CMOS 4-BIT MICROCONTROLLER

# TMP47C103N, TMP47C203N
# TMP47C103M, TMP47C203M

The 47C103/203 are high speed and high performance 4-bit single chip micro computers, integrating ROM, RAM, input/output ports and timer/counters on a ship.
The 47C103/203 are srandard LSI in the TLCS-47E series.
In addition, they have 8 bit SIO, watchdog timer and the output port with LED direct drive capabilty.

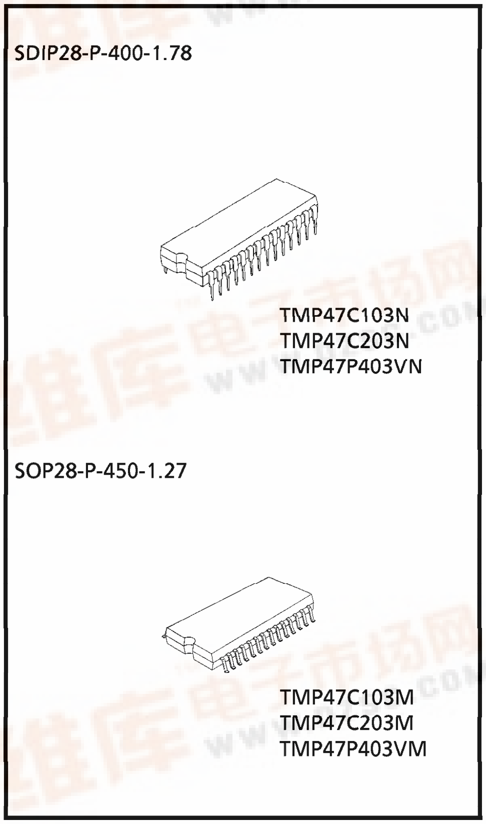
| PART No. | ROM | RAM | PACKAGE | OTP version |
|---|---|---|---|---|
| TMP47C103N | 1024 × 8-bit | 64 × 4-bit | SDIP28-P-400-1.78 | TMP47P403VN |
| TMP47C103M | | | SOP28-P-450-1.27 | TMP47P403VM |
| TMP47C203N | 2048 × 8-bit | 128 × 4-bit | SDIP28-P-400-1.78 | TMP47P403VN |
| TMP47C203M | | | SOP28-P-450-1.27 | TMP47P403VM |

## FEATURES

◆4-bit single chip microcomputer
◆Instruction execution time : 1.3 μs (at 6 MHz)
◆Low voltage operation : 2.2 V (at 2 MHz RC)
◆90 basic instructions
- ROM table look-up instructions
- 5-bit to 8-bit data conversion instruction

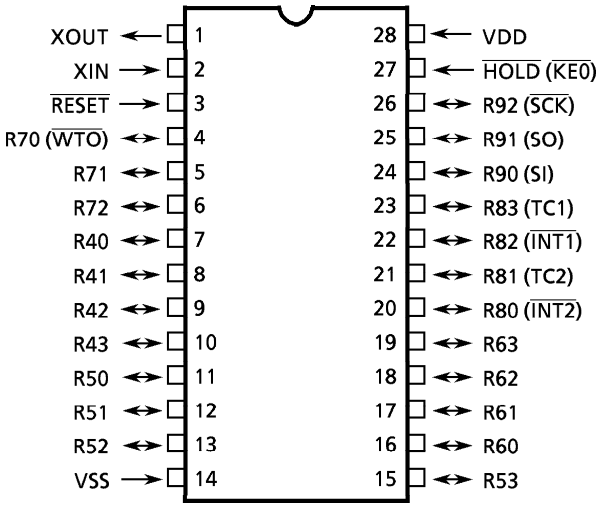
◆Subroutine nesting : 15 levels max.
◆6 interrupt sources (External : 2, Internal : 4)
All sources have independent latches each, and multiple interrupt control is available
◆I/O port (23 pins)
◆Two 12-bit Timer / Counters
Timer, event counter, and pulse width measurement mode
◆Interval Timer
◆Watchdog Timer
◆Serial Interface with 8-bit buffer
- Simultaneous transmission and reception capability
- 4/8-bit transfer, external / internal clock, and leading/trailing edge shift mode

◆High current outputs
LED direct drive capability : typ. 20 mA × 8 bits (port R5, R6)
typ. 7 mA × 4 bits (port R4)
◆Hold function
Battery / Capacitor back-up
◆Real Time Emulator : BM47C203

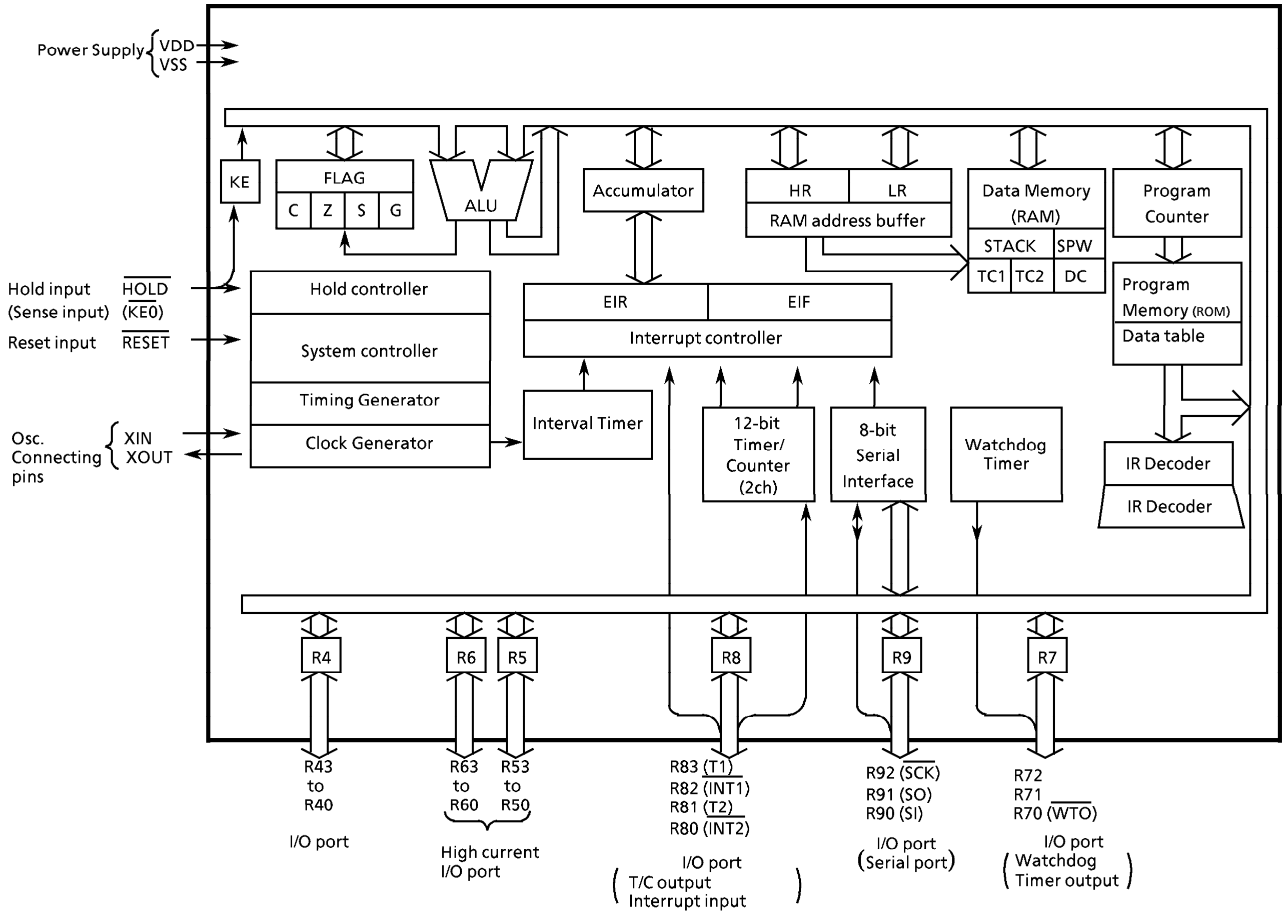


## PIN ASSIGNMENT (TOP VIEW)

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



## BLOCK DIAGRAM



## PIN FUNCTION

| PIN NAME | Input/Output | FUNCTIONS | |
|---|---|---|---|
| R43 to R40 | I/O | 4-bit I/O port with latch (R7 port has only 3-bit).<br>When used as input port, the latch must be set to “1”.<br>Every bit data is possible to be set, cleared and tested by the bit manipulation instruction of the L-register indirect addressing. | |
| R53 to R50 | | | 8-bit data are output by the 5-bit to 8-bit data conversion instruction [OUTB @HL] . |
| R63 to R60 | | | |
| R72 to R71 | I/O | | |
| R70 ($\overline{\text{WTO}}$) | I/O(Output) | | Watchdog timer output |
| R83 (T1) | I/O(Input) | 4-bit I/O port with latch.<br>When used as input port, external interrupt input pin, or timer/counter external input pin, the latch must be set to “1”. | Timer / C ounter 1 external input |
| R82 ($\overline{\text{INT1}}$) | | | External interrupt 1 input |
| R81 (T2) | | | Timer / Counter 2 external input |
| R80 ($\overline{\text{INT2}}$) | | | External interrupt 2 input |
| R92 ($\overline{\text{SCK}}$) | I/O(I/O) | 3-bit I/O port with latch.<br>When used as input port or serial port, the latch must be set to “1”. | Serial clock I/O |
| R91 (SO) | I/O(Output) | | Serial data output |
| R90 (SI) | I/O(Input) | | Serial data input |
| XIN | Input | Resonator connecting pins.<br>For inputting external clock, XIN is used and XOUT is opened. | |
| XOUT | Output | | |
| $\overline{\text{RESET}}$ | Input | Reset signal input | |
| $\overline{\text{HOLD}}$ ($\overline{\text{KE0}}$) | Input(Input) | Hold request / release signal input | Sense input |
| VDD | Power Supply | + 5 V | |
| VSS | | 0 V (GND) | |

## OPERATIONAL DESCRIPTION

Concerning the 47C103/203 the configuration and functions of hardwares are described. The basic instruction of configuration in the 47C103/203 is the same those of TLCS-47 serise.

## 1. SYSTEM CONFIGURATION

◆INTERNAL CPU FUNCTION

- 2.1 Program Counter (PC)
- 2.2 Program Memory (ROM)
- 2.3 H Register, L Register
- 2.4 Data Memory (RAM)
  - a. Stack
  - b. Stack Pointer Word (SPW)
  - c. Data Counter (DC)
- 2.5 ALU, Accumulator
- 2.6 Flags
- 2.7 Clock Generator and Timing Generator
- 2.8 INTERRUPT FUNCTION
- 2.9 RESET FUNCTION

◆PERIPHERAL HARDWARE FUNCTION

- 3.1 I/O Ports
- 3.2 Interval Timer
- 3.3 Timer / Counters (TC1, TC2)
- 3.4 Watchdog Timer
- 3.5 Serial Interface

## 2. INTERNAL CPU FUNCTION

### 2.1 Program Counter (PC)

The program counter is a 11-bit binary counter which indicates the address of the program memory storing the next instruction to be executed. Normally, the PC is incremented by the number of bytes of the instruction every time it is fetched. When a branch instruction or a subroutine instruction has been executed or an interrupt has been accepted, the specified values listed in Table 2-1 are set to the PC. The PC is initialized to "0" during reset.

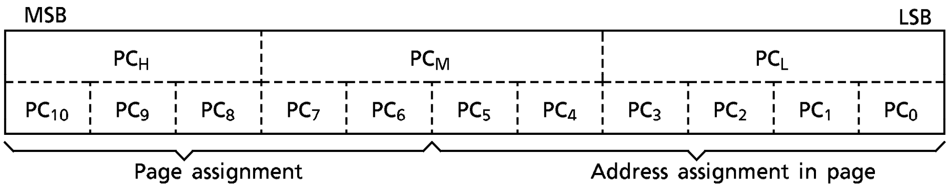


Figure 2-1. Configuration of Program Counter

The PC can directly address a 2048-byte address space. However, with the short branch and subroutine call instructions, the following points must be considered.

(1) Short branch instruction [BSS a]

In [BSS a] instruction execution, when the branch condition is satisfied, the value specified in the instruction is set to the lower 6 bits of the PC. That is, [BSS a] becomes the in-page branch instruction. When [BSS a] is stored at the last address of the page, the upper 6 bits of the PC point the next page, so that branch is made to the next page.

Table 2-1. Status Change of Program Counter

| Instruction or Operation | | Condition | | PC10 | PC9 | PC8 | PC7 | PC6 | PC5 | PC4 | PC3 | PC2 | PC1 | PC0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Execution of Instruction | BS a | SF = 1 (Branch condition is satisfied) | | Immediate data specified by the instruction | | | | | | | | | | |
| | | SF = 0 (Branch condition is not satisfied) | | + 2 | | | | | | | | | | |
| | BSS a | SF = 1 | Lower 6-bit address ≠ 111111 | Hold | | | | | Immediate data specified by the instruction | | | | | |
| | | | Lower 6-bit address = 111111 (last address in page) | + 1 | | | | | Immediate data specified by the instruction | | | | | |
| | | SF = 0 | | + 1 | | | | | | | | | | |
| | CALL a | | | Immediate data specified by the instruction | | | | | | | | | | |
| | CALLS a | | | 0 | 0 | 0 | The data generated by the immediate data specified by the instruction | | | | | 1 | 1 | 0 |
| | RET | | | The return address restored from stack | | | | | | | | | | |
| | RETI | | | The return address restored from stack | | | | | | | | | | |
| | Others | | | Incremented by the number of bytes in the instruction | | | | | | | | | | |
| Interrupt acceptance | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Interrupt vector | | | 0 |
| Reset | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 2.2 Program Memory (ROM)

Programs and fixed data are stored in the program memory. The instruction to be executed next is read from the address indicated by the contents of the PC.

The fixed data can be need by using the table look-up instructions on 5-bit to 8-bit data conversion instruction.

(1) Table look-up instructions

[LDL A, @DC], [LDH A, @DC + ]

The table look-up instructions read the lower and upper 4 bits of the fixed data stored at the address specified in the data counter (DC) to place them into the accumulator. [LDL A, @DC] instruction reads the lower 4 bits of fixed data, and [LDH A, @DC + ] instruction reads the upper 4 bits.

The DC is a 12-bit register, allowing it to address the entire program memory space.

In this case, the upper bit of the DC (MSB) is ignared.

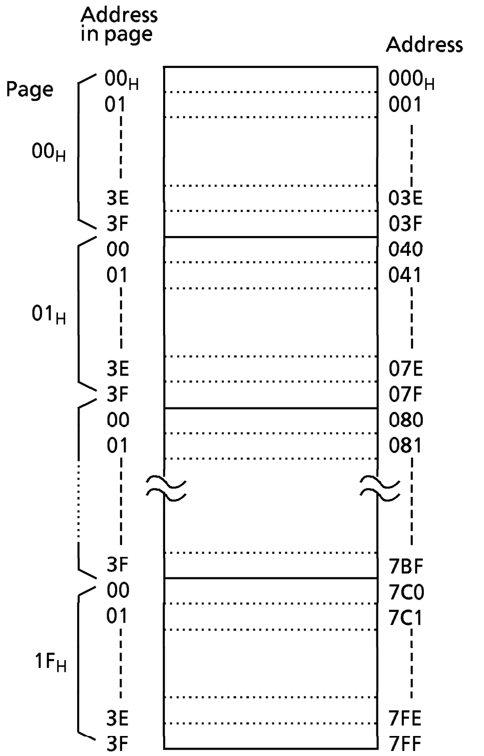


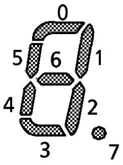
Figure 2-2. Configuration of Program Memory

(2) 5-bit to 8-bit data conversion instruction [OUTB @HL]

The 5-bit to 8-bit data conversion instruction reads the fixed data (8 bits) from the data conversion table in the program memory to output the upper 4 bits to port P6 and the lower 4 bits to port P5. The table is located in the last 32-byte space (addresses,$7E0_H$ through $7FF_H$ for the 47C203, $3E0_H$ through $3FF_H$ for the 47C103) in the program memory with the lower address consisting of the 5 bits obtained by concatenating the contents of the data memory specified by the HL register pair and the content of the carry flag. This instruction is usable for such applications as converting BCD data into an output code to the 7-segment display elements.

Example: The following shows that the BCD data at address $2F_H$ in the data memory is converted into the 7-segment code (e.g., anode common LED) to be output to ports P6 and P5.

```
LD        HL, #2FH ;  HL←2FH (Data memory address is set)
TEST      CF       ;  CF←0 (The table is specified at addresses 7E0H - 7EFH)
OUTB      @HL      ;  Ports P6, P5←fixed data
   ⋮
ORG       07E0H    ;  Data conversion table
DATA      0C0H, 0F9H, 0A4H, 0B0H, 99H, 92H, 82H, 0D8H, 80H, 98H
```



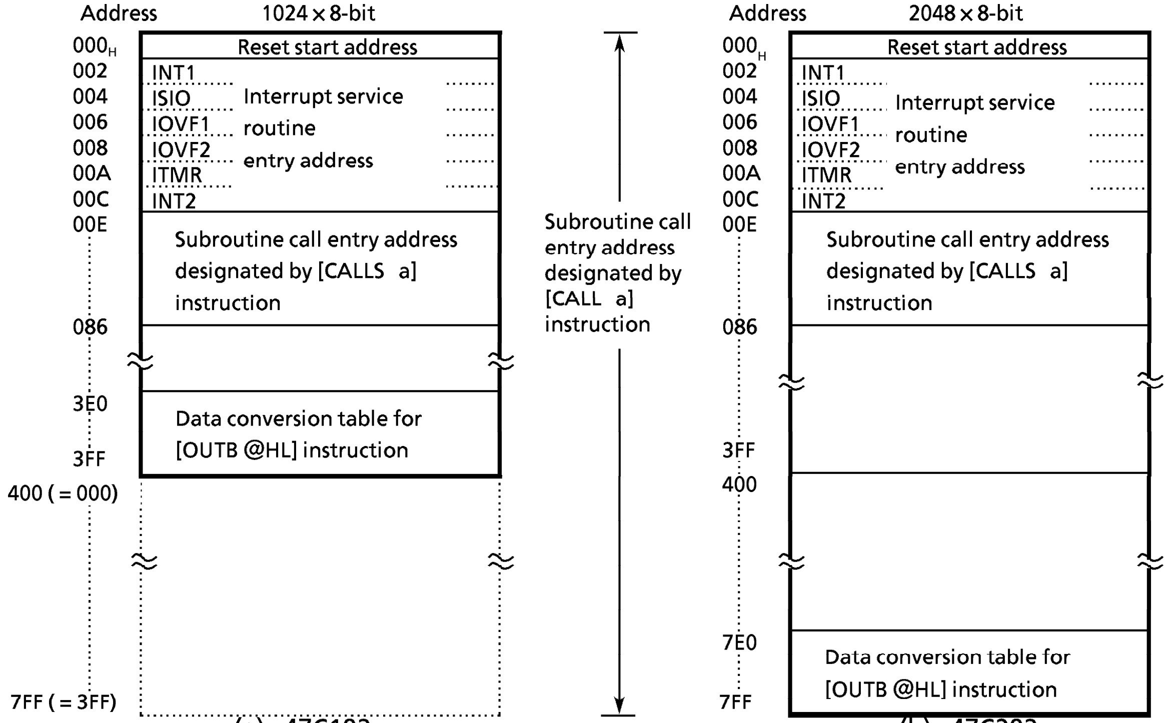
## 2.2.1 Program Memory Map

The 47C103 has 1024 × 8 bits (addresses $000_H$ through $3FF_H$) of program memory (mask ROM), the 47C203 has 2048 × 8 bits (addresses $000_H$ through $7FF_H$).

