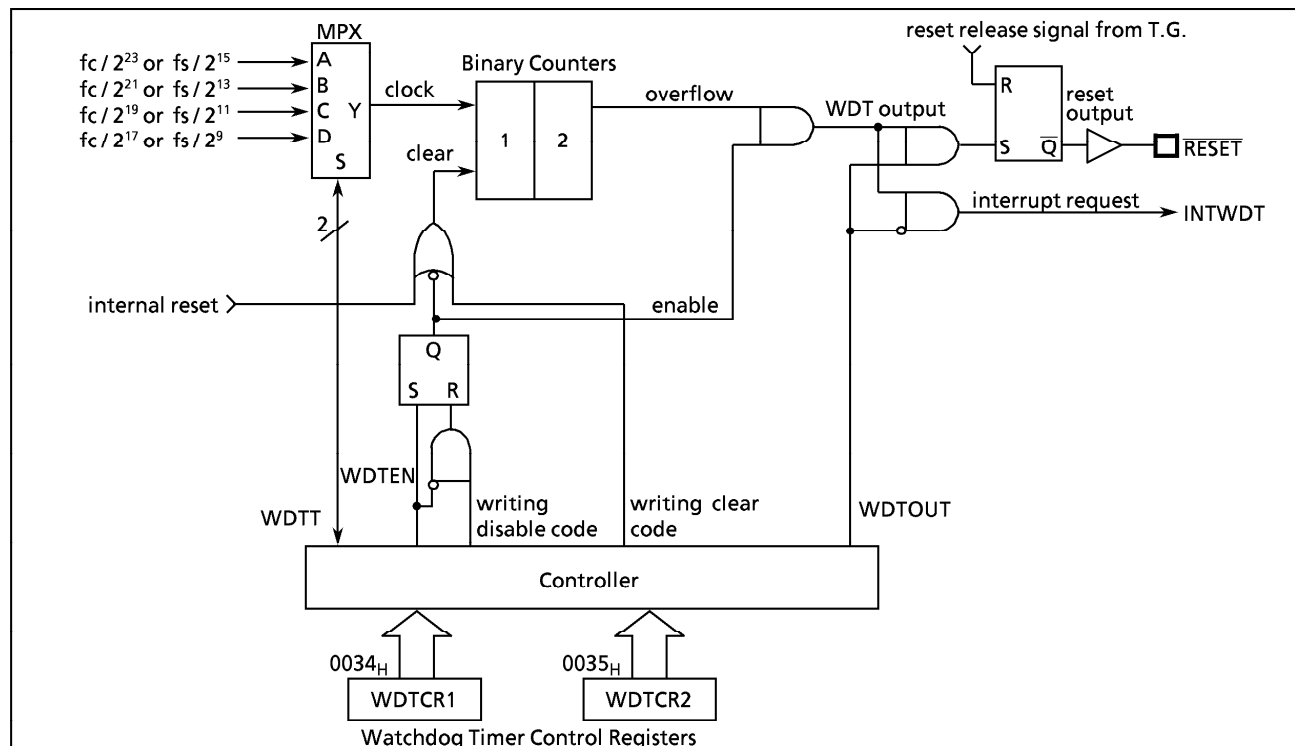


Figure 1-27. Watchdog Timer Configuration

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 a CPU malfunction occurs for any cause, the watchdog timer output will become active on the rise of an overflow from the binary counters unless the binary counters are cleared. At this time, if $WDTOUT = "1"$ a reset is generated, which drives the \overline{RESET} pin low to reset the internal hardware and the external circuit. If $WDTOUT = "0"$, a watchdog timer interrupt (INTWDT) is generated.

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

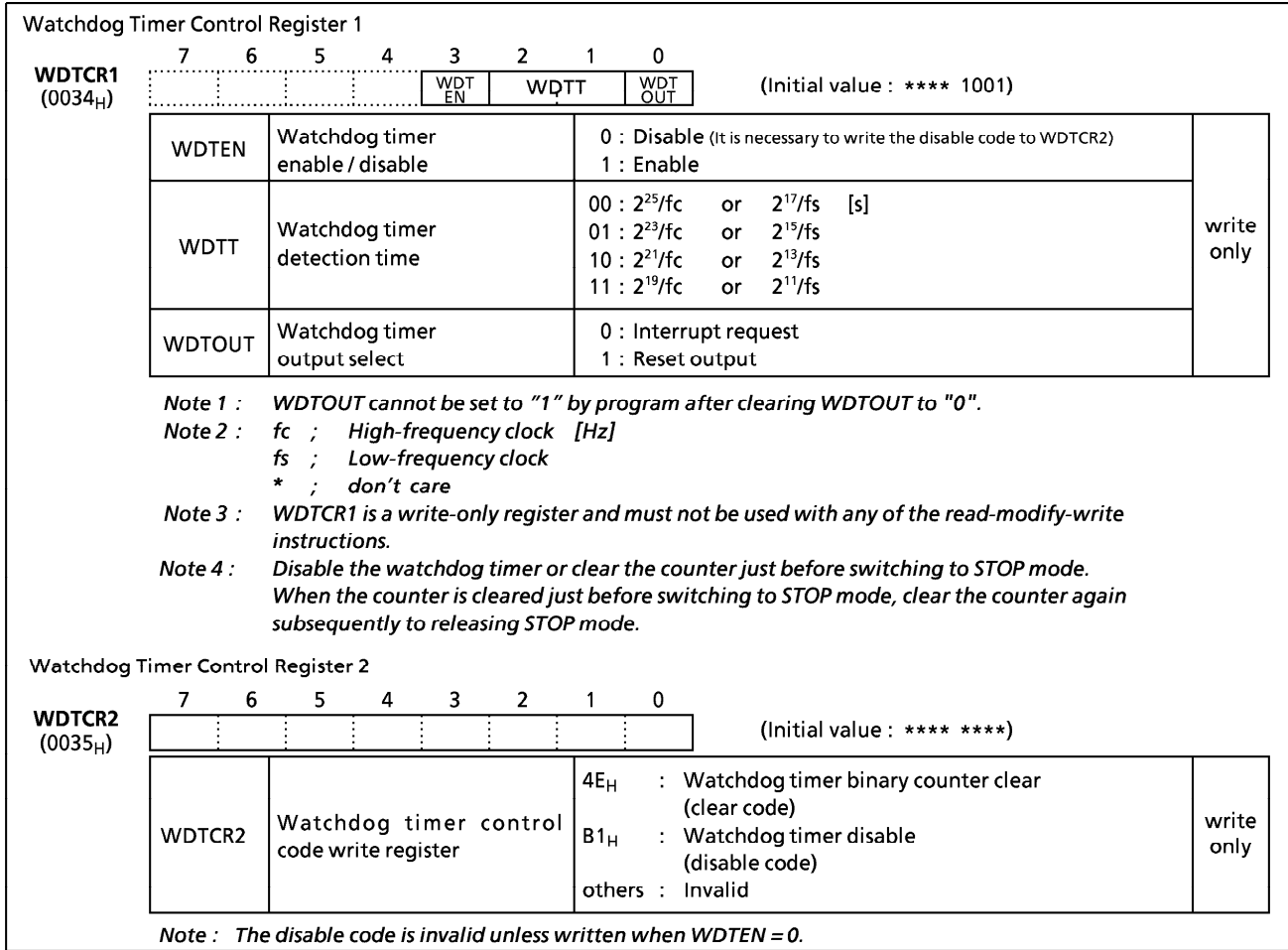


Figure 1-28. Watchdog Timer Control Registers

Table 1-4. Watchdog Timer Detection Time

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

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

```

LD (WDTCR2), 4EH ; Clears the binary counters
LD (WDTCR1), 00001101B ; WDTT←10, WDTOUT←1B
LD (WDTCR2), 4EH ; Clears the binary counters
                    (always clear immediately after changing WDTT)
LD (WDTCR2), 4EH ; Clears the binary counters
LD (WDTCR2), 4EH ; Clears the binary counters
    
```

Within WDT detection time {
 Within WDT detection time {

(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 software interrupt is already accepted, however, the new watchdog timer interrupt waits until the previous 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 : LD SP, 023FH ; 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 $\overline{\text{RESET}}$ pin (sink open drain output) low to reset the internal hardware and the external circuit. The reset output time is $2^{20}/f_c$ [sec] (131 ms at $f_c = 8\text{MHz}$). The high-frequency clock oscillator also turns on when a watchdog timer reset is generated in SLOW mode.

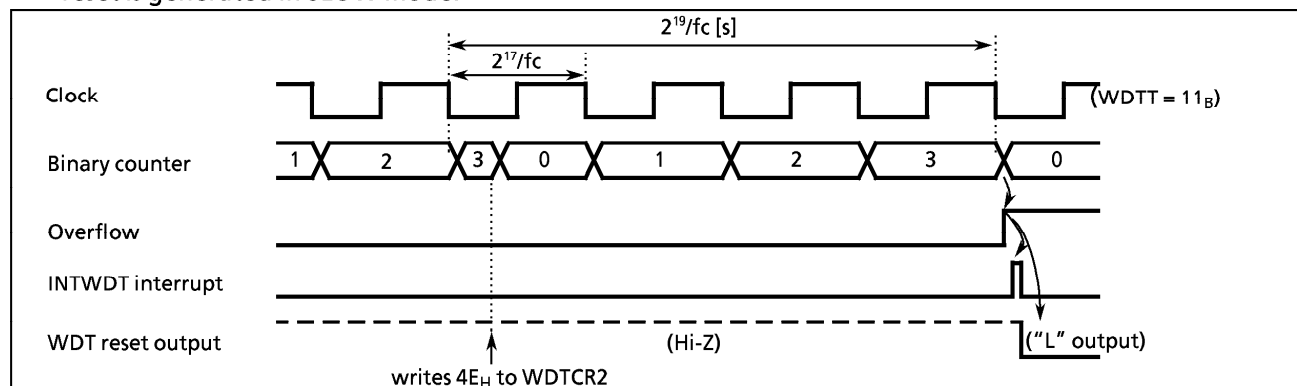


Figure 1-29. Watchdog Timer Interrupt / Reset

1.11 Reset Circuit

The 87CC20/H20/K20A/M20A each have four types of reset generation procedures: an external reset input, an address-trap- reset, a watchdog timer reset and a system-clock-reset. 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 $\overline{\text{RESET}}$ pin may go low ($2^{20}/f_c$ [s] (131ms at 8MHz), when power is turn 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) · (FFFE _H)	Divider of timing generator	0
Register bank selector (RBS)	0	Watchdog timer	Enable
Jump status flag (JF)	1	Output latches of I/O ports	Refer to I/O port circuitry
Interrupt master enable flag (IMF)	0	Control registers	Refer to each of control register
Interrupt individual enable flags (EF)	0		
Interrupt latches (IL)	0		

1.11.1 External Reset Input

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

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

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

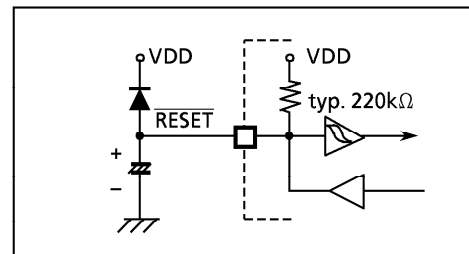


Figure 1-30. Simple Power-on-Reset Circuit

1.11.2 Address-Trap-Reset

An address-trap-reset is one of fail-safe function that detects CPU malfunction such as endless looping caused by noise or the like, and returns the CPU to the normal state. If the CPU attempts to fetch an instruction from addresses 0000_H to 023F_H (a part of RAM or SFRs), an internal reset (called address-trap-reset) will be generated. Then, the $\overline{\text{RESET}}$ pin output will go low. The reset time is $2^{20}/f_c$ [s] (131ms at 8MHz).

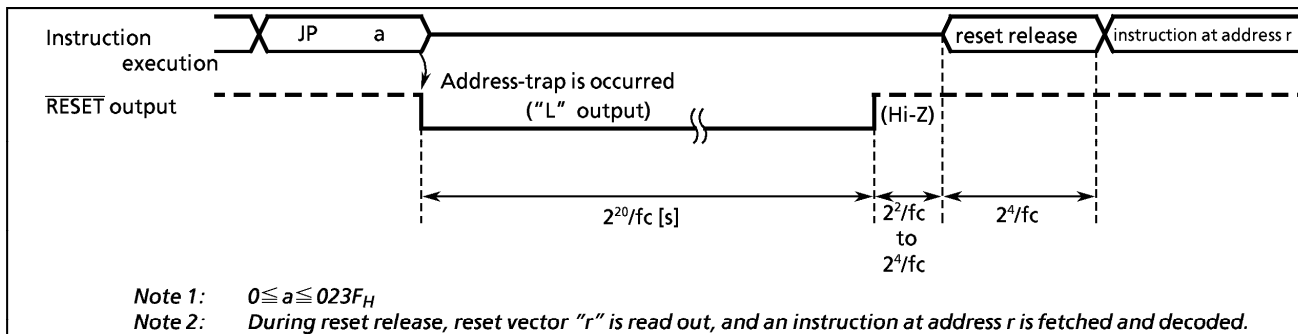


Figure 1-31. Address-Trap-Reset

1.11.3 Watchdog Timer Reset

Refer to "1.10 Watchdog Timer".

1.11.4 System-Clock-Reset

Clearing both XEN and XTEN (bits 7 and 6 in SYSCR2) to "0" stops 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 $\overline{\text{RESET}}$ pin output goes low from high-impedance. The reset time is $2^{20}/f_c$ [s] (131ms at 8MHz).

2. PERIPHERAL HARDWARE FUNCTIONS

2.1 Special Function Registers (SFR) and Data Buffer Registers (DBR)

The TLCS-870 Series uses the memory mapped I/O system, and all peripherals control and data transfers are performed through the special function registers (SFR) and data buffer registers (DBR).

The SFR are mapped to addresses 0000_H – 003F_H and the DBR to addresses 0F80_H – 0FFF_H.

Figure 2-1 shows the 87CC20/H20/K20A/M20A SFRs and DBRs.

Address	Read	Write	Address	Read	Write
0000 _H		P0 Port	0020 _H	SIOSR (SIO status register)	SIOCR1 (SIO control)
01		P1 Port	21	—	SIOCR2 (SIO control)
02		P2 Port	22		reserved
03		P3 Port	23		reserved
04		P4 Port	24		reserved
05		P5 Port	25		reserved
06		P6 Port	26		reserved
07		reserved	27		reserved
08		reserved	28	—	LCDCR (LCD control)
09		reserved	29	—	P6CR (P6 I/O control)
0A	—	P0CR (P0 I/O control)	2A		reserved
0B	—	P1CR (P1 I/O control)	2B		reserved
0C		reserved	2C		reserved
0D		reserved	2D		reserved
0E		reserved	2E		reserved
0F		reserved	2F		reserved
10	TREG1 _L	TREG1A (Timer register1A)	30		reserved
11	TREG1 _H		31		reserved
12	—	TREG1B (Timer register1B)	32		reserved
13	—	TC1CR (TC 1 control)	33		reserved
14		reserved	34	—	WDTCR1 (WDT control)
15		reserved	35	—	
16	—	TREG6 (Timer register6)	36	—	TBTCR (TBT control)
17	—	TC6CR (TC 6 control)	37	—	EINTCR (external interrupt control)
18	TREG3A (Timer register3A)		38		SYSCR1 (system control)
19	TREG3B (Timer register3B)		39		
1A	—	TC3CR (TC 3 control)	3A		EIR _L (interrupt enable register)
1B	—	TREG4 (Timer register4)	3B		
1C	—	TC4CR (TC 4 control)	3C		IL _L (interrupt latch)
1D	—	TREG5 (Timer register5)	3D		
1E	—	TC5CR (TC 5 control)	3E		reserved
1F		reserved	3F	PSW (program status word)	RBS (register bank selector)

(a) Special Function Registers

Address	Read	Write
0F80 _H		LCD display data buffer
0F8F		
0F90		reserved
0FEF		
0FF0		SIO transmit and receive data buffer
0FF7		
0FF8		reserved
0FFF		

(b) Data Buffer Registers

- Note 1 : Do not access reserved areas by the program.
- Note 2 : When defining address 003F_H with assembler symbols, use GPSW and GRBS.
- Note 3 : — ; do not access.
- Note 4 : Operations specified to writing registers and interrupt latches by read modifying write instructions (bit operation instructions such as SET, CLR, etc. , or operation instructions such as AND, OR, etc.) are not effective.

Figure 2-1. SFR & DBR

2.2 I/O Ports

The 87CC20/H20/K20A/M20A each have seven parallel input / output ports (45pins) each as follows:

	Primary Function	Secondary Functions
Port P0	8-bit I/O port	
Port P1	8-bit I/O port	External interrupt input, and divider output.
Port P2	3-bit I/O port	Low-frequency resonator connection, external interrupt input, STOP mode release signal input.
Port P3	8-bit I/O port	Timer / counter output.
Port P4	8-bit I/O port	Serial interface, timer / counter input and output.
Port P5	2-bit I/O port	Timer / counter input.
Port P6	8-bit I/O port	Segment output.

All output ports have latches, so output data are held by latching. All input ports have no latch, 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 read cycle during execution of the read instruction. This timing can not be recognized from outside, so that the transient input such as chattering must be processed by the program.

Data output is changed in the S2 state of write cycle during execution of the instruction which writes to an I/O port.

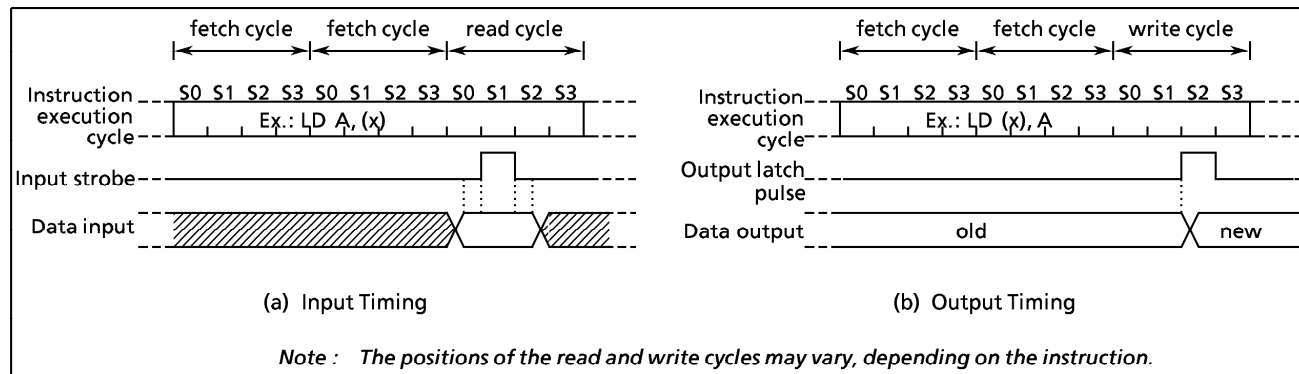


Figure 2-2. Input / Output Timing (example)

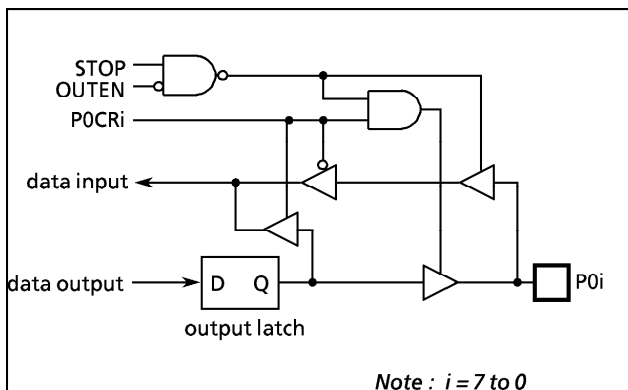
When reading an I/O port except programmable I/O ports, whether the pin input data or the output latch contents are read depends on the instructions, as shown below:

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

- (2) Instructions that read the pin input data
 - 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 written into the output latch regardless of the P0CR contents. Therefore initial output data should be written into the output latch before setting P0CR.



