

## 4-BIT SINGLE-CHIP MICROCOMPUTER

### DESCRIPTION

The μPD75237 is a microcomputer with a CPU capable of 1-, 4-, and 8-bit-wise data processing, a ROM, a RAM, I/O ports, a fluorescent display tube (FIP<sup>®</sup>) controller/driver, A/D converters, a watch timer, a timer/pulse generator capable of outputting 14-bit PWM, a serial interface and a vectored interrupt function integrated on a single-chip.

The μPD75237 has the more improved peripheral functions including the RAM capacity, FIP controller/driver display capabilities, I/O ports, A/D converter and serial interface than those of the μPD75217.

The μPD75237 is most suited for advanced and popular VCR timer and tuner applications, single-chip configurations of system computers, advanced CD players and advanced microwave ovens.

The μPD75P238 PROM product and various types of development tools (IE-75001-R, assemblers and others) are available for evaluation in system development or small-volume production.

### FEATURES

- Built-in, large-capacity ROM and RAM
  - Program memory (ROM): 24K × 8
  - Data memory (RAM): 1K × 4
- I/O port: 64 ports (except FIP dedicated pins)
- Minimum instruction execution time: 0.67 μs (when operated at 6.0 MHz)
- Instruction execution time varying function to achieve a wide range of power supply voltages
- Built-in programmable FIP controller/driver
  - Number of segments: 9 to 24
  - Number of digits: 9 to 16
- 8-bit A/D converter: 8 channels
- Powerful timer/counter function: 5 channels
- 8-bit serial interface: 2 channels
- Interrupt function with importance attached to applications
- Product with built-in PROM: μPD75P238

### ORDERING INFORMATION

Ordering Code	Package	Quality Grade
μPD75237GJ-xxx-5BG	94-pin plastic QFP (20 × 20 mm)	Standard

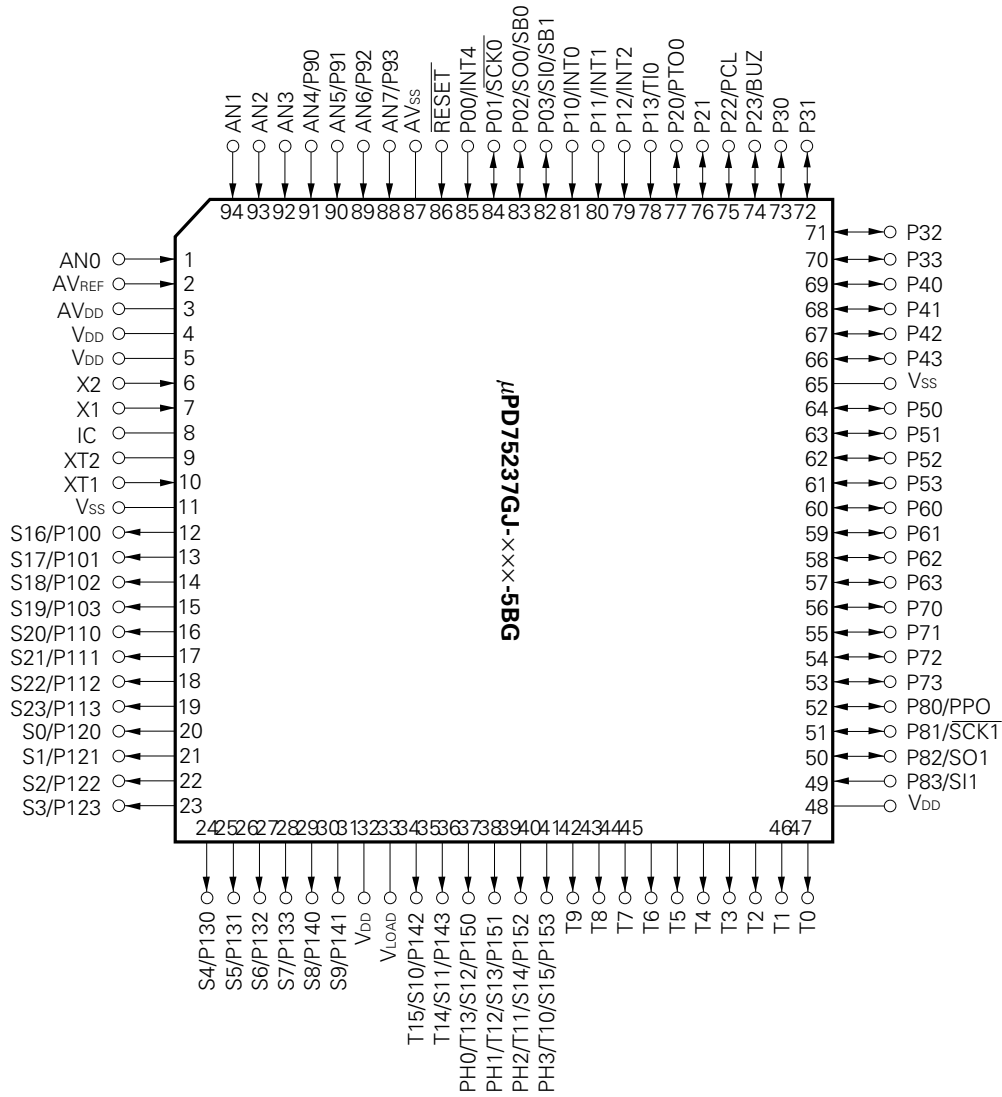
Please refer to "Quality grade on NEC Semiconductor Devices" (Document number IEI-1209) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

The information in this document is subject to change without notice.

LIST OF μPD75237 FUNCTIONS

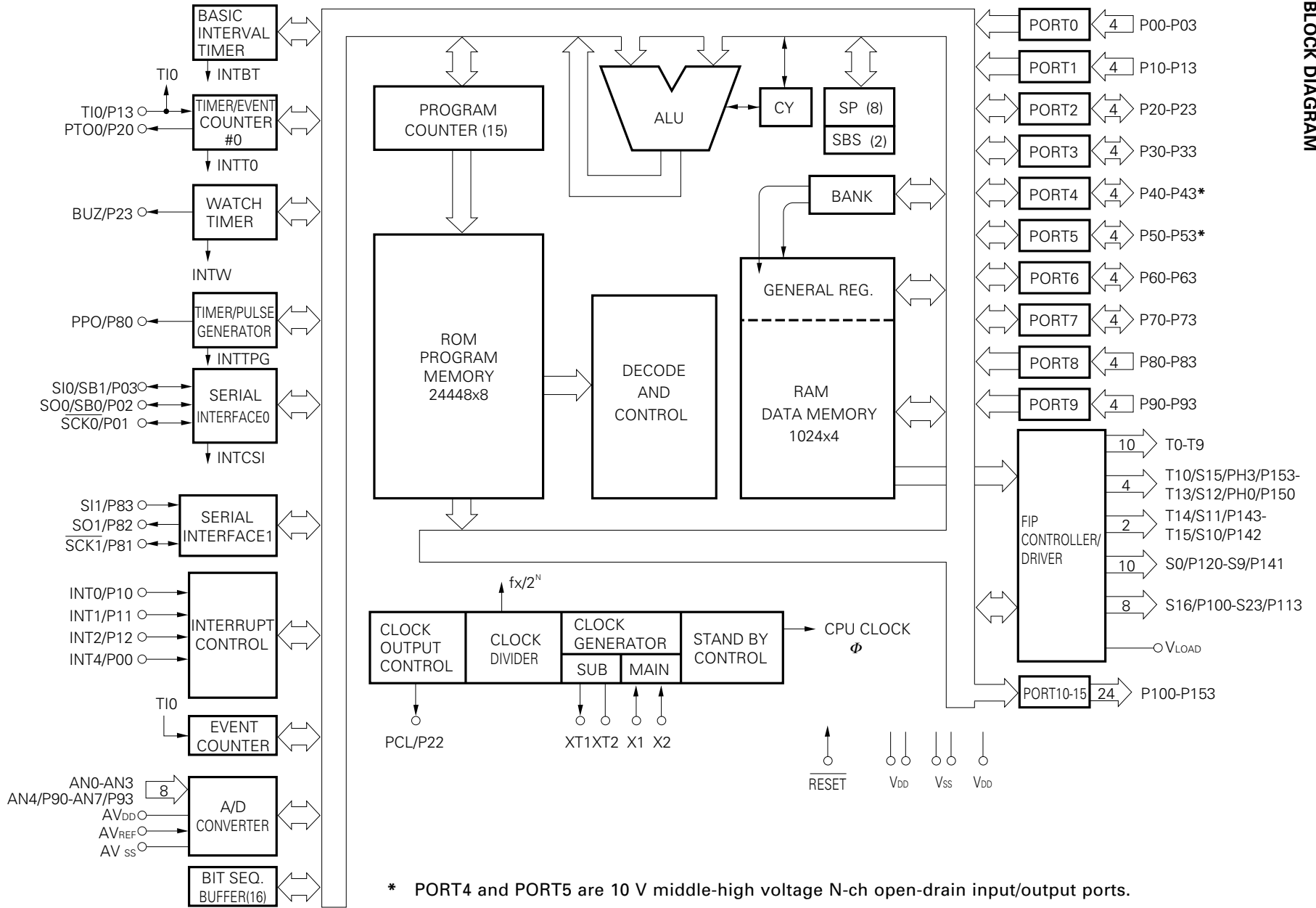
Item	Function
Built-in memory capacity	ROM: 24448 × 8 bits, RAM: 1024 × 4bits
I/O line (except FIP dedicated pins)	64 lines <ul style="list-style-type: none"> <li>○ Input pin : 16</li> <li>○ Input/output pin : 24</li> <li>○ Output pin : 24</li> </ul>
Instruction cycle	<ul style="list-style-type: none"> <li>○ 0.67 μs/1.33 μs/2.67 μs/10.7 μs (when operated at 6.0 MHz)</li> <li>○ 0.95 μs/1.91 μs/3.82 μs/15.3 μs (when operated at 4.19 MHz)</li> <li>○ 122 μs (when operated at 32.768 kHz)</li> </ul>
Fluorescent display tube (FIP) controller/driver	<ul style="list-style-type: none"> <li>○ Number of segments : 9 to 24</li> <li>○ Number of digits : 9 to 16</li> <li>○ Dimmer function : 8 levels</li> <li>○ Pull-down resistor mask option</li> <li>○ Key scan interrupt generation enabled</li> </ul>
Timer/counter	5 channels <ul style="list-style-type: none"> <li>○ Basic interval timer : Watchdog timer applicable</li> <li>○ Timer/event counter</li> <li>○ Watch timer : Buzzer output enabled</li> <li>○ Timer/pulse generator : 14-bit PWM output enabled</li> <li>○ Event counter</li> </ul>
Serial interface	2 channels <ul style="list-style-type: none"> <li>○ SBI/3-wire type</li> <li>○ 3-wire type</li> </ul>
Interrupt	<ul style="list-style-type: none"> <li>○ Multi-interrupt enabled by hardware</li> <li>○ External interrupt: 3 interrupts               <ul style="list-style-type: none"> <li>○ Both-edge detection</li> <li>○ Detected edge programmable (with noise remove function)</li> <li>○ Detected edge programmable</li> <li>○ Rising edge detection</li> </ul> </li> <li>○ External test input: 1 input               <ul style="list-style-type: none"> <li>○ Timer/pulse generator</li> <li>○ Timer/event counter</li> </ul> </li> <li>○ Internal interrupt: 5 interrupts               <ul style="list-style-type: none"> <li>○ Basic interval timer</li> <li>○ Serial interface #0</li> <li>○ Key scan interrupt</li> </ul> </li> <li>○ Internal test input: 2 inputs               <ul style="list-style-type: none"> <li>○ Clock timer</li> <li>○ Serial interface #1</li> </ul> </li> </ul>
System clock oscillator	<ul style="list-style-type: none"> <li>○ Main system clock : 6.0 MHz, 4.19 MHz</li> <li>○ Subsystem clock : 32.768 kHz standard</li> </ul>
Mask option	<ul style="list-style-type: none"> <li>○ High withstand voltage port : Pull-down resistor or open-drain output</li> <li>○ Ports 4 and 5 : Pull-up resistors</li> <li>○ Port 7 : Pull-down resistor</li> </ul>
Operating temperature range	-40 to +85 °C
Operating voltage	2.7 to 6.0 V (standby data hold: 2.0 to 6.0 V)
Package	94-pin plastic QFP (20 × 20 mm)

PIN CONFIGURATION



**Note** Be sure to supply power to AV<sub>DD</sub> , V<sub>DD</sub> , V<sub>SS</sub> and AV<sub>SS</sub> pins (pin Nos. 3, 4, 5, 11, 30, 48, 65 and 87) .

**Remarks** Connect the IC (Internally Connected) pin to GND.



BLOCK DIAGRAM

## CONTENTS

<b>1. PIN FUNCTIONS</b> .....	<b>7</b>
1.1 PORT PINS .....	7
1.2 NON-PORT PINS .....	9
1.3 PIN INPUT/OUTPUT CIRCUIT LIST .....	11
1.4 RECOMMENDED CONNECTIONS OF $\mu$ PD75237 UNUSED PINS .....	15
<b>2. <math>\mu</math>PD75237 ARCHITECTURE AND MEMORY MAP</b> .....	<b>16</b>
2.1 DATA MEMORY BANK CONFIGURATION AND ADDRESSING MODE .....	16
2.2 GENERAL REGISTER BANK CONFIGURATION .....	19
2.3 MEMORY MAPPED I/O .....	22
<b>3. INTERNAL CPU FUNCTIONS</b> .....	<b>28</b>
3.1 PROGRAM COUNTER (PC): 15 BITS .....	28
3.2 PROGRAM MEMORY (ROM): 24448 WORDS $\times$ 8 BITS .....	28
3.3 DATA MEMORY .....	30
3.4 GENERAL REGISTER: 8 $\times$ 4 BITS $\times$ 4 BANKS .....	32
3.5 ACCUMULATOR .....	33
3.6 STACK POINTER (SP) AND STACK BANK SELECT REGISTER (SBS) .....	33
3.7 PROGRAM STATUS WORD (PSW): 8 BITS .....	36
3.8 BANK SELECT REGISTER (BS) .....	40
<b>4. PERIPHERAL HARDWARE FUNCTIONS</b> .....	<b>41</b>
4.1 DIGITAL INPUT/OUTPUT PORTS .....	41
4.2 CLOCK GENERATOR .....	50
4.3 CLOCK OUTPUT CIRCUIT .....	58
4.4 BASIC INTERVAL TIMER .....	61
4.5 TIMER/EVENT COUNTER .....	63
4.6 WATCH TIMER .....	69
4.7 TIMER/PULSE GENERATOR .....	71
4.8 EVENT COUNTER .....	77
4.9 SERIAL INTERFACE .....	79
4.10 A/D CONVERTER .....	113
4.11 BIT SEQUENTIAL BUFFER: 16 BITS .....	119
4.12 FIP CONTROLLER/DRIVER .....	119
<b>5. INTERRUPT FUNCTIONS</b> .....	<b>131</b>
5.1 INTERRUPT CONTROL CIRCUIT CONFIGURATION .....	131
5.2 INTERRUPT CONTROL CIRCUIT HARDWARE DEVICES .....	133
5.3 INTERRUPT SEQUENCE .....	138
5.4 MULTI-INTERRUPT SERVICE CONTROL .....	139
5.5 VECTOR ADDRESS SHARING INTERRUPT SERVICING .....	141
<b>6. STANDBY FUNCTIONS</b> .....	<b>142</b>
6.1 STANDBY MODE SETTING AND OPERATING STATE .....	142
6.2 STANDBY MODE RELEASE .....	144
6.3 OPERATION AFTER STANDBY MODE RELEASE .....	146

7. RESET FUNCTIONS ..... 147

8. INSTRUCTION SET ..... 150

    8.1 CHARACTERISTIC INSTRUCTIONS OF μPD75237 ..... 150

    8.2 INSTRUCTION SET AND OPERATION ..... 153

    8.3 OPERATION CODES ..... 162

9. MASK OPTION SELECTION ..... 168

10. APPLICATION BLOCK DIAGRAM ..... 169

11. ELECTRICAL SPECIFICATIONS ..... 170

★ 12. CHARACTERISTIC CURVES (REFERENCE VALUES) ..... 183

13. PACKAGE INFORMATION ..... 185

14. RECOMMENDED SOLDERING CONDITIONS ..... 186

APPENDIX A. LIST OF μPD75238 SERIES PRODUCT FUNCTIONS ..... 187

APPENDIX B. DEVELOPMENT TOOLS ..... 188

1. PIN FUNCTIONS

1.1 PORT PINS (1/2)

Pin Name	I/O	Dual-Function Pin	Function	8-Bit I/O	After Reset	Input / Output Circuit Type *1	
P00	Input	INT4	4-bit input port (PORT0). Built-in pull-up resistor can be specified in 3-bit units by software for P01 to P03.	×	Input	Ⓑ	
P01		SCK0				Ⓕ - A	
P02		SO0/SB0				Ⓕ - B	
P03		SI0/SB1				M - C	
P10	Input	INT0	4-bit input port (PORT1). Built-in pull-up resistor can be specified in 4-bit units by software.	×	Input	Ⓑ - C	
P11		INT1					Noise removing function available
P12		INT2					
P13		TI0					
P20	Input/output	PTO0	4-bit input/ output port (PORT2). Built-in pull-up resistor can be specified in 4-bit units by software.	×	Input	E - B	
P21		—					
P22		PCL					
P23		BUZ					
P30 *2	Input/output	—	Programmable 4-bit input/ output port (PORT3). Input/ output specifiable in 1-bit units. Built-in pull-up resistor can be specified in 4-bit units by software.	×	Input	E - C	
P31 *2		—					
P32 *2		—					
P33 *2		—					
*2 P40 to P43	Input/output	—	N-ch open-drain 4-bit input/output port (PORT4). Pull-up resistor can be incorporated in 1-bit units (mask option). 10 V withstand voltage with open drain.	○	High level (when a pull-up resistor is incorporated) or high impedance	M	
*2 P50 to P53	Input/output	—	N-ch open-drain 4-bit input/ output port (PORT5). Pull-up resistor can be incorporated in 1-bit units (mask option). 10 V withstand voltage with open drain.		High level (when a pull-up resistor is incorporated) or high impedance	M	
P60	Input/output	—	Programmable 4-bit input/output port (PORT6). Input/output specifiable in 1-bit units. Built-in pull-up resistor can be specified in 4-bit units by software.	○	Input	E - C	
P61		—					
P62		—					
P63		—					
P70	Input/output	—	4-bit input/output port (PORT7). Built-in pull-down resistor can be incorporated in 1-bit units (mask option).	○	V <sub>SS</sub> level (when a pull-down resistor is incorporated) or high impedance	V	
P71		—					
P72		—					
P73		—					

- \* 1. Schmitt trigger inputs are circled.
- 2. Can drive LED directly.

1.1 PORT PINS (2/2)

Pin Name	I/O	Dual-Function Pin	Function	8-Bit I/O	After Reset	Input / Output Circuit Type *
P80	Input/output	PPO	4-bit input port (PORT8).	×	Input	A
P81	Input/output	$\overline{\text{SCK1}}$				Ⓣ
P82	Input/output	SO1				E
P83	Input	SI1				ⓑ
P90	Input	AN4	4-bit input port (PORT9).	×	Input	Y – A
P91		AN5				
P92		AN6				
P93		AN7				
P100	Output	S16	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option).	○	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated), V <sub>SS</sub> level (when a pull-down resistor to V <sub>SS</sub> is incorporated) or high impedance	I – F
P101		S17				
P102		S18				
P103		S19				
P110	Output	S20	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option).	○	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated), V <sub>SS</sub> level (when a pull-down resistor to V <sub>SS</sub> is incorporated) or high impedance	I – C
P111		S21				
P112		S22				
P113		S23				
P120	Output	S0	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option).	○	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated) or high impedance	I – C
P121		S1				
P122		S2				
P123		S3				
P130	Output	S4	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option).	○	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated) or high impedance	I – C
P131		S5				
P132		S6				
P133		S7				
P140	Output	S8	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option). P142 and P143 can drive LED directly.	○	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated) or high impedance	I – C
P141		S9				
P142		S10/T15				
P143		S11/T14				
P150	Output	S12/T13/PH0	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option). These ports can drive LED directly.	○	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated) or high impedance	I – C
P151		S13/T12/PH1				
P152		S14/T11/PH2				
P153		S15/T10/PH3				
PH0	Output	S12/T13/P150	P-ch open-drain 4-bit high-voltage output port. Pull-down resistor can be incorporated (mask option).	×	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated) or high impedance	I – C
PH1		S13/T12/P151				
PH2		S14/T11/P152				
PH3		S15/T10/P153				

\* Schmitt trigger inputs are circled.



1.2 NON-PORT PINS (1/2)

Pin Name	I/O	Dual-Function Pin	Function	After Reset	Input / Output Circuit Type *			
T0 to T9	Output	—	Digit output high-voltage high-current output pins.	V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated) or high impedance.	I – C			
T10/S15 to T13/S12		PH3/P153 to PH0/P150	Digit/segment output dual-function high-voltage high-current output pins. Extra pins can be used as PORTH. These pins can be used as PORT15 in the static mode.					
T14/S11		P143	Digit/segment output dual-function high-voltage high-current output pins. These pins can be used as POTR14 in the static mode.					
T15/S10		P142	FIP controller/driver output pins. Pull-down resistor can be incorporated in bit units (mask option).			Segment high-voltage output pins. These pins can be used as PORT12 to PORT14 in the static mode.		
S0 to S3		P120 to P123						
S4 to S7		P130 to P133						
S8		P140						
S9		P141	Segment high-voltage output pins. These pins can be used as PORT10 and PORT11 in the static mode.			V <sub>LOAD</sub> level (when a pull-down resistor to V <sub>LOAD</sub> is incorporated), V <sub>SS</sub> level (when a pull-down resistor to V <sub>SS</sub> is incorporated) or high impedance		
S16 to S19		P100 to P103						
S20 to S23		P110 to P113	External event pulse input to timer/event counter #0 and event counter #1.			—	Ⓑ – C	
T10	P13							
PTO0	Output	P20		Timer/event counter output.	Input			E – B
PCL	Output	P22		Clock output.	Input			E – B
BUZ	Output	P23		Fixed frequency output (for buzzer or system clock trimming).	Input			E – B
$\overline{\text{SCK0}}$	Input/output	P01		Serial clock input/output.	Input			Ⓕ – A
SO0/SB0	Input/output	P02		Serial data output. Serial bus input/output.	Input			Ⓕ – B
SI0/SB1	Input/output	P03		Serial data input. Serial bus input/output.	Input			M – C

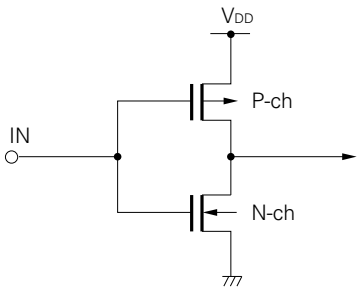
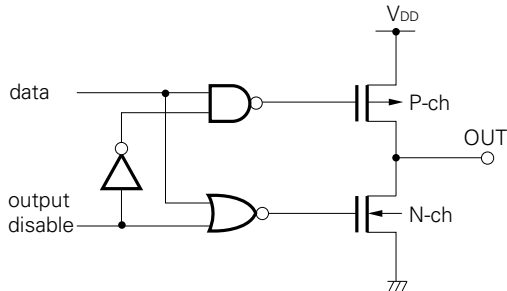
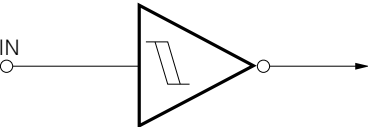
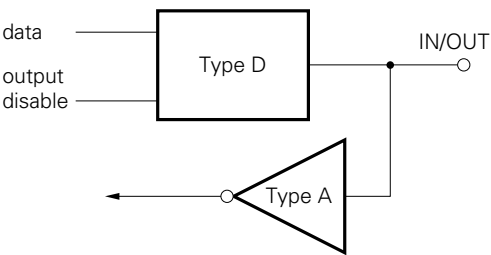
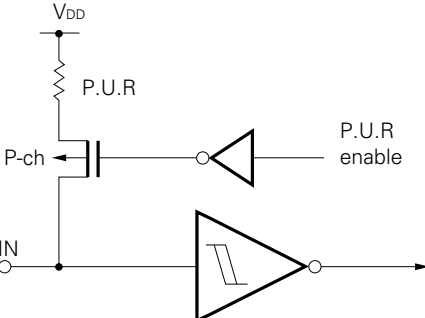
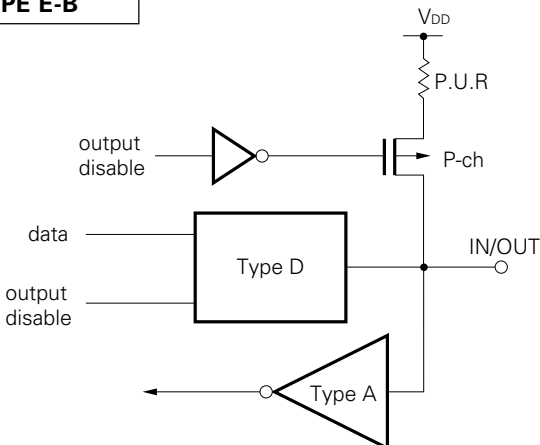
\* Schmitt trigger inputs are circled.

1.2 NON-PORT PINS (2/2)

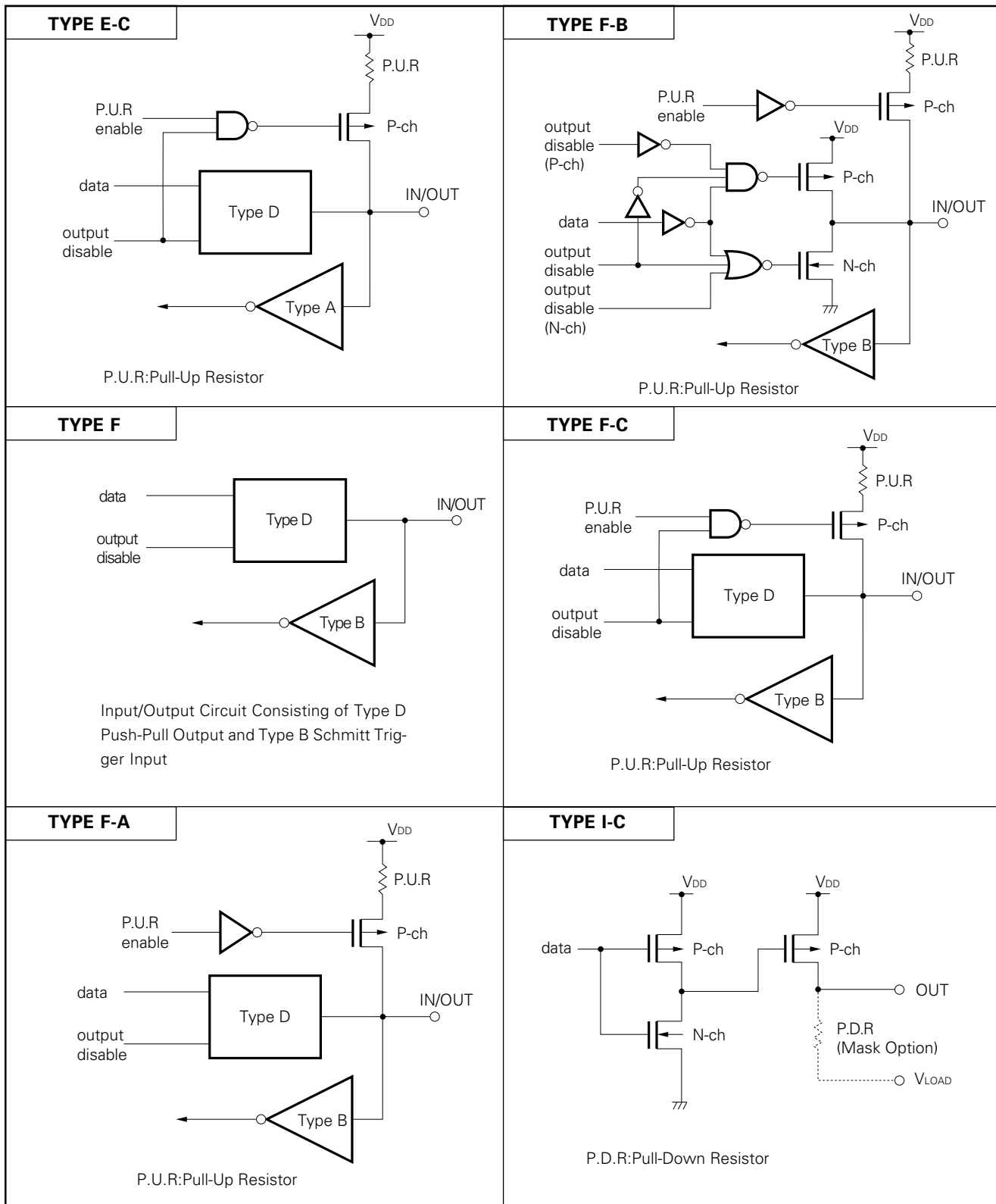
Pin Name	I/O	Dual-Function Pin	Function	After Reset	Input / Output Circuit Type *	
INT4	Input	P00	Edge-detected vectored interrupt input (valid for detection of rising and falling edges).	—	ⓑ	
INT0	Input	P10	Edge-detected vectored interrupt input (detected edge selection possible).	Clocked	ⓑ – C	
INT1		P11		Asynchronous		
INT2	Input	P12	Edge-detected testable input (rising edge detection).	Asynchronous	—	ⓑ – C
$\overline{\text{SCK1}}$	Input/output	P81	Serial clock input/output.	Input	Ⓕ	
SO1	Output	P82	Serial data output.	Input	E	
SI1	Input	P83	Serial data input.	Input	ⓑ	
AN0 to AN3	Input	—	Analog input to A/D converter.	—	Y	
AN4 to AN7		P90 to P93			Y – A	
AV <sub>DD</sub>	—	—	A/D converter power supply.	—	—	
AV <sub>REF</sub>	Input	—	A/D converter reference voltage input.	—	Z	
AV <sub>SS</sub>	—	—	A/D converter reference GND potential.	—	—	
X1, X2	Input	—	Main system clock oscillation crystal/ceramic connection. An external clock is input to X1 and an antiphase clock is input to X2.	—	—	
XT1	Input	—	Subsystem clock oscillation crystal connection. An external clock is input to XT1 and XT2 is made open.	—	—	
XT2	—					
$\overline{\text{RESET}}$	Input	—	System reset input.	—	ⓑ	
PPO	Output	P80	Timer/pulse generator pulse output.	Input	—	
V <sub>DD</sub> (3 – Pin)	—	—	Positive power supply.	—	—	
V <sub>SS</sub> (2 – Pin)	—	—	GND potential.	—	—	
V <sub>LOAD</sub>	—	—	FIP controller/driver pull-down resistor connect/power supply.	—	—	

\* Schmitt trigger inputs are circled.

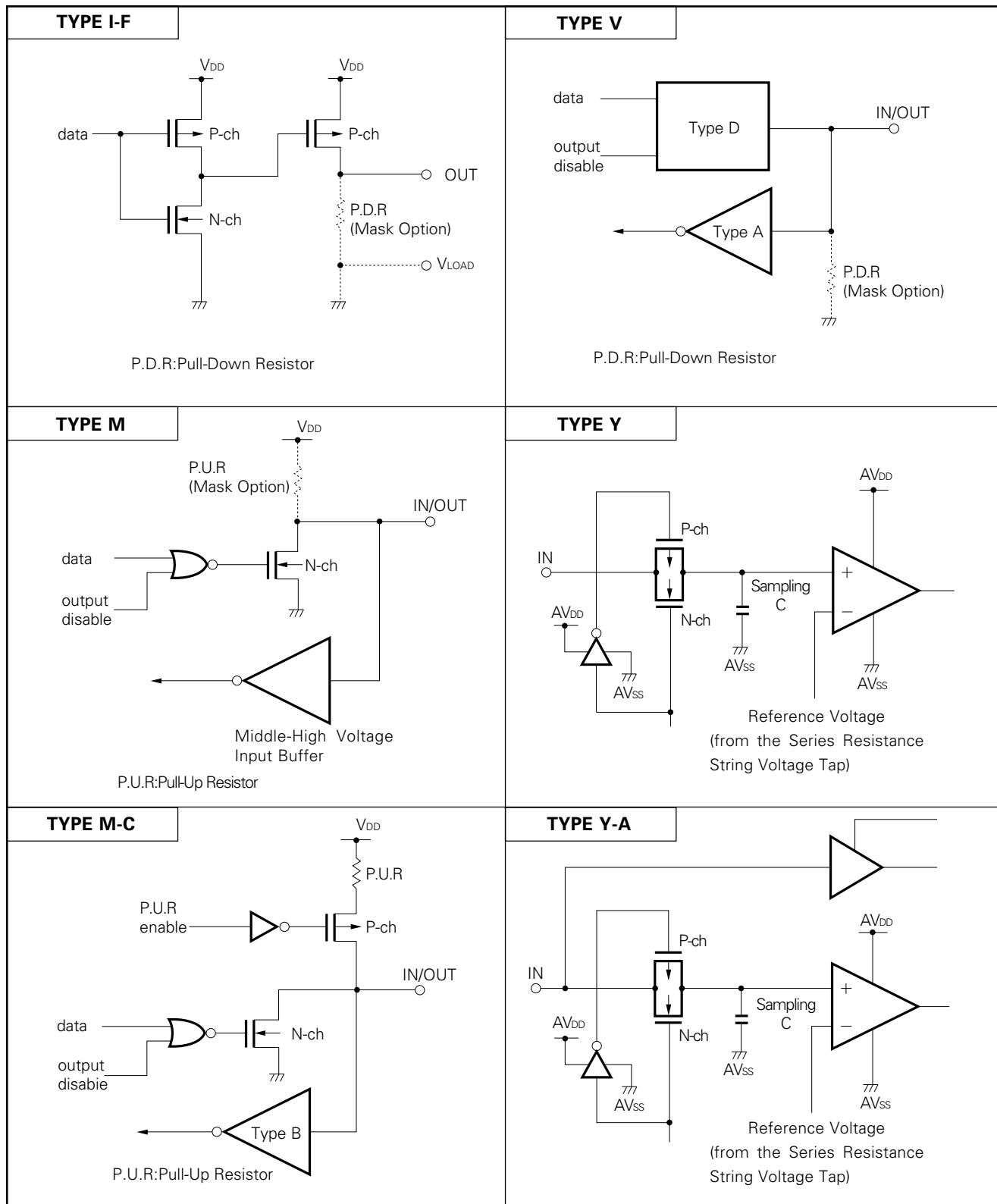
1.3 PIN INPUT/OUTPUT CIRCUIT LIST (1/4)

<p><b>TYPE A</b></p>  <p>CMOS-Specified Input Buffer</p>	<p><b>TYPE D</b></p>  <p>Push-Pull Output which can be Set to Output High Impedance (with Both P-ch and N-ch Set to OFF)</p>
<p><b>TYPE B</b></p>  <p>Schmitt Trigger Input Having Hysteresis Characteristics</p>	<p><b>TYPE E</b></p>  <p>Input/Output Circuit Consisting of Type D Push-Pull Output and Type A Input Buffer</p>
<p><b>TYPE B-C</b></p>  <p>P.U.R: Pull-Up Resistor</p> <p>Schmitt Trigger Input Having Hysteresis Characteristics</p>	<p><b>TYPE E-B</b></p>  <p>P.U.R: Pull-Up Resistor</p>

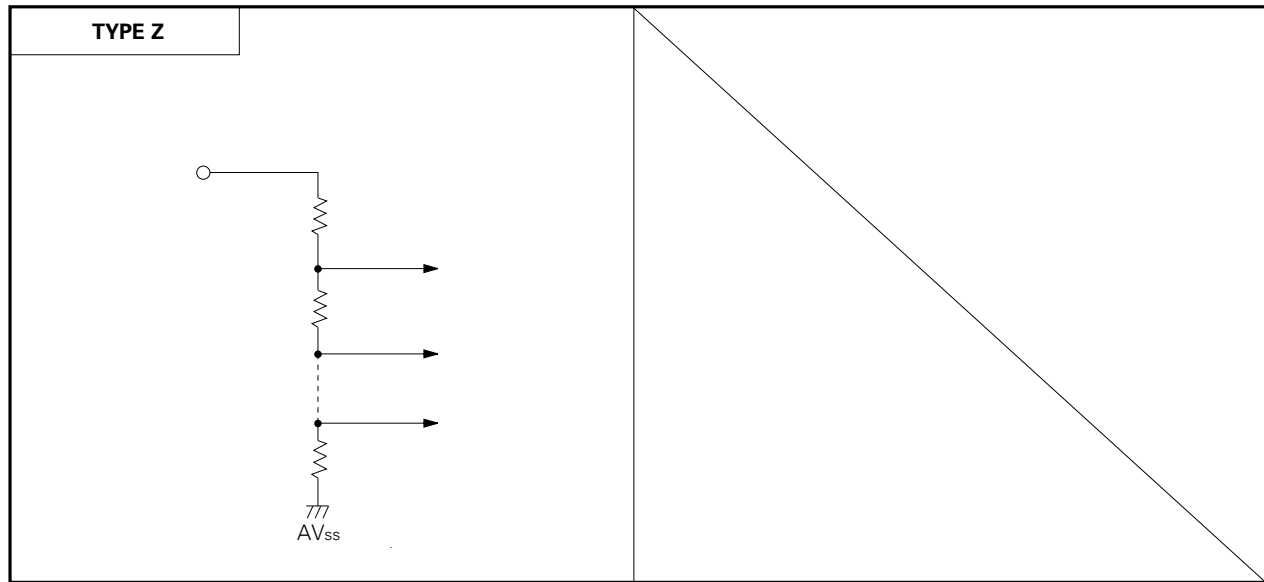
1.3 PIN INPUT/OUTPUT CIRCUIT LIST (2/4)



1.3 PIN INPUT/OUTPUT CIRCUIT LIST (3/4)



1.3 PIN INPUT/OUTPUT CIRCUIT LIST (4/4)



1.4 RECOMMENDED CONNECTIONS OF μPD75237 UNUSED PINS

Pin	Recommended Connection	
P00/INT4	Connect to V <sub>SS</sub>	
P01/SCK0	Connect to V <sub>SS</sub> or V <sub>DD</sub>	
P02/SO0/SB0		
P03/SI1/SB1		
P10/INT0 to P12/INT2	Connect to V <sub>SS</sub>	
P13/TI0		
P20/PTO0	Input state : Connect to V <sub>SS</sub> or V <sub>DD</sub>  Output state : Leave open	
P21		
P22/PCL		
P23/BUZ		
P30 to P33		
P40 to P43		
P50 to P53		
P60 to P63		
P70 to P73		
P80/PPO		Connect to V <sub>SS</sub>
P81/SCK1		
P82/SO1		
P83/SI1		
P90/AN4 to P93/AN7		
P100/S16 to P103/S19	Leave open	
P110/S20 to P113/S23		
P120 to P123		
P130 to P133		
P140 to P143		
P150 to P153		
AN0 to AN3	Connect to V <sub>SS</sub>	
AV <sub>REF</sub>		
AV <sub>DD</sub>	Connect to V <sub>DD</sub>	
AV <sub>SS</sub>	Connect to V <sub>SS</sub>	
XT1	Connect to V <sub>SS</sub> or V <sub>DD</sub>	
XT2	Leave open	
V <sub>LOAD</sub>	Connect to V <sub>SS</sub>	

**2. μPD75237 ARCHITECTURE AND MEMORY MAP**

The μPD75237 has the following three architectural features.

- (a) Data memory bank configuration
- (b) General register bank configuration
- (c) Memory mapped I/O

Each feature is outlined below.

**2.1 DATA MEMORY BANK CONFIGURATION AND ADDRESSING MODE**

As shown in Fig. 2-1, the μPD75237 incorporates a static RAM (928 words × 4 bits) at addresses 000H to 19FH and 200H to 3FFH in the data memory space and a display data memory (96 words × 4 bits) at addresses 1A0H to 1FFH and peripheral hardware (input/output ports, timers, etc.) at addresses F80H to FFFH. For addressing of this 12-bit address data memory space, the memory bank has a configuration wherein the lower 8 bits are directly or indirectly specified by an instruction and the higher 4-bit address is specified by a memory bank (MB).

A memory bank enable flag (MBE) and a memory bank select register (MBS) are incorporated to specify the memory bank (MB) and addressing operations shown in Fig. 2-1 and Table 2-1 can be carried out. (MBS is a register to select the memory bank and can set 0, 1, 2, 3 and 15. MBE is a flag to determine whether the memory bank selected by MBS should be validated or not. Since MBE is automatically saved/reset for interrupt or subroutine processing, it can be freely set for either processing.)

For data memory space addressing, set MBE = 1 normally and manipulate the memory bank static RAM specified by MBS. Efficient programming is possible by using the MBE = 0 or MBE = 1 mode for each program processing.

	Applicable Program Processing
MBE = 0 mode	<input type="radio"/> Interrupt service <input type="radio"/> Processing of repeating built-in hardware manipulation and static RAM manipulation <input type="radio"/> Subroutine processing
MBE = 1 mode	<input type="radio"/> Normal program processing



Fig. 2-1 Date Memory Configuration and Addressing Range in Each Addressing Mode

Addressing Mode	mem mem. bit		@HL @H+mem. bit		@DE @DL	Stack Addressing	fmem. bit	pmem. @L
	MBE = 0	MBE = 1	MBE = 0	MBE = 1	—	—	—	—
000H	Memory Bank Enable Flag							
01FH	↑ ↑ ↑ General Register Area							
020H	-----							
07FH	Data Area Static RAM (Memory Bank 0)		MBS = 0	MBS = 0		SBS = 0		
0FFH	↓							
100H	Data Area Static RAM (Memory Bank 1)		MBS = 1	MBS = 1		SBS = 1		
19FH	↓							
1A0H	↑ ↑ Display Data Memory Area							
1FFH	↓							
200H	Stack Area							
2FFH	Data Area Static RAM (Memory Bank 2)		MBS = 2	MBS = 2		SBS = 2		
300H	↓							
3FFH	Data Area Static RAM (Memory Bank 3)		MBS = 3	MBS = 3		SBS = 3		
-----								
	Not Incorporated							
F80H	↑							
FC0H	Peripheral Hardware Area (Memory Bank 15)		MBS = 15	MBS = 15				
FFFH	↓							

Remarks — : Don't care

**Table 2-1 Addressing Modes**

Addressing Mode	Identifier	Address Specified
1-bit direct addressing	mem.bit	Bit indicated by bit of address indicated by MB and mem, where: $\begin{matrix} \text{MBE} = 0 & \left\{ \begin{array}{l} \text{When mem} = 00\text{H to } 7\text{FH, MB} = 0 \\ \text{When mem} = 80\text{H to } \text{FFH, MB} = 15 \end{array} \right. \\ \text{MBE} = 1 & \text{MB} = \text{MBS} \end{matrix}$
4-bit direct addressing	mem	Address indicated by MB and mem, where : $\begin{matrix} \text{MBE} = 0 & \left\{ \begin{array}{l} \text{When mem} = 00\text{H to } 7\text{FH, MB} = 0 \\ \text{When mem} = 80\text{H to } \text{FFH, MB} = 15 \end{array} \right. \\ \text{MBE} = 1 & \text{MB} = \text{MBS} \end{matrix}$
8-bit direct addressing		Address indicated by MB and mem (mem is an even address), where: $\begin{matrix} \text{MBE} = 0 & \left\{ \begin{array}{l} \text{When mem} = 00\text{H to } 7\text{FH, MB} = 0 \\ \text{When mem} = 80\text{H to } \text{FFH, MB} = 15 \end{array} \right. \\ \text{MBE} = 1 & \text{MB} = \text{MBS} \end{matrix}$
4-bit register indirect addressing	@HL	Address indicated by MB and HL, where : $\text{MB} = \text{MBE} \cdot \text{MBS}$
	@HL+ @HL-	Address indicated by MB and HL, where : $\text{MB} = \text{MBE} \cdot \text{MBS}$ HL+ automatically increments L register after addressing. HL- automatically decrements L register after addressing.
	@DE	Address indicated by DE of memory bank 0
	@DL	Address indicated by DL of memory bank 0
8-bit register indirect addressing	@HL	Address indicated by MB and HL, where : $\text{MB} = \text{MBE} \cdot \text{MBS}$ Bit 0 of L register is ignored.
Bit manipulation addressing	fmem.bit	Bit indicated by bit of address indicated by fmem, where: $\text{fmem} = \left\{ \begin{array}{l} \text{FB0H to } \text{FBFH (interrupt-related hardware)} \\ \text{FF0H to } \text{FFFH (I/O port)} \end{array} \right.$
	pmem.@L	Bit indicated by the lower 2 bits of L register of the address indicated by the higher 10 bits of pmem and the higher 2 bits of L register, where: $\text{pmem} = \text{FC0H to } \text{FFFH}$
	@H + mem.bit	Bit indicated by bit of the address indicated by MB, H and the lower 4 bits of mem, where: $\text{MB} = \text{MBE} \cdot \text{MBS}$
Stack addressing		Address indicated by SP of memory banks 0, 1, 2 and 3 selected by SBS

As described in Table 2-1, direct and indirect addressing is possible for each of 1-bit, 4-bit and 8-bit data in μPD75237 data memory manipulation. Thus, easy-to-understand programs can be created very efficiently.

**2.2 GENERAL REGISTER BANK CONFIGURATION**

The μPD75237 incorporates four register banks, each bank consisting of eight general registers, X, A, B, C, D, E, H and L. This general register area is mapped at addresses 00H to 1FH of the memory bank 0 of the data memory (refer to **Fig. 2-2 General Register Configuration (4-Bit Processing)**). A register bank enable flag (RBE) and a register bank select register (RBS) are incorporated to specify the above general register banks. RBS is a register to select a register bank and RBE is a flag to determine whether the register bank selected by RBS should be validated or not. The register bank (RB) which is validated for instruction execution is given as

$$RB = RBE \cdot RBS.$$

As described above, with the μPD75237 having four register banks, programs can be created very efficiently by using different register banks for normal processing and interrupt service as described in Table 2-2. (RBE is automatically saved and set for interrupt service and automatically reset upon termination of the interrupt service.)

**Table 2-2 Recommended Use of Register Banks in Normal and Interrupt Routines**

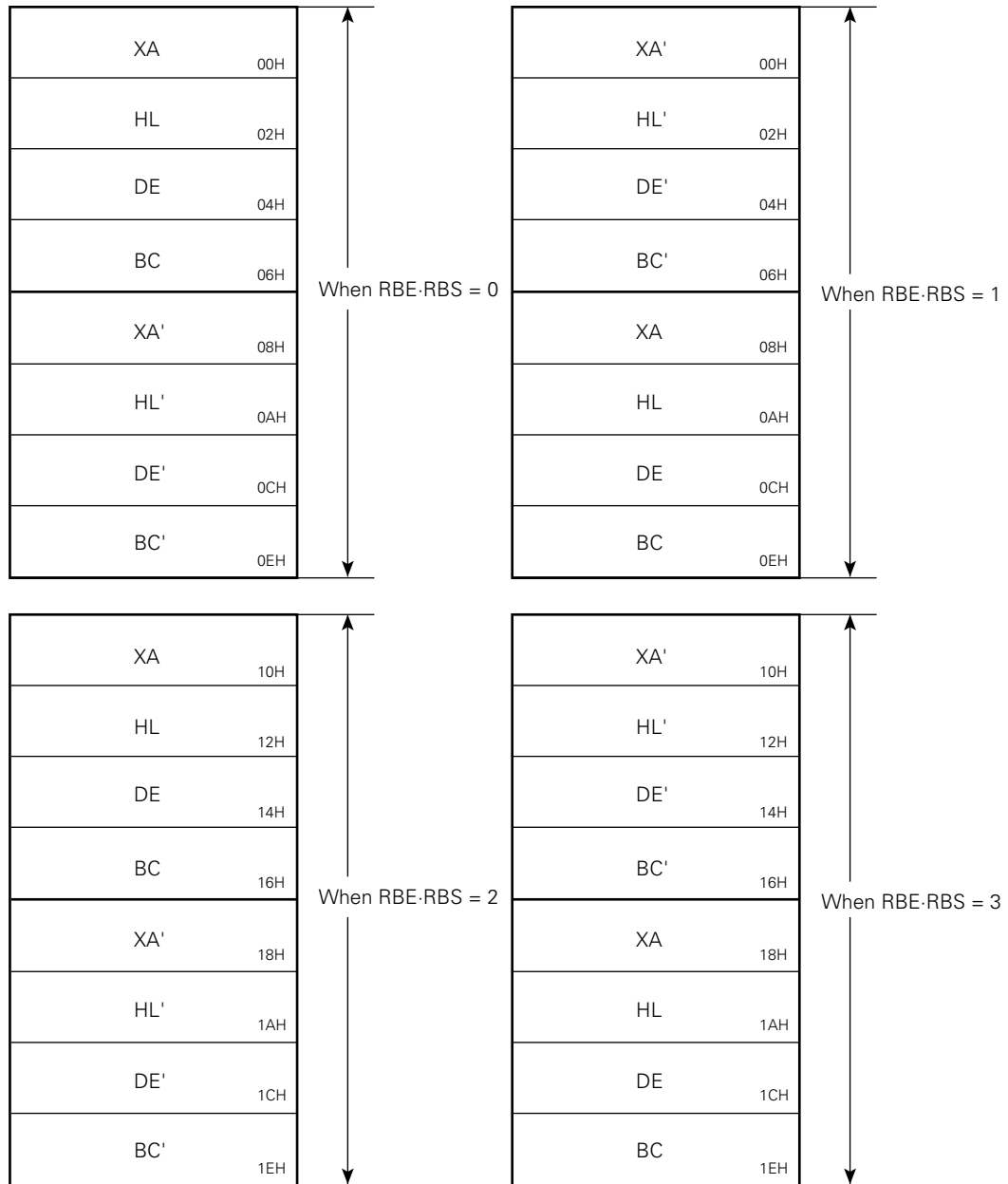
Normal processing	Use register banks 2 and 3 with RBE = 1.
Single interrupt service	Use register bank 0 with RBE = 0.
Double interrupt service	Use register bank 1 with RBE = 1. (It is necessary to save/reset RBS.)
Triple or more interrupt service	Save/reset registers by PUSH and POP.

Not only in 4-bit units, a register pair of XA, HL, DE or BC can transfer, compare, operate, increment or decrement data in 8-bit units. In this case, register pairs with the reversed bit 0 of the register bank specified by RBE•RBS can be specified as XA', HL', DE' and BC'. Thus, the μPD75237 has eight 8-bit registers (refer to **Fig. 2-3 General Register Configuration (8-Bit Processing)**).

Fig. 2-2 General Register Configuration (4-Bit Processing)

X	01H	A	00H	Register Bank 0 (RBE·RBS = 0)
H	03H	L	02H	
D	05H	E	04H	
B	07H	C	06H	
X	09H	A	08H	Register Bank 1 (RBE·RBS = 1)
H	0BH	L	0AH	
D	0DH	E	0CH	
B	0FH	C	0EH	
X	11H	A	10H	Register Bank 2 (RBE·RBS = 2)
H	13H	L	12H	
D	15H	E	14H	
B	17H	C	16H	
X	19H	A	18H	Register Bank 3 (RBE·RBS = 3)
H	1BH	L	1AH	
D	1DH	E	1CH	
B	1FH	C	1EH	

Fig. 2-3 General Register Configuration (8-Bit Processing)



2.3 MEMORY MAPPED I/O

As shown in Fig. 2-1, the μPD75237 employs the memory mapped I/O with the peripheral hardware including input/output ports and timers mapped at addresses F80H to FFFH in the data memory space. Thus, there are no special instructions to control the peripheral hardware and all operations are controlled by memory manipulation instructions. (Some hardware control mnemonics are available to make the program easy to understand.)

When operating the peripheral hardware, the addressing modes listed in Table 2-3 can be used.

Manipulate the display data memory, key scan register and port H mapped at addresses 1A0H to 1FFH by specifying memory bank 1.

**Table 2-3 Addressing Modes Applicable when Operating the Peripheral Hardware at Addresses F80H to FFFH**

	Applicable Addressing Mode	Applicable Hardware
Bit manipulation	Specify by direct addressing mem.bit with MBE = 0 or (MBE = 1, MBS = 15)	All hardware devices enabled for bit manipulation
	Specify by direct addressing fmem.bit irrespective of MBE and MBS	IST0, IST1, MBE, RBE, IExxx, IRQxxx, PORTn. 0 to 3
	Specify by indirect addressing pmem.@L irrespective of MBE and MBS	PORTn.
4-bit manipulation	Specify by direct addressing mem with MBE = 0 or (MBE = 1, MBS = 15)	All hardware devices enabled for 4-bit manipulation
	Specify by register indirect addressing @HL with (MBE = 1, MBS = 15)	
8-bit manipulation	Specify by direct addressing mem with MBE = 0 or (MBE = 1, MBS = 15) (mem is an even address.)	All hardware devices enabled for 8-bit manipulation
	Specify by register indirect addressing @HL with MBE = 1 and MBS = 15 (L register contents are even.)	

Table 2-4 shows the μPD75237 I/O map.

In the table, each item has the following meanings:

- Symbol ..... Name indicating the on-chip hardware address.  
Can be described in the instruction operand column.
- R/W ..... Indicates whether the corresponding hardware is enabled for read/write.  
R/W : Read/write enable  
R : Read only  
W : Write only
- No. of manipulatable bits ..... Indicates the number of applicable bits before operating the corresponding hardware.
- Bit manipulated addressing .... Indicates the applicable bit manipulated addressing before operating the applicable hardware.

Table 2-4 μPD75237 I/O Map (1/5)

Address	Hardware Name (Symbol)				R/W	No. of Manipulatable Bits			Bit Manipulated Addressing	Remarks
	b3	b2	b1	b0		1 Bit	4 Bits	8 Bits		
F80H	Stack pointer (SP)				R/W	—	—	○		Be sure to write 0 to bit 0.
F82H					Register bank select register (RBS)	R*1	—	○	○	
F83H	Memory bank select register (MBS)	—	○							
F84H	Stack bank select register (SBS)				R/W	—	○	—		Be sure to write 0 to bits 3 and 2.
F85H	Basic interval timer mode register (BTM)				W	△	○	—	mem.bit	Only bit 3 is bit-manipulatable.
F86H	Basic interval timer (BT)				R	—	—	○		
F88H					Display mode register (DSPM)	W	—	○	—	
F89H	Dimmer select register (DIMS)				W	—	○	—		
F8AH	KSF	Digit select register (DIGS)			R/W	△	○	—	mem.bit	Only bit 3 is bit-testable.
F90H	Timer pulse generator mode register (TPGM)				W	△	△	○	mem.bit	Only bit 3 is bit-manipulatable.
F94H					Timer pulse generator modulo register L (MODL)	R/W	—	—	○	
F96H	Timer pulse generator modulo register H (MODH)	R/W	—	—	○					
F98H	Watch mode register (WM)				W	—	—	○		

- \* 1. Can be read/written by the SEL instruction.
- 2. Individually manipulatable as RBS and MBS by 4-bit manipulation. Manipulatable as BS by 8-bit manipulation.

Table 2-4 μPD75237 I/O Map (2/5)

Address	Hardware Name (Symbol)				R/W	No. of Manipulatable Bits			Bit Manipulated Addressing	Remarks
	b3	b2	b1	b0		1 Bit	4 Bits	8 Bits		
FA0H	Timer/event counter 0 mode register (TM0)				W	△	—	○		Only bit3 is bit-manipulatable.
						—	—			
FA2H	TOE0				W	○	—	—		
FA4H	Timer/event counter 0 count register (T0)				R	—	—	○		
						—	—			
FA6H	Timer/event counter 0 modulo register (TMOD0)				W	—	—	○		
						—	—			
FA8H	Event counter mode register (TM1)				W	△	—	○		Only bit3 is bit-manipulatable.
						—	—			
FABH	Gate control register (GATEC)				W	—	○	—		
FACH	Counter register (T1)				R	—	—	○		
						—	—			



Table 2-4 μPD75237 I/O Map (3/5)

Address	Hardware Name (Symbol)				R/W	No. of Manipulatable Bits			Bit Manipulated Addressing	Remarks		
	b3	b2	b1	b0		1 Bit	4 Bits	8 Bits				
FB0H	IST1	IST0	MBE	RBE	R/W	○	○	○	fmem.bit			
	Program status word (PSW)					—	—					
FB2H	Interrupt priority select register (IPS)				W	○	○	—			★	
FB3H	Processor clock control register (PCC)				W	○	○					
FB4H	INT0 mode register (IM0)				W	—	○	—			Be sure to write 0 to bit 2.	
FB5H	INT1 mode register (IM1)				W	—	○				Be sure to write 0 to bits 3, 2 and 1.	
FB7H	System clock control register (SCC)				W	○	—	—			Only bits 3 and 0 are bit-manipulatable.	★
FB8H	IE4	IRQ4	IEBT	IRQBT	R/W	○	○	fmem.bit				
FB9H	/	/	/	EOT	R/W	○	○					
FBAH	/	/	IEW	IRQW	R/W	○	○					
FBBH	IEKS	IRQKS	IETPG	IRQTPG	R/W	○	○					
FBCH	/	IRQT1	IET0	IRQT0	R/W	○	○					
FBDH	/	/	IECSI0	IRQCSI0	R/W	○	○					
FBEH	IE1	IRQ1	IE0	IRQ0	R/W	○	○					
FBFH	/	/	IE2	IRQ2	R/W	○	○					
FC0H	Bit sequential buffer 0 (BSB0)				R/W	○	○		○			
FC1H	Bit sequential buffer 1 (BSB1)				R/W	○	○					
FC2H	Bit sequential buffer 2 (BSB2)				R/W	○	○					
FC3H	Bit sequential buffer 3 (BSB3)				R/W	○	○	○				
FC8H	/	/	CSIM11	CSIM10	W	—	—					
FC9H	CSIE1	/	/	/	W	○	—	○	★			
FCCH	Serial I/O shift register (SIO1)				R/W	—	—	○				

Table 2-4 μPD75237 I/O Map (4/5)

Address	Hardware Name (Symbol)				R/W	No. of Manipulatable Bits			Bit Manipulated Addressing	Remarks
	b3	b2	b1	b0		1 Bit	4 Bits	8 Bits		
FD0H	Clock output mode register (CLOM)				W	—	○	—		
FD4H	Static mode register B (STATB)				W	—	—	○		
FD6H	Static mode register A (STATA)				W	—	—	○		
★ FD8H	SOC	EOC			R/W	△	—	○		Write only in 8-bit manipulation
	A/D conversion mode register (ADM)					—	—			
FDAH	SA register (SA)				R	—	—	○		
FDCH	Pull-up register specification register group A (POGA)				W	—	—	○		
FE0H	Serial operating mode register (CSIM0)				W	—	—	○	mem.bit	Write only in 8-bit manipulation
	CSIE0	COI	WUP		R/W	○	○			
FE2H	CMDD	RELD	CMDT	RELT	R/W	○	—	—	mem.bit	
	SBI control register (SBIC)									
	BSYE	ACKD	ACKE	ACKT						
FE4H	Serial I/O shift register 0 (SIO0)				R/W	—	—	○		
FE6H	Slave address register (SVA)				W	—	—	○		
FE8H	PM33	PM32	PM31	PM30	W	—	—	○		
	Port mode register group A (PMGA)									
	PM63	PM62	PM61	PM60						
FECH	—	PM2	—	—	W	—	—	○		
	Port mode register group B (PMGB)									
	PM7	—	PM5	PM4						

Table 2-4 μPD75237 I/O Map (5/5)

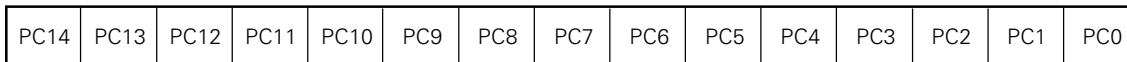
Address	Hardware Name (Symbol)				R/W	No. of Manipulatable Bits			Bit Manipulated Addressing	Remarks
	b3	b2	b1	b0		1 Bit	4 Bits	8 Bits		
FF0H	Port 0 (PORT0)				R	○	○	—	fmem.bit pmem.@L	
FF1H	Port 1 (PORT1)				R	○	○			
FF2H	Port 2 (PORT2)				R/W	○	○	—		
FF3H	Port 3 (PORT3)				R/W	○	○			
FF4H	Port 4 (PORT4)				R/W	○	○	○		
FF5H	Port 5 (PORT5)				R/W	○	○			
FF6H	Port 6 (PORT6)				R/W	○	○	○		
FF7H	Port 7 (PORT7)				R/W	○	○			
FF8H	Port 8 (PORT8)				R	○	○	—		
FF9H	Port 9 (PORT9)				R	○	○			
FFAH	Port 10 (PORT10)				W	○	○	○		
FFBH	Port 11 (PORT11)				W	○	○			
FFCH	Port 12 (PORT12)				W	○	○	○		
FFDH	Port 13 (PORT13)				W	○	○			
FFEH	Port 14 (PORT14)				W	○	○	○		
FFFH	Port 15 (PORT15)				W	○	○			
1A0H+4n	Display data memory: S16 to S23 (n = 0 to 15)				R/W	○	○	○	mem.bit	
1A1H+4n					R/W	○	○			
1BEH	Key scan register (KS2)				R/W	○	○	○		
1BFH					R/W	○	○			
1C0H+4n	Display data memory: S0 to S7 (n = 0 to 15)				R/W	○	○	○		
1C1H+4n					R/W	○	○			
1C2H+4n	Display data memory: S8 to S15 (n = 0 to 15)				R/W	○	○	○		
1C3H+4n					R/W	○	○			
1FCH	Key scan register (KS0)				R/W	○	○	○		
1FDH					R/W	○	○			
1FEH	Key scan register (KS1)				R/W	○	○	○		
1FFH	Port H (PORTH)				R/W	○	○			

### 3. INTERNAL CPU FUNCTIONS

#### 3.1 PROGRAM COUNTER (PC): 15 BITS

This is a 15-bit binary counter to hold the program memory address information.