Figure 2-3 shows the program memory map. Address $000_H$ to $086_H$ and $7E0_H$ to $7FF_H$ ($3E0_H$ to $3FF_H$ for the 47C103) of the program memory are also used for special purposes.

## 2.2.2 Program Memory Capacity

On the 47C103, no physical program memory exists in the address range $400_H$ through $7FF_H$. However, if this space is accessed by program, the most significant bit of each address is always regarded as "0" and the contents of the program memory corresponding to the address $000_H$ through $3FF_H$ are read.


Address
1024 × 8-bit
000H
002
004
006
008
00A
00C
00E
086
3E0
3FF
400 ( = 000)
7FF ( = 3FF)
Reset start address
INT1
ISIO
IOVF1
IOVF2
ITMR
INT2
Interrupt service routine entry address
Subroutine call entry address designated by [CALLS a] instruction
Data conversion table for [OUTB @HL] instruction
Subroutine call entry address designated by [CALL a] instruction
Address
2048 × 8-bit
000H
002
004
006
008
00A
00C
00E
086
3FF
400
7E0
7FF
Reset start address
INT1
ISIO
IOVF1
IOVF2
ITMR
INT2
Interrupt service routine entry address
Subroutine call entry address designated by [CALLS a] instruction
Data conversion table for [OUTB @HL] instruction

(a) 47C103 (b) 47C203

*Note. It is necessary to set two data conversion tables to check operation of 47C103/203 using 47P403V.* For details, see the technical documents of 47P403V.

Figure 2-3. Program Memory Map

## 2.3 H Register and L Register

The H register and the L register are 4-bit general registers. They are also used as a register pair (HL) for the data memory (RAM) addressing pointer. The RAM consists of pages, each page being 16 words long (1 word = 4 bits). The H register specifies a page and the L register specifies an address in the page.

The L register has the auto-post-increment / decrement capability, implementing the execution of composite instructions. For example, [ST A, @HL + ] instruction automatically increments the contents of the L register after data transfer.

During the execution of [SET @L], [CLR @L], or [TEST @L] instructions, the L register is also used to specify the bits corresponding to I/O port pins R72 through R40 (the indirect addressing of port bits by the L register).

Example 1: To write immediate values "5" and "$F_H$" to data memory addresses $10_H$ and $11_H$.

```
LD      HL,#10H          ;  HL←10H
ST      #5,@HL+          ;  RAM [10H] ←5H, LR←LR + 1
ST      #0FH,@HL+        ;  RAM [11H] ←FH, LR←LR + 1
```

Example 2: The output latch of R51 pin set "1" by the 2 register indirect addressing bit manipulation instruction.

```
LD      L,#0101B         ;  Set R51 pin address to L register
SET     @L               ;  R51←1
```

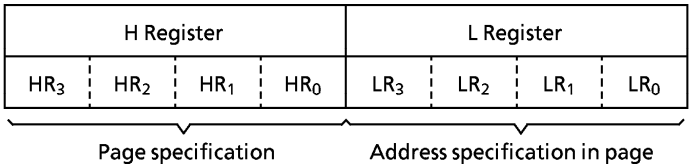


Figure 2-4. Configuration of H and L Registers

## 2.4 Data Memory (RAM)

The data memory stores usen-processed data. One page of this memory is 16 words long (1 word = 4 bits). It has 8 pages. The data memory is addressed in one of three ways (addressing modes).

The RAM is addressed in one of the three ways (addressing modes):

(1) Register-indirect addressing mode

In this mode, a page is specified by the H register and an address in the page by the L register.

Example: LD A, @HL ; Acc←RAM [HL]

(2) Direct addressing mode

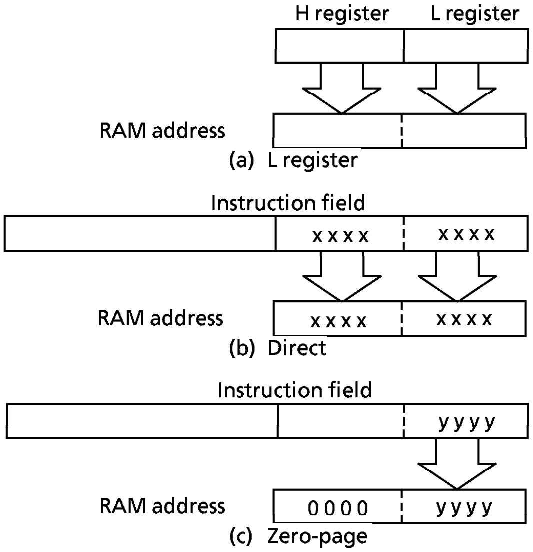
In this mode, an address is directly specified by the 8 bits of the second byte (operand) in the instruction field.

Example: LD A, 2CH ; Acc←RAM [$2C_H$]

(3) Zero-page addressing mode

In this mode, an address in zero-page (addresses $00_H$ through $0F_H$) is specified by the lower 4 bits of the second byte (operand) in the instruction field.

Example: ST #3, 05H ; RAM [$05_H$] ←3



(a) L register

(b) Direct

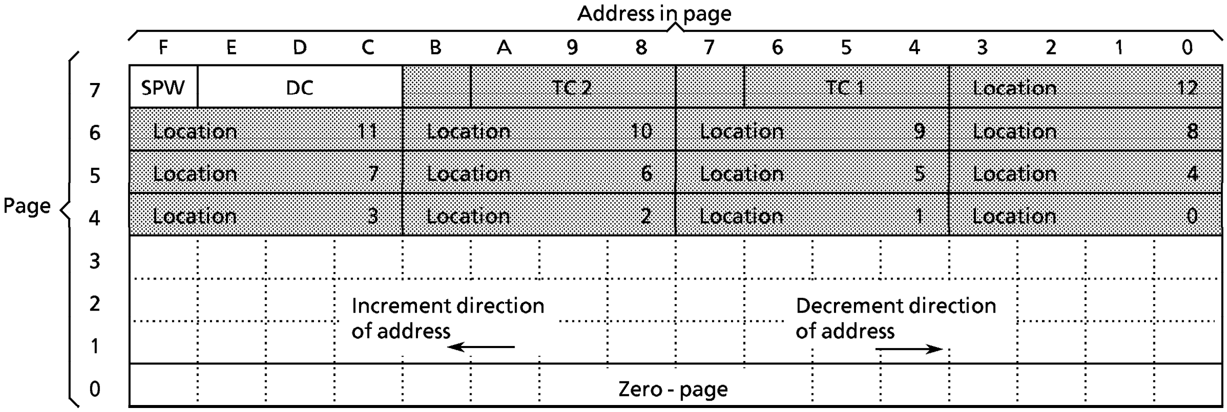
(c) Zero-page

Figure 2-5. Addressing mode

### 2.4.1 Data Memory Map

Figure 2-7 shows the data memory map. The data memory is also used for the following special purpose.

① Stack and Stack Pointer Word (SPW)
② Data Counter (DC)
③ Count registers of the timer/counters (TC1, TC2)
④ Zero-page



*Note1.* ▨ *denotes the stack area.*
*Note2. The TC1 and TC2 areas are shared by the locations 13 and 14.*

Figure 2-6. Data Memory Map (47C203)

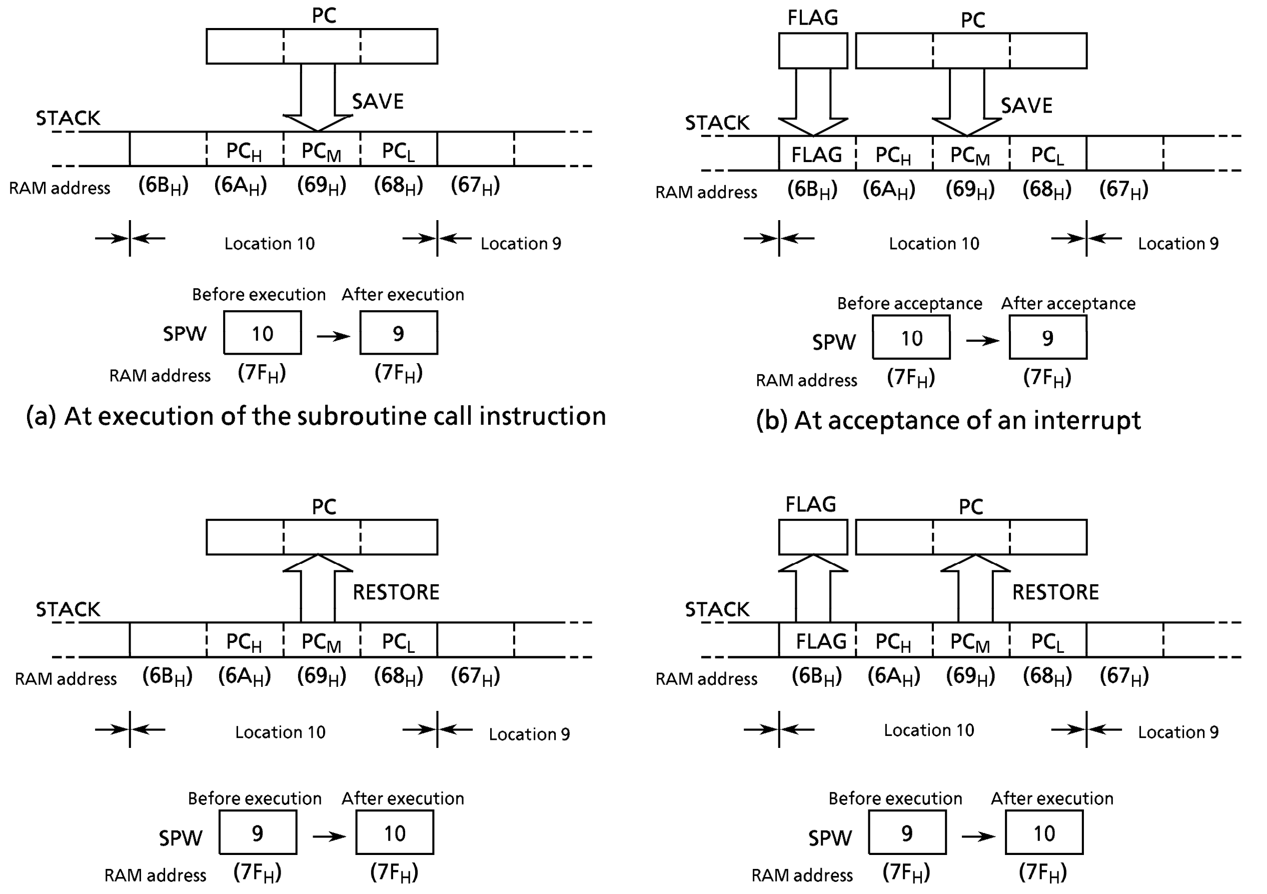
(1) Stack

The stack provides the area in which the return address is saved before a jump is performed to the processing routine at the execution of a subroutine call instruction or the acceptance of an interrupt. When a subroutine call instruction is executed, the contents (the return address) of the program counter are saved; when an interrupt is accepted, the contents of the program counter and flags are saved.

When returning from the processing routine, executing the subroutine return instruction [RET] restores the contents of the program counter from the stack; executing the interrupt return instruction [RETI] restores the contents of the program counter and flags.

The stack consists of up to 15 levels (locations 0 through 14) which are provided in the data memory (addresses $40_H$ through $7B_H$). Each location consists of 4-word data memory. Locations 13 and 14 are shared with the count registers of the timer / counters (TC1, TC2) to be described later.

The save / restore locations in the stack are determined by the stack pointer word (SPW). The SPW is automatically decremented after save, and incremented before restore. That is, the value of the SPW indicates the stack location number for the next save.



(a) At execution of the subroutine call instruction

(b) At acceptance of an interrupt

(c) At execution of the subroutine return instruction

(d) At execution of the interrupt return instruction

Figure 2-7. Accessing Stack (Save / Restore at the 47C203)

(2) Stack Pointer Word (SPW)

Address $7F_H$ ($3F_H$ for the 47C103) in the data memory is called the stack pointer word, which identifies the location in the stack to be accessed (save or restore).

Generally, location number 0 to 12 can be set to the SPW, providing up to 13 levels of stack nesting. Locations 13 and 14 are shared with the timer / counters to be described later; therefore, when the timer / counters are not used, the stack area of up to 15 levels is available. Address $7F_H$ is assigned to the SPW, so that the contents of the SPW cannot be set "15" in any case.

The SPW is automatically updated when a subroutine call is executed or an interrupt is accepted. However, if it is used in excess of the stack area permitted by the data memory allocating configuration, the user-processed data may be lost. (For example, when the user-processed data area is in an address range $00_H$ through $4F_H$, up to location 4 of the stacks are usable. If an interrupt is accepted with location 4 already used, the user-processed data stored in addresses $4C_H$ through $4F_H$ corresponding to the location 3 area is lost.)

The SPW is not initialized by hardware, requiring to write the initial value (the location with which the use of the stack starts) by using the initialization routine. Normally, the initial value of "12" is used.

Example: To initialize the SPW (when the stack is used from location 12)

```
        LD          A,#12      ;  SPW←12
        ST          A,0FFH
```

(3) Data Counter (DC)

The data counter is a 12-bit register to specify the address of the data table to be referenced in the program memory (ROM). Data table reference is performed by the table look-up instructions [LDL A, @DC] and [LDH A, @DC+] . The data table may be located anywhere within the program memory address space.
The DC is assigned with a RAM address in unit of 4 bits. Therefore, the RAM manipulation instruction is used to set the initial value or read the contents of the DC.

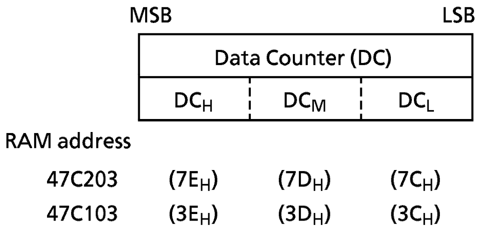


Figure 2-8. Data Counter

Example: To set the DC to $380_H$.

```
LD          HL,#07CH    ;  Sets RAM address of DCL to HL register pair.
ST          #0H,@HL+    ;  DC ← 380H
ST          #8H,@HL+
ST          #3H,@HL+
```

(4) Count registers of the timer / counters (TC1, TC2)

The 47C103/203 has two channels of 12-bit timer / counters. The count register of the timer / counter is assigned with a RAM addresses in unit of 4 bits, so that the initial value is set and the contents are read by using the RAM manipulation instruction.
The count registers are shared with the stack area (locations 13 and 14) described earlier, so that the stack is usable from location 13 when the timer / counter 1 is not used. When none of timer / counter 1 and timer / counter 2 are used, the stack is usable from location 14.
When both timer / counter 1 and timer / counter 2 are used, the data memory locations at addresses $77_H$ and $7B_H$ ($37_H$ and $3B_H$ for the 47C103) can be used to store the user-processed data.

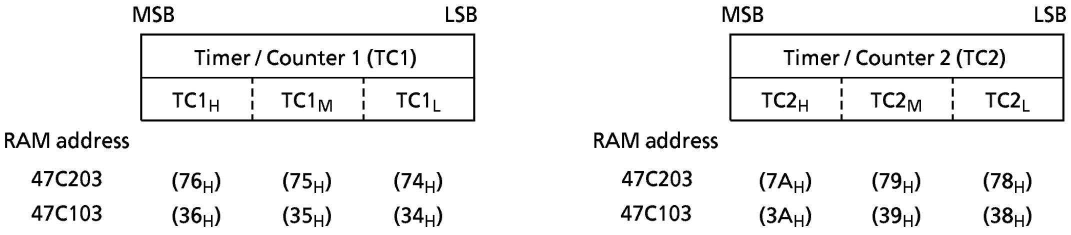


Figure 2-9. Count Registers of the Timer / Counters (TC1, TC2)

(5) Zero-page

The 16 words (at addresses $00_H$ through $0F_H$) of the zero page of the data memory can be used as the user flags or pointers by using zero-page addressing mode instructions (comparison, addition, transfer, and bit manipulation), providing enhanced efficiency in programming.

Example: To write immediate data "8" to address $09_H$ if bit 2 at address $04_H$ in the RAM is "1".

```
        TEST        04H,2       ;  Skips if bit 2 at address 04H in the RAM is "0".
        B           SKIP
        ST          #8,09H      ;  Writes "8" to address 09H in the RAM
   SKIP:
```

*Note : In the 47C103, Zero-page and Stack area (locations 0 through 3) are overlapped.*

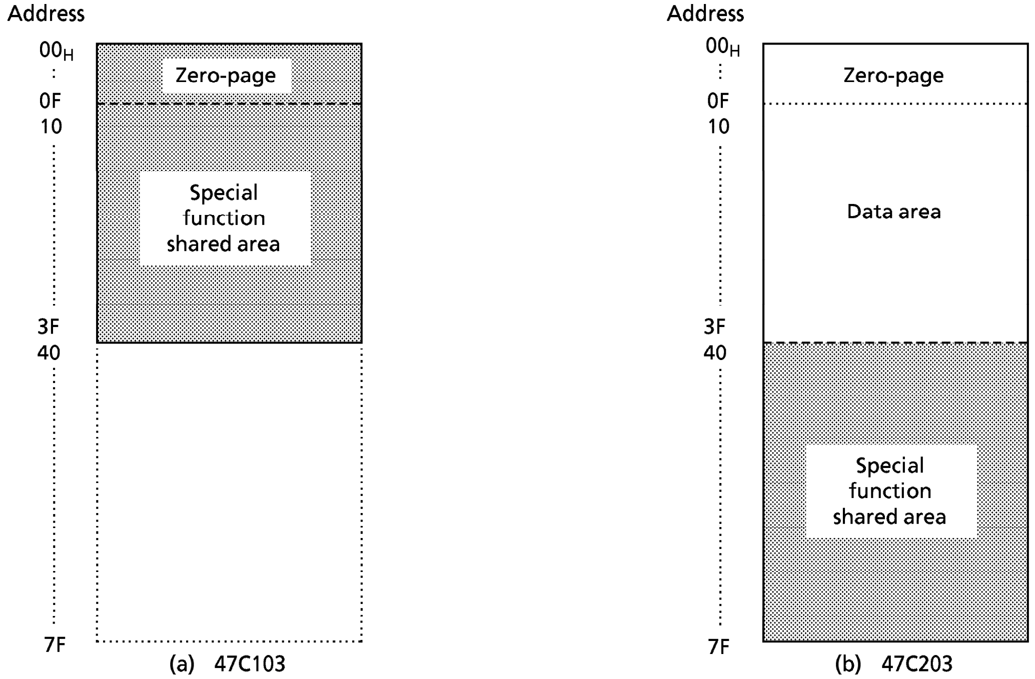
## 2.4.2 Data Memory (RAM)

The 47C103 has 64 × 4 bits (addresses $00_H$ through $3F_H$) of the data memory (RAM), and the 47C203 has 128 × 4 bits (addresses $00_H$ through $7F_H$).

When power-on is performed, the contents of the RAM become unpredictable, so that they must be initialized by the initialization routine.