P0 (0000 _H)	7	6	5	4	3	2	1	0	(Initial value: 0000 0000)
	P07	P06	P05	P04	P03	P02	P01	P00	
P0CR (000A _H)	7	6	5	4	3	2	1	0	(Initial value: 0000 0000)
P0CR	I/O control for port P0							0 : input mode 1 : output mode	Write only

Figure 2-3. Port P0 and P0CR

Example : Setting the upper 4 bits of port P0 as input ports and the lower 4 bits as output ports (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, P1CR is initialized to "0", which configures port P1 as an input. The P1 output latches are also initialized to "0".

Port P1 is also used as an external interrupt input, and a divider output. When used as these secondary function pins, the input pins should be set to the input mode, and the output pin 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, or input ports. The interrupt latch is set on the rising or falling edge of the output when used as an output port. Pin P10 ($\overline{INT0}$) can be configured as either an I/O port or an external interrupt input with the INT0EN (bit 6 in EINTCR). During reset, pin P10 ($\overline{INT0}$) is configured as an input port P10.

Example : Sets P17, P16 as output ports, P15 and P14 as input ports, and the others as function pins. Internal output data is "1" for the P17, and "0" for the P16 pin.

```
LD (EINTCR), 01000000B ; INT0EN←1
LD (P1), 10111111B ; P17←1, P16←0, P13←1
LD (P1CR), 11001000B
```

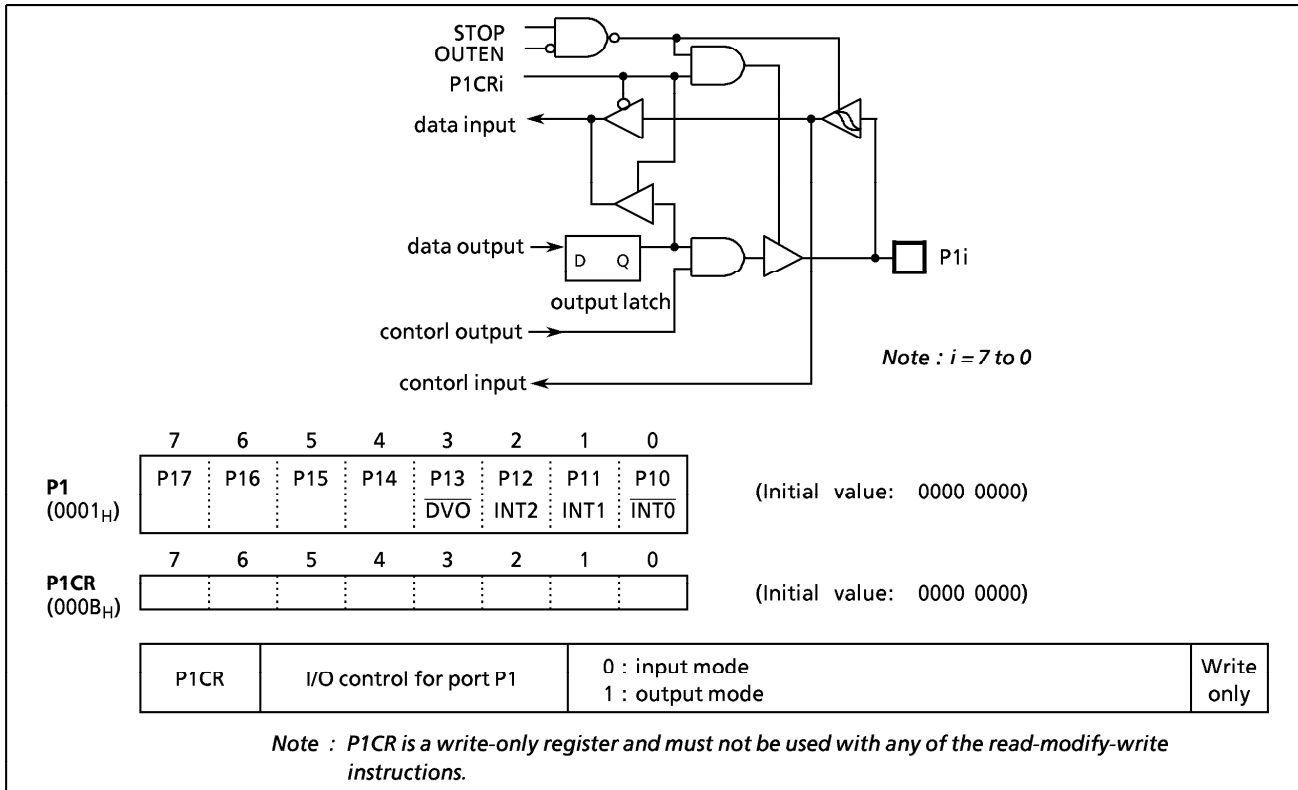


Figure 2-4. Port P1 and P1CR

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, STOP mode release signal input, and as low-frequency crystal connection pins. When used as an input port, or as 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 P20 output pulse. When a read instruction is executed for port P2, bits 7 to 3 in P2 are read in as undefined data.

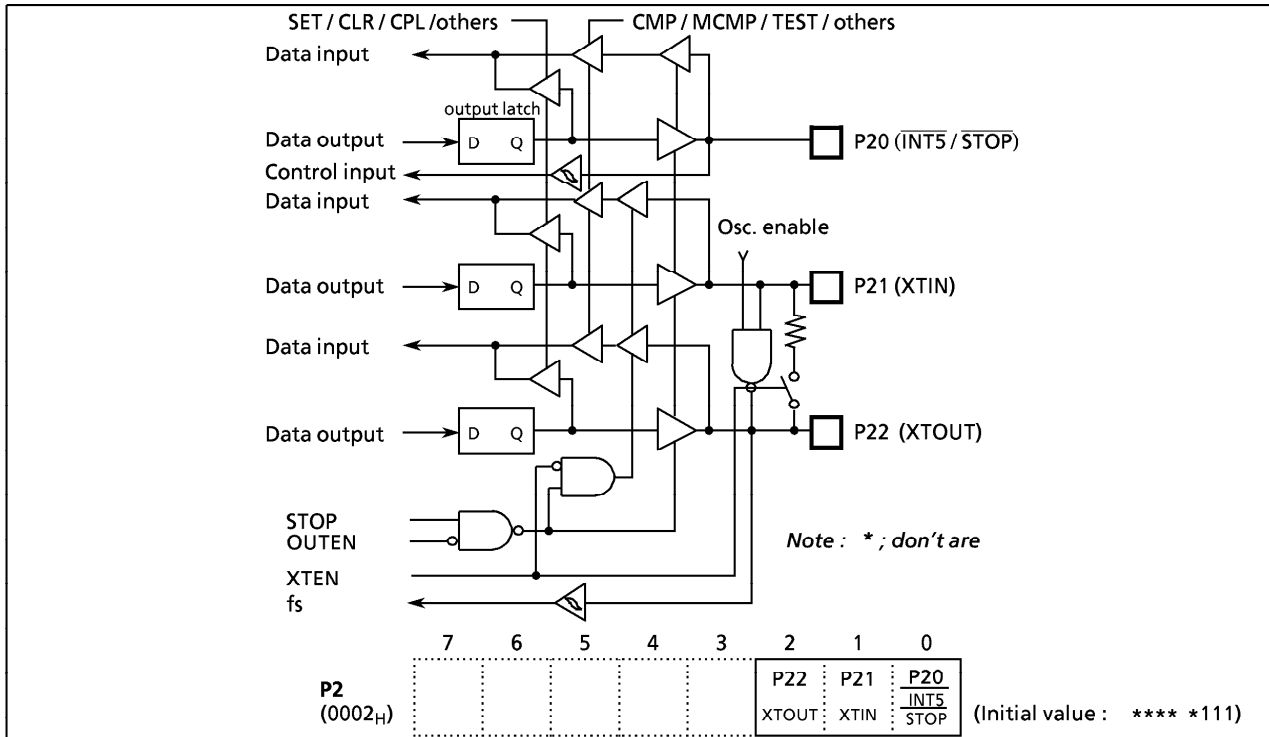


Figure 2-5. Port P2

2.2.4 Port P3 (P37 – P30)

Port P3 is an 8-bit input / output port. It is also used as a timer/counter output. Pins P31 and P30 can be also used as an output of PWM0/PDO0 and PWM1/PDO1 respectively, and only these pins are available high current output, so LEDs can be driven directly.

When used as an input or a secondary function pin, the output latch should be set to "1". The output latches are initialized to "1" during reset.

Example1 : Outputs an immediate data 5A_H to port P3.

```
LD (P3), 5AH ; P3 ← 5AH
```

Example2 : Inverts the output of the upper 4bits (P37 - P34) in port P3.

```
XOR (P3), 1111000B ; P37~P34 ←  $\overline{P37 \sim P34}$ 
```

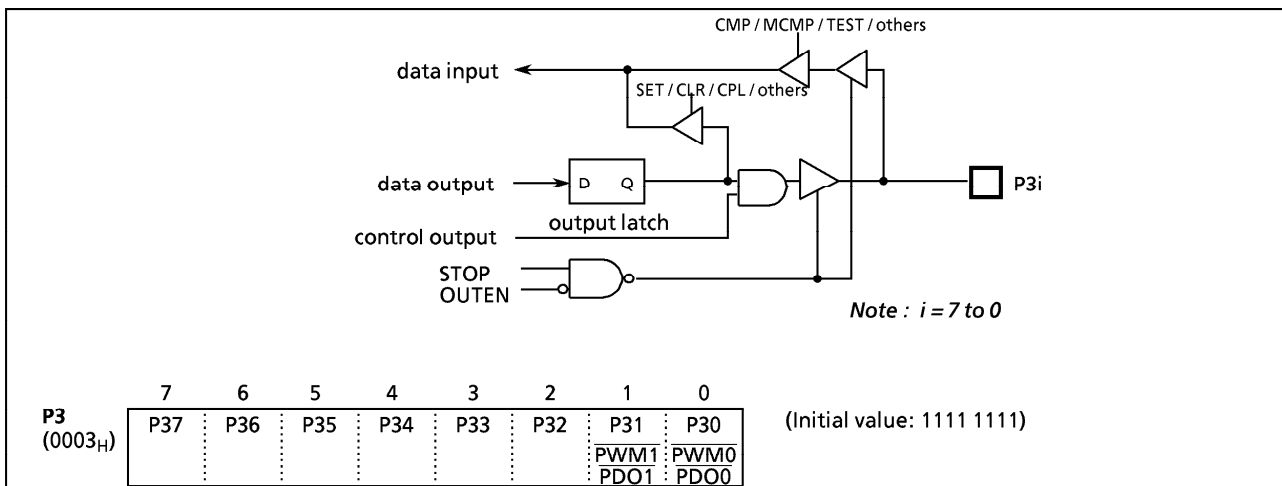


Figure 2-6. Port P3

2.2.5 Port P4 (P47 – P40)

Port P4 is an 8-bit input / output port. It is also used as serial interface (SIO) input / output, and a timer/counter input / output. When used as an input port or a secondary function pin, the output latch should be set to "1". The output latches are initialized to "1" during reset.

2.2.6 Port P5 (P51 – P50)

Port P5 is a 2-bit input/output port. It is also used as a timer/counter input. When used as an input port or a secondary function pin, the output latch should be set to "1". The output latches are initialized to "1" during reset. When a read instruction for port P5 is executed, bits 7 to 2 in P5 are read in as "1".

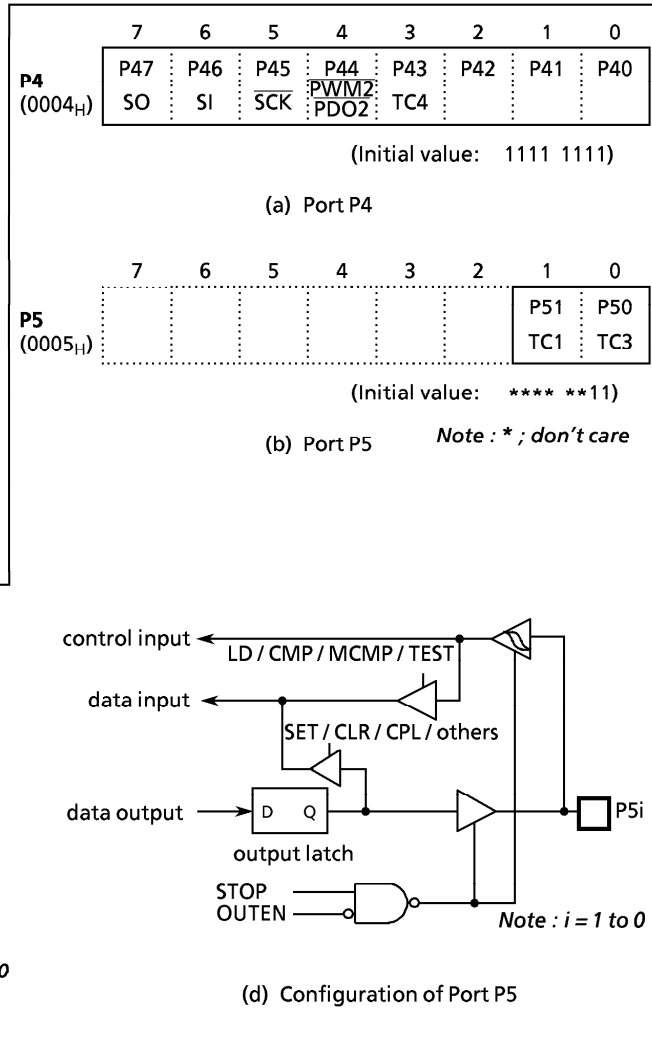


Figure 2-7. Ports P4 and P5

2.2.7 Ports P6 (P67 – P60)

Port P6 is an 8-bit input / output port and is also used as the segment output port. Input / output mode or segment output mode is specified by the corresponding bit in the P6 port control register (P6CR). During reset, P6CR is initialized to "0", which configure port P6 as input / output. Port P6 output latches are also initialized to "1". P6CR can only be written.

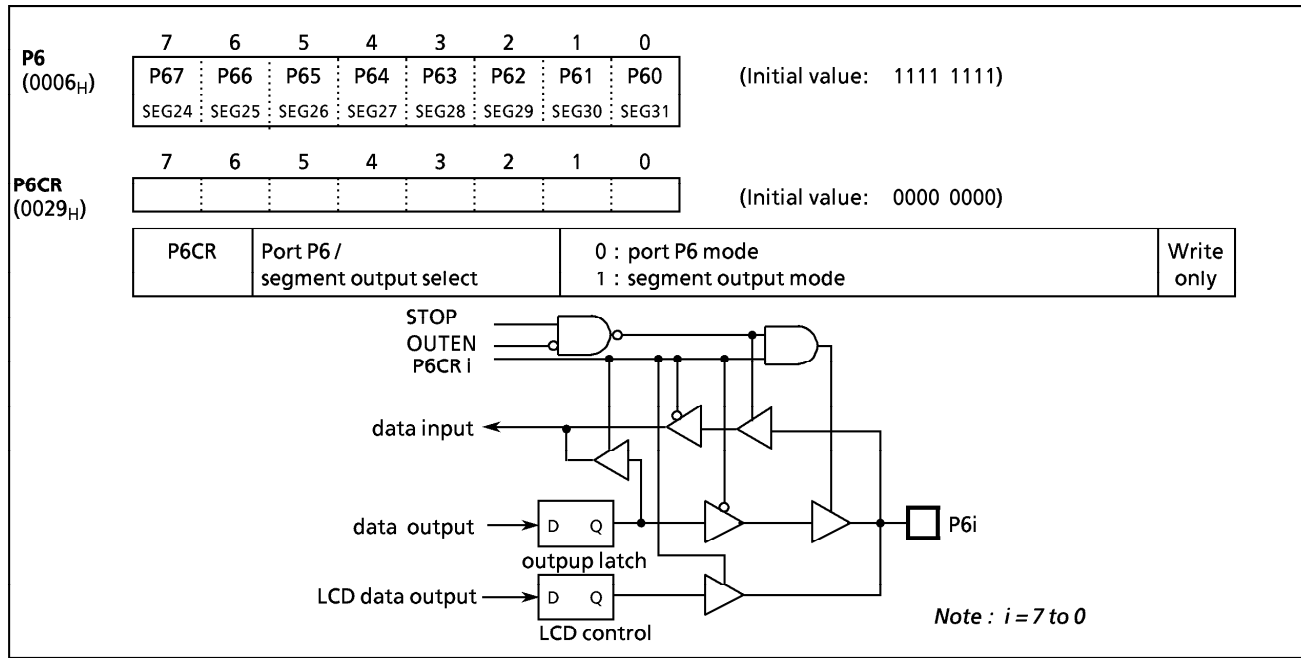


Figure 2-8. Port P6 and P6CR

Example : Setting the upper 2 bits of port P6 as a segment output port, and the others as input/output port.

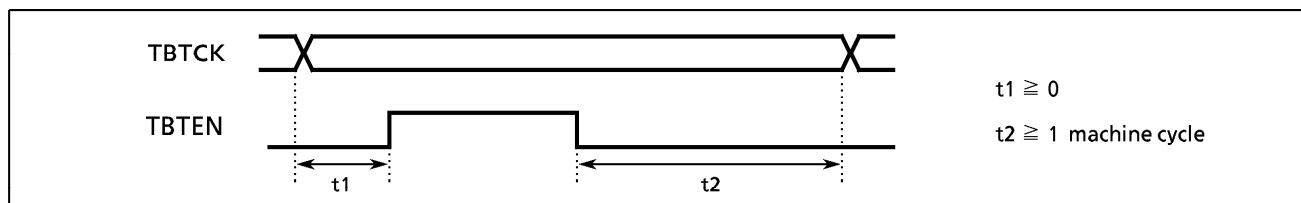
```
LD (P6CR), 11000000B
```

2.3 Time Base Timer (TBT)

The time base timer generates time base for key scanning or dynamic displaying, etc. And it provides a time base timer interrupt (INTTBT). The time base timer is controlled by the control register (TBTCR) shown in Figure 2-10.