Fig. 3-1 Program Counter Configuration



When  $\overline{\text{RESET}}$  is input, the lower 6 bits at address 0000H and the contents at address 0001H of the program memory are set to PC13 to PC8 and PC7 to PC0, respectively, and the PC is initialized. The reset start address should therefore be located in the 16K space (0000H to 3FFFH).

#### 3.2 PROGRAM MEMORY (ROM): 24448 WORDS × 8 BITS

This is a mask programmable ROM having a configuration of 24448 words × 8 bits to store programs, table data, etc.

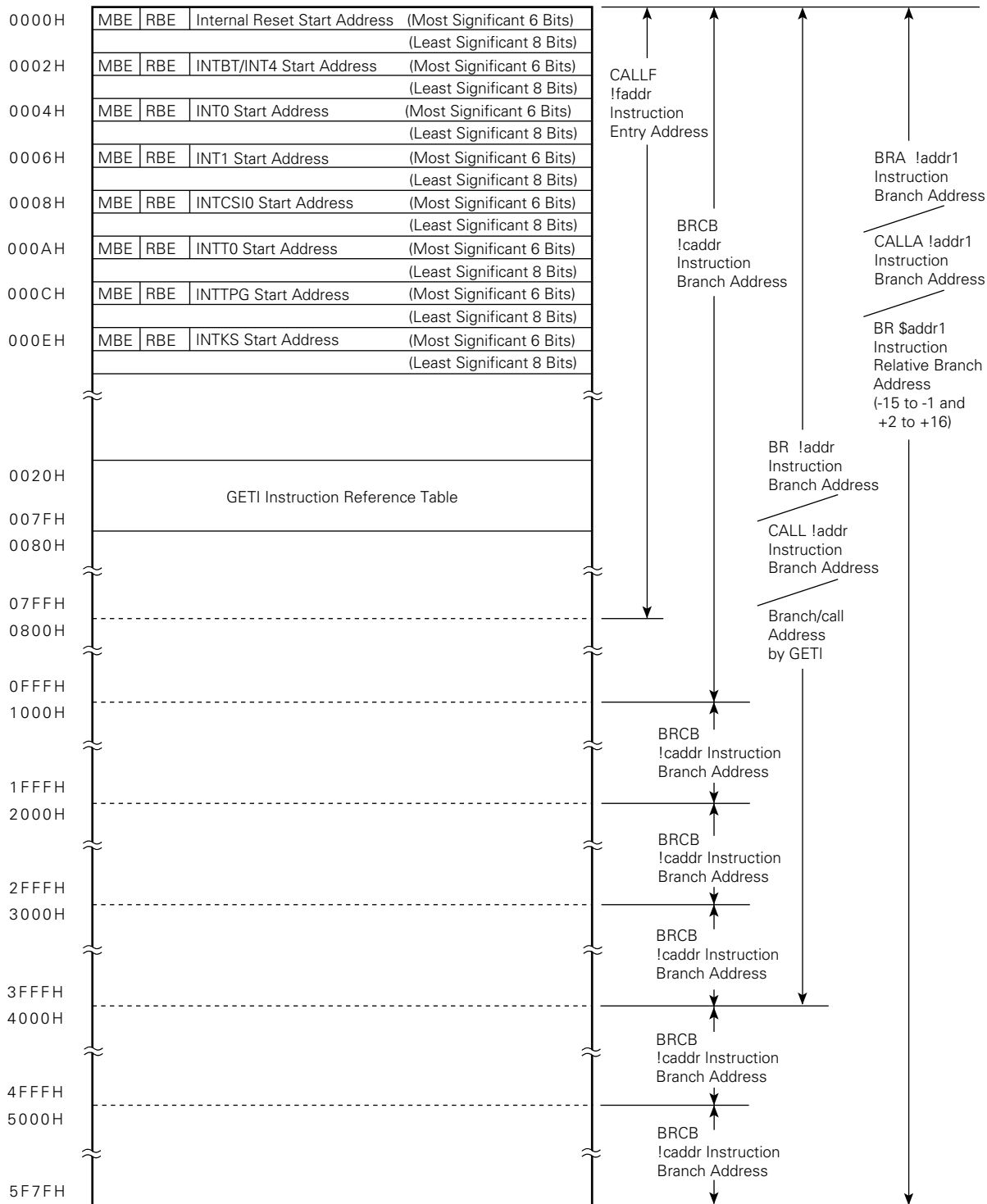
The program memory is addressed by the program counter. Table data can be referred to by the table reference instruction (MOVT).

The branch range enabled by the branch and subroutine call instructions is shown in Fig. 3-2. The entire space branch instruction (BRA !addr1) and the entire space call instruction (CALLA !addr1) allow direct branching to the entire space from 0000H to 5F7FH. The relative branch instruction (BR \$addr) enables branch to the [PC contents -15 to -1, +2 to +16] address irrespective of the block boundary.

The program memory addresses are 0000H to 5F7FH and the following addresses are especially assigned. (All areas except 0000H and 0001H can be used as the normal program memory.)

- Addresses 0000 and 0001H  
Vector address table for writing the program start address to be set upon  $\overline{\text{RESET}}$  input and the RBE and MBE set values. Can be reset and started at any address in a 16K space (0000H to 3FFFH).
  - Addresses 0002 to 000FH  
Vector address table for writing the program start address to be set by each vectored interrupt and the RBE and MBE set values. Interrupt service can be started at any address in a 16K space (0000H to 3FFFH).
  - Addresses 0020 to 007FH  
Table area to be referred to by GETI instruction\*.
- \* GETI instruction is an instruction to realize any 2-byte/3-byte instruction or two 1-byte instructions with one byte. It is used to decrease the number of program bytes. (Refer to **8.1 CHARACTERISTIC INSTRUCTIONS OF μPD75237.**)

Fig. 3-2 Program Memory Map



**Note** As stated above, the interrupt vector start address is 14 bits in length, and should therefore be set in the 16K space (0000H to 3FFFH).

**Remarks** In all cases other than those listed above, branch to the address with only the lower 8 bits of the PC changed is enabled by BR PCDE and BR PCXA instructions.

### 3.3 DATA MEMORY

The data memory consists of a static RAM and peripheral hardware.

The static RAM incorporates 160 words × 4 bits of memory banks 0, 2 and 3, 768 words × 4 bits of memory bank 1 and 96 words × 4 bits of memory bank 1 which also serves as a display data memory. It is used to store process data and to serve as a stack memory for interrupt execution.

General registers, display data memory and various registers of peripheral hardware are mapped at particular addresses of the data memory and such data is manipulated by the general register and memory manipulation instructions. (Refer to Fig. 2-1 Data Memory Configuration and Addressing Range in Each Addressing Mode.)

All addresses (000H to 3FFH) of memory banks 0, 1, 2 and 3 can be used as a stack area.

Although the data memory consists of one address and 4 bits, it can be manipulated in 8-bit units by the 8-bit memory manipulation instruction or in bit units by the bit manipulation instruction. Specify an even address by the 8-bit manipulation instruction.

The display data memory area (1A0H to 1FFH) is made up as shown in Fig. 3-4.

Fig. 3-3 Data Memory Map

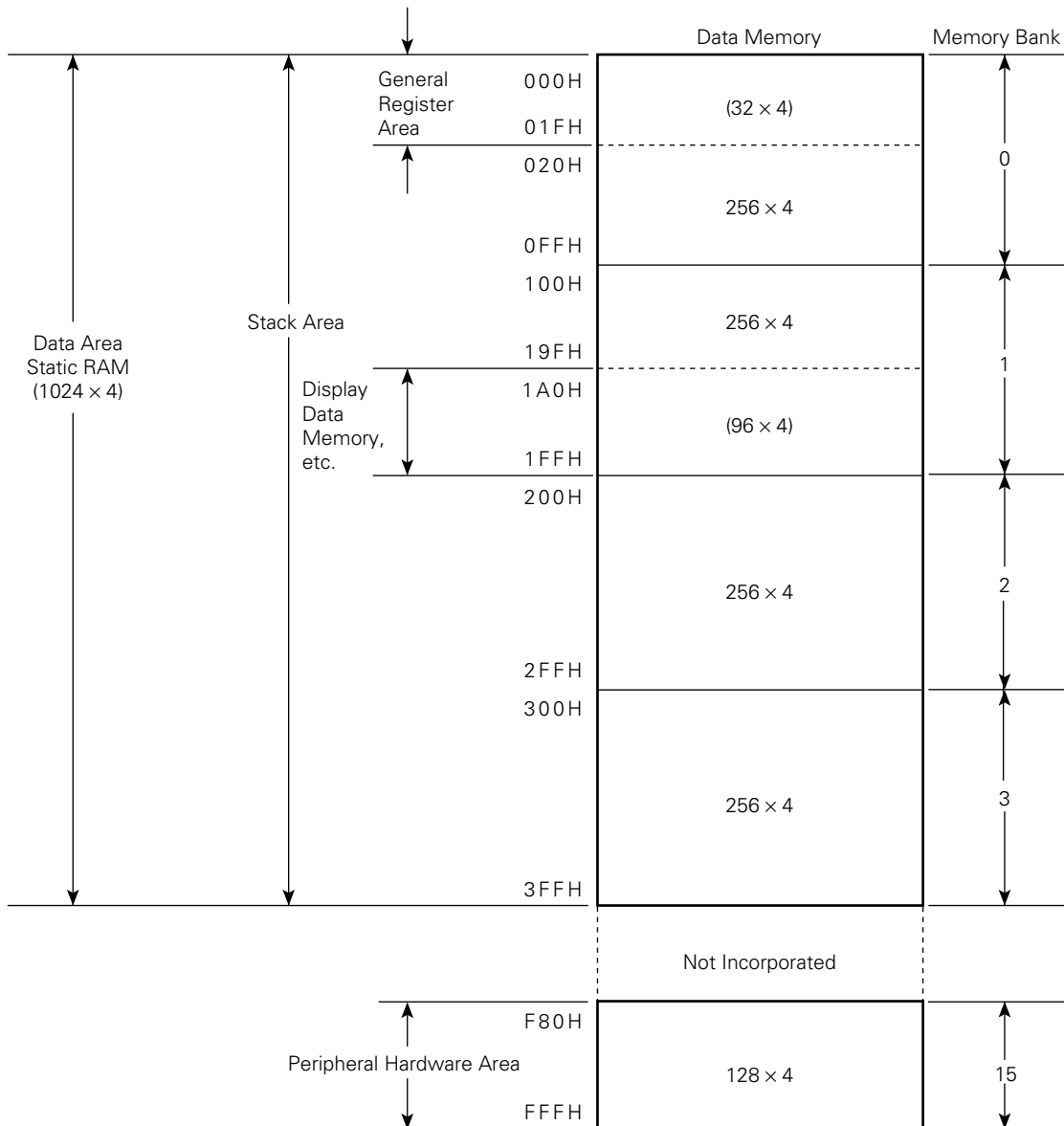


Fig. 3-4 Display Data Memory Configuration

		1 A 1 H	1 A 0 H	1 C 3 H	1 C 2 H	1 C 1 H	1 C 0 H
		1 A 3 H	1 A 2 H	1 C 7 H	1 C 6 H	1 C 5 H	1 C 4 H
		1 A 5 H	1 A 4 H	1 C B H	1 C A H	1 C 9 H	1 C 8 H
		1 A 7 H	1 A 6 H	1 C F H	1 C E H	1 C D H	1 C C H
		1 A 9 H	1 A 8 H	1 D 3 H	1 D 2 H	1 D 1 H	1 D 0 H
		1 A B H	1 A A H	1 D 7 H	1 D 6 H	1 D 5 H	1 D 4 H
		1 A D H	1 A C H	1 D B H	1 D A H	1 D 9 H	1 D 8 H
		1 A F H	1 A E H	1 D F H	1 D E H	1 D D H	1 D C H
		1 B 1 H	1 B 0 H	1 E 3 H	1 E 2 H	1 E 1 H	1 E 0 H
		1 B 3 H	1 B 2 H	1 E 7 H	1 E 6 H	1 E 5 H	1 E 4 H
		1 B 5 H	1 B 4 H	1 E B H	1 E A H	1 E 9 H	1 E 8 H
		1 B 7 H	1 B 6 H	1 E F H	1 E E H	1 E D H	1 E C H
		1 B 9 H	1 B 8 H	1 F 3 H	1 F 2 H	1 F 1 H	1 F 0 H
		1 B B H	1 B A H	1 F 7 H	1 F 6 H	1 F 5 H	1 F 4 H
		1 B D H	1 B C H	1 F B H	1 F A H	1 F 9 H	1 F 8 H
		1 B F H	1 B E H (KS2)	1 F H (PORTH)	1 F E H (KS1)	1 F D H	1 F C H (KS0)
No. of manipulatable bits	1 bit	○	○	○	○	○	○
	4 bits	○	○	○	○	○	○
	8 bits	○		○		○	

- Remarks**
1. KS0, KS1 and KS2: Key scan register
  2. PORTH: High-voltage, high-current output port which also serves as digit output port

**3.4 GENERAL REGISTER: 8 × 4 BITS × 4 BANKS**

The general registers are mapped at the special addresses of the data memory. There are 4-bank registers, each bank consisting of eight 4-bit registers (B, C, D, E, H, L, X, A).

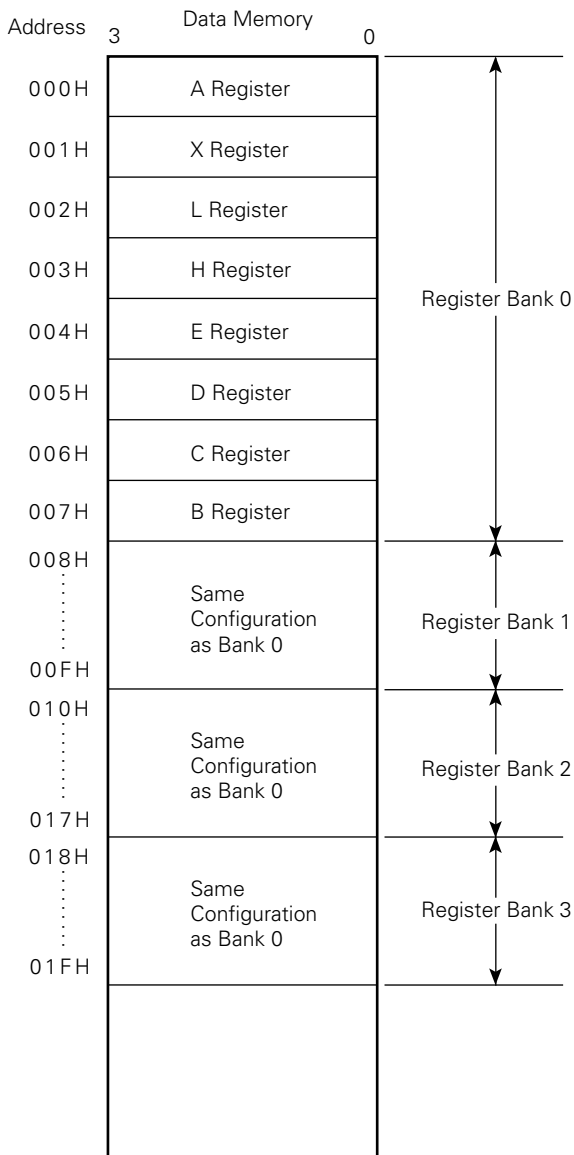
The register bank (RB) which becomes valid for instruction is given as

$$RB = RBE \cdot RBS \quad (RBS = 0 \text{ to } 3).$$

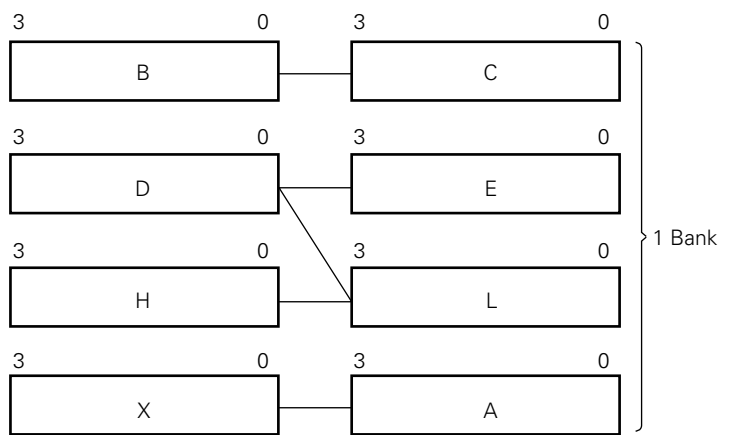
Each general register is operated in 4-bit units. BC, DE, HL and XA form register pairs and are used for 8-bit manipulation. In addition to DE and HL, DL also makes up a pair and these three pairs can be used as a data pointer.

The general register area can be accessed by address specification as a normal RAM whether or not it is used as a register.

**Fig. 3-5 General Register Configuraton**



**Fig. 3-6 Register Pair Configuration**

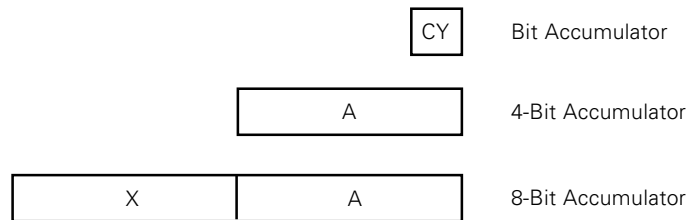




### 3.5 ACCUMULATOR

In the μPD75237, A register and XA register pair function as an accumulator. The 4-bit data processing instruction is executed mainly by A register and the 8-bit data processing instruction is executed mainly by XA register pair. For execution of the bit manipulation instruction, the carry flag (CY) functions as a bit accumulator.

**Fig. 3-7 Accumulator**



### 3.6 STACK POINTER (SP) AND STACK BANK SELECT REGISTER (SBS)

In the μPD75237, the static RAM is used as a static memory (LIFO type) and the 8-bit register which holds the start address information in the stack area is a stack pointer (SP).

The stack area is located at addresses 000H to 3FFH of memory banks 0, 1, 2 and 3. Specify one memory bank by a 4-bit SBS.

The SP is decremented prior to a write (save) to the stack memory and incremented after a read (restore) from the stack memory. Set SBS by the 4-bit memory manipulation instruction. In this case, set the higher 2-bits to 00.

The data to be saved/restored by each stack operation is shown in Figs. 3-9 and 3-10.

The SP initial value is set by the 8-bit memory manipulation instruction and the SBS initial value is set by the 4-bit memory manipulation instruction and then the stack area is determined. The SP and SBS contents can also be read.

**Table 3-1 Stack Areas to be Selected by SBS**

SBS		Stack Area
SBS1	SBS0	
0	0	Memory bank 0
0	1	Memory bank 1
1	0	Memory bank 2
1	1	Memory bank 3

When the SP initial value is set to 00H, stack starts with the most significant address (nFFH) of the memory bank (n: n = 0, 1, 2, 3) specified by SBS.

The stack area is limited to the memory bank specified by SBS. When stack operation is further carried out at address n00H, the address is reset to nFFH in the same bank. Linear stack past the memory bank boundary is not possible without rewriting SBS.

Since RESET input makes the SP and SBS undefined, be sure to initialize the SP and SBS to any desired value at the beginning of the program.

**Fig. 3-8 Stack Bank Select Register Configuration**

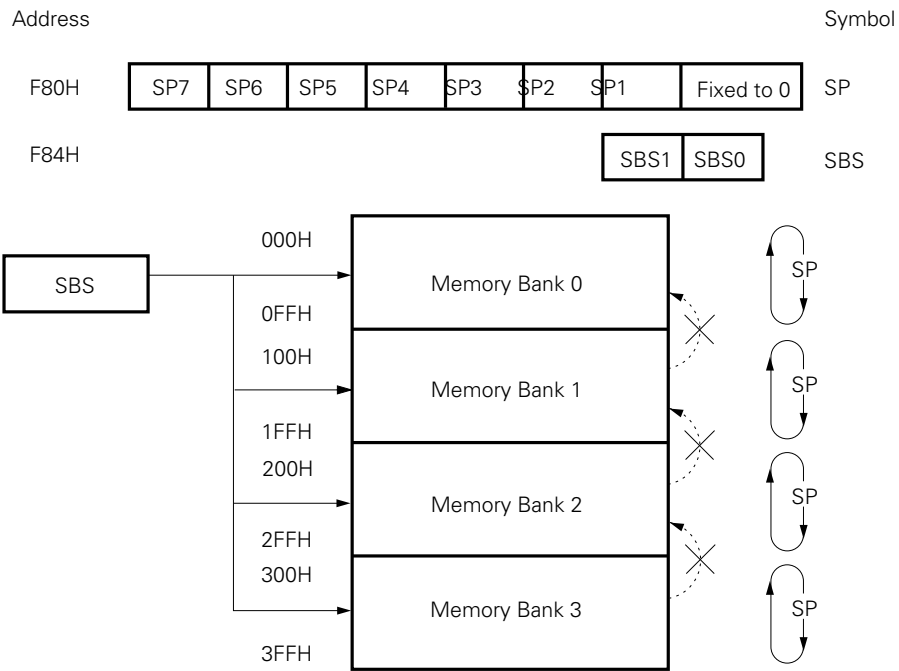


Fig. 3-9 Data to be Saved into Stack Memory

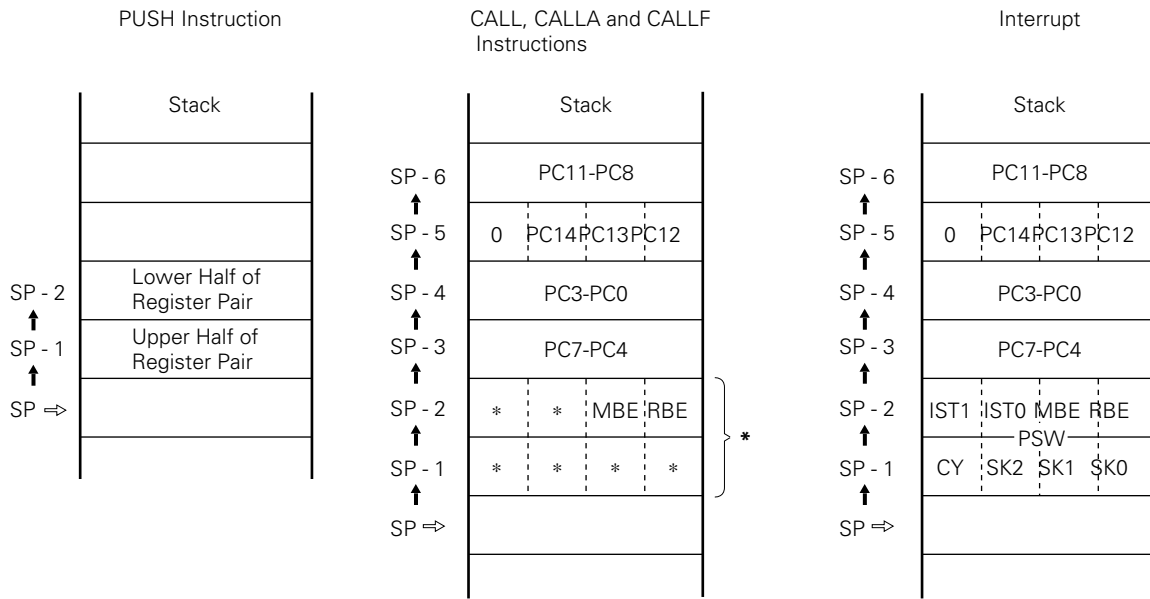
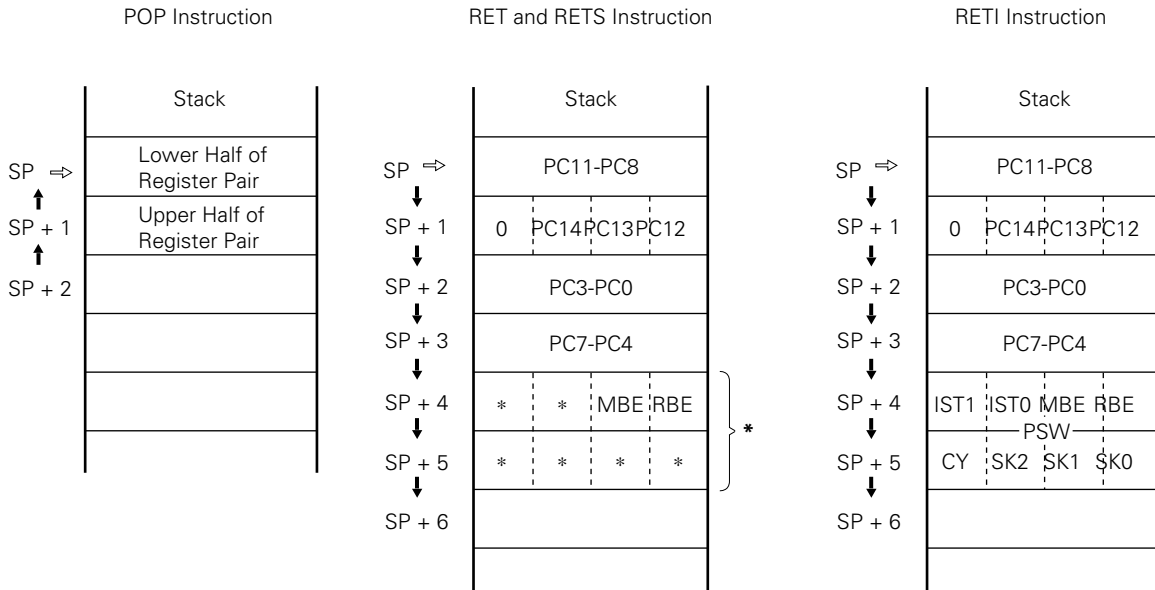


Fig. 3-10 Data to be Restored from Stack Memory



\* PSW except MBE and RBE are not saved/restored.

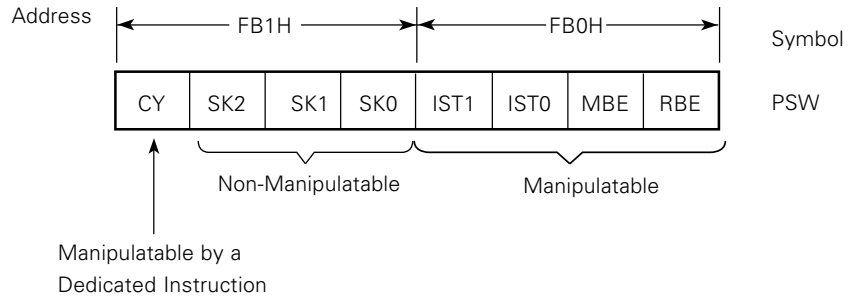
Remarks \* means undefined.

### 3.7 PROGRAM STATUS WORD (PSW): 8 BITS

The program status word (PSW) consists of various types of flags closely related to processor operation.

The PSW is mapped at addresses FB0H and FB1H in the data memory space and 4 bits at address FB0H can be operated by the memory manipulation instruction. Normal data memory manipulation instructions cannot be used at address FB1H.

**Fig. 3-11 Program Status Word Configuration**



**Table 3-2 PSW Flag to be Saved/Restored in Stack Operation**

		Flag to be Saved/Restored
Save	During CALL/CALLA/CALLF instruction execution	MBE and RBE saved
	Upon hardware interruption	All PSW bits saved
Restore	During RET/RETS instruction execution	MBE and RBE restored
	During RETI instruction execution	All PSW bits restored

**(1) Carry flag (CY)**

The carry flag is a 1-bit flag to store the overflow and underflow generate information when a carry operation instruction (ADDC, SUBC) is executed.

It has the bit accumulator function to execute Boolean algebraic operations with the data memory specified by the bit address and to store the result.

Carry flag manipulation is carried out using a dedicated instruction irrespective of other PSW bits. When  $\overline{\text{RESET}}$  signal is generated, the carry flag becomes undefined.

**Table 3-3 Carry Flag Manipulation Instructions**

	Instruction (Mnemonic)	Carry Flag Operation and Processing
Carry flag manipulation dedicated instruction	SET1 CY CLR1 CY NOT1 CY SKT CY	CY set (1) CY clear (0) CY contents invert SKip if CY contents are 1
Bit transfer instruction	MOV1 mem* .bit CY MOV1 CY, mem* .bit	CY contents transfer to the specified bit Specified bit contents transfer to CY
Bit Boolean instruction	AND1 CY, mem* .bit OR1 CY, mem* .bit XOR1 CY, mem* .bit	Specified bit contents ANDed/ORed/XORed with CY contents and the results set to CY
Interrupt service	During interrupt execution	Parallel save of other PSW bits and 8 bits to the stack memory
	RETI	Restore from the stack memory in parallel to other PSW bits

**Remarks** mem\*.bit indicates the following three bit manipulated addressing operations.

- fmem.bit
- pmem.@L
- @H + mem.bit

**(2) Skip flags (SK2, SK1, SK0)**

The skip flag is used to store the skipped state and is automatically set/reset when the CPU executes an instruction.

The user cannot directly operate the skip flags as operands.

**(3) Interrupt status flags (IST1, IST0)**

The interrupt status flag is a 2-bit flag to store the status of the processing currently being executed. (Refer to **Table 5-3 IST1 and IST0 Interrupt Servicing Statuses** for details.)

**Table 3-4 Interrupt Status Flag Directive Contents**

IST1	IST0	Status of Processing being Executed	Servicing Contents and Interrupt Control
0	0	Status 0	Normal program being executed. All interrupts acknowledgeable.
0	1	Status 1	Low or high interrupt being executed. Only high interrupt acknowledgeable.
1	0	Status 2	High interrupt being executed. All interrupts non-acknowledgeable.
1	1	—	Setting disable

The interrupt priority control circuit (see **Fig. 5-1 Interrupt Control Circuit Block Diagram**) identifies the interrupt status flag contents and executes multiple interrupt control.

If the interrupt is acknowledged, the IST1 and IST0 contents are saved to the stack memory as part of PSW and are automatically changed to the status higher by one level and the values prior to interruption by RETI instruction are restored.

The interrupt status flag can be operated by the memory manipulation instruction and the processing status being executed can be changed by program control.

**Note** Before operating this flag, be sure to disable interruption by executing DI instruction and enable interruption by execution EI instruction after operation.

**(4) Memory bank enable flag (MBE)**

This is a 1-bit flag to specify the mode to generate the address information of the most significant 4 bits of the 12 bits of the data memory address.

When this flag is set (1), the data memory address space is expanded and all data memory spaces become addressable.

When this flag is reset (0), the data memory address space is fixed irrespectively of MBS setting. (See **Fig. 2-1 Data Memory Configuration and Addressing Range in Each Addressing Mode.**)

When  $\overline{\text{RESET}}$  input is applied, the bit 7 contents at address 0 of the program memory are set and the MBE is automatically initialized.

In vectored interrupt service, the bit 7 contents of the corresponding vector address table are set and the MBE status in the interrupt service is automatically set.

Normally, set MBE = 0 for interrupt service and use the static RAM of memory bank 0.

**(5) Register bank enable flag (RBE)**

This is a 1-bit flag to determine whether or not the general register bank configuration should be expanded.

When this flag is set (1), one general register can be selected from register banks 0 to 3 depending on the register bank select register (RBS) contents.

When this flag is reset (0), register bank 0 is selected as a general register irrespectively of the register bank select register (RBS) contents.

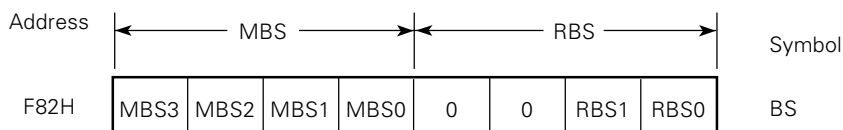
Upon  $\overline{\text{RESET}}$  input, the bit 6 contents at address 0 of the program memory are set and the flag is automatically initialized.

When a vectored interrupt is generated, the bit 6 contents of the corresponding vector address table are set and the RBE status in interrupt service is automatically set. Normally, set RBE = 0 for interrupt service. Use register bank 0 for 4-bit operation and register banks 0 and 1 for 8-bit operation.

### 3.8 BANK SELECT REGISTER (BS)

The bank select register (BS) consists of a register bank select register (RBS) and a memory bank select register (MBS). The RBS and MBS are used to specify the register bank and the memory bank to be used, respectively. The RBS and MBS are set by SEL RBn and SEL MBn instructions, respectively. The BS can be saved/restored the stack area in 8-bit units by PUSH BS/POP BS instruction.

**Fig. 3-12 Bank Select Register Configuration**



#### (1) Memory bank select register (MBS)

The memory bank select register is a 4-bit register to store the most significant 4-bit address information of the data memory address (12 bits) and the memory bank to be accessed is specified by the MBS contents. Banks 0, 1, 2, 3 and 15 can be specified.

The MBS is set by SEL MBn instruction. (n = 0, 1, 2, 3, 15)

The address range for MBE and MBS setting is shown in Fig. 2-1.

Upon RESET input, the MBS is initialized to "0".

#### (2) Register bank select register (RBS)

The register bank select register is used to specify the register bank for use as a general register and can set banks 0 to 3.

The RBS is set by SEL RBn instruction. (n = 0 to 3)

Upon RESET input, the RBS is initialized to "0".

**Table 3-5 RBE, RBS and Register Banks to be Selected**

RBE	RBS				Register Bank
	3	2	1	0	
0	0	0	×	×	Fixed to bank 0
1	0	0	0	0	Bank 0 selected
			0	1	Bank 1 selected
			1	0	Bank 2 selected
			1	1	Bank 3 selected

↑ ↑  
Fixed to 0

**Remarks** × : Don't care



4. PERIPHERAL HARDWARE FUNCTIONS

4.1 DIGITAL INPUT/OUTPUT PORTS

The μPD75237 employs the memory mapped I/O and all input/output ports are mapped in the data memory space.

Fig. 4-1 Digital Port Data Memory Address

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

**(1) Digital input/output port configuration**

The digital input/output port configurations are shown in Figs. 4-2 to 4-11.

**(2) Input/output mode setting**

The input/output mode of each input/output port is set by the port mode register as shown in Fig. 4-12.

Each port acts as an input when the corresponding port mode register bit is "0" and as an output port when the bit is "1".

Port mode register groups A and B each are set by the 8-bit memory manipulation instruction.

Upon RESET input, all bits of each port mode register are cleared to "0". Thus, the output buffer is turned OFF and all the ports are set to the input mode.

**(3) Digital input/output port operation**

The operations of the port and pin for instruction execution vary, depending on the input/output mode setting as shown in Table 4-1.

**Table 4-1 Input/Output Port Operations for Input/Output Instruction Execution**

	Input Mode (Corresponding Bit 0 of Mode Register) [Output Buffer OFF]	Output Mode (Corresponding Bit 1 of Mode Register) [Output Buffer ON]
When 1-bit test instruction, 1-bit input instruction, 4-bit or 8-bit instruction is executed	Each pin data input	Output latch contents input
When 4-bit or 8-bit output instruction is executed	Accumulator data transfer to output latch	Accumulator data output to output pin
When 1-bit output instruction* is executed	Output latch contents become undefined	Output pin status change according to instruction

\* SET1/CLR1/MOV1 PORTn.bit, CY, etc.

Fig. 4-2 Configurations of Port 0, 1 and 8

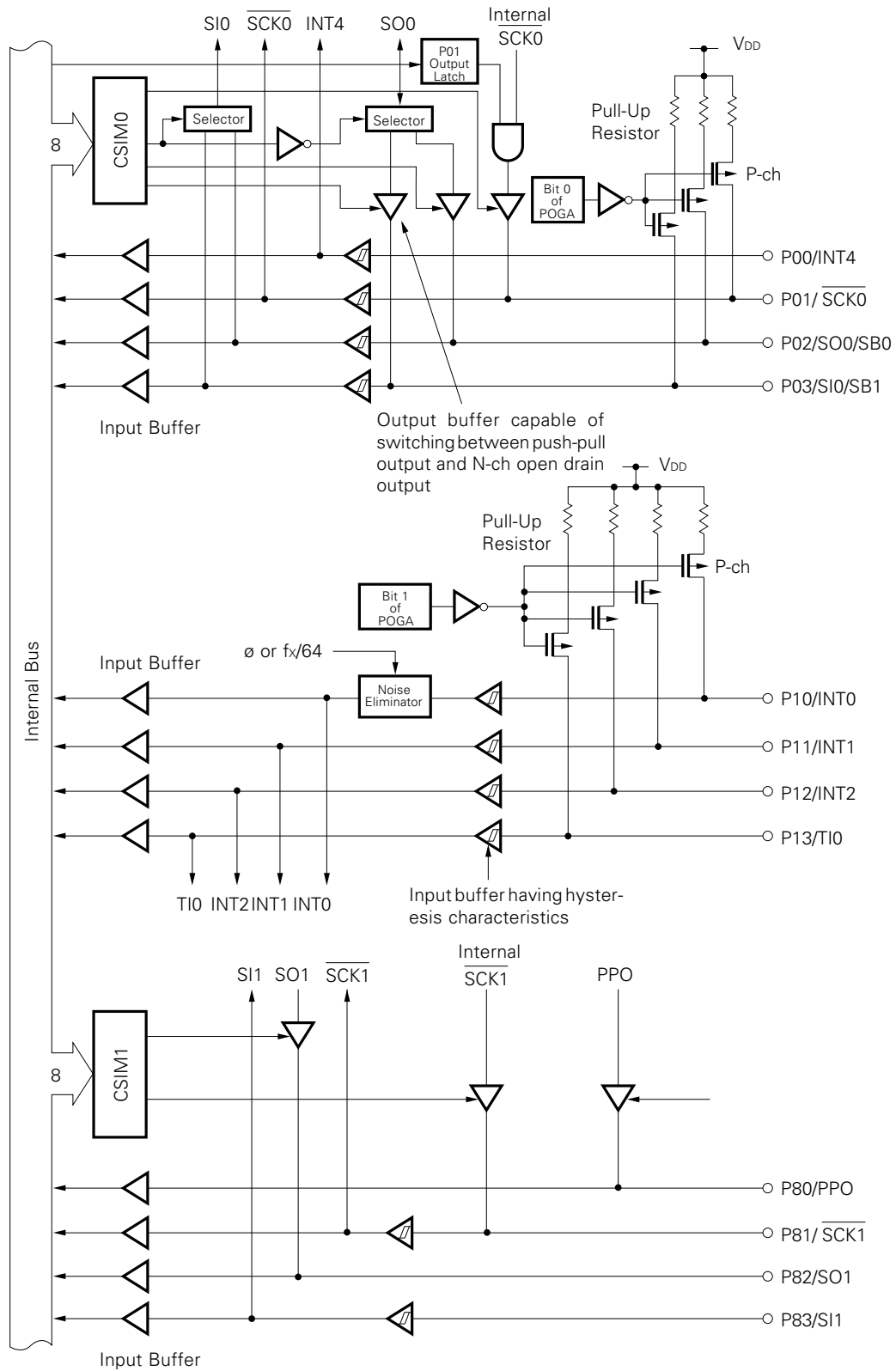


Fig. 4-3 Port 3n and Port 6n Configurations (n = 0 to 3)

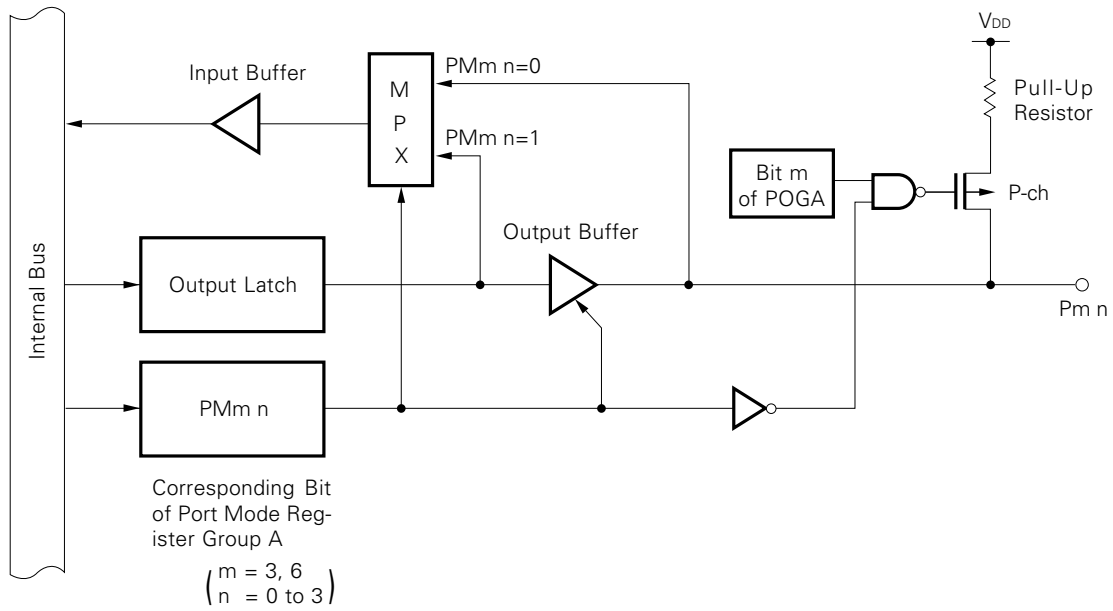


Fig. 4-4 Port 2 Configuration

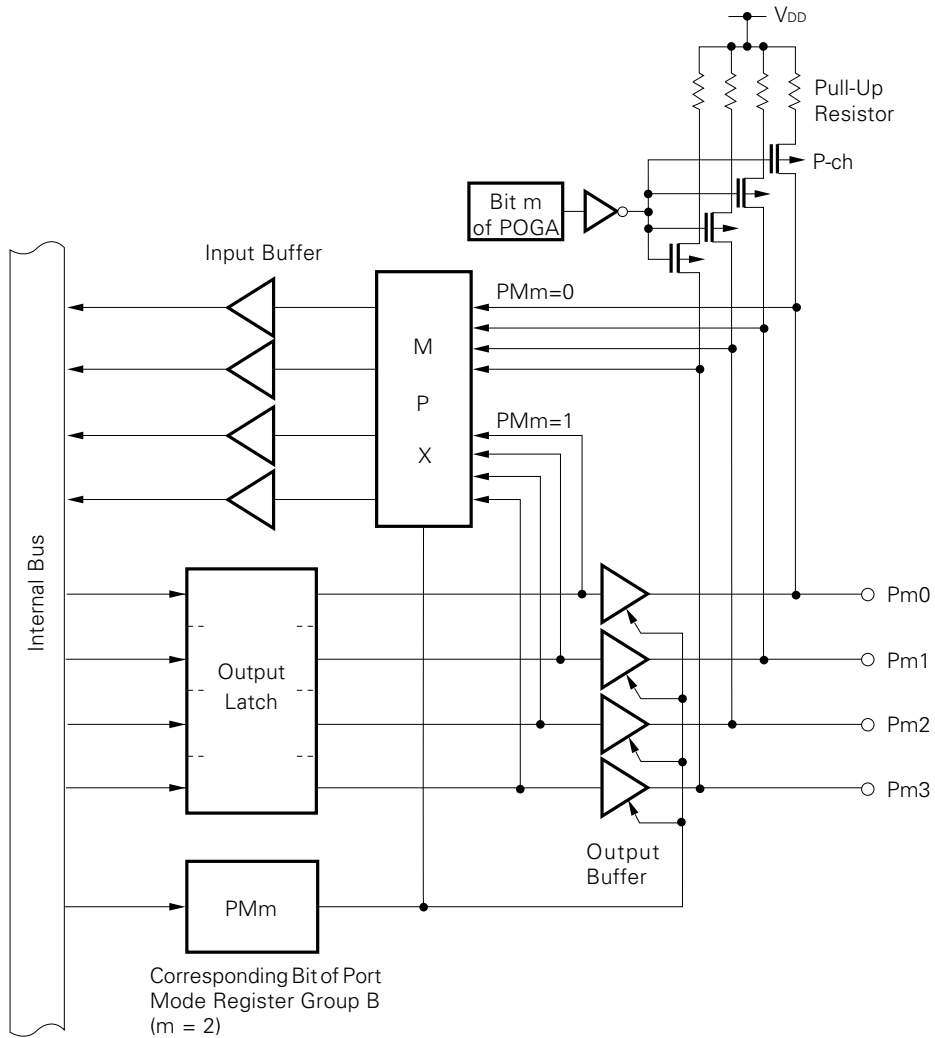


Fig. 4-5 Configurations of Ports 4 and 5

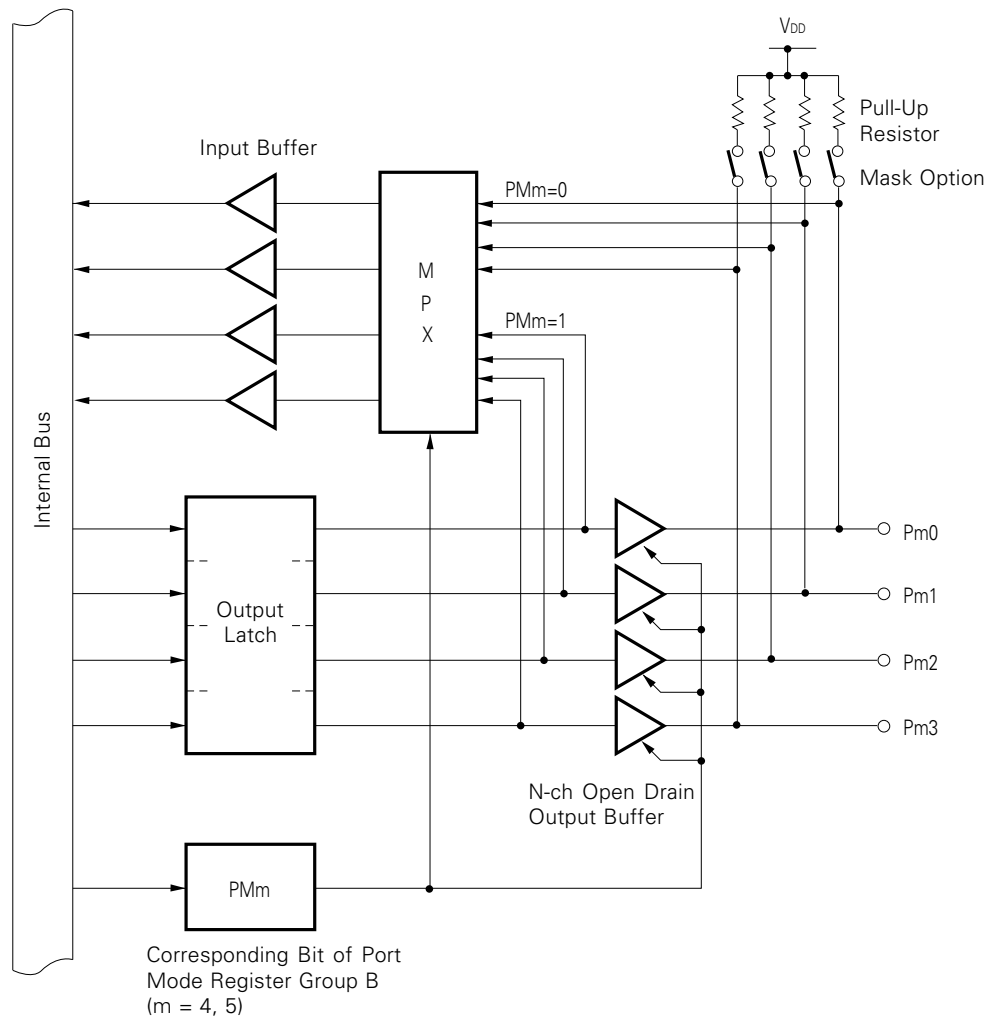


Fig. 4-6 Port 7 Configuration

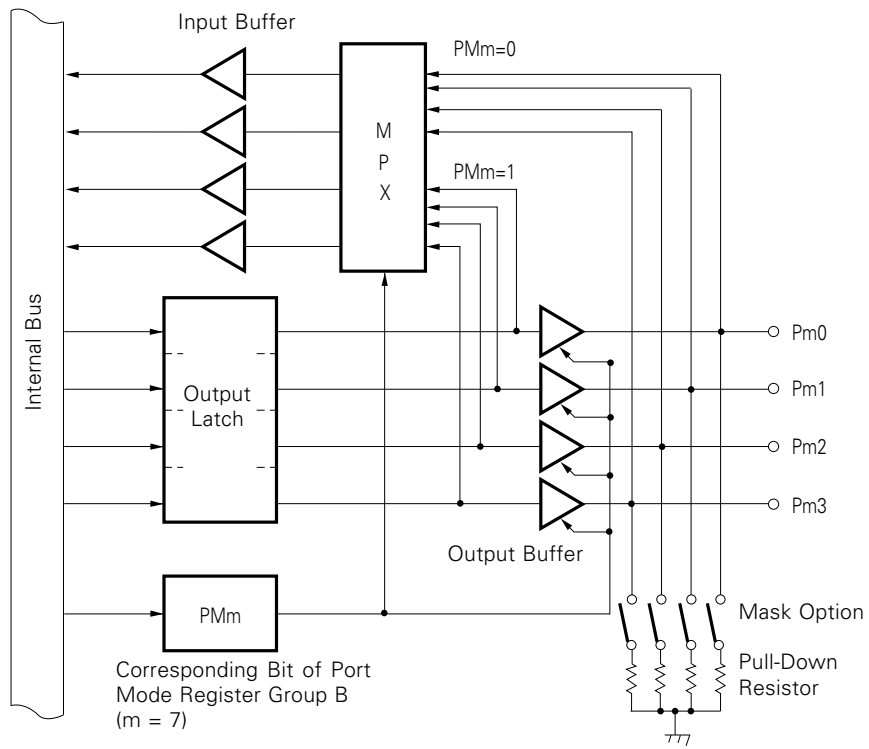
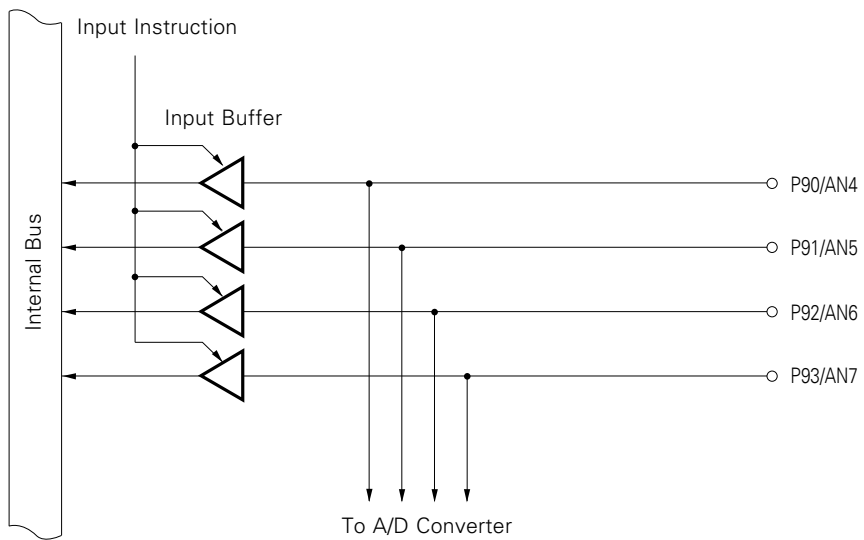
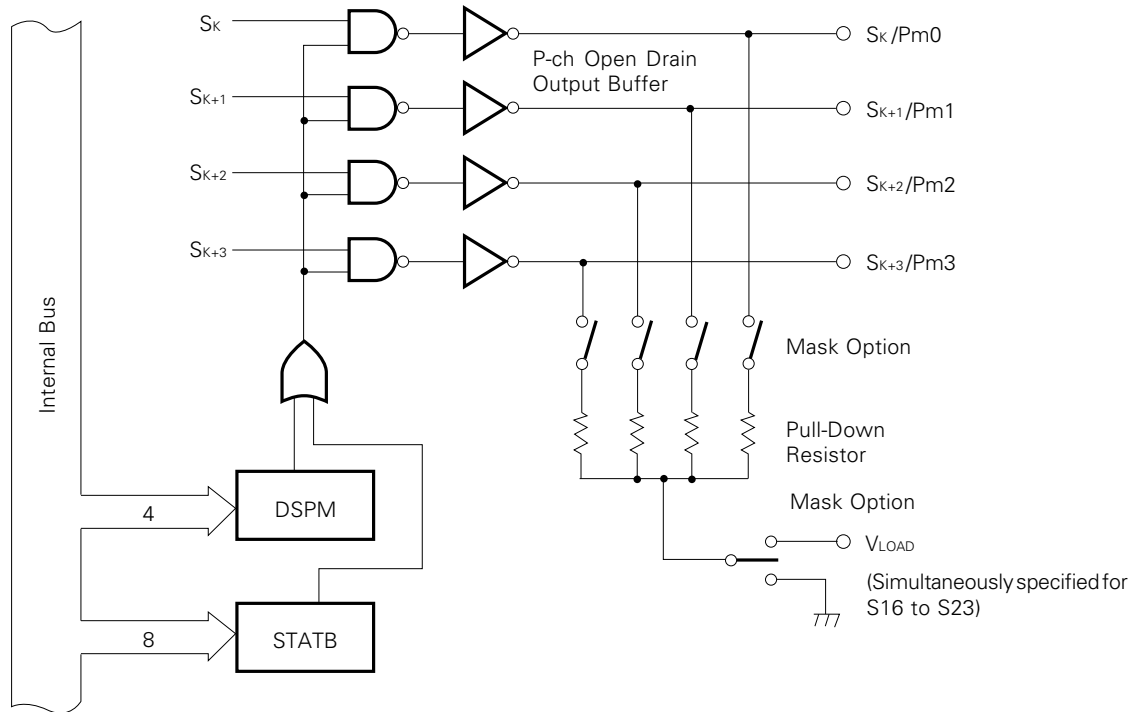


Fig. 4-7 Port 9 Configuration

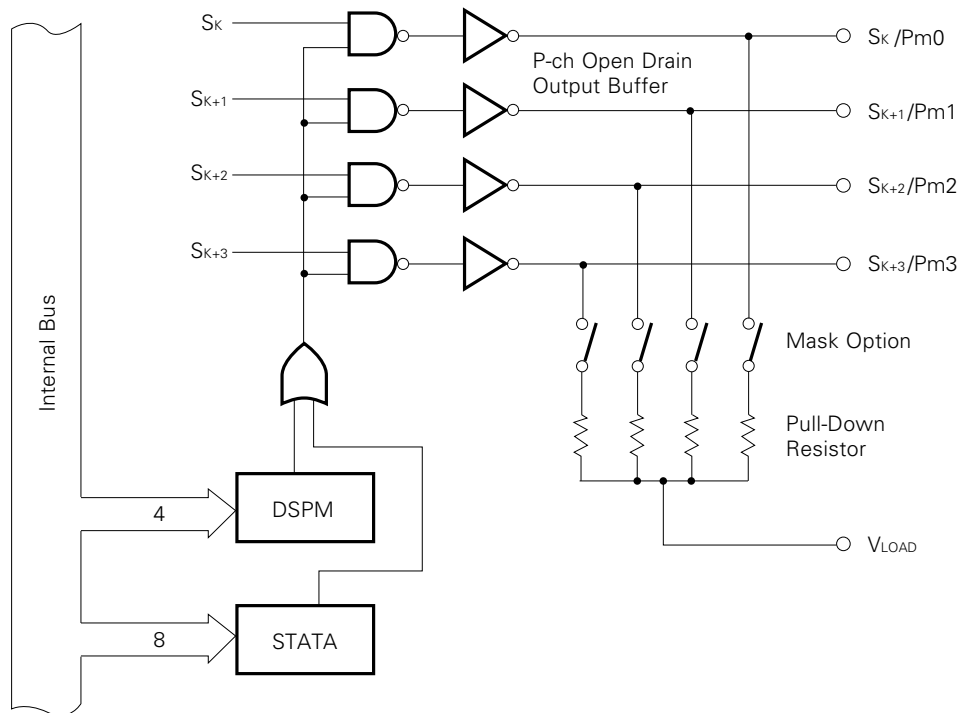


**Fig. 4-8 Configurations of Ports 10 and 11**



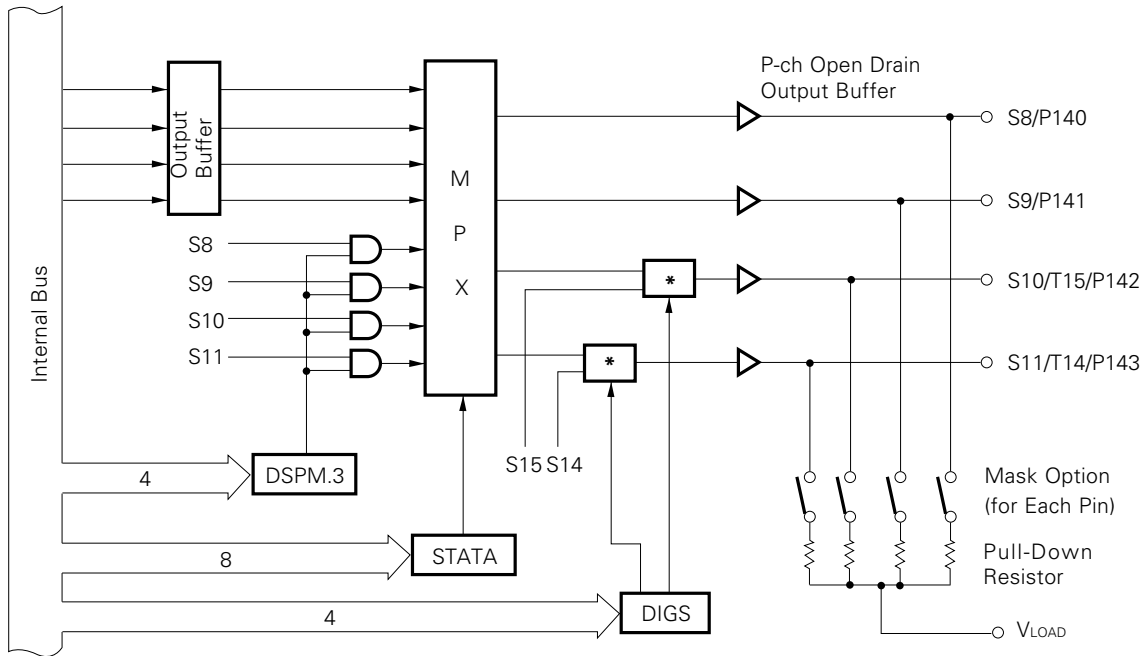
- Remarks**
1. Port 10:  $K = 16, m = 10$
  2. Port 11:  $K = 20, m = 11$

**Fig. 4-9 Configurations of Ports 12 and 13**



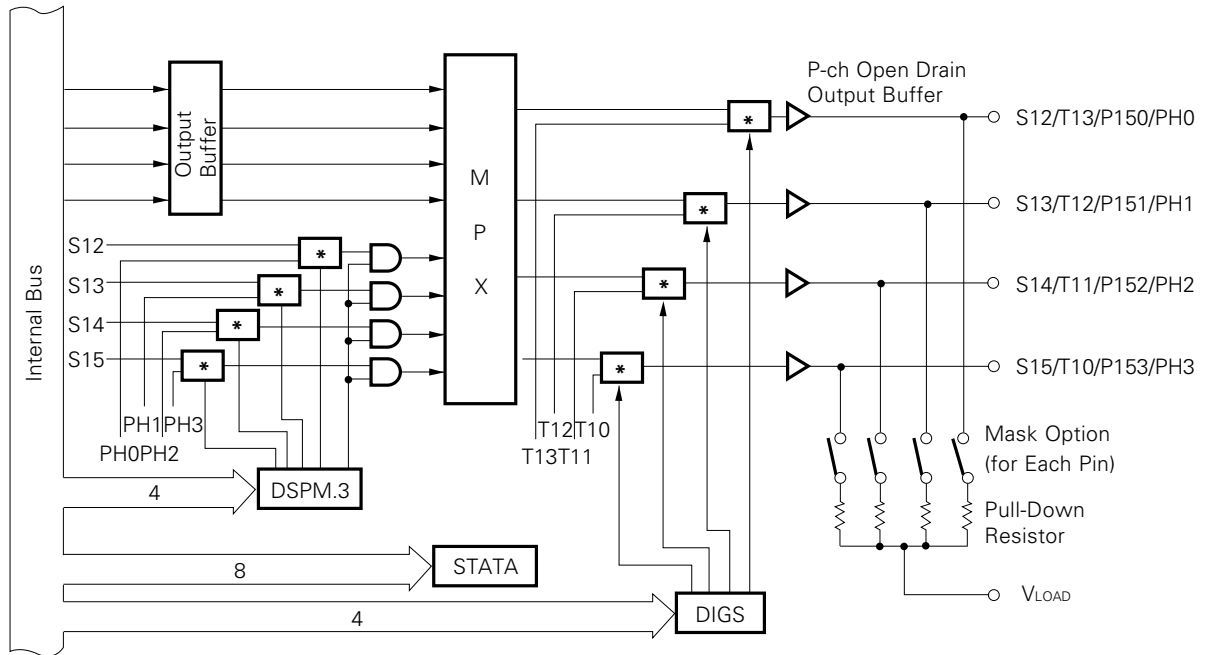
- Remarks**
1. Port 12:  $K = 0, m = 12$
  2. Port 13:  $K = 4, m = 13$

Fig. 4-10 Port 14 Configuration



\* Selector

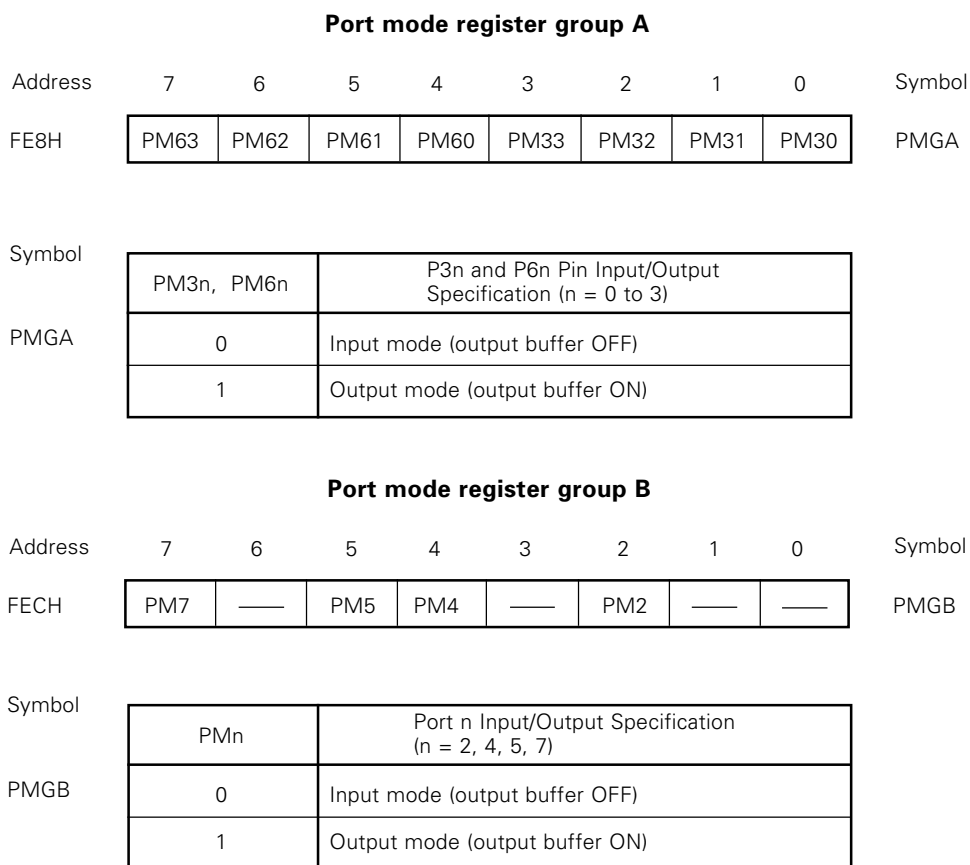
Fig. 4-11 Configurations of Ports 15 and H



\* Selector



**Fig. 4-12 Port Mode Register Format**



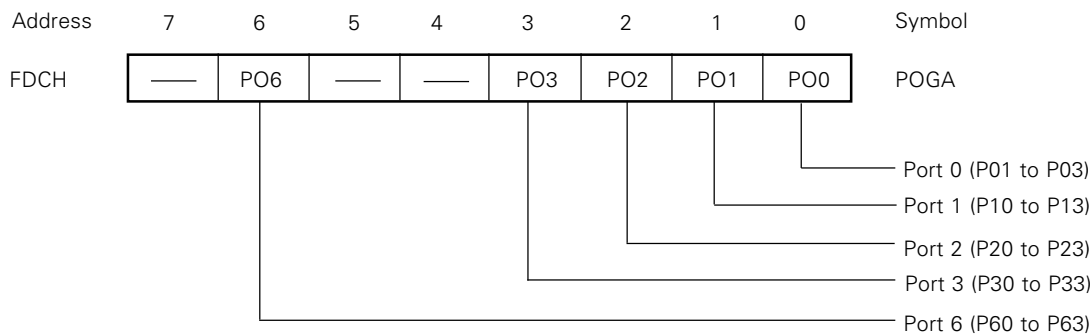
**Remarks** — : 0 or 1

**(4) Pull-up resistor register group A (POGA)**

Pull-up resistor register group A is intended to specify pull-up resistors to be built in ports 0 to 3 and port 6 (except P00). Fig. 4-13 shows the format.

