CHAPTER 5 PERIPHERAL HARDWARE FUNCTIONS

5.1 Digital I/O Ports

The uPD75238 contains digital I/O ports, port 0 to port 15.

The uPD75238 employs memory-mapped I/O. This means that all I/O ports are mapped to data memory space.

All data memory manipulation instructions are applicable to the ports of the uPD75238, enabling a wide range of control; 8-bit I/O and bit manipulation on a specified bit are enabled as well as 4-bit I/O.

Fig. 5-1 Data Memory Address Assigned to Digital I/O Ports

Address	3	2	1	0	Symbol
FF0H	P03	P02	P01	P00	PORT0
FF1H	P13	P12	P11	P10	PORT1
FF2H	P23	P22	P21	P20	PORT2
FF3H	P33	P32	P31	P30	PORT3
FF4H	P43	P42	P41	P40	PORT4
FF5H	P53	P52	P51	P50	PORT5
FF6H	P63	P62	P61	P60	PORT6
FF7H	P73	P72	P71	P70	PORT7
FF8H	P83	P82	P81	P80	PORT8
FF9H	P93	P92	P91	P90	PORT9
FFAH	P103	P102	P101	P100	PORT10
FFBH	P113	P112	P111	P110	PORT11
FFCH	P123	P122	P121	P120	PORT12
FFDH	P133	P132	P131	P130	PORT13
FFEH	P143	P142	P141	P140	PORT14
FFFH	P153	P152	P151	P150	PORT15

5.1.1 Types, features, and configurations of digital I/O ports

Table 5-1 lists the types of digital I/O ports.

Table 5-1 Types and Features of Digital I/O Ports

Port (pin name)	Function	Operation and feature	Remarks
PORT0	4-bit input	Allows input and test at any time regardless of the operation modes of the other functions with which the pins are shared.	Also used as INT4, $\overline{SCK0}$, SO0/SB0, and SI0/SB1 pins
PORT1			Also used as INT0-INT2 and TI0 pins
PORT8			Also used as PPO, $\overline{SCK1}$, SO1, and SI1
PORT9			Also used as AN4 to AN7
PORT3	4-bit I/O	Allows input or output mode setting bit-wise.	–
PORT6			
PORT2		Allows input or output mode setting in 4-bit units. Ports 6 and 7 can be paired for 8-bit data I/O.	–
PORT7			
PORT4	4-bit I/O (N-ch open-drain)	Allows input or output mode setting in 4-bit units. Ports 4 and 5 can be paired for 8-bit data I/O.	An internal pull-up resistor can be provided bit by bit by mask option.
PORT5			
PORT9	4-bit I/O	Allows input or output mode setting in 4-bit units.	An internal pull-down resistor can be provided bit by bit by mask option.
PORT10	4-bit output (N-ch open-drain)	Allows 4-bit output. Ports 10 and 11, ports 12 and 13, and ports 14 and 15 can be paired, respectively, for 8-bit data output.	Also used as S0-S23 pins. An internal pull-down resistor can be provided bit by bit by mask option. The uPD75P238 contains pull-down resistors for S0 to S8, and does not have a mask option.
PORT11			
PORT12			
PORT13			
PORT14			
PORT15 (PORTH)			

Ports 3 to 5 and 15, P142, and P143 can directly drive the LED.

Ports 4 and 5 have middle-voltage outputs in the N-ch transistors to allow effective interface with peripheral LSI devices having different power supply voltages.

In the uPD75236, uPD75237, and uPD75238, a pull-up resistor can be provided for ports 4 and 5, and a pull-down resistor can be provided for port 7 and ports 10 to 15 bit by bit.

Figures 5-2 to 5-11 show the configurations of the ports.

Fig. 5-2 Configurations of Ports 0, 1, and 8

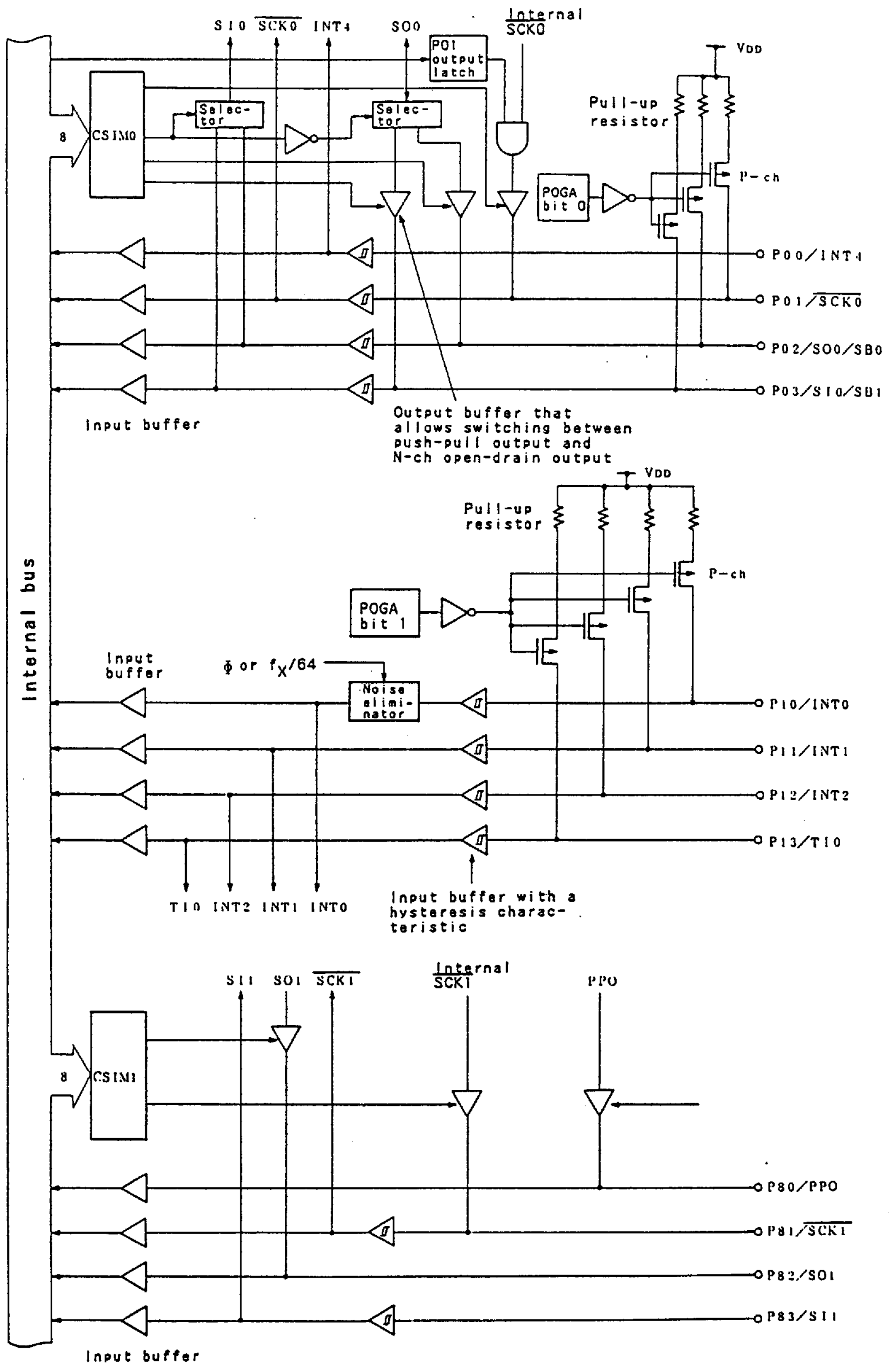


Fig. 5-3 Configurations of Ports 3n and 6n (n = 0 to 3)

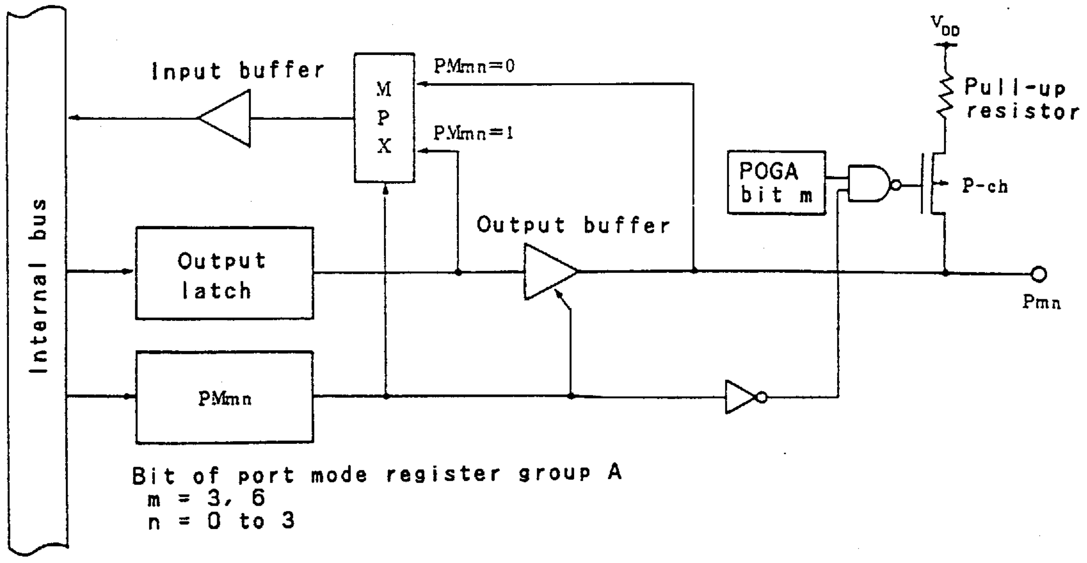


Fig. 5-4 Configuration of Port 2

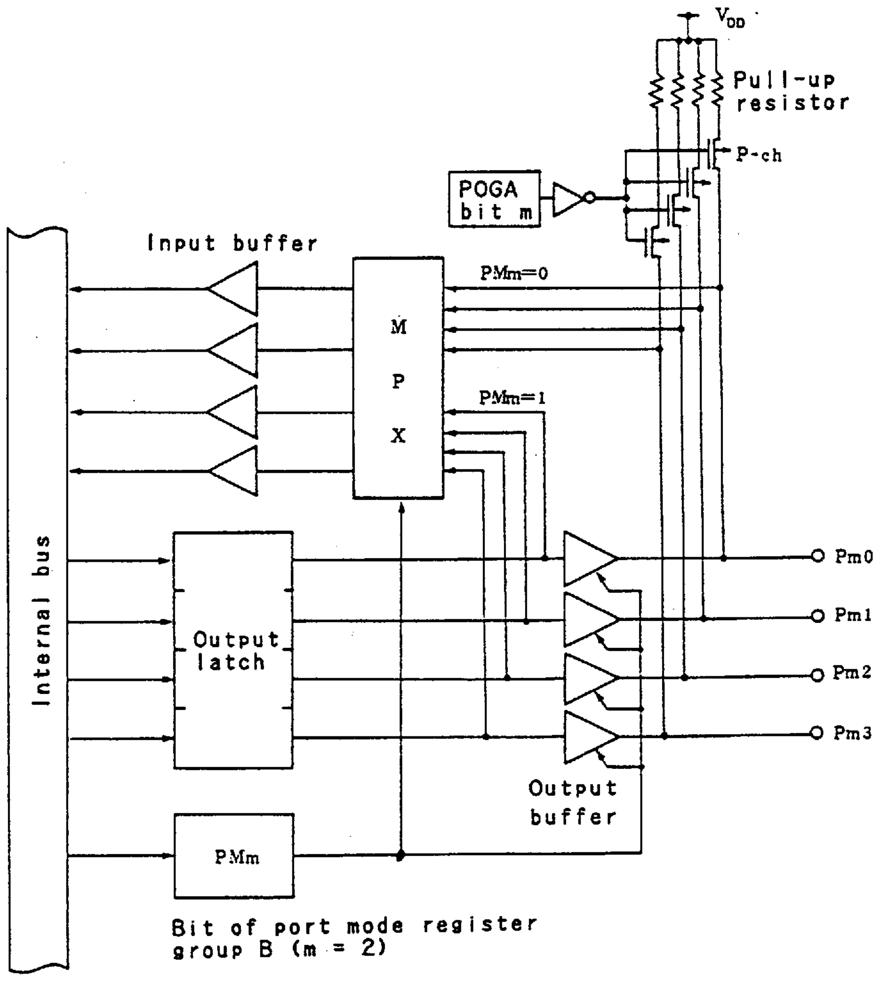


Fig. 5-5 Configurations of Ports 4 and 5

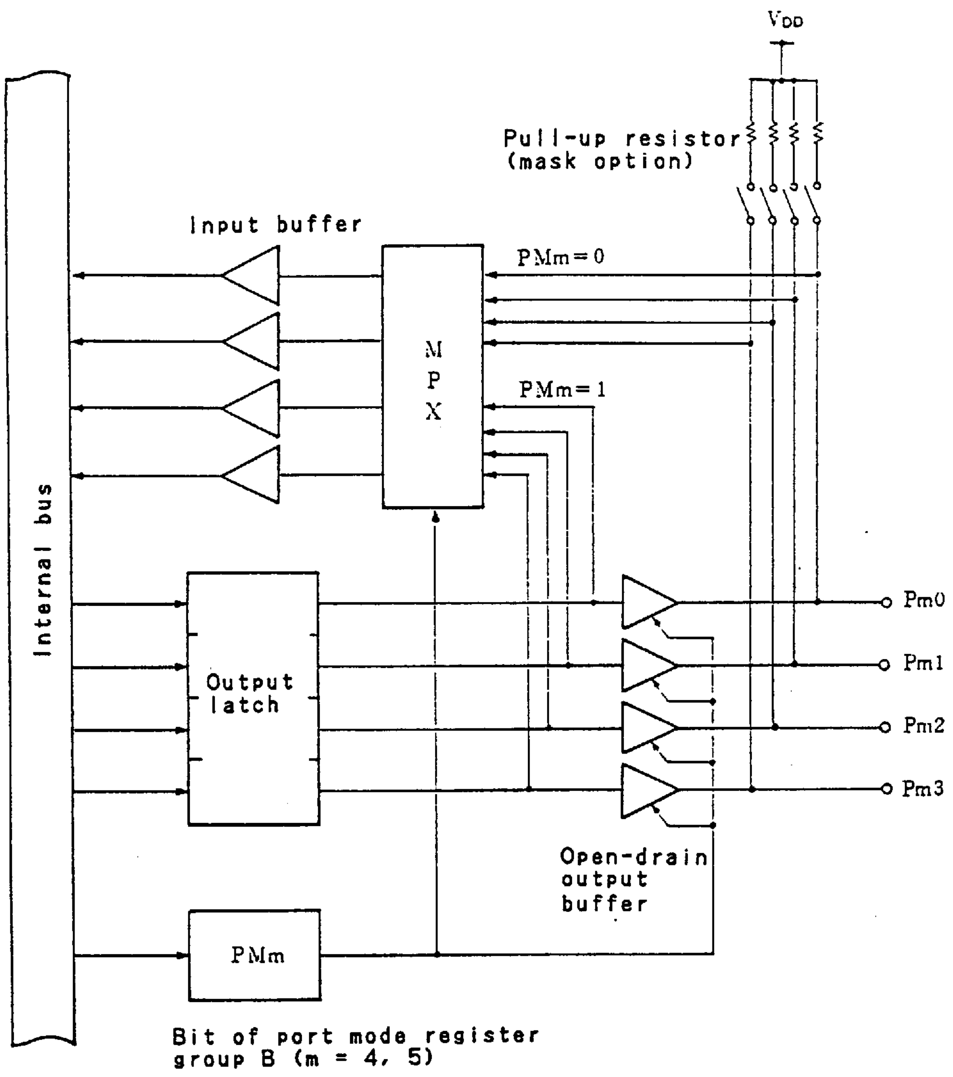


Fig. 5-6 Configuration of Port 7

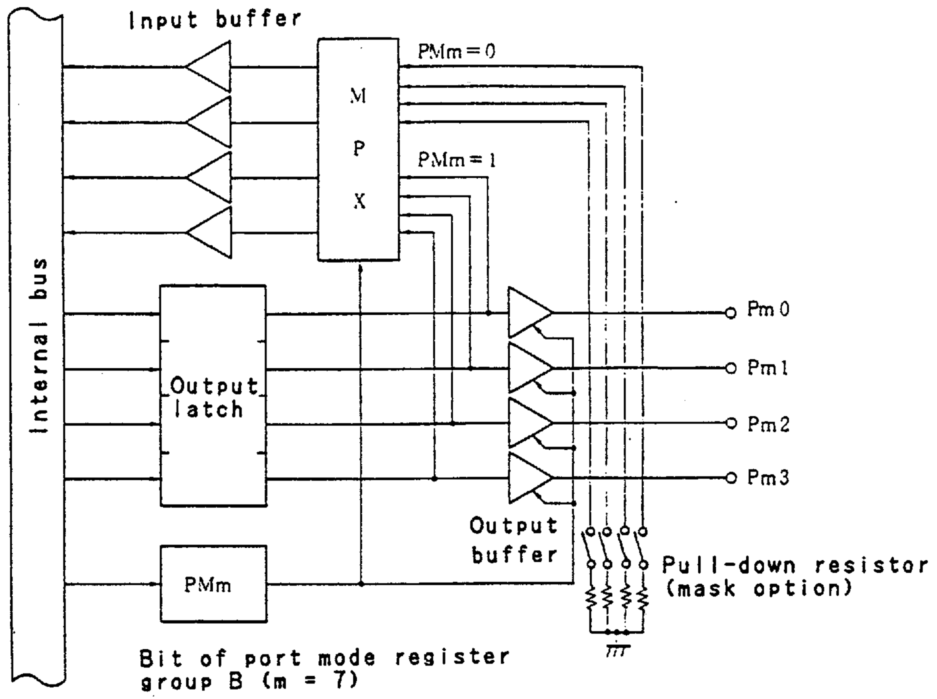


Fig. 5-7 Configuration of Port 9

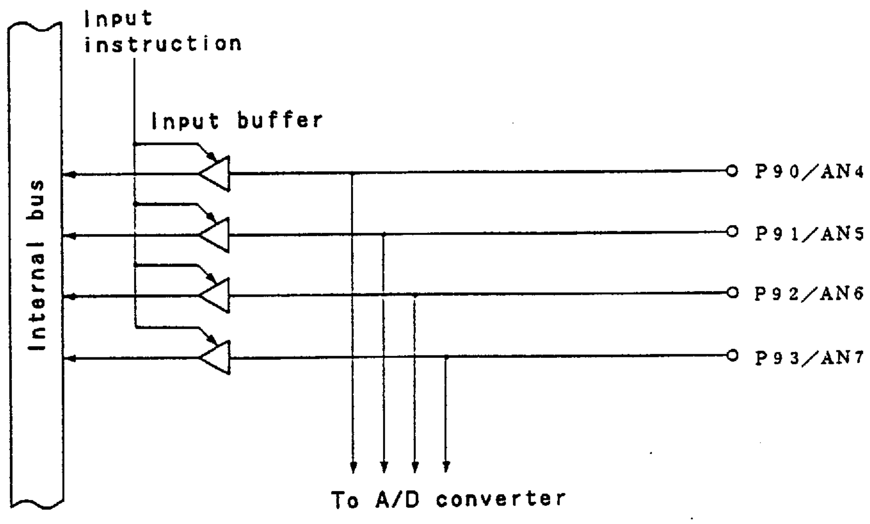
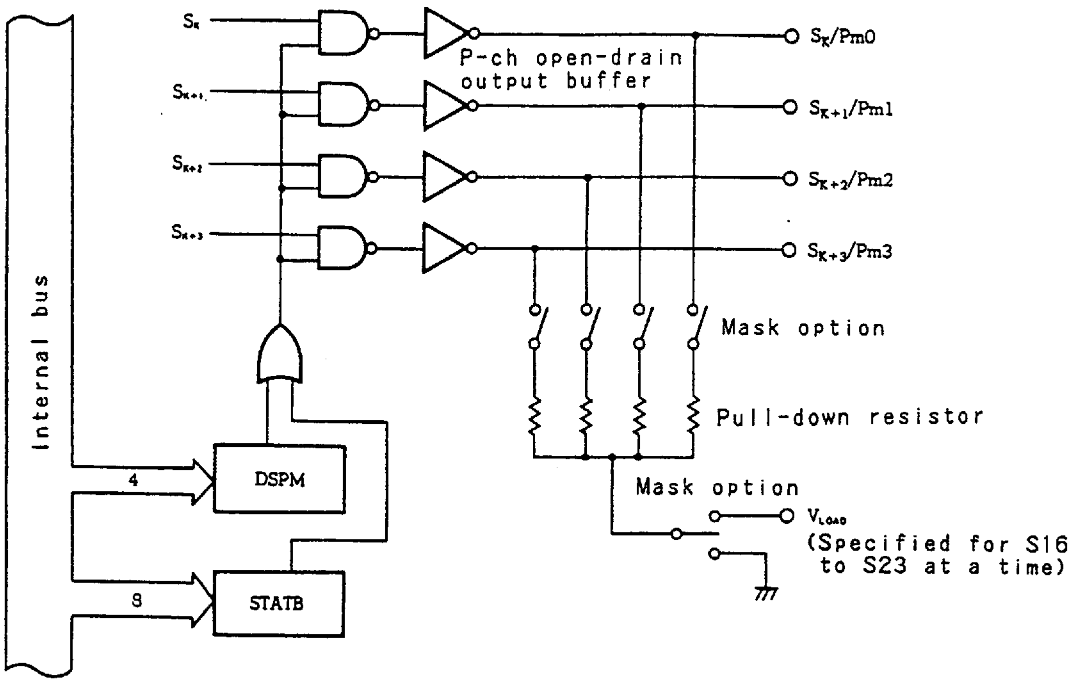


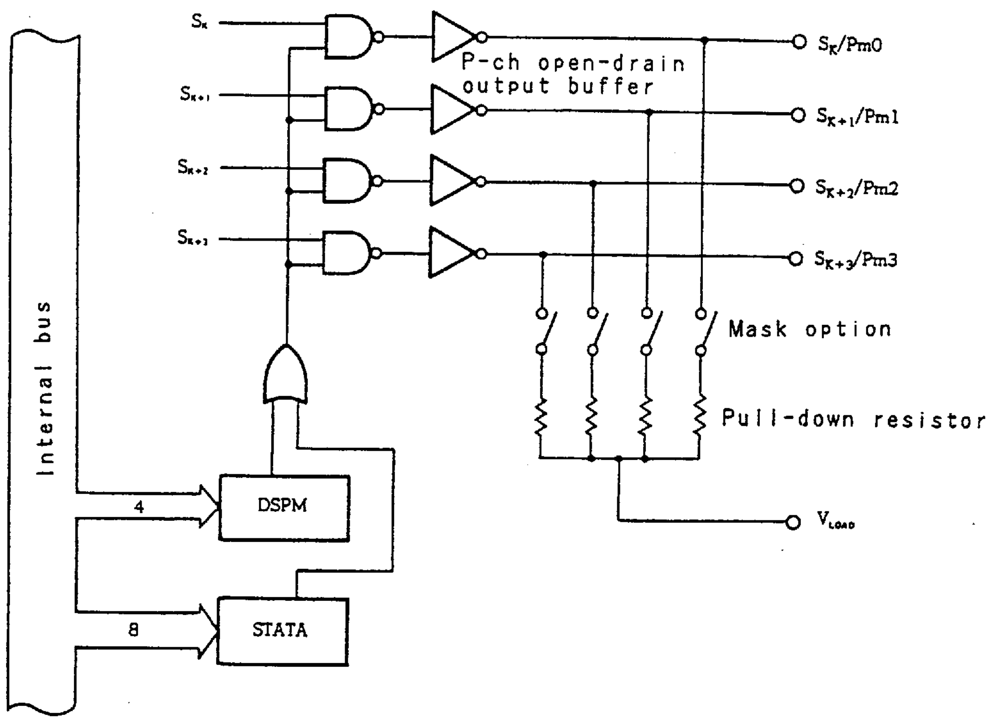
Fig. 5-8 Configurations of Ports 10 and 11



Remarks 1. Port 10: K = 16, m = 10

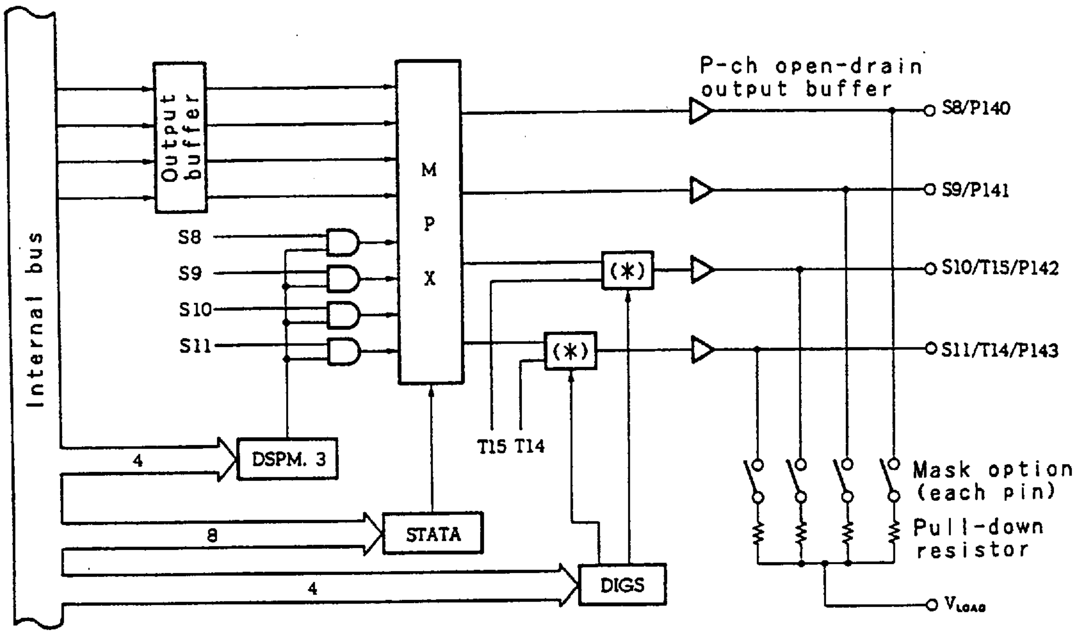
2. Port 11: K = 20, m = 11

Fig. 5-9 Configurations of Ports 12 and 13



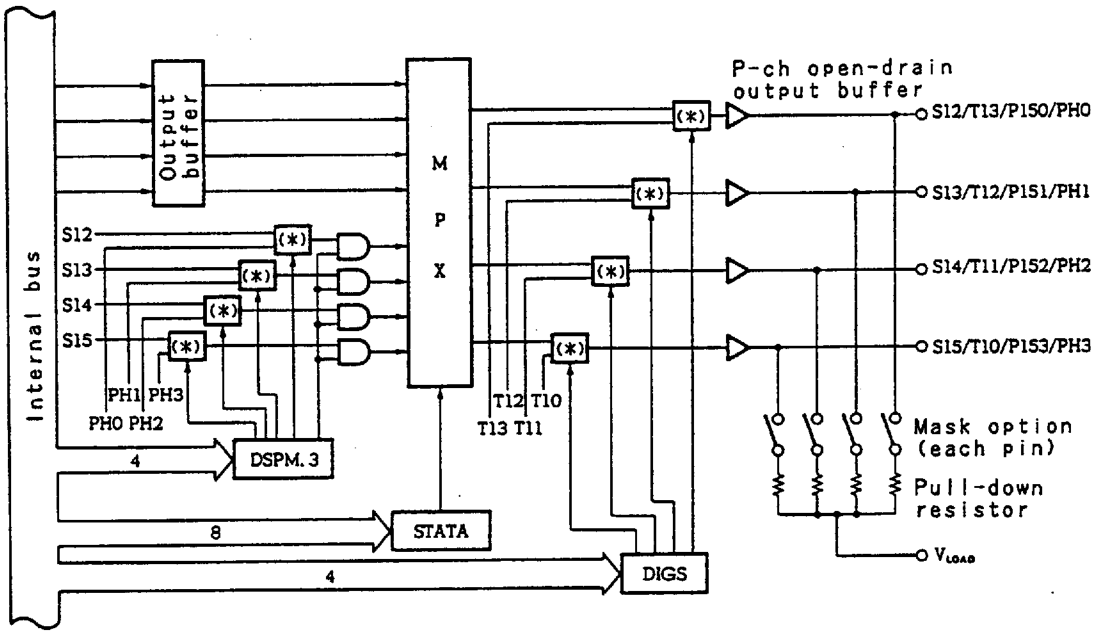
Remarks 1. Port 12: K = 0, m = 12
2. Port 13: K = 4, m = 13

Fig. 5-10 Configuration of Port 14



* Selector

Fig. 5-11 Configurations of Ports 15 and H



* Selector

5.1.2 I/O mode setting

The I/O mode of each I/O port is set by the port mode register as shown in Figure 5-12. The I/O modes of ports 3 and 6 can be set bit by bit by the port mode register group A (PMGA). The I/O modes of ports 2, 4, 5, and 7 can be set in units of four bits by the PMGB.

Each port functions as an input port when the corresponding bit of the port mode register is set to 0, and functions as an output port when the same corresponding bit is set to 1.

When the output mode is selected by the port mode register, the contents of output latch are output on the output pins, and so the contents of output latch must be changed to a desired value before the output mode is set.

An 8-bit memory manipulation instruction is used to set port mode register group A or B.

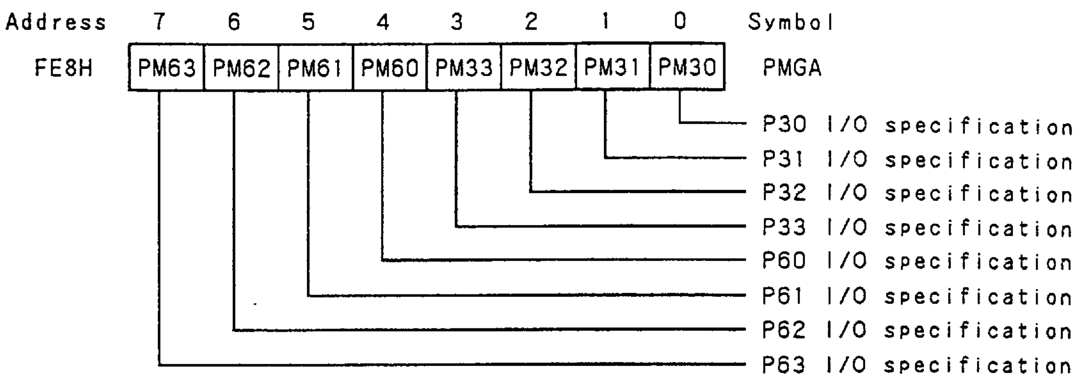
A $\overline{\text{RESET}}$ signal occurrence clears all bits of each port mode register to 0. This means that the output buffers are set off, and all ports are placed in the input mode.

Example: P30, P31, P62, and P63 are used as input pins, and P32, P33, P60, and P61 are used as output pins.

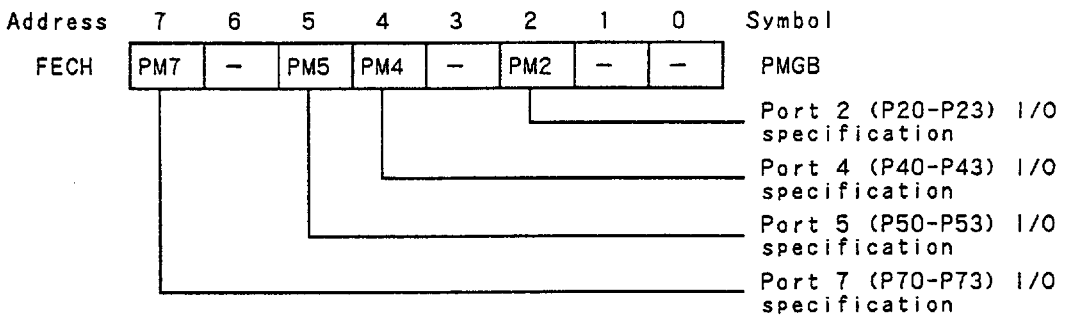
```
CLR1   MBE          ; Or SEL MB15
MOV    XA,#3CH
MOV    PMGA,XA
```

Fig. 5-12 Formats of Port Mode Registers

Port mode register group A



Port mode register group B



Address
7
6
5
4
3
2
1
0
Symbol
FECH
PM7
–
PM5
PM4
–
PM2
–
–
PMGB
Port 2 (P20-P23) I/O specification
Port 4 (P40-P43) I/O specification
Port 5 (P50-P53) I/O specification
Port 7 (P70-P73) I/O specification

	Specification
0	Input mode (output buffer off)
1	Output mode (output buffer on)

5.1.3 Digital I/O port manipulation instructions

All I/O ports contained in the uPD75238 are mapped to data memory space, so that all data memory manipulation instructions can be used. Table 5-2 lists the instructions that are particularly useful for I/O pin manipulation and their application ranges.

(1) Bit manipulation instructions

For all digital I/O ports, specific address bit direct addressing (fmem.bit) and specific address bit register indirect addressing (pmem.@L) can be used. This means that bit manipulation can be freely performed for these ports regardless of MBE and MBS settings.

Example: P50 is ORed with P81, then the result is output to P61.

```
MOV1 CY, PORT5.0    ; CY ← P50
OR1  CY, PORT8.1    ; CY ← CY ∨ P81
MOV1 PORT6.1,CY     ; P61 ← CY
```

(2) 4-bit manipulation instructions

All 4-bit manipulation instructions including the IN, OUT, MOV, XCH, ADDS, and INCS instructions can be used. However, before these instructions can be executed, memory bank 15 must be selected.

Example 1: The contents of the accumulator are output to port 3.

```
SEL   MB15          ; Or CLR1 MBE
OUT   PORT3,A
```

Example 2: The value of the accumulator is added to the data output on port 5, then the result is output.

```
SET1  MBE
SEL   MB15
MOV   HL,#PORT5
ADDS  A,@HL         ; A ← A+PORT5
NOP
MOV   @HL,A         ; PORT5 ← A
```

Example 3: Whether the data on port 4 is greater than the value of the accumulator is tested.

```
SET1   MBE
SEL    MB15
MOV    HL,#PORT4
SUBS   A,@HL          ; A < PORT4
BR     NO             ; NO
                      ; YES
```

(3) 8-bit manipulation instructions

The MOV, XCH, and SKE instructions as well as the IN and OUT instructions can be used for the ports which allow 8-bit manipulation. As with 4-bit manipulation, memory bank 15 must be selected in advance.

Example: The data contained in the BC register pair is output on the output port specified by 8-bit data applied to ports 4 and 5.

```
SET1   MBE
SEL    MB15
IN     XA,PORT4    ; XA ← ports 5,4
MOV    HL,XA       ; HL ← XA
MOV    XA,BC       ; XA ← BC
MOV    @HL,XA      ; Port (L) ← XA
```

Table 5-2 I/O Pin Manipulation Instructions

PORT	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
IN A,PORTn(*1)	o	o	o	o	o	o	o	o	o	o	–	–	–	–	–	–
IN XA,PORTn(*1)	–	–	–	–	o	o	o	o	–	–	–	–	–	–	–	–
OUT PORTn.A(*1)	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
OUT PORTn.XA(*1)	–	–	–	–	o	o	o	o	–	–	o	o	o	o	o	o
SET1 PORTn.bit	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
SET1 PORTn.@L(*2)	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
CLR1 PORTn.bit	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
CLR1 PORTn.@L(*2)	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
SKT PORTn.bit	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
SKT PORTn.@L(*2)	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
SKF PORTn.bit	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
SKF PORTn.@L(*2)	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
MOV1 CY,PORTn.bit	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
MOV1 CY,PORTn.@L(*2)	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
MOV1 PORTn.bit,CY	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
MOV1 PORTn.@L,CY(*2)	–	–	o	o	o	o	o	o	–	–	o	o	o	o	o	o
AND1 CY,PORTn.bit	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
AND1 CY,PORTn.@L(*2)	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
OR1 CY,PORTn.bit	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
OR1 CY,PORTn.@L(*2)	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
XOR1 CY,PORTn.bit	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-
XOR1 CY,PORTn.@L(*2)	o	o	o	o	o	o	o	o	o	o	-	-	-	-	-	-

*1 MBE = 0 or (MBE = 1, MBS = 15) must be set before execution.

*2 The low-order two bits of an address and bit address are indirectly specified using the L register.

5.1.4 Digital I/O port operation

When a data memory manipulation instruction is executed for a digital I/O port, the operation of the port and pins depends on the I/O mode setting (Table 5-3). This is because data taken in on the internal bus is the data input from the pins in the input mode, or the output latch data in the output mode, as obvious from the configurations of I/O ports.

(1) Operation when the input mode is set

Data from each pin is manipulated when a test instruction such as the SKT instruction, a bit input instruction by the MOV1 instruction, or an instruction for taking in port data on the internal bus in units of four or eight bits (such as an IN, OUT, arithmetic/logical or comparison instruction) is executed.

When an instruction (the OUT or MOV instruction) is executed to transfer the contents of the accumulator to a port in units of four or eight bits, the data of the accumulator is latched in the output latch, with the output buffers kept off.

When the XCH instruction is executed, the data on each pin is loaded into the accumulator, and the data in the accumulator is latched in the output latch, with the output buffers kept off.

When the INCS instruction is executed, the 4-bit data existing on the pins plus 1 is latched in the output latch, with the output buffers kept off.

When an instruction such as the SET1, CLR1, MOV1, or SKTCLR instruction is executed to rewrite a data memory bit, the output latch data of the specified bit can be rewritten according to the instruction, but the contents of the output latches of the other bits are unpredictable.

(2) Operation when the output mode is set

When a test instruction, bit input instruction, or instruction for taking in port data on the internal bus in units of four or eight bits is executed, output latch data is manipulated.

When an instruction is executed to transfer the contents of the accumulator in units of four or eight bits, the output latch data is rewritten, and is output on the pins.

When the XCH instruction is executed, the output latch data is transferred to the accumulator. The contents of the accumulator are latched in the output latches, and are output on the pins.

When the INCS instruction is executed, the contents of the output latch incremented by 1 are latched in the output latch, and are output on the pins.

When a bit output instruction is executed, the specified bit of the output latch is rewritten, and is output on the pin.

Table 5-3 Operations by I/O Port Manipulation Instructions

Instruction	Port and pin operation	
	Input mode	Output mode
SKT $\boxed{*}$ SKF $\boxed{*}$	Pin data is tested.	Output latch data is tested.
MOV1 CY, $\boxed{*}$	Pin data is transferred to CY.	Output latch data is transferred to CY.
AND1 CY, $\boxed{*}$ OR1 CY, $\boxed{*}$ XOR1 CY, $\boxed{*}$	An operation is performed on pin data and CY.	An operation is performed on output latch data and CY.
IN A,PORTn IN XA,PORTn MOV A,@HL MOV XA,@HL	Pin data is transferred to the accumulator.	Output latch data is transferred to the accumulator.
ADDS A,@HL ADDC A,@HL SUBS A,@HL SUBC A,@HL AND A,@HL OR A,@HL XOR A,@HL	An operation is performed on pin data and the accumulator.	An operation is performed on output latch data and the accumulator.
SKE A,@HL SKE XA,@HL	Pin data is compared with the accumulator.	Output latch data is compared with the accumulator.
OUT PORTn,A OUT PORTn,XA MOV @HL, A MOV @HL,XA	Accumulator data is transferred to the output latch (with the output buffers kept off).	Accumulator data is transferred to the output latch and is output on the pins.
XCH A,PORTn XCH XA,PORTn XCH A,@HL XCH XA,@HL	Pin data is transferred to the accumulator, and accumulator data is transferred to the output latch (with the output buffers kept off).	Data is exchanged between the output latch and accumulator.
INCS PORTn INCS @HL	Pin data incremented by 1 is latched in the output latch.	Output latch data is incremented by 1.
SET1 $\boxed{*}$ CLR1 $\boxed{*}$ MOV1 $\boxed{*}$,CY SKTCLR $\boxed{*}$	The output latch data of a specified bit is rewritten, but the output latch data of the other bits is unpredictable.	The output pin state is modified according to the instruction.

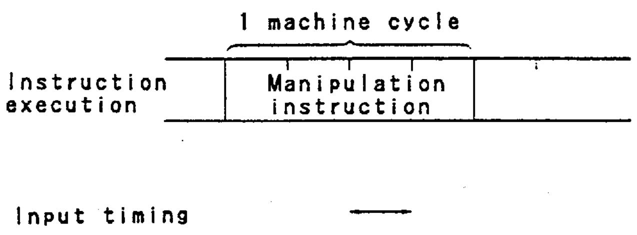
Remark: Two addressing modes PORTn.bit and PORTn.@L are indicated.

5.1.5 I/O timing of digital I/O ports

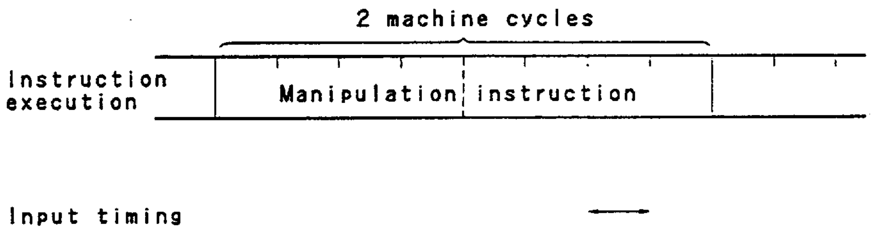
Figure 5-13 shows the timing of data output to an output latch and the timing of taking in pin data or output latch data on the internal bus.

Fig. 5-13 I/O Timing of Digital I/O Ports

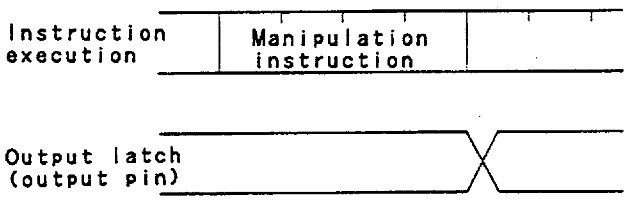
(a) When data is input by a 1-machine cycle instruction



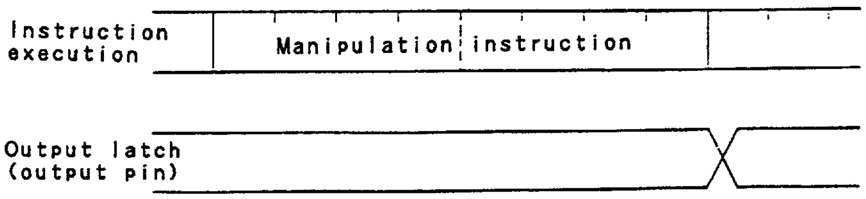
(b) When data is input by a 2-machine cycle instruction



(c) When data is latched by a 1-machine cycle instruction



(d) When data is latched by a 2-machine cycle instruction



5.1.6 Specification of internal pull-up/pull-down resistors

Each port (excluding P00 and the pins of ports 8 and 9) of the uPD75236, uPD75237, and uPD75238 can be provided with an internal pull-up or pull-down resistor. These resistors are specified as follows:

. Pull-up resistor: Software or mask option
. Pull-down resistor: Mask option

Table 5-4 shows how a pull-up or pull-down resistor is specified for each port pin.

Table 5-4 Specification of Internal Pull-Up/Pull-Down Resistors

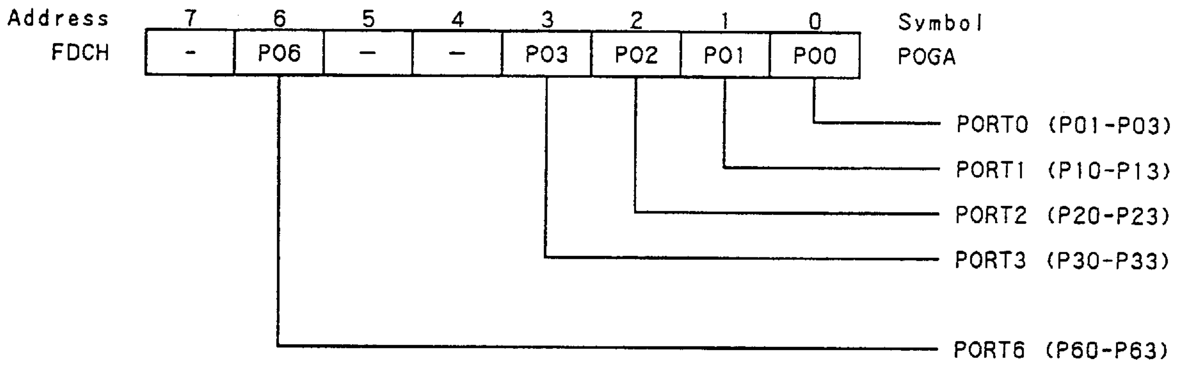
Port (pin name)	Specification of internal pull-up/pull-down resistor	Bit in POGA
Port 0 (P00-P03)(*)	Internal pull-up resistors are specified by software on a 3-bit basis.	bit 0
Port 1 (P10-P13)	Internal pull-up resistors are specified by software on a 4-bit basis.	bit 1
Port 2 (P20-P23)		bit 2
Port 3 (P30-P33)		bit 3
Port 6 (P60-P63)		bit 6
Port 4 (P40-P43)	An internal pull-up resistor is specified bit by bit by mask option.	–
Port 5 (P50-P53)		
Port 7 (P70-P73)	An internal pull-down resistor is specified bit by bit by mask option.	–
Port 10 (P100-P103)		
Port 11 (P110-P113)		
Port 12 (P120-P123)		
Port 13 (P130-P133)		
Port 14 (P140-P143)		
Port 15 (P150-P153)		

* The P00 pin cannot be provided with a pull-up resistor.

The software uses the pull-up resistor register group A (POGA) to specify whether an internal pull-up resistor is available. (See Figure 5-14.)

In ports 3 and 6, whether to use internal pull-up resistors can be specified only for the pins that have been specified as input mode pins. Output mode pins in ports 3 and 6 are not provided with pull-up resistors regardless of POGA setting.

Fig. 5-14 Format of the Pull-up Resistor Register Group A



	Specification
0	Do not use pull-up resistors.
1	Use pull-up resistors.

Caution: No pull-up resistors are specified for the uPD75P238 by mask option.

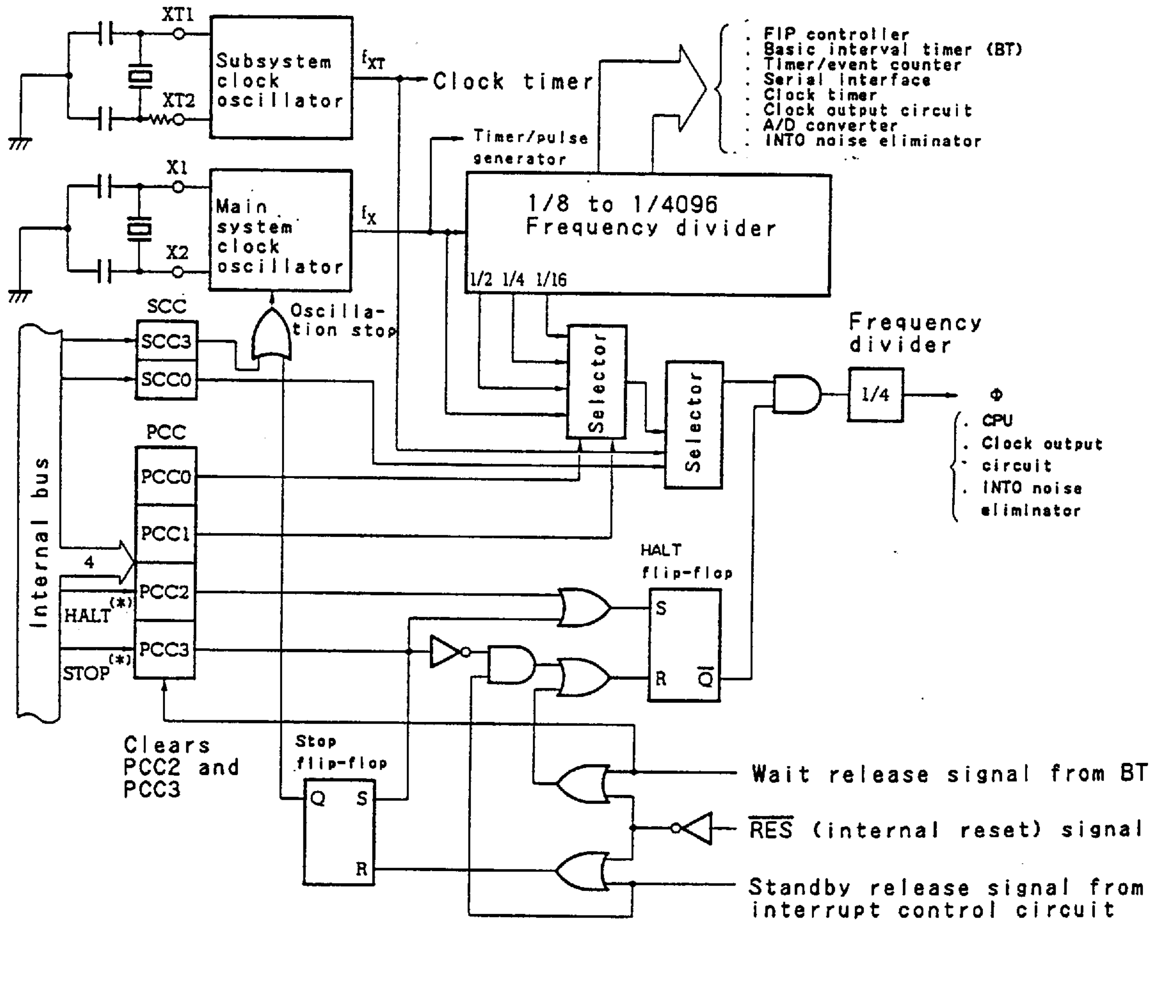
5.2 Clock Generator

The clock generator supplies various clock signals to the CPU and peripheral hardware to control the CPU operation mode.

5.2.1 Clock generator configuration

Figure 5-15 shows the configuration of the clock generator.

Fig. 5-15 Block Diagram of the Clock Generator



* Instruction execution

Remarks 1. f_X: Main system clock frequency .

2. f_{XT}: Subsystem clock frequency

3. Φ: CPU clock

4. PCC: Processor clock control register

5. SCC: System clock control register

6. One CPU clock cycle (t_{CY}) is equal to one machine cycle of an instruction.

5.2.2 Functions and operations of the clock generator

The clock generator generates clocks described below, and controls the standby mode and other CPU operation modes.

. Main system clock:	f_X
. Subsystem clock:	f_{XT}
. CPU clock:	Φ
. Clocks for peripheral hardware	

The operation of the clock generator is determined by the processor clock control register (PCC) and system clock control register (SCC). The function and operation of the clock generator are described below.

(a) A $\overline{\text{RESET}}$ signal occurrence selects the lowest-speed mode (10.7 us/6.0 MHz)(Note 1) for the main system clock (PCC = 0, SCC = 0).

(b) When the main system clock is selected, the PCC can be set to select one of four CPU clocks (0.67 us, 1.33 us, 2.67 us, and 10.7 us/6.0 MHz)(Note 2).

(c) When the main system clock is selected, the two standby modes, STOP mode and HALT mode, are available.

(d) The SCC can be set to select the subsystem clock for very low-speed, low-current operation (122 us/ 32.768 kHz).

(e) When the subsystem clock is selected, main system clock generation can be stopped with the SCC. In addition, the HALT mode can be used, but the STOP mode cannot be used. (Subsystem clock generation cannot be stopped.)