An INTTBT is generated at 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
```

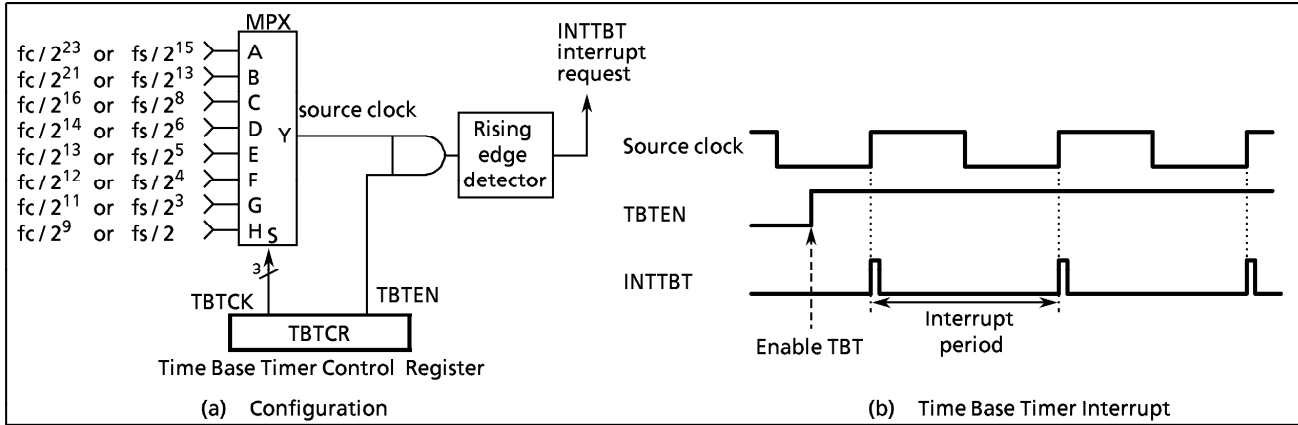


Figure 2-9. Time Base Timer

TBTCKR (0036H)		7	6	5	4	3	2	1	0	(Initial value : 0**0 0***)	
		(DVOEN)	(DVQCK)	(DV7CK)	TBTEN	TBTCK					
TBTEN	Time base timer enable/disable	0 : Disable 1 : Enable									
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 $fs/2$								write only	

Note1 : fc ; High-frequency clock [Hz], fs ; Low-frequency clock [Hz], * ; don't care
 Note2 : TBTCKR is a write-only register and must not be used with any of read-modify-write instructions.

Figure 2-10. Time Base Timer 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 $fc = 8\text{MHz}$	At $fs = 32.768\text{kHz}$
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 2-11.

Note that the TBTCR is a write-only register, so a read-modify-write instruction cannot be used.

		7	6	5	4	3	2	1	0		
TBTCR (0036 _H)		DVOEN	DVOCK	(DV7CK)	(TBTEN)	(TBTCR) ₁				(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 $fs / 2^2$									
<i>Note : fc ; High-frequency clock [Hz], fs ; Low-frequency clock [Hz], * ; don't care</i>											

Figure 2-11. Divider Output Control Register

Example : 1kHz pulse output (at $fc = 8\text{MHz}$)

```

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 $fc = 4.194304\text{MHz}$	At $fc = 8\text{MHz}$	At $fs = 32\text{kHz}$
00	$fc / 2^{13}$ or $fs / 2^5$	0.512 [kHz]	0.976 [kHz]	1.024 [kHz]
01	$fc / 2^{12}$ or $fs / 2^4$	1.024	1.953	2.048
10	$fc / 2^{11}$ or $fs / 2^3$	2.048	3.906	4.096
11	$fc / 2^{10}$ or $fs / 2^2$	4.096	7.812	8.192

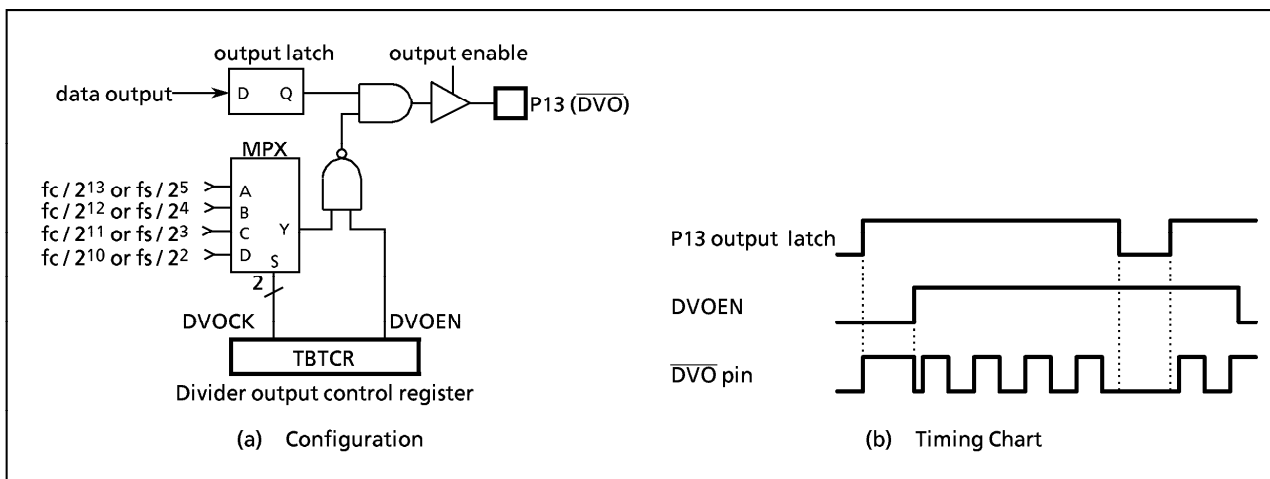


Figure 2-12. Divider Output

2.5 16-bit Timer / Counter 1 (TC1)

2.5.1 Configuration

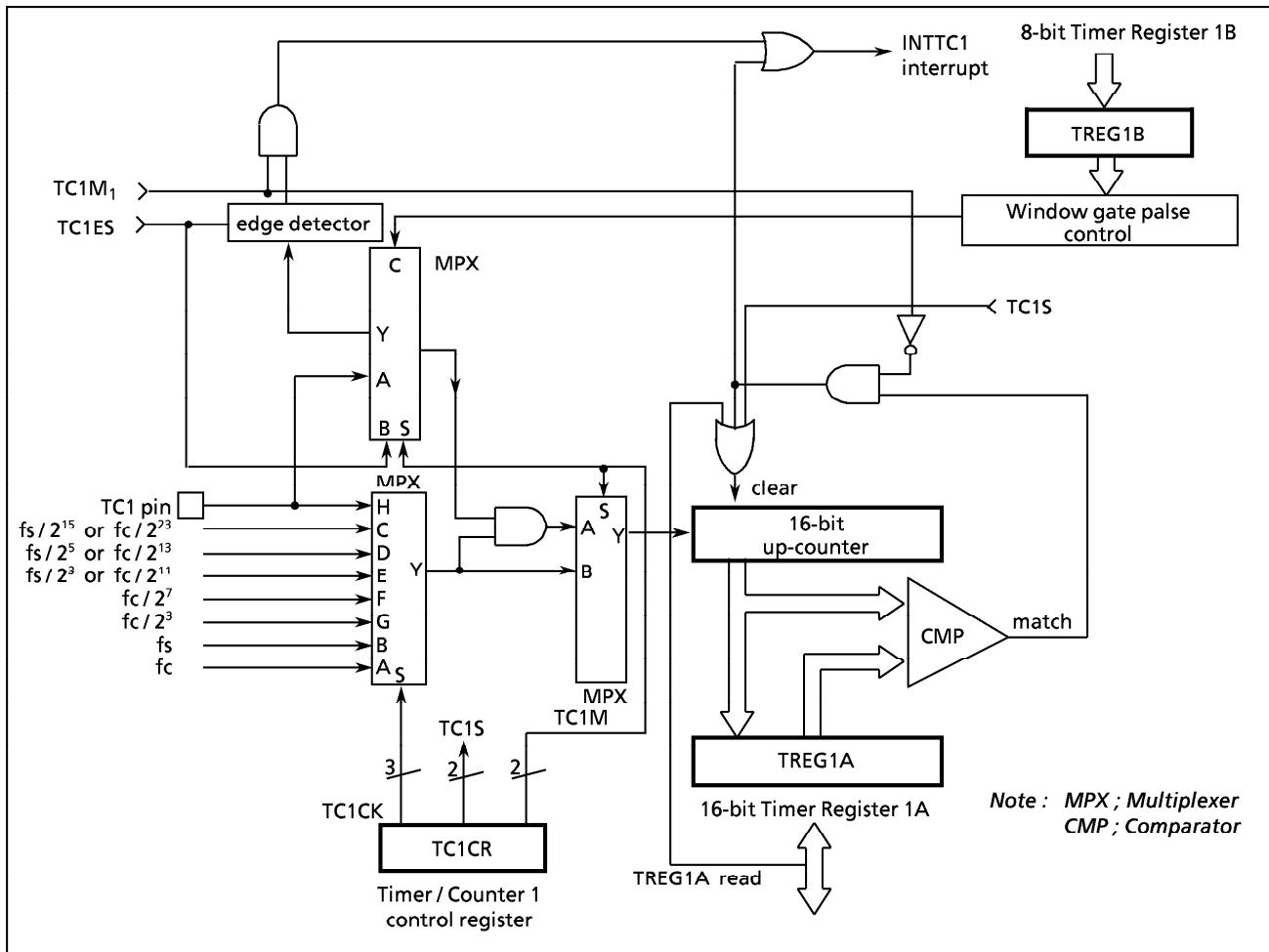


Figure 2-13. Timer / Counter 1

2.5.2 Control

Timer / counter 1 is controlled by a timer / counter 1 control register (TC1CR), an external interrupt control register (EINTCR), a 16-bit timer register (TREG1A) and an 8-bit window gate pulse register (TREG1B). Reset does not affect TREG1A and TREG1B.

2.5.3 Function

The TC1 has 3 operating modes. Also timer / counter 1 is used for warm-up when switching from SLOW mode to NORMAL2 mode.

(1) Timer mode

In this mode, counting up is performed using the internal source clock shown in Table 2-3. The contents of the timer register 1A (TREG1A) are compared with the contents of the up-counter. If a match is found, a INTTC1 interrupt is generated, and the up-counter is cleared to "0". Counting up is resumed after the up-counter is cleared.

When "fc" is selected as the source clock during SLOW mode, the lower 11bits of TREG1A are ignored to compare and an INTTC1 interrupt is generated by matching the upper 5 bits. Thus, in this case, only the TREG1A_H setting is necessary.

Table 2-3. Source Clock (internal clock) for TC1

Source Clock				Resolution		Maximum time setting	
NORMAL1 /2, IDLE1 /2 mode		SLOW mode	SLEEP mode	fc = 8MHz	fs = 32.768 kHz	fc = 8MHz	fs = 32.768 kHz
DV7CK = 0	DV7CK = 1						
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	1.07 min
fc / 2 ¹¹	fs / 2 ³	fs / 2 ³	fs / 2 ³	256 μs	244 μs	16.8 s	16 s
fc / 2 ⁷	—	—	—	16 μs	—	1 s	—
fc / 2 ³	—	—	—	1 μs	—	65.5 ms	—
fc (Note1)	—	fc (Note2)	—	125 ns	—	7.9 ms	—
fs	—	—	—	—	30.5 μs	—	2 s

Note1 : "fc" can be selected as the source clock only in the pulse width measurement mode.
 Note2 : "fc" can be selected as the source clock only in the timer mode when switching the main system clock.

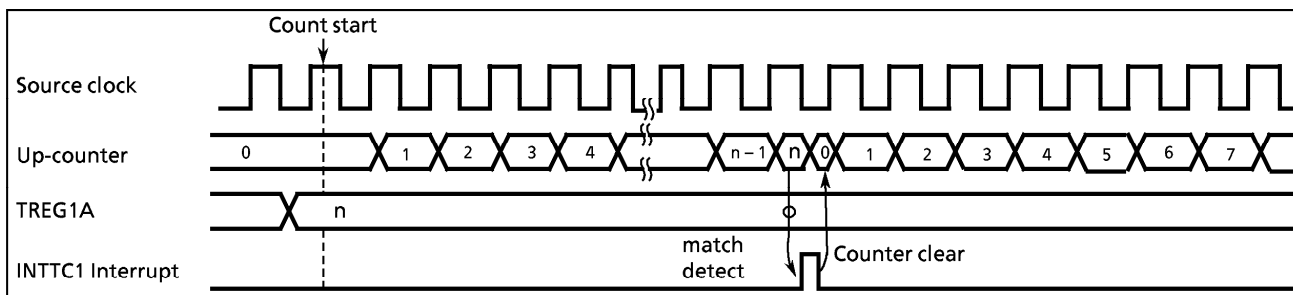


Figure 2-16. Timer Mode Timing chart

Example : Sets the timer mode with source clock fs/2³ [Hz] and generates an interrupt 1 s. later (at fs = 32.768kHz).

```
LD (TC1CR), 00010000B ; Sets TC1 mode and source clock
LDW (TREG1A), 1000H ; Sets the timer register (1s ÷ 23/fs = 1000H)
SET (EIRH).EF8 ; Enables INTTC1
EI
LD (TC1CR), 01010000B ; Starts TC1
```

(2) Event Counter mode

In this mode, events are counted on the edge of TC1 pin input. Either the rising or falling edge can be selected with TC1ES in EINTCR. The contents of TREG1A are compared with the contents of the up-counter. If a match is found, an INTTC1 interrupt is generated, and the up-counter is cleared to "0". The maximum applied frequency is $f_c/2^4$ [Hz] in NORMAL1/2 or IDLE1/2 mode, and $f_s/2^4$ [Hz] in SLOW or SLEEP mode. The contents of the up-counter cannot read.

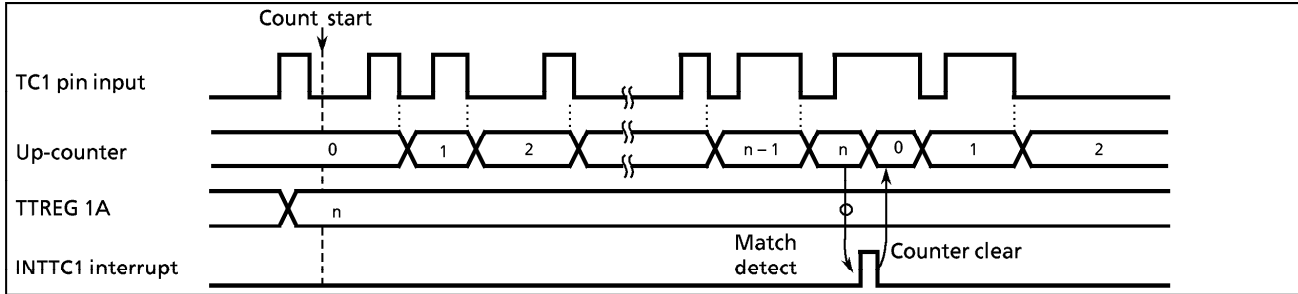


Figure 2-17. Event Counter mode Timing chart (TC1ES = 1)

(3) Pulse Width Measurement mode

Counting up is performed on the rising edge of the pulse that is the logical AND-ed product of the TC1 pin input (window pulse) and an internal clock shown in Table2-3. On the falling edge of the TC1 pin input contents of the up-counter are captured in TREG1A, and counting is stopped until the next rising edge of the TC1 pin input. An INTTC1 interrupt is generated at the edge (either rising edge or falling edge can be selected with TC1ES in EINTCR) of window pulse. Internal clock is selected with TC1CK. The counter will be cleared after TREG1A_L and TREG1A_H are read out. If the TREG1A is not read out, the counter will not be cleared.

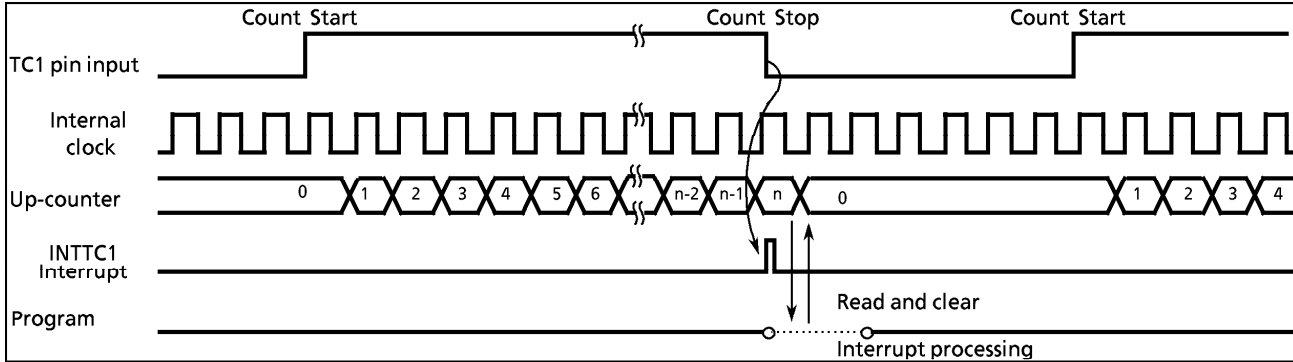


Figure 2-18. Pulse width measurement mode Timing chart (TC1ES = 1)

Example : To measure the width of the "H" level of pulse input from TC1 pin. The source clock is set to $f_c / 2^3$ ($f_c = 8$ MHz).

```

LD      (EINTCR), 00100000B ; Sets to window pulse edge selection mode. (TC1ES←1)
LD      (TC1CR), 00011010B ; Sets to source clock and pulse width measurement mode.
SET     (EIRH).EF8 ; Enables INTTC1
EI
LD      (TC1CR), 01011010B ; Starts the measurement.
:
:
PINTTC1: LD      RBS, 1
LD      WA, (TREG1A) ; Counter read and clear
: ; Pulse width calculation (WA × 23 / fc)
:
RETI
:
VINTTC1: DW      PINTTC1
    
```

(4) Frequency Measurement mode

This is the mode to measure the frequency of TC1 pin input (set TC1CK to "111"). While window gate pulse is at "H" level, the edge of input pulse is counted (leading edge or trailing edge is selected by TC1ES in EINTCR), generated an interrupt request (INTTC1) at the trailing edge of window gate pulse. The frequency of window gate pulse is set by TREG1B. The counter will be cleared when the value of the counter is taken in TREG1A and TREG1A_L and TREG1A_H are read in order. If the value of the counter is not taken in TREG1A, the counter will not be cleared and continue to count up at the next count start instruction.

Table 2-4. Setting Ta and Tb

Setting window gate pulse

Window gate pulse consists of counting time (Ta) and count halt time (Tb), which can be set independently. Accordingly, Ta + Tb make a one period.

Set time of Ta and Tb is expressed as follows.

$$(16 - n) \times 2^{13} / fc[s]$$

$$(16 - n) \times 2^5 / fs[s]$$

Set time when fc is 4.194304 MHz shown in the table at right.

The upper bits (bits 7-4) of TREG1B set Tb, While the lower bits (bits 3-0) set Ta.

Setting valve	Setting time	Setting valve	Setting time
0	31.25 ms	8	15.63 ms
1	29.30 ms	9	13.67 ms
2	27.34 ms	A	11.72 ms
3	25.39 ms	B	9.77 ms
4	23.44 ms	C	7.81 ms
5	21.48 ms	D	5.86 ms
6	19.53 ms	E	3.91 ms
7	17.58 ms	F	1.95 ms

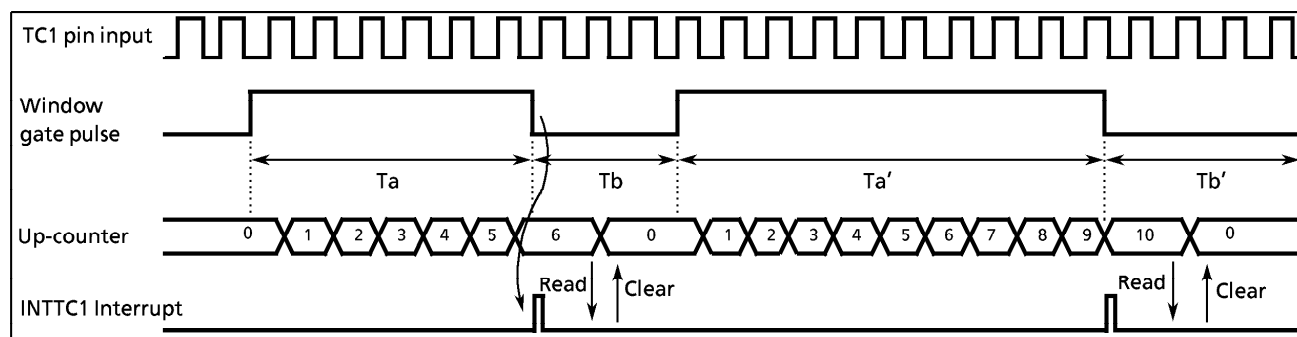


Figure 2-19. Frequency Measurement mode Timing Chart (TC1ES = 0)

Example: To measure the frequency of the pulse input from TC1 pin.