Set "1" when a pull-up resistor is incorporated or "0" when it is not incorporated.

**Fig. 4-13 Pull-Up Resistor Register Group A Format**



**Note** Mask option by which pull-up resistors at ports 4 and 5 and pull-down resistors at port 7 and ports 10 to 15 can be incorporated bit-wise.

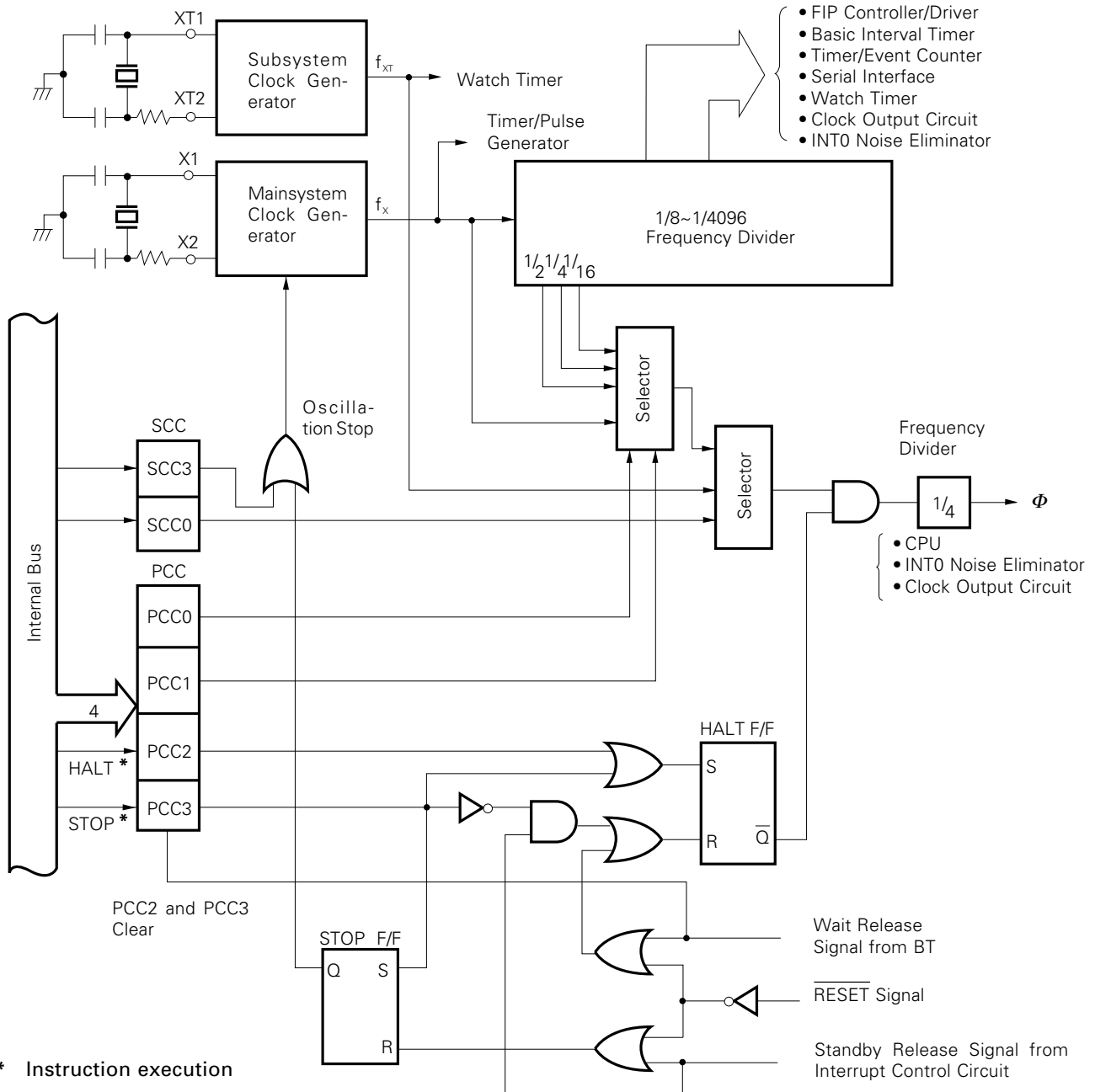
**Remarks** — : 0 or 1

4.2 CLOCK GENERATOR

(1) Clock generator configuration

The clock generator is a circuit to generate clocks to be supplied to the CPU and the peripheral hardware. Its configuration is shown in Fig. 4-14.

Fig. 4-14 Clock Generator Block Diagram



\* Instruction execution

- Remarks**
1.  $f_x$  = Main system clock frequency
  2.  $f_{XT}$  = Subsystem clock frequency
  3.  $\Phi$  = CPU clock
  4. PCC: Processor clock control register
  5. SCC: System clock control register

★ 6. 1 clock cycle ( $t_{CY}$ ) of  $\Phi$  is 1 machine cycle of an instruction. For  $t_{CY}$ , see "AC Characteristics" in 11.

**ELECTRICAL SPECIFICATIONS.**

## (2) Clock generator functions

The clock generator generates the following clocks and controls the CPU operating modes including the standby mode.

- Main system clock :  $f_x$
- Subsystem clock :  $f_{XT}$
- CUP CLOCK :  $\phi$
- Clocks for peripheral hardware

The following clock generator operations are determined by the processor clock control register (PCC) and the system clock control register (SCC):

- Upon  $\overline{\text{RESET}}$  input, the lowest speed mode (10.7  $\mu\text{s}$  : at 6.0 MHz operation)\*1 of the main system clock is selected. (PCC = 0, SCC = 0)
- One of the four-level CPU clocks can be selected by setting the PCC with the main system clock selected. (0.67  $\mu\text{s}$ , 1.33  $\mu\text{s}$ , 2.67  $\mu\text{s}$ , 10.7  $\mu\text{s}$  : at 6.0 MHz operation)\*2
- Two standby modes, the STOP and HALT modes, are available with the main system clock selected.
- The clock generator can be operated at an ultra-low speed and with low-level power consumption (122  $\mu\text{s}$  : at 32.768 KHz operation) by selecting the subsystem clock with SCC.
- Main system clock oscillation can be stopped by SCC with the subsystem clock selected. The HALT mode can also be used but the STOP mode cannot be used. (Subsystem clock oscillation cannot be stopped.)
- Divided system clocks are supplied to the peripheral hardware. Subsystem clocks can be directly supplied to the watch timer so that the timer function can be continued.
- When the subsystem clock is selected, the watch timer can operate normally. However, other hardware cannot be used if the main system clock is stopped.

- \* 1. 15.3  $\mu\text{s}$  : at 4.19 MHz operation  
2. 0.95  $\mu\text{s}$ , 1.91  $\mu\text{s}$ , 3.82  $\mu\text{s}$ , 15.3  $\mu\text{s}$  : at 4.19 MHz operation

**(3) Processor clock control register (PCC)**

The PCC is a 4-bit register to select the CPU clock  $\phi$  with the lower 2 bits and to control the CPU operating mode with the higher 2 bits. (See **Fig. 4-15 Processor Clock Control Register Format.**)

When bit 3 or 2 is set (1), the standby mode is set. If the standby mode is released by the standby release signal, both bits are automatically cleared and the normal operating mode is set. (For details, refer to **6. STANDBY FUNCTIONS.**)

The lower 2 bits of the PCC are set by the 4-bit memory manipulation instruction (with the higher 2 bits set to "0").

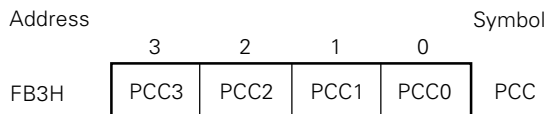
Bits 3 and 2 are reset "1" by the STOP and HALT instructions, respectively.

The STOP and HALT instructions can always be executed irrespective of the MBE contents.

The CPU clock selection is possible only when operated on the main system clock. When operated on the subsystem clock, the lower 2 bits of PCC are invalidated and  $f_{XT}/4$  is set. The STOP instruction is also enabled only when in operation with the main system clock.

$\overline{\text{RESET}}$  input clears PCC to "0".

Fig. 4-15 Processor Clock Control Register Format



**CPU Clock Select Bit**  
(When  $f_x = 6.0 \text{ MHz}$ )

		SCC = 0		SCC = 1	
		Values in parentheses are when $f_x = 6.0 \text{ MHz}$		Values in parentheses are when $f_{XT} = 32.768 \text{ kHz}$	
		CPU Clock Frequency	1 Machine Cycle	CPU Clock Frequency	1 Machine Cycle
0	0	$\Phi = f_x/64$ (93.7 kHz)	10.7 $\mu\text{s}$	$\Phi = f_{XT}/4$ (8.192 kHz)	122 $\mu\text{s}$
0	1	$\Phi = f_x/16$ (375 kHz)	2.67 $\mu\text{s}$	Setting prohibited	
1	0	$\Phi = f_x/8$ (750 kHz)	1.33 $\mu\text{s}$	$\Phi = f_{XT}/4$ (8.192 kHz)	122 $\mu\text{s}$
1	1	$\Phi = f_x/4$ (1.5 MHz)	0.67 $\mu\text{s}$		

(When  $f_x = 4.19 \text{ MHz}$ )

		SCC = 0		SCC = 1	
		Values in parentheses are when $f_x = 4.19 \text{ MHz}$		Values in parentheses are when $f_{XT} = 32.768 \text{ kHz}$	
		CPU Clock Frequency	1 Machine Cycle	CPU Clock Frequency	1 Machine Cycle
0	0	$\Phi = f_x/64$ (65.5 kHz)	15.3 $\mu\text{s}$	$\Phi = f_{XT}/4$ (8.192 kHz)	122 $\mu\text{s}$
0	1	$\Phi = f_x/16$ (262 kHz)	3.81 $\mu\text{s}$	Setting prohibited	
1	0	$\Phi = f_x/8$ (524 kHz)	1.91 $\mu\text{s}$	$\Phi = f_{XT}/4$ (8.192 kHz)	122 $\mu\text{s}$
1	1	$\Phi = f_x/4$ (1.05 MHz)	0.95 $\mu\text{s}$		

- Remarks**
1.  $f_x$  : Main system clock oscillator output frequency
  2.  $f_{XT}$  : Subsystem clock oscillator output frequency

**CPU Operating Mode Control Bit**

0	0	Normal operating mode
0	1	HALT mode
1	0	STOP mode
1	1	Setting prohibited

**(4) System clock control register (SCC)**

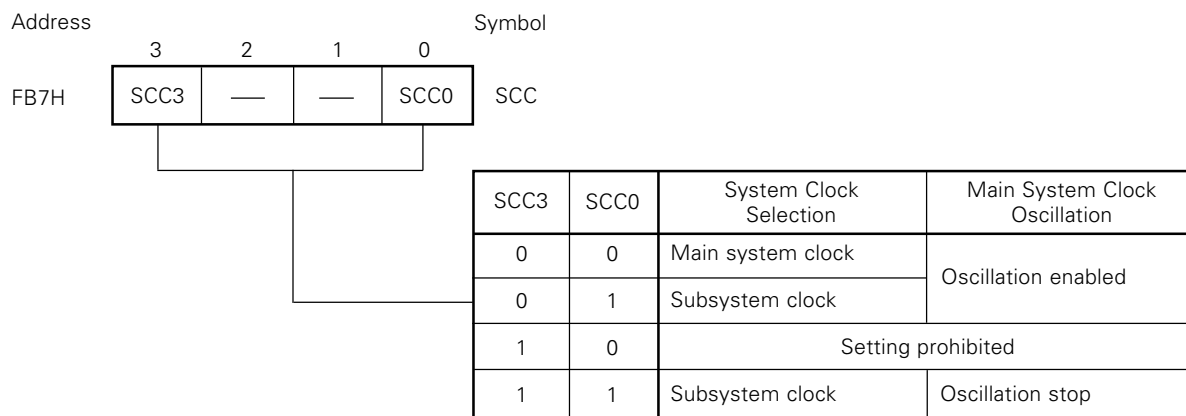
The SCC is a 4-bit register to select the CPU clock  $\Phi$  with the least significant bit and to control main system clock oscillation stop with the most significant bit (refer to Fig. 4-16).

Although SCC.0 and SCC.3 are located at the same data memory address, both bits cannot be changed simultaneously. Thus, SCC.0 and SCC.3 are set by the bit manipulation instruction. SCC.0 and SCC.3 can always be bit manipulated irrespective of the MBE contents.

Main system clock oscillation can be stopped by setting SCC.3 only when in operation with the subsystem clock. Oscillation when in operation with the main system clock is stopped by the STOP instruction.

RESET input clears SCC to "0".

**Fig. 4-16 System Clock Control Register Format**



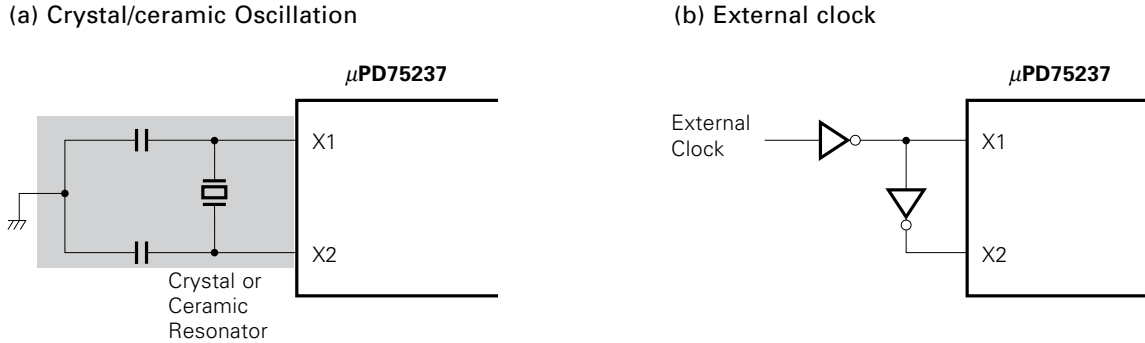
- Note**
1. A maximum of  $1/f_{XT}$  is required to change the system clock. Thus, when stopping the main system clock oscillation, change the clock to the subsystem clock and set SCC.3 following the passage of more than the machine cycles described in Table 4-2.
  2. The normal STOP mode cannot be set if oscillation is stopped by setting SCC.3 while in operation with the main system clock.
  3. If SCC.3 is set to "1", X1 input is internally short-circuited to V<sub>SS</sub> (GND potential) to suppress crystal oscillator leakage. Thus, when using an external clock for the main system clock do not set SCC.3 to "1".
  4. When PCC = 0001B ( $\Phi = f_X/16$  selected), do not set SCC.0 to "1". When switching from the main system clock to the subsystem clock, do so after setting PCC to another value (PCC ≠ 0001B). Do not set PCC = 0001B while in operation with the subsystem clock.

(5) **System clock oscillator**

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

External clocks can be input to this oscillator.

**Fig. 4-17 External Circuit of Main System Clock Oscillator**

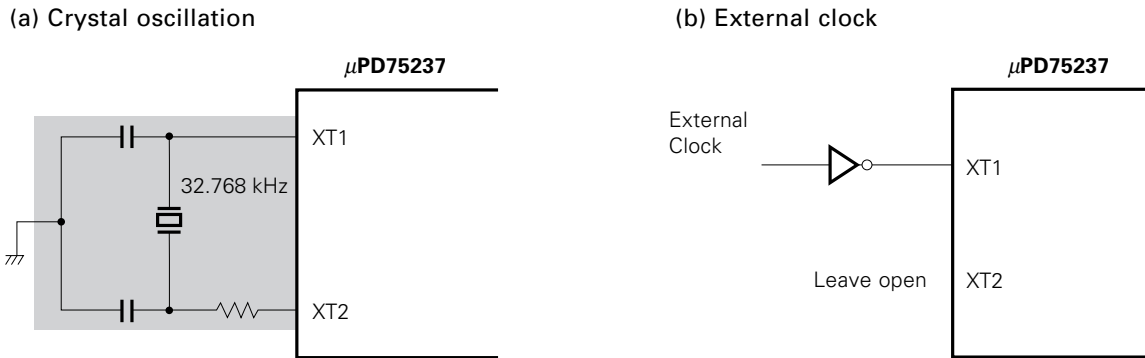


**Note** The STOP mode cannot be set while an external clock is input because the X1 pin is short-circuited to V<sub>SS</sub> in the STOP mode.

The subsystem clock oscillator oscillates with a crystal resonator (with a standard frequency of 32.768 kHz) connected to the XT1 and XT2 pins.

External clocks can be input to this oscillator.

**Fig. 4-18 External Circuit of Subsystem Clock Oscillator**



**Note** When using a main system clock and subsystem clock oscillator, wire the crosshatched section in Figs. 4-17 and 4-18 as follows to prevent any effect of the wiring capacity.

- Make the wiring as short as possible.
- Do not allow wiring to intersect with other signal conductors. Do not allow wiring to be near a line through which varying high current flows.
- Set the oscillator capacitor grounding point to the same potential as that of V<sub>SS</sub>. Do not ground to a ground pattern through which high current flows.
- Do not fetch signals from the oscillator.

The subsystem clock oscillator has a low amplification factor to maintain low current consumption and is more likely to malfunction due to noise than the main system clock oscillator. Thus, take extra care when using a subsystem clock.

**(6) Time required for system clock and CPU clock switching**

The system clock and the CPU clock can be switched to each other with the least significant bit of the SCC and the lower 2 bits of the PCC. This switching is not executed just after register rewrite and operation continues with the previous clock during the specified machine cycle. Thus, to stop main system clock oscillation, it is necessary to execute the STOP instruction or to set SCC.3 after the specified switching time.

**Table 4-2 Maximum Time Required for System Clock and CPU Clock Switching**

Set Value before Switching			Set Value 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	×	×
0	0	0	/			1 machine cycle			1 machine cycle			1 machine cycle			$\frac{f_x}{64f_{XT}}$ machine cycle (3 machine cycle)		
	0	1				4 machine cycle			4 machine cycle			4 machine cycle			Setting prohibited		
	1	0				8 machine cycle			8 machine cycle			8 machine cycle			$\frac{f_x}{8f_{XT}}$ machine cycle (23 machine cycle)		
	1	1				16 machine cycle			16 machine cycle			16 machine cycle			$\frac{f_x}{4f_{XT}}$ machine cycle (64 machine cycle)		
1	×	×	1 machine cycle			Setting prohibited			1 machine cycle			1 machine cycle			/		

- Remarks**
1. CPU clock  $\phi$  is a clock to be supplied to the internal CPU of μPD75237 and its inverse number is the minimum instruction time (defined as "one machine cycle" in this manual).
  2. Values in parentheses are when  $f_x = 6.0$  MHz and  $f_{XT} = 32.768$  kHz.

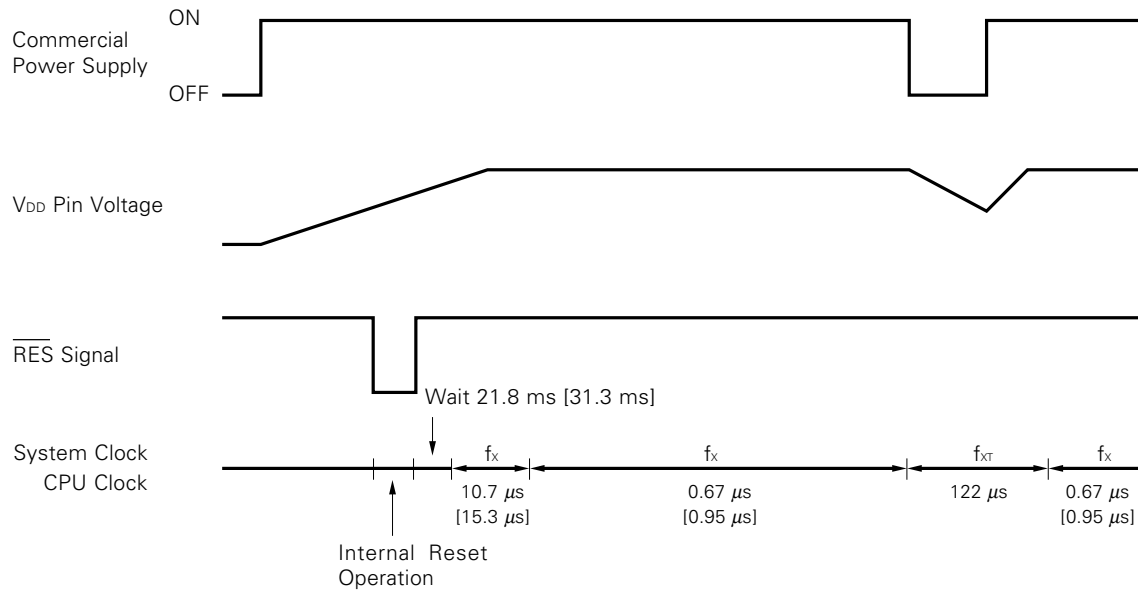
**Note** When PCC = 0001B ( $\phi = f_x/16$  selected), do not set SCC.0 to "1". When switching from the main system clock to the subsystem clock, do so after setting PCC to another value (PCC ≠ 0001B). Do not set PCC = 0001B while in operation with the subsystem clock.



(7) System clock and CPU clock switching procedure

System clock and CPU clock switching is described referring to Fig. 4-19.

Fig. 4-19 System Clock and CPU Clock Switching



**Remarks** ( $f_x=6.0\text{ MHz}$ ,  $f_{xt}=32.768\text{ kHz}$ ), values in brackets are when  $f_x = 4.19\text{ MHz}$ .

- ① RESET input starts the CPU at the lowest speed (21.8 ms : at 6.0 MHz operation)\*1 of the main system clock after the wait time (10.7 μs : at 6.0 MHz operation)\*2 for maintaining the oscillation stabilize time.
- ② The CPU rewrites the PCC and operates at its maximum available speed after the lapse of sufficient time for the V<sub>DD</sub> pin voltage to increase to a voltage allowing the highest speed operation.
- ③ The CPU detects commercial power-off from the interrupt input (INT4 is effective), sets SCC.0 and operates with the subsystem clock. (At this time, subsystem clock oscillation must have started beforehand.) After the passage of time required for the CPU clock to switch to the subsystem clock (32 machine cycles), the CPU sets SCC.3 to stop main system clock oscillation.
- ④ After the CPU detects the commercial power restored from the interrupt, it clears SCC.3 and starts main system clock oscillation. Following the passage of time required for oscillation stabilization, the CPU clears SCC.0 and operates at its highest speed.

- \* 1. 31.3 ms at 4.19 MHz operation
- 2. 15.3 μs at 4.19 MHz operation

4.3 CLOCK OUTPUT CIRCUIT

(1) Clock output circuit configuration

The clock output circuit is configured as shown in Fig. 4-20.

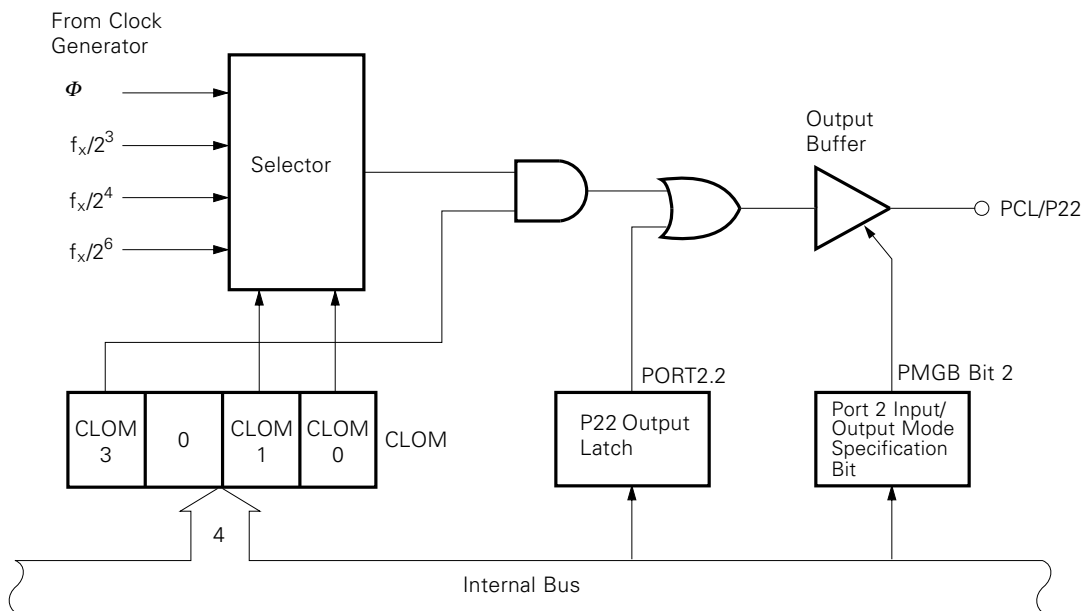
(2) Clock output circuit functions

The clock output circuit is intended to generate clock pulses from the P22/PCL pin. It is used for remote-controlled output or clock pulse supply to the peripheral LSI.

Follow the procedure below to generate clock pulses.

- (a) Select the clock output frequency. Do not output clocks.
- (b) Write 0 to P22 output latch.
- (c) Set the port 2 input/output mode to 'output'.
- (d) Enable clock output.

Fig. 4-20 Clock Output Circuit Configuration



**Remarks** The clock output circuit has such a configuration as to prevent pulses having short widths when switching clock output enable/disable.

**(3) Clock output mode register (CLOM)**

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

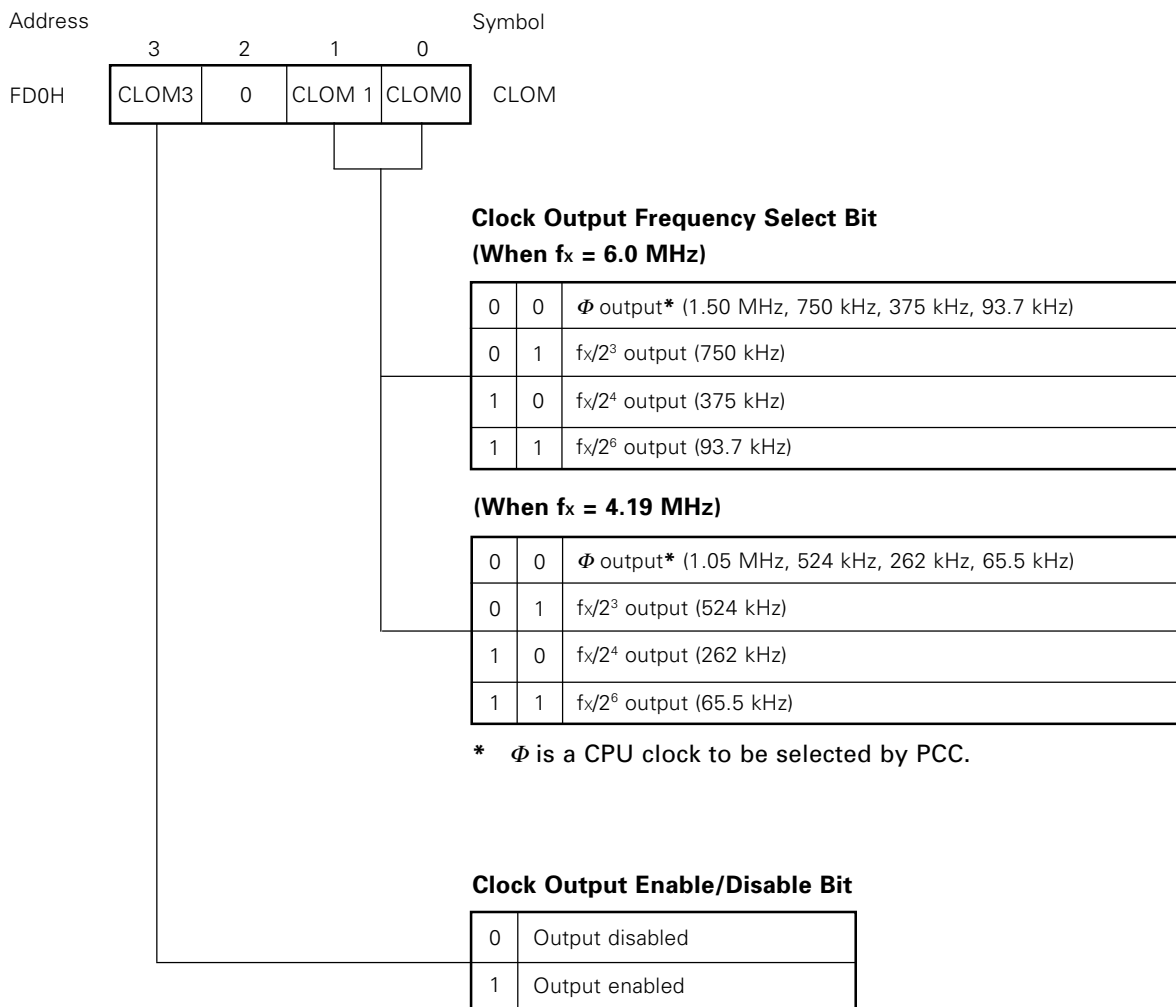
The CLOM is set by a 4-bit memory manipulation instruction. Data cannot be read from the CLOM.

```

Example   CPU clock  $\Phi$  output from PCL/P22 pin
SEL    MB15      ; Or CLR1 MBE
MOV    A, #1000B
MOV    CLOM, A
    
```

$\overline{\text{RESET}}$  input clears the CLOM to 0 and disables clock output.

**Fig. 4-21 Clock Output Mode Register Format**



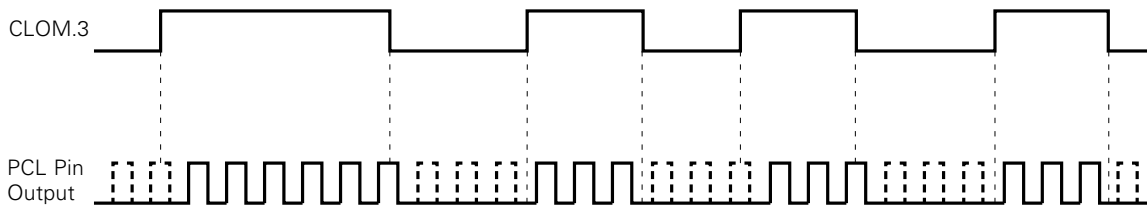
**Note** Be sure to write "0" to bit 2 of CLOM.

**(4) Example of application to remote-controlled output**

The clock output function of the μPD75237 can be applied to remote-controlled output. The carrier frequency of remote-controlled output is selected by the clock frequency select bit of the clock output mode register. Pulse output is enabled/disabled by controlling the clock output enable/disable bit by software.

The clock output circuit has such a configuration as to prevent pulses having short widths when switching clock output enable/disable.

**Fig. 4-22 Remote-Controlled Output Application Example**



4.4 BASIC INTERVAL TIMER

(1) Basic interval timer configuration

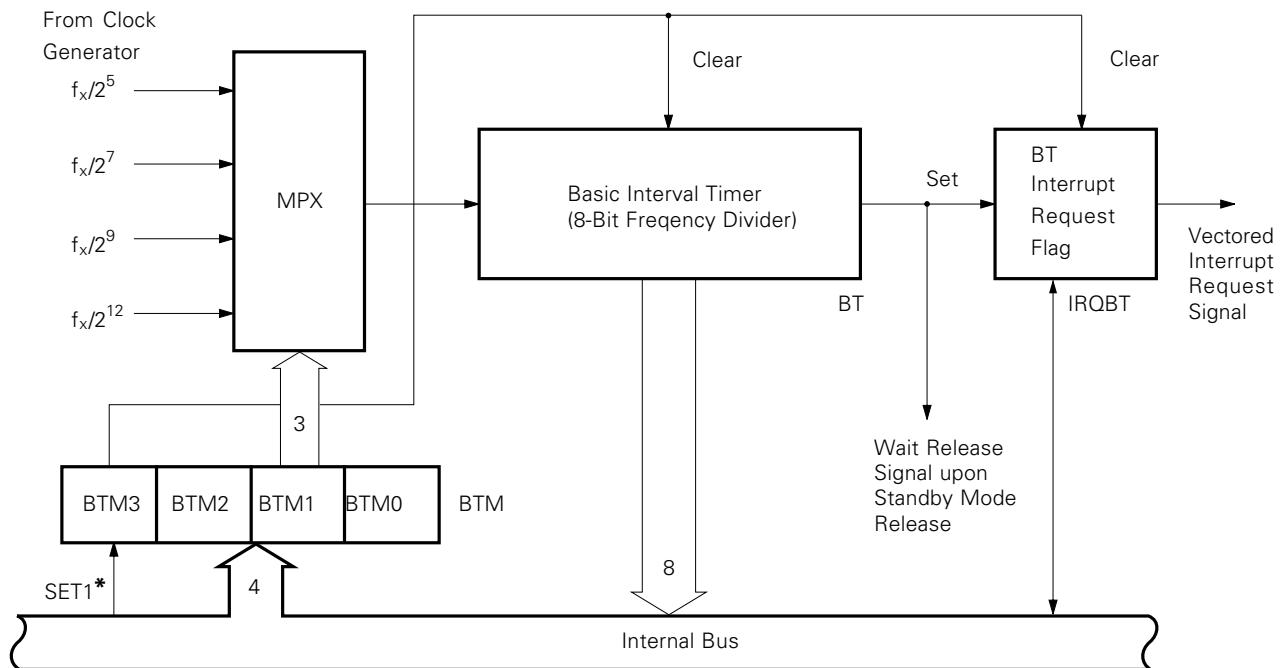
The basic interval timer configuration is shown in Fig. 4-23.

(2) Basic interval timer functions

The basic interval timer has the following functions:

- (a) Interval timer operation to generate reference time (at any of four time intervals)
- (b) Watchdog timer application to detect inadvertent program loop
- (c) Wait time select and count upon standby mode release
- (d) Count contents read

Fig. 4-23 Basic Interval Timer Configuration



\* Instruction execution

**(3) Basic interval timer mode register (BTM)**

The BTM is a 4-bit register to control basic interval timer operations.

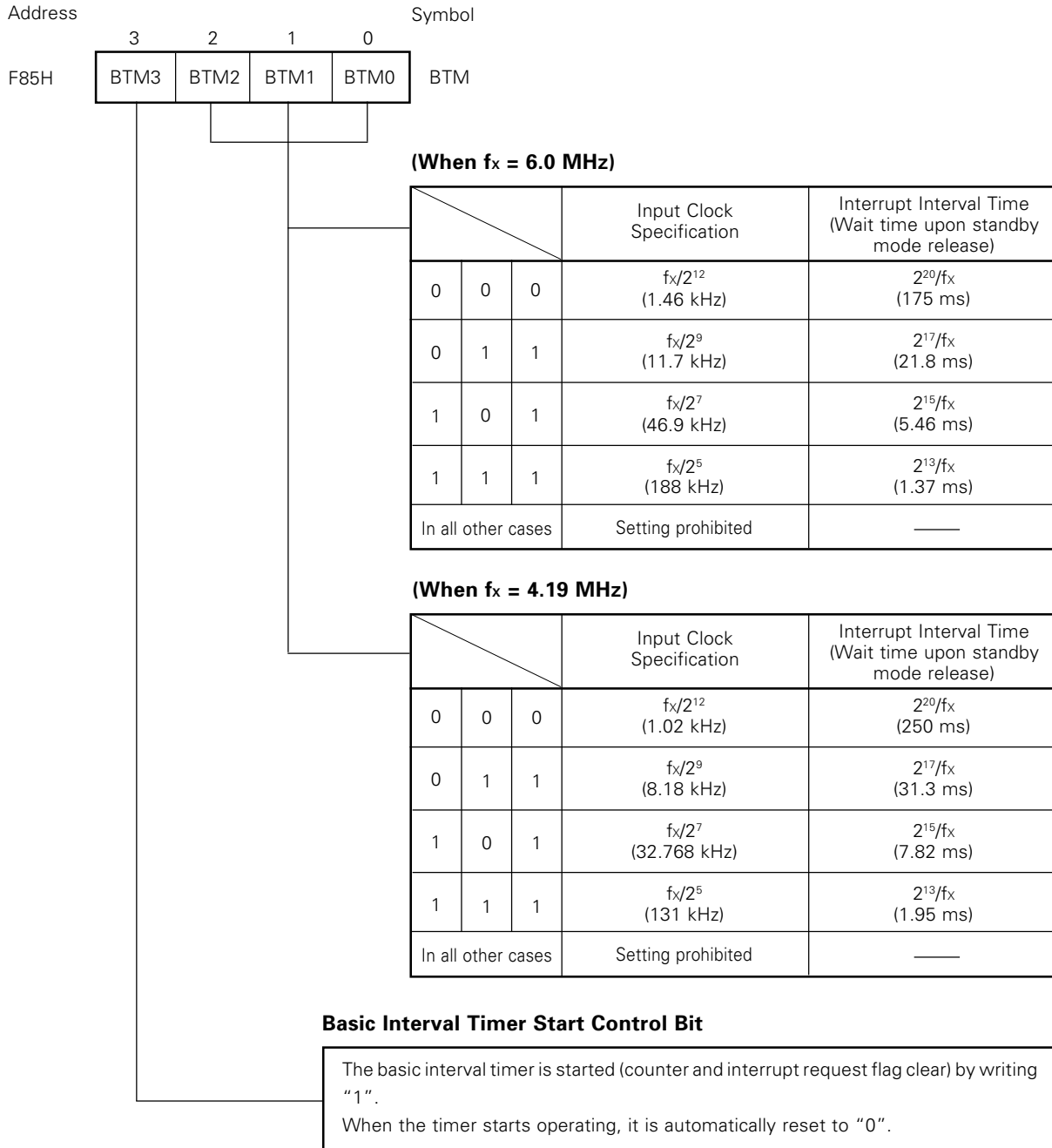
The BTM is set by a 4-bit memory manipulation instruction.

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

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

RESET input clears the contents to "0" and sets the interrupt request signal generation interval time to its maximum value.

**Fig. 4-24 Basic Interval Timer Mode Register Format**



#### (4) Basic interval timer operation

The basic interval timer (BT) is always incremented by clocks from the clock generator and sets the interrupt request flag (IRQBT) due to an overflow. BT count operation cannot be stopped.

Four interrupt generate intervals are available by setting the BTM (refer to **Fig. 4-24 Basic Interval Timer Mode Register Format**).

The basic interval timer and the interrupt request flag can be cleared by setting bit 3 of the BTM (1) (interval timer start instruction).

The count state can be read from the basic interval timer (BT) by the 8-bit manipulation instruction. Data cannot be written to the BT.

**Note** When reading the basic interval timer count contents, execute the read instruction twice and compare the two read contents so as not to read unstable data undergoing count update. If the two values are both acceptable, use the second read value as the correct one. If they differ completely, execute reading again from the beginning.

To obtain the oscillation stabilize time from STOP mode release to system clock oscillation stabilization, the wait function is available to stop CPU operation until the basic interval timer overflows.

Wait time after RESET input is fixed, however, if the STOP mode has been released by interrupt generation, the wait time can be selected by BTM setting. In that case, the wait time is equal to the interval time shown in Fig. 4-24.

BTM setting must be done before STOP mode setting. (For details, refer to **6 . STANDBY FUNCTIONS**.)

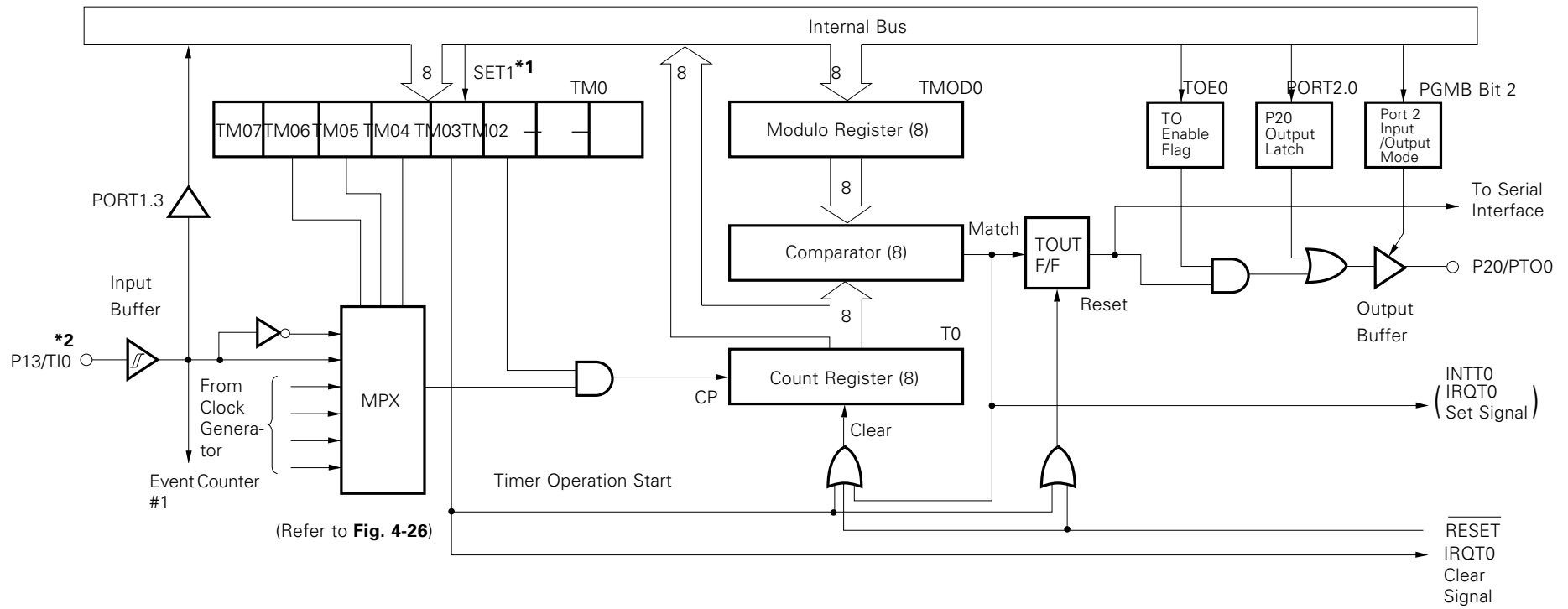
## 4.5 TIMER/EVENT COUNTER

### (1) Timer/event counter functions

The timer/event counter has the following functions.

- (a) Program interval timer operation
- (b) Output of square wave with any frequency to PTO0 pin
- (c) Event counter operation
- (d) Output of N-divided TI0 pin input to PTO0 pin (frequency divider operation)
- (e) Serial shift clock supply to the serial interface circuit
- (f) Count state read function

Fig. 4-25 Timer/Event Counter Block Diagram



- \* 1. Instruction execution
- \* 2. P13/TI0 pin is an external event pulse input pin which serves as timer/event counter and event counter.



**(2) Timer/event counter mode register (TM0) and timer/event counter output enable flag (TOE0)**

The timer/event counter mode register (TM0) is an 8-bit register to control the timer/event counter and is set by an 8-bit memory manipulation instruction.

Fig. 4-27 shows the timer/event counter mode register format.

Bit 3 is a timer start command bit which can be set independently. When the timer starts operating, this bit is automatically reset to "0".

RESET input clears all bits of the TM0 to 0.

The timer/event counter output enable flag (TOE0) controls enable/disable for output to the PTO0 pin in the timer out F/F (TOUT F/F) state.

Fig. 4-26 shows the timer/event counter output enable flag format.

The timer out F/F (TOUT F/F) is an F/F which is reversed by a match signal transmitted from the comparator. The timer out F/F is reset by an instruction which sets bit 3 of the TM0.

RESET input clears TOE0 and TOUT F/F to 0.

**Fig. 4-26 Timer/Event Counter Output Enable Flag Format**

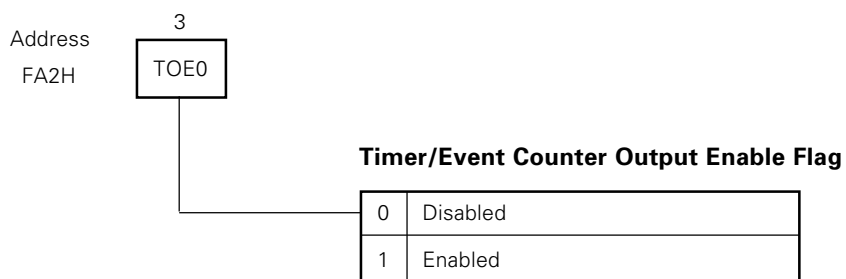
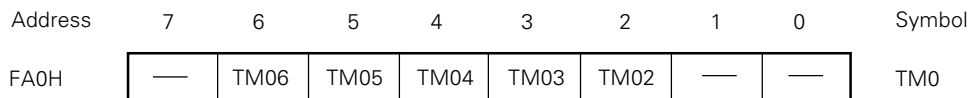


Fig. 4-27 Timer/Event Counter Mode Register Format



**Operating Mode**

	Count operation
0	Stop (with count contents held)
1	Count operation

**Timer Start Command Bit**

Writing "1" clears the counter and IRQT0 flag.  
If bit 2 has been set to "1", the counter operation starts.

**Count Pulse (CP) Select Bit**

(When  $f_x = 6.0$  MHz)

TM06	TM05	TM04	Count Pulse (CP)
0	0	0	T10 input rising edge
0	0	1	T10 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)
In all other cases			Setting prohibited

(When  $f_x = 4.19$  MHz)

TM06	TM05	TM04	Count Pulse (CP)
0	0	0	T10 input rising edge
0	0	1	T10 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)
In all other cases			Setting prohibited

**(3) Timer/event counter operating modes**

The count operation stop mode and the count operating mode are available by setting the mode register for the timer/event counter operation.

The following operations are enabled irrespective of the mode register setting:

- (a) TI0 pin signal input and test (Dual-function pin P13 input testable)
- (b) Output of the timer out F/F state to PTO0
- (c) Modulo register (TMOD0) setting
- (d) Count register (T0) read
- (e) Interrupt request flag (IRQT0) set/clear/test

**(a) Count operation stop mode**

When TM0 bit 2 is 0, this mode is set. In this mode, count operation is not carried out because count pulse (CP) supply to the count register is stopped.

**(b) Count operating mode**

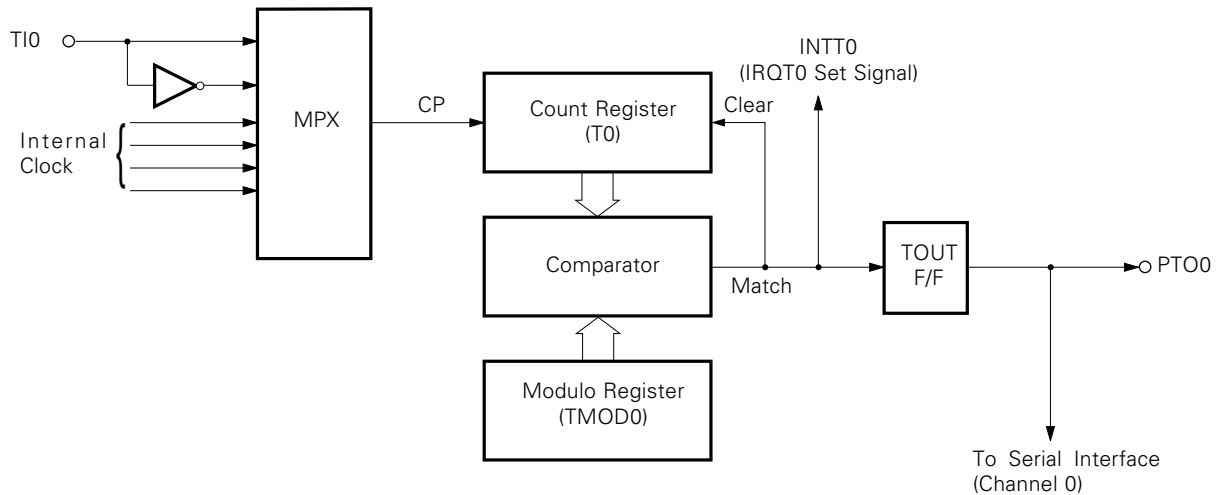
When TM0 bit 2 is 1, this mode is set. The count pulse selected by bits 4 to 6 is supplied to the count register and the count operation shown in Fig. 4-28 is carried out.

The timer operation is normally started by the following operations in the described order.

- ① Set the number of counts to the modulo register (TMOD0).
- ② Set the operating mode, count clock and start command to the mode register (TM0).

Set the modulo register by an 8-bit data transfer instruction.

**Fig. 4-28 Operation in Count Operating Mode**



**(4) Timer/event counter time setting**

[Timer set time] (cycle) is obtained by dividing [Modulo register contents + 1] by [Count pulse frequency] selected by timer mode register setting.

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

T (sec) : Timer set time (sec)

f<sub>CP</sub> (Hz) : Count pulse frequency (Hz)

n : Modulo register value (n ≠ 0)

Once the timer is set, an interrupt request signal (IRQT0) is generated at the set intervals. Table 4-3 shows the resolutions with each count pulse of the timer/event counter and the maximum set time (with FFH set to the modulo register).

**Table 4-3 Resolution and Maximum Set Time**

**(When f<sub>x</sub> = 6.0 MHz)**

Mode Register			Timer Channel 0	
TM06	TM05	TM04	Resolution	Maximum Set Time
1	0	0	171 μs	43.7 ms
1	0	1	42.7 μs	10.9 ms
1	1	0	10.7 μs	2.73 ms
1	1	1	2.67 μs	683 μs

**(When f<sub>x</sub> = 4.19 MHz)**

Mode Register			Timer Channel 0	
TM06	TM05	TM04	Resolution	Maximum Set Time
1	0	0	244 μs	62.5 ms
1	0	1	61.1 μs	15.6 ms
1	1	0	15.3 μs	3.91 ms
1	1	1	3.81 μs	977 μs

4.6 WATCH TIMER

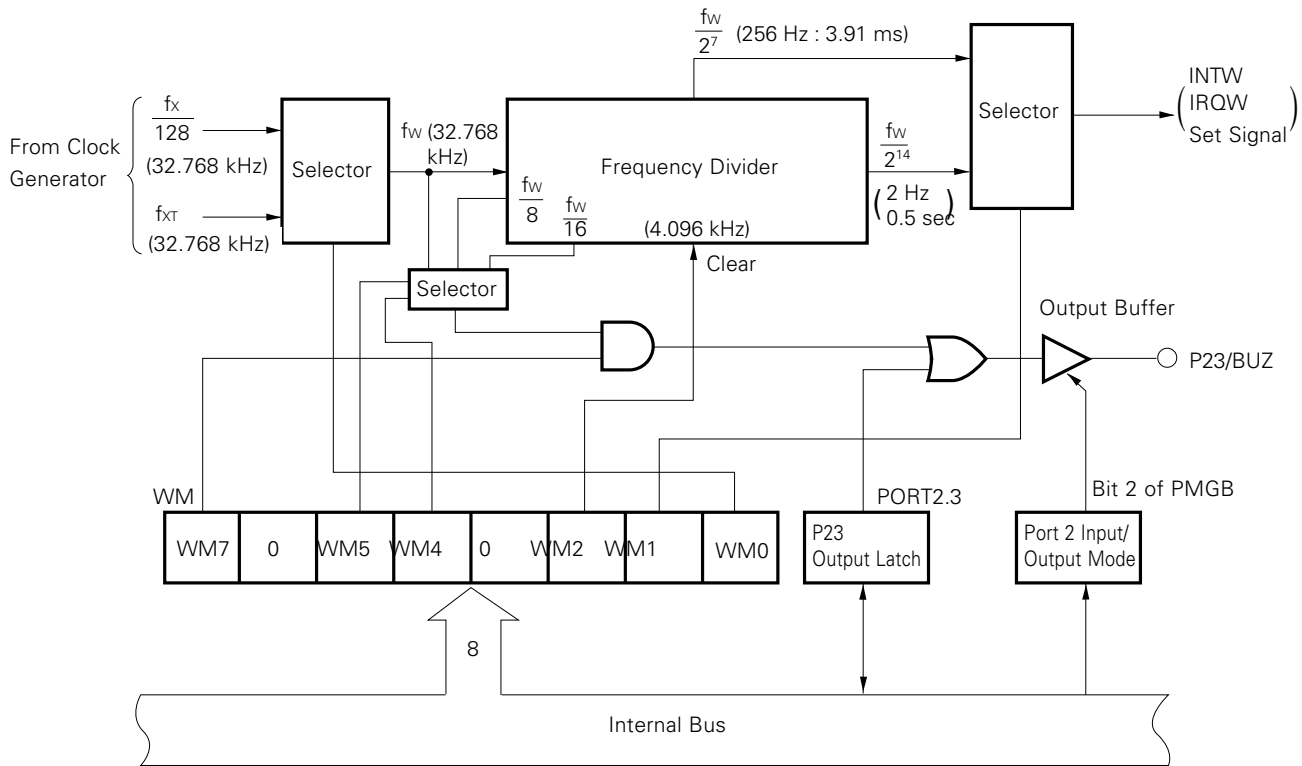
(1) Watch timer

The μPD75237 incorporates one channel of watch timer having a configuration shown in Fig. 4-29.

(2) Watch timer functions

- (a) Sets the test flag (IRQW) at 0.5-sec intervals.  
The standby mode can be released by IRQW.
- (b) 0.5-second interval can be set with the main system clock (4.19 MHz) or subsystem clock (32.768 kHz).
- (c) The fast mode enables to set 128-time (3.91 ms) interval useful to program debugging and inspection.
- (d) The fixed frequencies (2.048 kHz, 4.096 kHz and 32.768 kHz) can be output to the P23/BUZ pin for use to generate buzzer sound and trim the system clock oscillator frequency.
- (e) Since the frequency divider can be cleared, the watch can be started from zero second.

Fig. 4-29 Watch Timer Block Diagram



**Remarks** Values at  $f_x = 4.194304$  MHz and  $f_{xT} = 32.768$  kHz are indicated in parentheses.

**Note** In the main system clock 6.0 MHz operation, 0.5-second interval can be generated. Therefore, after switching to the subsystem clock, 0.5-second interval should be generated.

**(3) Watch mode register (WM)**

The watch mode register (WM) is an 8-bit register to control the watch timer. Its format is shown in Fig. 4-30.

The watch mode register is set by an 8-bit memory manipulation instruction.  $\overline{\text{RESET}}$  input clears all bits to "0".

**Fig. 4-30 Watch Mode Register Format**

Address	7	6	5	4	3	2	1	0	Symbol
F98H	WM7	0	WM5	WM4	0	WM2	WM1	WM0	WM

**Count Clock (fw) Select Bit**

WM0	0	System clock divided output: $\frac{f_x}{128}$ selected
	1	Subsystem clock: $f_{XT}$ selected

**Operating Mode Select Bit**

WM1	0	Normal watch mode $\left( \frac{f_w}{2^{14}} : \text{IRQW set at 0.5 sec} \right)$
	1	Fast watch mode $\left( \frac{f_w}{2^7} : \text{IRQW set at 3.91 ms} \right)$

**Watch Operation Enable/Disable Bit**

WM2	0	Watch operation stopped (frequency divider clear)
	1	Watch operation enabled

**BUZ Output Frequency Select Bit**

WM5	WM4	BUZ Output Frequency
0	0	$f_w/2^4$ (2.048 kHz)
0	1	$f_w/2^3$ (4.096 kHz) *
1	0	Setting prohibited
1	1	$f_w$ (32.768 kHz) *

\* Not supported with IE-75000-R

**BUZ Output Enable/Disable Bit**

WM7	0	BUZ output disabled
	1	BUZ output enabled

## 4.7 TIMER/PULSE GENERATOR

### (1) Timer/pulse generator functions

The μPD75237 incorporates one channel of timer/pulse generator which can be used as a timer or a pulse generator. The timer/pulse generator has the following functions.

#### (a) Functions available in the timer mode

- 8-bit interval timer operation (IRQTPG generation) enabling the clock source to be varied at 5 levels
- Square wave output to PPO pin

#### (b) Functions available in the PWM pulse generate mode

- 14-bit accuracy PWM pulse output to the PPO pin (Used as a digital-to-analog converter and applicable to tuning)
- Interrupt generation of fixed time interval ( $\frac{2^{15}}{f_x} = 5.46 \text{ ms}$  : at 6.0 MHz operation) \*

\* 7.81 ms at 4.19 MHz operation

If pulse output is not necessary, the PPO pin can be used as a 1-bit output port.

**Note** If the STOP mode is set while the timer/pulse generator is in operation, miss-operation may result. To prevent that from occurring, preset the timer/pulse generator to the stop state using its mode register.

**(2) Timer/pulse generator mode register (TPGM)**

The timer/pulse generator mode register (TPGM) is an 8-bit register to control timer/pulse generator operations. Its format is shown in Fig. 4-31.