Example: To clear RAM (use common to the 47C103 and 47C203)

```
          LD     HL, #00H     ; HL←00H
SCLRRAM:  ST     #0, @HL+     ; RAM [HL] ←0, LR←LR + 1
          B      SCLRRAM
          ADD    H, #1        ; HR←HR + 1
          B      SCLRRAM
```



(a) 47C103 (b) 47C203

*Note: In the **47C103**, the zero-page and thespecial function shared area (stack location 3 to 0) are overlapped. At programming, note that addresses 10 to 3FH are assigned to address 50 to 1FH in the **47C103**. The technical data sheets for the **47P403V** shall also be referred to.*

Figure 2-10. Data Memory Capacity and Address Assignment

## 2.5 ALU and Accumulator

### 2.5.1 Arithmetic / Logic Unit (ALU)

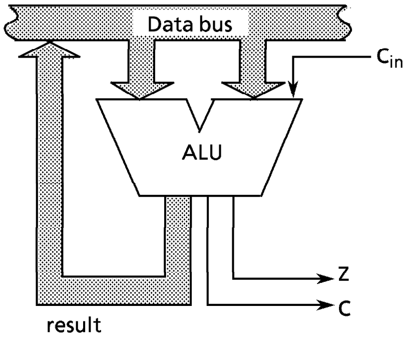
The ALU performs the arithmetic and logic operations specified by instructions on 4-bit binary data and outputs the result of the operation, the carry information (C), and the zero detect information (Z).

(1) Carry information (C)

The carry information indicates a carry-out from the most significant bit in an addition. A subtraction is performed as addition of two's complement, so that, with a subtraction, the carry information indicates that there is no borrow to the most significant bit. With a rotate instruction, the information indicates the data to be shifted out from the accumulator.

(2) Zero detect information (Z)

This information is "1" when the operation result or the data to be transferred to the accumulator / data memory is "$0000_B$".



*Note. $C_{in}$ indicates the carry input specified by instruction*

Figure 2-11. ALU

Example: The carry information (C) and zero detect information (Z) for 4-bit additions and subtractions.

| Operation | Result | C | Z |
|---|---|---|---|
| 4 + 2 = | 6 | 0 | 0 |
| 7 + 9 = | 0 | 1 | 1 |
| 8 − 1 = | 7 | 1 | 0 |
| 2 − 2 = | 0 | 1 | 1 |
| 5 − 8 = | −3 ($1101_B$) | 0 | 0 |

### 2.5.2 Accumulator (Acc)

The accumulator is a 4-bit register used to hold source data or results of the operations and data manipulations.

MSB LSB

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| | | | |

Figure 2-12. Accumulator

## 2.6 Flags

There are a carry flag (CF), a zero flag (ZF), a status flag (SF), and a general flag (GF), each consisting of 1 bit. These flags are set or cleared according to the condition specified by an instruction. When an interrupt is accepted, the flags are saved on the stack along with the program counter. When the [RETI] instruction is executed, the flags are restored from the stack to the states set before interrupt acceptance.

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| CF | ZF | SF | GF |

Figure 2-13. Flags

(1) Carry flag (CF)

The carry flag holds the carry information received from the ALU at the execution of an addition/subtraction with carry instruction, a compare instruction, or a rotate instruction. With a carry flag test instruction, the CF holds the value specified by it.

① Addition/subtraction with carry instructions [ADDC A, @HL], [SUBRC A, @HL]
The CF becomes the input ($C_{in}$) to the ALU to hold the carry information.

② Compare instructions [CMPR A, @HL], [CMPR A, #k]
The CF holds the carry information (non-borrow).

③ Rotate instructions [ROLC A], [RORC A]
The CF is shifted into the accumulator to hold the carry information (the data shifted out from the accumulator).

④ Carry flag test instructions [TESTP CF], [TEST CF]
With [TESTP CF] instruction, the content of the CF is transferred to the SF then the CF is set to "1".
With [TEST CF] instruction, the value obtained by inverting the content of the CF is transferred to the SF then the CF is cleared to "0".

(2) Zero flag (ZF)
The zero flag holds the zero detect information (Z) received from the ALU at the execution of an operational instruction, a rotate instruction, an input instrcution, or a transfer-to-accumulator instruction.

(3) Status flag (SF)
The status flag provides the branch condition for a branch instruction. Branch is performed when this flag is set to "1". Normally the SF is set to "1", so that any branch instruction can be regarded as an unconditional branch instruction. When a branch instruction is executed upon set or clear of the SF according to the condition specified by an instruction, this instruction becomes a conditional branch instruction. During reset, the SF is initialized to "1", other flags are not affected.

(4) General flag (GF)
This is a 1-bit general-purpose flag which can be set, cleared, or tested by program.

Example 1: When the following instructions are executed with the accumulator, H register, L register, data memory (address $07_H$), and carry flag being set to "$C_H$", "0", "7", "5", and "1" respectively, the contents of the accumulator and flags become as follows:

| Instruction | Acc after execution | Flag after execution | | |
|---|---|---|---|---|
| | | CF | ZF | SF |
| ADDC A, @HL | $2_H$ | 1 | 0 | 0 |
| SUBRC A, @HL | $9_H$ | 0 | 0 | 0 |
| CMPR A, @HL | $C_H$ | 0 | 0 | 1 |
| AND A, @HL | $4_H$ | 1 | 0 | 1 |
| LD A, @HL | $5_H$ | 1 | 0 | 1 |

| Instruction | Acc after execution | Flag after execution | | |
|---|---|---|---|---|
| | | CF | ZF | SF |
| LD A, #0 | $0_H$ | 1 | 1 | 1 |
| ADD A, #4 | $0_H$ | 1 | 1 | 0 |
| DEC A | $B_H$ | 1 | 0 | 1 |
| ROLC A | $9_H$ | 1 | 0 | 0 |
| RORC A | $E_H$ | 0 | 0 | 1 |

Example 2: When the accumulator (Acc) is 0≦Acc≦9, the general flag (GF) is set to "1".

```
        CLR     GF          ; GF←0
        CMPR    A, #9       ; Skip if Acc ≧ 9.
        TEST    CF
        B       SKIPC
        SET     GF          ; GF←1
SKIPC:
```

## 2.7 Clock Generator and Timing Generator

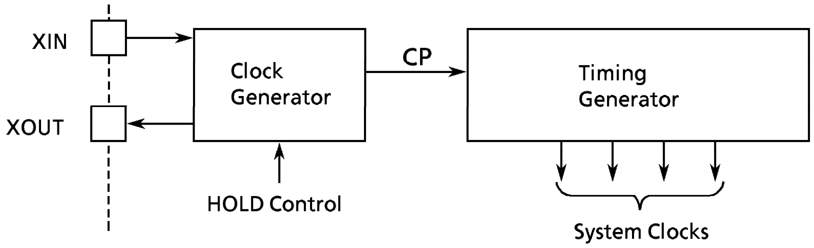
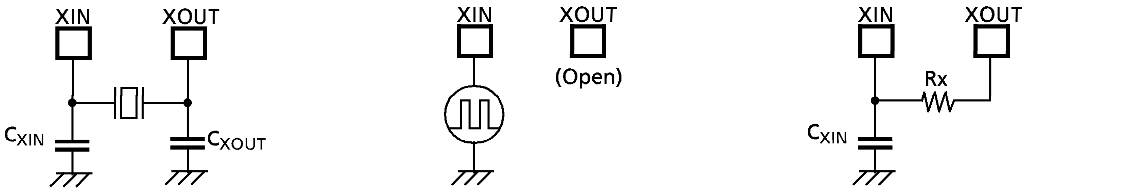


Figure 2-14. Clock Generator and Timing Generator

### 2.7.1 Clock Generator

The clock generator provides the basic clock pulse (CP) by which the system clock to be supplied to the CPU and the peripheral hardware is produced. The CP can be easily obtained by connecting the resonator to the XIN and XOUT pins. (RC oscillation is also possible, depending on the mask option) The clock from the external oscillator is also available. In the hold operating mode, the clock generator stops oscillating.



(a) Crystal or ceramic resonator (b) External oscillator (c) RC oscillation (mask option)

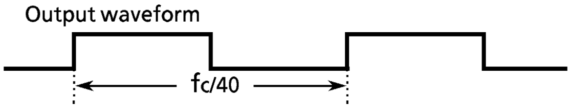
Figure 2-15. Examples of Oscillator Connection

*Note: Accurate adjustment of the oscillation frequency*

*Although the hardware to externally and directly monitor the CP is not provided, the oscillation frequency can be adjusted by making the program to output the pulse with a fixed frequency to the port with the all interrupts disabled and timer/counters stopped and monitoring this pulse. With a system requiring the oscillation frequency adjustment, the adjusting program must be created beforehand.*

Example: To output the oscillation frequency adjusting monitor pulse to port R40.

```
SFCCHK:    SET      %OP04,0
           CLR      %OP04,0
           BSS      SFCCHK
```



### 2.7.2 Timing Generator

The timing generator produces the system clocks from basic clock pulse which are supplied to the CPU and the peripheral hardware.

The interval timer consists of a 18-stage binary counter with a divided-by-4 prescaler. The basic clock (frequency: fc) provides the interval timer. Therefore, the output frequency at the last stage is $fc/2^{22}$[Hz]. During reset, the binary counter is cleared to "0", however, the prescaler is not cleared.

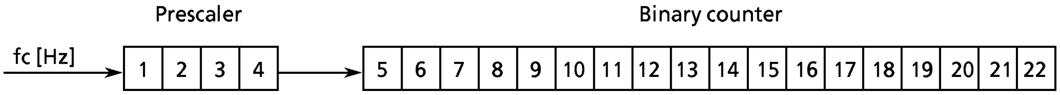


Figure 2-16. Configuration of Interval Timer

The Timing generator produces the following function:

① Internal pulse for internal timer
② Internal pulse for timer/counters
③ Internal serial clock for a serial interface
④ warm-up time at release of the hold operation

### 2.7.3 Instruction Cycle

The instruction execution and the on-chip peripheral hardware operations are performed in synchronization with the basic clock pulse (CP: fc [Hz]). The smallest unit of instruction execution is called an instruction cycle. The instruction set of the TLCS-47 series consists of 1-cycle instructions and 2-cycle instructions. The former requires 1 cycle for their execution; the latter, 2 cycles. Each instruction cycle consists of 4 states (S1 through S4). Each state consists of 2 basic clock pulses.

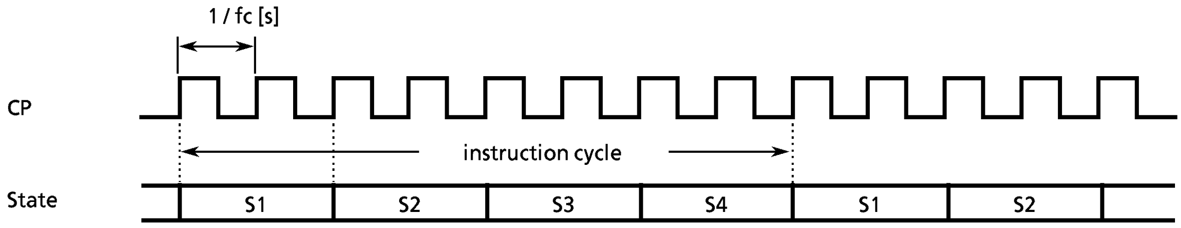


Figure 2-17. Instruction Cycle

### 2.7.4 Hold Operating Mode

The hold feature stops the system and holds the the system's internal states active before stop with a low power. The hold operation is controlled by the command register (OP10) and the $\overline{\text{HOLD}}$ pin input. The $\overline{\text{HOLD}}$ pin input state can be known by the status register (IP0E). The $\overline{\text{HOLD}}$ pin is shared with the $\overline{\text{KE0}}$ pin.

(1) Starts Hold Operating Mode

The hold operation is started when the command is set to the command register and holds the following states during the hold operation:

① The oscillator stops and the system's internal operations are all held up.
② The interval timer is cleared to "0".
③ The states of the data memory, registers, and latches valid immediately before the system is put in the hold state are all held.
④ The program counter holds the address of the instruction to be executed after the instruction which starts the hold operating mode.

Hold operating mode command register
(Port address OP10) (Initial value *0**)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| HLDMS | | HWUT | |

| HLDMS | Sets mode/starts hold operation |
|---|---|
| 01 | Starts hold operation in edge-release mode |
| 11 | Starts hold operation in level-release mode |
| *0 | Unused |

Hold operating mode status register
(Port address IP0E)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| (SIOF) | (SEF) | | HOLD<br>($\overline{\text{KE0}}$) |

| HOLD | $\overline{\text{HOLD}}$ pin input state |
|---|---|
| 0 | $\overline{\text{HOLD}}$ pin is high |
| 1 | $\overline{\text{HOLD}}$ pin is low (HOLD operation request) |

| HWUT | Sets the warm-up time at release of the hold operating mode | |
|---|---|---|
| | | Example : At fc = 4 MHz |
| 00 | $2^{18}$/fc [s] | ··· 65.5 [ms] |
| 01 | $2^{14}$/fc | ··· 4.1 |
| 10 | Unused | |
| 11 | $2^{6}$/fc | ··· 0.016 |

*Note 1. * ; don't care*
*Note 2. Do not access the OP10 when HOLD mode is not used.*
*Note 3. "1" is read as the bit 1 of the IOPE by the input instruction.*

Figure 2-18. Hold Operating Mode Command Register / Status Register

The hold operating mode consists of the level-sensitive release mode and the edge-sensitive release mode.

a. Level-sensitive release mode

In this mode, the hold operation is released by setting the $\overline{\text{HOLD}}$ pin to the high level. This mode is used for the capacitor backup with power off or for the battery backup for long hours.

If the instruction to start the hold operation is executed with the $\overline{\text{HOLD}}$ pin input being high, the hold operation does not start but the release sequence (warm-up) starts immediately. Therefore, to start the hold operation in the level-sensitive release mode, that the $\overline{\text{HOLD}}$ pin input being low (the hold operation request) must be recognized in program. This recognition is performed in one of the two ways below:

① Testing HOLD (bit 0 of the status register)
② Applying the $\overline{\text{HOLD}}$ pin input also to the $\overline{\text{INT1}}$ pin to generate the external interrupt 1 request.

Example: To test HOLD to start the hold operation in the level-sensitive release mode (the warm-up time = $2^{14}$/fc).

```
SHOLDH: TEST    %IP0E, 0    ;  Waits until HOLD pin input goes low.
        B       SHOLDH
        LD      A, #1101B     ;  OP10←1101B
        OUT     A, %OP10
```

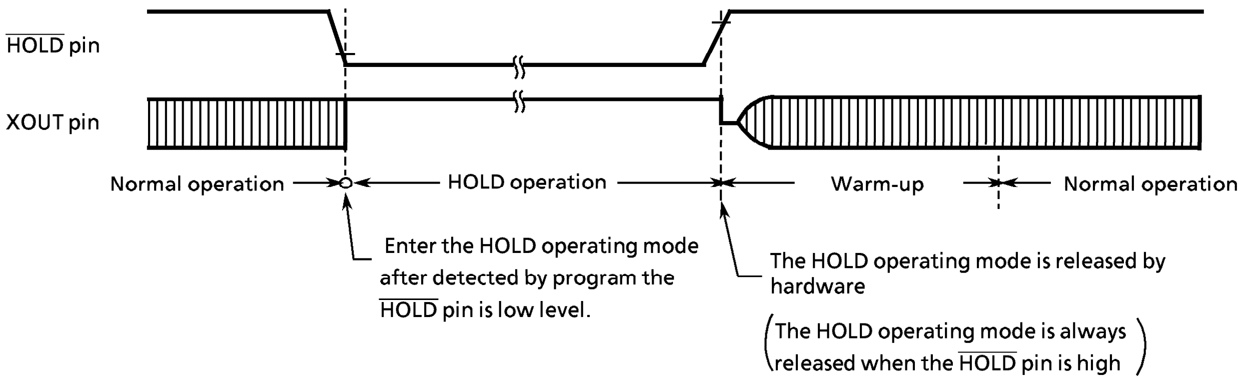

HOLD pin
XOUT pin
Normal operation
HOLD operation
Warm-up
Normal operation
Enter the HOLD operating mode after detected by program the HOLD pin is low level.
The HOLD operating mode is released by hardware
(The HOLD operating mode is always released when the HOLD pin is high)

Figure 2-19. Level-sensitive release mode

b. Edge-sensitive release mode

In this mode, the hold operation is released at the rising edge of the $\overline{\text{HOLD}}$ pin input. This mode is used for applications in which a relatively short-time program processing is repeated at a certain cycle. This cyclic signal (for example, the clock supplied from the low power dissipation oscillator). In the edge-sensitive mode, even if the $\overline{\text{HOLD}}$ pin input is high, the hold operation is performed.

Example: To start the hold operation in the edge-sensitive release mode (the warm-up time = $2^{14}$/fc).

```
LD      A, #0101B     ;  OP10←0101B
OUT     A, %OP10
```

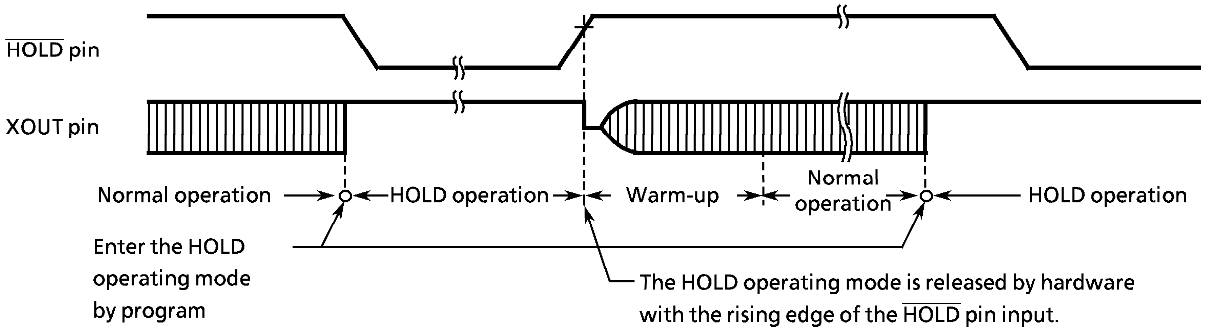

HOLD pin
XOUT pin
Normal operation
HOLD operation
Warm-up
Normal operation
HOLD operation
Enter the HOLD operating mode by program
The HOLD operating mode is released by hardware with the rising edge of the HOLD pin input.

Figure 2-20. Edge-sensitive release mode

*Note: In the hold operation, the dissipation of the power associated with the oscillator and the internal hardware is lowered; however, the power dissipation associated with the pin interface (depending on the external circuitry and program) is not directly determined by the hardware operation of the hold feature. This point should be considered in the system design and the interface circuit design.*

*In the CMOS circuitry, a current does not flow when the input level is stable at the power voltage level ($V_{DD}$ / $V_{SS}$); however, when the input level gets higher than the power voltage level (by approximately 0.3 to 0.5 V), a current begins to flow. Therefore, if cutting off the output transistor at an I/O port (the open drain output pin with an input transistor connected) puts the pin signal into the high-impedance state, a current flows across the ports input transistor, requiring to fix the level by pull-up or other means.*

(2) Releases Hold Operating Mode

The hold operating mode is released in the following sequence:

① The oscillator starts

② Warm-up is performed to acquire the time for stabilizing oscillation. During the warm-up , the internal operations are all stopped. One of three warm-up times can be selected by program depending on the characteristics of the oscillator used.

③ When the warm-up time has passed, an ordinary operation restarts from the instruction next to the instruction which starts the hold operation. At this time, the interval timer starts from the reset state "0".

