

**4-BIT SINGLE-CHIP MICROCONTROLLER WITH  
DIGITAL TUNING SYSTEM HARDWARE**

The  $\mu$ PD17012 is a 4-bit single-chip CMOS microcontroller equipped with hardware for digital tuning systems.

The CPU employs 17K architecture and can directly manipulate the data memory, execute various operations, and control the peripheral hardware with a single instruction. All the instructions are one-word 16-bit instructions.

Besides various I/O ports, an LCD controller/driver, A/D converter, D/A converter (PWM output), and BEEP output, this microcontroller has a prescaler that can operate at up to 250 MHz, a PLL frequency synthesizer, and frequency counter for digital tuning systems.

Consequently, a high-performance, multi-functional digital tuning system can be configured on a single chip.

In addition to the  $\mu$ PD17012, a one-time PROM version for program evaluation and small-scale production, the  $\mu$ PD17P012, is also available.

**FEATURES**

- 17K architecture: General-purpose register method
- Program memory (ROM)  
8 KB ( $4,096 \times 16$  bits)
- General-purpose data memory (RAM)  
 $316 \times 4$  bits
- Instruction execution time  
4.44  $\mu$ s (with 4.5 MHz crystal resonator)
- Decimal operation
- Table reference
- Hardware for PLL frequency synthesizer  
Dual modulus prescaler (250 MHz max.), programmable divider, phase comparator, charge pump
- Various peripheral hardware  
General-purpose I/O ports, LCD controller/driver, serial interface, A/D converter, D/A converter (PWM output), BEEP output, frequency counter
- Interrupts  
External: 1  
Internal: 3
- Power-on-reset, reset by CE pin, and power failure detector
- Power-saving CMOS
- Supply voltage:  $V_{DD} = 5 \text{ V} \pm 10\%$

The information in this document is subject to change without notice. Before using this document, please confirm that this is the latest version.

Not all devices/types available in every country. Please check with local NEC representative for availability and additional information.

## ORDERING INFORMATION

	Part Number	Package
	$\mu$ PD17012GF-xxx-3BE	64-pin plastic QFP (14 × 20 mm)
★	$\mu$ PD17012GC-xxx-8BT	80-pin plastic QFP (14 × 14 mm)

**Remark** xxx indicates the ROM code suffix.

## OVERVIEW OF FUNCTIONS

Item	Function
Program memory (ROM)	<ul style="list-style-type: none"> <li>8 KB (4,096 × 16 bits)</li> <li>Table reference area: 4,096 × 16 bits</li> </ul>
General-purpose data memory (RAM)	<ul style="list-style-type: none"> <li>316 × 4 bits</li> </ul>
Register file	33 × 4 bits (control register)
Port register	12 × 4 bits (functions alternately as LCD segment register)
Instruction execution time	<ul style="list-style-type: none"> <li>4.44 <math>\mu</math>s (with 4.5 MHz crystal resonator)</li> </ul>
Stack levels	<ul style="list-style-type: none"> <li>Address stack: 5 levels (stack operation enabled)</li> <li>Interrupt stack: 2 levels (stack operation disabled)</li> </ul>
General-purpose ports	<ul style="list-style-type: none"> <li>I/O ports: 14 pins</li> <li>Input ports: 8 pins</li> <li>Output ports: 8 pins (+20: LCD segment pin)</li> </ul>
BEEP output	<ul style="list-style-type: none"> <li>2 pins (frequency can be set individually)</li> <li>Selectable frequency (200 Hz, 1 kHz, 3 kHz, 9 kHz)</li> </ul>
LCD controller/driver	<ul style="list-style-type: none"> <li>20 segments, 3 commons</li> <li>1/3 duty, 1/2 bias, frame frequency: 167 Hz, drive voltage: <math>V_{DD}</math>, segment pins also used as key source pins: 16</li> <li>All 20 pins can be used as output port pins</li> <li>(4 pins can be set in output mode individually and the rest are set at once)</li> </ul>
Serial interface	<ul style="list-style-type: none"> <li>1 channel</li> <li>3-wire (serial I/O)</li> </ul>
D/A converter	<ul style="list-style-type: none"> <li>8 bits × 2 channels (PWM output)</li> </ul>
A/D converter	<ul style="list-style-type: none"> <li>6 bits × 2 channels (successive approximation method by software)</li> </ul>
Interrupts	<ul style="list-style-type: none"> <li>4 (maskable)</li> <li>External: 1 (INT pin)</li> <li>Internal: 3 (timer × 2, serial interface)</li> </ul>
Timer	<ul style="list-style-type: none"> <li>3 channels</li> <li>12-bit timer (10, 50 <math>\mu</math>s)</li> <li>Basic timer 0 (1, 5, 100, 250 ms)</li> <li>Basic timer 1 (1, 5, 100, 250 ms)</li> </ul>
Reset	<ul style="list-style-type: none"> <li>Power-ON reset (on power application)</li> <li>Reset by CE pin (CE pin low level → high level)</li> <li>Power failure detection function</li> </ul>

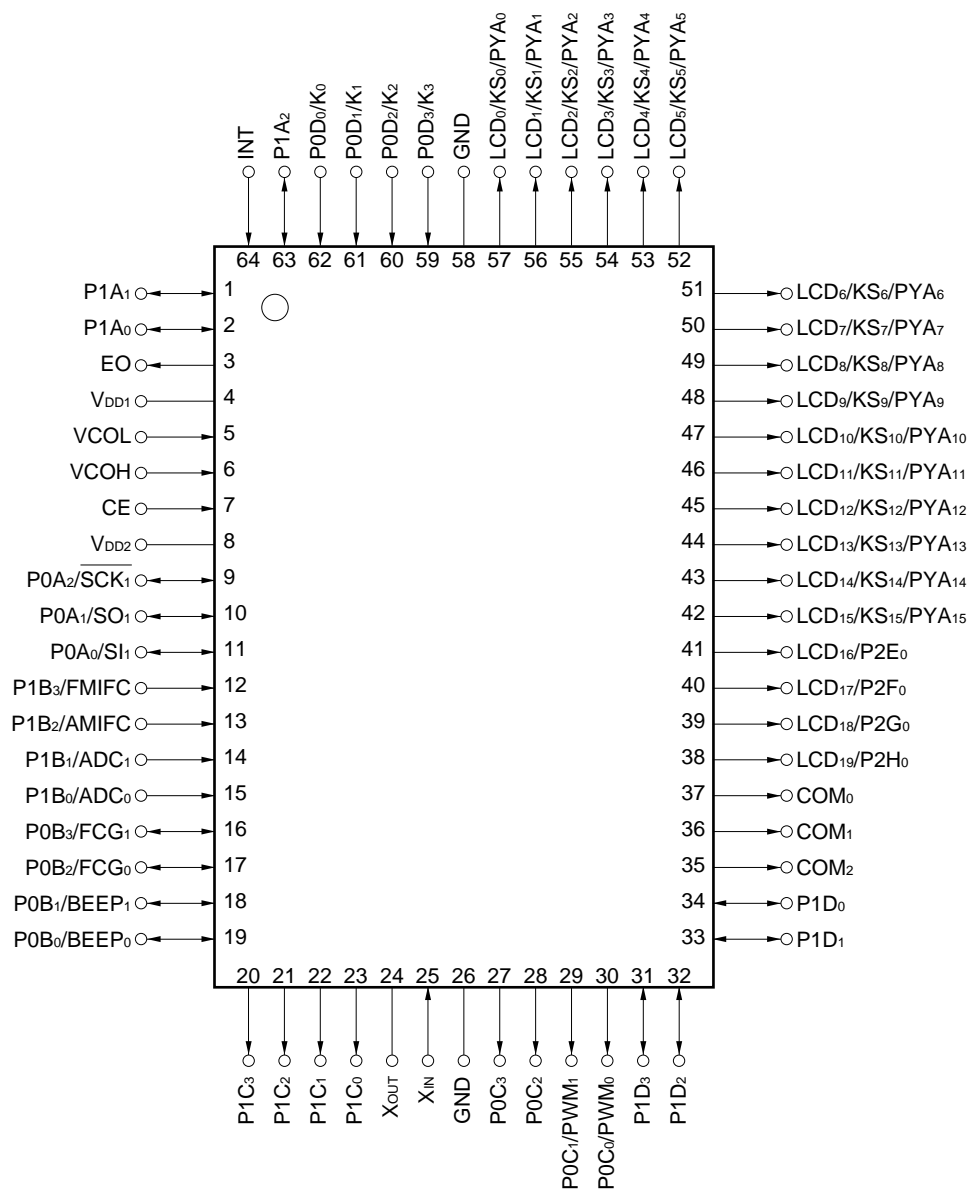
Item		Function
PLL frequency synthesizer	Division mode	<ul style="list-style-type: none"> <li>Two types               <ul style="list-style-type: none"> <li>Direct division mode (VCOL pin: 20 MHz MAX.)</li> <li>Pulse swallow mode (VCOL pin: 30 MHz MAX.)</li> <li>(VCOH pin: 250 MHz MAX.)</li> </ul> </li> </ul>
	Reference frequency	<ul style="list-style-type: none"> <li>12 programmable frequencies 1, 1.25, 2.5, 3, 5, 6.25, 9, 10, 12.5, 25, 50, 100 kHz</li> </ul>
	Charge pump	Error-out outputs: 1
	Phase comparator	Unlock detection by program
Frequency counter		<ul style="list-style-type: none"> <li>Frequency measurement               <ul style="list-style-type: none"> <li>P1B<sub>3</sub>/FMIFC pin: In FMIF mode 5 to 15 MHz</li> <li>In AMIF mode 0.3 to 1 MHz</li> <li>P1B<sub>2</sub>/AMIFC pin: 0.3 to 1 MHz</li> </ul> </li> <li>External gate width measurement P0B<sub>3</sub>/FCG<sub>1</sub>, P0B<sub>2</sub>/FCG<sub>0</sub> pins</li> </ul>
Supply voltage		$V_{DD} = 5\text{ V} \pm 10\%$
Package		64-pin plastic QFP (14 × 20 mm) 80-pin plastic QFP (14 × 14 mm)

★

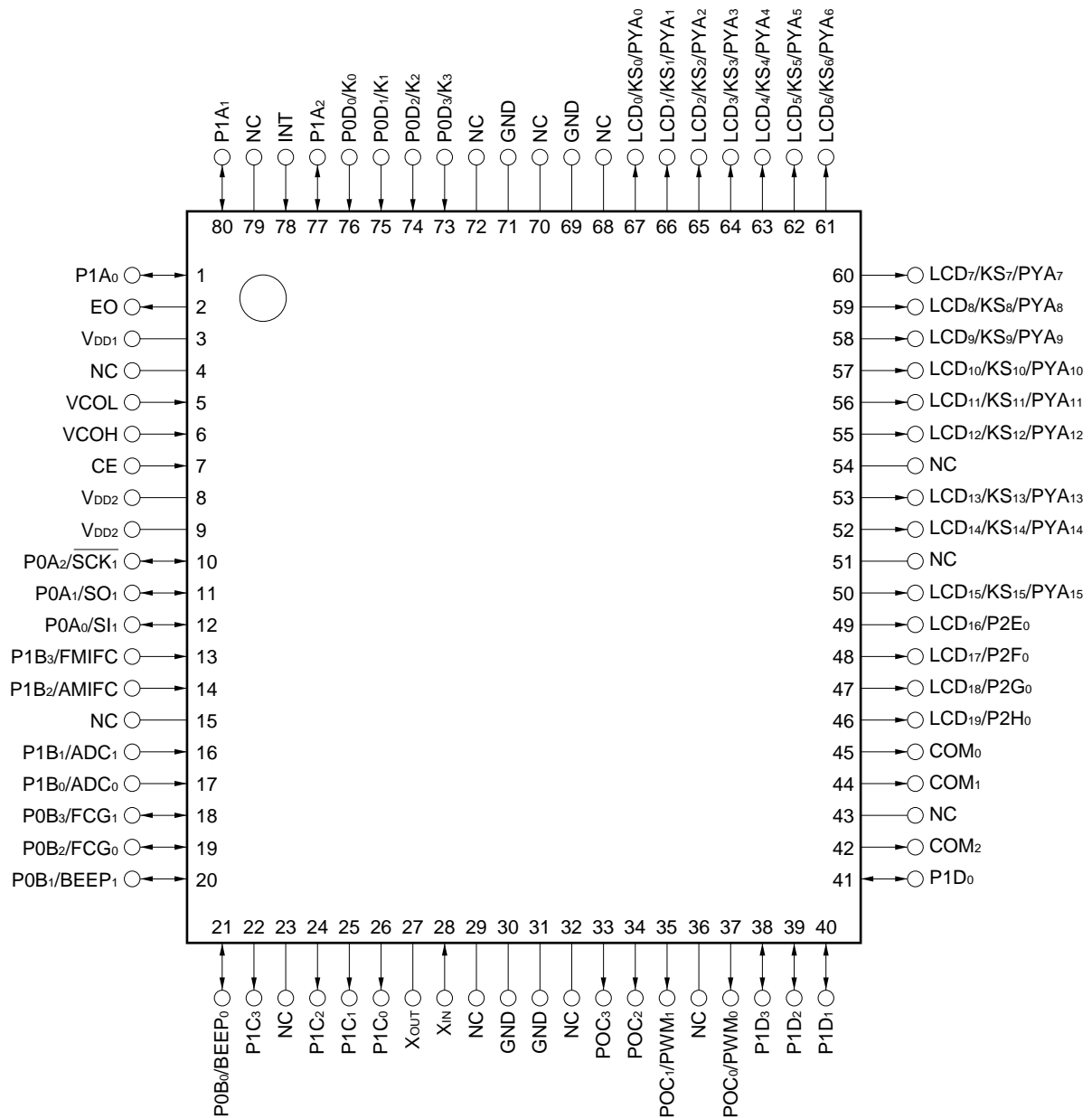
# PIN CONFIGURATION (TOP VIEW)

64-pin plastic QFP (14 × 20 mm)

μPD17012GF-xxx-3BE



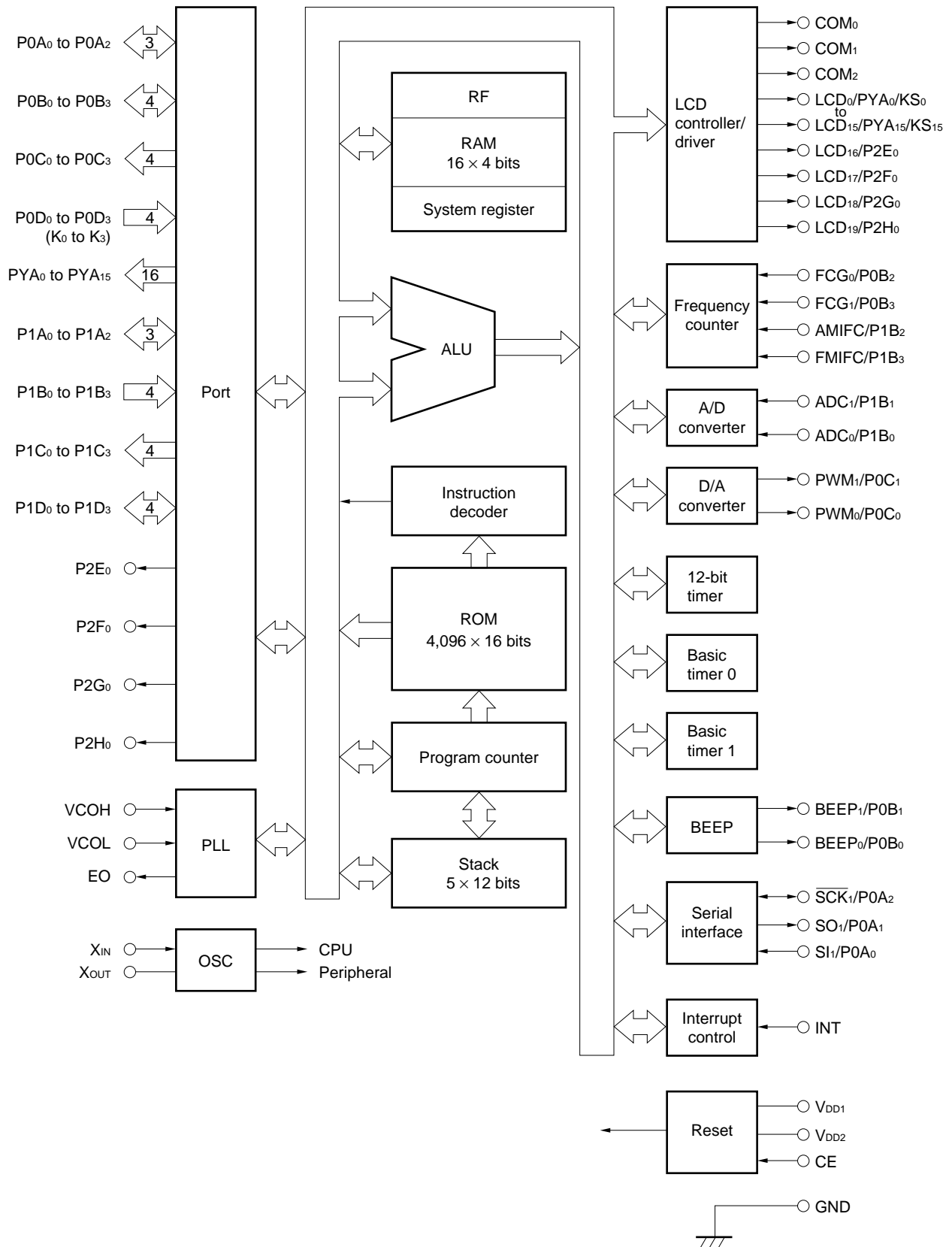
- ★ **80-pin plastic QFP (14 × 14 mm)**  
 μPD17012GC-xxx-8BT



# PIN IDENTIFICATION

ADC <sub>0</sub> , ADC <sub>1</sub> :	A/D converter input	P1A <sub>0</sub> to P1A <sub>2</sub> :	Port 1A
AMIFC:	AM intermediate frequency counter input	P1B <sub>0</sub> to P1B <sub>3</sub> :	Port 1B
BEEP <sub>0</sub> , BEEP <sub>1</sub> :	BEEP output	P1C <sub>0</sub> to P1C <sub>3</sub> :	Port 1C
CE:	Chip enable input	P1D <sub>0</sub> to P1D <sub>3</sub> :	Port 1D
COM <sub>0</sub> to COM <sub>2</sub> :	LCD common signal output	P2E <sub>0</sub> :	Port 2E
EO:	Error out output	P2F <sub>0</sub> :	Port 2F
FCG <sub>0</sub> , FCG <sub>1</sub> :	External gate counter input	P2G <sub>0</sub> :	Port 2G
FMIFC:	FM intermediate frequency counter input	P2H <sub>0</sub> :	Port 2H
GND:	Ground	PWM <sub>0</sub> , PWM <sub>1</sub> :	D/A converter output
INT:	External interrupt input	PYA <sub>0</sub> to PYA <sub>15</sub> :	Port YA
K <sub>0</sub> to K <sub>3</sub> :	Key source signal input	$\overline{\text{SCK}}_1$ :	Serial clock I/O
KS <sub>0</sub> to KS <sub>15</sub> :	Key source signal output	SI <sub>1</sub> :	Serial data input
LCD <sub>0</sub> to LCD <sub>19</sub> :	LCD segment signal output	SO <sub>1</sub> :	Serial data output
NC:	No connection	VCOH:	Local oscillation high input
P0A <sub>0</sub> to P0A <sub>2</sub> :	Port 0A	VCOL:	Local oscillation low input
P0B <sub>0</sub> to P0B <sub>3</sub> :	Port 0B	VDD <sub>1</sub> , VDD <sub>2</sub> :	Power supply
P0C <sub>0</sub> to P0C <sub>3</sub> :	Port 0C	XIN, XOUT:	Crystal resonator connection
P0D <sub>0</sub> to P0D <sub>3</sub> :	Port 0D		

# BLOCK DIAGRAM



## Contents

<b>1. PIN FUNCTIONS</b> .....	<b>13</b>
1.1 Pin Function List .....	13
1.2 Pin Equivalent Circuits .....	16
1.3 Recommended Connection of Unused Pins .....	20
1.4 Notes on Using CE and INT Pins .....	21
<b>2. PROGRAM MEMORY (ROM)</b> .....	<b>22</b>
2.1 Outline of Program Memory .....	22
2.2 Program Memory .....	23
2.3 Program Counter .....	23
2.4 Program Flow .....	23
<b>3. ADDRESS STACK (ASK)</b> .....	<b>26</b>
3.1 Outline of Address Stack .....	26
3.2 Address Stack Registers (ASR) .....	26
3.3 Stack Pointer (SP) .....	27
3.4 Operation of Address Stack .....	28
3.5 Notes on Using Address Stack .....	28
<b>4. DATA MEMORY (RAM)</b> .....	<b>29</b>
4.1 Outline of Data Memory .....	29
4.2 Configuration and Function of Data Memory .....	30
4.3 Addressing of Data Memory .....	32
4.4 Notes on Using Data Memory .....	33
<b>5. SYSTEM REGISTER (SYSREG)</b> .....	<b>34</b>
5.1 Outline of System Register .....	34
5.2 System Register List .....	35
5.3 Address Register (AR) .....	36
5.4 Window Register (WR) .....	38
5.5 Bank Register (BANK) .....	39
5.6 Index Register (IX) and Data Memory Row Address Pointer (MP: Memory Pointer) ...	40
5.7 General Register Pointer (RP) .....	42
5.8 Program Status Word (PSWORD) .....	44
5.9 Notes on Using System Register .....	45
<b>6. GENERAL REGISTER (GR)</b> .....	<b>46</b>
6.1 Outline of General Register .....	46
6.2 General Register Body .....	46
6.3 Address Generation of General Register by Instructions .....	47
6.4 Notes on Using General Register .....	48
<b>7. ALU (Arithmetic Logic Unit) BLOCK</b> .....	<b>49</b>
7.1 Outline of ALU Block .....	49



7.2	Configuration and Function of Each Block .....	50
7.3	ALU Processing Instruction List .....	50
7.4	Notes on Using ALU .....	54
8.	REGISTER FILE (RF) .....	55
8.1	Outline of Register File .....	55
8.2	Configuration and Function of Register File .....	56
8.3	Register File Manipulation Instructions (“PEEK WR, rf” and “POKE rf, WR”) .....	57
8.4	Control Registers .....	57
8.5	Notes on Using Register File .....	63
9.	DATA BUFFER (DBF) .....	64
9.1	Outline of Data Buffer .....	64
9.2	Data Buffer .....	65
9.3	Peripheral Hardware and Data Buffer List .....	66
9.4	Notes on Using Data Buffer .....	70
10.	GENERAL-PURPOSE PORTS .....	71
10.1	Configuration and Classification of General-Purpose Ports .....	71
10.2	Functional Outline of General-Purpose Ports .....	73
10.3	General-Purpose I/O Ports (P0A, P0B, P1A, and P1D) .....	78
10.4	General-Purpose Input Ports (P0D and P1B) .....	84
10.5	General-Purpose Output Ports (P0C and P1C) .....	86
10.6	General-Purpose Output Ports (P2E to P2H and PYA) .....	88
11.	INTERRUPTS .....	95
11.1	Outline of Interrupt Block .....	95
11.2	Interrupt Control Block .....	97
11.3	Interrupt Stack Register .....	101
11.4	Stack Pointer, Address Stack Register, Program Counter .....	102
11.5	Interrupt Enable Flip-Flop (INTE) .....	102
11.6	Acknowledging Interrupts .....	103
11.7	Operations After Acknowledging Interrupt .....	108
11.8	Restoring from Interrupt Servicing Routine .....	108
11.9	External (INT Pin) Interrupt .....	109
11.10	Internal Interrupt .....	111
12.	TIMER .....	112
12.1	General .....	112
12.2	Basic Timer 0 .....	113
12.3	Basic Timer 1 .....	126
12.4	12-Bit Timer .....	134
13.	A/D CONVERTER (ADC) .....	142
13.1	General .....	142
13.2	Input Selector Block .....	143
13.3	Compare Voltage Generator Block and Compare Block .....	144
13.4	Comparison Timing Chart .....	148
13.5	Performance of A/D Converter .....	148

13.6	Using A/D Converter .....	149
13.7	Status on Reset .....	154
14.	D/A CONVERTER (DAC) .....	155
14.1	Configuration of D/A Converter .....	155
14.2	Functional Outline of D/A Converter .....	155
14.3	Output Select Blocks .....	156
14.4	Duty Setting Blocks and Clock Generation Block .....	158
14.5	Cautions When Using D/A Converter .....	161
14.6	Status on Reset .....	162
15.	SERIAL INTERFACE .....	163
15.1	Configuration of Serial Interface .....	164
15.2	Functional Outline of Serial Interface .....	165
15.3	Shift Clock and Serial Data I/O Pin Control Blocks .....	166
15.4	Clock Generation Block .....	168
15.5	Clock Counter .....	170
15.6	Presetable Shift Register (SIO1SFR) .....	171
15.7	Wait Control Block .....	173
15.8	Outline of Serial Interface Operation .....	175
15.9	Status of Serial Interface on Reset .....	176
16.	PLL FREQUENCY SYNTHESIZER .....	177
16.1	Configuration of PLL Frequency Synthesizer .....	177
16.2	Functional Outline of PLL Frequency Synthesizer .....	178
16.3	Input Select Block and Programmable Divider .....	179
16.4	Reference Frequency Generator .....	184
16.5	Phase Comparator ( $\phi$ -DET), Charge Pump, and Unlock Detection Block .....	186
16.6	PLL Disabled Status .....	189
16.7	Using PLL Frequency Synthesizer .....	189
16.8	Status on Reset .....	193
17.	FREQUENCY COUNTER .....	194
17.1	Outline of Frequency Counter .....	194
17.2	Input/Output Select Block and Gate Time Control Block .....	195
17.3	Start/Stop Control Block and IF Counter .....	198
17.4	Using IF Counter Function .....	204
17.5	Error of External Gate Counter .....	206
17.6	Status on Reset .....	206
18.	BEEP .....	207
18.1	General .....	207
18.2	I/O Select Block and Output Select Block .....	208
18.3	Clock Select Block and Clock Generator Block .....	210
18.4	Output Waveform of BEEP .....	211
18.5	Status on Reset .....	211

19. LCD CONTROLLER/DRIVER .....	212
19.1 Configuration of LCD Controller/Driver .....	212
19.2 Functional Outline of LCD Controller/Driver .....	213
19.3 LCD Segment Register .....	215
19.4 Segment Signal/General-Purpose Output Port Select Block .....	218
19.5 Common Signal Output Timing Control Block and Segment Signal/Key Source Signal Output Timing Control Block .....	220
19.6 Output Waveforms of Common and Segment Signals .....	222
19.7 Using LCD Controller/Driver .....	226
19.8 Status on Reset .....	228
20. KEY SOURCE CONTROLLER/DECODER .....	229
20.1 Configuration of Key Source Controller/Decoder .....	229
20.2 Functional Outline of Key Source Controller/Decoder .....	230
20.3 Key Source Data Setting Block .....	231
20.4 Output Timing Control Blocks and Segment/Port Select Block .....	233
20.5 Key Input Control Block .....	237
20.6 Using Key Source Controller/Decoder .....	240
20.7 Status on Reset .....	248
21. STANDBY .....	249
21.1 Configuration of Standby Block .....	249
21.2 Standby Function .....	250
21.3 Selecting Device Operation Mode with CE Pin .....	250
21.4 Halt Function .....	252
21.5 Clock Stop Function .....	260
21.6 Device Operations in Halt and Clock Stop Status .....	263
21.7 Notes on Processing Each Pin in Halt and Clock Stop Status .....	264
22. RESET .....	267
22.1 Configuration of Reset Block .....	267
22.2 Reset Function .....	268
22.3 CE Reset .....	269
22.4 Power-on Reset .....	273
22.5 Relationship Between CE Reset and Power-on Reset .....	276
22.6 Power Failure Detection .....	280
23. INSTRUCTION SET .....	288
23.1 Outline of Instruction Set .....	288
23.2 Legend .....	289
23.3 Instruction Set List .....	290
23.4 Assembler (RA17K) Embedded Macro Instructions .....	292
24. RESERVED SYMBOLS .....	293
24.1 Data Buffer (DBF) .....	293
24.2 System Register (SYSREG) .....	293
24.3 LCD Segment Register .....	294
24.4 Port Register .....	295

24.5	Register File (Control Register) .....	296
24.6	Peripheral Hardware Register .....	298
24.7	Others .....	298
25.	ELECTRICAL SPECIFICATIONS .....	299
26.	PACKAGE DRAWINGS .....	302
27.	RECOMMENDED SOLDERING CONDITIONS .....	304
APPENDIX A. NOTES ON CONNECTING CRYSTAL RESONATOR .....		305
APPENDIX B. DEVELOPMENT TOOLS .....		306

## 1. PIN FUNCTIONS

### ★ 1.1 Pin Function List

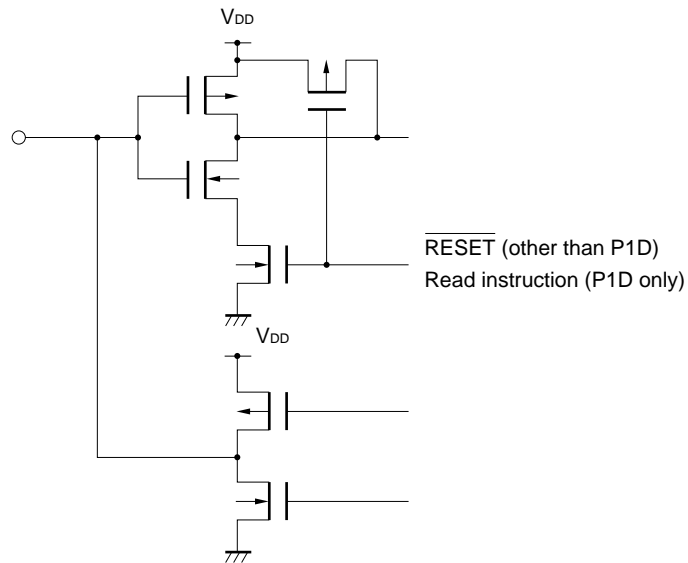
Pin No.		Symbol	Function	Output Format	After Power-on Reset
64-pin	80-pin				
63 1 2	77 80 1	P1A <sub>2</sub> P1A <sub>1</sub> P1A <sub>0</sub>	3-bit I/O port (port 1A). Input/output can be specified in 1-bit units.	CMOS push-pull	Input
3	2	EO	Output from PLL frequency synthesizer charge pump. The division value of the local oscillation frequency and the phase of the reference frequency are compared at this pin, and the result is output.	CMOS 3-state	High impedance
4 8	3 8, 9	V <sub>DD1</sub> V <sub>DD2</sub>	Positive power-supply pins. 5 V+/-10% is supplied to these pins. When only the CPU is operating, 3.5 to 5.5 V is supplied to these pins. 2.3 to 5.5 V is supplied when the clock is stopped. The same potential voltage is supplied to V <sub>DD1</sub> and V <sub>DD2</sub> .	—	—
5 6	5 6	VCOL VCOH	PLL local oscillation frequency is input.	—	Input
7	7	CE	Device selection and reset signal input.	—	Input
9 10 11	10 11 12	P0A <sub>2</sub> /SCK <sub>1</sub> P0A <sub>1</sub> /SO <sub>1</sub> P0A <sub>0</sub> /SI <sub>1</sub>	Port 0A and serial interface I/O pins. <ul style="list-style-type: none"> <li>• P0A<sub>2</sub> to P0A<sub>0</sub></li> <li>• 3-bit I/O port</li> <li>• Input/output can be specified in 1-bit units.</li> <li>• SCK<sub>1</sub></li> <li>• Serial clock I/O</li> <li>• SO<sub>1</sub></li> <li>• Serial data output</li> <li>• SI<sub>1</sub></li> <li>• Serial data input</li> </ul>	CMOS push-pull	Input
12 13 14 15	13 14 16 17	P1B <sub>3</sub> /FMIFC P1B <sub>2</sub> /AMIFC P1B <sub>1</sub> /ADC <sub>1</sub> P1B <sub>0</sub> /ADC <sub>0</sub>	Port 1B. Frequency counter input and analog input to A/D converter pins. <ul style="list-style-type: none"> <li>• P1B<sub>3</sub> to P1B<sub>0</sub></li> <li>• 4-bit input port</li> <li>• FMIFC, AMIFC</li> <li>• Frequency counter inputs</li> <li>• ADC<sub>1</sub>, ADC<sub>0</sub></li> <li>• Analog inputs to A/D converter</li> </ul>	—	Input

Pin No.		Symbol	Function	Output Format	After Power-on Reset
64-pin	80-pin				
16 17 18 19	18 19 20 21	P0B <sub>3</sub> /FCG <sub>1</sub> P0B <sub>2</sub> /FCG <sub>0</sub> P0B <sub>1</sub> /BEEP <sub>1</sub> P0B <sub>0</sub> /BEEP <sub>0</sub>	Port 0B. External gate counter input and BEEP output pins. <ul style="list-style-type: none"> <li>P0B<sub>3</sub> to P0B<sub>0</sub></li> <li>4-bit I/O port</li> <li>Input/output can be specified in 1-bit units.</li> <li>FCG<sub>1</sub>, FCG<sub>0</sub></li> <li>External gate counter inputs</li> <li>BEEP<sub>1</sub>, BEEP<sub>0</sub></li> <li>BEEP outputs</li> </ul>	CMOS push-pull	Input
20 21 22 23	22 24 25 26	P1C <sub>3</sub> P1C <sub>2</sub> P1C <sub>1</sub> P1C <sub>0</sub>	4-bit output port (port 1C)	CMOS push-pull	Low-level output
24	27	X <sub>OUT</sub>	Pins for connecting crystal resonator for system clock oscillation.	CMOS push-pull	—
25	28	X <sub>IN</sub>		—	
26 58	30, 69 31, 71	GND	Ground pins. These pins must be connected to the same potential.	—	—
27 28 29 30	33 34 35 37	P0C <sub>3</sub> P0C <sub>2</sub> P0C <sub>1</sub> /PWM <sub>1</sub> P0C <sub>0</sub> /PWM <sub>0</sub>	Port 0C. D/A converter output pins. <ul style="list-style-type: none"> <li>P0C<sub>3</sub> to P0C<sub>0</sub></li> <li>4-bit output port</li> <li>PWM<sub>1</sub>, PWM<sub>0</sub></li> <li>D/A converter outputs</li> </ul>	N-ch open-drain (+12 V withstand voltage)	Low-level output
31 32 33 34	38 39 40 41	P1D <sub>3</sub> P1D <sub>2</sub> P1D <sub>1</sub> P1D <sub>0</sub>	4-bit I/O port (port 1D). Input/output can be specified in 4-bit units.	CMOS push-pull	Input
35 36 37	42 44 45	COM <sub>2</sub> COM <sub>1</sub> COM <sub>0</sub>	These pins output the common signals of the LCD controller/driver.	CMOS ternary output	Low-level output
38 39 40 41	46 47 48 49	LCD <sub>19</sub> /P2H <sub>0</sub> LCD <sub>18</sub> /P2G <sub>0</sub> LCD <sub>17</sub> /P2F <sub>0</sub> LCD <sub>16</sub> /P2E <sub>0</sub>	Port 2H, 2G, 2F, and 2E. LCD controller/driver segment signal output pins. <ul style="list-style-type: none"> <li>P2H<sub>0</sub>, P2G<sub>0</sub>, P2F<sub>0</sub>, P2E<sub>0</sub></li> <li>1-bit output ports</li> <li>LCD<sub>19</sub> to LCD<sub>16</sub></li> <li>LCD controller/driver segment signal outputs</li> </ul>	CMOS push-pull	Low-level output

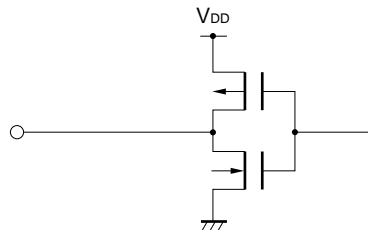
Pin No.		Symbol	Function	Output Format	After Power-on Reset
64-pin	80-pin				
42 to 57	50 52 53 55 to 67	LCD <sub>15</sub> /KS <sub>15</sub> /PYA <sub>15</sub> to LCD <sub>0</sub> /KS <sub>0</sub> /PYA <sub>0</sub>	Port YA. Segment signal output of LCD controller/driver and key source signal output of key matrix. <ul style="list-style-type: none"> <li>• PYA<sub>15</sub> to PYA<sub>0</sub></li> <li>• 16-bit output port</li> <li>• LCD<sub>15</sub> to LCD<sub>0</sub></li> <li>• LCD controller/driver segment signal outputs</li> <li>• KS<sub>15</sub> to KS<sub>0</sub></li> <li>• Key matrix key source signal outputs</li> </ul>	CMOS push-pull	Low-level output
59 to 62	73 to 76	P0D <sub>3</sub> /K <sub>3</sub> to P0D <sub>0</sub> /K <sub>0</sub>	Port 0D. Key source signal return input of LCD segment. <ul style="list-style-type: none"> <li>• P0D<sub>3</sub> to P0D<sub>0</sub></li> <li>• 4-bit input port</li> <li>• K<sub>3</sub> to K<sub>0</sub></li> <li>• Key source signal return inputs</li> </ul>	—	Input with pull-down resistor
64	78	INT	Vector interrupt pin for edge detection. Rising or falling edge can be selected.	—	Input

## 1.2 Pin Equivalent Circuits

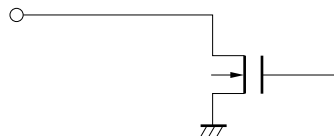
- (1) P0A (P0A<sub>2</sub>/ $\overline{\text{SCK}}_1$ , P0A<sub>1</sub>/SO<sub>1</sub>, P0A<sub>0</sub>/SI<sub>1</sub>)  
 P0B (P0B<sub>3</sub>/FCG<sub>1</sub>, P0B<sub>2</sub>/FCG<sub>0</sub>, P0B<sub>1</sub>/BEEP<sub>1</sub>, P0B<sub>0</sub>/BEEP<sub>0</sub>)  
 P1A (P1A<sub>2</sub>, P1A<sub>1</sub>, P1A<sub>0</sub>)  
 P1D (P1D<sub>3</sub>, P1D<sub>2</sub>, P1D<sub>1</sub>, P1D<sub>0</sub>)
- (I/O)



- (2) P1C (P1C<sub>3</sub>, P1C<sub>2</sub>, P1C<sub>1</sub>, P1C<sub>0</sub>)  
 LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub>  
 LCD<sub>19</sub>/P2H<sub>0</sub>, LCD<sub>18</sub>/P2G<sub>0</sub>, LCD<sub>17</sub>/P2F<sub>0</sub>, LCD<sub>16</sub>/P2E<sub>0</sub>,
- (Output)

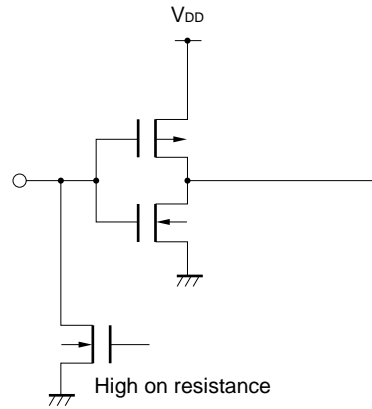


- (3) P0C (P0C<sub>3</sub>, P0C<sub>2</sub>, P0C<sub>1</sub>/PWM<sub>1</sub>, P0C<sub>0</sub>/PWM<sub>0</sub>) (Output)

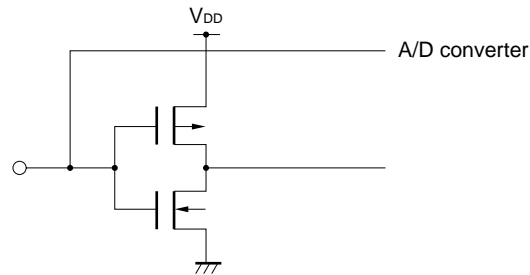




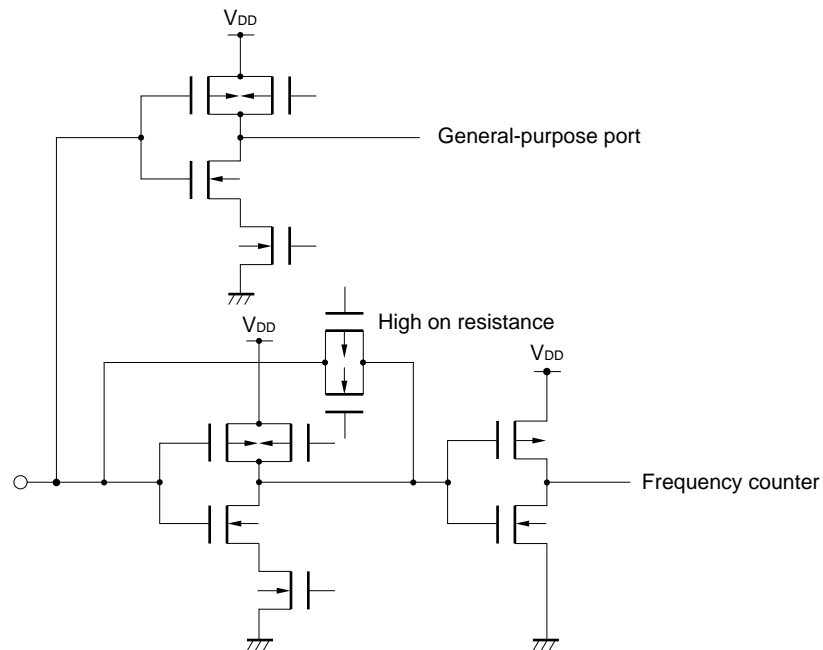
(4) P0D (P0D3/K3, P0D2/K2, P0D1/K1, P0D0/K0) (Input)



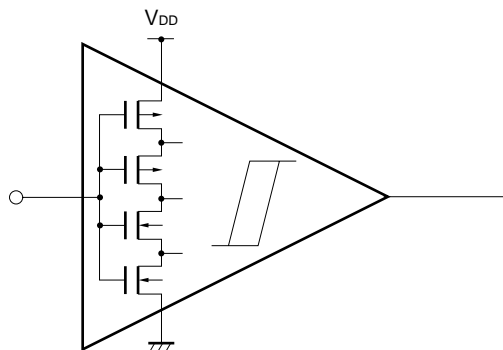
(5) P1B (P1B1/ADC1, P1B0/ADC0) (Input)



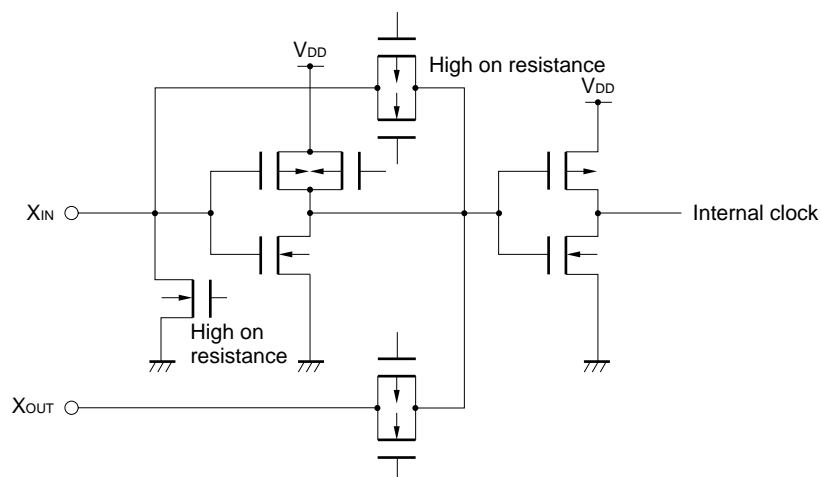
(6) P1B (P1B3/FMIFC, P1B2/AMIFC) (Input)



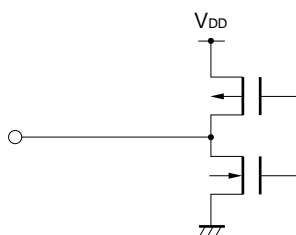
(7) CE  
INT } (Schmitt trigger input)



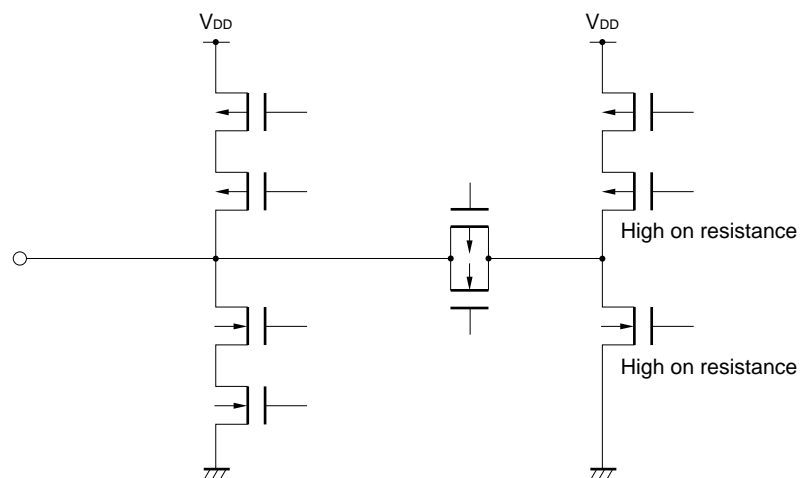
(8) X<sub>OUT</sub> (Output), X<sub>IN</sub> (Input)



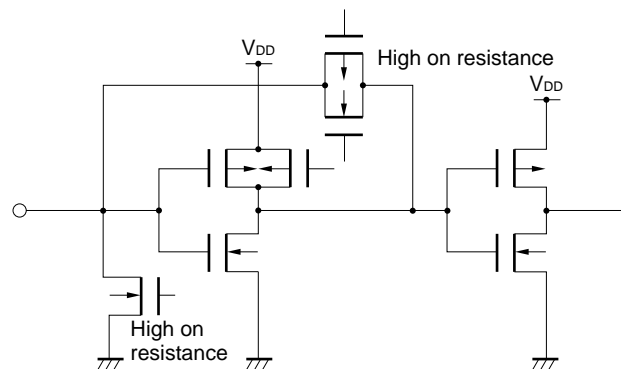
(9) EO (Output)



(10) COM<sub>2</sub>  
COM<sub>1</sub>  
COM<sub>0</sub> } (Output)



(11) VCOH  
VCOL } (Input)



### 1.3 Recommended Connection of Unused Pins

The following connections are recommended for unused pins.