The TPGM is set by the 8-bit memory manipulation instruction.

Bit 3 enables or disables the timer/pulse generator modulo register (MODH, MODL) contents to be transferred (reloaded) to the modulo latch and can be manipulated individually.

The timer/pulse generator operation can be stopped and current consumption can be decreased by setting the TPGM1 to "0".

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

**Fig. 4-31 Timer/Pulse Generator Mode Register Format**

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

**Timer/Pulse Generator Operating Mode Select Bit**

TPGM0	0	PWM pulse generate mode selected
	1	Timer mode selected

**Timer/Pulse Generator Operation Enable/Disable Bit**

TPGM1	0	Timer/pulse generator operation stopped
	1	Timer/pulse generator operation enabled

**Modulo Register Reload Enable/Disable Bit**

TPGM3	0	Modulo register reload disabled
	1	Modulo register reload enabled

**PPO Output Latch Data**

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

**PPO Pin Output Select Bit Static/Pulse**

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

**PPO Pin Output Enable/Disable Bit**

TPGM7	0	PPO pin output disabled (high impedance)
	1	PPO pin output enabled



**(3) Configuration and operation for use in the timer mode**

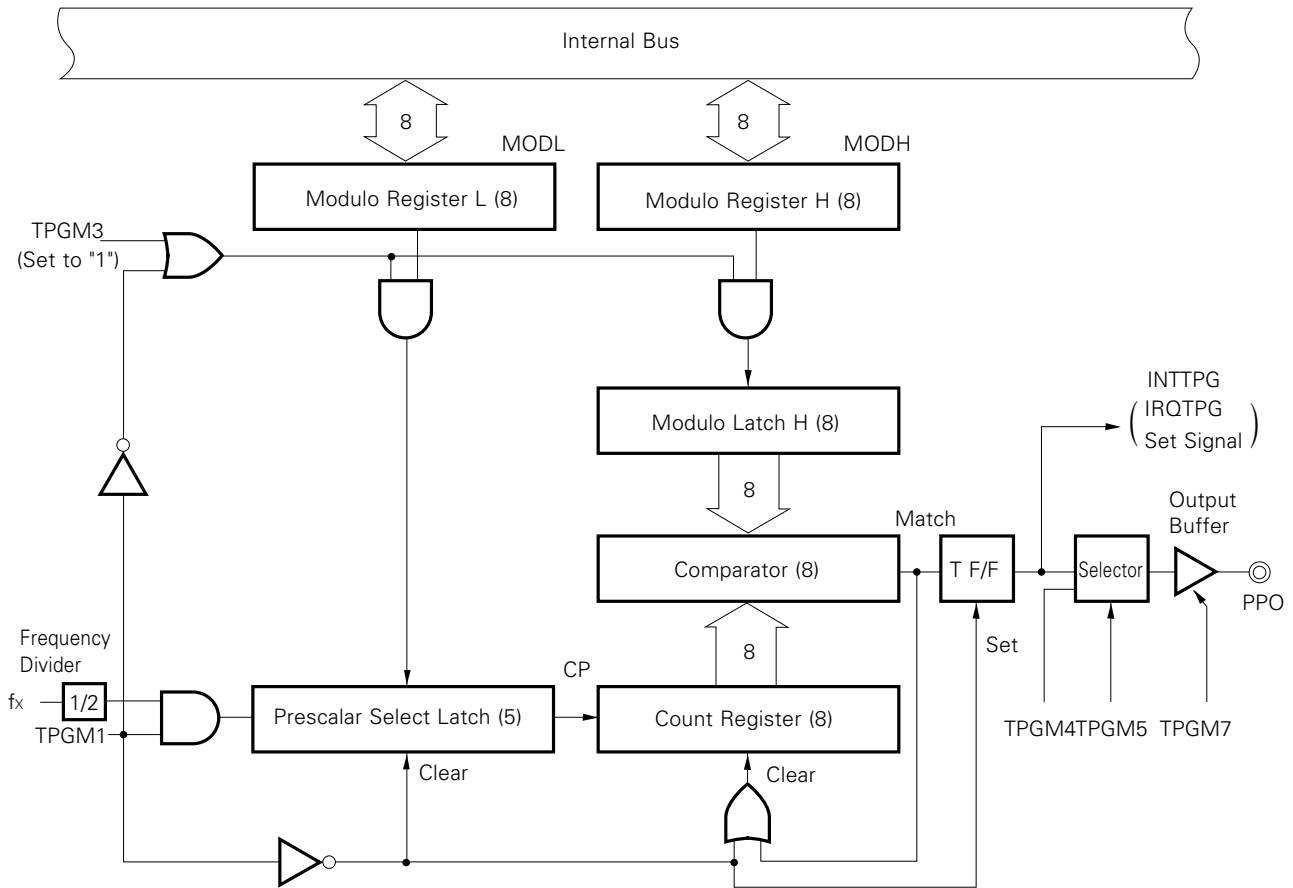
The timer/pulse generator configuration for use in the timer mode is shown in Fig. 4-32.

The timer mode is selected by setting TPGM bit 0 to "1". In the timer mode, enable modulo register reload by setting TPGM3 to "1".

In the timer mode, select the prescaler with modulo register L (MODL) and set the frequency or interrupt interval set value to modulo register H (MODH). Start the timer by resetting the TPGM1 from 0 to 1. The operation timing for MODH setting is shown in Fig. 4-33 and the frequency or interrupt interval setting is shown in Table 4-4.

Square wave output or static output to the PPO pin can be switched. In the case of square wave output, set TPGM5 to "1" and TPGM7 to "1".

**Fig. 4-32 Block Diagram of Timer/Pulse Generator (Timer Mode)**



**Note** If the timer is stopped in the timer operating mode, the IRQTPG may be set because the T F/F is set. Thus, when stopping the timer, do so with interruption disabled, and after the timer has stopped, clear the IRQTPG.

Fig. 4-33 Timer Mode Operation Timing

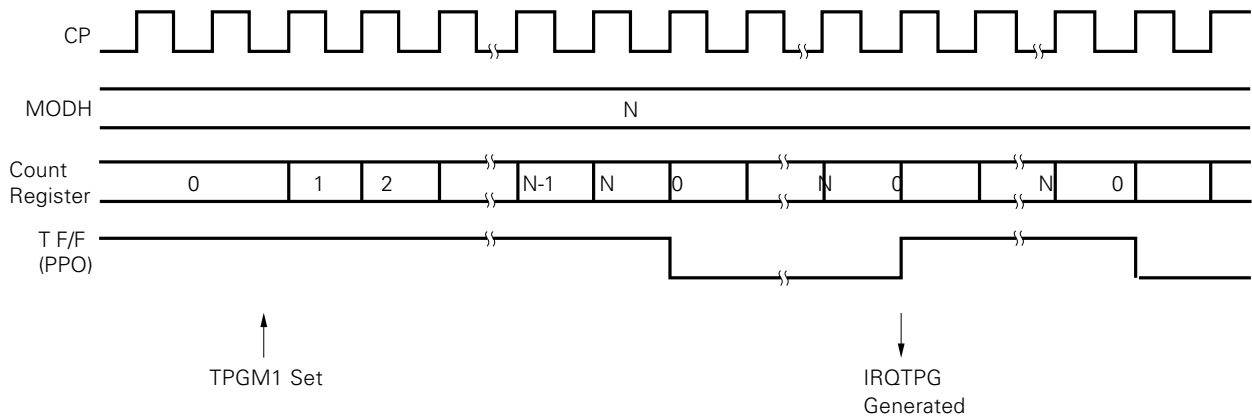


Table 4-4 Modulo Register Setting

(When  $f_x = 6.0 \text{ MHz}$ )

MODL Bits 2 to 6					Interrupt Generate Interval ( $f_x = 6.0 \text{ MHz}$ )	Square Wave Output Frequency ( $f_x = 6.0 \text{ MHz}$ )
6	5	4	3	2		
0	0	0	0	1	$\frac{256(N+1)}{f_x} = 85.3 \mu\text{s to } 10.9 \text{ ms}$	$\frac{f_x}{256(N+1)} = 91.6 \text{ Hz to } 11.7 \text{ kHz}$
0	0	0	1	0	$\frac{128(N+1)}{f_x} = 42.7 \mu\text{s to } 5.45 \text{ ms}$	$\frac{f_x}{128(N+1)} = 183 \text{ Hz to } 23.4 \text{ kHz}$
0	0	1	0	0	$\frac{64(N+1)}{f_x} = 21.3 \mu\text{s to } 2.73 \text{ ms}$	$\frac{f_x}{64(N+1)} = 366 \text{ Hz to } 46.9 \text{ kHz}$
0	1	0	0	0	$\frac{32(N+1)}{f_x} = 10.7 \mu\text{s to } 1.37 \text{ ms}$	$\frac{f_x}{32(N+1)} = 732 \text{ Hz to } 93.8 \text{ kHz}$
1	0	0	0	0	$\frac{16(N+1)}{f_x} = 5.33 \mu\text{s to } 683 \mu\text{s}$	$\frac{f_x}{16(N+1)} = 1465 \text{ Hz to } 188 \text{ kHz}$

(When  $f_x = 4.19 \text{ MHz}$ )

MODL Bits 2 to 6					Interrupt Generate Interval ( $f_x = 4.19 \text{ MHz}$ )	Square Wave Output Frequency ( $f_x = 4.19 \text{ MHz}$ )
6	5	4	3	2		
0	0	0	0	1	$\frac{256(N+1)}{f_x} = 122 \mu\text{s to } 15.6 \text{ ms}$	$\frac{f_x}{256(N+1)} = 64 \text{ Hz to } 8 \text{ kHz}$
0	0	0	1	0	$\frac{128(N+1)}{f_x} = 61.0 \mu\text{s to } 7.81 \text{ ms}$	$\frac{f_x}{128(N+1)} = 128 \text{ Hz to } 16 \text{ kHz}$
0	0	1	0	0	$\frac{64(N+1)}{f_x} = 30.5 \mu\text{s to } 3.91 \text{ ms}$	$\frac{f_x}{64(N+1)} = 256 \text{ Hz to } 32 \text{ kHz}$
0	1	0	0	0	$\frac{32(N+1)}{f_x} = 15.3 \mu\text{s to } 1.95 \text{ ms}$	$\frac{f_x}{32(N+1)} = 512 \text{ Hz to } 65 \text{ kHz}$
1	0	0	0	0	$\frac{16(N+1)}{f_x} = 7.63 \mu\text{s to } 977 \mu\text{s}$	$\frac{f_x}{16(N+1)} = 1024 \text{ Hz to } 131 \text{ kHz}$

- Note**
1. Only the above values can be set to MODL. Be sure to set "0" to bits 0, 1 and 7.
  2. N is the MODH set value. "0" cannot be set to N.  
Be sure to set a value in the range from 1 to 255 to N.

**(4) Configuration and operation for use in the PWM pulse generate mode**

The timer/pulse generator for use in the PWM pulse generate mode is shown in Fig. 4-34.

The PWM pulse generate mode is selected by setting TPGM0 to "0". Pulse output is enabled by setting TPGM5 and TPGM7 to "1". In the PWM mode, PWM pulse can be output from the PPO pin and the IRQTPG can be set at the fixed interval ( $2^{15}/f_x = 5.46 \text{ ms}$  : at 6.0 MHz operation)\*1.

The PWM pulse generated by the μPD75237 is an active-low, 14-bit accuracy pulse. This pulse is converted to an analog voltage by integrating it using an external low-pass filter and can be applied for electronic tuning and DC motor control. (Refer to **Fig. 4-35 Example of D/A Conversion Configuration with μPD75237.**)

The PWM pulse is generated by combining the fundamental period determined by  $2^{10}/f_x$  ( $171 \mu\text{s}$ : at 6.0 MHz operation)\*2 and the sub period of  $2^{15}/f_x$  ( $5.46 \text{ ms}$ : at 6.0 MHz operation)\*1 and the time constant of the external low-pass filter can be shortened.

The low-level width of the PWM pulse is determined by the 14-bit modulo latch value. The modulo latch value is determined as a result of transfer of MODH 8 bits to the most significant 8 bits of the modulo latch and MODL most significant 6 bits to the least significant 6 bits of the modulo latch.

The digital-to analog converted output voltage is given as

$$V_{AN} = V_{ref} \times \frac{\text{Modulo latch value}}{2^{14}}$$

where  $V_{ref}$ : External switching circuit reference voltage

In the μPD75237, all 14 bits can be transferred simultaneously to the modulo latch after correct data has been written to MODH and MODL by the 8-bit manipulation instruction. This aims at preventing the PWM from being generated with an unstable value in the process of modulo latch rewrite. This transfer is called "reload" and is controlled by TPGM3.

- Note**
1. Setting "0" to modulo register H (MODH) disables the PWM pulse generator to operate normally. Be sure to set to MODH a value in the range from 1 to 255.
  2. When the least significant 2 bits of modulo register L (MODL) are read, an undefined value is read.
  3. The fundamental period of the PWM pulse is  $2^{10}/f_x$  ( $171 \mu\text{s}$ : at 6.0 MHz operation)\*2. If the module latch is changed with a shorter period, the PWM pulse remains unchanged.

- \* 1. 7.81 ms at 4.19 MHz operation  
2. 244 μs at 4.19 MHz operation

**(5) Static output to the PPO pin**

If pulse output is not necessary, the PPO pin can be used for normal static output. In this case, set output data to TPGM4 with TPGM5 and TPGM7 set to "0" and "1", respectively.

Fig. 4-34 Timer/Pulse Generator Block Diagram (PWM Pulse Generate Mode)

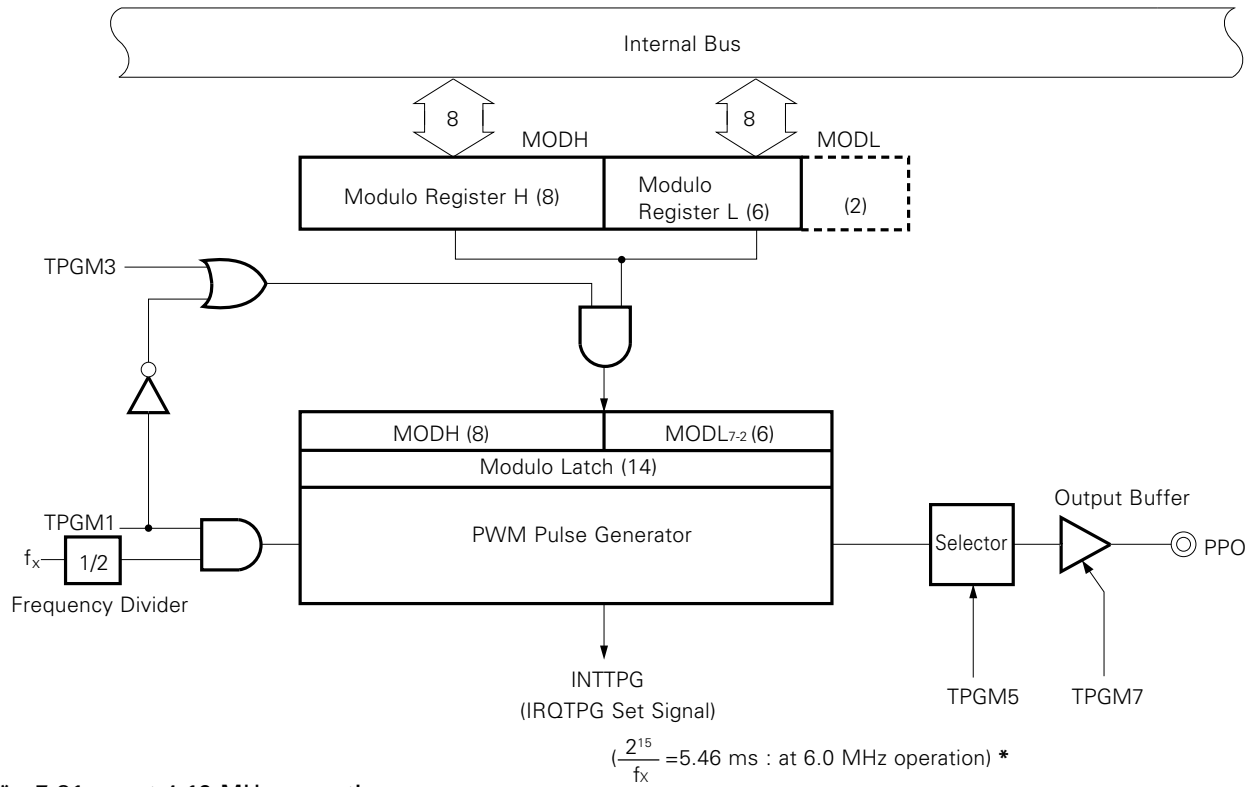
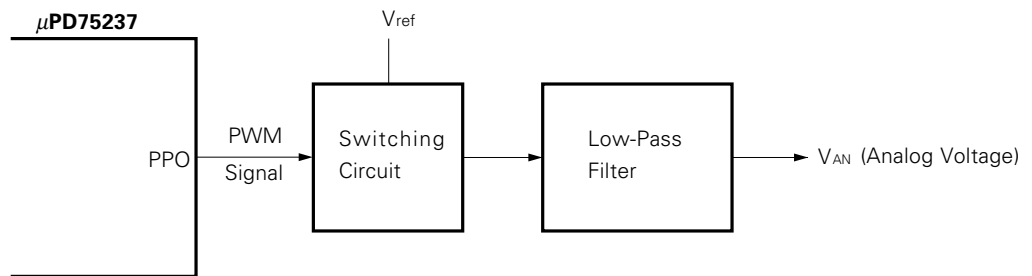


Fig. 4-35 Example of D/A Conversion Configuration with μPD75237

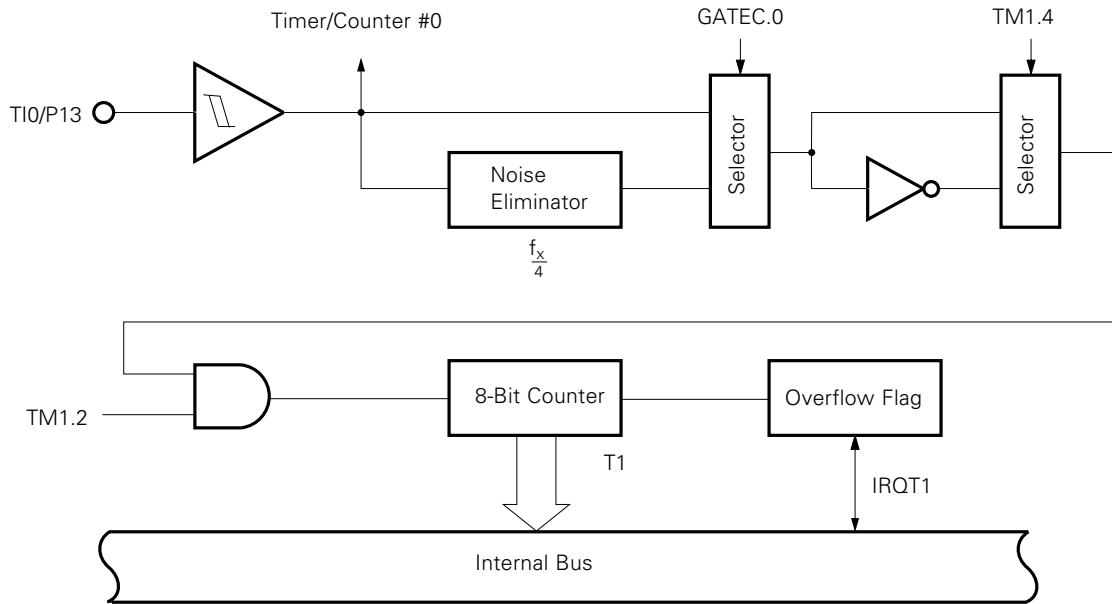


4.8 EVENT COUNTER

(1) Event counter configuration

The event counter of the μPD75237 incorporates a noise eliminator and has a configuration shown in Fig. 4-36.

Fig. 4-36 Event Counter Block Diagram



**Note** TI0/P13 pin is an external event pulse input pin which serves as timer/event counter #0 and event counter #1.

(2) Event counter functions

The event counter has the following functions.

- (a) Event counter operation
- (b) Count state read function
- (c) Count pulse edge specification
- (d) Noise eliminating function

**(3) Event counter mode register**

The event counter mode register (TM1) is an 8-bit register to control the event counter. Its format is shown in Fig. 4-37.

TM1 is set by an 8-bit memory manipulation instruction.

Bit 3 is an event counter start bit and can be set independently. When the counter starts operating, bit 3 is automatically reset to "0".

**Fig. 4-37 Event Counter Mode Register Format**

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	Count operation stopped (with count value held)
	1	Count operation enabled

**Event Counter Start Command Bit**

TM13	Writing "1" clears the counter and IRQT1 flag. If TM12 is "1", count operation starts.
------	--

**Count Pulse Edge Specification**

TM14	0	T10 input rising edge
	1	T10 input falling edge

**(4) Overflow flag (IRQT1)**

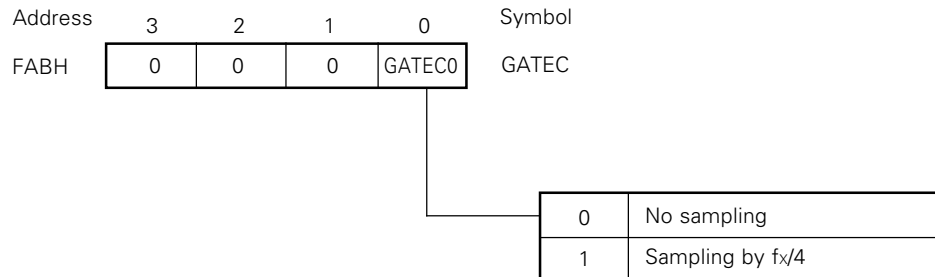
The overflow flag is a flag which is set (1) by an overflow of the event counter count register and is cleared (0) by a count operation start command.

**(5) Event counter control register (GATEC)**

This is a register to select sampling with a sampling clock ( $f_x/4$ ). A pulse having a smaller width than that of two sampling clock cycles ( $8/f_x$ ) is eliminated as noise by a noise eliminator and a pulse having a width larger than that of the sampling clock is securely acknowledged as an interrupt signal.

Its format is shown in Fig. 4-38.

**Fig. 4-38 Event Counter Control Register Format**



4.9 SERIAL INTERFACE

The μPD75237 incorporates two channels of clocked 8-bit serial interfaces. Table 4-5 gives differences between channel 0 and channel 1.

**Table 4-5 Differences between Channels 0 and 1**

Serial Transfer Mode and Function		Channel 0	Channel 1
3-wire serial I/O	Clock selection	$f_x/2^4$ , $f_x/2^3$ , TOUT F/F, external clock	$f_x/2^4$ , $f_x/2^3$ , external clock
	Transfer mode	MSB first/LSB first switchable	MSB first
	Transfer end flag	Serial transfer end interrupt request flag (IRQCSI0)	Serial transfer end flag (EOT)
2-wire serial I/O	Use enabled	None	
Serial bus interface			

**(1) Serial interface (channel 0) functions**

The following four modes are available for the μPD75237 serial interface (channel 0). The functions of each mode are outlined below.

**• Operation stop mode**

This is the mode used when no serial transfer is performed. Low power consumption operation is possible in this mode.

**• 3-wire serial I/O mode**

8-bit data is transferred using three lines of serial clock ( $\overline{\text{SCK0}}$ ), serial output (SO0) and serial input (SI0).

The 3-wire serial I/O mode enables simultaneous transmission/reception, thus shortening the data transfer processing time.

Since the start bit of 8-bit data for serial transfer can be switched between MSB and LSB, channel 0 can be connected to a device having either start bit.

In the 3-wire serial I/O mode, channel 0 can be connected to the 75X series, 78K series and various types of peripheral I/O devices.

**• 2-wire serial I/O mode**

8-bit data is transferred using two lines of serial clock ( $\overline{\text{SCK0}}$ ) and serial data bus (SB0 or SB1).

Communication is possible with two or more devices by controlling the level of output to the two lines by software.

Since the output level of  $\overline{\text{SCK0}}$  and SB0 (or SB1) can be controlled by software, any transfer format is applicable. Thus, the number of handshake lines previously required to connect two or more devices can be decreased and so the input/output ports can be used efficiently.

**• SBI mode (serial bus interface mode)**

This mode enables communication with two or more devices with two lines of serial clock ( $\overline{\text{SCK0}}$ ) and serial data bus (SB0 or SB1).

This mode is compliant with the NEC serial bus format.

In the SBI mode, the transmitter can output an "address" for selection of a serial communication target device on the serial data bus, a "command" to provide instructions to the target device and actual "data".

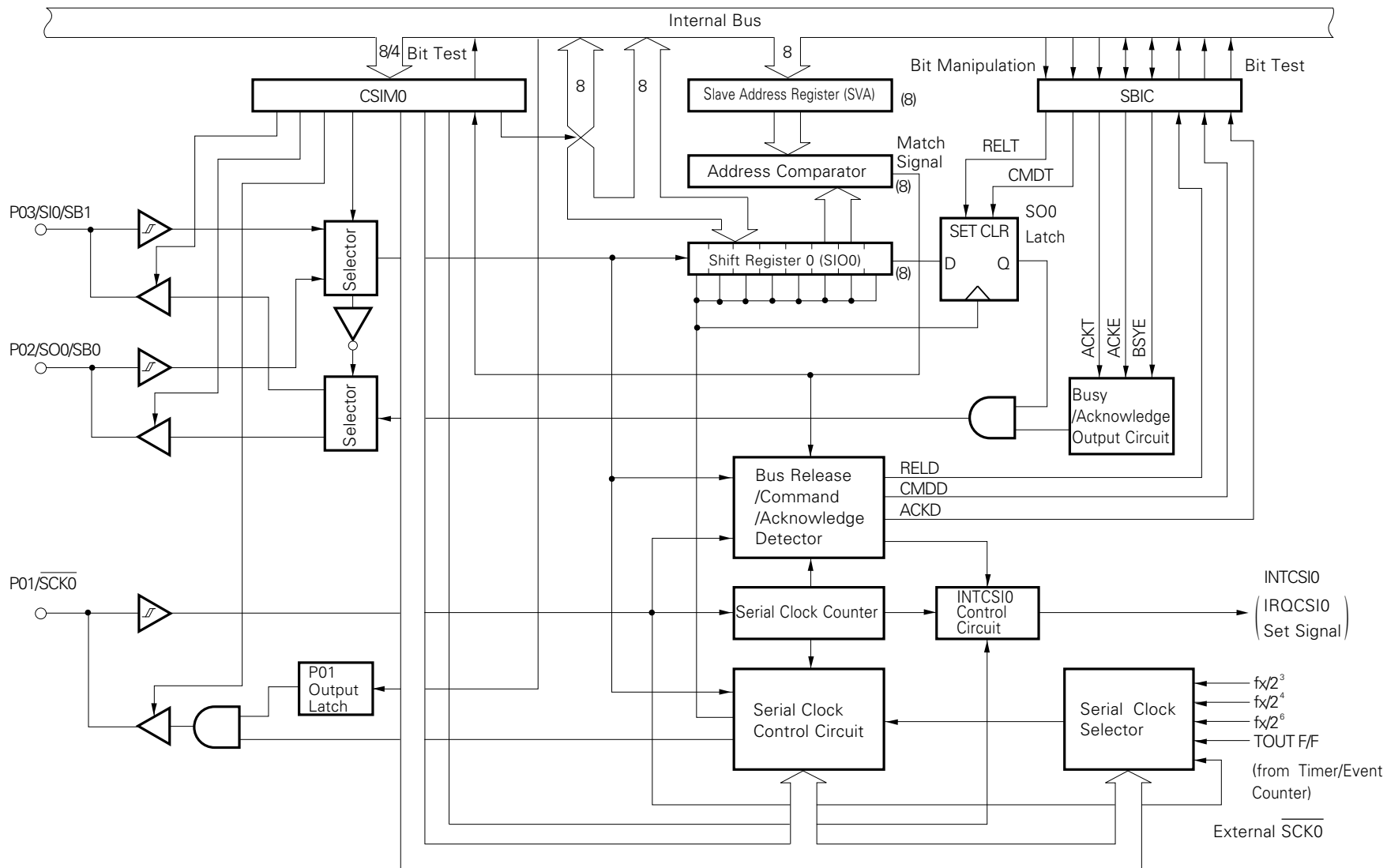
The receiver can distinguish between "address", "command" and "data" by hardware. As in the 2-wire serial I/O mode, this function enables the input/output ports to be used efficiently and the serial interface control portions of any applied program to be simplified.

**(2) Serial interface (channel 0) configuration**

Fig. 4-39 is a block diagram of serial interface (channel 0).



Fig. 4-39 Serial Interface (Channel 0) Block Diagram



**(3) Serial interface (channel 0) register functions**

**(a) Serial operating mode register 0 (CSIM0)**

Fig. 4-40 shows a serial operating mode register 0 (CSIM0) format.

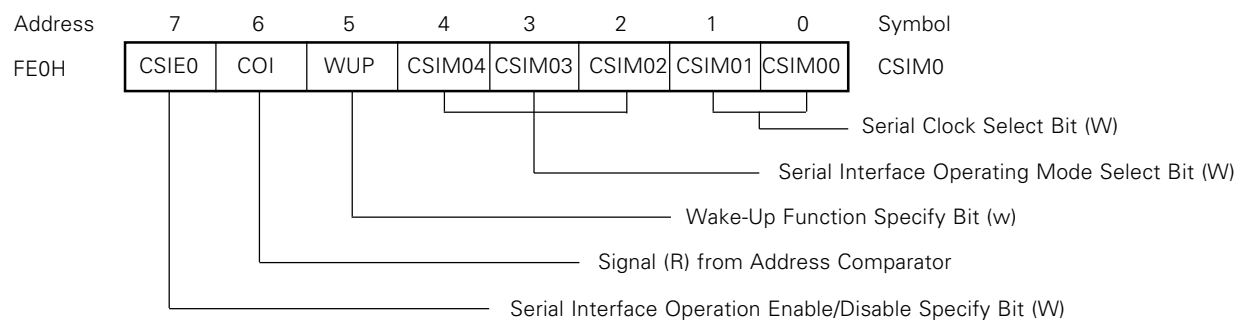
CSIM0 is an 8-bit register to specify the serial interface (channel 0) operating mode, serial clock and the wake-up function.

An 8-bit memory manipulation instruction is used for CSIM0 operations. The higher 3 bits can be manipulated in 1-bit units. Use each bit name for bit manipulation.

Read/write operation is enabled/disabled depending on the bit (refer to Fig. 4-40). Bit 6 is only enabled for test and the written data is invalidated.

RESET input clears all bits to 0.

**Fig. 4-40 Serial Operating Mode Register 0 (CSIM0) Format (1/3)**



- Remarks**
1. (R) : Read only
  2. (W) : Write only

Fig. 4-40 Serial Operating Mode Register 0 (CSIM0) Format (2/3)

**Serial Clock Select Bit (W)**

CSIM01	CSIM00	Serial Clock			SCK0 Pin Mode
		3-Wire Serial I/O Mode	SBI Mode	2-Wire Serial I/O Mode	
0	0	Input clock to SCK0 pin from outside.			Input
0	1	Timer/event counter output (T0)			Output
1	0	$f_x/2^4$ (375 kHz, or 262 kHz) *		$f_x/2^6$ (93.8 kHz, or 65.5 kHz) *	
1	1	$f_x/2^3$ (750 kHz, or 524 kHz) *			

\* Values in parentheses are when  $f_x = 6.0$  MHz or when  $f_x = 4.19$  MHz.

**Serial Interface Operating Mode Select Bit (W)**

CSIM04	CSIM03	CSIM02	Operating Mode	Bit Order of Shift Register 0	SO0 Pin Function	SIO Pin Function
×	0	0	3-wire serial I/O mode	SIO <sub>7-0</sub> ↔XA (transferred with MSB first)	SO0/P02 (CMOS output)	SIO/P03 (input)
		1		SIO <sub>0-7</sub> ↔XA (transferred with LSB first)		
0	1	0	SBI mode	SIO <sub>7-0</sub> ↔XA (transferred with MSB first)	SB0/P02 (N-ch open drain input/output)	P03 input
1						
0	1	1	2-wire serial I/O mode	SIO <sub>7-0</sub> ↔XA (transferred with MSB first)	SB0/P02 (N-ch open drain input/output)	P03 input
1						

Remarks × : Don't care

**Wake-Up Function Specify Bit (W)**

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

**Note** When WUP = 1 is set during  $\overline{\text{BUSY}}$  signal output,  $\overline{\text{BUSY}}$  is not released. In SBI,  $\overline{\text{BUSY}}$  signal continues to be output up to the falling edge of the next serial clock (SCK0) after  $\overline{\text{BUSY}}$  release. Ensure to set WUP = 1 after releasing  $\overline{\text{BUSY}}$  and confirming that the SB0 (or SB1) pin has become high level.

**Fig. 4-40 Serial Operating Mode Register 0 (CSIMO) Format (3/3)**

**Signal (R) from Address Comparator**

	Clear Condition (COI = 0)	Set Condition (COI = 1)
COI*	When the slave address register (SVA) data unmatches the shift register 0 data.	When the slave address register (SVA) data matches the shift register 0 data.

\* COI read is only valid before serial transfer and after its completion. Only undefined value is read during transfer. The COI data written by an 8-bit manipulation instruction is ignored.

**Serial Interface Operation Enable/Disable Specify Bit (W)**

		Shift Register 0 Operation	Serial Clock Counter	IRQCSI0 Flag	SO0/SB0, SI0/SB1 Pins
CSIE0	0	Shift operation disabled	Clear	Hold	Dedicated to port 0 functions
	1	Shift operation enabled	Count operation	Settable	Functions in each mode and operations with port 0

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

CSIE0	CSIM03	CSIM02	Operating Mode
0	×	×	Operation stop mode
1	0	×	3-wire serial I/O mode
1	1	0	SBI mode
1	1	1	2-wire serial I/O mode

**2.** P01/SCK0 pin becomes as follows depending on the settings of CSIE0, CSIM01 and CSIM00.

CSIE0	CSIM01	CSIM00	P01/SCK0 Pin Status
0	0	0	Input port
1	0	0	High impedance
0	0	1	High-level output
0	1	0	
0	1	1	
1	0	1	Serial clock output (high-level output)
1	1	0	
1	1	1	

**Remarks** 3. Clear CSIE0 during serial transfer using the following procedure.

- ① Disable interrupt by clearing the interrupt enable flag.
- ② Clear CSIE0.
- ③ Clear the interrupt request flag.

**Example** 1. Select  $fx/2^4$  for serial clock and generate serial interrupt IRQCSI0 upon termination of each serial transfer and select a serial transfer mode in the SBI mode using the SB0 pin as serial data bus.

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

2. Enable serial transfer in accordance with the CSIM0 contents.

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

(b) **Serial bus interface control register (SBIC)**

Fig. 4-41 shows a serial bus interface control register (SBIC) format.

SBIC is an 8-bit register which consists of a serial bus control bit and flags indicating various statuses of input data received from the serial bus.

SBIC is manipulated using a bit manipulation instruction.

It cannot be manipulated using a 4-bit or 8-bit manipulation instruction.

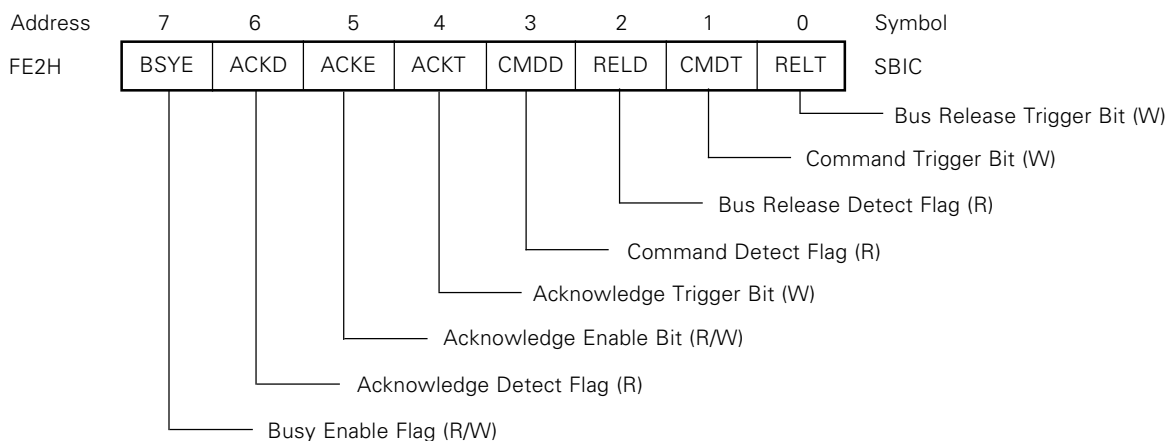
Read/write operation enable/disable depends on the bit (refer to Fig. 4-41).

RESET input clears all bits to 0.

**Note** Only the following bits can be used in the 3-wire and 2-wire serial I/O modes.

- Bus release trigger bit (RELT) ..... SO0 latch set
- Command trigger bit (CMDT) ..... SO0 latch clear

**Fig. 4-41 Serial Bus Interface Control Register (SBIC) Format (1/3)**



- Remarks**
1. (R) Only read
  2. (W) Only write
  3. (R/W) Read/write enabled

Fig. 4-41 Serial Bus Interface Control Register (SBIC) Format (2/3)

**Bus Release Trigger Bit (W)**

RELT	Bus release signal (REL) trigger output control bit. When set (RELT = 1), SO0 latch is set (1) and then the RELT bit is automatically cleared (0).
------	--

**Note** Do not clear SB0 (or SB1) during serial transfer. Be sure to do so before transfer start or after transfer end.

**Command Trigger Bit (W)**

CMDT	Command signal (CMD) trigger output control bit. When set (CMDT = 1), SO0 latch is cleared (0) and then the CMDT bit is automatically cleared (0).
------	--

**Note** Do not clear SB0 (or SB1) during serial transfer. Be sure to do so before transfer start or after transfer end.

**Bus Release Detect Flag (R)**

	Clearing Conditions (RELD = 0)	Setting Conditions (RELD = 1)
RELD	<ul style="list-style-type: none"> <li>① Transfer start instruction execution</li> <li>② RESET input</li> <li>③ CSIE0 = 0 (refer to Fig. 4-40)</li> <li>④ SVA and SIO0 mismatch upon address reception.</li> </ul>	Bus release signal (REL) detection

**Command Detect Flag (R)**

	Clearing Conditions (CMDD = 0)	Setting Conditions (CMDD = 1)
CMDD	<ul style="list-style-type: none"> <li>① Transfer start instruction execution</li> <li>② Bus release signal (REL) detection</li> <li>③ RESET input</li> <li>④ CSIE0 = 0 (refer to Fig. 4-40)</li> </ul>	Command signal (CMD) detection

**Acknowledge Trigger Bit (W)**

ACKT	Setting this bit after termination of transfer outputs $\overline{ACK}$ in synchronization with the next $\overline{SCK0}$ . After output of $\overline{ACK}$ signal, this bit is automatically cleared (0).
------	--

**Note**

1. Do not set (1) this bit during serial transfer.
2. ACKT cannot be cleared by software.
3. When setting ACKT, set ACKE = 0.

**Acknowledge Enable Bit (R/W)**

ACKE	0	Automatic output of acknowledge signal $\overline{ACK}$ is disabled (output by ACKT enabled).	
	1	When set before termination of transfer	$\overline{ACK}$ is output in synchronization with the 9th clock of $\overline{SCK0}$ .
		When set after termination of transfer	$\overline{ACK}$ is output in synchronization with $\overline{SCK0}$ just after execution of a set instruction.

Fig. 4-41 Serial Bus Interface Control Register (SBIC) Format (3/3)

**Acknowledge Detect Flag (R)**

	Clearing Conditions (ACKD = 0)	Setting Conditions (ACKD = 1)
ACKD	① Transfer start instruction execution ② RESET input	Acknowledge signal ( $\overline{ACK}$ ) detection (at the rising edge of $\overline{SCK0}$ )

**Busy Enable Bit (R/W)**

BSYE	0	① Busy signal automatic output disabled ② Busy signal output stopped at the falling edge of $\overline{SCK0}$ just after clear instruction execution.
	1	Busy signal output at the falling edge of $\overline{SCK0}$ following the acknowledge signal.

**Example 1.** Output the command signal.

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

**2.** Identify the receive data type by testing RELD and CMDD for proper processing.

Set WUP = 1 for this interrupt routine so that processing is carried out only in the case of a match address.

```
SEL    MB15
SKF    RELD    ; RELD test
BR     !ADRS
SKT    CMDD    ; CMDD test
BR     !DATA
```

CMD : ..... ; Command interpret

DATE : ..... ; Data processing

ADRS : ..... ; Address decode



(c) **Shift register 0 (SIO0)**

Fig. 4-42 shows a shift register 0 peripheral configuration. SIO0 is an 8-bit register which executes parallel-to-serial conversion and carries out serial transmission/reception (shift operation) in synchronization with a serial clock.

Serial transfer is started by writing data to SIO0.

In transmission, the data written to SIO0 is output to the serial output (SO0) or serial data bus (SB0/SB1).

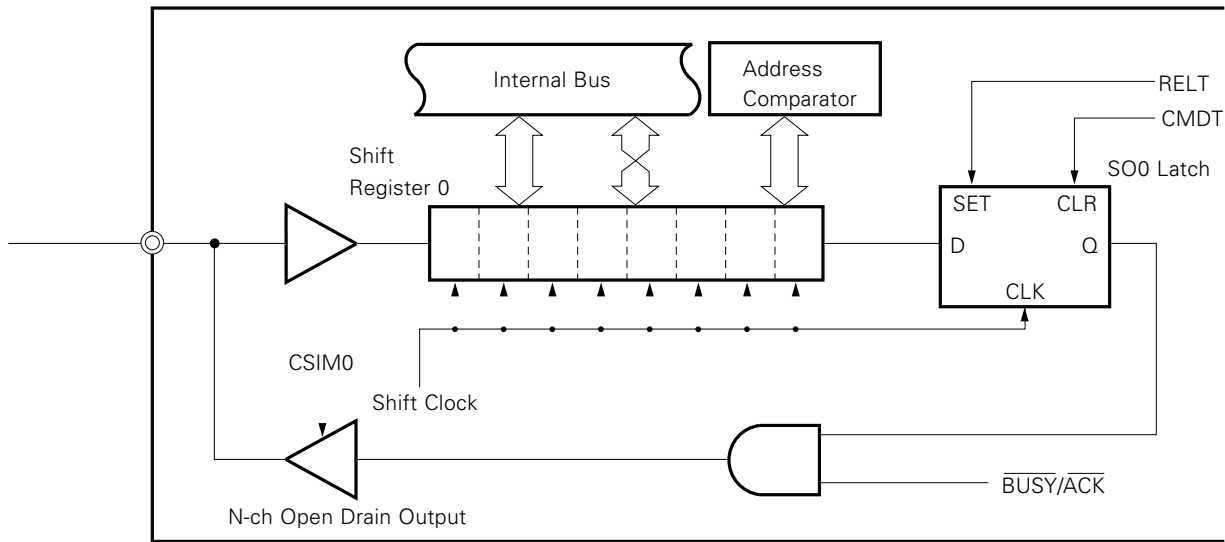
In reception, data is read from the serial input (SI0) or SB0/SB1 to SIO0.

This register can be read/written by an 8-bit manipulation instruction.

$\overline{\text{RESET}}$  input during operation makes the SIO0 value undefined.  $\overline{\text{RESET}}$  input in the standby mode holds the SIO0 value.

Shift operation stops after 8-bit transmission /reception.

**Fig. 4-42 Shift Register 0 peripheral Configuration**



SIO0 read and serial transfer start (write) are enabled at the following timings.

- Serial interface operation enable/disable bit (CSIE0) = 1 except when CSIE0 is set to "1" after data write to the shift register.
- When the serial clock is masked after 8-bit serial transfer.
- When  $\overline{\text{SCK0}}$  is at a high level

Be sure to write/read data to SIO0 when  $\overline{\text{SCK0}}$  is at a high level.

In the 2-wire serial I/O or SBI mode, the data bus has a configuration that the input pins serve as output pins and vice versa. Each output pin has an N-ch open drain configuration.

Thus, set FFH to SIO0 for the device for data reception.

**(d) Slave address register (SVA)**

The slave address register (SVA) has the following two functions.

Only write is enabled for the SVA by an 8-bit manipulation instruction.

$\overline{\text{RESET}}$  input makes the SVA value undefined.  $\overline{\text{RESET}}$  input in the standby mode holds the SVA value.

- **Slave address detection**

- [SBI mode]**

Use this mode to connect the  $\mu$ PD75237 as a slave device to the serial bus. The SVA is an 8-bit register for the slave to set the slave address value (own specification number). The master outputs a slave address for particular slave selection to the connected slave. These two data (slave address and SVA values output from the master) are compared by an address comparator. When they match, the slave has been selected.

In this case, bit 6 (COI) of the serial operating mode register 0 (CSIM0) is set to "1".

**Note 1. The slave selection or non-selection status is checked by detecting the matching of the slave address received after bus release (RELD = 1).**

**Use the address match interrupt (IRQCSI0) to be normally generated with WUP = 1 to detect the matching. Thus, detect selection or non-selection by slave address when WUP = 1.**

**2. If selection or non-selection is to be detected without using an interrupt when WUP = 0, do so by transmitting/receiving the command preset by a program without using the method of detecting address matching.**

- **Error detection**

- [2-wire serial I/O and SBI modes]**

When an address, a command and data are to be transmitted using the  $\mu$ PD75237 as the master device or data is to be transmitted using the  $\mu$ PD75237 as the slave device, the SVA detects errors.

**(4) Various types of signals**

Table 4-6 gives a list of various types of signals. Figs. 4-43 to 4-48 show the various types of signals and flag operation.

Table 4-6 Various Types of Signals in SBI Mode (1/2)

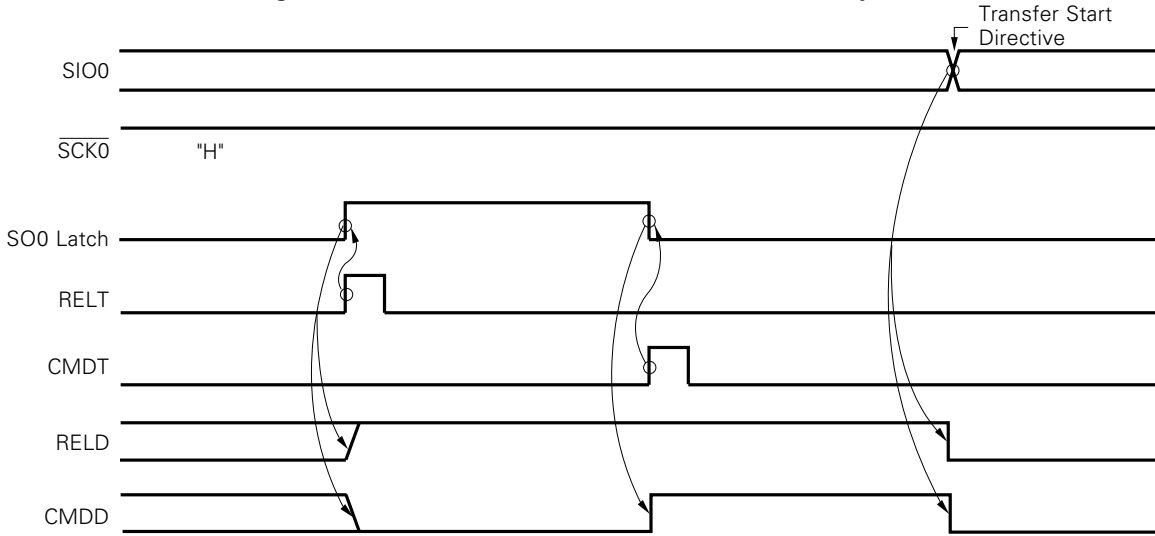
Signal Name	Output Device	Definition	Timing Chart	Output Condition	Effect on Flag	Meaning of Signal
Bus release signal (REL)	Master	Rising edge of SB0/SB1 when $\overline{SCK0} = 1$		• RELT set	• RELD set • CMDD clear	CMD signal is output to indicate that transmit data is an address.
Command signal (CMD)	Master	Falling edge of SB0/SB1 when $\overline{SCK0} = 1$		• CMDT set	• CMDD set	i) Transmit data is an address after REL signal output ii) No REL signal output. Transmit data is a command.
Acknowledge signal ( $\overline{ACK}$ )	Master/slave	Low-level signal to be output to SB0/SB1 during one-clock period of $\overline{SCK0}$ after completion of serial reception	<p>[Synchronous Busy Output]</p>	① ACKE = 1 ② ACKT set	• ACKD set	Completion of reception
Busy signal ( $\overline{BUSY}$ )	Slave	[Synchronous busy signal] Low-level signal to be output to SB0/SB1 following the acknowledge signal		• BSYE = 1	—	Serial reception disabled because of processing
Ready signal (READY)	Slave	High-level signal to be output to before serial transfer start or after its completion		① BSYE = 0 ② Execution of an instruction for data write to SIO0 (transfer start command)	—	Serial reception enabled

Table 4-6 Various Types of Signals in SBI Mode (2/2)

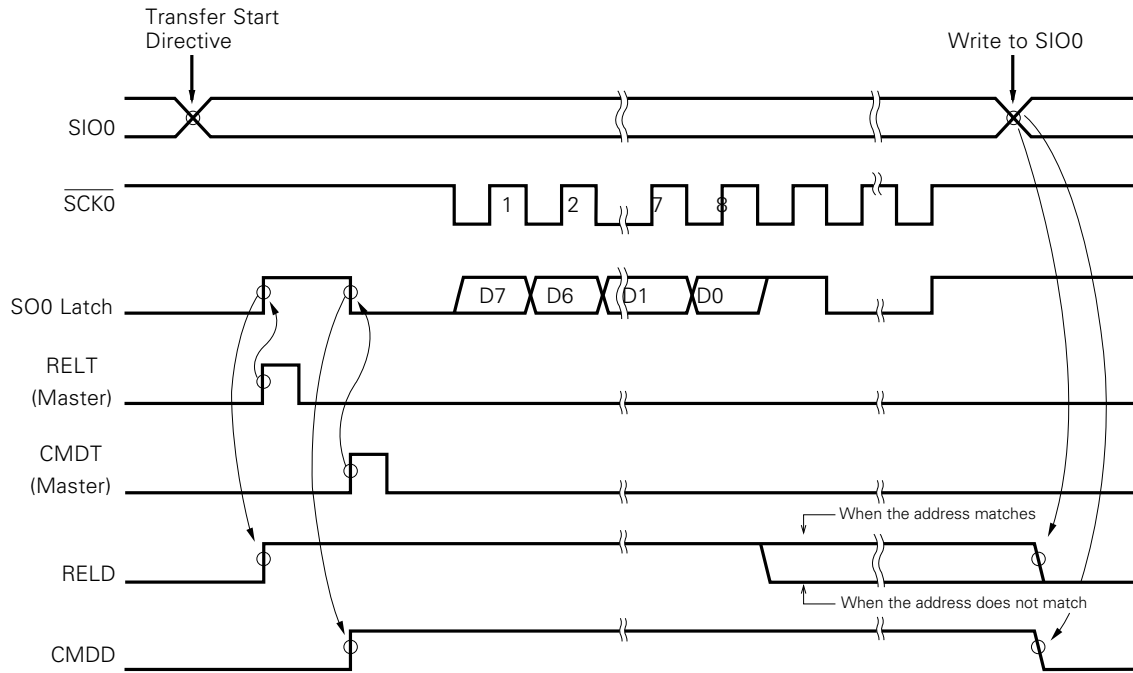
Signal Name	Output Device	Definition	Timing Chart	Output Condition	Effect on Flag	Meaning of Signal
Serial clock (SCK0)	Master	Synchronous clock to output address, command, data, ACK signal and synchronous BUSY signal. Address, command and data are transferred by the first eight clocks.		Execution of an instruction for data write to SIO0 when CSIE0 = 1 (serial transfer start command)*2	IRQCSI0 set (rising edge of 9th clock) *1	Timing of signal output to the serial data bus
Address (A7 to 0)	Master	8-bit data to be transferred in synchronization with SCK0 after output of REL and CMD signals				Address value of slave device on the serial bus
Command (C7 to 0)	Master	8-bit data to be transferred in synchronization with SCK0 after output of CMD signal only without REL signal output				Command and message for the slave device
Data (D7 to 0)	Master/slave	8-bit data to be transferred in synchronization with SCK0 without output of REL and CMD signals				Numeric value to be processed by a slave or master device

- \* 1. When WUP = 0, IRQCSI0 is always set at the rising edge of the 9th clock of SCK0. When WUP = 1, an address is received. Only when the received address matches the slave address register (SVA) value, IRQCSI0 is set at the rising edge of the 9th clock of SCK0.
2. Transfer starts after the BUSY state is changed to the READY state.

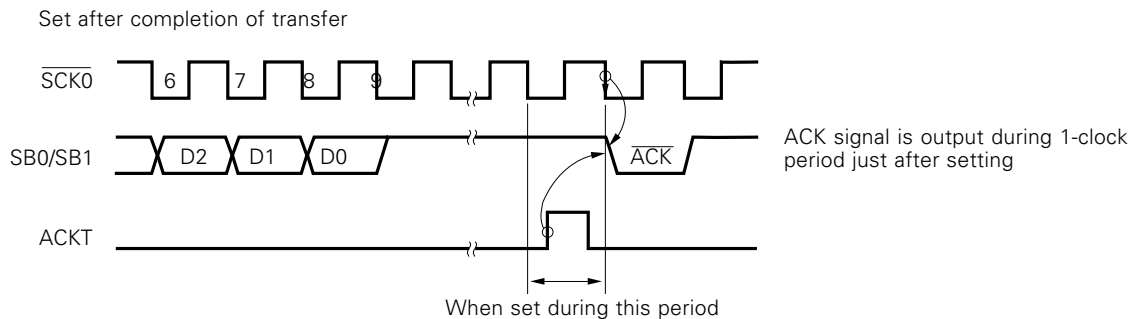
**Fig. 4-43 RELT, CMDT, RELD and CMDD (Master) Operations**



**Fig. 4-44 RELT, CMDT, RELD and CMDD (Slave) Operations**



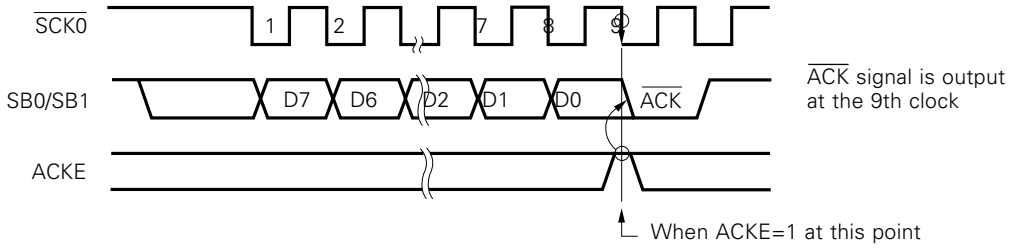
**Fig. 4-45 ACKT Operations**



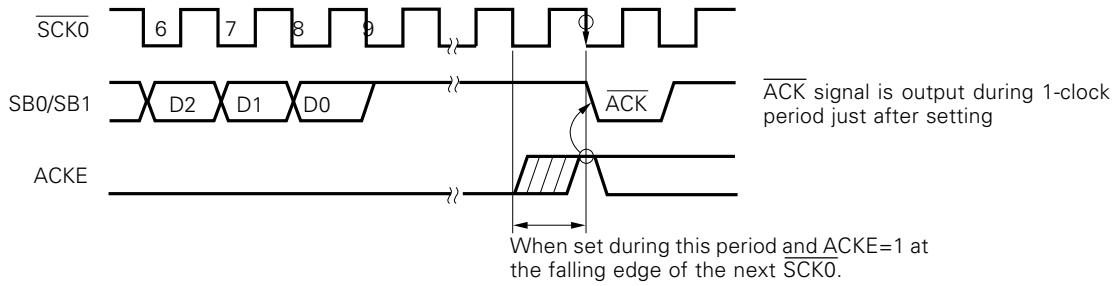
**Note** Do not set ACKT just before termination of transfer.

Fig. 4-46 ACKE Operation

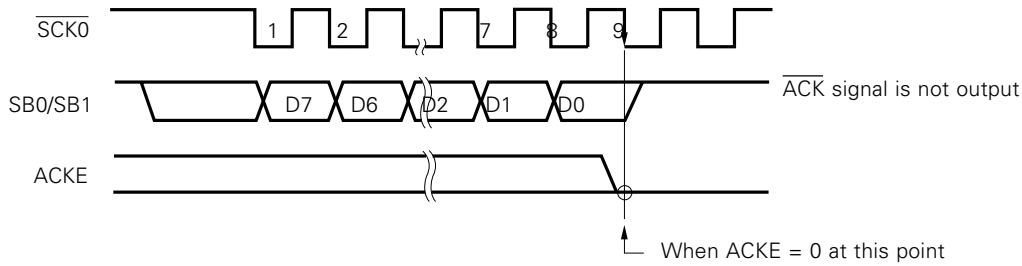
(a) When ACKE = 1 upon completion of transfer



(b) When set after completion of transfer



(c) When ACKE = 0 upon completion of transfer



(d) When the ACKE = 1 period is short

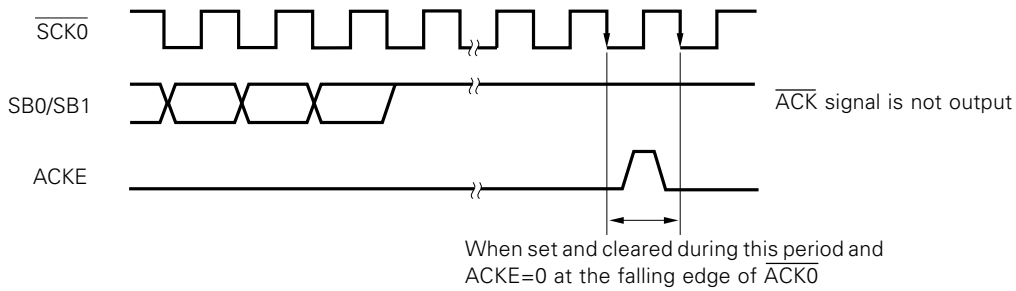
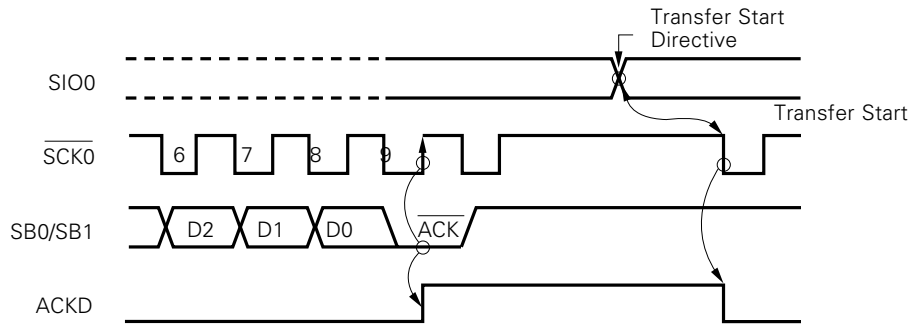
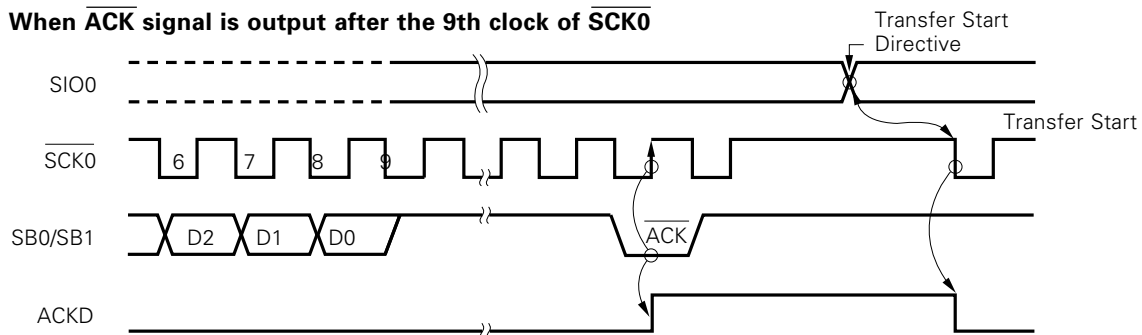


Fig. 4-47 ACKD Operations

(a) When  $\overline{\text{ACK}}$  signal is output during the 9th clock period of  $\overline{\text{SCK0}}$ .