The warm-up time is obtained by dividing the basic clock by the interval timer, so that, if the frequency at releasing the hold operation is unstable, the warm-up time shown in Figure 2-18. includes an error. Therefore, the warm-up time must be handled as an approximate value. The hold operation is also released by setting the $\overline{\text{RESET}}$ pin to the low level. In this case, the normal reset operation follows immediately.

## 2.8 INTERRUPT FUNCTION

(1) Interrupt Controller

There are 6 interrupt sources (2 external and 4 internal). The prioritized multiple interrupt capability is supported. The interrupt latches ($IL_5$ through $IL_0$) to hold interrupt requests are provided for the interrupt sources. Each interrupt latch is set to "1" when an interrupt request is made, asking the CPU to accept the interrupt. The acceptance of interrupt can be permitted or prohibited by program through the interrupt enable master flip-flop (EIF) and interrupt enable register (EIR). When two or more interrupts occur simultaneously, the one with the highest priority determined by hardware is serviced first.

Table 2-2. Interrupt Sources

| Interrupt Source | | | Priority | Interrupt Latch | Enable conditions | Entry address |
|---|---|---|---|---|---|---|
| External | Extenal Interrupt 1 | (INT1) | (highest) 1 | $IL_5$ | EIF = 1 | $002_H$ |
| Internal | Serial Interface Interrupt | (ISIO) | 2 | $IL_4$ | EIF = 1, $EIR_3$ = 1 | $004_H$ |
| | TC1 overflow Interrupt | (IOVF1) | 3 | $IL_3$ | EIF = 1, $EIR_2$ = 1 | $006_H$ |
| | TC2 overflow Interrupt | (IOVF2) | 4 | $IL_2$ | EIF = 1, $EIR_1$ = 1 | $008_H$ |
| | Interval Timer Interrupt | (ITMR) | 5 | $IL_1$ | | $00A_H$ |
| External | External Interrupt 2 | (INT2) | (lowest) 6 | $IL_0$ | EIF = 1, $EIR_0$ = 1 | $00C_H$ |

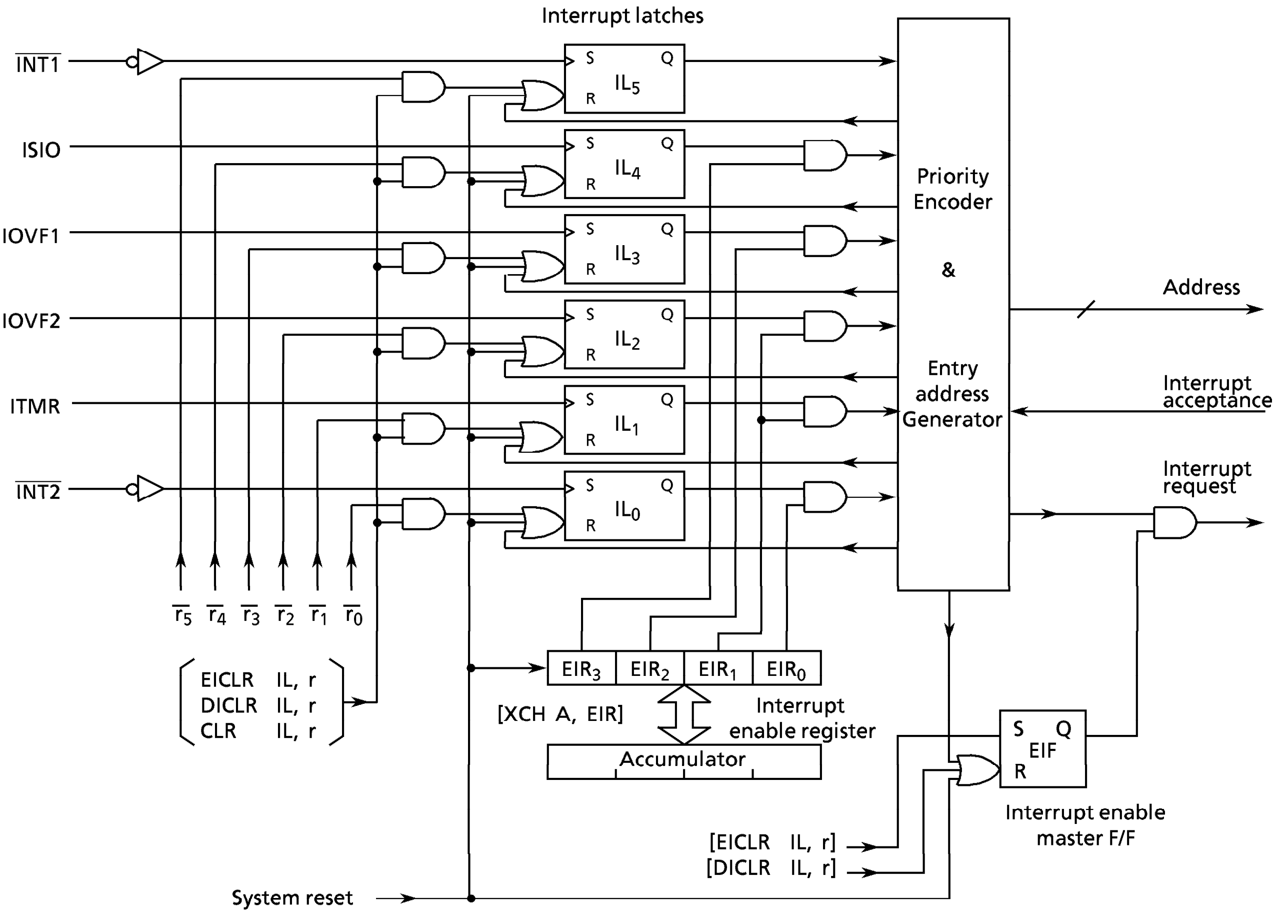


Figure 2-21. Interrupt Controller Block Diagram

a. Interrupt enable master flip-flop (EIF)

The EIF controls the enable / disable of all interrupts. When this flip-flop is cleared to "0", all interrupts are disabled; when it is set to "1", the interrupts are enabled.

When an interrupt is accepted, the EIF is cleared to "0", temporarily disabling the acceptance of subsequent interrupts. When the interrupt service program has been executed, the EIF is set to "1" by the execution of the interrupt return instruction [RETI], being put in the enabled state again.

Set or clear of the EIF in program is performed by instructions [EICLR IL, r] and [DICLR IL, r], respectively. The EIF is initialized to "0" during reset.

b. Interrupt enable register (EIR)

The EIR is a 4-bit register specifies the enable or disable of each interrupt except INT1. An interrupt is enabled when the corresponding bit of the EIR is "1", and an interrupt is disabled when the corresponding bit of the EIR is "0". Bit 1 of the EIR ($EIR_1$) is shared by both IOVF2 and ITMR interrupts.

Read/write on the EIR is performed by executing [XCH A, EIR] instruction. The EIR is initialized to "0" during reset.

c. Interrupt latch (IL)

An interrupt latch is provided for each interrupt source. The IL is set to "1" when an interrupt request is made to ask the CPU for accepting the interrupt. Each IL is cleared to "0" upon acceptance of the interrupt. It is initialized to "0" during reset.

The ILs can be cleared independently by interrupt latch operation instructions ([EICLR IL, r], [DICLR IL, r], and [CLR IL, r]) to make them cancel interrupt requests or initialize by program. When the value of instruction field (r) is "0", the interrupt latch is cleared; when the value is "1", the IL is held. Note that the ILs cannot be set by instruction.

Example 1: To enable IOVF1, INT1, and INT2 interrupts.

```
LD      A,#0101B   ;  EIR←0101B
XCH     A,EIR
EICLR   IL,111111B ;  EIF←1
```

Example 2: To set the EIF to "1", and to clear the interrupt latches except ISIO to "0".

```
EICLR   IL,010000B ;  EIF←1, IL0←0, IL2 – IL5←0
```

(2) Interrupt Processing

An interrupt request is held until the interrupt is accepted or the IL is cleared by the reset or the interrupt latch operation instruction. The interrupt acknowledge processing is performed in 2 instruction cycles after the end of the current instruction execution (or after the timer/counter processing if any). The interrupt service program terminates upon execution of the interrupt return instruction [RETI].

The interrupt acknowledge processing consists of the following sequence:

① The contents of the program counter and the flags are saved on the stack.

② The interrupt entry address corresponding to the interrupt source is set to the program counter.

③ The status flag is set to "1".

④ The EIF is cleared to "0", temporarily disabling the acceptance of subsequent interrupts.

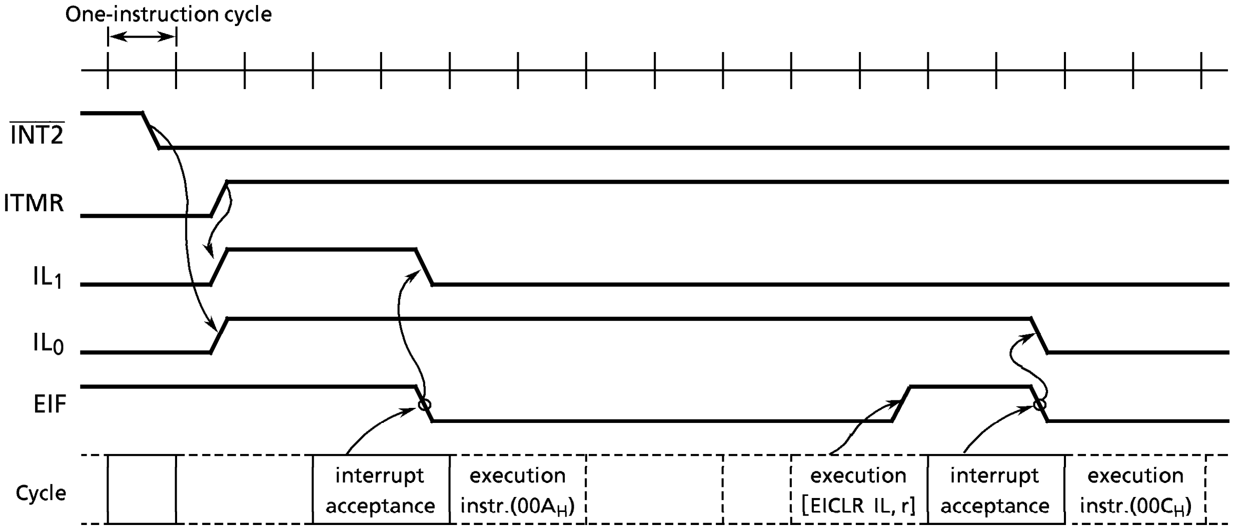
⑤ The interrupt latch for the accepted interrupt source is cleared to "0".

⑥ The instruction stored at the interrupt entry address is executed. (Generally, in the program memory space at the interrupt entry address, the branch instruction to each interrupt processing program is stored.)

To perform the multi-interrupt, the EIF is set to "1" in the interrupt service program, and the acceptable interrupt source is selected by the EIR. However, for the INT1 interrupt, the interrupt service is disabled under software control because it is not disabled by the EIR.

Example: The INT1 interrupt service is disabled under software control (Bit 0 of RAM [$05_H$] are assigned to the disabling switch of interrupt service).

```
PINT1:  TEST    05H,0    ;  Skips if RAM [05H]0 is "1"
        B       SINT1
        RETI
SINT1:   ⋮
```



*Notes.*

*1. It is assumed that there is no other interrupt request and EIR = $0011_B$.*

*2. The value r in the [EICLR IL, r] instruction is assumed as $111111_B$.*

*3. ⬚ denotes the execution of an instruction.*

Figure 2-22. Interrupt Timing chart (Example)

The interrupt return instruction [RETI] performs the following operations :

① Restores the contents of the program counter and the flags from the stack.

② Sets the EIF to "1" to provide the interrupt enable state again.

Note. *When the timer reguired for the instrupt service is longer than that for the interrupt request, only the interrupt service program is executed without executing the main program.*

In the interrupt processing, the program counter and flags are automatically saved or restored but the accumulator and other registers are not. If it is necessary to save or restore them, it must be performed by program as shown in the following example. To perform the multi-interrupt, the saving RAM area never be overlapped.

Example: To save and restore the accumulator and HL register pair.

```
XCH     HL, GSAV1      ;   RAM [GSAV1] ↔ HL
XCH     A, GSAV1 + 2   ;   RAM [GSAV1 + 2] ↔ Acc
```

*Note. The lower 2 bits of GSAV1 should be "0's".*

(3) External Interrupt

When an external interrupt (INT1 or INT2) occurs, the interrupt latch is set at the falling edge of the corresponding pin input ($\overline{\text{INT1}}$ or $\overline{\text{INT2}}$).

The INT1 interrupt cannot be disabled by the EIR, so that it is always accepted in the interrupt enable state (EIF = "1"). Therefore, INT1 is used for an interrupt with high priority such as an emergency interrupt. When R82 ($\overline{\text{INT1}}$) pin is used for the I/O port, the INT1 interrupt occurs at the falling edge of the pin input, so that the interrupt return [RETI] instruction must be stored at the interrupt entry address to perform dummy interrupt processing.

Because the external interrupt input is the hysteresis type, each of high and low level time requires 2 or more instruction cycles for a correct interrupt operation.

The INT2 interrupt can be enabled/disable by the EIR. When R80 ($\overline{\text{INT2}}$) pin is used as the I/O port, the INT2 interrupt ouurs at the falling edge of the pin input.

However the interrupt request is not auepted by clearing bit 0 of the EIR to "0".

## 2.9 RESET FUNCTION

When the $\overline{\text{RESET}}$ pin is held to the low level for three or more instruction cycles when the power voltage is within the operating voltage range and the oscillation is stable, reset is performed to initialize the internal states.

When the $\overline{\text{RESET}}$ pin input goes high, the reset is cleared and program execution starts from address $000_H$. The $\overline{\text{RESET}}$ pin is a hysteresis input with a pull-up resistor (220 kΩ typ.). Externally attaching a capacitor and a diode implement a simplified power-on-reset operation.

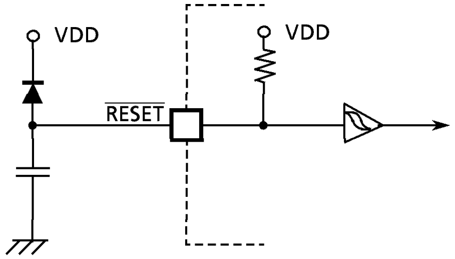


Figure 2-23. Simplified Power-On-Reset Circuit

Table 2-3. Initialization of Internal States by Reset Operation

| On-chip hardware | Initial value | On-chip hardware | Initial value |
|---|---|---|---|
| Program counter (PC) | $000_H$ | Output latch (I/O ports or Output ports) | Refer to "INPUT / OUTPUT Circuitry". |
| Status flag (SF) | 1 | | |
| Interrupt enable master flip-flop (EIF) | 0 | | |
| Interrupt enable register (EIR) | $0_H$ | Command register | Refer to the description of each relative command register. |
| Interrupt latch (IL) | "0" | | |
| Interval timer | "0" | | |

## 3. PERIPHERAL HARDWARE FUNCTION

### 3.1 Ports

The data transfer with the external circuit and the command / status / data transfer with the internal circuit are performed by using the I/O instructions (13 kinds). There are 4 types of ports:

| | | |
|---|---|---|
| ① I/O port | ; | Data transfer with external circuit |
| ② Command register | ; | Control of internal circuit |
| ③ Status register | ; | Reading the status signal from internal circuit |
| ④ Data register | ; | Data transfer with internal circuit |

These ports are assigned with port addresses ($00_H$ through $1F_H$). Each port is selected by specifying its port address in an I/O instruction. Table 3-2 lists the port address assignments and the I/O instructions that can access the ports.

### 3.1.1 I/O Timing

(1) Input timing

External data is read from an input port or an I/O port in the S3 state of the second instruction cycle during the input instruction (2-cycle instruction) execution. This timing cannot be recognized from the outside, so that the transient input such as chattering must be processed by program.

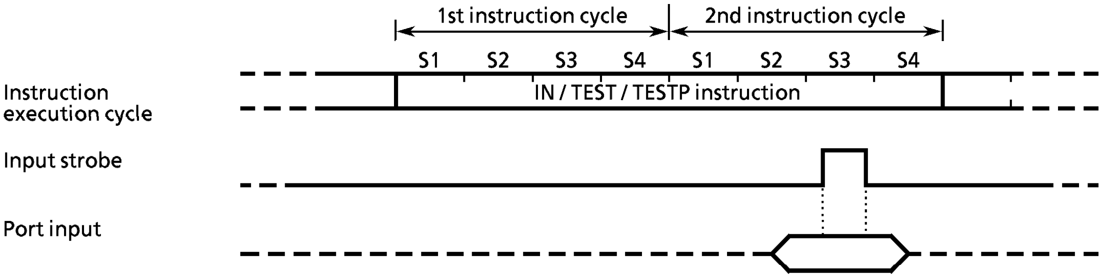


Figure 3-1. Input Timing

(2) Output timing

Data is output to an output port or an I/O port in the S4 state of the second instruction cycle during the output instruction (2-cycle instruction) execution.

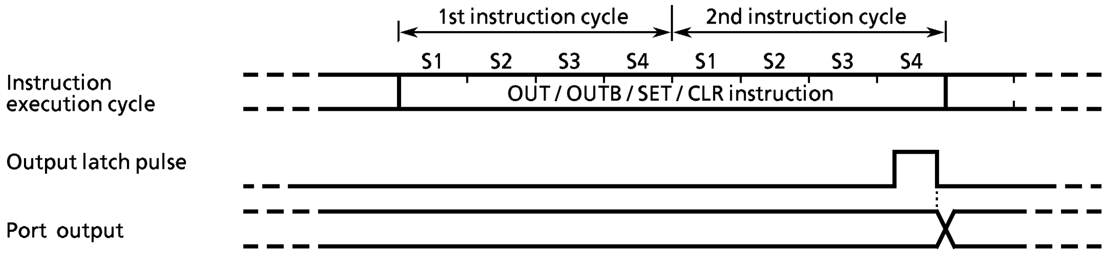


Figure 3-2. Output Timing

### 3.1.2 I/O Ports

The 47C103/203 have 7 I/O ports (23 pins) each as follows:

| | | |
|---|---|---|
| ① R4, R5, R6 | ; | 4-bit output |
| ② R7 | ; | 3-bit input / output (shared with watchdog timer output; R70) |
| ③ R8 | ; | 4-bit input / output (shared with external interrupt input and timer / counter input) |
| ④ R9 | ; | 3-bit input / output (shared with serial port) |
| ⑤ KE | ; | 1-bit sense input (shared with hold request / release signal input) |

Each output port contains a latch, which holds the output data. The input ports have no latch; therefore, it is desired to hold data externally until it is read or read twice or more before processing it.

(1) Port R4, R5, R6, R7

Port R4 is a 4-bit (Port R7 is 3 bits) I/O port with a latch. When used as an input port, the latch must be set to "1".

The latch is initialized to "1" during reset.

These 4 ports (15 pins) can be set, cleared, and tested for each bit as specified by L register indirect addressing bit manipulation instructions ([SET @L], [CLR @L], and [TEST @L]). Table 3-1 lists the pins (I/O ports) that correspond to the contents of L register.