(f) The clock to be supplied to peripheral hardware is produced by dividing the main system clock signal. Only to the clock timer, the subsystem clock can be directly supplied to continue the clock function.

(g) When the subsystem clock is selected, the clock timer can operate normally, but other hardware cannot be used because they operate with the main system clock.

Notes 1. 15.3 us wnen the microcomputer operates at 4.19 MHz
2. 0.95 us, 1.91 us, 3.82 us, or 15.3 us wnen the microcomputer operates at 4.19 MHz

The following explains block operations.

(1) Processor clock control register (PCC)

The PCC is a 4-bit register for selecting CPU clock Φ with the low-order two bits and for controlling the CPU operation mode with the high-order two bits. (See Figure 5-16.)

When bit 3 or bit 2 is set to 1, the standby mode is set. When the standby mode is released by the standby release signal, these bits are automatically cleared to return to the normal operation mode. (See Chapter 7 for details.)

A 4-bit memory manipulation instruction is used to set the low-order two bits of the PCC. (The high-order two bits are set to 0.)

CPU operation clocks can be switched from the main system clock to the subsystem clock regardless of the value of the low-order two bits of the PCC.

Bit 3 and bit 2 are set to 1 using the STOP instruction and HALT instruction, respectively.

The STOP instruction and HALT instruction can be executed regardless of MBE setting.

The STOP instruction can be executed only when the main system clock is used for operation.

Example 1: The machine cycle is set to 0.67 us (in operation at f_X = 6.0 MHz$^{(Note)}$).

```
SEL MB15
MOV A,#0011B
MOV PCC,A
```

Example 2: The machine cycle is set to 1.63 us (in operation at f_X = 4.91 MHz).

```
SEL MB15
MOV A,#0010B
MOV PCC,A
```

Example 3: The STOP mode is set. (A STOP instruction or HALT instruction must always be followed by an NOP instruction.)

```
STOP
NOP
```

Note: 0.95 us for the uPD75236 (in operation at f_X = 4.19 MHz)

A $\overline{\text{RESET}}$ signal occurrence clears the PCC to 0.

Fig. 5-16 Format of the Processor Clock Control Register (1/2)

(a) For uPD75236

Address	3	2	1	0	Symbol
FB3H	PCC3	PCC2	PCC1	PCC0	PCC

CPU clock select bit
(When $f_X \leqq 4.19$ MHz)

		SCC0 = 0 ( ) indicates frequency when f_X = 4.19 MHz		SCC0 = 1 ( ) indicates frequency when f_{XT} = 32.768 kHz	
		CPU clock frequency	1 machine cycle	CPU clock frequency	1 machine cycle
0	0	$\Phi = f_X/64$ (65.5 kHz)	15.3 us	$\Phi = f_{XT}/4$ (8.192 kHz)	122 us
0	1	$\Phi = f_X/16$ (262 kHz)	3.82 us	Not to be set	
1	0	$\Phi = f_X/8$ (524 kHz)	1.91 us	$\Phi = f_{XT}/4$ (8.192 kHz)	122 us
1	1	$\Phi = f_X/4$ (1.05 MHz)	0.95 us		

(When 4.19 MHz < $f_X \leqq 5.0$ MHz)

		SCC0 = 0 ( ) indicates frequency when f_X = 4.91 MHz		SCC0 = 1 ( ) indicates frequency when f_{XT} = 32.768 kHz	
		CPU clock frequency	1 machine cycle	CPU clock frequency	1 machine cycle
0	0	$\Phi = f_X/64$ (76.7 kHz)	13 us	$\Phi = f_{XT}/4$ (8.192 kHz)	122 us
0	1	$\Phi = f_X/16$ (307 kHz)	3.26 us	Not to be set	
1	0	$\Phi = f_X/8$ (614 kHz)	1.63 us	$\Phi = f_{XT}/4$ (8.192 kHz)	122 us
1	1	Not to be set		Not to be set	

Remarks 1. f_X: Output frequency from the main system clock oscillator

2. f_{XT}: Output frequency from the subsystem clock oscillator

CPU operation mode control bit

0	0	Normal operation mode
0	1	HALT mode
1	0	STOP mode
1	1	Not to be set

Caution: When f_X is 4.19 MHz < $f_X \leqq$ 5 MHz, specify the maximum speed mode = f_X/4 (PCC1 = PCC0 = 10 or 01 or 00) for the CPU clock frequency because when PCC1 = PCC0 = 11, 1 machine cycle falls short of 0.95 us, the minimum value for the standard. This means that the combination of f_X = 4.19 MHz and PCC1 = PCC0 = 11 is selected for the maximum CPU clock speed (1 machine cycle = 0.95 us).

Fig. 5-16 Format of the Processor Clock Control Register (2/2)

(b) For uPD75237, uPD75238 and uPD75P238

Address	3	2	1	0	Symbol
FB3H	PCC3	PCC2	PCC1	PCC0	PCC

CPU clock select bit
(When f_X = 6.0 MHz)

		SCC = 0 ( ) indicates frequency when f_X = 6.0 MHz		SCC = 1 ( ) indicates frequency when f_{XT} = 32.768 kHz	
		CPU clock frequency	1 machine cycle	CPU clock frequency	1 machine cycle
0	0	Φ = f_X/64 (93.7 kHz)	10.7 us	Φ = f_{XT}/4 (8.192 kHz)	122 us
0	1	Φ = f_X/16 (375 kHz)	2.67 us	Not to be set	
1	0	Φ = f_X/8 (750 kHz)	1.33 us	Φ = f_{XT}/4 (8.192 kHz)	122 us
1	1	Φ = f_X/4 (1.5 MHz)	0.67 us		

(When f_X = 4.19 MHz)

		SCC = 0 ( ) indicates frequency when f_X = 4.19 MHz		SCC = 1 ( ) indicates frequency when f_{XT} = 32.768 kHz	
		CPU clock frequency	1 machine cycle	CPU clock frequency	1 machine cycle
0	0	Φ = f_X/64 (65.5 kHz)	15.3 us	Φ = f_{XT}/4 (8.192 kHz)	122 us
0	1	Φ = f_X/16 (262 kHz)	3.82 us	Not to be set	
1	0	Φ = f_X/8 (524 kHz)	1.91 us	Φ = f_{XT}/4 (8.192 kHz)	122 us
1	1	Φ = f_X/4 (1.05 MHz)	0.95 us		

Remarks 1. f_X: Output frequency from the main system clock oscillator

2. f_{XT}: Output frequency from the subsystem clock oscillator

CPU operation mode control bit

0	0	Normal operation mode
0	1	HALT mode
1	0	STOP mode
1	1	Not to be set

(2) System clock control register (SCC)

The SCC is a 4-bit register for selecting CPU clock Φ f_{XX} with the least significant bit and for controlling the termination of main system clock generation with the most significant bit. (See Figure 5-17.)

SCC.0 and SCC.3 are located at the same data memory address, but both bits cannot be changed at the same time. Accordingly, SCC.0 and SCC.3 are set using bit manipulation instructions. SCC.0 and SCC.3 can be manipulated regardless of MBE setting.

Main system clock generation can be terminated by setting SCC.3 only when the subsystem clock is used for operation. The STOP instruction must be used to terminate main system clock generation.

A $\overline{\text{RESET}}$ signal clears the SCC to 0.

Fig. 5-17 Format of the System Clock Control Register

Address	3	2	1	0	Symbol
FB7H	SCC3	–	–	SCC0	SCC

SCC3	SCC0	CPU clock selection	Main system clock oscillation
0	0	Main system clock	Oscillation enabled
0	1	Subsystem clock	
1	0	Not to be set	
1	1	Subsystem clock	Oscillation disabled

Cautions 1. A time period of up to $1/f_{XT}$ is needed to change the system clock. This means that to terminate main system clock generation, SCC.3 must be set when the machine cycles indicated in Table 5-5 or more have elapsed after the clock is switched from the main system clock to the subsystem clock.

2. When the main system clock is used for operation, setting SCC.3 to stop clock generation does not enter the normal STOP mode.

3. When SCC.3 is set to 1, the X1 input pin is connected to V_{SS} (GND electric potential) to prevent leakage in the crystal oscillator. When an external clock is used as the main system clock, never set SCC.3 to 1.

4. When the four bits of PCC are set to 0001 (= $f_X/16$), do not set SCC.0 to 1. Before switching the main system clock to the subsystem clock, be sure to manipulate the PCC bits so other than 0001 is set. When the system operates on the subsystem clock, the PCC bits must also be other than 0001.

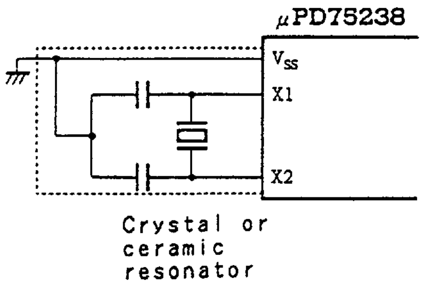
(3) System clock oscillator

(i) The main system clock oscillator operates with a crystal or a ceramic resonator (6.0 MHz standard) connected to the X1 and X2 pins.

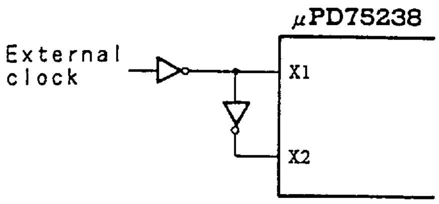
An external clock can also be input.

Fig. 5-18 External Circuit for the Main System Clock Oscillator

(a) Crystal/ceramic oscillation



(b) External clock



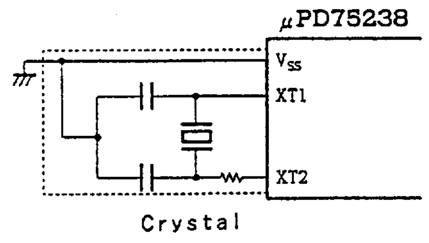
Caution: When an external clock is used, the STOP mode cannot be set. This is because the X1 pin is connected to V_{SS} in the STOP mode.

(ii) The subsystem clock oscillator operates with a crystal (32.768 kHz standard) connected to the XT1 and XT2 pins.

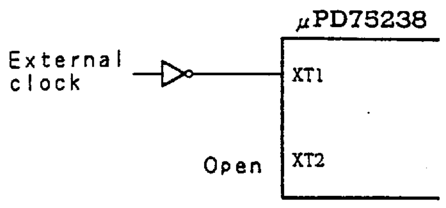
An external clock can also be input.

Fig. 5-19 External Circuit for the Subsystem Clock Oscillator

(a) Crystal oscillation



(b) External clock



Cautions 1. When the main system clock or subsystem clock oscillator is used, conform to the following guidelines when wiring at the portions surrounded by dotted lines in Figures 5-18 and 5-19 to eliminate the influence of the wiring capacity.

. The wiring must be as short as possible.

. Other signal lines must not run in these areas. Any line carrying a high fluctuating current must be kept away as far as possible.

. The grounding point of the capacitor of the oscillator must have the same potential as that of V_{SS}. It must not be grounded to ground patterns carrying a large current.

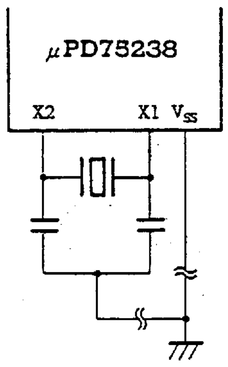
. No signal must be taken from the oscillator.

Take special care of the subsystem clock oscillator because it has low amplification to minimize current consumption.

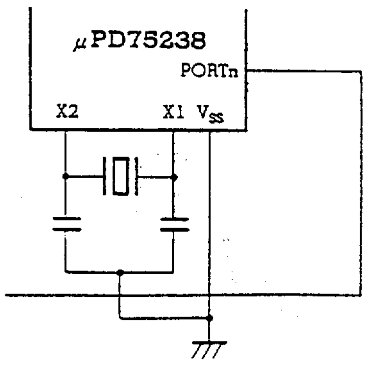
Figure 5-20 is an example of how not to connect the oscillator.

Fig. 5-20 Example of How Not to Connect the Oscillator

(a) The wiring is too long.

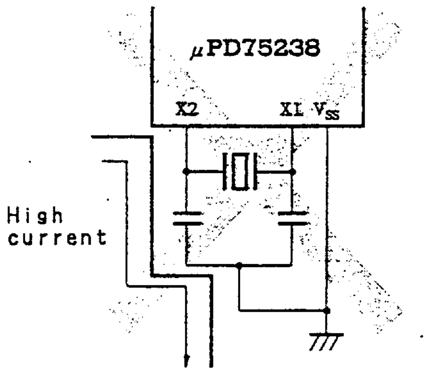


(b) The signal lines cross.

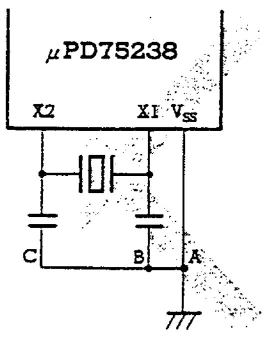


Remark: The subsystem clock user can replace X1 and X2 with XT1 and XT2 respectively. In this case, a resistor must be added to XT2 in series.

(c) A high fluctuating current is too close to the signal line.



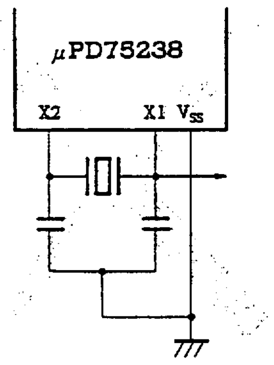
(d) The current flows through the ground line of the oscillator.
(The potential at points A, B, and C fluctuates.)



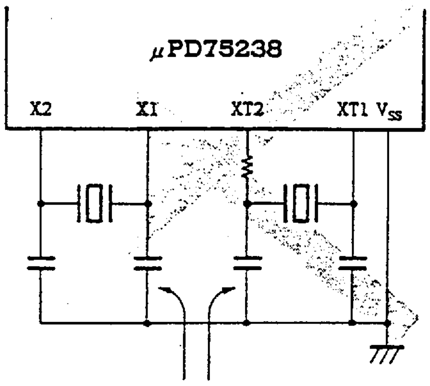
(to be continued)

Fig. 5-20 Example of How Not to Connect the Oscillator (Cont'd)

(e) A signal is taken from the oscillator.



(f) The signal lines of the main system clock and subsystem clock are parallel and adjacent to each other.

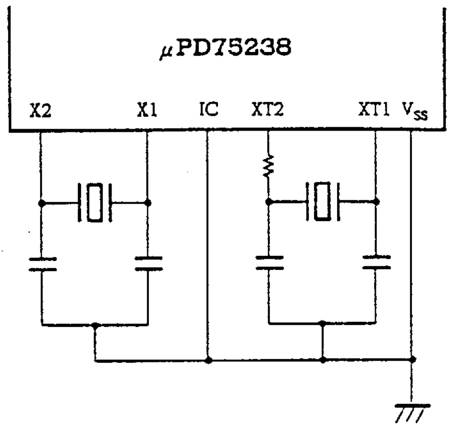


XT2 and X1 are wired in parallel. (See caution 2 for the countermeasure.)

Remark: The subsystem clock user can replace X1 and X2 with XT1 and XT2 respectively. In this case, a resistor must be added to XT2 in series.

Cautions 2. In (f) of Figure 5-20, XT2 and X1 are wired in parallel. This may cause the system to malfunction due to crosstalk noise in XT2 from X1.

To prevent this, XT2 and X1 must not be wired in parallel. It is also recommended that the IC pin placed between XT2 and X1 be connected to V_{SS}.



(4) Frequency divider

The frequency divider divides the output (f_X) of the main system clock oscillator to generate various clocks.

(5) For no use of the subsystem clock

If the subsystem clock is not required for low power consumption or clock operation, XT1 and XT2 pins are treated as follows:

XT1: Connected to V_{SS} or V_{DD}.
XT2: Open

In these states, however, a little leakage current flows through the feedback resistor of the subsystem clock oscillator when the main system clock is stopped. To prevent this, the feedback resistor can be disconnected by mask option. XT1 and XT2 pins are also treated as shown above. (Specify the resistor by mask option with the order.)

5.2.3 System clock and CPU clock setting

(1) Time required to change the system clock and CPU clock

The system clock and CPU clock can be changed by using the low-order two bits of the PCC and the least significant bit of the SCC. This switching is not performed immediately after the contents of the registers are rewritten, but the system operates with the previous clock for some machine cycles. Accordingly, after this time period, the STOP instruction must be executed or SCC.3 must be set to terminate main system clock generation.

Table 5-5 Maximum Time Required to Change the System Clock and CPU Clock

Setting before switching			Setting after switching				
SCC	PCC	PCC	SCC0 PCC1 PCC0	SCC0 PCC1 PCC0	SCC0 PCC1 PCC0	SCC0 PCC1 PCC0	SCC0 PCC1 PCC0
0	1	0	0 0 0	0 0 1	0 1 0	0 1 1	1 x x
0	0	0		1 machine cycle	1 machine cycle	1 machine cycle	$\frac{f_X}{64 f_{XT}}$ machine cycles (3 machine cycles)
	0	1	4 machine cycles		4 machine cycles	4 machine cycles	Not to be set
	1	0	8 machine cycles	8 machine cycles		8 machine cycles	$\frac{f_X}{8 f_{XT}}$ machine cycles (23 machine cycles)
	1	1	16 machine cycles	16 machine cycles	16 machine cycles		$\frac{f_X}{4 f_{XT}}$ machine cycles (46 machine cycles)
1	x	x	1 machine cycle	Not to be set	1 machine cycle	1 machine cycle	

x: Don't care

Remarks 1. CPU clock Φ is supplied to the CPU in the uPD75238. The reciprocal of this frequency is a minimum instruction time (defined as one machine cycle in this manual).

2. Time enclosed in parentheses is required when f_X = 6.0 MHz and f_{XT} = 32.768 kHz.

Cautions 1. When the four bits of PCC are set to 0001 (Φ = $f_X/16$), do not set SCC.0 to 1. Before switching the main system clock to the subsystem clock, be sure to manipulate the PCC bits so other than 0001 is set. When the system operates on the subsystem clock, the PCC bits must also be other than 0001.

2. The fluctuation of the ambient temperature around an oscillator and the performance of a load capacity changes f_X and f_{XT}. In particular, when f_X is higher than the nominal value or f_{XT} is lower than the nominal value, the machine cycles calculated by $f_X/64f_{XT}$, $f_X/8f_{XT}$, and $f_X/4f_{XT}$ in Table 5-5 are longer than the machine cycle calculated by the nominal values of f_X and f_{XT}.

Therefore, the wait time required to change the system clock and CPU clock should be longer than the machine cycle calculated by the nominal values of f_X and f_{XT}.

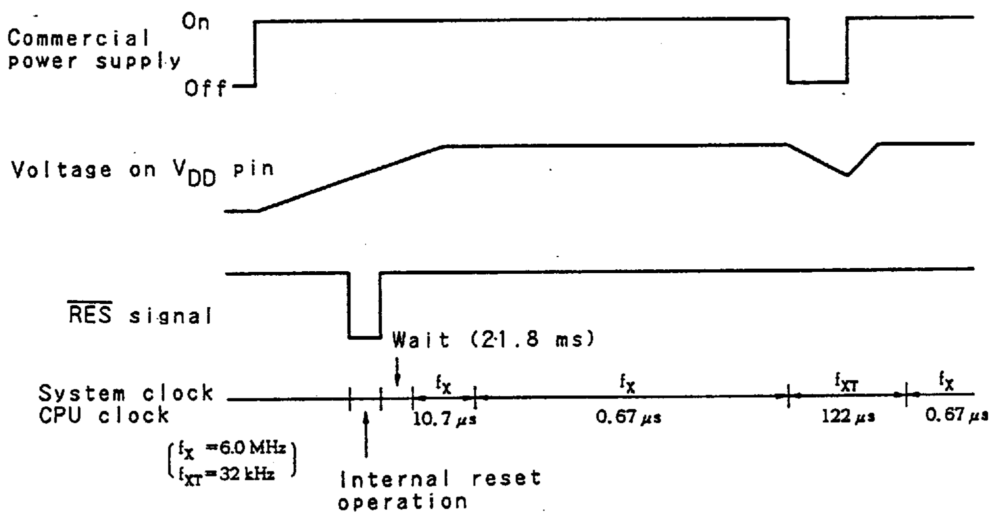
Example:

```
                       |  SEL   MB15       ; SCC = 0, PCC = 0011
                       |  MOV   A,#0000
   0.67 us/6.0 MHz     |  MOV   PCC,A      ; PCC <- 0000
                       |  MOV ┐
                       |  BR  ┘ 16 machine cycles
                       |   :
              ---------+------------------------
   10.7 us/6.0 MHz     |   :
```

(2) Procedure for changing the system clock and CPU clock

The procedure for changing the system clock and CPU clock is explained using Figure 5-21.

Fig. 5-21 Changing the System Clock and CPU Clock



① A $\overline{RESET}$ signal occurrence starts CPU operation at the lowest speed of the main system clock (10.7 us/6.0 MHz)(Note 1) after a wait time (21.8 ms/6.0 MHz)(Note 2) for stable oscillation.

② The PCC is rewritten for highest-speed operation after a time elapse which is sufficient for the voltage on the V_{DD} pin to be high enough for highest-speed operation.

③ The removal of commercial power is detected using, for example, an interrupt input (INT4 is useful), then SCC.0 is set to operate with the subsystem clock. (In this case, the start of subsystem clock generation must be confirmed beforehand.) After a time (46 machine cycles) required to switch to the subsystem clock elapses, SCC.3 is set to terminate main system clock generation.

④ After detecting the input of commercial power by using an interrupt, SCC.3 is cleared to start main system clock generation. After a time required for stable generation, SCC.0 is cleared to operate at highest speed.

Notes 1. 15.3 ms when the microcomputer operates at 4.19 MHz

2. 31.3 us when the microcomputer operates at 4.19 MHz

5.2.4 Clock output circuit

(1) Configuration of the clock output circuit

Figure 5-22 shows the configuration of the clock output circuit.

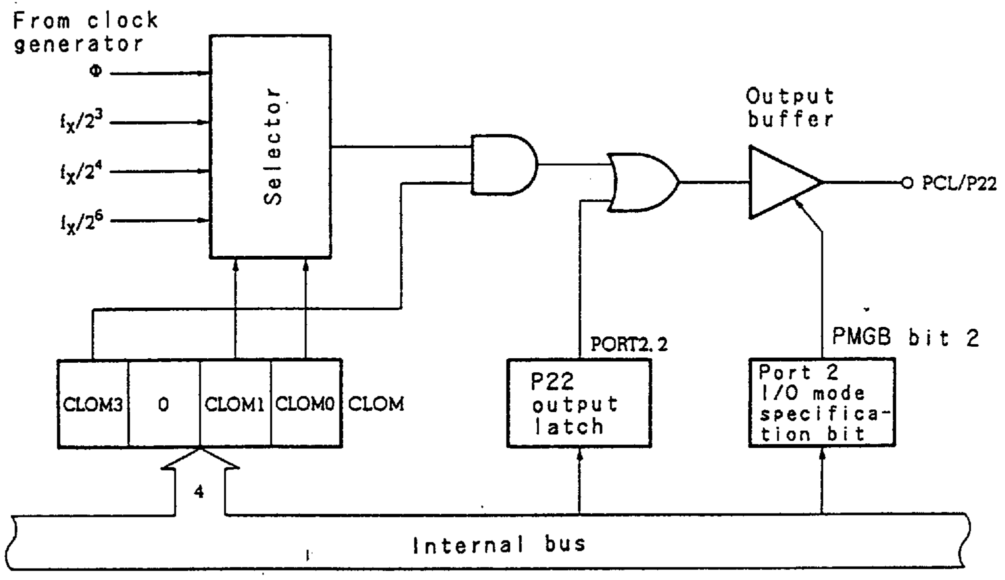
(2) Functions of the clock output circuit

The clock output circuit outputs a clock pulse signal on the P22/PCL pin to output remote control signals or to supply clock pulses to a peripheral LSI device.

The procedure for outputting a clock pulse signal is as follows:

(a) Select a clock output frequency, and disable clock output.
(b) Write a 0 in the P22 output latch.
(c) Set the output mode for port 2.
(d) Enable clock output.

Fig. 5-22 Configuration of the Clock Output Circuit



Remark: The clock output circuit is designed so that pulses with short widths do not appear in enabling or disabling clock output.

(3) Clock output mode register (CLOM)

The CLOM is a 4-bit register to control clock output.

The CLOM is set with a 4-bit memory manipulation instruction. No read operation is allowed on this register.

Example: CPU clock Φ is output on the PCL/P22 pin.

```
SEL MB15        ; Or CLR1 MBE
MOV A,#1000B
MOV CLOM,A
```

A $\overline{\text{RESET}}$ input clears the CLOM to 0, disabling clock output.

Fig. 5-23 Format of the Clock Output Mode Register

Address	3	2	1	0	Symbol
FD0H	CLOM3	0	CLOM1	CLOM0	CLOM

Clock output frequency selection bit
(Frequency when f_X = 6.0 MHz)

0	0	Φ output(*) (1.50 MHz, 750 kHz, 375 kHz, 93.7 kHz)
0	1	$f_X/2^3$ output (750 kHz)
1	0	$f_X/2^4$ output (375 kHz)
1	1	$f_X/2^6$ output (93.7 kHz)

(Frequency when f_X = 4.19 kHz)

0	0	Φ output(*) (1.05 MHz, 524 kHz, 262 kHz, 65.5 kHz)
0	1	$f_X/2^3$ output (524 kHz)
1	0	$f_X/2^4$ output (262 kHz)
1	1	$f_X/2^6$ output (65.5 kHz)

* Φ is a CPU clock selected by the PCC.

Clock output enable/disable bit

0	Disables output.
1	Enables output.

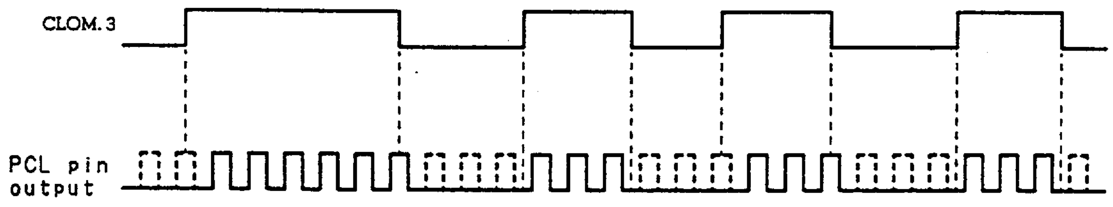
Caution: Be sure to write a 0 in bit 2 of the CLOM.

(4) Application to remote control output

The clock output function of the uPD75238 is applicable to remote control output. The frequency of the carrier for remote control output is selected by the clock frequency select bit of the clock output mode register. Pulse output is enabled or disabled by controlling the clock output enable/disable bit by software.

The clock output circuit is designed so that pulses with short widths do not appear in enabling or disabling clock output.

Fig. 5-24 Application to Remote Control Output



5.3 Basic Interval Timer

The uPD75238 contains an 8-bit basic interval timer (BT). BT has the following functions.

(a) Reference time generation (four time intervals)

(b) Selection of a wait time for releasing the standby mode, and counting

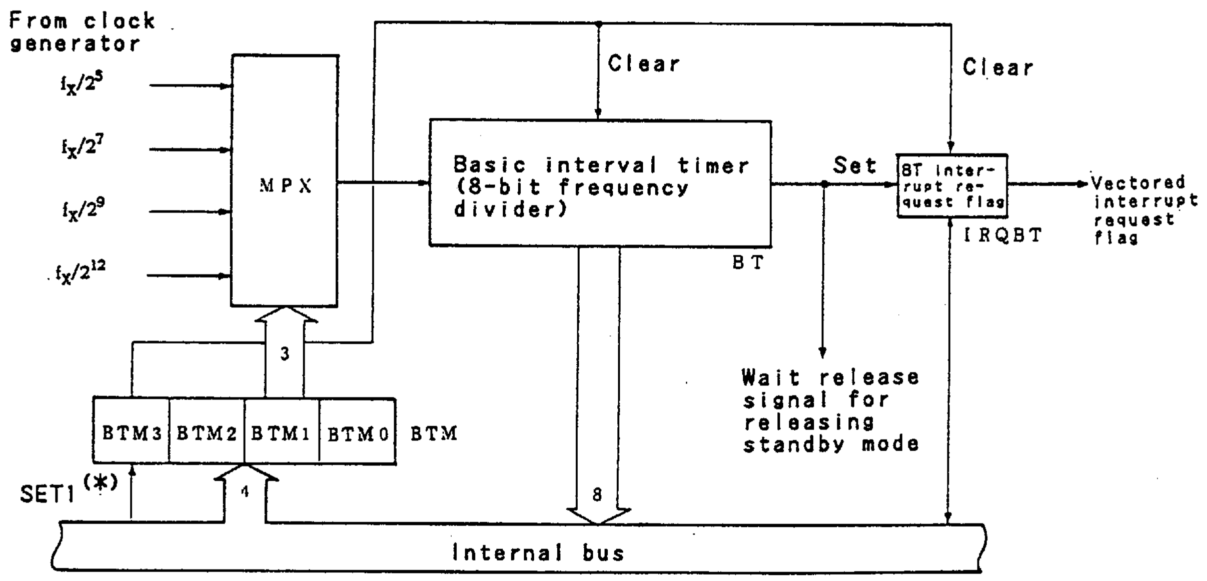
(c) Reading the count value

The basic interval timer can also be used as a watchdog timer for detecting program clash.

5.3.1 Configuration of the basic interval timer

Figure 5-25 shows the configuration of the basic interval timer.

Fig. 5-25 Configuration of the Basic Interval Timer



From clock generator
fx/2^5
fx/2^7
fx/2^9
fx/2^12
MPX
Clear
Clear
Basic interval timer (8-bit frequency divider)
BT
Set
BT inter-rupt re-quest flag
IRQBT
Vectored interrupt request flag
3
BTM3
BTM2
BTM1
BTM0
BTM
SET1 (*)
4
8
Wait release signal for releasing standby mode
Internal bus

* Instruction execution

5.3.2 Basic interval timer mode register (BTM)

The BTM is a 4-bit register for controlling operation of the basic interval timer.

A 4-bit memory manipulation instruction is used to set the BTM.

Bit 3 can be independently set using a bit manipulation instruction.

Example 1: The interrupt generation interval is set to 1.37 ms (6.0 MHz).

```
SEL    MB15         ; Or CLR1 MBE
MOV    A,#0111B
MOV    BTM,A        ; BTM ← 0111B
```

Example 2: The BT and IRQBT are cleared (application to watchdog timer).

```
SEL    MB15         ; Or CLR1 MBE
SET1   BTM.3        ; Set bit 3 of BTM to 1
```

When bit 3 is set to 1, the contents of the basic interval timer are cleared, and the basic interval timer interrupt request flag (IRQBT) is also cleared (to start the basic interval timer).

A $\overline{\text{RESET}}$ input clears the contents to 0, and the longest interrupt request signal generation interval time is set.

Fig. 5-26 Format of the Basic Interval Timer Mode Register

Address	3	2	1	0	Symbol
F85H	BTM3	BTM2	BTM1	BTM0	BTM

(When f_X = 6.0 MHz)

			Input clock specification	Interrupt interval time (wait time for releasing standby)
0	0	0	$f_X/2^{12}$ (1.46 kHz)	$2^{20}/f_X$ (175 ms)
0	1	1	$f_X/2^{9}$ (11.7 kHz)	$2^{17}/f_X$ (21.8 ms)
1	0	1	$f_X/2^{7}$ (46.9 kHz)	$2^{15}/f_X$ (5.46 ms)
1	1	1	$f_X/2^{5}$ (188 kHz)	$2^{13}/f_X$ (1.37 ms)
Other setting			Not to be set	–

(When f_X = 4.19 MHz)

			Input clock specification	Interrupt interval time (wait time for releasing standby)
0	0	0	$f_X/2^{12}$ (1.02 kHz)	$2^{20}/f_X$ (250 ms)
0	1	1	$f_X/2^{9}$ (8.18 kHz)	$2^{17}/f_X$ (31.3 ms)
1	0	1	$f_X/2^{7}$ (32.768 kHz)	$2^{15}/f_X$ (7.82 ms)
1	1	1	$f_X/2^{5}$ (131 kHz)	$2^{13}/f_X$ (1.95 ms)
Other setting			Not to be set	–

Basic interval timer start control bit

Setting this bit to 1 starts the basic interval timer (clears the counter and interrupt request flag). This bit is automatically reset to 0 at the start of operation.

5.3.3 Operation of the basic interval timer

The basic interval timer (BT) is always incremented by the clock supplied from the clock generator, and when it overflows (the timer value changes from FFH to 00), the interrupt request flag (IRQBT) is set. The count operation of the BT cannot be stopped.

One of four interrupt generation intervals can be selected by setting the BTM. (See Figure 5-26.)

The basic interval timer and the interrupt request flag can be cleared by setting bit 3 of the BTM to 1 (instruction for starting as an interval timer).

The count status of the BT can be read using an 8-bit manipulation instruction. The BT cannot be loaded with data.

Caution: When reading the count value of the basic interval timer, execute a read instruction twice so that unstable data being counted will not be read. If two read values are reasonable, use the second one as the result. If two read values are far apart form each other, retry all over again.

Example: The count value of the BT is read.

```
     SET1 MBE
     SEL  MB15
     MOV  HL,#BT  ; Set the address of BT in HL
LOOP:MOV  XA,@HL  ; First read
     SKE  XA,@HL  ; Second read
     BR   LOOP
```

To obtain a time required for stable system clock generation in releasing the STOP mode, a wait function is available which stops the operation of the CPU until the basic interval timer overflows.

The wait time after a $\overline{\text{RESET}}$ input is fixed. On the other hand, a wait time can be selected by setting the BTM when the STOP mode is released with an interrupt occurrence. In this case, possible wait times are the same as the interval times shown in Figure 5-26. The BTM must be set before the STOP mode is set. (See Chapter 7 for details.)

5.3.4 Application examples of the basic interval timer

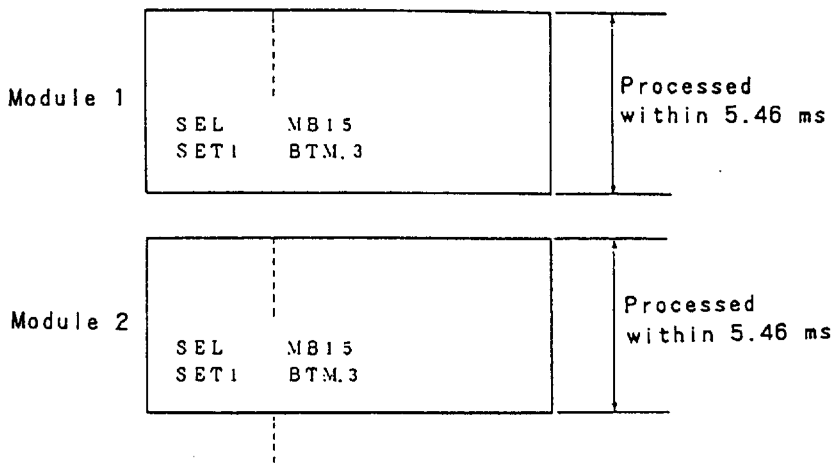
Example 1: A basic interval timer interrupt is enabled, and an interrupt generation interval of 1.37 ms (when operating at 6.0 MHz) is set.

```
SEL   MB15
MOV   A,#1111B
MOV   BTM,A        ; Setting and start
EI                 ; Enable an interrupt
EI    IEBT         ; Enable a BT interrupt
```

Example 2: Application to a watchdog timer

A program is divided into modules each of which can be executed within the time set in the BT, and the BT and IRQBT are cleared when the execution of each module is completed. If an interrupt occurs, it is treated as a crush. Set an interrupt occurrence interval to 5.46 ms (when operating at 6.0 MHz). And divide a program into modules so that the execution time of each module is less than 5.46 ms.

```
          ┌ SEL   MB15         ; or CLR MBE
Initial-  │ MOV   A,#1101B
ization   │ MOV   BTM,A        ; Setting and start
          │ EI
          └ EI    IEBT
```



Example 3: The wait time 5.46 ms (when operating at 6.0 MHz) is set in releasing the STOP mode with an interrupt.

```
        SEL    MB15
        MOV    A,#1101B
        MOV    BTM,A        ; BTM ← 1101B
        STOP                ; Set STOP mode
        NOP
```

Example 4: The high level width of pulses applied to the INT4 interrupt pin (both edges detected). (The pulse width is assumed not to exceed the value set in the BT. The set value in the BT is 5.46 ms or longer.)

```
LOOP:   MOV    XA,BT        ; INT4 interrupt routine
        MOV    BC,XA          (MBE=0, RBE=0)
        MOV    XA,BT
        SKE    XA,BC
        BR     LOOP
        SKT    PORT0.0      ; P00 = 1?
        BR     AA           ; NO
        MOV    BUFF,XA      ; Store data in data memory
        CLR1   FLAG         ; Clear data presence flag
        RETI
AA:     MOV    XA,BUFF
        SUBS   BC,XA
        NOP
        MOV    XA,BC
        MOV    BUFF,XA      ; Store data
        SET1   FLAG         ; Set data presence flag
        RETI
```

5.4 Clock Timer

The uPD75238 contains one channel of clock timer, which has the following functions.

(a) The clock timer sets the test flag (IRQW) every 0.5 seconds.

The IRQW can release the standby mode.

(b) Either the main system clock or the subsystem clock can be used to produce 0.5-second intervals.

(c) The fast-forward mode produces an interval 128 times faster (3.91 ms), which is useful for program debugging and testing.

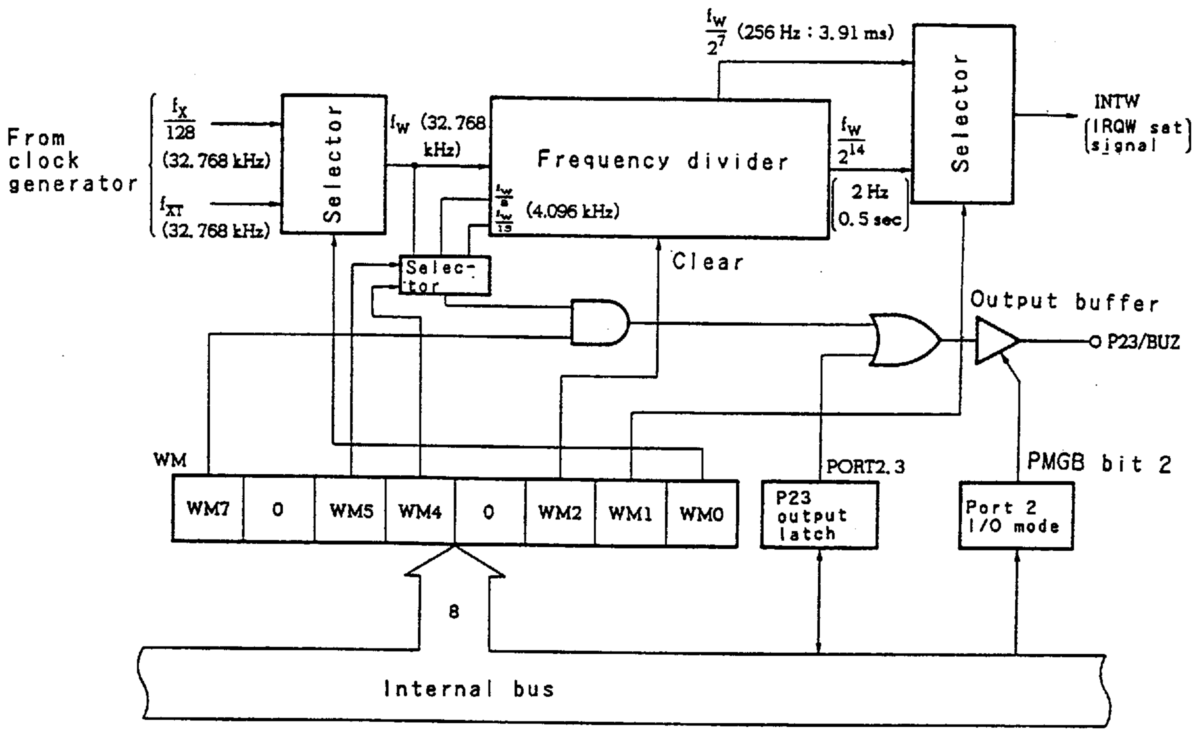
(d) A fixed frequency (2.048, 4.096, or 32.768 kHz) can be output to P23/BUZ, so that it can be used for sounding the buzzer and system clock frequency trimming.

(e) The frequency divider can be cleared to start the clock from zero second.

5.4.1 Configuration of the clock timer

Figure 5-27 shows the configuration of the clock timer.

Fig. 5-27 Block Diagram of the Clock Timer



Remark: The values in parentheses are for f_X = 4.194304 MHz and f_{XT} = 32.768 kHz.

Caution: When the main system clock operates at 6.0 MHz, a time interval of 0.5 s cannot be produced. Before producing this time interval, the main system clock must be changed to the subsystem clock.

5.4.2 Watch mode register

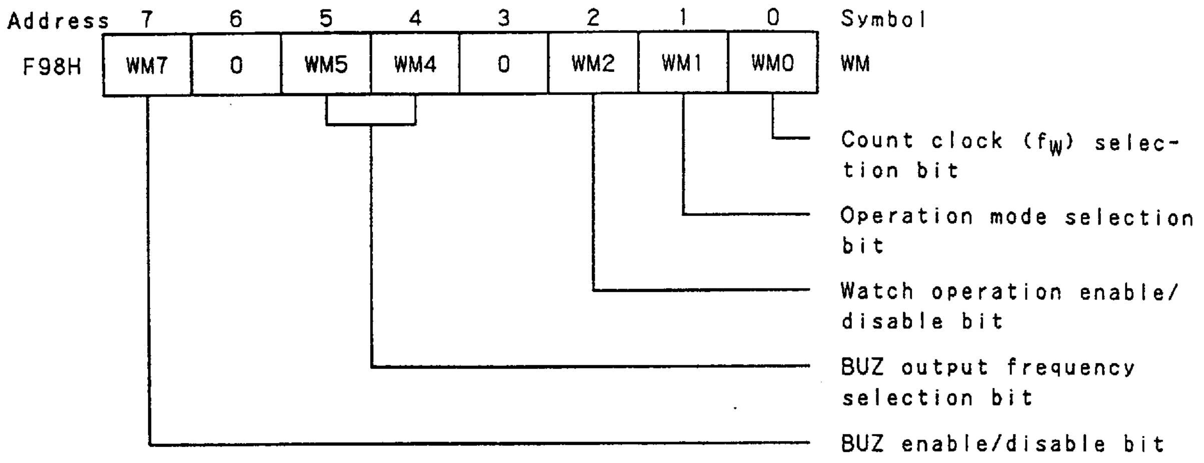
The watch mode register (WM) is an 8-bit register for controlling the clock timer. Figure 5-28 shows its format.

An 8-bit memory manipulation instruction is used to set the watch mode register. A $\overline{\text{RESET}}$ input clears all bits to 0.

Example: Time is set using the main system clock (4.19 MHz), and buzzer output is enabled.

```
CLR1  MBE
MOV   XA,#84H
MOV   WM,XA     ; Set WM
```

Fig. 5-28 Format of the Watch Mode Register



Address
7
6
5
4
3
2
1
0
Symbol
F98H
WM7
0
WM5
WM4
0
WM2
WM1
WM0
WM
Count clock (f_W) selection bit
Operation mode selection bit
Watch operation enable/disable bit
BUZ output frequency selection bit
BUZ enable/disable bit

Count clock (fw) selection bit

WM0	0	Selects system clock frequency-divided output: $f_X/128$
	1	Selects subsystem clock: f_{XT}

Operation mode selection bit

WM1	0	Normal watch mode ($f_W/2^{14}$: Sets IRQW every 0.5 s)
	1	Fast-forward watch mode ($f_W/2^7$: Sets IRQW every 3.91 ms)

Watch operation enable/disable bit

WM2	0	Disables watch operation (clears the frequency divider)
	1	Enables watch operation

BUZ output frequency selection bit

WM5	WM4	BUZ output frequency
0	0	$fw/2^4$ (20.48 kHz)
0	1	$fw/2^3$ (4.096 kHz)(*)
1	0	Not to be set
1	1	fw (32.768 kHz)(*)

* Not supported by the IE-75000-R.

BUZ enable/disable bit

WM7	0	Disables BUZ output
	1	Enables BUZ output

5.5 Timer/Event Counter

5.5.1 Configuration of the timer/event counter

The uPD75238 contains one channel of timer/event counter, which is configured as shown in Figure 5-29.

The timer/event counter has the following functions.

(a) Programmable interval timer operation

(b) Output of a square wave at a given frequency to the PTO0 pin

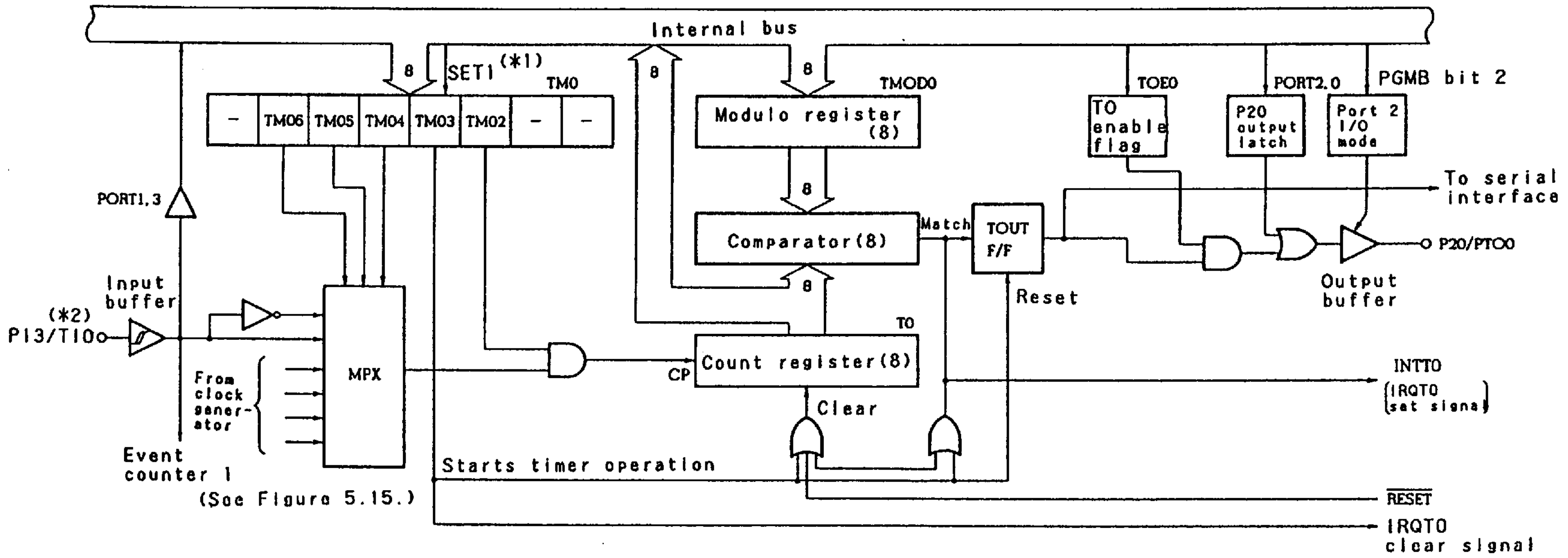
(c) Event counter operation

(d) Frequency divider operation that divides TI0 pin input by N and outputs the result to the PTO0 pin

(e) Supply of serial shift clock signal to a serial interface circuit

(f) Function of reading the state of counting

Fig. 5-29 Block Diagram of the Timer/Event Counter



*1 Instruction execution

*2 The P13/TI0 pin is an external event pulse input pin shared between timer/event counter and event counter.

5.5.2 Basic configuration and operation of the timer/event counter

The timer/event counter has several operation modes. A mode can be selected by the timer/event counter mode register (TMO). The basic configuration and operation of the timer/event counter are described below.

(1) The count pulse (CP) is selected by setting TMO, and is set in the timer/event counter count register (TO).

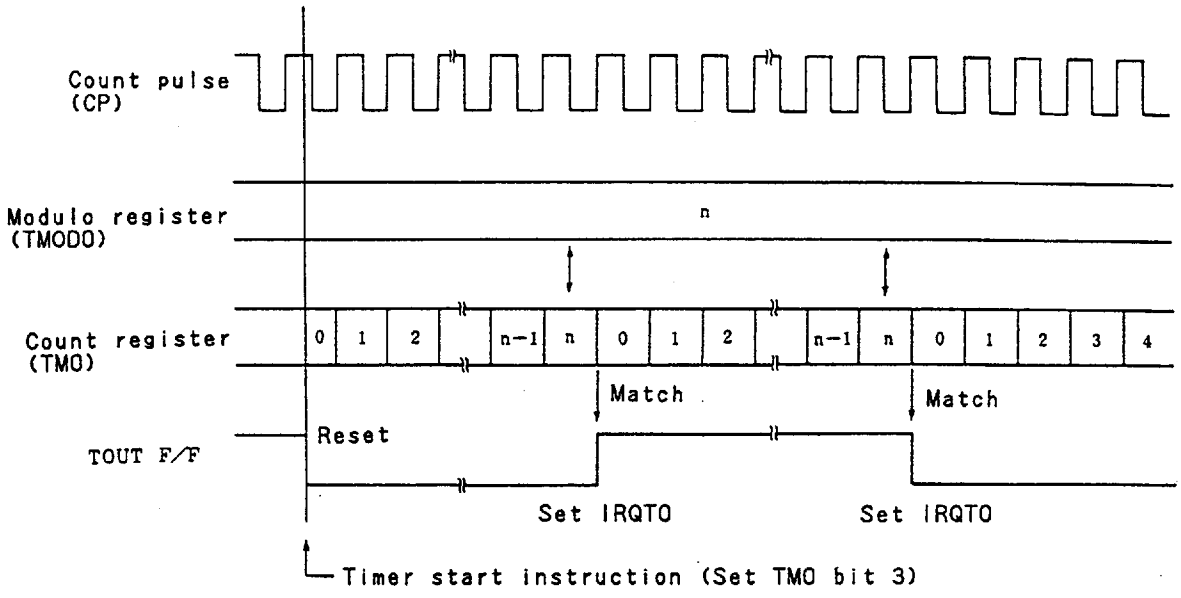
(2) TO is a binary 8-bit up-counter incremented by one when CP is applied. TO is cleared to 0 when a $\overline{\text{RESET}}$ input occurs, bit 3 of TMO is set (to start the timer), or a match signal occurs. TO can be read at any time with an 8-bit memory manipulation instruction, but cannot be written to.

(3) The timer/event counter modulo register (TMODO) is an 8-bit register for determining the count of TO. A value can be set in TMODO with an 8-bit memory manipulation instruction, but cannot be read. A $\overline{\text{RESET}}$ input occurrence initializes TMODO to FFH.

(4) The comparator compares the contents of TO with the contents of TMODO. If they match, the comparator generates a match signal, and sets the interrupt request flag (IRQTO).

Figure 5-30 shows the timing of count operation.

Fig. 5-30 Timing of Count Operation



5.5.3 Timer/event counter mode register and timer/event counter output enable flag

The timer/event counter mode register (TM0) is an 8-bit register for controlling the timer/event counter. Figure 5-31 shows its format.

An 8-bit memory manipulation instruction is used to set TM0.

Bit 3 is the timer start bit, and can be set independently of the other bits. Bit 3 is automatically reset to 0 when the timer starts instruction operation.