(b) When  $\overline{\text{ACK}}$  signal is output after the 9th clock of  $\overline{\text{SCK0}}$



(c) Clear timing with transfer start command during BUSY

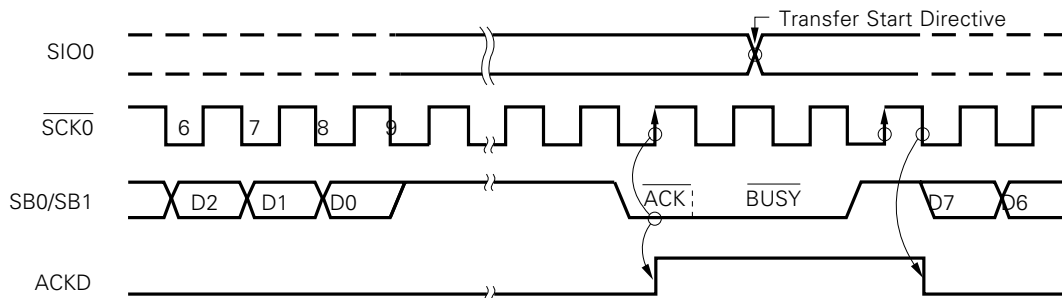
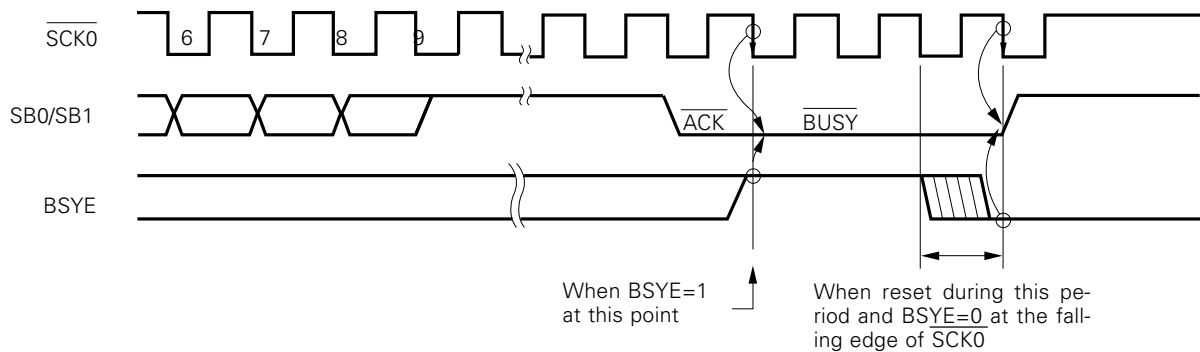


Fig. 4-48 BSYE Operation



(5) Serial interface (channel 0) operations

(a) Operation stop mode

The operation stop mode is used when serial transfer is not carried out. Power consumption is decreased in this mode.

In this mode, shift register 0 does not carry out shift operation and thus can be used as a normal 8-bit register.

RESET input sets the operation stop mode. The P02/SO0/SB0 pin and P03/SI0/SB1 pins are fixed to the input port. P01/SCK0 can be used as an input port by setting serial operating mode register 0.

(b) 3-wire serial I/O mode operations

The 3-wire serial I/O mode allows connection with the methods employed with another 75X series and 78K series.

Communication is carried out using three lines of serial clock (SCK0), serial output (SO0) and serial input (SI0).

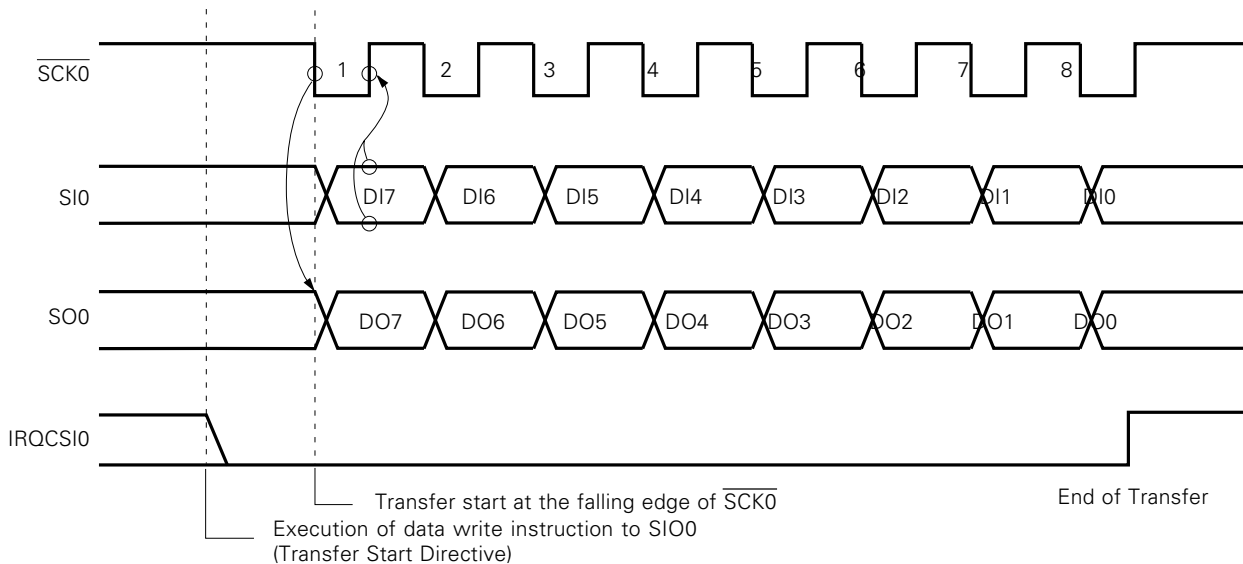
(i) Communication

The 3-wire serial I/O mode is used for data transmission and reception in 8-bit units. Bit-wise data transmission/reception is carried out in synchronization with the serial clock.

Shift operation of shift register 0 is carried out at the falling edge of serial clock (SCK0). Transmit data is held at the SO0 latch and output from the SO0 pin. Receive data input to the SI0 pin is latched to the shift register 0 at the rising edge of SCK0.

Shift register 0 operation automatically stops upon termination of 8-bit transfer and the interrupt request flag (IRQCSI0) is set.

Fig. 4-49 3-Wire Serial I/O Mode Timing





The S00 pin serves as CMOS output to output the S00 latch status. Thus, the S00 pin output status can be manipulated by setting the RELT and CMDT bits.

However, do not carry out this manipulation during serial transfer.

The  $\overline{SCK0}$  pin can control the output status by manipulating the P01 output latch in the output mode (internal system clock mode) (refer to 4.9 (7) **SCK0 pin output manipulation**).

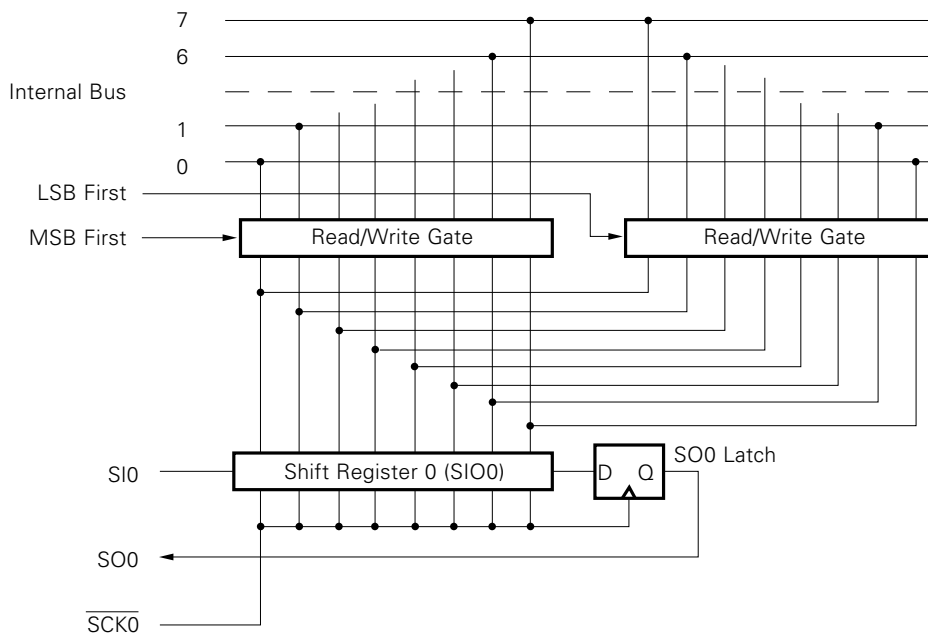
(ii) **MSB/LSB first switching**

The 3-wire serial I/O mode has a function which allows MSB-first or LSB-first transfer to be selected.

Fig. 4-50 shows shift register 0 (SIO0) and internal bus configurations. As shown in Fig. 4-50, MSB/LSB can be reversed and read/written.

MSB/LSB first switching can be specified by bit 2 of serial operating mode register 0 (CSIM0).

**Fig. 4-50 Transfer Bit Switching Circuit**



First bit switching is realized by switching the bit order of data write to the shift register 0 (SIO0). The SIO0 shift order remains the same.

Thus, switch the MSB/LSB first bit before writing data to the shift register 0.

**(c) 2-wire serial I/O mode operations**

The 2-wire serial I/O mode can be applied to any communication format by program.

Communication is basically carried out using two lines of serial clock ( $\overline{SCK0}$ ) and serial data input/output (SB0 or SB1).

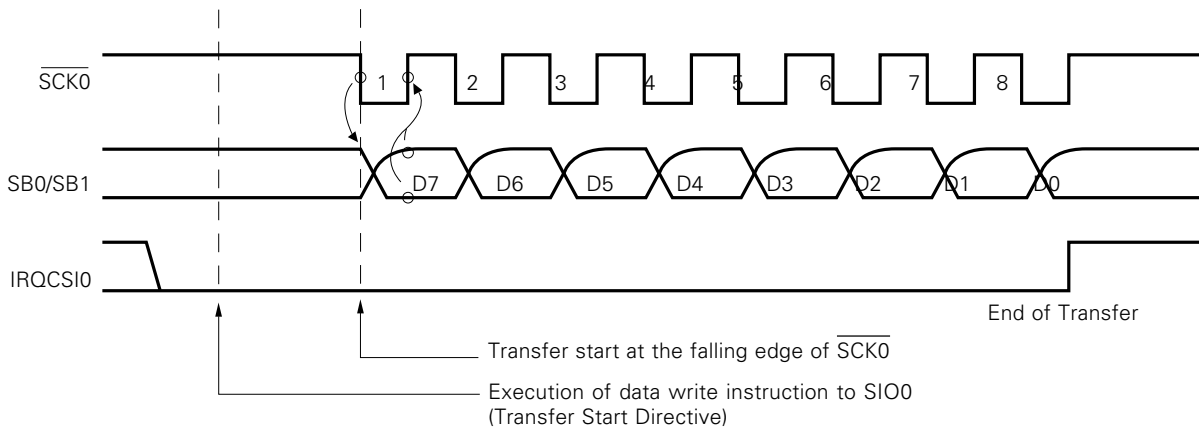
**(i) Communication**

The 2-wire serial I/O mode is used for data transmission and reception in 8-bit units. Bit-wise data transmission/reception is carried out in synchronization with the serial clock.

Shift operation of shift register 0 is carried out at the falling edge of serial clock ( $\overline{SCK0}$ ). Transmit data is held at the SO0 latch and output from the SB0/P02 (or SB1/P03) pin with MSB set as the first bit. Receive data input from the SB0 (or SB1) pin at the  $\overline{SCK0}$  rising edge is latched to the shift register 0.

Upon termination of 8-bit transfer, the shift register 0 operation automatically stops and the interrupt request flag (IRQCSI0) is set.

**Fig. 4-51 2-Wire Serial I/O Mode Timing**



Since the pin specified for the serial data bus of the SB0 (or SB1) pin becomes an N-ch open drain input/output, it must be pulled up externally.

Since the SB0 (or SB1) pin outputs the SO0 latch status, the SB0 (or SB1) pin status can be manipulated by setting the RELT and CMDT bits.

However, do not carry out this operation during serial transfer.

The  $\overline{SCK0}$  pin can control the output status by manipulating the P01 output latch in the output mode (internal system clock mode) (refer to **4.9 (7)  $\overline{SCK0}$  pin output manipulation**).

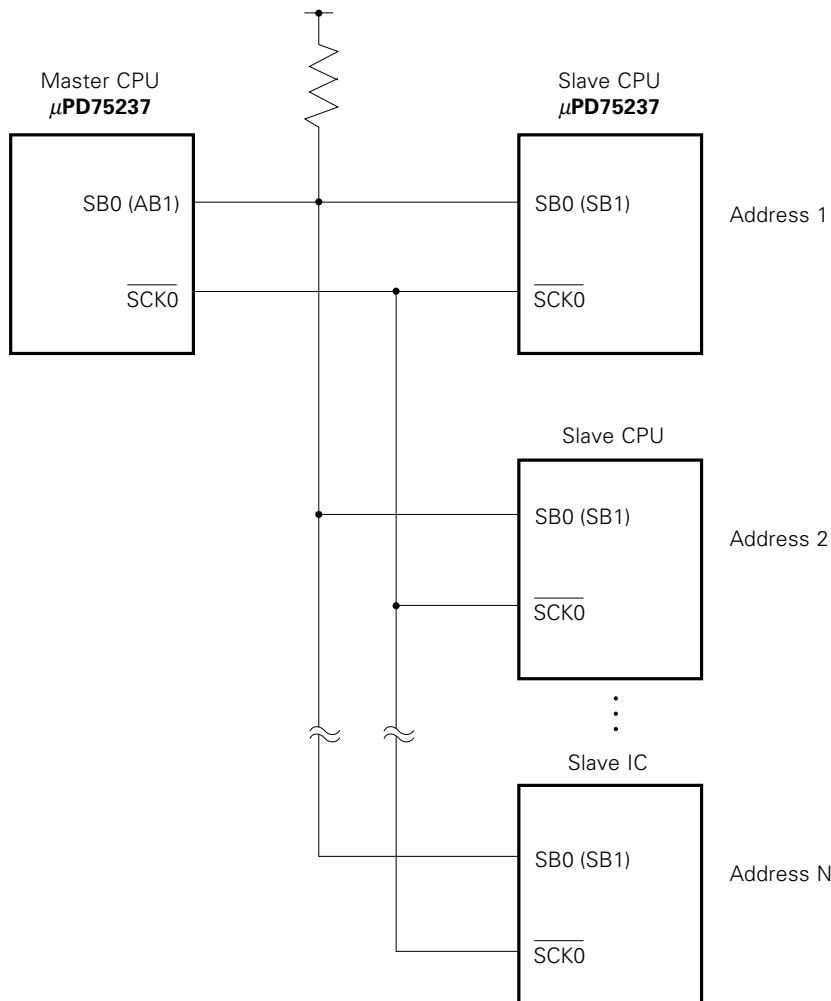
**(d) SBI mode operations**

SBI (serial bus interface) is a high-speed serial interface method compliant with the NEC serial bus format.

SBI is a single master high-speed serial bus based on the format with bus configuration functions added to the clocked serial synchronization I/O method so that communication can be carried out with two or more devices using two signal conductors. Thus, the number of ports used and that of wires on the board can be decreased for serial bus configuration with two or more microcomputers and peripheral ICs.

Fig. 4-52 shows the SBI system configuration example.

**Fig. 4-52 SBI System Configuration Example**

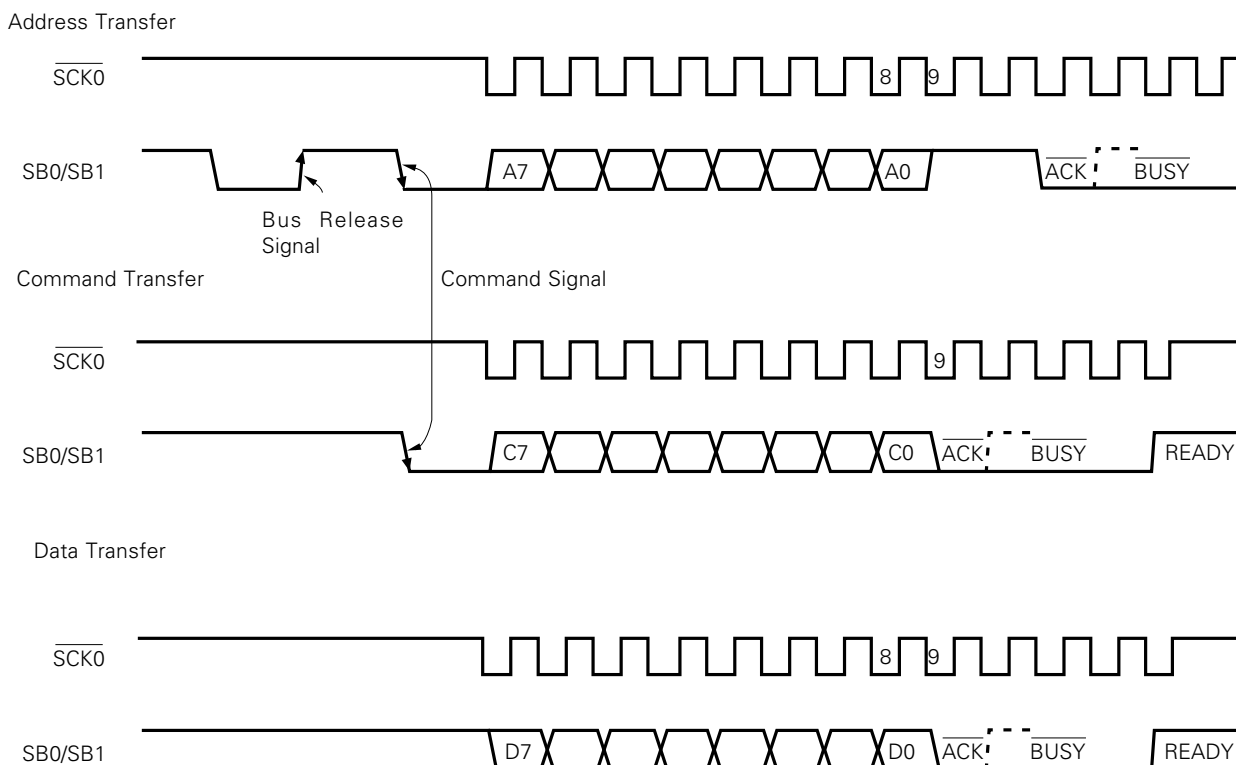


- Note 1.** Because in the SBI the serial data bus pin SB0 (or SB1) is an open drain output, the serial data bus line is wired-OR. A pull-up resistor is necessary for the serial data bus line.
- 2.** For master/slave replacement, a pull-up resistor is necessary for SCK0 because serial clock line (SCK0) input/output switching is executed asynchronously between the master and slave.

(i) **SBI functions**

- Address/command/data identification  
SBI distinguishes serial data between address, command and data.
- Chip select function by address  
The master executes slave chip selection by address transmission.
- Wake-up function  
The slave can easily make an address receive judgment (chip select judgment) using the wake-up function (which can be set/cancelled by software).  
When the wake-up function is set, an interrupt (IRQCSI0) is generated upon reception of a match address. Thus, when communication is carried out with two or more devices, CPUs except the selected slave can operate irrespective of serial communication.
- Acknowledge signal ( $\overline{\text{ACK}}$ ) control function  
Acknowledge signal is controlled to confirm serial data reception.
- Busy signal ( $\overline{\text{BUSY}}$ ) control function  
The busy signal is controlled to inform the slave busy status.

**Fig. 4-53 SBI Transfer Timing**



**(ii) Communication**

In the SBI, the master normally selects one slave device for communication target from among two or more devices by outputting an "address" to the serial bus.

After the communication target device has been determined, serial communication is achieved through command and data transmission/reception between the master and slave devices.

Figs. 4-54 to 4-57 show the timing charts of data communication.

In the SBI mode, shift operation of shift register 0 is carried out at the falling edge of serial clock ( $\overline{SCK0}$ ) and transmit data is output from the SB0/P02 or SB1/P03 pin with MSB as the first bit. Receive data input to the SB0 (or SB1) pin at the rising edge of  $\overline{SCK0}$  is latched to the shift register 0.

Fig. 4-54 Address Transmission from Master Device to Slave Device (WUP = 1)

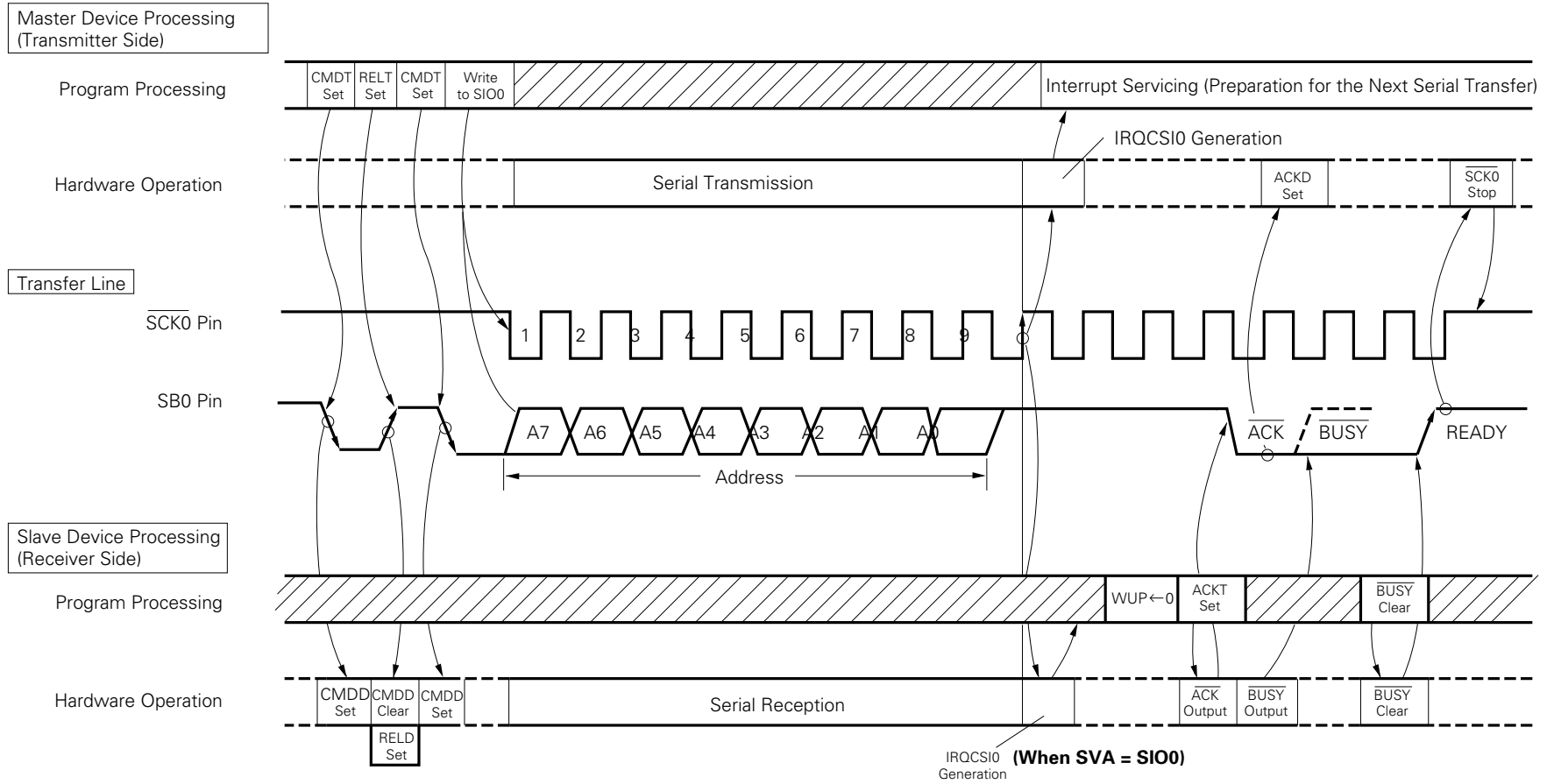


Fig. 4-55 Command Transmission from Master Device to Slave Device

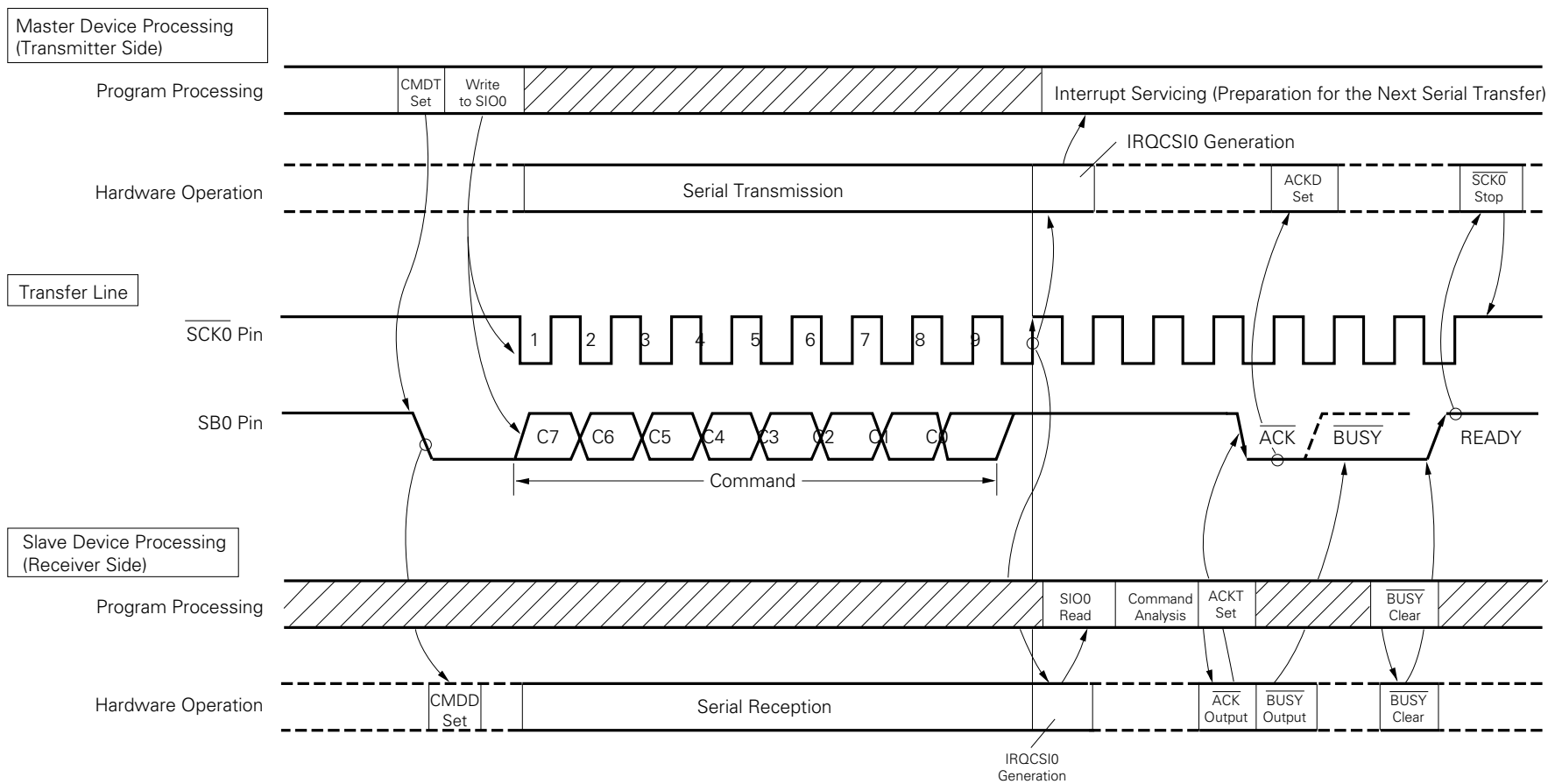


Fig. 4-56 Data Transmission from Master Device to Slave Device

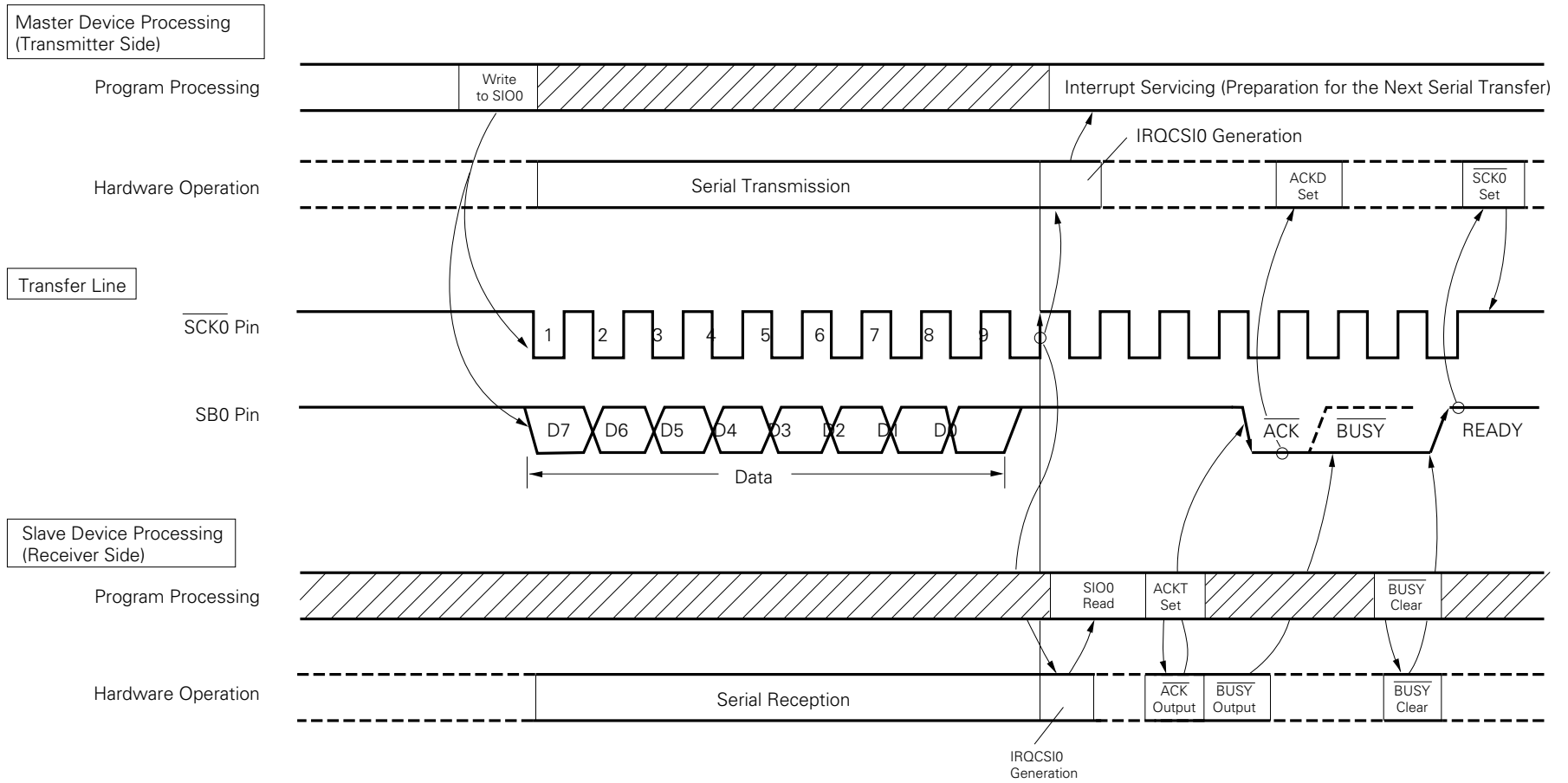
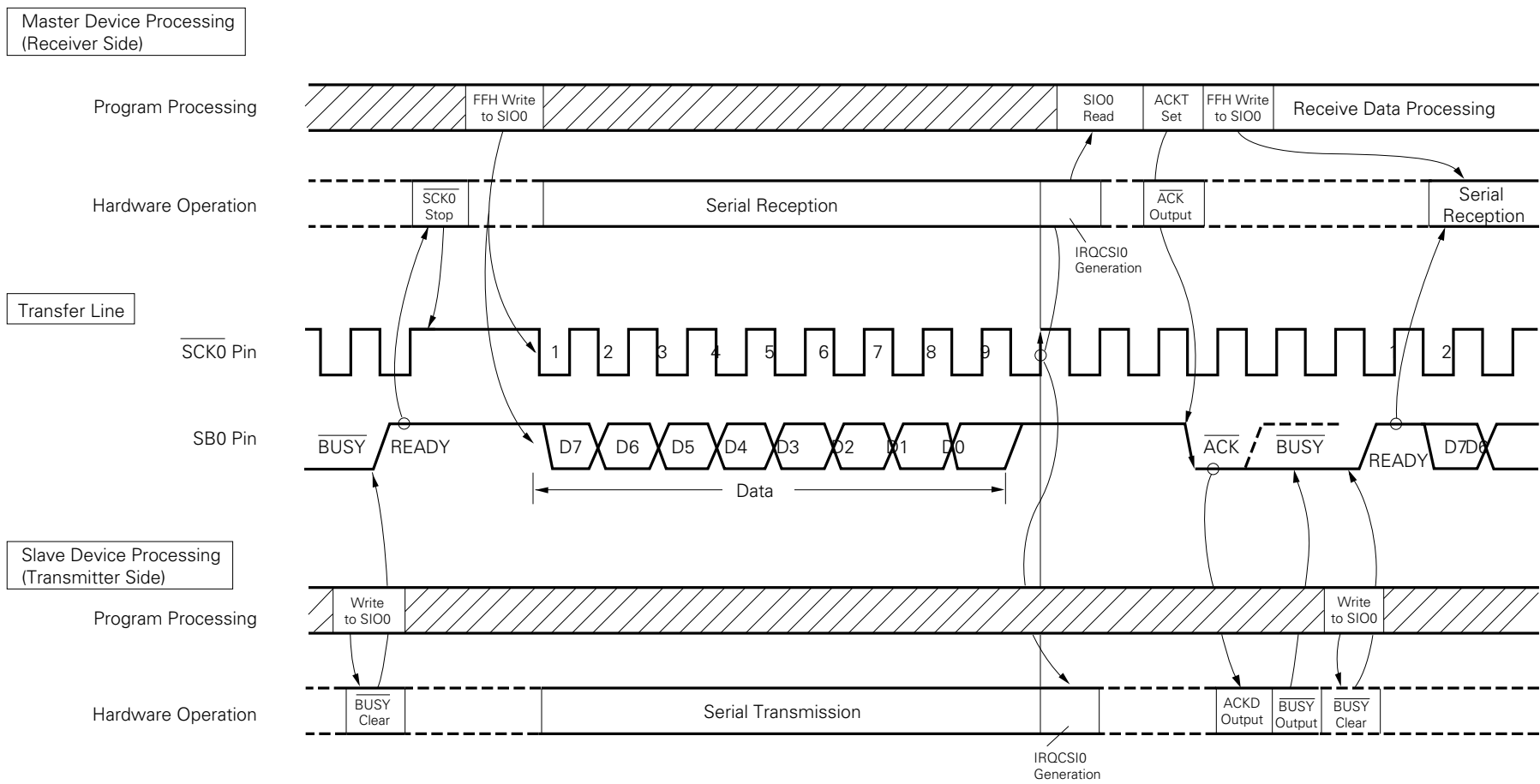




Fig. 4-57 Data Transmission from Slave Device to Master Device★



**(6) Transfer start in each mode**

In each of the 3-wire and 2-wire serial I/O modes and the SBI mode, serial transfer is started by setting transfer data to the shift register 0 (SIO0) under the following two conditions.

- Serial interface operation enable/disable bit (CSIE0) = 1
- The internal serial clock has stopped or  $\overline{\text{SCK0}}$  is at high level after 8-bit serial transfer.

**Note** Transfer does not start if CSIE0 is set to "1" after data is written to the shift register 0.

Serial transfer automatically stops and the interrupt request flag (IRQCSI0) is set upon termination of 8-bit transfer.

**[2-wire serial I/O mode transfer start precautions]**

**Note** Because it is necessary to turn off the N-ch transistor upon data reception, write FFH to SIO0 in advance.

**[SBI mode transfer start precautions]**

- Note**
1. Because it is necessary to turn off the N-ch transistor upon data reception, write FFH to SIO0 in advance. However, in the case of wake-up function specify bit (WUP) = 1, the N-ch transistor remains OFF. Thus, it is not necessary to write FFH to SIO0 before reception.
  2. If data is written to SIO0 when the slave is busy, the written data is not lost. Transfer starts when the busy status is cancelled and the SB0 (or SB1) input becomes high level (ready status).

**Example** The RAM data specified by the HL register is transferred to SIO0 and simultaneously the SIO0 data is fetched into the accumulator and serial transfer is started.

```
MOV  XA, @HL    ; Transmit data is fetched from the RAM.
SEL  MB15      ; Or CLR1 MBE
XCH  XA, SIO0   ; Transmit data is exchanged with receive data and transfer is started.
```

**(7)  $\overline{SCK0}$  pin output manipulation**

Because the  $\overline{SCK0}/P01$  pin incorporates an output latch, static output is possible by software in addition to normal serial clocks.

P01 output latch manipulation enables to set any number of  $\overline{SCK0}$  by software (SO0/SB0/SB1 pin is controlled by the RELT and CMDT bits of SBIC).

$\overline{SCK0}/P01$  pin output manipulation is described below.

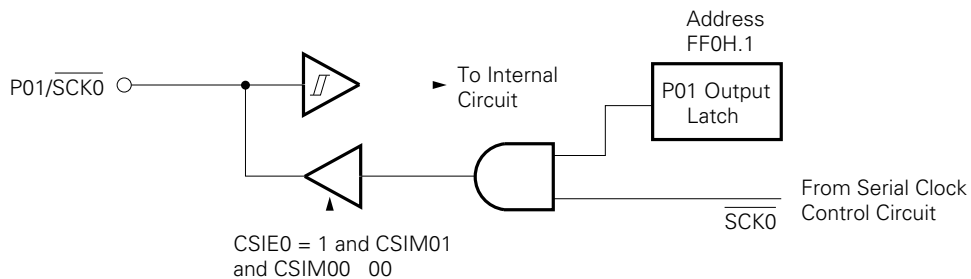
- ① Set the serial operating mode register 0 (CSIM0) ( $\overline{SCK0}$  pin: output mode, serial operation: enabled). While serial transfer is stopped,  $\overline{SCK0}$  from the serial clock control circuit remains 1.
- ② Manipulate the P01 output latch by a bit manipulation instruction.

**Example** 1 clock output to  $\overline{SCK0}/P01$  pin by software.

```

SEL   MB15           ; Or CLR1 MBE
MOV   XA,#10000011B ;  $\overline{SCK0}(f_x/2^3)$ , output mode
MOV   CSIM0,XA
CLR1  OFF0H.1       ;  $\overline{SCK0}/P01 \leftarrow 0$ 
SET1  OFF0H.1       ;  $\overline{SCK0}/P01 \leftarrow 1$ 
    
```

**Fig. 4-58  $\overline{SCK0}/P01$  Pin Configuration**



The P01 output latch is mapped at bit 1 of address FF0H.  $\overline{RESET}$  signal generation sets the P01 output latch to "1".

- Note**
1. It is necessary to set the P01 output latch to 1 during normal serial transfer.
  2. The P01 output latch address cannot be set by "PORT0.1" as shown below. Describe address (0FF0H.1) directly for the operand.

However, it is necessary to preset MBE = 0 or (MBE = 1 and MBS = 15) for instruction execution.

```

CLR1  PORT0.1 } Use disabled
SET1  PORT0.1 }
CLR1  OFF0H.1 } Use enabled
SET1  OFF0H.1 }
    
```

**(8) Serial interface (channel 1) functions**

The following two modes are available to the  $\mu$ PD75237 serial interface (channel 1).  
The summary of each mode is shown below.

**• Operation stop mode**

The operation stop mode is used when serial transfer is not carried out. Power consumption is decreased in this mode.

**• 3-wire serial I/O mode**

8-bit data transfer is carried out using three lines of serial clock ( $\overline{\text{SCK1}}$ ), serial output (SO1) and serial input (SI1).

In the 3-wire serial I/O mode which enables simultaneous transmission and reception, the data transfer rate is improved.

★

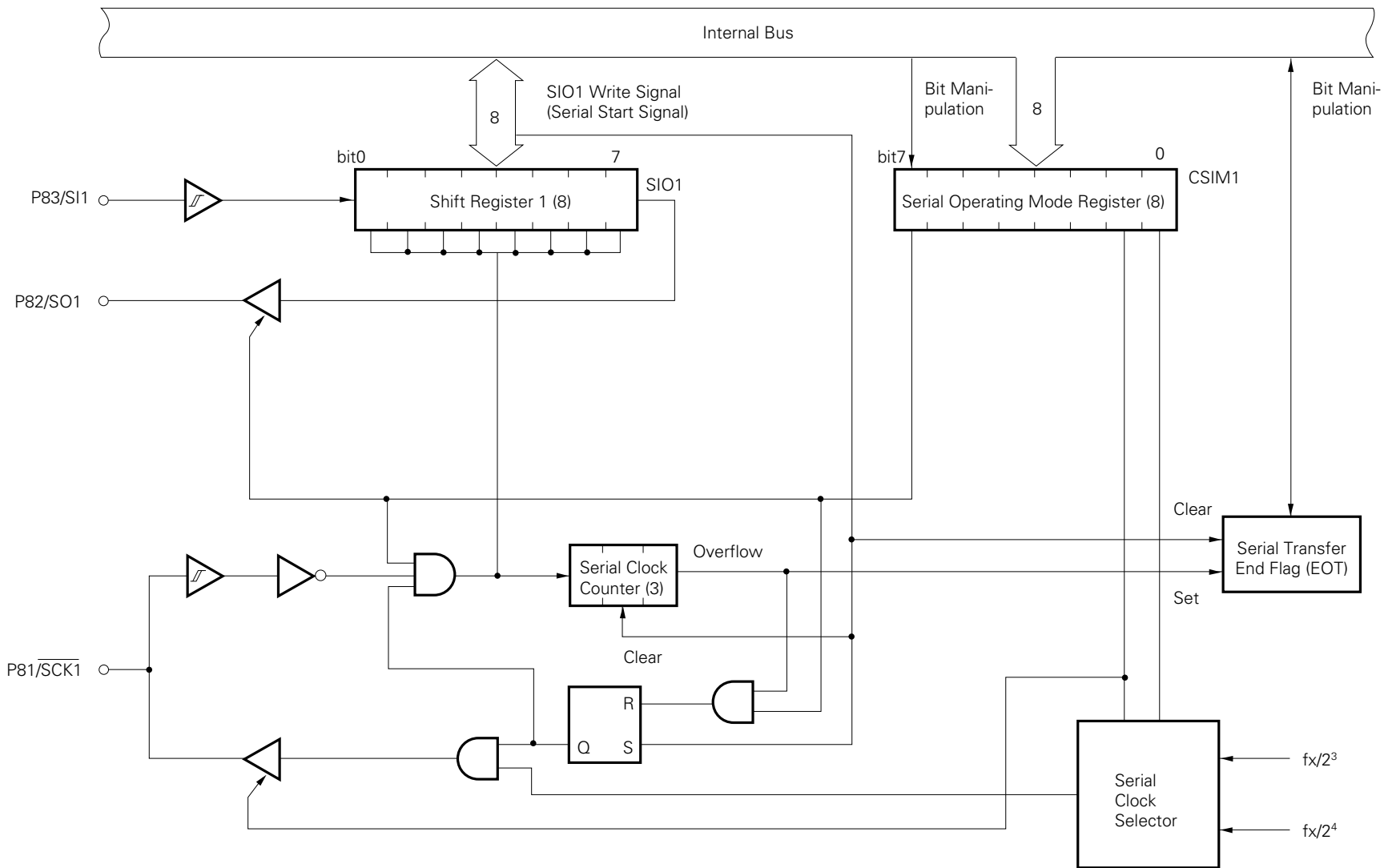
The first bit of 8-bit data for serial transfer is fixed to MSB.

In the 3-wire serial I/O mode, channel 1 can be connected to the 75X series, 78K series and various types of peripheral I/O devices.

**(9) Serial interface (channel 1) configuration**

Fig. 4-59 shows a serial interface (channel 1) block diagram.

Fig. 4-59 Serial Interface (Channel 1) Block Diagram



(10) Serial interface (channel 1) register functions

(a) Serial operating mode register 1 (CSIM1)

Fig. 4-60 shows a serial operating mode register 1 (CSIM1) format.

CSIM1 is an 8-bit register to specify the serial interface (channel 1) operating mode and serial clock.

It is manipulated by an 8-bit memory manipulation instruction. The higher 1 bit can be manipulated bit-wise. Use each bit name for bit manipulation.

RESET input clears all bits to 0.

Fig. 4-60 Serial Operating Mode Register 1 Format

Address	7	6	5	4	3	2	1	0	Symbol
FC8H	CSIE1	0	0	0	0	0	CSIM11	CSIM10	CSIM1

Serial Clock Select Bit (W)

CSIM11	CSIM10	Serial Clock 3-Wire Serial I/O Mode	SCK Pin Mode
0	0	External input clock to $\overline{\text{SCK1}}$ pin	Input
0	1	Setting disabled	—
1	0	$f_x/2^4$ (375 kHz, or 262 kHz) *	Output
1	1	$f_x/2^3$ (750 kHz, or 524 kHz) *	

\* Values in parentheses are when  $f_x = 6.0$  MHz or  $f_x = 4.19$  MHz.

Serial Interface Operation Enable/Disable Specify Bit (W)

		Shift Register 1 Operation	Serial Clock Counter	IRQCSI Flag	SO1 and SI1 Pins	
★	CSIE1	0	Shift operation disabled	Clear	Hold	Dedicated to port 8 functions
★		1	Shift operation enabled	Count operation	Settable	Functions in each mode and operations with port 8

Note Be sure to write "0" to bits 2 to 6 of the serial operating mode register.

**(b) Shift register 1 (SIO1)**

SIO1 is an 8-bit register which executes parallel to serial conversion and carries out serial transmission/reception (shift operation) in synchronization with a serial clock.

Serial transfer is started by writing data to SIO1.

In transmission, the data written to SIO1 is output to the serial output (SO1). In reception, data is read from the serial input (SI1) to SIO1.

This register can be read/written by an 8-bit manipulation instruction.

$\overline{\text{RESET}}$  input during operation makes the SIO1 value undefined.  $\overline{\text{RESET}}$  input in the standby mode holds the SIO1 value.

Shift operation stops after 8-bit transmission/reception.

SIO1 read and serial transfer start (write) are enabled at the following timings.

- Serial interface operation enable/disable bit (CSIE1) = 1 except when CSIE1 is set to "1" after data write to the shift register.
- When the serial clock is masked after 8-bit serial transfer.
- When  $\overline{\text{SCK1}}$  is at a high level.

(11) Serial interface (channel 1) operations

(a) Operation stop mode

The operation stop mode is used when serial transfer is not carried out. Power consumption is decreased in this mode.

In this mode, shift register 1 does not carry out shift operation and thus can be used as a normal 8-bit register.

RESET input sets the operation stop mode. The P82/SO1 pin and P83/SI1 pin are fixed to the input port. P81/SCK1 can be used as an input port by setting serial operating mode register 1.

(b) 3-wire serial I/O mode operations

The 3-wire serial I/O mode allows connection with the methods employed with another 75X series and 78K series, etc.

Communication is carried out using three lines of serial clock (SCK1), serial output (SO1) and serial input (SI1).

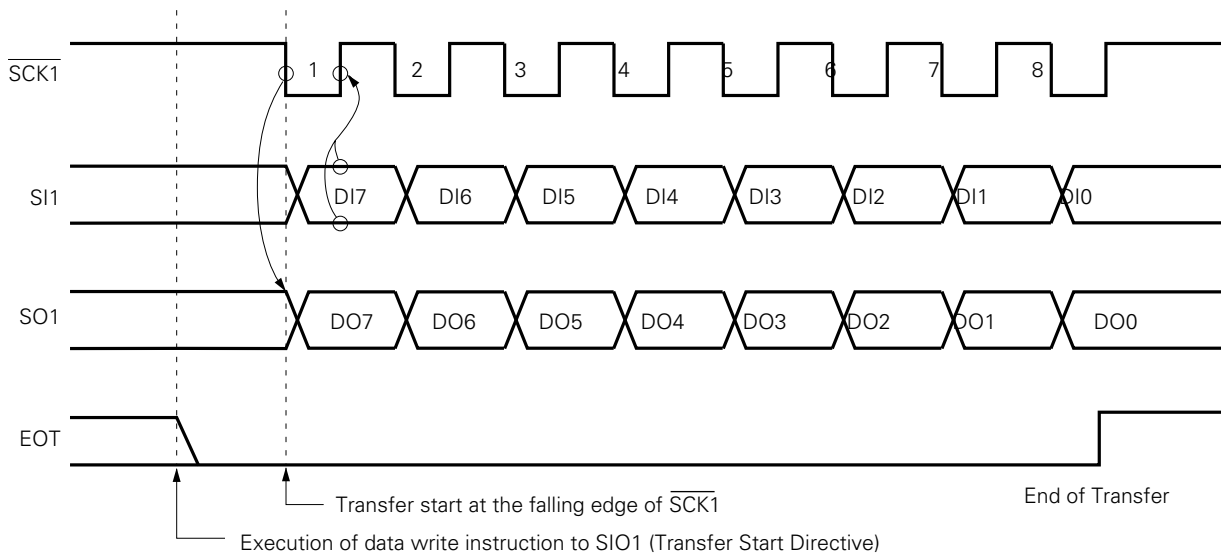
The 3-wire serial I/O mode is used for data transmission and reception in 8-bit units. Bit-wise data transmission/reception is carried out in synchronization with the serial clock.

Shift operation of shift register 1 is carried out at the falling edge of serial clock (SCK1). Transmit data is held at the SO1 latch and output from the SO1 pin.

Receive data input to the SI1 pin is latched to the shift register 1 at the rising edge of SCK1.

Shift register 1 operation automatically stops upon termination of 8-bit transfer and the serial transfer end flag (EOT) is set.

Fig. 4-61 3-Wire Serial I/O Mode Timing





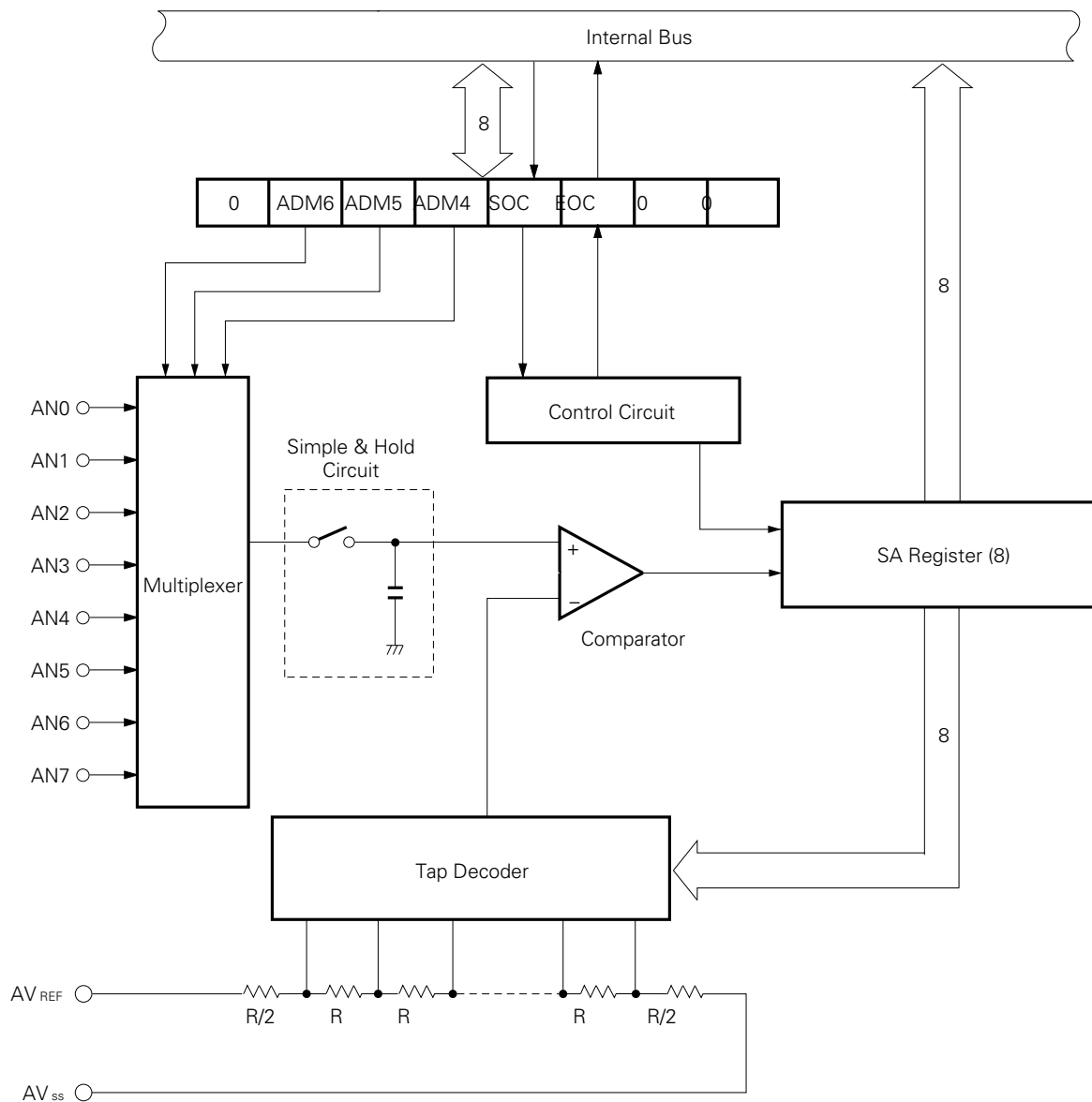
4.10 A/D CONVERTER

The μPD75237 incorporates an 8-bit resolution A/D converter with 8-channel analog inputs (AN0 to AN7). The A/D converter employs successive approximation.

(1) A/D converter configuration

Fig. 4-62 shows an A/D converter configuration.

Fig. 4-62 A/D Converter Block Diagram



**(2) A/D converter pin functions****(a) AN0 to AN7**

These are 8-channel analog signal input pins to the A/D converter. An analog signal to undergo A/D conversion is input to these pins.

The A/D converter incorporates a sample hold circuit. The analog input voltage is internally held during A/D conversion.

**(b) AV<sub>REF</sub> and AV<sub>SS</sub>**

The A/D converter reference voltage is input to these pins.

Signals input to AN0 to AN7 are converted to digital signals in accordance with the voltage applied between AV<sub>REF</sub> and AV<sub>SS</sub>.

★ AV<sub>SS</sub> should always be set to the same voltage as V<sub>SS</sub>.

**(c) AV<sub>DD</sub>**

AV<sub>DD</sub> is a power supply pin for the A/D converter. It should be set to the same voltage as V<sub>DD</sub>, even when the A/D converter is not used, or in standby mode.

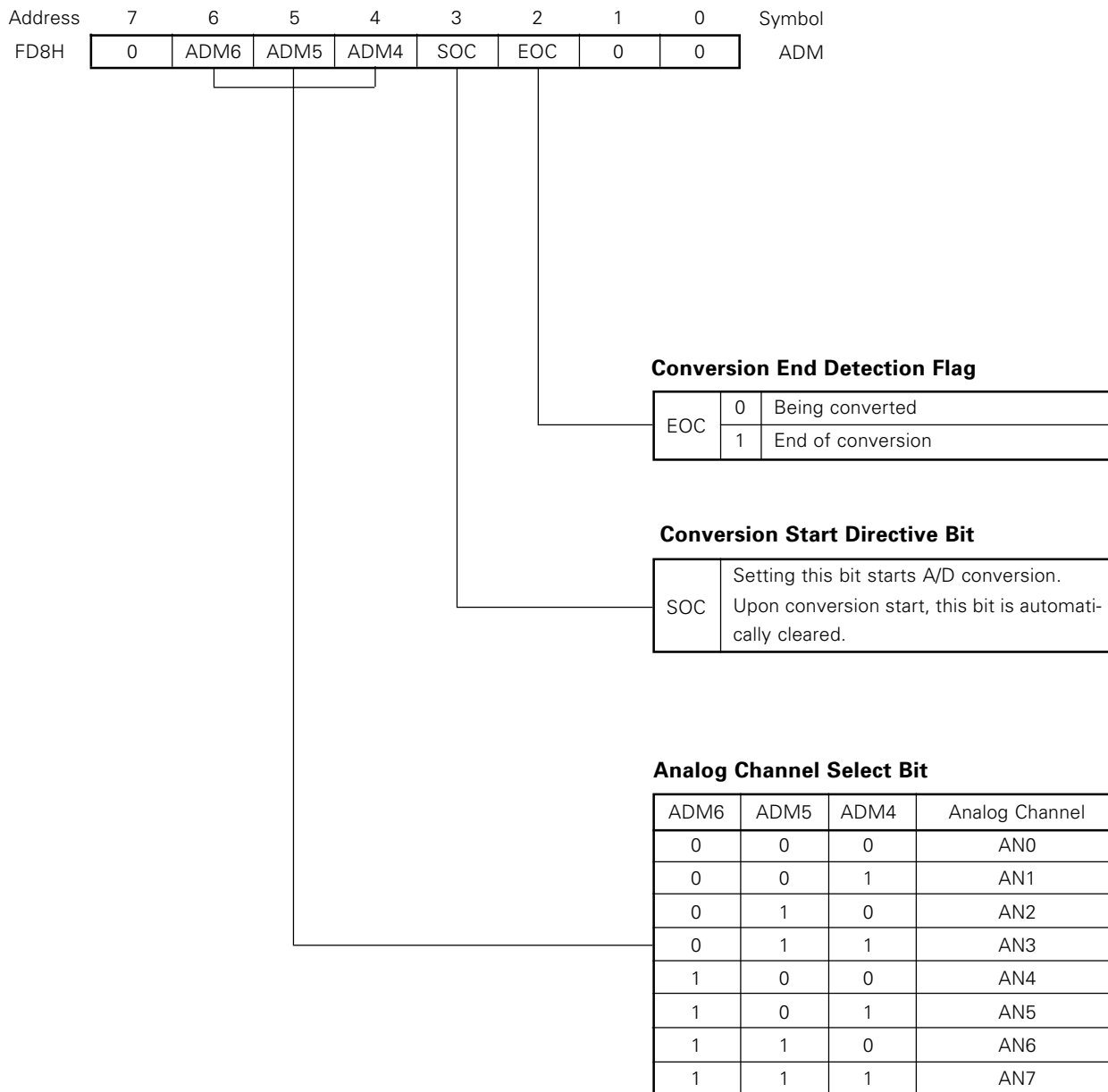
**(3) A/D conversion mode register**

The A/D conversion mode register (ADM) is an 8-bit register for analog input channel selection, conversion start command and conversion end detection (see **Fig. 4-63**).

The ADM is set by an 8-bit manipulation instruction. The bit 2 conversion end detection flag (EOC) and the bit 3 conversion start command bit (SOC) can be manipulated in bit units.

RESET input initializes the ADM to 04H (only EOC is set to "1" and all other bits are cleared to "0").

Fig. 4-63 A/D Conversion Mode Register Format



**Note** A/D conversion starts with a maximum delay of  $2^4/f_x$  sec ( $2.67 \mu\text{s}$ : at 6.0 MHz operation)\* after SOC setting (refer to 4.10 (5) A/D converter operations).

\*  $3.81 \mu\text{s}$  at 4.19 MHz operation

**(4) SA register (SA)**

The SA register (Successive Approximation Register) is an 8-bit register to store the result of A/D conversion by successive approximation.

The SA register is read by an 8-bit manipulation instruction. Data cannot be written to this register by software.

$\overline{\text{RESET}}$  input sets the SA register undefined.

**(5) A/D converter operations**

The analog input signal to undergo A/D conversion is specified by setting bits 6, 5 and 4 (ADM6, 5 and 4) of the A/D conversion mode register.

A/D conversion is started by setting (1) ADM bit 3 (SOC). SOC is automatically cleared (0) after the setting. A/D conversion is executed using successive approximation by hardware and the 8-bit conversion result data is stored into the SA register. Upon termination of conversion, bit 2 (EOC) of ADM is set (1).

Fig. 4-64 is an A/D conversion timing chart.

Use the A/D converter as follows.

- ① Select the analog input channel (ADM 6, 5 and 4 setting).
- ② Instruct A/D conversion start (SOC setting).
- ③ Wait for A/D conversion to terminate (wait for EOC to be set or wait with a software timer).
- ④ Read the A/D conversion result (SA register reading).

**Note** 1. ① and ② can be carried out simultaneously.  
 2. A maximum delay of  $2^4/f_x$  sec (2.67 μs: at 6.0 MHz operation)\* occurs from A/D conversion start to EOC clear after SOC setting. Thus, test EOC after the passage of time indicated in Table 4-7 after SOC setting. Table 4-7 shows A/D conversion times as well.