Example: To clear R43 output as specified by the L register indirect addrressing bit manipulation instruction.

```
LD     L, #0011B     ;  Sets R43 pin address to L register
CLR    @L            ;  R43←0
```

| L register | | | | PIN |
|---|---|---|---|---|
| 3 | 2 | 1 | 0 | |
| 0 | 0 | 0 | 0 | R40 |
| 0 | 0 | 0 | 1 | R41 |
| 0 | 0 | 1 | 0 | R42 |
| 0 | 0 | 1 | 1 | R43 |
| 0 | 1 | 0 | 0 | R50 |
| 0 | 1 | 0 | 1 | R51 |
| 0 | 1 | 1 | 0 | R52 |
| 0 | 1 | 1 | 1 | R53 |
| 1 | 0 | 0 | 0 | R60 |
| 1 | 0 | 0 | 1 | R61 |
| 1 | 0 | 1 | 0 | R62 |
| 1 | 0 | 1 | 1 | R63 |
| 1 | 1 | 0 | 0 | R70 |
| 1 | 1 | 0 | 1 | R71 |
| 1 | 1 | 1 | 0 | R72 |

Table 3-3. Relationship between L register contents and I/O port bits

(a) Port R4 (Port address OP04 / IP04)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| R43 | R42 | R41 | R40 |

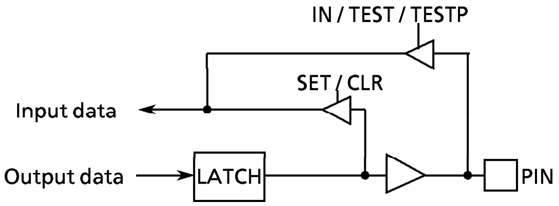


Figure 3-4. Ports R4

(b) Ports R5 (R53 to R50) and R6 (R63 to R60)

Ports R5 and R6 are 4-bit high current output ports with a latch, which can directly drive LEDs. When an input instruction is executed, the latch data is read in these ports. They can be accessed separately at port addresses OP05/IP05 and OP06/IP06. Additionally, 8-bit data can be set to these ports (the upper 4 bits to port R6, the lower 4 bits to port R5) by using the 5-bit to 8-bit data conversion instruction [OUTB @HL].

Example 1: To output immediate value "5" to port R5

```
OUT      #5,%OP05        ;  Port R5←5
```

Example 2: To read the latch data from port R6 and store it in the accumulator

```
IN       %IP06,A         ;  Acc←Port R6
```

Example 3: To read from ROM, the 8-bit data corresponding to the 5 bits obtained by linking the content (1 bit) of the carry flag with the contents (4 bits) of at address $90_H$ in RAM to output the 8-bit data to ports R6 and R5.

```
LD       HL,#90H         ;  HL←90H (Sets the data memory address)
OUTB     @HL             ;  Ports R6, R5←ROM data
```

Port R5 (Port address OP05 / IP05)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| R53 | R52 | R51 | R50 |

Port R6 (Port address OP06 / IP06)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| R63 | R62 | R61 | R60 |

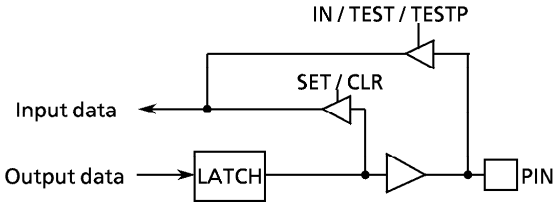


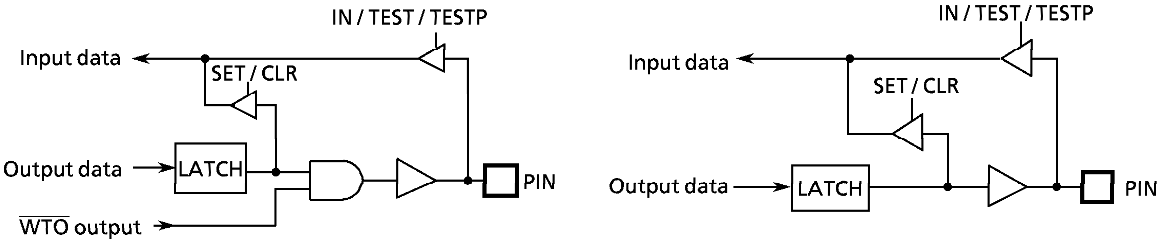
Figure 3-5. Ports R5, R6

(c) Port R7 (R72 to R70)

Port R7 is 3-bits I/O port with latch. R70 pin is shared by the watchdog timer output. To use R70 pin for the watchdog timer output, the latch should be set to "1". The latch is initialized to "1" during reset. R73 pin does not exist actually but "1" is read when an input instruction is executed.

Port R7 (Port address OP07 / IP07)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
|  | R72 | R71 | R70<br>($\overline{\text{WTO}}$) |



(a) R70 (WTO)

(b) R71, R72

Figure 3-6. Port R7

(2) Port R8 (R83 to R80)

Port R8 is a 4-bit I/O port with a latch. When used as an input port, the latch must be set to "1". The latch is initialized to "1" during reset.

Port R8 is shared with the external interrupt input pin and the timer / counter input pin. To use this port for one of these functional pins, the latch should be set to "1". To use it for an ordinary I/O port, the acceptance of external interrupt should be disabled or the event counter / pulse width measurement modes of the timer / counter should be disabled.

Note: *When R82* ($\overline{\text{INT1}}$) *pin is used for an I/O port, external interrupt 1 occurs upon detection of the falling edge of pin input, and if the interrupt enable master flip-flop is enabled, the interrupt request is always accepted. So that a dummy interrupt processing must be performed (only the interrupt return instruction [RETI] is executed).*

*With R80* ($\overline{\text{INT2}}$) *pin, external interrupt 2 occurs like R82 in but bit 0 of the interrupt enable register (EIR*$_0$*) is only kept at "0", not accepting the interrupt request.*

Port R8 (Port address OP08 / IP08)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| R83<br>(T1) | R82<br>($\overline{\text{INT1}}$) | R81<br>(T2) | R80<br>($\overline{\text{INT2}}$) |

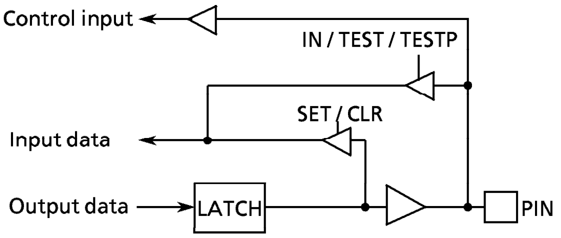


Figure 3-7. Port R8

(3) Port R9 (R92 to R90)

Port R9 is a 3-bit I/O port with a latch. When used as an input, the latch must be set to "1". The latch is initialized to "1" during reset.

Port R9 is shared with the serial port. To use port R9 for the serial port, the latch should be set to "1". To use port R9 for an ordinary I/O port, the serial interface must be disabled.

Note that R93 pin does not exist actually but "1" is read when an input instruction is executed.

Port R9 (Port address OP09 / IP09)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
|  | R92<br>($\overline{\text{SCK}}$) | R91<br>(SO) | R90<br>(SI) |

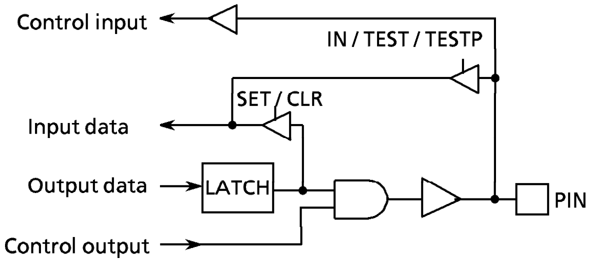


Figure 3-8. Port R9

(4) Port KE ($\overline{\text{KE0}}$)

Port KE is a 1-bit sense input port shared with the hold request / release signal input in ($\overline{\text{HOLD}}$). This input port is assigned to the least significant bit of port address IP0E and is processed as the data with inverted polarity. For example, if an input instruction is executed with the pin on the high level, "0" is read. The bit 1 of port KE, an undefined value is read when an input instruction is executed.

Example: To wait until $\overline{\text{KE0}}$ pin goes low.

```
SWAIT  :  TEST     %IP0E,0         ; Waits if KE0 pin = "H".
          B        SWAIT
```

Port KE (Port address IP0E)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| (SIOF) | (SEF) |  | KE0<br>($\overline{\text{HOLD}}$) |

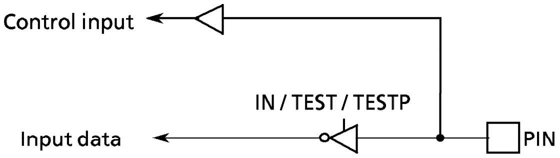


Figure 3-9. Port KE

Table 3-2. Port Address Assignments and Available I/O Instructions

| Port Address (**) | Port |  | Input/Output instruction |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|
|  | Input (IP**) | Output (OP**) | IN %p, A<br>IN %p, @HL | OUT A,%p<br>OUT @HL,%p | OUT #k, %p | OUTB @HL | SET %p, b<br>CLR %p, b | TEST %p, b<br>TESTP %p,b | SET @L<br>CLR @L<br>TEST @L |
| $00_H$ | —— | —— | – | – | – | – | – | – | – |
| 01 | —— | —— | – | – | – | – | – | – | – |
| 02 | —— | —— | – | – | – | – | – | – | – |
| 03 | —— | —— | – | – | – | – | – | – | – |
| 04 | R4 input port | R4 output port | ○ | ○ | ○ | – | ○ | ○ | ○ |
| 05 | R5 input port | R5 output port | ○ | ○ | ○ | ○ (Note 2) | ○ | ○ | ○ |
| 06 | R6 input port | R6 output port | ○ | ○ | ○ | ○ | ○ | ○ | ○ |
| 07 | R7 input port | R7 output port | ○ | ○ | ○ | – | ○ | ○ | ○ |
| 08 | R8 input port | R8 output port | ○ | ○ | ○ | – | ○ | ○ | – |
| 09 | R9 input port | R9 output port | ○ | ○ | ○ | – | ○ | ○ | – |
| 0A | —— | —— | – | – | – | – | – | – | – |
| 0B | —— | —— | – | – | – | – | – | – | – |
| 0C | —— | —— | – | – | – | – | – | – | – |
| 0D | —— | —— | – | – | – | – | – | – | – |
| 0E | Hold status | —— | ○ | – | – | – | – | ○ | – |
| 0F | Serial receive buffer | Serial transmit buffer | ○ | ○ | ○ | – | – | – | – |
| $10_H$ | Undefined | Hold operating mode control | – | ○ | – | – | – | – | – |
| 11 | Undefined | —— | – | – | – | – | – | – | – |
| 12 | Undefined | —— | – | – | – | – | – | – | – |
| 13 | Undefined | —— | – | – | – | – | – | – | – |
| 14 | Undefined | —— | – | – | – | – | – | – | – |
| 15 | Undefined | Watchdog Timer control | – | ○ | – | – | – | – | – |
| 16 | Undefined | —— | – | – | – | – | – | – | – |
| 17 | Undefined | —— | – | – | – | – | – | – | – |
| 18 | Undefined | —— | – | – | – | – | – | – | – |
| 19 | Undefined | Interval Timer interrupt control | – | ○ | – | – | – | – | – |
| 1A | Undefined | —— | – | – | – | – | – | – | – |
| 1B | Undefined | —— | – | – | – | – | – | – | – |
| 1C | Undefined | Timer/Counter 1 control | – | ○ | – | – | – | – | – |
| 1D | Undefined | Timer/Counter 2 control | – | ○ | – | – | – | – | – |
| 1E | Undefined | SIO control 1 | – | ○ | – | – | – | – | – |
| 1F | Undefined | SIO control 2 | – | ○ | – | – | – | – | – |

*Note 1. "____" means the reserved state. Unavailable for the user programs.*

*Note 2. The 5-bit to 8-bit data conversion instruction [OUTB @HL], automatic access to ports R5 and R6.*

## 3.2 Interval Timer Interrupt (ITMR)

The interval timer can be used to generate an interrupt with a fixed frequency. Internal time interrupt is control by the command register (OP19). An interval timer interrupt is generated at the first rising edge of the binary counters output after the command has been set. The interval timer is not cleared by command, so that the first interrupt may occur earlier than the preset interrupt period.

Example: To set the interval timer interrupt frequency to fc/$2^{12}$ [Hz].

```
LD      A,  #0110B    ;  OP19←0110B
OUT     A,  %OP19
```

Interval Timer interrupt command register
(Port address OP19)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| TMRE | | TMRF | |

(Initial value 0000)

| TMRE | Interrupt disable/enable |
|---|---|

00 : Stop
01 : Enable
1* : (Reserved)

Note. * ; don't care

| TMRF | Interrupt frequency |
|---|---|

| | | Example : fc = 4.19 MHz |
|---|---|---|
| 00 : | fc / $2^{10}$ [Hz] | 4096 [Hz] |
| 01 : | fc / $2^{11}$ | 2048 |
| 10 : | fc / $2^{12}$ | 1024 |
| 11 : | fc / $2^{13}$ | 512 |

Figure 3-10. Interval Timer Interrupt Command Register

## 3.3 Timer/Counters (TC1, TC2)

The 47C103/203 contain two 12-bit timer/counters (TC1, TC2). RAM addresses are assigned to the count register in unit of 4 bits, permitting the initial value setting and counter reading through the RAM manipulation instruction. When the timer/counter is not used, the mode selection may be set to "stopped" to use the RAM at the address corresponding to the timer / counter for storing the ordinary user-processed data.

| | MSB Timer / Counter 1 (TC1) | | LSB |
|---|---|---|---|
| | $TC1_H$ | $TC1_M$ | $TC1_L$ |
| RAM address | | | |
| 47C203 | ($76_H$) | ($75_H$) | ($74_H$) |
| 47C103 | ($36_H$) | ($35_H$) | ($34_H$) |

| | MSB Timer / Counter 2 (TC2) | | LSB |
|---|---|---|---|
| | $TC2_H$ | $TC2_M$ | $TC2_L$ |
| RAM address | | | |
| 47C203 | ($7A_H$) | ($79_H$) | ($78_H$) |
| 47C103 | ($3A_H$) | ($39_H$) | ($38_H$) |

Figure 3-11. The Count registers of the Timer / Counters (TC1, TC2)

### 3.3.1 Functions of Timer / Counters

The timer / counters provide the following functions:

① Event counter
② Programmable timer
③ Pulse width measurement

### 3.3.2 Control of Timer / Counters

The timer / counters are controlled by the command registers. The command register is accessed as port address OP1C for TC1 and port address OP1D for TC2. These registers are initialized to "0" during reset.

Timer / Counter 1 control command register
(Port address OP1C) (Initial value 0000)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| TC1MS | | IPR1 | |

| TC1MS | Mode selection |
|---|---|
| 00 : | Stop |
| 01 : | Event counter mode |
| 10 : | Timer mode |
| 11 : | Pulse width measurement mode |

| IPR1 | Internal pulse rate (interval timer output) selection | |
|---|---|---|
| | Example: | At fc = 4.19 MHz |
| 00 : | $fc/2^{10}$[Hz] | ··· 4096 [Hz] |
| 01 : | $fc/2^{14}$ | ··· 256 |
| 10 : | $fc/2^{18}$ | ··· 16 |
| 11 : | $fc/2^{22}$ | ··· 1 |

Timer / Counter 2 control command register
(Port address OP1D) (Initial value 0000)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| TC2MS | | IPR2 | |

| TC2MS | Mode selection |
|---|---|
| 00 : | Stop |
| 01 : | Event counter mode |
| 10 : | Timer mode |
| 11 : | Pulse width measurement mode |

| IPR2 | Internal pulse rate (interval timer output) selection | |
|---|---|---|
| | Example: | At fc = 4.19 MHz |
| 00 : | $fc/2^{10}$[Hz] | ··· 4096 [Hz] |
| 01 : | $fc/2^{14}$ | ··· 256 |
| 10 : | $fc/2^{18}$ | ··· 16 |
| 11 : | $fc/2^{22}$ | ··· 1 |

*Note. fc; Basic clock frequency [Hz]*

Figure 3-12. Timer / Counter Control Command Registers

The timer/counter increments at the rising edge of each count pulse. Counting starts with the first rising edge of the count pulse generated after the command has been set. Count operation is performed in one instruction cycle after the current instruction execution, during which the execution of a next instruction and the acceptance of an interrupt are delayed. If counting is requested by both TC1 and TC2 simultaneously, the request by TC1 is preferred. The request by TC2 is accepted in the next instruction cycle. Therefore, during count operation, the apparent instruction execution speed drops as counting occurs more frequently.

The timer / counter causes an interrupt upon occurrence of an overflow (a transition of the count value from $FFF_H$ to $000_H$). If the timer / counter is in the interrupt enabled state and the overflow interrupt is accepted immediately after its occurrence, the interrupt is processed in the sequence shown in Figure 3-13. Note that counting continues if there is a count request after overflow occurrence.

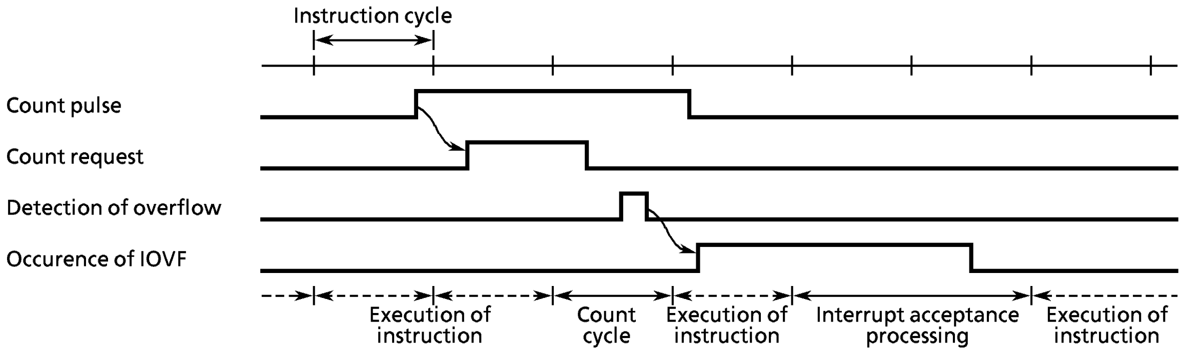


Figure 3-13. Timer / Counter Overflow Interrupt Timing

(1) Event counter mode

In the event counter mode, the timer / counter increments at each rising edge of the external pin (T1, T2) input. T1 and T2 pins are also used as the R83 and R81 pins.

To use these pins as the T/C input, set the output latch of R83 and R83 and R81 to "1".

At reset, the output latch is initialized to "1". The maximum applied frequency of the external pin input is fc/32 for the 1-channel operation; for the 2-channel operation, the frequency is fc/32 for TC1 and fc/40 for TC2. The apparent instruction execution speed drops most to (9/11) × 100 = 82 % when TC1 and TC2 are operated at the maximum applied frequency because the count operation is inserted once every 4 instruction cycles for TC1 and every 5 cycles for TC2. For example, the instruction execution speed of 2 $\mu$s drops to 3.64 $\mu$s.

Example: To operate TC2 in the event counter mode

```
        LD      A,  #0100B    ;  OP1D←01**B
        OUT     A,  %OP1D
```

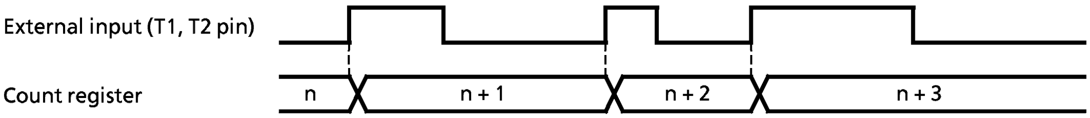


Figure 3-14. Event Counter Mode Timing chart

(2) Timer mode

In the timer mode, the timer/counter increments at the rising edge of the internal pulse generated from the interval timer. One of 4 internal pulse rates can be selected by the command register. The selected rate can be initially set to the timer/counter to generate an overflow interrupt in order to create a desired time interval.