Example 1: The timer is started in the interval timer mode with CP = 5.86 kHz.

```
SEL  MB15             ; Or CLR1 MBE
MOV  XA,#01001100B
MOV  TM0,XA           ; TM0 ← 4CH
```

Example 2: The timer is restarted according to the setting of TM0.

```
SEL    MB15     ; Or CLR1 MBE
SET1   TM0.3    ; TM0.bit3 ← 1
```

A $\overline{\text{RESET}}$ input occurrence clears all the bits of TM0 to 0.

The timer/event counter output enable flag (TOE0) enables or disables output of the timer out flip-flop (TOUT flip-flop) status to the PTO0 pin.

The timer out flip-flop is inverted by a match signal sent out from the comparator. The timer out flip-flop is reset when an instruction sets bit 3 of TM0.

A $\overline{\text{RESET}}$ input occurrence clears the TOE0 and TOUT flip-flop to 0.

Fig. 5-31 Format of the Timer/Event Counter Mode Register

Address	7	6	5	4	3	2	1	0	Symbol
FA0H	—	TM06	TM05	TM04	TM03	TM02	—	—	TM0

Operation mode

	Count operation
0	Stop (count value retained)
1	Count operation

Timer start bit

When this bit is set to 1, the counter and IRQT0 flag are cleared. If bit 2 is set to 1, count operation starts.

Count pulse (CP) select bit
(When f_X = 6.0 MHz)

TM06	TM05	TM04	Count pulse (CP)
0	0	0	TI0 input rising edge
0	0	1	TI0 input falling edge
1	0	0	$f_X/2^{10}$ (5.86 kHz)
1	0	1	$f_X/2^8$ (23.4 kHz)
1	1	0	$f_X/2^6$ (93.8 kHz)
1	1	1	$f_X/2^4$ (375 kHz)
Other setting			Not to be set

(When f_X = 4.19 MHz)

TM06	TM05	TM04	Count pulse (CP)
0	0	0	TI0 input rising edge
0	0	1	TI0 input falling edge
1	0	0	$f_X/2^{10}$ (4.09 kHz)
1	0	1	$f_X/2^8$ (16.4 kHz)
1	1	0	$f_X/2^6$ (65.5 kHz)
1	1	1	$f_X/2^4$ (262 kHz)
Other setting			Not to be set

Fig. 5-32 Format of the Timer/Event Counter Output Enable Flag

Address	3
FA2H	TOE0

Timer/event counter output enable flag

0	Disables output
1	Enables output

5.5.4 Operation mode of the timer/event counter

The timer/event counter operates in the count operation disable mode or in the count operation mode, depending on the setting of the mode register.

The following operations are possible, regardless of the setting of the mode register:

(a) TI0 pin signal input and test (input test is possible for the other function (P13) of the pin)

(b) Output of the timer out flip-flop status to the PTO0

(c) Setting of the modulo register (TMOD0)

(d) Reading from the count register (T0)

(e) Setting, clearing, and testing of the interrupt request flag (IRQT0)

(1) Count operation disable mode

This mode is set when bit 2 of TM0 is set to 0. In this mode, count operation is not performed because count pulse (CP) supply to the count register is stopped.

(2) Count operation mode

This mode is set when bit 2 of TM0 is set to 1. In this mode, a count pulse signal selected with bits 4 to 6 is supplied to the count register for count operation as shown in Figure 5-30.

Timer operation is usually started in the following steps:

① A count value is set in the modulo register (TMOD0).

② An operation mode, count clock, and start instruction are set in the mode register (TM0).

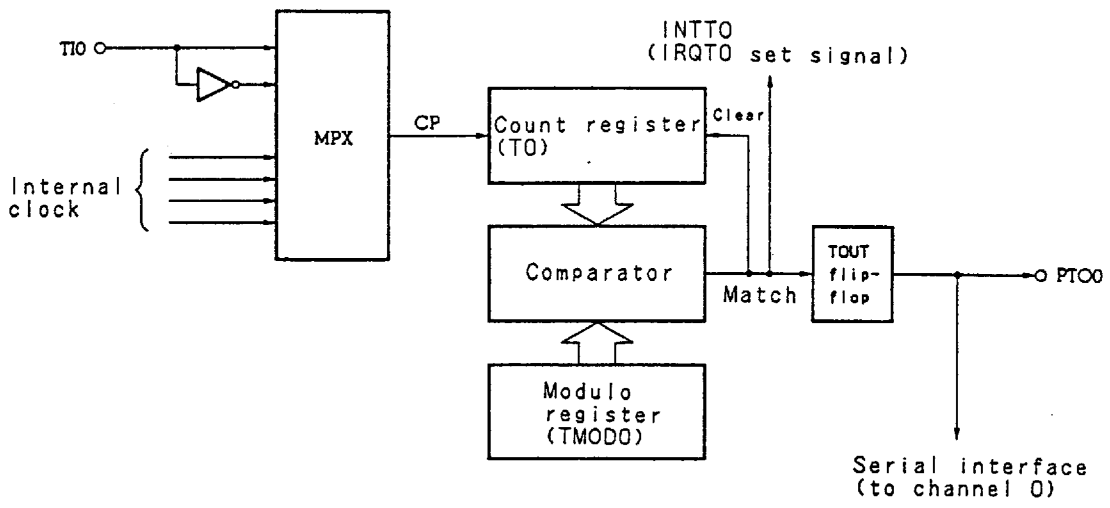
An 8-bit data transfer instruction is used to set the modulo register.

Caution: A value other than 0 must be set in the modulo register.

Example: The value 3FH is set in the modulo register of channel 0.

```
SEL  MB15       ; Or CLR1 MBE
MOV  XA,#3FH
MOV  TMOD0,XA
```

Fig. 5-33 Operation in the Count Operation Mode



5.5.5 Time setting in the timer/event counter

[Timer set time](period) is [value of the modulo register + 1] divided by [count pulse frequency] selected by the timer mode register.

$$T(\text{sec}) = \frac{(n + 1)}{F_{CP}} = (n + 1) \cdot (\text{resolution})$$

T(sec):	Timer set time (seconds)
F_{CP} (Hz):	Count pulse frequency (Hz)
n:	Value in the modulo register (n = 0)

Once the timer is set, the interrupt request signal (IRQT0) is generated at set time intervals.

Table 5-6 indicates the resolution and maximum set time (set when FFH is set in the modulo register) of the timer/event counter for each count pulse signal.

Example: A time interval of 30 ms is produced (f_{XX} = 6.0 MHz).

In this case, the mode with a maximum set time of 43.7 ms is used.

$$\frac{30\text{ ms}}{171\text{ us}} = 175.4 \fallingdotseq \text{AFH}$$

Accordingly, AFH is set in the modulo register.

```
SEL   MB15
MOV   XA,#AFH
MOV   TMOD0,XA
```

Table 5-6 Resolution and Maximum Set Time

(When f_X = 6.0 MHz)

Mode register			Timer channel 0	
TM06	TM05	TM04	Resolution	Maximum set time
1	0	0	171 us	43.7 ms
1	0	1	42.7 us	10.9 ms
1	1	0	10.7 us	2.73 ms
1	1	1	2.67 us	683 us

(When f_X = 4.19 MHz)

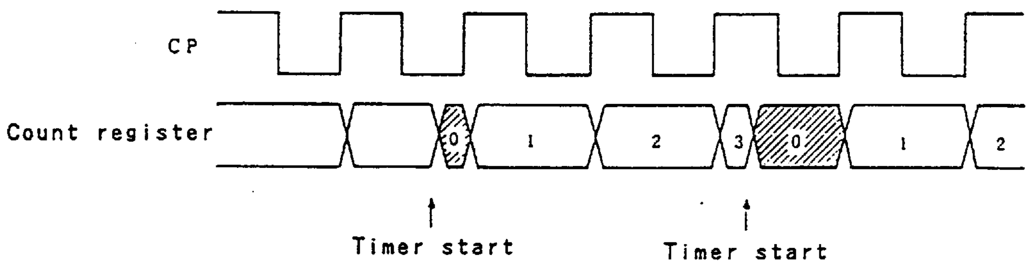
Mode register			Timer channel 0	
TM06	TM05	TM04	Resolution	Maximum set time
1	0	0	244 us	62.5 ms
1	0	1	61.1 us	15.6 ms
1	1	0	15.3 us	3.91 ms
1	1	1	3.81 us	977 us

5.5.6 Notes on timer/event counter applications

(1) Time error at the start of the timer

A maximum error of one count pulse (CP) cycle from a value calculated according to Section 5.5.5 occurs in a time period from the start of the timer (with TM0.3 set) to the generation of a match signal. This is because the count register is cleared not in phase with the CP as shown in Figure 5-34.

Fig. 5-34 Error at the Start of the Timer



(2) Notes on the start of the timer

Usually, when the timer is started (with bit 3 of TM0 set), the count register (T0) and the interrupt request flag (IRQT0) are cleared. However, when the timer is placed in the operation mode, and the setting of IRQT0 and the start of the timer occur at the same time, IRQT0 may not be cleared. This causes no problem if IRQT0 is used for a vectored interrupt. However, if IRQT0 is being tested, a problem arises because IRQT0 is set even if the timer is started. Accordingly, in a situation where the timer is started on such timing that IRQT0 may be set, the timer must be restarted after it is once stopped (with bit 2 of TM0 set to 0), or timer start operation must be performed twice.

Example: The timer is started on such timing that IRQT0 may be set.

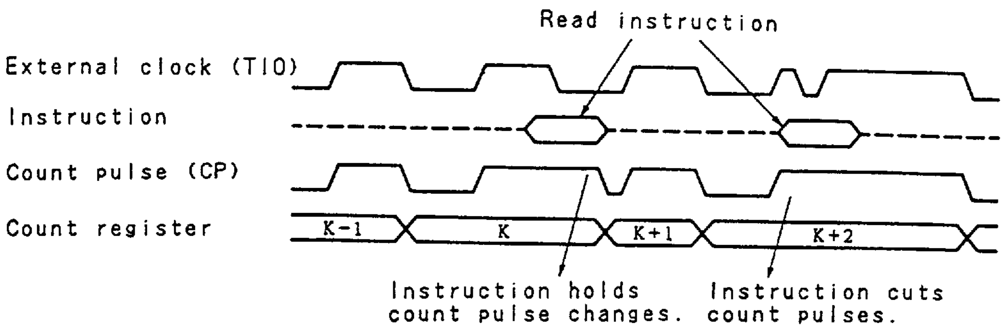
```
 SEL    MB15
 MOV    XA,#0
 MOV    TM0,XA    ; Stop the timer
 MOV    XA,#4CH
 MOV    TM0,XA    ; Restart
or
 SEL    MB15
 SET1   TM03
 SET1   TM03      ; Restart
```

(3) Error in reading the count register

The contents of the count register can be read using an 8-bit data memory manipulation instruction at any time. During operation by such an instruction, all count pulse changes are held not to change the count register. This means that if the count pulse signal source is applied to the TI0 input, as many count pulses as corresponding to the time required to execute the instruction are cut. (When an internal clock is used for the count pulse signal, this problem does not occur because of synchronization with the instruction.)

Accordingly, in an attempt to read the contents of the count register with a count pulse signal applied to TI0, the signal must have a pulse wide enough to avoid incorrect counting even if count pulses are cut. That is, the contents of the count register are held by a read instruction for one machine cycle, so that a signal applied to the TI0 pin must have a pulse wider than that.

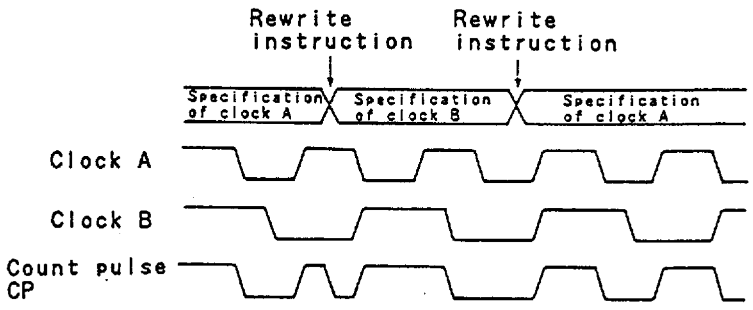
Fig. 5-35 Error in Reading the Count Register



(4) Notes on changing the count pulse

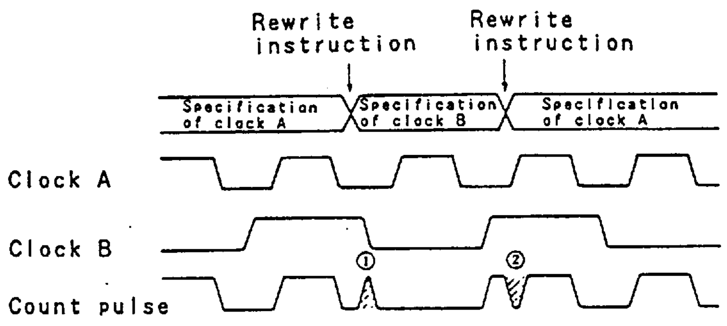
When the count pulse is changed by rewriting the contents of the timer mode register, the change specification becomes effective immediately after the rewrite instruction is executed.

Fig. 5-36 Note 1 on Changing the Count Pulse



A combination of clocks used for changing count pulse signals can generate a spike (① or ②) count pulse as shown in the figure below. In this case, an incorrect count operation may occur, or the contents of the count register may be destroyed. So when the count pulse is changed, bit 3 of the count mode register must be set to 1, and the timer must be restarted at the same time.

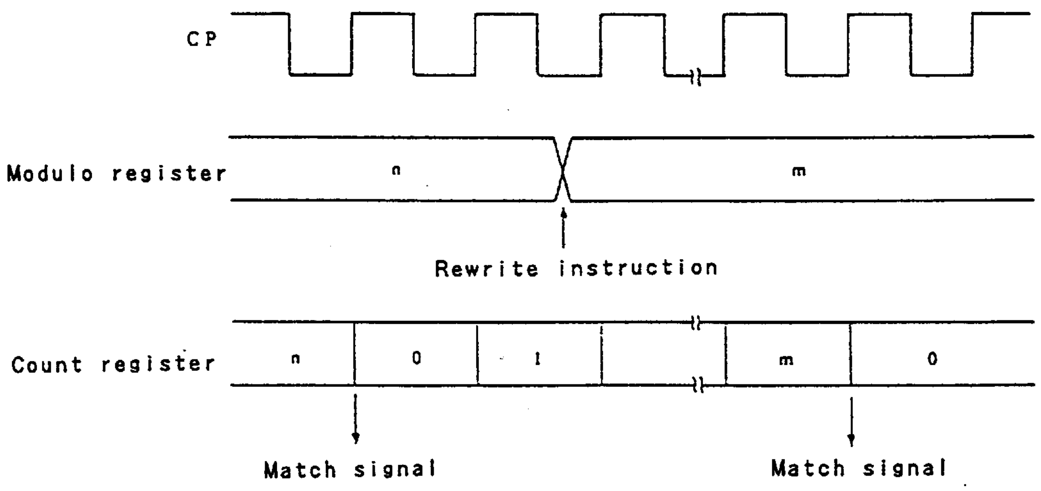
Fig. 5-37 Note 2 on Changing the Count Pulse



(5) Operation after the modulo register is changed

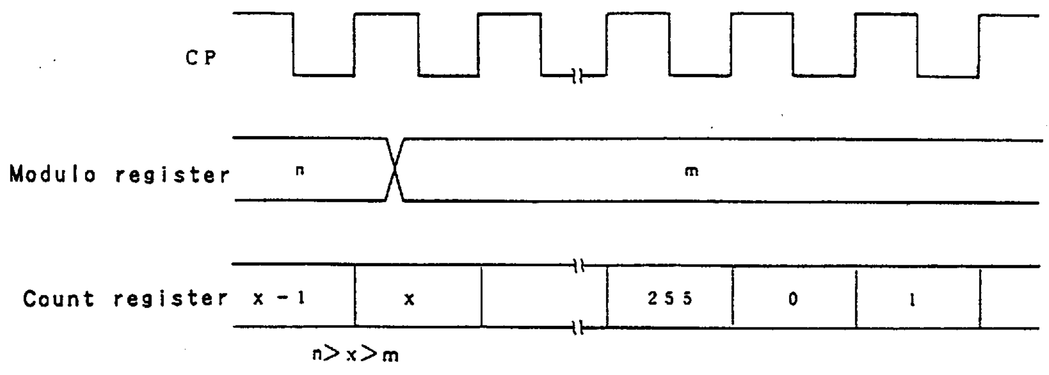
The contents of the modulo register are changed when an 8-bit data memory manipulation instruction is executed.

Fig. 5-38 Operation after the Modulo Register is Changed 1



If the new value of the modulo register is less than the value of the count register, the count register continues count operation until it overflows, then it restarts count operation from 0. Accordingly, the new value (m) of the modulo register is less than the value (n) before it is changed, the timer must be restarted after the contents of the modulo register are changed.

Fig. 5-39 Operation after the Modulo Register is Changed 2



5.5.7 Applications of the timer/event counter

(1) Timer 0 is used as an interval timer that generates interrupts at intervals of 10 ms.

. The high-order four bits of the mode register are set to 0100B to select 10.9 ms.

. The low-order four bits of the mode register are set to 1100B.

. The modulo register is set to the following value:

$$\frac{10\ \text{ms}}{42.7\ \text{us}} = 234 \fallingdotseq \text{E9H}$$

<Sample program>

```
SEL  MB15
MOV  XA,#E9H
MOV  TMOD0,XA          ; Set the modulo register
MOV  XA,#01001100B
MOV  TM0,XA            ; Set the mode register and start
                         the timer
EI                     ; Enable an interrupt
EI   IET0              ; Enable a timer interrupt
```

Remark: In this application, the TI0 pin can be used as an input pin.

(2) An interrupt is caused when the number of pulses (active high) applied to the TI0 pin reaches 100.

. The high-order four bits of the mode register are set to 0000 to select the rising edge.

. The low-order four bits of the mode register are set to 1100B.

. The modulo register is set to 99 = 100 - 1.

<Sample program>

```
SEL  MB15
MOV  XA,#100-1
MOV  TMOD0,XA          ; Set the modulo register
MOV  XA,#00001100B
MOV  TM0,XA            ; Set the mode register
EI
EI   IET0              ; Enable INTT0
```

5.6 Event Counter

5.6.1 Configuration of the event counter

The uPD75238 contains one channel for an event counter. Figure 5-40 shows the configuration of the counter.

The timer/event counter provides the following functions:

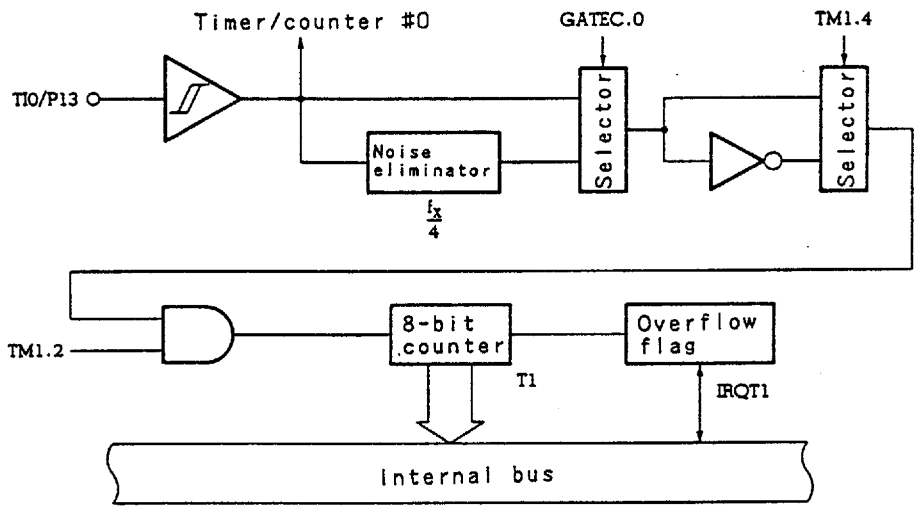
(a) Event counter operation

(b) Function of reading the state of counting

(c) Count pulse edge specification

(d) Noise elimination function

Fig. 5-40 Block Diagram of the Event Counter



Caution: The TI0/P13 pin is a dual function pin shared between timer/event counter 0 and counter 1, and also used as an external event pulse input pin.

5.6.2 Event counter mode register

The event counter mode register (TM1) is an 8-bit register that controls the event counter. Figure 5-41 shows the format of the register.

The TM1 is set with an 8-bit memory manipulation instruction. Bit 3 is an event counter start bit, and can be set independently of the other bits. Bit 3 is automatically reset to 0 when the timer starts operation.

Fig. 5-41 Format of the Event Counter Mode Register

Address	7	6	5	4	3	2	1	0	Symbol
FA8H	0	0	0	TM14	TM13	TM12	0	0	TM1

Event count operation enable/disable bit

TM12	0	Disables count operation (count value retained)
	1	Enables count operation

Event counter start instruction bit

TM13	When 1 is written, the counter and the IRQT1 flag are cleared. IF TM12 is set to 1, count operation starts.

Count pulse edge specification

TM14	0	TI0 input rising edge
	1	TI0 input falling edge

5.6.3 Overflow flag (IRQT1) and event counter count register (T1)

The overflow flag (IRQT1) is set to 1 when the event counter count register (T1) overflows. THe flag is cleared to 0 by a count operation start instruction (TM13 = 1).

IRQT1 is set with 1- or 4-bit memory manipulation instruction.

T1 is a binary 8-bit up-counter which is incremented by one when CP is applied.

The count register is cleared to 0 when a $\overline{\text{RESET}}$ input occurs, when bit 3 of TM1 is set to 1, or when an overflow occurs. If an overflow is followed by a CP, however, the count register restarts counting.

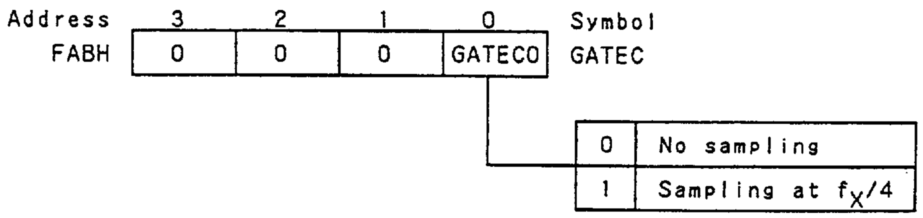
T1 can be read with an 8-bit memory manipulation instruction, but cannot be written to. A $\overline{\text{RESET}}$ input clears T1 to 0.

5.6.4 Event counter control register (GATEC)

GATEC specifies sampling by sampling clock ($f_X/4$). Figure 5-42 shows the format of the GATEC.

GATEC is set with a 4-bit manipulation instruction.

Fig. 5-42 Format of the Event Counter Control Register



A noise eliminator removes any pulse shorter than two-sampling-clock-cycle pulses ($8/f_X$) as noise, and surely accepts pulses wider than this as interrupt signals. (See Figure 5-43.)

Example 1: Sampling is performed with $f_X/4$, and falling edges on the TI0 input are counted. Counting is stopped.

```
SEL   MB15            ; Or CLR1 MBE
MOV   A,#0001B
MOV   GATEC,A
MOV   XA,#00010100B ; Counting is enabled, but operation is
                      disabled.
MOV   TM1,XA          ; TM0 ← 14H
```

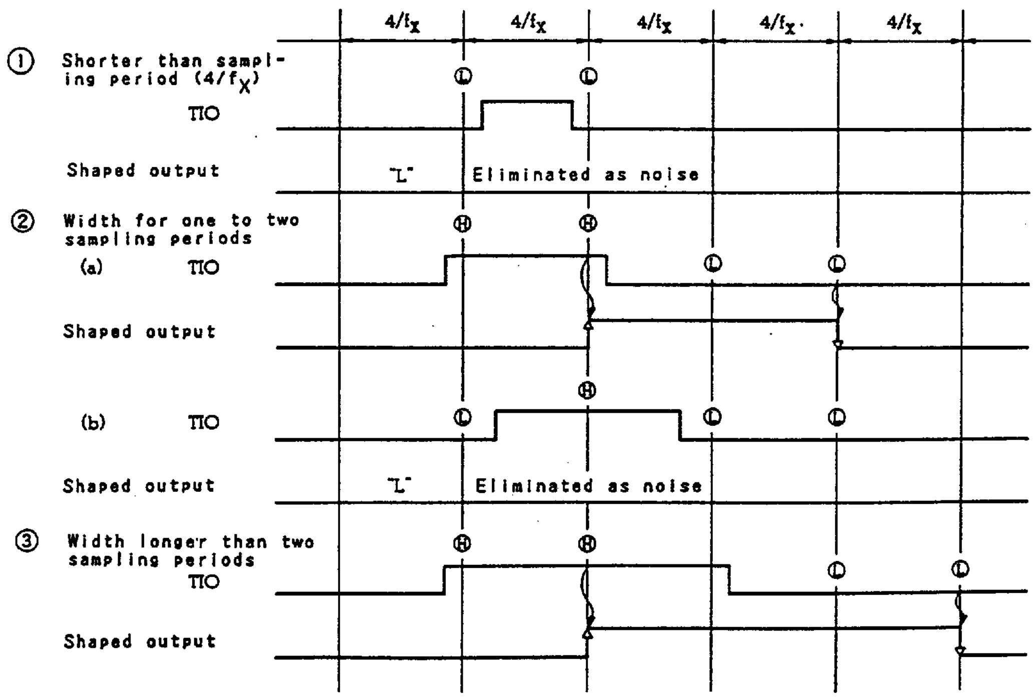
Example 2: The timer is started according to the setting of the event counter mode register.

```
SEL   MB15            ; Or CLR1 MBE
SET1  TM1.3           ; TM1, 3 ← 1
```

A $\overline{\text{RESET}}$ input clears GATEC to 0.

The noise eliminator outputs a high when high inputs occur consecutively, and outputs a low when low inputs occur consecutively.

Fig. 5-43 I/O Timing of the Noise Eliminator



5.7 Timer/Pulse Generator

5.7.1 Functions of the timer/pulse generator

The uPD75238 contains one channel of a timer/pulse generator that can be used as a timer or a pulse generator. It has the following functions:

(a) Functions available when the timer/pulse generator is used in the timer mode

- 8-bit interval timer operation using one of five clock sources (occurrence of IRQTPG)
- Square wave output to the PPO pin

(b) Functions available when the timer/pulse generator is used in the PWM pulse generation mode

- PWM pulse output to the PPO pin with an accuracy of 14 bits (applicable for electronic tuning when used as an D/A converter)
- Generation of interrupts at regular intervals ($2^{15}/f_X$: 5.46 ms at 6.0 MHz)(Note)

Note: 7.81 ms at 4.19 MHz

If pulse output is unnecessary, the PPO pin can be used as a 1-bit output port.

Caution: If the timer/pulse generator is operating when the STOP mode is set, it may malfunction. So the timer/pulse generator must be disabled with the mode register in advance.

5.7.2 Timer/pulse generator mode register (TPGM)

The timer/pulse generator mode register (TPGM) is an 8-bit register that controls operation of the timer/pulse generator. Figure 5-44 shows the format of the register.

TPGM is set with an 8-bit memory manipulation instruction.

Bit 3 enables or disables the transfer (reloading) of the timer/pulse generator modulo register (MODH and MODL) contents to the modulo latch. Bit 3 can be manipulated independently of the other bits.

By setting TPGM1 to 0, timer/pulse generator operation can be stopped to decrease current consumption.

A $\overline{\text{RESET}}$ input clears all bits to 0.

Fig. 5-44 Format of Timer/Pulse Generator Mode Register

Address	7	6	5	4	3	2	1	0	Symbol
F90H	TPGM7	–	TPGM5	TPGM4	TPGM3	0	TPGM1	TPGM0	TPGM

Timer/pulse generator operation mode selection bit

TPGM0	0	Select PWM pulse generation mode
	1	Select timer mode

Timer/pulse generator operation enable/disable bit

TPGM1	0	Disable timer/pulse generator operation
	1	Enable timer/pulse generator operation

Modulo register reload enable/disable bit

TPGM3	0	Disable reloading of modulo register
	1	Enable reloading of modulo register

PPO output latch data

TPGM4	0	Output 0 to PPO output latch
	1	Output 1 to PPO output latch

PPO pin output selection bit static/pulse

TPGM5	0	Static output on PPO pin
	1	Pulse output (square wave/PWM) on PPO pin

PPO pin output enable/disable bit

TPGM7	0	Disable output on PPO pin (high-impedance)
	1	Enable output on PPO pin

5.7.3 Configuration and operation of the timer/pulse generator

(1) When the timer/pulse generator is used in the timer mode

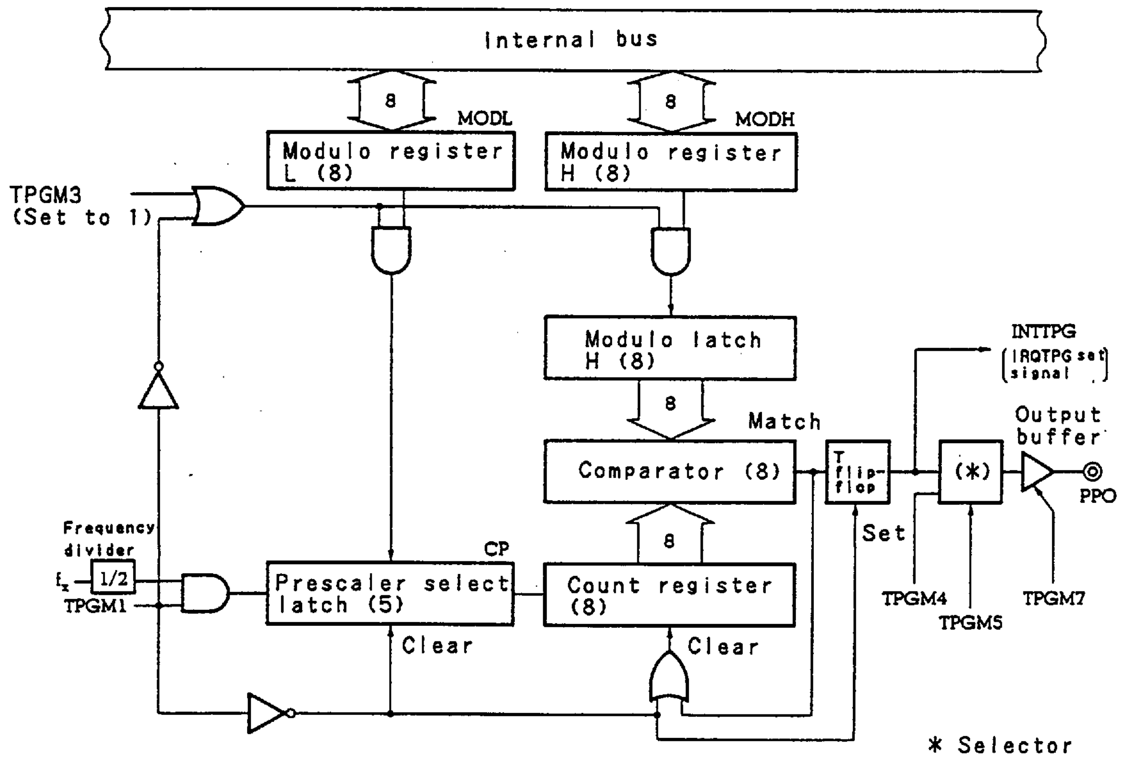
Figure 5-45 shows the configuration of the timer/pulse generator when it is used in the timer mode.

The timer mode is selected by setting bit 0 of TPGM to 1. In the timer mode, TPGM3 must be set to 1, allowing the modulo register to be reloaded at any time.

In the timer mode, a prescaler is selected by the modulo register L (MODL), and a frequency or interrupt interval value is set by the modulo register H (MODH). The timer starts when the TPGM1 is set to 1. Figure 5-46 shows the operation timing for the MODH setting, and Table 5-7 lists frequencies and interrupt intervals that can be set.

The output to the PPO pin can be switched between the square wave output and static output. To output a square wave, set TPGM5 and TPGM7 to 1.

Fig. 5-45 Block Diagram of the Timer/Pulse Generator (Timer Mode)



* Selector

Example: Set IRQTPG every 1.37 ms, and output a high on the PPO pin.

```
CLR1 MBE               ; Or SEL MB15
MOV  XA, #00100000B
MOV  MODL, XA
MOV  XA, #0FFH
MOV  MODH, XA
MOV  XA, #10011011B
MOV  TPGM, XA          ; Timer start, PPO ← 1
```

Caution: When the timer operating in the timer operation mode is stopped, IRQTPG may be set because T flip-flop is set. Therefore, the timer must be stopped with an interrupt being disabled, then IRQTPG must be cleared.

Example:

```
DI
CLR1 MBE
MOV  XA, #0
MOV  TPGM, XA
CLR1 IRQTPG
EI
```

Fig. 5-46 Timer Mode Operation Timing

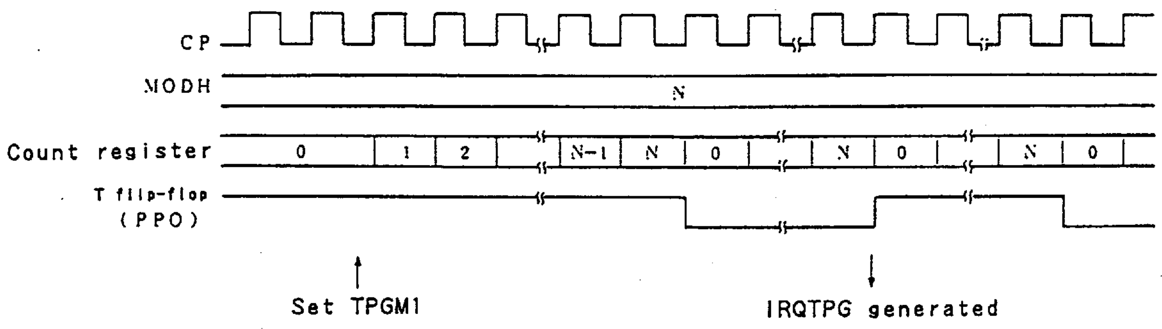


Table 5-7 Modulo Register Settings

(When f_X = 6.0 MHz)

MODL bits 2-6					Interrupt generation interval (f_X = 6.0 MHz)	Square wave output frequency (f_X = 6.0 MHz)
6	5	4	3	2		
0	0	0	0	1	$\frac{256(N+1)}{f_X}$ = 85.3 us - 10.9 ms	$\frac{f_X}{256(N+1)}$ = 91.6 Hz - 11.7 kHz
0	0	0	1	0	$\frac{128(N+1)}{f_X}$ = 42.7 us - 5.45 ms	$\frac{f_X}{128(N+1)}$ = 183 Hz - 23.4 kHz
0	0	1	0	0	$\frac{64(N+1)}{f_X}$ = 21.3 us - 2.73 ms	$\frac{f_X}{64(N+1)}$ = 366 Hz - 46.9 kHz
0	1	0	0	0	$\frac{32(N+1)}{f_X}$ = 10.7 us - 1.37 ms	$\frac{f_X}{32(N+1)}$ = 732 Hz - 93.8 kHz
1	0	0	0	0	$\frac{16(N+1)}{f_X}$ = 5.33 us - 683 us	$\frac{f_X}{16(N+1)}$ = 1465 Hz - 188 kHz

(When f_X = 4.19 MHz)

MODL bits 2-6					Interrupt generation interval (f_X = 4.19 MHz)	Square wave output frequency (f_X = 4.19 MHz)
6	5	4	3	2		
0	0	0	0	1	$\frac{256(N+1)}{f_X}$ = 122 us - 15.6 ms	$\frac{f_X}{256(N+1)}$ = 64 Hz - 8 kHz
0	0	0	1	0	$\frac{128(N+1)}{f_X}$ = 61.0 us - 7.81 ms	$\frac{f_X}{128(N+1)}$ = 128 Hz - 16 kHz
0	0	1	0	0	$\frac{64(N+1)}{f_X}$ = 30.5 us - 3.91 ms	$\frac{f_X}{64(N+1)}$ = 256 Hz - 32 kHz
0	1	0	0	0	$\frac{32(N+1)}{f_X}$ = 15.3 us - 1.95 ms	$\frac{f_X}{32(N+1)}$ = 512 Hz - 65 kHz
1	0	0	0	0	$\frac{16(N+1)}{f_X}$ = 7.63 us - 977 us	$\frac{f_X}{16(N+1)}$ = 1024 Hz - 131 kHz

Cautions 1. A value other than the above cannot be set in MODL. Bits 0, 1, and 7 must be set to 0.

2. N is the set value of MODH. 0 must not be set for N. Be sure to set a value from 1 to 255 for N.

(2) When the timer/pulse generator is used in the PWM pulse generation mode

Figure 5-47 shows the configuration of the timer/pulse generator when it is used in the PWM pulse generation mode.

The PWM pulse generation mode is selected by setting TPGM0 to 0. TPGM5 and TPGM7 are set to 1 to enable pulse output. In the PWM mode, the PWM pulse signal can be output on the PPO pin, and IRQTPG can be set at intervals of a fixed time period ($2^{15}/f_X$ = 5.46 ms/6.0 MHz).

PWM pulses output by the uPD75238 are active-low and have an accuracy of 14 bits. This pulse signal is applicable for electronic tuning and control of a DC motor when it is integrated by an external low-pass filter and is converted to analog voltage. (See Figure 5-48.)

The PWM pulse signal is generated by combining the basic period determined by $2^{10}/f_X$ (171 us/ 6.0 MHz)(Note) and the secondary period by $2^{15}/f_X$ (5.46 ms/6.0 MHz) so that the time constant of the external low-pass filter can be decreased.

Note: 244 us when the microcomputer operates at 4.19 MHz

The low-level width of a PWM pulse depends on the 14-bit modulo latch value. The upper 8 bits of the modulo latch are sent from the 8 bits of MODH, and the lower 6 bits of the latch are sent from the upper 6 bits of MODL.

When the PWM pulse signal is converted to analog form in the configuration as shown in Figure 5-48, the voltage level of the analog output is obtained as follows:

$$V_{AN} = V_{ref} \times \frac{\text{Value of modulo latch}}{2^{14}}$$

V_{ref}: Reference voltage of external switching circuitry

To prevent an incorrect PWM pulse from being output by unstable modulo latch data being rewritten, in the uPD75238, correct data can be written into MODH and MODL beforehand with 8-bit manipulation instructions, then the written 14-bit data can be transferred to the modulo latch at one time. This transfer operation is referred to as reloading, and it is controlled by TPGM3. If TPGM3 is 0, reloading is disabled, and if it is 1, reloading is enabled. Follow the procedure below to rewrite the modulo latch contents:

① Clear TPGM3 to disable reloading.
② Change the MODH and MODL contents.
③ Set TPGM3 to enable reloading.

Cautions 1. If the modulo register H (MODH) is set to 0, the PWM pulse generator cannot function normally. So be sure to set MODH to a value from 1 to 255.

2. If the lower 2 bits of the modulo register L (MODL) is read, the read result is unpredictable.

Cautions 3. If the modulo latch is changed in a shorter period than the PWM pulse basic period $2^{10}/f_X$ (171 us/6.0 MHz)(Note), PWM pulses do not change.

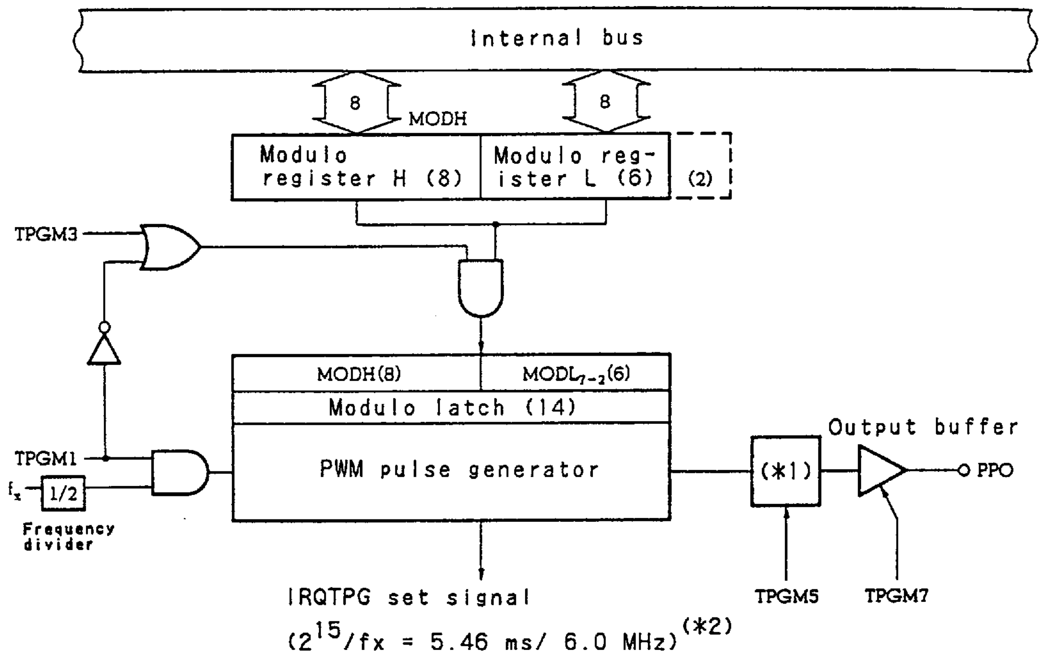
Note: 244 us when the microcomputer operates at 4.19 MHz

Example: Decrease analog output voltage to the lowest level, then increase it to the highest level.

```
CLR1  MBE
MOV   XA, #01H
MOV   MODH, XA         ; MODH ← 01
MOV   MODL, XA         ; MODL ← 01
MOV   XA, #10101010B
MOV   TPGM, XA         ; Enable PWM pulse
                         output

CLR1   TPGM. 3         ; Disable reloading
MOV    XA, #0FFH
MOV    MODH, XA
MOV    MODL, XA
SET1   TPGM. 3         ; Enable reloading
```

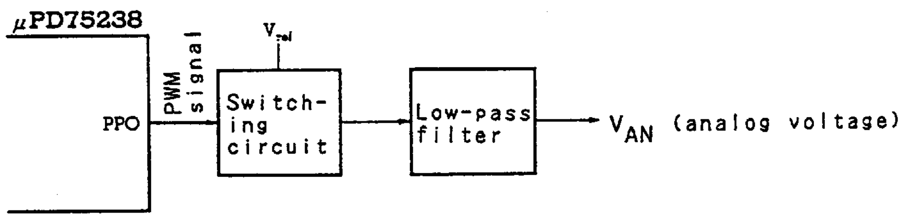
Fig. 5-47 Block Diagram of the Timer/Pulse Generator (PWM Pulse Generation Mode)



*1 Selector

*2 7.81 ms when the microcomputer operates at 4.19 MHz

Fig. 5-48 Sample Configuration of D/A Conversion Using uPD75238



(3) Static output to the PPO pin

When pulse output is unnecessary, the PPO pin can be used as normal static output. In this case, the output data is set in TPGM4 with TPGM5 set to 0 and TPGM7 to 1.

5.7.4 Application of the timer/pulse generator

(1) The timer/pulse generator is used as an interval timer which generates interrupts at 1-ms intervals (when f_X = 6.0 MHz).

From Table 5-7, an equation for setting the timer to 1 ms is selected. Since N is an integer from 1 to 255, there is some difference between the set time and an actual time. To obtain high resolution (that is, to suppress the difference as little as possible), the value of N must be as large as possible.

$$T = \frac{32(N + 1)}{f_X} \quad \text{(modulo register L (MODL) = 20H)}$$

Since T = 1 ms, the value N to be set in modulo register H (MODH) is

$$N = \frac{f_X}{32} \times T - 1 = \frac{6.0 \times 10^6}{32} \times 1 \times 10^{-3} - 1 = 186.500$$

Therefore, N is set to 187.
When N = 187, the actual time is obtained as follows:

$$T = \frac{32(N + 1)}{f_X} = \frac{32(187 + 1)}{6.0 \times 10^6} = 1.003 \text{ (ms)}$$

The error is 0.03%.

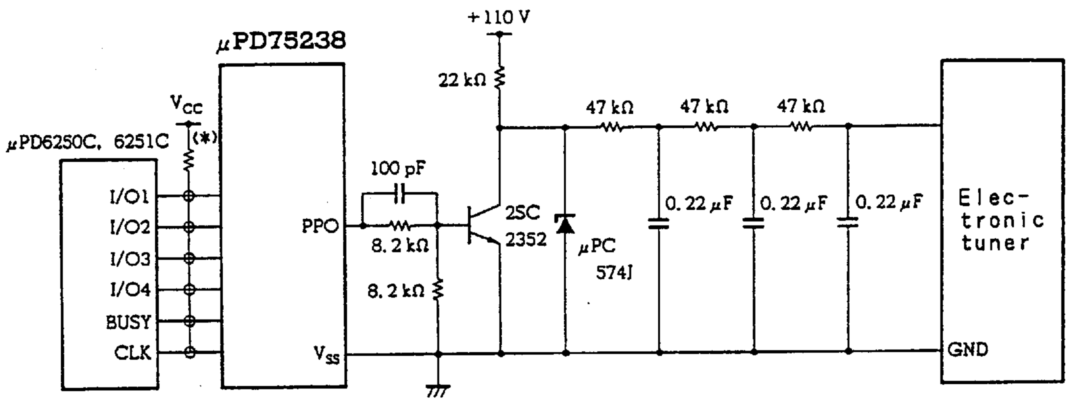
<Sample program>

```
CLR1   MBE
MOV    XA,#20H          ; Select resolution
MOV    MODL,XA
MOV    XA,#ABH          ; Determine set value
MOV    MODH,XA
MOV    XA,#10011011B
MOV    TPGM,XA          ; Start timer
EI                      ; Enable interrupt
EI     IETPG            ; Enable timer interrupt
```

(2) The timer/pulse generator is used for electronic tuning by converting the PWM pulse signal output on the PPO pin to analog voltage form.

Figure 5-49 shows a sample circuit which applies to a voltage synthesizer TV tuner.

Fig. 5-49 Sample Circuit Applicable to TV Tuners



* Connect pull-up resistors to I/O1 to I/O4, BUSY, and CLK pins.

Even in the above simple circuit configuration, high characteristics, including a ripple voltage of 2 mV_{p-p} and a response rate of 100 ms, can be obtained.

If f_X is 6.0 MHz, the ripple voltage components include a 5.86-Hz component using the basic period and a 183-Hz component using the secondary period. (The 183-Hz component is very small, and the frequency 256 Hz can be regarded as the lowest frequency.)

As shown in Figure 5-49, the pulse signal output on the PPO is inverted by an external transistor. The period of a low output on the PPO corresponds to the value written in the timer/pulse generator modulo register.

The tuning voltage is preset with the AFC and horizontal synchronizing signals. The resultant tuning data is written to the non-volatile memory uPD6250C or uPD6251C.

(3) Notes are output on the PPO pin. (A square wave is output on the PPO pin at a given frequency.)

The frequency of clock oscillation is set to 4.194304 MHz.

Although various frequencies from 64 Hz to 131 kHz are possible by setting the timer/pulse generator modulo register, the resolution is restricted. (See Table 5-7.) This means that generated notes have errors. (See Table 5-8.) According to the table, to generate a one-octave higher tone the MODL value is doubled.

As shown in Table 5-7, to increase the resolution (for fine tuning), the value N must be as large as possible. In Table 5-8, a large value is set in MODL to obtain a large value for N.

<Sample program>

Tone la: Output 440 Hz.

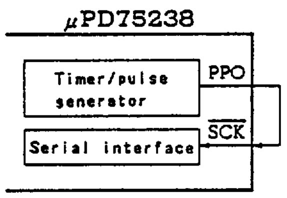
```
CLR1   MBE
MOV    XA,#10H         ; Select output frequency
MOV    MODL,XA
MOV    XA,#94H         ; 440Hz
MOV    MODH,XA
MOV    XA,#10101011B
MOV    TPGM,XA         ; Start note output
```

Table 5-8 Generation of Note Using the Timer/Pulse Generator

Temperament		Modulo register setting		Output from PPO	
Note	Frequency [Hz]	MODL	MODH	Frequency [Hz]	Error [Hz]
La A	440	00010000B	94H	439.8	-0.2 (-0.045%)
$A^{\#}$	446.16	00010000B	8CH	464.8	-1.36 (-0.29%)
Ti B	493.88	00010000B	84H	492.8	-1.08 (-0.22%)
Do C_5	523.248	00100000B	F9H	524.29	1.04 (0.20%)
$C^{\#}$	554.368	00100000B	FBH	555.39	1.02 (0.18%)
Re D	587.328	00100000B	DEH	587.77	0.442 (0.075%)
$D^{\#}$	662.256	00100000B	D2H	621.2	-1.06 (-0.17%)
Mi E	659.248	00100000B	C6H	658.7	-0.548 (-0.083%)
Fa F	698.464	00100000B	BBH	697.2	-1.264 (-0.12%)
$F^{\#}$	739.984	00100000B	B0H	740.5	0.516 (0.070%)
So G	783.984	00100000B	A6H	784.9	0.916 (0.117%)
$G^{\#}$	830.608	00100000B	9DH	829.7	-0.908 (-0.11%)

(4) A square wave output on the PPO pin is used as a baud rate generator for the serial interface.

PPO is connected to $\overline{\text{SCK0}}$ as shown below.



The timer/pulse generator uses the main system clock whose frequency (f_X) is divided by two as count pulse.

To obtain a baud rate of 9600, use a system clock whose frequency (f_X) is 4.9152 MHz.

. The mode register (TPGM) is set to 10111011B (the square-wave output mode)
. The modulo register is set as listed below.
  Modulo register L (MODL) is set to 08H.

Baud rate	Modulo register H (MODH)
9600	03H
4800	07H
2400	0FH
1200	1FH
600	3FH
300	7FH
150	FFH

Caution: In this application, a main system clock whose frequency is 4.9152 MHz is used. When selecting CPU clock Φ for the uPD75236, set the PCC to 0010B (1 machine cycle = 1.63 us at 614 kHz) or 0000B (1 machine cycle = 13 us at 76.7 kHz). If the PCC is set to 0011B, one machine cycle falls short of 0.95 us, the minimum value in the specifications.

<Sample program>

Data is output at 9600 baud starting with LSB.

```
CLR1  MBE
MOV   XA,#08H          ; Select output frequency band
MOV   MODL,XA
MOV   XA,#03H          ; 9600 baud
MOV   MODH,XA
MOV   XA,#10111011B
MOV   TPGM,XA          ; Start timer, high output on PPO
MOV   XA,#1FH          ; LSB, PPO → SCK, SO → H
MOV   SIOM,XA          ; Transfer start
MOV   XA,DATA          ; Transfer data set
MOV   SIO,XA
```

5.8 Serial Interface (Channel 0)

The uPD75238 has two channels of serial interface: Channel 0 and channel 1. Table 5-9 lists the differences between channel 0 and channel 1.

Table 5-9 Differences between Channel 0 and Channel 1

Serial transfer mode, function		Channel 0	Channel 1
3-wire serial I/O	Clock selection	$f_X/2^4$, $f_X/2^3$, TOUT flip-flop external clock	$f_X/2^4$, $f_X/2^3$, external clock
	Transfer method	Start bit switchable: MSB/LSB	Start bit: MSB
	Transfer end flag	Serial transfer end interrupt request flag (IRQCSIO)	Serial transfer end flag (EOT)
2-wire serial I/O		Available	Not available
Serial bus interface			

5.8.1 Serial interface (channel 0) functions

The serial interface (channel 0) of the uPD75238 has four modes.

The functions of the four modes are outlined below.

(1) Operation halt mode

This mode is used when serial transfer is not performed. This mode reduces power consumption.