\* 3.81 μs at 4.19 MHz operation

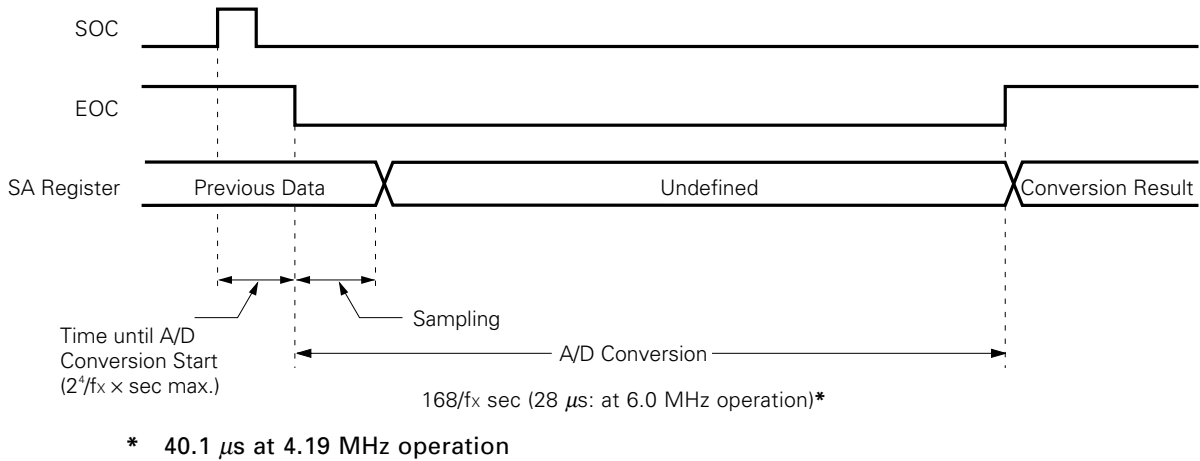
**Table 4-7 SCC and PCC Settings**

SCC and PCC Set Value				A/D Conversion Time	Wait time till EOC test after SOC setting	Wait time till the end of A/D conversion after SOC setting
SCC3	SCC0	PCC1	PCC0			
0	0	0	0	168/f <sub>x</sub> (28 μs : at 6.0 MHz operation)*	Wait not required	3 machine cycles
		0	1		1 machine cycle	11 machine cycles
		1	0		2 machine cycles	21 machine cycles
		1	1		4 machine cycles	42 machine cycles
0	1	×	×		Wait not required	Wait not required
1	×	×	×	Conversion operation stopped	—	—

\* 40.1 μs at 4.19 MHz operation

**Remarks** x : Don't care

Fig. 4-64 A/D Conversion Timing Chart



**(6) Standby mode precautions**

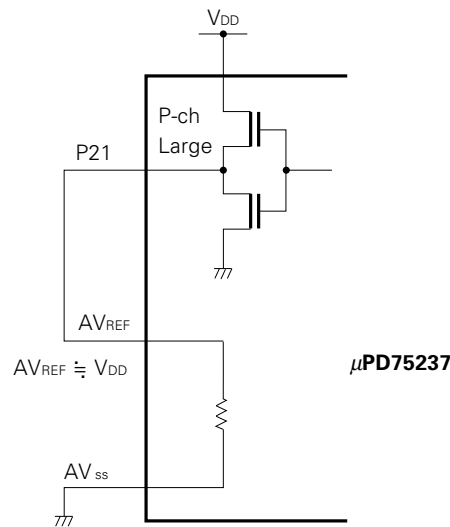
The A/D converter operates with the main system clock. Thus, the converter operation stops in the STOP mode or in the HALT mode with the subsystem clock. In this case also, current flows to the AV<sub>REF</sub> pin. Thus, it is necessary to cut the current to decrease the power consumption of the whole system. The P21 pin has a more improved driving capacity than any other port and so can directly supply a voltage to the AV<sub>REF</sub> pin.

However, in this case, the actual AV<sub>REF</sub> voltage have no accuracy. Thus, the conversion value itself has no accuracy and can only be used for relative comparison.

In the standby mode, power consumption can be decreased by generating a low level to P21.

The AV<sub>DD</sub> pin should be set to the same voltage as V<sub>DD</sub> in the standby mode.

Fig. 4-65 AV<sub>REF</sub> Pin Processing in Standby Mode



(7) Others and operating precautions

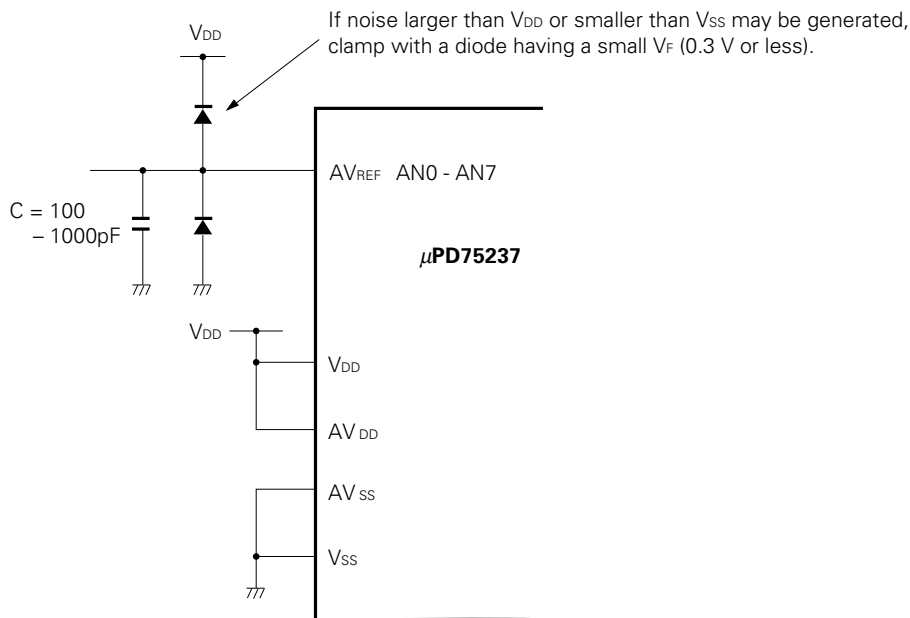
(a) AN0 to AN7 input range

Use AN0 to AN7 input voltages in the specified range. If a voltage larger than  $V_{DD}$  or smaller than  $V_{SS}$  is input (if in the absolute maximum range), the conversion value of the channel becomes undefined and may affect the conversion values of other channels.

★ (b) Countermeasures against noise

To maintain 8-bit accuracy, extra attention must be paid to noise in the  $AV_{REF}$  and AN0 to AN7 pins. The higher the analog input source output impedance becomes the more the noise effect becomes. To prevent that from occurring, mount C externally as shown in Fig. 4-66.

Fig. 4-66 Analog Input Pin Processing



(c) AN4/P90 to AN7/P93 pins

Analog inputs AN4 to AN7 also serve as the input port (PORT9) pin.

Do not execute a PORT9 input instruction during A/D conversion with any one of AN4 to AN7 selected.

The conversion accuracy may be deteriorated.

If a digital pulse is applied to a pin contiguous to the pin undergoing A/D conversion, the expected A/D conversion value may not be obtained because of coupling noise.

Thus, do not apply pulses to such pins.

★ (d)  $AV_{DD}$  pin

$AV_{DD}$  pin should be set to the same voltage even when A/D converter is not used, or in standby mode.

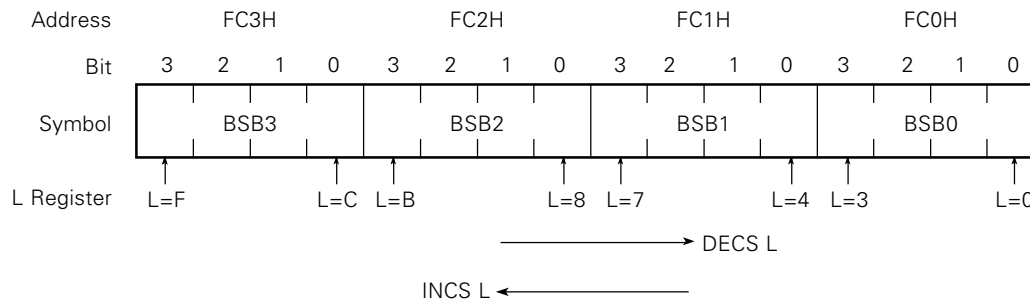
### 4.11 BIT SEQUENTIAL BUFFER : 16 BITS

The bit sequential buffer (BSB0 to BSB3) is a special data memory for bit manipulation.

Since this buffer can easily carry out bit manipulation by sequentially changing address and bit specification, it is useful to process data having long bit lengths in bit units.

This data memory consists of 16 bits and can execute the pmem.@L addressing of bit manipulation instructions. Thus, it can indirectly specify bits with the L register. In this case, processing can be carried out by sequentially shifting the specified bit by simply incrementing/decrementing the L register in the program loop.

**Fig. 4-67 Bit Sequential Buffer Format**



**Remarks** In pmem.@L addressing, the specified bit shifts in accordance with the L register. The bit sequential buffer can be operated irrespective of MBE or MBS specification.

Data manipulation is also possible by direct addressing. 1-, 4- and 8-bit direct addressing can be combined with pmem.@L addressing for applications to continuous 1-bit data input/output. In the case of 8-bit manipulation, the most and least significant 8 bits each are manipulated by specifying BSB0 and BSB2, respectively.

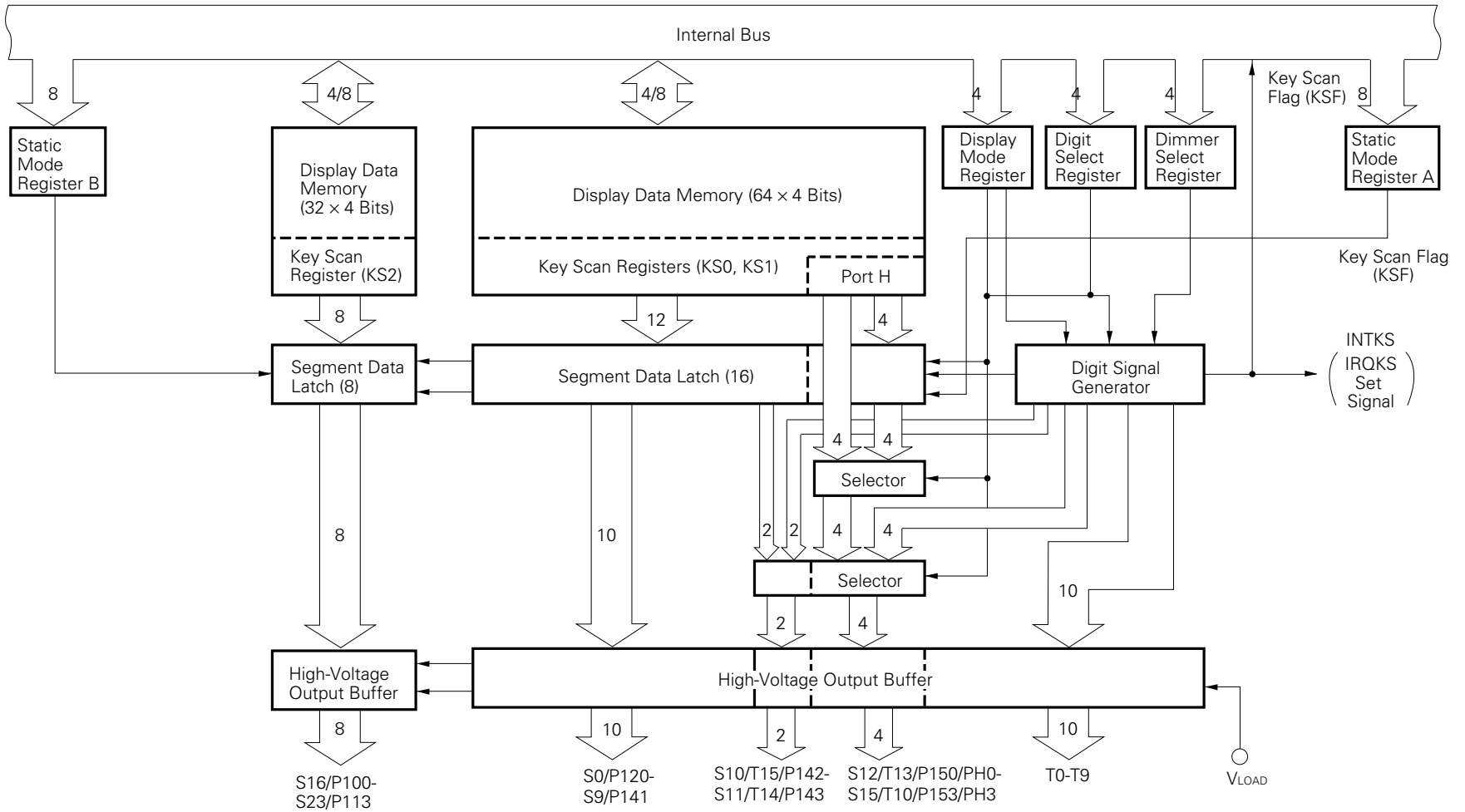
### 4.12 FIP CONTROLLER/DRIVER

#### (1) FIP controller/driver configuration

The μPD75237 incorporates a display controller which automatically generates the digit and segment signals by reading the display data memory contents by carrying out DMA operation and a high-voltage output buffer which can directly drive the fluorescent display tube (FIP). The FIP controller/driver configuration is shown in Fig. 4-68.

**Note** The FIP controller/driver can only operate at high and intermediate speeds (PCC = 0011B or 0010B) of the main system clock (SCC.0 = "0"). It may malfunction with any other clock or in the standby mode. Thus, be sure to stop FIP controller operation (DSPM.3 = "0") and then shift the unit to any other clock mode or the standby mode.

Fig. 4-68 FIP Controller/Driver Block Diagram





**(2) FIP controller/driver functions**

The FIP controller/driver built in the μPD75237 has the following functions:

- (a) Segment signal output (DMA operation) and automatic digit signal output are possible by automatic read of display data.
- (b) The FIP with 9 to 24 segments and 9 to 16 digits (up to a total of 34 display outputs) can be controlled using the display mode register (DSPM), digit select register (DIGS), static mode register A (STATA) and static mode register B (STATB).
- (c) Output not used for dynamic display can be used for static output or output port.
- (d) 8 brightness levels can be adjusted using the dimmer function.
- (e) Hardware is incorporated for key scan application.
  - Key scan interrupt (IRQS) generation (key scan timing detection)
  - Key scan data output from segment output is possible with the key scan buffers (KS0, KS1 and KS2).
- (f) High-voltage output pin (40 V) capable of directly driving FIP.
  - Segment output pins (S0 to S9, S16 to S23) :  $V_{OD} = 40\text{ V}$ ,  $I_{OD} = 3\text{ mA}$
  - Digit output pins (T0 to T15) :  $V_{OD} = 40\text{ V}$ ,  $I_{OD} = 15\text{ mA}$
- (g) Display output pin mask option
  - T0 to T9 and S0 to S15 can incorporate a pull-down resistor in bit units to  $V_{LOAD}$ .
  - S16 to S23 can incorporate a pull-down resistor in bit units to  $V_{LOAD}$  or  $V_{SS}$ . Determine in 8-bit units whether a pull-down resistor should be incorporated to  $V_{LOAD}$  or  $V_{SS}$ .

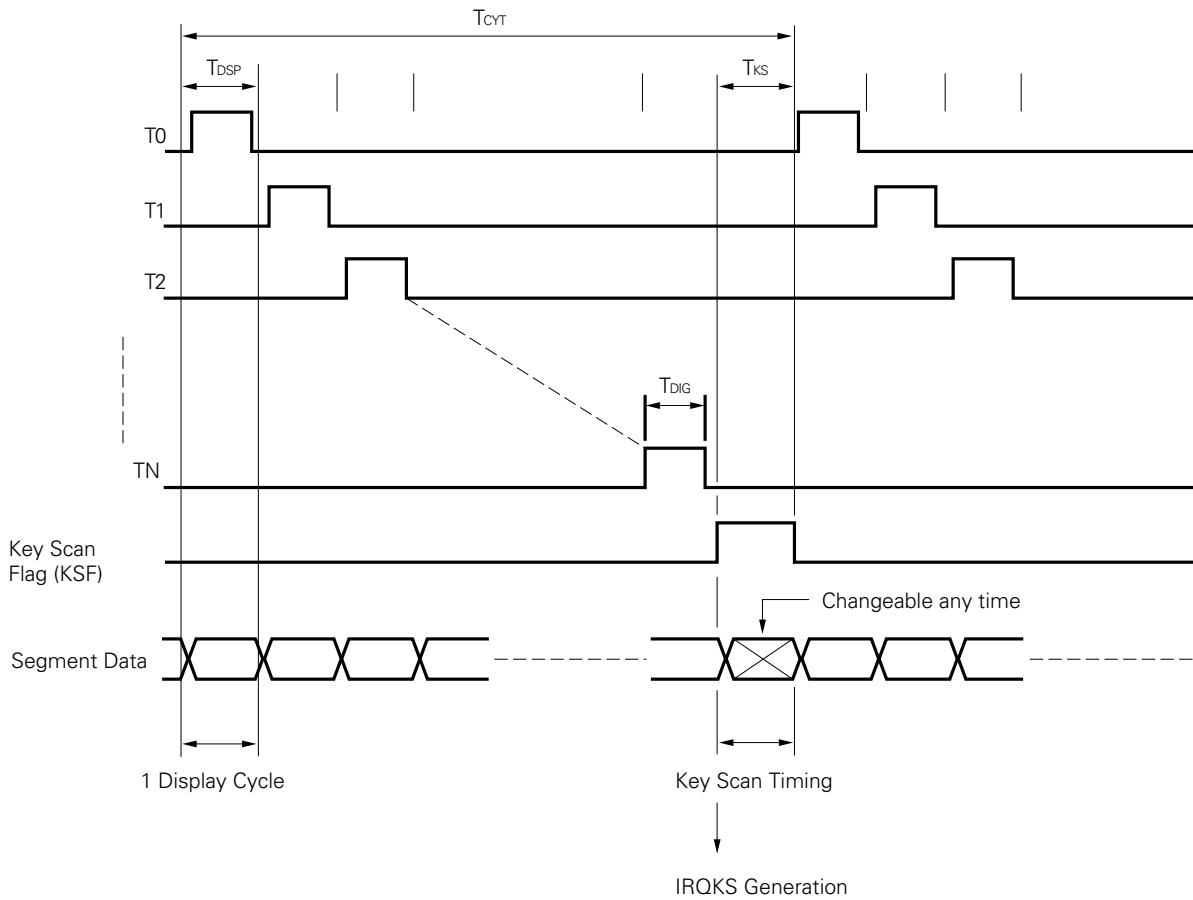
**(3) Display output function differences between μPD75237 and μPD75216A/μPD75217**

Table 4-8 shows display output function differences between μPD75237 and μPD75216A/μPD75217.

**Table 4-8 Display Output Function Differences between μPD75237 and μPD75216A/μPD75217**

	μPD75237	μPD75216A, 75217
High-voltage output display	FIP output total : 34 outputs Segment output : 9 to 24 outputs Digit output : 9 to 16 outputs	FIP output total : 26 outputs Segment output : 9 to 16 outputs Digit output : 9 to 16 outputs
Display data area	1A0H to 1FFH	1C0H to 1FFH
Output dual-function pin	S0 to S23 (PORT10 to PORT15)	S12 to S15 (PORTH)
Key scan register	KS0 to KS2	KS0, KS1

Fig. 4-69 FIP Controller Operation Timing



N : Digit select register set value

$T_{DSP}$  : 1 display cycle

$$\left( \frac{1024}{f_x} = 171 \mu s: \text{at } 6.0 \text{ MHz operation*1} \text{ or } \frac{2048}{f_x} = 341 \mu s: \text{at } 6.0 \text{ MHz operation*2} \right)$$

$T_{CYT}$  : Display period ( $T_{CYT} = T_{DSP} \times (N + 2)$ )

$T_{DIG}$  : Digit signal pulse width variable at 8 levels using a dimmer select register

- \* 1. 244 μs at 4.19 MHz operation
- 2. 489 μs at 4.19 MHz operation

**(4) Display mode register (DSPM)**

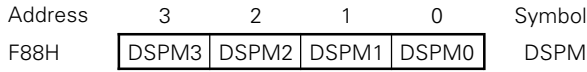
The display mode register (DSPM) is a 4-bit register to enable/disable display operation and to specify the number of display segments. Its format is shown in Fig. 4-70.

The display mode register is set by the 4-bit memory manipulation instruction.

When setting the standby mode (STOP mode, HALT mode) or operating the DSPM with the subsystem clock (fxT), stop the display operation by presetting DSPM.3 to "0".

RESET input clears all bits to "0".

**Fig. 4-70 Display Mode Register Format**



**Display Segment Number Specify Bit**

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)

**Remarks** Values when S16 to S23 are set to the dynamic mode by STATB are in parentheses.

**Display Operation Enable/Disable Bit**

DSPM3	0	Display stopped
	1	Display enabled

**(5) Digit select register (DIGS)**

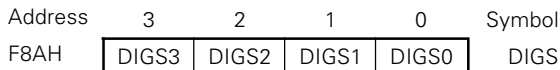
The digit select register (DIGS) is a 4-bit register to specify the number of digits to be displayed. Its format is shown in Fig. 4-71.

DIGS is set by the 4-bit memory manipulation instruction. The number of digits to be displayed can be set in the range from 9 to 16 by DIGS setting.

The value of 8-digit or less cannot be selected.

RESET input initializes DIGS to "1000B" and selects 9-digit display.

**Fig. 4-71 Digit Select Register Format**



**Note** 0 to 7 cannot be set in N.

DIGS0 to 3 Set Value	No. of Digits to be Displayed
N (= 8 to 15)	N + 1

**(6) Dimmer select register (DIMS)**

The dimmer select register (DIMS) is a 4-bit register to specify the digit signal cut width to prevent display light emission from leaking and to maintain the dimmer (brightness adjustment) function. It is also used to select the display cycle (T<sub>DSP</sub>).

The DIMS format is shown in Fig. 4-72.

The DIMS is set by the 4-bit memory manipulation instruction.

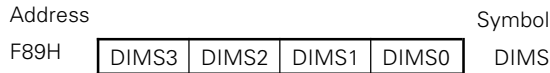
The display cycle of 341 μs: at 6.0 MHz operation\*1 is normally selected with DIMS.0 set to "1" to minimize light emission leakage. Because if the number of digits to be displayed increases, the display period becomes equivalent to the commercial power supply frequency and display flickers, select 171 μs: at 6.0 MHz operation\*2.

If any light emission leakage occurs, adjust the digit signal cut width with DIMS.1 to DIMS.3.

RESET input clears all bits to "0".

- \* 1. 489 μs at 4.19 MHz operation
- 2. 244 μs at 4.19 MHz operation

**Fig. 4-72 Dimmer Select Register Format**



**Display Cycle Specify Bit**

DIMS0	0	Sets $\frac{1024}{f_x}$ as one display cycle (1 cycle = 171 μs:6.0 MHz)*1
	1	Sets $\frac{2048}{f_x}$ as one display cycle (1 cycle = 341 μs:6.0 MHz)*2

**Digit Signal Cut Width Specify 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 μs at 4.19 MHz operation
- 2. 489 μs at 4.19 MHz operation

**(7) Static mode register**

The static mode register is intended to specify the static output/dynamic output of the segment output pin.

There are two types of static mode registers: static mode register A, static mode register B. Figs. 4-73 and 4-74 show their formats, respectively.

These two types of static mode registers are set by an 8-bit manipulation instruction.  $\overline{\text{RESET}}$  input clears all bits to "0".

**(a) Static mode register A (STATA)**

Static mode register A (STATA) is intended to specify the static output/dynamic output of the S0/P120 to S15/P153/T10/PH3 pins.

**Fig. 4-73 Static Mode Register A (STATA)**

Address	7	6	5	4	3	2	1	0	Symbol
FD6H	0	0	0	0	STATA3	STATA2	STATA1	STATA0	STATA

**S0 to S15 Pin Static Output/Dynamic Output Select Bit**

STATA3	STATA2	STATA1	STATA0	S0 to S15 Pins Output Status
0	0	0	0	S0 to S15 become dynamic output. The numbers of segments and digits are set by DSPM and DIGS.
1	1	1	1	S0 to S15 become static output. Perform static data output using an output instruction for ports 12 to 15. These pins are not affected by the DSPM.3 value.

**Note** It is not possible to set some of the S0 to S15 pins to dynamic output and the remaining pins to static output.

**(b) Static mode register B (STATB)**

Static mode register B (STATB) is intended to specify the static output/dynamic output of the S16/P100 to S23/P113 pins.

**Fig. 4-74 Static Mode Register B (STATB)**

Address	7	6	5	4	3	2	1	0	Symbol
FD4H	0	0	STATB5	STATB4	0	0	0	0	STATB

**S16 to S23 Pin Static Output/Dynamic Output Select Bit**

STATB5	STATB4	S16 to S23 Pins Output Status
0	0	Dynamic output. Dynamic output is generated in accordance with 1A0H to 1BDH contents.
1	1	Static output. Perform static data output using an output instruction for ports 10 and 11. These pins are not affected by the DSPM.3 value.

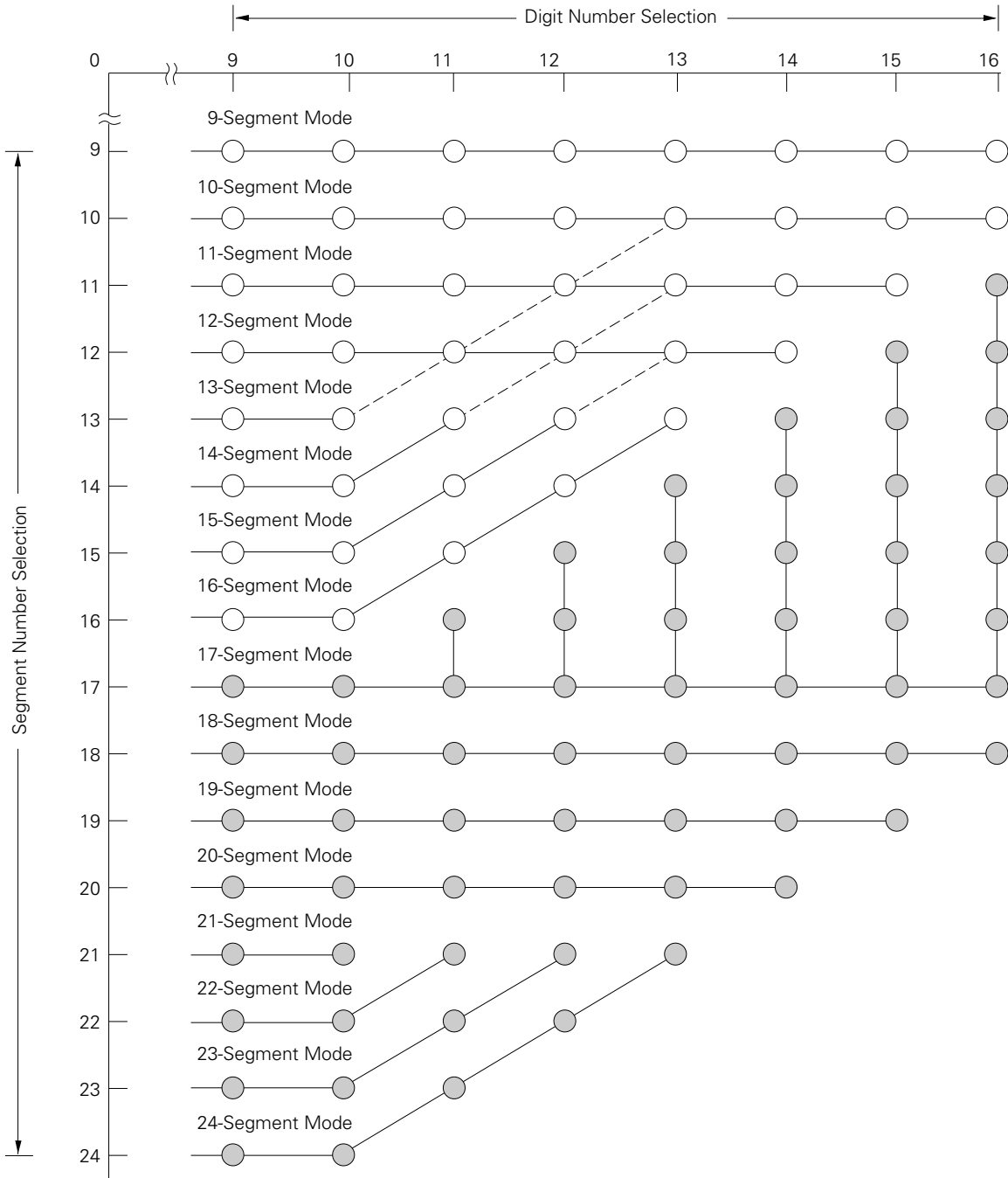
**Note** It is not possible to set some of the S16 to S23 pins to dynamic output and the remaining pins to static output.

**(8) Display mode selection**

The numbers of segments and digits which can be displayed using the built-in FIP controller/driver depend on the display mode.

Fig. 4-75 shows a display mode selection diagram.

**Fig. 4-75 Display Mode Selection Diagram**



**Remarks** The circled modes with shading are those expanded from the μPD75216A and μPD75217.

**(9) Display data memory**

The display data memory is an area storing the displayed segment data and is mapped at addresses 1A0H to 1FFH of the data memory. Display data is automatically read by a display controller (DMA operation). The areas not used for display can be used as normal data memory.

Display data operation is carried out by a data memory manipulation instruction. Data manipulation is possible in 1-, 4- and 8-bit units. Only even addressed can be specified for 8-bit manipulation instruction execution.

Addresses 1FCH to 1FFH, 1BEH and 1BFH of the display data memory also serve as key scan registers (KS0, KS1 and KS2).

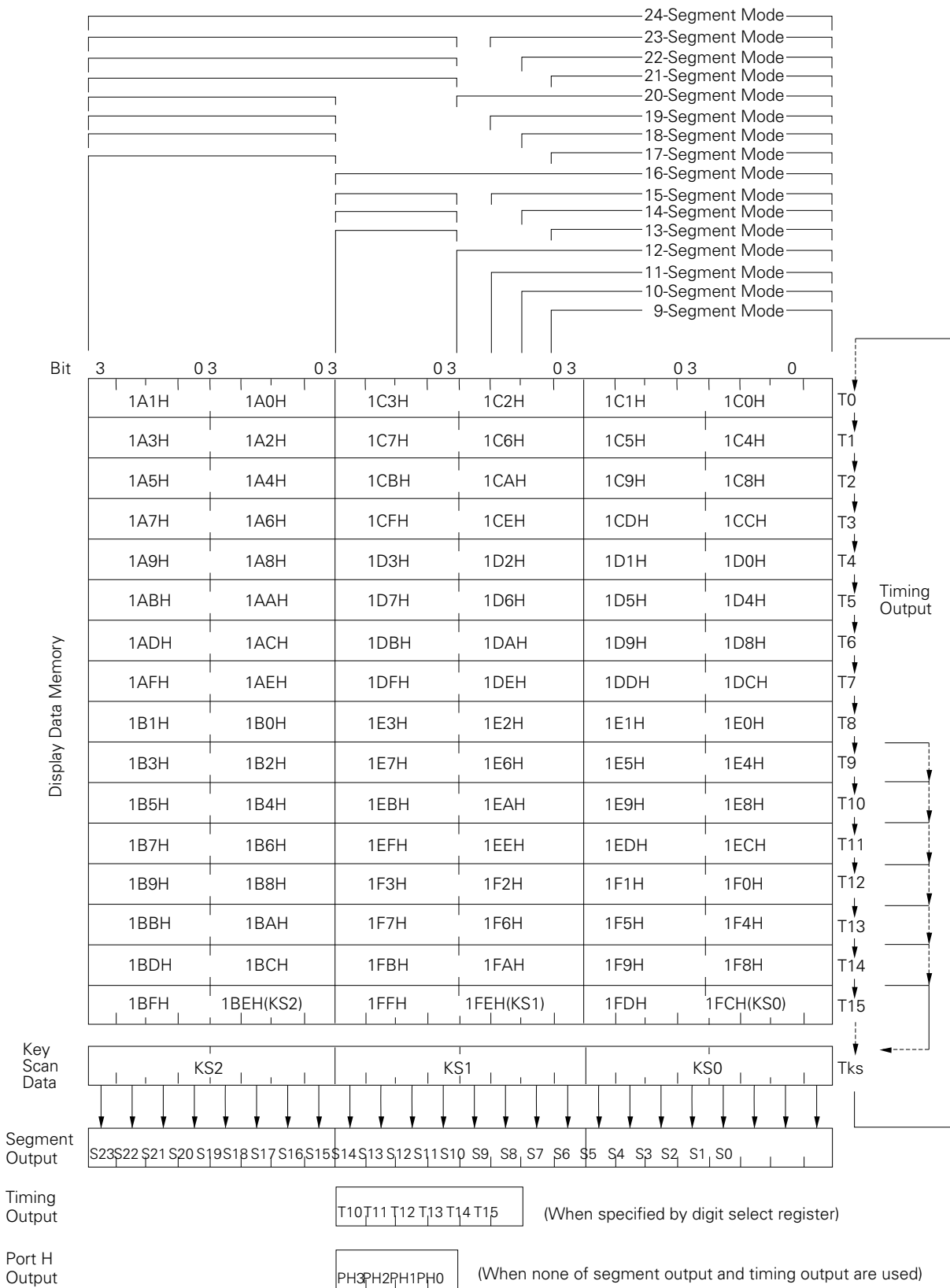
**Table 4-9 Data Memories which also Serve as Key Scan Registers**

Key Scan Register	Data Memory which also Serves as Key Scan Register
KS0	1FCH, 1FDH
KS1	1FEH, 1FFH
KS2	1BEH, 1BFH

**Note** Extra caution is necessary when transferring a program developed for the μPD75237 to one for the μPD75216A and μPD75217 because a maximum of 16 segments are displayed and no data memory is incorporated at addresses (1A0H + 4n and 1A1H + 4n) in the case of the μPD75216A and μPD75217.



Fig. 4-76 Display Data Memory Contents and Segment Outputs



**(10) Key scan registers (KS0, KS1 and KS2)**

The key scan registers (KS0, KS1 and KS2) are used to set the segment output data in the key scan timing mapped in the part of the display data memory (addresses 1FCH, 1FDH, 1FEH, 1FFH, 1BEH and 1BFH).

KS0, KS1 and KS2 are 8-bit registers and are normally manipulated by an 8-bit manipulation instruction (the lower 4 bits can be manipulated bit-wise or in 4-bit units).

Data set to KS0, KS1 and KS2 is output from the segment output pin at the key scan timing. During the key scan timing the segment output data can be immediately changed by rewriting KS0, KS1 and KS2. Key scan can be performed using the segment output.

**(11) Key scan flag (KSF)**

The key scan flag is set ("1") during the key scan timing and is automatically reset ("0") in all other timings. The KSF is mapped at bit 3 of address F8AH and is bit-wise testable. No write is possible. Whether the KSF is at the key scan timing can be checked by testing it. Thus, it is possible to check whether key input data is correct or not.

## 5. INTERRUPT FUNCTIONS

The μPD75237 has eight types of interrupt sources and can generate multiple interrupts with priority order. It is also equipped with two types of test sources. INT2 is an edge detected testable input.

**Table 5-1 Interrupt Source Types**

Interrupt Source		Internal/ External	Interrupt Order *1	Vectored Interrupt Request Signal (Vector Table Address)
INTBT (Reference timer interval signal from the basic interval timer)		Internal	1	VRQ1 (0002H)
INT4 (Rising or falling edge detection)		External		
INT0	(Rising and falling detected edge selection)	External	2	VRQ2 (0004H)
INT1		External	3	VRQ3 (0006H)
INTCSI0 (Serial data transfer end signal)		Internal	4	VRQ4 (0008H)
INTT0 (Match signal from timer event/counter 0)		Internal	5	VRQ5 (000AH)
INTTPG (Match signal from timer/pulse generator)		Internal	6	VRQ6 (000CH)
INTKS (Key scan timing signal from display controller)		Internal	7	VRQ7 (000EH)
INT2 *2 (Rising edge detection)		External	Testable input signal (IRQ2 and IRQW set)	
INTW *2 (Signal from watch timer)		Internal		

- \* 1. Interrupt order is priority order to be applied when two or more interrupt requests are generated simultaneously.
- 2. These are test sources. They are affected by interrupt enable flags as in the case of interrupt sources, but no vectored interrupt is generated.

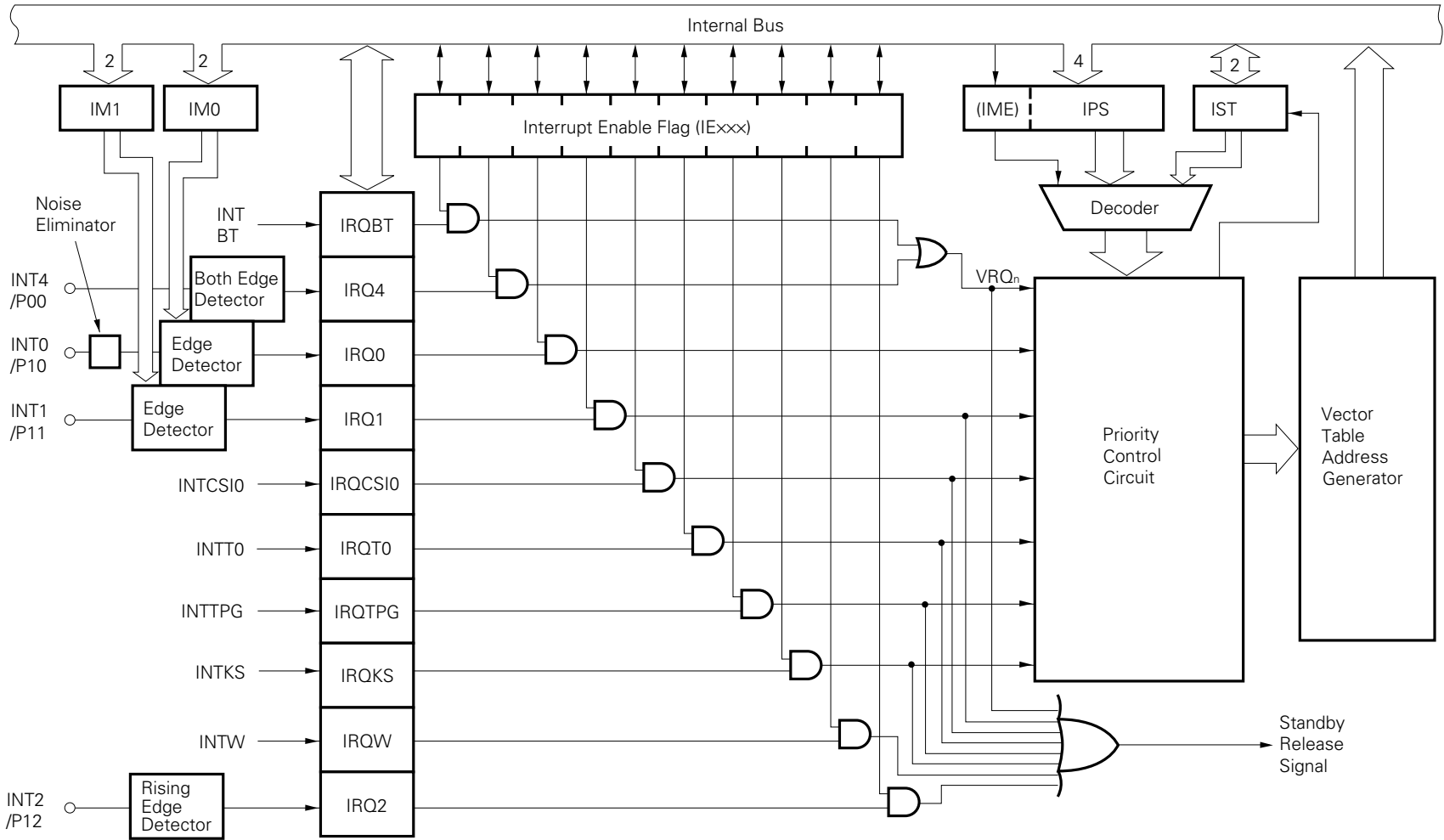
The μPD75237 interrupt control circuit has the following functions:

- (a) Hardware-controller vectored interrupt function which can control interrupt acknowledge with the interrupt enable flag (IE<sub>xxx</sub>) and the interrupt master enable flag (IME).
- (b) Function of setting any interrupt start address.
- (c) Multiple interrupt function which can specify priority order with the interrupt priority select register (IPS).
- (d) Interrupt request flag (IRQ<sub>xxx</sub>) test function. (Interrupt generation can be checked by software.)
- (e) Standby mode release function. (Interrupt to be released by interrupt enable flag can be selected.)

### 5.1 INTERRUPT CONTROL CIRCUIT CONFIGURATION

The interrupt control circuit has a configuration shown in Fig. 5-1 and each hardware is mapped in the data memory space.

Fig. 5-1 Interrupt Control Circuit Block Diagram



## 5.2 INTERRUPT CONTROL CIRCUIT HARDWARE DEVICES

### (1) Interrupt request flag, interrupt enable flag

There are ten interrupt request flags (IRQ<sub>xxx</sub>) corresponding to interrupt sources (interrupt :8, test :2) as shown below. ★

INT0 interrupt request flag (IRQ0)	Serial interface interrupt request flag (IRQCSIO)
INT1 interrupt request flag (IRQ1)	Timer/event counter interrupt request flag (IRQTO)
INT2 interrupt request flag (IRQ2)	Timer/pulse generator interrupt request flag (IRQTPG)
INT4 interrupt request flag (IRQ4)	Key scan interrupt request flag (IRQKS)
BT interrupt request flag (IRQBT)	Watch timer interrupt request flag (IRQW)

Interrupt request flag is set to "1" at generation of an interrupt request and is automatically cleared ("0") upon execution of interrupt service. IRQBT and IRQ4 carry out clear operation differently because they share the vector address. (See **5.5 VECTOR ADDRESS SHARING INTERRUPT SERVICING.**)

There are ten interrupt enable flags (IE<sub>xxx</sub>) corresponding to interrupt request flags as shown below. ★

INT0 interrupt enable flag (IE0)	Serial interface interrupt enable flag (IECSIO)
INT1 interrupt enable flag (IE1)	Timer/event counter interrupt enable flag (IETO)
INT2 interrupt enable flag (IE2)	Timer/pulse generator interrupt enable flag (IETPG)
INT4 interrupt enable flag (IE4)	Key scan interrupt enable flag (IEKS)
BT interrupt enable flag (IEBT)	Watch timer interrupt enable flag (IEW)

When the contents of interrupt enable flag is "1", interrupt is enabled and when it is "0", interrupt is disabled. When the interrupt request flag is set and the interrupt enable flag has enabled interrupt, the vectored interrupt request (VRQ<sub>n</sub>) is generated.

This signal is also used to release the standby mode.

Both the interrupt request flag and interrupt enable flag are operated by the bit manipulation instruction and 4-bit memory manipulation instruction. They can be operated directly by the bit manipulation instruction irrespective of MBE setting. The interrupt enable flag is operated by the EI IE<sub>xxx</sub> and DI IE<sub>xxx</sub> instruction. The SKTCLR instruction is normally used to test the interrupt request flag.

When the interrupt request flag is set by an instruction even if an interrupt has not been generated, the vectored interrupt is executed in the same way as when an interrupt had been generated.

$\overline{\text{RESET}}$  input clears the interrupt request flag and the interrupt enable flag ("0") and disables all interrupts.

Table 5-2 Interrupt Request Flag Set Signals

Interrupt Request Flag	Interrupt Request Flag Set Signal	Interrupt Enable Flag
IRQBT	Set by the reference time interval signal generated by the basic interval timer.	IEBT
IRQ4	Set upon detection of the rising or falling edge of the INT4/P00 input signal.	IE4
IRQ0	Set upon detection of the INT0/P10 pin input signal edge. The detected edge is selected using the INT0 mode register (IM0).	IE0
IRQ1	Set upon detection of the INT1/P11 pin input signal edge. The detected edge is selected using the INT1 mode register (IM1).	IE1
IRQCSI0	Set by the serial data transfer operation end signal of the serial interface.	IECSI0
IRQT0	Set by the match signal from the timer/event counter #0.	IET0
IRQTPG	Set by the match signal from the timer/pulse generator.	IETPG
IRQKS	Set by the key scan timing signal from the display controller.	IEKS
IRQW	Set by a signal from the watch timer.	IEW
IRQ2	Set upon detection of the rising edge of the INT2/P12 pin input signal.	IE2

★ (2) **Noise eliminator and edge detection mode register**

INT0, INT1 and INT2 each have the configuration shown in Figs. 5-2 and 5-3 and serve as the external interrupt input capable of selecting detected edges.

INT0 has a function of eliminating noise with sampling clock. Pulses having a shorter width than 2 sampling clock cycles\* are eliminated as noise by noise eliminator.

However, pulses having a larger width than 1 sampling clock cycle may be acknowledged as an interrupt signal depending on the sampling timing. Pulses having a larger width than 2 sampling clock cycles are securely acknowledged as an interrupt signal.

INT0 has two sampling clocks,  $\Phi$  and  $f_x/64$  and can select and use either clock. Selection is made by bit 3 (IM03) of the edge detection mode register (refer to Fig. 5-4).

IRQ2 is set by detecting the rising edge of INT2 pin input.

Edge detection mode registers (IM0 and IM1) to select detection edge have the format shown in Fig. 5-4.

IM0 and IM1 each are set by a 4-bit memory manipulation instruction.  $\overline{\text{RESET}}$  input clears all bits to 0 and specifies INT0, INT1 and INT2 for the rising edge.

- \* When sampling clock is  $\Phi$  :  $2t_{cy}$   
 When sampling clock is  $f_x/64$  :  $128/f_x$

- Note**
1. Since INT0 samples by clock, it is not operated in the standby mode.
  2. Pulses are input to the INT0/P10 pin serving as a port via the noise eliminator. Thus, input pulses having two sampling clock cycles or larger.

Fig. 5-2 INT0 and INT1 Configuration

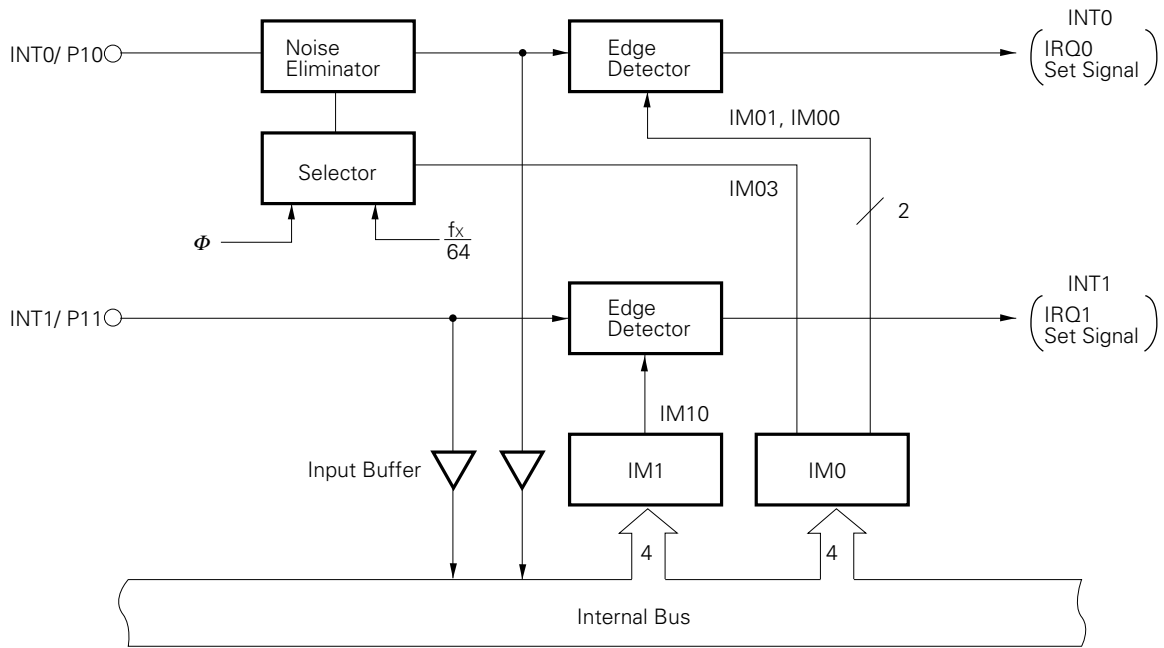


Fig. 5-3 INT2 Configuration

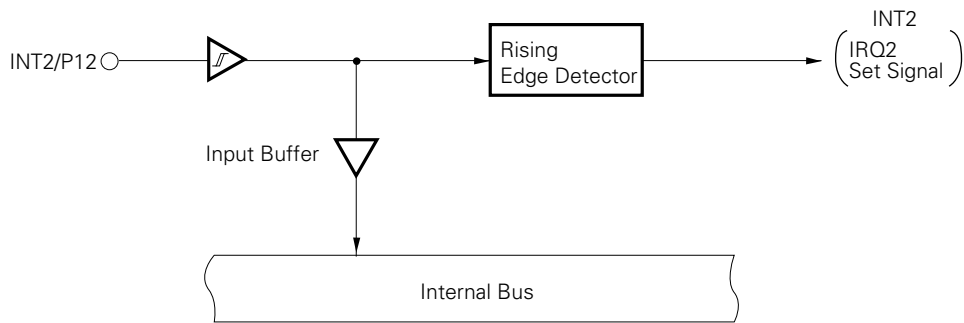
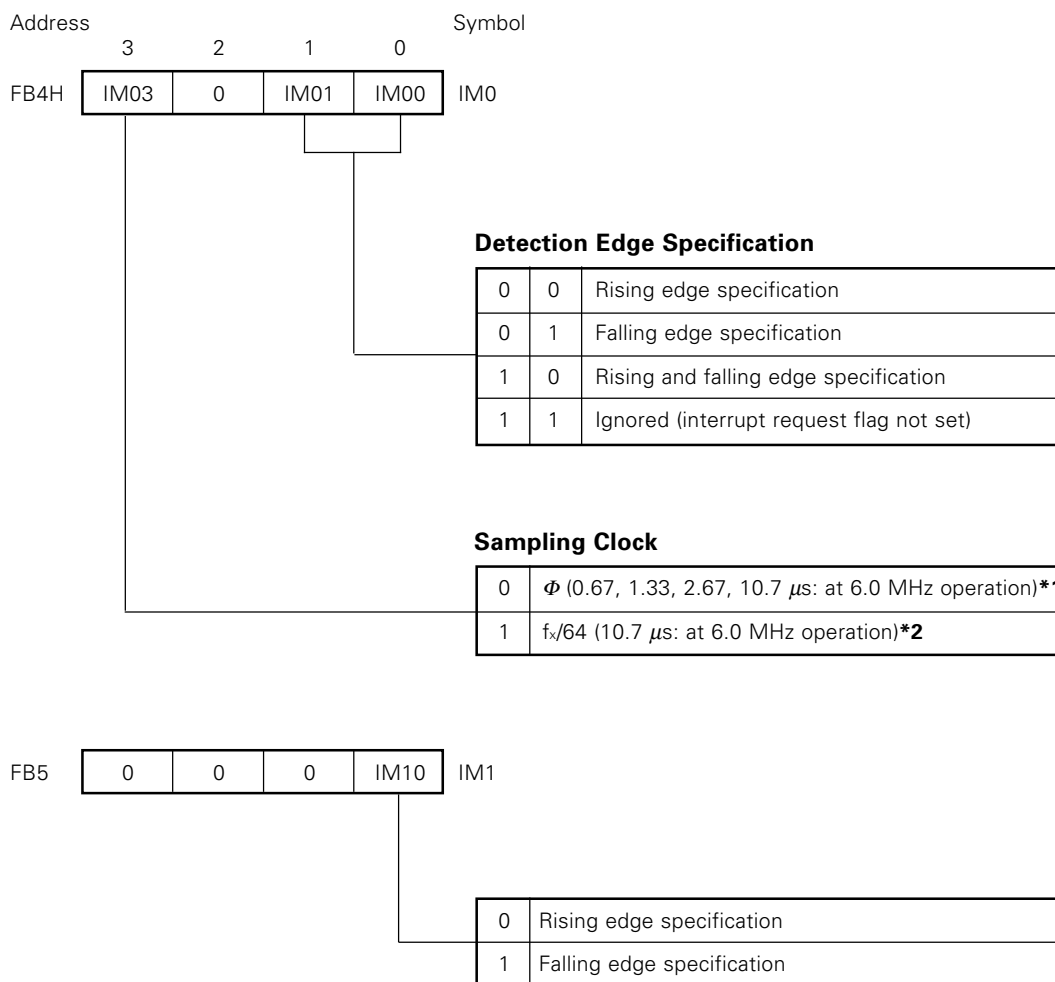


Fig. 5-4 Edge Detection Mode Register Format



- \* 1. 0.95, 1.91, 3.82, 15.3  $\mu$ s at 4.19 MHz operation
- 2. 15.3  $\mu$ s at 4.19 MHz operation

**Note** If the edge detection mode register is changed, the interrupt request flag may be set. To prevent that from occurring, disable interrupt and change edge detection mode register first, then enable interruption after clearing the interrupt request flag by the CLR1 instruction. If  $f_x/64$  has been selected as sampling clock by changing IM0, it is necessary to clear the interrupt request flag 16 machine cycles after the mode register has been changed.



**(3) Interrupt priority select register (IPS)**

The interrupt priority select register is used to select high interrupt enabled for multiple interrupt and is specified by the least significant 3 bits.

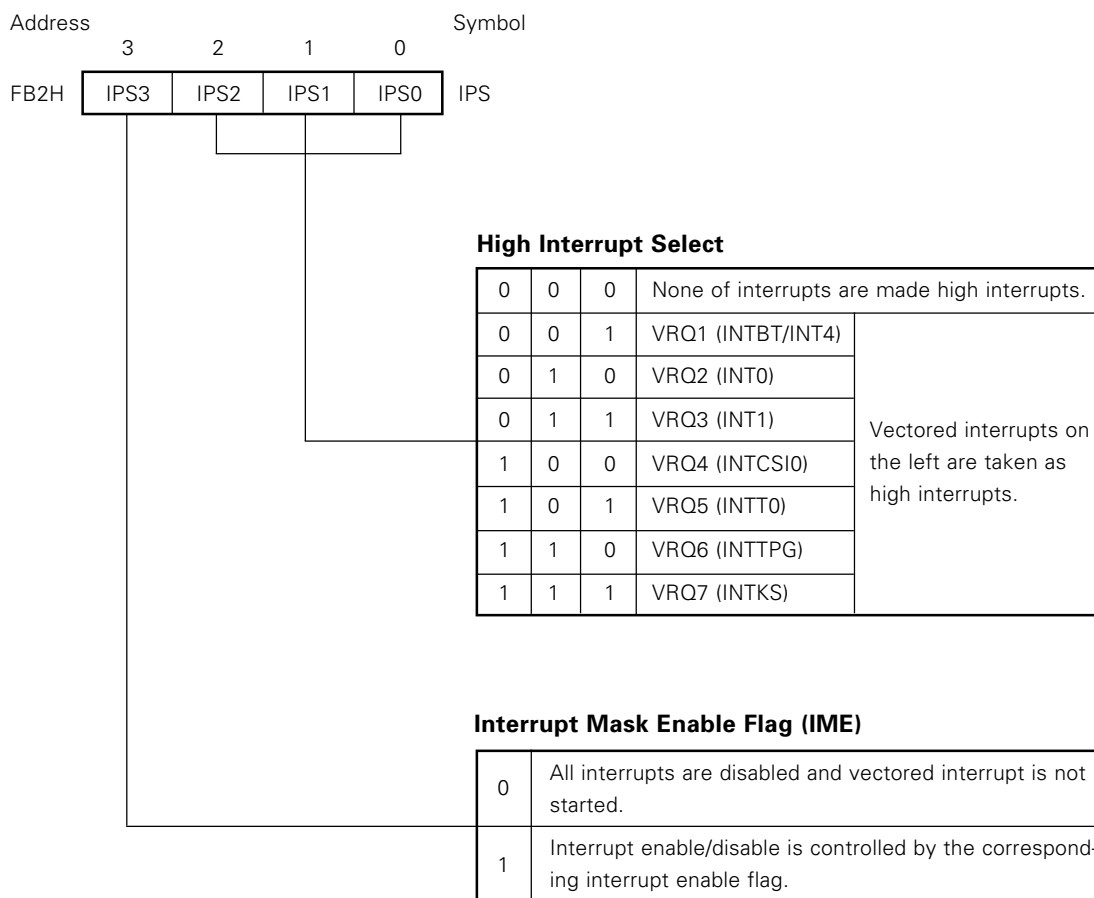
Bit 3 is an interrupt master enable flag (IME) to specify whether all interrupts should be disabled or not.

The IPS is set by the 4-bit memory manipulation instruction and bit 3 is set/reset by the EI/DI instruction.

When changing the low-order 3-bit contents of IPS, it is necessary to do so with interrupt disabled (IME = 0).

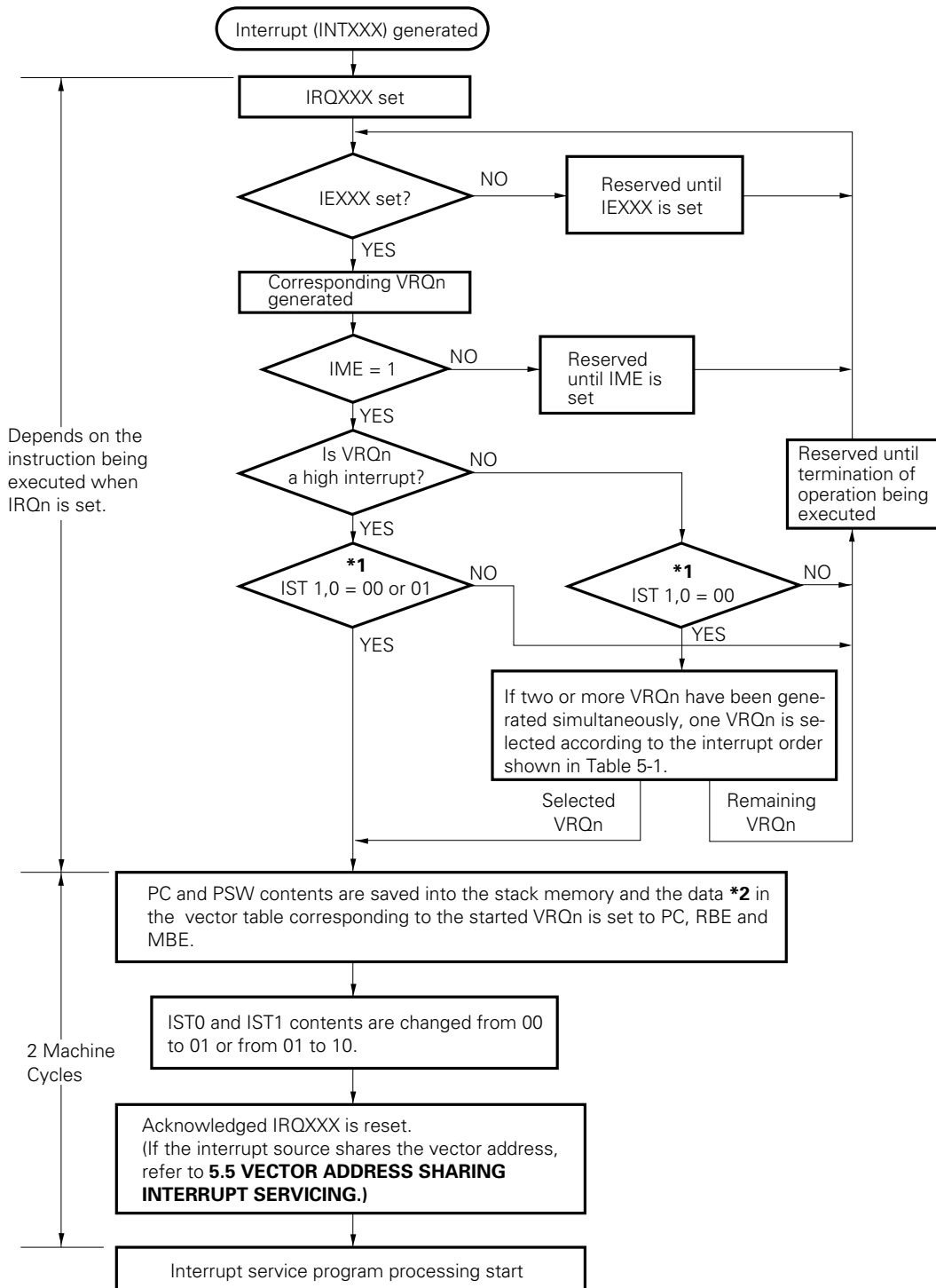
RESET input clears all bits to "0".

**Fig. 5-5 Interrupt Priority Select Register**



5.3 INTERRUPT SEQUENCE

If interrupt is generated, it is processed using the following procedure:



- \* 1. IST1 and IST0 : Interrupt status flags (PSW bits 3, 2: Refer to Table 5-3 IST1 and IST0 Interrupt Servicing Statuses).
- 2. The start address of the interrupt service program and the MBE and RBE set values at the start of interrupt are stored in each vector table.

5.4 MULTI-INTERRUPT SERVICE CONTROL

The following two methods are available for the μPD75237 to generate multi-interrupts.

(1) Multi-interruption specifying high interrupt

This is a standard multi-interrupt method of the μPD75237 in which one interrupt source is selected and multi-interruption (dual interrupt) is enabled.

In other words, the high interrupt specified using the interrupt priority select register (IPS) is enabled when the status of the operation being executed is 0 or 1. All other interrupts (low interrupts) are only enabled when the status is 0. (Refer to Fig. 5-6 and Table 5-3.)