When an internal pulse rate of fc/$2^{10}$ is used, a count operation is inserted once every 128 instruction cycles, so that the apparent instruction execution speed drops by (1/127) × 100 = 0.8 %. For example, the instruction execution speed of 2 $\mu$s drops to 2.016 $\mu$s.

In the timer mode, R83 (T1) and R81 (T2) pins provide the ordinary I/O ports.

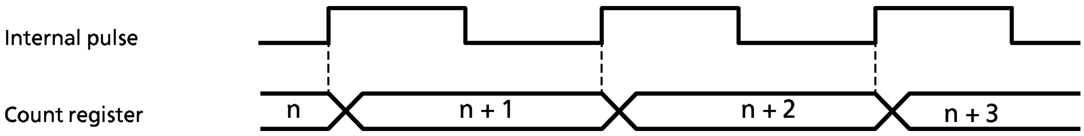


Figure 3-15. Timer Mode Timing chart

Example: To generate an overflow interrupt (at fc = 4 MHz) by the TC1 after 100 ms.

```
LD          HL,  #0F4H     ;  TC1←E79H (setting of the count register)
ST          #9,  @HL+
ST          #7,  @HL+
ST          #0EH,  @HL+
LD          A,  #1000B     ;  OP1C←1000B
OUT         A,  %OP1C
LD          A,  #0100B     ;  EIR←0100B (enables interrupt)
XCH         A,  EIR
EICLR       IL,  110111B   ;  EIF←1, IL3←0
```

**** Calculating the preset value of the counter register***

The preset value of the count register is obtained from the following relation:

$2^{12} -$ (interrupt setting time) × (internal pulse rate)

For example, to generate an overflow interrupt after 100 ms at fc = 4 MHz with the internal pulse rate of fc/$2^{10}$, set the following value to the count register as the preset value:

$2^{12} - (100 \times 10^{-3}) \times (4 \times 10^{6}/2^{10}) = 3705 = \text{E79}_H$

* The apparent execution rate is caleulated as following.

$$1 \div \{ \frac{\text{(Fundamental clock frequency) / 8}}{\text{(Internal pulse rate)}} - 1\} \times 100 \quad [\%]$$

Table 3-3. Internal Pulse Rate Selection

| Internal pulse rate | Max. setting time | Example : At fc = 4.194304 MHz | |
|---|---|---|---|
| | | Internal pulse rate | Max. setting time |
| fc/$2^{10}$ [Hz] | $2^{22}$/fc [s] | 4096 [Hz] | 1 [s] |
| fc/$2^{14}$ | $2^{26}$/fc | 256 | 16 |
| fc/$2^{18}$ | $2^{30}$/fc | 16 | 256 |
| fc/$2^{22}$ | $2^{34}$/fc | 1 | 4096 |

(3) Pulse width measurement mode

In the pulse width measurement mode, the timer / counter increments with the pulse obtained by sampling the external pin (T1, T2) by the internal pulse. As shown in Figure 3-16, the timer / counter increments only while the external pin input is high. The maximum applied frequency to the external pin input must be one that is enough for analyzing the count value. Normally, a frequency sufficient slower than the internal pulse rate setting is applied to the external pin.

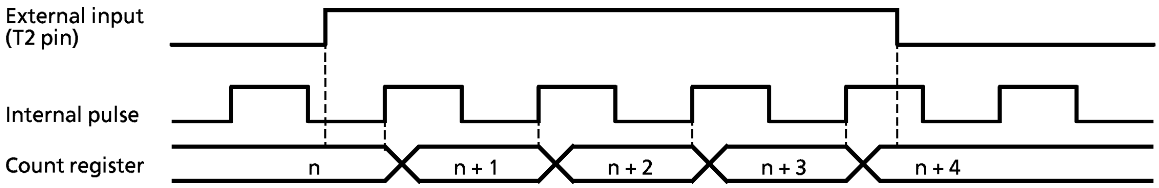


Figure 3-16. Pulse Width Measurement Mode Timing chart

## 3.4 Watchdog Timer (WDT)

The purpose of the watchdog timer is to detect the malfunction (ranaway) of program due to external noise or other causes and return the operation to the normal condition.

The watchdog timer output is output to R70 must be set to "1". Further, during reset, the output latch of R70 is set to "1", and the watchdog timer becomes disable state.

The initialization at time of runaway will become possible when the $\overline{\text{WTO}}$ pin and $\overline{\text{RESET}}$ pin are connected each other.

### 3.4.1 Configuration of Watchdog Timer

The watchdog timer consists of 3-stage binary counter, flip-flop (F/F), and its control circuit. The F/F is set to "1" during reset, and cleared to "0" at the rising edge of the binary counter output.

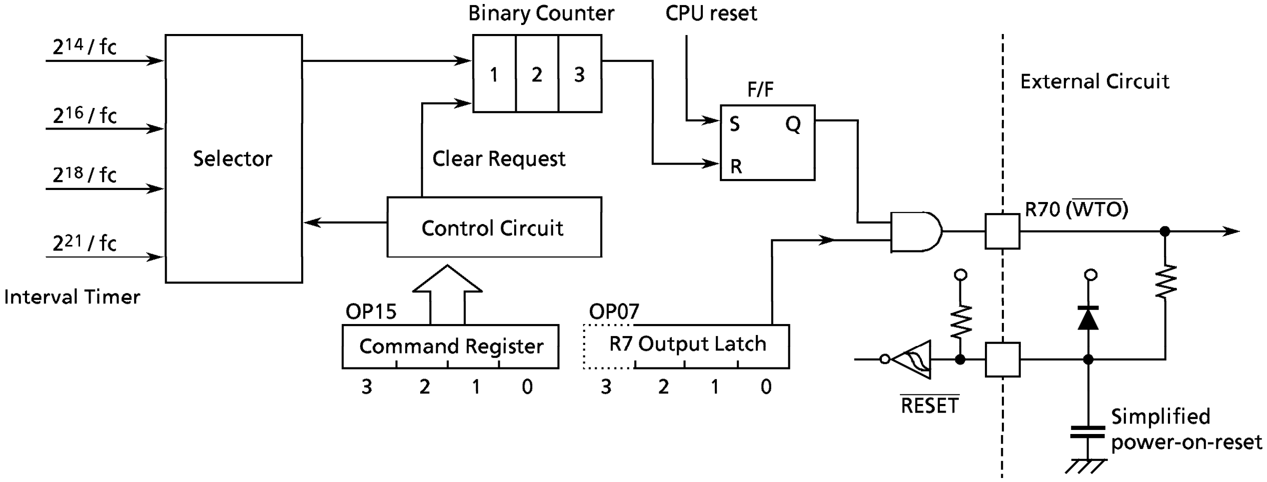


Figure 3-17. Watchdog Timer

### 3.4.2 Control of watchdog timer

The watchdog timer is controlled by the command register (OP15). This command register is initialized to "$1000_B$" during reset. The following are procedure to detect the malfunction (runaway) of CPU by the watchdog timer.

① At first, detection time of the watchdog timer should be set and binary counter should be cleared.

② The watchdog timer should be become enable.

③ Binary counter must be cleared before the detection time of the watchdog timer. When the runaway of CPU is taken place for some reason and binary counter is not cleared, the F/F is cleared to "0"at the rising edge of the binary counter and signal of runaway detection is become active ($\overline{\text{WTO}}$ output is "L" ).

Watchdog Timer control command register
(Port address OP15)

| 3 | 2 | 1 | 0 | |
|---|---|---|---|---|
| RWT | EWT | TWT | | (Initial value 1000) |

| RWT | Clears binary counter |
|---|---|

0 : Clears binary counter (After clear, automatically "1" is set)

| EWT | Enable/Disable |
|---|---|

0 : Disable
1 : Enable

| TWT | Setting of watchdog timer detection time |
|---|---|

| | | | Example : At fc = 4.19 MHz |
|---|---|---|---|
| 00 : | $2^{17}/fc$ [s] | …… | 31.25 [ms] |
| 01 : | $2^{19}/fc$ | …… | 125 |
| 10 : | $2^{21}/fc$ | …… | 500 |
| 11 : | $2^{24}/fc$ | …… | 4000 |

*Note. fc ; Basic clock frequency [Hz]*

Figure 3-18. Command Register

Example : To set the watchdog detection time ($2^{21}$ / fc [s] ). And to enable the watchdog timer.

```
                    LD      A, #0010B     ;  OP15 ← 0010B
                                             (Sets WDT detection time. Clears binary counter)
                    OUT     A, %OP15
                    LD      A, #0110B     ;  OP15 ← 0110B  (Enables WDT)
                  ┌ OUT     A, %OP15
                  │  :
Within WDT        │  :
detection time    │  :
                  │ LD      A, #0110B     ;  OP15 ← 0110B  (Clears binary counter)
                  └ OUT     A, %OP15
                     :
                     :
```

*Note. It is not necessary to set RWT to "1". Note that both EWT (Enable Watchdog Timer) and RWT should not be set to "1" at the same time.*

## 3.5 Serial Interface (SIO)

The 47C103/203 have a serial interface with an 8-bit buffer. 4-bit / 8-bit tramsfer mode can be selected. In the 8-bit transfer mode, data may be transmitted and received simultaneously. The serial interface is connected to the exterenal device via 3 pins (the serial port): R92 ($\overline{\text{SCK}}$), R91 (SO), and R90 (SI). The serial port is shared by port R9. For the serial port, the output latch of port R9 must be set to "1". In the transmit mode, R90 pin provides the I/O port; in the receive mode, R91 pin provides the I/O port.

### 3.5.1 Configuration of Serial Interface

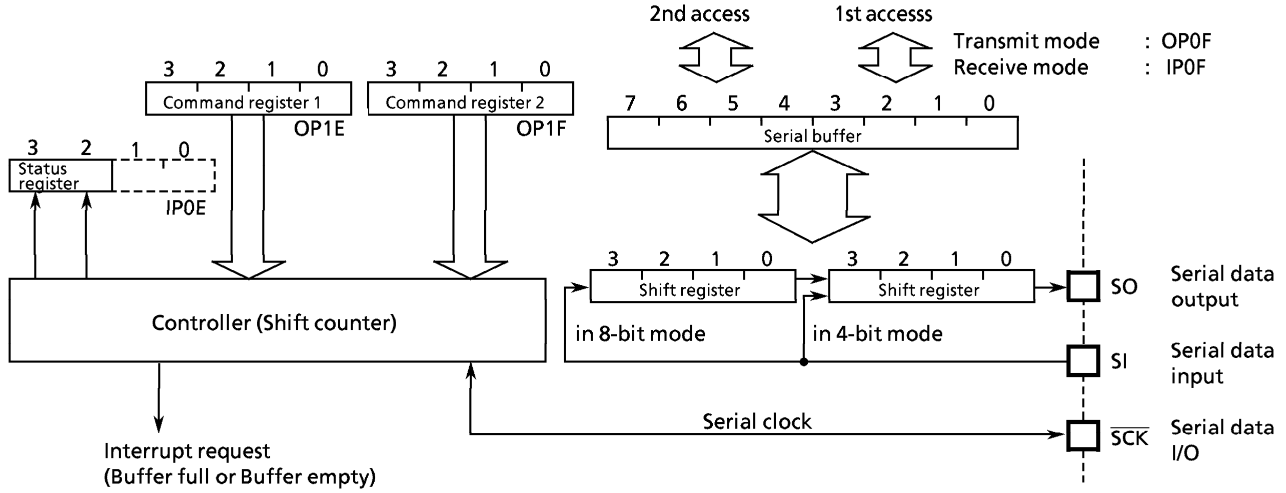


Figure 3-19. Configuration of Serial Interface

### 3.5.2 Control of Serial Interface

The serial interface is controlled by command registers (OP1E, OP1F). The operating states of the serial interface can be monitored by the status register (IP0E).

Serial interface status register (Port address IP0E).

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| SIOF | SEF | | (HOLD) |

| SIOF | Monitor serial transfer operation state |
|---|---|
| 0 : | Transfer is terminated |
| 1 : | Transfer is in progress |

| SEF | Monitors shift operation status |
|---|---|
| 0 : | Shift operation is terminated |
| 1 : | Shift operation is in progress |

Figure 3-20. Serial Interface Status Register

Serial interface control command register1
(port address OP1E) (Initial value 0001)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| INH | SBIT | CKR | |

| INH | Forcible stop of serial transfer |
|---|---|
| 0 : | Transfer continue |
| 1 : | Automatically clearing after stopped |

| SBIT | Transfer bit number |
|---|---|
| 0 : | 4-bit serial transfer |
| 1 : | 8-bit serial transfer |

| CKR | Select serial clock frequency |
|---|---|
| 00 : | fc/$2^6$ [Hz] |
| 01 : | fc/$2^7$ |
| 10 : | fc/$2^9$ |
| 11 : | fc/$2^{12}$ |

*Note 1. When setting the transfer mode, ESIO must be "0"*
*Note 2. Transmit/Receive mode : don't care*
*Transmit mode : 2M = 1*

Serial interface control command register 2
(port address OP1F) (Initial value 0000)

| 3 | 2 | 1 | 0 |
|---|---|---|---|
| ESIO | RM | LM | ECKM |

| ESIO | Instructs serial transfer start / end |
|---|---|
| 0 : | Instructs serial transfer end |
| 1 : | Instructs serial transfer start |

| RM | Select transfer mode | |
|---|---|---|
| | 4 bit transfer | 8 bit transfer |
| 0 : | Transmit mode | Transmit mode |
| 1 : | Receive mode | Transmit / Receive mode |

| LM | Select shift edge |
|---|---|
| 0 : | Shift at the trailing edge of serial clock |
| 1 : | Shift at the leading edge of serial clock |

| ECKM | Select shift clock |
|---|---|
| 0 : | Internal clock (output to $\overline{\text{SCK}}$ pin) |
| 1 : | External clock (input from $\overline{\text{SCK}}$ pin) |

Figure 3-21. Serial Interface Control Command Register

### 3.5.3 Serial clock

For the serial clock, one of the following can be selected according to the contents of the command registers:

(1) Clock source selection

a. Internal clock

The serial clock frequency is selected by command register.

The serial clock is output on the $\overline{\text{SCK}}$ pin. Note that the start of transfer, the $\overline{\text{SCK}}$ pin output goes high. When writing (transmiting) or reading (receiving) data in a program does not catch up the serial clock rate, this device stops the serial clock automatically. Additionally it provides the wait function in which the shift is not occurred until this processing is completed.

The highest transfer rate based on the internal clock is 93750 bits / second (at fc = 6 MHz).

b. External clock

The signal obtained by the clock supplied to the $\overline{\text{SCK}}$ pin from the outside is used for the serial clock. In this case, the output latch of R92 ($\overline{\text{SCK}}$) must be set to "1" beforehand. For the shift operation to be performed correctly, each of the serial clock high and low levels needs 2 instruction cycles or more to be completed.

(2) Shift edge selection

a. Leading edge

Date is shifted at the leading edge (the falling edge of $\overline{\text{SCK}}$ pin input) of the serial clock.

b. Trailing edge

Data is shifted at the trailing edge (the rising edge of $\overline{\text{SCK}}$ pin input) of the serial clock. However, in the transmit mode, the trailing-edge shift is not supported.

### 3.5.4 Transfer bit number

SBIT (bit 2 of the command register 1) can select 4-bit / 8-bit serial transfer.

(1) 4-bit serial transfer

In this mode, transmission / reception is performed on 4-bit basis. ISIO interrupt is generated every 4-bit transfer. Transmit / receive data is written / read by accessing the buffer register (OP0F/IP0F) respectively.

(2) 8-bit serial transfer

In this mode, transmission/reception is performed on 8-bit basis. ISIO interrupt is generated every 8-bit transfer. Transmit / receive data is written / read by accessing the buffer register (OP0F / IP0F) twice.

At the first access after setting transfer mode or generating the interrupt request, the write / read operation of lower 4-bit is performed to from the buffer register. At the second access, that of upper 4-bit is performed.

### 3.5.5 Transfer modes

Selection between the transmit mode and the receive mode is performed by RM (bit 2 of the command register2).

Example : Transfers the data stored in the data memory (specified by DMB, HL register pair) in the internal clock operation (fc/$2^7$). (8 bit serial transferring).

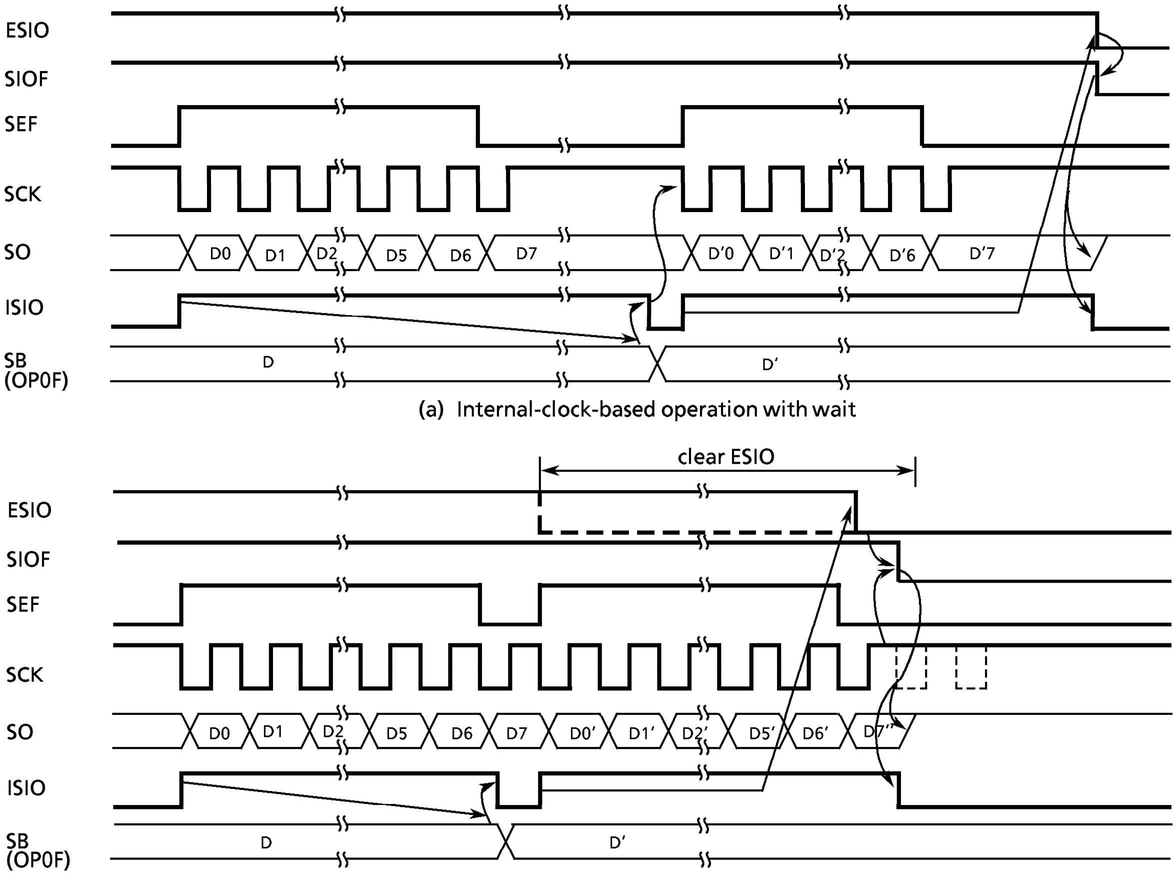
```
LD    A,    #0101B
OUT   A,    %OP1E
LD    A,    #0010B
OUT   A,    %OP1F
OUT   @HL,  %OP0F
INC   L
OUT   @HL,  %OP0F
LD    A,    #1010B
OUT   A,    %OP1F
```

(1) Transmit mode

The transmit mode is set to the command register than writes the first transmit data (4 bits or 8 bits) is written to the buffer register (OP0F). (If the transmit mode is not set, the data is not written to the buffer register). In the 8-bit transfer mode, the 8-bit data is wirtten by accessing the buffer register (OP0F) twice. The transmit data is written after the 8-bit transfer mode is set or an interrupt request occurs: the lower 4 bits are written by the first access and the upper 4 bits by the next access. Then, setting ESIO to "1" starts transmission. The transmit data is output to the SO pin in synchronization with the serial clock from the LSB side sequentially. When the LSB is output, the transmit data is moved from the buffer register to the shift register. When the buffer register becomes empty, the buffer empty interrupt (ISIO) to request for the next transmit data is generated. In the interrupt service program, when the nexttransmit data tis written o the buffer register, the interrupt request is reset.