Sets to Ta = 19.53ms and Tb = 9.77ms (at fc = 4.194304MHz).

```

LD      (TC1CR), 00011111B      ; Sets to frequency measurement mode.
                                       (TC1M = 11, TC1CK = 111)
LD      (TREG1B), 0B6H          ; Sets Ta and Tb.
SET     (E1RH). EF8            ; Enables INTTC1
EI
LD      (TC1CR), 01011111B      ; Starts the measurement.
:
PINTTC1: LD      RBS, 1
LD      WA, (TREG1A)           ; Counter read and clear.
:                                   ; Frequency calculation (WA ÷ 19.53ms)
RETI
:
VINTTC1: DW      PINTTC1
    
```

2.6 8-Bit Timer / Counter 3 (TC3)

2.6.1 Configuration

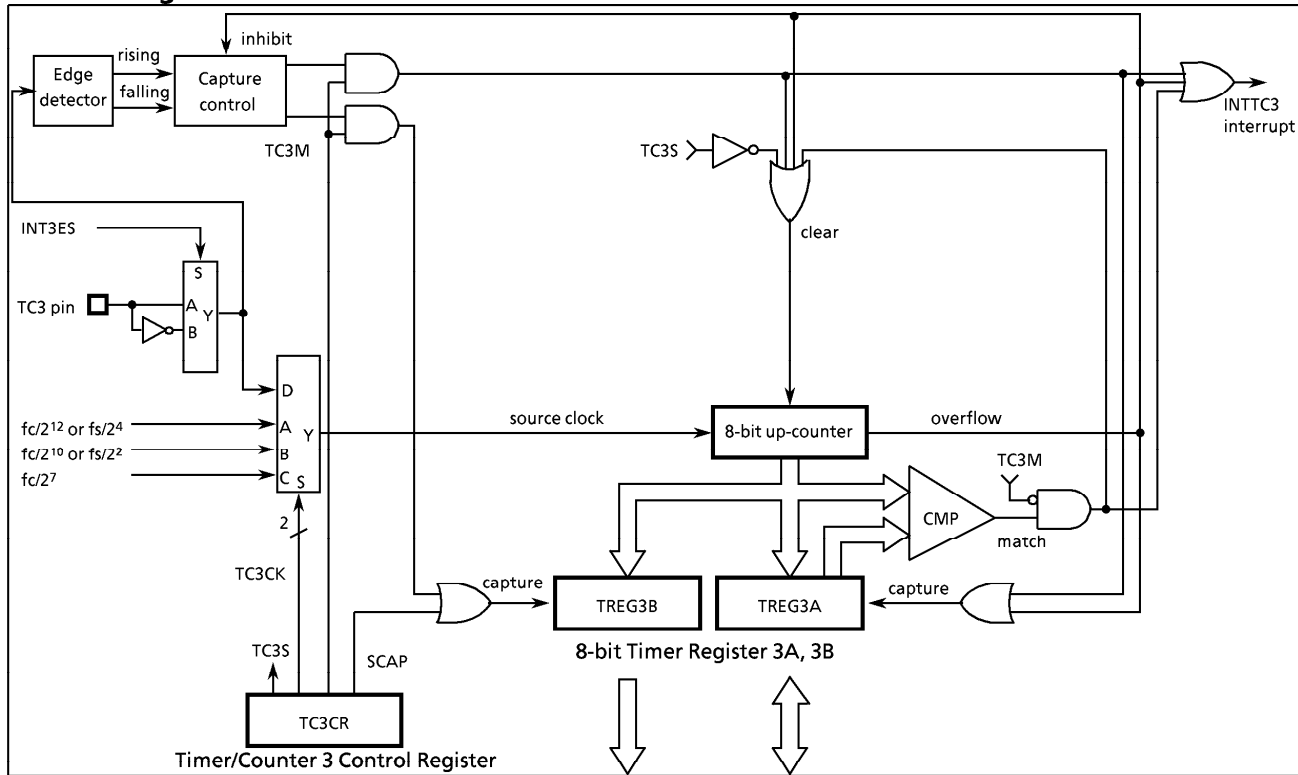


Figure 2-20. Timer/Counter 3

2.6.2 Control

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

TREG3A (0018 _H)	7 6 5 4 3 2 1 0	Read / Write
TREG3B (0019 _H)	7 6 5 4 3 2 1 0	Read only
TC3CR (001A _H)	7 6 5 4 3 2 1 0	(Initial value: *0*0 00*0)
	SCAP TC3S TC3CK TC3M	
TC3M	TC3 operating mode select	0 : Timer / Event counter 1 : Capture
TC3CK	TC3 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	TC3 Start control	0 : Stop & clear 1 : Start
SCAP	Software capture control	0 : - 1 : Software capture

Note1 : fc ; high-frequency clock [Hz] fs ; low-frequency clock [Hz] * ; don't care
Note2 : Set the operating mode, the source clock selection, and edge selection (TC3ES) when the TC3 stops (TC3S = 0).
Note3 : Values to be loaded to timer register 3A must satisfy the following condition.
 $TREG3A > 0$ (in the timer and event counter modes)
Note4 : TC3CR is write-only register and must not be used with any of the read-modify-write instruction.

Figure 2-21. Timer Register 3 and TC3 Control Register

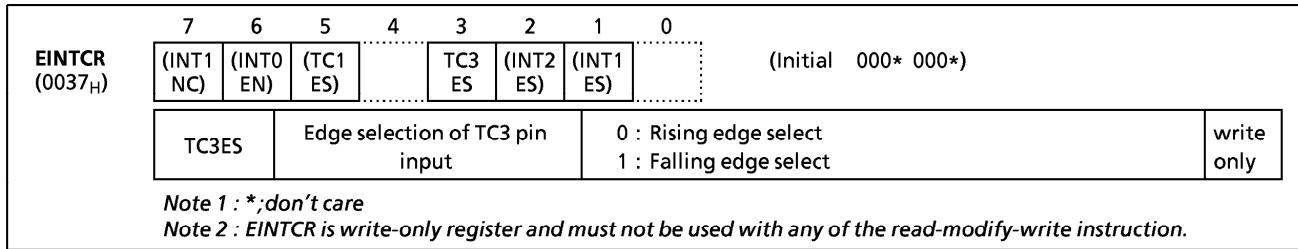


Figure 2-22. External Interrupt Control Register

2.6.3 Function

The TC3 has three operating modes : timer, event counter, and capture mode.

(1) Timer mode

In this mode, the internal clock shown in Table2-5 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 to "0". 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 to "0" after capturing.

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

Source clock		Resolution			maximum setting time	
NORMAL1/2, IDLE1/2 mode		SLOW, SLEEP mode	fc = 8MHz	At fs = 32.768 kHz	fc = 8MHz	At fs = 32.768 kHz
DV7CK = 0	DV7CK = 1					
$fc/2^{12}$ [Hz]	$fs/2^4$ [Hz]	$fs/2^4$ [Hz]	512 μ s	488.28 μ s	131.1 ms	125 ms
$fc/2^{10}$	$fs/2^2$	–	128 μ s	122.07 μ s	32.8 ms	31.2 ms
$fc/2^1$	–	–	16 μ s		4.1 ms	

(2) Event Counter mode

In this mode, the TC3 pin input pulses are used for counting up. Either the rising or falling edge can be selected with TC3ES (bit 3 in EINTCR). The contents of TREG3A are compared with the contents of 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 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, inputting 50Hz pulses to the TC3 pin.

```
LD (TC3CR), 00001100B ; Sets TC3 mode and source clock
LD (TREG3A), 19H ; 0.5s ÷ 1/50 = 25 = 19H
SET (EIRH).EF11 ; Enables INTTC3
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 shown in Table 2-5. On the rising (falling) edge of the TC3 pin input, the current contents of up-counter are loaded into TREG3A, then the up-counter is cleared to "0" and an INTTC3 interrupt is generated. On the falling (rising) edge of the TC3 pin input, the current contents of up-counter are loaded into to TREG3B. In this case, counting continues. On the next rising (falling) edge, 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 into TREG3A and an overflow interrupt (INTTC3) is generated. During interrupt processing, it can be determined whether or not there is an overflow by checking whether or not the TREG3A value is FF_H. Also, after an interrupt (TREG3A capture or overflow detection) is generated, capture and overflow detection are halted until TREG3A has been read out; however, the counter continues.

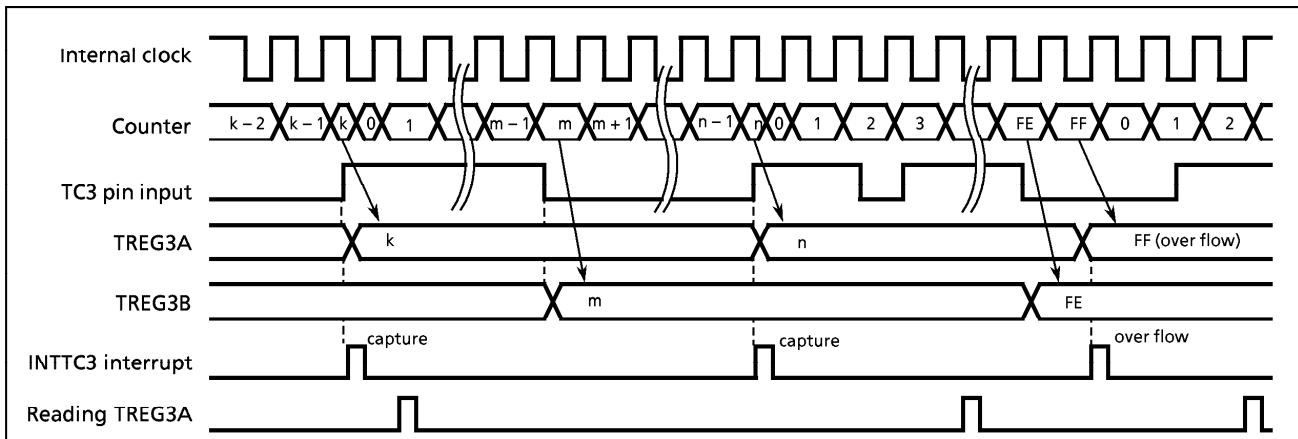


Figure 2-23. Capture Mode Timing Chart (TC3ES = 0)

2.7 8-bit Timer / Counter (TC4)

2.7.1 Configuration

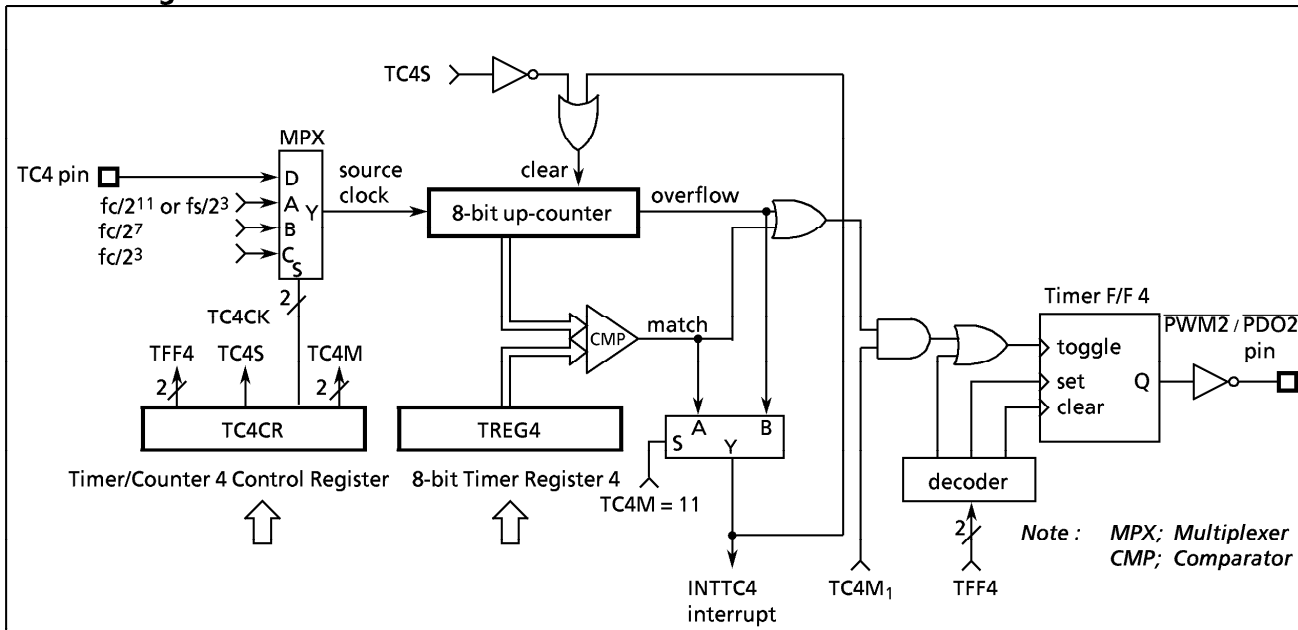


Figure 2-24. Timer / Counter 4

2.7.2 Control

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

TREG4 (001B _H)	7	6	5	4	3	2	1	0		
	Write only									
TC4CR (001C _H)	7	6	5	4	3	2	1	0		
	TFF4			TC4S	TC4CK			TC4M		
	(Initial value: 00*0 0000)									

TC4M	TC4 operating mode select	00 : Timer / event counter mode 01 : Reserved 10 : Programmable divider output (PDO) mode 11 : Pulse width modulation (PWM) output mode	
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 : External clock (TC4 pin input)	Write only
TC4S	TC4 start control	0 : Stop & clear 1 : Start	
TFF4	Timer F/F4 control	00 : Clear 01 : Toggle 10 : Set 11 : -	

*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 and timer F/F4 control when the TC4 stops (TC4S = 0).

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

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

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

Figure 2-25. Timer Register 4 and TC4 Control Register

2.7.3 Function

The TC4 has four operating modes : timer, event counter, programmable divider output, and pulse width modulation output mode.

(1) **Timer mode**

In this mode, the internal clock is used for counting up. The contents of TREG4 are compared with the contents of up-counter. Matching with TREG4 generates a timer / counter 4 interrupt (INTTC4) and clears the counter. Counting up resumes after the up-counter is cleared.

Table 2-6. Source Clock (Internal Clock) for TC4

Source clock		SLOW, SLEEP mode	Resolution		maximum setting time	
NORMAL1/2, IDLE1/2 mode DV7CK = 0	IDLE1/2 mode DV7CK = 1		$fc = 8\text{MHz}$	At $fs = 32.768$ kHz	$fc = 8\text{MHz}$	At $fs = 32.768$ kHz
$fc/2^{11}$ [Hz]	$fs/2^3$ [Hz]	$fs/2^3$ [Hz]	256 μs	244.14 μs	65.536 ms	62.5 ms
$fc/2^7$		-	16 μs	-	4.096 ms	-
$fc/2^3$		-	1 μs	-	256 μs	-

(2) **Event Counter mode**

In this mode, the TC4 pin input (external clock) pulse is used for counting up. Matching with TREG4 generates an INTTC4 interrupt and clears the counter. 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.

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

The internal clock is used for counting up. The contents of the TREG4 are compared with the contents of the up-counter. Timer F/F4 output is toggled and the counter is cleared each time a match is found. Timer F/F4 output is inverted and output to the $\overline{PDO2}$ (P44) pin. This mode can be used for 50% duty pulse output. Timer F/F4 can be initialized by program, and is initialized to "0" during reset. An INTTC4 interrupt is generated each time the $\overline{PDO2}$ output is toggled.

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

```

SET   (P4).4           ; P44 output latch←1
LD    (TC4CR), 00000110B ; Initializes the TC4 mode, source clock and timer F/F 4
LD    (TREG4), 10H      ; 1/2048 ÷ 27/fc = 10H
LD    (TC4CR), 00010110B ; Starts TC4
    
```

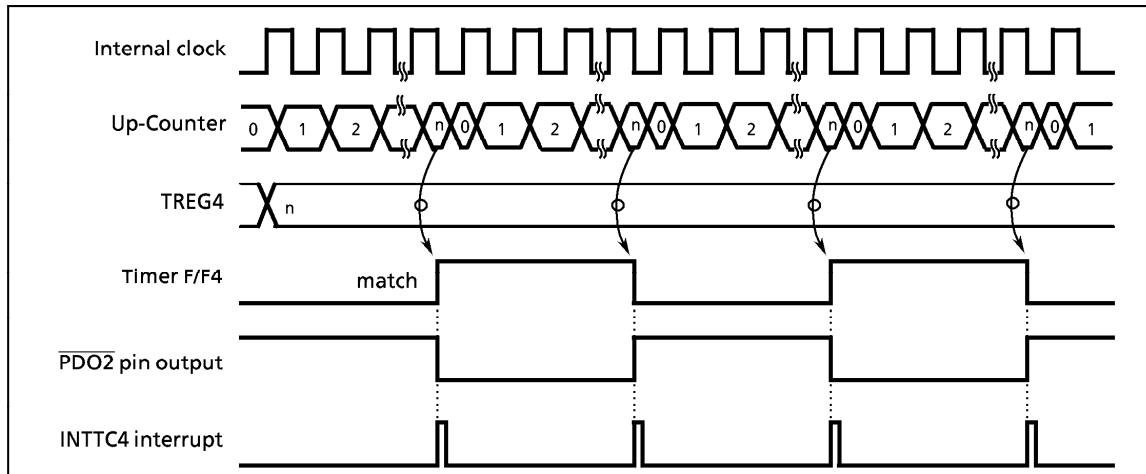


Figure 2-26. PDO Mode Timing Chart

(4) **Pulse width modulation (PWM) output mode**

PWM output with a resolution of 8 bits is possible. The internal clock is used for counting up. The contents of TREG4 is compared with the contents of up-counter. If a match is found, the timer F/F4 output is toggled. The counter continues counting and, when an overflow occurs, the timer is again toggled and the counter is cleared. Timer F/F4 output is inverted and output to the $\overline{PWM2}$ (P44) pin. An INTTC4 interrupt is generated when an overflow occurs.

TREG4 is configured a 2-stage shift register and, during output, will not switch until one output cycle is completed even if TREG4 is overwritten; therefore, output can be altered continuously. Also, the first time, TREG4 is shifted by setting TC45 (bit 4 in TC4CR) to "1" after data are loaded to TREG4.

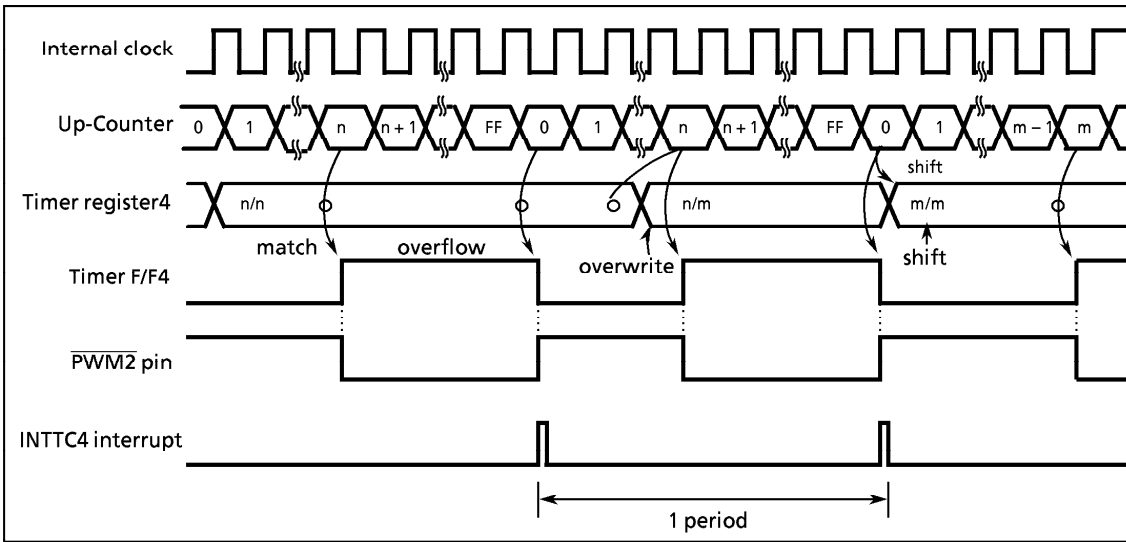


Figure 2-27. Timing Chart for PWM Mode

Table 2-7. PWM Output Mode

Source clock		SLOW, SLEEP mode	Resolution		Repeat cycle	
NORMAL1/2, IDLE1/2 mode	DV7CK = 1		At $f_c = 8\text{MHz}$	At $f_s = 32.768\text{kHz}$	At $f_c = 8\text{MHz}$	At $f_s = 32.768\text{kHz}$
DV7CK = 0	DV7CK = 1	$f_s / 2^3$ [Hz]	256 μs	244.14 μs	65.536 ms	62.5 ms
$f_c / 2^{11}$ [Hz]	$f_s / 2^3$ [Hz]	-	16 μs	-	4.096 ms	-
$f_c / 2^7$	-	-	1 μs	-	256 μs	-
$f_c / 2^3$	-	-	-	-	-	-

2.8 8-bit Timer/Counter 5 (TC5) and 8-bit Timer/Counter 6 (TC6)

The difference between timer/counter 5 (TC5) and timer/counter 6 (TC6) is the address of control register and timer register. Only timer/counter 5 will be explained below if not specifically related.

2.8.1 Configuration

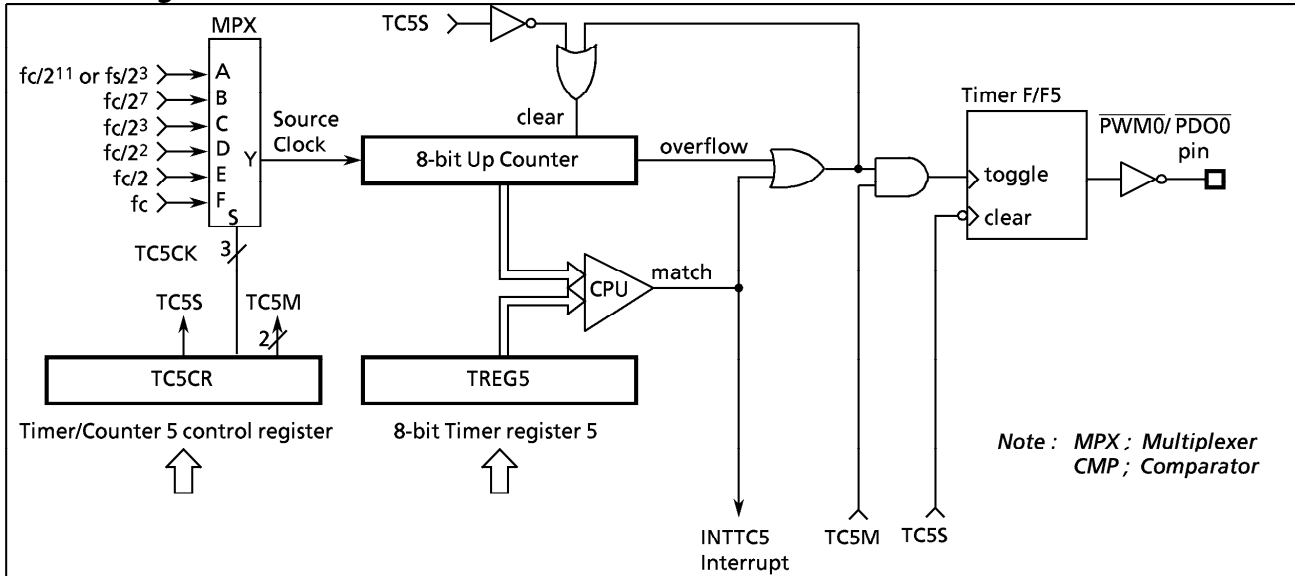


Figure 2-28. Timer/Counter 5 (TC5)

2.8.2 Control

The TC5 is controlled by a timer/counter 5 control register (TC5CR) and an 8-bit timer register 5 (TREG5). The address of timer/counter 6 register (TC6CR) is "0017H", while that of timer register 6 (TREG6) is "0016H".

TREG5 (001DH)	7 6 5 4 3 2 1 0	Write only
TC5CR (001EH)	7 6 5 4 3 2 1 0	(Initial **00 0000)
	TC5S TC5CK TC5M	
TC5M	TC5 Operating mode select	00 : Timer mode 01 : Reserved 10 : Programmable divider output (PDO) mode 11 : Pulse width modulation (PWM) output mode
TC5CK	TC5 Source clock select	000 : Reserved 001 : Internal clock $fc/2^{11}$ or $fs/2^3$ [Hz] 010 : Internal clock $fc/2^7$ 011 : Internal clock $fc/2^3$ 100 : Internal clock $fc/2^2$ 101 : Internal clock $fc/2$ 110 : Internal clock fc 111 : Reserved
TC5S	TC5 Start control	0 : Stop & clear 1 : Start

Note 1: fc ; High-frequency clock [Hz], fs ; Low-frequency clock [Hz], * ; don't care
 Note 2: The set value of timer register must satisfy the following conditions.
 (a) When in PWM output mode, $5 < TREG5 < 251$
 (b) When in any other mode than PWM output mode, $0 < TREG5$
 Note 3: Source clock $fc/2^2$, $fc/2$, and fc cannot be used except in PWM output mode.
 Note 4: Set the operating mode and the source clock selection when timer/counter stops ($TC5S = 0$).

Figure 2-29. Timer/Counter 5 Timer register, Control register

2.8.3 Function

TC5 has 3 operating modes : timer, programmable divider output, and pulse width modulation output mode.

(1) Timer mode

In this mode, the internal clock is used for counting up. The contents of the timer register 5 (TREG5) is compared with the contents of the up-counter. Matching with TREG5 generates a timer/counter 5 interrupt (INTTC5) and clears the counter. Counting up resumes after the counter is cleared.

Table 2-8. Source Clock (Internal clock) for TC5

Source clock		SLOW, SLEEP mode	resolution		maximum setting time	
NORMAL1/2, IDLE1/2 mode DV7CK = 0	DV7CK = 1		fc = 8MHz	fs = 32.768 kHz	fc = 8MHz	fs = 32.768 kHz
fc/2 ¹¹ [Hz]	fs/2 ³ [Hz]	fs/2 ³ [Hz]	256 μs	244.14 μs	65.536 ms	62.5 ms
fc/2 ⁷		–	16 μs		4.096 ms	
fc/2 ³		–	1 μs		256 μs	

(2) Programmable divider output (PDO) mode

The internal clock is used for counting up. The contents of the TREG5 are compared with the contents of the up-counter. The timer F/F5 output is toggled and the counter is cleared each time a match is found. The timer F/F5 output is inverted and output to the P $\overline{D}O0$ (P30) pin. This mode can be used for 50% duty pulse output. INTTC5 interrupt is generated each time the P $\overline{D}O0$ output is toggled.

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