**Table 1-1. Recommended Connection of Unused Pins**

Pin Name		I/O Mode	Recommended Connection of Unused Pins
Port pins	P0D <sub>0</sub> /K <sub>0</sub> to P0D <sub>3</sub> /K <sub>3</sub>	Input	Independently connect to GND via a resistor <sup>Note 1</sup> .
	P1B <sub>0</sub> /ADC <sub>0</sub>		Independently connect to V <sub>DD</sub> or GND via a resistor <sup>Note 1</sup> .
	P1B <sub>1</sub> /ADC <sub>1</sub>		
	P1B <sub>2</sub> /AMIFC <sup>Notes 2, 3</sup>		Set to P1B <sub>2</sub> and connect to V <sub>DD</sub> or GND via a resistor <sup>Note 1</sup> .
	P1B <sub>3</sub> /FMIFC <sup>Notes 2, 3</sup>		Set to P1B <sub>3</sub> and connect to V <sub>DD</sub> or GND via a resistor <sup>Note 1</sup> .
	P1C <sub>0</sub> /P1C <sub>3</sub>	CMOS push-pull output	Leave open.
	P2E <sub>0</sub> /LCD <sub>16</sub>		
	P2F <sub>0</sub> /LCD <sub>17</sub>		
	P2G <sub>0</sub> /LCD <sub>18</sub>		
	P2H <sub>0</sub> /LCD <sub>19</sub>		
	PYA <sub>0</sub> /LCD <sub>0</sub> /KS <sub>0</sub> to PYA <sub>15</sub> /LCD <sub>15</sub> /KS <sub>15</sub>	N-ch open-drain output	Set to low-level output by software, and leave open.
	P0C <sub>0</sub> /PWM <sub>0</sub>		
	P0C <sub>1</sub> /PWM <sub>1</sub>		
	P0C <sub>2</sub> , P0C <sub>3</sub>		
	P0A <sub>0</sub> /SI <sub>1</sub>	I/O <sup>Note 4</sup>	Set to general-purpose input port by software, and independently connect to V <sub>DD</sub> or GND via a resistor <sup>Note 1</sup> .
	P0A <sub>1</sub> /SO <sub>1</sub>		
	P0A <sub>2</sub> /SCK <sub>1</sub>		
	P0B <sub>0</sub> /BEEP <sub>0</sub>		
	P0B <sub>1</sub> /BEEP <sub>1</sub>		
	P0B <sub>2</sub> /FCG <sub>0</sub> <sup>Notes 2, 3</sup>		
	P0B <sub>3</sub> /FCG <sub>1</sub> <sup>Notes 2, 3</sup>		
	P1A <sub>0</sub> to P1A <sub>2</sub>		
	P1D <sub>0</sub> to P1D <sub>3</sub>		
Non-port pins	CE	Input	Connect to V <sub>DD</sub> via a resistor <sup>Note 1</sup> .
	INT		Connect to GND via a resistor <sup>Note 1</sup> .
	VCOH, VCOL		Disable by software, and leave open.
	COM <sub>0</sub> to COM <sub>2</sub>	Output	Leave open.
	EO		

- Notes 1.** Note that when pulling up (connecting to V<sub>DD</sub> via a resistor) or pulling down (connecting to GND via a resistor) a pin externally using a high resistance value, the current consumption (through current) of the port increases because the pin approaches the high-impedance state. Generally, a resistance value of several tens of kΩ suffices for pull up and pull down, although this value depends on each application circuit.
- 2.** This general-purpose input port has a circuit designed so that the current consumption does not increase even in the high-impedance state.
- 3.** Do not set this pin to AMIFC, FMIFC, FCG<sub>0</sub> or FCG<sub>1</sub>, or the current consumption will increase.
- 4.** These input/output ports become general-purpose input ports at power-on, clock stop, and CE reset.

#### 1.4 Notes on Using CE and INT Pins

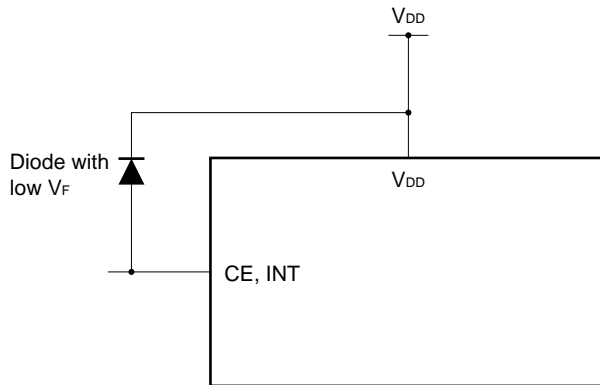
The CE and INT pins have a function to set a test mode (for IC test) in which the internal operations of the  $\mu$ PD17012 are tested, in addition to the functions listed in **1.1 Pin Function List**.

If a voltage higher than  $V_{DD}$  is applied to either of these pins, the test mode is set. Therefore, if noise exceeding  $V_{DD}$  is applied to these pins even during normal operation, the test mode may be set by mistake, affecting normal operation.

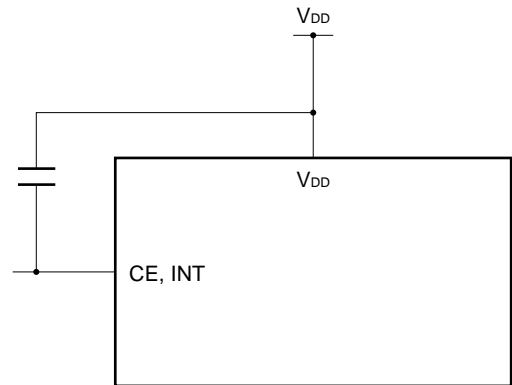
Noise may be superimposed on these pins if the length of the wiring of these pins is too long.

Therefore, keep the wiring length as short as possible. If noise is inevitable, take noise suppression measures by using an external component as illustrated below.

- Connect a diode with low  $V_F$  between CE or INT and  $V_{DD}$



- Connect a capacitor between CE or INT and  $V_{DD}$



## 2. PROGRAM MEMORY (ROM)

### 2.1 Outline of Program Memory

Figure 2-1 illustrates the program memory.

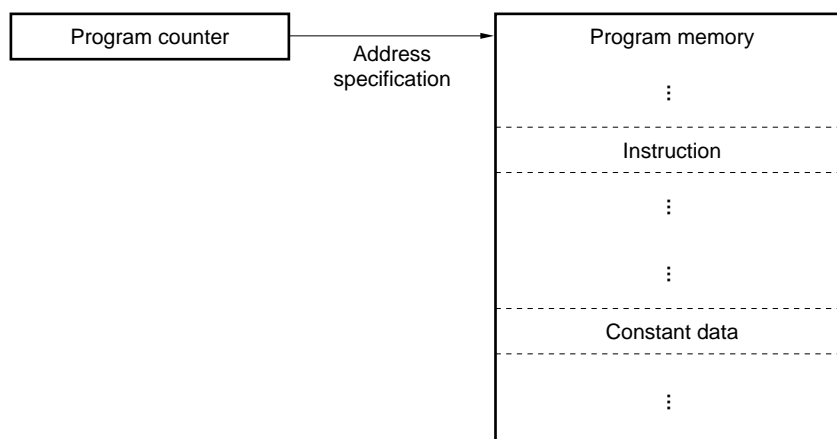
As shown in this figure, the program memory consists of program memory and a program counter.

The addresses of the program memory are specified by the program counter.

The program memory has the following two major functions.

- (1) Storing programs
- (2) Storing constant data

**Figure 2-1. Outline of Program Memory**



## 2.2 Program Memory

Figure 2-2 shows the configuration of the program memory.

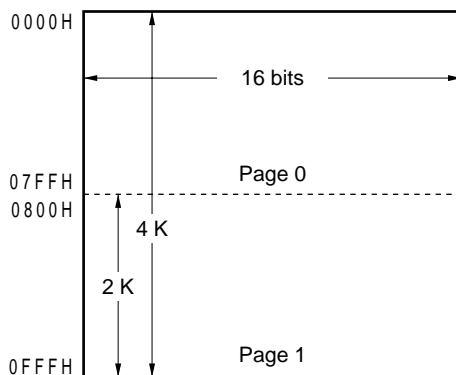
As shown in this figure, the program memory has a configuration of 4,096 steps × 16 bits.

Therefore, program memory addresses are addresses 0000H to 0FFFH.

All instructions are 1-word instructions, 16 bits long, so that one instruction can be stored in one address of the program memory.

The constant data in the program memory contents is read to the data buffer by using a table reference instruction.

**Figure 2-2. Configuration of Program Memory**



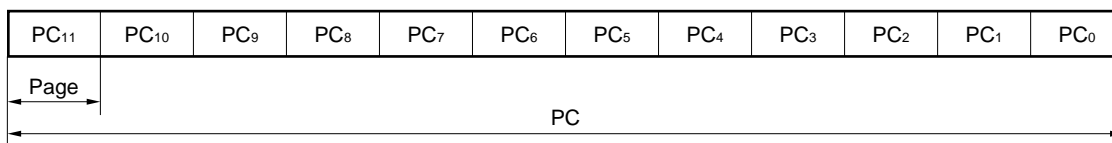
## 2.3 Program Counter

Figure 2-3 shows the configuration of the program counter.

As shown in this figure, the program counter is configured as a 12-bit binary counter. The most significant bit, b<sub>11</sub>, indicates a page.

The program counter specifies an address of the program memory.

**Figure 2-3. Configuration of Program Counter**



## 2.4 Program Flow

The execution flow of the program is controlled by the program counter, which specifies an address of the program memory.

Figure 2-4 shows the value set to the program counter when each instruction is executed.

Table 2-1 shows the vector address when an interrupt is acknowledged.

### 2.4.1 Branch instructions

#### (1) Direct branch ("BR addr")

The branch destination address of the direct branch instruction is in the area of addresses 0000H to 0FFFFH, i.e. all the addresses of the program memory.

#### (2) Indirect branch ("BR @AR")

The branch destination address of the indirect branch instruction is in the area of addresses 0000H to 0FFFFH, i.e. all the addresses of the program memory.

Also refer to **5.3 Address Register (AR)**.

### 2.4.2 Subroutine

#### (1) Direct subroutine call ("CALL addr")

The top address of the subroutine that can be called by the direct subroutine call instruction is within page 0 (addresses 0000H to 07FFH) in the program memory.

#### (2) Indirect subroutine call ("CALL @AR")

The top address of the subroutine that can be called by the indirect subroutine call instruction is in the area of addresses 0000H to 0FFFFH, i.e. all the addresses of the program memory.

Also refer to **5.3 Address Register (AR)**.

### 2.4.3 Table referencing

Addresses that can be referenced by the table reference instruction ("MOVT DBF, @AR") are in the area of addresses 0000H to 0FFFFH, i.e. all the addresses of the program memory.

Also refer to **5.3 Address Register (AR)** and **9.2.2 Table reference instruction ("MOVT DBF, @AR")**.



Figure 2-4. Specification of Program Counter on Instruction Execution

Instruction \ Program Counter		Contents of Program Counter (PC)												
		b <sub>11</sub>	b <sub>10</sub>	b <sub>9</sub>	b <sub>8</sub>	b <sub>7</sub>	b <sub>6</sub>	b <sub>5</sub>	b <sub>4</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	
BR addr	Page 0	0	Instruction operand (addr)											
	Page 1	1												
CALL addr		0	Instruction operand (addr)											
BR @AR CALL @AR MOVT DBF, @AR		Address register contents												
RET RETSK RETI		Contents of address stack register (ASR) specified by stack pointer (SP) (Return address)												
When interrupt is acknowledged		Vector address of each interrupt												
Power-on reset, CE reset		0	0	0	0	0	0	0	0	0	0	0	0	

Table 2-1. Interrupt Vector Address

Priority	Internal/External	Interrupt Source	Vector address
1	External	INT pin	0004H
2	Internal	12-bit timer	0003H
3	Internal	Basic timer 1	0002H
4	Internal	Serial interface	0001H

### 3. ADDRESS STACK (ASK)

#### 3.1 Outline of Address Stack

Figure 3-1 illustrates the address stack.

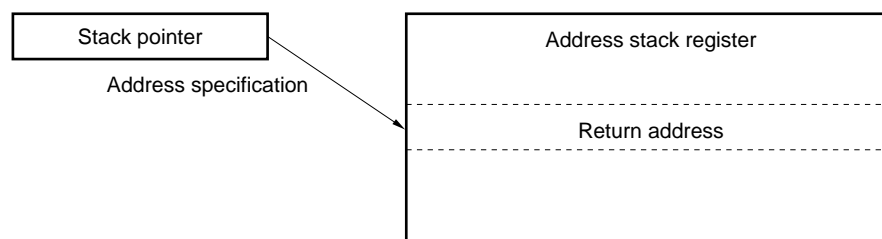
The address stack consists of a stack pointer and address stack registers.

The addresses of the address stack registers are specified by the stack pointer.

The address stack saves a return address when a subroutine call instruction is executed or when an interrupt is acknowledged.

The address stack is also used when a table reference instruction is executed.

**Figure 3-1. Outline of Address Stack**



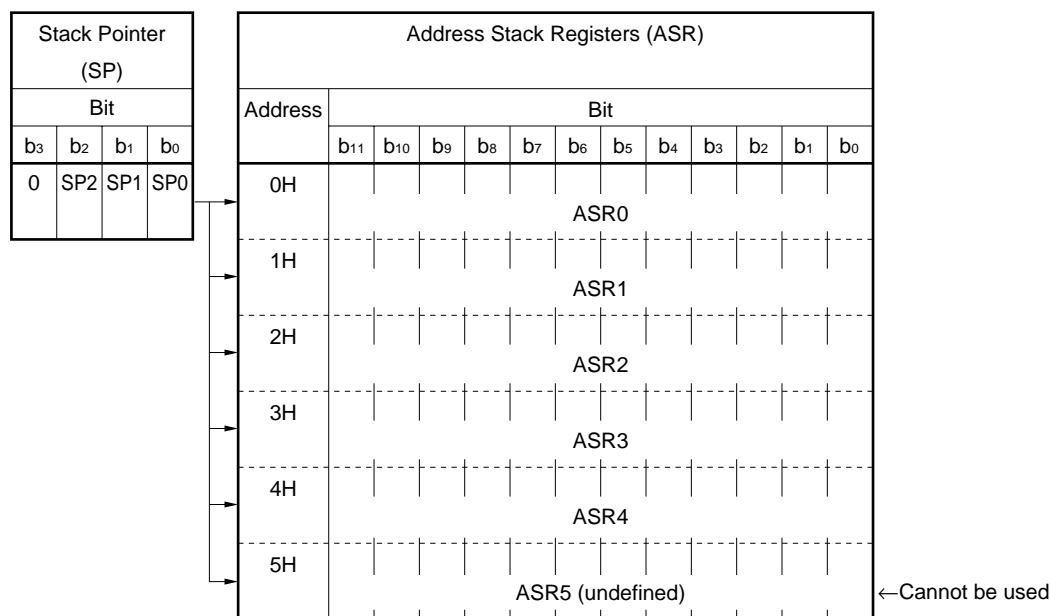
#### 3.2 Address Stack Registers (ASR)

Figure 3-2 shows the configuration of the address stack registers.

Although there are six 12-bit address stack registers: ASR0 to ASR5, no register is assigned to ASR5, and five 12-bit registers, ASR0 to ASR4, are used.

The address stack saves a return address when a subroutine call instruction or table reference instruction is executed, or when an interrupt is acknowledged.

**Figure 3-2. Configuration of Address Stack Registers**



### 3.3 Stack Pointer (SP)

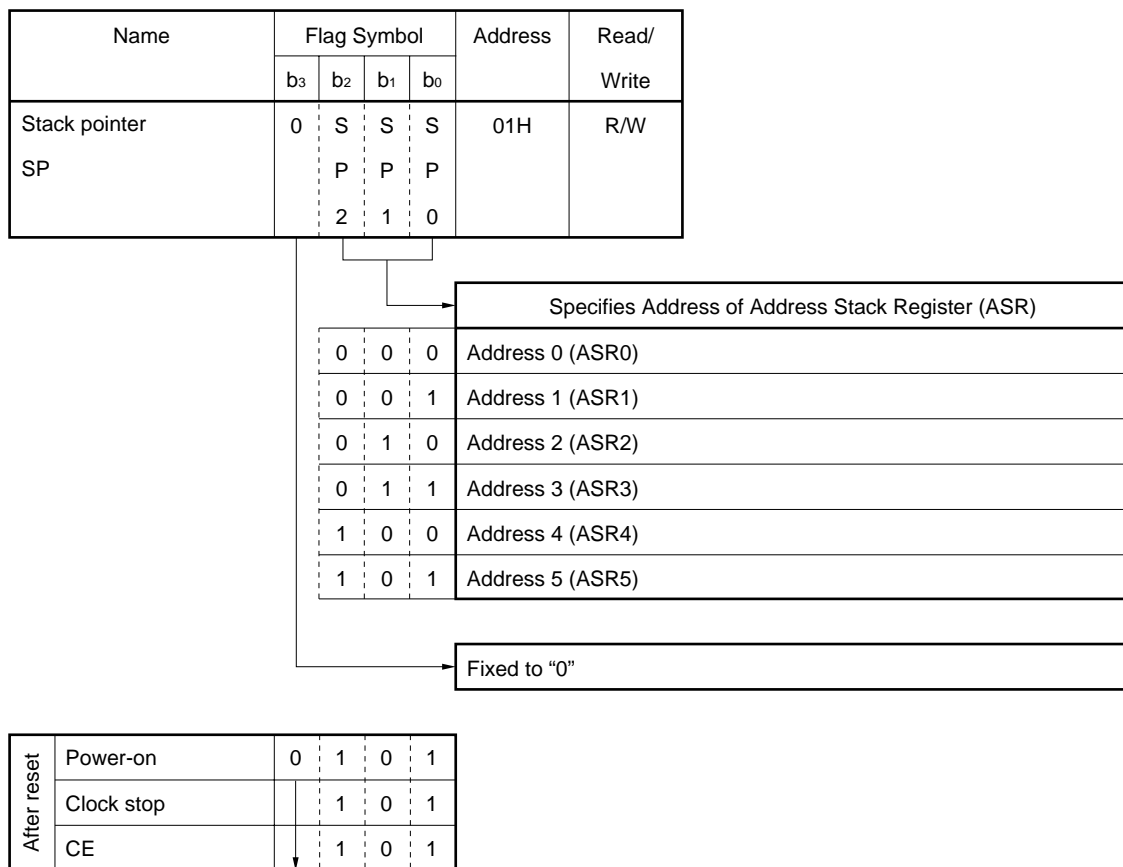
Figure 3-3 shows the configuration and function of the stack pointer.

The stack pointer is a 4-bit binary counter.

It specifies the address of an address stack register.

The value of the stack pointer can be directly read or written by using a register manipulation instruction.

**Figure 3-3. Configuration and Function of Stack Pointer**



### 3.4 Operation of Address Stack

#### 3.4.1 On execution of subroutine call (“CALL addr”, “CALL @AR”) or return (“RET”, “RETSK”) instruction

When a subroutine call instruction is executed, the value of the stack pointer is decremented by one and a return address is saved to the address stack register specified by the stack pointer.

When a return instruction is executed, the contents (return address) of the address stack register specified by the stack pointer are restored to the program counter, and the value of the stack pointer is incremented by one.

#### 3.4.2 On execution of table reference instruction (“MOVT DBF, @AR”)

When a table reference instruction is executed, the value of the stack pointer is decremented by one, and a return address is saved to the address stack register specified by the stack pointer.

Next, the contents of the program memory specified by the address register are read to the data buffer, the contents (return address) of the address stack register specified by the stack pointer are restored to the program counter, and then the value of the stack pointer is incremented by one.

#### 3.4.3 On acknowledgement of interrupt and execution of return instruction (“RETI”)

When an interrupt is acknowledged, the value of the stack pointer is decremented by one, and the return address is saved to the address stack register specified by the stack pointer.

When a return instruction is executed, the contents (return address) of the address stack register specified by the stack pointer are restored to the program counter, and the value of the stack pointer is incremented by one.

#### 3.4.4 On execution of address stack manipulation instruction (“PUSH AR”, “POP AR”)

When the “PUSH” instruction is executed, the value of the stack pointer is decremented by one, and the contents of the address register are transferred to the address stack register specified by the stack pointer.

When the “POP” instruction is executed, the contents of the address stack register specified by the stack pointer are transferred to the address register, and the value of the stack pointer is incremented by one.

### 3.5 Notes on Using Address Stack

The nesting level of the address stack is 5 and the value of the address stack register (ASR5) is “undefined” when the value of the stack pointer is 05H.

Do not use a subroutine call or interrupt exceeding level 5 without manipulating the stack; otherwise, execution returns to an undefined address.

## 4. DATA MEMORY (RAM)

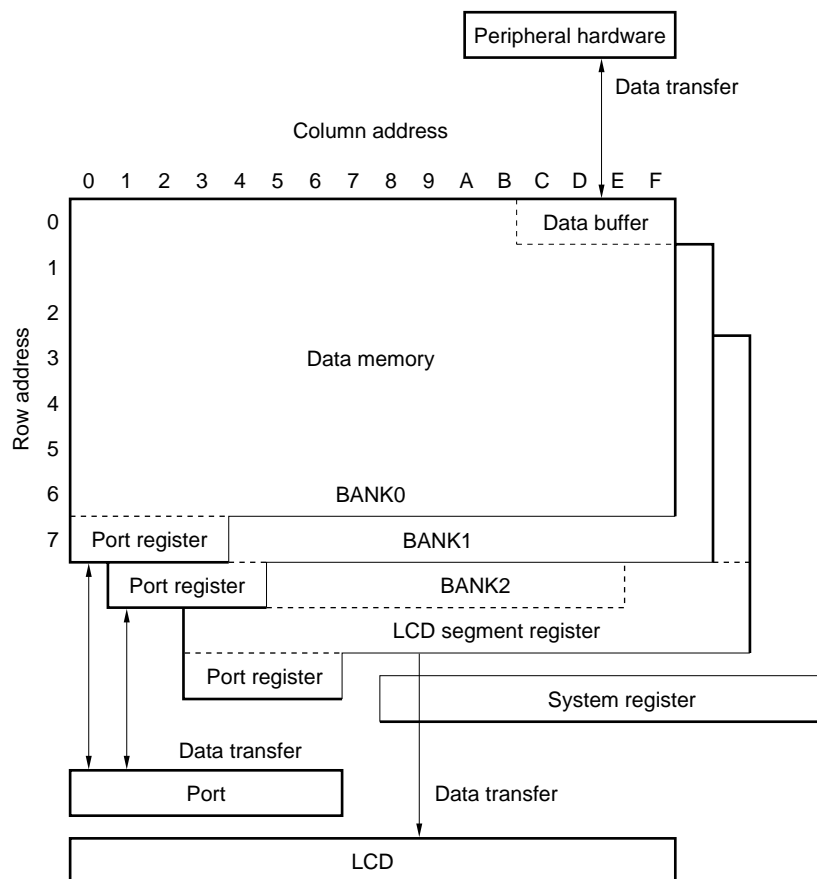
### 4.1 Outline of Data Memory

Figure 4-1 illustrates the data memory.

As shown in this figure, the data memory consists of a general-purpose data memory, system register, data buffer, LCD segment register, and port registers.

The data memory stores data, transfers data with the peripheral hardware units, sets display data, transfers data with the ports, and controls the CPU.

**Figure 4-1. Outline of Data Memory**



## 4.2 Configuration and Function of Data Memory

Figure 4-2 shows the configuration of the data memory.

As shown in the figure, the data memory consists of banks.

Each bank consists of 128 nibbles with 7H row addresses and 0FH column addresses.

The data memory can be divided by classification of function into the blocks explained in 4.2.1 through 4.2.6 below.

The contents of the data memory can be operated, compared, judged, and transferred in 4-bit units by using a data memory manipulation instruction.

Table 4-1 lists the available data memory manipulation instructions.

### 4.2.1 System register (SYSREG)

The system register is allocated to addresses 74H to 7FH.

Because the system register is allocated to every bank, the identical system register exists at addresses 74H to 7FH of any bank.

For details, refer to **5. SYSTEM REGISTER (SYSREG)**.

### 4.2.2 Data buffer (DBF)

The data buffer is allocated to addresses 0CH to 0FH of BANK0.

For details, refer to **9. DATA BUFFER (DBF)**.

### 4.2.3 LCD segment register

The LCD segment register is allocated to addresses 5CH to 6FH of BANK2.

For details, refer to **19. LCD CONTROLLER/DRIVER**.

### 4.2.4 Port registers

The port registers are allocated to addresses 70H to 73H of each bank.

For details, refer to **10. GENERAL-PURPOSE PORTS**.

### 4.2.5 General-purpose data memory

The general-purpose data memory is allocated to the addresses of the data memory excluding those of the system register, LCD segment register, and port registers.

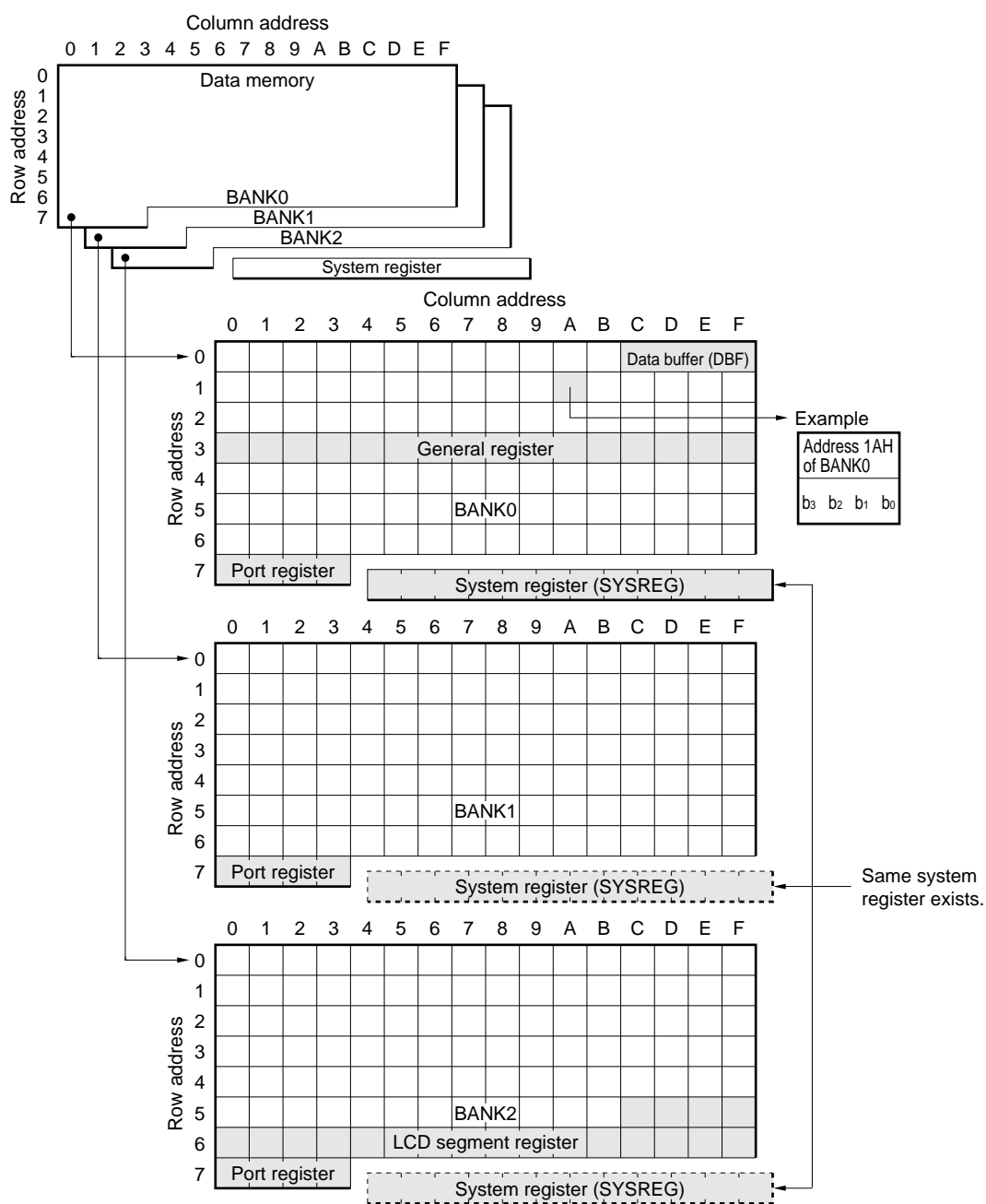
The general-purpose data memory of the μPD17012 consists of a total of 316 nibbles ( $316 \times 4$  bits): 112 nibbles for each of BANK0 and BANK1, and 92 nibbles for BANK2.

### 4.2.6 Unallocated data memory

Data memory areas to which nothing is actually allocated exist in part of the port registers.

For the details of these data memory areas, refer to **4.4.2 Notes on unallocated data memory**, and **10. GENERAL-PURPOSE PORTS**.

Figure 4-2. Configuration of Data Memory



**Table 4-1. Data Memory Manipulation Instructions**

Function		Instruction
Operation	Addition	ADD ADDC
	Subtraction	SUB SUBC
	Logical	AND OR XOR
Compare		SKE SKGE SKLT SKNE
Transfer		MOV LD ST
Judgement		SKT SKF

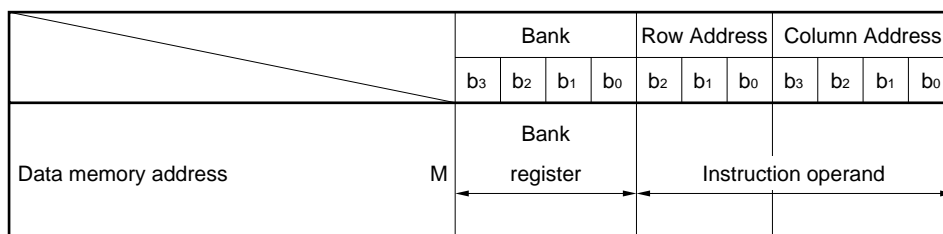
### 4.3 Addressing of Data Memory

Figure 4-3 shows addressing of the data memory.

An address of the data memory is specified by a bank, a row address, and a column address.

The row and column addresses are directly specified by using a data memory manipulation instruction. The bank is specified by the contents of the bank register.

For the details of the bank register, refer to **5. SYSTEM REGISTER (SYSREG)**.

**Figure 4-3. Addressing of Data Memory**



## 4.4 Notes on Using Data Memory

### 4.4.1 On power-on reset

The contents of the general-purpose data memory are undefined on power-on reset.  
Initialize the general-purpose data memory as necessary.

### 4.4.2 Notes on unallocated data memory

If a data memory manipulation instruction is executed to a data memory address to which nothing has been allocated, the following operations are performed.

#### (1) Device operation

If a read instruction is executed, "0" is read.  
Nothing is affected even if a write instruction is executed.

#### (2) Assembler (RA17K) operation

Assembly is performed normally.  
An error does not occur.

#### (3) In-circuit emulator (IE-17K) operation

If a read instruction is executed, "0" is read.  
Nothing is affected even if a write instruction is executed.  
An error does not occur.

## 5. SYSTEM REGISTER (SYSREG)

### 5.1 Outline of System Register

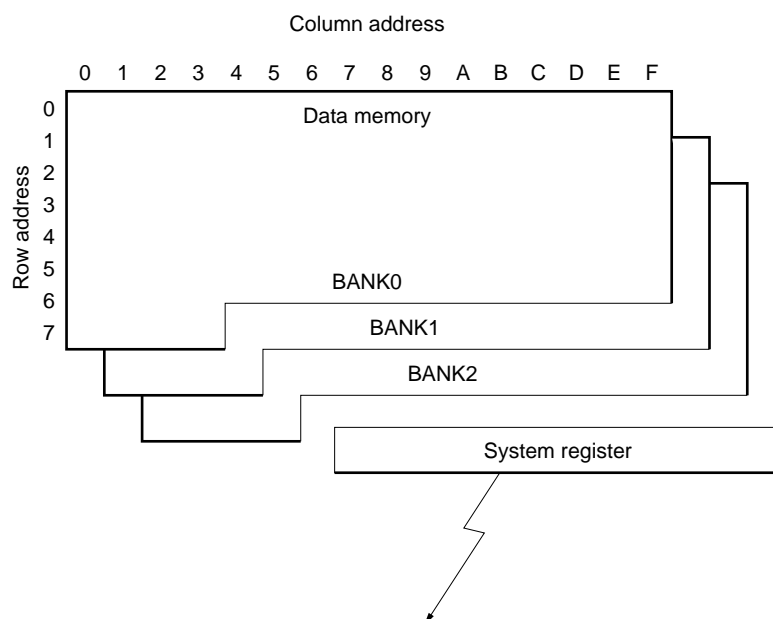
Figure 5-1 shows the location in the data memory and outline of the system register.

As shown in this figure, the system register is allocated to addresses 74H to 7FH of each bank of the data memory. Therefore, an identical system register exists at addresses 74H to 7FH of any bank.

Because the system register is located in the data memory, it can be manipulated by any data memory manipulation instruction.

The system register consists of seven registers.

**Figure 5-1. Location on Data Memory and Outline of System Register**



Address	74H	75H	76H	77H	78H	79H
Name	Address register (AR)				Window register (WR)	Bank register (BANK)
Function	Controls program memory address.				Transfers data with register file.	Specifies bank of data memory.

Address	7AH	7BH	7CH	7DH	7EH	7FH
Name	Index register (IX)  Data memory row address pointer (MP)			General register pointer (RP)		Program status word (PSWORD)
Function	Modifies address of data memory.			Specifies address of general register		Controls operation.

## 5.2 System Register List

Figure 5-2 shows the configuration of the system register.

**Figure 5-2. Configuration of System Register**

Address	74H				75H				76H				77H				78H				79H				
Name	System register																								
	Address register (AR)												Window register (WR)				Bank register (BANK)								
Symbol	AR3				AR2				AR1				AR0				WR				BANK				
Bit	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	
Data	0	0	0	0																		0	0		

Address	7AH				7BH				7CH				7DH				7EH				7FH				
Name	System register																								
	Index register (IX)												General register pointer (RP)								Program status word (PSWORD)				
	Data memory row address pointer (MP)																								
Symbol	IXH				IXM				IXL				RPH				RPL				PSW				
	MPH				MPL																				
Bit	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	
Data	M																				B	C	C	Z	I
	P	0	0					(IX)					0	0			(RP)				C	M	Y		X
	E							(MP)													D	P			E

## 5.3 Address Register (AR)

### 5.3.1 Configuration of address register

Figure 5-3 shows the configuration of the address register.

As shown in this figure, the address register consists of 16 bits, or 74H to 77H (AR3 to AR0) of the system register. However, since the higher 4 bits are always fixed to 0, this register actually operates as a 12-bit register.

**Figure 5-3. Configuration of Address Register**

Address		74H				75H				76H				77H			
Name		Address register (AR)															
Symbol		AR3				AR2				AR1				AR0			
Bit		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data <sup>Note</sup>		⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
0 0 0 0				⌈ M S B ⌋													
After reset	Power-on	0				0				0				0			
	Clock stop	0				0				0				0			
	CE	0				0				0				0			

**Remark** Power-on: Power-on reset  
 Clock stop: Execution of clock stop instruction  
 CE: CE reset

**Note** Bits marked as "0" are fixed to 0.

### 5.3.2 Function of address register

The address register specifies a program memory address when a table reference (“MOVT DBF, @AR”), stack manipulation (“PUSH AR”, “POP AR”), indirect branch (“BR @AR”) or indirect subroutine call (“CALL @AR”) instruction is executed.

A dedicated instruction (“INC AR”) that can increment the value of the address register by one is also available.

The following paragraphs (1) through (5) explain the operations to be performed when the respective instructions are executed.

#### (1) Table reference instruction (“MOVT DBF, @AR”)

This instruction reads the constant data (16-bit) of the program memory address specified by the contents of the address register to the data buffer.

The addresses for storing constant data specified by the address register are 0000H to 0FFFH.

#### (2) Stack manipulation instructions (“PUSH AR”, “POP AR”)

When the “PUSH AR” instruction is executed, the value of the stack pointer is decremented by one, and the contents of the address register (AR) are stored to the address stack register specified by the value of the decremented stack pointer.

When the “POP AR” instruction is executed, the contents of the address stack register specified by the stack pointer are transferred to the address register, and the value of the stack pointer is incremented by one.

#### (3) Indirect branch instruction (“BR @AR”)

This instruction branches execution to the program memory address specified by the contents of the address register.

The branch addresses specified by the address register are 0000H to 0FFFH.

#### (4) Indirect subroutine call instruction (“CALL @AR”)

This instruction calls the subroutine at the program memory address specified by the contents of the address register.

The top address of the subroutine specified by the address register are 0000H to 0FFFH.

#### (5) Address register increment instruction (“INC AR”)

This instruction increments the contents of the address register by one.

Because the address register of the  $\mu$ PD17012 consists of 12 bits, its contents are cleared to “0000H” if the “INC AR” instruction is executed when the contents of the address register are “0FFFH”.

### 5.3.3 Address register and data buffer

The address register can transfer data via the data buffer as part of the peripheral hardware.

For details, refer to **9. DATA BUFFER (DBF)**.

## 5.4 Window Register (WR)

### 5.4.1 Configuration of window register

Figure 5-4 shows the configuration of the window register.

As shown in the figure, the window register consists of 4 bits at address 78H of the system register.

**Figure 5-4. Configuration of Window Register**

Address		78H			
Name		Window register (WR)			
Symbol		WR			
Bit		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data		MSB			LSB
After reset	Power-on	Undefined			
	Clock stop	Holds previous status			
	CE				

### 5.4.2 Function of window register

The window register is used to transfer data with the register file (RF) which is explained later.

To transfer data between the window register and register file, the dedicated instructions “PEEK WR, rf” and “POKE rf, WR” are used (rf: address of register file).

The following paragraphs (1) and (2) explain the operations to be performed when each of these instructions is executed.

Also refer to **8. REGISTER FILE (RF)**.

#### (1) “PEEK WR, rf” instruction

When this instruction is executed, the contents of the register file addressed by “rf” are transferred to the window register.

#### (2) “POKE rf, WR” instruction

When this instruction is executed, the contents of the window register are transferred to the register file addressed by “rf”.

## 5.5 Bank Register (BANK)

### 5.5.1 Configuration of bank register

Figure 5-5 shows the configuration of the bank register.

As shown in the figure, the bank register consists of 4 bits at address 79H (BANK) of the system register. Actually, however, this register is a 2-bit register because the higher 2 bits are always fixed to 0.

**Figure 5-5. Configuration of Bank Register**

Address		79H			
Name		Bank register (BANK)			
Symbol		BANK			
Bit		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data		0	0	$\widehat{\text{MSB}}$	$\widehat{\text{LSB}}$
After reset	Power-on	0			
	Clock stop	0			
	CE	0			

### 5.5.2 Function of bank register

The bank register specifies a bank of the data memory.

Table 5-1 shows the relationship between the value of the bank register and the bank of the data memory specified by each value of the bank register.

Because the bank register exists on the system register, its contents can be rewritten no matter which bank may be currently specified.

In other words, the bank register can be manipulated independently of the current status of the bank.

**Table 5-1. Specifying Bank of Data Memory**

Bank Register (BANK)				Data Memory Bank
b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	
0	0	0	0	BANK0
0	0	0	1	BANK1
0	0	1	0	BANK2
0	0	1	1	Setting prohibited

## 5.6 Index Register (IX) and Data Memory Row Address Pointer (MP: Memory Pointer)

### 5.6.1 Configuration of index register and data memory row address pointer

Figure 5-6 shows the configuration of the index register and data memory row address pointer.

As shown in the figure, the index register consists of an index register (IX) and an index enable flag (IXE). IX is an 11-bit register consisting of the lower 3 bits (IXH) of system register address 7AH, and addresses 7BH and 7CH (IXM and IXL). IXE is the least significant bit of address 7FH (PSW).

The data memory row address pointer (memory pointer) consists of a data memory row address pointer, which consists of 7 bits with the lower 3 bits of address 7AH (MPH) and address 7BH (MPL), and a data memory row address pointer enable flag (memory pointer enable flag: MPE), which is the most significant bit of address 7AH (MPH).

In other words, the higher 7 bits of the index register are shared with the data memory row address pointer.

Note, however, that the higher 2 bits of the index register and data memory row address pointer (bits b<sub>2</sub> and b<sub>1</sub> of address 7AH) are always fixed to 0.

**Figure 5-6. Configuration of Index Register and Data Memory Row Address Pointer**

Address		7AH				7BH				7CH				7EH				7FH			
Name		Index register (IX)												Program status word (PSWORD)							
		Memory pointer (MP)																			
Symbol		IXH				IXM				IXL								PSW			
		MPH				MPL															
Bit		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data		$\widehat{M}$ P E $\widehat{<}$	0	0	$\widehat{M}$ S B $\widehat{<}$								$\widehat{L}$ S B $\widehat{<}$								I X E
After reset	Power-on		0				0				0										0
	Clock stop		0				0				0										0
	CE		0				0				0										0



### 5.6.2 Functions of index register and data memory row address pointer

The index register and data memory row address pointer modify the addresses of the data memory.

The following paragraphs (1) and (2) explain the functions of the index register and data memory row address pointer, respectively.

A dedicated instruction ("INC IX") that can increment the value of the address register by one is also available. For details on address modification, refer to **7. ALU (Arithmetic Logic Unit) BLOCK**.

#### (1) Index register

The index register modifies a specified data memory address according to the contents of the index register when a data memory manipulation instruction is executed.

This modification, however, is valid only when the IXE flag is set to 1.

To modify an address, the bank, row address, and column address of the data memory are ORed with the contents of the index register, and the instruction is executed to the data memory whose address (called an actual address) is specified by the result of this OR operation.

All the data memory manipulation instructions are subject to address modification by the index register.

The following instructions are not subject to modification by the index register.

INC	AR	RORC r
INC	IX	CALL addr
MOVT	DBF, @AR	CALL @AR
PUSH	AR	RET
POP	AR	RETSK
PEEK	WR, rf	RETI
POKE	rf, WR	EI
GET	DBF, p	DI
PUT	p, DBF	STOP s
BR	addr	HALT h
BR	@AR	NOP

#### (2) Data memory row address pointer

The data memory row address pointer modifies the address at the indirect transfer destination when a general register indirect transfer instruction ("MOV @r, m" or "MOV m, @r") is executed.

However, this modification is valid only when the MPE flag is set to "1".

To modify the address, the bank and row address at the transfer destination are replaced with the contents of the data memory row address pointer.

Instructions other than general register indirect transfer instructions are not subject to address modification.