In the operation based on the internal clock, if no more data is set after the transmission of the 4-bit or 8-bit data, the serial clock is stopped and the wait state sets in. In the operation based on the external clock, the data must be set in the buffer register by the time the next data shift operation starts. Therefore, the transfer rate is determined by the maximum delay time between the occurrence of the interrupt request and the writing of data to the buffer register by the interrupt serviced program.

To end transmission, ESIO is cleared to "0" instead of writing the next transmit data by the buffer empty interrupt service program. When ESIO is cleared,transmission stops upon termination of the currently shifted-out data. The transmission end can be known by the SIOF state (SIOF goes "0" upon transmission end). In the operation based on the external clock, ESIO must be cleared to "0" before the next data is shifted out. If ESIO is not cleared before, the transmission stops upon sending the next 4-bit or 8-bit data(dummy).



(a) Internal-clock-based operation with wait

(b) External-clock-based operation

Figure 3-22. Transmit Mode

(2) 4-bit receive mode

Data can be received when ESIO is set to "1" after setting the receive mode to the command register. The data is put from the SI pin to the shift register in synchronization with the serial clock. Then the 4/8-bit data is transferred from the shift register to the buffer register (IP0F),upon which the (buffer full)interrupt (ISIO) to reguest for readingreceived data is generated. The receive data is read from the buffer register by the interrupt service program. When the data has been read, the interrupt request is reset and the next data is put in the shift register to be transferred to the buffer register.

In the operation based on the internal clock, if the previous receive data has not been read from the buffer register at the end of capturing the next data, the serial clock is stopped and the wait operation is performed until the data has been read. In the operation based on the external clock, the shift operation is performed in synchronization with the externally-suppliedclock, so that the data must be read from the buffer register before the next receive data is transferred to it. The maximum transfer rate in the external-clock-based operation is determined by the maximum delay time between the generation of interrupt request and the reading of receive data. In the receive mode, the shift operation may be performed at either the leading edge or the trasiling edge. In the leading edge shift operation,data is captured at the leading edge of the serial clock, so that the first shift data must be put in the SI pin before the first serial clock is applied at the start of transfer.

Example: To instruct the receive start operation with the 4-bit serial transfer, internal clock and leadingedge shift (with the interrupt enable register already set).

```
LD      A,  #0000B     ;  OP1E ← 0000B (Sets the 4-bit serial transfer)
OUT     A,  %OP1E
LD      A,  #0110B     ;  OP1F ← 0110B (Sets the receive mode)
OUT     A,  %OP1F
EI                     ;  EIF ← 1 (Enables interrupt)
LD      A,  #1110B     ;  ESIO ← 1 (Instructs reception start)
OUT     A,  %OP1F
```

To end the receive operation, ESIO must be cleared to "0". When ESIO is cleared, the completion of the transfer of the current 4-bit data to the buffer register terminates the receive operation. To confirm the end of the receive operation by program, SIOF (bit 3 of the status register) must be sensed. SIOF goes "0" upon the end of receive operation.

*Note: If the transfer modes are changed, the contents of the buffer register are lost. Therefore, the modes should not be changed until the last received data is read even after the end of reception is instructed (by clearing ESIO to "0").*

The receive operation can be terminated in one of the following approaches determined by the transfer rate:

a. When the transfer rate is sufficiently low (the external-clock-based operation):
If ESIO can be cleared to "0" before the next serial clock is applied upon occurrence of buffer full interrupt in the external-clock-based operation, ESIO is cleared to "0" by the interrupt service program, then the last received data is read.

Example: To instruct reception end when transfer rate LOW (leading edge shift).

```
LD    A,      #0110B   ;  ESIO ← 0 (The end of receive operation)
OUT   A,      %OIF
IN    %IP0F,  A        ;  Acc ← IP0F (Reads received data)
```

b. When the transfer rate is high (the internal / external clock-based operation):
If the transfer rate is high and,therefore, it is possible that the capture of the next data starts before ESIO is cleared to "0" upon acceptance of any interrupt, ESIO must be cleared to "0" by confirming that SEF (bit 2 of the status register) is set at reading the data proceeding the last data. Then, the data is read. In the interrupt serevicing following the reception of the last data, no operation is needed for termination; only the reading of the received data is performed. This method is generally employed for the internal-clock-based operations. For an external-clock-based operation, ESIO must be cleared and the received data must be read before the last data is transferred to the buffer register.

Example: To instruct reception end when transfer rate is high (the internal clock, leading-edge shift).

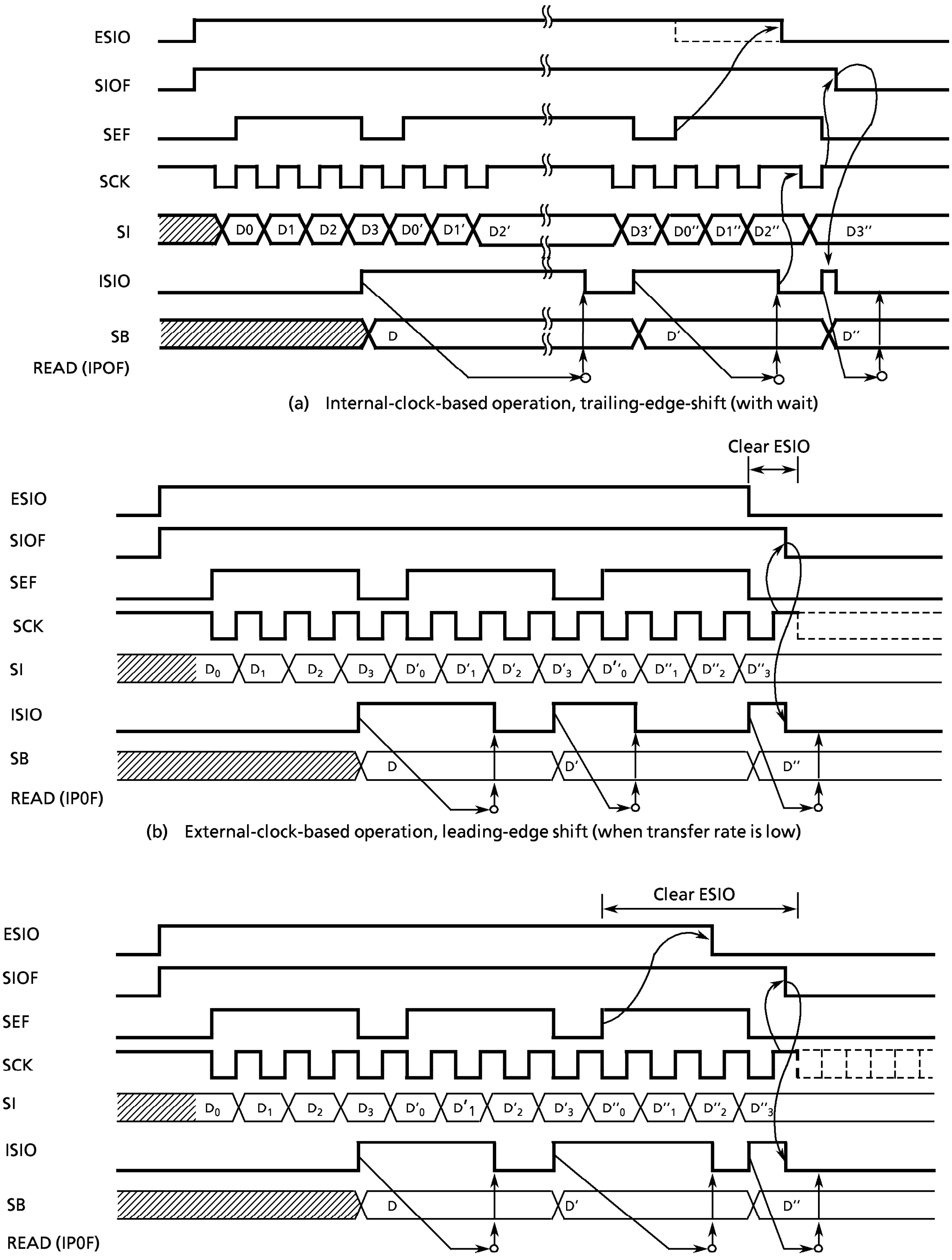
```
SSEF0 :   TEST      %IP0E,  2     ; Waits until SEF = "1"
          B         SSEF0
          LD        A,  #0110B    ; ESIO ← 0
          OUT       A,  %OP1F
          IN        %IP0F,  A     ; Acc ← IP0F (Reads received data)
```

c. One-word reception

When receiving only 1 word, ESIO is set to "1" then it is cleared to "0" after confirming that SEF has gone "1". In this case, buffer full interrupt is caused only once, so that the received data is read by the interrupt service program.

Example: To instruct the start/end of 1-word reception (the internal clock, the trailing edge shift).

```
                LD      A,  #0100B      ; OP1F ←0100B (Sets in the receive mode)
                OUT     A,  %OP1F
                EI                      ; EIF ← 1 (Enables interrupt)
                LD      A,  #1110B      ; ESIO ← 1 (Instructs reception start)5
                OUT     A,  %OP1F
   SSEF0 :      TEST    %IP0E,  2       ; Confirms that SEF = "1"
                B       SSEF0
                LD      A,  #0110B      ; ESIO ← 0 (Instructs reception end)
                OUT     A,  %OP1F
```



(a) Internal-clock-based operation, trailing-edge-shift (with wait)

(b) External-clock-based operation, leading-edge shift (when transfer rate is low)

(c) Internal-clock-based operation, leading-edge-shift (when transfer rate is high)

Figure3-23. 4-bit Receive Mode

(3) 8-bit Transmit / Receive Mode

After setting the transmition / reception mode to the command register, write first transmit data into the buffer register. Then, when "1" is set to ESIO, data transmition / reception becomes possible. The transmit data is output to the SO pin at the leading edge of serial clock and the receive data is input from the SI pin at the trailing edge. If the shift register is filled with the receive data, the data is transferred to the buffer register and ISIO (buffer full) interrupt is generated to request data read. The received data is read from the buffer register by the interrupt service program, and then write the transmit data to the buffer register.

Lower order 4 bits of both transmit and receive data are read/written from / into the buffer register by first access after setting of transmition / reception mode or generation of ISIO and higher 4 bits by next access.

In the operation based on the internal clock, SIO becomes the wait state until the received data are read out and the next data to be transmitted are written.

In the operation based on the external clock, the shift operation is synchronized with the external clock ; therefore, it is necessary to read the data received and to write data to be sent next before starting the next shift operation. The maximum transfer rate using an external clock is determined by the maximum delay time between the generation of the interrupt request and the writing of the data to be transmitted after the reading of the received data.

Also, the buffer register is used for both transmission and reception, therefore, the data must be written after reading 8 bits of receive data.

This operation is ended by clearing ESIO to "0". When ESIO is cleared, this operation is ended after transfer of the current 8 bits od data to the buffer register is completed. Programs can confirm that the operation has been completed by sensing SIOF (bit 3 of the status register) because SIOF is cleared to "0" when the operation is completed.

Example 1 : To write data to be transmitted and to instruct the transmit / receive start.

```
LD      A, #0110B     ; Sets the 8-bit transfer and serial clock frequency.
OUT     A, %OP1E
LD      A, #0110B     ; Sets the transmit / receive mode of internal clock
                        operation
OUT     A, %OP1F
LD      HL, #20H      ; OP0F←RAM[20H] (Writes lower 4-bit data to be
                        transmitted)
OUT     @HL, %OP0F
INC     L             ; OP0F←RAM[21H] (Writes upper 4-bit data to be
                        transmitted)
OUT     @HL, %OP0F
LD      A, #1110B     ; ESIO ← 1 (Instructs serial transfer start)
OUT     A, %OP1F
  ⋮                   ; Data transfer
```

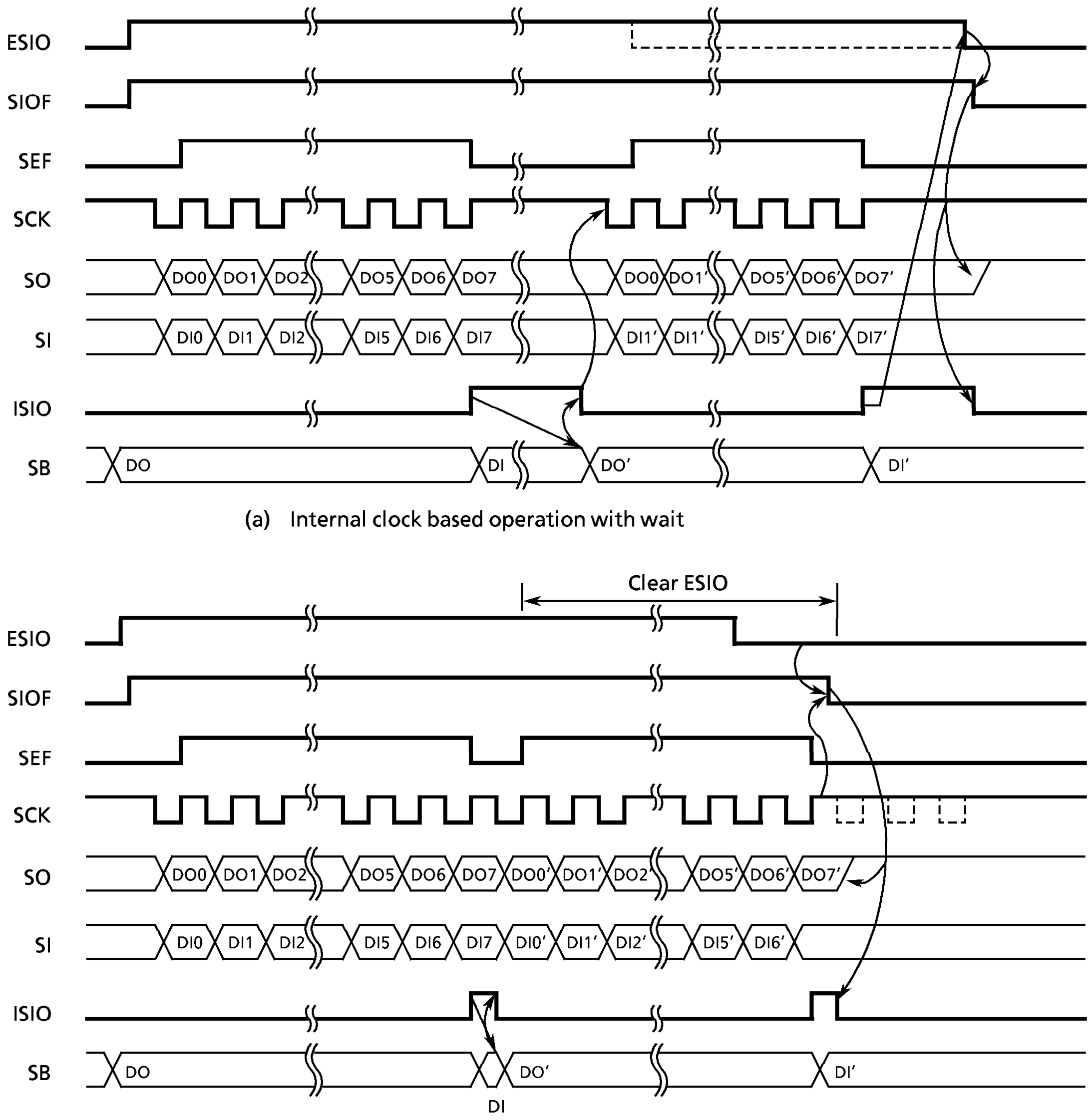
Example 2 : To read data received and to write next data to be transmitted.

```
LD      HL, #30H      ; Stores lower 4-bit data received in RAM[30H].
IN      %IP0F, @HL
INC     L             ; Stores upper 4-bit received in RAM[31H].
IN      %IP0F, @HL
LD      HL, #22H      ; Writes next lower 4-bit data to be transmitted.
OUT     @HL, %OP0F
INC     L             ; Writes next upper 4-bit data to be transmitted.
OUT     @HL, %OP0F
```

### 3.5.6 Stopping serial transfer

A serial transfer operation can be stopped forcibly.

It is stopped by setting INH (bit 3 of command register 1) to "1", clearing the shift counter. When the serial transfer is over, INH is automatically cleared to "0" with no other bits of command register affected. In the transmit mode of this case, $\overline{\text{SCK}}$ and SO output are initialized to "H" level whereas the shift register is not cleared. Therefore, after the resumption of transmit, SO holds the data just before forcible stop via the shift register until the 1st shift data comes to SO.



(a) Internal clock based operation with wait

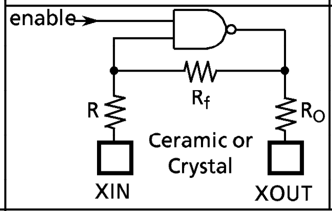
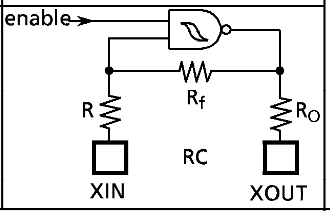
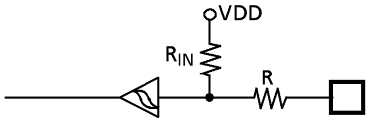
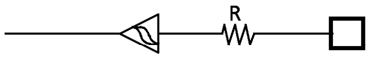
(b) External clock based operation

Figure3-24. 8-bit Transmit/Receive Mode

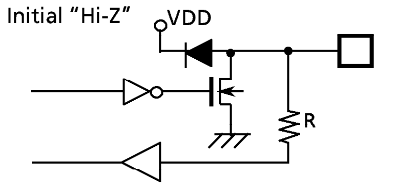
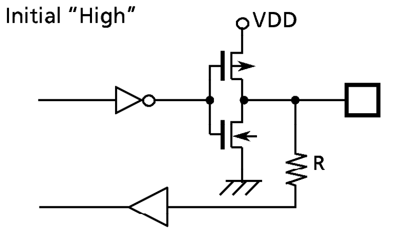
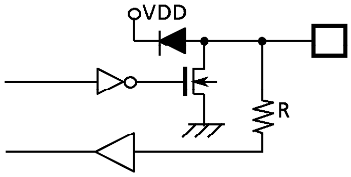
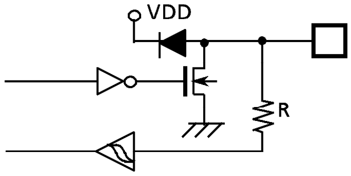
## INPUT / OUTPUT CIRCUITRY

The input / output circuitries of the 47C103/203 are shown as below, any one of the circuitries can be chosen by a code (FA, FB, FD or FE) as a mask option.