```

SET (P3).0 ; P30 output latch←1
LD (TC5CR), 00001010B ; Sets to TC5 modes and source clock
LD (TREG5), 10H ; 1/2048 ÷ 27/fc = 10H
LD (TC5CR), 00101010B ; Starts TC5
    
```

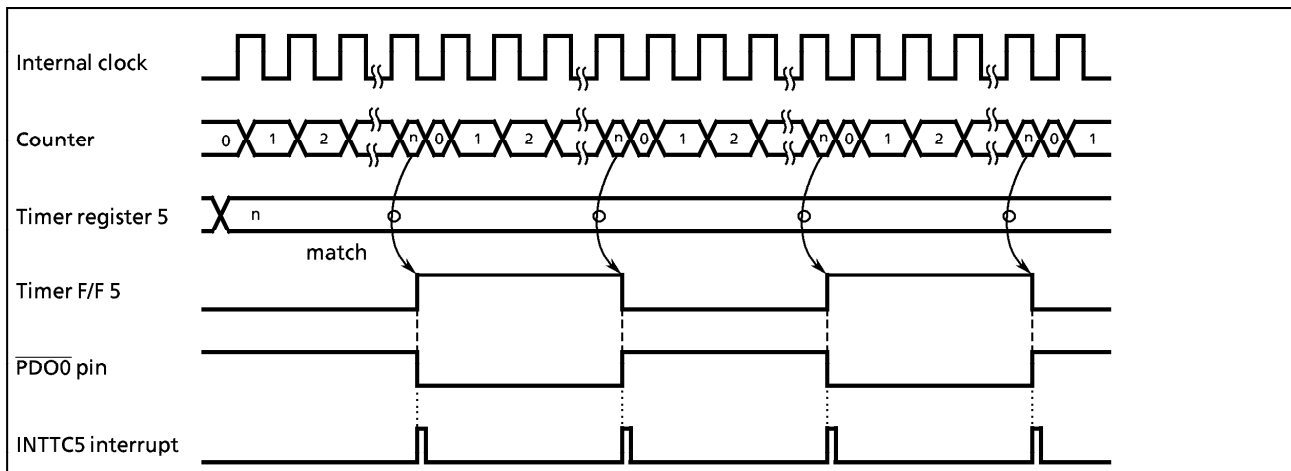


Figure 2-30. PDO Mode Timing Chart

(3) Pulse width modulation (PWM) output mode

PWM output with a resolution of 8-bits is possible. The internal clock is used for counting up. The contents of the TREG5 is compared with the contents of the up-counter. If a match is found, the timer F/F5 output is toggled. The counter continues counting and, when an overflow occurs, the timer is again toggled and the counter is cleared. The timer F/F5 output is inverted and output to the $\overline{\text{PWM}}$ (P30) pin. An INTTC5 interrupt is generated when an overflow occurs.

TREG5 is configured a 2-stage shift register and, during output, will not switch until one output cycle is completed even if TREG5 is overwritten; therefore, output can be altered continuously. Also, the first time, TREG5 is shifted by setting TC5S (bit 5 in TC5CR) to "1" after data are loaded to TREG5.

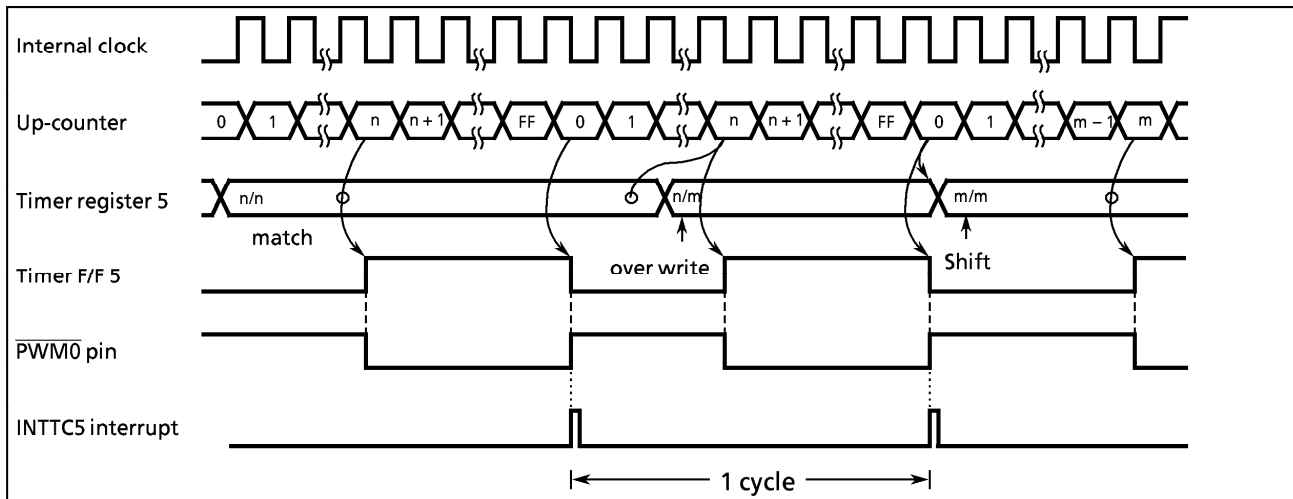


Figure 2-31. PWM Output Mode Timing Chart

Table 2-9. PWM Output Mode

Source clock		Resolution		Repeat cycle	
NORMAL1/2, IDLE1/2 mode		At $f_c = 8\text{MHz}$	At $f_c = 4.194304\text{MHz}$	At $f_c = 8\text{MHz}$	At $f_c = 4.194304\text{MHz}$
DV7CK = 0	DV7CK = 1				
$f_c/2^2$ [Hz]		500ns	953.7ns	128 μs	244 μs
$f_c/2$		250ns	476.8ns	64 μs	122 μs
f_c		125ns	238.4ns	32 μs	61 μs

2.9 Serial Interface (SIO)

The 87CC20/H20/K20A/M20A each have a clocked-synchronous 8-bit serial interface. Serial interface has an 8-byte transmit and receive data buffer that can automatically and continuously transfer up to 64 bits of data.

The serial interface is connected to external devices via the pins P47 (SO), P46 (SI), P45 (\overline{SCK}). The serial interface pins are also used as P4. When used as serial interface pins, set these pins of port P4 output latch to "1", in the transmit mode, the pin P46 can be used as a normal I/O port; and in the receive mode, pin P47 can be used as a normal I/O port.

2.9.1 Configuration

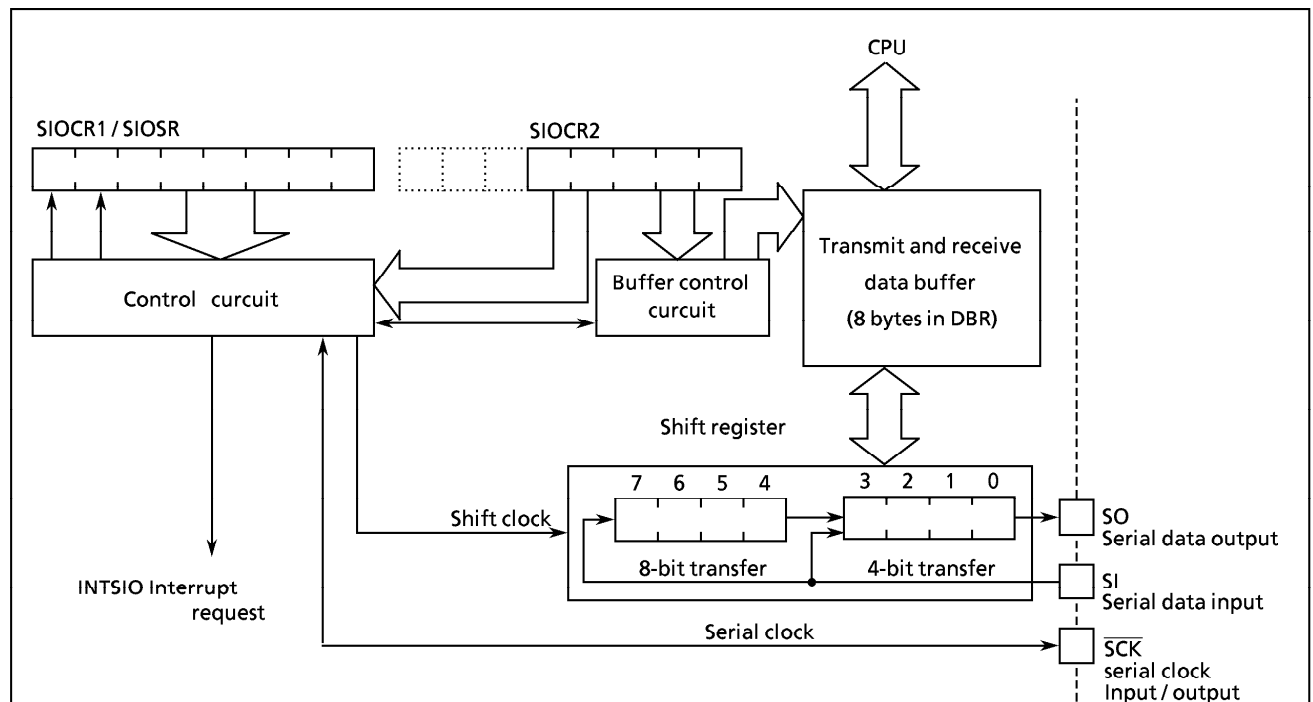


Figure2-32. Serial Interface

2.9.2 Control

The SIO is controlled by SIOCR1, SIOCR2. The SIO operating status can be determined by reading SIOSR. The transmit and receive data buffer is controlled by SIOCR2. The data buffer is assigned to address 0FF0-0FF7H in the DBR area, and can continuously transfer up to 8 word (bytes or nibbles) at one time. When the specified number of bytes has been transferred, a buffer empty (in the transmit mode) or a buffer full (in the receive mode or transmit/receive mode receiving) interrupt (INTSIO) is generated.

When the internal clock is used as the serial clock, in the 8-bit receive mode and the 8-bit transmit/receive mode a fixed internal wait can be applied to the serial clock for each word transferred. Any one of four different wait times can be selected with SIOCR2.

Serial Interface Control Register

SIOCR1 (0020 _H)	7	6	5	4	3	2	1	0	(Initial value: 0000 0000)
	SIOS	SIOINH		SIOM			SCK		
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	000 : 8-bit transmit mode 010 : 4-bit transmit mode 100 : 8-bit transmit / receive mode 101 : 8-bit receive mode 110 : 4-bit receive mode							
SCK	Serial clock select	000 : Internal clock $fc/2^{13}$ or $fs/2^5$ [Hz] 001 : Internal clock $fc/2^8$ 010 : Internal clock $fc/2^6$ 011 : Internal clock $fc/2^5$ 111 : External clock (input from \overline{SCK} pin)							
<p><i>Note 1 : Set SIOS = 0 and SIOINH to "1" when setting the transfer mode, and serial clock.</i></p> <p><i>Note 2 : fc ; High-frequency clock [Hz], fs ; Low-frequency clock [Hz]</i></p> <p><i>Note 3 : SIOCR1 is write-only register, which cannot access any of in read-modify-write instruction such as bit operate, etc.</i></p>									
Serial Interface Status Register									
SIOSR (0020 _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 2-33. Serial Interface Control Register1 / Status Register

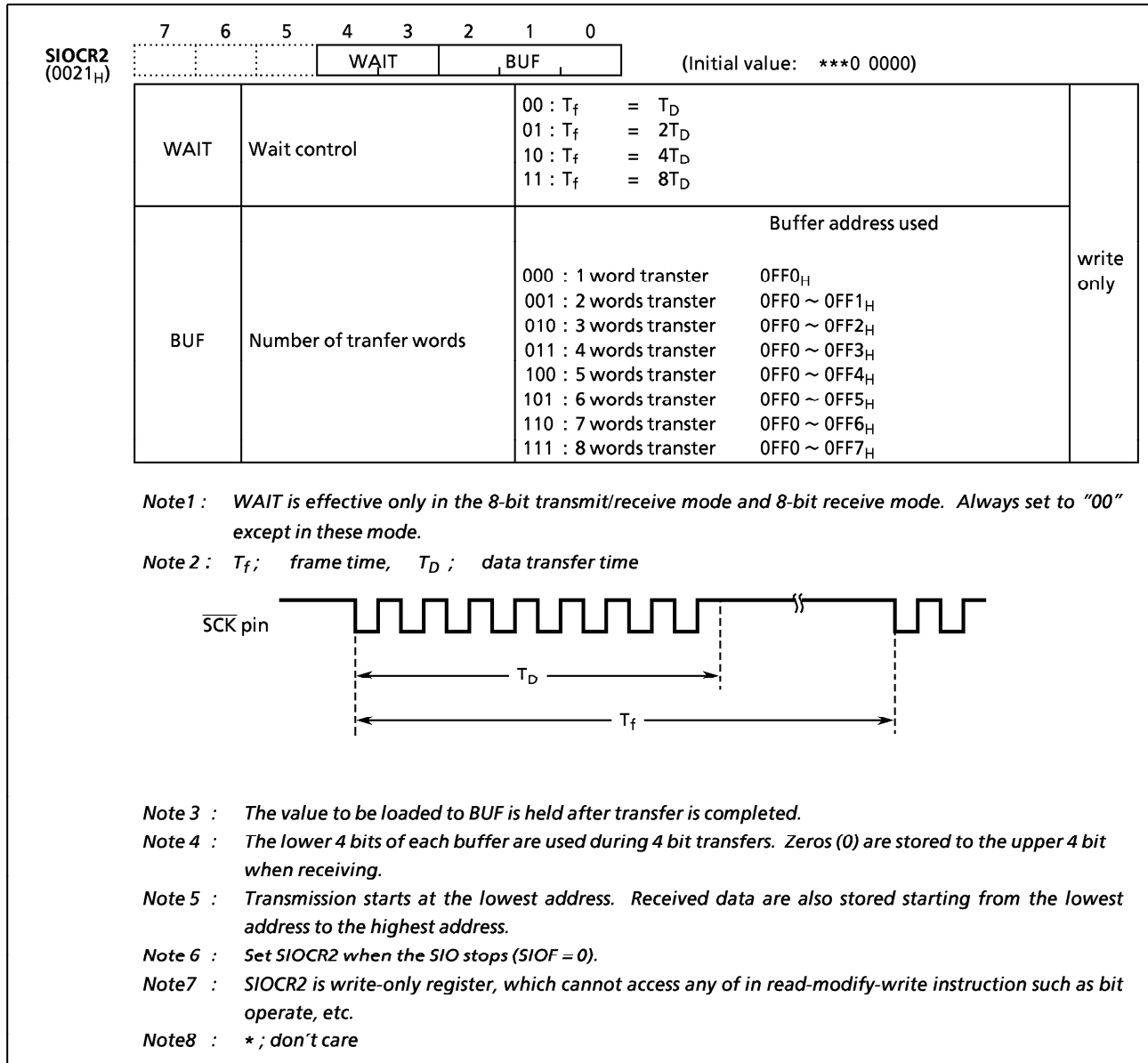


Figure 2-34. Serial Interface Control Register2

(1) Serial clock

SCK (bits 2 - 0 in SIOCR1) are able to select the following:

① Internal clock

Any of four frequencies can be selected. The serial clock is output on the $\overline{\text{SCK}}$ pin. The $\overline{\text{SCK}}$ pin goes high when transfer starts.

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

Table 2-10. Serial Clock Rate

Serial clock		SLOW, SLEEP mode	Maximum transfer rate	
NORMAL1/2, IDLE1/2 mode			At $f_c = 8\text{MHz}$	At $f_s = 32.768\text{kHz}$
DV7CK = 0	DV7CK = 1			
$f_c / 2^{13}$ [Hz]	$f_s / 2^5$ [Hz]	$f_s / 2^5$ [Hz]	0.977 Kbit/s	1 Kbit/s
$f_c / 2^8$	$f_c / 2^8$	-	31.2	-
$f_c / 2^6$	$f_c / 2^6$	-	125	-
$f_c / 2^5$	$f_c / 2^5$	-	250	-

Note : 1Kbit = 1024bit

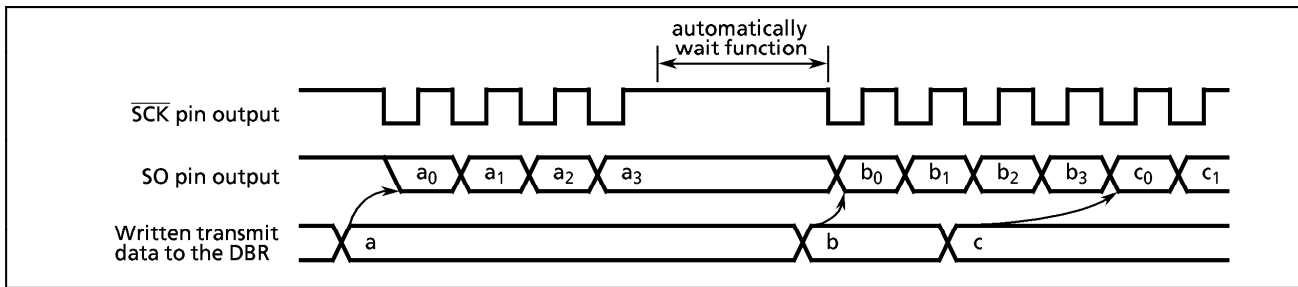
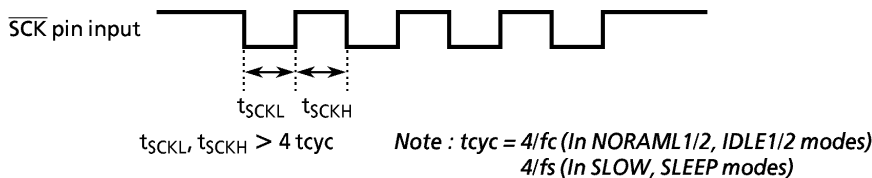


Figure 2-35. Clock Source (Internal Clock)

② External Clock

An external clock connected to the $\overline{\text{SCK}}$ pin is used as the serial clock. In this case, the P45 ($\overline{\text{SCK}}$) output latch must be set to "1". To ensure shifting, a pulse width of at least 4 machine cycles is required. Thus, the maximum transfer speed is 250K-bit/s. (at $f_c = 8\text{MHz}$).



b. Shift edge

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

① Leading Edge

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

② Trailing Edge

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