(2) Three-wire serial I/O mode

In this mode, 8-bit data is transferred through three lines: Serial clock ($\overline{\text{SCK0}}$), serial output (SO0), and serial input (SI0).

The three-wire serial I/O mode allows full-duplex transmission, so data transfer can be performed at higher speed.

The user can choose 8-bit data transfer starting with the MSB or LSB, so devices starting with either the MSB or LSB can be connected.

The three-wire serial I/O mode enables connections to be made with the 75X series, 78K series, and many other types of peripheral I/O devices.

(3) Two-wire serial I/O mode

In this mode, 8-bit data is transferred through two lines: Serial clock ($\overline{\text{SCK0}}$) and serial data bus (SB0 or SB1). By controlling output levels on the two lines by software, communication with multiple devices is enabled.

The output levels of $\overline{\text{SCK0}}$ and SB0 (or SB1) can be controlled by software, so the user can match an arbitrary transfer format. This means that a line that has been required for handshaking to connect multiple lines can be eliminated for more efficient input/output port utilization.

(4) Serial bus interface (SBI) mode

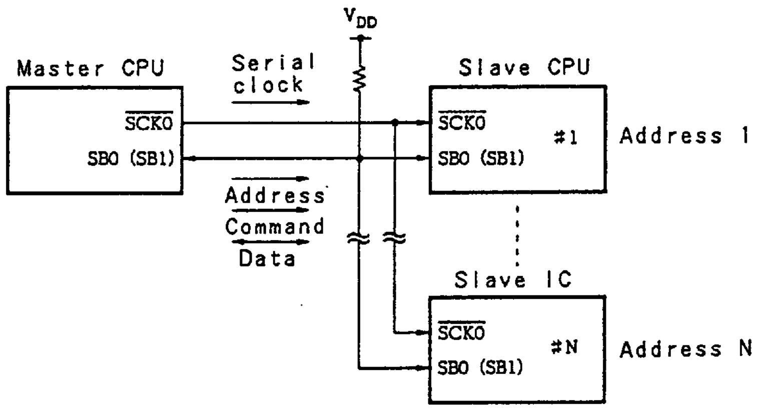
In this mode, communication with multiple devices can be performed using two lines: Serial clock ($\overline{\text{SCK0}}$) and serial data bus (SB0 or SB1).

This mode conforms to the NEC serial bus format.

In this mode, the sender can output, on the serial data bus, an address for selecting a device subject to serial communication, commands directed to the remote device, and data. The receiver can identify an address, commands, and data from received data by hardware.

This function enables more efficient input/output port utilization as in the case of the two-wire serial I/O mode. In addition, this function can simplify the serial interface control portion of an application program.

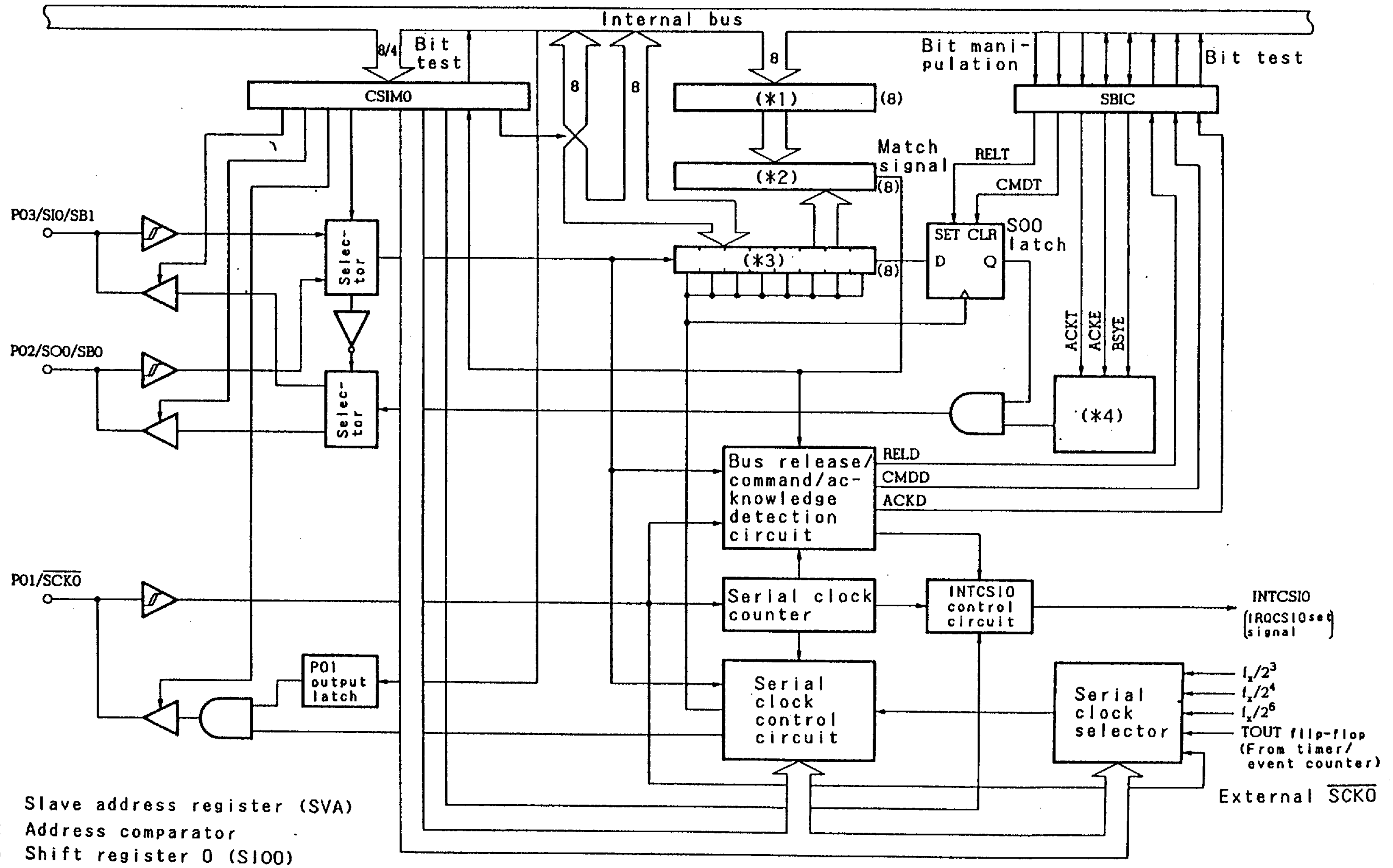
Fig. 5-50 Example of the SBI System Configuration



5.8.2 Configuration of serial interface (channel 0)

Figure 5-51 shows the block diagram of the serial interface (channel 0).

Fig. 5-51 Block Diagram of the Serial Interface (Channel 0)



*1 Slave address register (SVA)
*2 Address comparator
*3 Shift register 0 (SIO0)
*4 Busy/acknowledge output circuit

(1) Serial operation mode register 0 (CSIM0)

CSIM0 is an 8-bit register which specifies a serial interface operation mode, serial clock, wake-up function, and so forth. (See (1) in Section 5.8.3 for details.)

(2) Serial bus interface control register (SBIC)

SBIC is an 8-bit register consisting of bits for controlling the serial bus and flags for indicating the states of input data from the serial bus. SBIC is used mainly in the SBI mode. (See (2) in Section 5.8.3 for details.)

(3) Shift register 0 (SIO0)

SIO0 is an 8-bit register which converts 8-bit serial data to parallel data, and 8-bit parallel data to serial data. SIO0 performs transfer (shift) operations in phase with the serial clock. Transfer operations are controlled by writing data to SIO0. (See (3) in Section 5.8.3 for details.)

(4) SO0 latch

SO0 is a latch to hold the levels of pins SO0 and SB0, or SI0 and SB1, which can be controlled directly by software. In the SBI mode, SO0 is set when the eighth clock of $\overline{\text{SCK0}}$ has been output. (See (2) in Section 5.8.3 for details.)

(5) Serial clock selector

The serial clock selector selects the serial clock to be used.

(6) Serial clock counter

The serial clock counter counts the serial clock to be output or input during transfer operation, and checks whether 8-bit data has been transferred.

(7) Slave address register (SVA) and address comparator

. In the SBI mode

SVA is used when the uPD75238 is used as a slave device. A slave sets the number assigned to it (slave address) in SVA. The master outputs a slave address to select a particular slave.

Two data values (a slave address output from the master and the value of SVA) are compared with each other by the address comparator. If a match is found, the slave is selected.

. In the two-wire serial I/O mode or SBI mode

SVA detects an error when data is transferred with the uPD75238 operating as the master or a slave. (See (4) in Section 5.8.3 for details.)

(8) INTCSIO control circuit

The IRQCSIO control circuit controls occurrence of interrupt requests. An interrupt request (INTCSIO) is generated and an interrupt request flag (IRQCSIO) is set in the following cases:

. In the three-wire or two-wire serial I/O mode

An interrupt is generated each time the eighth serial clock is counted.

. In the SBI mode

When WUP$^{(Note)}$ = 0, an interrupt is generated each time the eighth serial clock is counted. When WUP = 1, an interrupt is generated when values of SVA and SIO0 match after an address is received.

Note: WUP: Wake-up function specification bit (bit 5 of CSIM0)

(9) Serial clock control circuit

The serial clock control circuit controls the serial clock to be supplied to the shift register, or controls the clock to be output to the $\overline{\text{SCK0}}$ pin when the internal system clock is used.

(10) Busy/acknowledge output circuit and bus release/command/acknowledge detection circuit

The busy/acknowledge output circuit and bus release/command/acknowledge detection circuit output and detect control signals generated in the SBI mode.

These circuits do not operate in the three-wire and two-wire serial I/O modes.

(11) P01 output latch

The P01 output latch generates serial clock by software after the eighth serial clock has been output.

When the $\overline{\text{RESET}}$ signal is entered, this latch is set to 1.

To select the internal system clock as the serial clock, set the P01 output latch to 1.

5.8.3 Register functions

(1) Serial operation mode register 0 (CSIM0)

Figure 5-52 shows the format of serial operation mode register 0 (CSIM0).

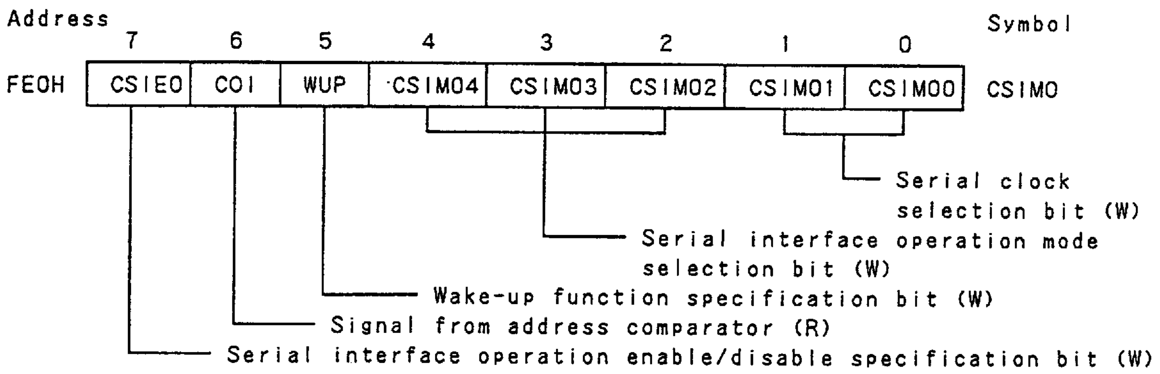
CSIM0 is an 8-bit register which specifies a serial interface (channel 0) operation mode, serial clock, wake-up function, and so forth.

CSIM0 is manipulated using an 8-bit memory manipulation instruction. The higher three bits can be manipulated bit by bit. Each bit can be manipulated using its name.

Each bit may or may not allow read and/or write operation (see Figure 5-52). Bit 6 allows bit test operation only; any data written to this bit is invalid.

When the $\overline{\text{RESET}}$ signal is input, this register is set to 00H.

Fig. 5-52 Format of Serial Operation Mode Register 0 (CSIM0)


Address
Symbol
7
6
5
4
3
2
1
0
FE0H
CSIE0
COI
WUP
CSIM04
CSIM03
CSIM02
CSIM01
CSIM00
CSIM0
Serial clock selection bit (W)
Serial interface operation mode selection bit (W)
Wake-up function specification bit (W)
Signal from address comparator (R)
Serial interface operation enable/disable specification bit (W)

Remark: (R): Read only
(W): Write only

(to be continued)

Fig. 5-52 Format of Serial Operation Mode Register 0 (CSIM0) (Cont'd)

Serial clock selection bit (W)

CSIM01	CSIM00	Serial clock: 3-wire serial I/O mode	Serial clock: SBI mode	Serial clock: 2-wire serial I/O mode	$\overline{SCK0}$ pin mode
0	0	Input clock externally applied to $\overline{SCK0}$ pin			Input
0	1	Timer/event counter output (TO)			Output
1	0	$f_X/2^4$ (262 kHz or 375 kHz)(*)		$f_X/2^6$ (65.5 kHz or 93.8 kHz)(*)	
1	1	$f_X/2^3$ (524 kHz or 750 kHz)(*)			

* The values in parentheses are for f_X = 4.19 MHz or 6.0 MHz.

Serial interface operation mode selection bit (W)

CSIM04	CSIM03	CSIM02	Operation mode	Bit order of shift register 0	SO0 pin function	SI0 pin function
x	0	0	3-wire serial I/O mode	$SIO0_{7-0} \leftrightarrow XA$ (Transfer starting with MSB)	SO0/P02 (CMOS output)	SI0/P03 (Input)
		1		$SIO0_{0-7} \leftrightarrow XA$ (Transfer starting with LSB)		
0	1	0	SBI mode	$SIO0_{7-0} \leftrightarrow XA$ (Transfer starting with MSB)	SB0/P02 (N-ch open-drain I/O)	P03 input
1					P02 input	SB1/P03 (N-ch open-drain I/O)
0	1	1	2-wire serial I/O mode	$SIO0_{7-0} \leftrightarrow XA$ (Transfer starting with MSB)	SB0/P02 (N-ch open-drain I/O)	P03 input
1					P02 input	SB1/P03 (N-ch open-drain I/O)

Remark: x: Don't care

(to be continued)

Fig. 5-52 Format of Serial Operation Mode Register 0 (CSIM0) (Cont'd)

Wake-up function specification bit (W)

WUP	0	Sets IRQCSI0 each time serial transfer is completed in each mode.
	1	Used in the SBI mode only to set IRQCSI0 only when an address received after bus release matches the data in the slave address register (wake-up state). SB0/SB1 goes to high-impedance state.

Caution: When WUP = 1 is set during $\overline{\text{BUSY}}$ signal output, $\overline{\text{BUSY}}$ is not released. In the SBI mode, the $\overline{\text{BUSY}}$ signal is output until the next falling edge of the serial clock ($\overline{\text{SCK0}}$) appears after release of $\overline{\text{BUSY}}$ is directed. Before setting WUP = 1, be sure to confirm that the SB0 (or SB1) pin is high after releasing $\overline{\text{BUSY}}$.

Signal from address comparator (R)

	Condition for being cleared (COI = 0)	Condition for being set (COI = 1)
COI(*)	When the slave address register (SVA) does not match the data of shift register 0	When the slave address register (SVA) matches the data of shift register 0

* COI can be read only before serial transfer is started or after serial transfer is completed. An undefined value may be read during transfer.

COI data written by an 8-bit manipulation instruction is ignored.

(to be continued)

Fig. 5-52 Format of Serial Operation Mode Register 0 (CSIM0) (Cont'd)

Serial interface operation enable/disable specification bit (W)

		Shift register 0 operation	Serial clock counter	IRQCSI0 flag	SO0/SB0, SI0/SB1 pin
CSIE0	0	Shift operation disabled	Cleared	Held	Used only for port 0
	1	Shift operation enabled	Count operation	Can be set.	Used in each mode as well as for port 0

Remarks 1. Each mode can be selected by setting CSIE0, CSIM03, and CSIM02.

CSIE0	CSIM03	CSIM02	Operation mode
0	x	x	Operation halt mode
1	0	x	Three-wire serial I/O mode
1	1	0	SBI mode
1	1	1	Two-wire serial I/O mode

2. The P01/$\overline{\text{SCK0}}$ pin assumes the following state according to the setting of CSIE0, CSIM01, and CSIM00:

CSIE0	CSIM01	CSIM00	P01/$\overline{\text{SCK0}}$ pin state
0	0	0	Input port
0	0	1	High level output
0	1	0	
0	1	1	
1	0	0	High impedance
1	0	1	Serial clock output (High level output)
1	1	0	
1	1	1	

Remarks 3. When clearing CSIE0 during serial transfer, use the following procedure:

① Disable interrupts by clearing the interrupt enable flag (IECSIO).

② Clear CSIE0.

③ Clear the interrupt request flag (IRQCSIO).

Example 1: $f_X/2^4$ is selected as the serial clock, serial interrupt IRQCSIO, is generated each time serial transfer is completed, and serial transfer is performed in the SBI mode with the SB0 pin used as the serial data bus.

```
SEL   MB15            ; Or CLR1  MBE
MOV   XA,#10001010B
MOV   CSIM0,XA        ; CSIM0 ← 10001010B
```

Example 2: Serial transfer dependent on the contents of CSIM0 is enabled.

```
SEL    MB15           ; Or CLR1  MBE
SET1   CSIE0
```

(2) Serial bus interface control register (SBIC)

Figure 5-53 shows the format of the serial bus interface control register (SBIC).

SBIC is an 8-bit register consisting of bits for controlling the serial bus and flags for indicating the states of input data from the serial bus. SBIC is used mainly in the SBI mode.

SBIC is manipulated using a bit manipulation instruction. SBIC cannot be manipulated using a 4-bit or 8-bit memory manipulation instruction.

Each bit may or may not allow read and/or write operation (Figure 5-53).

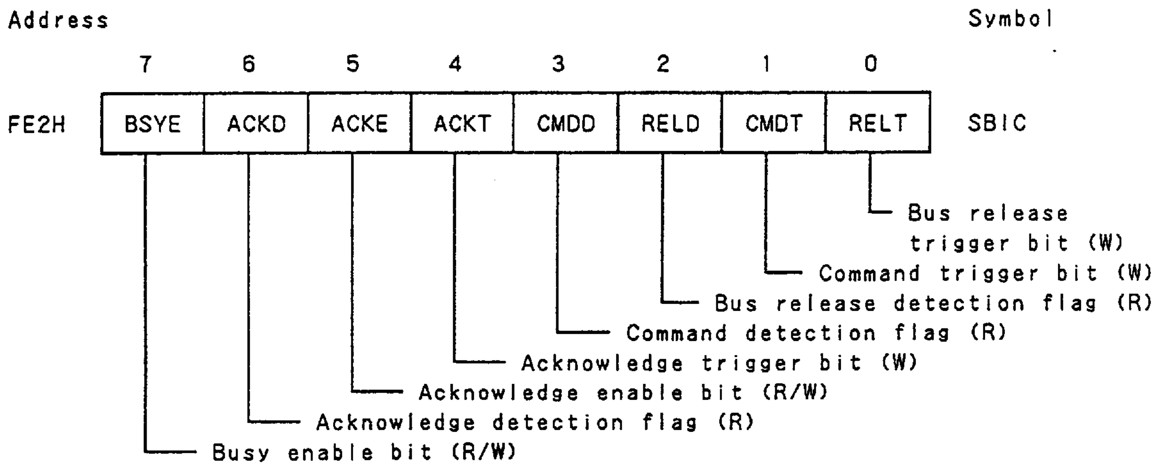
When the $\overline{\text{RESET}}$ signal is input, this register is set to 00H.

Caution: Only the following bits can be used in the three-wire and two-wire serial I/O modes:

. Bus release trigger bit (RELT): Sets the SO0 latch.

. Command trigger bit (CMDT): Clears the SO0 latch.

Fig. 5-53 Format of Serial Bus Interface Control Register (SBIC)


Address
Symbol
7
6
5
4
3
2
1
0
FE2H
BSYE
ACKD
ACKE
ACKT
CMDD
RELD
CMDT
RELT
SBIC
Bus release trigger bit (W)
Command trigger bit (W)
Bus release detection flag (R)
Command detection flag (R)
Acknowledge trigger bit (W)
Acknowledge enable bit (R/W)
Acknowledge detection flag (R)
Busy enable bit (R/W)

Remark:	(R):	Read only
	(W):	Write only
	(R/W):	Read/write

(to be continued)

Fig. 5-53 Format of Serial Bus Interface Control Register (SBIC) (Cont'd)

Bus release trigger bit (W)

RELT	Control bit for bus release signal (REL) trigger output. By setting RELT = 1, the SO0 latch is set to 1. Then the RELT bit is automatically cleared to 0.

Caution: Never clear SB0 (or SB1) during serial transfer. Be sure to clear SB0 (or SB1) before or after serial transfer.

Command trigger bit (W)

CMDT	Control bit for command signal (CMD) trigger output. By setting CMDT = 1, the SO0 latch is cleared. Then the CMDT bit is automatically cleared.

Caution: Never clear SB0 (or SB1) during serial transfer. Be sure to clear SB0 (or SB1) before or after serial transfer.

Bus release detection flag (R)

	Condition for being cleared (RELD = 0)	Condition for being set (RELD = 1)
RELD	① The transfer start instruction is executed. ② The $\overline{\text{RESET}}$ signal is entered. ③ CSIE0 = 0 (Figure 5-52) ④ SVA does not match SIO0 when an address is received.	The bus release signal (REL) is detected.

(to be continued)

Fig. 5-53 Format of Serial Bus Interface Control Register (SBIC) (Cont'd)

Command detection flag (R)

	Condition for being cleared (CMDD = 0)	Condition for being set (CMDD = 1)
CMDD	① The transfer start instruction is executed. ② The bus release signal (REL) is detected. ③ The $\overline{\text{RESET}}$ signal is entered. ④ CSIE0 = 0 (Figure 5-52)	The command signal (CMD) is detected.

Acknowledge trigger bit (W)

ACKT	When set after transfer, $\overline{\text{ACK}}$ is output in phase with the next $\overline{\text{SCK0}}$. After $\overline{\text{ACK}}$ signal output, this bit is automatically cleared to 0.

Cautions 1. Never set ACKT before or during serial transfer.
2. ACKT cannot be cleared by software.
3. Before setting ACKT, set ACKE = 0.

Acknowledge enable bit (R/W)

ACKE	0	Disables automatic output of the acknowledge signal ($\overline{\text{ACK}}$). (Output by ACKT is possible.)	
	1	When set before transfer	$\overline{\text{ACK}}$ is output in phase with the 9th clock of $\overline{\text{SCK0}}$.
		When set after transfer	$\overline{\text{ACK}}$ is output in phase with $\overline{\text{SCK0}}$ immediately following set instruction execution.

(to be continued)

Fig. 5-53 Format of Serial Bus Interface Control Register (SBIC) (Cont'd)

Acknowledge detection flag (R)

	Cleared to 0 when	Set to 1 when
ACKD	① Transfer starts. ② The $\overline{\text{RESET}}$ signal is entered.	The acknowledge signal ($\overline{\text{ACK}}$) is detected (in phase with the rising edge of $\overline{\text{SCK0}}$).

Busy enable bit (R/W)

BSYE	0	① The busy signal is automatically disabled. ② Busy signal output is stopped in phase with the falling edge of $\overline{\text{SCK0}}$ immediately after clear instruction execution.
	1	The busy signal is output after the acknowledge signal in phase with the falling edge of $\overline{\text{SCK0}}$.

Example 1: A command signal is output.

```
        SEL    MB15    ; Or CLR1  MBE
        SET1   CMDT
```

Example 2: RELD and CMDD are tested to identify the types of received data and the types of processing accordingly. By setting WUP = 1, this interrupt routine is processed only when an address match is found.

```
        SEL   MB15
        SKF   RELD     ; RELD test
        BR    !ADRS
        SKT   CMDD     ; CMDD test
        BR    !DATA
CMD  :        .....    ; Command analysis
DATA :        .....    ; Data processing
ADRS :        .....    ; Address decode
```

(3) Shift register 0 (SIO0)

Figure 5-54 shows the configuration of peripheral hardware of shift register 0. SIO0 is an 8-bit register which performs parallel-serial conversion and serial transfer (shift) operation in phase with the serial clock.

Serial transfer is started by writing data to SIO0.

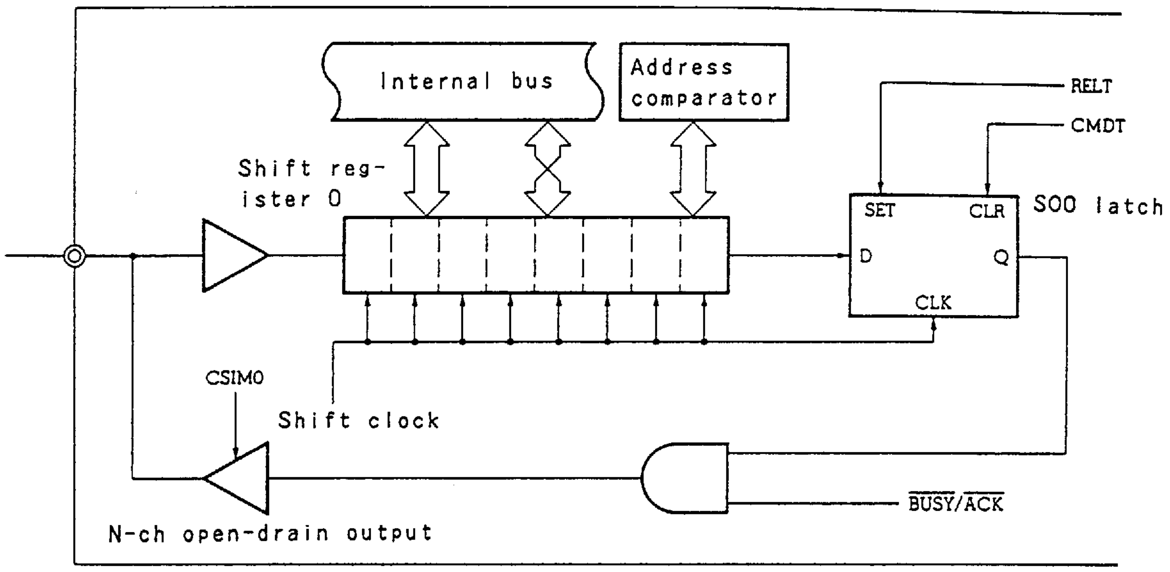
In send operation, data written to SIO0 is output on the serial output (SO0) or serial data bus (SB0/SB1). In receive operation, data is read from the serial input (SI0) or SB0/SB1 into SIO0.

Data can be read from or written to SIO0 by using an 8-bit manipulation instruction.

When the $\overline{\text{RESET}}$ signal is entered during operation, the value of SIO0 is undefined. When the $\overline{\text{RESET}}$ signal is entered in the standby mode, the value of SIO0 is preserved.

Shift operation is stopped after 8-bit send or receive operation is completed.

Fig. 5-54 Peripheral Hardware of Shift Register 0



The timing for reading SIO0 and start of serial transfer (writing to SIO0) is as follows:

. When the serial interface operation enable/disable bit (CSIE0) = 1. However, the case where CSIE0 is set to 1 after data is written to the shift register is excluded.

. When the serial clock is masked after 8-bit serial transfer

. $\overline{\text{SCK0}}$ is high.

When reading from or writing to SIO0, make sure that $\overline{\text{SCK0}}$ is high.

In the two-wire serial I/O mode and SBI mode, the pins specified for the data bus are used for both input and output. Because the configuration of output pins is N-ch open-drain, write FFH in SIO0 for devices that are to receive data.

(4) Slave address register (SVA)

The slave address register (SVA) is an 8-bit register for a slave to set its slave address (number assigned to it).

SVA is manipulated using an 8-bit manipulation instruction. SVA allows only write operation.

When the $\overline{\text{RESET}}$ signal is entered, the value of SVA is undefined. However, the value of SVA is preserved when the $\overline{\text{RESET}}$ signal is entered in the standby mode.

SVA has the following two functions:

(a) Slave address detection (in the SBI mode)

SVA is used when the uPD75238 is connected as a slave device to the serial bus. The master outputs a slave address to the connected slaves to select a particular slave. Two data values (a slave address output from the master and the value of SVA) are compared with each other by the address comparator. If a match is found, the slave is selected.

At this time, bit 6 (COI) of serial operation mode register 0 (CSIM0) is set to 1.

If a match with received address data is not found, the bus release detection flag (RELD) is cleared to 0. When WUP = 1 (wake-up state detection)., IRQCSIO is set only when a match is found. With this interrupt request, the uPD75238 can be informed of a communication request sent from the master.

(b) Error detection (In the two-wire serial I/O mode or SBI mode)

SVA detects an error in either of the following cases:

. When addresses, commands, or data is transferred with the uPD75238 operating as the master

. When data is transferred with the uPD75238 operating as a slave

For details, see (6) in Section 5.8.6 and (8) in Section 5.8.7.

5.8.4 Operation halt mode

The operation halt mode is used when serial transfer is not performed. This mode reduces power consumption.

The shift register does not perform shift operation in this mode, so the shift register can be used as a normal 8-bit register.

When the $\overline{\text{RESET}}$ signal is entered, the operation halt mode is set. The P02/SO0/SB0 pin and P03/SI0/SB1 pin function as input-only port pins. The P01/$\overline{\text{SCK0}}$ pin can be used as an input port pin by setting the serial operation mode register.

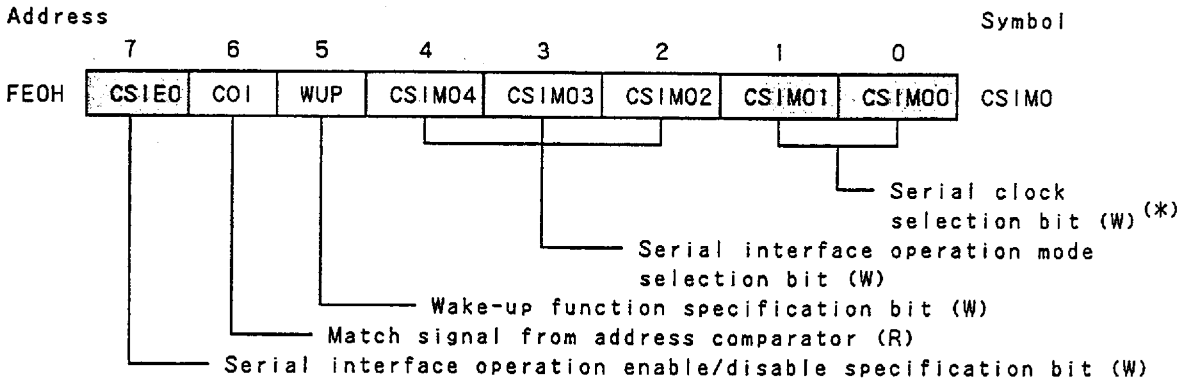
[Register setting]

To set the operation halt mode, manipulate serial operation mode register 0 (CSIM0). (For details on CSIM0, see (1) in Section 5.8.3.)

CSIM0 is manipulated with an 8-bit manipulation instruction. Only the CSIE0 bit of CSIM0 can be independently manipulated. CSIM0 can also be manipulated using the name of each bit.

When the $\overline{\text{RESET}}$ signal is entered, CSIM0 is set to 00H.

In the figure below, hatched portions indicate bits used in the operation halt mode.



Address
Symbol
7 6 5 4 3 2 1 0
FE0H
CSIE0
COI
WUP
CSIM04
CSIM03
CSIM02
CSIM01
CSIM00
CSIM0
Serial clock selection bit (W) (*)
Serial interface operation mode selection bit (W)
Wake-up function specification bit (W)
Match signal from address comparator (R)
Serial interface operation enable/disable specification bit (W)

* The status of the P01/$\overline{\text{SCK0}}$ pin is selectable.

Remark:	(R):	Read only
	(W):	Write only

Serial interface operation enable/disable specification bit (W)

		Shift register 0 operation	Serial clock counter	IRQCSIO flag	SO0/SB0, SI0/SB1 pin
CSIE0	0	Shift operation disabled	Cleared	Held	Used only for port 0

Serial clock selection bit (W)

The P01/$\overline{\text{SCK0}}$ pin assumes the following state according to the setting of CSIM00 and CSIM01:

CSIM01	CSIM00	P01/$\overline{\text{SCK0}}$ pin state
0	0	High impedance
0	1	High level output
1	0	
1	1	

When clearing CSIE0 during serial transfer, use the following procedure:

① Disable interrupts by clearing the interrupt enable flag (IECSIO).

② Clear CSIE0.

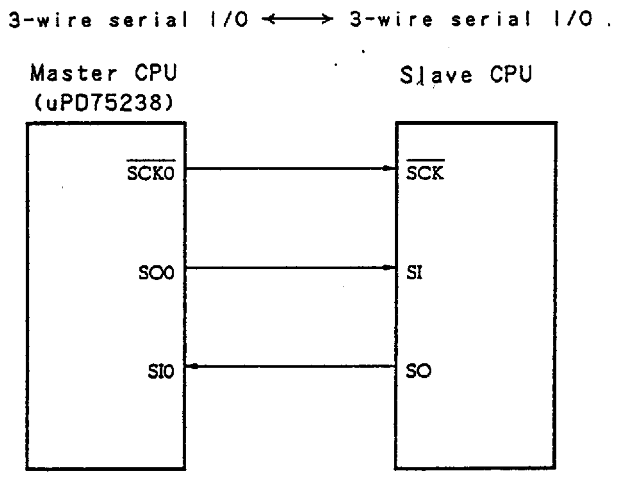
③ Clear the interrupt request flag (IRQCSIO).

5.8.5 Three-wire serial I/O mode operations

The three-wire serial I/O mode is compatible with other modes used in the 75X series, uPD7500 series, and 78K series.

Communication is performed using three lines:
Serial clock ($\overline{\text{SCK0}}$), serial output (SO0), and serial input (SI0).

Fig. 5-55 Example of Three-wire Serial I/O System Configuration



Remark: The uPD75238 can also be used as a slave CPU.

(1) Register setting

To set the three-wire serial I/O mode, manipulate the following two registers:

. Serial operation mode register 0 (CSIM0)
. Serial bus interface control register (SBIC)

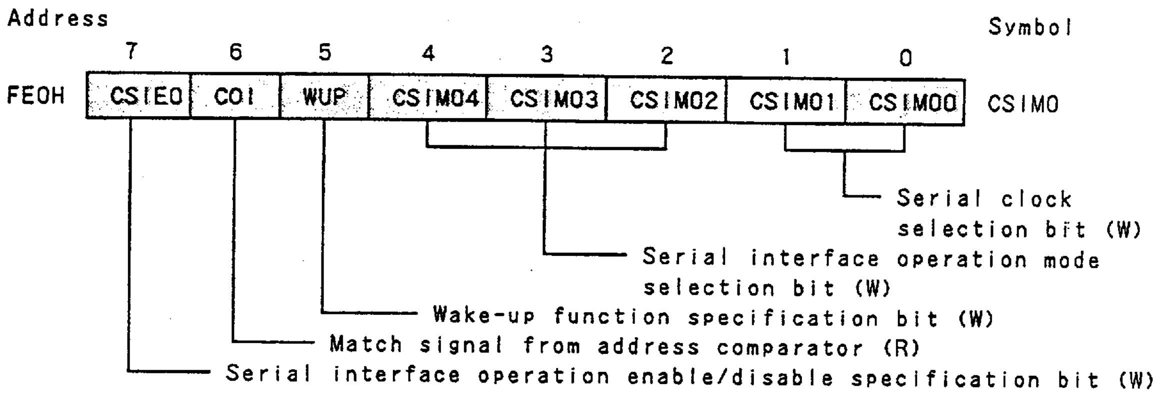
(a) Serial operation mode register 0 (CSIM0)

To use the three-wire serial I/O mode, set CSIM0 as shown below. (For details on CSIM0, see (1) in Section 5.8.3.)

CSIM0 is manipulated using an 8-bit manipulation instruction. Bits 7, 6, and 5 of CSIM0 can be manipulated bit by bit.

When the $\overline{\text{RESET}}$ signal is input, CSIM0 is set to 00H.

In the figure below, hatched portions indicate the bits used in the three-wire serial I/O mode.



Address
Symbol
7 6 5 4 3 2 1 0
FE0H
CSIE0
COI
WUP
CSIM04
CSIM03
CSIM02
CSIM01
CSIM00
CSIM0
Serial clock selection bit (W)
Serial interface operation mode selection bit (W)
Wake-up function specification bit (W)
Match signal from address comparator (R)
Serial interface operation enable/disable specification bit (W)

Remark: (R) : Read only
(W) : Write only

Serial clock selection bit (W)

CSIM01	CSIM00	Serial clock	$\overline{\text{SCK0}}$ pin mode
0	0	External clock applied to $\overline{\text{SCK0}}$ pin	Input
0	1	Timer/event counter output (TO)	Output
1	0	$f_X/2^4$ (262 kHz or 375 kHz)(*)	
1	1	$f_X/2^3$ (524 kHz or 750 kHz)(*)	

* The values at 4.19 MHz and 6.0 MHz are indicated in parentheses.

Serial interface operation mode selection bit (W)

CSIM04	CSIM03	CSIM02	Shift register 0 sequence	SO0 pin function	SI0 pin function
x	0	0	$SIO0_{7-0} \leftrightarrow XA$ (Transfer starting with MSB)	SO0/P02 (CMOS output)	SI0/P03 (Input)
		1	$SIO0_{0-7} \leftrightarrow XA$ (Transfer starting with LSB)		

Remark: x: Don't care

Wake-up function specification bit (W)

WUP	0	Sets IRQCSI0 each time serial transfer is completed.

Signal from address comparator (R)

	Condition for being cleared (COI = 0)	Condition for being set (COI = 1)
COI(*)	When the slave address register (SVA) does not match the data of shift register 0	When the slave address register (SVA) matches the data of shift register 0

* COI can be read only before serial transfer is started or after serial transfer is completed. An undefined value may be read during transfer. COI data written by an 8-bit manipulation instruction is ignored.

Serial interface operation enable/disable specification bit (W)

		Shift register 0 operation	Serial clock counter	IRQCSIO flag	SO0/SB0, SI0/SB1 pin
CSIE0	1	Shift operation enabled	Count operation	Can be set.	Used in each mode as well as for port 0

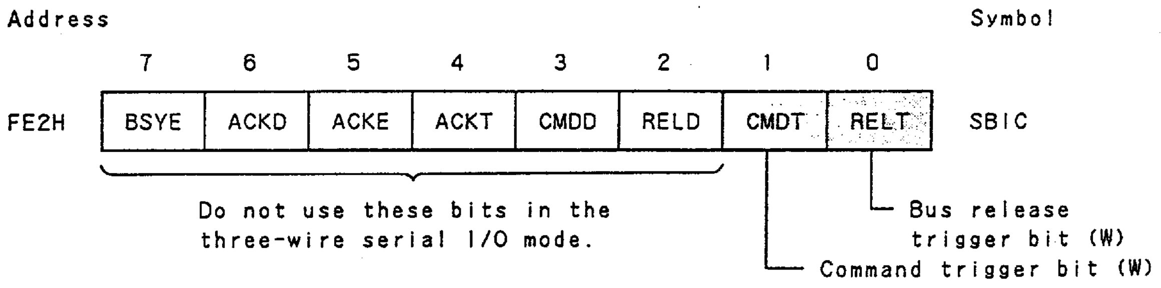
(b) Serial bus interface control register (SBIC)

To use the three-wire serial I/O mode, set SBIC as shown below. (For details on SBIC, see (2) in Section 5.8.3.)

SBIC is manipulated using a bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, SBIC is set to 00H.

In the figure below, hatched portions indicate the bits used in the three-wire serial I/O mode.



Remark: (W): Write only

Bus release trigger bit (W)

RELT	Control bit for bus release signal (REL) trigger output. By setting RELT = 1, the SO0 latch is set to 1. Then the RELT bit is automatically cleared to 0.

Command trigger bit (W)

CMDT	Control bit for command signal (CMD) trigger output. By setting CMDT = 1, the SO0 latch is cleared. Then the CMDT bit is automatically cleared.

Caution: Never use bits other than RELT and CMDT in the three-wire serial I/O mode.

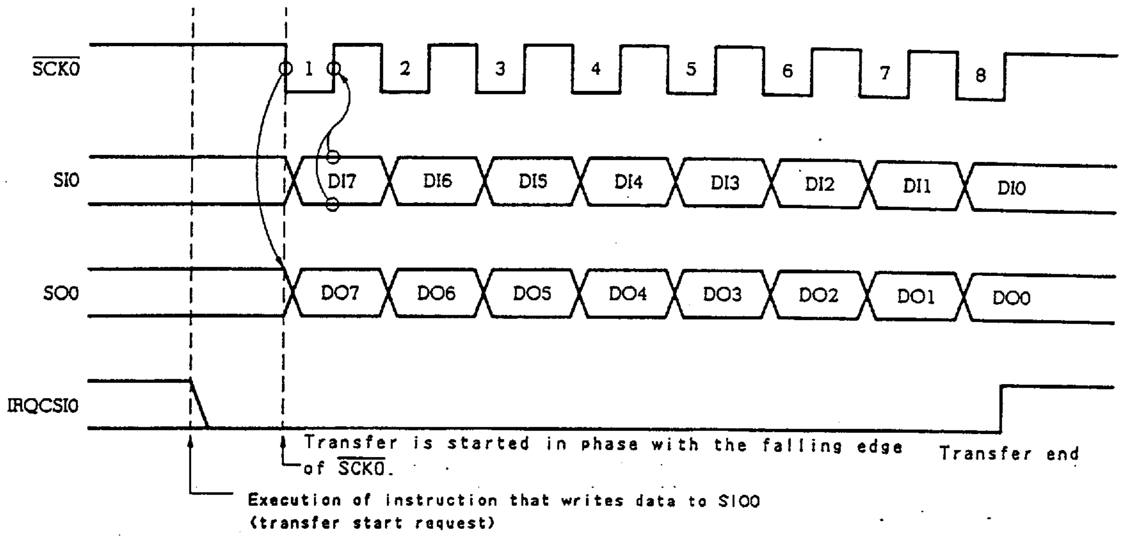
(2) Communication operation

The three-wire serial I/O mode transfers data, with eight bits as one block. Data is transferred bit by bit in phase with the serial clock.

The shift register performs shift operation on the falling edge of the serial clock ($\overline{\text{SCK0}}$). Send data is latched on the SO0 latch, and is output on the SO0 pin. Receive data applied to the SI0 pin is latched in the shift register on the rising edge of $\overline{\text{SCK0}}$.

When eight bits have been transferred, shift register operation automatically terminates setting the interrupt request flag (IRQCSI0).

Fig. 5-56 Timing of Three-wire Serial I/O Mode



$\overline{\text{SCK0}}$
SI0
SO0
IRQCSI0
1
2
3
4
5
6
7
8
DI7
DI6
DI5
DI4
DI3
DI2
DI1
DI0
DO7
DO6
DO5
DO4
DO3
DO2
DO1
DO0
Transfer is started in phase with the falling edge of $\overline{\text{SCK0}}$.
Transfer end
Execution of instruction that writes data to SIO0 (transfer start request)

The SO0 pin becomes a CMOS output and outputs the state of the SO0 latch. So the output state of the SO0 pin can be manipulated by setting the RELT bit and CMDT bit.

However, this manipulation must not be performed during serial transfer operation.

The output level of the $\overline{\text{SCK0}}$ pin can be controlled by manipulating the P01 output latch in the output mode (internal system clock mode). (See Section 5.8.8.)

(3) Serial clock selection

To select the serial clock, manipulate bits 0 and 1 of serial operation mode register 0 (CSIM0). The serial clock can be selected out of the following four clocks:

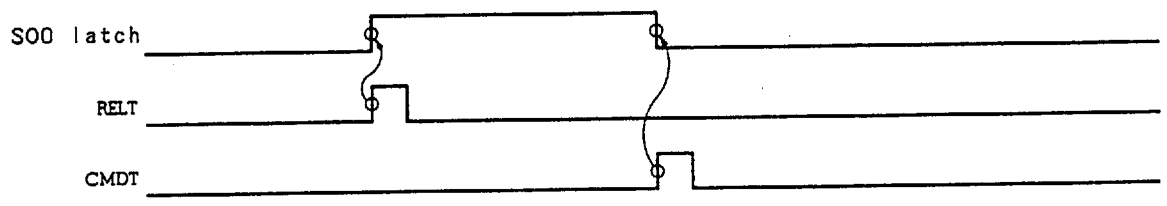
Table 5-10 Serial Clock Selection and Application
(In the Three-wire Serial I/O Mode)

Mode register		Serial clock		Timing for shift register R/W and start of serial transfer	Application
CSIM0 1	CSIM0 0	Source	Masking of serial clock		
0	0	External $\overline{\text{SCK0}}$	Automatically masked when 8-bit data transfer is completed	① In the operation halt mode (CSIE0 = 0) ② When the serial clock is masked after 8-bit transfer ③ When $\overline{\text{SCK0}}$ is high	Slave CPU
0	1	TOUT flip -flop			Half-duplex asynchronous transfer (software control)
1	0	$f_X/2^4$			Middle-speed serial transfer
1	1	$f_X/2^3$			High-speed serial transfer

(4) Signals

Figure 5-57 shows operations of RELT and CMDT.

Fig. 5-57 Operations of RELT and CMDT



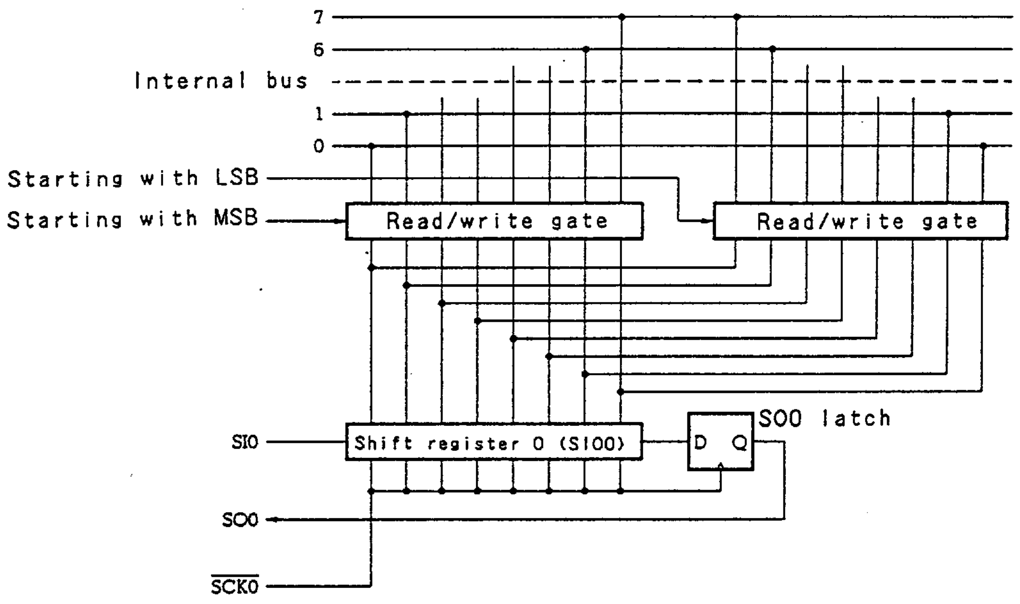
(5) Switching between MSB and LSB as the first transfer bit

The three-wire serial I/O mode has a function that can switch between the MSB and LSB as the first bit of transfer.

Figure 5-58 shows the configuration of shift register 0 (SIO0) and internal bus. As shown in Figure 5-58, read or write operation can be performed by switching between the MSB and LSB.

This switching can be specified using bit 2 of serial operation mode register 0 (CSIM0).

Fig. 5-58 Transfer Bit Switching Circuit



The first bit is switched by changing the order of data bits written to shift register 0 (SIO0). The shift operation order of SIO0 is always the same.

Accordingly, the first bit must be switched between the MSB and LSB before writing data to the shift register.

(6) Transfer start

Serial transfer is started by writing transfer data into shift register 0 (SIO0), provided that the following two conditions are satisfied:

. The serial interface operation enable/disable specification bit (CSIE0) is set to 1.

. The internal serial clock is not operating after 8-bit serial transfer, or $\overline{\text{SCK0}}$ is high.

Caution: Setting CSIE0 after writing data to the shift register does not start transfer.

When eight bits have been transferred, serial transfer automatically terminates setting the interrupt request flag (IRQCSIO).

Example: To transfer the RAM data specified with the HL register to SIO0, load the SIO0 data to the accumulator and start serial transfer operation:

```
MOV  XA,@HL   : Fetch send data from RAM
SEL  MB15     : Or CLR1  MBE
XCH  XA,SIO0  : Exchange send data and receive data,
                and start transfer
```

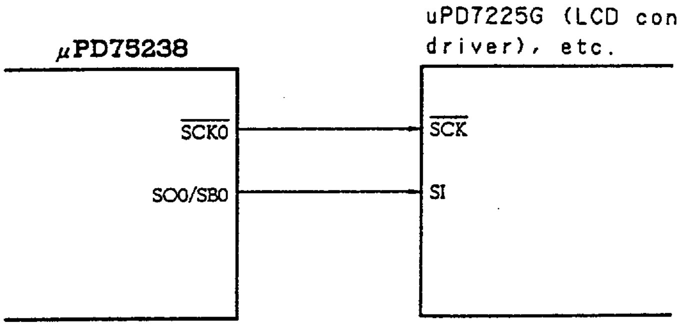
(7) Application of the three-wire serial I/O mode

Example 1: Data is transferred starting with the MSB on a transfer clock of 262 kHz (in 4.19-MHz operation). (Master operation)

<Sample program>

```
CLR1  MBE
MOV   XA,#10000010B
MOV   CSIM0,XA         ; Set transfer mode
MOV   XA,TDATA         ; TDATA is transfer data
                         storage address
MOV   SIO0,XA          ; Set transfer data, and start
                         transfer
```

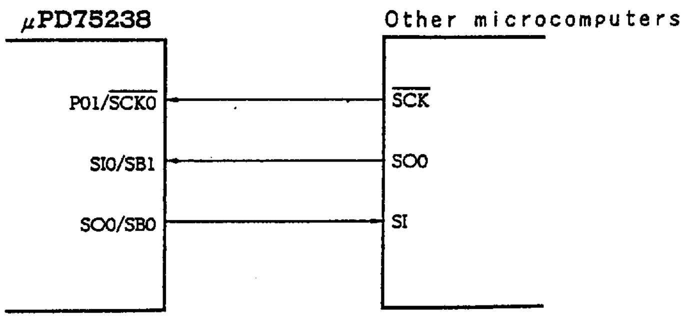
Caution: A second or subsequent transfer operation can be started by setting data in SIO0 (MOV SIO0,XA or XCH XA,SIO0).



In this case, the SI0/SB1 pin on the uPD75238 can be used as an input.

Example 2: Data is sent and received starting with the LSB on an external clock (slave operation).

(In this case, the function of inverting the MSB/LSB is used for shift register read/write operation.)