#### (3) Index register increment instruction ("INC IX")

This instruction increments the contents of the index register by one.

Because the index register is configured from 9 bits, the contents of the index register are cleared to "000H" if the "INC IX" instruction is executed when the contents of the index register are "1FFH".

## 5.7 General Register Pointer (RP)

### 5.7.1 Configuration of general register pointer

Figure 5-7 shows the configuration of the general register pointer.

As shown in this figure, the general register pointer consists of 7 bits: 4 bits of address 7DH (RPH) of the system register and the higher 3 bits of address 7EH (RPL). However, because the higher 2 bits of address 7DH are always fixed to 0, actually the lower 5 bits of this register (the lower 2 bits of address 7DH and the higher 3 bits of address 7EH) are valid.

**Figure 5-7. Configuration of General Register Pointer**

Address		7DH				7EH			
Name		General register pointer (RP)							
Symbol		RPH				RPL			
Bit		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data		0	0	M S B <				L S B <	B C D
After reset	Power-on	0				0			
	Clock stop	0				0			
	CE	0				0			

### 5.7.2 Function of general register pointer

The general register pointer specifies a general register in the data memory.

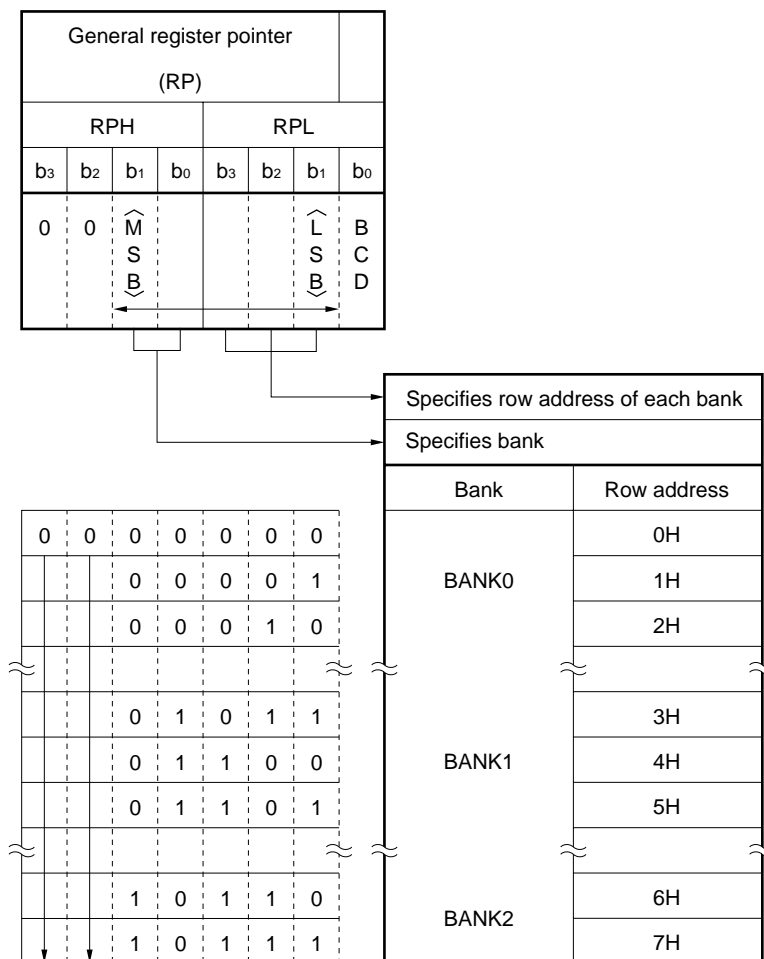
Figure 5-8 shows the address of the general register specified by the general register pointer.

As shown in the figure, the higher 4 bits of the general register pointer (RPH: address 7DH) specify a bank, and the lower 3 bits (RPL: address 7EH) specify a row address.

Because the valid number of bits of the general register pointer is 5, the row addresses (0H to 7H) of BANK0 and BANK1 can be specified as general registers.

For the details on the operations of the general registers, refer to **6. GENERAL REGISTER (GR)**.

**Figure 5-8. Address of General Register Specified by General Register Pointer**



### 5.7.3 Notes on using general register pointer

The least significant bit of address 7EH (RPL) to which the general register pointer is allocated is used as the BCD flag of the program status word.

When rewriting the value of RPL, therefore, pay attention to the value of the BCD flag.

## 5.8 Program Status Word (PSWORD)

### 5.8.1 Configuration of program status word

Figure 5-9 shows the configuration of the program status word.

As shown in the figure, the program status word consists of 5 bits: the least significant bit of address 7EH (RPL) of the system register and the 4 bits of the address 7FH (PSW).

The program status word consists of five flags, each of which functions independently: BCD (BCD), compare (CMP), carry (CY), zero (Z), and index enable (IXE) flags.

**Figure 5-9. Configuration of Program Status Word**

Address		7EH				7FH			
Name		(RP)				Program status word (PSWORD)			
Symbol		RPL				PSW			
Bit		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data					B	C	C	Z	I
					C	M	Y		X
					D	P			E
After reset	Power-on	0				0			
	Clock stop	0				0			
	CE	0				0			

### 5.8.2 Function of program status word

The program status word is a register that sets the condition of an operation or transfer instruction of the ALU (Arithmetic Logic Unit) or indicates the result of an executed operation.

Table 5-2 outlines the function of each flag of the program status word.

For details, refer to 7. **ALU (Arithmetic Logic Unit) BLOCK.**

**Table 5-2. Functional Outline of Program Status Word**

(RP)				Program status word (PSWORD)			
RPL				PSW			
b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
			B	C	C	Z	I
			C	M	Y		X
			D	P			E

Flag Name	Function
Index enable flag (IXE)	This flag modifies the address of the data memory when a data memory manipulation instruction is executed. 0: No modification 1: Modification
Zero flag (Z)	This flag indicates that the result of an arithmetic operation executed is 0. Note that the status of 0 or 1 differs depending on the contents of the compare flag.
Carry flag (CY)	This flag indicates the occurrence of a carry or borrow as a result of execution of an addition or subtraction instruction. It is reset to 0 if a carry/borrow does not occur; it is set to 1 if a carry/borrow occurs. This flag is also used as the shift bit of the "RORC r" instruction.
Compare flag (CMP)	This flag is used avoid storing the result of an arithmetic operation to the data memory or general register. 0: Result stored 1: Result not stored
BCD flag (BCD)	This flag is used to execute an arithmetic operation in decimal. 0: Binary operation executed 1: Decimal operation executed

### 5.8.3 Notes on using program status word

If an arithmetic operation (addition or subtraction) instruction is executed for the program status word, the result of the arithmetic operation is stored in the program status word.

For example, even if an operation that causes a carry to occur is executed, if the result of the operation is 0000B, 0000B is stored in the PSW.

### 5.9 Notes on Using System Register

The data of the system register fixed to 0 is not affected even if a write instruction is executed to it.  
This data is always 0 when it is read.

## 6. GENERAL REGISTER (GR)

### 6.1 Outline of General Register

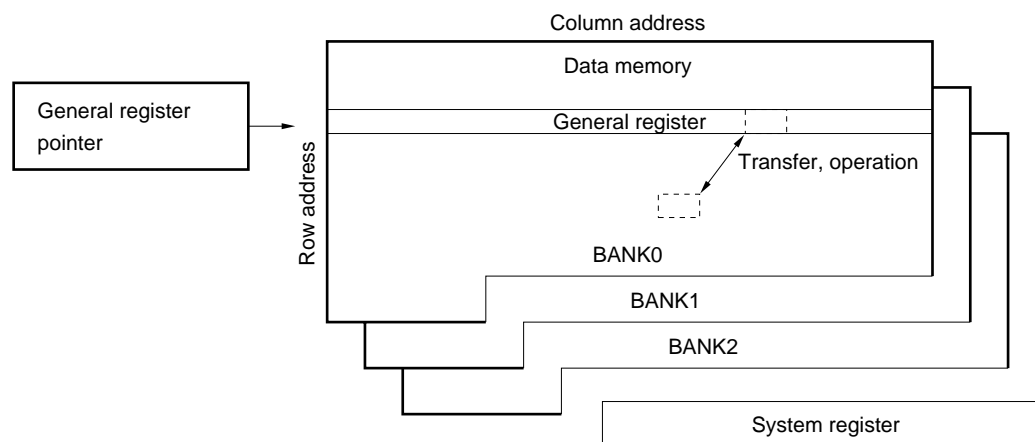
Figure 6-1 illustrates the general register.

As shown in the figure, the general register consists of a general register pointer and general register body.

The bank and row address of the general register body are specified by the general register pointer.

The general register body is used to transfer data and execute operations between data memory addresses.

**Figure 6-1. Outline of General Register**



### 6.2 General Register Body

The general register body consists of 16 nibbles (16 x 4 bits) at the same row addresses in the data memory.

For the range of the banks and row addresses that can be specified by the general register pointer and general register, refer to **5.7 General Register Pointer (RP)**.

The 16 nibbles of the same row address specified as a general register executes operations and transfers data with the data memory using a single instruction.

In other words, operations or transfer between data memory addresses can be executed with a single instruction.

The general register can be controlled by a data memory manipulation instruction like the other data memory areas.

### 6.3 Address Generation of General Register by Instructions

The following subsections 6.3.1 and 6.3.2 explain how the addresses of the general register are generated when each instruction is executed.

For the details of the operation of each instruction, refer to **7. ALU (Arithmetic Logic Unit) BLOCK**.

#### 6.3.1 Addition (“ADD r, m”, “ADDC r, m”), subtraction (“SUB r, m”, “SUBC r, m”), logical operation (“AND r, m”, “OR r, m”, “XOR r, m”), direct transfer (“LD r, m”, “ST m, r”), and rotation processing (“RORC r”) instructions

Table 6-1 shows the address of general register “R” specified by operand “r” of an instruction. Only the column address is specified as operand “r”.

**Table 6-1. Address Generation of General Register**

		Bank				Row Address			Column Address			
		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
General register address	R	Contents of general register pointer							r			

#### 6.3.2 Indirect transfer (“MOV @r,m”, “MOV m, @r”) instructions

Table 6-2 shows the address of the general register “R” specified by operand “r” of an instruction and an indirect transfer address specified by “@R”.

**Table 6-2. Address Generation of General Register**

		Bank				Row Address			Column Address			
		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
General register address	R	Contents of general register pointer							r			
Indirect transfer address	@R					Same as data memory			Contents of R			

## 6.4 Notes on Using General Register

### 6.4.1 Row address of general register

Note that because the row address of the general register is specified by the general register pointer, the bank currently specified may differ from the bank of the general register.

### 6.4.2 Operation between general register and immediate data

No instruction that executes an operation between the general register and immediate data is provided.

To execute an operation between the general register and immediate data, the general register must be treated as a data memory area.



## 7. ALU (Arithmetic Logic Unit) BLOCK

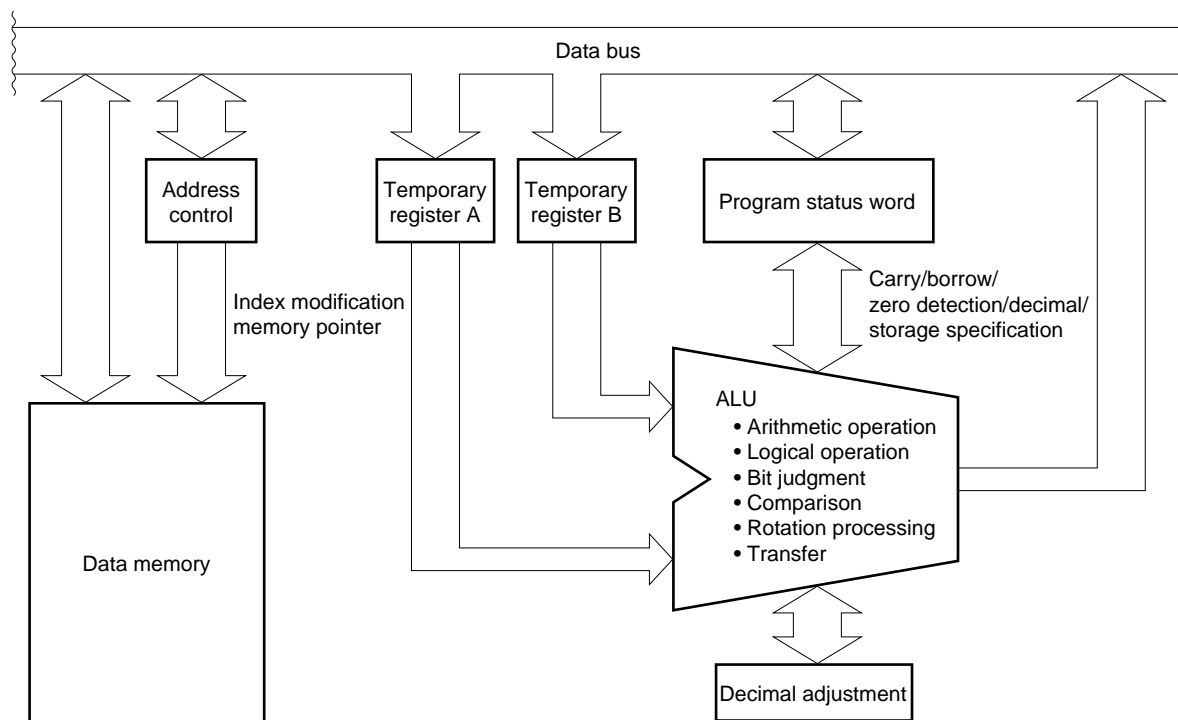
### 7.1 Outline of ALU Block

Figure 7-1 outlines the ALU block.

As shown in the figure, the ALU block consists of an ALU, temporary registers A and B, a program status word, decimal adjuster, and data memory address controller.

The ALU operates, judges, compares, rotates, and transfers 4-bit data in the data memory.

Figure 7-1. Outline of ALU Block



## 7.2 Configuration and Function of Each Block

### 7.2.1 ALU

The ALU executes arithmetic or logical operations, bit judgment, comparison, rotation processing, and transfer of 4-bit data according to an instruction specified by the program.

### 7.2.2 Temporary registers A and B

Temporary registers A and B temporarily store 4-bit data.

These registers are automatically used when an instruction is executed and are not controlled by the program.

### 7.2.3 Program status word

The program status word controls the operation of the ALU and stores the status of the ALU.

For the details of the program status word, refer to **5.8 Program Status Word (PSWORD)**.

### 7.2.4 Decimal adjuster

If the BCD flag of the program status word is set to 1 as a result of an executed arithmetic operation, the arithmetic operation result is converted into a decimal number by the decimal adjuster.

### 7.2.5 Address controller

The address controller specifies an address of the data memory.

At this time, address modification by the index register and data memory row address pointer is also controlled.

## 7.3 ALU Processing Instruction List

Table 7-1 lists the ALU operations when each instruction is executed.

Table 7-2 shows modification of data memory addresses by the index register and data memory row address pointer.

Table 7-3 shows the decimal adjustment data when a decimal operation is executed.

Table 7-1. List of ALU Processing Instruction Operations

ALU Function	Instruction		Difference in Operation Based on Program Status Word (PSWORD)					Address Modification	
			Value of BCD Flag	Value of CMP Flag	Arithmetic Operation	Operation of CY Flag	Operation of Z Flag	Index	Memory Pointer
Addition	ADD	r, m	0	0	Stores result of binary operation.	Set if carry or borrow occurs; otherwise, reset.	Set if result of operation is 0000B; otherwise, reset.	Executed	Not executed
		m, #n4							
	ADDC	r, m	0	1	Does not store result of binary operation.		Holds status if result of operation is 0000B; otherwise, reset.		
		m, #n4							
Subtrac- tion	SUB	r, m	1	0	Stores result of decimal operation.		Set if result of operation is 0000B; otherwise, reset.		
		m, #n4							
	SUBC	r, m	1	1	Does not store result of decimal operation.		Holds status if result of operation is 0000B; otherwise, reset.		
		m, #n4							
Logical operation	OR	r, m	Any (held)	Any (held)	Not affected	Holds previous status.	Holds previous status.	Executed	Not executed
		m, #n4							
	AND	r, m							
		m, #n4							
	XOR	r, m							
		m, #n4							
Judgment	SKT	m, #n	Any (held)	Any (reset)	Not affected	Holds previous status.	Holds previous status.	Executed	Not executed
	SKF	m, #n							
Compare	SKE	m, #n4	Any (held)	Any (held)	Not affected	Holds previous status.	Holds previous status.	Executed	Not executed
	SKNE	m, #n4							
	SKGE	m, #n4							
	SKLT	m, #n4							
Transfer	LD	r, m	Any (held)	Any (held)	Not affected	Holds previous status	Holds previous status	Executed	Not executed
	ST	m, r							
	MOV	m, #n4							
		@r, m							Executed
		m, @r							
Rotation	RORC	r	Any (held)	Any (held)	Not affected	Value of general register bo	Holds previous value	Not executed	Not executed

**Table 7-2. Modification of Data Memory Address and Modification of Indirect Transfer Address by Index Register and Data Memory Row Address Pointer**

IXE	MPE	General Register Address Specified by r												Data Memory Address Specified by m												Indirect Transfer Address Specified by @r											
		Bank				Row Address				Column Address				Bank				Row Address				Column Address				Bank				Row Address				Column Address			
		b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>			
0	0	RP				r				BANK				m				BANK				m <sub>R</sub>				(r)											
0	1					ditto								ditto								MP				(r)											
1	0					ditto				BANK				m				BANK				m <sub>R</sub>				Logical IXH, IXM				OR (r)							
1	1					ditto								ditto								MP				(r)											

BANK: Bank register

IX: Index register

IXE: Index enable flag

IXH: Bits 10 through 8 of index register

IXM: Bits 7 through 4 of index register

IXL: Bits 3 through 0 of index register

m: Data memory address indicated by m<sub>R</sub>, m<sub>C</sub>

m<sub>R</sub>: Data memory row address (higher)

m<sub>C</sub>: Data memory column address (lower)

MP: Data memory row address pointer

MPE: Memory pointer enable flag

r: General register column address

RP: General register pointer

(x): Contents addressed by x

x: Direct address such as m and r

Table 7-3. Decimal Adjustment Data

Operation Result	Hexadecimal Addition		Decimal Addition	
	CY	Operation Result	CY	Operation Result
0	0	0000B	0	0000B
1	0	0001B	0	0001B
2	0	0010B	0	0010B
3	0	0011B	0	0011B
4	0	0100B	0	0100B
5	0	0101B	0	0101B
6	0	0110B	0	0110B
7	0	0111B	0	0111B
8	0	1000B	0	1000B
9	0	1001B	0	1001B
10	0	1010B	1	0000B
11	0	1011B	1	0001B
12	0	1100B	1	0010B
13	0	1101B	1	0011B
14	0	1110B	1	0100B
15	0	1111B	1	0101B
16	1	0000B	1	0110B
17	1	0001B	1	0111B
18	1	0010B	1	1000B
19	1	0011B	1	1001B
20	1	0100B	1	1110B <sup>Note</sup>
21	1	0101B	1	1111B <sup>Note</sup>
22	1	0110B	1	1100B <sup>Note</sup>
23	1	0111B	1	1101B <sup>Note</sup>
24	1	1000B	1	1110B <sup>Note</sup>
25	1	1001B	1	1111B <sup>Note</sup>
26	1	1010B	1	1100B <sup>Note</sup>
27	1	1011B	1	1101B <sup>Note</sup>
28	1	1100B	1	1010B <sup>Note</sup>
29	1	1101B	1	1011B <sup>Note</sup>
30	1	1110B	1	1100B <sup>Note</sup>
31	1	1111B	1	1101B <sup>Note</sup>

Operation Result	Hexadecimal Subtraction		Decimal Subtraction	
	CY	Operation Result	CY	Operation Result
0	0	0000B	0	0000B
1	0	0001B	0	0001B
2	0	0010B	0	0010B
3	0	0011B	0	0011B
4	0	0100B	0	0100B
5	0	0101B	0	0101B
6	0	0110B	0	0110B
7	0	0111B	0	0111B
8	0	1000B	0	1000B
9	0	1001B	0	1001B
10	0	1010B	1	1100B <sup>Note</sup>
11	0	1011B	1	1101B <sup>Note</sup>
12	0	1100B	1	1110B <sup>Note</sup>
13	0	1101B	1	1111B <sup>Note</sup>
14	0	1110B	1	1100B <sup>Note</sup>
15	0	1111B	1	1101B <sup>Note</sup>
-16	1	0000B	1	1110B <sup>Note</sup>
-15	1	0001B	1	1111B <sup>Note</sup>
-14	1	0010B	1	1100B <sup>Note</sup>
-13	1	0011B	1	1101B <sup>Note</sup>
-12	1	0100B	1	1110B <sup>Note</sup>
-11	1	0101B	1	1111B <sup>Note</sup>
-10	1	0110B	1	0000B
-9	1	0111B	1	0001B
-8	1	1000B	1	0010B
-7	1	1001B	1	0011B
-6	1	1010B	1	0100B
-5	1	1011B	1	0101B
-4	1	1100B	1	0110B
-3	1	1101B	1	0111B
-2	1	1110B	1	1000B
-1	1	1111B	1	1001B

**Note** The operation results are not correctly adjusted by the decimal adjustment circuit.

## 7.4 Notes on Using ALU

### 7.4.1 Notes on using operations for program status word

If an arithmetic operation is executed for the program status word, the result of the arithmetic operation is stored in the program status word.

The CY and Z flags of the program status word are set or reset depending on the result of the arithmetic operation. If an arithmetic operation is executed on the program status word itself, the result of the operation is stored in the program status word, which makes it impossible to judge occurrence of a carry or a borrow, or whether the result of the operation is zero.

If the CMP flag is set, however, the result of the operation is not stored in the program status word, and the CY and Z flags are set or reset normally.

### 7.4.2 Notes on using decimal operations

A decimal operation can be executed only if the result falls within the following range:

- (1) Result of addition: 0 to 19 in decimal
- (2) Result of subtraction: 0 to 9 or -10 to -1 in decimal

If a decimal operation is executed exceeding this range, the CY flag is set, and the result is a value greater than 1010B (0AH).

## 8. REGISTER FILE (RF)

### 8.1 Outline of Register File

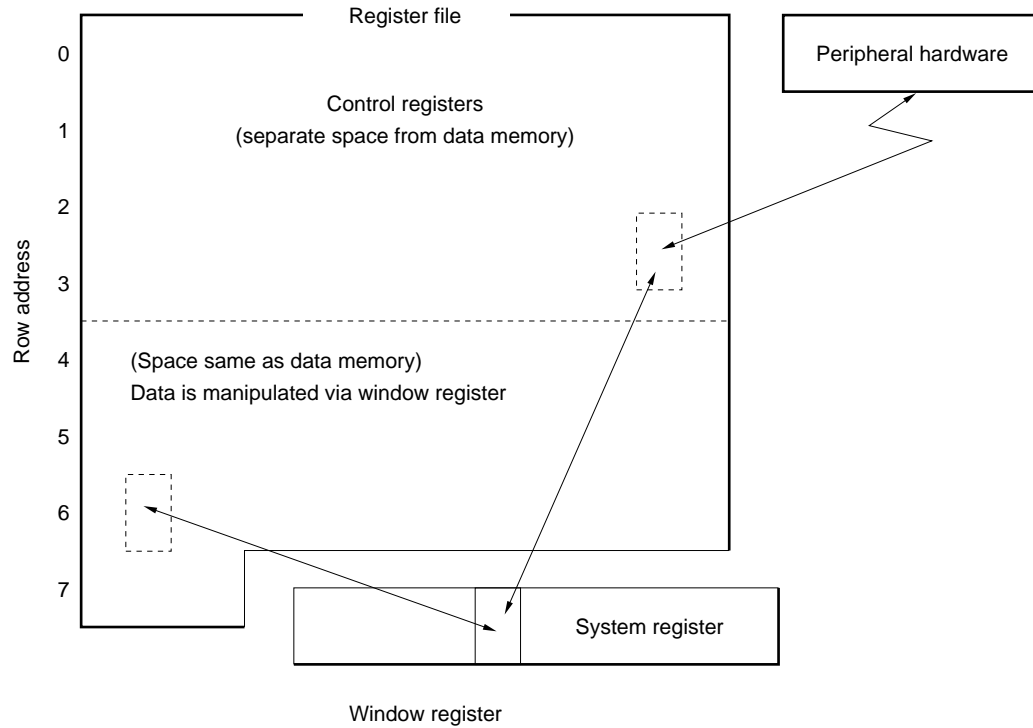
Figure 8-1 illustrates the register file.

As shown in the figure, the register file consists of control registers existing on a space different from that of the data memory, and a portion overlapping the data memory.

The control registers set the conditions of the peripheral hardware units.

Data is read from or written to the register file via the window register.

**Figure 8-1. Outline of Register File**



## 8.2 Configuration and Function of Register File

Figure 8-2 shows the configuration of the register file and its relationship with the data memory.

Addresses are allocated to the register file in 4-bit units, like the data memory, and the register file has a total of 128 nibbles with row addresses 0H to 7H and column addresses 0H to 0FH.

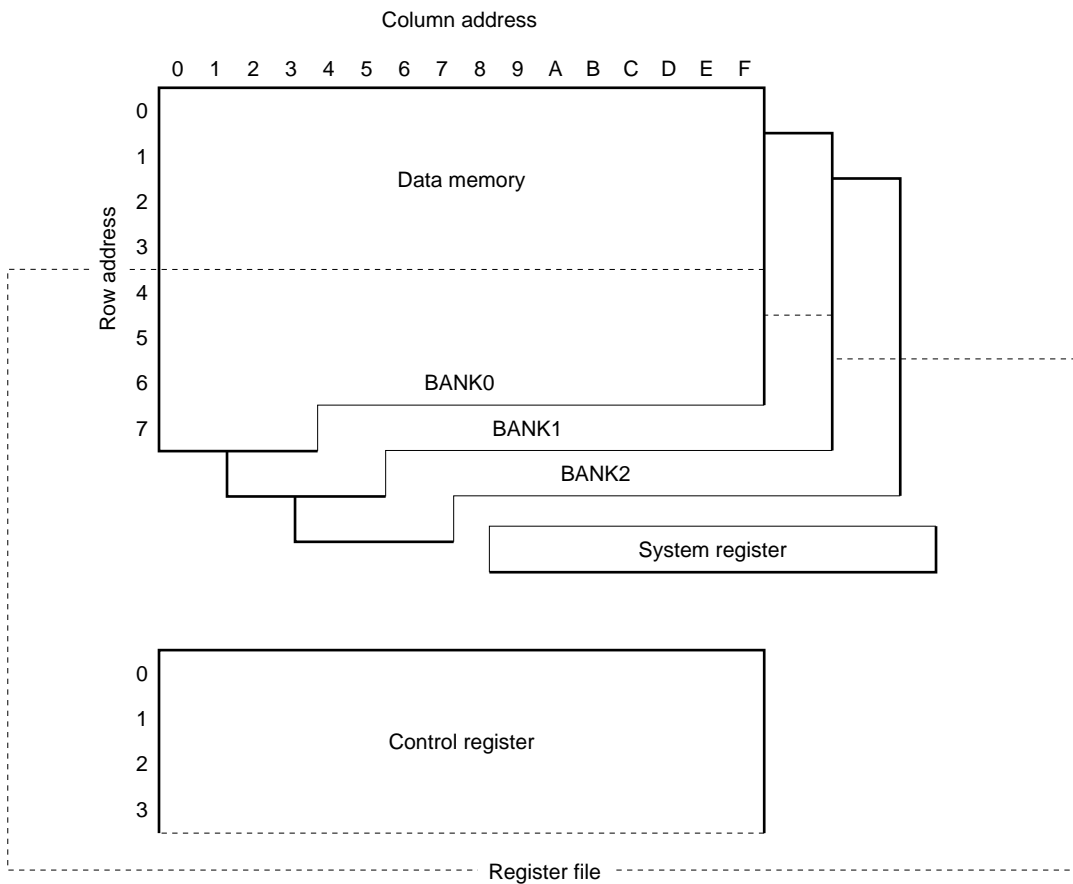
Control registers that set the conditions of the peripheral hardware units are allocated to addresses 00H to 3FH.

Addresses 40H to 7FH overlap the data memory.

To put it another way, the addresses 40H to 7FH of the register file are the memory addresses of the data memory bank currently selected.

These addresses, 40H to 7FH, can be treated in the same manner as the normal data memory areas, except that they can be manipulated by a register file manipulation instruction ("PEEK WR, rf" or "POKE rf, WR"), because they overlap the data memory.

**Figure 8-2. Configuration of Register File and Its Relationship with Data Memory**





### 8.3 Register File Manipulation Instructions (“PEEK WR, rf” and “POKE rf, WR”)

Data is read from or written to the register file via the window register in the system register by using a register file manipulation instruction (“PEEK WR, rf” or “POKE rf, WR”). The operation of each instruction is explained below.

**(1) “PEEK WR, rf”**

This instruction reads the data of the register file addressed by “rf” to the window register.

**(2) “POKE rf, WR”**

This instruction writes the data of the window register to the register file addressed by “rf”.

### 8.4 Control Registers

Figure 8-3 shows the configuration of the control registers.

As shown in this figure, a total of 64 nibbles (64 words  $\times$  4 bits) at addresses 00H to 3FH of the register file can be used as control registers.

Of these nibbles, however, 33 nibbles are actually used. The remaining 31 nibbles are unused registers that are prohibited from being read or written.

Each control register has an attribute of 1 nibble, and is classified into four types: read/write (R/W), read-only (R), write-only (W), and read-and-reset (R & Reset).

Nothing is changed even if data is written to a read-only (R and R & Reset) register.

An undefined value is read if a write-only (W) register is read.

Of the 4-bit data in 1 nibble, the bit fixed to 0 is always 0 when it is read or written.

The 31 nibbles of unused registers are undefined when they are read, and nothing is changed when data is written to them.

Figure 8-3. Configuration of Control Registers (1/2)

Column	Address	0	1	2	3	4	5	6	7
Row Address	Item								
0 (8) <sup>Note</sup>	Name		Stack pointer (SP)	Serial I/O mode select register		I/F count gate judge register	PLL unlock FF judge register	A/D converter compare judge register	CE pin level judge register
	Symbol		0 (SP2) (SP1) (SP0)	SIO1TS SIO1HZ SIO1CK1 SIO1CK0		0 0 0 IFCG	0 0 0 PLLUL	0 0 0 ADCOMP	0 0 0 CE
	Read/Write		R/W	R/W		R	R & Reset	R	R
1 (9) <sup>Note</sup>	Name	LCD mode select register	LCD port select register	IF counter mode select register	PWM mode select register	A/D converter channel select register	BEEP select register	Key input judge register	Basic timer 0 carry FF judge register
	Symbol	0 KSEN LCDEN PYASEL P2HSEL P2GSEL P2FSEL P2ESEL IFCMD1 IFCMD0 IFCK1 IFCK0			0 0 PWM1SEL PWMSEL	0 0 ADCCH1 ADCCH0	0 0 BEEP1SEL BEEPSEL	0 0 0 KEYJ	0 0 0 BTMOCY
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R & Reset	R & Reset
2 (A)	Name		PLL mode select register		IF counter control register	FCG channel select register	BEEP clock select register		Port 1D group I/O select register
	Symbol		0 0 PLLMD1 PLLMD0		0 0 IFCTR1 IFCTRS	0 0 FCGCH1 FCGCH0	BEEP1CK0 BEEP1CK1 BEEP0CK1 BEEP0CK0		0 0 0 P1DGIO
	Read/Write		R/W		W	R/W	R/W		R/W
3 (B) <sup>Note</sup>	Name		PLL reference clock select register				Port 1A bit I/O select register	Port 0B bit I/O select register	Port 0A bit I/O select register
	Symbol		PLLRCK3 PLLRCK2 PLLRCK1 PLLRCK0				0 P1ABIO2 P1ABIO1 P1ABIO0 P0BBIO3 P0BBIO2 P0BBIO1 P0BBIO0		0 P0ABIO2 P0ABIO1 P0ABIO0
	Read/Write		R/W				R/W	R/W	R/W

**Note** Addresses in parentheses are for when an assembler (RA17K) is used.

**Figure 8-3. Configuration of Control Registers (2/2)**

8				9				A				B				C				D				E				F							
				Basic timer clock select register												12-bit timer clock select register				12-bit timer overflow register				12-bit timer control register											
				B T M 1 C K 1	B T M 1 C K 0	B T M 0 C K 1	B T M 0 C K 0									0	0	0	T M C K	0	0	0	T M O V F	0	T M R P T	T M R E S	T M E N								
				R/W												R/W				R				R/W											
																												Interrupt edge select register							
																													0	0	0	I E G			
																												R/W							
																												Interrupt enable register							
																													I P S I O 1	I P B T M 1	I P T M	I P			
																												R/W							
																Interrupt request register 4				Interrupt request register 3				Interrupt request register 2				Interrupt request register 1							
																	0	0	0	I R Q S I O 1	0	0	0	I R Q B T M 1	0	0	0	I R Q T M	I N T	0	0	I R Q			
																R/W				R/W				R/W				R				R/W			

Table 8-1. Peripheral Hardware Control Functions of Control Registers (1/4)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function			After Reset						
	Name	Address	Read/ Write	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	Functional Outline	Set Value		Power-on	S T O P	C E				
						0	1							
General-purpose ports	Port 1D group I/O select register	27H	R/W	0	Fixed to 0			0	0	0				
				0										
				0										
				P1DGIO	I/O setting of port 1D (Group I/O)	Input	Output							
	Port 1A bit I/O select register	35H	R/W	0	Fixed to 0									
				P1ABIO2							<div><div></div><div>P1A<sub>2</sub> pin P1A<sub>1</sub> pin P1A<sub>0</sub> pin P0B<sub>3</sub> pin P0B<sub>2</sub> pin P0B<sub>1</sub> pin P0B<sub>0</sub> pin</div><div>I/O setting (bit I/O)</div></div>	Input	Output	
				P1ABIO1										
				P1ABIO0										
	Port 0B bit I/O select register	36H	R/W	P0BBIO3	Fixed to 0									
				P0BBIO2										
				P0BBIO1										
				P0BBIO0										
	Port 0A bit I/O select register	37H	R/W	0	Fixed to 0									
				P0ABIO2								<div><div></div><div>P0A<sub>2</sub> pin P0A<sub>1</sub> pin P0A<sub>0</sub> pin</div><div>I/O setting (bit I/O)</div></div>	Input	Output
				P0ABIO1										
				P0ABIO0										
Interrupts	Interrupt edge select register	1FH	R/W	0	Fixed to 0			0	0	0				
				0										
				0										
				IEG	Sets interrupt issuance edge (INT)	Rising edge	Falling edge							
	Interrupt enable register	2FH	R/W	IPSIO1	<div><div></div><div>Serial interface Basic timer 1 12-bit timer INT pin</div><div>Enables interrupt</div></div>	Disables interrupt	Enables Interrupt	0	0	0				
				IPBTM1										
				IPTM										
				IP										
	Interrupt request register 4	3CH	R/W	0	Fixed to 0			0	0	0				
				0										
				0										
				IRQSIO1	Detects interrupt request (serial interface)	Not requested	Requested							
	Interrupt request register 3	3DH	R/W	0	Fixed to 0			0	0	0				
				0										
				0										
				IRQBTM1	Detects interrupt request (basic timer 1)	Not requested	Requested							
Interrupt request register 2	3EH	R/W	0	Fixed to 0			0	0	0					
			0											
			0											
			IRQTM	Detects interrupt request (12-bit timer)	Not requested	Requested								

Table 8-1. Peripheral Hardware Control Functions of Control Registers (2/4)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				After Reset			
	Name	Address	Read/ Write	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	Functional Outline	Set Value		Power-on	S T O P	C E		
						0	1					
Interrupts	Interrupt request register 1	3FH	R	INT	Detects status of INT pin	Low level		High level		0	0	0
			R/W	0	Fixed to 0							
				0								
				IRQ	Detects interrupt request (INT pin)	Not requested		Requested				
Timers	Basic timer clock select register	09H	R/W	BTM1CK1	Sets clock of basic timer 1	0 0 1 1		100 ms 250 ms 5 ms 1 ms		0	0	Retained
				BTM1CK0		0 1		0 1				
				BTM0CK1	Sets clock of basic timer 0	0 0 1 1		100 ms 250 ms 5 ms 1 ms				
				BTM0CK0		0 1		0 1				
	12-bit timer clock select register	0CH	R/W	0	Fixed to 0					0	0	Retained
				0								
				0								
				TMCK	Sets clock of 12-bit timer	50 μs		10 μs				
	12-bit timer overflow register	0DH	R	0	Fixed to 0					0	0	Retained
				0								
				0								
				TMOVF	Detects overflow of timer/counter	No overflow		Overflow				
	12-bit timer control register	0EH	R/W	0	Fixed to 0					0	0	Retained
				TMRPT	Selects operation mode of 12-bit timer	Free-run count mode		Modulo count mode				
				TMRES	Resets timer/counter	Does not reset		Resets				
				TMEN	Sets operation of timer/counter	Does not operate		Operates				
	Basic timer 0 carry FF judge register	17H	R & Reset	0	Fixed to 0					0	1	1
				0								
				0								
				BTM0CY	Detects status of carry FF	Reset		Set				
A/D converter	A/D converter compare judge register	06H	R	0	Fixed to 0					Undefined	Retained	Retained
				0								
				0								
				ADCCMP	Detects comparison result	V <sub>ADCI</sub> < V <sub>REF</sub>		V <sub>ADCI</sub> > V <sub>REF</sub>				
A/D converter	A/D converter channel select register	14H	R/W	0	Fixed to 0					3	3	3
				0								
				ADCC1	Selects pins to be used for A/D converter	0	0	1	1			
				ADCC0		ADC <sub>0</sub>	ADC <sub>1</sub>	Not used	Not used			
D/A converter	PWM mode select register	13H	R/W	0	Fixed to 0					0	0	Retained
				0								
				PWM1SEL	PWM <sub>1</sub> pin } Set for D/A converter	General-purpose output port		D/A converter				
				PWM0SEL								

Table 8-1. Peripheral Hardware Control Functions of Control Registers (3/4)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				After Reset			
	Name	Address	Read/ Write	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	Functional Outline	Set Value		Power-on	S T O P	C E		
						0	1					
Serial interface	Serial I/O mode select register	02H	R/W	SIO1TS	Starts/stops operation	Stops operation		0	0	0		
				SIO1HIZ	Sets SO <sub>1</sub> pin as serial output pin	General-purpose I/O port						
				SIO1CK1	Sets clock of serial interface	0	1					
				SIO1CK0		External clock	0				1	
PLL frequency synthesizer	PLL unlock FF judge register	05H	R&Reset	0	Fixed to 0			Undefined	Retained	Retained		
				0								
				0								
				PLLUL	Detects status of unlock FF	Lock status	Unlock status					
	PLL mode select register	21H	R/W	0	Fixed to 0			0	0	Retained		
				0								
				PLLMD1							Sets division method of PLL	0
				PLLMD0	Disable	MF	VHF					HF
	PLL reference clock select register	31H	R/W	PLLRFCK3	Sets reference frequency of PLL	0: 1.25 kHz, 1: 2.5 kHz, 2: 5 kHz, 3: 10 kHz, 4: 6.25 kHz, 5: 12.5 kHz, 6: 25 kHz, 7: 50 kHz, 8: 3 kHz, 9, A, B: Setting prohibited, C: 1 kHz, D: 9 kHz, E: 100 kHz, F: Off		F	F	Retained		
				PLLRFCK2								
				PLLRFCK1								
				PLLRFCK0								
Frequency counter	IF counter gate judge register	04H	R	0	Fixed to 0			0	0	Retained		
				0								
				0								
				IFCG	Detects opening/closing of gate of frequency counter	Close	Open					
	IF counter mode select register	12H	R/W	IFCMD1	Sets mode of frequency counter	0	1	0	0	Retained		
				IFCMD0		FCG	AMIFC pin				FMIFC pin	
				IFCCK1	Sets gate time of frequency counter	0	1				8 ms	Open
				IFCCK0		1 ms	4 ms					
	IF counter control register	23H	W	0	Fixed to 0			0	0	Retained		
				0								
				IFCSTRT							Does not start	Starts
				IFCRES	Resets IF counter	Does not reset	Resets					
FCG channel select register	24H	R/W	0	Fixed to 0			3	3	Retained			
			0									
			FCGCH1							0	1	
			FCGCH0	Sets pin to be used as FCG	FCG <sub>0</sub>	FCG <sub>1</sub>				Not used	Not used	
BEEP	BEEP select register	15H	R/W	0	Fixed to 0			0	0	0		
				0								
				BEEP1SEL	BEEP <sub>1</sub> pin	General-purpose I/O port	BEEP					
				BEEP0SEL	BEEP <sub>0</sub> pin							

Table 8-1. Peripheral Hardware Control Functions of Control Registers (4/4)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				After Reset							
	Name	Address	Read/ Write	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	Functional Outline	Set Value		Power-on	S T O P	C E						
						0	1									
BEEP	BEEP clock select register	25H	R/W	BEEP1CK1	Sets output frequency of BEEP <sub>1</sub>	0	0	1	1	0	0	Retained				
				1 kHz		3 kHz	200 Hz	9 kHz								
				BEEP1CK0	0	1	0	1	Sets output frequency of BEEP <sub>0</sub>				0	0	1	1
				BEEP0CK1	0	0	1	1					1 kHz	3 kHz	200 Hz	9 kHz
BEEP0CK0	0	1	0	1												
LCD controller/driver	LCD mode select register	10H	R/W	0	Fixed to 0					0	0	Retained				
				KSEN	Sets key source output signal	Key source off	Key source on									
				LCDEN	Sets LCD display output	Display off	Display on									
				PYASEL												
	LCD port select register	11H	R/W	P2HSEL	PYA <sub>0</sub> -PYA <sub>15</sub> pins P2H <sub>0</sub> pin P2G <sub>0</sub> pin P2F <sub>0</sub> pin P2E <sub>0</sub> pin	Set as general-purpose output port	LCD segment	General-purpose output port								
				P2GSEL												
				P2FSEL												
				P2ESEL												
	Key input judge register	16H	R&Reset	0	Fixed to 0					0	0	0				
				0												
0																
KEYJ				Detects if key input latch is valid	Latch invalid	Latch valid										
Standby	CE pin level judge register	07H	R	0	Fixed to 0					-	-	-				
				0												
				0												
				CE	Detects status of CE pin	Low level	High level									

**Remark** —: Determined according to the status of the pin.

## 8.5 Notes on Using Register File

Note the following points (1) through (3) when manipulating the write-only registers (W), read-only registers (R), and unused registers of the control registers (addresses 00H to 3FH of the register file).

- (1) When a write-only register is read, an undefined value is read.
- (2) Nothing is changed even if data is written to a read-only register.
- (3) An undefined value is read if an unused register is read. Nothing is changed even if data is written to an unused register.

## 9. DATA BUFFER (DBF)

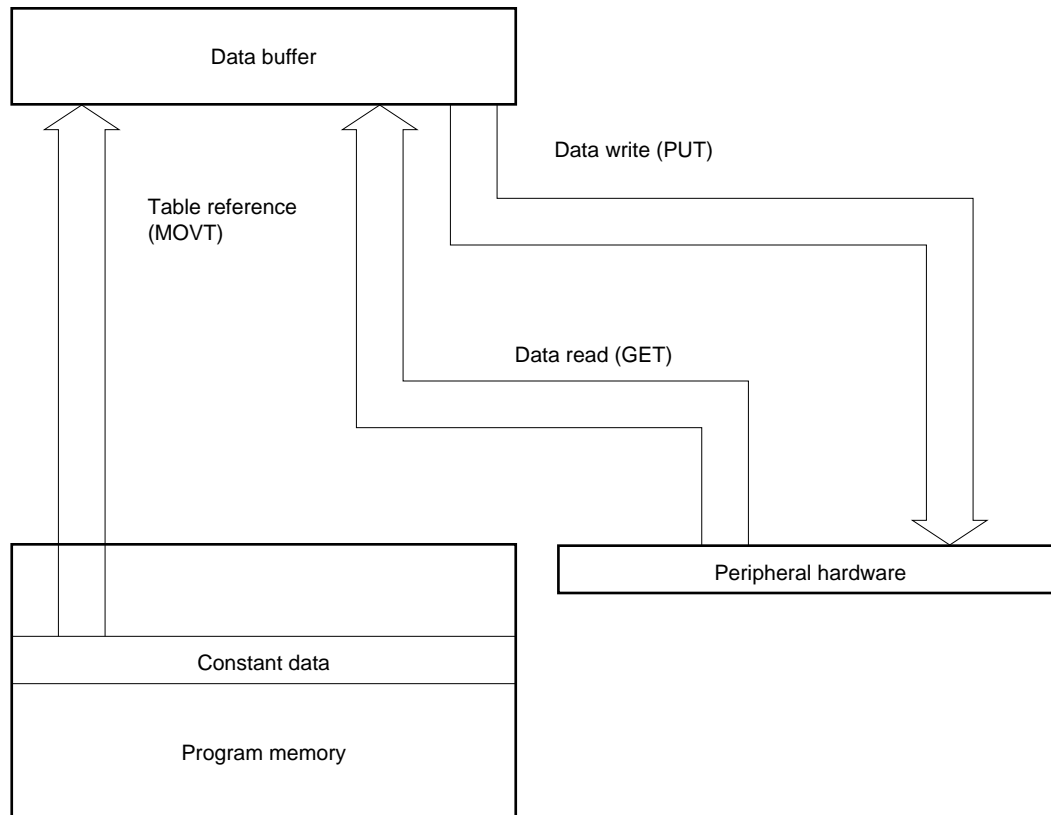
### 9.1 Outline of Data Buffer

Figure 9-1 illustrates the data buffer.