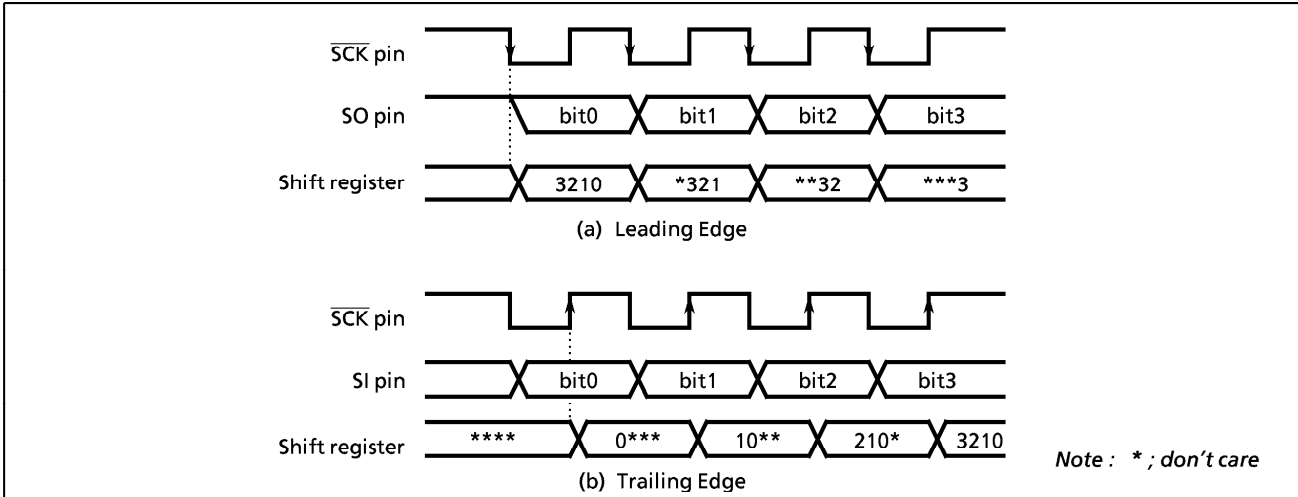


Figure 2-36. Shift Edge

(2) Number of bits transferred

Either 4-bit or 8-bit serial transfer can be selected. When 4-bit serial transfer selected, only the lower 4 bits of the transmit / receive data buffer are used. The upper 4 bits are cleared to "0" when receiving.

The data are transferred in sequence starting at the least significant bit (LSB).

(3) Number of Words to Transfer

Up to 8 words consisting of 4 bits of data (4-bit serial transfer) or 8 bits (8-bit serial transfer) of data can be transferred continuously. The number of words to be transferred is loaded to BUF in SIOCR2. An INTSIO interrupt is generated when the specified number of words has been transferred. If the number of words is to be changed during transfer, the serial interface must be stopped before making the change.

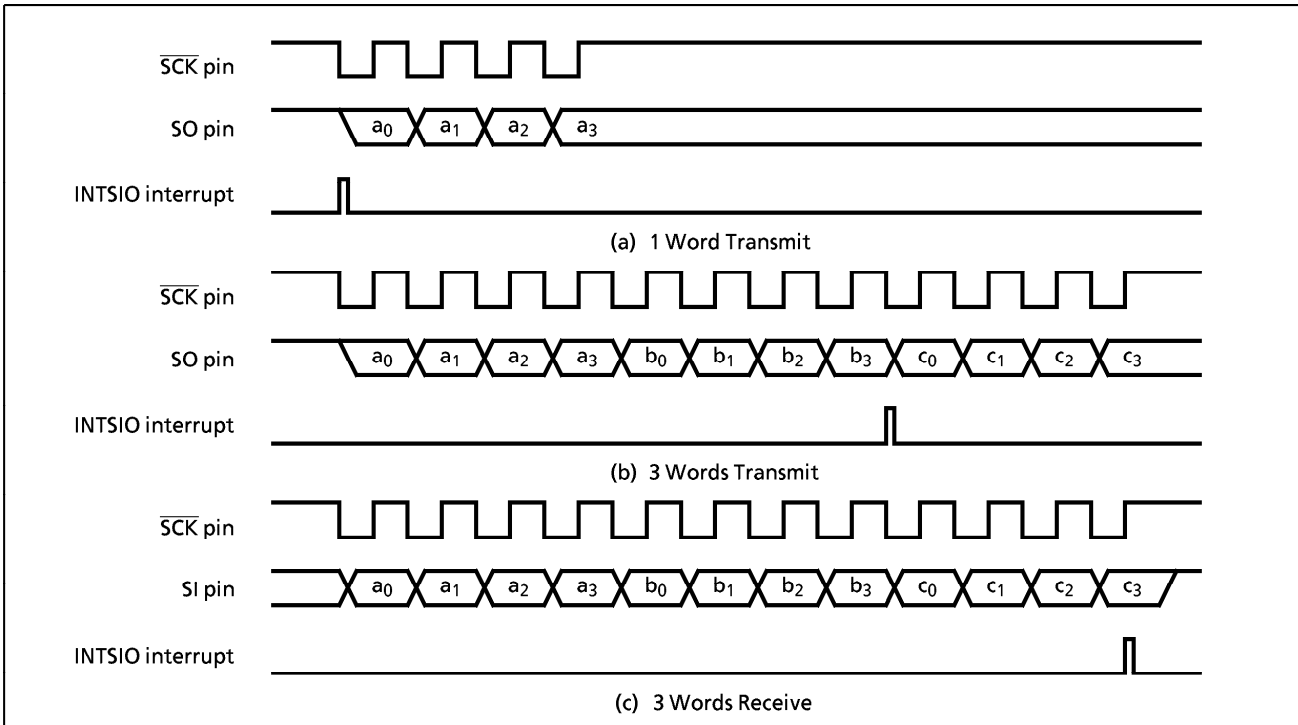


Figure 2-37. Number of Bits to Transfer (Example : 4-bit serial transfer)

2.9.3 Transfer Mode

S_{IOM} (bits 5 - 3 in S_{IOCR1}) is used to select the transmit, receive, or transmit/receive mode.

(1) 4-bit and 8-bit Transmit Modes

In these modes, the S_{IOCR1} is set to the transmit mode and then the data to be transmitted first are written to the data buffer registers (DBR). After the data are written, the transmission is started by setting S_{IOS} to "1". The data are then output sequentially to the SO pin in synchronous with the serial clock, starting with the least significant bit (LSB). As soon as the LSB has been output, the data are transferred from the data buffer register to the shift register. When the final data bit has been transferred and the data buffer register is empty, an INTS_{IO} (buffer empty) interrupt is generated to request the next transmitted data.

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

<p><i>Note :</i> <i>Waits are also canceled by writing to a DBR not being used as a transmit data buffer register; therefore, during SIO do not use such DBR for other applications.</i></p>

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

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

The transmission is ended by clearing S_{IOS} to "0" at the time that the final bit of the data being shifted out has been transferred. That the transmission has ended can be determined from the status of S_{IOF} (bit 7 in S_{IOSR}) because S_{IOF} is cleared to "0" when a transfer is completed.

When an external clock is used, it is also necessary to clear S_{IOS} to "0" before shifting the next data; otherwise, dummy data will be transmitted and the operation will end.

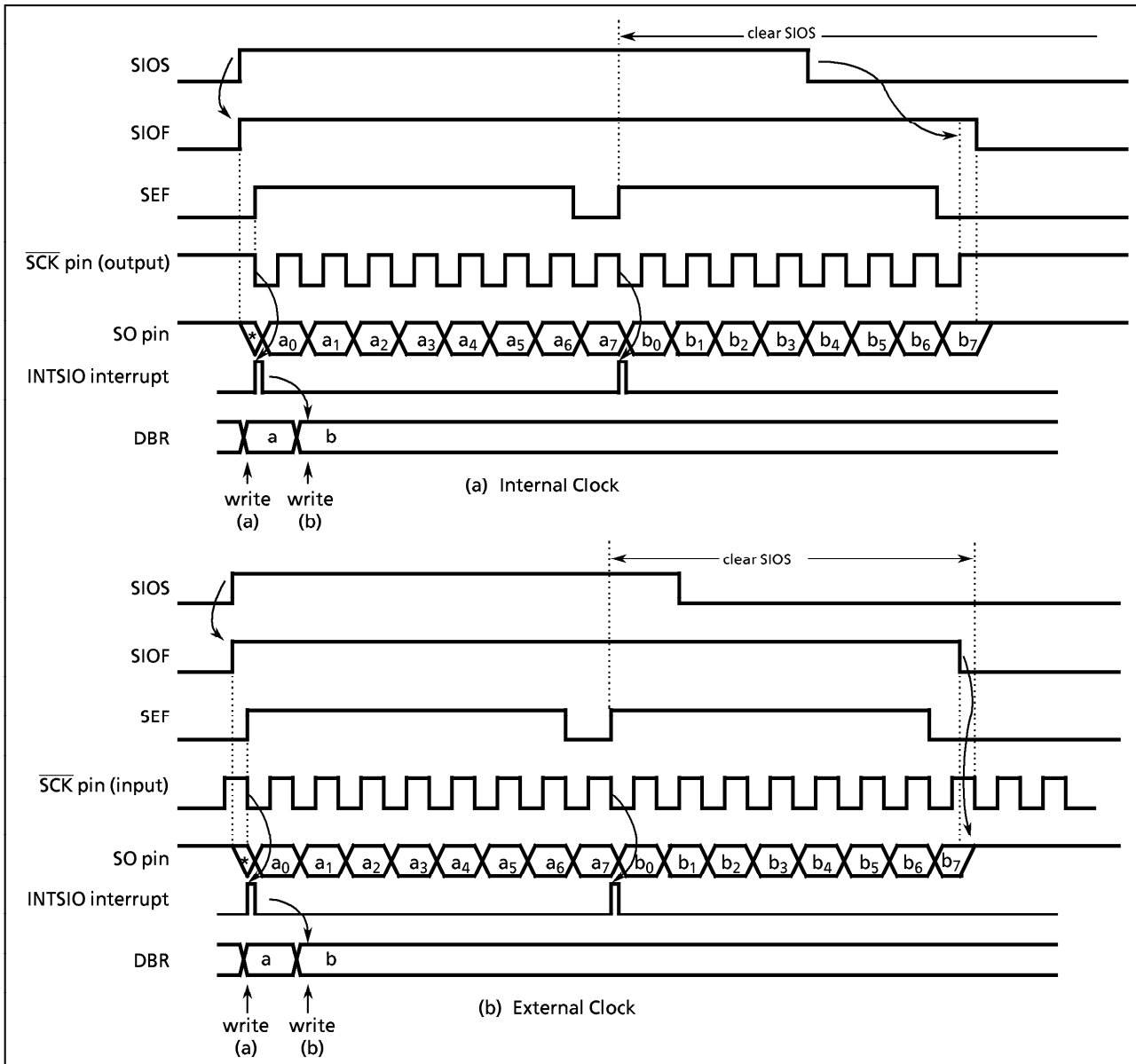


Figure 2-38. Transfer Mode (Example: 8-bit, 1 Word Transfer)

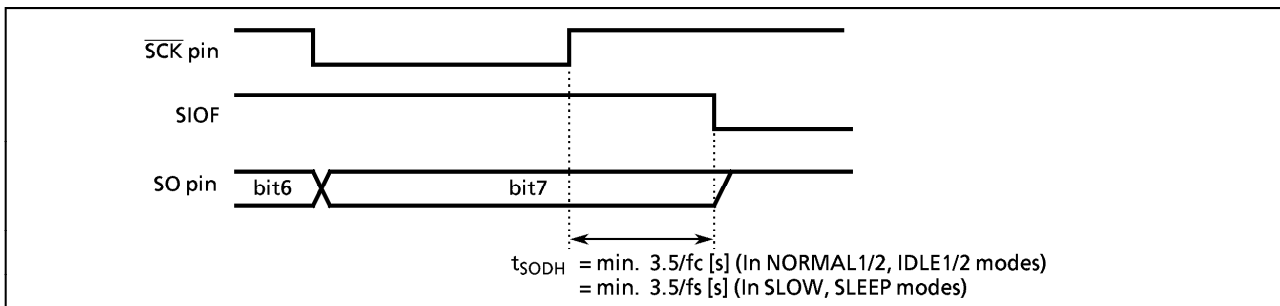


Figure 2-39. Transmitted Data Hold Time at End of Transmit

(2) 4-bit and 8-bit Receive Modes

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

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

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

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

Clear SIOS to "0" to end receiving. When SIOS is cleared, the current data are transferred to the buffer in 4-bit or 8-bit blocks. The receiving mode ends when the transfer is completed. SIOF is cleared to "0" when receiving is ended and thus can be sensed by program to confirm that receiving has ended.

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

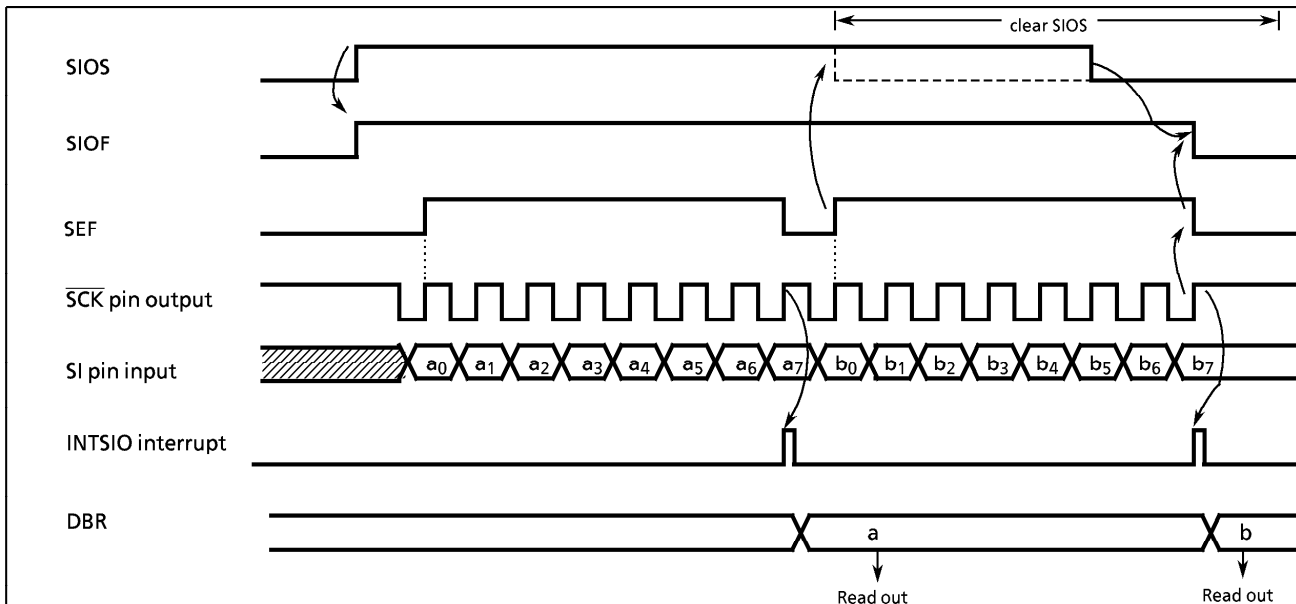


Figure 2-40. Receive Mode (Example : 8-bit, 1 word, internal clock)

(3) 8-bit Transmit/Receive Mode

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

When the internal clock is used, a wait is initiated until the received data are read and the next data are written.

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

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} .

Clear SIOS to "0" to enable the transmit mode. When SIOS is cleared, the current data are transferred to the data buffer register in 8-bit blocks. The transmit mode ends when the transfer is completed. SIOF is cleared to "0" when receiving is ended and thus can be sensed by program to confirm that receiving has ended.

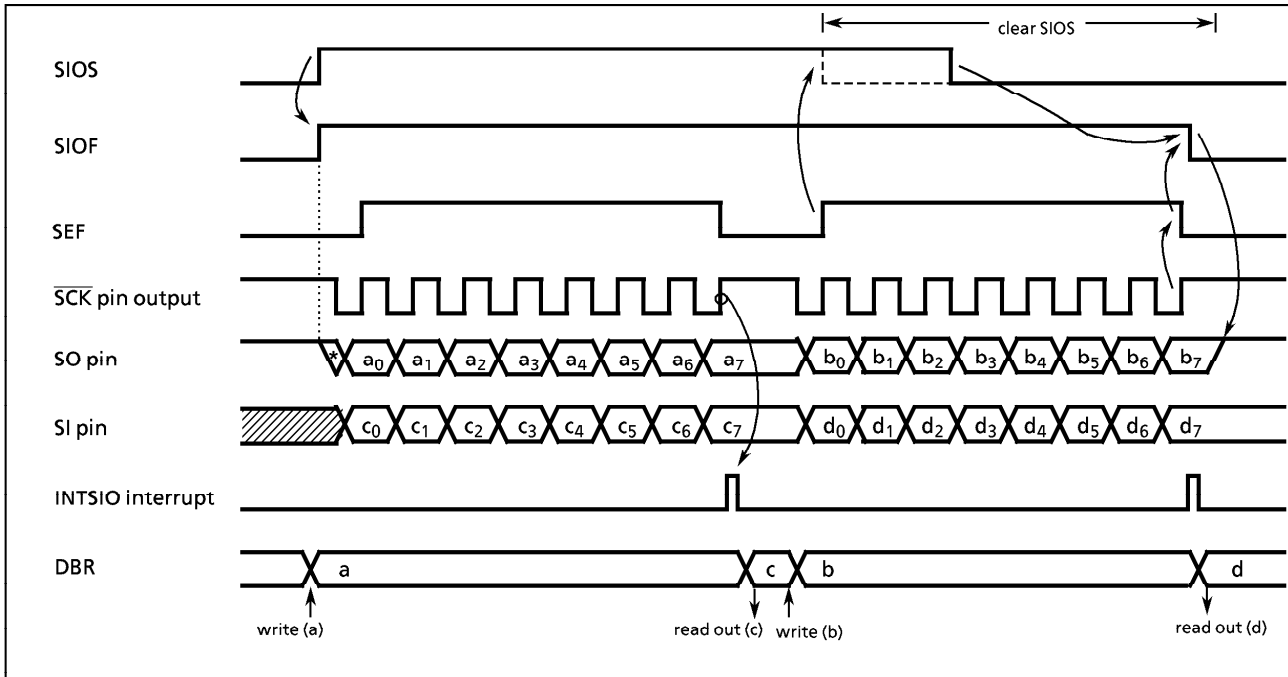


Figure 2-41. Transmit/Receive Mode (Example : 8-bit, 1word, internal clock)

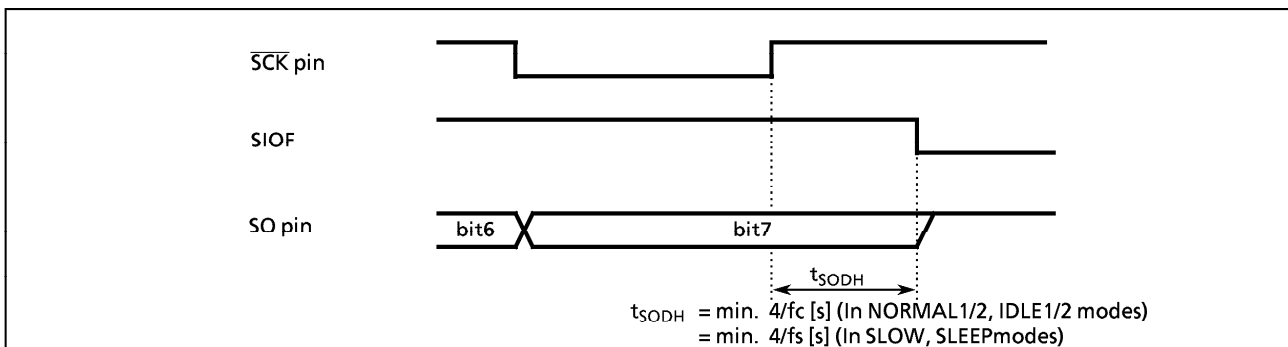


Figure 2-42. Transmitted Data Hold Time at End of Transmit/Receive

2.10 LCD Driver

The 87CC20/H20/K20A/M20A each have a driver and control circuit to directly drive the liquid crystal display (LCD). The pins to be connected to LCD are as follows:

- ① Segment output port 24 pins (SEG23 to SEG0)
- ② Segment output or P6 input / output port 8 pins (SEG31 to SEG24)
- ③ Common output port 4 pins (COM3 to COM0)

In addition, VLC pin is provided as the drive power pins

The devices that can be directly driven is selectable from LCD of the following drive methods:

- ① 1/4 Duty (1/3 Bias) LCD Max.128 Segments (8-segment x 16 digits)
- ② 1/3 Duty (1/3 Bias) LCD Max. 96 Segments (8-segment x 12 digits)
- ③ 1/3 Duty (1/2 Bias) LCD Max. 96 Segments (8-segment x 12 digits)
- ④ 1/2 Duty (1/2 Bias) LCD Max. 64 Segments (8-segment x 8 digits)
- ⑤ Static LCD Max. 32 Segments (8-segment x 4 digits)

2.10.1 Configuration

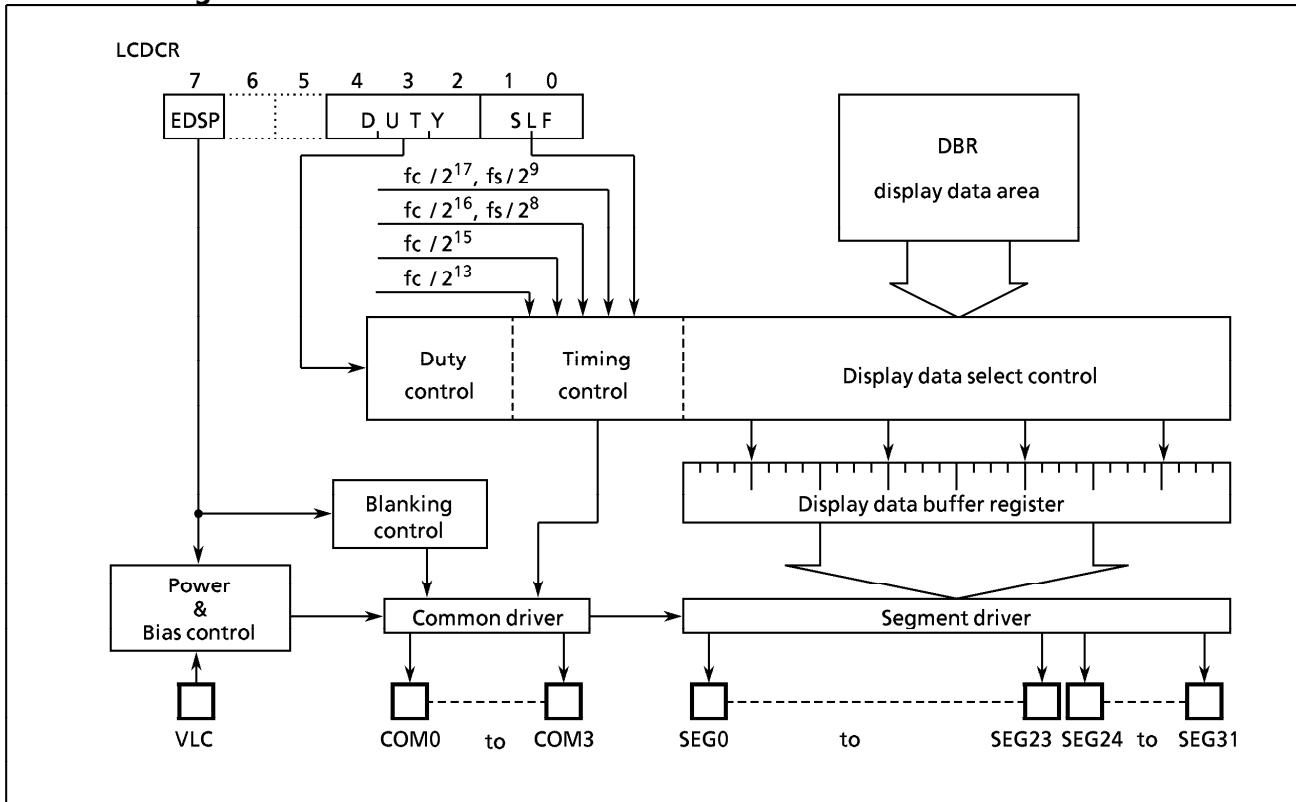


Figure 2-43. LCD Driver

2.10.2 Control

The LCD driver is controlled by the LCD control register (LCDCR).

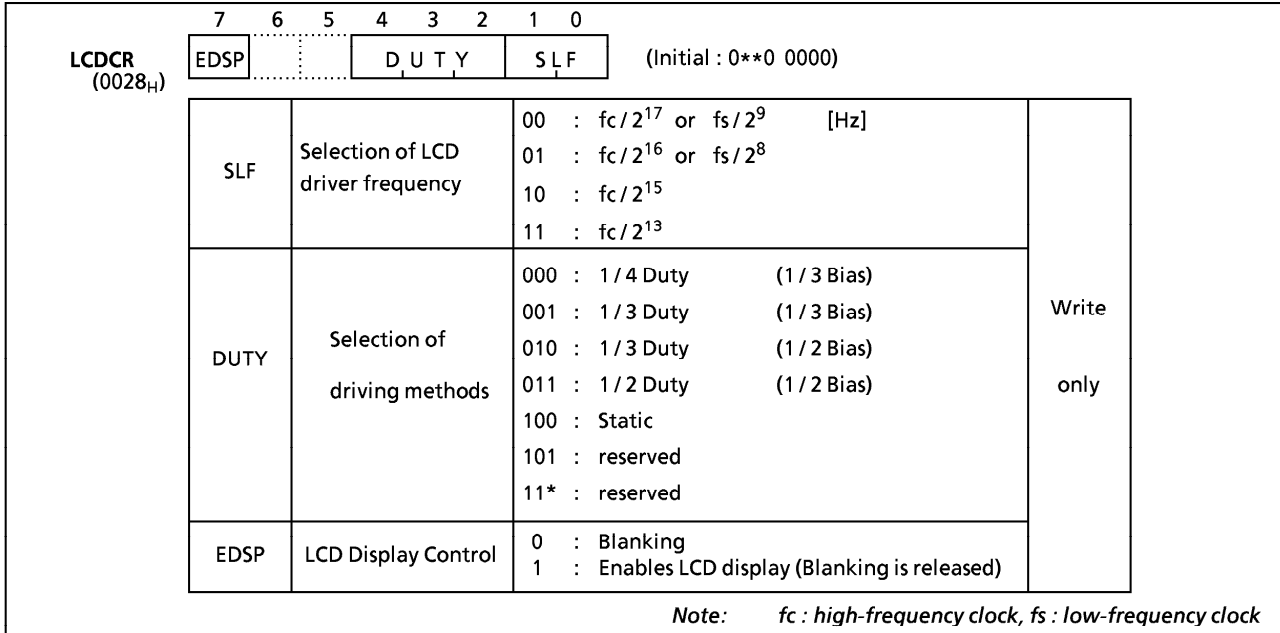


Figure 2-44. LCD Driver Control Register

(1) LCD driving methods