Fig. 5-6 Multi-Interruption by High Interrupt

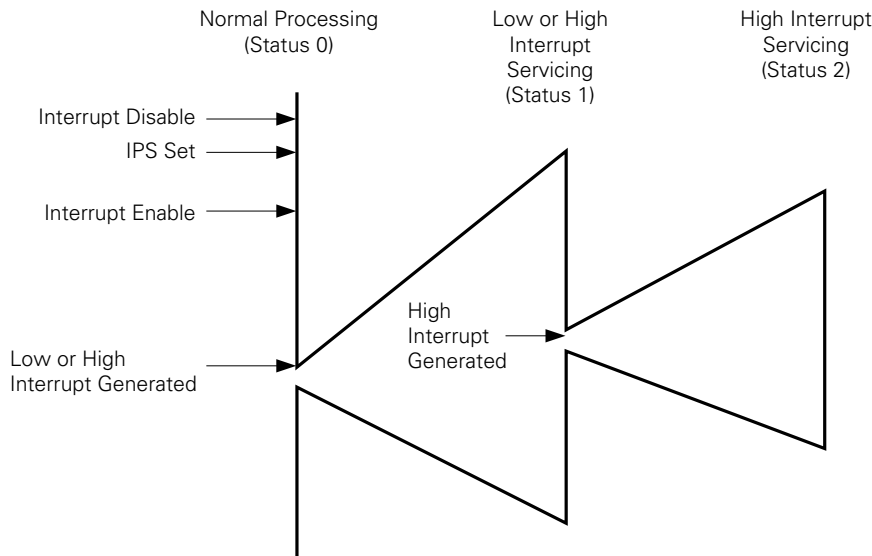


Table 5-3 IST1 and IST0 Interrupt Servicing Statuses

IST1	IST0	Status of Servicing being Executed	CPU Processing Contents	Interrupt Acknowledgeable Interrupt Request	After Interrupt Acknowledgement		
					IST1	IST0	
0	0	Status 0	Normal program being processed	All interrupts acknowledgeable	0	1	
0	1	Status 1	Low or high interrupt being servicing	Only high interrupt acknowledgeable	1	0	
1	0	Status 2	High interrupt being servicing	All interrupts not acknowledgeable	-	-	
1	1	Setting prohibited					

When an interrupt is acknowledged, IST1 and IST0 are saved into the stack memory together with other PSW and is changed to a status higher by one level. When RET1 instruction is executed, the original IST1 and IST0 values are reset.

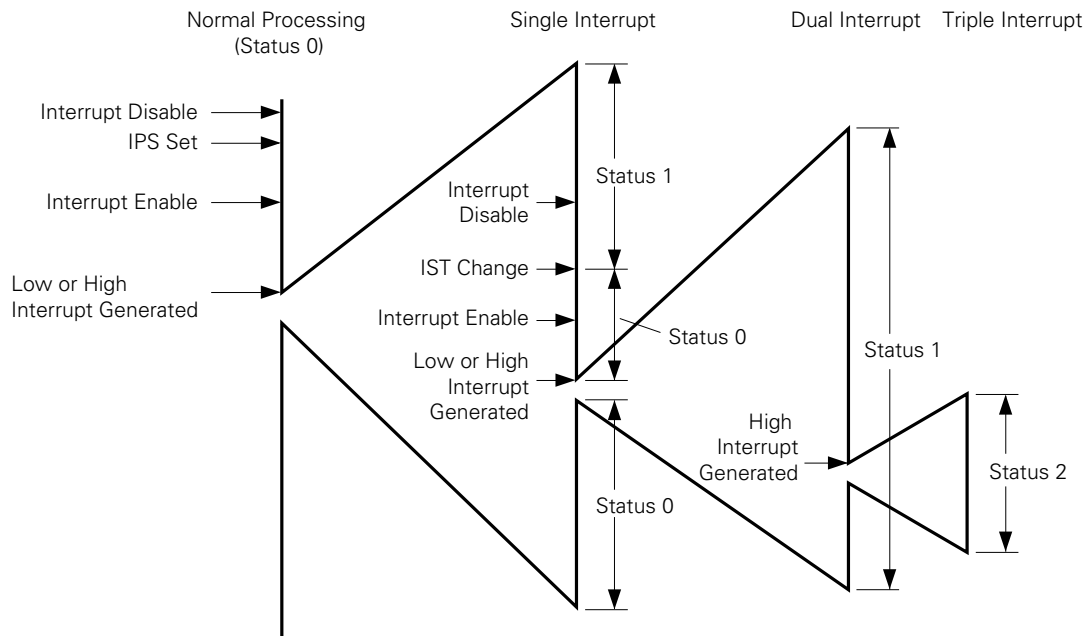
**(2) Multi-interruption changing the interrupt status flag**

As is clear from Table 5-3, multi-interrupt is enabled by changing the interrupt status flag using the program. That is, multi-interrupt is enabled by changing IST1 and IST0 each to "0" using the interrupt servicing program and setting status 0.

This method is used to enable multi-interrupt with two to more interrupts or multi-interruption with triple or more interrupts.

Before changing IST1 and IST0, disable interruption by DI instruction.

**Fig. 5-7 Multi-Interruption by Changing the Interrupt Status Flag**



## 5.5 VECTOR ADDRESS SHARING INTERRUPT SERVICING

Since the INTBT and INT4 interrupt sources share the vector table, interrupt source selection is carried out as follows:

### (1) When only one interrupt source is used

Among the two interrupt sources sharing the vector table, set the interrupt enable flag of the necessary interrupt source ("1") and clear the other interrupt enable flag ("0"). In this case, an interrupt request is generated by the enabled interrupt source (IE<sub>xxx</sub>=1). When the request is acknowledged, the corresponding interrupt request flag is reset (as is the case with an interrupt not sharing the vector address).

### (2) When both interrupt sources are used

Set the interrupt enable flags corresponding to the two interrupt sources ("1"). In this case, the logical sum of the interrupt request flags of the two interrupt sources becomes an interrupt request.

And, if an interrupt request by the setting of one or both interrupt request flags is acknowledged, none of the interrupt request flag is reset.

Accordingly, it is necessary to check in the interrupt service routing by which interrupt source the interrupt has been generated. It can be done by executing the DI instruction at the beginning of the interrupt service routine and checking the interrupt request flag by the SKTCLR instruction.

## 6. STANDBY FUNCTIONS

Two standby modes (STOP mode and HALT mode) are available for the μPD75237 to decrease power consumption in the program standby mode.

### 6.1 STANDBY MODE SETTING AND OPERATING STATE

**Table 6-1 Operation Status in Standby Mode**

		STOP Mode	HALT Mode
Set instruction		STOP instruction	HALT instruction
System clock when set		Setting enabled only with main system clock.	Setting enabled with either main system clock or subsystem clock.
Operating State	Clock oscillator	Oscillator stops only with main system clock.	Stops only with CPU clock $\phi$ (Oscillation continued).
	Serial interface (channel 0)	Operation enabled only when external $\overline{SCK0}$ input is selected for serial clock.	Operation enabled when the main system clock oscillates or with external $\overline{SCK0}$ .
	Serial interface (channel 1)	Operation enabled only when external $\overline{SCK1}$ input is selected for serial clock.	Operation enabled only when the main system clock oscillates.
	Basic interval timer	Operation stopped.	Operation (IRQBT set at reference time intervals).
	Timer/event counter	Operation enabled only when TI0 pin input is specified for count clock.	Operation enabled.
	Watch timer	Operation enabled only fxt is selected for count clock.	Operation enabled.
	Timer/pulse generator	Operation stopped.	Operation enabled only when the main system clock oscillates.
	Event counter	Operation stopped.	Operation enabled only when the main system clock oscillates.
	A/D converter	Operation stopped.	Operation enabled only when the main system clock oscillates.
	FIP controller/driver	Operation disabled (display off mode set before disabling).	
	External interrupt	INT0 operation disabled. INT1, INT2 and INT4 operation enabled.	
CPU	Operation stopped.		
Release signal		Interrupt request signal or $\overline{RESET}$ input from operational hardware enabled by interrupt enable flag.	

The STOP and HALT modes are set by STOP and HALT instructions, respectively. (The two instructions are instructions to set PCC bit 3 and bit 2, respectively.)

When changing the CPU operation clock with the least significant 2 bits of PCC, a delay may result from PCC rewrite to CPU clock change as shown in Table 4-1. Thus, when changing the operation clock before the standby mode is set or the CPU clock after the standby mode is released, set the standby mode after the passage of the machine cycle required for CPU clock change following PCC rewrite.

In the standby mode, the data of all registers and data memories which stop operating is held. Such units include general registers, flag, mode registers and output latches.

- Note 1. When the STOP mode is set, X1 input is internally short-circuited to V<sub>ss</sub> (GND potential) to prevent leakage from the crystal resonator unit. Thus, the use of STOP mode is prohibited in a system using external clocks.**
- 2. Because the interrupt request signal is used to release the standby mode, the standby mode is immediately released if there is an interrupt source with both the interrupt request flag and interrupt enable flag set. Thus, the STOP mode is set to the HALT mode just after STOP instruction execution. After waiting for the time period set by the BTM register, the operating mode is reset.**

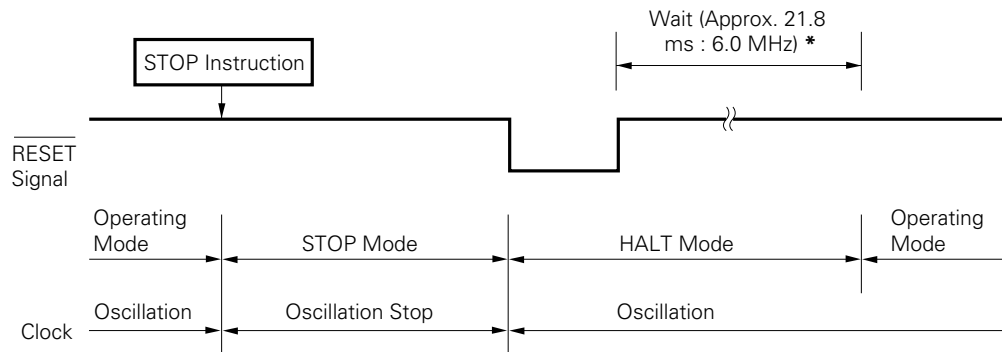
6.2 STANDBY MODE RELEASE

The STOP and HALT modes each are released upon generation of the interrupt request signal\* enabled by the interrupt enable flag or by RESET input. Fig. 6-1 shows release operation in each mode.

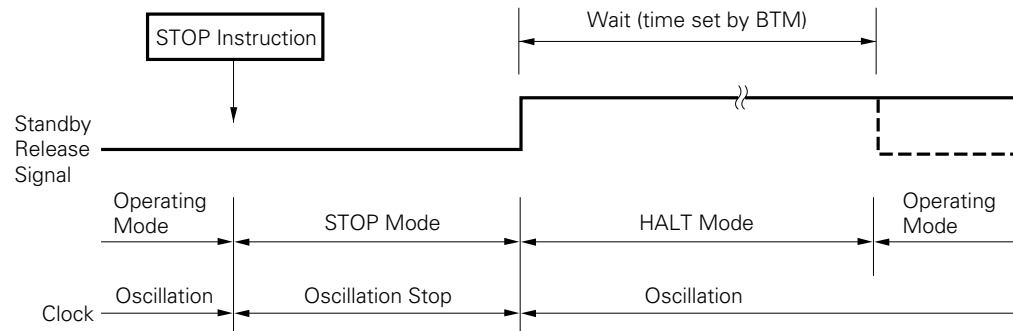
\* Except INT0 to INT2.

Fig. 6-1 Standby Mode Release Operation (1/2)

(a) Release by RESET input in STOP mode

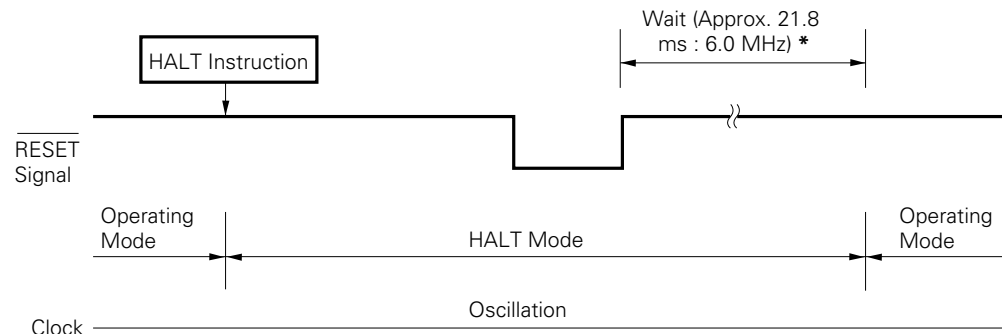


(b) Release by interrupt generation in STOP mode



**Remarks** The broken line shows the case in which the interrupt request which released the standby mode has been acknowledged (IME = 1).

(c) Release by RESET input in HALT mode

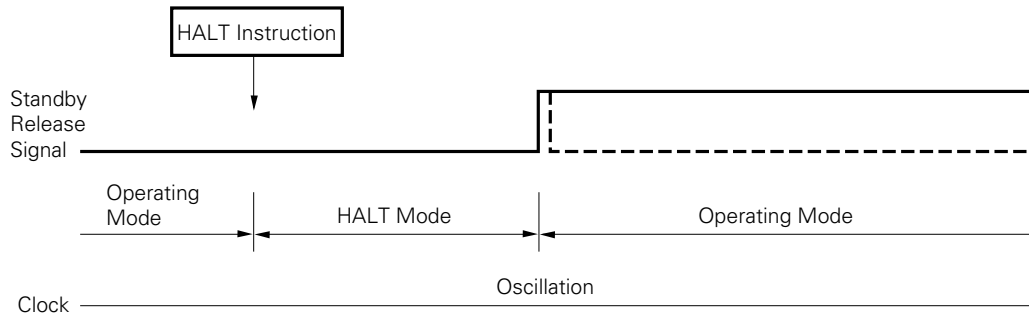


\* 31.3 ms at 4.19 MHz operation



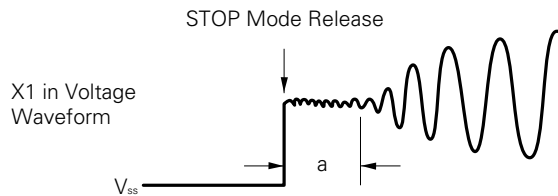
Fig. 6-1 Standby Mode Release Operation (2/2)

(d) Release by interrupt generation in HALT mode



**Remarks** The broken line shows the case in which the interrupt request which released the standby mode has been acknowledged (IME = 1).

The wait time upon STOP mode release does not include a time from STOP mode release to clock oscillation start ("a" below) whether the STOP mode is released by RESET input or interrupt generation.



If the STOP mode has been released by interrupt generation, the wait time is determined by BTM setting. (Refer to Table 6-2.)

Table 6-2 Wait Time Selection by BTM

(When f<sub>x</sub> = 6.0 MHz)

BTM3	BTM2	BTM1	BTM0	Wait Time* (Values at f <sub>x</sub> = 6.0 MHz are shown in parentheses)
-	0	0	0	Approx. 2 <sup>20</sup> /f <sub>x</sub> (approx. 175 ms)
-	0	1	1	Approx. 2 <sup>17</sup> /f <sub>x</sub> (approx. 21.8 ms)
-	1	0	1	Approx. 2 <sup>15</sup> /f <sub>x</sub> (approx. 5.46 ms)
-	1	1	1	Approx. 2 <sup>13</sup> /f <sub>x</sub> (approx. 1.37 ms)
In all other cases				Setting prohibited

(When f<sub>x</sub> = 4.19 MHz)

BTM3	BTM2	BTM1	BTM0	Wait Time* (Values at f <sub>x</sub> = 4.19 MHz are shown in parentheses)
-	0	0	0	Approx. 2 <sup>20</sup> /f <sub>x</sub> (approx. 250 ms)
-	0	1	1	Approx. 2 <sup>17</sup> /f <sub>x</sub> (approx. 31.3 ms)
-	1	0	1	Approx. 2 <sup>15</sup> /f <sub>x</sub> (approx. 7.82 ms)
-	1	1	1	Approx. 2 <sup>13</sup> /f <sub>x</sub> (approx. 1.95 ms)
In all other cases				Setting prohibited

\* Wait time does not include a time from STOP mode release to oscillation start.

### 6.3 OPERATION AFTER STANDBY MODE RELEASE

- (1) If the STOP mode has been released by  $\overline{\text{RESET}}$  input, normal reset operation is carried out.
- (2) If the STOP mode has been released by interrupt generation, the bit 3 (IME) contents of the IPS determine whether a vectored interrupt should be executed when the CPU resumes instruction execution.

**(a) When IME = "0"**

Execution is resumed with the instruction (NOP instruction) following standby mode setting after the standby mode has been released. The interrupt request flag is held.

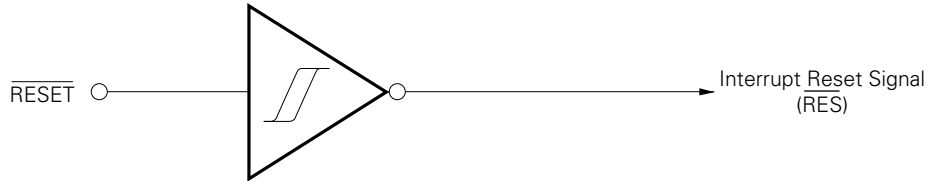
**(b) When IME = "1"**

Vectored interrupt is executed following execution of two instructions after the standby mode has been released. If the standby mode has been released by INTW (testable input), no vectored interrupt is generated; so the same processing as with (a) is carried out.

### 7. RESET FUNCTIONS

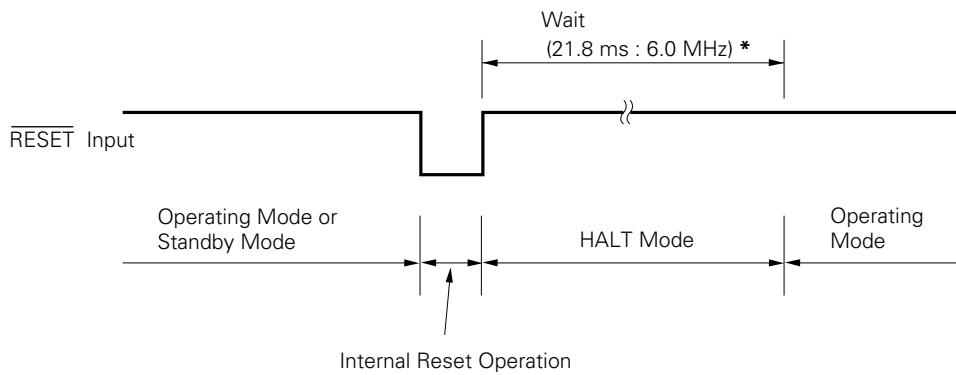
The reset signal ( $\overline{\text{RES}}$ ) generator has a configuration shown in Fig. 7-1.

**Fig. 7-1 Reset Signal Generator**



Reset operation is shown in Fig. 7-2.  
 The output buffer is turned OFF upon  $\overline{\text{RESET}}$  input.  
 Table 7-1 shows each hardware status after reset.

**Fig. 7-2 Reset Operation by  $\overline{\text{RESET}}$  input**



\* 31.3 ms at 4.19 MHz operation

Table 7-1 shows each hardware status after reset.

Table 7-1 Hardware Statuses after Reset (1/2)

Hardware		$\overline{\text{RESET}}$ Input in Standby Mode	$\overline{\text{RESET}}$ Input in Operation
Program counter (PC)		Sets the low-order 6 bits of program memory address 0000H to PC <sub>13-8</sub> and the contents of address 0001H to PC <sub>7-0</sub> .	←
PSW	Carry flag (CY)	Hold	Undefined
	Skip flag (SK0-SK2)	0	0
	Interrupt status flag (IST1, IST2)	0	0
	Bank enable flags (MBE, RBE)	Sets bit 6 of program memory address 0000H to RBE and bit 7 to MBE.	←
Data memory (RAM)		Hold	Undefined
General registers (X, A, H, L, D, E, B, C)		Hold	Undefined
Bank select registers (MBS, RBS)		0, 0	0, 0
Stack pointer (SP)		Undefined	Undefined
Stack bank select register (SBS)		Undefined	Undefined
Basic interval timer	Counter (BT)	Undefined	Undefined
	Mode register (BTM)	0	0
Timer/event counter	Counter (T0)	0	0
	Modulo register (TMOD0)	FFH	FFH
	Mode register (TM0)	0	0
	TOE0, TOUT F/F	0,0	0, 0
Watch timer	Mode register (WM)	0	0
Timer/pulse generator	Modulo register (MODH, MODL)	Hold	Hold
	Mode register (TPGM)	0	0
Event counter	Counter (T1)	0	0
	Mode register (TM1)	0	0
	Gate control register (GATEC)	0	0
Serial interface (channel 0)	Shift register (SIO0)	Hold	Undefined
	Operating mode register (CSIM0)	0	0
	SBI control register (SBIC)	0	0
	Slave address register (SVA)	Hold	Undefined
	P01/ $\overline{\text{SCK0}}$ output latch	1	1
Serial interface (channel 1)	Shift register (SIO1)	Hold	Undefined
	Operating mode register (CSIM1)	0	0
	Serial transfer end flag (EOT)	0	0

Table 7-1 Hardware Statuses after Reset (2/2)

Hardware		$\overline{\text{RESET}}$ Input in Standby Mode	$\overline{\text{RESET}}$ Input in Operation
A/D converter	Mode register (ADM), EOC	04H (EOC = 1)	04H (EOC = 1)
	SA register	Undefined	Undefined
Bit sequential buffer (BSB0 to BSB3)		Hold	Undefined
FIP controller/ driver	Mode register (DSPM)	0	0
	Dimmer select register (DIMS)	0	0
	Digit select register (DIGS)	8H	8H
	Display data memory	Hold	Hold
	Output buffer	OFF	OFF
	Static mode register (STATA, STATB)	0, 0	0, 0
Clock genera- tor and clock output circuit	Processor clock control register (PCC)	0	0
	System clock control register (SCC)	0	0
	Clock output mode register (CLOM)	0	0
Interrupt function	Interrupt request flag (IRQ <sub>xxx</sub> )	Reset	Reset
	Interrupt enable flag (IE <sub>xxx</sub> )	0	0
	Interrupt master enable flag (IME)	0	0
	INT0 and INT1 mode registers (IM0, IM1)	0, 0	0, 0
Digital port	Output buffer (ports 2 to 7)	OFF	OFF
	Output latch (ports 2 to 7)	Clear	Clear
	Input/output mode register (PMGA, PMGB)	0	0
	Pull-up resistor specify register (POGA)	0	0
Ports 10 to 15	Output buffer	OFF	OFF
	Output latch	0	0
Port H	Output latch	Hold	Undefined

8. INSTRUCTION SET

8.1 CHARACTERISTIC INSTRUCTIONS OF μPD75237

(1) GETI instruction

The GETI instruction is a 1-byte instruction to execute the following three types of operations by referring to the 2-byte table in the program memory.

It can considerably help to decrease the number of program steps.

- (a) Subroutine call to 16K-byte space (0000H to 3FFFH) of table data as call instruction call address.
- (b) Branch to 16K-byte space (0000H to 3FFFH) of table data as branch instruction branch address
- (c) Execution of table data as 2-byte instruction (except BRCB and CALLF instructions)
- (d) Execution of table data as 1-byte instruction and 2 operation codes.

As shown in Fig. 3-2, the table addressed referred to by GETI instruction as 0020H to 007FH of the program memory and data can be set in 48 tables.

When describing table addresses as operands, describe even addresses.

- Note**
- 1. 2-byte instructions which can be referred to by GETI instruction are limited to 2-machine cycle instructions.
  - 2. When referring to two 1-byte instructions by GETI instruction, combinations are limited as follows.

1st Byte Instruction	2nd Byte Instruction
MOV A, @HL	( INCS L DECS L
MOV @HL, A	( INCS H DECS H
XCH A, @HL	INCS HL
MOV A, @DE	( INCS E DECS E
XCH A, @DE	( INCS D DECS D INCS DE
MOV A, @DL	( INCS L DECS L
XCH A, @DL	( INCS D DECS D

- 3. When a branch instruction or subroutine instruction is referenced by a GETI instruction, the relevant branch destination or subroutine call address must be within 16K bytes (0000H to 3FFFH). It is not possible to use a GETI instruction to reference a branch instruction or subroutine call instruction to an address outside this range (4000H to 5F7FH).

Since the PC does not increment during execution of GETI instruction, it continues processing with the address following GETI instruction.

If an instruction preceding the GETI instruction has the skip function, the GETI instruction is skipped as is the case with all other 1-byte instructions. If the instruction referred to by the GETI instruction has the skip function, an instruction following the GETI instruction is skipped.

When instructions having stack effects are referred to by the GETI instruction, the following operations are carried out:

- If an instruction preceding GETI instruction also has the stack effects of the same group, the execution of GETI instruction eliminates the stack effects and the instructions referred to are not skipped.
- If an instruction following GETI instruction also has the stack effects of the same group, the stack effects derived from the instructions referred to are valid and the following instruction is skipped.

**(2) Bit manipulation instruction**

In addition to normal bit manipulation instructions (set and clear instructions), the bit test instruction, bit transfer instruction and bit Boolean instructions (AND, OR, XOR) are available for the μPD75237. Manipulation bits are specified by bit manipulation addressing.

Three types of available addressing operations and bits manipulated by each addressing are shown below.

Addressing	Specifiable Peripheral Hardware	Specifiable Bit Address Range
fmem.bit	RBE/MBE/IST1, IST0/IExxx/IRQxxx	FB0H to FBFH
	PORT0 to PORT15	FF0H to FFFH
pmem.@L	BSB0 to BSB3, PORT0 to PORT15	FC0H to FFFH
@H+mem.bit	All peripheral hardware devices enabled for bit manipulation	All manipulatable bits of the memory bank specified by MB

- Remarks**
1. xxx : 0, 1, 2, 3, 4, BT, T0, TPG, CSI0, KS, W
  2. MB = MBE• MBS

**(3) Stack instructions**

If the instructions of the same group of the following three instructions are stacked (set at two or more continuous addresses) in the program, the stack instruction placed at the start point is executed. In the subsequent execution, one stack instruction is replaced with one NOP instruction.

Group A: MOV A, #n4,      MOV XA, #n8  
Group B: MOV HL, #n8

**(4) Radix adjustment instructions**

Radix adjustment instructions to adjust the result of 4-bit data addition or subtraction to any radix is available for the  $\mu$ PD75237.

When the radix to be adjusted is m.

- ADD      ADDS A, #16-m  
            ADDC A, @HL  
            ADDS A, #m
- Subtract    SUBC A, @HL  
            ADDS A, #m

Using the above combinations, the addition/subtraction result with the memory addressed by the accumulator and register pair HL is adjusted to a m-ary radix. In the case of subtraction, m's complement of the subtraction result is set to the accumulator. The overflow/underflow remains in the carry flag (in these instruction combinations, the "ADDS A, #m" instruction skip function is disabled).



## 8.2 INSTRUCTION SET AND OPERATION

### (1) Operand identifier and description

Enter an operand in the operand column of each instruction using the description method relating to the operand identifier of the instruction (refer to the assembler specifications for details). If more than one description method is available, select one. Capital alphabetic letters, plus and minus signs are keywords. Describe them as they are.

In the case of immediate data, describe appropriate numerical values or labels.

Symbols in the register and flag format diagrams in chapters 3 to 5 can be described as labels in place of mem, fmem, pmem, bit, etc. (Available labels are limited for fmem and pmem. Refer to **8.1 (2) Bit manipulation instruction.**)

Identifier	Description Method
reg	X, A, B, C, D, E, H, L
reg 1	X, B, C, D, E, H, L
rp	XA, BC, DE, HL
rp1	BC, DE, HL
rp2	BC, DE
rp'	XA, BC, DE, HL, XA', BC', DE', HL'
rp'1	BC, DE, HL, XA', BC', DE', HL'
rpa	HL, HL+, HL-, DE, DL
rpa1	DE, DL
n4	4-bit immediate data or label
n8	8-bit immediate data or label
mem	8-bit immediate data or label*
bit	2-bit immediate data or label
fmem	FB0H to FBFH and FF0H to FFFH immediate data or labels
pmem	FC0H to FFFH immediate data or labels
addr1	0000H to 5F7FH immediate data or labels
addr	0000H to 3F7FH immediate data or labels
caddr	12-bit immediate data or label
faddr	11-bit immediate data or label
taddr	20H to 7FH immediate data (bit0 = 0) or label
PORTn	PORT0 to PORT15
IExxx	IEBT, IECSi0, IET0, IETPG, IE0, IE1, IE2, IEKS, IEW, IE4
RBn	RB0 to RB3
MBn	MB0, MB1, MB2, MB3, MB15

\* For 8-bit data processing, only even addresses can be specified.

**(2) Legend for operation description**

A	: A register; 4-bit accumulator
B	: B register
C	: C register
D	: D register
E	: E register
H	: H register
L	: L register
X	: X register
XA	: Register pair (XA); 8-bit accumulator
BC	: Register pair (BC)
DE	: Register pair (DE)
HL	: Register pair (HL)
XA'	: Expanded register pair (XA')
BC'	: Expanded register pair (BC')
DE'	: Expanded register pair (DE')
HL'	: Expanded register pair (HL')
PC	: Program counter
SP	: Stack pointer
SBS	: Stack bank select register
CY	: Carry flag; Bit accumulator
PSW	: Program status word
MBE	: Memory bank enable flag
RBE	: Register bank enable flag
PORT <sub>n</sub>	: Port n (n = 0 to 15)
IME	: Interrupt master enable flag
IPS	: Interrupt priority select register
IE <sub>xxx</sub>	: Interrupt enable flag
RBS	: Register bank select register
MBS	: Memory bank select register
PCC	: Processor clock control register
•	: Address and bit delimiter
(xx)	: Contents addressed by xx
xxH	: Hexadecimal data

(3) Description of symbols in the addressing area column

* 1	MB = MBE • MBS (MBS = 0, 1, 2, 3, 15)	Data Memory Addressing
* 2	MB = 0	
* 3	MBE = 0 : MB = 0 (00H to 7FH) MB = 15 (80H to FFH) MBE = 1 : MB = MBS (MBS = 0, 1, 2, 3, 15)	
* 4	MB = 15, fmem = FB0H to FBFH, FF0H to FFFH	
* 5	MB = 15, pmem = FC0H to FFFH	
* 6	addr = 0000H to 3FFFH	Program Memory Addressing
* 7	addr = (Current PC) – 15 to (Current PC) – 1, (Current PC) + 2 to (Current PC) + 16	
* 8	caddr = 0000H to 0FFFH (PC <sub>14, 13, 12</sub> = 00B) or 1000H to 1FFFH (PC <sub>14, 13, 12</sub> = 01B) or 2000H to 2FFFH (PC <sub>14, 13, 12</sub> = 10B) or 3000H to 3FFFH (PC <sub>14, 13, 12</sub> = 11B) or 4000H to 4FFFH (PC <sub>14, 13, 12</sub> = 100B) or 5000H to 5F7FH (PC <sub>14, 13, 12</sub> = 101B)	
* 9	faddr = 0000H to 07FFH	
* 10	taddr = 0020H to 007FH	
* 11	addr1 = 0000H to 5F7FH	

- Remarks**
1. MB indicates accessible memory bank.
  2. In \*2, MB = 0 irrespective of MBE and MBS.
  3. In \*4 and \*5, MB = 15 irrespective of MBE and MBS.
  4. \*6 to \*10 indicate addressable areas.

(4) Description of the machine cycle column

S indicates the number of machine cycles required for skip operation by an instruction having skip function. The S value varies as follows:

- When not skipped ..... S = 0
- When 1-byte or 2-byte instructions are skipped ..... S = 1
- When 3-byte instructions are skipped ..... S = 2

**Note** GETI instruction is skipped in one machine cycle.

One machine cycle is equal to one cycle of CPU clock  $\Phi$  and five time periods are available according to PCC and SCC setting. (Refer to 4.2 (3) Processor clock control register (PCC).)

Note	Mnemonic	Operands	No. of Bytes	Machine Cycle	Operation	Addressing Area	Skip Condition
Transfer	MOV	A, #n4	1	1	A←n4		Stack A
		reg1, #n4	2	2	reg1←n4		
		XA, #n8	2	2	XA←n8		Stack A
		HL, #n8	2	2	HL←n8		Stack B
		rp2, #n8	2	2	rp2←n8		
		A, @HL	1	1	A←(HL)	*1	
		A, @HL+	1	2 + S	A←(HL), then L←L+1	*1	L = 0
		A, @HL-	1	2 + S	A←(HL), then L←L-1	*1	L = FH
		A, @rpa1	1	1	A←(rpa1)	*2	
		XA, @HL	2	2	XA←(HL)	*1	
		@HL, A	1	1	(HL)←A	*1	
		@HL, XA	2	2	(HL)←XA	*1	
		A, mem	2	2	A←(mem)	*3	
		XA, mem	2	2	XA←(mem)	*3	
		mem, A	2	2	(mem)←A	*3	
		mem, XA	2	2	(mem)←XA	*3	
		A, reg	2	2	A←reg		
		XA, rp'	2	2	XA←rp'		
		reg1, A	2	2	reg1←A		
	rp'1, XA	2	2	rp'1←XA			
	XCH	A, @HL	1	1	A↔(HL)	*1	
		A, @HL+	1	2 + S	A↔(HL), then L←L+1	*1	L = 0
		A, @HL-	1	2 + S	A↔(HL), then L←L-1	*1	L = FH
		A, @rpa1	1	1	A↔(rpa1)	*2	
		XA, @HL	2	2	XA↔(HL)	*1	
		A, mem	2	2	A↔(mem)	*3	
		XA, mem	2	2	XA↔(mem)	*3	
		A, reg1	1	1	A↔reg1		
XA, rp'		2	2	XA↔rp'			
Table reference	MOVT	XA, @PCDE	1	3	XA←(PC <sub>14-8</sub> +DE) <sub>ROM</sub>		
		XA, @PCXA	1	3	XA←(PC <sub>14-8</sub> +XA) <sub>ROM</sub>		
		XA, @BCDE	1	3	XA←(BCDE) <sub>ROM</sub>	*11	
		XA, @BCXA	1	3	XA←(BCXA) <sub>ROM</sub>	*11	

**Note** Instruction Group

Note 1	Mnemonic	Operand	No. of Bytes	Machine Cycle	Operation	Addressing Area	Skip Condition
Bit transfer	MOV1	CY, fmem.bit	2	2	$CY \leftarrow (\text{fmem.bit})$	*4	
		CY, pmem.@L	2	2	$CY \leftarrow (\text{pmem}_{7-2} + L_{3-2}.\text{bit}(L_{1-0}))$	*5	
		CY, @H+mem.bit	2	2	$CY \leftarrow (\text{H} + \text{mem}_{3-0}.\text{bit})$	*1	
		fmem.bit, CY	2	2	$(\text{fmem.bit}) \leftarrow CY$	*4	
		pmem.@L, CY	2	2	$(\text{pmem}_{7-2} + L_{3-2}.\text{bit}(L_{1-0})) \leftarrow CY$	*5	
		@H+mem.bit, CY	2	2	$(\text{H} + \text{mem}_{3-0}.\text{bit}) \leftarrow CY$	*1	
Operation	ADDS	A, #n4	1	1 + S	$A \leftarrow A + n4$		carry
		XA, #n8	2	2 + S	$XA \leftarrow XA + n8$		carry
		A, @HL	1	1 + S	$A \leftarrow A + (\text{HL})$	*1	carry
		XA, rp'	2	2 + S	$XA \leftarrow XA + rp'$		carry
		rp'1, XA	2	2 + S	$rp'1 \leftarrow rp'1 + XA$		carry
	ADDC	A, @HL	1	1	$A, CY \leftarrow A + (\text{HL}) + CY$	*1	
		XA, rp'	2	2	$XA, CY \leftarrow XA + rp' + CY$		
		rp'1, XA	2	2	$rp'1, CY \leftarrow rp'1 + XA + CY$		
	SUBS	A, @HL	1	1 + S	$A \leftarrow A - (\text{HL})$	*1	borrow
		XA, rp'	2	2 + S	$XA \leftarrow XA - rp'$		borrow
		rp'1, XA	2	2 + S	$rp'1 \leftarrow rp'1 - XA$		borrow
	SUBC	A, @HL	1	1	$A, CY \leftarrow A - (\text{HL}) - CY$	*1	
		XA, rp'	2	2	$XA, CY \leftarrow XA - rp' - CY$		
		rp'1, XA	2	2	$rp'1, CY \leftarrow rp'1 - XA - CY$		
	AND	A, #n4	2	2	$A \leftarrow A \wedge n4$		
		A, @HL	1	1	$A \leftarrow A \wedge (\text{HL})$	*1	
		XA, rp'	2	2	$XA \leftarrow XA \wedge rp'$		
		rp'1, XA	2	2	$rp'1 \leftarrow rp'1 \wedge XA$		
	OR	A, #n4	2	2	$A \leftarrow A \vee n4$		
		A, @HL	1	1	$A \leftarrow A \vee (\text{HL})$	*1	
XA, rp'		2	2	$XA \leftarrow XA \vee rp'$			
rp'1, XA		2	2	$rp'1 \leftarrow rp'1 \vee XA$			
XOR	A, #n4	2	2	$A \leftarrow A \veebar n4$			
	A, @HL	1	1	$A \leftarrow A \veebar (\text{HL})$	*1		
	XA, rp'	2	2	$XA \leftarrow XA \veebar rp'$			
	rp'1, XA	2	2	$rp'1 \leftarrow rp'1 \veebar XA$			
Note 2	RORC	A	1	1	$CY \leftarrow A_0, A_3 \leftarrow CY, A_{n-1} \leftarrow A_n$		
	NOT	A	2	2	$A \leftarrow \bar{A}$		

**Note** 1. Instruction Group  
 2. Accumulator manipulation

Note	Mnemonic	Operands	No. of Bytes	Machine Cycle	Operation	Addressing Area	Skip Condition
Increment/decrement	INCS	reg	1	1 + S	reg←reg+1		reg = 0
		rp1	1	1 + S	rp1←rp1+1		rp1 = 00H
		@HL	2	2 + S	(HL)←(HL)+1	*1	(HL) = 0
		mem	2	2 + S	(mem)←(mem)+1	*3	(mem) = 0
	DECS	reg	1	1 + S	reg←reg-1		reg = FH
		rp'	2	2 + S	rp'←rp'-1		rp' = FFH
Compare	SKE	reg, #n4	2	2 + S	Skip if reg = n4		reg = n4
		@HL, #n4	2	2 + S	Skip if (HL) = n4	*1	(HL) = n4
		A, @HL	1	1 + S	Skip if A = (HL)	*1	A = (HL)
		XA, @HL	2	2 + S	Skip if XA = (HL)	*1	XA = (HL)
		A, reg	2	2 + S	Skip if A = reg		A = reg
		XA,rp'	2	2 + S	Skip if XA = rp'		XA = rp'
Carry flag manipulation	SET1	CY	1	1	CY←1		
	CLR1	CY	1	1	CY←0		
	SKT	CY	1	1 + S	Skip if CY = 1		CY = 1
	NOT1	CY	1	1	CY← $\overline{CY}$		

**Note** Instruction Group

Note	Mnemonic	Operands	No. of Bytes	Machine Cycle	Operation	Addressing Area	Skip Condition
Memory bit manipulation	SET1	mem.bit	2	2	(mem.bit)←1	*3	
		fmem.bit	2	2	(fmem.bit)←1	*4	
		pmem.@L	2	2	(pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> ))←1	*5	
		@H + mem.bit	2	2	(H+mem <sub>3-0</sub> .bit)←1	*1	
	CLR1	mem.bit	2	2	(mem.bit)←0	*3	
		fmem.bit	2	2	(fmem.bit)←0	*4	
		pmem.@L	2	2	(pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> ))←0	*5	
		@H+mem.bit	2	2	(H+mem <sub>3-0</sub> .bit)←0	*1	
	SKT	mem.bit	2	2 + S	Skip if (mem.bit) = 1	*3	(mem.bit) = 1
		fmem.bit	2	2 + S	Skip if (fmem.bit) = 1	*4	(fmem.bit) = 1
		pmem.@L	2	2 + S	Skip if (pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> )) = 1	*5	(pmem.@L) = 1
		@H+mem.bit	2	2 + S	Skip if (H+mem <sub>3-0</sub> .bit) = 1	*1	(@H+mem.bit) = 1
	SKF	mem.bit	2	2 + S	Skip if (mem.bit) = 0	*3	(mem.bit) = 0
		fmem.bit	2	2 + S	Skip if (fmem.bit) = 0	*4	(fmem.bit) = 0
		pmem.@L	2	2 + S	Skip if (pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> )) = 0	*5	(pmem.@L) = 0
		@H+mem.bit	2	2 + S	Skip if (H+mem <sub>3-0</sub> .bit) = 0	*1	(@H+mem.bit) = 0
	SKTCLR	fmem.bit	2	2 + S	Skip if (fmem.bit) = 1 and clear	*4	(fmem.bit) = 1
		pmem.@L	2	2 + S	Skip if (pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> ))=1 and clear	*5	(pmem.@L) = 1
		@H+mem.bit	2	2 + S	Skip if (H+mem <sub>3-0</sub> .bit)=1 and clear	*1	(@H+mem.bit)=1
	AND1	CY, fmem.bit	2	2	CY←CY∧(fmem.bit)	*4	
		CY, pmem.@L	2	2	CY←CY∧(pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> ))	*5	
		CY, @H+mem.bit	2	2	CY←CY∧(H+mem <sub>3-0</sub> .bit)	*1	
	OR1	CY, fmem.bit	2	2	CY←CY∨(fmem.bit)	*4	
		CY, pmem.@L	2	2	CY←CY∨(pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> ))	*5	
CY, @H+mem.bit		2	2	CY←CY∨(H+mem <sub>3-0</sub> .bit)	*1		
XOR1	CY, fmem.bit	2	2	CY←CY⊕(fmem.bit)	*4		
	CY, pmem.@L	2	2	CY←CY⊕(pmem <sub>7-2</sub> +L <sub>3-2</sub> .bit(L <sub>1-0</sub> ))	*5		
	CY, @H+mem.bit	2	2	CY←CY⊕(H+mem <sub>3-0</sub> .bit)	*1		
Branch	BR	addr	—	—	PC <sub>14-0</sub> ←addr1 (Optimum instruction is selected from among BR !addr, BRA !addr1, BRCB !caddr and BR \$addr1 by an assembler.)	*11	
		\$addr1	1	2	PC <sub>14-0</sub> ←addr1	*7	
		!addr	3	3	PC <sub>14</sub> ←0, PC <sub>13-0</sub> ←!addr	*6	
		PCDE	2	3	PC <sub>14-0</sub> ←PC <sub>14-8</sub> +DE		
		PCXA	2	3	PC <sub>14-0</sub> ←PC <sub>14-8</sub> +XA		
		BCDE	2	3	PC <sub>14-0</sub> ←BCDE		
		BCXA	2	3	PC <sub>14-0</sub> ←BCXA		
	BRA	!addr1	3	3	PC <sub>14-0</sub> ←!addr1	*11	
BRCB	!caddr	2	2	PC <sub>14-0</sub> ←PC <sub>14, 13,12</sub> +caddr <sub>11-0</sub>	*8		

★

Note Instruction Group

Note	Mnemonic	Operands	No. of Bytes	Machine Cycle	Operation	Addressing Area	Skip Condition	
Subroutine stack control	CALL	laddr	3	4	(SP-5) (SP-6) (SP-3) (SP-4)←PC <sub>14-0</sub> (SP-2)←X, X, MBE, RBE PC <sub>14-0</sub> ←0, PC <sub>13-0</sub> ←addr, SP←SP-6	*6		
	CALLA	laddr1	3	3	(SP-5) (SP-6) (SP-3) (SP-4)←PC <sub>14-0</sub> (SP-2)←X, X, MBE, RBE PC <sub>14-0</sub> ←addr1, SP←SP-6	*11		
	CALLF	lfaddr	2	3	(SP-5) (SP-6) (SP-3) (SP-4)←PC <sub>14-0</sub> (SP-2)←X, X, MBE, RBE PC <sub>14-0</sub> ←0000, faddr, SP←SP-6	*9		
	RET		1	3	X, X, MBE, RBE←(SP+4) PC <sub>14-0</sub> ←(SP+1) (SP) (SP+3) (SP+2) SP←SP+6			
	RETS		1	3 + S	X, X, MBE, RBE←(SP+4) PC <sub>14-0</sub> ←(SP+1) (SP) (SP+3) (SP+2) SP←SP+6 then skip unconditionally		Unconditional	
	RETI		1	3	X, PC <sub>14, 13, 12</sub> ←(SP+1) PC <sub>11-0</sub> ←(SP) (SP+3) (SP+2) PSW←(SP+4) (SP+5), SP←SP+6			
	PUSH	rp		1	1	(SP-1) (SP-2)←rp, SP←SP-2		
		BS		2	2	(SP-1)←MBS, (SP-2)←RBS, SP←SP-2		
POP	rp		1	1	rp←(SP+1) (SP), SP←SP+2			
	BS		2	2	MBS←(SP+1), RBS←(SP), SP←SP+2			
Interrupt control	EI		2	2	IME(IPS.3)←1			
		IE <sub>xxx</sub>	2	2	IE <sub>xxx</sub> ←1			
	DI		2	2	IME(IPS.3)←0			
		IE <sub>xxx</sub>	2	2	IE <sub>xxx</sub> ←0			
Input/output	IN *	A, PORT <sub>n</sub>	2	2	A←PORT <sub>n</sub>			
		XA, PORT <sub>n</sub>	2	2	XA←PORT <sub>n+1</sub> , PORT <sub>n</sub>			
	OUT *	PORT <sub>n</sub> , A	2	2	PORT <sub>n</sub> ←A			
		PORT <sub>n</sub> , XA	2	2	PORT <sub>n+1</sub> , PORT <sub>n</sub> ←XA			
CPU control	HALT		2	2	Set HALT Mode (PCC.2←1)			
	STOP		2	2	Set STOP Mode (PCC.3←1)			
	NOP		1	1	No Operation			

\* MBE = 0 or MBE = 1 and MBE = 15 must be set for execution of IN/OUT instruction

**Note** Instruction Group



Note	Mnemonic	Operands	No. of Bytes	Machine Cycle	Operation	Addressing Area	Skip Condition
Special	SEL	R <sub>Bn</sub>	2	2	R <sub>B</sub> ← n (n = 0 to 3)		
		M <sub>Bn</sub>	2	2	M <sub>B</sub> ← n (n = 0, 1, 2, 3, 15)		
	GET1 *	taddr	1	3	• TBR instruction PC <sub>13-0</sub> ← (taddr) <sub>5-0</sub> + (taddr+1) PC <sub>14</sub> ← 0	*10	Depends on instructions referred to.
				4	• TCALL instruction (SP-5)(SP-6)(SP-3)(SP-4) ← PC <sub>14-0</sub> (SP-2) ← x, x, MBE, RBE PC <sub>13-0</sub> ← (taddr) <sub>5-0</sub> + (taddr+1) SP ← SP-6, PC <sub>14</sub> ← 0		
				3	• (taddr) (taddr+1) instruction executed in the case of instruction except TBR and TCALL instructions		

★

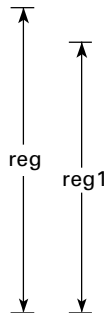
\* TBR and TCALL instructions are assembled pseudo-instructions to define the GETI instruction table.

**Note** Instruction Group

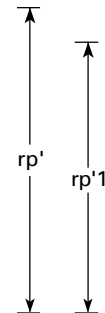
8.3 OPERATION CODES

(1) Description of operation code symbols

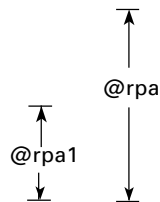
R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>	reg
0	0	0	A
0	0	1	X
0	1	0	L
0	1	1	H
1	0	0	E
1	0	1	D
1	1	0	C
1	1	1	B



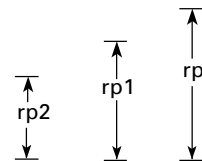
P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>	reg-pair
0	0	0	XA
0	0	1	XA'
0	1	0	HL
0	1	1	HL'
1	0	0	DE
1	0	1	DE'
1	1	0	BC
1	1	1	BC'



Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>	addressing
0	0	1	@HL
0	1	0	@HL+
0	1	1	@HL-
1	0	0	@DE
1	0	1	@DL



P <sub>2</sub>	P <sub>1</sub>	reg-pair
0	0	XA
0	1	HL
1	0	DE
1	1	BC



N <sub>5</sub>	N <sub>2</sub>	N <sub>1</sub>	N <sub>0</sub>	IE <sub>xxx</sub>
0	0	0	0	IEBT
0	0	1	0	IEW
0	0	1	1	IETPG
0	1	0	0	IETO
0	1	0	1	IECSI0
0	1	1	0	IE0
0	1	1	1	IE2
1	0	0	0	IE4
1	0	1	1	IEKS
1	1	1	0	IE1

In : Immediate data for n4 and n8

Dn : Immediate data for mem

Bn : Immediate data for bit

Nn : Immediate data for n and IE<sub>xxx</sub>

Tn : Immediate data for taddr × 1/2

An : Immediate data for [Relative address distance from branch destination address (2 to 16)]-1

Sn : Immediate data for one's complement of [Relative address distance from branch destination address (15 to 1)]

(2) **Operation codes of bit manipulation addressing**

\*1 in the operand column indicates that the following three addressings are available.

- fmem.bit
- pmem.@L
- @H+mem.bit

The 2nd byte \*2 of the operation code corresponding to the above addressing is shown below:

*1	2nd Byte of Operation Code	Accessible Bits
fmem.bit	1 0 B <sub>1</sub> B <sub>0</sub> F <sub>3</sub> F <sub>2</sub> F <sub>1</sub> F <sub>0</sub>	Manipulatable bits of FB0H to FBFH
	1 1 B <sub>1</sub> B <sub>0</sub> F <sub>3</sub> F <sub>2</sub> F <sub>1</sub> F <sub>0</sub>	Manipulatable bits of FF0H to FFFH
pmem.@L	0 1 0 0 G <sub>3</sub> G <sub>2</sub> G <sub>1</sub> G <sub>0</sub>	Manipulatable bits of FC0H to FFFH
@H+mem.bit	0 0 B <sub>1</sub> B <sub>0</sub> D <sub>3</sub> D <sub>2</sub> D <sub>1</sub> D <sub>0</sub>	Manipulatable bits of accessible memory banks

B<sub>n</sub> : Immediate data for bit

F<sub>n</sub> : Immediate data for fmem (indicating the low-order 4-bits of address)

G<sub>n</sub> : Immediate data for pmem (indicating the bits 5 to 2 of address)

D<sub>n</sub> : Immediate data for mem (indicating the low-order 4 bits of address)

Note 1	Mnemonic	Operands	Operation Code															
			B <sub>1</sub>				B <sub>2</sub>				B <sub>3</sub>							
Transfer	MOV	A, #n4	0	1	1	1	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>								
		reg1, #n4	1	0	0	1	1	0	1	0	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>	1	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>
		rp, #n8	1	0	0	0	1	P <sub>2</sub>	P <sub>1</sub>	1	l <sub>7</sub>	l <sub>6</sub>	l <sub>5</sub>	l <sub>4</sub>	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>
		A, @rpa	1	1	1	0	0	Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>								
		XA, @HL	1	0	1	0	1	0	1	0	0	0	0	1	1	0	0	0
		@HL, A	1	1	1	0	1	0	0	0								
		@HL, XA	1	0	1	0	1	0	1	0	0	0	0	1	0	0	0	0
		A, mem	1	0	1	0	0	0	1	1	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
		XA, mem	1	0	1	0	0	0	1	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	0
		mem, A	1	0	0	1	0	0	1	1	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
		mem, XA	1	0	0	1	0	0	1	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	0
		A, reg	1	0	0	1	1	0	0	1	0	1	1	1	1	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>
		XA, rp'	1	0	1	0	1	0	1	0	0	1	0	1	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
		reg1, A	1	0	0	1	1	0	0	1	0	1	1	1	0	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>
rp'1, XA	1	0	1	0	1	0	1	0	0	1	0	1	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>		
Transfer	XCH	A, @rpa	1	1	1	0	1	Q <sub>2</sub>	Q <sub>1</sub>	Q <sub>0</sub>								
		XA, @HL	1	0	1	0	1	0	1	0	0	0	0	1	0	0	0	1
		A, mem	1	0	1	1	0	0	1	1	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>
		XA, mem	1	0	1	1	0	0	1	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	0
		A, reg1	1	1	0	1	1	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>								
		XA, rp'	1	0	1	0	1	0	1	0	0	1	0	0	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
Table reference	MOV <sub>T</sub>	XA, @PCDE	1	1	0	1	0	1	0	0								
		XA, @PCXA	1	1	0	1	0	0	0	0								
		XA, @BCDE	1	1	0	1	0	1	0	1								
		XA, @BCXA	1	1	0	1	0	0	0	1								
Note 2	MOV <sub>1</sub>	CY, #1	1	0	1	1	1	1	0	1								*2
		#1, CY	1	0	0	1	1	0	1	1								*2

**Note** 1. Instruction Group  
 2. Bit transfer

Note 1	Mnemonic	Operands	Operation Code															
			B <sub>1</sub>				B <sub>2</sub>				B <sub>3</sub>							
Operate	ADDS	A, #n4	0	1	1	0	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>								
		XA, #n8	1	0	1	1	1	0	0	1	l <sub>7</sub>	l <sub>6</sub>	l <sub>5</sub>	l <sub>4</sub>	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>
		A, @HL	1	1	0	1	0	0	1	0								
		XA, rp'	1	0	1	0	1	0	1	0	1	1	0	0	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
		rp'1, XA	1	0	1	0	1	0	1	0	1	1	0	0	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
	ADDC	A, @HL	1	0	1	0	1	0	0	1								
		XA, rp'	1	0	1	0	1	0	1	0	1	1	0	1	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
		rp'1, XA	1	0	1	0	1	0	1	0	1	1	0	1	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
	SUBS	A, @HL	1	0	1	0	1	0	0	0								
		XA, rp'	1	0	1	0	1	0	1	0	1	1	1	0	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
		rp'1, XA	1	0	1	0	1	0	1	0	1	1	1	0	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
	SUBC	A, @HL	1	0	1	1	1	0	0	0								
		XA, rp'	1	0	1	0	1	0	1	0	1	1	1	1	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
		rp'1, XA	1	0	1	0	1	0	1	0	1	1	1	1	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
	AND	A, #n4	1	0	0	1	1	0	0	1	0	0	1	1	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>
		A, @HL	1	0	0	1	0	0	0	0								
		XA, rp'	1	0	1	0	1	0	1	0	1	0	0	1	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
		rp'1, XA	1	0	1	0	1	0	1	0	1	0	0	1	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>
	OR	A, #n4	1	0	0	1	1	0	0	1	0	1	0	0	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>
		A, @HL	1	0	1	0	0	0	0	0								
XA, rp'		1	0	1	0	1	0	1	0	1	0	1	0	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>	
rp'1, XA		1	0	1	0	1	0	1	0	1	0	1	0	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>	
XOR	A, #n4	1	0	0	1	1	0	0	1	0	1	0	1	l <sub>3</sub>	l <sub>2</sub>	l <sub>1</sub>	l <sub>0</sub>	
	A, @HL	1	0	1	1	0	0	0	0									
	XA, rp'	1	0	1	0	1	0	1	0	1	0	1	1	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>	
	rp'1, XA	1	0	1	0	1	0	1	0	1	0	1	1	0	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>	
Note 2	RORC	A	1	0	0	1	1	0	0	0								
	NOT	A	1	0	0	1	1	0	0	1	0	1	0	1	1	1	1	1

**Note** 1. Instruction Group  
 2. Accumulator manipulation

Note	Mnemonic	Operands	Operation Code																	
			B <sub>1</sub>						B <sub>2</sub>						B <sub>3</sub>					
Increment/decrement	INCS	reg	1	1	0	0	0	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>										
		rp1	1	0	0	0	1	P <sub>2</sub>	P <sub>1</sub>	0										
		@HL	1	0	0	1	1	0	0	1	0	0	0	0	0	0	1	0		
		mem	1	0	0	0	0	0	1	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>		
	DECS	reg	1	1	0	0	1	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>										
		rp'	1	0	1	0	1	0	1	0	0	1	1	0	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>		
Compare	SKE	reg, #n4	1	0	0	1	1	0	1	0	I <sub>3</sub>	I <sub>2</sub>	I <sub>1</sub>	I <sub>0</sub>	0	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>		
		@HL, #n4	1	0	0	1	1	0	0	1	0	1	1	0	I <sub>3</sub>	I <sub>2</sub>	I <sub>1</sub>	I <sub>0</sub>		
		A, @HL	1	0	0	0	0	0	0	0										
		XA, @HL	1	0	1	0	1	0	1	0	0	0	0	1	1	0	0	1		
		A, reg	1	0	0	1	1	0	0	1	0	0	0	0	1	R <sub>2</sub>	R <sub>1</sub>	R <sub>0</sub>		
		XA, rp'	1	0	1	0	1	0	1	0	0	1	0	0	1	P <sub>2</sub>	P <sub>1</sub>	P <sub>0</sub>		
Carry flag manipulation	SET1	CY	1	1	1	0	0	1	1	1										
	CLR1	CY	1	1	1	0	0	1	1	0										
	SKT	CY	1	1	0	1	0	1	1	1										
	NOT1	CY	1	1	0	1	0	1	1	0										
Memory bit manipulation	SET1	mem.bit	1	0	B <sub>1</sub>	B <sub>0</sub>	0	1	0	1	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>		
		*1	1	0	0	1	1	1	0	1	*2									
	CLR1	mem.bit	1	0	B <sub>1</sub>	B <sub>0</sub>	0	1	0	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>		
		*1	1	0	0	1	1	1	0	0	*2									
	SKT	mem.bit	1	0	B <sub>1</sub>	B <sub>0</sub>	0	1	1	1	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>		
		*1	1	0	1	1	1	1	1	1	*2									
	SKF	mem.bit	1	0	B <sub>1</sub>	B <sub>0</sub>	0	1	1	0	D <sub>7</sub>	D <sub>6</sub>	D <sub>5</sub>	D <sub>4</sub>	D <sub>3</sub>	D <sub>2</sub>	D <sub>1</sub>	D <sub>0</sub>		
		*1	1	0	1	1	1	1	1	0	*2									
	SKTCLR	*1	1	0	0	1	1	1	1	1	*2									
	AND1	CY, *1	1	0	1	0	1	1	0	0	*2									
	OR1	CY, *1	1	0	1	0	1	1	1	0	*2									
	XOR1	CY, *1	1	0	1	1	1	1	0	0	*2									

**Note** Instruction Group

Note	Mnemonic	Operands	Operation Code			
			B <sub>1</sub>	B <sub>2</sub>	B <sub>3</sub>	
Branch	BR	!addr	1 0 1 0 1 0 1 1	0 0 ←	addr →	
		\$addr1	(+16) to (+2)	0 0 0 0 A <sub>3</sub> A <sub>2</sub> A <sub>1</sub> A <sub>0</sub>		
			(-1) to (-15)	1 1 1 1 S <sub>3</sub> S <sub>2</sub> S <sub>1</sub> S <sub>0</sub>		
	BRA	!addr1	1 0 1 1 1 0 1 0	0 ←	addr1 →	
	BRCB	!caddr	0 1 0 1 ←	caddr →		
	BR	PCDE	1 0 0 1 1 0 0 1	0 0 0 0 0 1 0 0		
		PCXA	1 0 0 1 1 0 0 1	0 0 0 0 0 0 0 0		
		BCDE	1 0 0 1 1 0 0 1	0 0 0 0 0 1 0 1		
BCXA		1 0 0 1 1 0 0 1	0 0 0 0 0 0 0 1			
Subroutine stack control	CALL	!addr	1 0 1 0 1 0 1 1	0 1 ←	addr →	
	CALLA	!addr1	1 0 1 1 1 0 1 1	0 ←	addr1 →	
	CALLF	!faddr	0 1 0 0 0 ←	faddr →		
	RET		1 1 1 0 1 1 1 0			
	RETS		1 1 1 0 0 0 0 0			
	RETI		1 1 1 0 1 1 1 1			
	PUSH	rp	0 1 0 0 1 P <sub>2</sub> P <sub>1</sub> 1			
		BS	1 0 0 1 1 0 0 1	0 0 0 0 0 1 1 1		
POP	rp	0 1 0 0 1 P <sub>2</sub> P <sub>1</sub> 0				
	BS	1 0 0 1 1 0 0 1	0 0 0 0 0 1 1 0			
Input/output	IN	A, PORT <sub>n</sub>	1 0 1 0 0 0 1 1	1 1 1 1 N <sub>3</sub> N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
		XA, PORT <sub>n</sub>	1 0 1 0 0 0 1 0	1 1 1 1 N <sub>3</sub> N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
	OUT	PORT <sub>n</sub> , A	1 0 0 1 0 0 1 1	1 1 1 1 N <sub>3</sub> N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
		PORT <sub>n</sub> , XA	1 0 0 1 0 0 1 0	1 1 1 1 N <sub>3</sub> N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
Interrupt control	EI		1 0 0 1 1 1 0 1	1 0 1 1 0 0 1 0		
		IE <sub>xxx</sub>	1 0 0 1 1 1 0 1	1 0 N <sub>5</sub> 1 1 N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
	DI		1 0 0 1 1 1 0 0	1 0 1 1 0 0 1 0		
		IE <sub>xxx</sub>	1 0 0 1 1 1 0 0	1 0 N <sub>5</sub> 1 1 N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
CPU control	HALT		1 0 0 1 1 1 0 1	1 0 1 0 0 0 1 1		
	STOP		1 0 0 1 1 1 0 1	1 0 1 1 0 0 1 1		
	NOP		0 1 1 0 0 0 0 0			
Special	SEL	RB <sub>n</sub>	1 0 0 1 1 0 0 1	0 0 1 0 0 0 N <sub>1</sub> N <sub>0</sub>		
		MB <sub>n</sub>	1 0 0 1 1 0 0 1	0 0 0 1 N <sub>3</sub> N <sub>2</sub> N <sub>1</sub> N <sub>0</sub>		
	GETI	taddr	0 0 T <sub>5</sub> T <sub>4</sub> T <sub>3</sub> T <sub>2</sub> T <sub>1</sub> T <sub>0</sub>			

★

Note Instruction Group

**9. MASK OPTION SELECTION**

The μPD75237 has the following mask options enabling or disabling on-chip components.