<Sample program>

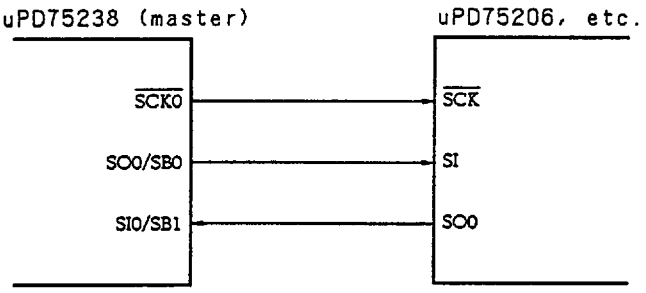
Main routine

```
CLR1   MBE
MOV    XA,#84H
MOV    CSIMO,XA     ; Serial operation halt, MSB/LSB
                      invert mode, external clock
MOV    XA,TDATA
MOV    SIOO,XA      ; Set transfer data, and start
                      transfer
EI     IECSIO
EI
```

Interrupt routine (MBE = 0)

```
MOV    XA,TDATA
XCH    XA,SIOO      ; Receive data — send data, start
                      transfer
MOV    RDATA,XA     ; Save receive data
RETI
```

Example 3: Data is sent and received at high speed by using a transfer clock of 524 kHz (at 4.19 MHz).



<Sample program> (master side):

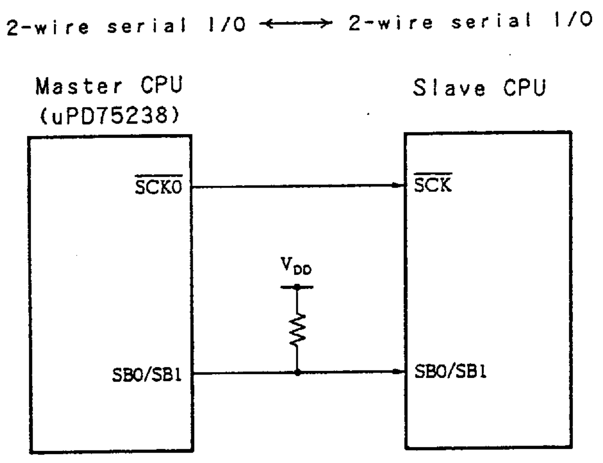
```
        CLR1   MBE
        MOV    XA,#10000011B
        MOV    CSIMO,XA         ; Set transfer mode
        MOV    XA,TDATA
        MOV    SIOO,XA          ; Set transfer data, and
          .                       start transfer
          .
          .
          .
          .
LOOP:SKTCLR    IRQCSIO          ; Test IRQCSIO
     BR        LOOP
     MOV       XA,SIOO          ; Read in receive data
```

5.8.6 Two-wire serial I/O mode

The two-wire serial I/O mode can be made compatible with any communication format by programming.

In this mode, communication is basically performed using two lines: Serial clock ($\overline{SCK0}$) and serial data input/output (SB0 or SB1).

Fig. 5-59 Example of Two-wire Serial I/O System Configuration



Remark: The uPD75238 can also be used as a slave CPU.

(1) Register setting

To set the two-wire serial I/O mode, manipulate the following two registers:

. Serial operation mode register 0 (CSIM0)
. Serial bus interface control register (SBIC)

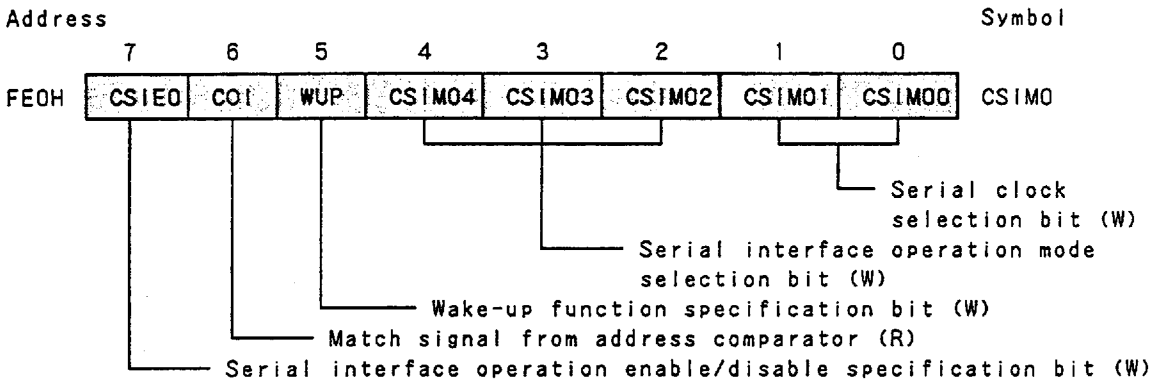
(a) Serial operation mode register 0 (CSIM0)

To use the two-wire serial I/O mode, set CSIM0 as shown below. (For details on CSIM0, see (1) in Section 5.8.3.)

CSIM0 is manipulated using an 8-bit manipulation instruction. Bits 7, 6, and 5 of CSIM0 can be manipulated bit by bit.

When the $\overline{\text{RESET}}$ signal is input, CSIM0 is set to 00H.

In the figure below, hatched portions indicate the bits used in the two-wire serial I/O mode.


Address
Symbol
7
6
5
4
3
2
1
0
FE0H
CSIE0
COI
WUP
CSIM04
CSIM03
CSIM02
CSIM01
CSIM00
CSIM0
Serial clock selection bit (W)
Serial interface operation mode selection bit (W)
Wake-up function specification bit (W)
Match signal from address comparator (R)
Serial interface operation enable/disable specification bit (W)

Remark: (R): Read only
(W): Write only

Serial clock selection bit (W)

CSIM01	CSIM00	Serial clock	$\overline{\text{SCK0}}$ pin mode
0	0	External clock applied to $\overline{\text{SCK0}}$ pin	Input
0	1	Timer/event counter output (TO)	Output
1	0	$f_X/2^6$ (65.5 kHz or 93.8 kHz)(*)	
1	1		

* The values at 4.19 MHz and 6.0 MHz are indicated in parentheses.

Serial interface operation mode selection bit (W)

CSIM04	CSIM03	CSIM02	Shift register 0 bit sequence	SO0 pin function	SI0 pin function
0	1	1	$SIO0_{7-0} \longleftrightarrow XA$ (Transfer starting with MSB)	SB0/P02 (N-ch open-drain I/O)	P03 input
1				P02 input	SB1/P03 (N-ch open-drain I/O)

Wake-up function specification bit (W)

WUP	0	Sets IRQCSIO each time serial transfer is completed.

Signal from address comparator (R)

	Condition for being cleared (COI = 0)	Condition for being set (COI = 1)
COI(*)	When the slave address register (SVA) does not match the data of shift register 0	When the slave address register (SVA) matches the data of shift register 0

* COI can be read only before serial transfer is started or after serial transfer is completed. An undefined value may be read during transfer. COI data written by an 8-bit manipulation instruction is ignored.

Serial interface operation enable/disable specification bit (W)

		Shift register 0 operation	Serial clock counter	IRQCSIO flag	SO0/SB0, SI0/SB1 pin
CSIE0	1	Shift operation enabled	Count operation	Can be set.	Used in each mode as well as for port 0

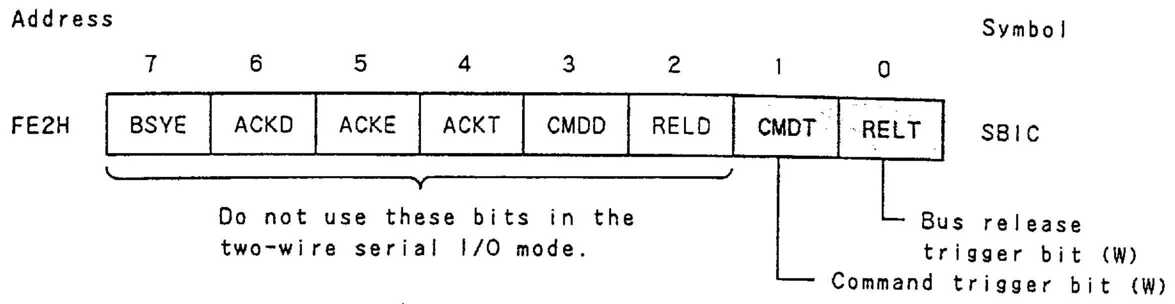
(b) Serial bus interface control register (SBIC)

To use the two-wire serial I/O mode, set SBIC as shown below. (For details on SBIC, see (2) in Section 5.8.3.)

SBIC is manipulated using a bit manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, SBIC is set to 00H.

In the figure below, the hatched portions indicate the bits used in the two-wire serial I/O mode.



Remark: (W): Write only

Bus release trigger bit (W)

RELT	Control bit for bus release signal (REL) trigger output. By setting RELT = 1, the SOO latch is set to 1. Then the RELT bit is automatically cleared to 0.

Command trigger bit (W)

CMDT	Control bit for command signal (CMD) trigger output. By setting CMDT = 1, the SOO latch is cleared. Then the CMDT bit is automatically cleared.

Caution: Never use bits other than RELT and CMDT in the two-wire serial I/O mode.

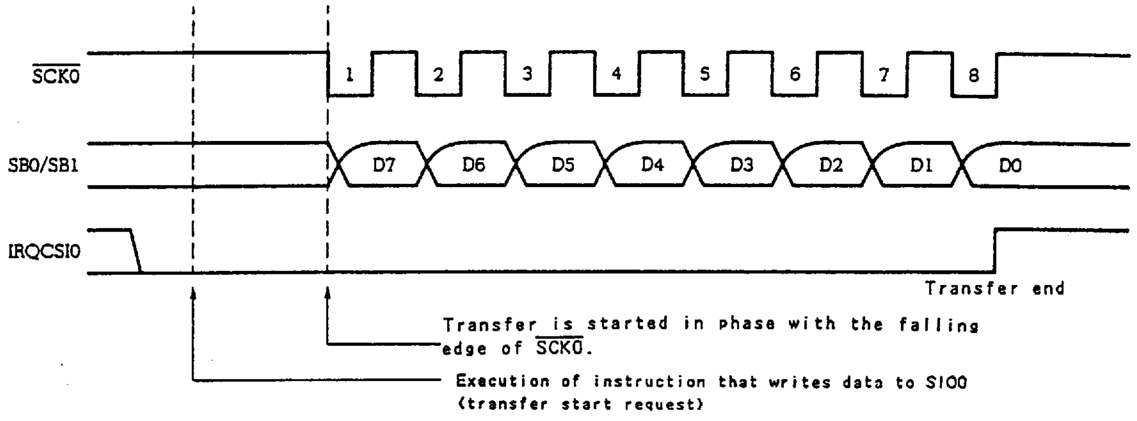
(2) Communication operation

The two-wire serial I/O mode transfers data, with eight bits as one block. Data is transferred bit by bit in phase with the serial clock.

The shift register performs shift operation on the falling edge of the serial clock ($\overline{\text{SCK0}}$). Send data is latched on the SOO latch, and is output on the SB0/P02 pin or SB1/P03 pin starting with the MSB. Receive data applied to the SB0 pin or SB1 pin is latched in the shift register on the rising edge of $\overline{\text{SCK0}}$.

When eight bits have been transferred, shift register operation automatically terminates setting the interrupt request flag (IRQCSI0).

Fig. 5-60 Timing of Two-wire Serial I/O Mode



$\overline{SCK0}$
SB0/SB1
IRQCSI0
1
2
3
4
5
6
7
8
D7
D6
D5
D4
D3
D2
D1
D0
Transfer end
Transfer is started in phase with the falling edge of $\overline{SCK0}$.
Execution of instruction that writes data to SIO0 (transfer start request)

The SB0 or SB1 pin becomes an N-ch open-drain I/O when specified as the serial data bus, so the voltage level on that pin must be pulled up externally. When data is received, the N-ch transistor must be turned off, so FFH must be written to SIO0 beforehand.

The state of the SO0 latch is output on the SB0/SB1 pin, so the SB0/SB1 pin output states can be controlled by setting the RELT or CMDT bit.

However, this operation must not be performed during serial transfer operation.

The output level of the $\overline{SCK0}$ pin can be controlled by manipulating the P01 output latch in the output mode (internal system clock mode). (See Section 5.8.8.)

(3) Serial clock selection

To select the serial clock, manipulate bits 0 and 1 of serial operation mode register 0 (CSIM0). The serial clock can be selected out of the following three clocks:

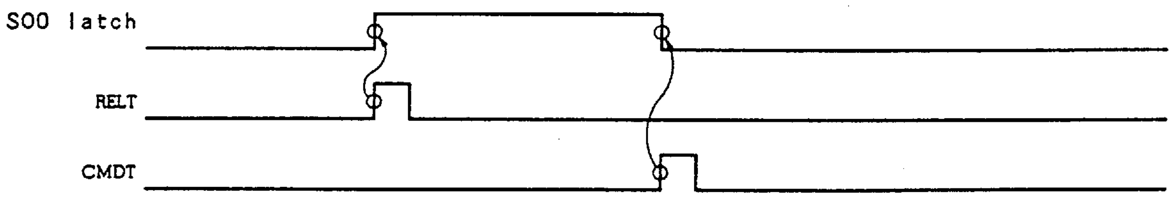
Table 5-11 Serial Clock Selection and Application (In the Two-wire Serial I/O Mode)

Mode register		Serial clock		Timing for shift register R/W and start of serial transfer	Application
CSIM0 1	CSIM0 0	Source	Masking of serial clock		
0	0	External $\overline{\text{SCK0}}$	Automatically masked when 8-bit data transfer is completed	① In the operation halt mode (CSIE0 = 0) ② When the serial clock is masked after 8-bit transfer ③ When $\overline{\text{SCK0}}$ is high	Slave CPU
0	1	TOUT flip-flop			Arbitrary speed serial transfer
1	0	$f_X/2^6$			Low-speed serial transfer
1	0				

(4) Signals

Figure 5-61 shows operations of RELT and CMDT.

Fig. 5-61 Operations of RELT and CMDT



(5) Transfer start

Serial transfer starts by writing transfer data into shift register 0 (SIO0), provided that the following two conditions are satisfied:

. The serial interface operation enable/disable specification bit (CSIE0) is set to 1.

. The internal serial clock is not operating after 8-bit serial transfer, or $\overline{\text{SCK0}}$ is high.

Cautions 1. Setting CSIE0 to 1 after writing data to the shift register does not start transfer.

2. When data is received, the N-ch transistor must be turned off, so FFH must be written to SIO0 beforehand.

When eight bits have been transferred, serial transfer automatically terminates setting the interrupt request flag (IRQCSIO).

(6) Error detection

In the two-wire serial I/O mode, the state of serial bus SB0 or SB1 being used for communication is loaded into the shift register (SIO0) of the transmitting device. So a transmission error can be detected by the methods described below.

(a) Comparing SIO0 data before start of transmission with SIO0 data after start of transmission

With this method, the occurrence of a transmission error is assumed when two SIO0 values disagree with each other.

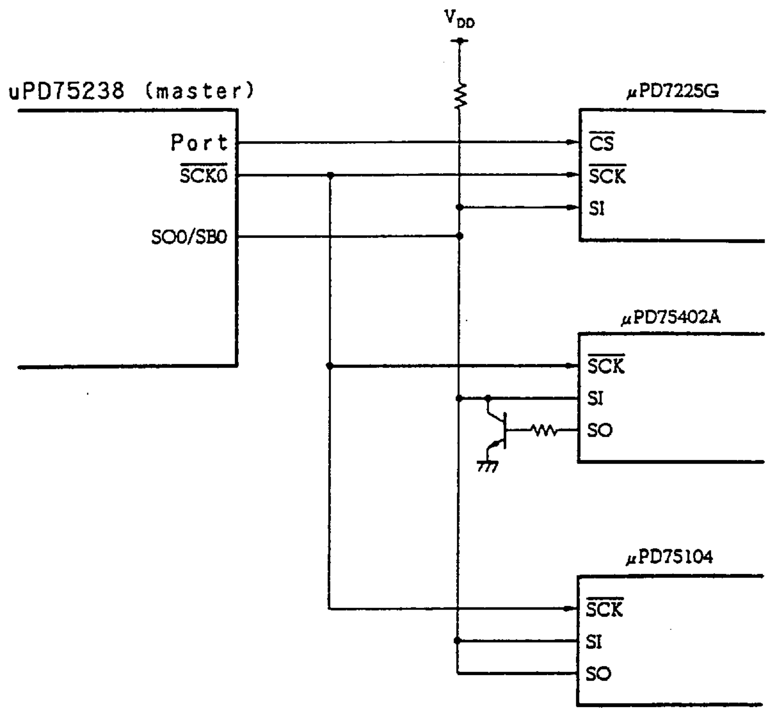
(b) Using the slave address register (SVA)

Send data is set in SIO0 and SVA as well before the data is transmitted. On completion of transmission, the COI bit (match signal from the address comparator) of serial operation mode register 0 (CSIM0) is tested. If the result is 1, the transmission is regarded as successful. If the result is 0, the occurrence of a transmission error is assumed.

(7) Application of two-wire serial I/O mode

A serial bus is configured, and multiple devices are connected to it.

Example: A system is configured with a uPD75238 as the master to which a uPD75104, uPD75402A, and uPD7225G are connected as slaves.



The uPD75104 connects the SI and SO pins, manipulates the serial operation mode register except when serial data is output, and frees the bus by setting off the output buffer.

The SO pin of the uPD75402A cannot go into a high-impedance state, so that a transistor must be connected as shown in the figure to make open collector output appear on the pin. When data is input, 00H must be set beforehand in the shift register to set the transistor off.

The timing of data output by each microcomputer must be predetermined.

The uPD75238, which is the master microcomputer, outputs a serial clock, and all slave microcomputers operate with an external clock.

5.8.7 SBI mode operation

The SBI (serial bus interface) is a high-speed serial interface that conforms to the NEC serial bus format.

To allow communication with multiple devices on a single-master and high-speed serial bus using two signal lines, the SBI has a bus configuration function added to the clock synchronous serial I/O method. So the SBI can reduce ports and wires on boards when multiple microcomputers and peripheral ICs are used to configure a serial bus.

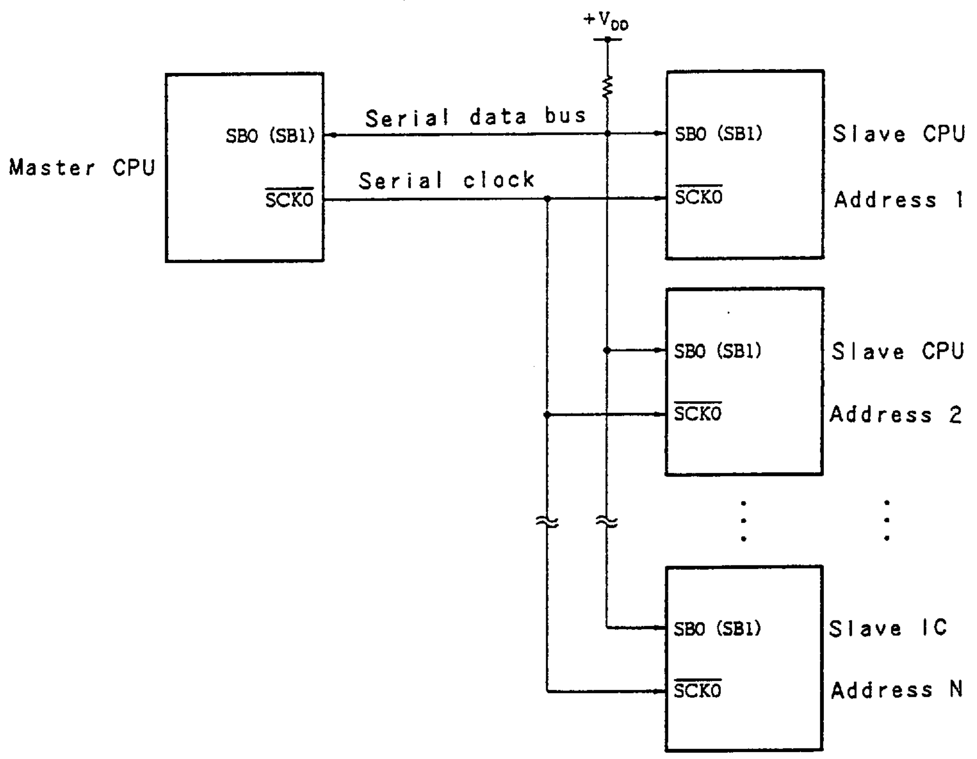
The master can output, on the serial data bus, an address for selecting a device subject to serial communication, commands directed to the remote device, and data. A slave can identify an address, commands, and data from received data by hardware. This function simplifies the serial interface (channel 0) control portion of an application program.

The SBI function is available with devices such as the 75X series and 78K series 8- and 16-bit single chip microcomputers.

Figure 5-62 is an example of the SBI system configuration when the CPU with a serial interface conforming to SBI or peripheral ICs are used.

In the SBI mode, the serial data bus pin SB0 (or SB1) is an open-drain output. So the serial data bus line is placed in the wired OR state. A pull-up resistor is required for the serial data bus line.

Fig. 5-62 Example of SBI System Configuration



Caution: To switch between the master and slave, a pull-up resistor is required also for the serial clock line ($\overline{\text{SCK0}}$), because $\overline{\text{SCK0}}$ input/output switching is performed between the master and slave asynchronously.

(1) SBI functions

Conventional serial I/O methods provide only data transfer functions. Therefore, many ports and wires are required to identify chip select signals, commands, and data, and to detect busy states, when the serial bus is configured with multiple devices. Also, these processes are too burdensome to be controlled by software.

The SBI method can configure a serial bus with two signal lines: Serial clock $\overline{\text{SCK0}}$ and serial data bus SB0/SB1. For this reason, the number of ports on a microcomputer can be reduced and the wiring on a circuit board can be simplified.

SBI functions are described below.

(a) Address/command/data identification function

Serial data is classified into three types: Address, command, and data.

(b) Address-based chip select function

The master selects a chip for a slave by address transfer.

(c) Wake-up function

A slave can easily check address reception (for chip select identification) with the wake-up function. This function can be set or released by software.

When the wake-up function is set, an interrupt (IRQCSIO) is generated when a match address is received.

For this reason, in communication with multiple devices, a CPU other than a selected slave can operate independently of serial communication.

(d) Acknowledge signal ($\overline{\text{ACK}}$) control function

The acknowledge signal, which is used to confirm the reception of serial data, can be controlled.

(e) Busy signal ($\overline{\text{BUSY}}$) control function

The busy signal, which is used to post the busy state of a slave, can be controlled.

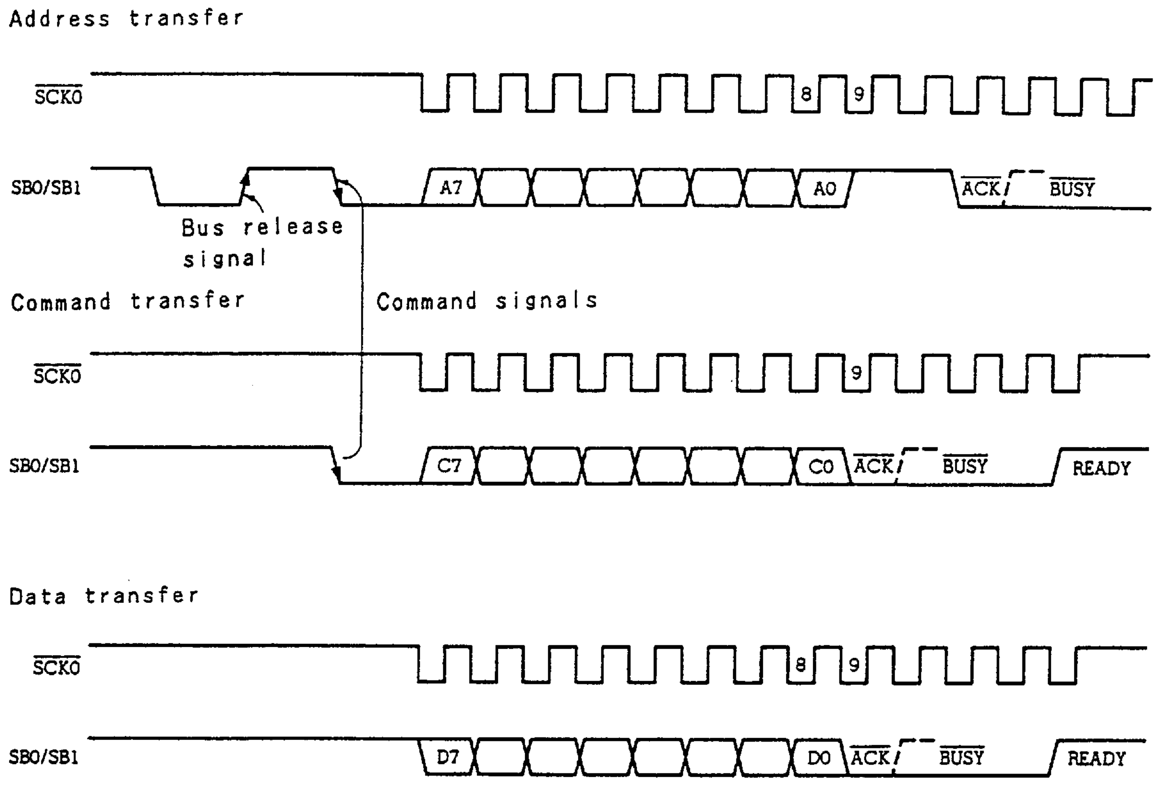
(2) SBI definition

The format of serial data and signal used in the SBI mode are described below.

Serial data to be transferred in the SBI mode is classified into three types: Address, command, and data.

Figure 5-63 is a timing chart for transferring address, command, and data.

Fig. 5-63 Timing of SBI Transfer



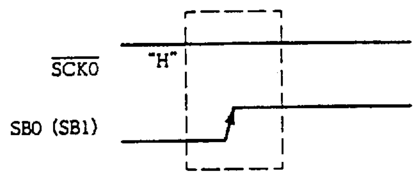
The bus release signal and command signal are output by the master. $\overline{\text{BUSY}}$ is output by a slave. $\overline{\text{ACK}}$ is output by either the master or a slave. (Normally, the device which received 8-bit data outputs $\overline{\text{ACK}}$.)

The master continues to output the serial clock from when 8-bit data transfer starts to when $\overline{\text{BUSY}}$ is released.

(a) Bus release signal (REL)

When the $\overline{\text{SCK0}}$ line is high (the serial clock is not output), the SB0 (or SB1) line changes from low to high. This signal is called the bus release signal, and is output by the master.

Fig. 5-64 Bus Release Signal

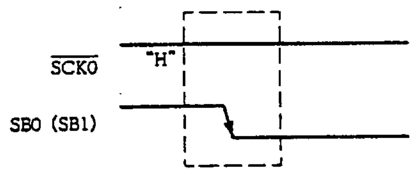


This signal indicates that the master is to send an address to a slave. Slaves contain hardware to detect the bus release signal.

(b) Command signal (CMD)

When the $\overline{\text{SCK0}}$ line is high (the serial clock is not output), the SB0 (or SB1) line changes from high to low. This signal is called the command signal, which is output by the master.

Fig. 5-65 Command Signal

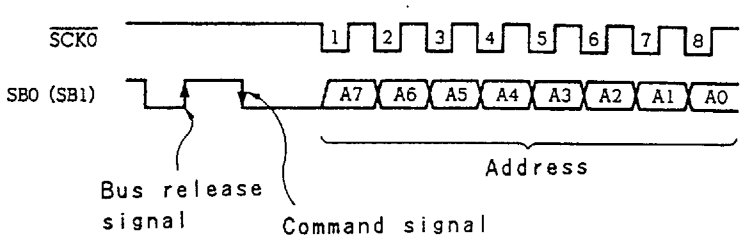


Slaves contain hardware to detect the command signal.

(c) Address

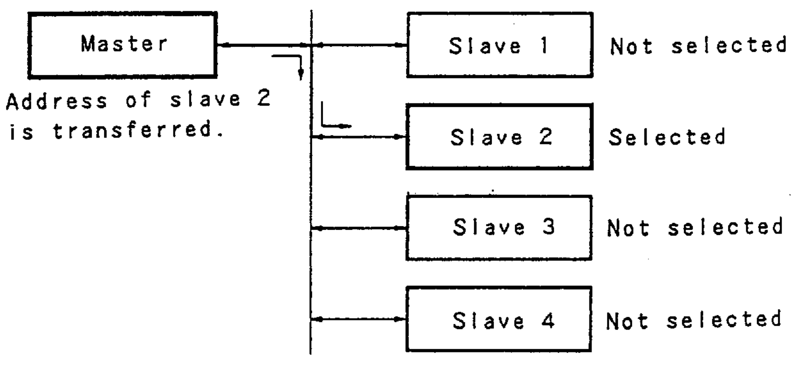
An address is 8-bit data and is output by the master to connected slaves to select a particular slave.

Fig. 5-66 Address



The 8-bit data following the bus release signal or command signal is defined as an address. A slave detects the condition for the addresses by hardware, and checks whether the 8-bit data matches the number assigned to the slave (slave address). If the 8-bit data matches the slave address, that slave is selected. The selected slave continues to communicate with the master until disconnection is directed by the master.

Fig. 5-67 Slave Selection Using an Address



(d) Command and data

The master sends commands to the slave selected by sending an address. The master also transfers data to or from the slave.

Fig. 5-68 Command

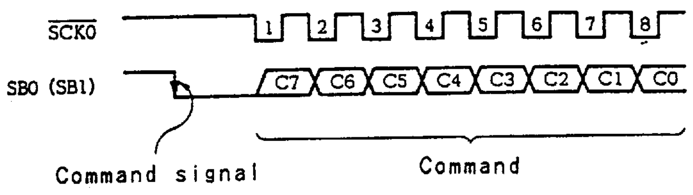
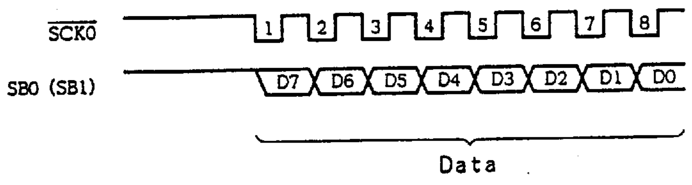


Fig. 5-69 Data



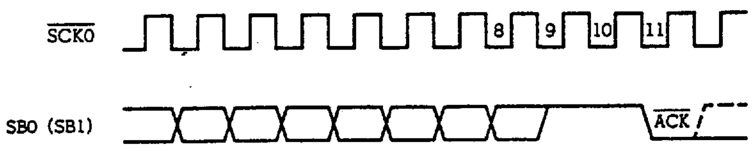
The 8-bit data following the command signal is defined as a command. The 8-bit data without the command signal is defined as data. The usage of commands or data can be selected optionally according to the communication specifications.

(e) Acknowledge signal ($\overline{\text{ACK}}$)

The acknowledge signal confirms the reception of serial data between the sender and the receiver.

Fig. 5-70 Acknowledge Signal

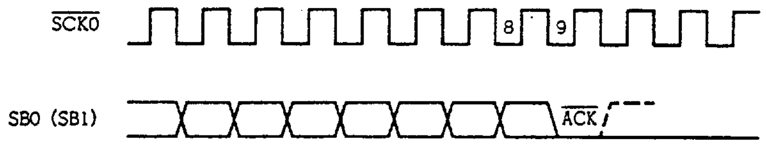
[When output in phase with the 11th clock of $\overline{\text{SCK0}}$]



(to be continued)

Fig. 5-70 Acknowledge Signal (Cont'd)

[When output in phase with the 9th clock of $\overline{\text{SCK0}}$]



The acknowledge signal is a one-shot pulse output in phase with the falling edge of $\overline{\text{SCK0}}$ after 8-bit data transfer. This signal may be synchronized with any clock of $\overline{\text{SCK0}}$.

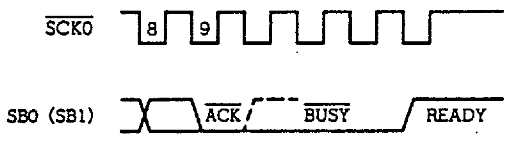
The sender checks if the receiver returns the acknowledge signal after 8-bit data transfer. If the acknowledge signal is not returned after a specified period of time, the sender can assume that the reception failed.

(f) Busy signal ($\overline{\text{BUSY}}$) and ready signal ($\overline{\text{READY}}$)

The busy signal informs the master that a slave is getting ready for data transfer.

The ready signal informs the master that a slave is ready for data transfer.

Fig. 5-71 Busy and Ready Signals



In the SBI mode, a slave notifies the master of the busy state by changing SB0 (or SB1) from high to low.

The busy signal is output following the acknowledge signal output by the master or a slave. The busy signal is set and released in phase with the falling edge of $\overline{\text{SCK0}}$. The master automatically terminates output of serial clock $\overline{\text{SCK0}}$ when the busy signal is released.

The master can transfer the next data when the busy signal is released and a slave enters the state in which the ready signal is to be output.

(3) Register setting

To set the SBI mode, manipulate the following two registers:

. Serial operation mode register 0 (CSIM0)
. Serial bus interface control register (SBIC)

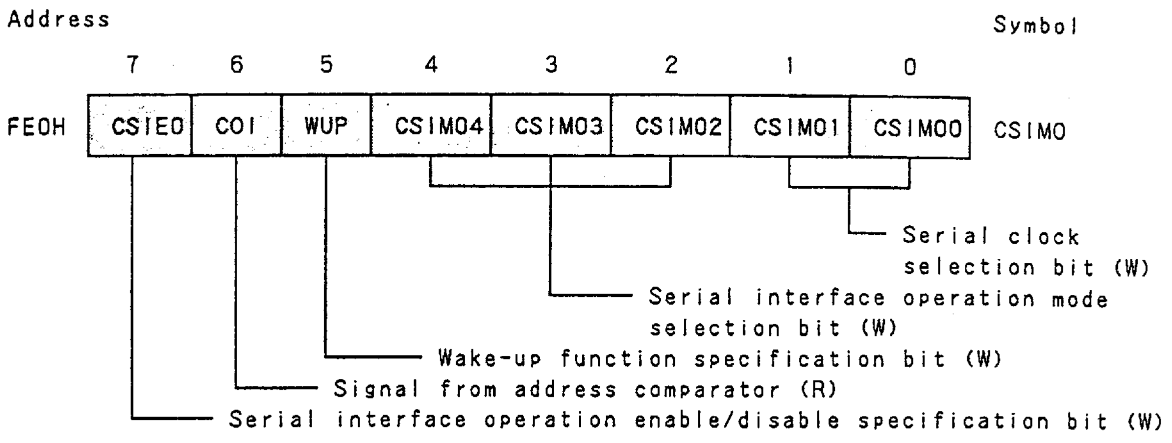
(a) Serial operation mode register 0 (CSIM0)

To use the SBI mode, set CSIM0 as shown below. (For details on CSIM0, see (1) in Section 5.8.3.)

CSIM0 is manipulated using an 8-bit manipulation instruction. Bits 7, 6, and 5 of CSIM0 can be manipulated bit by bit.

When the $\overline{\text{RESET}}$ signal is input, CSIM0 is set to 00H.

In the figure below, hatched portions indicate the bits used in the SBI mode.


Address
Symbol
7 6 5 4 3 2 1 0
FE0H
CSIE0
COI
WUP
CSIM04
CSIM03
CSIM02
CSIM01
CSIM00
CSIM0
Serial clock selection bit (W)
Serial interface operation mode selection bit (W)
Wake-up function specification bit (W)
Signal from address comparator (R)
Serial interface operation enable/disable specification bit (W)

Remark: (R): Read only
(W): Write only

Serial clock selection bit (W)

CSIM01	CSIM00	Serial clock	$\overline{\text{SCK0}}$ pin mode
0	0	External clock applied to $\overline{\text{SCK0}}$ pin	Input
0	1	Timer/event counter output (TO)	Output
1	0	$f_X/2^4$ (262 kHz or 375 kHz) (*)	
1	1	$f_X/2^3$ (524 kHz or 750 kHz) (*)	

* The values at 4.19 MHz and 6.0 MHz are indicated in parentheses.

Serial interface operation mode selection bit (W)

CSIM04	CSIM03	CSIM02	Shift register 0 bit sequence	SO0 pin function	SI0 pin function
0	1	0	SIO07-0 $\leftrightarrow$ XA (Transfer starting with MSB)	SB0/P02 (N-ch open-drain I/O)	P03 input
1				P02 input	SB1/P03 (N-ch open-drain I/O)

Wake-up function specification bit (W)

WUP	0	Sets IRQCSIO each time serial transfer is completed in each mode.
	1	Used in the SBI mode only to set IRQCSIO only when an address received after bus release matches the data in the slave address register (wake-up state). SB0/SB1 goes to high-impedance state.

Caution: When WUP = 1 is set during $\overline{\text{BUSY}}$ signal output, $\overline{\text{BUSY}}$ is not released. In the SBI mode, the $\overline{\text{BUSY}}$ signal is output until the next falling edge of the serial clock ($\overline{\text{SCK}}$) appears after release of $\overline{\text{BUSY}}$ is directed. Before setting WUP = 1, be sure to confirm that the SB0 (or SB1) pin is high after releasing $\overline{\text{BUSY}}$.

Signal from address comparator (R)

	Condition for being cleared (COI = 0)	Condition for being set (COI = 1)
COI (*)	When the slave address register (SVA) does not match the data of shift register 0	When the slave address register (SVA) matches the data of shift register 0

* COI can be read only before serial transfer is started or after serial transfer is completed. An undefined value may be read during transfer.

COI data written by an 8-bit manipulation instruction is ignored.

Serial interface operation enable/disable specification bit (W)

		Shift register 0 operation	Serial clock counter	IRQCSIO flag	SO0/SB0, SI0/SB1 pin
CSIE0	1	Shift operation enabled	Count operation	Can be set.	Used in each mode as well as for port 0

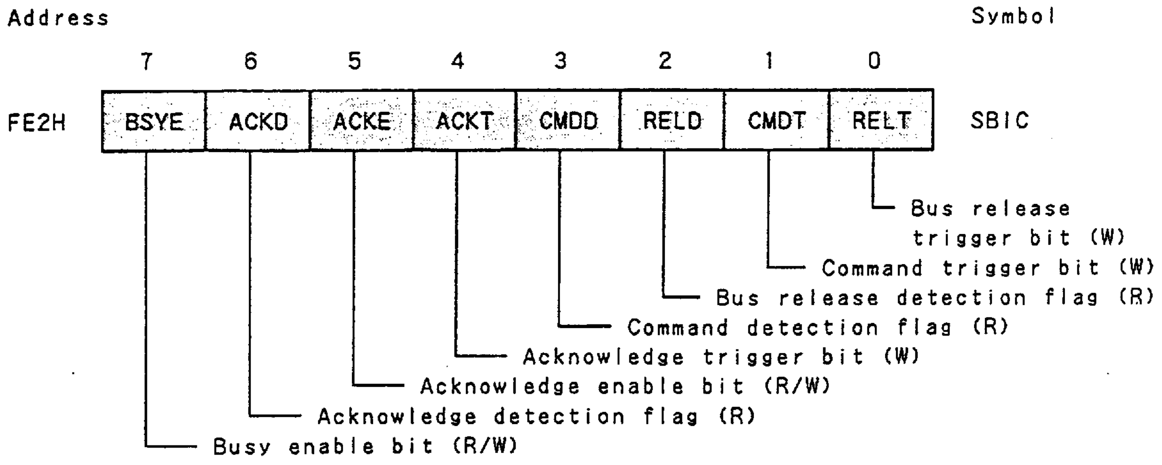
(b) Serial bus interface control register (SBIC)

To use the SBI mode, set SBIC as shown below. (For details on SBIC, see (2) in Section 5.8.3.)

SBIC is manipulated using a bit manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, SBIC is set to 00H.

In the figure below, hatched portions indicate the bits used in the SBI mode.


Address
Symbol
7
6
5
4
3
2
1
0
FE2H
BSYE
ACKD
ACKE
ACKT
CMDD
RELD
CMDT
RELT
SBIC
Bus release trigger bit (W)
Command trigger bit (W)
Bus release detection flag (R)
Command detection flag (R)
Acknowledge trigger bit (W)
Acknowledge enable bit (R/W)
Acknowledge detection flag (R)
Busy enable bit (R/W)

Remark:	(R):	Read only
	(W):	Write only
	(R/W):	Read/write

Bus release trigger bit (W)

RELT	Control bit for bus release signal (REL) trigger output. By setting RELT = 1, the SO0 latch is set to 1. Then the RELT bit is automatically cleared to 0.

Caution: Never clear SB0 (or SB1) during serial transfer. Be sure to clear SB0 (or SB1) before or after serial transfer.

Command trigger bit (W)

CMDT	Control bit for command signal (CMD) trigger output. By setting CMDT = 1, the SO0 latch is cleared. Then the CMDT bit is automatically cleared.

Caution: Never clear SB0 (or SB1) during serial transfer. Be sure to clear SB0 (or SB1) before or after serial transfer.

Bus release detection flag (R)

	Condition for being cleared (RELD = 0)	Condition for being set (RELD = 1)
RELD	① The transfer start instruction is executed. ② The $\overline{\text{RESET}}$ signal is entered. ③ CSIE0 = 0 (Figure 5-52) ④ SVA does not match SIO0 when an address is received.	The bus release signal (REL) is detected.

Command detection flag (R)

	Condition for being cleared (CMDD = 0)	Condition for being set (CMDD = 1)
CMDD	① The transfer start instruction is executed. ② The bus release signal (REL) is detected. ③ The $\overline{\text{RESET}}$ signal is entered. ④ CSIE0 = 0 (Figure 5-52)	The command signal (CMD) is detected.

Acknowledge trigger bit (W)

ACKT	When set after transfer, $\overline{\text{ACK}}$ is output in phase with the next $\overline{\text{SCK0}}$. After $\overline{\text{ACK}}$ signal output, this bit is automatically cleared to 0.

Cautions 1. Never set ACKT before or during serial transfer.
2. ACKT cannot be cleared by software.
3. Before setting ACKT, set ACKE = 0.

Acknowledge enable bit (R/W)

ACKE	0	Disables automatic output of the acknowledge signal ($\overline{\text{ACK}}$). (Output by ACKT is possible.)	
	1	When set before transfer	$\overline{\text{ACK}}$ is output in phase with the 9th clock of $\overline{\text{SCK0}}$.
		When set after transfer	$\overline{\text{ACK}}$ is output in phase with $\overline{\text{SCK0}}$ immediately following set instruction execution.

Acknowledge detection flag (R)

	Cleared to 0 when	Set to 1 when
ACK	① Transfer starts ② The $\overline{\text{RESET}}$ signal is entered.	The acknowledge signal ($\overline{\text{ACK}}$) is detected (in phase with the rising edge of $\overline{\text{SCK0}}$).

Busy enable bit (R/W)

BSYE	0	① The busy signal is automatically disabled. ② Busy signal output is stopped in phase with the falling edge of $\overline{\text{SCK0}}$ immediately after clear instruction execution.
	1	The busy signal is output after the acknowledge signal in phase with the falling edge of $\overline{\text{SCK0}}$.

(4) Serial clock selection

To select the serial clock, manipulate bits 0 and 1 of serial operation mode register 0 (CSIM0). The serial clock can be selected out of the following four clocks:

Table 5-12 Serial Clock Selection and Application (In the SBI Mode)

Mode register		Serial clock		Timing for shift register R/W and start of serial transfer	Application
CSIM0 1	CSIM0 0	Source	Masking of serial clock		
0	0	External $\overline{\text{SCK0}}$	Automatically masked when 8-bit data transfer is completed	① In the operation halt mode (CSIE0 = 0) ② When the serial clock is masked after 8-bit transfer ③ When $\overline{\text{SCK0}}$ is high	Slave CPU
0	1	TOUT flip-flop			Arbitrary speed serial transfer
1	0	$f_X/2^4$			Middle-speed serial transfer
1	1	$f_X/2^3$			High-speed serial transfer

When the internal system clock is selected, $\overline{\text{SCK0}}$ is internally terminated when the 8th clock has been output, and is externally counted until the slave enters the ready state.

(5) Signals

Figures 5-72 to 5-77 show signals to be generated in the SBI mode and flag operations on the SBIC. Table 5-13 lists signals used in the SBI mode.

Fig. 5-72 Operations of RELT, CMDT, RELD, and CMDD (Master)

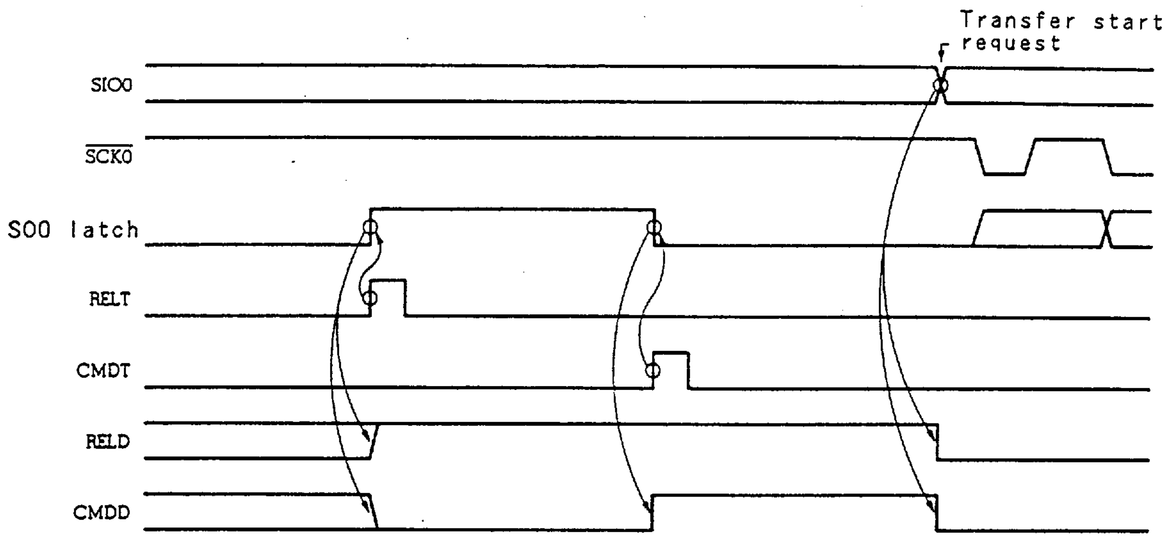
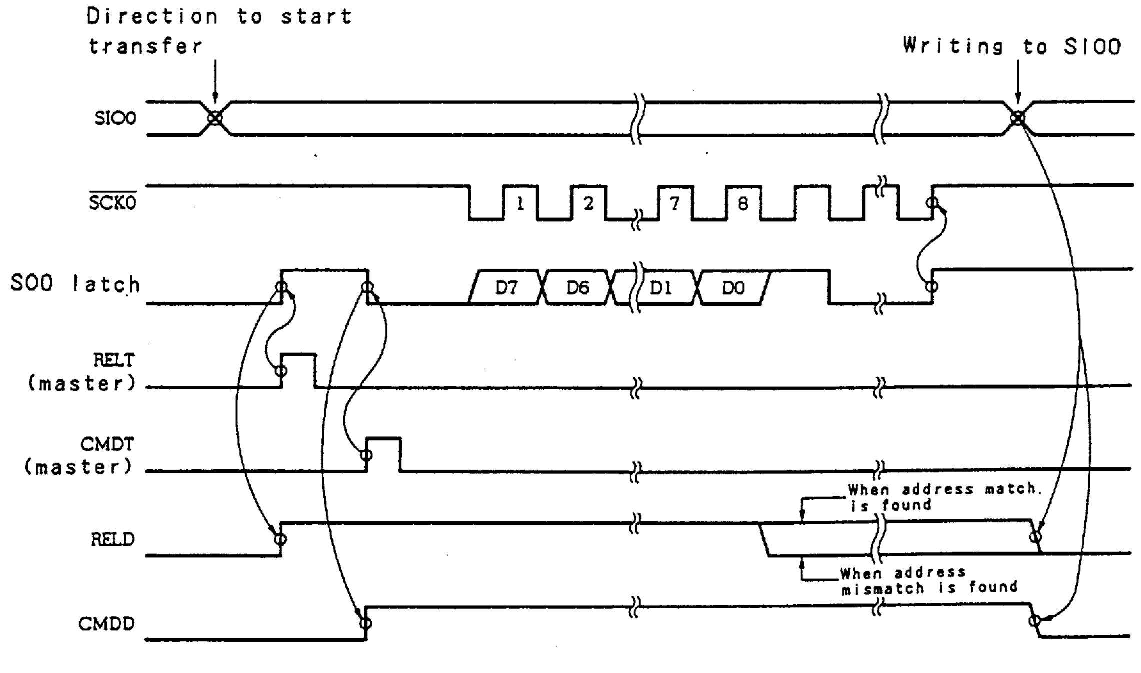
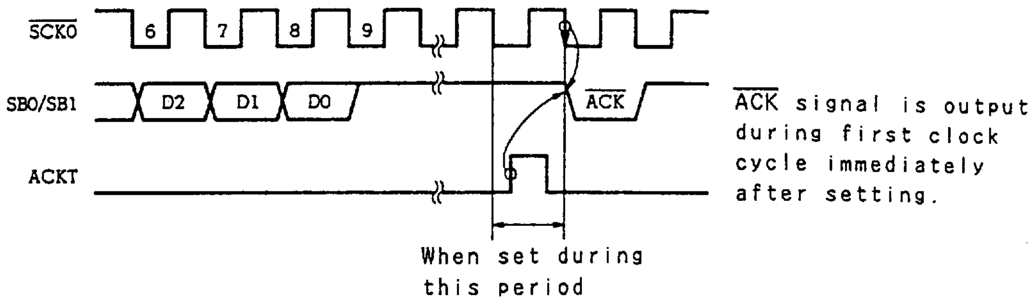


Fig. 5-73 Operations of RELT, CMDT, RELD, and CMDD (Slave)



Direction to start transfer
Writing to SI00
SIO0
SCK0
1
2
7
8
SO0 latch
D7
D6
D1
D0
RELT (master)
CMDT (master)
When address match is found
When address mismatch is found
RELD
CMDD

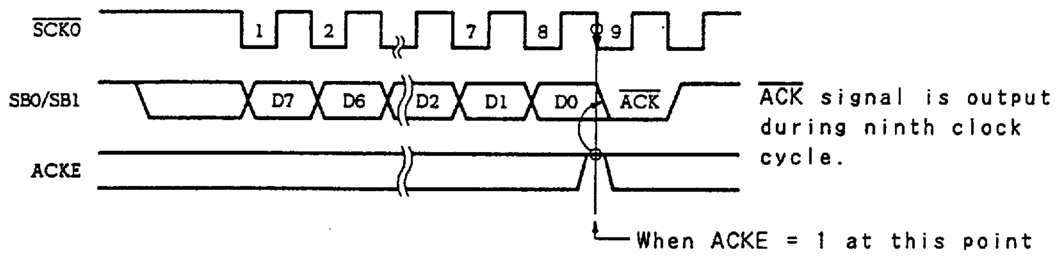
Fig. 5-74 Operation of ACKT


$\overline{SCK0}$
SB0/SB1
ACKT
6
7
8
9
D2
D1
D0
$\overline{ACK}$
$\overline{ACK}$ signal is output during first clock cycle immediately after setting.
When set during this period

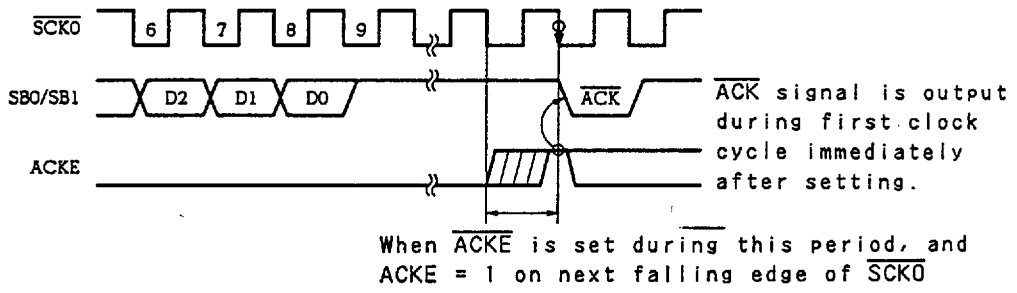
Caution: Do not set the ACKT until the transfer is completed.