As for LCD driving method, 5 types can be selected by DUTY (bit 4 to bit 2 of LCDCR). The driving method is initialized in the initial program according to the LCD used.

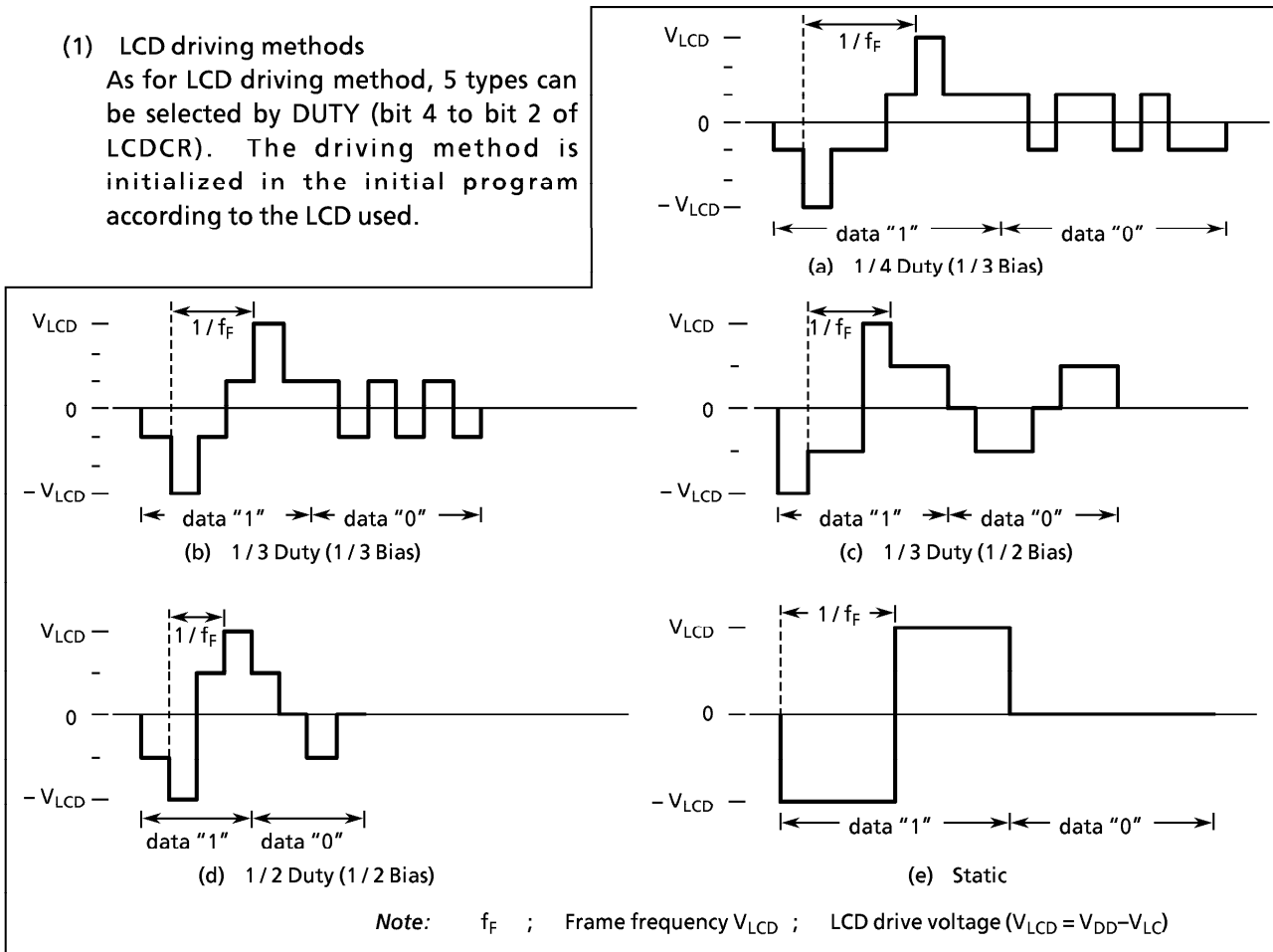


Figure 2-45. LCD Drive Waveform (COM – SEG Pins)

(2) Frame frequency

Frame frequency (f_F) is set according to driving method and base frequency as shown in the following Table 2-11. The base frequency is selected by SLF (Lower two bits of command register) according to the frequency f_c and f_s of the basic clock to be used.

a. At the single clock mode

Table 2-11. Setting of LCD Frame Frequency - 1

SLF	Base frequency [Hz]	Frame frequency [Hz]			
		1 / 4 Duty	1 / 3 Duty	1 / 2 Duty	Static
00	$\frac{f_c}{2^{17}}$	$\frac{f_c}{2^{17}}$	$\frac{4}{3} \cdot \frac{f_c}{2^{17}}$	$\frac{4}{2} \cdot \frac{f_c}{2^{17}}$	$\frac{f_c}{2^{17}}$
	($f_c = 8\text{MHz}$)	61	81	122	61
01	$\frac{f_c}{2^{16}}$	$\frac{f_c}{2^{16}}$	$\frac{4}{3} \cdot \frac{f_c}{2^{16}}$	$\frac{4}{2} \cdot \frac{f_c}{2^{16}}$	$\frac{f_c}{2^{16}}$
	($f_c = 4\text{MHz}$)	61	81	122	61
10	$\frac{f_c}{2^{15}}$	$\frac{f_c}{2^{15}}$	$\frac{4}{3} \cdot \frac{f_c}{2^{15}}$	$\frac{4}{2} \cdot \frac{f_c}{2^{15}}$	$\frac{f_c}{2^{15}}$
	($f_c = 4\text{MHz}$)	122	162	244	122
11	$\frac{f_c}{2^{13}}$	$\frac{f_c}{2^{13}}$	$\frac{4}{3} \cdot \frac{f_c}{2^{13}}$	$\frac{4}{2} \cdot \frac{f_c}{2^{13}}$	$\frac{f_c}{2^{13}}$
	($f_c = 1\text{MHz}$)	122	162	244	122

Note f_c ; High-frequency clock [Hz]

b. At the dual clock mode

Table 2-11. Setting of LCD Frame Frequency - 2

SLF	Base frequency [Hz]	Frame frequency [Hz]			
		1 / 4 Duty	1 / 3 Duty	1 / 2 Duty	Static
00	$\frac{f_s}{2^9}$	$\frac{f_s}{2^9}$	$\frac{4}{3} \cdot \frac{f_s}{2^9}$	$\frac{4}{2} \cdot \frac{f_s}{2^9}$	$\frac{f_s}{2^9}$
	($f_s = 32.768\text{kHz}$)	64	85	128	64
01	$\frac{f_s}{2^8}$	$\frac{f_s}{2^8}$	$\frac{4}{3} \cdot \frac{f_s}{2^8}$	$\frac{4}{2} \cdot \frac{f_s}{2^8}$	$\frac{f_s}{2^8}$
	($f_s = 32.768\text{kHz}$)	128	171	256	128

Note f_s ; Low-frequency clock [Hz]

(3) LCD drive voltage

LCD driving voltage V_{LCD} is given as potential difference $V_{DD} - V_{LC}$ between pins VDD and VLC. Therefore, when the CPU voltage and LCD drive voltage are the same, VLC pin will be connected to VSS pin. The LCD lights when the potential difference between segment output and common output is $\pm V_{LCD}$. Otherwise it turns off.

During reset, the power switch of LCD driver is automatically turned off, shutting off the VLC voltage. At the same time, both segment outputs and common outputs become at V_{DD} level, turning off the LCD. The power switch is turned on to supply VLC voltage to LCD driver by setting with EDSP (bit 7 in LCDCR) to "1". After that, the power switch will not turned off even during blanking (clearing EDSP to "0") and the VLC voltage continues flow. When STOP mode starts, the power switch will be turned off. Therefore, LCD light out, and stop operation is executed at low power consumption. When STOP mode is released the status in effect immediately before the STOP operation is reinstated.

2.10.3 LCD Display Operation

(1) Display data setting

Display data is stored to the display data area (assigned to address 0F80 to 0F8FH) in the DBR. The display data which are stored in the display data area is automatically read out and sent to the LCD driver by the hardware. The LCD driver generates the segment signal and common signal according to the display data and driving method. Therefore, display patterns can be changed by only over writing the contents of display data area by the program. Figure 2-46 shows the correspondence between the display data area and SEG/COM pins.

LCD light when display data is "1" and turn off when "0". According to the driving method of LCD, the number of pixels which can be driven becomes different, and the number of bits in the display data area which is used to store display data also becomes different.

Therefore, the bits which are not used to store display data as well as the data buffer which corresponds to the addresses not connected to LCD can be used to store general user process data (see Table 2-12.)

address	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0F80H			SEG1					SEG0
81			SEG3					SEG2
82			SEG5					SEG4
83			SEG7					SEG6
84			SEG9					SEG8
85			SEG11					SEG10
86			SEG13					SEG12
87			SEG15					SEG14
88			SEG17					SEG16
89			SEG19					SEG18
8A			SEG21					SEG20
8B			SEG23					SEG22
8C			SEG25					SEG24
8D			SEG27					SEG26
8E			SEG29					SEG28
8F			SEG31					SEG30
			COM3	COM2	COM1	COM0	COM3	COM2
							COM1	COM0

Figure 2-46. LCD Display Data Area (DBR)

Table 2-12. Driving Method and Bit for Display Data

Driving methods	bit 7/3	bit 6/2	bit 5/1	bit 4/0
1 / 4 Duty	COM3	COM2	COM1	COM0
1 / 3 Duty	-	COM2	COM1	COM0
1 / 2 Duty	-	-	COM1	COM0
Static	-	-	-	COM0

Note - ; This bit is not used for display data

(2) Blanking

Blanking is enabled when EDSP is cleared to "0".

Blanking turns off LCD through outputting a non-selective level to COM pin. A signal level is continuously output to SEG pin according to display data and driving method. For static drive, lights-out by data (clearing display data to "0") does not apply any voltage between pins COM and SEG. On the other hand, lights-out by blanking makes the output to COM pin at a constant $V_{LCD}/2$ level, so that the part between pins COM and SEG becomes in the state driven by $V_{LCD}/2$.

2.10.4 Control Method of LCD Driver

(1) Initial setting

Figure 2-47 shows the flowchart of initialization.

Example : To operate a 1/4 duty LCD of 32 segments \times 4 commons at frame frequency $f_c/2^{16}$ [Hz]