(1) Control pins

| CONTROL PIN | I/O | CIRCUITRY and CODE | | REMARKS |
|---|---|---|---|---|
| | | FA, FB | FD, FE | |
| XIN<br>XOUT | Input<br>Output |   enable<br>R<br>$R_f$<br>$R_O$<br>Ceramic or Crystal<br>XIN<br>XOUT |  | Resonator connecting pins<br>R = 1 kΩ ( typ.)<br>$R_f$ = 1.5 MΩ ( typ.)<br>$R_O$ = 2 kΩ ( typ.) |
| $\overline{\text{RESET}}$ | Input |  | | Hysteresis input<br>Pull-up resistor<br>$R_{IN}$ = 220 kΩ ( typ.)<br>R = 1 kΩ ( typ.) |
| $\overline{\text{HOLD}}$<br>($\overline{\text{KE0}}$) | Input<br>(Input) |  | | Hysteresis input<br>R = 1 kΩ (typ.) |

(2) I/O ports

| PORT | I/O | INPUT / OUTPUT CIRCUITRY and CODE | | REMARKS |
|---|---|---|---|---|
| | | FA, FD | FB, FE | |
| R4 | I/O |  |  | Sink open drain<br>or<br>push-pull output<br>R = 1 kΩ (typ.) |
| R5<br>R6<br>R7 | I/O |  | | Sink open drain output<br>Initial “Hi-Z”<br>R = 1 kΩ (typ.) |
| R8<br>R9 | I/O |  | | Sink open drain output<br>Initial “Hi-Z”<br>Hysteresis input<br>R = 1 kΩ (typ.) |

## ELECTRICAL CHARACTERISTICS

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

| PARAMETER | SYMBOL | PINS | | RATING | UNIT |
|---|---|---|---|---|---|
| Supply Voltage | $V_{DD}$ | | | – 0.3 to 6.5 | V |
| Input Voltage | $V_{IN}$ | | | – 0.3 to $V_{DD}$ + 0.3 | V |
| Output Voltage | $V_{OUT}$ | | | – 0.3 to $V_{DD}$ + 0.3 | V |
| Output Current (Per 1 pin) | $I_{OUT1}$ | Port R5, R6 | | 30 | mA |
| | $I_{OUT2}$ | Port R4 | | 15 | |
| | $I_{OUT3}$ | Ports R7, R8, R9 | | 3.2 | |
| Output Current (Total) | $\Sigma I_{OUT1}$ | Port R4, R5, R6 | | 120 | mA |
| Power Dissipation [$T_{opr}$ = 70 °C] | PD | | DIP | 300 | mW |
| | | | SOP | 180 | |
| 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 ($V_{SS}$ = 0 V, $T_{opr}$ = – 30 to 70 °C)

| PARAMETER | SYMBOL | PINS | | CONDITIONS | | Min. | Max. | UNIT |
|---|---|---|---|---|---|---|---|---|
| Supply Voltage | $V_{DD}$ | | Normal mode | Crystar or ceramic | fc = 6.0 MHz | 4.5 | 5.5 | V |
| | | | | | fc = 4.2 MHz | 2.7 | | |
| | | | | RC | fc = 2.5 MHz | 2.2 | | |
| | | | HOLD mode | – | – | 2.0 | | |
| Input High Voltage | $V_{IH1}$ | Except Hysteresis Input | | In the normal operating area | | $V_{DD} \times 0.7$ | $V_{DD}$ | V |
| | $V_{IH2}$ | Hysteresis Input | | | | $V_{DD} \times 0.75$ | | |
| | $V_{IH3}$ | | | In the HOLD mode | | $V_{DD} \times 0.9$ | | |
| Input Low Voltage | $V_{IL1}$ | Except Hysteresis Input | | In the normal operating area | | 0 | $V_{DD} \times 0.3$ | V |
| | $V_{IL2}$ | Hysteresis Input | | | | | $V_{DD} \times 0.25$ | |
| | $V_{IL3}$ | | | In the HOLD mode | | | $V_{DD} \times 0.1$ | |
| Clock Frequency | fc | XIN, XOUT | | $V_{DD}$ = 4.5 to 5.5 V | | 0.4 | 6.0 | MHz |
| | | | | $V_{DD}$ = 2.7 to 5.5 V | | | 4.2 | |
| | | | | $V_{DD}$ = 2.2 to 5.5 V (RC) | | | 2.5 | |

## D.C. CHARACTERISTICS

($V_{SS}$ = 0 V, $T_{opr}$ = – 30 to 70 °C)

| PARAMETER | SYMBOL | PINS | CONDITIONS | Min. | Typ. | Max. | UNIT |
|---|---|---|---|---|---|---|---|
| Hysteresis Voltage | $V_{HS}$ | Hysteresis Input | | – | 0.7 | – | V |
| Input Current | $I_{IN1}$ | $\overline{\text{RESET}}$, $\overline{\text{HOLD}}$ | $V_{DD}$ = 5.5 V, $V_{IN}$ = 5.5 V / 0 V | – | – | ± 2 | μA |
| | $I_{IN2}$ | Open drain output ports | | | | | |
| Input Resistance | $R_{IN}$ | $\overline{\text{RESET}}$ | | 100 | 220 | 450 | kΩ |
| Input Low Current | $I_{IL}$ | Push-pull output ports | $V_{DD}$ = 5.5 V, $V_{IN}$ = 0.4 V | – | – | – 2 | mA |
| Output Leakage Current | $I_{LO}$ | Open drain output ports | $V_{DD}$ = 5.5 V, $V_{OUT}$ = 5.5 V | – | – | 2 | μA |
| Output High Voltage | $V_{OH}$ | Push-pull output ports | $V_{DD}$ = 4.5 V, $I_{OH}$ = – 200 μA | 2.4 | – | – | V |
| | | | $V_{DD}$ = 2.2 V, $I_{OH}$ = – 5 μA | 2.0 | – | – | |
| Output Low Voltage | $V_{OL}$ | Port R7, R8, R9 | $V_{DD}$ = 4.5 V, $I_{OL}$ = 1.6 mA | – | – | 0.4 | V |
| | | | $V_{DD}$ = 2.2 V, $I_{OL}$ = 20 mA | – | – | 0.1 | |
| Output Low Current | $I_{OL1}$ | Port R5, R6 | $V_{DD}$ = 4.5 V, $V_{OL}$ = 1.0 V | – | 20 | – | mA |
| | $I_{OL2}$ | Port R4 | | – | 7 | – | |
| Supply Current (in the Normal operating mode) | $I_{DD}$ | | $V_{DD}$ = 5.5 V, fc = 4 MHz | – | 2 | 4 | mA |
| | | | $V_{DD}$ = 3.0 V, fc = 4 MHz | – | 1 | 2 | |
| | | | $V_{DD}$ = 3.0 V, fc = 400 kHz | – | 0.5 | 1 | |
| Supply Current (in the HOLD operating mode) | $I_{DDH}$ | | $V_{DD}$ = 5.5 V | – | 0.5 | 10 | μA |

*Note 1. Typ. values show those at $T_{opr}$ = 25 °C, $V_{DD}$ = 5 V.*

*Note 2. Input Current $I_{IN1}$ : The current through resistor is not included.*

*Note 3. Supply Current : $V_{IN}$ = 5.3 V / 0.2 V ($V_{DD}$ = 5.5 V), 2.8 V / 0.2 V ($V_{DD}$ = 3.0 V)*

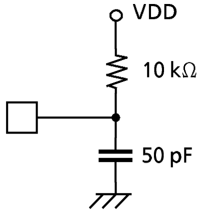
## A.C. CHARACTERISTICS

($V_{SS}$ = 0 V, $T_{opr}$ = – 30 to 70 °C)

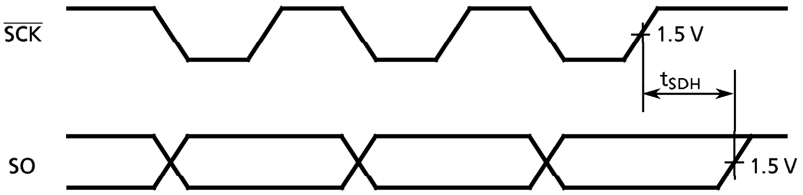
| PARAMETER | SYMBOL | CONDITIONS | | Min. | Typ. | Max. | UNIT |
|---|---|---|---|---|---|---|---|
| Instruction Cycle Time | tcy | | $V_{DD}$ = 4.5 to 5.5 V | 1.3 | – | 20 | μs |
| | | | $V_{DD}$ = 2.7 to 5.5 V | 1.9 | | | |
| | | | $V_{DD}$ = 2.2 to 5.5 V | 3.2 | | | |
| High level Clock pulse Width | $t_{WCH}$ | For external clock operation | $V_{DD}$ ≧ 2.7 V | 80 | – | – | ns |
| | | | $V_{DD}$ < 2.7 V | 160 | | | |
| Low level Clock pulse Width | $t_{WCL}$ | | $V_{DD}$ ≧ 2.7 V | 80 | | | |
| | | | $V_{DD}$ < 2.7 V | 160 | | | |
| Shift data Hold Time | $t_{SDH}$ | | | 0.5 $t_{cy}$ – 300 | – | – | ns |

*Note. Shift data Hold Time:*

External circuit for pins $\overline{SCK}$ and SO



Serial port (completed of transmission)



## RECOMMENDED OSCILLATING CONDITIONS

($V_{SS}$ = 0 V, $V_{DD}$ = 4.5 to 5.5 V, $T_{opr}$ = – 30 to 70 °C)

(1) 6 MHz

Ceramic Resonator

| | |
|---|---|
| CSA6.00MGU (MURATA) | $C_{XIN}$ = $C_{XOUT}$ = 30 pF |
| KBR-6.00MS (KYOCERA) | $C_{XIN}$ = $C_{XOUT}$ = 30 pF |

(2) 4 MHz

Ceramic Resonator

| | |
|---|---|
| CSA4.00MG (MURATA) | $C_{XIN}$ = $C_{XOUT}$ = 30 pF |
| KBR-4.00MS (KYOCERA) | $C_{XIN}$ = $C_{XOUT}$ = 30 pF |

Crystal Oscillator

204B-6F 4.0000 (TOYOCOM) $C_{XIN}$ = $C_{XOUT}$ = 20 pF

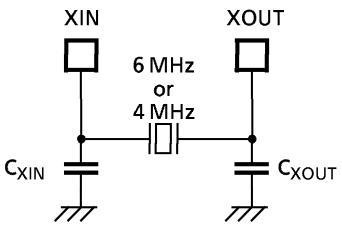
(3) 400 kHz

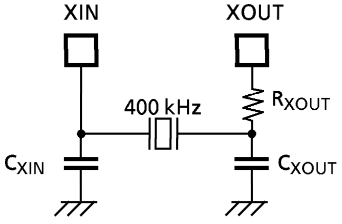
Ceramic Resonator

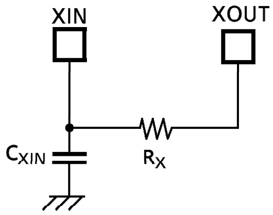
| | |
|---|---|
| CSB400B (MURATA) | $C_{XIN}$ = $C_{XOUT}$ = 220 pF, $R_{XOUT}$ = 6.8 kΩ |
| KBR-400B (KYOCERA) | $C_{XIN}$ = $C_{XOUT}$ = 100 pF, $R_{XOUT}$ = 10 kΩ |

(4) RC Oscillation (VSS = 0 V, VDD = 5.0 V, $T_{opr}$ = 25 °C)

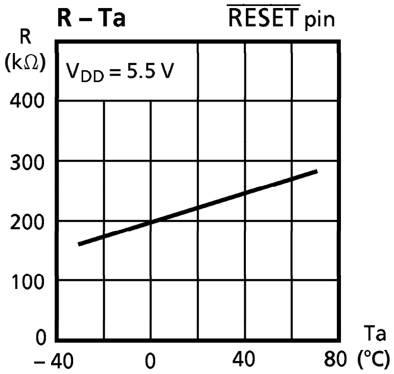
| | |
|---|---|
| 2 MHz (Typ.) | $C_{XIN}$ = 33 pF, $R_X$ = 10 kΩ |
| 400 kHz (Typ.) | $C_{XIN}$ = 100 pF, $R_X$ = 28 kΩ |

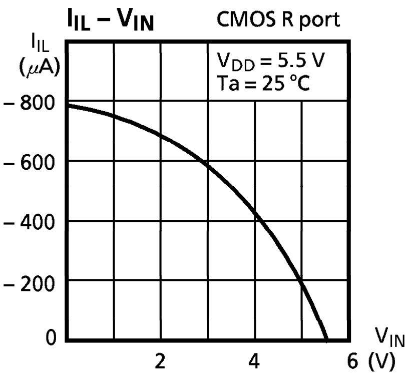


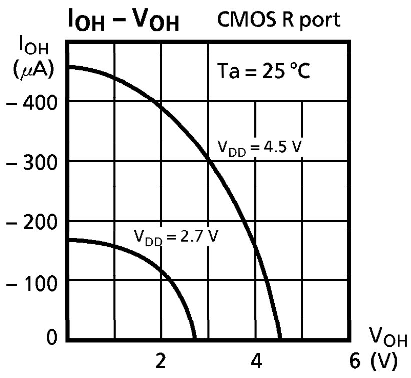


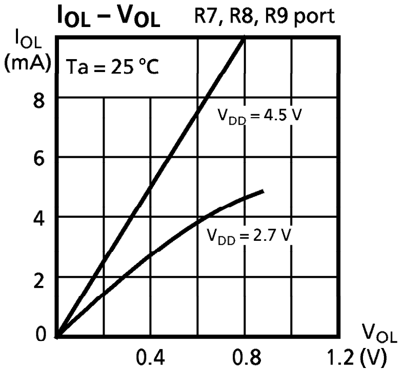


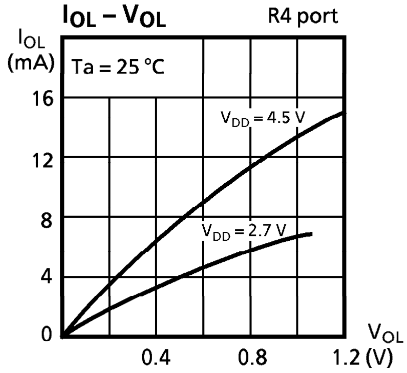
## TYPICAL CHARACTERISTICS

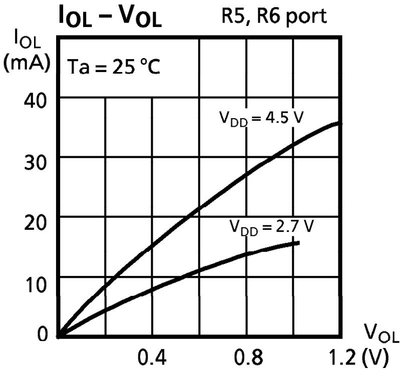

R – Ta
RESET pin
R (kΩ)
V_DD = 5.5 V
Ta (°C)

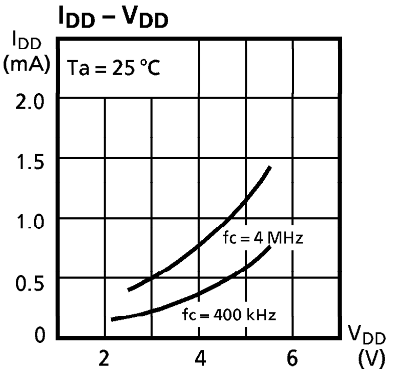

I_IL – V_IN
CMOS R port
I_IL (μA)
V_DD = 5.5 V
Ta = 25 °C
V_IN (V)

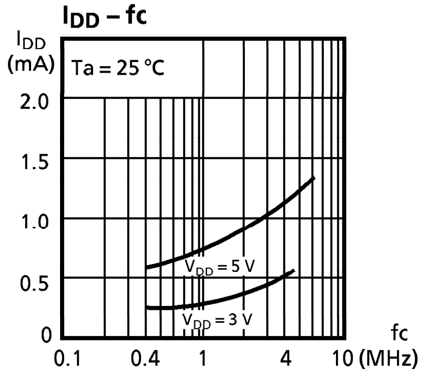

I_OH – V_OH
CMOS R port
I_OH (μA)
Ta = 25 °C
V_DD = 4.5 V
V_DD = 2.7 V
V_OH (V)

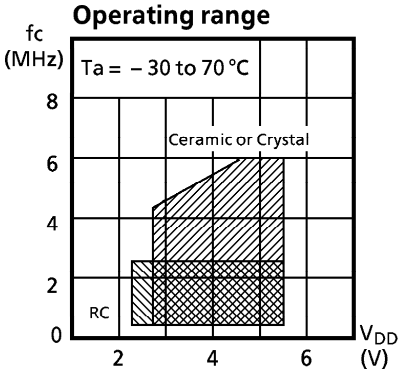

I_OL – V_OL
R7, R8, R9 port
I_OL (mA)
Ta = 25 °C
V_DD = 4.5 V
V_DD = 2.7 V
V_OL (V)

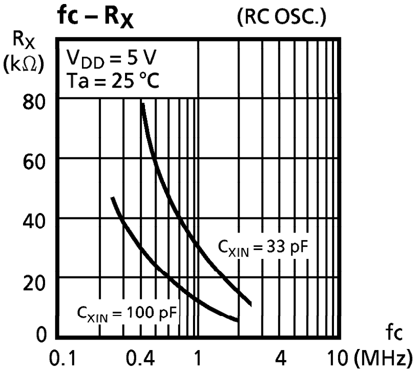

I_OL – V_OL
R4 port
I_OL (mA)
Ta = 25 °C
V_DD = 4.5 V
V_DD = 2.7 V
V_OL (V)

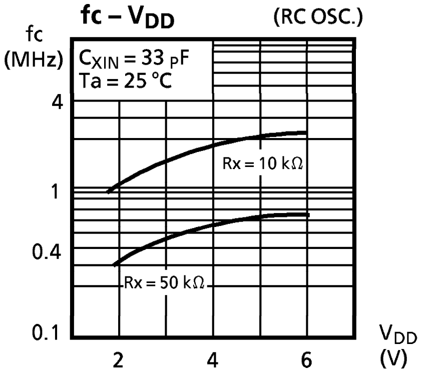

I_OL – V_OL
R5, R6 port
I_OL (mA)
Ta = 25 °C
V_DD = 4.5 V
V_DD = 2.7 V
V_OL (V)


I_DD – V_DD
I_DD (mA)
Ta = 25 °C
fc = 4 MHz
fc = 400 kHz
V_DD (V)


I_DD – fc
I_DD (mA)
Ta = 25 °C
V_DD = 5 V
V_DD = 3 V
fc (MHz)


Operating range
fc (MHz)
Ta = – 30 to 70 °C
Ceramic or Crystal
RC
V_DD (V)


fc – R_X
(RC OSC.)
R_X (kΩ)
V_DD = 5 V
Ta = 25 °C
C_XIN = 33 pF
C_XIN = 100 pF
fc (MHz)


fc – V_DD
(RC OSC.)
fc (MHz)
C_XIN = 33 pF
Ta = 25 °C
Rx = 10 kΩ
Rx = 50 kΩ
V_DD (V)