The data buffer is located in the data memory and has the following two functions:

- (1) Reads constant data from program memory (table reference)
- (2) Transfers data with peripheral hardware unit

**Figure 9-1. Outline of Data Buffer**





## 9.2 Data Buffer

### 9.2.1 Configuration of data buffer

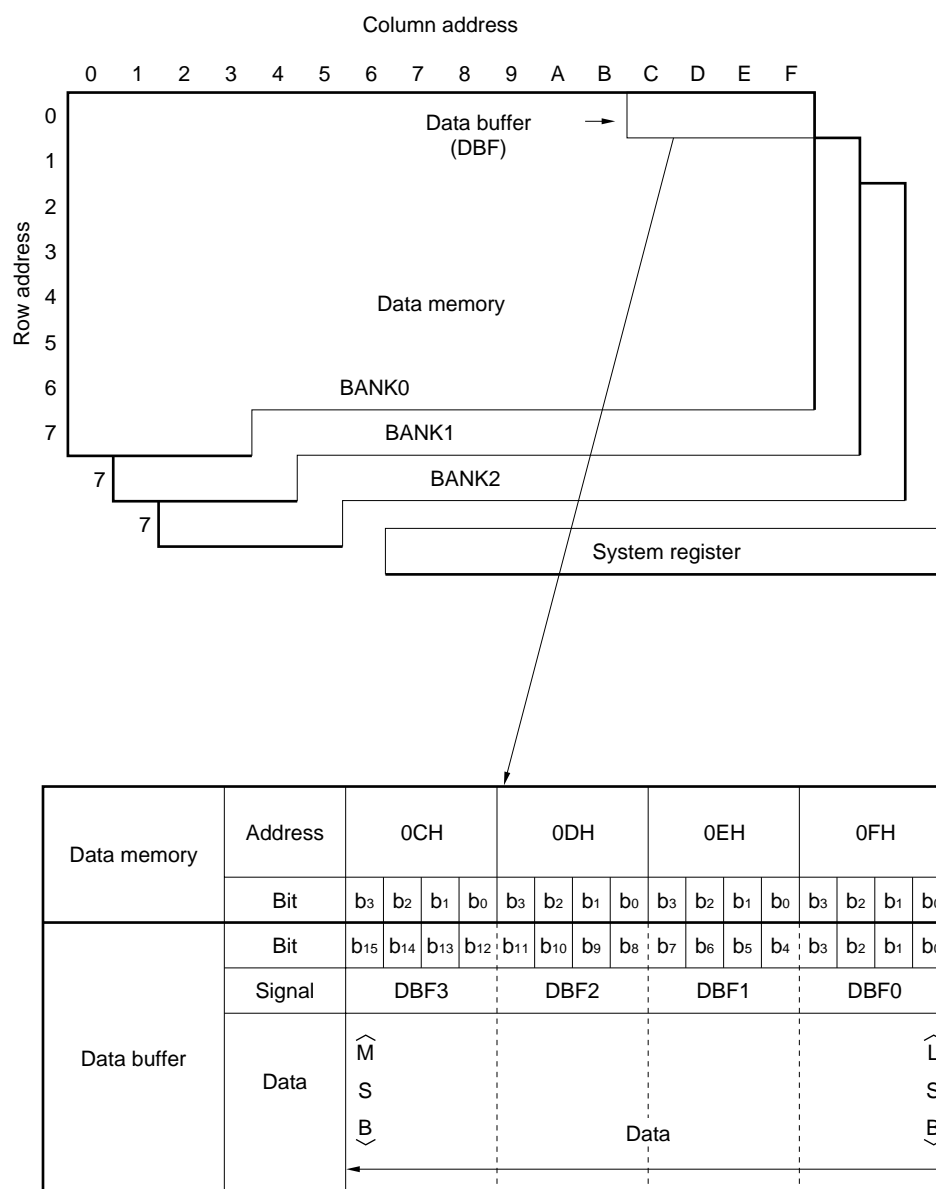
Figure 9-2 shows the configuration of the data buffer.

As shown in the figure, the data buffer consists of a total of 16 bits at addresses 0CH to 0FH of BANK 0 on the data memory.

The 16-bit data consists of bit  $b_3$  at address 0CH as the MSB and bit  $b_0$  at address 0FH as the LSB.

Because the data buffer is located in the data memory, it can be manipulated by all data memory manipulation instructions.

Figure 9-2. Configuration of Data Buffer



### 9.2.2 Table reference instruction (“MOV<sub>T</sub> DBF, @AR”)

The operation of the “MOV<sub>T</sub> DBF, @AR” instruction is indicated below.

MOV<sub>T</sub> DBF, @AR

This instruction reads the contents of the program memory addressed by the contents of the address register to the data buffer.

One stack level is used when the table reference instruction is used.

The program memory addresses that can be referenced by the table are all the addresses from 0000H to 0FFFFH of the program memory.

### 9.2.3 Peripheral hardware control instructions (“PUT”, “GET”)

The operations of the “PUT” and “GET” instructions are as follows:

#### (1) GET DBF, p

This instruction reads the data of the peripheral register addressed by p to the data buffer.

#### (2) PUT p, DBF

This instruction sets the data of the data buffer to the peripheral register addressed by p.

## 9.3 Peripheral Hardware and Data Buffer List

Table 9-1 lists the peripheral hardware units and the functions of the data buffer.

[MEMO]

Table 9-1. Relationship Between Peripheral Hardware and Data Buffer (1/2)

Peripheral Hardware		Peripheral Register That Transfers Data with Data Buffer			
		Name	Symbol	Peripheral Address	Execution of PUT/GET Instruction
A/D converter		A/D converter data register	ADCR	02H	PUT/GET
Serial interface		Presetable shift register	SIO1SFR	03H	PUT/GET
D/A converter (PWM output)	PWM <sub>0</sub> pin	PWM data register 0	PWMR0	04H	PUT/GET
	PWM <sub>1</sub> pin	PWM data register 1	PWMR1	05H	
Address register (AR)		Address register	AR	40H	PUT/GET
PLL frequency synthesizer		PLL data register	PLL	41H	PUT/GET
Key source controller/decoder		Key source data register	KSR	42H	PUT/GET
Port YA		Port YA group register	PYA	42H	PUT/GET
Frequency counter		IF counter data register	IFC	43H	GET
12-bit timer	Timer modulo	Timer modulo register	TMM	46H	PUT/GET
	Timer counter	Timer counter	TMC	47H	GET

Table 9-1. Relationship Between Peripheral Hardware and Data Buffer (2/2)

Function		
Number of I/O Bits of Data Buffer	Number of Bits Actually Used	Outline
8	6	Sets compare voltage $V_{REF}$ data of A/D converter $V_{REF} = \frac{x - 0.5}{64} \times V_{DD}$ , $1 \leq x \leq 63$
8	8	Sets serial out data and reads serial in data.
8	8	Sets duty factor of output signal of D/A converter. $\text{Duty } D = \frac{x + 0.25}{256} \times 100\%, \quad 0 \leq x \leq 255$ Frequency $f = 4.3945 \text{ kHz}$
16	13	Transfers data with address register.
16	16	Sets division ratio (N value) of PLL.
16	16	Sets output data of key source signal.
16	16	Sets output data of port YA 0: low level 1: high level
16	16	Reads count value of frequency counter.
16	12	Sets reference data of timer modulo.
16	12	Read data of up-counter

#### 9.4 Notes on Using Data Buffer

Note the following points (1) through (3) concerning unused peripheral address and write-only peripheral registers (PUT only) and read-only peripheral registers (GET only) when transferring data with the peripheral hardware units via the data buffer.

- (1) When a write-only register is read, an undefined value is read.
- (2) Nothing is changed even if data is written to a read-only register.
- (3) An undefined value is read if an unused register is read. Nothing is changed even if data is written to an unused register.

## 10. GENERAL-PURPOSE PORTS

The general-purpose ports output a high-level, low-level, or floating signal to an external circuit, and read a high-level or low-level signal from the external circuit.

### 10.1 Configuration and Classification of General-Purpose Ports

Figure 10-1 shows the block diagram of the general-purpose ports.

Table 10-1 classifies the general-purpose ports.

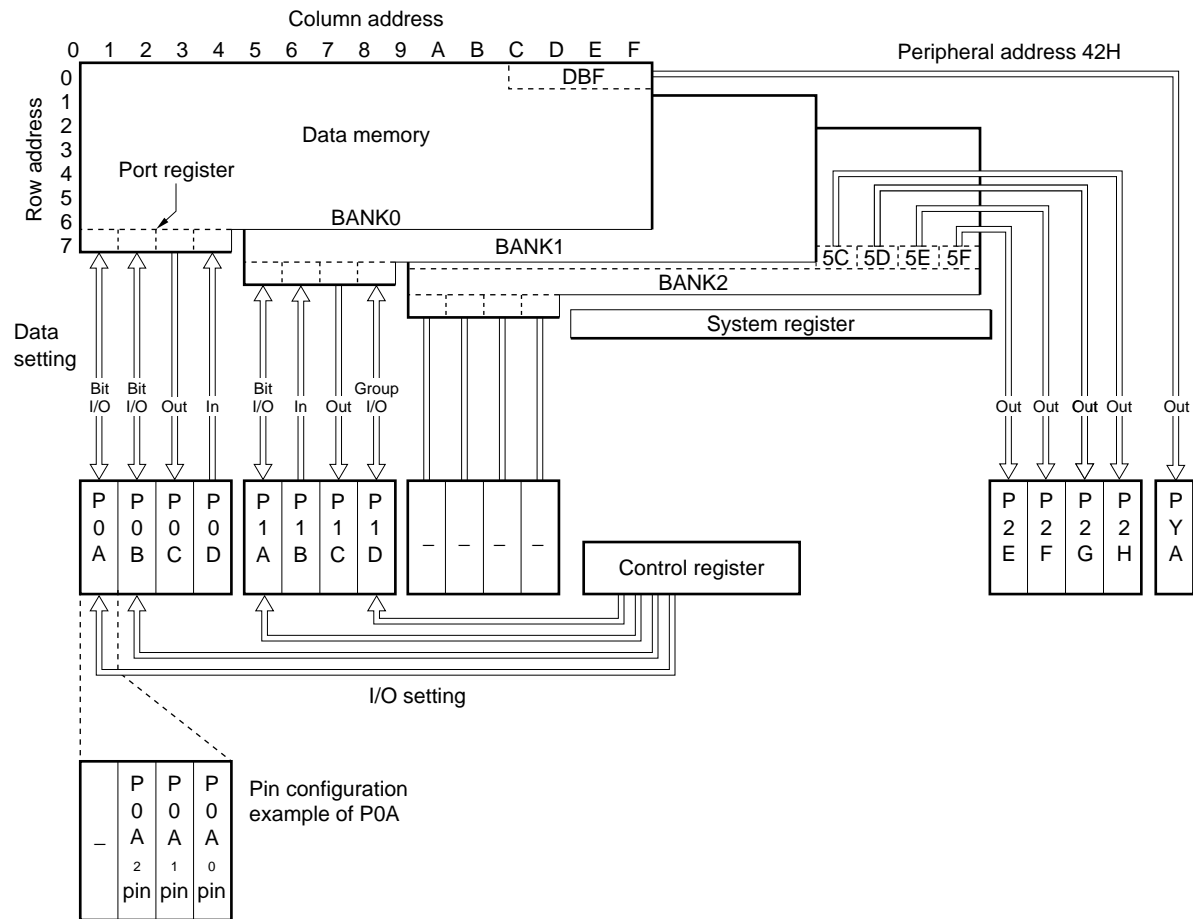
As shown in Figure 10-1, the general-purpose ports include ports 0A (P0A) to 1D (P1D) to which are set by addresses 70H to 73H (port registers) of each bank of the data memory, ports 2E (P2E) to 2H (P2H) to which data are set by addresses 5CH to 5FH of bank 2 of the data memory, and port YA (PYA) to which data is set via a data buffer (DBF).

Each port consists of general-purpose port pins (e.g., port 0A consists of the P0A<sub>2</sub> to P0A<sub>0</sub> pins).

As shown in Table 10-1, the general-purpose ports are classified into input/output ports (I/O ports), input-only ports (input ports), and output-only ports (output ports).

The I/O ports are further subdivided into bit I/O ports that can be set in the input or output mode in 1-bit units (1-pin units) and group I/O ports that can be set in the input or output mode in 4-bit units (4-pin units).

**Figure 10-1. Block Diagram of General-Purpose Port**





**Table 10-1. Classification of General-Purpose Ports**

Classification of General-Purpose Ports			Port	Data Set by:
General-purpose ports	I/O dedicated ports	Bit I/O	Port 0A Port 0B Port 1A	Port register
		Group I/O	Port 1D	Port register
	Input dedicated port		Port 0D Port 1B	Port register
	Output dedicated port		Port 0C Port 1C	Port register
			Port 2E Port 2F Port 2G Port 2H	Port register (multiplexed with LCD segment register)
			Port YA	Peripheral register

## ★ 10.2 Functional Outline of General-Purpose Ports

The general-purpose output ports and the general-purpose I/O ports set in the output mode output a high or low level from the corresponding pins when data is set to the corresponding port register or port group register.

The general-purpose input ports and the general-purpose I/O ports set in the input mode detect the level of the signals input to the corresponding pins by reading the contents of the corresponding port register.

The general-purpose I/O ports are set in the input or output mode by the corresponding control register.

In other words, these ports can be set in the input or output mode by program.

P0A to P0D and P1A to P1D are set in the general-purpose port mode on power-on reset.

P2E to P2H and PYA are used as LCD segment signal output pins on power-on reset. To use these ports as general-purpose output ports, the corresponding control registers must be set independently.

The following subsections 10.2.1 to 10.2.4 explain the port registers, the function of the port group register, and the functional outline of each port.

### 10.2.1 General-purpose port data register (port register)

A port register sets the output data and reads the input data of the corresponding general-purpose port.

Because the port registers are mapped in the data memory, they can be manipulated by any data memory manipulation instruction.

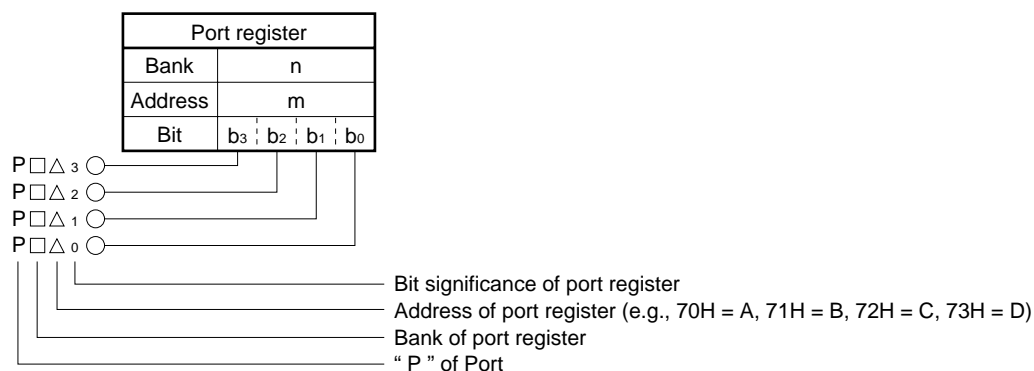
Figure 10-2 shows the relationship between a port register and the corresponding port pins.

By setting data to the port register corresponding to the port pins set in the general-purpose output port mode, the output of each pin is set.

By reading the contents of the port register corresponding to the port pins set in the general-purpose input port mode, the input status of each pin is detected.

Table 10-2 shows the relationship between each port (each pin) and port register.

**Figure 10-2. Relationship Between Port Register and Pins**



Reserved words are defined for the port registers by the assembler.

Because these reserved words are defined in flag (bit) units, the assembler-embedded macro instructions can be used.

Note that data memory type reserved words are not defined for the port registers.

P2E to P2H are multiplexed with LCD segment signal output pins. The port registers of P2E to P2H are also multiplexed with LCD segment registers.

Because the LCD segment registers are also mapped in the data memory, they can be treated in the same manner as the port registers.

### 10.2.2 Port YA (PYA) group register

The port YA (PYA) group register sets the output data of PYA. Port YA functions alternately as the key source signal output pin. Therefore, the PYA group register is also used as the key source data register and is allocated to address 42H of the peripheral addresses. For details, refer to **10.6.7**.

### 10.2.3 General-purpose I/O ports (P0A, P0B, P1A, and P1D)

P0A, P0B, P1A, and P1D can be set in the input or output mode by the port 0A bit I/O select register (RF address 37H), port 0B bit I/O select register (RF address 36H), port 1A bit I/O select register (RF address 35H), and port 1D group I/O select register (RF address 27H), respectively.

The input/output data of the P0A, P0B, P1A, and P1D are set by port registers P0A (address 70H of BANK0), P0B (address 71H of BANK0), P1A (address 70H of BANK1), and P1D (address 73H of BANK1), respectively.

Refer to **Table 10-2**.

For details, refer to **10.3**.

#### 10.2.4 General-purpose input ports (P0D and P1B)

The input data of P0D and P1B is read by port registers P0D (address 73H of BANK0) and P1B (address 71H of BANK1), respectively.

Refer to **Table 10-2**.

For details, refer to **10.4**.

#### 10.2.5 General-purpose output ports (P0C, P1C, P2E, P2F, P2G, P2H, and PYA)

##### (1) P0C, P1C

The output data of P0C and P1C is set by port registers P0C (address 72H of BANK0) and P1C (address 72H of BANK1).

Refer to **Table 10-2**.

For details, refer to **10.5**.

##### (2) P2E, P2F, P2G, P2H, and PYA

P2E, P2F, P2G, P2H, and PYA usually operate as LCD segment signal output pins. To use these ports as the output ports select the port using the P2ESEL to P2HSEL and PYASEL flags of the LCD port select register and LCD mode select register.

The port to be used can be selected individually using P2E to P2H and PYA.

The output data of P2E, P2F, P2G, and P2H can be set by the P2E register (also used as LCDD16 of the LCD segment register, address 5FH of BANK2), P2F register (also used as LCDD17, address 5EH of BANK2), P2G register (also used as LCDD18, address 5DH of BANK2), and P2H register (also used as LCDD19, address 5CH of BANK2).

Refer to **Table 10-2**.

For details, refer to **10.6**.

Table 10-2. Relationship Between Each Port (Pin) and Port Register (1/2)

Port	Pin			Data Setting Method					
	No.	Symbol	I/O	Port Register (Data Memory)				Remarks	
				Bank	Address	Symbol	Bit Symbol (Reserved Word)		
Port 0A	No pin		I/O (bit I/O)	BANK0	70H	P0A	b <sub>3</sub>	P0A3	Fixed to “0”
	9 (10)	P0A <sub>2</sub>					b <sub>2</sub>	P0A2	
	10 (11)	P0A <sub>1</sub>					b <sub>1</sub>	P0A1	
	11 (12)	P0A <sub>0</sub>					b <sub>0</sub>	P0A0	
Port 0B	16 (18)	P0B <sub>3</sub>	I/O (bit I/O)		71H	P0B	b <sub>3</sub>	P0B3	
	17 (19)	P0B <sub>2</sub>					b <sub>2</sub>	P0B2	
	18 (20)	P0B <sub>1</sub>					b <sub>1</sub>	P0B1	
	19 (21)	P0B <sub>0</sub>					b <sub>0</sub>	P0B0	
Port 0C	27 (33)	P0C <sub>3</sub>	Output		72H	P0C	b <sub>3</sub>	P0C3	
	28 (34)	P0C <sub>2</sub>					b <sub>2</sub>	P0C2	
	29 (35)	P0C <sub>1</sub>					b <sub>1</sub>	P0C1	
	30 (37)	P0C <sub>0</sub>					b <sub>0</sub>	P0C0	
Port 0D	59 (73)	P0D <sub>3</sub>	Input		73H	P0D	b <sub>3</sub>	P0D3	
	60 (74)	P0D <sub>2</sub>					b <sub>2</sub>	P0D2	
	61 (75)	P0D <sub>1</sub>					b <sub>1</sub>	P0D1	
	62 (76)	P0D <sub>0</sub>					b <sub>0</sub>	P0D0	
Port 1A	No pin		I/O (bit I/O)	BANK1	70H	P1A	b <sub>3</sub>	P1A3	Fixed to “0”
	63 (77)	P1A <sub>2</sub>					b <sub>2</sub>	P1A2	
	1 (80)	P1A <sub>1</sub>					b <sub>1</sub>	P1A1	
	2 (1)	P1A <sub>0</sub>					b <sub>0</sub>	P1A0	
Port 1B	12 (13)	P1B <sub>3</sub>	Input		71H	P1B	b <sub>3</sub>	P1B3	
	13 (14)	P1B <sub>2</sub>					b <sub>2</sub>	P1B2	
	14 (16)	P1B <sub>1</sub>					b <sub>1</sub>	P1B1	
	15 (17)	P1B <sub>0</sub>					b <sub>0</sub>	P1B0	
Port 1C	20 (22)	P1C <sub>3</sub>	Output		72H	P1C	b <sub>3</sub>	P1C3	
	21 (24)	P1C <sub>2</sub>					b <sub>2</sub>	P1C2	
	22 (25)	P1C <sub>1</sub>					b <sub>1</sub>	P1C1	
	23 (26)	P1C <sub>0</sub>					b <sub>0</sub>	P1C0	
Port 1D	31 (38)	P1D <sub>3</sub>	I/O (group I/O)		73H	P1D	b <sub>3</sub>	P1D3	
	32 (39)	P1D <sub>2</sub>					b <sub>2</sub>	P1D2	
	33 (40)	P1D <sub>1</sub>					b <sub>1</sub>	P1D1	
	34 (41)	P1D <sub>0</sub>					b <sub>0</sub>	P1D0	

**Remark** Numbers in parentheses are pin numbers for pin 80.

Table 10-2. Relationship Between Each Port (Pin) and Port Register (2/2)

Port	Pin			Data Setting Method						
	No.	Symbol	I/O	Port Register (Data Memory)				Port Group Register (Peripheral Register)		
				Bank	Address	Symbol	Bit Symbol (Reserved Word)	Peripheral Address	Symbol (Reserved Word)	Bit
				BANK2	70H	---	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	Fixed to "0"		
					71H	---	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>			
					72H	---	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>			
					73H	---	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>			
Port 2E	No pin 41 (49)	P2E <sub>0</sub>	Output	BANK2	5FH (multiplexed with LCDD16)	P2E	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	P2E3 P2E2 P2E1 P2E0	Can be used as data memory	
Port 2F	No pin 40 (48)	P2F <sub>0</sub>	Output		5EH (multiplexed with LCDD17)	P2F	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	P2F3 P2F2 P2F1 P2F0	Can be used as data memory	
Port 2G	No pin 39 (47)	P2G <sub>0</sub>	Output		5DH (multiplexed with LCDD18)	P2G	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	P2G3 P2G2 P2G1 P2G0	Can be used as data memory	
Port 2H	No pin 38 (46)	P2H <sub>0</sub>	Output		5CH (multiplexed with LCDD19)	P2H	b <sub>3</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>	P2H3 P2H2 P2H1 P2H0	Can be used as data memory	
Port YA	42 (50) 43 (52) 44 (53) 55 (65) 56 (66) 57 (67)	PYA <sub>15</sub> PYA <sub>14</sub> PYA <sub>13</sub> PYA <sub>2</sub> PYA <sub>1</sub> PYA <sub>0</sub>	Output					42H (multiplexed with KSR)	PYAR	b <sub>15</sub> b <sub>14</sub> b <sub>13</sub> b <sub>2</sub> b <sub>1</sub> b <sub>0</sub>

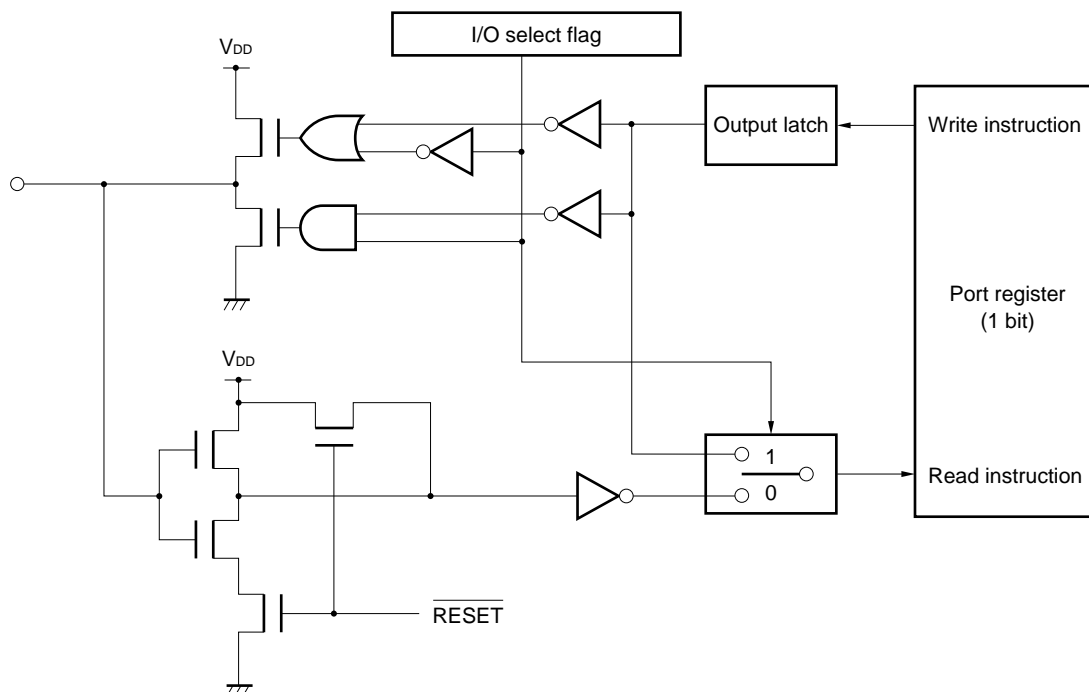
**Remark** Numbers in parentheses are pin numbers for pin 80.

### 10.3 General-Purpose I/O Ports (P0A, P0B, P1A, and P1D)

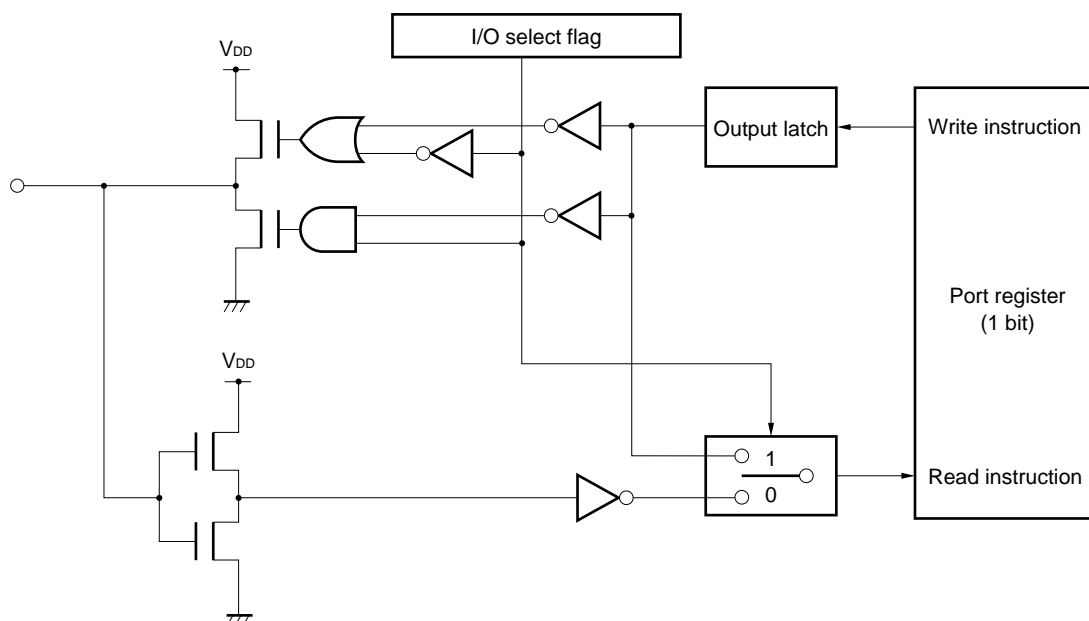
#### 10.3.1 Configuration of I/O ports

The following paragraphs (1) through (3) indicate the configuration of the I/O ports.

- (1) P0A (P0A<sub>2</sub>, P0A<sub>1</sub>, and P0A<sub>0</sub> pins),  
 P0B (P0B<sub>3</sub>, P0B<sub>2</sub>, P0B<sub>1</sub>, and P0B<sub>0</sub> pins),  
 P1A (P1A<sub>2</sub>, P1A<sub>1</sub>, and P1A<sub>0</sub> pins)



- (2) P1D (P1D<sub>3</sub>, P1D<sub>2</sub>, P1D<sub>1</sub>, and P1D<sub>0</sub> pins)



### 10.3.2 Using I/O ports

The I/O ports are set in the input or output mode by I/O select registers P0A, P0B, P1A, and P1D of the control registers.

The bit I/O ports (P0A, P0B, and P1A) can be set in the input or output mode in 1-bit units, and group I/O port (P1D) can be set in the input or output mode in 4-bit units.

Output data can be set to a port by writing the data to the corresponding port register, and the input data of the port can be read by executing an instruction that reads the port register.

10.3.3 explains the I/O select register of each port.

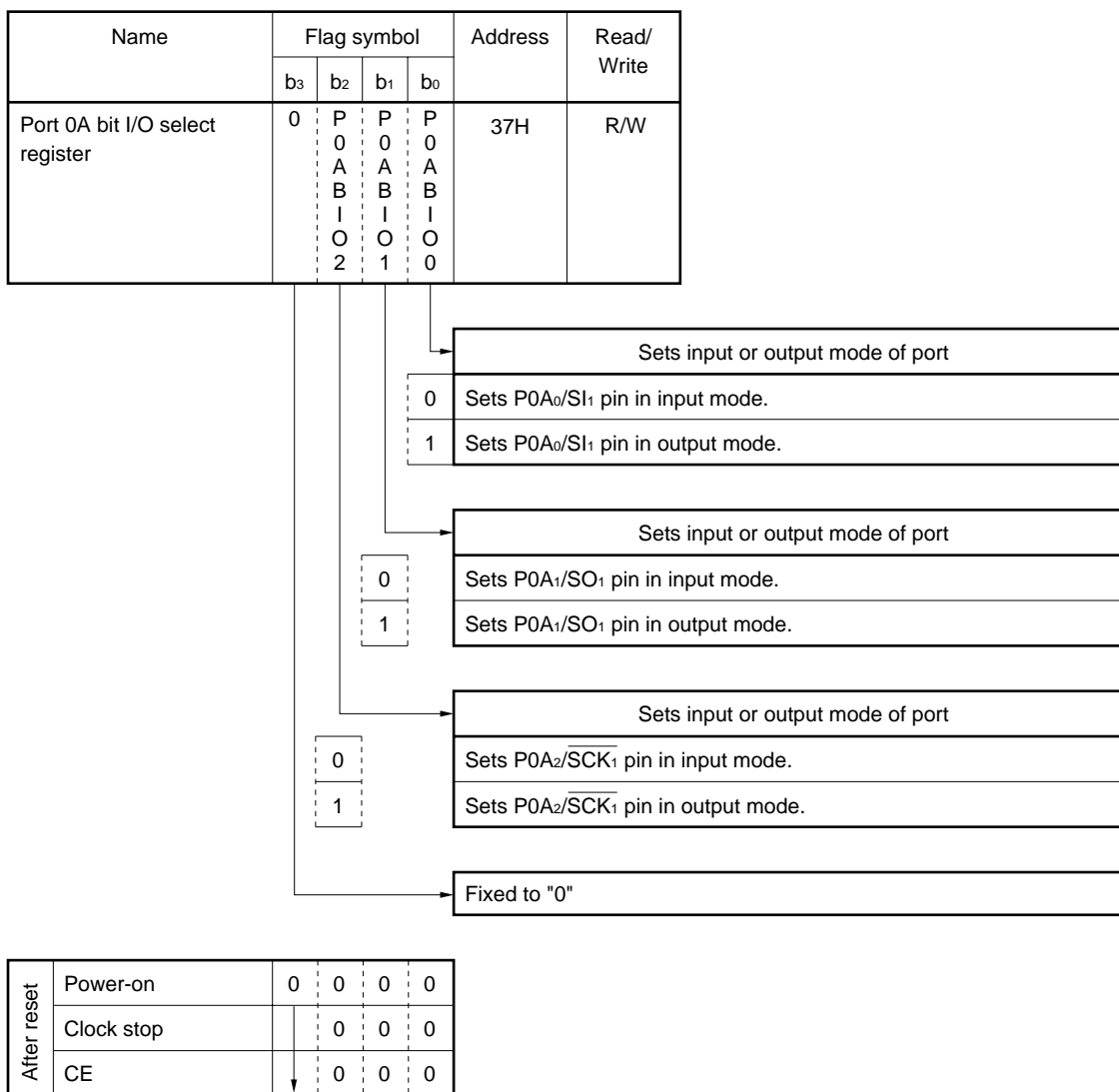
10.3.4 and 10.3.5 explain how to use the input and output ports.

### 10.3.3 I/O port control register

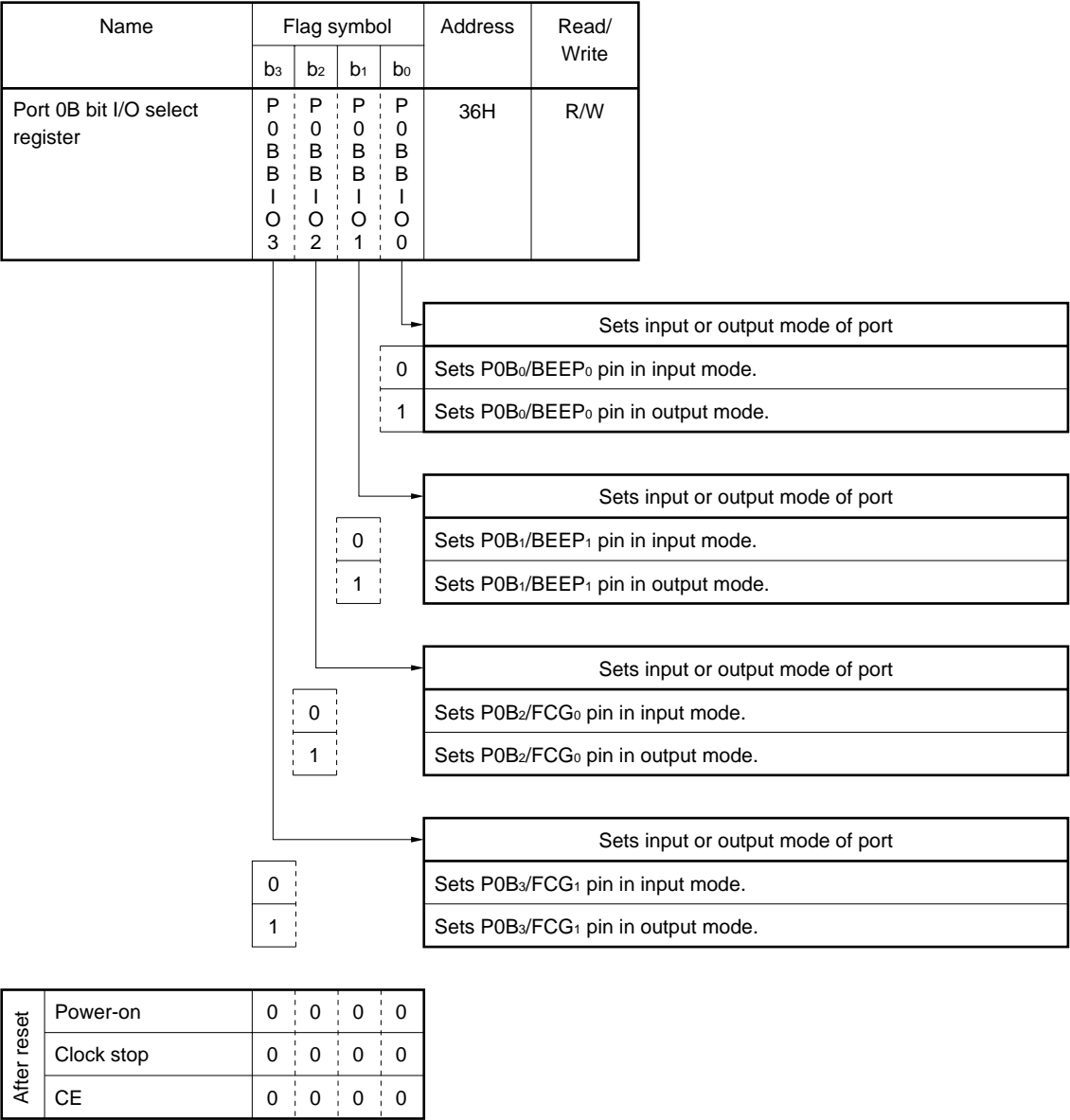
The port 0A bit I/O, port 0B bit I/O, port 1A bit I/O, and port 1D group I/O select registers set each pin of the P0A, P0B, P1A, and P1D in the input or output mode.

The configuration and functions of these registers are shown below.

#### (1) Port 0A bit I/O select register



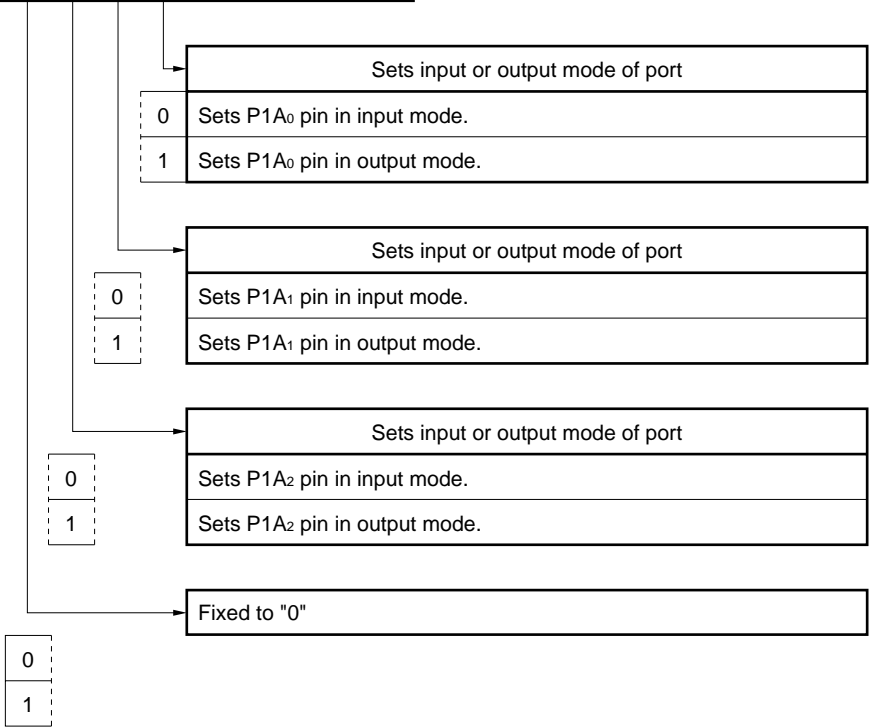
(2) Port 0B bit I/O select register





(3) Port 1A bit I/O select register

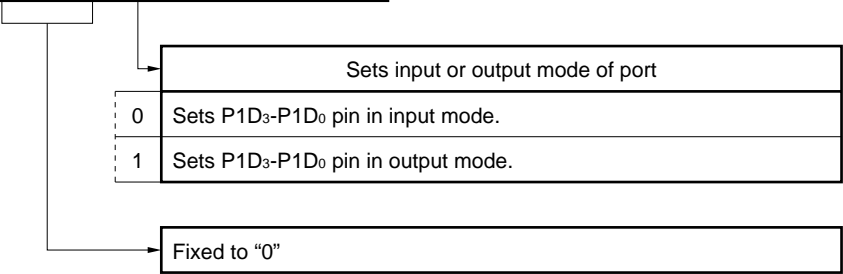
Name	Flag symbol				Address	Read/ Write
	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>		
Port 1A bit I/O select register	0	P 1 A B I O 2	P 1 A B I O 1	P 1 A B I O 0	35H	R/W



After reset	Power-on	0	0	0	0
	Clock stop		0	0	0
	CE	↓	0	0	0

(4) Port 1D group I/O select register

Name	Flag symbol				Address	Read/ Write
	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>		
Port 1D group I/O select register	0	0	0	P 1 D G I O	27H	R/W



After reset	Power-on	0	0	0	0
	Clock stop				0
	CE				0

### 10.3.4 Using I/O ports (P0A, P0B, P1A, and P1D) as input ports

Select the pin to be used as an input port pin by the I/O select register corresponding to each port.

Note that P1D can be set in the input or output mode in 4-bit units only.

The pin specified as an input port pin is floated (Hi-Z), and waits for input of an external signal.

The input data can be read by executing an instruction that reads the contents of the port register corresponding to each port, such as the SKT instruction.

When a high level is input to each pin, "1" is read to the corresponding port register; when a low level is input, "0" is read.

If a write instruction, such as MOV, is executed to the port register corresponding to the port pin specified as an input port pin, the contents of the output latch are rewritten.

### 10.3.5 Using I/O ports (P0A, P0B, P1A, and P1D) as output ports

Select the pin to be used as an output port pin by the I/O select register corresponding to each port.

Note that P1D can be set in the input or output mode in 4-bit units only.

The pin specified as an output port pin outputs the contents of the output latch.

The output data can be set by executing an instruction that writes the contents of the corresponding port register to each pin, such as the MOV instruction.

To output a high level to each pin, write "1" to the corresponding port register; to output a low level, write "0".

The port pin can also be floated when it is specified as an input port pin.

When an instruction, such as SKT, that reads the contents of the port register corresponding to a port specified as an output port is executed, the contents of the output latch are read.

### 10.3.6 Status of I/O ports (P0A, P0B, P1A, and P1D) on reset

#### (1) On power-on reset

All the I/O ports are set in the input mode.

Because the contents of the output latch are undefined, the output latch must be initialized by program, as necessary, before setting the corresponding port in the output mode.

#### (2) On CE reset

All the I/O ports are set in the input mode.

The contents of the output latch are retained.

#### (3) On execution of clock stop instruction

All the I/O ports are set in the input mode.

The contents of the output latch are retained.

I/O ports other than P1D prevent an increase in the current consumption due to the noise of the input buffer by using the  $\overline{\text{RESET}}$  signal when the clock stop instruction is executed, as explained in 10.3.1.

If P1D is floated on execution of the clock stop instruction, the current consumption may increase due to external noise. Externally pull this port down or up as necessary.