Fig. 5-75 Operation of ACKE

a. When ACKE = 1 at time of transfer operation completion


$\overline{SCK0}$
SB0/SB1
ACKE
1
2
7
8
9
D7
D6
D2
D1
D0
$\overline{ACK}$
$\overline{ACK}$ signal is output during ninth clock cycle.
When ACKE = 1 at this point

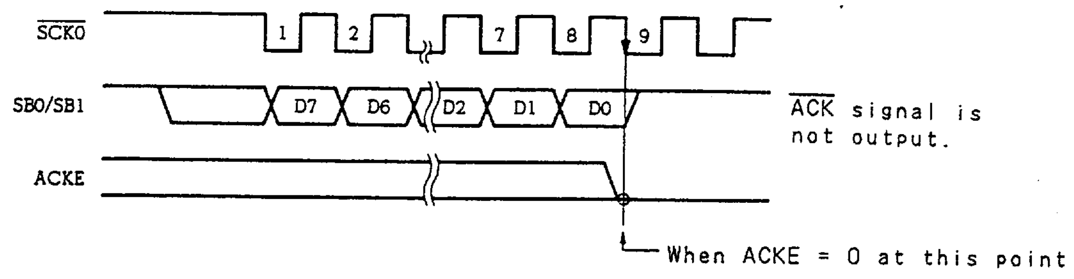
b. When ACKE is set after transfer operation completion


$\overline{SCK0}$
SB0/SB1
ACKE
6
7
8
9
D2
D1
D0
$\overline{ACK}$
$\overline{ACK}$ signal is output during first clock cycle immediately after setting.
When $\overline{ACKE}$ is set during this period, and ACKE = 1 on next falling edge of $\overline{SCK0}$

(to be continued)

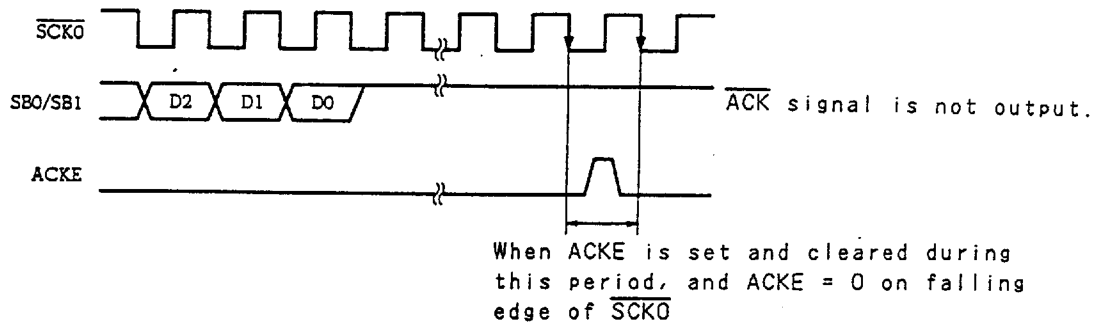
Fig. 5-75 Operation of ACKE (Cont'd)

c. When ACKE = 0 at time of transfer operation completion



$\overline{SCK0}$
SB0/SB1
ACKE
1
2
7
8
9
D7
D6
D2
D1
D0
$\overline{ACK}$ signal is not output.
When ACKE = 0 at this point

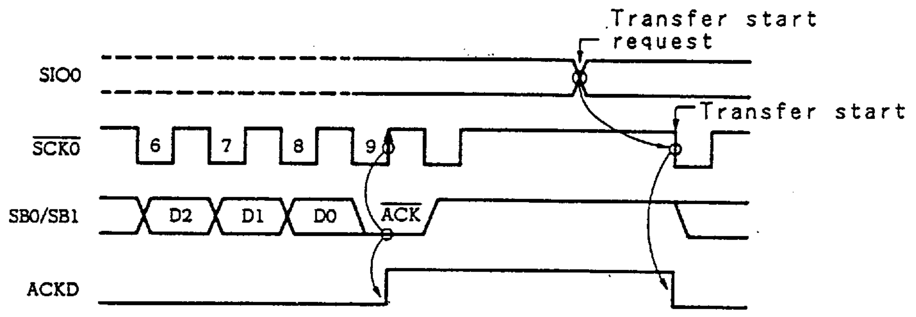
d. When ACKE = 1 period is too short



$\overline{SCK0}$
SB0/SB1
ACKE
D2
D1
D0
$\overline{ACK}$ signal is not output.
When ACKE is set and cleared during this period, and ACKE = 0 on falling edge of $\overline{SCK0}$

Fig. 5-76 Operation of ACKD

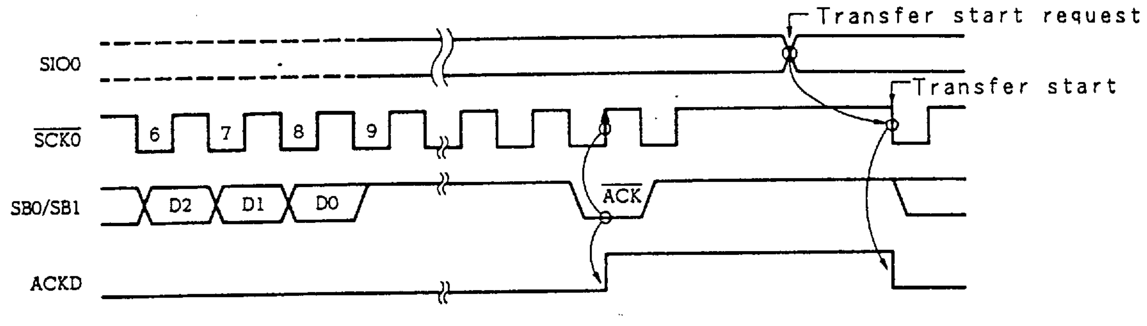
a. When $\overline{ACK}$ signal is output during the ninth $\overline{SCK0}$ clock



(to be continued)

Fig. 5-76 Operation of ACKD (Cont'd)

b. When $\overline{ACK}$ signal is output after the ninth $\overline{SCK0}$ clock



c. Clear timing for case where start of transfer is directed during $\overline{BUSY}$

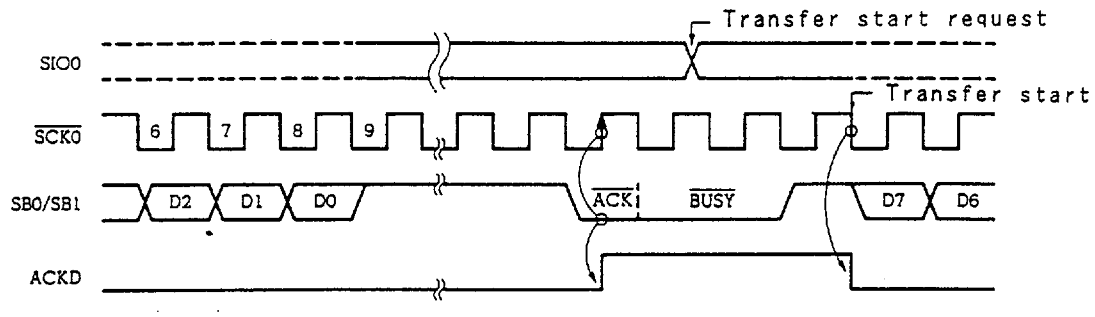


Fig. 5-77 Operation of BSYE

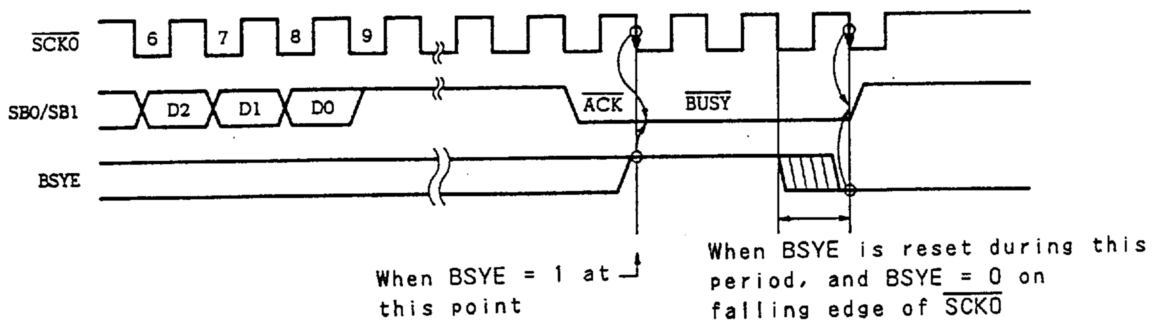
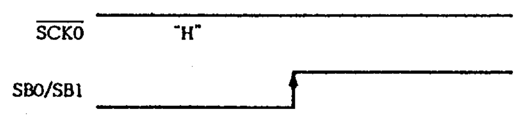
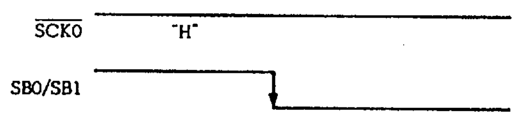
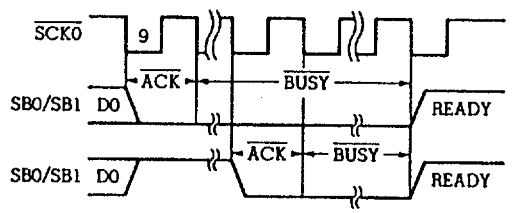
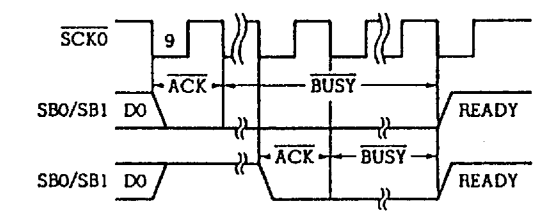
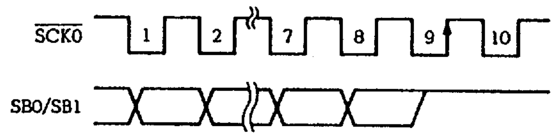
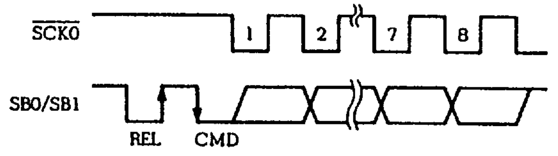

$\overline{SCK0}$
6
7
8
9
SB0/SB1
D2
D1
D0
$\overline{ACK}$
$\overline{BUSY}$
BSYE
When BSYE = 1 at this point
When BSYE is reset during this period, and BSYE = 0 on falling edge of $\overline{SCK0}$

Table 5-13 Various Signals Used in the SBI Mode

Signal name	Output device	Definition	Timing chart	Condition for output	Flag operation	Meaning of signal
Bus release signal (REL)	Master	Rising edge of SBO/SB1 when SCKO=1		. RELT is set.	. RELD is set. . CMDD is cleared.	Indicates that CMD signal will follow this signal, and data transmitted is address data.
Command signal (CMD)	Master	Falling edge on SBO/SB1 when $\overline{SCKO}$=1		. CMDT is set.	. CMDD is set.	i) Data transmitted after REL signal output is address. ii) Data transmitted without REL signal output is command.
Acknowledge signal ($\overline{ACK}$)	Master/ slave	Low-level signal output on SBO/SB1 during one SCKO clock cycle after serial receive operation is completed.	[Sync busy signal output]  $\overline{SCKO}$ 9 SBO/SB1 DO $\overline{ACK}$ $\overline{BUSY}$ READY SBO/SB1 DO $\overline{ACK}$ $\overline{BUSY}$ READY	① ACKE=1 ② ACKT is set.	. ACKD is set.	Completion of receive operation
Busy signal ($\overline{BUSY}$)	Slave	[Sync busy signal] Low-level signal output on SBO/SB1 after acknowledge signal		. BSYE=1	—	Indicates that processing is in progress so the serial receive operation is impossible.

(to be continued)

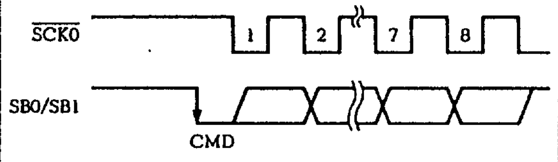
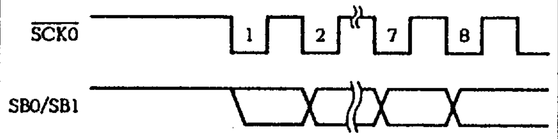
Table 5-13 Various Signals Used in the SBI Mode (Cont'd)

Signal name	Output device	Definition	Timing chart	Condition for output	Flag operation	Meaning of signal
Ready signal (READY)	Slave	High-level signal output on SB0/SB1 before or after serial transfer	[Sync busy signal output]  SCK0 9 ACK BUSY SB0/SB1 D0 READY SB0/SB1 D0 ACK BUSY READY	① BSYE=0 ② Execution of instruction to write data to SIO0 (transfer start request)	—	Indicates that serial receive operation is possible.
Serial clock (SCK0)	Master	Sync clock for outputting address/command/data, ACK signal, sync BUSY signal, etc. Address/command/data are transferred during first 8 clock cycles.		Execution of instruction to write data to SIO0 when CSIE=1 (serial transfer start request)(*2)	IRQCSI0 is set (on the ninth rising edge of SCK0 clock)(*1)	Timing of signal output on serial data bus
Address (A7-A0)	Master	8-bit data transferred in phase with SCK0 after REL signal and CMD signal are output				Address value of slave device on serial bus

(to be continued)

6427525 0094607 940

Table 5-13 Various Signals Used in the SBI Mode (Cont'd)

Signal name	Output device	Definition	Timing chart	Condition for output	Flag operation	Meaning of signal
Command (C7-C0)	Master	8-bit data transferred in phase with SCKO after only CMD signal is output, with REL signal not output		Execution of instruction to write data to SIO0 when CSIE=1 (serial transfer start request)(*2)	IRQCSIO is set (on the ninth rising edge of $\overline{SCKO}$ clock)(*1)	Specification of message to slave device
Data (D7-D0)	Master/ slave	8-bit data transferred in phase with SCKO, with REL and CMD signals not output				Value processed by slave or master device

*1 When WUP = 0, IRQCSIO is always set on the ninth rising edge of the $\overline{SCKO}$ signal.
When WUP = 1, IRQCSIO is set only when the received address matches the value held in the slave address register (SVA).

*2 In the $\overline{BUSY}$ state, transfer operation is initiated after the READY state is set.

(6) Pin configuration

The configurations of serial clock pin $\overline{\text{SCK0}}$ and serial data bus pin (SB0 or SB1) are as follows:

(a) $\overline{\text{SCK0}}$: Pin for serial clock I/O

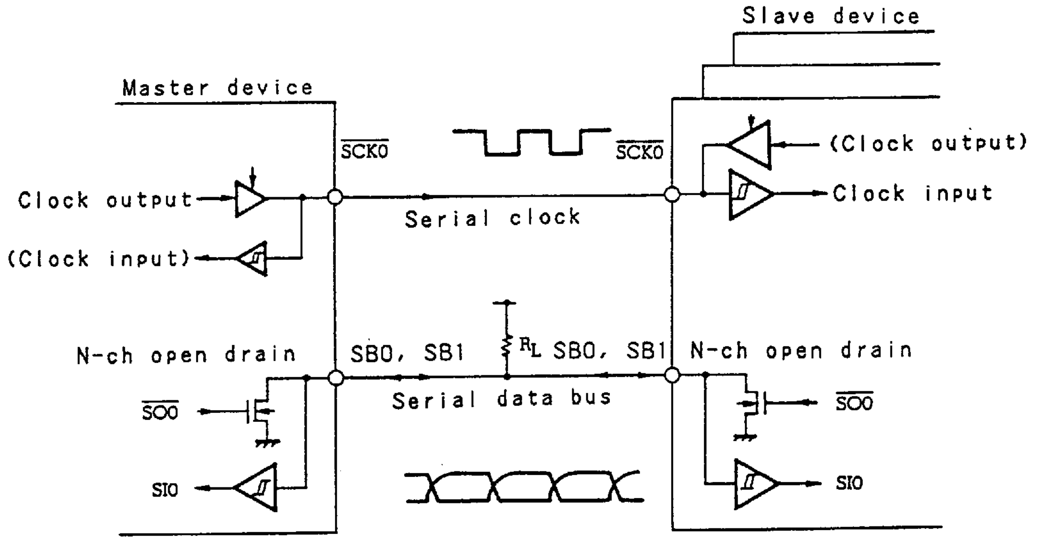
① Master: CMOS, push-pull output
② Slave: Schmitt input

(b) SB0, SB1: Pin for serial data I/O

Output to SB0 or SB1 is an N-ch open-drain output and input is Schmitt input for both the master and a slave.

The serial data bus line becomes an N-ch open-drain output, so it needs to be externally pulled up.

Fig. 5-78 Pin Configuration



Caution: When data is received, the N-ch transistor must be turned off, so FFH must be written to SIO0 beforehand. The N-ch transistor can be turned off at any time during transfer. However, when the wake-up function specification bit (WUP) is set to 1, the N-ch transistor is always off, so there is no need to write FFH to SIO0 before receive operation.

(7) Address match detection method

In the SBI mode, communication starts when the master selects a particular slave device by outputting an address.

An address match is detected by hardware. The slave address register (SVA) is available. In the wake-up state (WUP = 1), IRQCSI0 is set only when the address sent by the master and the value held in SVA match.

Cautions 1. Slave selection or nonselection state is detected by detecting a match for a slave address received after bus release (in the state of RELD = 1).

For this match detection, an address match interrupt (IRQCSI0) generated when WUP is set to 1 is usually used. So detect selection/nonselection state by slave address when WUP is set to 1.

2. When detecting selection/nonselection state without using an interrupt when WUP is 0, do not use the address match detection method. Instead, use transfer of commands set in advance in a program.

(8) Error detection

In the SBI mode, the state of serial bus SB0 or SB1 being used for communication is loaded into the shift register (SIO0) of the transmitting device. So a transmission error can be detected by the methods described below.

(a) Comparing SIO0 data before start of transmission with SIO0 data after start of transmission

With this method, the occurrence of a transmission error is assumed when two SIO0 values disagree with each other.

(b) Using the slave address register (SVA)

Send data is set in SIO0 and SVA as well before the data is transmitted. On completion of transmission, the COI bit (match signal from the address comparator) of serial operation mode register 0 (CSIM0) is tested. If the result is 1, the transmission is regarded as successful. If the result is 0, the occurrence of a transmission error is assumed.

(9) Communication operation

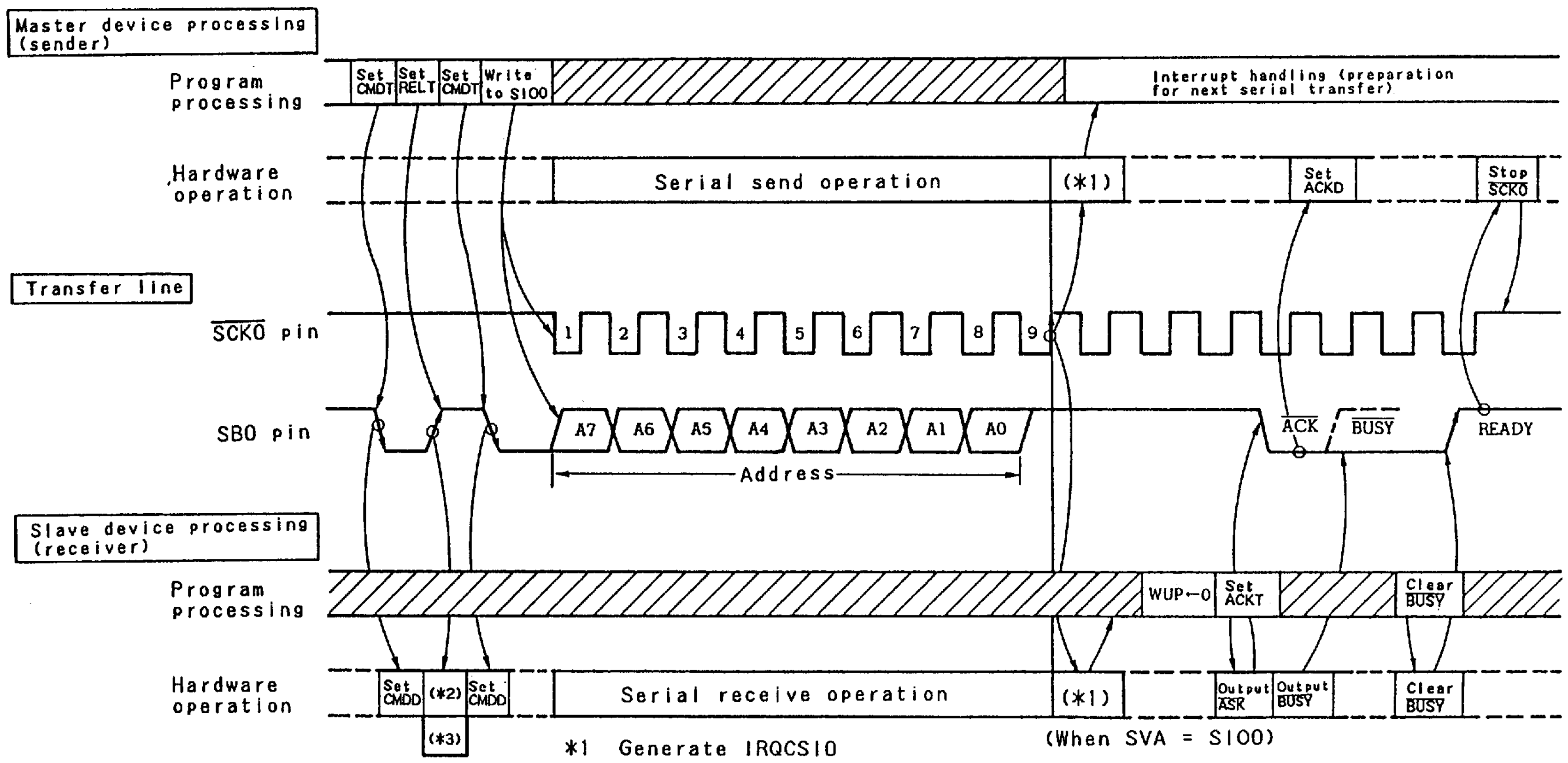
In the SBI mode, the master usually selects a slave device to communicate with from multiple devices by outputting the address of the slave in the serial bus.

After selecting a device to communicate with, the master exchanges commands and data with the slave device, thus establishing serial communication.

Figures 5-79 to 5-82 show the timing charts of data communication operations.

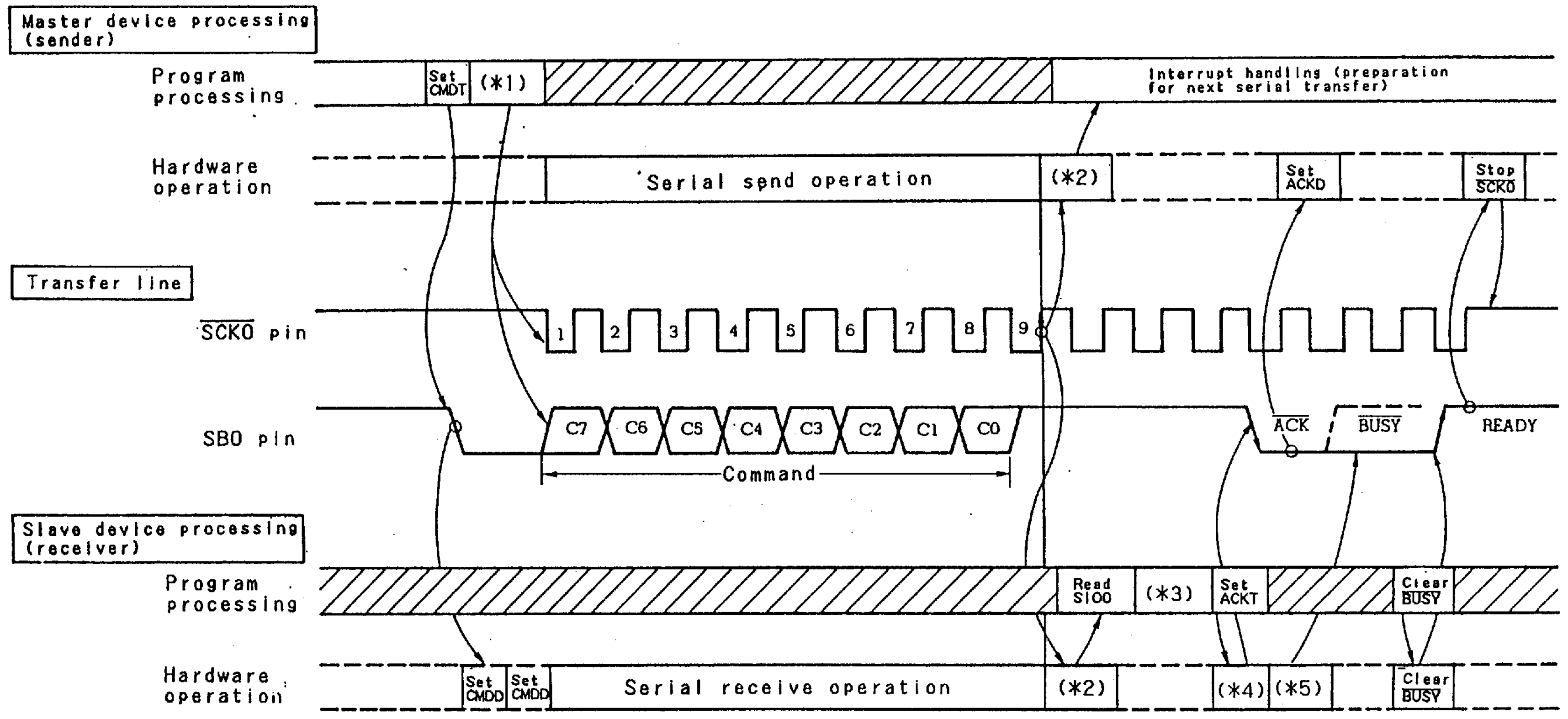
In the SBI mode, the shift register performs shift operation on the falling edge of the serial clock ($\overline{\text{SCK0}}$). Send data is held on the SO0 latch, and is output on the SB0/P02 or SB1/P03 pin starting with the MSB. Receive data applied to the SB0 (or SB1) pin is latched in the shift register on the rising edge of $\overline{\text{SCK0}}$.

Fig. 5-79 Address Transfer Operation from Master Device to Slave Device (WUP = 1)



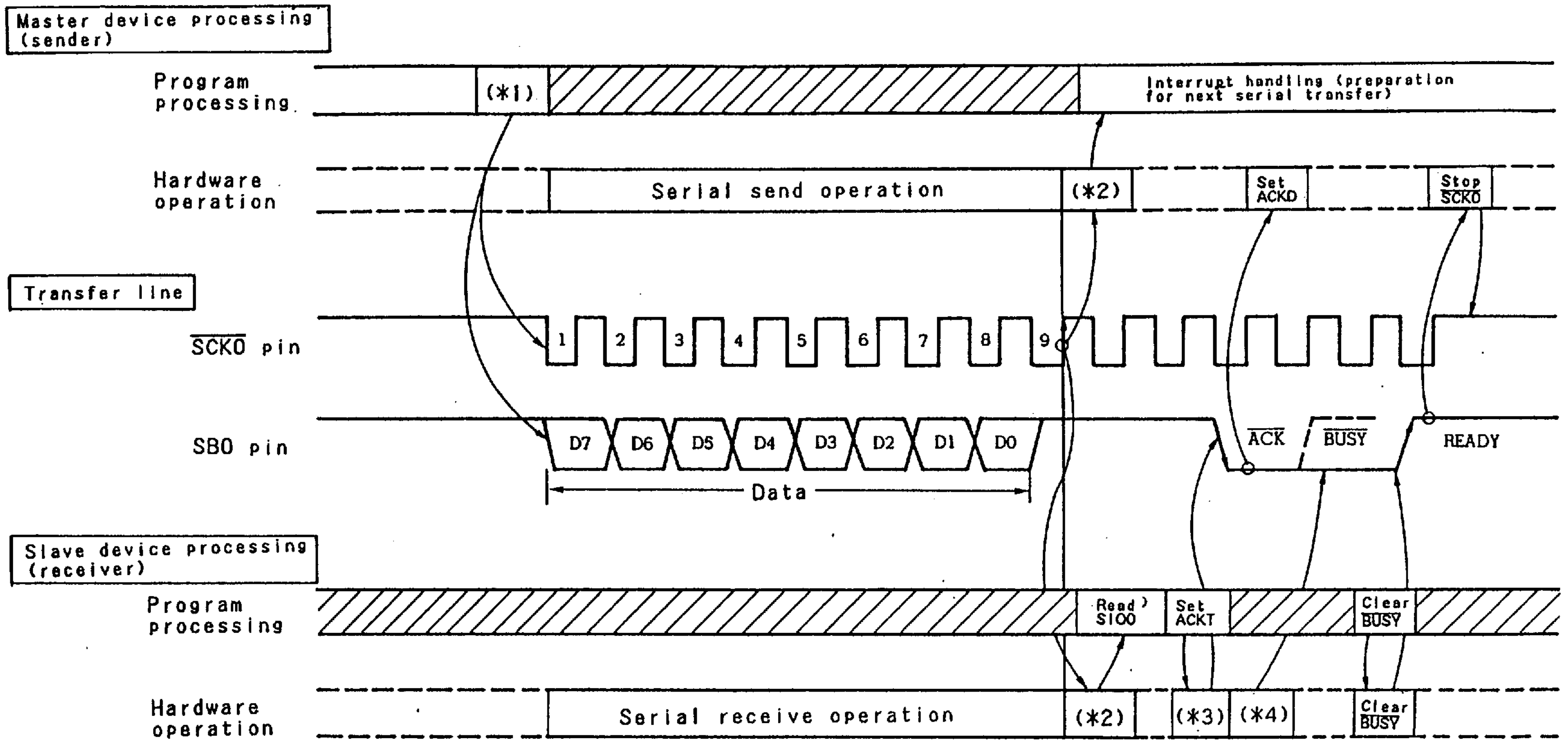
*1 Generate IRQCSIO
*2 Clear CMDD
*3 Set RELD

Fig. 5-80 Command Transfer Operation from Master Device to Slave Device



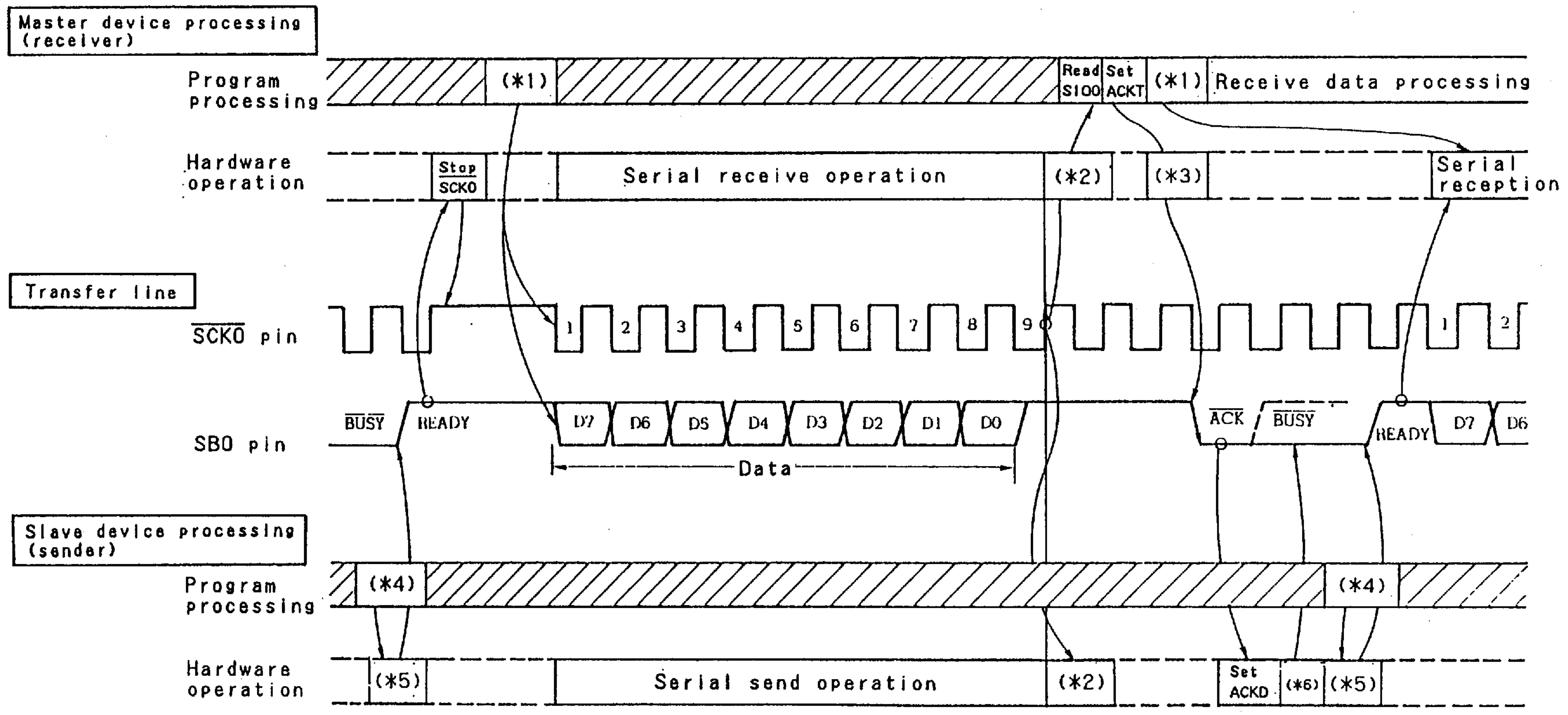
*1 Write to SIO0
*2 Generate IRQCSI0
*3 Analyze command
*4 Output $\overline{\text{ACK}}$
*5 Output $\overline{\text{BUSY}}$

Fig. 5-81 Data Transfer Operation from Master Device to Slave Device



*1 Write to SIO0
*2 Generate IRQCSIO
*3 Output $\overline{\text{ACK}}$
*4 Output $\overline{\text{BUSY}}$

Fig. 5-82 Data Transfer Operation from Slave Device to Master Device



*1 Write FFH to SIO0
*2 Generate IRQCSIO
*3 Output $\overline{\text{ACK}}$
*4 write to SIO0
*5 Clear $\overline{\text{BUSY}}$
*6 Output $\overline{\text{BUSY}}$

(10) Transfer start

Serial transfer is started by writing transfer data in shift register 0 (SIO0), provided that the following two conditions are satisfied:

. The serial interface operation enable/disable bit (CSIE0) is set to 1.

. The internal serial clock is not operating after 8-bit serial transfer, or $\overline{\text{SCK0}}$ is high.

Cautions 1. Transfer operation cannot be started by setting CSIE0 to 1 after writing data to the shift register.

2. The N-ch transistor needs to be turned off when data is received. So FFH must be written to SIO0 beforehand.

However, when the wake-up function specification bit (WUP) is set to 1, the N-ch transistor is always off. So FFH need not be written to SIO0 beforehand for reception.

3. If data is written to SIO0 when the slave is busy, the data is not lost.

Transfer operation is started when the busy state is released and input to SB0 (or SB1) goes high.

When eight bits have been transferred, serial transfer automatically terminates setting the interrupt request flag (IRQCSI0).

Example: When RAM data specified by the HL register is transferred to SIO0, SIO0 data is loaded into the accumulator at the same time, and serial transfer is started.

```
MOV  XA,@HL   ; Extracts send data from RAM
SEL  MB15     ; Or CLR1  MBE
XCH  XA,SIO0  ; Exchanges send data with receive data
                and starts transfer
```

(11) Notes on the SBI mode

(a) Slave selection or nonselection state is detected by detecting a match for a slave address received after bus release (in the state of RELD = 1).

For this match detection, an address match interrupt (IRQCSIO) generated when WUP is 1 is usually used. So detect selection/nonselection state by slave address when WUP is set to 1.

(b) When detecting selection/nonselection state without using an interrupt when WUP = 0, do not use the address match detection method. Instead, use transfer of commands set in advance in a program.

(c) When WUP is set to 1 during $\overline{\text{BUSY}}$ signal output, $\overline{\text{BUSY}}$ is not released. In the SBI mode, after release of $\overline{\text{BUSY}}$ is directed, the $\overline{\text{BUSY}}$ signal is output until the next falling edge of the serial clock ($\overline{\text{SCK0}}$) appears. Before setting WUP to 1, be sure to confirm that the SB0 (or SB1) pin is high after releasing $\overline{\text{BUSY}}$.

(12) SBI mode

This section describes an example of application which performs serial data communication in the SBI mode. In the example, the uPD75238 can be used as either the master CPU or a slave CPU on the serial bus.

The master can be switched to another CPU with a command.

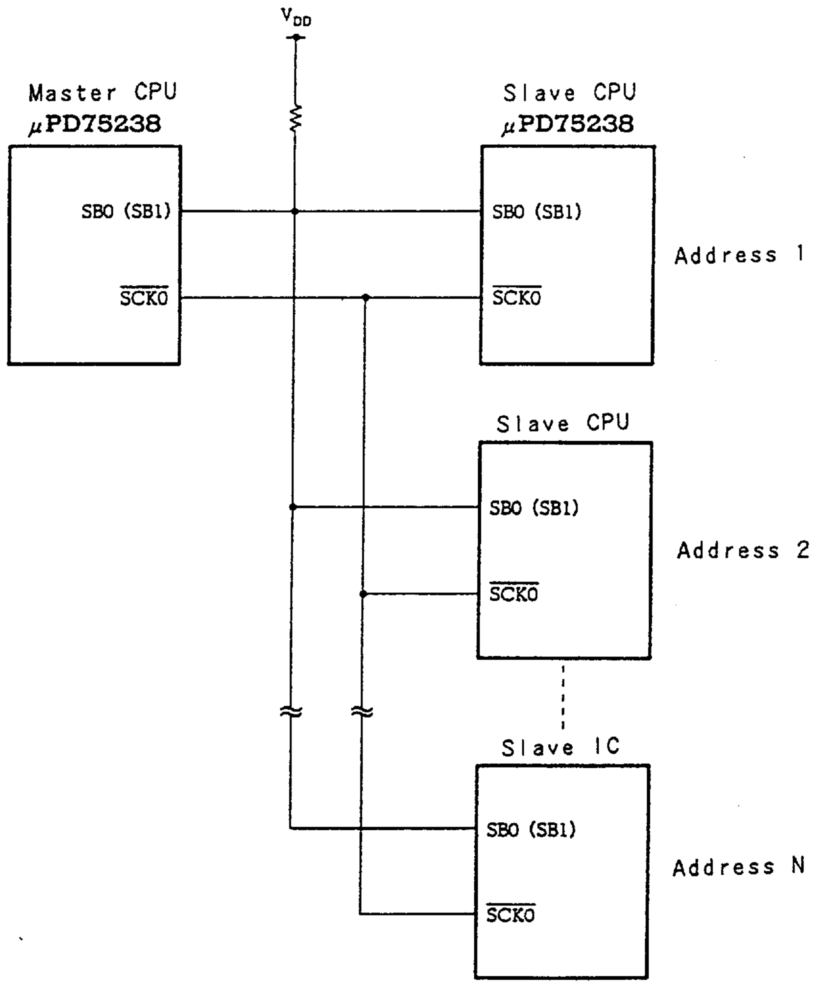
(a) Serial bus configuration

In the serial bus configuration used for the example of this section, a uPD75238 is connected to the bus line as a device on the serial bus.

Two pins on the uPD75238 are used: the serial data bus SB0 (SO2/SO0) and serial clock $\overline{\text{SCK0}}$ (P01).

Figure 5-83 shows an example of the serial bus configuration.

Fig. 5-83 Example of Serial Bus Configuration



(b) Explanation of commands

<Types of commands>

This example uses the following commands:

①	READ command:	Transfers data from slave to master.
②	WRITE command:	Transfers data from master to slave.
③	END command:	Informs slave of WRITE command completion.
④	STOP command:	Informs slave of WRITE command interruption.
⑤	STATUS command:	Reads slave status.
⑥	RESET command:	Sets currently selected slave as non-selected slave.
⑦	CHGMST command:	Passes master authority to slave.

<Protocol>

The following protocol is used for communication between the master and slaves.

① The address of a slave with which the master intends to communicate is transmitted to select the slave (chip select). This starts communication.

The slave that has received the address returns $\overline{\text{ACK}}$ to engage in communication with the master. (The state of the slave is changed from the non-selected state to selected state.)

② Commands and data are transferred between the master and the slave selected in ①.

Command and data are transferred between the master and the selected slave on a one-to-one basis, so the other slaves must be placed in the non-selected state.

③ Communication is completed when the selected slave is placed in the non-selected state. This state is caused in the following cases:

. The selected slave is placed in the non-selected state when the slave receives a RESET command from the master.

. The device that is switched from the master to a slave with a CHGMST command is placed in the non-selected state.

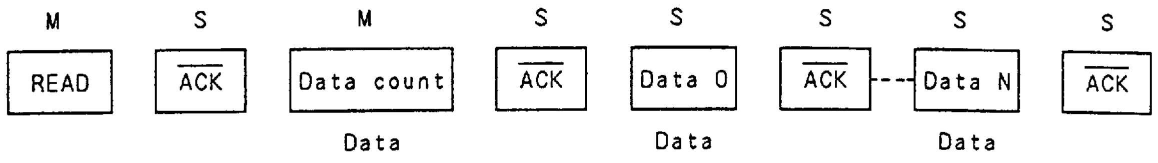
<Command format>

The transfer format of each command is described below.

① READ command

The READ command reads data from a slave. One to 256 bytes of data can be read. The data length is specified in a parameter by the master. When 00H is specified as the data length, the 256-byte data transfer is assumed.

Fig. 5-84 Transfer Format of the READ Command



Remarks: M: Output by the master
S: Output by the slave

If the slave has data enough for transmitting the specified number of bytes of data, the slave returns $\overline{ACK}$. If the slave does not have enough data for transmission, an error occurs; $\overline{ACK}$ is not returned in this case.

The master sends $\overline{ACK}$ to the slave whenever it receives one byte.

② WRITE command, END command, STOP command

These commands write data to a slave.
One to 256 bytes of data can be written.
The data length is specified in a parameter by the master. When 00H is specified as the data length, the 256-byte data transfer is assumed.

Fig. 5-85 Transfer Format of the WRITE and END Commands

M	S	M	S	M	S	M	S	M	S
WRITE	$\overline{\text{ACK}}$	Data count	$\overline{\text{ACK}}$	Data 0	$\overline{\text{ACK}}$ ---	Data N	$\overline{\text{ACK}}$	END	$\overline{\text{ACK}}$
Command		Data		Data		Data		Command	

Remarks: M: Output by the master
S: Output by the slave

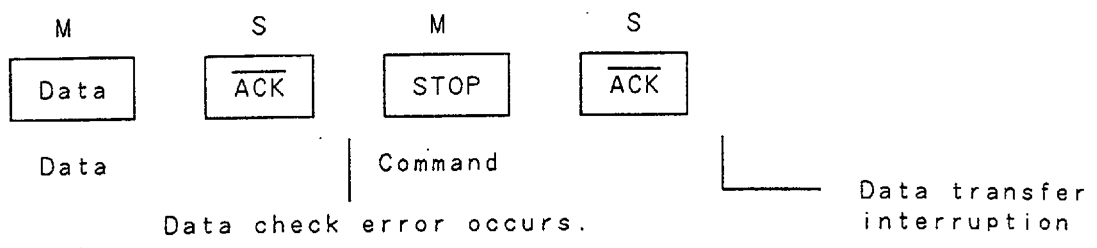
If the slave has an enough area for storing receive data of the specified length, the slave returns $\overline{\text{ACK}}$. If the slave does not have an enough area, an error occurs; $\overline{\text{ACK}}$ is not returned in this case.

The master transmits an END command when all data have been transferred. The END command informs the slave that all data have been transferred correctly.

The slave accepts an END command even before data reception is uncompleted. In this case, the data received just before the acceptance of the END command becomes valid.

The master compares the contents of SIOO before transfer with the contents of SIOO after transfer to check if the data has been output onto the bus correctly. If the contents of SIOO disagree with each other, the master interrupts data transfer by transmitting a STOP command.

Fig. 5-86 Transfer Format of the STOP Command



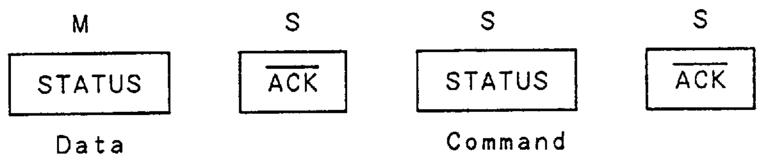
Remarks: M: Output by the master
S: Output by the slave

When the slave receives a STOP command, the slave invalidates the 1-byte data received immediately before the STOP command.

③ STATUS command

The STATUS command reads the status of the slave currently selected.

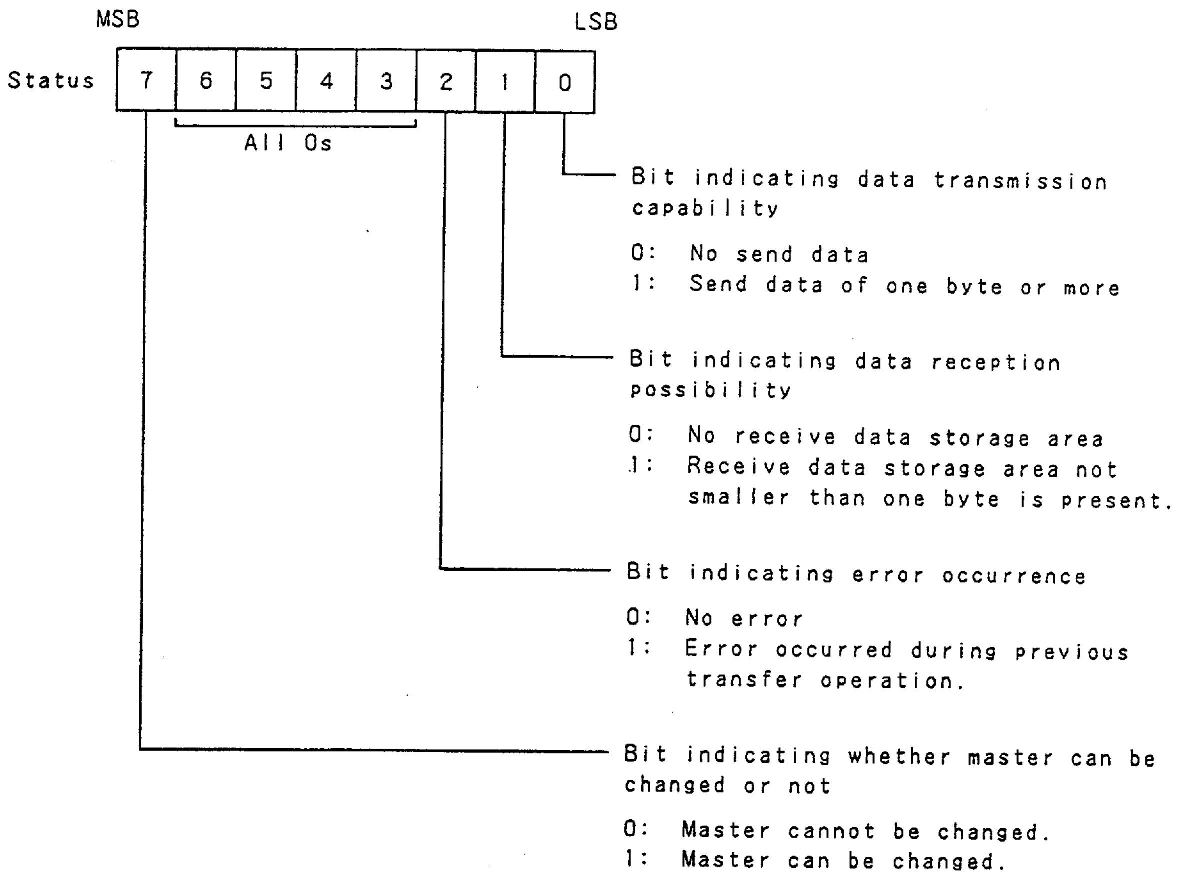
Fig. 5-87 Transfer Format of the STATUS Command



Remarks: M: Output by the master
S: Output by the slave

The slave returns the status in the format shown in Figure 5-88.

Fig. 5-88 Status Format of the STATUS Command

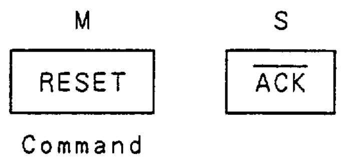

MSB
LSB
Status
7
6
5
4
3
2
1
0
All 0s
Bit indicating data transmission capability
0: No send data
1: Send data of one byte or more
Bit indicating data reception possibility
0: No receive data storage area
1: Receive data storage area not smaller than one byte is present.
Bit indicating error occurrence
0: No error
1: Error occurred during previous transfer operation.
Bit indicating whether master can be changed or not
0: Master cannot be changed.
1: Master can be changed.

When the master receives a status, it returns $\overline{\text{ACK}}$ to the slave currently selected.

④ RESET command

The RESET command changes the currently selected slave to a non-selected slave. When a RESET command is transmitted, all slaves can be placed in the non-selected state.

Fig. 5-89 Transfer Format of the RESET Command

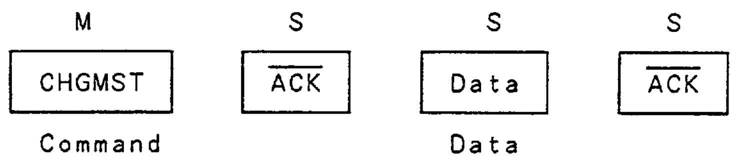


Remarks: M: Output by the master
S: Output by the slave

⑤ CHGMST command·

The CHGMST command passes the master authority to the currently selected slave.

Fig. 5-90 Transfer Format of the CHGMST Command



Remarks: M: Output by the master
S: Output by the slave

When the slave receives a CHGMST command, the slave returns one of the following data to the master after checking whether the slave can receive the master authority:

. OFFH: Master changeable
. OOH: Master not changeable

The slave compares the contents of SIOO before transfer with the contents of SIOO after transfer. If the contents of SIOO disagree with each other, an error occurs: $\overline{\text{ACK}}$ is not returned in this case.

If the master receives 0FFH, the master returns $\overline{\text{ACK}}$ to the slave, and starts to operate as a slave. The slave which sent 0FFH starts to operate as the master after it receives $\overline{\text{ACK}}$.

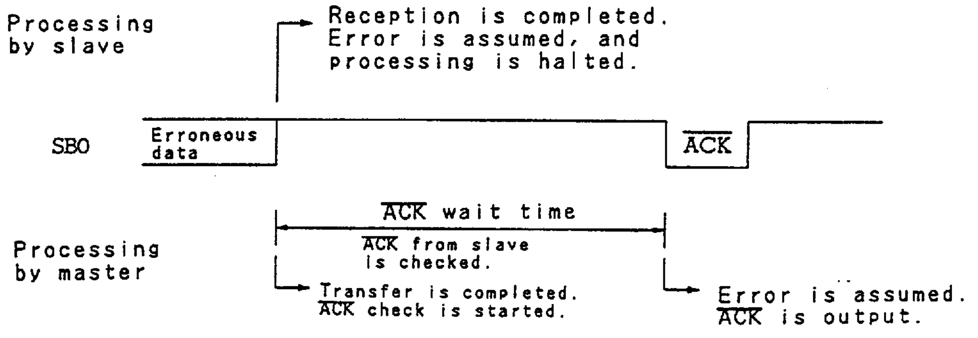
<Error occurrence>

If a communication error occurs, the operation described below is performed.