```
LD    (LCDCR),0000001B    ; Sets LCD driving method and
                           ; frame frequency.
LD    (P6CR),0FFH        ; Sets P6 port as segment
                           ; output .
      ...                 ; Sets the initial value of
                           ; display data.
LD    (LCDCR),1000001B    ; Display enable
```

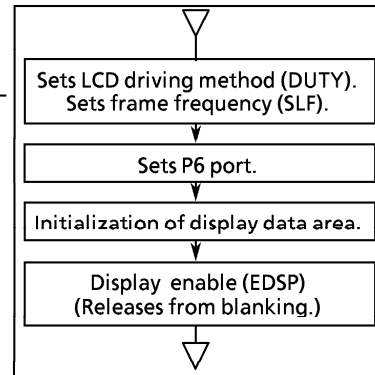


Figure 2-47. Initial Setting of LCD Driver

(2) Store of display data

Generally, display data are prepared as fixed data in program memory and stored in display data area by load command.

Example 1 : To display using 1/4 duty LCD a numerical value which corresponds to the LCD data stored in data memory at address 80_H (when pins COM and SEG are connected to LCD as in Figure 2-48), display data become as shown in Table 2-13.

```
LD    A, (80H)
ADD   A, TABLE - $ - 5
LD    HL, 0F80H
LD    (HL), (PC + A)
JRS   T, SNEXT
TABLE : DB 11011111B, 00000110B,
           11100011B, 10100111B,
           00110110B, 10110101B,
           11110101B, 00010111B,
           11110111B, 10110111B
SNEXT :
```

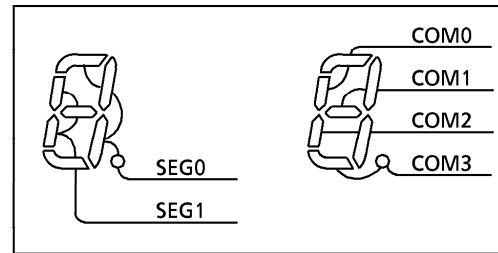
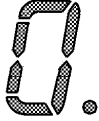



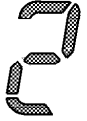

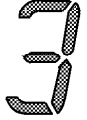

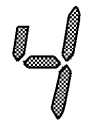
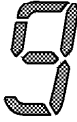


Figure 2-48. Example of COM, SEG Pin Connection (1/4 Duty)

Note : DB is a byte data definition instruction.

Table 2-13. Example of Display Data (1/4 Duty)

No.	display	display data	No.	display	display data
0		11011111	5		10110101
1		00000110	6		11110101
2		11100011	7		00010111
3		10100111	8		11110111
4		00110110	9		10110111

Example 2 : Table 2-14 shows an example of display data which are displayed using 1/2 duty LCD in the same way as Table 2-13. The connection between pins COM and SEG are the same as shown in Figure 2-49.

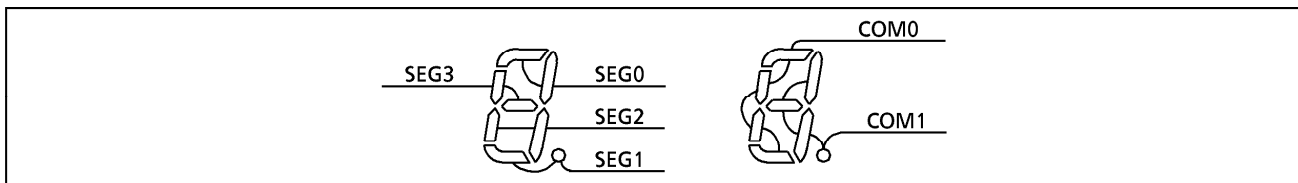


Figure 2-49. Example of COM, SEG Pin Connection

Table 2-14. Example of Display Data (1/2 Duty)

Number	display data		Number	display data	
	High order address	Low order address		High order address	Low order address
0	**01**11	**01**11	5	**11**10	**01**01
1	**00**10	**00**10	6	**11**11	**01**01
2	**10**01	**01**11	7	**01**10	**00**11
3	**10**10	**01**11	8	**11**11	**01**11
4	**11**10	**00**10	9	**11**10	**01**11

Note * ; don't care

(3) Example of LCD drive output

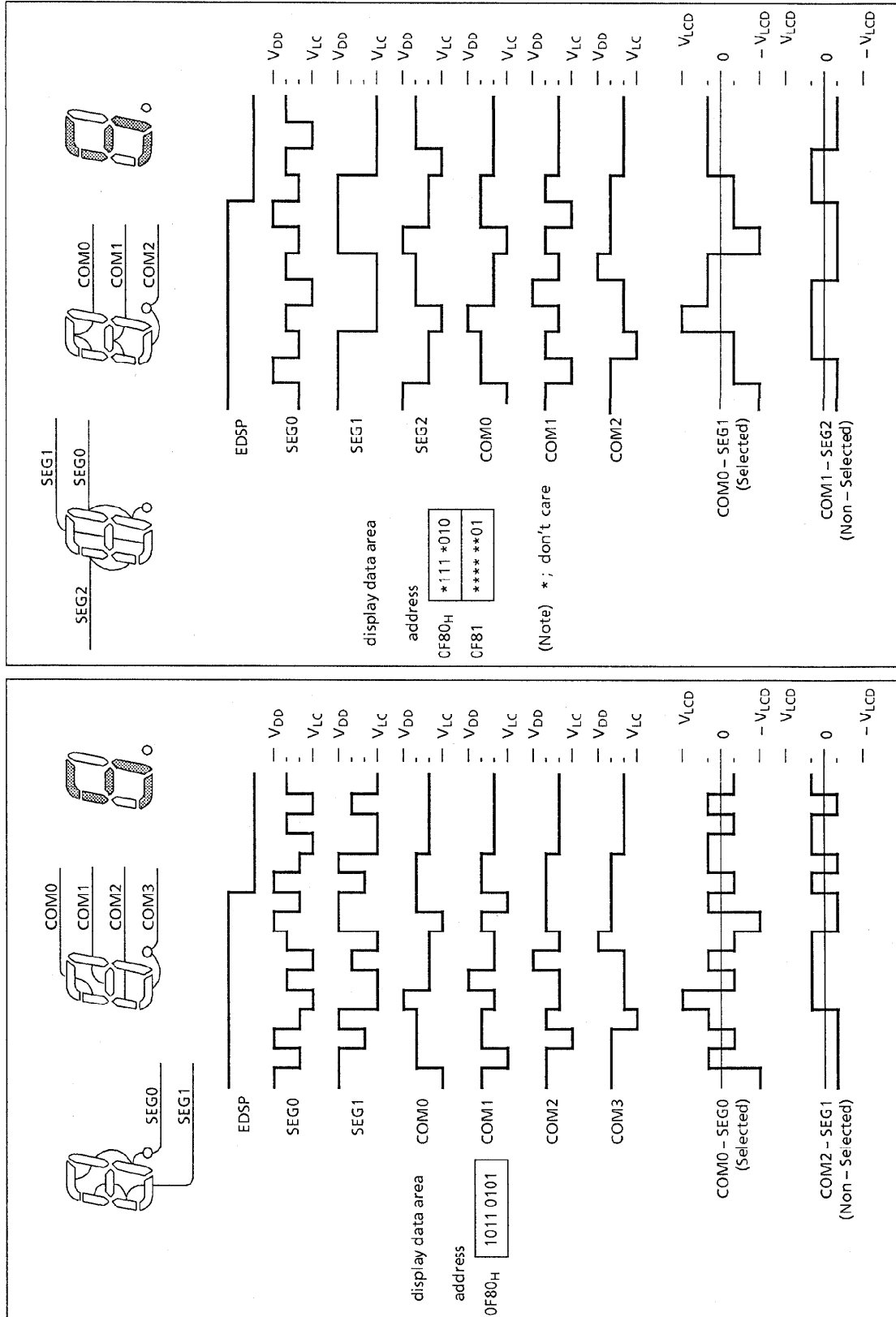


Figure 2-51. 1/3 Duty (1/3 Bias) Drive

Figure 2-50. 1/4 Duty (1/3 Bias) Drive

INPUT / OUTPUT CIRCUITRY

(1) Control pins

The input / output circuitries of the 87CC20/H20/K20A/M20A control pins are shown below.

Please specify either the single-clock mode or the dual-clock mode by a code (NM1 or NM2) as an option for an operating mode during reset.

CONTROL PIN	I/O	INPUT / OUTPUT CIRCUITRY and CODE	REMARKS			
XIN XOUT	Input Output		Resonator connecting pins (high-frequency) $R_f = 1.2M\Omega$ (typ.) $R_o = 1.5k\Omega$ (typ.)			
XTIN (P21) XTOUT (P22)	Input Output	<table border="1"> <tr> <td>NM1</td> <td rowspan="2"> </td> </tr> <tr> <td>Refer to port P2</td> </tr> </table>	NM1		Refer to port P2	Resonator connecting pins (low-frequency) $R_f = 6M\Omega$ (typ.) $R_o = 220k\Omega$ (typ.) In only dual-clock mode
NM1						
Refer to port P2						
RESET	I/O		Sink open drain output Hysteresis input Pull-up resistor $R_{IN} = 220k\Omega$ (typ.) $R = 1k\Omega$ (typ.)			
$\overline{STOP/INT5}$ (P20)	Input		Hysteresis input $R = 1k\Omega$ (typ.)			
TEST	Input		Pull-down resistor $R_{IN} = 70k\Omega$ (typ.) $R = 1k\Omega$ (typ.)			

Note1 : The TEST pin of the 87PH20/PM20 does not have a pull-down resistor.
 Note2 : The 87PH20/PM20 is placed in the single-clock mode during reset.

(2) Input / Output Ports

The input / output circuitries of the 87CC20/H20/K20A/M20A input/output ports are shown below, any one of the circuitries can be chosen by a code (A or B) as a mask option.

PORT	I/O	INPUT / OUTPUT CIRCUITRY and CODE	REMARKS
P0	I/O	<p>initial "Hi-Z"</p>	<p>Tri-state I/O</p> <p>R = 1kΩ (typ.)</p>
P1	I/O	<p>initial "Hi-Z"</p>	<p>Tri-state I/O Hysteresis input</p> <p>R = 1kΩ (typ.)</p>
P2	I/O	<p>P20</p> <p>initial "Hi-Z"</p>	<p>Sink open drain output</p> <p>R = 1kΩ</p>
		<p>P21, P22</p> <p>initial "Hi-Z"</p>	
P3	I/O	<p>initial "Hi-Z"</p>	<p>Sink open drain</p> <p>High current output (only P31, P30)</p> <p>R = 1kΩ (typ.)</p>
P4 P5	I/O	<p>A</p> <p>initial "Hi-Z"</p>	<p>Sink open drain or Push-pull output</p> <p>Hysteresis input</p> <p>R = 1kΩ (typ.)</p>
		<p>B</p> <p>initial "High"</p>	
P6	I/O	<p>initial "Hi-Z"</p> <p>Segment output</p> <p>disable</p>	<p>Sink open drain or Segment output</p> <p>R = 1kΩ (typ.)</p>

ELECTRICAL CHARACTERISTICS

(2) 87CK20A/M20A

ABSOLUTE MAXIMUM RATINGS		(V _{SS} = 0V)		
PARAMETER	SYMBOL	CONDITION	RATINGS	UNIT
Supply Voltage	V _{DD}		- 0.3 to 6.5	V
Input Voltage	V _{IN}		- 0.3 to V _{DD} + 0.3	V
Output Voltage	V _{OUT1}	Except sink open drain pin, but include P21, P22, P6 and $\overline{\text{RESET}}$	- 0.3 to V _{DD} + 0.3	V
	V _{OUT2}	Sink open drain pin except port P21, P22, P6 and $\overline{\text{RESET}}$	- 0.3 to 5.5	
Output Current (Per 1 pin)	I _{OUT1}	Ports P0, P1, P2, P3 (Except P30, P31), P4, P5, P6	3.2	mA
	I _{OUT2}	Only P30, P31	30	
Output Current (Total)	$\sum I_{OUT1}$	Ports P0, P1, P2, P3 (Except P30, P31), P4, P5, P6	120	mA
	$\sum I_{OUT2}$	Only P30, P31	60	
Power Dissipation [T _{opr} = 70°C]	PD		350	mW
Soldering Temperature (time)	T _{sld}		260 (10 s)	°C
Storage Temperature	T _{stg}		- 55 to 125	°C
Operating Temperature	T _{opr}		- 30 to 70	°C

RECOMMENDED OPERATING CONDITIONS		(V _{SS} = 0V, T _{opr} = - 30 to 70°C)					
PARAMETER	SYMBOL	PINS	CONDITIONS	Min.	Max.	UNIT	
Supply Voltage	V _{DD}		f _c = 8MHz	NORMAL1, 2 mode	4.5	5.5	V
				IDLE1, 2 mode			
			f _c = 4.2MHz	NORMAL1, 2 mode	2.7		
				IDLE1, 2 mode			
			f _s = 32.768kHz	SLOW mode	2.0		
SLEEP mode							
	STOP mode						
Input High Voltage	V _{IH1}	Except hysteresis input	V _{DD} ≥ 4.5V	V _{DD} × 0.70	V _{DD}	V	
	V _{IH2}	Hysteresis input		V _{DD} × 0.75			
	V _{IH3}		V _{DD} < 4.5V	V _{DD} × 0.90			
Input Low Voltage	V _{IL1}	Except hysteresis input	V _{DD} ≥ 4.5V	0	V _{DD} × 0.30	V	
	V _{IL2}	Hysteresis input			V _{DD} × 0.25		
	V _{IL3}		V _{DD} < 4.5V		V _{DD} × 0.10		
Clock Frequency	f _c	XIN, XOUT	V _{DD} = 4.5 to 5.5V	0.4	8.0	MHz	
			V _{DD} = 2.7 to 5.5V		4.2		
	f _s	XTIN, XTOUT		30.0	34.0	kHz	

D.C.CHARACTERISTICS

(V_{SS} = 0V, Topr = -30 to 70°C)

PARAMETER	SYMBOL	PINS	CONDITIONS	Min.	Typ.	Max.	UNIT
Hysteresis Voltage	V _{HS}	Hysteresis input		-	0.9	-	V
Input Current	I _{IN1}	TEST	V _{DD} = 5.5V V _{IN} = 5.5V / 0V	-	-	± 2	μA
	I _{IN2}	Open drain port and tri-state port					
	I _{IN3}	RESET, STOP					
Input Low Current	I _{IL}	Push-pull port	V _{DD} = 5.5V, V _{IN} = 0.4V	-	-	- 2	mA
Input Resistance	R _{IN2}	RESET		100	220	450	kΩ
Output Leakage Current	I _{LO}	Open drain port and	V _{DD} = 5.5V, V _{OUT} = 5.5V	-	-	2	μA
		tri-state port	V _{DD} = 5.5V, V _{OUT} = 5.5V/0V	-	-	± 2	
Output High Voltage	V _{OH1}	Push-pull port	V _{DD} = 4.5V, I _{OH} = -200μA	2.4	-	-	V
	V _{OH2}	Tri-state port	V _{DD} = 4.5V, I _{OH} = -0.7mA	4.1	-	-	
Output Low Voltage	V _{OL}	Except XOUT and P30, P31	V _{DD} = 4.5V, I _{OL} = 1.6mA	-	-	0.4	V
Output Low Current	I _{OL3}	Only P30, P31	V _{DD} = 4.5V, V _{OL} = 1.0V	-	20	-	mA
Supply Current in NORMAL 1, 2 mode	I _{DD}		V _{DD} = 5.5V f _c = 8MHz	-	10	16	mA
Supply Current in IDLE 1, 2 mode			f _s = 32.768kHz V _{IN} = 5.3V / 0.2V	-	4.5	6	mA
Supply Current in NORMAL 1, 2 mode			V _{DD} = 3.0V, V _{IN} = 2.8V / 0.2V f _c = 4.19MHz	-	3.5	4.5	mA
Supply Current in IDLE 1, 2 mode			f _s = 32.768kHz	-	1.5	2.0	mA
Supply Current in SLOW mode			V _{DD} = 3.0V f _s = 32.768kHz	-	30	60	μA
Supply Current in SLEEP mode			V _{IN} = 2.8V / 0.2V LCD driver is not enable	-	15	30	μA
Supply Current in STOP mode			V _{DD} = 5.5V V _{IN} = 5.3V / 0.2V	-	0.5	10	μA
Segment Output Low Resistance			R _{OS1}	SEG31-SEG0	V _{DD} = 5V V _{DD} - V _{LC} = 3V	-	20
Common Output Low Resistance	R _{OC1}	COM3-COM0					
Segment Output High Resistance	R _{OS2}	SEG31-SEG0					kΩ
Common Output High Resistance	R _{OC2}	COM3-COM0					
Segment /Common Output Voltage	V _{O 2/3}	SEG31-SEG0 and COM3-COM0 pins		3.8	4.0	4.2	V
	V _{O 1/2}			3.3	3.5	3.7	
	V _{O 1/3}			2.8	3.0	3.2	

Note 1 : Typical values show those at Topr = 25°C, V_{DD} = 5V.

Note 2 : Input Current ; The current through pull-up or pull-down resistor is not included.

Note 3 : Output resistance R_{OS} and R_{OC} indicate "on" when switching levels.

Note 4 : V_{O2/3} indicates an output current at the 2/3 level when operating in the 1/4 or 1/3 duty mode.

Note 5 : V_{O1/2} indicates an output current at the 1/2 level when operating in the 1/2 duty or static mode.

Note 6 : V_{O1/3} indicates an output current at the 1/3 level when operating in the 1/4 or 1/3 duty mode.

Note 7 : When you use a liquid crystal display (LCD), it is necessary to give careful consideration to the value of the output resistor R_{OS 1/2}, R_{OC 1/2}.

Note 8 : R_{OS1}, R_{OC1} : On time of the lower output resistor is 2⁶/f_c, 2/f_s [s]

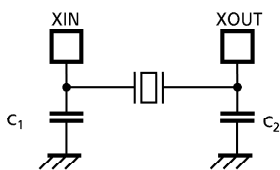
Note 9 : R_{OS2}, R_{OC2} : On time of the higher output resistor is 1/(n/f_F). (1/n duty, f_F : frame frequency)

A.C. CHARACTERISTICS ($V_{SS} = 0V, V_{DD} = 4.5 \text{ to } 5.5V, T_{opr} = -30 \text{ to } 70^\circ\text{C}$)

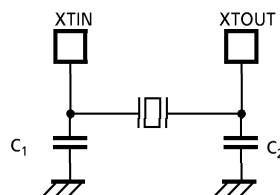
PARAMETER	SYMBOL	CONDITIONS	Min.	Typ.	Max.	UNIT
Machine Cycle Time	t_{cy}	In NORMAL 1, 2 mode	0.5	—	10	μs
		In IDLE 1, 2 mode				
		In SLOW mode	117.6	—	133.3	
		In SLEEP mode				
High Level Clock Pulse Width	t_{WCH}	For external clock operation	62.5	—	—	ns
Low Level Clock Pulse Width	t_{WCL}	(XIN input), $f_c = 8\text{MHz}$				
High Level Clock Pulse Width	t_{WSH}	For external clock operation	14.7	—	—	μs
Low Level Clock Pulse Width	t_{WSL}	(XTIN input), $f_s = 32.768\text{kHz}$				
Frequency of TC1 input	f_{TC1}	Frequency Measurement mode	—	—	fc	MHz

RECOMMENDED OSCILLATING CONDITION ($V_{SS} = 0V, V_{DD} = 4.5 \text{ to } 5.5V, T_{opr} = -30 \text{ to } 70^\circ\text{C}$)

PARAMETER	OSCILLATOR	FREQUENCY	RECOMMENDED OSCILLATOR		RECOMMENDED CONDITION	
					C ₁	C ₂
High-frequency	Ceramic Resonator	8MHz	KYOCERA	KBR8.0M	30pF	30pF
		4MHz	KYOCERA	KBR4.0MS		
	Crystal Oscillator	8MHz	TOYOCOM	210B 8.0000	20pF	20pF
		4MHz	TOYOCOM	204B 4.0000		
Low-frequency	Crystal Oscillator	32.768kHz	NDK	MX-38T	15pF	15pF



(1) High-frequency



(2) Low-frequency

Note : An electrical shield by metal shield plate on the surface of 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.