#### (4) In halt status

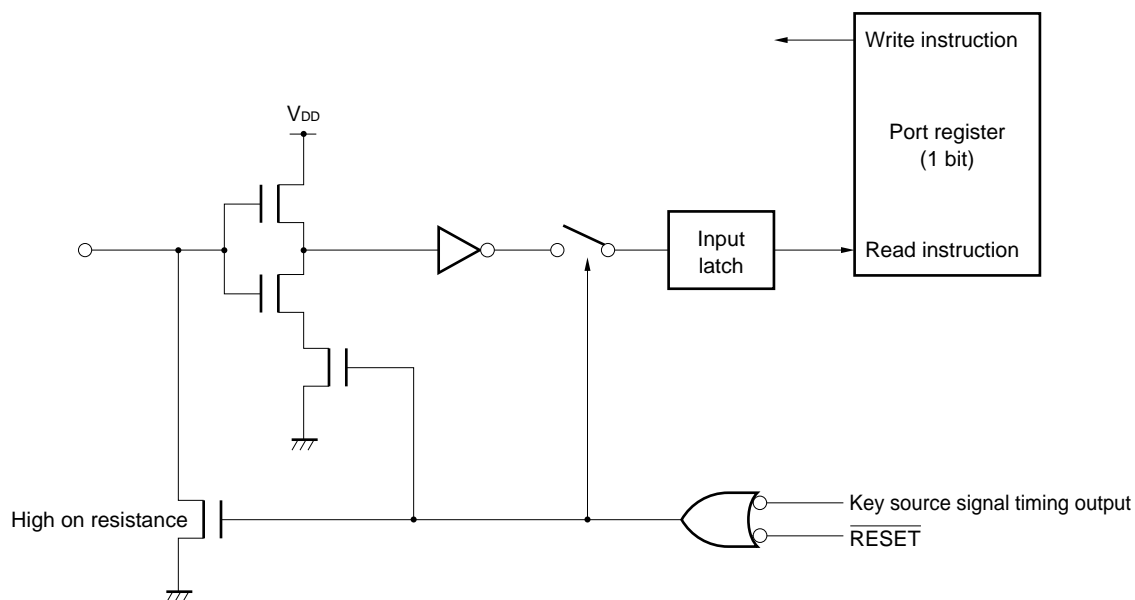
The previous status is retained.

## 10.4 General-Purpose Input Ports (P0D and P1B)

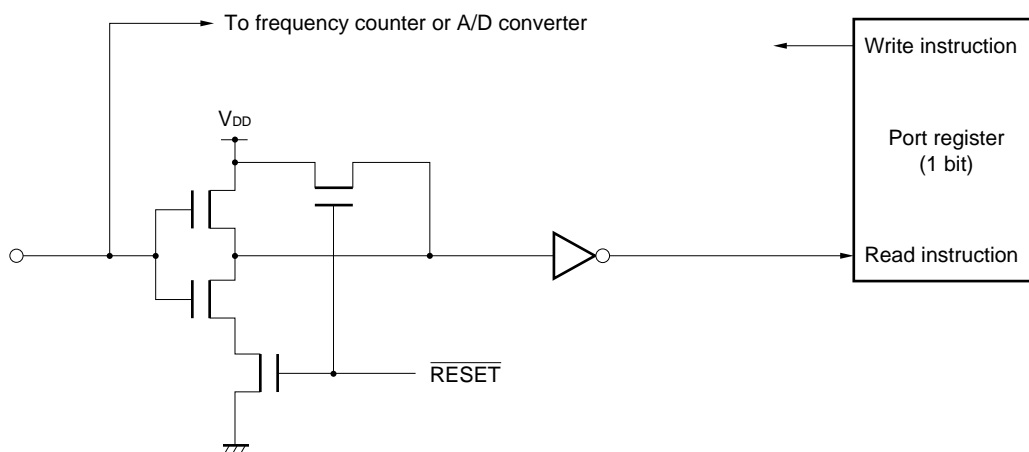
### 10.4.1 Configuration of input ports

The following paragraphs (1) and (2) indicate the configuration of the input ports.

#### (1) P0D (P0D<sub>3</sub>, P0D<sub>2</sub>, P0D<sub>1</sub>, and P0D<sub>0</sub> pins)



#### (2) P1B (P1B<sub>3</sub>, P1B<sub>2</sub>, P1B<sub>1</sub>, and P1B<sub>0</sub> pins)



#### 10.4.2 Using input ports (P0D and P1D)

The input data is read by executing an instruction, such as SKT, that reads the contents of the port register corresponding to each port pin.

When a high level is input to each pin, "1" is read to the corresponding port register; when a low level is input, "0" is read.

Nothing is changed even if a write instruction, such as MOV, is executed to the port register.

#### 10.4.3 Notes on using input port (P0D)

The P0D is internally pulled down when it is used as a general-purpose port.

#### 10.4.4 Status of input ports (P0D and P1B) on reset

##### (1) On power-on reset

All the input ports are specified as general-purpose input ports.

##### (2) On CE reset

All the input ports are specified as general-purpose input ports.

##### (3) On execution of clock stop instruction

All the input ports are specified as general-purpose input ports.

Because the  $\overline{\text{RESET}}$  signal is output when the clock stop instruction is executed, P1B prevents an increase in the current dissipation due to the noise of the input buffer as described in 10.4.1.

P0D is internally pulled down.

##### (4) In halt status

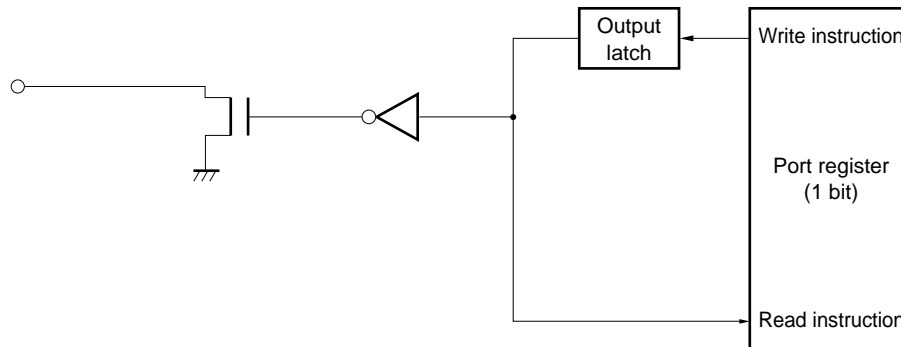
The previous status is retained.

## 10.5 General-Purpose Output Ports (P0C and P1C)

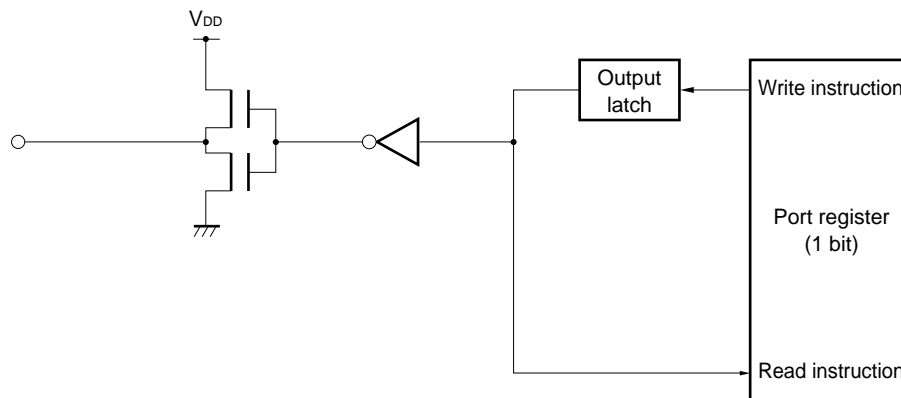
### 10.5.1 Configuration of output ports (P0C and P1C)

The following paragraphs (1) and (2) indicate the configuration of the output ports.

#### (1) P0C (P0C<sub>3</sub>, P0C<sub>2</sub>, P0C<sub>1</sub>, and P0C<sub>0</sub> pins)



#### (2) P1C (P1C<sub>3</sub>, P1C<sub>2</sub>, P1C<sub>1</sub>, and P1C<sub>0</sub> pins)



### 10.5.2 Use of output ports (P0C and P1C)

The output ports output the contents of the output latches.

The output data is set by executing an instruction, such as MOV, that writes the data to the port register corresponding to the output port.

Write "1" to the port register to output a high level to the corresponding port; write "0" to output a low level.

Note, however, that the P0C<sub>3</sub>, P0C<sub>2</sub>, P0C<sub>1</sub>, and P0C<sub>0</sub> pins are N-ch open-drain output pins and are floated when a high level is output.

When an instruction, such as SKT, that reads the contents of the port register is read, the contents of the output latch are read.

### 10.5.3 Status of output ports (P0C and P1C) on reset

#### (1) On power-on reset

The contents of the output latch are output.

Because the contents of the output latch are undefined, an undefined value is output for a fixed period (until the output latch is initialized by program).

#### (2) On CE reset

The contents of the output latch are output.

The contents of the output latch are retained and the output data is not changed on CE reset.

#### (3) On execution of clock stop instruction

The contents of the output latch are output.

The contents of the output latch are retained and the output data is not changed on execution of the clock stop instruction.

Therefore, initialize the output latch in the program as necessary.

#### (4) In halt status

The contents of the output latch are output.

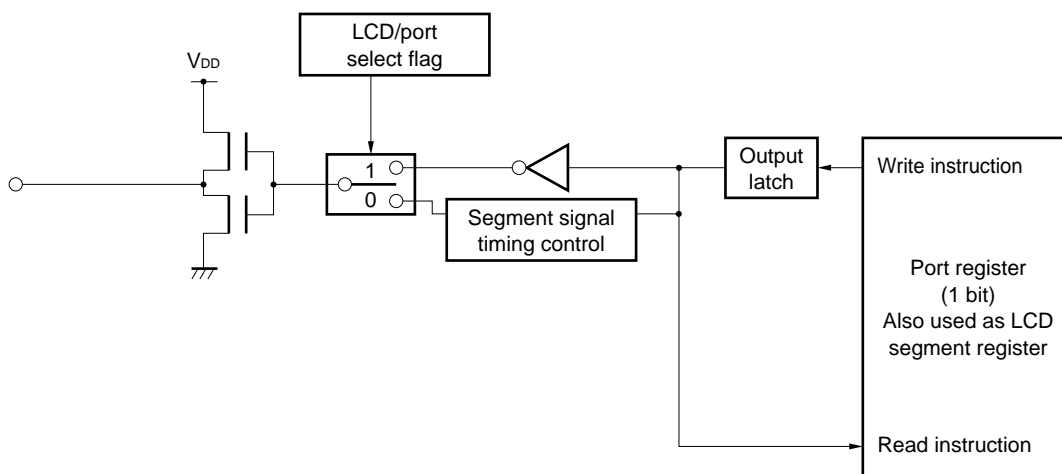
The contents of the output latch are retained and the output data is not changed in the halt status.

## 10.6 General-Purpose Output Ports (P2E to P2H and PYA)

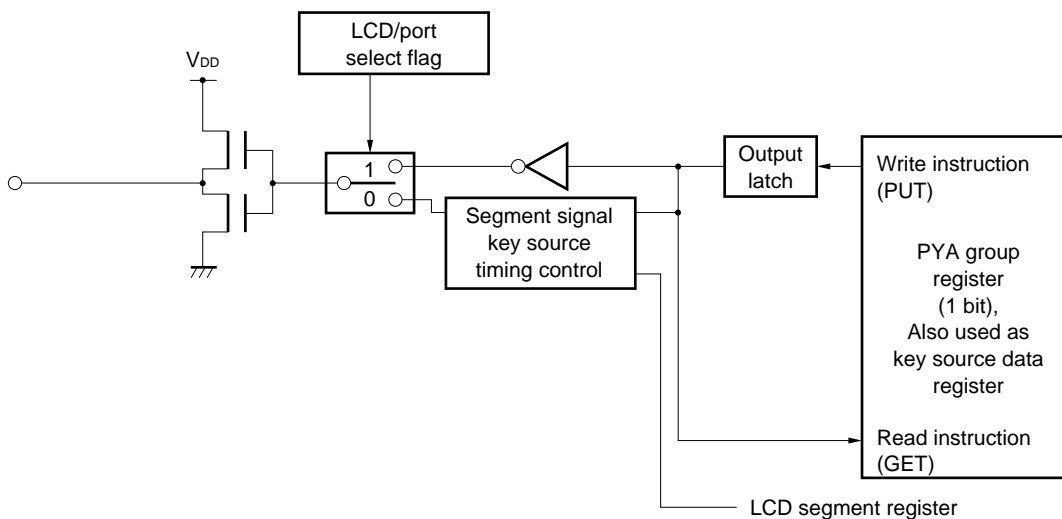
### 10.6.1 Configuration of output ports (P2E to P2H and PYA)

The configuration of the output ports is shown below.

- (1) P2E (P2E<sub>0</sub> pin)
- P2F (P2F<sub>0</sub> pin)
- P2G (P2G<sub>0</sub> pin)
- P2H (P2H<sub>0</sub> pin)



- (2) PYA (PYA<sub>15</sub> to PYA<sub>0</sub>)





### 10.6.2 Example of using output ports (P2E to P2H and PYA)

Each pin of P0E and P0F are used as an LCD segment signal output pin on power-on reset.

To use it as an output port pin, select the port to be used by the P2ESEL to P2HSEL and PYASEL flags of the LCD port select register and LCD mode select register.

The port to be used can be selected by P2E to P2H and PYA independently.

The pins not set in the output port mode by the LCD port select register and LCD mode select register can be used as LCD segment signal output pins.

The setting of P2E to P2H and PYA output data is described in section 10.6.3 and 10.6.4.

The configuration and functions of the LCD port select register, LCD mode select register, and port YA (PYA) group register are described in section 10.6.5 to 10.6.7.

### 10.6.3 Setting data to P2E to P2H

Output data is set to P2E to P2H by executing an instruction, such as MOV, that writes data to the port registers corresponding to the ports.

To output a high level to each port pin, write “1” to the corresponding port register; to output a low level, write “0”.

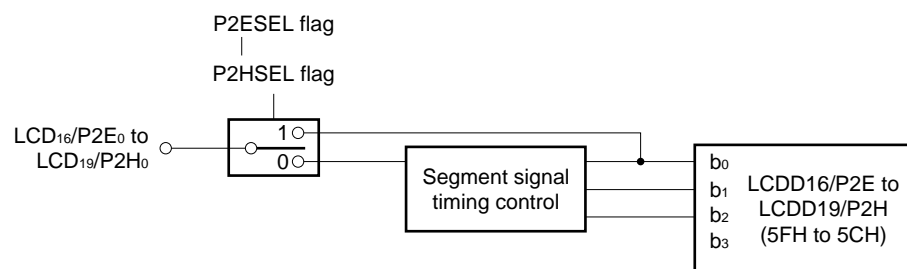
The contents of the output latch are read when an instruction, such as SKT, that reads the contents of the port register is executed.

Figure 10-3 shows the relationship between the P2E to P2H port registers and LCD segment register.

As shown in this figure, the LCD segment register's higher 3 bits, LCDD16 to LCDD19, can be used as a general-purpose data memory area when P2E to P2H are used.

Refer to **Figure 19-5. Relationship Between LCD Display Dots, Output of Each Pin, and Data Setting Registers** in **14. LCD CONTROLLER/DRIVER**.

**Figure 10-3. Relationship Between Port Registers P2E to P2H and LCD Segment Register**



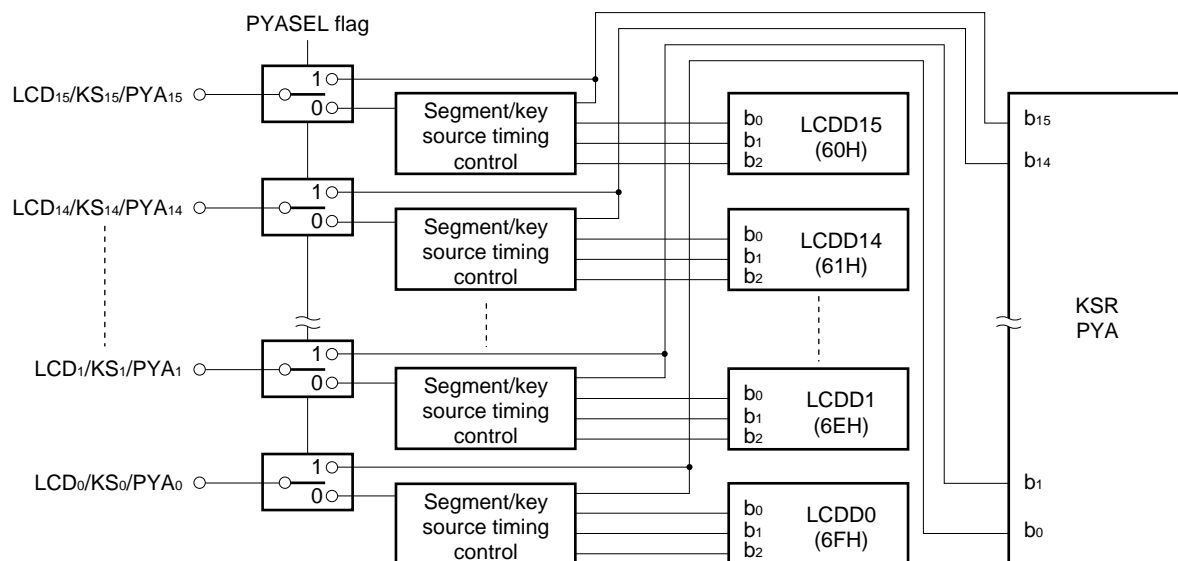
#### 10.6.4 PYA data setting

To set output data to PYA, execute the write the instruction “PUT PYA, DBF” to the port YA (PYA) group register corresponding to each pin.

When the instruction “GET DBF, PYA”, which reads the contents of a PYA group register, is executed, the contents of the output latch are read.

To output a high level to each pin, write “1” to the corresponding port register; to output a low level, write “0”.

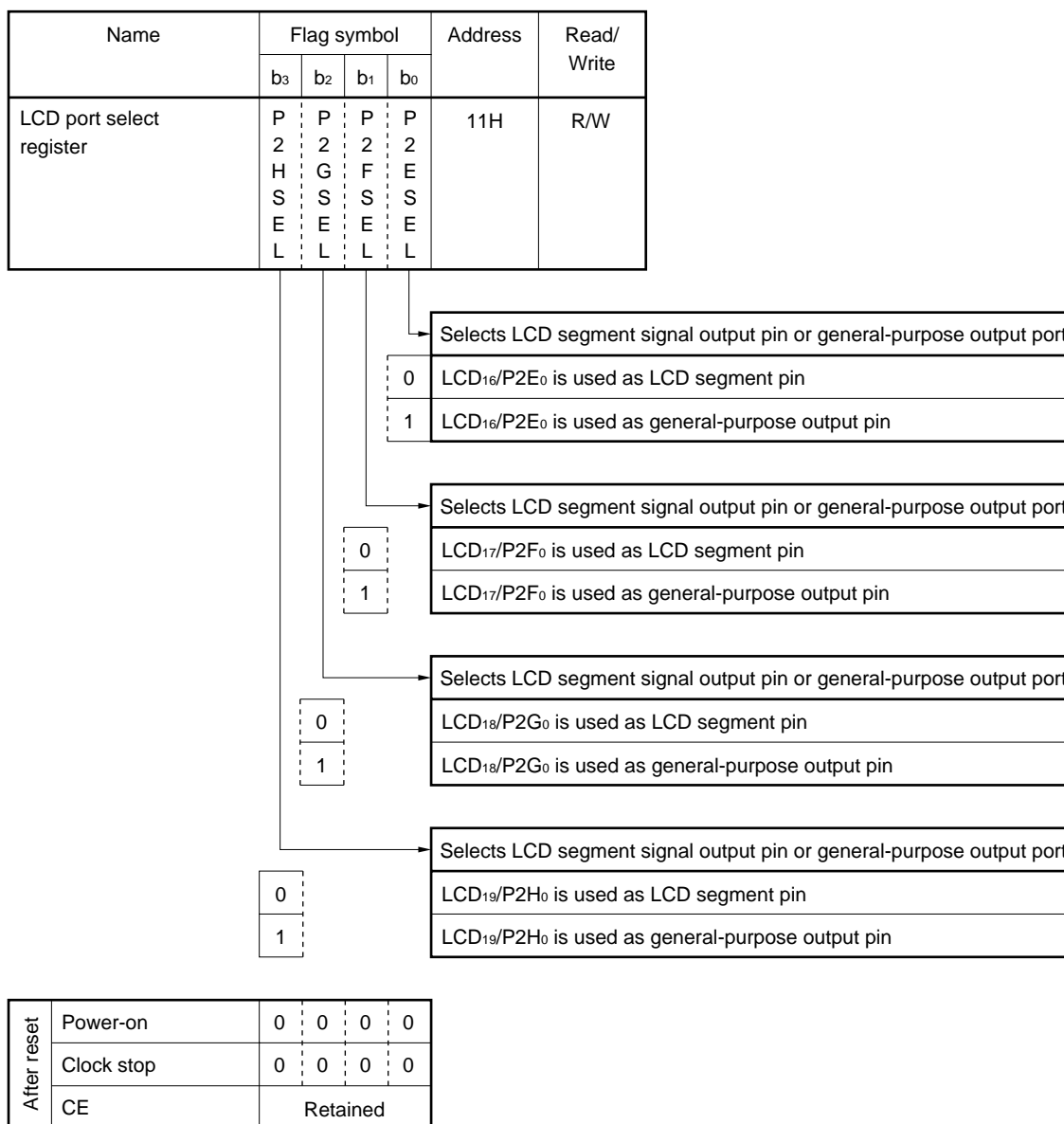
**Figure 10-4. Relationship Between PYA Group Register and LCD Segment Register**



### 10.6.5 Configuration and functions of LCD port select register

The LCD port select register specifies whether P2E, P2F, P2G, and P2H are used as LCD segment signal output pins or as general-purpose output port pins.

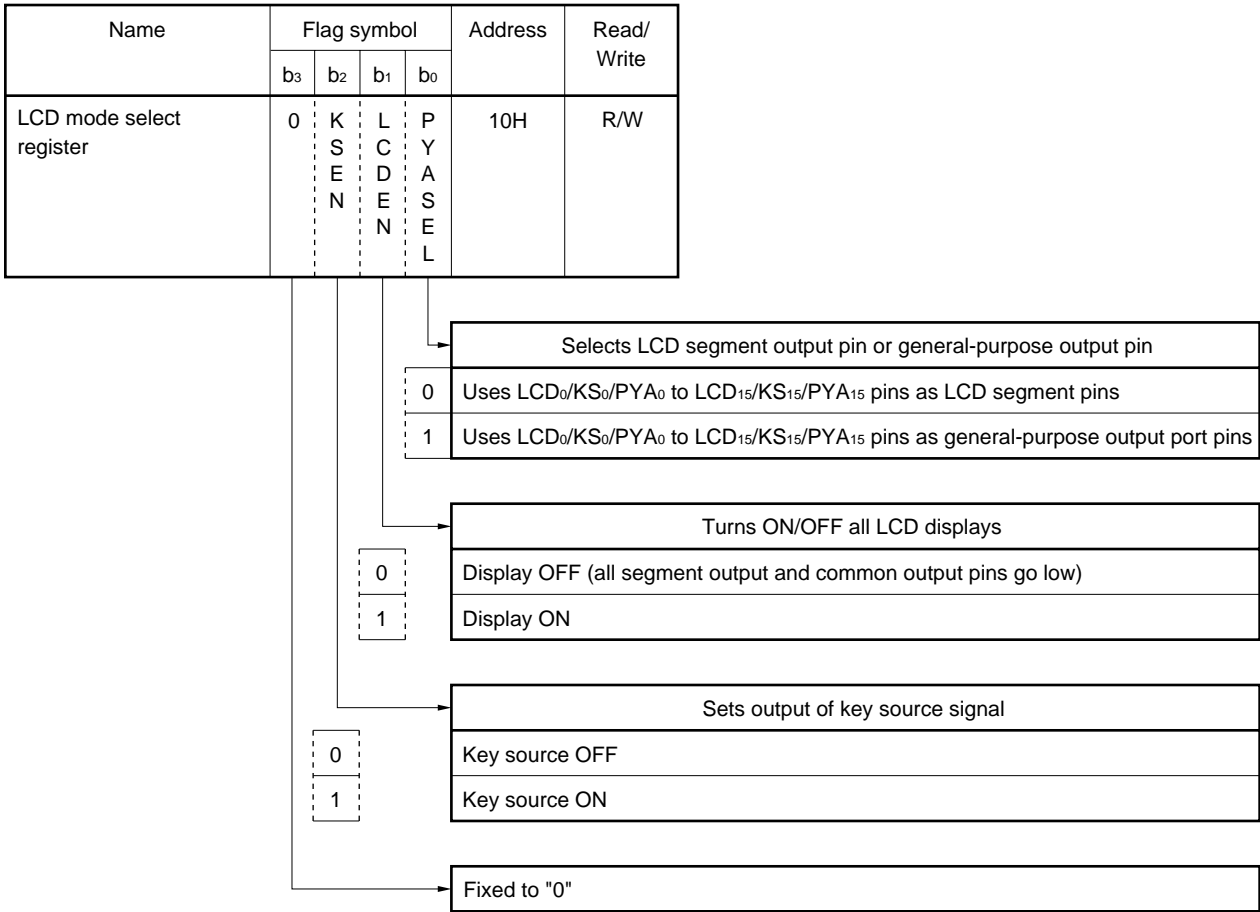
The configuration and function of this register are illustrated below.



### 10.6.6 Configuration and function of LCD mode select register

The LCD mode select register specifies whether the PYA pins are used as LCD segment signal output pins or as general-purpose port pins. This register also turns ON/OFF all the LCD displays, and outputs key source signals.

The configuration and function of this register are illustrated below.



After reset	Power-on	0	0	0	0
	Clock stop		0	0	0
	CE	Retained			

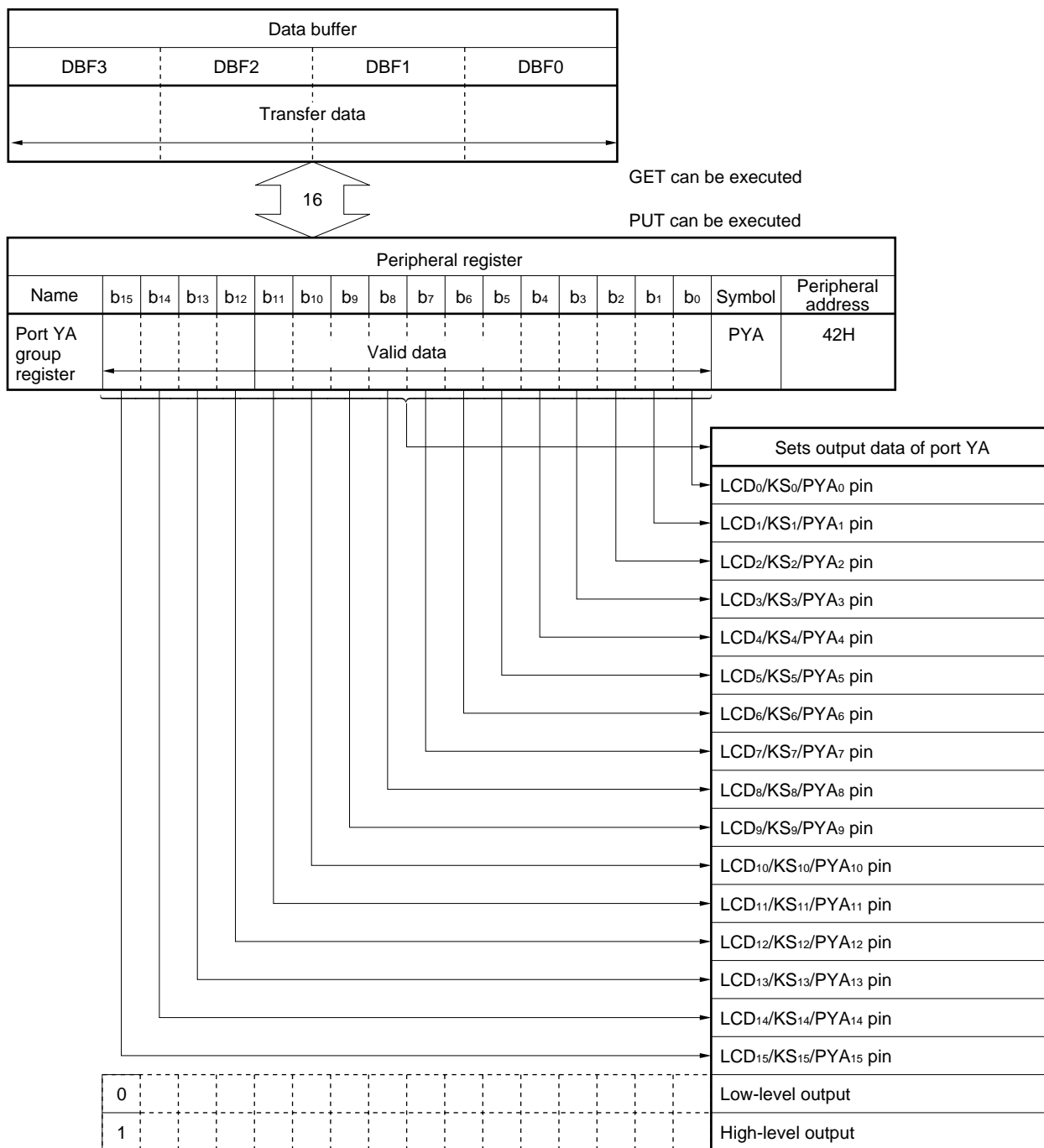
The 16 pins, LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> to LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub>, function alternately as LCD segment signal outputs and key source signal outputs. When any of these pins is set as a general-purpose output port pin, however, neither the LCD segment signal nor key source signal is output.

### 10.6.7 Port YA (PYA) group register

The PYA group register sets the output data of the PYA pins (PYA<sub>0</sub> through PYA<sub>15</sub>).

The PYA pins can set 16-bit output data all at once.

The function of the PYA group register is illustrated below.



Port YA is alternately used with key source signal output pins.

Therefore, the PYA group register (peripheral address: 42H) is alternately used with the key source data register (peripheral address: 42H), which is to be described later.

Consequently, the PYA group register is used to set the output data of port YA when the LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> to LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> pins are specified as output port pins, and key source signal output data when these pins are specified as key source signal output pins.

### 10.6.8 Status of output ports (P2E to P2H and PYA) on reset

#### (1) On power-on reset

P0E and P0F are set as LCD segment signal output pins and output a low level.

Because the contents of the output latch are undefined, undefined data is output if these ports are set in the output mode as is. Initialize the ports in the program as necessary.

#### (2) On CE reset

P0E and P0F are set as LCD segment signal output pins and output a low level.

Because the contents of the output latch are retained, the previous values are retained if these ports are set in the output mode as is.

#### (3) On execution of clock stop instruction

P0E and P0F are set as LCD segment signal output pins and output a low level.

Because the contents of the output latch are retained, the previous values are retained if these ports are set in the output mode as is.

#### (4) In halt status

The contents of the output latch are output.

Because the contents of the output latch are retained, the output data is not changed in the halt status.

## 11. INTERRUPTS

### 11.1 Outline of Interrupt Block

Figure 11-1 illustrates the interrupt block.

As shown in the figure, the interrupt block temporarily stops the program currently being executed in response to an interrupt request output from any peripheral hardware unit and branches execution to an interrupt vector address.

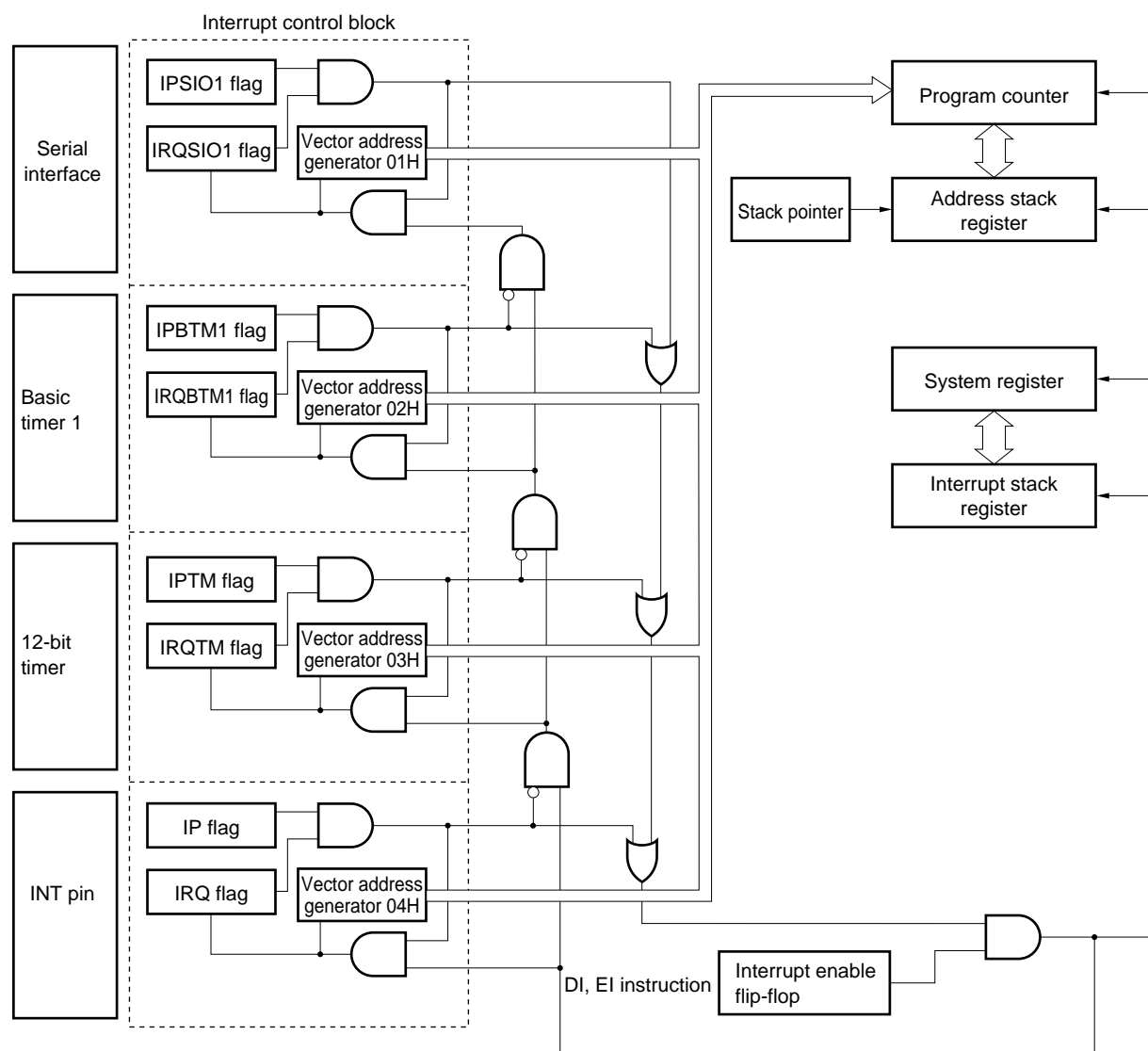
The interrupt block consists of an “interrupt control block” for each peripheral hardware unit, “interrupt enable flip-flop” that enables all the interrupts, “stack pointer” that is controlled when an interrupt is acknowledged, “address stack register”, “program counter”, and “interrupt stack”.

The “interrupt control block” of each peripheral hardware unit consists of an “interrupt request flag (IRQxxx)” that detects each interrupt request, “interrupt enable flag (IPxxx)” that enables each interrupt, and “vector address generator (VAG)” that specifies a vector address when an interrupt is acknowledged.

The peripheral hardware units that have an interrupt function are as follows:

- INT pin (rising-edge detection)
- 12-bit timer
- Basic timer 1
- Serial interface

Figure 11-1. Outline of Interrupt Block





## 11.2 Interrupt Control Block

The interrupt control block is provided for each peripheral hardware unit and detects an interrupt request, enables the interrupt, and generates a vector address when the interrupt is acknowledged.

### 11.2.1 Configuration and function of interrupt request flag (IRQ<sub>xxx</sub>)

Each interrupt request flag (IRQ<sub>xxx</sub>) is set to 1 when an interrupt request is issued from the corresponding peripheral hardware unit, and is reset to 0 when the interrupt is acknowledged. It cannot be set by software.

The “issued” state of each interrupt request can be detected by the detection of these interrupt request flags when interrupts are not enabled.

Also, when “1” is directly written to the interrupt request flag via a window register, it means that the interrupt request has been issued.

Once this flag has been set to “1”, it is not reset until the corresponding interrupt is acknowledged or “0” is written via a window register.

If more than one interrupt request is issued at the same time, the interrupt request flag corresponding to the interrupt that has not been acknowledged is not reset.

The interrupt request flag is assigned to the register file's interrupt request register.

The configuration and function of the interrupt request register are shown in Figures 11-2 to 11-5.

**Figure 11-2. Configuration of Interrupt Request Register 1**

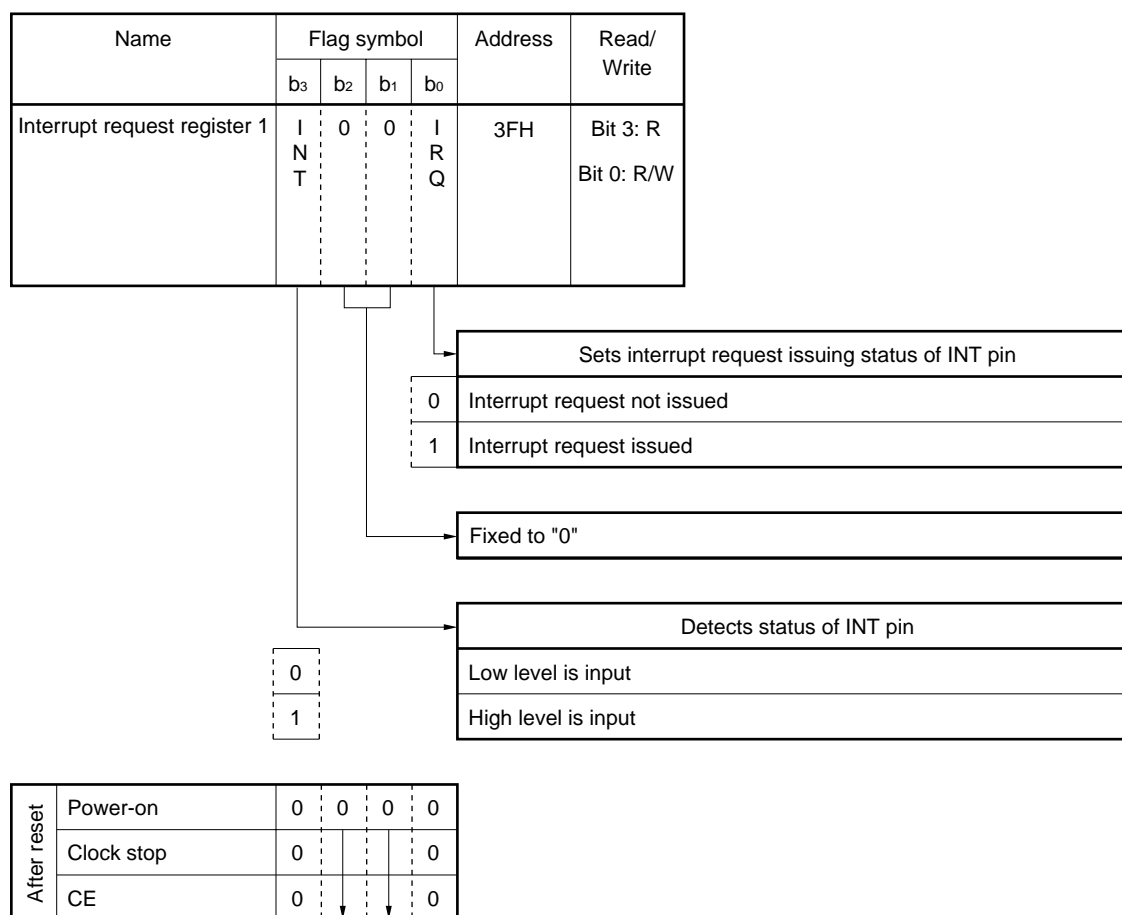


Figure 11-3. Configuration of Interrupt Request Register 2

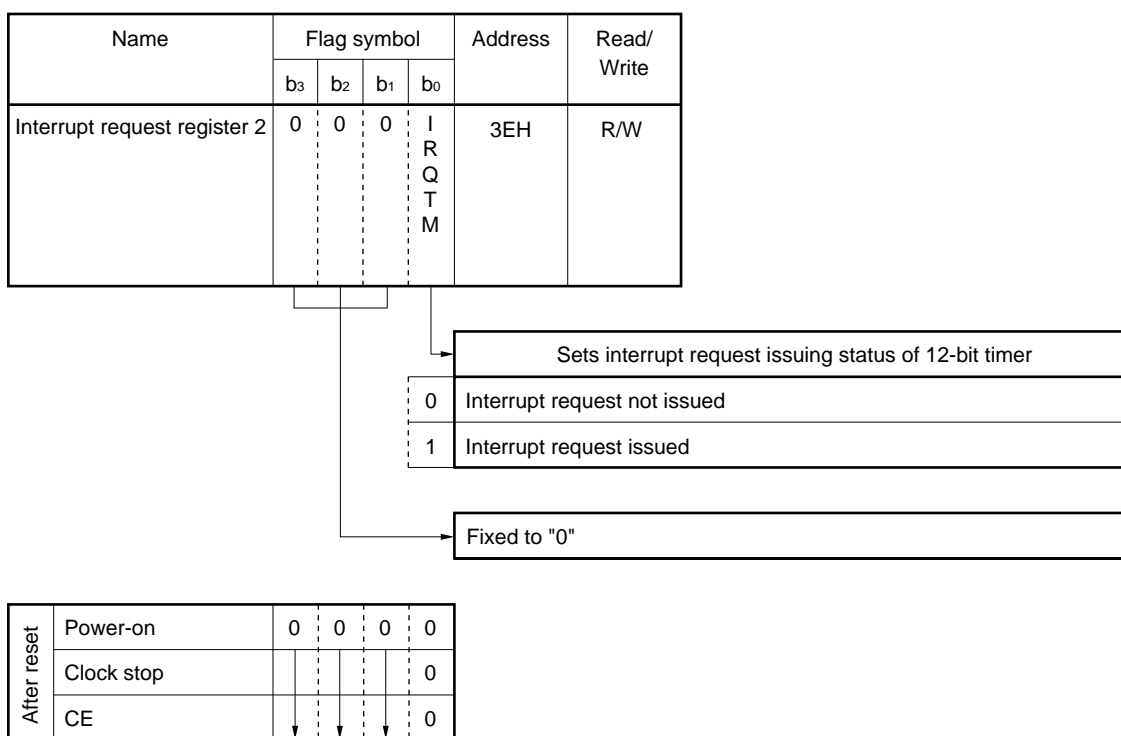


Figure 11-4. Configuration of Interrupt Request Register 3

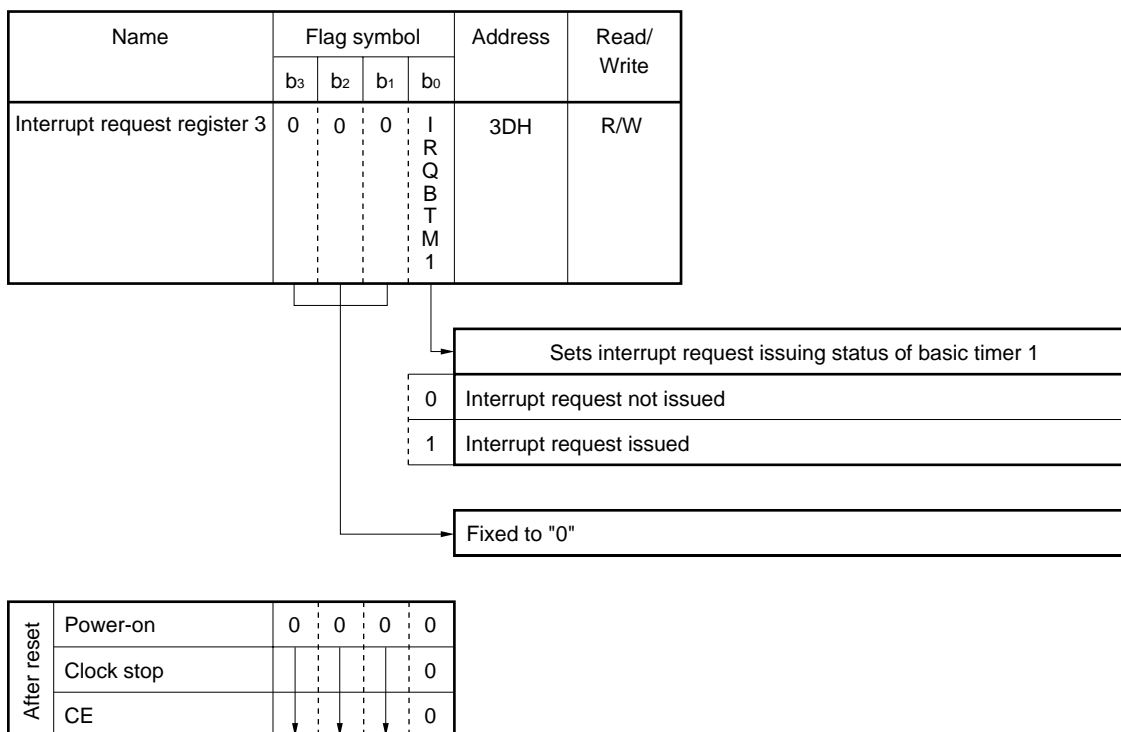
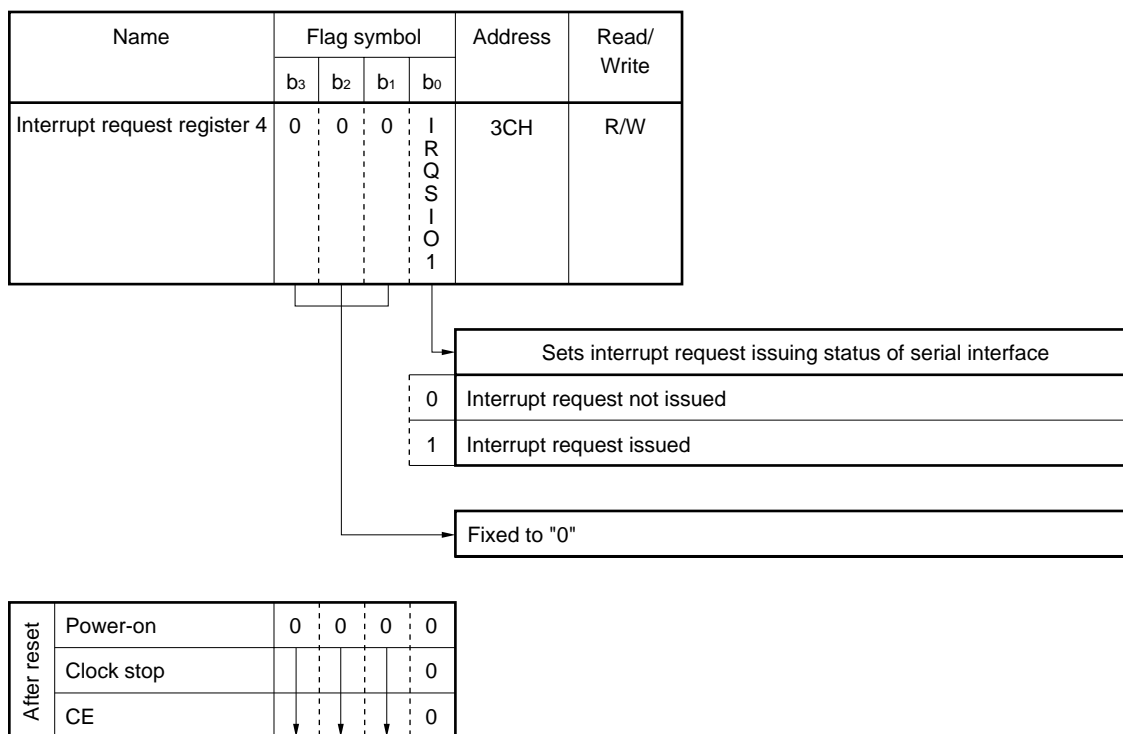