The slave reports the occurrence of an error by not returning $\overline{\text{ACK}}$ to the master. If an error occurs during receipt of data, the slave sets the status bit for indicating error occurrence, and cancels all command processing being performed.

When the transmission of one byte is completed, the master checks for $\overline{\text{ACK}}$ from the slave. If $\overline{\text{ACK}}$ is not returned from the slave within a predetermined period after transmission completion, the occurrence of an error is assumed; the master outputs the $\overline{\text{ACK}}$ signal as a dymmy.

Fig. 5-91 Master and Slave Operation in Case of Error


Processing by slave
Reception is completed. Error is assumed, and processing is halted.
SBO
Erroneous data
ACK
ACK wait time
ACK from slave is checked.
Processing by master
Transfer is completed. ACK check is started.
Error is assumed. ACK is output.

6427525 0094628 675

The following errors may occur:

. Error that may occur on the slave side

① Invalid command transfer format
② Reception of an undefined command
③ Insufficient number of transfer data bytes for a READ command
④ Insufficient area to contain data for a WRITE command

⑤ Changes in data being transmitted with a READ, STATUS, or CHGMST Command

If any of the above types of errors occur, $\overline{\text{ACK}}$ is not returned.

. Error that may occur on the master side

If data transmitted with a WRITE command changes during transmission, the master transmits a STOP command to the slave.

5.8.8 Manipulation of $\overline{\text{SCK0}}$ pin output

The $\overline{\text{SCK0}}$/P01 pin has a built-in output latch, so that this pin allows static output by software manipulation in addition to normal serial clock output.

The number of $\overline{\text{SCK0}}$s can be software-set arbitrarily by manipulating the P01 output latch. (The SO0/SB0/SB1 pin is controlled by manipulating the RELT and CMDT bits of SBIC.)

The procedure for manipulating $\overline{\text{SCK0}}$/P01 pin output is explained below.

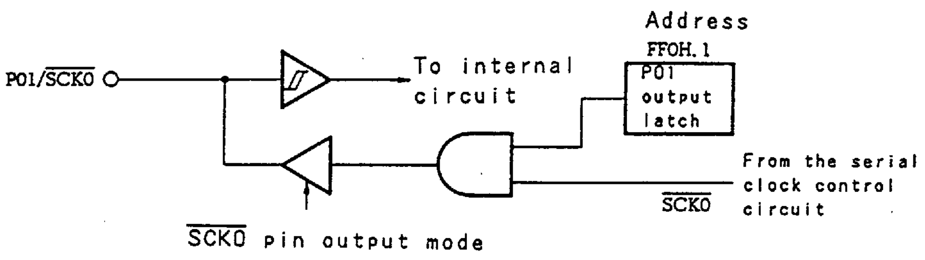
① Set serial operation mode register 0 (CSIM0) ($\overline{\text{SCK0}}$ pin: output mode, serial operation: enabled). When serial transfer operation is halted, $\overline{\text{SCK0}}$ = 1.

② Manipulate the P01 output latch by using a bit manipulation instruction.

Example: To output one $\overline{\text{SCK0}}$ clock cycle by software

```
SEL   MB15            ; Or CLR1  MBE
MOV   XA,#00000011B   ; SCK0 (fX/2^3), output mode
MOV   CSIM0,XA
CLR1  0FF0H.1         ; SCK0/P01 <- 0
SET1  0FF0H.1         ; SCK0/P01 <- 1
```

Fig. 5-92 $\overline{\text{SCK0}}$/P01 Pin Circuit Configuration



The P01 output latch is mapped to bit 1 of address FF0H. A $\overline{\text{RESET}}$ signal sets the P01 output latch to 1.

Cautions 1. During normal serial transfer operation, the P01 output latch must be set to 1.

2. The P01 output latch cannot be addressed by specifying PORT0.1 (as described below). The address of the latch (FF0H.1) must be coded in the operand of an instruction directly. However, MBE = 0 (or MBE = 1, MBS = 15) must be specified before the instruction is executed.

```
CLR1   PORT0.1 ┐
SET1   PORT0.1 ┘ Not allowed
CLR1   0FF0H.1 ┐
SET1   0FF0H.1 ┘ Allowed
```

5.9 Serial Interface (Channel 1)

5.9.1 Serial interface (channel 1) functions

The uPD75238 has two modes. The functions of the two modes are outlined below.

(1) Operation halt mode

This mode is used when serial transfer is not performed. This mode reduces power consumption.

(2) Three-wire serial I/O mode

8-bit data transfer is performed using three lines: Serial clock ($\overline{\text{SCK1}}$), serial output (SO1), and serial input (SI1).

The three-wire serial I/O mode allows full-duplex transmission, so data transfer can be performed at higher speed.

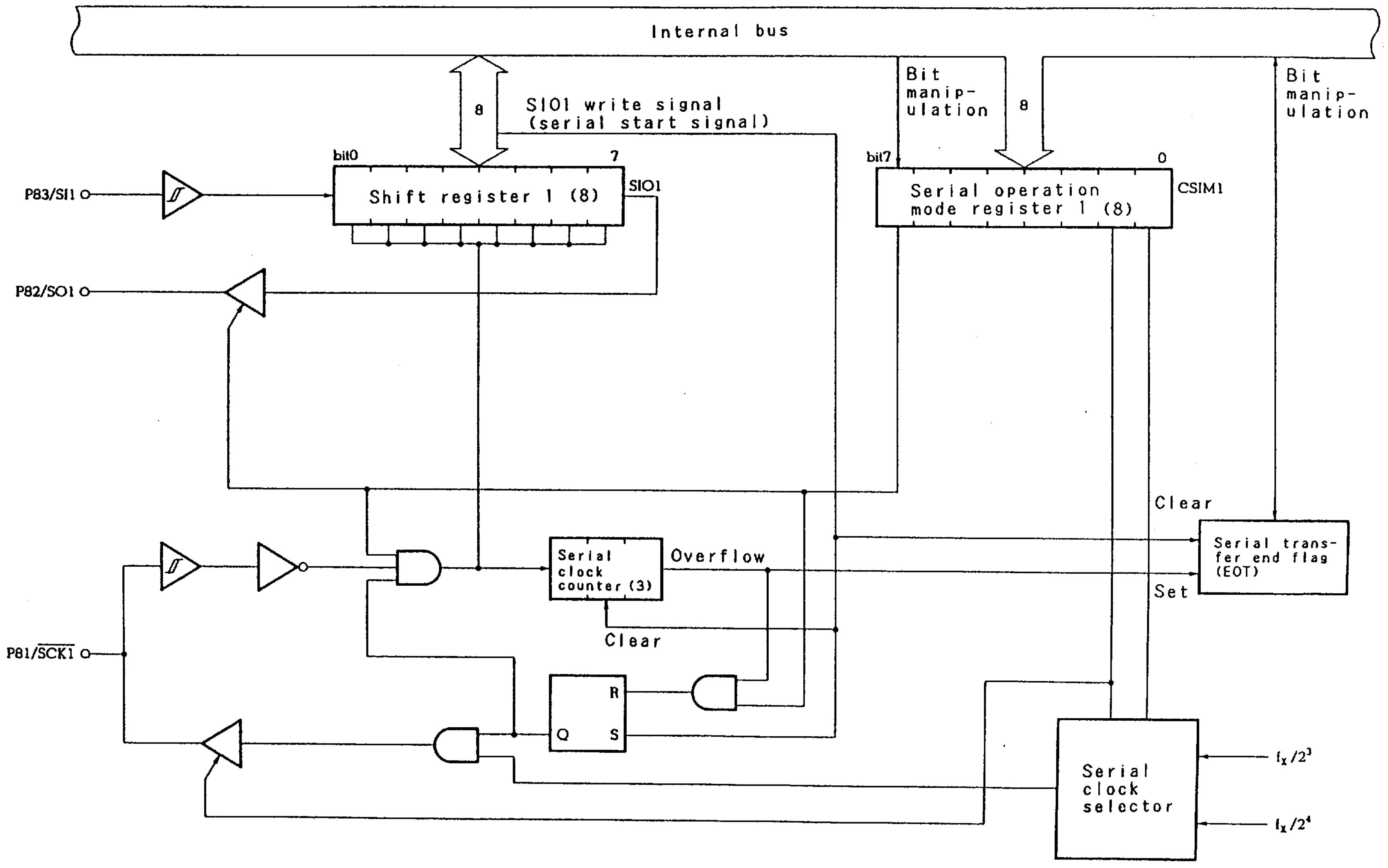
The user can choose 8-bit data transfer starting with the MSB or LSB, so devices starting with either the MSB or LSB can be connected.

The three-wire serial I/O mode enables connections to be made with the 75X series, 78K series, and many other types of peripheral I/O devices.

5.9.2 Serial interface (channel 1) configuration

Figure 5-93 shows the block diagram of the serial interface (channel 1).

Fig. 5-93 Block Diagram of the Serial Interface (Channel 1)



5.9.3 Register functions

(1) Serial operation mode register 1 (CSIM1)

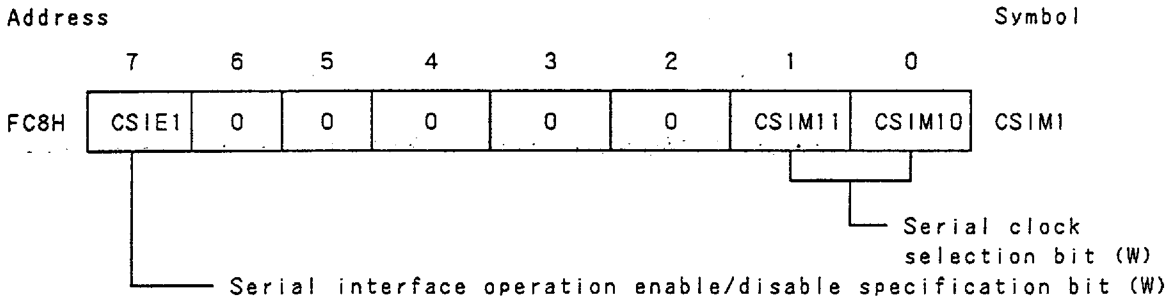
Figure 5-94 shows the format of serial operation mode register 1 (CSIM1).

CSIM1 is an 8-bit register which specifies a serial interface (channel 1) operation mode and serial clock.

CSIM1 is manipulated using an 8-bit memory manipulation instruction. Only the high-order one bit can be manipulated independently. Each bit can be manipulated using its name.

When the $\overline{\text{RESET}}$ signal is input, this register is set to 00H.

Fig. 5-94 Format of Serial Operation Mode Register 1 (CSIM1)



Address
Symbol
7
6
5
4
3
2
1
0
FC8H
CSIE1
0
0
0
0
0
CSIM11
CSIM10
CSIM1
Serial clock selection bit (W)
Serial interface operation enable/disable specification bit (W)

Remark: (W): Write only

Serial clock selection bit (W)

CSIM11	CSIM10	Serial clock (three-wire serial I/O mode)	$\overline{SCK1}$ pin mode
0	0	External clock applied to $\overline{SCK1}$ pin	Input
0	1	Not to be set	–
1	0	$f_X/2^4$ (262 kHz or 375 kHz)$^{(*)}$	Output
1	1	$f_X/2^3$ (524 kHz or 750 kHz)$^{(*)}$	

* The values at 4.19 MHz and 6.0 MHz are indicated in parentheses.

Serial interface operation enable/disable specification bit (W)

		Shift register 1 operation	Serial clock counter	IRQCSI flag	SO1, SI1 pin
CSIE1	0	Shift operation disabled	Cleared	Held	Used only for port 0
	1	Shift operation enabled	Count operation	Can be set.	Used in each mode as well as for port 0

Caution: Be sure to write 0 in bits 2 to 6 of the serial operation mode register.

Example: To select $f_X/2^4$ as the serial clock, and set the serial transfer end flag EOT to 1. each time serial transfer terminates

```
SEL   MB15             ; Or CLR1   MBE
MOV   XA,#10000010B
MOV   CSIM1,XA         ; CSIM1 ← 10000010B
```

(2) Shift register 1 (SIO1)

SIO1 is an 8-bit register which performs parallel-serial conversion and serial transfer (shift) operation in phase with the serial clock.

Serial transfer is started by writing data to SIO1.

In send operation, data written to SIO1 is output on the serial output (SO1). In receive operation, data is read from the serial input (SI1) into SIO1.

Data can be read from or written to SIO1 using an 8-bit manipulation instruction.

When the $\overline{\text{RESET}}$ signal is entered during operation, the value of SIO1 is undefined. When the $\overline{\text{RESET}}$ signal is entered in the standby mode, the value of SIO1 is preserved.

Shift operation is stopped after 8-bit send or receive operation is completed.

The timing for reading SIO1 and start of serial transfer (writing to SIO1) is as follows:

. When the serial interface operation enable/disable bit (CSIE1) is set to 1. However, the case where CSIE1 is set to 1 after data is written to the shift register is excluded.

. When the serial clock is masked after 8-bit serial transfer

. When $\overline{\text{SCK1}}$ is high

5.9.4 Operation halt mode

The operation halt mode is used when serial transfer is not performed, which is set by setting 0 in CSIE1. This mode reduces power consumption.

Shift register 1 does not perform shift operation in this mode, so the shift register can be used as a normal 8-bit register.

When the $\overline{\text{RESET}}$ signal is entered, the operation halt mode is set. The P82/SO1 pin and P83/SI1 pin function as input-only port pins. The P81/$\overline{\text{SCK1}}$ pin can be used as an input port pin by setting serial operation mode register 1.

5.9.5 Three-wire serial I/O mode operations

The three-wire serial I/O mode is compatible with other modes used in the 75X series, uPD7500 series, and 78K series. This mode is set by setting CSIE1 to 1.

Communication is performed using three lines: Serial clock ($\overline{\text{SCK1}}$), serial output (SO1), and serial input (SI1).

The three-wire serial I/O mode transfers data with eight bits as one block. Data is transferred bit by bit in phase with the serial clock.

Shift register 1 performs shift operation on the falling edge of the serial clock ($\overline{\text{SCK1}}$). Send data is latched on the SO1 latch, and is output on the SO1 pin. Receive data applied to the SI1 pin is latched in the shift register 1 on the rising edge of $\overline{\text{SCK1}}$.

When eight bits have been transferred, operation of shift register 1 automatically terminates setting the serial transfer end flag (EOT).

Fig. 5-95 Timing of the Three-wire Serial I/O Mode

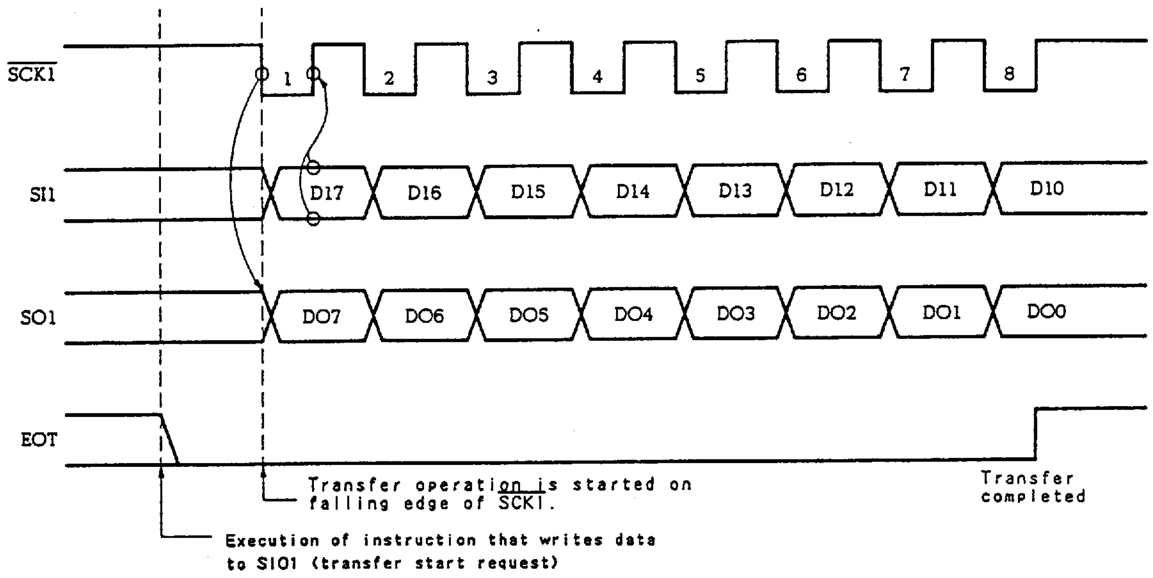

SCK1
1
2
3
4
5
6
7
8
SI1
D17
D16
D15
D14
D13
D12
D11
D10
SO1
DO7
DO6
DO5
DO4
DO3
DO2
DO1
DO0
EOT
Transfer operation is started on falling edge of SCK1.
Transfer completed
Execution of instruction that writes data to SIO1 (transfer start request)

Table 5-14 Serial Clock Selection and Application

Mode register		Serial clock		Timing for shift register R/W and start of serial transfer	Application
CSIM1 1	CSIM1 0	Source	Masking of serial clock		
0	0	External $\overline{\text{SCK1}}$	Automatically masked when 8-bit data transfer is completed	① In the operation halt mode (CSIE1 = 0) ② When the serial clock is masked after 8-bit transfer ③ When $\overline{\text{SCK1}}$ is high	Slave CPU
0	1	-			–
1	0	$f_X/2^4$			Middle-speed serial transfer
1	1	$f_X/2^3$			High-speed serial transfer

An 8-bit transfer instruction is used to write to or read from the shift register. The transfer format starts with the MSB.

Example: To transfer the RAM data specified by the HL register pair to SIO1, load the SIO1 data to the accumulator, and start serial transfer operation:

```
MOV  XA,@HL   ; Fetch send data from RAM
SEL  MB15     ; Or CLR1  MBE
XCH  XA,SIO1  ; Exchange send data and receive data, and
                start transfer
```

5.9.6 Application of the serial interface (channel 1)

This section describes how to use the serial interface (channel 1) in the three-wire serial I/O mode with an example.

Usually, communication using the serial interface is performed in the following steps.

① A transfer mode is set. (Data is set in CSIM1.)

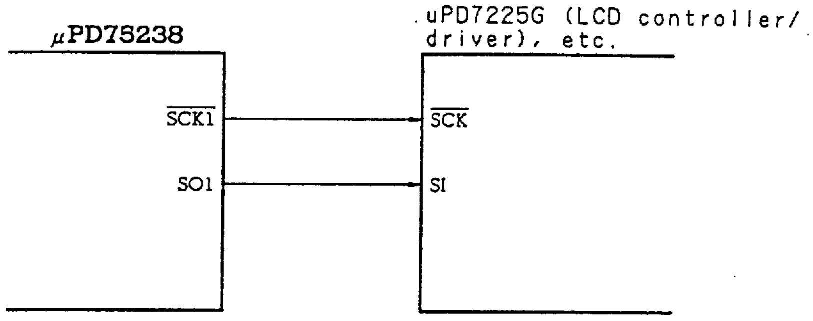
② Data is written to SIO1, then start of transfer is directed. (MOV SIO1,XA or XCH XA,SIO1; in this case, transfer operation is automatically directed.)

Example: Data is transferred starting with the MSB on a transfer clock of 262 kHz (in 4.19-MHz operation) (master operation).

<Sample program>

```
CLR1  MBE
MOV   XA,#10000010B
MOV   CSIM1,XA        ; Set transfer mode
MOV   XA,TDATA        ; TDATA is transfer data storage
                        address
MOV   SIO1,XA         ; Set transfer data, and start
                        transfer
```

Caution: A second or subsequent transfer operation can be started by setting data in SIO1 (MOV SIO1,XA or XCH XA,SIO1).



In this case, the SI1 pin on the uPD75238 can be used as an input.

5.10 A/D Converter

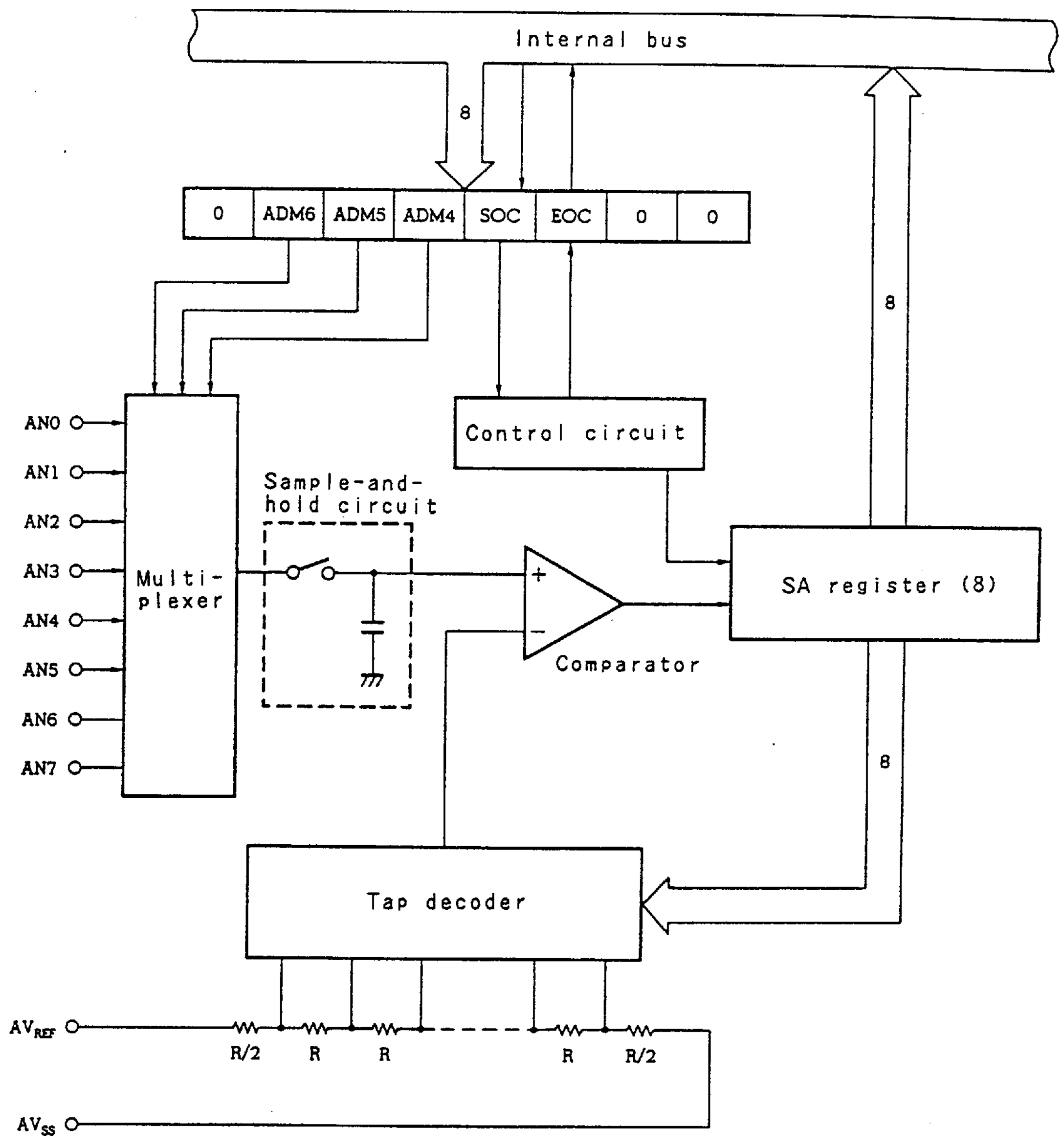
The uPD75238 contains an 8-bit analog/digital (A/D) converter that has eight analog input channels (AN0 to AN7).

The A/D converter employs the successive-approximation method.

5.10.1 Configuration of the A/D converter

Figure 5-96 shows the configuration of the A/D converter.

Fig. 5-96 Block Diagram of the A/D Converter



5.10.2 Pins of the A/D converter

(1) AN0 to AN7

AN0 to AN7 are the input pins for eight analog signal channels. Analog signals subject to A/D conversion are applied to these pins.

The A/D converter contains a sample-and-hold circuit, and analog input voltages are internally maintained during A/D conversion.

(2) AV_{REF}, AV_{SS}

A reference voltage for the A/D converter is applied to these pins.

By using an applied voltage across AV_{REF} and AV_{SS}, signals applied to AN0 to AN7 are converted to digital signals.

(3) AV_{DD}

This is the supply voltage pin for the A/D converter. When the A/D converter is not used or is in the standby mode, the potential of the AV_{DD} pin must be equal to that of the V_{DD} pin.

5.10.3 A/D conversion mode register

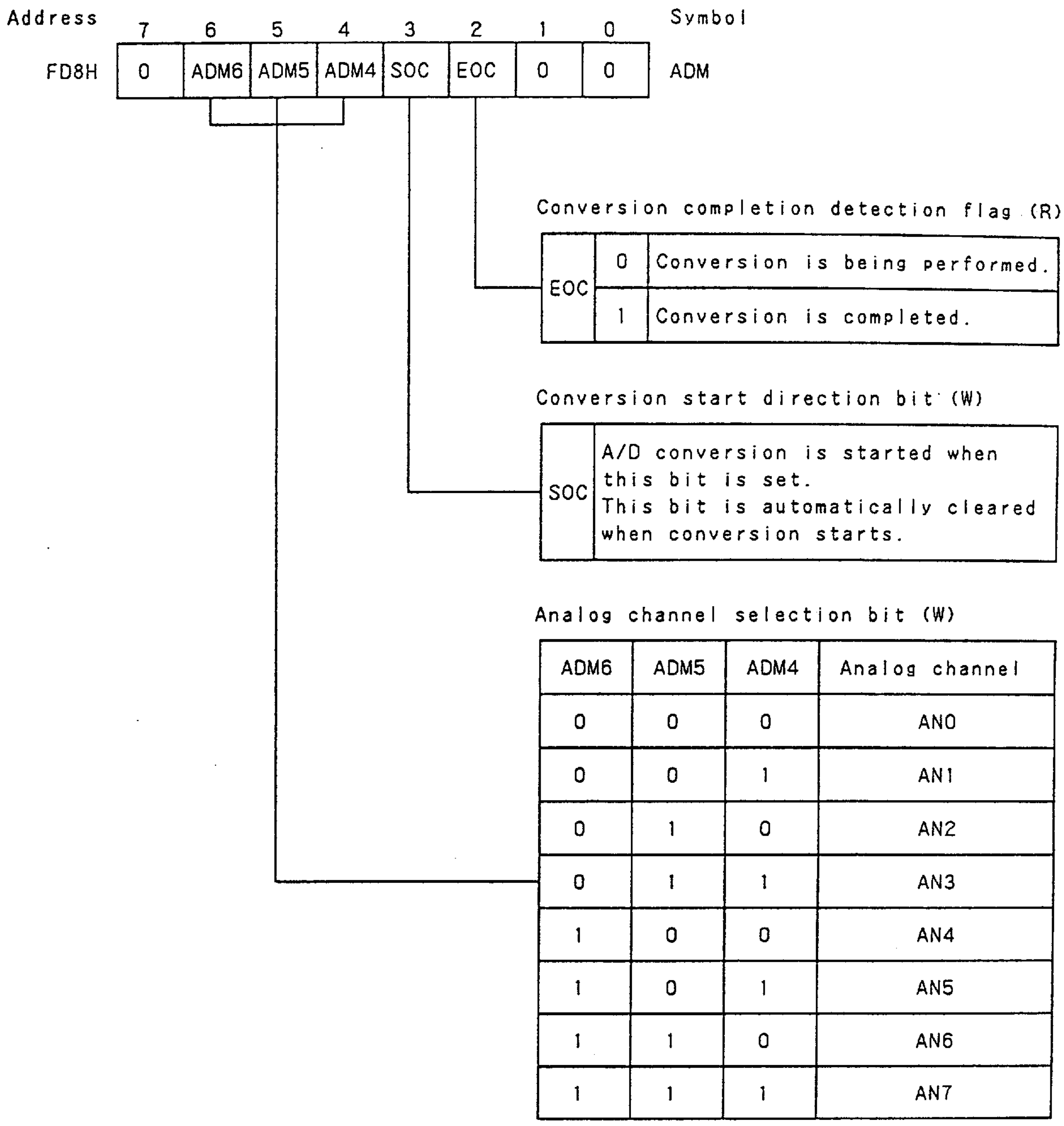
The A/D conversion mode register (ADM) is an 8-bit register used to select analog input channels, direct the start of conversion, and detect the completion of conversion. (See Figure 5-97.)

ADM is set with an 8-bit manipulation instruction.

The conversion completion detection flag of bit 2 (EOC) and the conversion start direction bit of bit 3 (SOC) can be manipulated on a bit-by-bit basis.

A $\overline{\text{RESET}}$ input signal initializes ADM to 04H. That is, only EOC is set to 1, with all bits cleared to 0.

Fig. 5-97 Format of the A/D Conversion Mode Register



Address	7	6	5	4	3	2	1	0	Symbol
FD8H	0	ADM6	ADM5	ADM4	SOC	EOC	0	0	ADM

Conversion completion detection flag (R)

EOC	0	Conversion is being performed.
	1	Conversion is completed.

Conversion start direction bit (W)

SOC	A/D conversion is started when this bit is set. This bit is automatically cleared when conversion starts.

Analog channel selection bit (W)

ADM6	ADM5	ADM4	Analog channel
0	0	0	AN0
0	0	1	AN1
0	1	0	AN2
0	1	1	AN3
1	0	0	AN4
1	0	1	AN5
1	1	0	AN6
1	1	1	AN7

Caution: A/D conversion is started a maximum of $2^4/f_X$ seconds (2.67 us at f_X = 6.0 MHz)$^{(Note)}$ after SOC is set. (For details, see Section 5.10.5.)

Note: 3.81 us at 4.19 MHz

5.10.4 SA register (SA)

The SA register (Successive Approximation Register) is an 8-bit register to hold the result of A/D conversion in successive approximation.

SA is read with an 8-bit manipulation instruction. No data can be written to SA by software.

Receiving the $\overline{\text{RESET}}$ signal makes the SA undefined.

SA is mapped to address FDAH.

5.10.5 A/D converter operation

Analog input signals subject to A/D conversion are specified with bits 6, 5, and 4 of the A/D conversion mode register (ADM6, ADM5, and ADM4).

A/D conversion is started by setting bit 3 (SOC) of ADM to 1. After that, SOC is automatically cleared to 0. A/D conversion is performed by hardware using the successive-approximation method. The resultant 8-bit data is loaded into the SA register. Upon completion of A/D conversion, ADM bit 2 (EOC) is set to 1.

Figure 5-98 shows the timing chart of A/D conversion.

The A/D converter is used as follows:

① Select analog input channels and comparator bias voltage (by setting ADM6, ADM5, ADM4, and ADM1).

② Direct the start of A/D conversion (by setting SOC).

③ Wait for the completion of A/D conversion (wait for EOC to be set or wait using a software timer).

④ Read the result of A/D conversion (read the SA register).

Cautions 1. ① and ② above can be performed at the same time.

2. There is a delay of up to $2^4/f_X$ seconds (2.67 us at 6.0 MHz)$^{(Note)}$ from the setting of SOC to the clearing of EOC after A/D conversion is started. EOC must be tested when a time indicated in Table 5-15 has elapsed after the setting of SOC. Table 5-15 also indicates A/D conversion times.

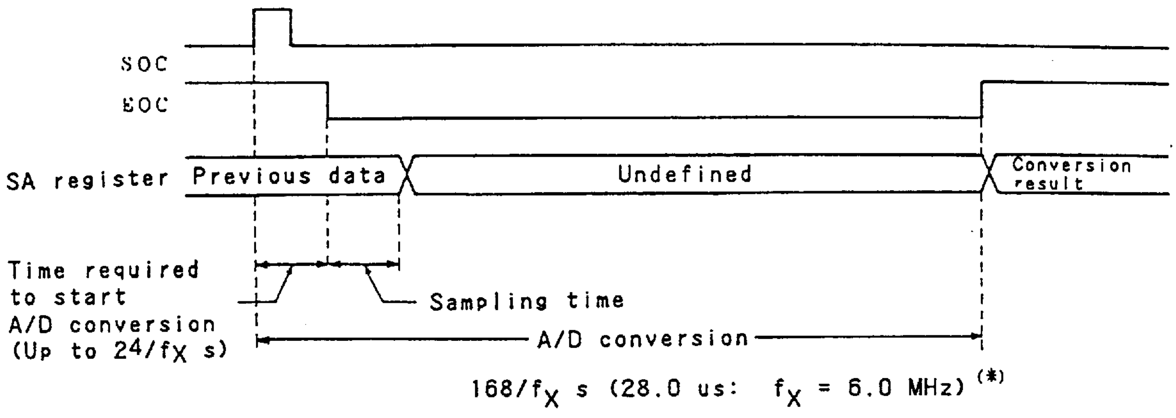
Note: 3.81 us at 4.19 MHz

Table 5-15 Setting of SCC and PCC

Setting values of SCC, PCC				A/D conversion time	Wait time from SOC setting to EOC test	Wait time from SOC setting to A/D conversion completion
SCC3	SCC0	PCC1	PCC0			
0	0	0	0	$168/f_X$ (28.0 us/ f_X = 6.0 MHz)	Waiting not required	3 machine cycles
0	0	0	1	$168/f_X$ (28.0 us/ f_X = 6.0 MHz)	1 machine cycle	11 machine cycles
0	0	1	0	$168/f_X$ (28.0 us/ f_X = 6.0 MHz)	2 machine cycles	21 machine cycles
0	0	1	1	$168/f_X$ (28.0 us/ f_X = 6.0 MHz)	4 machine cycles	42 machine cycles
0	1	x	x	$168/f_X$ (28.0 us/ f_X = 6.0 MHz)	Waiting not required	Waiting not required
1	x	x	x	Conversion stopped	−	−

Remark: x: Don't care

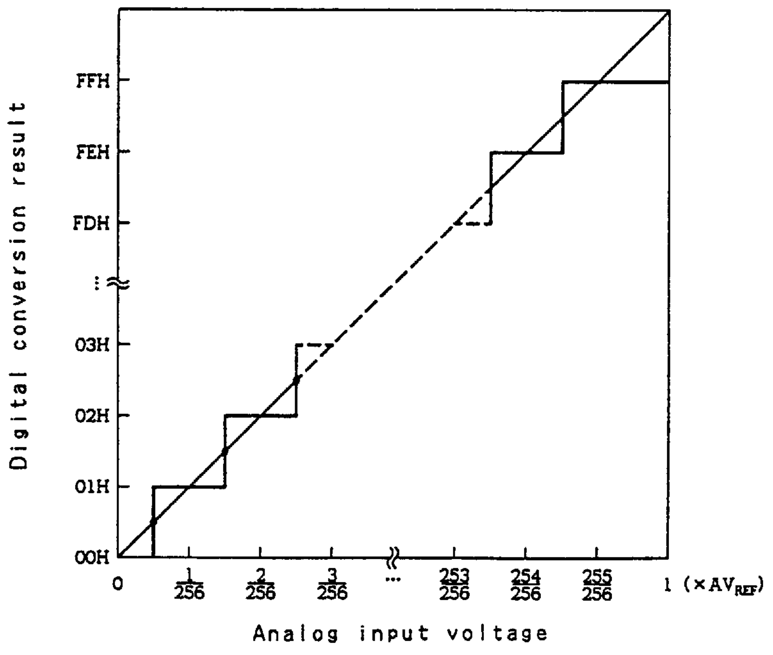
Fig. 5-98 Timing Chart of A/D Conversion


SOC
EOC
SA register
Previous data
Undefined
Conversion result
Time required to start A/D conversion (Up to $24/f_X$ s)
Sampling time
A/D conversion
$168/f_X$ s (28.0 us: f_X = 6.0 MHz) (*)

* 40.1 us at 4.19 MHz

Figure 5-99 shows the relationship between analog input voltages and 8-bit digital data obtained by A/D conversion.

Fig. 5-99 Relationship (Ideal) between Analog Input Voltages and Results of A/D Conversion

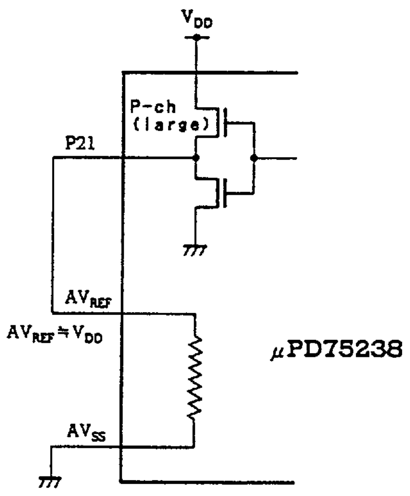


5.10.6 Notes on the standby mode

The A/D converter operates on the main system clock. So its operation stops in the STOP mode, or when the subsystem clock is used, in the HALT mode. A current flows through the AV_{REF} pin even when the A/D converter is not operating. This current should be stopped to reduce the overall system power consumption. Since pin P21 has a higher drive capability than the other ports, it can supply voltage to the AV_{REF} pin directly. In this case, however, the actual AV_{REF} voltage does not have a high accuracy. This means that the value resulting from conversion does not have a high accuracy either and can be used only for comparison. In the standby mode, outputting a low on pin P21 can reduce power consumption.

Pin P113 of the peripheral hardware emulator uPD75390 used for emulation using an in-circuit emulator has the same drive capability as the other ports. Accordingly, V_{ref} observed during emulation is lower than V_{ref} observed on the uPD75238, and the conversion result obtained during emulation is not equal to that on the uPD75238.

Fig. 5-100 Reducing Power Consumption in the Standby Mode



■ 6427525 0094648 463 ■

5.10.7 Other notes on use

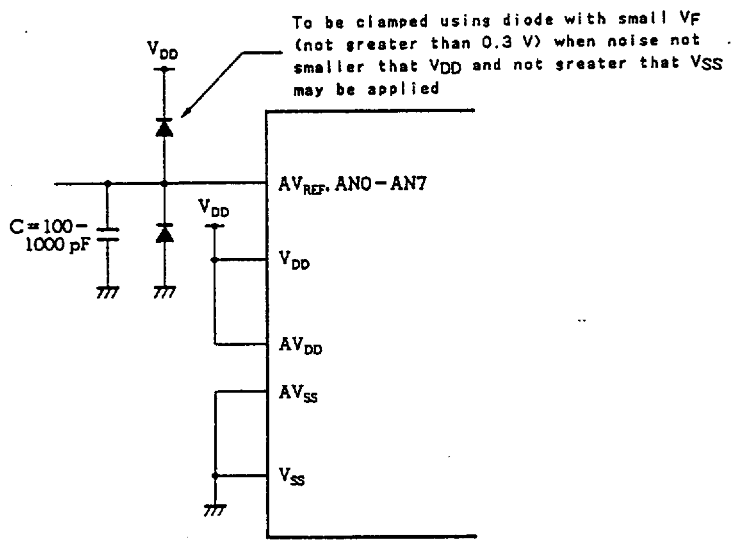
(a) AN0 to AN7 input range

Specified voltages must be applied to AN0 to AN7 inputs. If a voltage higher than V_{DD} or lower than V_{SS} is applied even when the maximum absolute rating is not exceeded, the conversion result for an associated channel becomes unpredictable. In addition, the conversion results for other channels may be affected.

(b) Noise protection

To maintain 8-bit resolution, the user should pay attention to noise that may be applied to the AV_{REF}, and AN0 to AN7 pins. Noise adversely affects operation to a greater extent when the analog input source has a higher output impedance. As shown in Figure 5-101, a resistor and capacitor should be externally connected.

Fig. 5-101 Analog Input Pin Connection


To be clamped using diode with small V_F (not greater than 0.3 V) when noise not smaller that V_{DD} and not greater that V_{SS} may be applied
V_{DD}
AV_{REF}, AN0 - AN7
V_{DD}
C = 100 - 1000 pF
V_{DD}
AV_{DD}
AV_{SS}
V_{SS}

(c) AN4/P90 to AN7/P93 pins

The analog input pins (AN4 to AN7) are also used for an input port (PORT9).

When any of AN4 to AN7 is selected for A/D conversion, no input instruction must be executed for PORT9 during A/D conversion. Otherwise, the resolution of conversion may deteriorate.

If a digital pulse signal is applied to a pin adjacent to a pin being used for A/D conversion, an expected A/D conversion value may not be obtained because of coupling noise. So no digital pulse signal should be applied to a pin adjacent to a pin being used for A/D conversion.

5.10.8 Application of A/D converter

Example: Select channel 0 (AN0) for analog input, and set the following:

. SCC3, SCC0 = 0, 0
. PCC1, PCC0 = 1, 0

Use middle-speed mode.

<Sample program>

```
       CLR1  MBE      ; Or SEL MB15
       MOV   XA,#08H  ; XA ← 00001000B
       MOV   ADM,XA   ; Select AN0, and start A/D conversion
       NOP            ; Wait until EOC flag test
       NOP            ; Wait until EOC flag test
TEST:  SKT   EOC      ; EOC flag test
       BR    TEST
       MOV   XA,SA    ; XA ← A/D conversion result
```

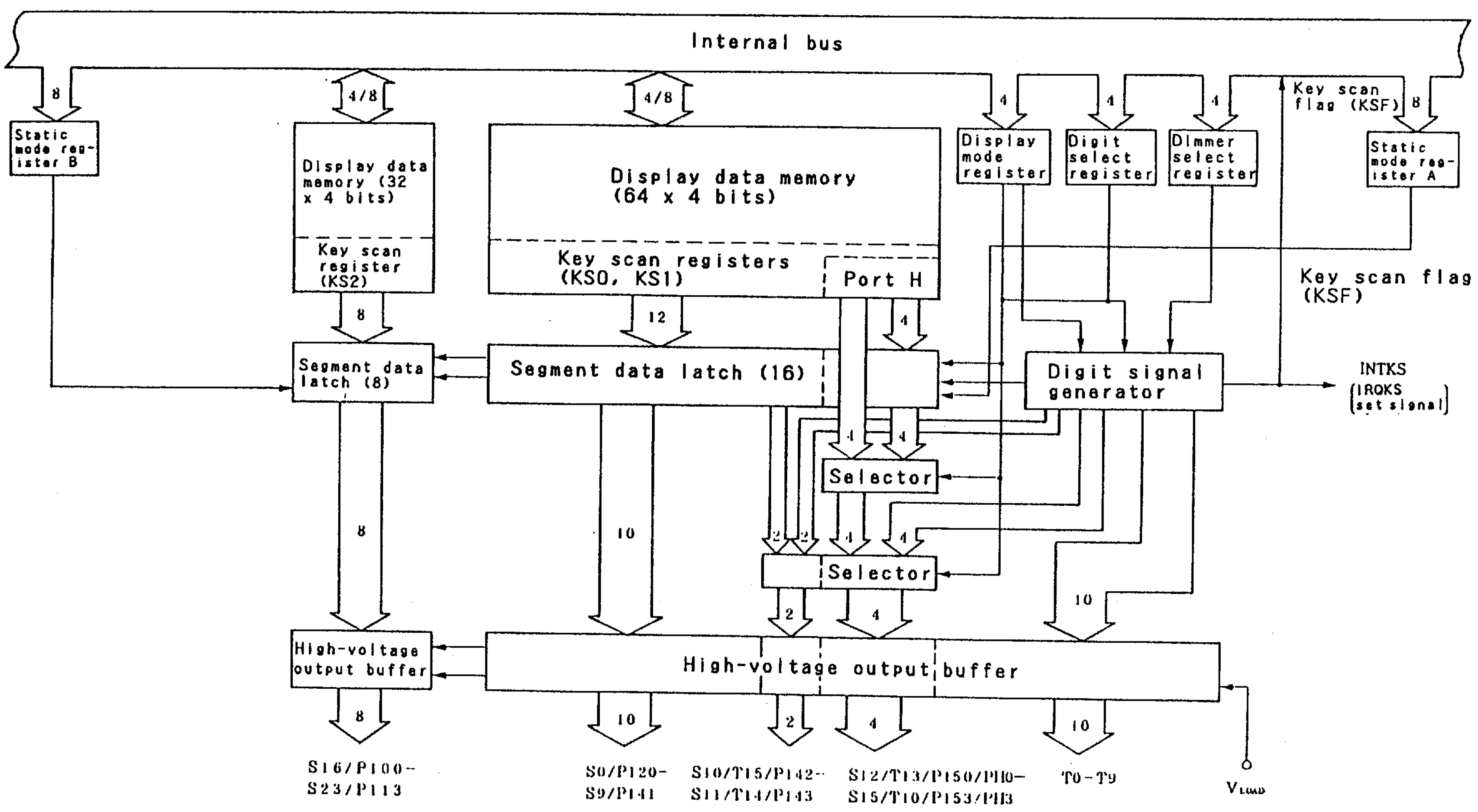
5.11 FIP Controller/Driver

5.11.1 Function of the FIP controller/driver

The uPD75238 contains a display controller that reads the contents of display data memory and generates digit and segment signals automatically. It also contains a high-voltage output buffer that can directly drive a fluorescent indicator lamp (FIP).

Figure 5-102 shows the configuration of the FIP controller/driver.

Fig. 5-102 Block Diagram of the FIP Controller/Driver



The FIP controller/driver contained in the uPD75238 has the following functions:

(a) The FIP controller/driver automatically reads display data and generates a segment signal (DMA operation) and a digit signal.

(b) An FIP having nine to 24 segments and nine to 16 digits can be controlled with the display mode register (DSPM), digit select register (DIGS), and static mode registers (STATA and STATB). (Up to 34 display outputs are allowed.)

(c) Any outputs not used for dynamic display can be used as static outputs or output ports.

(d) The dimmer function provides eight levels of intensity.

(e) Hardware that makes key scan application possible

- An interrupt (IRQKS) is caused for key scanning (detection of key scan timing.)
- Key scan data can be output on a segment output with key scan buffers (KS0, KS1, and KS2).

(f) A high-voltage output pin (40 V) is provided which can directly drive the FIP.

- Segment output pins (S0 to S9, S16 to S23): V_{OD} = 40 V, I_{OD} = 3 mA
- Digit output pins (T0 to T15): V_{OD} = 40 v, I_{OD} = 15 mA

(g) Mask options for display output pins(Note)

- A pull-down resistor to V_{LOAD} can be incorporated bit by bit for T0 to T9 and S0 to S15.
- A pull-down resistor to V_{LOAD} or V_{SS} can be incorporated bit by bit for S16 to S23. V_{LOAD} or V_{SS} must be selected in 8-bit units.

Note: Pull-down resistors are internally connected to the T0 to T9 and S0 to S8 signals in the uPD75P238.

Figure 5-103 shows the timing of FIP controller operation.

A display cycle (T_{CYT}) consists of a display timing in which the FIP is driven and a key scanning timing.

In the display timing, the FIP controller sequentially reads data stored in display data memory for output on segment pins (S_N).

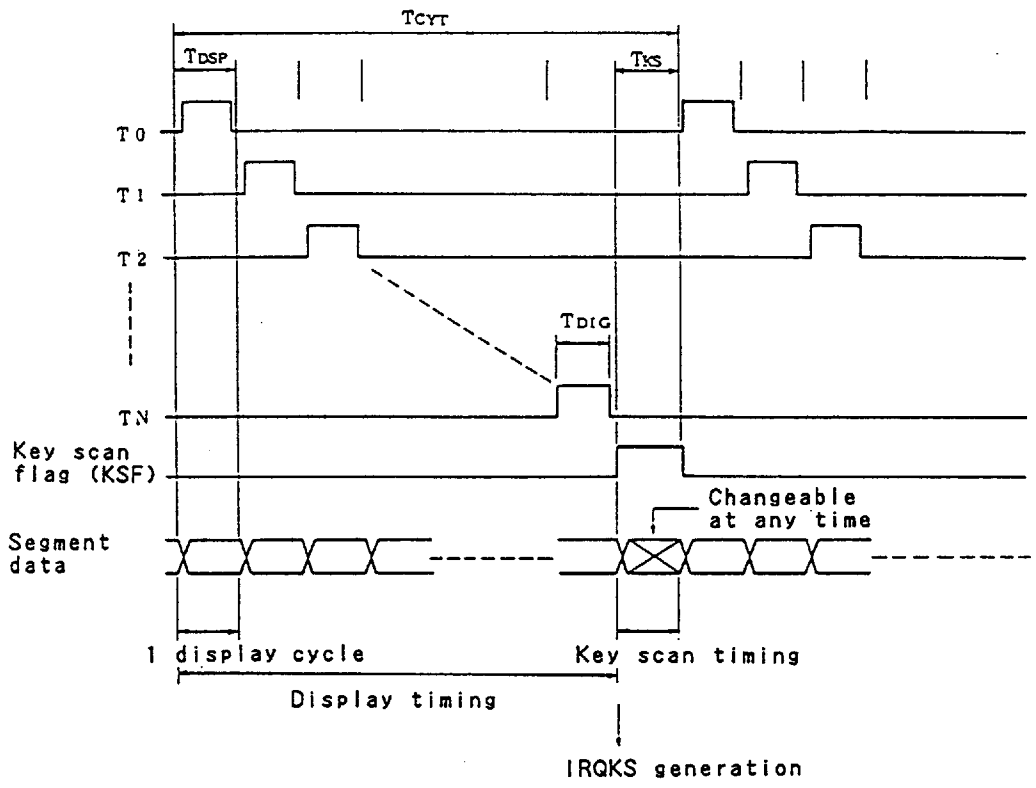
A display mode is determined by a combination of a digit signal carrying the number of digits set with the digit select register (DIGS) and a segment signal carrying the number of segments set with the display mode register (DSPM) and static mode registers (STATA and STATB).

A digit signal is output which has a cut width specified with the dimmer select register (DIMS) to prevent display light leakage and to control intensity (dimmer).

In the key scan timing, the contents of the key scan register are output as the segment signal, and no digit signal is output (all digit outputs are in the low level).

Remark: the key scan registers (KS0, KS1, and KS2) are mapped to an area of display memory. A write of key scan data to KS0, KS1, and KS2 is triggered by a key scan interrupt (IRQKS).

Fig. 5-103 FIP Controller Operation Timing



N:	Value set in the digit select register
T_{DSP}:	One display cycle (1024/f_X: 171 us/ 6.0 MHz$^{\text{(Note 1)}}$ or 2048/f_X: 341 us/ 6.0 MHz$^{\text{(Note 2)}}$)
T_{CYT}:	Display cycle (T_{CYT} = T_{DSP} x (N + 2))
T_{DIG}:	Digit signal pulse width, which can be selected from eight levels with the dimmer select register
T_{KS}:	Key scan timing

Notes 1. 244 us at 4.19 MHz

2. 489 us at 4.19 MHz

Cautions 1. The FIP controller/driver can operate only in the high-speed or medium speed mode (PCC = 0011B or 0010B) of the main system clock (SCC.0 = 0). With the other clocks or in the standby mode, the FIP controller/driver may malfunction, so that FIP controller operation must be disabled (by setting DSPM.3 = 0).

2. In the display off mode, ports 10 to H used also as display outputs go into a high-impedance state (without pull-down resistors) or are low (with pull-down resistors).

Accordingly, use ports 10 to H in such a way that these ports are allowed to be set off when the display is turned off in the standby mode.

5.11.2 Registers of the FIP controller/driver

The FIP controller/driver is controlled with the following five registers.