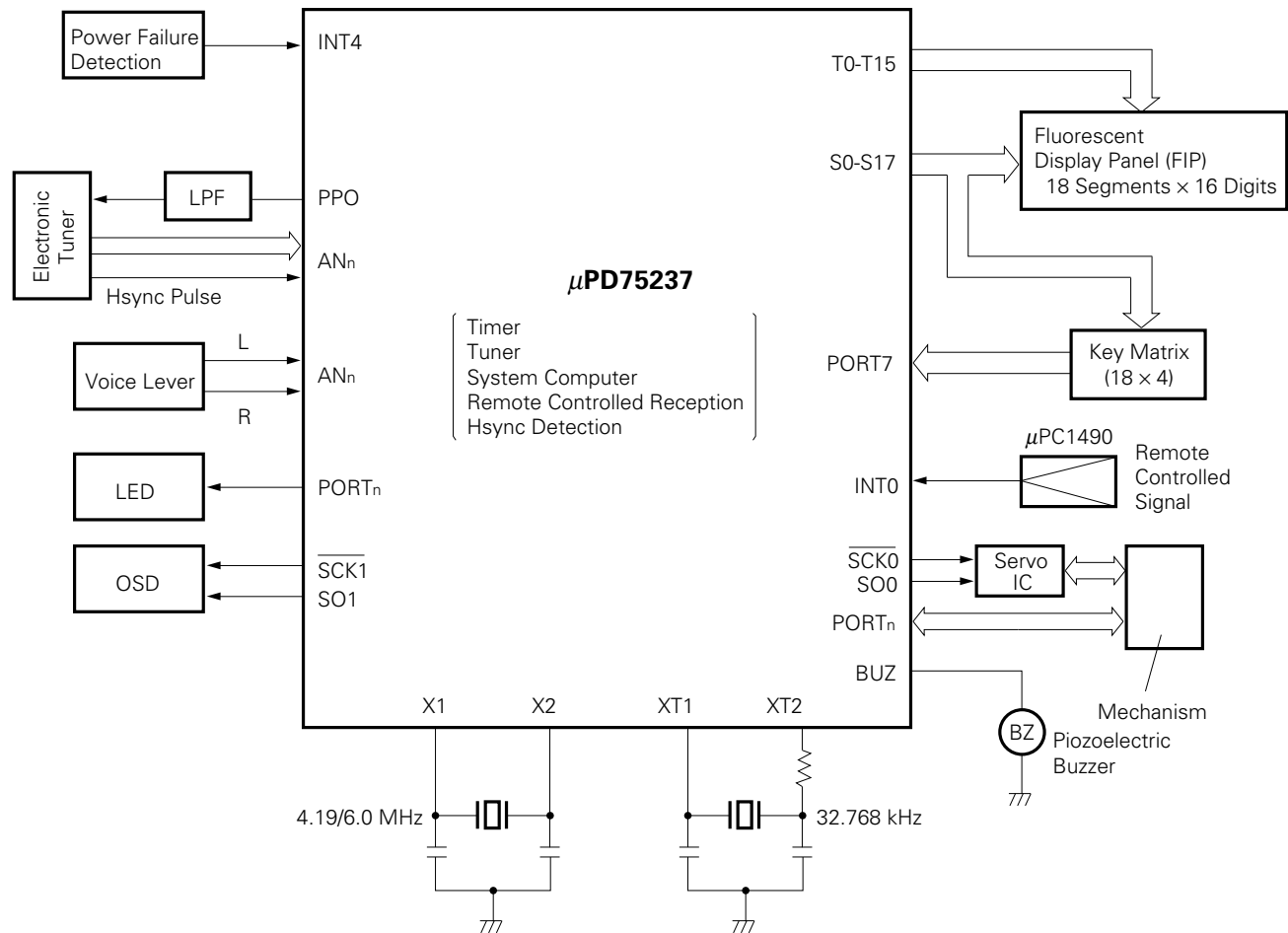
Pin	Mask Option
P40 to P43	Pull-up resistor incorporation enabled bit-wise
P50 to P53	
P70 to P73	Pull-down resistor incorporation enabled bit-wise
S0/P120 to S3/P123	Pull-down resistor incorporation to V <sub>LOAD</sub> enabled bit-wise
S4/P130 to S7/P133	
S8/P140, S9/P141	
S10/T15/P142, S11/T14/P143	
S12/T13/P150/PH0 to S15/T10/P153/PH3	
S16/P100 to S19/P103	Pull-down resistor incorporation to V <sub>LOAD</sub> or V <sub>SS</sub> bit-wise *
S20/P110 to S23/P113	
XT1, XT2	Deletion of subsystem clock oscillator feedback resistor possible

\* Select pull-down resistor incorporation to V<sub>LOAD</sub> or V<sub>SS</sub> in 8-bit units.

**Note** In a system not using subsystem clocks, power consumption in the STOP mode can be decreased by removing the feedback resistor from the oscillator.



10. APPLICATION BLOCK DIAGRAM



**Remarks** LPF : Low Pass Filter  
 OSD : On Screen Display  
 Hsync : Horizontal Synchronous

11. ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS (Ta = 25 °C)

PARAMETER	SYMBOL	TEST CONDITIONS		RATING	UNIT
Power supply voltage	V <sub>DD</sub>			-0.3 to +7.0	V
	V <sub>LOAD</sub>			V <sub>DD</sub> -40 to V <sub>DD</sub> +0.3	V
Input voltage	V <sub>I1</sub>	Except ports 4 and 5		-0.3 to V <sub>DD</sub> +0.3	V
	V <sub>I2</sub>	Ports 4 and 5	Pull-up resistor	-0.3 to V <sub>DD</sub> +0.3	V
			Open-drain	-0.3 to +11	V
Output voltage	V <sub>O</sub>	Pins except display output pins		-0.3 to V <sub>DD</sub> +0.3	V
	V <sub>OD</sub>	Display output pins		V <sub>DD</sub> -40 to V <sub>DD</sub> +0.3	V
Output current high	I <sub>OH</sub>	1 pins except display output pins		-15	mA
		S0 to S9, S16 to S23 1 pin		-15	mA
		T0 to T15 1 pin		-30	mA
		Total of pins except display output pins		-30	mA
		Total of display output pins		-120	mA
Output current low	I <sub>OL</sub>	1 pin	Peak value	30	mA
			Effective value	15	mA
		Total of ports 0, 2, 3 and 4	Peak value	100	mA
			Effective value	60	mA
		Total of ports 5 to 8	Peak value	100	mA
			Effective value	60	mA
Total loss	P <sub>T</sub>	Plastic QFP	( Ta = -40 to +70 °C )	700	mW
			( Ta = -40 to +85 °C )	510	mW
Operating temperature	T <sub>opt</sub>			-40 to +85	°C
Storage temperature	T <sub>stg</sub>			-65 to +150	°C

POWER SUPPLY VOLTAGE RANGE (Ta = -40 to +85 °C)

PARAMETER	TEST CONDITIONS	MIN.	MAX.	UNIT
CPU *1		*2	6.0	V
Display controller		4.5	6.0	V
Time/pulse generator		4.5	6.0	V
Other hardware *1		2.7	6.0	V

- \* 1. Except the system clock oscillator, display controller and timer/pulse generator.
- 2. The power supply voltage range varies, depending on the cycle time. Refer to the description of AC characteristics.

**MAIN SYSTEM CLOCK OSCILLATOR CHARACTERISTICS (Ta = -40 to +85 °C, VDD = 2.7 to 6.0 V)**

RESONATOR	RECOMMENDED CIRCUIT	PARAMETER	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Ceramic resonator		Oscillator frequency (fx) *1	VDD = Oscillation voltage range	2.0		6.2	MHz
		Oscillation stabilization time *2	After VDD reaches the minimum value in the oscillation voltage range			4	ms
Crystal resonator		Oscillator frequency (fx) *1		2.0	4.19	6.2	MHz
		Oscillation stabilization time *2	VDD = 4.5 to 6.0 V			10	ms
						30	ms
External clock		X1 input frequency (fx) *1		2.0		6.2	MHz
		X1 high and low level widths (txH, txL)		81		250	ns

- \* 1. Oscillator characteristics only. Refer to the description of AC characteristics for details of instruction execution time.
- 2. Time required for oscillation to become stabilized after VDD application or STOP mode release.

**SUBSYSTEM CLOCK OSCILLATOR CHARACTERISTICS (Ta = -40 to +85 °C, VDD = 2.7 to 6.0 V)**

RESONATOR	RECOMMENDED CIRCUIT	PARAMETER	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Crystal resonator		Oscillator frequency (fxT) *1		32	32.768	35	kHz
		Oscillation stabilization time *2	VDD = 4.5 to 6.0 V		1.0	2	s
						10	s
External clock		XT1 input frequency (fxT) *1		32		100	kHz
		XT1 high and low level widths (txTH, txTL)		5		15	μs

- \* 1. Oscillator characteristics only. Refer to the description of AC characteristics for instruction execution time.
- 2. Time required for oscillation to become stabilized after VDD application or STOP mode release.

**CAPACITANCE (Ta = 25 °C, V<sub>DD</sub> = 0 V)**

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Input capacitance	C <sub>i</sub>	f = 1 MHz Unmeasured pin returned to 0 V			15	pF
Output capacitance (except display output)	C <sub>o</sub>				15	pF
Input /output capacitance	C <sub>io</sub>				15	pF
Output capacitance ( display output )	C <sub>o</sub>				35	pF

DC CHARACTERISTICS (Ta = -40 to 85 °C, VDD = 2.7 to 6.0 V)

PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
Input voltage high	V <sub>IH1</sub>	Except below		0.7 V <sub>DD</sub>		V <sub>DD</sub>	V
	V <sub>IH2</sub>	Ports 0, 1, <u>RESET</u> , P81, P83		0.8 V <sub>DD</sub>		V <sub>DD</sub>	V
	V <sub>IH3</sub>	X1, X2, XT1		V <sub>DD</sub> -0.4		V <sub>DD</sub>	V
	V <sub>IH4</sub>	Port 7	V <sub>DD</sub> = 4.5 to 6.0 V	0.65 V <sub>DD</sub>		V <sub>DD</sub>	V
				0.7 V <sub>DD</sub>		V <sub>DD</sub>	V
V <sub>IH5</sub>	Ports 4, 5	Pull-up resistor incorporated	0.7 V <sub>DD</sub>		V <sub>DD</sub>	V	
		Open-drain	0.7 V <sub>DD</sub>		10	V	
Input Voltage low	V <sub>IL1</sub>	Except below		0		0.3 V <sub>DD</sub>	V
	V <sub>IL2</sub>	Ports 0, 1, <u>RESET</u> , P81, P83		0		0.2 V <sub>DD</sub>	V
	V <sub>IL3</sub>	X1, X2, XT1		0		0.4	V
Output voltage high	V <sub>OH</sub>	All output pins except ports 4, 5 and P03	V <sub>DD</sub> = 4.5 to 6.0V I <sub>OH</sub> = -1 mA	V <sub>DD</sub> -1.0			V
			V <sub>DD</sub> = 2.7 to 6.0V I <sub>OH</sub> = -100 μA	V <sub>DD</sub> -0.5			V
Output voltage low	V <sub>OL</sub>	Ports 3, 4, 5	V <sub>DD</sub> = 4.5 to 6.0V I <sub>OL</sub> = 15 mA		0.4	2.0	V
		All output pins	V <sub>DD</sub> = 4.5 to 6.0V I <sub>OL</sub> = 1.6 mA			0.4	V
			V <sub>DD</sub> = 2.7 to 6.0V I <sub>OL</sub> = 400 μA			0.5	V
	SB0, SB1	Open-drain pull-up resistance ≥ 1k Ω				0.2 V <sub>DD</sub>	V
Input leakage current high	I <sub>LIH1</sub>	Except below	V <sub>IN</sub> = V <sub>DD</sub>			3	μA
	I <sub>LIH2</sub>	X1, X2, XT1				20	μA
	I <sub>LIH3</sub>	Ports 4, 5	Open-drain V <sub>IN</sub> = 10 V			20	μA
Input leakage current low	I <sub>LIL1</sub>	Except below	V <sub>IN</sub> = 0 V			-3	μA
	I <sub>LIL2</sub>	X1, X2, XT1				-20	μA
Output leakage current high	I <sub>LOH1</sub>	Except below	V <sub>OUT</sub> = V <sub>DD</sub>			3	μA
	I <sub>LOH2</sub>	Ports 4, 5	(Open-drain) V <sub>OUT</sub> = 10 V			20	μA
Output leakage current low	I <sub>LOL1</sub>	Except below	V <sub>OUT</sub> = 0 V			-3	μA
	I <sub>LOL2</sub>	Display output	V <sub>OUT</sub> = V <sub>LOAD</sub> = V <sub>DD</sub> -35 V			-10	μA
Display output current	I <sub>OD</sub>	S0 to S9, S16 to S23	V <sub>DD</sub> = 4.5 to 6.0 V V <sub>OD</sub> = V <sub>DD</sub> -2 V	-3	-5.5		mA
		T0 to T15		-15	-22		mA
Built-in pull-down resistor (mask option)	R <sub>P7</sub>	Port 7 V <sub>IN</sub> = V <sub>DD</sub>	V <sub>DD</sub> = 4.5 to 6.0 V	20	80	200	kΩ
				20		1000	kΩ
	R <sub>L</sub>	Display output	V <sub>DD</sub> -V <sub>LOAD</sub> = 35 V	25	50	135	kΩ
Built-in pull-up resistor	R <sub>V1</sub>	Ports 0, 1, 2, 3, and 6 (except P00) V <sub>IN</sub> = 0 V	V <sub>DD</sub> = 5 V ± 10%	15	40	80	kΩ
			V <sub>DD</sub> = 3 V ± 10%	30		300	kΩ
	R <sub>V2</sub>	Ports 4 and 5 V <sub>OUT</sub> = V <sub>DD</sub> -2.0 V	V <sub>DD</sub> = 5 V ± 10%	15	40	70	kΩ
			V <sub>DD</sub> = 3 V ± 10%	10		60	kΩ

**DC CHARACTERISTICS (Ta = -40 to 85 °C, VDD = 2.7 to 6.0 V)**

PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT		
Supply current *1	IDD1	6 MHz crystal oscillation C1 = C2 = 22 pF *4	Operating mode	VDD = 5 V ± 10% *2		4.5	13.5	mA	
				VDD = 3 V ± 10% *3		0.6	1.8	mA	
	IDD2		HALT mode	VDD = 5 V ± 10%		600	1800	μA	
				VDD = 3 V ± 10%		200	600	μA	
	IDD1		4.19 MHz crystal oscillation C1 = C2 = 22 pF *4	Operating mode	VDD = 5 V ± 10% *2		3	9	mA
					VDD = 3 V ± 10% *3		0.5	1.5	mA
	IDD2	HALT mode		VDD = 5 V ± 10%		600	1800	μA	
				VDD = 3 V ± 10%		200	600	μA	
	IDD3	32 kHz crystal oscillation *5		Operating mode	VDD = 3 V ± 10%		40	120	μA
	IDD4			HALT mode	VDD = 3 V ± 10%		5	15	μA
	IDD5	XT1 = 0 V STOP mode	VDD = 5 V ± 10%			0.5	20	μA	
			VDD = 3 V ± 10%		0.3	10	μA		
Ta = 25 °C						5	μA		

- \* 1. Current flowing to the built-in pull-down (pull-up) resistor excluded.
- 2. When operated in the high speed mode with the processor clock control register (PCC) set to 0011.
- 3. When operated in the low speed mode with PCC = 0000.
- 4. Subsystem clock oscillation included.
- 5. When operated with subsystem clock with system clock control register (SCC) set to 1001 and the main system clock stopped.

**A/D CONVERTER CHARACTERISTICS (Ta = -40 to +85 °C, VDD = 2.7 to 6.0 V, AVSS = VSS = 0 V, AVDD = VDD)**

PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
Resolution				8	8	8	bit
Absolute accuracy *1		2.5 V ≤ AVREF ≤ AVDD	-10 ≤ Ta ≤ +85 °C			±1.5	LSB
			-40 ≤ Ta < -10 °C			±2.0	
Conversion time	tCONV	*2				168/fx	μs
Sampling time	tsAMP	*3				44/fx	μs
Analog input voltage	VIAN			AVSS		AVREF	V
Analog input impedance	RAN				1000		MΩ
AVREF current	IAREF				1.0	2.0	mA

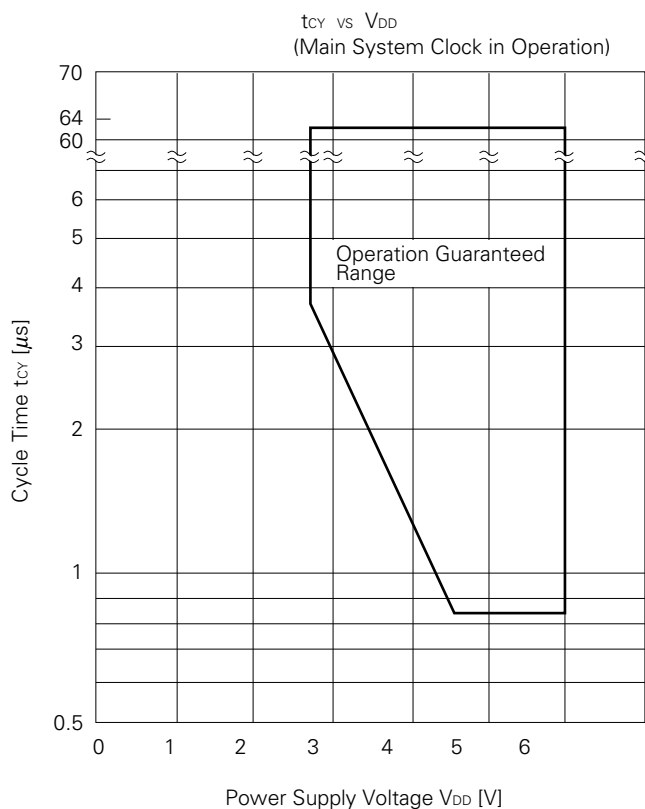
- \* 1. Absolute accuracy with any quantization error (±1/2 LSB) excluded.
- 2. Time until EOC = 1 after execution of conversion start instruction (fx = 28.0 μs at 6.0 MHz, fx = 40.1 μs at 4.19 MHz).
- 3. Time until the end of sampling after execution of conversion start instruction (fx = 7.33 μs at 6.0 MHz, fx = 10.5 μs at 4.19 MHz).

AC CHARACTERISTICS (Ta = -40 to +85 °C , VDD = 2.7 to 6.0 V)

(1) Basic Operation

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT	
CPU clock cycle time (minimum instruction execution time = 1 machine cycle) *1	tcy	Operation with main system clock	VDD = 4.5 to 6.0 V	0.67		64	μs
				2.6		64	μs
		Operation with subsystem clock	114	122	125	μs	
TIO input frequency	fTI	VDD = 4.5 to 6.0 V	0		1	MHZ	
			0		275	kHz	
TIO input high and low- level widths	tTIH, tTIL	VDD = 4.5 to 6.0 V	0.48			μs	
			1.8			μs	
Interrupt input high and low-level widths	tINTH, tINTL	INT0	*2			μs	
		INT1, 2, 4	10			μs	
RESET low-level width	trSL		10			μs	

- \* 1. CPU clock (Φ) cycle time is determined by the oscillator frequency of the connected resonator, the system clock control register (SCC) and the processor clock control register (PCC). The cycle time tcy characteristics for power supply voltage VDD when the main system clock is in operation is shown below.
- 2. 2tcy or 128/fx is set by interrupt mode register (IM0) setting.



(2) Serial Transfer Operation

(a) 2-wire and 3-wire serial I/O mode ( $\overline{\text{SCK}}$ ...Internal clock output)

PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
$\overline{\text{SCK}}$ cycle time	$t_{\text{KCY1}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$	$f_{\text{X}} = 6.0 \text{ MHz}$	1340			ns
			$f_{\text{X}} = 4.19 \text{ MHz}$	1600			ns
			$f_{\text{X}} = 6.0 \text{ MHz}$	2680			ns
			$f_{\text{X}} = 4.19 \text{ MHz}$	3800			ns
$\overline{\text{SCK}}$ high and low level widths	$t_{\text{KL1}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$		$(t_{\text{KCY1}}/2)-50$			ns
	$t_{\text{KH1}}$			$(t_{\text{KCY1}}/2)-150$			ns
SI setup time (to $\overline{\text{SCK}}\uparrow$ )	$t_{\text{SIK1}}$			150			ns
SI hold time (from $\overline{\text{SCK}}\uparrow$ )	$t_{\text{KSI1}}$			400			ns
SO output delay time from $\overline{\text{SCK}}\downarrow$	$t_{\text{KSO1}}$	$R_{\text{L}} = 1 \text{ k } \Omega$ $C_{\text{L}} = 100 \text{ pF}^*$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$			250	ns
						1000	ns

\*  $R_{\text{L}}$  and  $C_{\text{L}}$  are SO output line load resistance and load capacitance, respectively.

(b) 2-wire and 3-wire serial I/O mode ( $\overline{\text{SCK}}$ ...External clock input)

PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
$\overline{\text{SCK}}$ cycle time	$t_{\text{KCY2}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$		800			ns
				3200			ns
$\overline{\text{SCK}}$ high and low level widths	$t_{\text{KL2}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$		400			ns
	$t_{\text{KH2}}$			1600			ns
SI setup time (to $\overline{\text{SCK}}\uparrow$ )	$t_{\text{SIK2}}$			100			ns
SI hold time (from $\overline{\text{SCK}}\uparrow$ )	$t_{\text{KSI2}}$			400			ns
SO output delay time from $\overline{\text{SCK}}\downarrow$	$t_{\text{KSO2}}$	$R_{\text{L}} = 1 \text{ k } \Omega$ $C_{\text{L}} = 100 \text{ pF}^*$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$			300	ns
						1000	ns

\*  $R_{\text{L}}$  and  $C_{\text{L}}$  are SO output line load resistance and load capacitance, respectively.



(c) SBI mode ( $\overline{\text{SCK}}$ ...Internal clock output (master))

PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
$\overline{\text{SCK}}$ cycle time	$t_{\text{KCY3}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$	$f_x = 6.0 \text{ MHz}$	1340			ns
			$f_x = 4.19 \text{ MHz}$	1600			ns
			$f_x = 6.0 \text{ MHz}$	2680			ns
			$f_x = 4.19 \text{ MHz}$	3800			ns
$\overline{\text{SCK}}$ high and low level widths	$t_{\text{KL3}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$		$t_{\text{KCY}}/2-50$			ns
	$t_{\text{KH3}}$			$t_{\text{KCY}}/2-150$			ns
SB0 and SB1 setup time (to $\overline{\text{SCK}}\uparrow$ )	$t_{\text{SIK3}}$			150			ns
SB0 and SB1 hold time (from $\overline{\text{SCK}}\uparrow$ )	$t_{\text{KSI3}}$			$t_{\text{KCY}}/2$			ns
SB0 and SB1 output delay time from $\overline{\text{SCK}}\downarrow$	$t_{\text{KSO3}}$	$R_L = 1 \text{ k } \Omega$ $C_L = 100 \text{ pF}^*$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$	0		250	ns
				0		1000	ns
SB0, SB1 $\downarrow$ from $\overline{\text{SCK}}\uparrow$	$t_{\text{KSB}}$			$t_{\text{KCY}}$			ns
$\overline{\text{SCK}}$ from SB0, SB1 $\downarrow$	$t_{\text{SBK}}$			$t_{\text{KCY}}$			ns
SB0 and SB1 low-level widths	$t_{\text{SBL}}$			$t_{\text{KCY}}$			ns
SB0 and SB1 high-level widths	$t_{\text{SBH}}$			$t_{\text{KCY}}$			ns

\*  $R_L$  and  $C_L$  are SO output line load resistance and load capacitance, respectively.

(d) SBI mode ( $\overline{\text{SCK}}$ ...External clock input (slave))

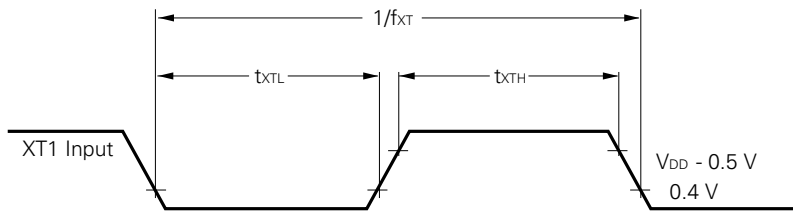
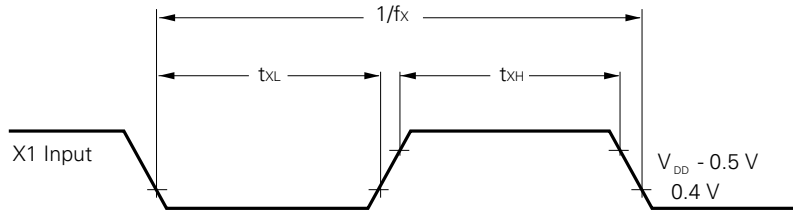
PARAMETER	SYMBOL	TEST CONDITIONS		MIN.	TYP.	MAX.	UNIT
$\overline{\text{SCK}}$ cycle time	$t_{\text{KCY4}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$		800			ns
				3200			ns
$\overline{\text{SCK}}$ high and low level widths	$t_{\text{KL4}}$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$		400			ns
	$t_{\text{KH4}}$			1600			ns
SB0 and SB1 setup time (to $\overline{\text{SCK}}\uparrow$ )	$t_{\text{SIK4}}$			100			ns
SB0 and SB1 hold time (from $\overline{\text{SCK}}\uparrow$ )	$t_{\text{KSI4}}$			$t_{\text{KCY}}/2$			ns
SB0 and SB1 output delay time from $\overline{\text{SCK}}\downarrow$	$t_{\text{KSO4}}$	$R_L = 1 \text{ k } \Omega$ $C_L = 100 \text{ pF}^*$	$V_{\text{DD}} = 4.5 \text{ to } 6.0 \text{ V}$	0		300	ns
				0		1000	ns
SB0, SB1 $\downarrow$ from $\overline{\text{SCK}}\uparrow$	$t_{\text{KSB}}$			$t_{\text{KCY}}$			ns
$\overline{\text{SCK}}\downarrow$ from SB0, SB1 $\downarrow$	$t_{\text{SBK}}$			$t_{\text{KCY}}$			ns
SB0 and SB1 low-level widths	$t_{\text{SBL}}$			$t_{\text{KCY}}$			ns
SB0 and SB1 high-level widths	$t_{\text{SBH}}$			$t_{\text{KCY}}$			ns

\*  $R_L$  and  $C_L$  are SO output line load resistance and load capacitance, respectively.

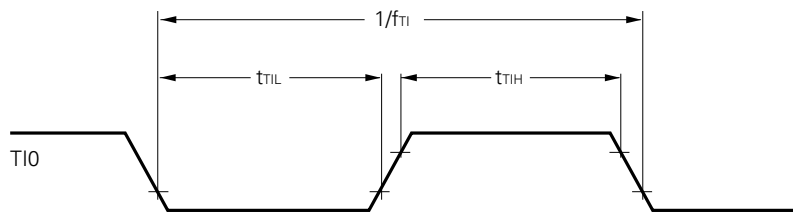
**AC Timing Test Points (Except X1 and XT1 Inputs)**



**Clock Timing**

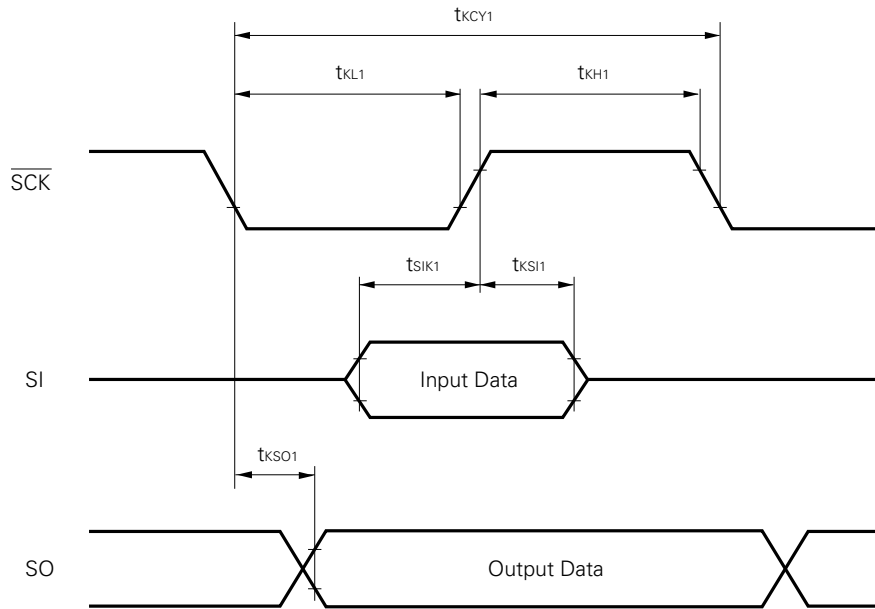


**T10 Timing**

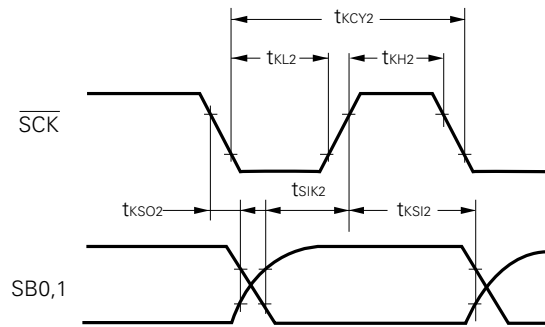


Serial Transfer Timing

3-wire serial I/O mode:

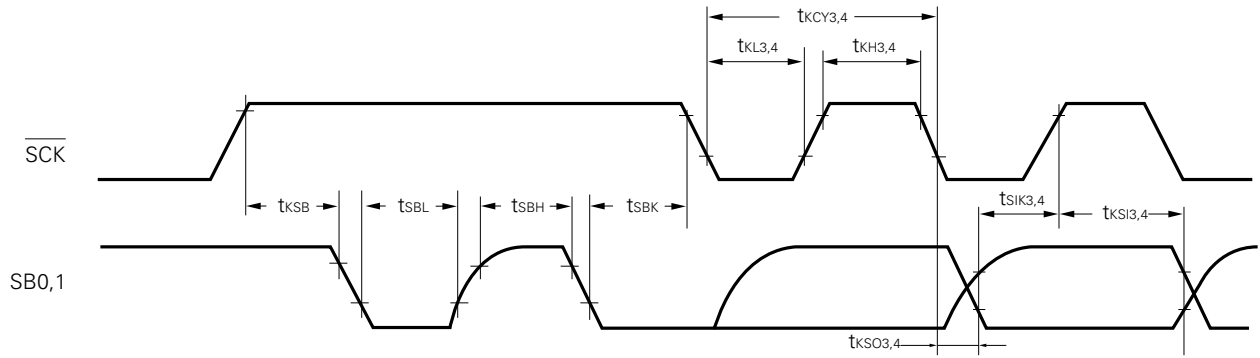


2-wire serial I/O mode:

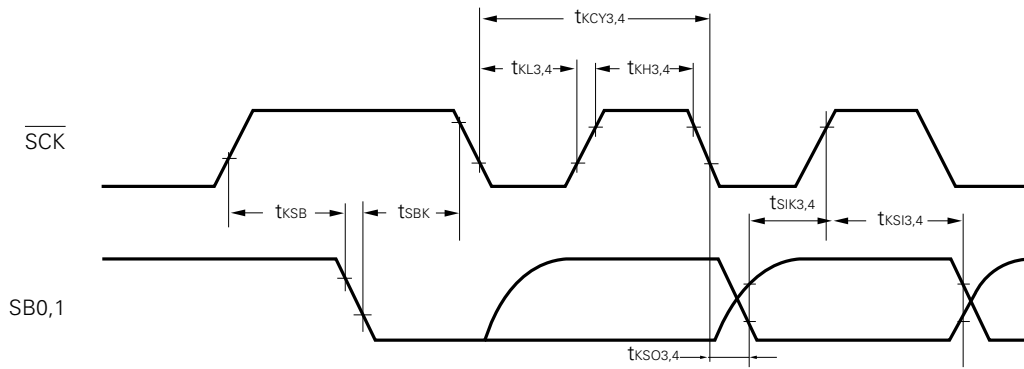


**Serial Transfer Timing**

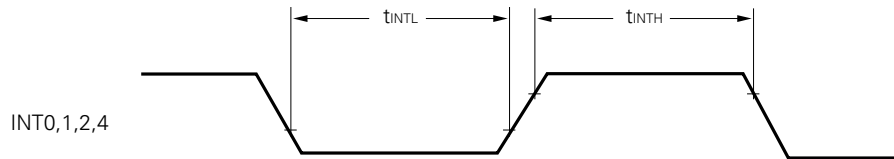
**Bus release signal transfer:**



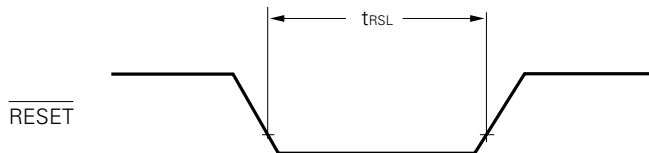
**Command signal transfer:**



**Interrupt Input Timing**



**RESET Input Timing**



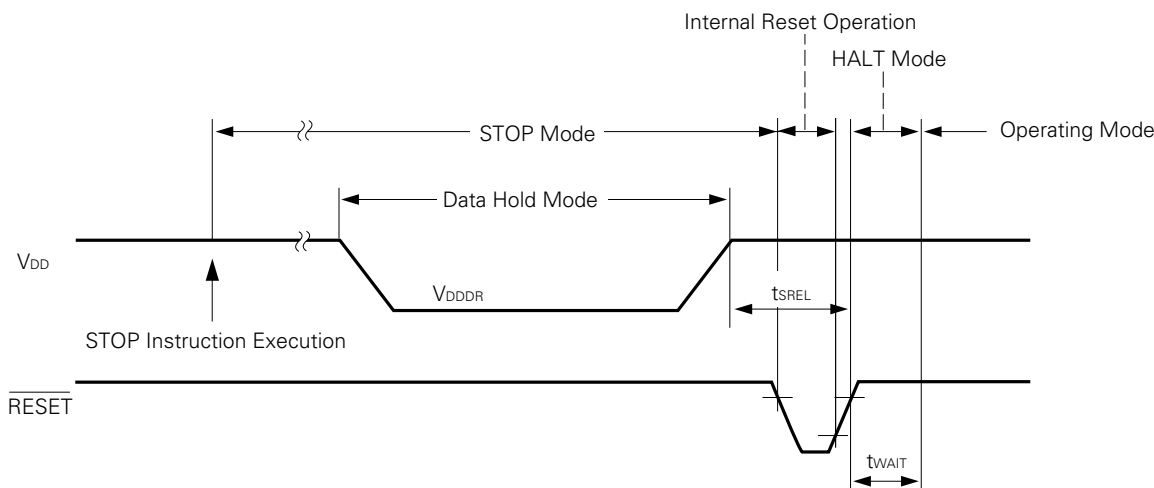
**DATA MEMORY STOP MODE LOW POWER SUPPLY VOLTAGE DATA HOLD CHARACTERISTICS (Ta = -40 to +85 °C)**

PARAMETER	SYMBOL	TEST CONDITIONS	MIN.	TYP.	MAX.	UNIT
Data hold power supply voltage	V <sub>DDDR</sub>		2.0		6.0	V
Data hold power supply current *1	I <sub>DDDR</sub>	V <sub>DDDR</sub> = 2.0 V		0.1	10	μA
Release signal set time	t <sub>SREL</sub>		0			μs
Oscillation stabilization wait time *2	t <sub>WAIT</sub>	Release by $\overline{\text{RESET}}$		2 <sup>17</sup> /f <sub>x</sub>		ms
		Release by interrupt request		*3		ms

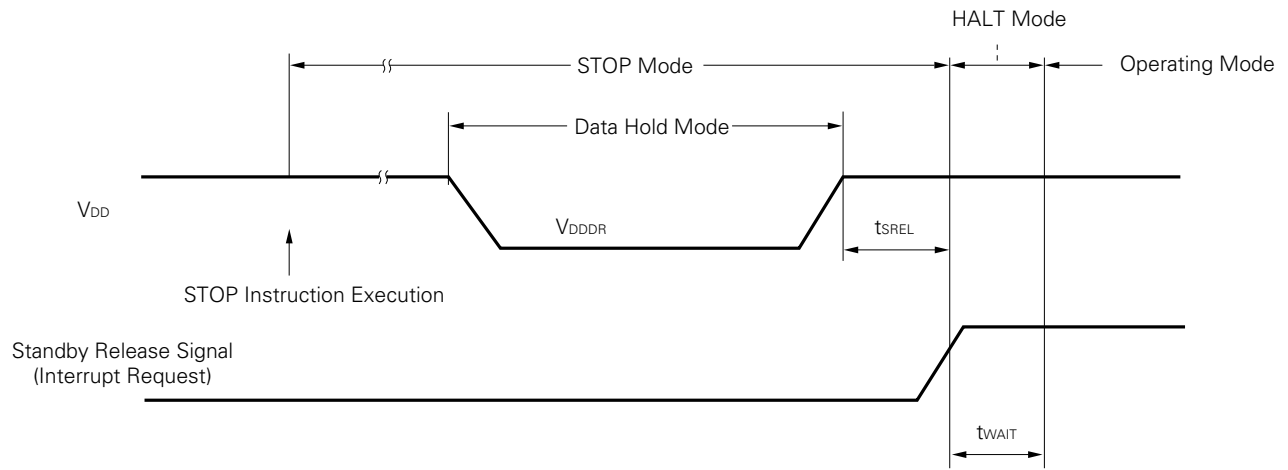
- \* 1. Current to the on-chip pull-up (pull-down) resistor is not included.
- 2. Oscillation stabilization wait time is time to stop CPU operation to prevent unstable operation upon oscillation start.
- 3. According to the setting of the basic interval timer mode register (BTM) (see below).

BTM3	BTM2	BTM1	BTM0	Wait Time	
				(Values at f <sub>x</sub> = 6.0 MHz in parentheses)	(Values at f <sub>x</sub> = 4.19 MHz in parentheses)
—	0	0	0	2 <sup>20</sup> /f <sub>x</sub> (approx. 175 ms)	2 <sup>20</sup> /f <sub>x</sub> (approx. 250 ms)
—	0	1	1	2 <sup>17</sup> /f <sub>x</sub> (approx. 21.8 ms)	2 <sup>17</sup> /f <sub>x</sub> (approx. 31.3 ms)
—	1	0	1	2 <sup>15</sup> /f <sub>x</sub> (approx. 5.46 ms)	2 <sup>15</sup> /f <sub>x</sub> (approx. 7.82 ms)
—	1	1	1	2 <sup>13</sup> /f <sub>x</sub> (approx. 1.37 ms)	2 <sup>13</sup> /f <sub>x</sub> (approx. 1.95 ms)

**Data Hold Timing (STOP Mode Release by  $\overline{\text{RESET}}$ )**



**Data Hold Timing (Standby Release Signal: STOP Mode Release by Interrupt Signal)**

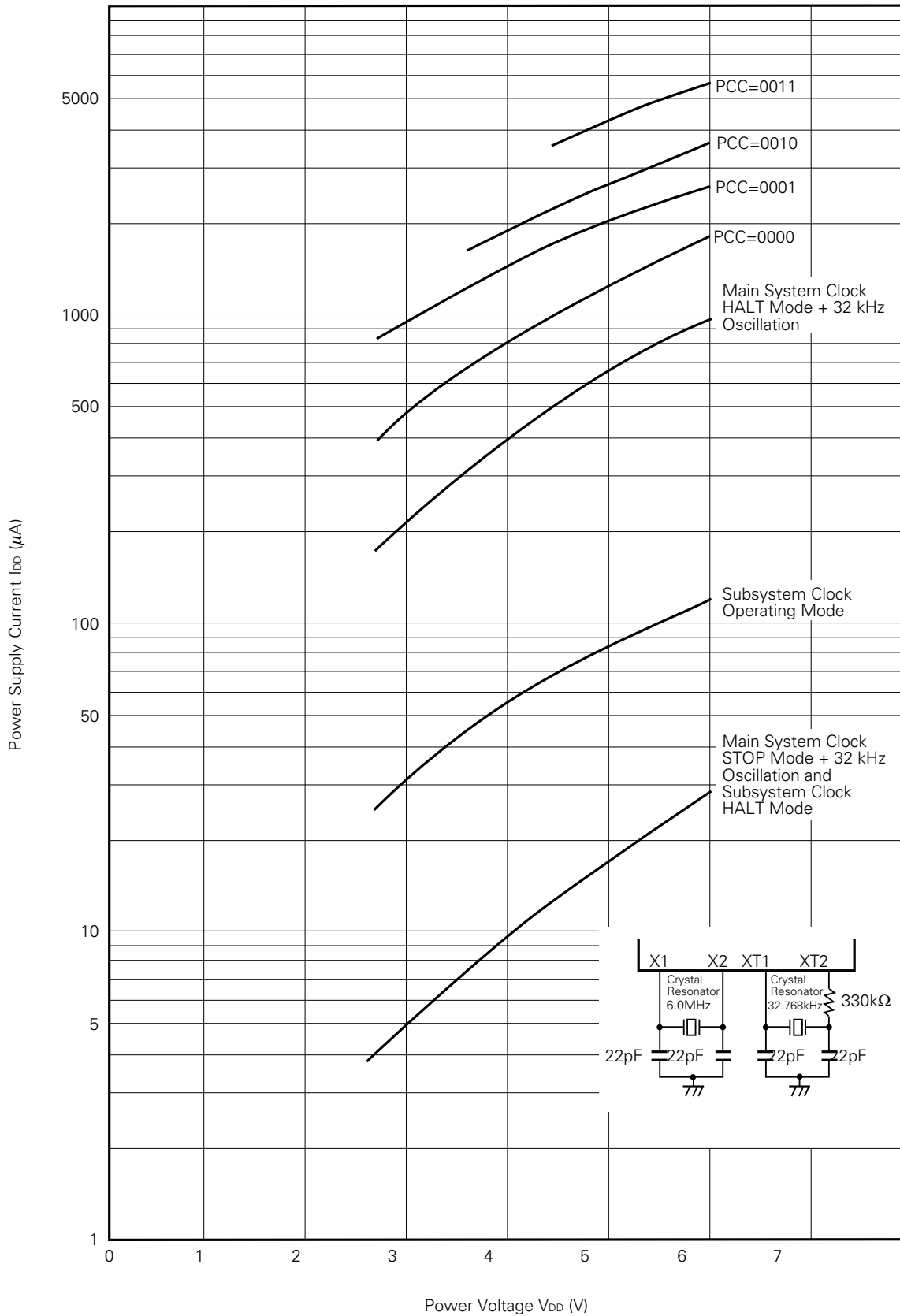


12. CHARACTERISTIC CURVES (REFERENCE VALUES)



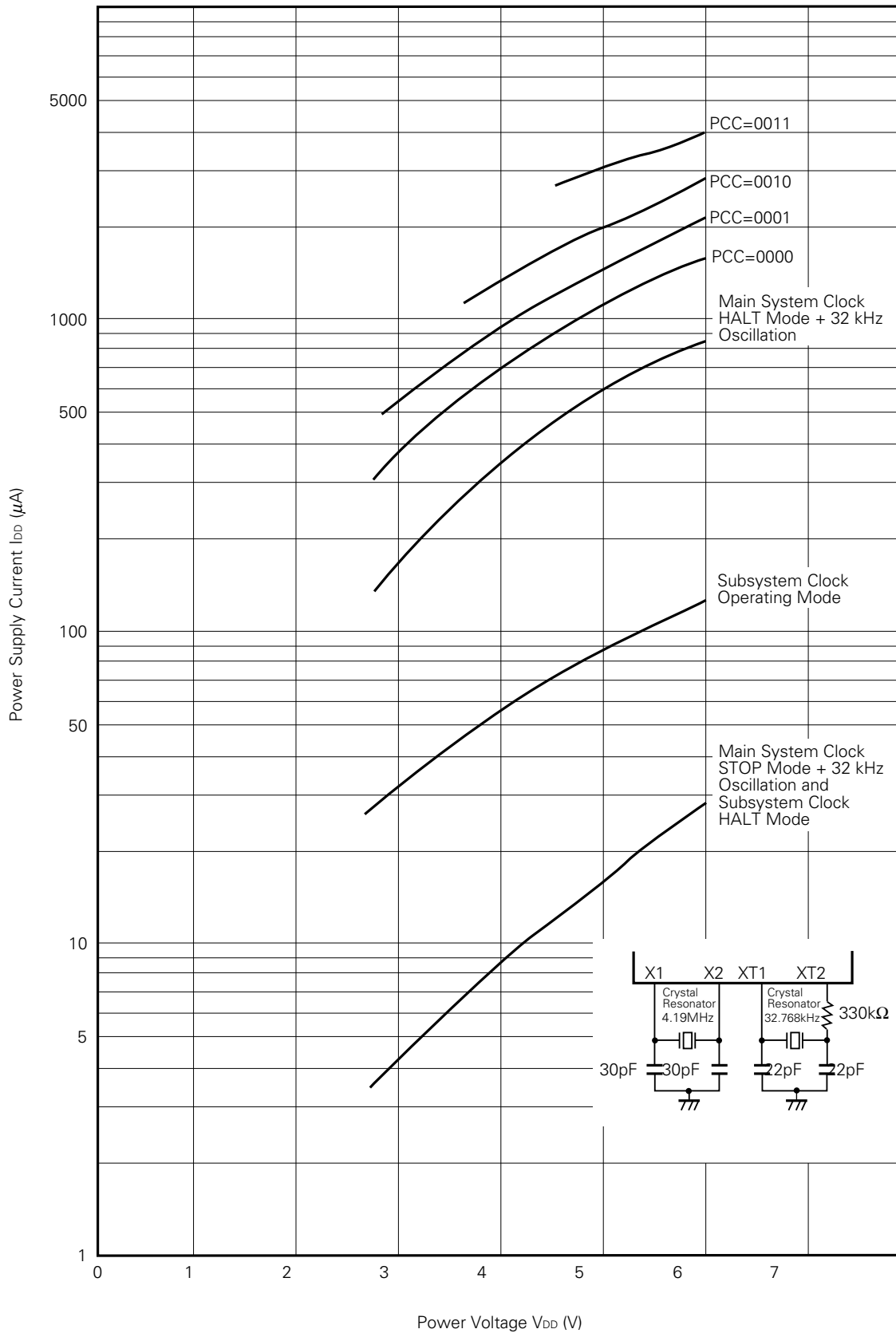
I<sub>DD</sub> vs V<sub>DD</sub> (Main System Clock : 6.0 MHz)

(T<sub>a</sub>=25°C)



I<sub>DD</sub> vs V<sub>DD</sub> (Main System Clock : 4.19 MHz)

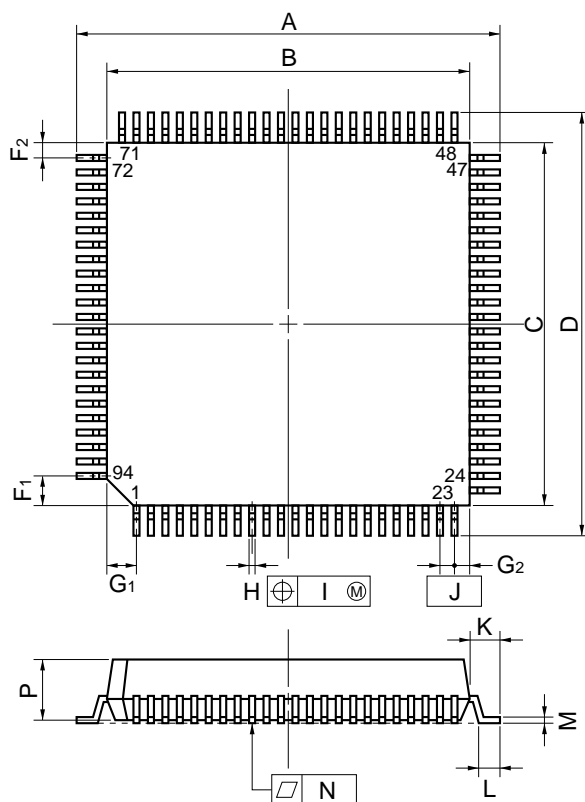
(T<sub>a</sub>=25°C)



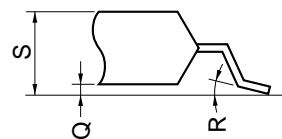


13. PACKAGE INFORMATION

94 PIN PLASTIC QFP (□ 20)



detail of lead end



NOTE

Each lead centerline is located within 0.15 mm (0.006 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
A	23.2±0.4	0.913 <sup>+0.017</sup> <sub>-0.016</sub>
B	20.0±0.2	0.787 <sup>+0.009</sup> <sub>-0.008</sub>
C	20.0±0.2	0.787 <sup>+0.009</sup> <sub>-0.008</sub>
D	23.2±0.4	0.913 <sup>+0.017</sup> <sub>-0.016</sub>
F1	1.6	0.063
F2	0.8	0.031
G1	1.6	0.063
G2	0.8	0.031
H	0.35±0.10	0.014 <sup>+0.004</sup> <sub>-0.005</sub>
I	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
K	1.6±0.2	0.063±0.008
L	0.8±0.2	0.031 <sup>+0.009</sup> <sub>-0.008</sub>
M	0.15 <sup>+0.10</sup> <sub>-0.05</sub>	0.006 <sup>+0.004</sup> <sub>-0.003</sub>
N	0.10	0.004
P	3.7	0.146
Q	0.1±0.1	0.004±0.004
R	5°±5°	5°±5°
S	4.0 MAX.	0.158 MAX.

S94GJ-80-5BG-3

★ 14. RECOMMENDED SOLDERING CONDITIONS

The μPD75237 should be soldered and mounted under the conditions recommended in the table below. For soldering methods and conditions other than those recommended below, contact our salesman.

**Table 14-1 List of Recommended Soldering Conditions**

Product Name	Package	Recommended Condition Symbol
μPD75237GJ-xxx-5BG	94-pin plastic QFP	WS60-107-1 IR30-107-1 VP15-107-1 Pin part heating

**Table 14-2 Soldering Conditions**

Recommended Condition Symbol	Soldering Method	Soldering Conditions
WS60-107-1	Wave Soldering	Solder bath temperature: 260 °C or less Duration: 10 sec. max. Number of times: Once Time limit: 7 days* (thereafter 10 hours prebaking required at 125 °C) Preheating temperature: 120 °C max. (package surface temperature)
IR30-107-1	Infrared reflow	Package peak temperature: 230 °C Duration: 30 sec. max. (at 210 °C or above) Number of times: Once Time limit: 7 days* (thereafter 10 hours prebaking required at 125 °C)
VP15-107-1	VPS	Package peak temperature: 215 °C Duration: 40 sec. max. (at 200 °C or above) Number of times: Once Time limit: 7 days* (thereafter 10 hours prebaking required at 125 °C)
Pin part heating	Pin part heating	Pin part temperature: 300 °C or less Duration: 3 sec. max. (Per device side)

\* For the storage period after dry-pack decompression, storage conditions are max. 25°C, 65% RH.

**Note** Use of more than one soldering method should be avoided (except in the case of pin part heating).

**Remarks** For details of recommended soldering conditions for the surface mounting type, refer to the document "Semiconductor Device Mount Technology" (IEI-1207).

APPENDIX A. LIST OF μPD75238 SERIES PRODUCT FUNCTIONS

Item \ Product Name		μPD75217	μPD75236	μPD75237	μPD75238	μPD75P238
ROM		24448 × 8	16256 × 8	24448 × 8	32640 × 8	
RAM		768 × 4		1024 × 4		
Instruction cycle	Main system clock selected	0.95 μs/1.91 μs/ 15.3 μs (Operation at 4.19 MHz)	0.95 μs/1.91 μs/ 3.82 μs/15.3 μs (Operation at 4.19 MHz)	0.67 μs/1.33 μs/2.67 μs/10.7 μs (Operation at 6.0 MHz)		
	Subsystem clock selected	122 μs (Operation at 32.768 kHz)				
I/O line (FIP dual-function pin included and FIP dedicated pin excluded)	Total	33	64			
	Input	8	16			
	Input/output	20: 8 for LED drive	24: 12 for LED drive			
	Ouptut	5	24			
A/D converter		None	8: 8-bit resolution			
FIP controller/ driver	High-voltage output	26: 40 V max.	34: 40 V max.			
	No. of segments	9 to 16 segments	9 to 24 segments			
	No. of digits	9 to 16 digits				
Timer		4 channels	5 channels			
Serial interface		1 channel 3-wire	2 channels { SBI/3-wire 3-wire			
Interrupt source		10	11			
Operating temperature range		-40 to +85 °C				-40 to +70 °C
Operating voltage		2.7 to 6.0 V				
Package		64-pin plastic shrink DIP 64-pin plastic QFP	94-pin plastic QFP			94-pin plastic QFP 94-pin ceramic LCC with window

★

★ APPENDIX B. DEVELOPMENT TOOLS

The following development tools are available for the development of systems using the μPD75237.

**Language Processor**

	Host Machine	OS	Supply Medium	Ordering Code (Product Name)
		RA75X relocatable assembler	PC-9800 series	
			5-inch 2HD	μS5A10RA75X
	IBM PC series	PC DOS™ (Ver.3.1)	5-inch 2HC	μS7B10RA75X

**PROM Write Tools**

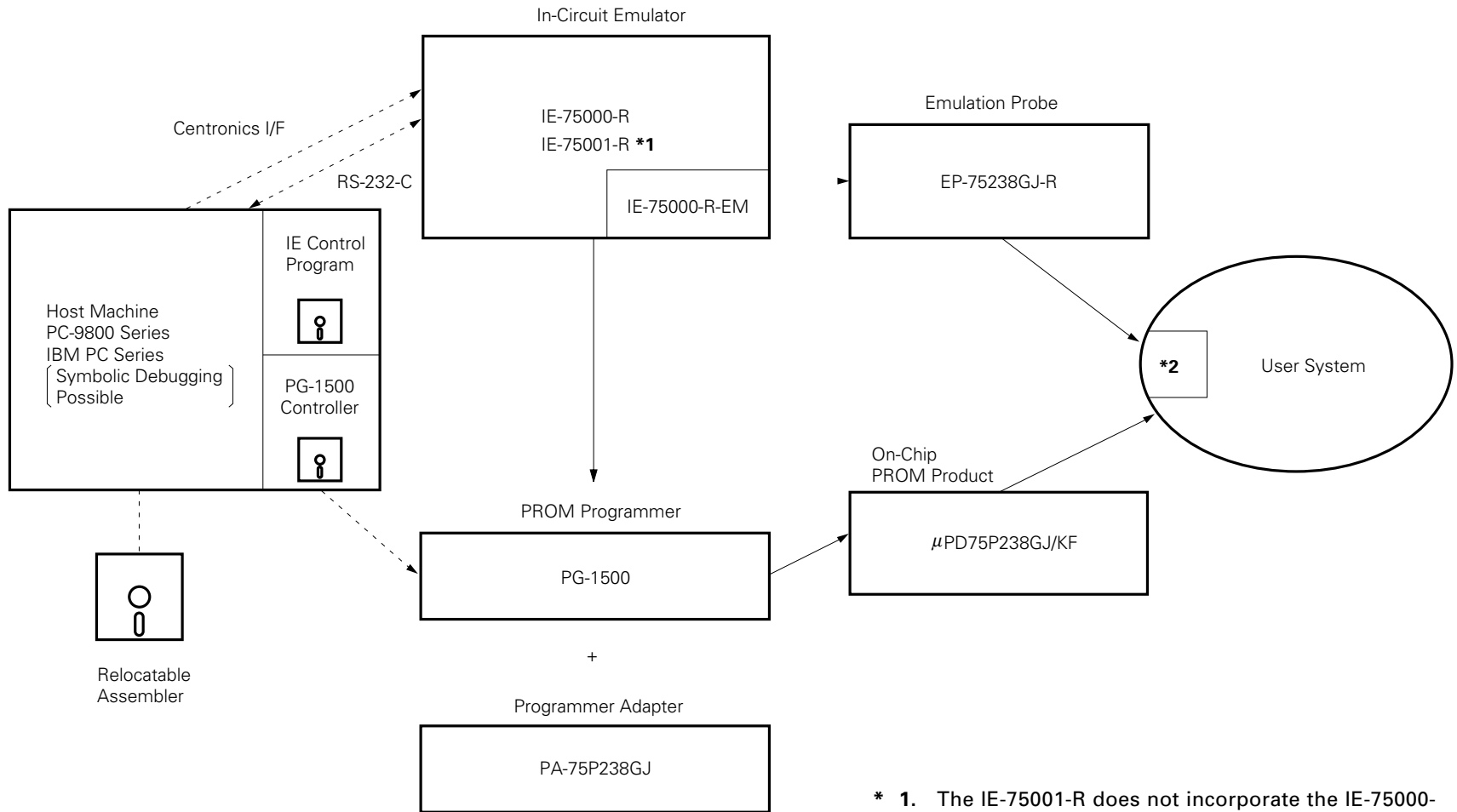
Hardware	PG-1500	PROM programmer which can easily program representative 256K-bit to 1M-bit PROMs and single-chip microcomputers with on-chip PROM from the keyboard or by remote control by connecting a board provided and a separately sold socket board.			
	PA-75P238GJ	PROM programmer adapter for μPD75P238 used in connection with PG-1500.			
Software	PG-1500 controller	PG-1500 is connected to the host machine via serial and parallel interfaces to control the PG-1500 on the host machine.			
		Host Machine	OS	Supply Medium	Ordering Code (Product Name)
		PC-9800 series	MS-DOS (Ver.3.10 to Ver.3.30C)	3.5-inch 2HD	
				5-inch 2HD	μS5A10PG1500
IBM PC series	PC DOS (Ver.3.1)	5-inch 2HC	μS7B10PG1500		

Debugging Tools

Hardware	IE-75000-R *	The IE-75000-R is an in-circuit emulator corresponding to the 75X series. Use the IE-75000-R and emulation probe in combinations for the development of μPD75237. Debugging can be carried out efficiently by connecting the IE-75000-R to the host machine and the PROM programmer.			
	IE-75000-R-EM	The IE-75000-R-EM is an emulation board for the IE-75000-R and IE-75001-R. It is incorporated in the IE-75000-R. Use the IE-75000-R-EM and IE-75000-R or IE-75001-R in combinations for the evaluation of μPD75237.			
	IE-75001-R	The IE-75001-R is an in-circuit emulator corresponding to the 75X series. Use the IE-75001-R and emulation board IE-75000-R-EM which is sold separately, and emulation probe in combinations for the development of μPD75237. Debugging can be carried out efficiently by connecting the IE-75001-R to the host machine and the PROM programmer.			
	EP-75238GJ-R IE-9200G-94	Emulation probe for μPD75237 (94-pin plastic QFP). Used in combination with the IE-75000-R or IE-75001-R. 94-pin conversion socket EV-9200G-94 is also provided to facilitate connection with the user system.			
Software	IE control program	Controls the IE-75000-R and IE-75001-R on the host machine with the IE-75000-R and IE-75001-R, connected to the host machine via RS-232-C.			
		Host Machine	OS	Supply Medium	Ordering Code (Product Name)
		PC-9800 series	MS-DOS (Ver.3.10 to Ver.3.30C)	3.5-inch 2HD	μS5A13IE75X
				5-inch 2HD	μS5A10IE75X
IBM PC series	PC DOS (Ver.3.1)	5-inch 2HC	μS7B10IE75X		

\* Maintenance product

### Development Tool Configuration



- \* 1. The IE-75001-R does not incorporate the IE-75000-R-EM (sold separately).
- \* 2. EV-9200G-94

[MEMO]

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

The devices listed in this document are not suitable for use in aerospace equipment, submarine cables, nuclear reactor control systems and life support systems. If customers intend to use NEC devices for above applications or they intend to use "Standard" quality grade NEC devices for applications not intended by NEC, please contact our sales people in advance.

Application examples recommended by NEC Corporation

Standard : Computer, Office equipment, Communication equipment, Test and Measurement equipment, Machine tools, Industrial robots, Audio and Visual equipment, Other consumer products, etc.

Special : Automotive and Transportation equipment, Traffic control systems, Antidisaster systems, Anticrime systems, etc.

FIP® is a trademark of NEC Corporation.

MS-DOS™ is a trademark of MicroSoft Corporation.

PC DOS™ is a trademark of IBM Corporation.