Figure 11-5. Configuration of Interrupt Request Register 4



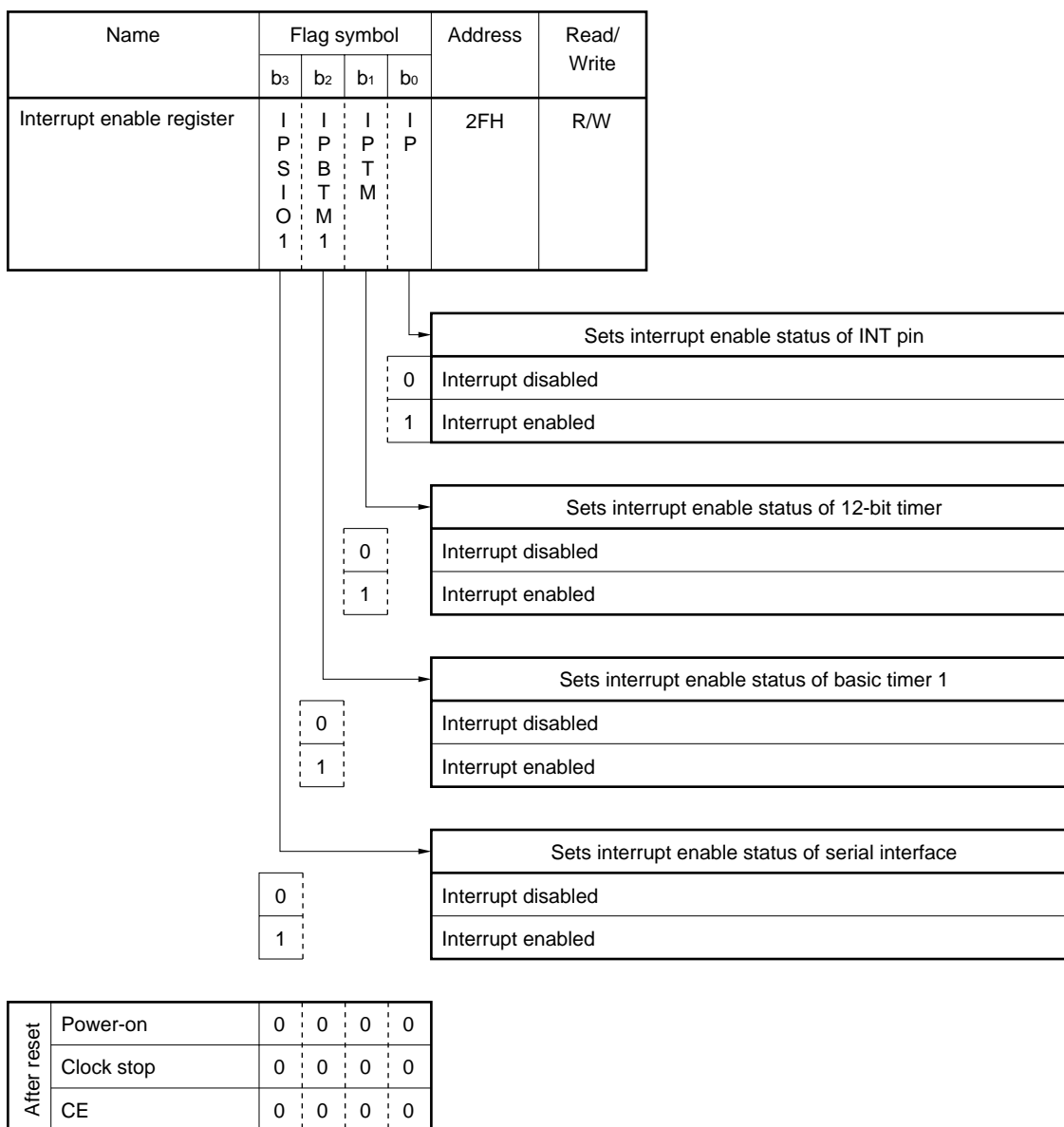
### 11.2.2 Configuration and function of interrupt enable flag (IP<sub>xxx</sub>)

Each interrupt enable flag enables the interrupt of the corresponding peripheral hardware unit. So that an interrupt is acknowledged, all the following three conditions must be satisfied.

- The interrupt must be enabled by the corresponding interrupt enable flag.
- An interrupt request must be issued by the corresponding interrupt request flag.
- The "EI" instruction (that enables all the interrupts) must be executed.

The interrupt enable flag is assigned to the register file's interrupt enable register. Figure 11-6 shows the configuration and function of the interrupt enable register.

Figure 11-6. Configuration of Interrupt Enable Register



### 11.2.3 Vector address generator (VAG)

The vector address generator generates a branch address (vector address) of the program memory for the interrupt source acknowledged when a peripheral hardware interrupt has been acknowledged.

Table 11-1 shows the vector address of each interrupt source.

Table 11-1. Vector Address of Each Interrupt Source

Interrupt Source	Vector Address
INT pin	04H
12-bit timer	03H
Basic timer 1	02H
Serial interface	01H

## 11.3 Interrupt Stack Register

### 11.3.1 Configuration and function of interrupt stack register

Figure 11-7 shows the configuration of the interrupt stack register and the system register whose contents are saved to the interrupt stack register.

The interrupt stack register saves the contents of the following system registers when an interrupt is acknowledged.

- Bank register (BANK)
- General register pointer (RP)
- Program status word (PSWORD)

When an interrupt is acknowledged and the contents of the above system registers are saved to the interrupt stack register, the contents of the above system registers are reset to 0.

The interrupt stack can save up to 2 levels of the contents of the above system registers.

Therefore, up to 2 levels of multiple interrupts can be executed.

The contents of the interrupt stack register are restored to the system registers when an interrupt return instruction ("RETI") is executed.

**Figure 11-7. Configuration of Interrupt Stack Register**

Interrupt Stack Register (INTSK)																
Name	Bank Stack				Register pointer Stack High				Register pointer Stack Low				Status Stack			
Bit	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
0H	—	—			—	—										
1H	—	—			—	—										

**Remark** —: Bit not saved

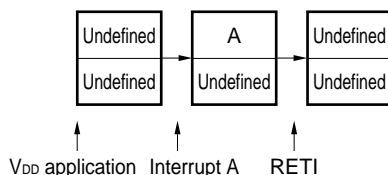
### 11.3.2 Interrupt stack register operation

Figure 11-8 illustrates the operation of the interrupt stack register.

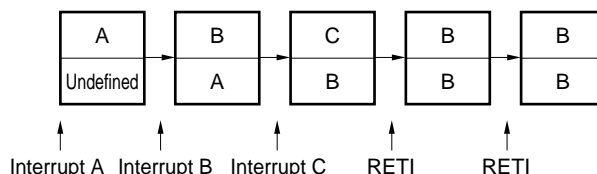
If multiple interrupts exceeding 2 levels are acknowledged, the first saved contents are discarded and therefore must be saved by program.

**Figure 11-8. Operation of Interrupt Stack Register**

**(a) If interrupt does not exceed 2 levels**



**(b) If interrupt exceeds 2 levels**



### 11.4 Stack Pointer, Address Stack Register, Program Counter

The address stack register saves the return address to which execution is to be returned from an interrupt processing routine.

The stack pointer specifies the address of the address stack register.

When an interrupt is acknowledged, therefore, the value of the stack pointer is decremented by one and the value of the program counter at that time is saved to the address stack register specified by the stack pointer.

When the dedicated return instruction RETI is executed after the processing of the interrupt servicing routine has been executed, the contents of the address stack register specified by the stack pointer are restored to the program counter, and the value of the stack pointer is incremented by one.

For further information, also refer to **3. ADDRESS STACK (ASK)**.

### 11.5 Interrupt Enable Flip-Flop (INTE)

The interrupt enable flip-flop enables all the interrupts.

When this flip-flop is set, all the interrupts are enabled. When it is reset, all the interrupts are disabled.

This flip-flop is set or reset by using dedicated instructions EI (to set) and DI (to reset).

The EI instruction sets this flip-flop when the instruction next to the EI instruction is executed, and the DI instruction resets the flip-flop while the DI instruction is executed.

When an interrupt is acknowledged, this flip-flop is automatically reset.

Nothing is affected even if the DI instruction is executed in the DI state, or if the EI instruction is executed in the EI state.

This flip-flop is reset on power-on reset, CE reset, and on execution of the clock stop instruction.

## 11.6 Acknowledging Interrupts

### 11.6.1 Acknowledging interrupts and priority

An interrupt is acknowledged in the following procedure:

- (1) Each peripheral hardware unit outputs an interrupt request signal to the corresponding interrupt control block if a given interrupt condition is satisfied (e.g., if a rising signal is input to the INT pin).
- (2) When the interrupt control block has received the interrupt request signal from the peripheral hardware unit, it sets the corresponding interrupt request flag (e.g., IRQ flag if the peripheral unit is the INT pin) to 1.
- (3) If the interrupt enable flag corresponding to the interrupt request flag (e.g., IP flag for IRQ flag) is set to 1 when the interrupt request flag is set to 1, the interrupt control block outputs "1".
- (4) The signal output by the interrupt control block is ORed with the output of the interrupt enable flip-flop, and an interrupt acknowledge signal is output.

This interrupt enable flip-flop is set to 1 by the EI instruction and reset to 0 by the DI instruction.

If the interrupt control block outputs "1" while the interrupt enable flip-flop is "1", the interrupt is acknowledged.

As shown in Figure 11-1, the interrupt acknowledge signal is input to each interrupt control block when the interrupt has been acknowledged.

The interrupt request flag is reset to 0 by the signal input to the interrupt control block, and a vector address corresponding to the interrupt is output.

If more than one interrupt block outputs "1" at this time, the interrupt acknowledge signal is not transferred to the next stage. If more than one interrupt request is issued at the same time, therefore, the interrupts are acknowledged in the following priority order.

INT pin > 12-bit timer > basic timer 1 > serial interface

The interrupt of an interrupt source is not acknowledged unless the corresponding interrupt enable flag is set to 1.

If the interrupt enable flag is reset to 0, therefore, an interrupt with a high hardware priority can be disabled.

### 11.6.2 Timing chart for acknowledging interrupt

Figure 11-9 shows the timing chart illustrating acknowledging interrupts.

(1) in this figure illustrates how one interrupt is acknowledged.

(a) in (1) shows the case where the interrupt request flag is the last to be set to 1, and (b) in (1) shows the case where the interrupt enable flag is the last to be set to 1.

In either case, the interrupt is acknowledged when each of the interrupt request flag, interrupt enable flip-flop, and interrupt enable flag are set to 1.

If the last flag or flip-flop that was set to 1 satisfies the first instruction cycle of the MOV<sub>T</sub> DBF, @AR instruction or a given skip condition, the interrupt is acknowledged after the second instruction cycle of the MOV<sub>T</sub> DBF, @AR instruction or the instruction that is skipped (NOP) has been executed.

The interrupt enable flip-flop is set in the instruction cycle next to the one in which the EI instruction is executed.

(2) in Figure 11-9 illustrates how more than one interrupt is used.

In this case, the interrupts are sequentially acknowledged according to the hardware priority if all the interrupt enable flags are set. The hardware priority can be changed by manipulating the interrupt enable flag by program.

“Interrupt cycle” shown in Figure 11-9 is a special cycle in which the interrupt request flag is reset, a vector address is specified, and the contents of the program counter are saved after an interrupt has been acknowledged, and lasts for 4.44  $\mu$ s, which is equivalent to the execution time of one instruction.

For further information, refer to **11.7 Operations After Acknowledging Interrupt**.

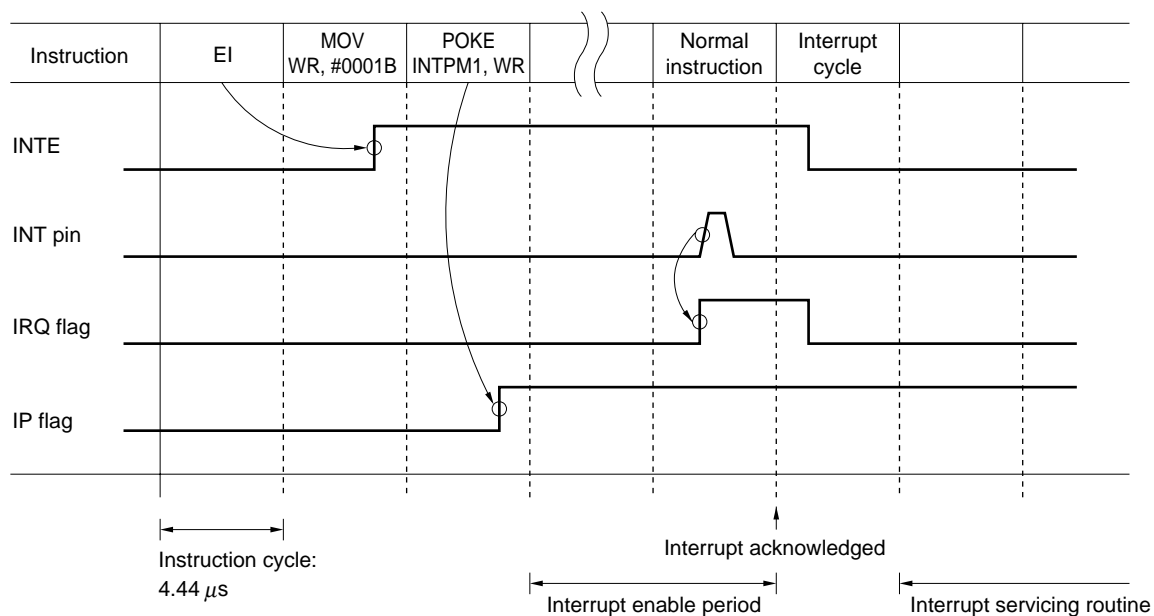


Figure 11-9. Timing Chart of Acknowledging Interrupt (1/3)

## (1) When one interrupt (e.g., rising of INT pin) is used

(a) If interrupt is not masked by interrupt enable flag (IP<sub>xxx</sub>)

<1> If the MOVT instruction or a normal instruction that does not satisfies the skip condition is executed when an interrupt is acknowledged



<2> If the MOVT instruction or an instruction that satisfies the skip condition is executed when an interrupt is acknowledged

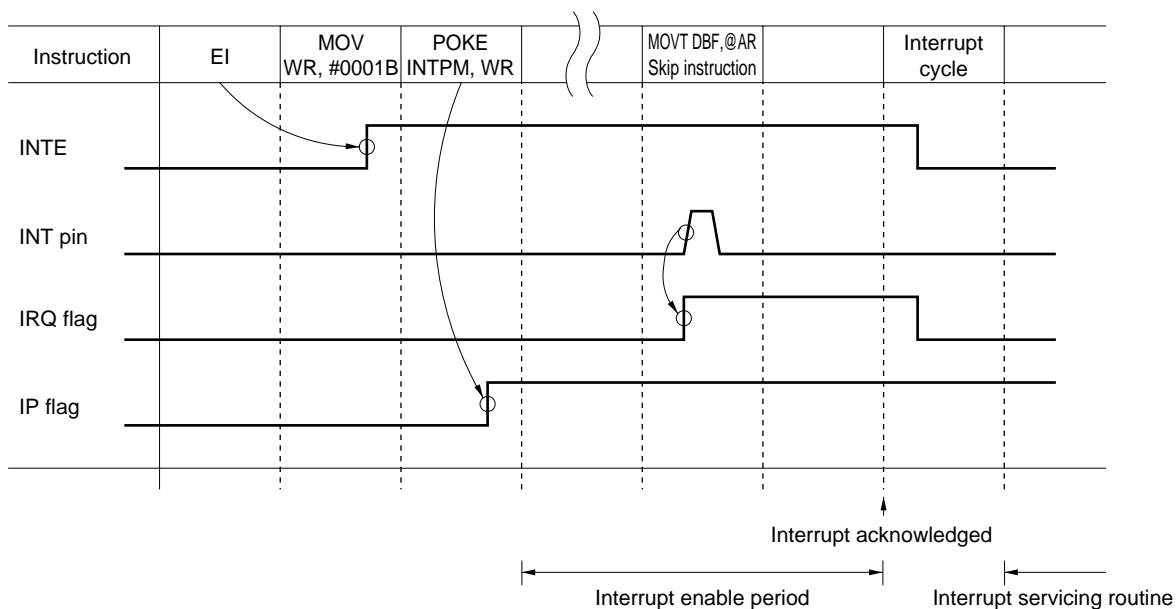


Figure 11-9. Timing Chart of Acknowledging Interrupt (2/3)

(b) If interrupt is kept pending by interrupt enable flag

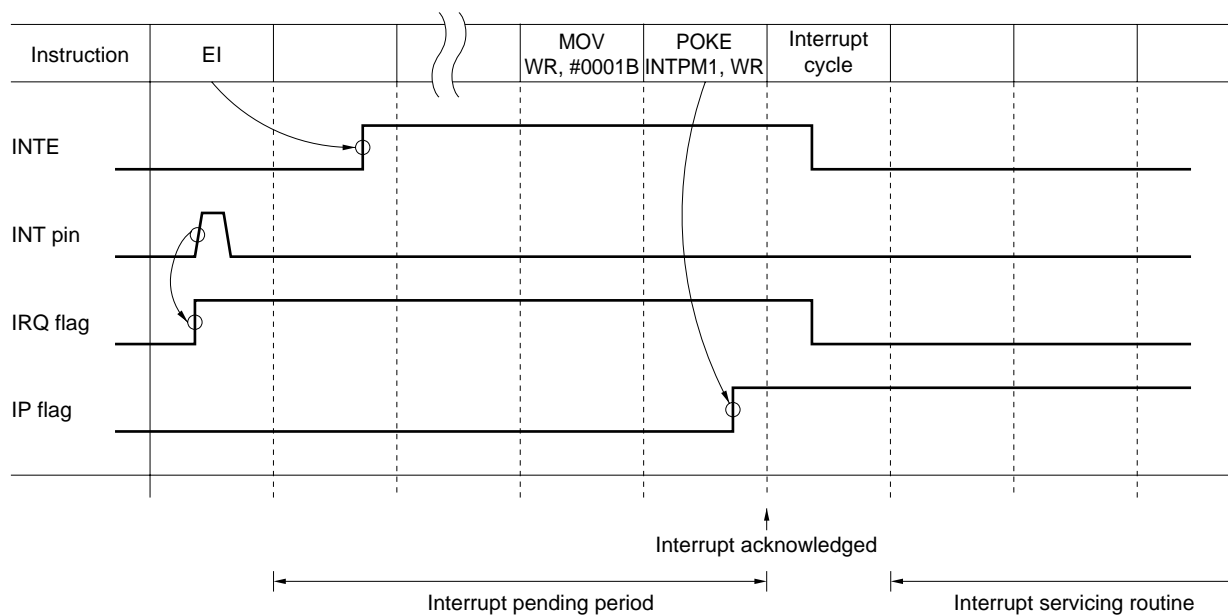
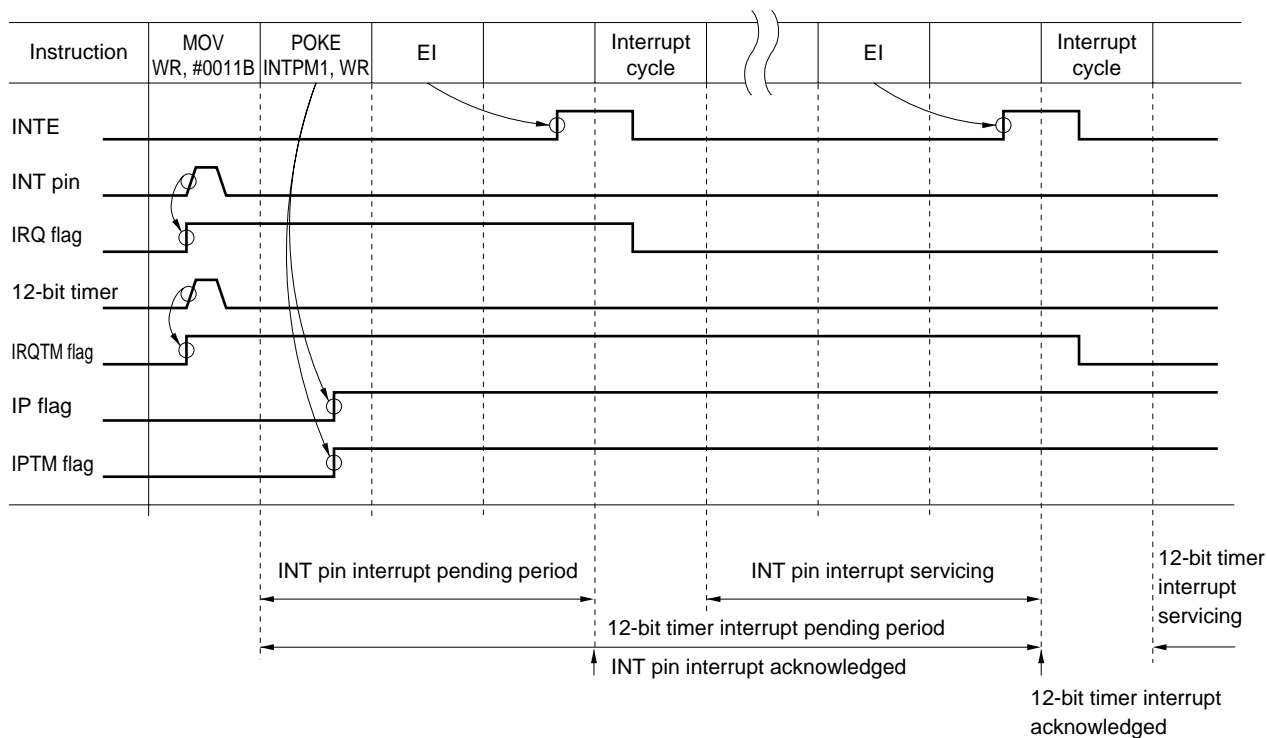


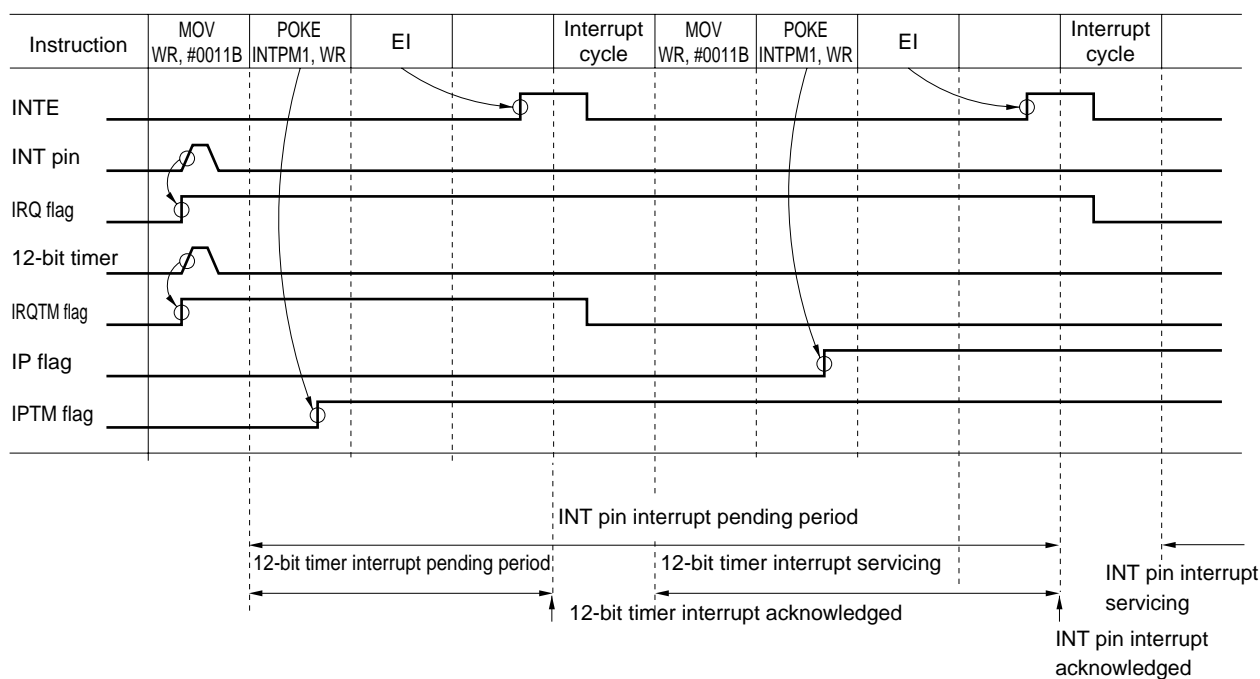
Figure 11-9. Timing Chart of Acknowledging Interrupt (3/3)

(2) When two or more interrupts are used (e.g. INT pin and 12-bit timer)

(a) Hardware priority



(b) Software priority



### 11.7 Operations After Acknowledging Interrupt

When an interrupt has been acknowledged, the following processing is sequentially executed.

- (1) The interrupt enable flip-flop and the interrupt request flag corresponding to the acknowledged interrupt are reset to 0, disabling the interrupts.
- (2) The contents of the stack pointer are decremented by one.
- (3) The contents of the program counter are saved to the address stack register specified by the stack pointer. The contents saved at this time are the next program memory address that is used after the interrupt has been acknowledged. For example, if a branch instruction is executed, the contents saved are the branch destination address; if a subroutine call instruction is executed, they are the called address. Because the interrupt is acknowledged after the next instruction is executed as a NOP instruction if a skip condition is satisfied by a skip instruction, the saved contents are the skipped address.
- (4) The lower 2 bits of the bank register (BANK), lower 5 bits of the general register pointer (RP), and 5 bits of the program status word (PSWORD) are saved to the interrupt stack.
- (5) The contents of the vector address generator corresponding to the acknowledged interrupt are transferred to the program counter. In other words, execution branches to an interrupt servicing routine.

The processing (1) through (5) above is executed in one special instruction cycle (4.44  $\mu$ s) in which the normal instruction is not executed. This instruction cycle is called an interrupt cycle.

In other words, one instruction cycle time is necessary since an interrupt has been acknowledged until execution branches to the corresponding vector address.

### 11.8 Restoring from Interrupt Servicing Routine

To restore execution from an interrupt servicing routine to the processing that was being performed when the interrupt occurred, a dedicated instruction, "RETI", is used.

When the RETI instruction is executed, the following processing is sequentially executed.

- (1) The contents of the address stack register specified by the stack pointer are saved to the program counter.
- (2) The contents of the interrupt stack are restored to the lower 2 bits of the bank register (BANK), lower 5 bits of the general register pointer (RP), and 5 bits of the program status word (PSWORD).
- (3) The contents of the stack pointer are incremented by one.

The processing (1) through (3) above is executed in one instruction cycle in which the RETI instruction is executed.

The difference between the RETI instruction and subroutine return instructions "RET" and "RETSK" is only the restoring operation of the system register in (2) above.

## 11.9 External (INT Pin) Interrupt

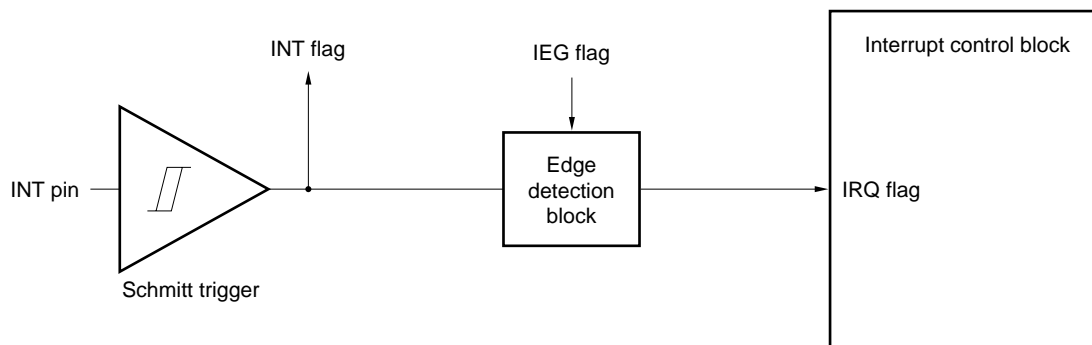
### 11.9.1 Outline of external interrupt

Figure 11-10 illustrates the external interrupt.

As shown in the figure, the external interrupt issues an interrupt request at the rising edge of the signal input to the INT pin.

The INT pin is a Schmitt trigger input pin to prevent malfunctioning due to noise, and does not accept a pulse less than 1 μs wide.

**Figure 11-10. Outline of External Interrupt**



**Remark** INT: Detects pin status  
 IEG: Selects interrupt edge

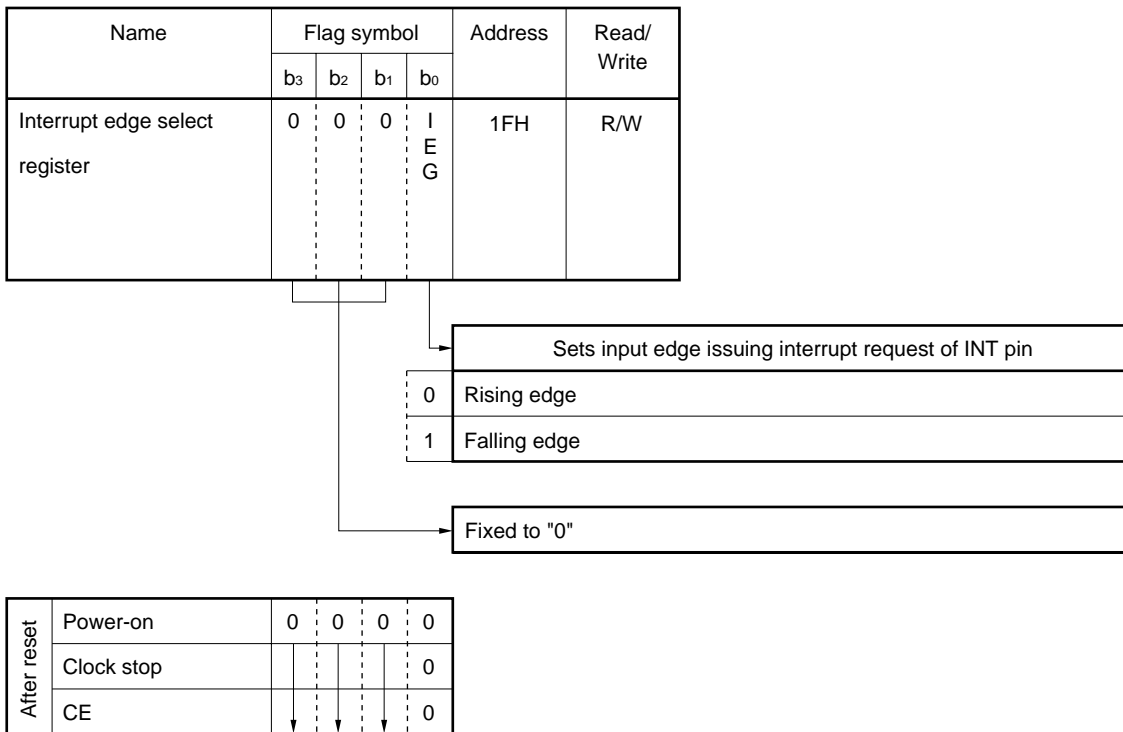
### 11.9.2 Edge detection block

The edge detection block sets and detects the input signal edge (rising or falling edge) and that issues the interrupt request of the INT pin.

The edge setting is made by the IEG flag.

Figure 11-11 shows the configuration and function of the interrupt edge select register.

Figure 11-11. Configuration of Interrupt Edge Select Register



Note that as soon as the interrupt request issuing edge is changed by the IEG flag, the interrupt request signal may be issued.

Suppose that the IEG flag is set to 1 (specifying the falling edge) and that a high level is input to the INT pin, as shown in Table 11-2. If the IEG flag is reset to 0 at this time, the edge detector judges that a rising edge has been input, and issues an interrupt request.

Table 11-2. Issuing Interrupt Request by Changing IEG Flag

Changes in IEG Flag	INT Pin Status	Interrupt Request	IRQ Flag Status
1 → 0 (falling) (rising)	Low level	Not issued	Retains previous status
	High level	Issued	Set to 1
0 → 1 (rising) (falling)	Low level	Issued	Set to 1
	High level	Not issued	Retains previous status

### 11.9.3 Interrupt control block

The level of a signal input to the INT pin can be detected by using the INT flag.

This flag is set or reset independently of interrupts; therefore, it can be used as a 1-bit general-purpose input port when the interrupt function is not used.

The INT flag can also be used as a general-purpose port that can detect the rising or falling edge by reading an interrupt request flag if the interrupt corresponding to the flag is not enabled.

In this case, however, the interrupt request flag is not automatically reset and must be reset by program.

Also refer to **11.2.1 Interrupt request flag (IRQxxx)**.

## 11.10 Internal Interrupt

Three internal interrupt sources, 12-bit timer, basic timer 1, and serial interface, are available.

### 11.10.1 Interrupt by 12-bit timer

This interrupt request is issued at fixed time intervals.

For details, refer to **12. TIMER**.

### 11.10.2 Interrupt by basic timer 1

This interrupt request is issued at fixed time intervals.

For details, refer to **12. TIMER**.

### 11.10.3 Interrupt by serial interface

This interrupt request is issued when a serial output or serial input operation has been completed.

For details, refer to **15. SERIAL INTERFACE**.

## 12. TIMER

The timers are used to control the program execution time.

### 12.1 General

Figure 12-1 illustrates the timers of the  $\mu$ PD17012.

As shown in this figure, the  $\mu$ PD17012 is provided with the following three timers:

- Basic timer 0
- Basic timer 1
- 12-bit timer (modulo timer)

Basic timer 0 is used to detect the status of a flip-flop that is set at fixed time intervals.

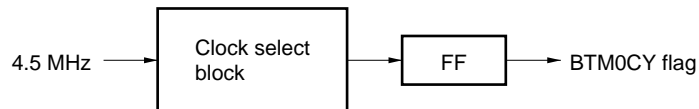
Basic timer 1 is used to issue an interrupt request at fixed time intervals.

The 12-bit timer is a modulo timer that issues an interrupt request at fixed time intervals.

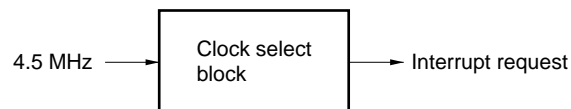
Basic timer 0 can also be used to detect a power failure. The clock of each timer is generated by dividing the system clock (4.5 MHz).

**Figure 12-1. Outline of Timer**

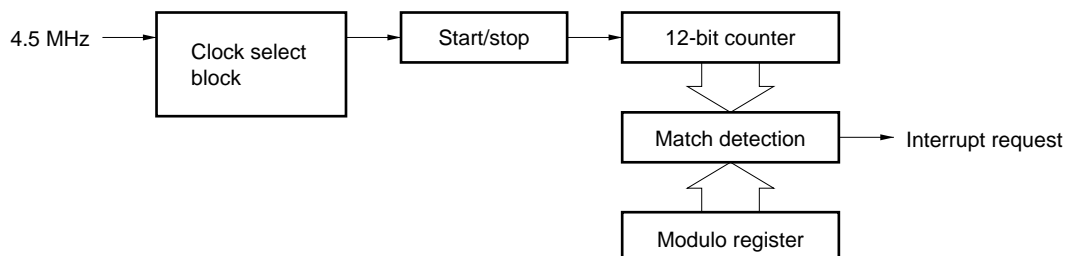
#### (a) Basic timer 0



#### (b) Basic timer 1



#### (c) 12-bit timer





## 12.2 Basic Timer 0

### 12.2.1 Outline of basic timer 0

Figure 12-2 illustrates basic timer 0.

Basic timer 0 is used as a timer by detecting the status of a flip-flop that is set at fixed intervals (100, 250, 5, or 1 ms), using the BTM0CY flag (RF: address 17H, bit 0).

The contents of the flip-flop correspond to the BTM0CY flag.

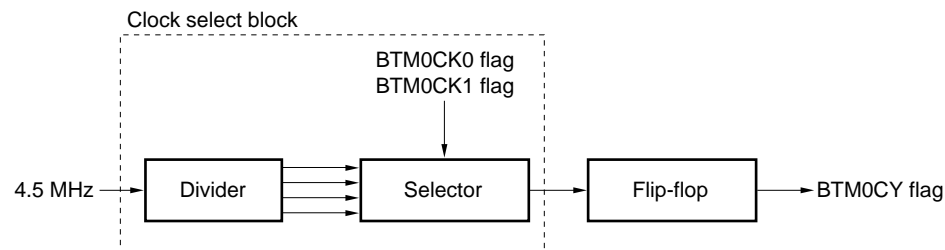
If the BTM0CY flag is read first after power-on reset, "0" is always read. After that, the flag is set to 1 at fixed intervals.

If the CE pin goes high from low, CE reset is effected in synchronization with the timing at which the BTM0CY flag is set next.

Therefore, a power failure can be detected by reading the contents of the BTM0CY flag at system reset (power-on reset or CE reset).

For the details of power failure detection, refer to **22. RESET**.

**Figure 12-2. Outline of Basic Timer 0**



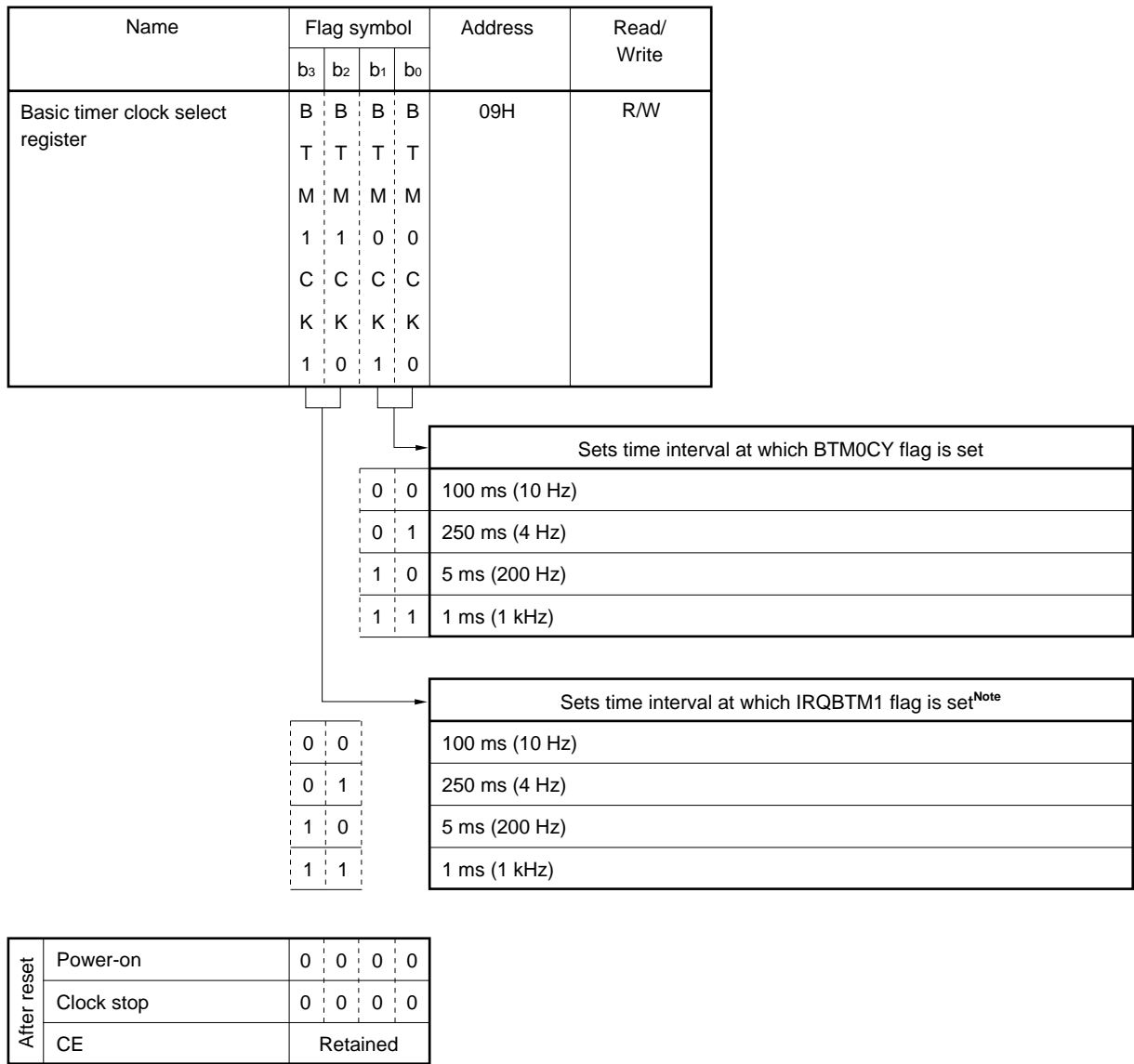
- Remarks**
1. BTM0CK1 and BTM0CK0 (bits 1 and 0 of the basic timer clock select register: refer to **Figure 12-3**) set the time intervals at which the BTM0CY flag is set.
  2. BTM0CY (bit 0 of the basic timer 0 carry FF judge register: refer to **Figure 12-4**) detects the status of the flip-flop.

12.2.2 Clock select block

The clock select block divides the system clock (4.5 MHz) and sets the time interval at which the BTM0CY flag is to be set, by using the basic timer clock select register.

Figure 12-3 shows the configuration of the basic timer clock select register.

Figure 12-3. Configuration of Basic Timer Clock Select Register



**Note** For Basic timer 1, refer to 12.3.

### 12.2.3 Flip-flop and BTM0CY flag

The flip-flop is set at fixed intervals and its status is detected by the BTM0CY flag of the basic timer 0 carry FF judge register.