Table 5-16 Registers for Controlling the FIP Controller/Driver

Register name	Symbol	Description of control
Display mode register	DSPM	Selects display segments (9 to 16 or 17 to 24 segments). Enables or disables display.
Digit select register	DIGS	Selects display digits (9 to 16 digits).
Dimmer select register	DIMS	Specifies display cycle. Selects cut width of digit signal (8 levels).
Static mode register A	STATA	Selects either display output or static output for S0/P120 to S15/T10/P153/PH3.
Static mode register B	STATB	Selects either display output or static output for S16/P100 to S23/P113.

The following shows the format of each register.

(1) Display mode register (DSPM)

The display mode register (DSPM) is a 4-bit register for enable/disable setting for the display operation and for specifying the number of display segments. Figure 5-104 shows the format of the register.

The display mode register is set with a 4-bit memory manipulation instruction.

Caution: Before a standby mode (STOP mode or HALT mode) can be set or the subsystem clock (f_{XT}) can be used for operation, display must be disabled by setting DSPM.3 to 0.

A $\overline{\text{RESET}}$ input clears all bits to 0.

Example: Enables display with 20 segments.

```
SEL   MB15
MOV   A,#00H
MOV   STATA,XA
MOV   STATB,XA
MOV   A,#1011B
MOV   DSPM,A
```

Fig. 5-104 Display Mode Register Format

Address	3	2	1	0	Symbol
F88H	DSPM3	DSPM2	DSPM1	DSPM0	DSPM

Bit for specifying the number of display segments

DSPM2	DSPM1	DSPM0	Number of display segments
0	0	0	9 segments (+ 8 segments)
0	0	1	10 segments (+ 8 segments)
0	1	0	11 segments (+ 8 segments)
0	1	1	12 segments (+ 8 segments)
1	0	0	13 segments (+ 8 segments)
1	0	1	14 segments (+ 8 segments)
1	1	0	15 segments (+ 8 segments)
1	1	1	16 segments (+ 8 segments)

Remark: Segments in parentheses are added when pins S16 to S23 are specified as dynamic mode in STATB.

Display operation enable/disable bit

DSPM3	0	Display disabled
	1	Display enabled

(2) Digit select register (DIGS)

The digit select register (DIGS) is a 4-bit register that specifies the number of display digits.
Figure 5-105 shows the format of the register.

DIGS is set with a 4-bit memory manipulation instruction. This register enables the number of display digits to be selected from nine to 16 digits. No other values can be selected.

A $\overline{\text{RESET}}$ input clears the register to 1000B so that 9-digit display is selected.

Example: Select 10-digit display.

```
SEL  MB15
MOV  A,#9
MOV  DIGS, A
```

Fig. 5-105 Digit Select Register Format

Address	3	2	1	0	Symbol
F8AH	DIGS3	DIGS2	DIGS1	DIGS0	DIGS

DIGS 0-3 set value	Number of display digits
N (= 8 to 15)	N + 1

Cautions 1. Do not set a value from 0 to 7 for N.

2. Be sure to set the digit select resister to 1000B when using the S0/P120 to S15/P153/T10/PH3 pins as static outputs.

(3) Dimmer select register (DIMS)

The dimmer select register (DIMS) is a 4-bit register for specifying a digit signal cut width to prevent display light leakage and to use the dimmer function (for intensity control). In addition, the register is used to select a display cycle (T_{DSP}).

Figure 5-106 shows the format of DIMS. Figure 5-107 shows the waveforms of the digit signals associated with set values.

DIMS is set with a 4-bit memory manipulation instruction.

One of two types of display cycle is selected. If DIMS.0 is set to 1, a display cycle of 341 us (when operating at 6.0 MHz)(Note 1) is selected. If DIMS.0 is set to 0, a display cycle of 171 us (when operating at 6.0 MHz)(Note 2) is selected.

Note that, however, as the number of display digits increases, the display cycle becomes closer to the frequency of commercial power, resulting in flickering display.

A $\overline{\text{RESET}}$ input clears all bits to 0.

Notes 1. 489 us when the microcomputer operates at 4.19 MHz

2. 244 us when the microcomputer operates at 4.19 MHz

Fig. 5-106 Dimmer Select Register Format

Address	3	2	1	0	Symbol
F89H	DIMS3	DIMS2	DIMS1	DIMS0	DIMS

Display cycle specification bit

DIMS0	0	A display cycle is $\frac{1024}{f_X}$. (One cycle = 171 us/6.0 MHz)(*1)
	1	A display cycle is $\frac{2048}{f_X}$. (One cycle = 341 us/6.0 MHz)(*2)

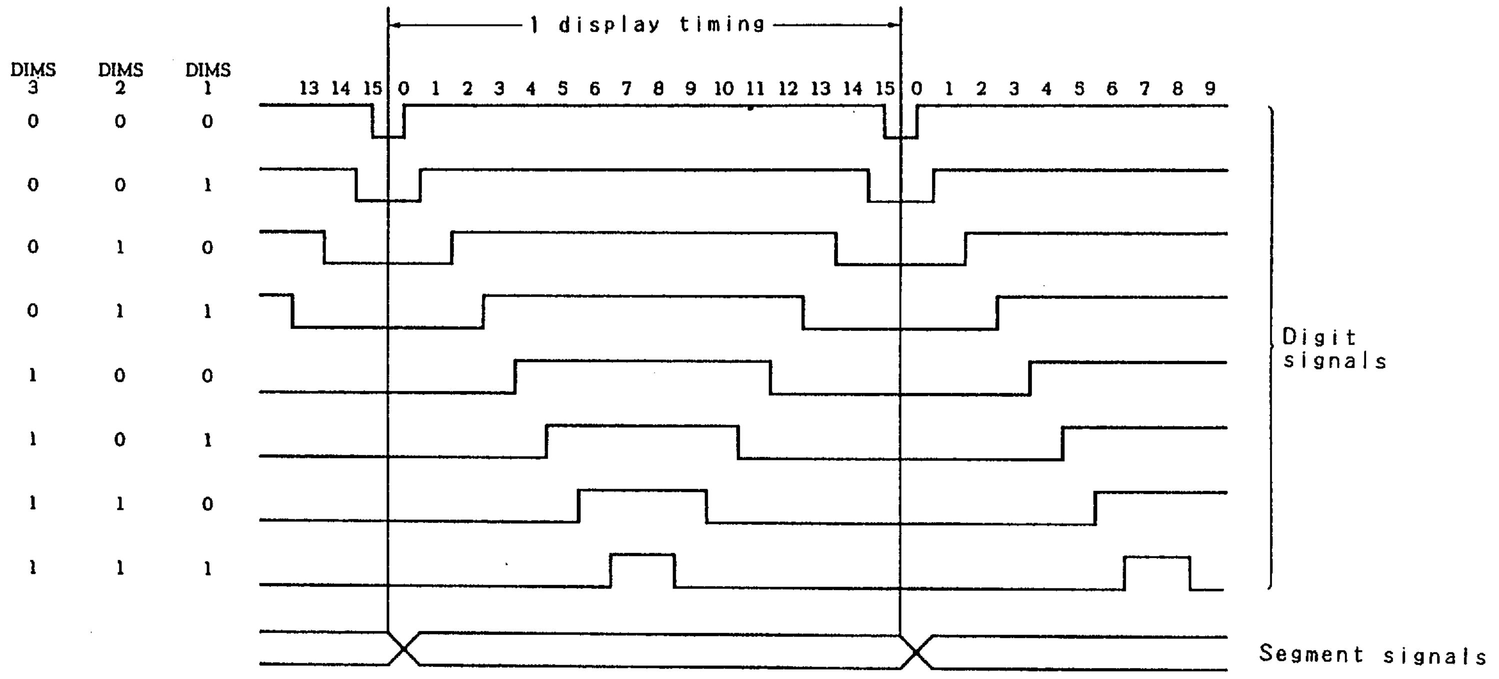
Digit signal cut width specification bit

DIMS3	DIMS2	DIMS1	Digit signal cut width
0	0	0	1/16
0	0	1	2/16
0	1	0	4/16
0	1	1	6/16
1	0	0	8/16
1	0	1	10/16
1	1	0	12/16
1	1	1	14/16

*1 244 us when the microcomputer operates at 4.19 MHz

*2 489 us when the microcomputer operates at 4.19 MHz

Fig. 5-107 Digit Signal Waveform



Remark: One display timing is 2048/fX (when DIMS0 = 1)
= 341 us (when operating at 6.0 MHz)(*1), or 1024/fX (when DIMS0 = 0)
= 171 us (when operating at 6.0 MHz)(*2).

*1 489 us when the microcomputer operates at 4.19 MHz
*2 244 us when the microcomputer operates at 4.19 MHz

(4) Static mode registers

The display output pins of the uPD75238, except T0 to T9, are also used as general output ports (P-ch open-drain).

A static mode register enables or disables display operation for the display output pins (that is, it specifies whether these pins are used as general ports or display outputs).

There are two static mode registers: static mode register A and static mode register B. Figures 5-108 and 5-109 show their formats.

These registers are set with an 8-bit manipulation instruction.

A $\overline{\text{RESET}}$ input clears all bits to 0.

(a) Static mode register A (STATA)

Static mode register A (STATA) enables or disables display operation of the S0/P120 to S15/P153/T10/PH3 pins.

Example: To set these pins as display segment outputs

```
SEL  MB15
MOV  XA,#00H
MOV  STATA,XA
```

Fig. 5-108 Format of Static Mode Register A

Address	7	6	5	4	3	2	1	0	Symbol
FD6H	0	0	0	0	STATA3	STATA2	STATA1	STATA0	STATA

S0 to S15 pins Static output/dynamic output selection bit

STATA3	STATA2	STATA1	STATA0	Output status of S0 to S15 pins
0	0	0	0	S0 to S15 become dynamic outputs. The number of segments and the number of digits are set by DSPM and DIGS.
1	1	1	1	S0 to S15 become static outputs. Static data is output by issuing an output instruction for ports 12 to 15.

Caution: Part of the S0 to S15 pins cannot be set as dynamic outputs while the other pins are set as static outputs.

(b) Static mode register B (STATB)

Static mode register B (STATB) enables or disables display operation of the S16/P100 to S23/P113 pins (that is, it specifies whether these pins are used as general ports or display outputs).

Example: To set these pins as display segment outputs

```
SEL   MB15
MOV   XA,#00H
MOV   STATB,XA
```

Fig. 5-109 Format of Static Mode Register B

Address	7	6	5	4	3	2	1	0	Symbol
FD4H	0	0	STATB5	STATB4	0	0	0	0	STATB

S16 to S23 pins Static output/dynamic output selection bit

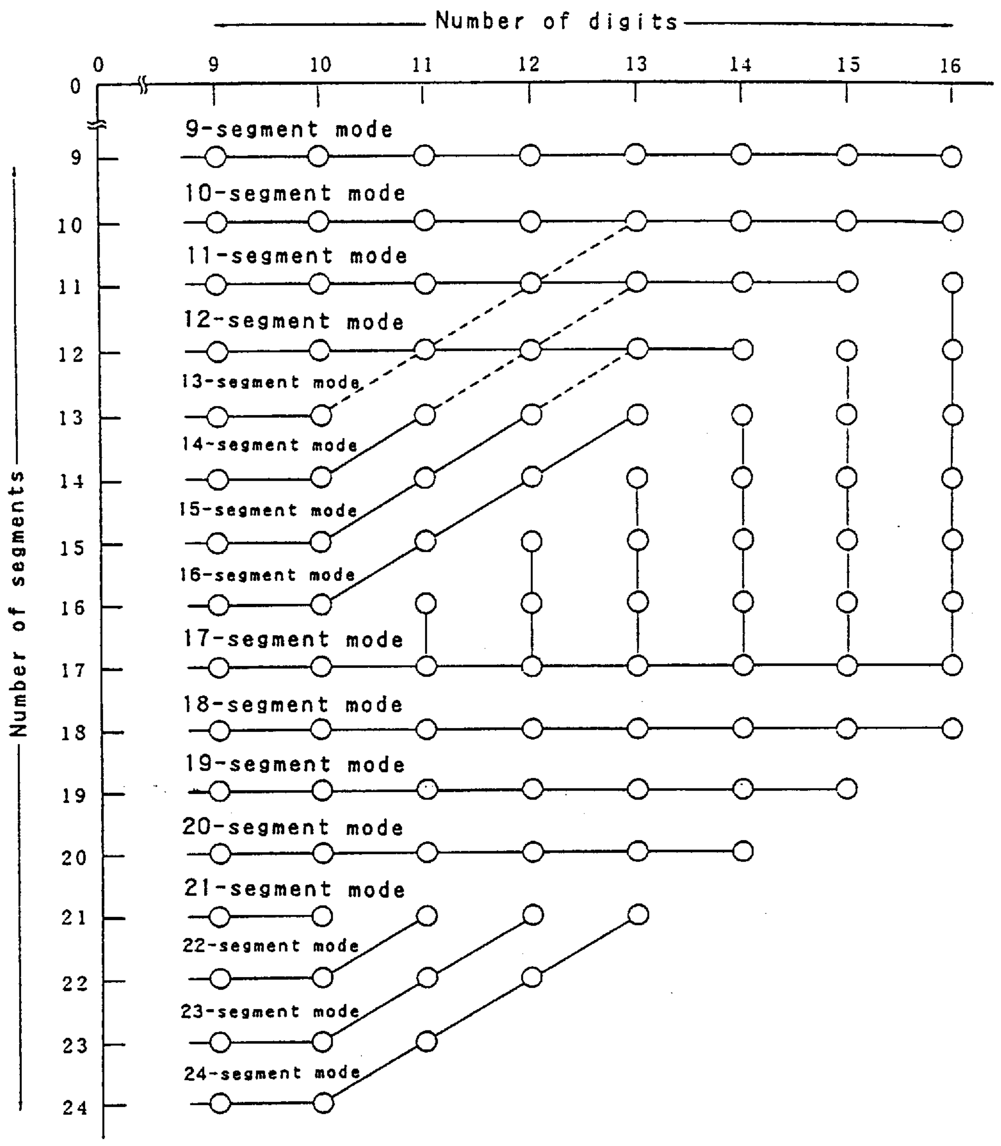
STATB5	STATB4	Output status of S16 to S23 pins
0	0	S16 to S23 become dynamic outputs. Output depends on the contents of 1A0H to 1B0H.
1	1	S16 to S23 become static outputs. Static data is output by issuing an output instruction for ports 10 and 11.

Caution: Part of the S16 to S23 pins cannot be set as dynamic outputs while the other pins are set as static outputs.

5.11.3 Selection of display mode

Figure 5-110 shows the possible numbers of segments and digits that the FIP controller/driver can display.

Fig. 5-110 Selection of Display Mode



Remark: Circles (o) indicate extended modes derived from the uPD75216A and uPD75217.

5.11.4 Display data memory

(1) Function and configuration of display data memory of the uPD75238

Display data memory holds segment data for display. It is mapped to addresses 1A0H to 1FFH in data memory. The display controller automatically reads display data (DMA operation). Area not used for display can be used as normal data memory.

Display data is manipulated with data memory manipulation instructions in 1-, 4-, and 8-bit units. With 8-bit manipulation instructions, only even-numbered addresses can be specified.

Example: To transfer the XA register contents to addresses 1A0H and 1A1H.

```
SEL    MB1
SET1   MBE        ; Select memory bank 1
MOV    HL,#0A0H   ; Transfer A0H to HL
MOV    @HL,XA     ; Transfer XA contents to
                    1A1H and 1A0H
```

Display data memory locations at 1FCH to 1FFH, 1BEH, and 1BFH are also used as key scan registers KS0, KS1, and KS2.

Table 5-17 Data Memory Locations Also Used as Key Scan Registers

Key scan register	Data memory locations
KS0	1FCH, 1FDH
KS1	1FEH, 1FFH
KS2	1BEH, 1BFH

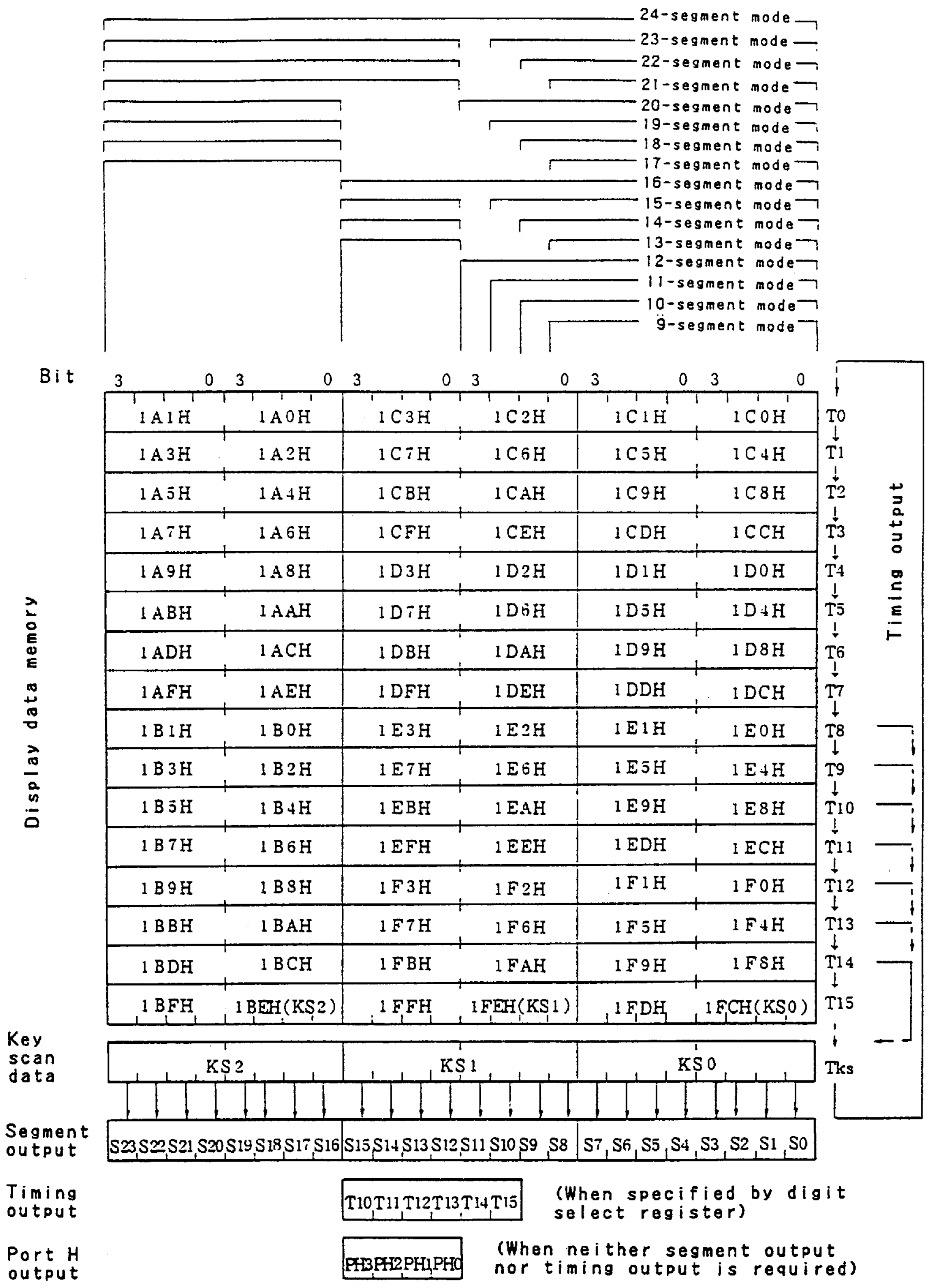
Figure 5-111 shows the correspondence between display data memory and segment outputs.

Cautions 1. Change of static output lags a maximum of one display cycle (a cycle of T output) behind the rewriting of KS1 data.

2. If segment output (S15 to S10) and timing output (T10 to T15) are specified at a time, timing output is assumed.

When S12 to S15 are not used as segment outputs nor timing output, these pins can function as port H (PH0 to PH3) to output data at address 1FFH.

Fig. 5-111 Correspondence between Display Data Memory and Segment Output



Bit	3 0	3 0	3 0	3 0	3 0	3 0	
	1A1H	1A0H	1C3H	1C2H	1C1H	1C0H	T0
	1A3H	1A2H	1C7H	1C6H	1C5H	1C4H	T1
	1A5H	1A4H	1CBH	1CAH	1C9H	1C8H	T2
	1A7H	1A6H	1CFH	1CEH	1CDH	1CCH	T3
	1A9H	1A8H	1D3H	1D2H	1D1H	1D0H	T4
	1ABH	1AAH	1D7H	1D6H	1D5H	1D4H	T5
	1ADH	1ACH	1DBH	1DAH	1D9H	1D8H	T6
	1AFH	1AEH	1DFH	1DEH	1DDH	1DCH	T7
	1B1H	1B0H	1E3H	1E2H	1E1H	1E0H	T8
	1B3H	1B2H	1E7H	1E6H	1E5H	1E4H	T9
	1B5H	1B4H	1EBH	1EAH	1E9H	1E8H	T10
	1B7H	1B6H	1EFH	1EEH	1EDH	1ECH	T11
	1B9H	1B8H	1F3H	1F2H	1F1H	1F0H	T12
	1BBH	1BAH	1F7H	1F6H	1F5H	1F4H	T13
	1BDH	1BCH	1FBH	1FAH	1F9H	1F8H	T14
	1BFH	1BEH(KS2)	1FFH	1FEH(KS1)	1FDH	1FCH(KS0)	T15

6427525 0094669 198

(2) Difference between the uPD75238 display data memory and the uPD75216A and uPD75217 display data memory

A special care must be paid when a program developed for the uPD75238 is transported to the uPD75216A or the uPD75217. This is because the uPD75216A and the uPD75217 allow up to 16 display segments, and do not contain data memory at addresses (1A0H + 4n, 1A1H + 4n).

5.11.5 Key scan registers and flag

(1) Key scan registers (KS0, KS1, and KS2)

The key scan registers (KS0, KS1, and KS2) set segment output data on the key scan timing that is mapped to display data memory locations (1FCH, 1FDH, 1FEH, 1FFH, 1BEH, and 1BFH).

KS0, KS1, and KS2 are 8-bit registers, which are usually manipulated with an 8-bit manipulation instruction. (One-bit or 4-bit manipulation is also possible for the lower 4 bits.)

The data set in KS0, KS1, and KS2 is output on the segment output pins on the key scan timing. During the key scan timing, the rewriting of KS0, KS1, and KS2 is immediately reflected in segment output data, which can be used for key scanning.

(2) Key scan flag (KSF)

The key scan flag (KSF) is set to 1 on the key scan timing, and is automatically reset to 0 on any other timing. KSF is mapped to bit 3 at address F8AH, and can be tested on a single-bit basis. KSF cannot be written to.

Testing this flag can determine whether the key scan timing is present, so that the validity of key input data can be determined.

5.11.6 Light leakage from the fluorescent indicator lamp

When the fluorescent indicator lamp is driven with the uPD75238, light may leak mainly for the causes described below.

(1) Short blanking time

Figure 5-112 shows the waveforms present when two-digit display is used, with the first digit turned on and the second digit turned off. When blanking time is short as shown in the figure, the T1 signal goes high before the segment signal is removed, resulting in light leakage. In general, a blanking time of 20 us is required. This requirement must be taken into consideration when a dimmer select register is set.

(2) Capacitance between the segment and grid of the fluorescent indicator lamp

Even if a sufficient blanking time is provided as shown in Figure 5-114, light may leak. A capacitance (indicated as C_{SG} in Figure 5-113) is present between the segment and grid of the fluorescent indicator lamp. This capacitance raises the voltage on the timing signal pin via C_{SG} when the segment signal is set on. If this voltage is higher than the cutoff voltage as shown in Figure 5-114, light leaks.

This spike changes with the value of C_{SG} and the value of the incorporated pull-down resistor R_L. As C_{SG} or R_L increases, this voltage becomes higher, which increases the chance of light leakage.

Fig. 5-112 Concept of Light Leakage Due to a Short Blanking Time

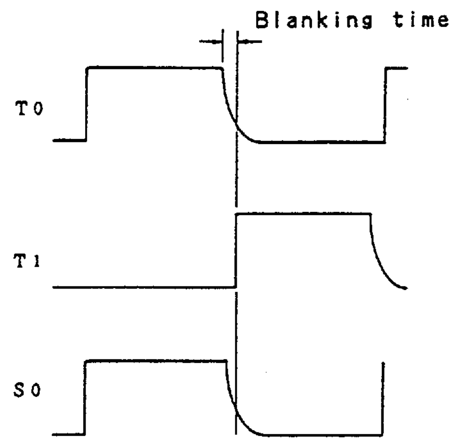
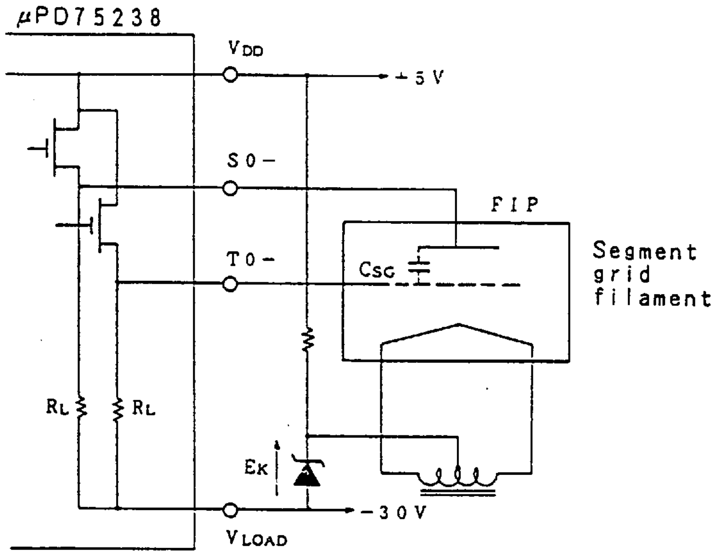
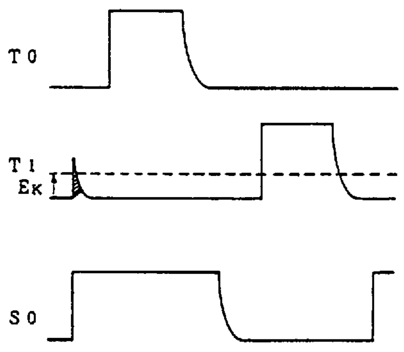


Fig. 5-113 Circuit Diagram Explaining Light Leakage Due to C_{SG}



E_K: Cutoff voltage
R_L: Internal pull-down resistor

Fig. 5-114 Concept of Light Leakage Due to C_{SG}



The value of C_{SG} depends on the display area of the fluorescent indicator lamp. As this area becomes larger, the value of C_{SG} increases. This means that the value of the pull-down resistor for preventing light leakage is determined by the size of the fluorescent indicator lamp.

On the other hand, the power loss allowance of the flat package limits the value of a pull-down resistor contained in the uPD75238 by a mask option to a relatively large value of 40 to 120 kΩ.
In some cases, light leakage may not be prevented with a pull-down resistor alone.

If sufficient display quality cannot be obtained, quality should be improved by a deeper back bias (to raise E_K), by attaching a filter to the fluorescent indicator lamp, or by connecting external tens-kilohm pull-down resistors to the timing signal pins.

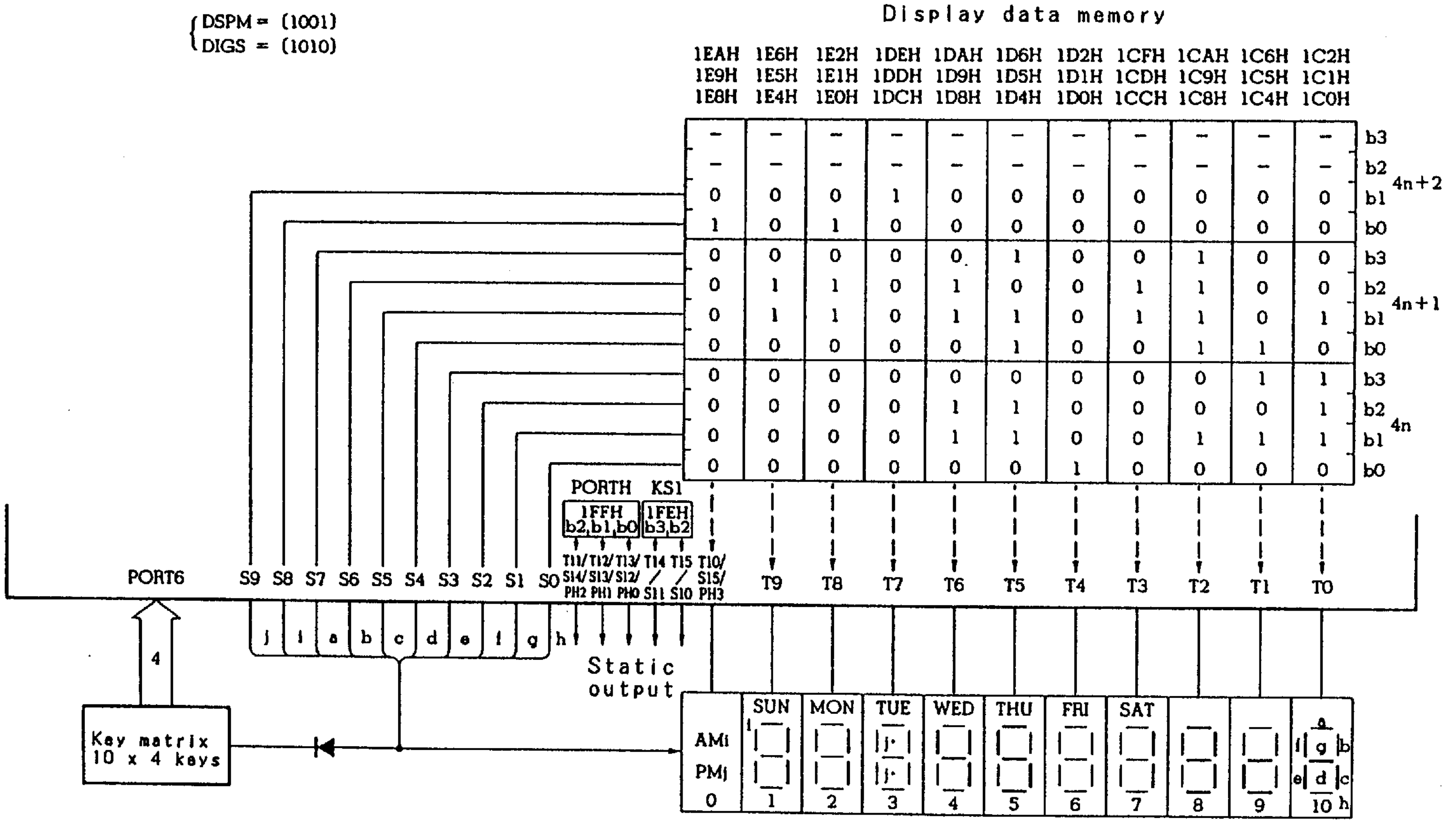
The possibility of light leakage caused by C_{SG} depends on the duty cycle of a spike to the entire display cycle. This means that more light leakage tends to occur with a smaller number of display digits. For the same reason, a slower display controller operation clock is better, and bit 0 (DIMS0) of the dimmer select register should be set to 1. In addition, light leakage can be reduced by lowering display intensity, or by setting more display digits (for example, setting 9-segment/16-digit display when 9-segment/8-digit display is actually required) if unused timing pins are available.

5.11.7 Examples of display

This section gives some examples of display in the following three modes:

(1)	10-segment mode:	10-segment/11-digit display (See Figure 5-115.)
(2)	14-segment mode:	14-segment/10-digit display (See Figure 5-116.)
(3)	16-segment mode:	16-segment/8-digit display (See Figure 5-117.)

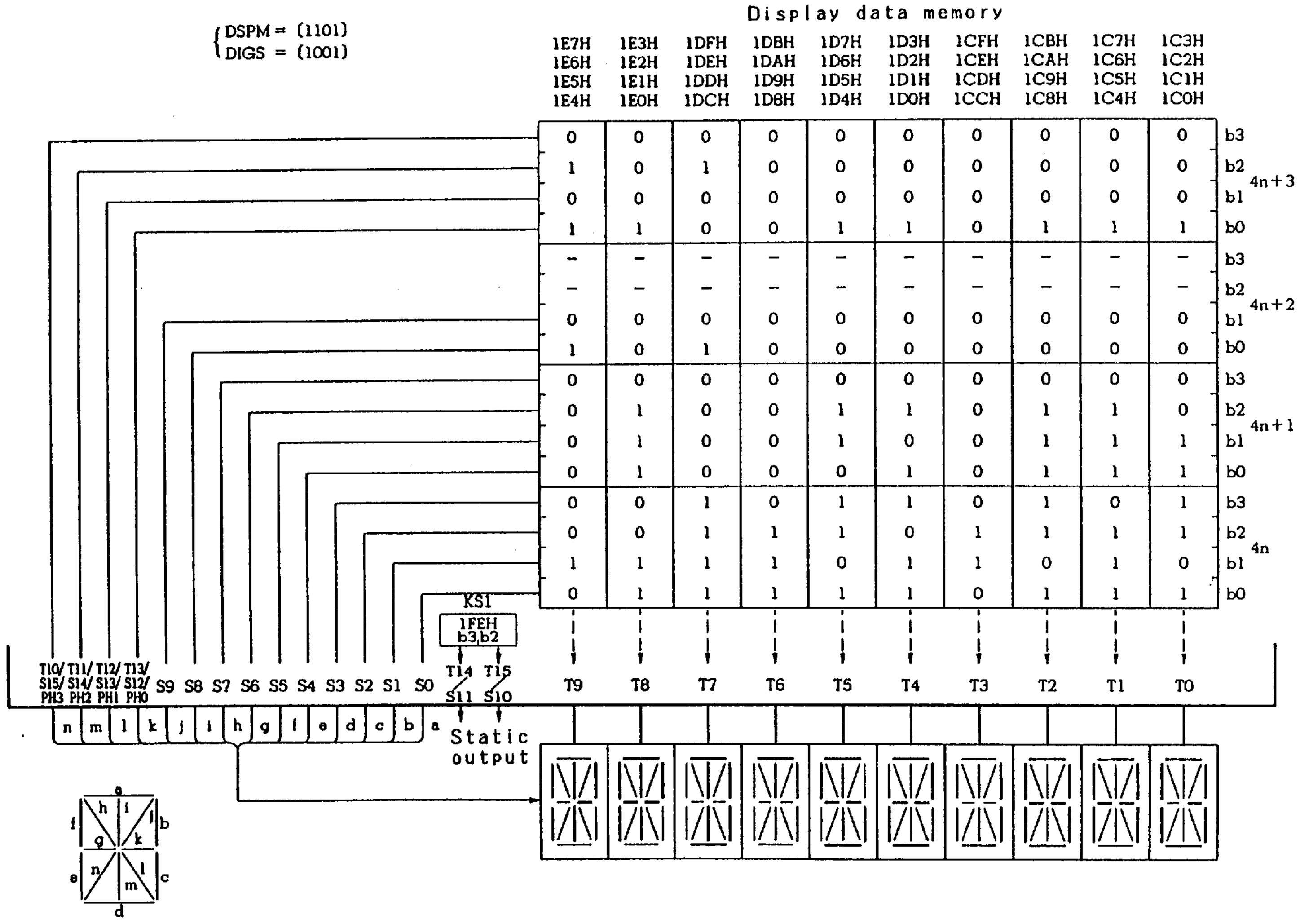
Fig. 5-115 10-Segment/11-Digit Display



	1EAH 1E9H 1E8H	1E6H 1E5H 1E4H	1E2H 1E1H 1E0H	1DEH 1DDH 1DCH	1DAH 1D9H 1D8H	1D6H 1D5H 1D4H	1D2H 1D1H 1D0H	1CFH 1CDH 1CCH	1CAH 1C9H 1C8H	1C6H 1C5H 1C4H	1C2H 1C1H 1C0H	
	T10/S15/PH3	T9	T8	T7	T6	T5	T4	T3	T2	T1	T0	
b3	–	–	–	–	–	–	–	–	–	–	–	4n+2
b2	–	–	–	–	–	–	–	–	–	–	–	
b1	0	0	0	1	0	0	0	0	0	0	0	
b0	1	0	1	0	0	0	0	0	0	0	0	
b3	0	0	0	0	0	1	0	0	1	0	0	4n+1
b2	0	1	1	0	1	0	0	1	1	0	0	
b1	0	1	1	0	1	1	0	1	1	0	1	
b0	0	0	0	0	0	1	0	0	1	1	0	
b3	0	0	0	0	0	0	0	0	0	1	1	4n
b2	0	0	0	0	1	1	0	0	0	0	1	
b1	0	0	0	0	1	1	0	0	1	1	1	
b0	0	0	0	0	0	0	1	0	0	0	0	

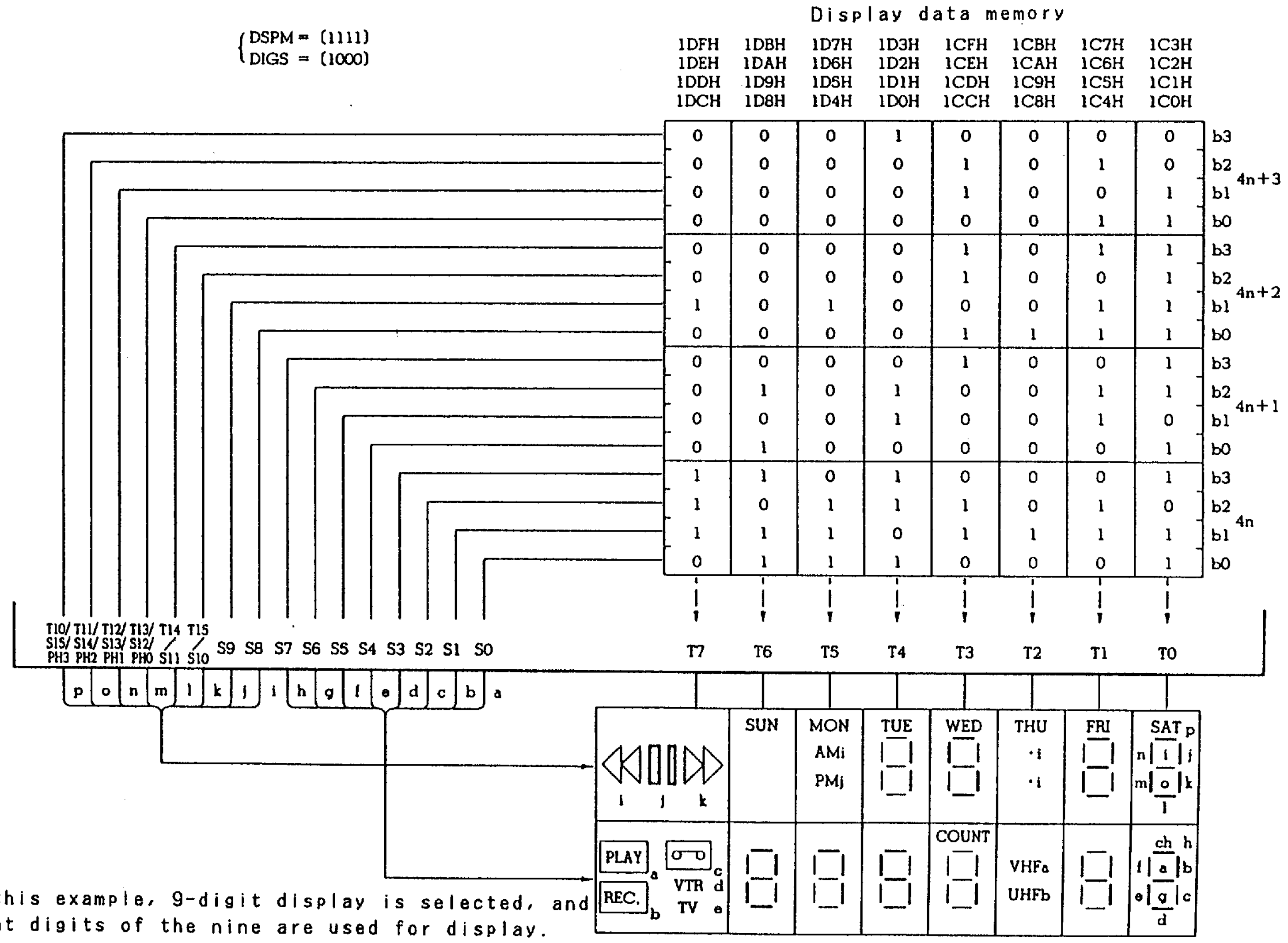
Caution: As shown in this example, the a to g display elements are assigned to the S1 to S7 pins.

Fig. 5-116 14-Segment/10-Digit Display



		1E7H 1E6H 1E5H 1E4H	1E3H 1E2H 1E1H 1E0H	1DFH 1DEH 1DDH 1DCH	1DBH 1DAH 1D9H 1D8H	1D7H 1D6H 1D5H 1D4H	1D3H 1D2H 1D1H 1D0H	1CFH 1CEH 1CDH 1CCH	1CBH 1CAH 1C9H 1C8H	1C7H 1C6H 1C5H 1C4H	1C3H 1C2H 1C1H 1C0H
4n+3	b3	0	0	0	0	0	0	0	0	0	0
	b2	1	0	1	0	0	0	0	0	0	0
	b1	0	0	0	0	0	0	0	0	0	0
	b0	1	1	0	0	1	1	0	1	1	1
4n+2	b3	–	–	–	–	–	–	–	–	–	–
	b2	–	–	–	–	–	–	–	–	–	–
	b1	0	0	0	0	0	0	0	0	0	0
	b0	1	0	1	0	0	0	0	0	0	0
4n+1	b3	0	0	0	0	0	0	0	0	0	0
	b2	0	1	0	0	1	1	0	1	1	0
	b1	0	1	0	0	1	0	0	1	1	1
	b0	0	1	0	0	0	1	0	1	1	1
4n	b3	0	0	1	0	1	1	0	1	0	1
	b2	0	0	1	1	1	0	1	1	1	1
	b1	1	1	1	1	0	1	1	0	1	0
	b0	0	1	1	1	1	1	0	1	1	1
		T9	T8	T7	T6	T5	T4	T3	T2	T1	T0

Fig. 5-117 16-Segment/8-Digit Display



	1DFH 1DEH 1DDH 1DCH	1DBH 1DAH 1D9H 1D8H	1D7H 1D6H 1D5H 1D4H	1D3H 1D2H 1D1H 1D0H	1CFH 1CEH 1CDH 1CCH	1CBH 1CAH 1C9H 1C8H	1C7H 1C6H 1C5H 1C4H	1C3H 1C2H 1C1H 1C0H		
S15/T10/PH3	0	0	0	1	0	0	0	0	b3	4n+3
S14/T11/PH2	0	0	0	0	1	0	1	0	b2	
S13/T12/PH1	0	0	0	0	1	0	0	1	b1	
S12/T13/PH0	0	0	0	0	0	0	1	1	b0	
T14/S11	0	0	0	0	1	0	1	1	b3	4n+2
T15/S10	0	0	0	0	1	0	0	1	b2	
S9	1	0	1	0	0	0	1	1	b1	
S8	0	0	0	0	1	1	1	1	b0	
S7	0	0	0	0	1	0	0	1	b3	4n+1
S6	0	1	0	1	0	0	1	1	b2	
S5	0	0	0	1	0	0	1	0	b1	
S4	0	1	0	0	0	0	0	1	b0	
S3	1	1	0	1	0	0	0	1	b3	4n
S2	1	0	1	1	1	0	1	0	b2	
S1	1	1	1	0	1	1	1	1	b1	
S0	0	1	1	1	0	0	0	1	b0	
	T7	T6	T5	T4	T3	T2	T1	T0		

Remark: In this example, 9-digit display is selected, and eight digits of the nine are used for display.

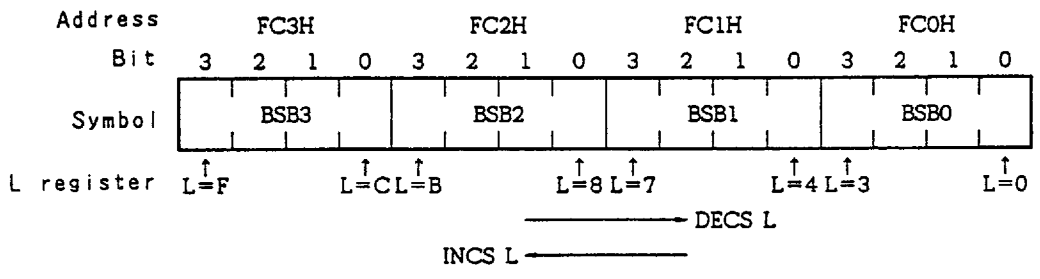
5.12 Bit Sequential Buffer: 16-Bit

The bit sequential buffer (BSB0 to BSB3) is special data memory for bit manipulations.

In particular, BSB0 to BSB3 allow bit manipulations to be performed very easily by sequentially changing address and bit specifications. So BSB0 to BSB3 are useful in processing long data bit by bit.

As shown in Figure 5-118, this data memory consists of 16 bits, and allows pmem.@L addressing with a bit manipulation instruction. This addressing uses the L register for indirect bit specification. In this case, only by incrementing or decrementing the L register in a program loop, the bit to be manipulated can be sequentially shifted for continued processing.

Fig. 5-118 Format of the Bit Sequential Buffer



Remark: In pmem.@L addressing, bit specification is shifted according to the L register. With pmem.@L addressing, BSB can be manipulated at any time regardless of MBE/MBS specification.

Data can also be manipulated by direct addressing. The buffer can be used for applications such as continuous 1-bit data input or output operations by combining direct 1-bit, 4-bit, and 8-bit addressing with pmem.@L addressing. In 8-bit manipulation, the higher eight bits or lower eight bits are manipulated by specifying BSB0 or BSB2.

5.12.1 Applications of the bit sequential buffer

(1) 16-bit data in certain data memory areas BUFF1 and BUFF2 in memory bank 0 is output serially from bit 0 in port 3.

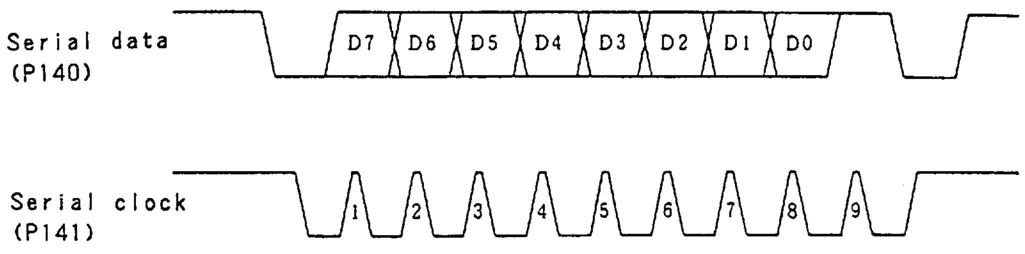
<Sample program>

```
     CLR1  MBE
     MOV   XA,BUFF1
     MOV   BSB0,XA        ; Set BSB0 and BSB1
     MOV   XA,BUFF2
     MOV   BSB2,XA        ; Set BSB2 and BSB3
     MOV   L,#0
LOOP:MOV1  CY,BSB0,@L     ; CY ← specified bit in BSB
     MOV1  PORT3.0 CY     ; Bit 0 in port 3 ← CY
     INCS  L              ; L ← L + 1
     BR    LOOP
     RET
```

(2) Data is transmitted in arbitrary serial transfer format (1)

A serial transfer format shown in Figure 5-119 is used for data output. (P140 is used for serial data output, and P141 is used for the clock. The data to be transferred is already set in BSB2 and BSB3, and bit 3 of BSB1 is set to 1, and bit 2 is set to 0.)

Fig. 5-119 Serial Transfer Format (I)



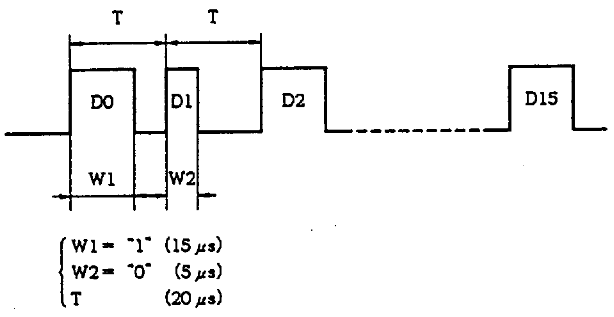
<Sample program>

```
       MOV    L,#0FH
       CLR1   PORT14.0      ; Start transfer
LOOP:CLR1   PORT14.1      ; CLK ←— 0
       MOV1   CY,BSB0.@L
       MOV1   PORT14.0,CY   ; Data output
       SET1   PORT14.1      ; CLK ←— 1
       DECS   L
       SKE    L,#5
       BR     LOOP
       SET1   PORT14.0      ; End transfer
```

(3) Data is transferred in arbitrary serial transfer format (2)

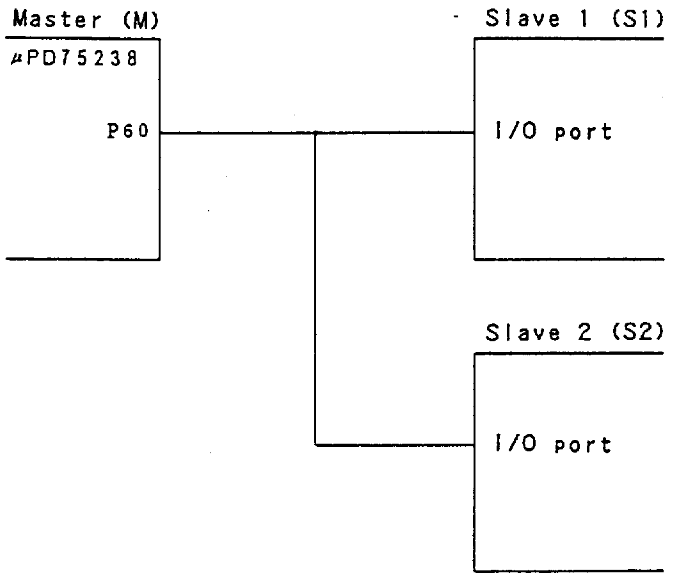
A serial transfer format shown in Figure 5-120 is used for data transmission. (Transmission data is 16-bit serial data, and is input and output on the P60 pin. f_X = 4.19 MHz, and the minimum instruction execution time is 1 us.)

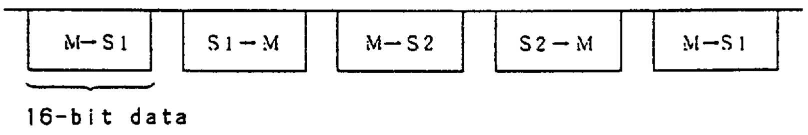
Fig. 5-120 Serial Transfer Format (II)



When data is sent and received alternately in the above data transfer format, only one signal line is needed for high-speed data transfer, as shown in Figure 5-121.

Fig. 5-121 Bus Configuration Example





<Sample program for reception>

```
        MOV   L,#0
CHECK:SKT   PORT6.0       ; P60 = 1?
        BR    CHECK         ; NO
        NOP                 ; YES
        NOP
        NOP
        MOV1  CY,PORT6.0    ;
        MOV   BSB0.@L,CY
        NOP
        NOP
        INCS  L
        BR    CHECK
```

<Sample program for sending>

```
      MOV   L,#0
LOOP:SET1  PORT6.0       ; P60 ← 1
      MOV1  CY,BSB0.@L    ; CY ← transfer data
      NOP
      MOV1  PORT6.0,CY    ; P60 ← data
      MOV   A,#0EH        ; Wait for 8 machine cycles
WAIT:INCS  A
      BR    WAIT
      NOP
      NOP
      CLR1  PORT6.0       ; P60 ← 0
      INCS  L
      BR    LOOP
```