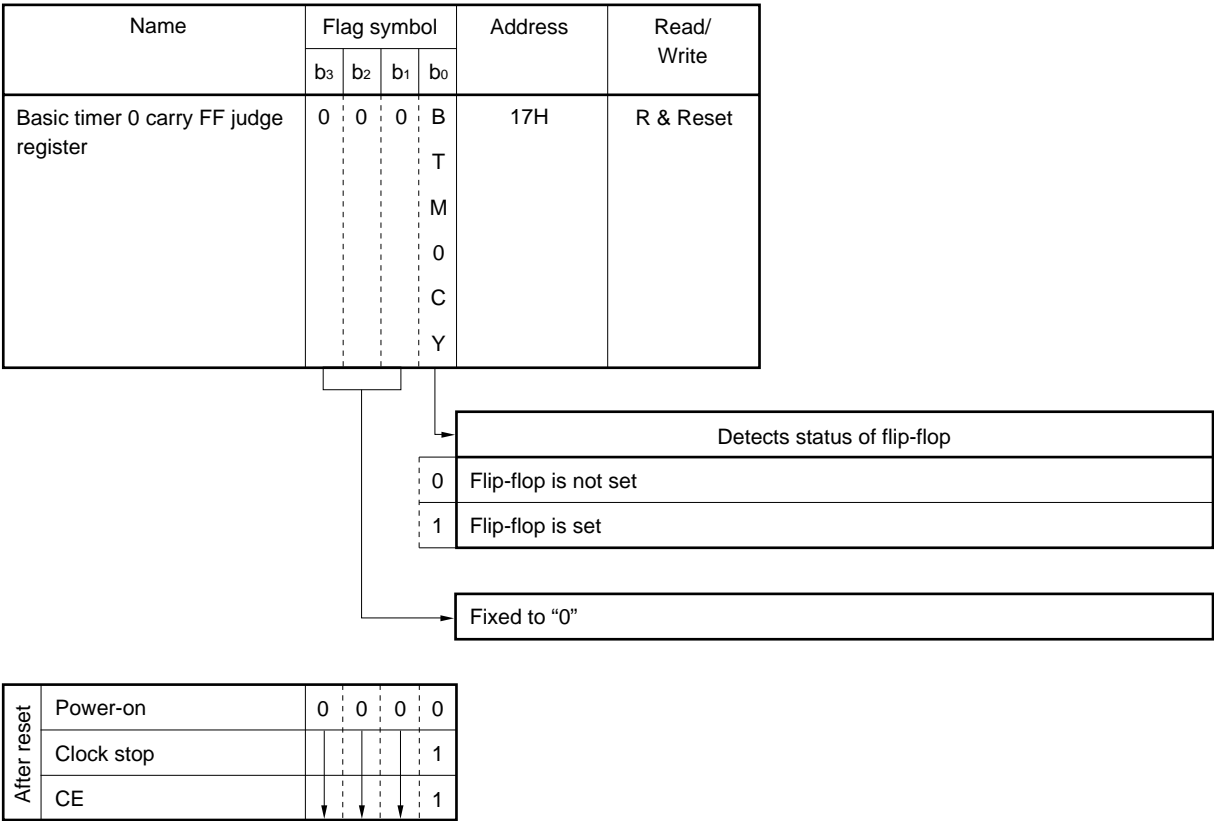
When the BTM0CY flag reads out its contents to the window register by PEEK instruction execution, it is reset to 0 (Read & Reset).

The BTM0CY flag is “0” at power-ON reset, and is “1” at CE reset and on execution of the clock stop instruction. Therefore, this flag can be used to detect a power failure.

The BTM0CY flag is not set after power application until an instruction that reads it is executed. Once the read instruction has been executed, the flag is set at fixed intervals.

Figure 12-4 shows the configuration of the basic timer 0 carry FF judge register.

**Figure 12-4. Configuration of Basic Timer 0 Carry FF Judge Register**



### 12.2.4 Example of using basic timer 0

An example of a program using basic timer 0 is shown below.

This program executes processing A every 1 second.

#### Example

```

CLR2  BTM0CK1, BTM0CK0  ; Sets BTM0CY flag setting pulse to 10 Hz (100 ms)
MOV   M1, #0
LOOP:
SKT1  BTM0CY              ; Branches to NEXT if BTM0CY flag is "0"
BR    NEXT
ADD   M1, #1              ; Adds 1 to M1
SKE   M1, #0AH            ; Executes processing A if M1 is "10" (1 second has elapsed)
BR    NEXT
MOV   M1, #0



Processing A



NEXT:



Processing B

      ; Executes processing B and branches to LOOP

BR    LOOP
    
```

### 12.2.5 Errors of basic timer 0

Errors of basic timer 0 include an error due to the detection time of the BTM0CY flag, and an error that occurs when the time interval at which the BTM0CY flag is to be set is changed.

The following paragraphs (1) and (2) describe each error.

#### (1) Error due to detection time of BTM0CY flag

The time to detect the BTM0CY flag must be shorter than the time at which the BTM0CY flag is set (refer to 12.2.6 Notes on using basic timer 0).

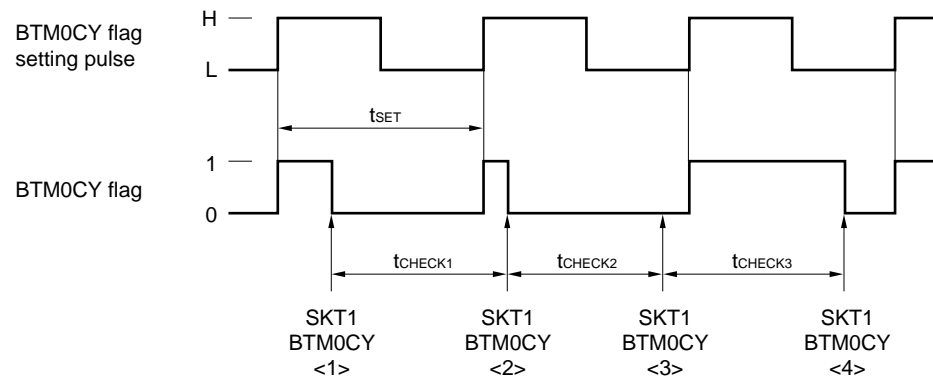
Where the time interval at which the BTM0CY flag is detected is  $t_{CHECK}$  and the time interval at which the flag is set is  $t_{SET}$  (250, 10, 5, or 1 ms),  $t_{CHECK}$  and  $t_{SET}$  must relate as follows.

$$t_{CHECK} < t_{SET}$$

At this time, the error of the timer when the BTM0CY flag is detected is as follows, as shown in Figure 12-5.

$$0 < \text{Error} < t_{SET}$$

**Figure 12-5. Error of Basic Timer 0 due to Detection Time of BTM0CY Flag**



As shown in Figure 12-5, the timer is updated because BTM0CY flag is "1" when it is detected in step <2>.

When the flag is detected next in step <3>, it is "0". Therefore, the timer is not updated until the flag is detected again in <4>.

This means that the timer is extended by the time of  $t_{CHECK3}$ .

## (2) Error when time interval to set BTM0CY flag is changed

The BTM0CK1 and BTM0CK0 flags set the time of the BTM0CY flag.

As described in 12.2.2, four types of timer time-setting pulses can be selected: 1 kHz, 200 Hz, 10 Hz, and 4 Hz.

At this time, these four pulses operate independently. If the timer time-setting pulse is changed by using the BTM0CK1 and BTM0CK0 flags, an error occurs as described in the example below.

### Example

```
; <1>
INITFLG BTM0CK1, NOT BTM0CK0           ; Sets BTM0CY flag setting pulse to 200 Hz (5 ms)
```

Processing A

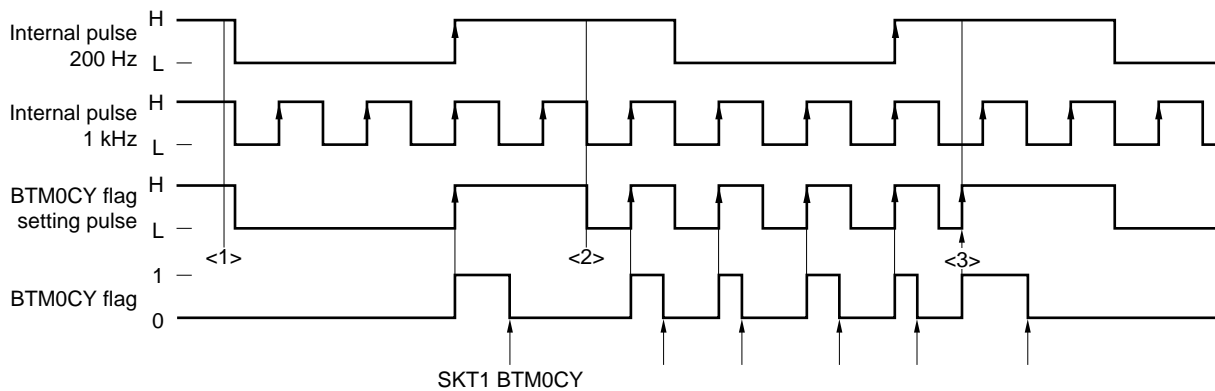
```
; <2>
INITFLG BTM0CK1, BTM0CK0               ; Sets BTM0CY flag setting pulse to 1 kHz (1 ms)
```

Processing A

```
; <3>
INITFLG BTM0CK1, NOT BTM0CK0           ; Sets BTM0CY flag setting pulse to 200 Hz (5 ms)
```

At this time, the BTM0CY flag setting pulse is changed as shown in Figure 12-6.

**Figure 12-6. Changing BTM0CY Flag Setting Pulse**



As shown in Figure 12-6, if the BTM0CY flag setting time is changed and the new pulse falls, the BTM0CY flag retains the previous status (<2> in the figure). If the new pulse rises, however, the BTM0CY flag is set to 1 (<3> in the figure).

Although changing the pulse setting between 200 Hz (5 ms) and 1 kHz (1 ms) is described in this example, the same applies to changing the pulse in respect to 4 Hz (250 ms) and 10 Hz (100 ms).

Therefore, as shown in Figure 12-7, the error of the time until the BTM0CY flag is first set after the BTM0CY flag setting time has been changed is as follows:

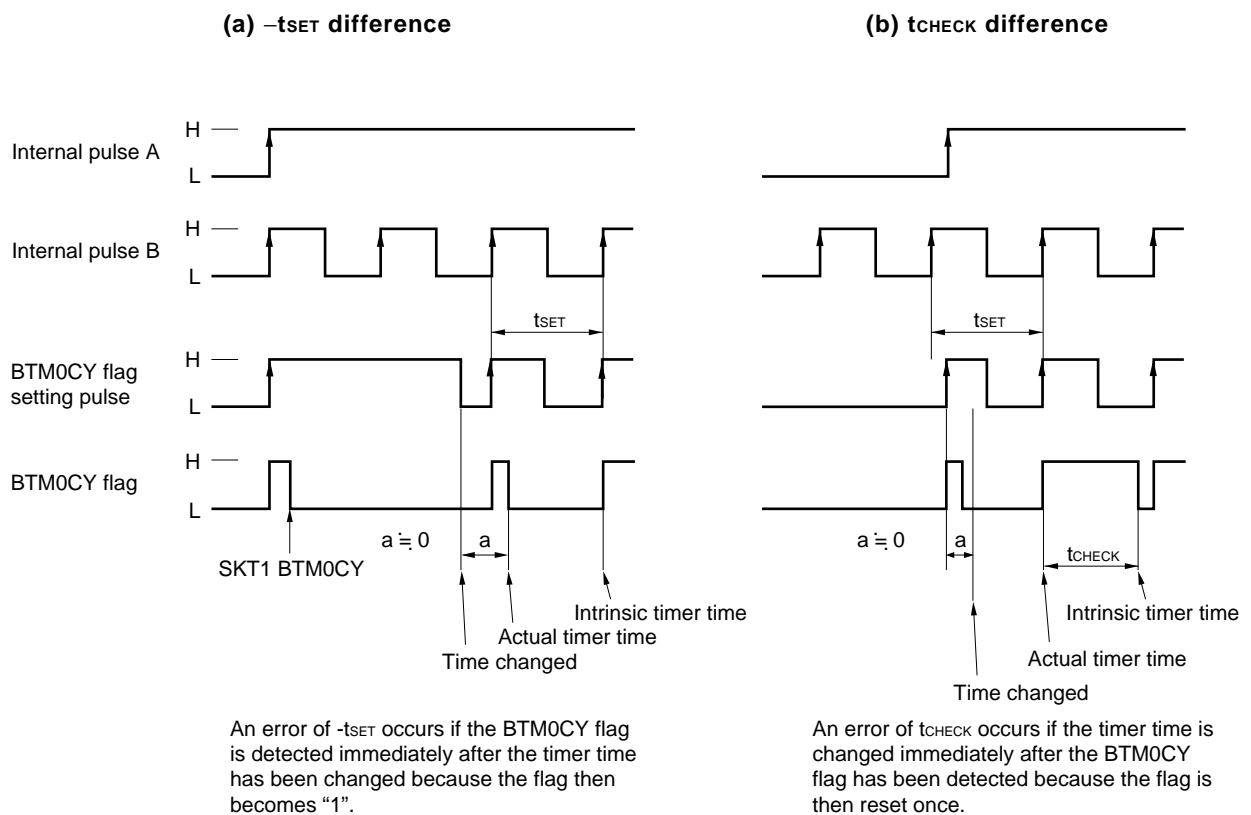
$$-t_{SET} < \text{Error} < t_{CHECK}$$

$t_{SET}$ : New setting time of BTM0CY flag

$t_{CHECK}$ : Time to detect BTM0CY flag

Phase differences are provided among the internal pulses of 4, 10, 200 Hz, and 1 kHz. Because these phase differences are shorter than the newly set pulse time, they are included in the above error. For the phase difference of each pulse, refer to **12.3.5 Notes on using basic timer 1**.

**Figure 12-7. Timer Error When BTM0CY Flag Setting Time Is Changed from A to B**

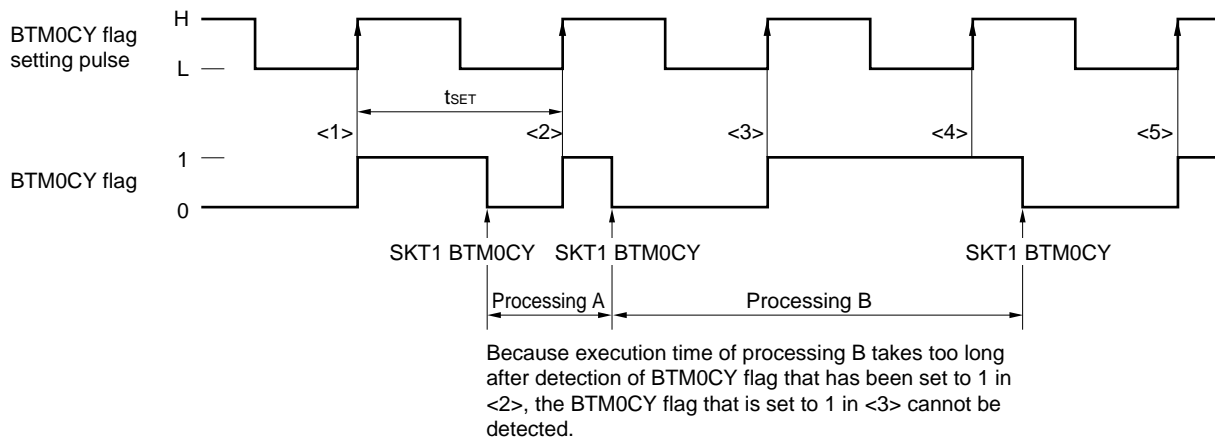


### 12.2.6 Cautions on using basic timer 0

#### (1) BTM0CY flag detection time interval

Keep the time to detect the BTM0CY flag shorter than the time at which the BTM0CY flag is set. This is because if the time of processing B is longer than the time interval at which the BTM0CY flag is set as shown in Figure 12-8, setting of the BTM0CY flag is overlooked.

Figure 12-8. BTM0CY Flag Detection and BTM0CY Flag



#### (2) Timer updating processing time and BTM0CY flag detection time interval

As described in (1) above, time interval  $t_{SET}$  at which the BTM0CY flag is detected must be shorter than the time for which to set the BTM0CY flag.

At this time, even if the time interval at which the BTM0CY flag is detected is short, if the updating processing time of the timer is long the processing of the timer may not be executed normally at CE reset. Therefore, the following condition must be satisfied.

$$t_{CHECK} + t_{TIMER} < t_{SET}$$

$t_{CHECK}$ : Time to detect BTM0CY flag  
 $t_{TIMER}$ : Timer updating processing time  
 $t_{SET}$ : Time to set BTM0CY flag

An example is given below.



**Example** Example of timer updating processing and BTM0CY flag detection time interval

```

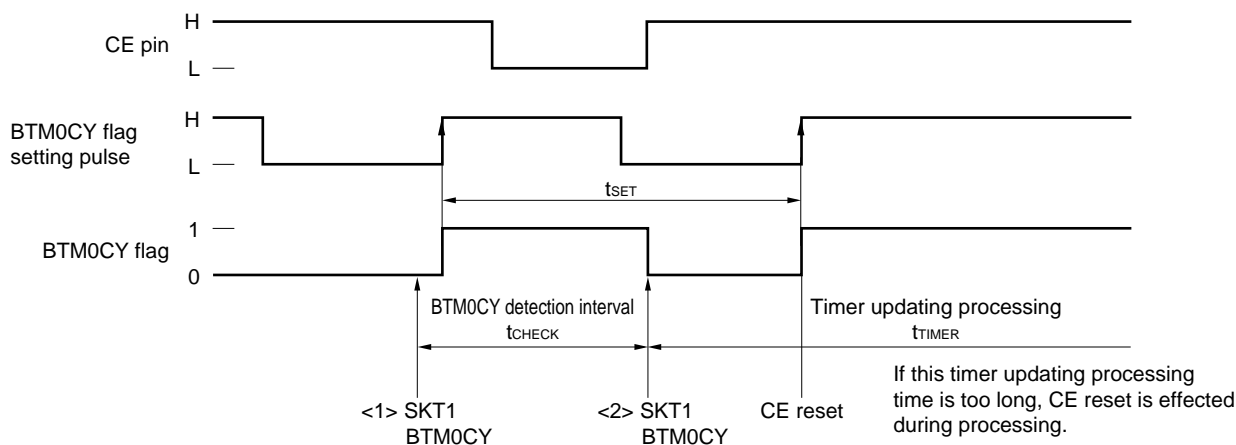
START:
    CLR2    BTMOCK1, BTMOCK0    ; Sets BTM0CY flag setting pulse to 10 Hz (100 ms)
BTIMER:
    ; <1>
    SKT1    BTM0CY              ; Updates timer if BTM0CY flag is "1"
    BR      AAA                ; Branches to AAA if BTM0CY flag is "0"

    [Timer updating]

    BR      BTIMER
AAA:
    [Processing A]

    BR      BTIMER
  
```

The timing chart of the above program is shown below.



### (3) Correcting basic timer 0 carry at CE reset

Next, an example of correcting the timer at CE reset is described below.

As shown in the example below, the timer must be corrected at CE reset if the BTM0CY flag is used for power failure detection and if the BTM0CY flag is used for a watch timer.

The BTM0CY flag is reset (to 0) first on power application (power-on reset), and is disabled from being set until it is read once by the PEEK instruction.

If the CE pin goes high from low, a CE reset is effected in synchronization with the rising edge of the BTM0CY flag setting pulse. At this time, the BTM0CY flag is set (to 1) and the timer is started.

By detecting the status of the BTM0CY flag at system reset (power-on reset or CE reset), therefore, it can be identified whether a power-on reset or CE reset has been effected (power failure detection). That is, a power-on reset has been effected if the flag is "0", and a CE reset has been effected if it is "1". At this time, the watch timer must continue operating even if a CE reset has been effected.

However, because the BTM0CY flag is reset to 0 when it is read to detect a power failure, the set status (1) of the BTM0CY flag is overlooked once.

Consequently, the watch timer must be updated if a CE reset is identified by means of power failure detection.

For the details of power failure detection, refer to **22. RESET**.

**Example**

Example of correcting timer at CE reset (to detect power failure and update watch timer using BTM0CY flag)

```

START:                                ; Program address 0000H

    Processing A

    ; <1>
    SKT1 BTM0CY                       ; Embedded macro
                                       ; Tests BTM0CY flag
    BR    INITIAL                     ; if "0", branches to INITIAL (power failure detection)
BACKUP:
    ; <2>

    100 ms watch updating             ; Corrects watch timer because of backup (CE reset)

LOOP:
    ; <3>

    Processing B                       ; While performing processing B,

    SKF1 BTM0CY                       ; tests BTM0CY flag and updates watch timer
    BR    BACKUP
    BR    LOOP
INITIAL:
    CLR2 BTM0CK1, BTM0CK0
                                       ; Embedded macro
                                       ; Because power failure (power-ON reset) occurs,
                                       ; sets setting time of BTM0CY flag to 100 ms, and
                                       ; executes processing C

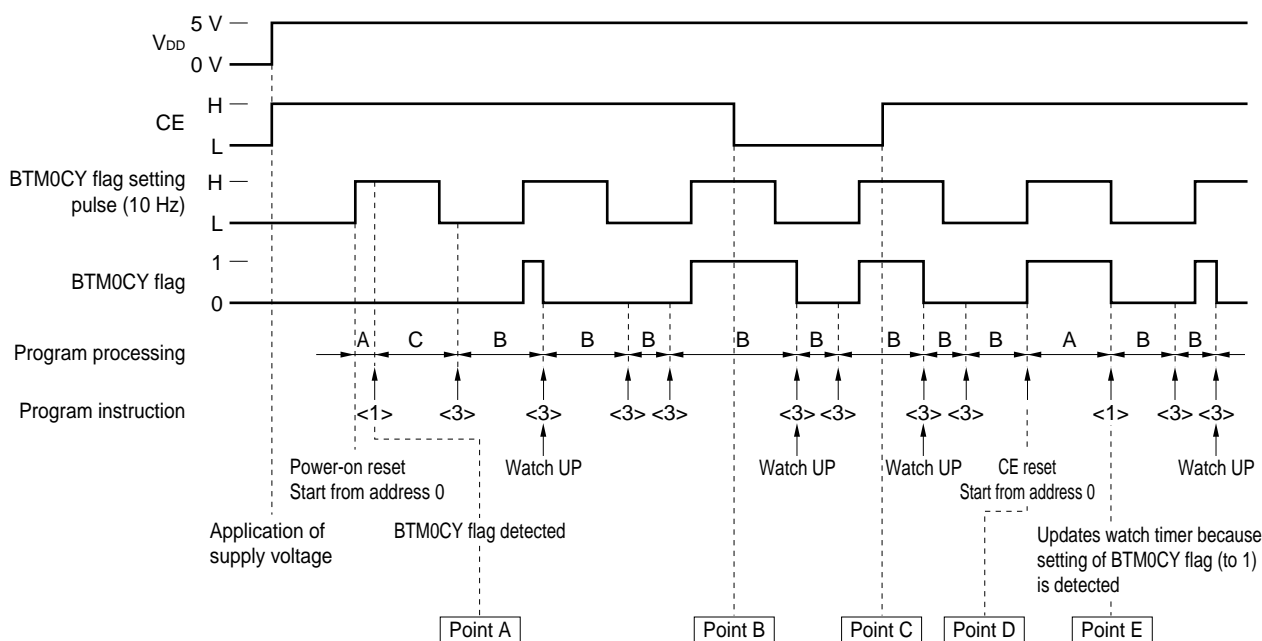
    Processing C

    BR    LOOP

```

Figure 12-9 shows the timing chart of the above program.

Figure 12-9. Timing Chart



As shown in Figure 12-9, the program is started from address 0000H because the internal 10-Hz pulse rises when supply voltage  $V_{DD}$  is first applied.

When the BTM0CY flag is detected at point A, it is judged that the BTM0CY flag is reset (to 0) and that a power failure (power-on reset) has occurred because the power has just been applied.

Therefore, “processing C” is executed, and the BTM0CY flag setting pulse is set to 100 ms.

Because the content of the BTM0CY flag is read once at point A, the BTM0CY flag will be set to 1 every 100 ms.

Next, even if the CE pin goes low at point B and high at point C, the program counts up the watch timer while executing “processing B”, unless the clock stop instruction is executed.

At point C, because the CE pin goes high from low, CE reset is effected at point D at which the BTM0CY flag setting pulse rises next time, and the program is started from address 0000H.

When the BTM0CY flag is detected at point E at this time, it is set to 1. Therefore, this is judged to be a back up (CE reset).

As is evident from the above figure, unless the watch is updated by 100 ms at point E, the watch is delayed by 100 ms each time CE reset is effected.

If processing A takes longer than 100 ms when a power failure is detected at point E, the setting of the BTM0CY flag is overlooked two times. Therefore, processing A must be completed within 100 ms.

The above description also applies when the BTM0CY flag setting pulse is set to 250, 5, or 1 ms.

Therefore, the BTM0CY flag must be detected for power failure detection within the BTM0CY flag setting time after the program has been started from address 0000H.

#### (4) If BTM0CY flag is detected at the same time as CE reset

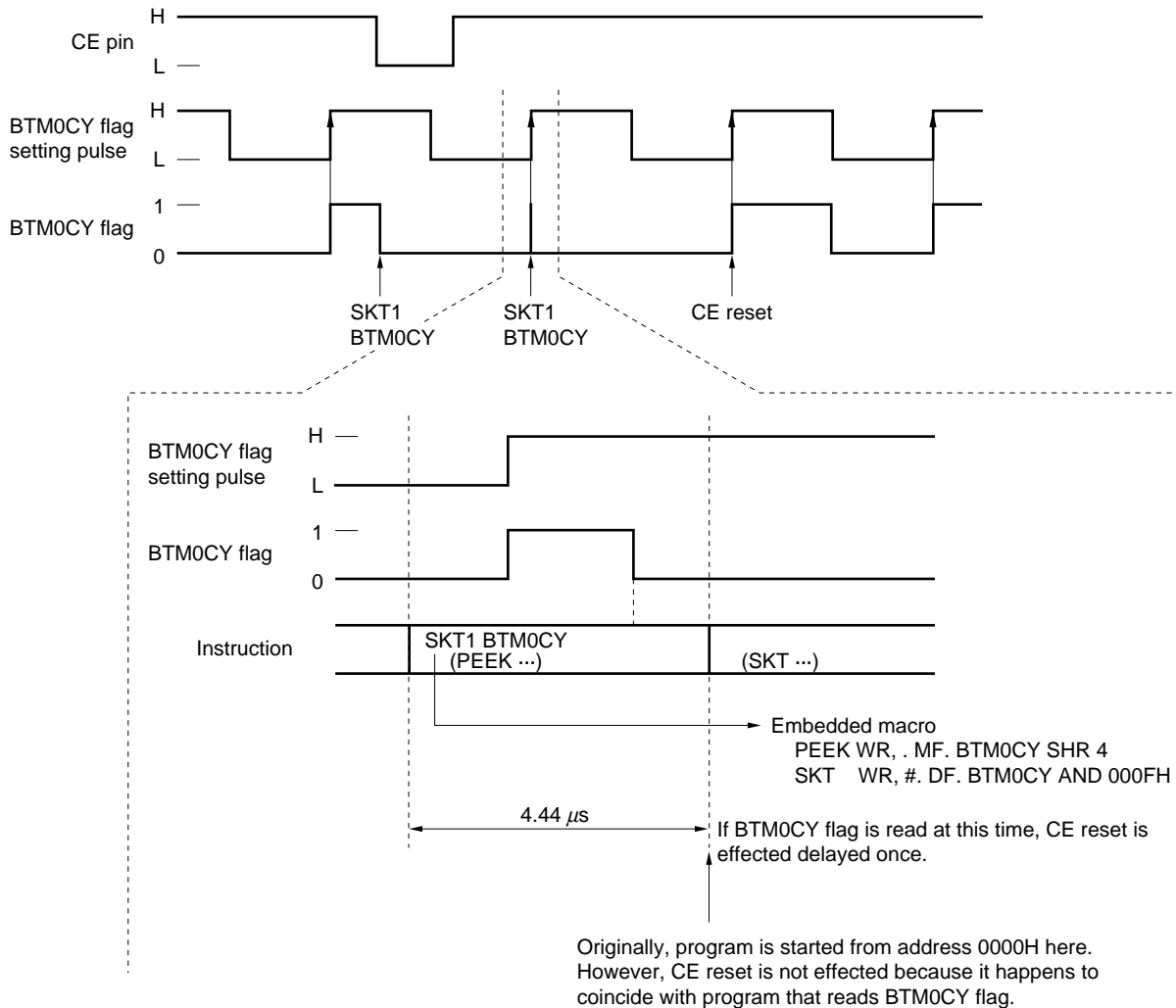
As described in (3) above, CE reset is effected as soon as the BTM0CY flag is set to 1.

If the instruction that reads the BTM0CY flag happens to be executed at the same time as CE reset at this time, the BTM0CY flag reading instruction takes precedence.

Therefore, if the next setting the BTM0CY flag (rising of BTM0CY flag setting pulse) after the CE pin has gone high coincides with execution of the BTM0CY flag reading instruction, CE reset is effected at the next timing at which the BTM0CY flag is set.

This operation is illustrated in Figure 12-10.

**Figure 12-10. Operation When CE Reset Coincides with BTM0CY Flag Reading Instruction**



Consequently, if the BTM0CY flag detection time interval coincides with the BTM0CY flag setting time in a program that cyclically detects the BTM0CY flag, CE reset is never effected.

Therefore, the following point must be noted.

Because one instruction cycle is 4.44 μs (1/225 kHz), a program that detects the BTM0CY flag once, for example, every 225 instructions, reads the BTM0CY flag every  $4.44 \mu\text{s} \times 225 = 1 \text{ ms}$ .

Even if any of 1 ms, 5 ms, 100ms, or 250 ms is selected as the timer time setting pulse, if setting and detection of the BTM0CY flag coincide once, CE reset is never effected.

Therefore, do not create a cyclic program that satisfies the following condition.

$$\frac{t_{\text{SET}} \times 225}{X} = n \text{ (n: natural number)}$$

t<sub>SET</sub>: BTM0CY flag setting time

X: Cycle X step of instruction that reads BTM0CY flag

An example of a program that satisfies the above condition is shown below. Do not create such a program.

### Example

```

Processing A
SET    BTM0CK1, BTM0CK0 ; Embedded macro
                        ; Sets BTM0CY flag setting pulse to 1 ms
LOOP:
; <1>
SKT1   BTM0CY           ; Embedded macro
BR     BBB
AAA:
221 steps
BR     LOOP
BBB:
221 steps
BR     LOOP

```

Because the BTM0CY flag reading instruction in <1> is repeatedly executed every 225 instructions in this example, CE reset is not effected if the BTM0CY flag happens to be set at the timing of the instruction in <1>.

## 12.3 Basic Timer 1

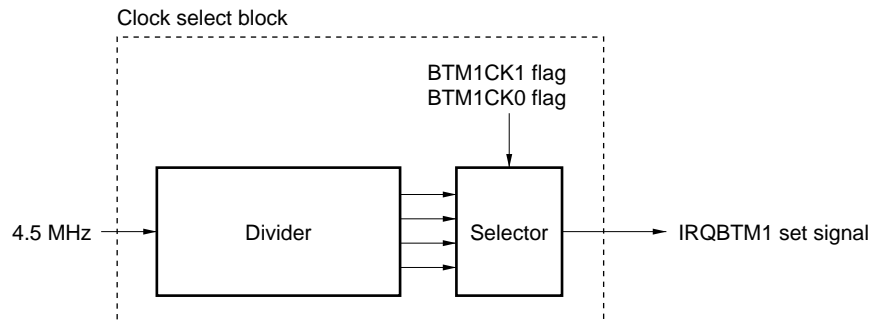
### 12.3.1 General

Figure 12-11 illustrates basic timer 1.

Basic timer 1 issues an interrupt request at a fixed time interval and sets the IRQBTM1 flag to 1.

The interrupt generated by basic timer 1 is acknowledged when the IRQBTM1 flag is set, if the EI instruction has been issued and the IPBTM1 flag has been set (refer to **11. INTERRUPTS**).

**Figure 12-11. Outline of Basic Timer 1**



**Remark** BTM1CK1 and BTM1CK0 (bits 3 and 2 of the basic timer clock select register, refer to **Figure 12-3**) set the time interval at which the IRQBTM1 flag is set.

### 12.3.2 Clock select block

The clock select block divides the system clock (4.5 MHz) and sets the time interval at which the IRQBTM1 flag is to be set, by using the basic timer clock select register.

For the configuration and function of the basic timer clock select register, refer to Figure 12-3.

### 12.3.3 Application example of basic timer 1

A program example is shown below.

#### Example

```

M1      MEM      0.10H      ; 80 ms counter
BTIMER1 DAT      0002H      ; Symbol definition of basic timer 1 interrupt vector address

        BR       START      ; Branches to START
ORG     BTIMER1      ; Program address (0002H)
        ADD      M1, #0001B  ; Adds 1 to M1
        SKT1     CY         ; Tests CY flag
        BR       EI_RET1    ; Returns if no carry
        Processing A
EI_RET1:
        EI
        RETI

START:
        INITFLG   BTM1CK1, NOT BTM1CK0
                                ; Embedded macro
                                ; Sets basic timer 1 interrupt pulse to 5 ms
        MOV       M1, #0000B ; Clears contents of M1 to 0
        SET1      IPBTM1     ; Enables basic timer 1 interrupt
        EI        ; Enables all interrupts

LOOP:
        Processing B
        BR       LOOP

```

This program executes processing A every 80 ms.

The points to be noted in this case are that the DI status is automatically set when an interrupt has been acknowledged, and that the IRQBTM1 flag is set to 1 even in the DI status.

This means that the interrupt is acknowledged even if execution exits from an interrupt routine by execution of the RETI instruction, if processing A takes longer than 5 ms.

Consequently, processing B is not executed.

### 12.3.4 Error of basic timer 1

As described in 12.3.3, the interrupt generated by basic timer 1 is acknowledged each time the basic timer 1 interrupt pulse falls, if the EI instruction has been executed, and if the interrupt has been enabled.

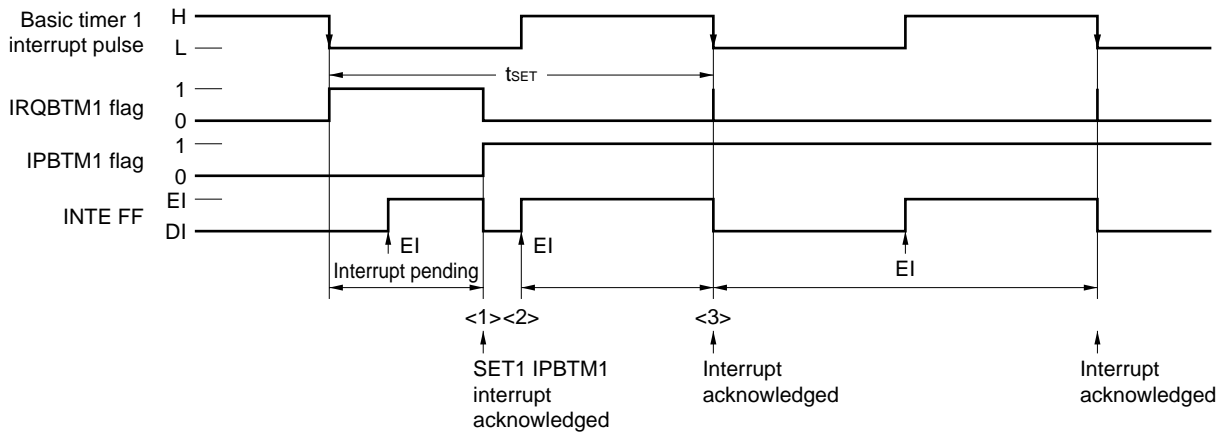
Therefore, an error of basic timer 1 occurs only when any of the following operations (1) to (3) is performed:

- (1) When the first interrupt after the basic timer 1 interrupt has been enabled has been acknowledged
- (2) When the time interval at which the IRQBTM1 flag is to be set is changed, i.e., when the first interrupt is acknowledged after the interrupt pulse has been changed
- (3) When data has been written to the IRQBTM1 flag

Figure 12-12 shows an error in each of the above operations.

**Figure 12-12. Error of Basic Timer 1 (1/2)**

**(a) When interrupt by basic timer 1 is enabled**



At point <1> in the above figure, the interrupt by basic timer 1 is acknowledged as soon as the interrupt is enabled.

At this time, the error is  $-t_{SET}$ .

If an interrupt is enabled by the "EI" instruction at the next point <2>, the interrupt occurs at the falling edge of the basic timer 1 interrupt pulse.

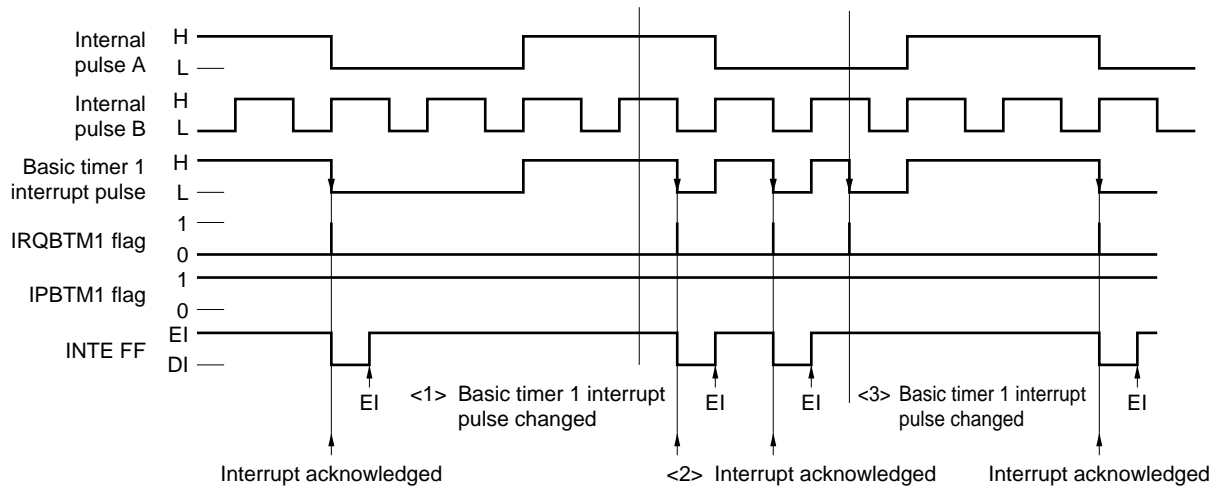
At this time, the error is:

$$-t_{SET} < \text{error} < 0$$



Figure 12-12. Error of Basic Timer 1 (2/2)

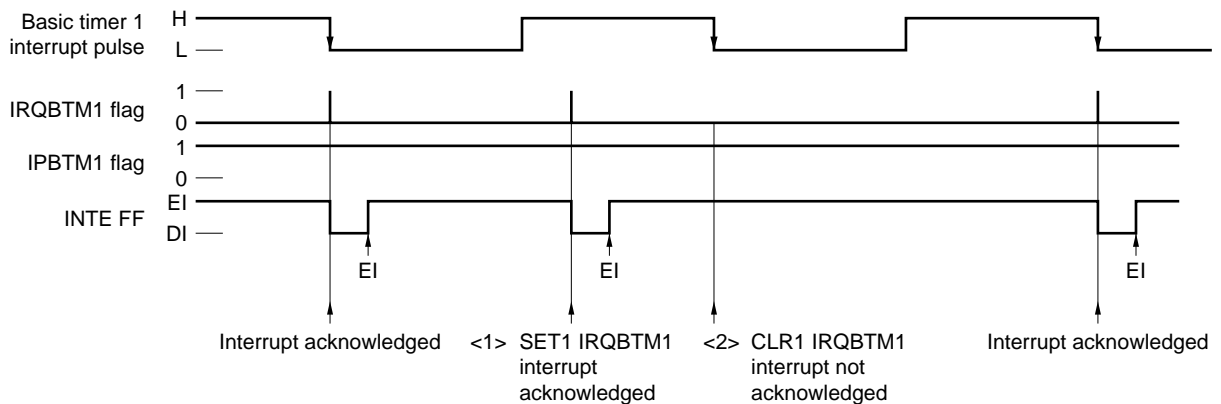
(b) When basic timer 1 interrupt pulse is changed



Even if the basic timer 1 interrupt pulse is changed to B at point <1> in the above figure, the interrupt is acknowledged at the next point <2> because the basic timer 1 interrupt pulse does not fall.

If the basic timer 1 interrupt pulse is changed to A at <3>, the interrupt is immediately acknowledged because the basic timer 1 interrupt pulse falls.

(c) When IRQBTM1 flag is manipulated



The interrupt is immediately acknowledged if the IRQBTM1 flag is set to 1 at <1>.

If clearing the IRQBTM1 flag to 0 overlaps with the falling of the basic timer 1 interrupt pulse at <2>, the interrupt is not acknowledged.

### 12.3.5 Notes on using basic timer 1

When creating a program, such as a watch program, in which processing is always performed at fixed time intervals using basic timer 1 after the supply voltage has been applied (power-on reset), the basic timer 1 interrupt servicing must be completed in a fixed time.

Let's take the following example:

#### Example

```

M1      MEM      0.10H      ; 1 ms counter
BTIMER1 DAT      0002H      ; Symbol definition of interrupt vector address of basic timer 1

        BR       START      ; Branches to START
ORG     BTIMER1      ; Program address (0002H)
        ADD      M1, #0100B ; Adds 0100B to M1
        SKT1     CY        ; Watch processing if carry occurs
        BR       EI_RET1    ; Returns if no carry occurs
; <1>
        Watch processing
EI_RET1:
        EI
        RETI
START:
        INITFLG   NOT BTM1CK1, BTM1CK0, NOT BTM0CK1, NOT BTM0CK0
                                ; Embedded macro
                                ; Sets time of interrupt by basic timer 1 to 250 ms
                                ; and set time of BTM0CY flag to 100 ms
        SET1      IPBTM1    ; Embedded macro
                                ; Enables interrupt by basic timer 1
        EI        ; Enables all interrupts
LOOP:
        Processing A
        BR        LOOP

```

In this example, watch processing <1> is executed every 1 second while processing A is executed.

If the CE pin goes high as shown in Figure 12-13 (a), CE reset is effected in synchronization with the rising of the BTM0CY flag setting pulse.

If issuance of an interrupt request by the basic timer 1 happens to overlap with the setting of the BTM0CY flag at this time, CE reset takes precedence.

When CE reset is effected, the basic timer 1 interrupt request (IRQBTM1) flag is cleared. Consequently, the timer processing is skipped once.

To prevent this, a delay is actually provided to the rising of the BTM0CY flag setting pulse and falling of the basic timer 1 interrupt pulse as shown in Figure 12-13 (b).

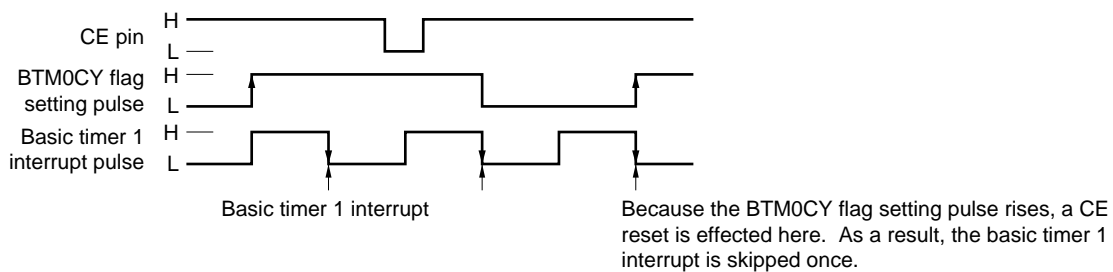
In the above example, therefore, skipping of the basic timer 1 interrupt is prevented, even if a CE reset is effected, by performing the watch processing within 10 ms.

Because the BTM0CY flag setting pulse and basic timer 1 interrupt time setting pulse can be independently set to 4 Hz (250 ms), 10 Hz (100 ms), 200 Hz (5 ms), or 1 kHz (1 ms), a time difference is provided as shown in Figure 12-14 and Table 12-1.

Consequently, if the basic timer 1 interrupt must be enabled even when a CE reset is effected, the servicing of the basic timer 1 interrupt must be completed within the delay time of the pulse shown in Figure 12-14.

**Figure 12-13. Timing Chart**

(a)



(b)

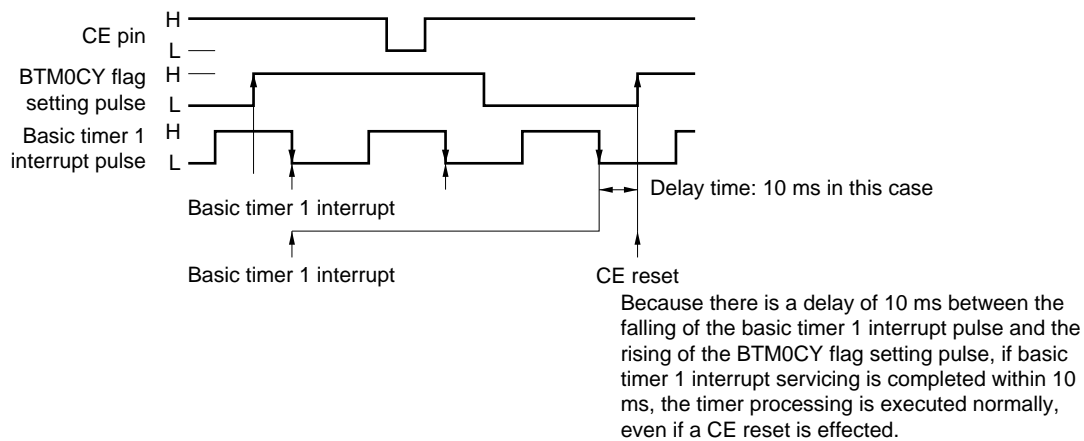
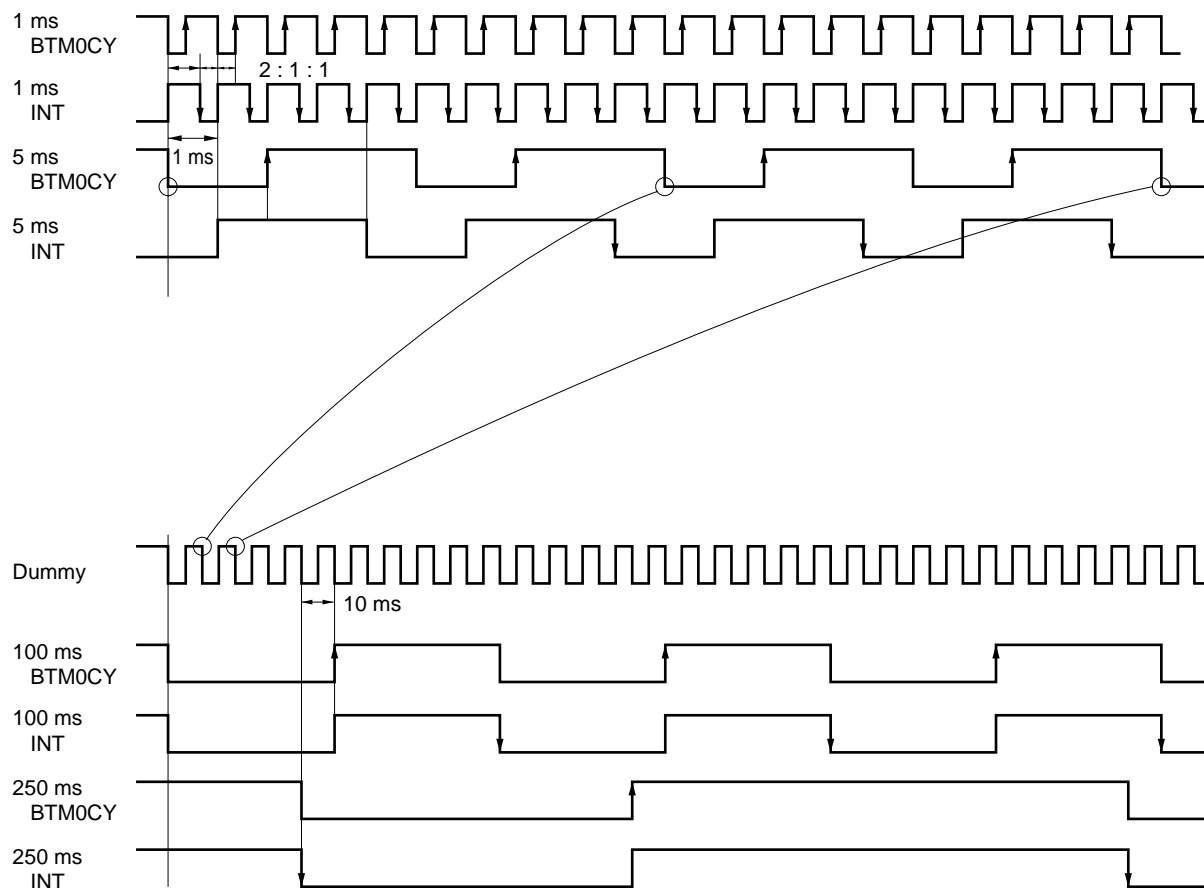
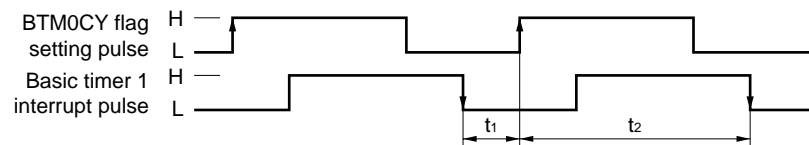


Figure 12-14. Time Difference Between BTM0CY Flag Setting Pulse and Basic Timer 1 Interrupt Pulse



**Table 12-1. Time Difference Between Rising Edge of BTM0CY Flag Setting Pulse and Falling Edge of Basic Timer 1 Interrupt Pulse**

Internal Pulse		Minimum Value of Time Difference (Refer to Figure Below.)	
BTM0CY Flag Setting Pulse	Basic Timer 1 Interrupt Pulse	t <sub>1</sub>	t <sub>2</sub>
1 ms	1 ms	666 $\mu$ s	333 $\mu$ s
1 ms	5 ms	333 $\mu$ s	666 $\mu$ s
1 ms	100 ms	333 $\mu$ s	666 $\mu$ s
1 ms	250 ms	333 $\mu$ s	666 $\mu$ s
5 ms	1 ms	333 $\mu$ s	666 $\mu$ s
5 ms	5 ms	3 ms	2 ms
5 ms	100 ms	2 ms	3 ms
5 ms	250 ms	2 ms	3 ms
100 ms	1 ms	333 $\mu$ s	666 $\mu$ s
100 ms	5 ms	1 ms	4 ms
100 ms	100 ms	50 ms	50 ms
100 ms	250 ms	10 ms	40 ms
250 ms	1 ms	333 $\mu$ s	666 $\mu$ s
250 ms	5 ms	1 ms	4 ms
250 ms	100 ms	40 ms	10 ms
250 ms	250 ms	100 ms	150 ms



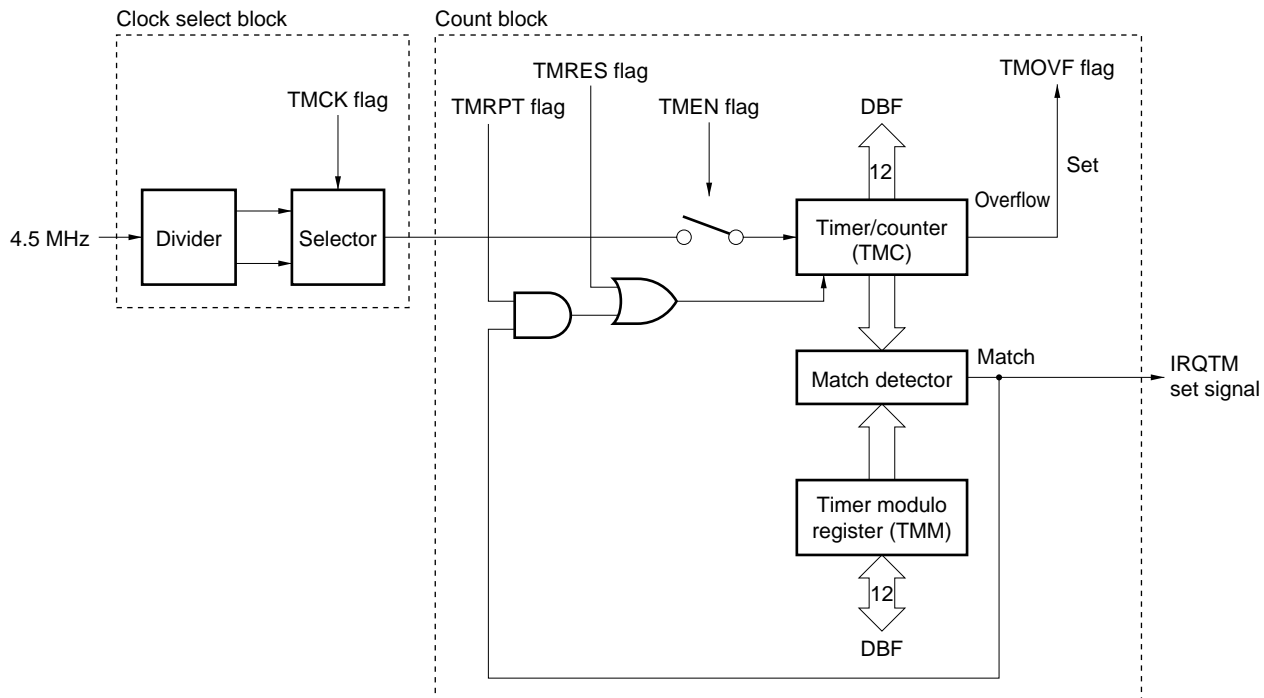
## 12.4 12-Bit Timer

### 12.4.1 General

Figure 12-15 illustrates the 12-bit timer.

The 12-bit timer operates as a timer by counting the basic clock (100 kHz or 20 kHz) by using a 12-bit counter, and comparing its count value with a value set in advance.

**Figure 12-15. Outline of 12-Bit Timer**



- Remarks**
1. TMCK (bit 0 of 12-bit timer clock select register; refer to **Figure 12-16**) sets the basic clock frequency.
  2. TMEN (bit 0 of 12-bit timer control register; refer to **Figure 12-17**) starts/stops the 12-bit timer.
  3. TMRES (bit 1 of 12-bit timer control register; refer to **Figure 12-17**) controls resetting the timer/counter.
  4. TMRPT (bit 2 of 12-bit timer control register; refer to **Figure 12-17**) selects the modulo count mode/free-run count mode.
  5. TMOVF (bit 0 of 12-bit timer overflow register; refer to **Figure 12-18**) detects an overflow in the timer/counter.

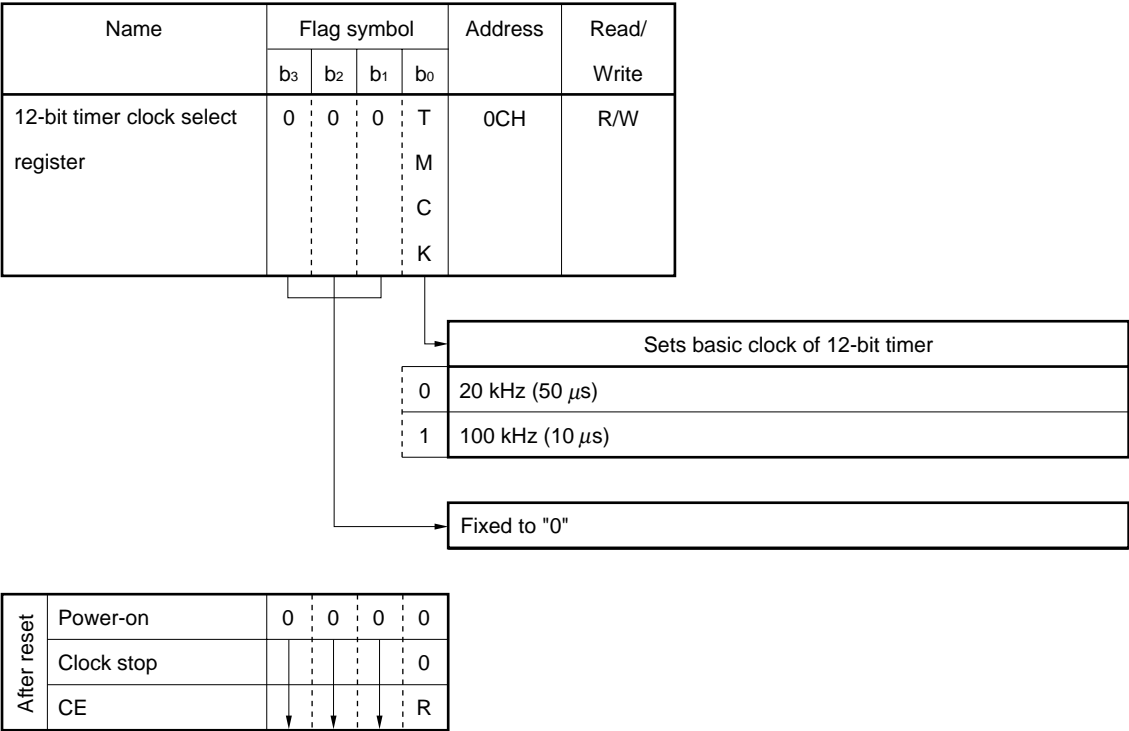
12.4.2 Clock select block

The clock select block selects the basic clock that is used for the operation of the timer/counter.

Two basic clocks can be selected by using the TMCK flag.

Figure 12-16 shows the configuration and function of the 12-bit timer clock select register.

Figure 12-16. Configuration of 12-Bit Timer Clock Select Register



**Remark** R: retained

### 12.4.3 Count block

The count block counts the basic clock by using a 12-bit timer/counter. When the count value matches the value of the timer modulo register, the count block issues an interrupt request.

The value of the timer/counter can be written or read via the data buffer.

The basic clock that is input to the timer/counter can be started or stopped by the TMEN flag.

The timer/counter can be reset by the TMRES flag.

The timer/counter is not automatically reset even when its count value matches the value of the timer modulo register.

Either the modulo count mode or free-run count mode can be set by the TMRPT flag.

In the free-run count mode, the contents of the timer/counter are not reset even after a match between the value of the timer/counter and the contents of the timer modulo register has been detected; therefore, the timer/counter continues counting up.

In the modulo counter mode, the contents of the timer/counter are reset and then the timer/counter continues counting when a match between the count value of the timer/counter and the contents of the timer modulo register has been detected.

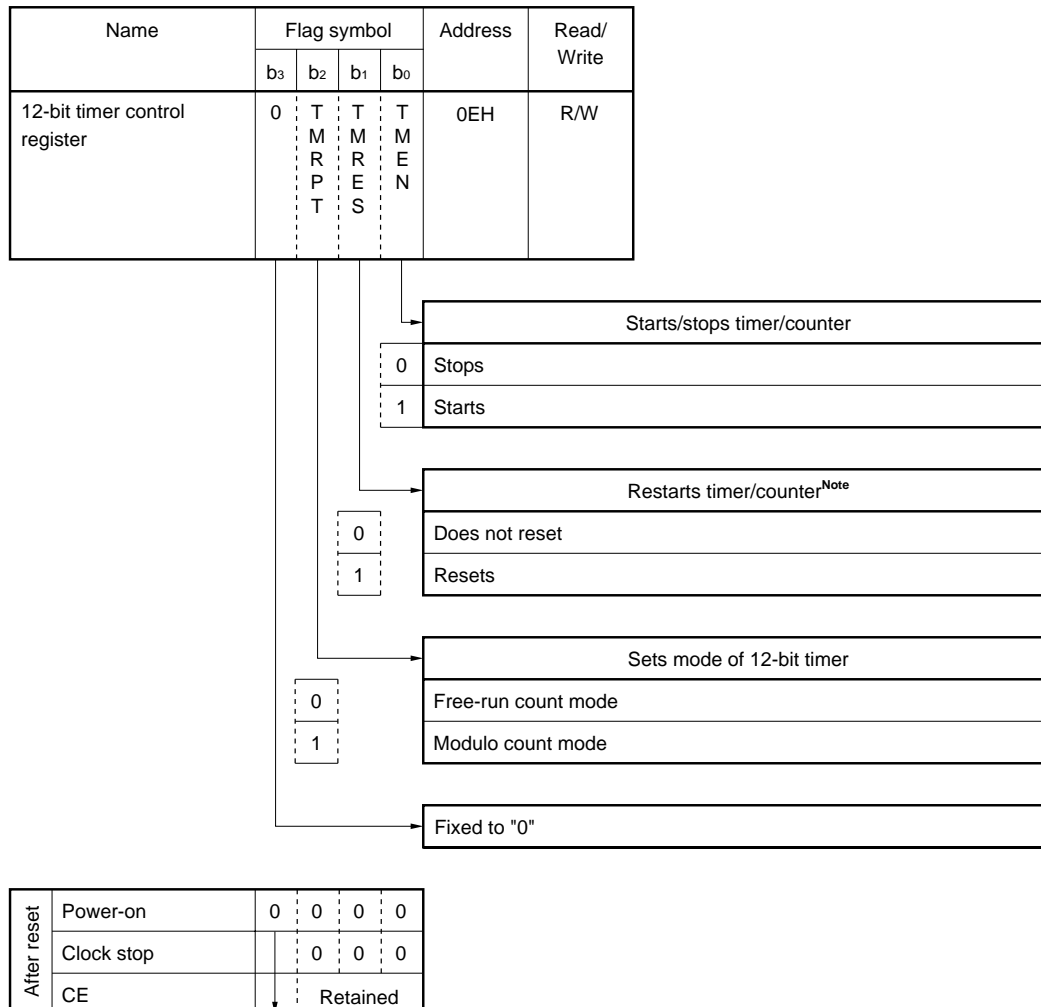
An overflow in the counter, if any, can be detected by the TMOVF flag. If an overflow has been detected, the counting operation is stopped.

Figure 12-17 shows the configuration and function of the 12-bit timer control register.

Figure 12-18 shows the configuration and function of the 12-bit timer overflow register.

Figures 12-19 and 12-20 show the configurations of the timer/counter and timer modulo register respectively.

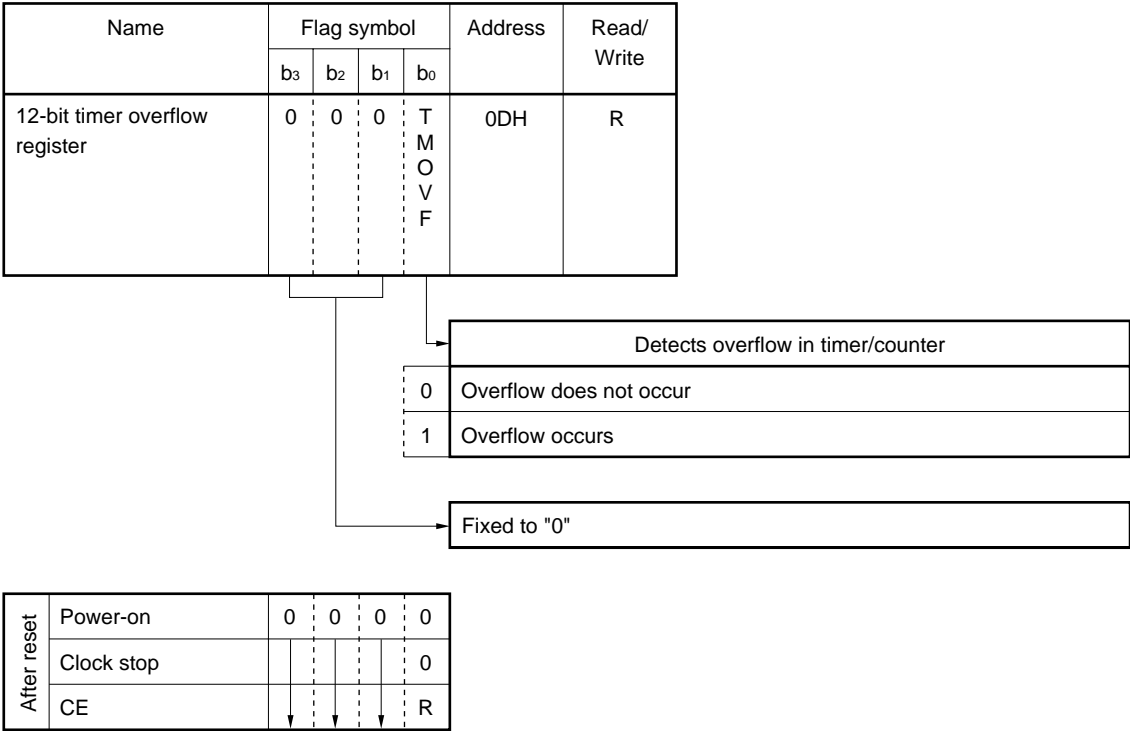
**Figure 12-17. Configuration of 12-Bit Timer Control Register**



**Note** The TMRES flag is always “0” when it is read.



Figure 12-18. Configuration of 12-Bit Timer Overflow Register



Remark R: retained

Figure 12-19. Configuration of Timer/Counter

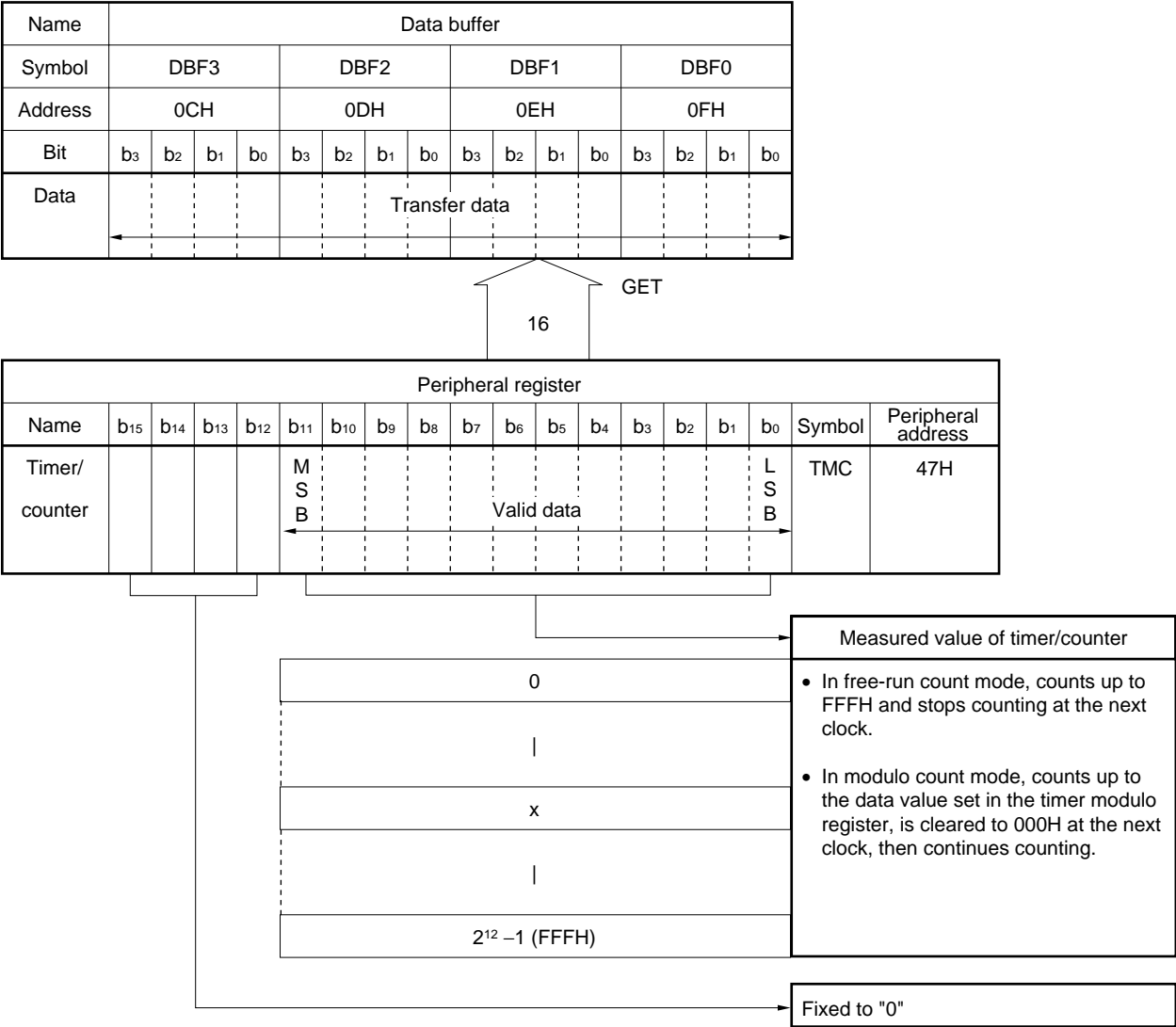
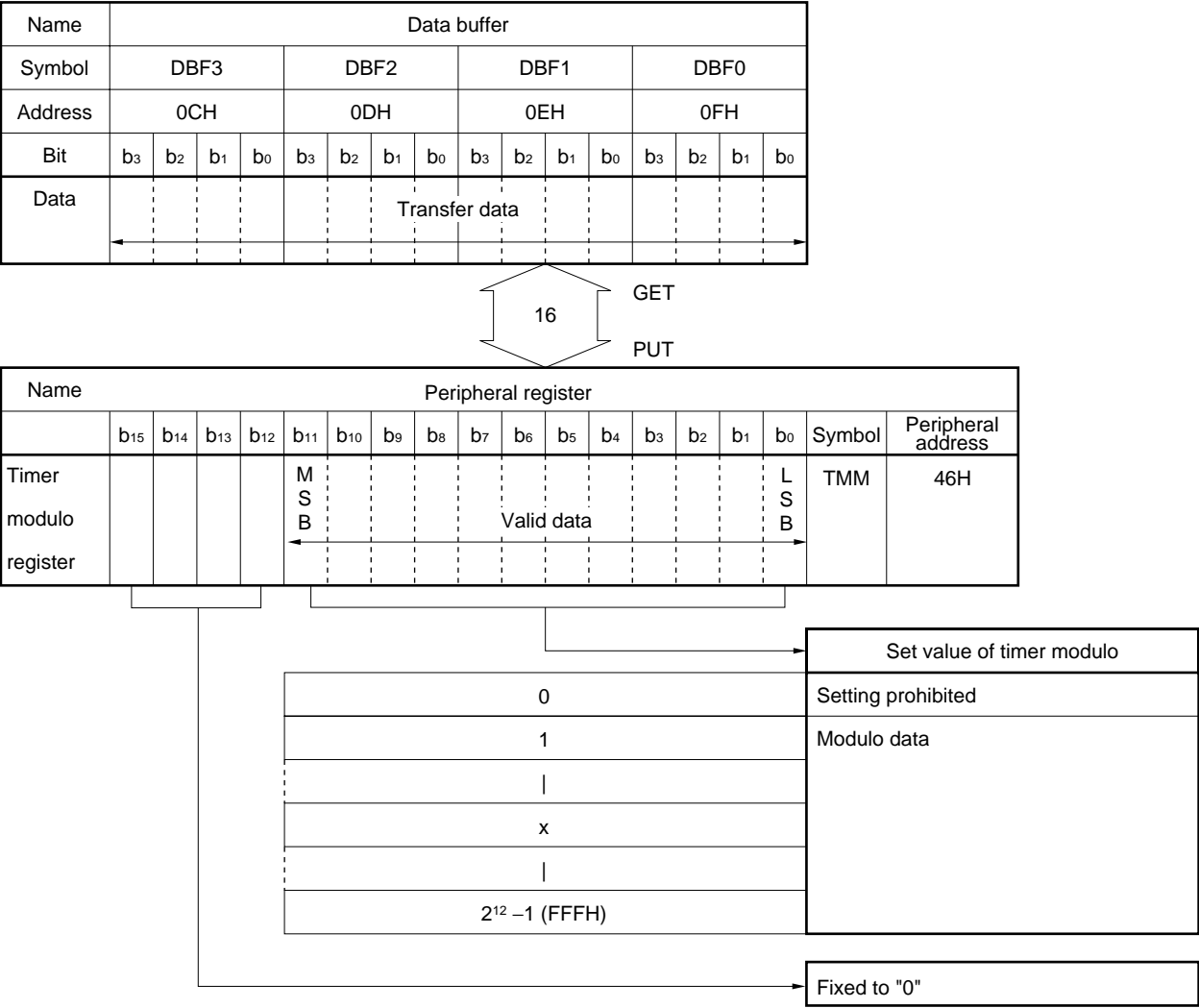


Figure 12-20. Configuration of Timer Modulo Register



#### 12.4.4 Application example of 12-bit timer

### Example 1. Modulo count mode

```

TMINT    DAT    0003H           ; Symbol definition of 12-bit timer interrupt vector address
        BR      START
ORG      TMINT                   ; Program address (0003H)
        Processing A
        EI
        RETI

START:
        INITFLG  TMCK           ; Sets count clock to 100 kHz (10  $\mu$ s)
        MOV      DBF2, #50 SHR 8 AND 0FH
        MOV      DBF1, #50 SHR 4 AND 0FH
        MOV      DBF0, #50 AND 0FH
        PUT      TMM, DBF
        SET1     IPTM
        EI
        SET3     TMRPT, TMRES, TMEN

LOOP:
        Main processing
        BR       LOOP

```

This program executes processing A every 500  $\mu$ s.  
However, processing A must be completed within 500  $\mu$ s.

### Example 2. Free-run count mode

```

BR          Start
          ⋮
Start:      ⋮
          ⋮
INITFLG    TMCK          ; Sets count clock to 100 kHz (10 μs)
INITFLG    NOT TMRPT, TMRES, TMEN
Processing A
SKF1       TMOVF
BR         Overflow occurs
GET        DBF, TMC
          ⋮
Overflow occurs
          ⋮
          ⋮
          ⋮

```

This program is to measure the time required for processing A. The measurable time range is from 10  $\mu$ s to 40,950  $\mu$ s (the software in Example 2 cannot measure time exceeding 40,950  $\mu$ s and therefore, execution must branch to another routine to measure the time longer than 40,950  $\mu$ s).

This program is used to measure the pulse width of a remote controller signal.

The modulo count mode is useful for issuing an interrupt request at fixed time intervals, but the free-run count mode is better to measure total time.

#### 12.4.5 Error of 12-bit timer

The 12-bit timer produces an error of a maximum of 1 basic clock in the following cases:

**(1) When TMEN flag is manipulated**

When the TMEN flag is set, an error of 0 to +1 clock occurs.

When the TMEN flag is cleared, an error of 0 to -1 clock occurs.

**(2) When counter in operation is reset**

When the counter is reset, an error of 0 to +1 clock occurs.

**(3) When basic clock is changed during counter operation**

An error of 0 to +1 clock of the new clock occurs.

#### 12.4.6 Notes on using 12-bit timer

The interrupt by the 12-bit timer may be generated at the same time as the interrupt by basic timer 1 and CE reset.

If the timer must be updated even at CE reset, do not use the 12-bit timer. Instead, use basic timer 1.

## 13. A/D CONVERTER (ADC)

### 13.1 General

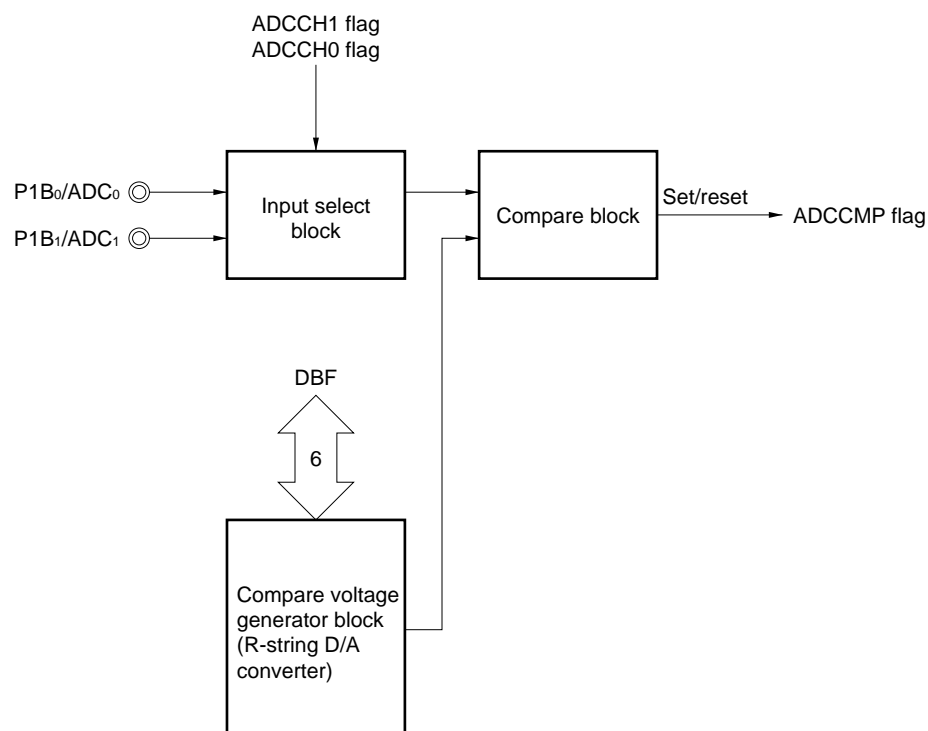
Figure 13-1 illustrates the A/D converter.

The A/D converter compares an analog voltage input to the ADC<sub>0</sub> or ADC<sub>1</sub> pins with the internal compare voltage, judges the comparison result via software, and converts the analog signal into a 6-bit digital signal.

The comparison result can be detected by the ADCCMP flag.

As the comparison method, successive approximation is employed.

**Figure 13-1. Outline of A/D Converter**



- Remarks**
1. ADCCH0 and ADCCH1 (bits 0 and 1 of the A/D converter channel select register; refer to **Figure 13-3**) select the pin used for the A/D converter.
  2. ADCCMP (bit 0 of the A/D converter compare judge register; refer to **Figure 13-5**) detects the result of comparison.

### 13.2 Input Selector Block

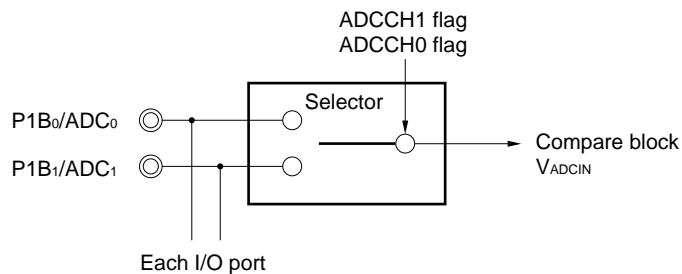
Figure 13-2 shows the configuration of the input selector block.

The input selector block selects the pin to be used via the A/D converter channel select register.

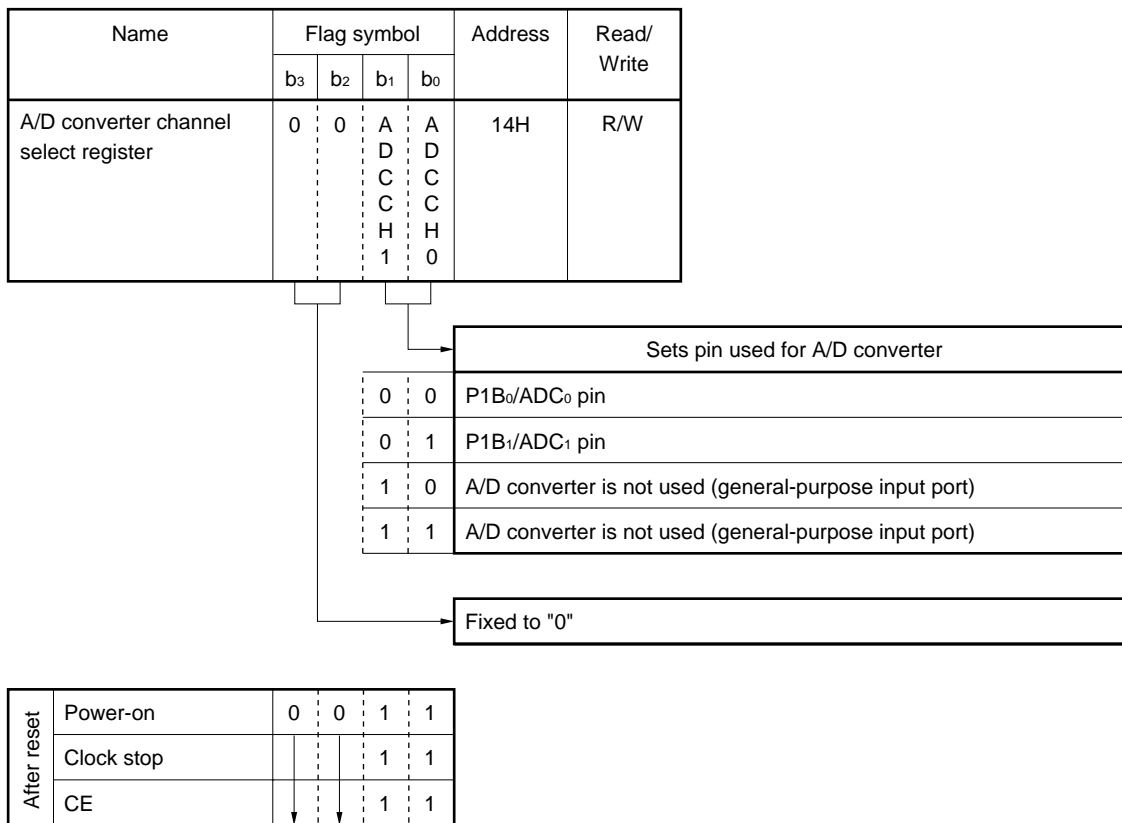
Two or more pins cannot be used at the same time with the A/D converter.

Figure 13-3 shows the configuration and function of the A/D converter channel select register.

**Figure 13-2. Configuration of Input Selector Block**



**Figure 13-3. Configuration of A/D Converter Channel Select Register**



### 13.3 Compare Voltage Generator Block and Compare Block

Figure 13-4 shows the configuration of the compare voltage generator block and compare block.

The compare voltage generator block switches over the tap decoder by using 6-bit data set to the A/D converter data register to generate 64 steps of compare voltage  $V_{REF}$ .

In other words, this block is an R-string D/A converter.

The power source of the R string is the same as the  $V_{DD}$  supplied to the device.

The voltage applied to the resistor of the R string is only supplied when the ADCCMP flag is read by using the PEEK instruction.

The compare block judges which of the voltage  $V_{ADCIN}$  input from a pin and compare voltage  $V_{REF}$  is greater.

Comparison is made by a comparator when the ADCCMP flag is read. Therefore, one compare time of the A/D converter is equal to one instruction execution time ( $4.44 \mu s$ ).

Figures 13-5 and 13-6 show the configuration and function of the A/D converter compare judge register and A/D converter data register. Table 13-1 lists the compare voltages.

**Figure 13-4. Configuration of Compare Voltage Generator Block and Compare Block**

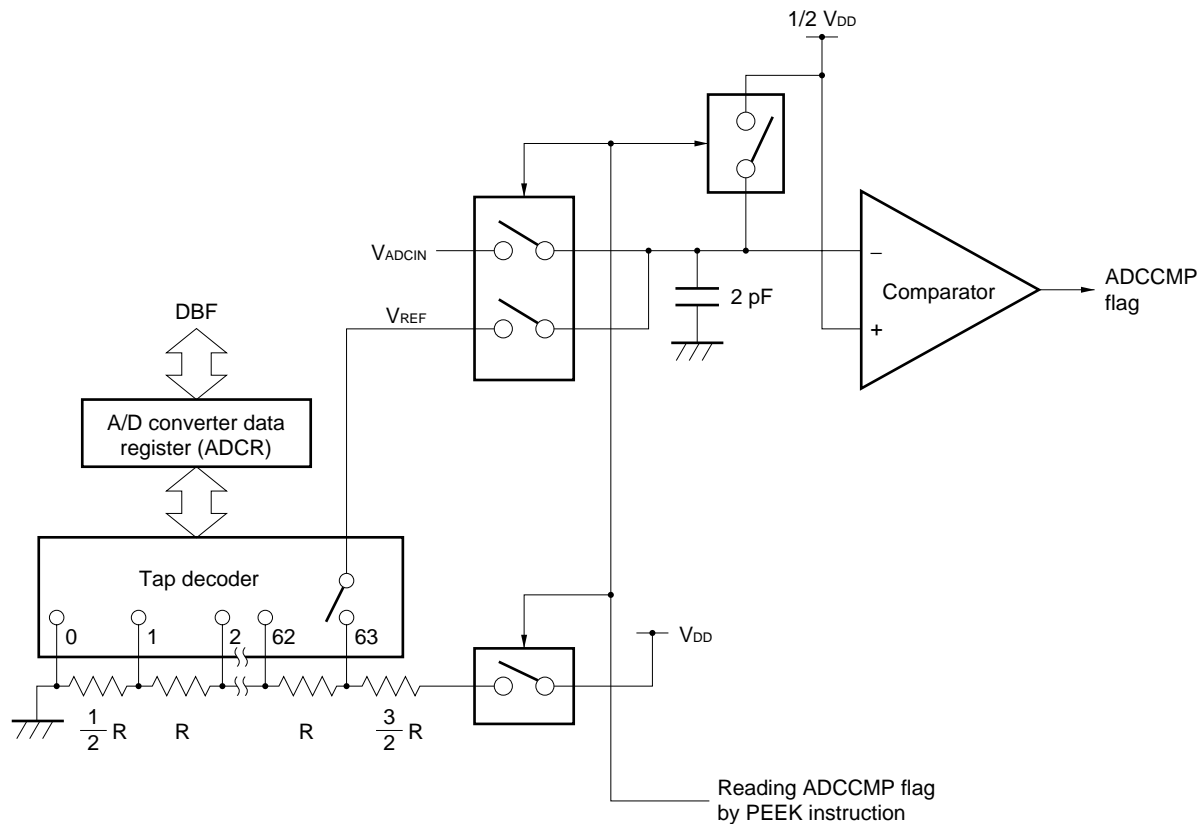
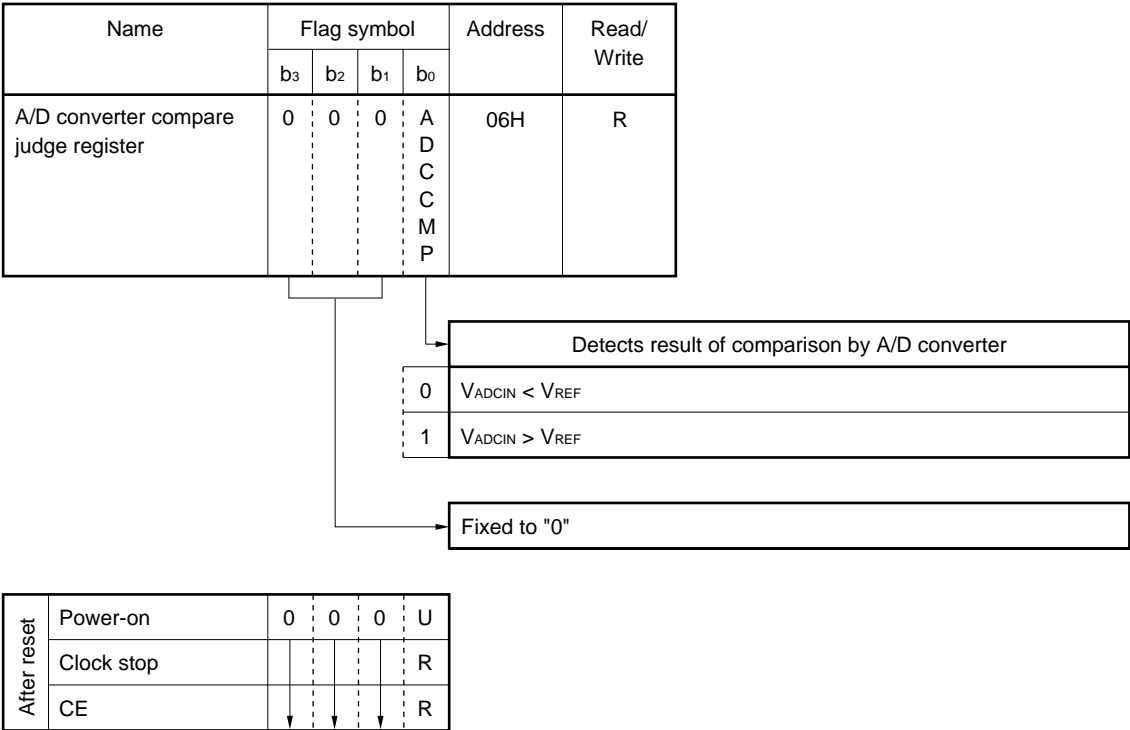




Figure 13-5. Configuration of A/D Converter Compare Judge Register



**Remark** U: Undefined  
R: Retained

Figure 13-6. Configuration of A/D Converter Data Register

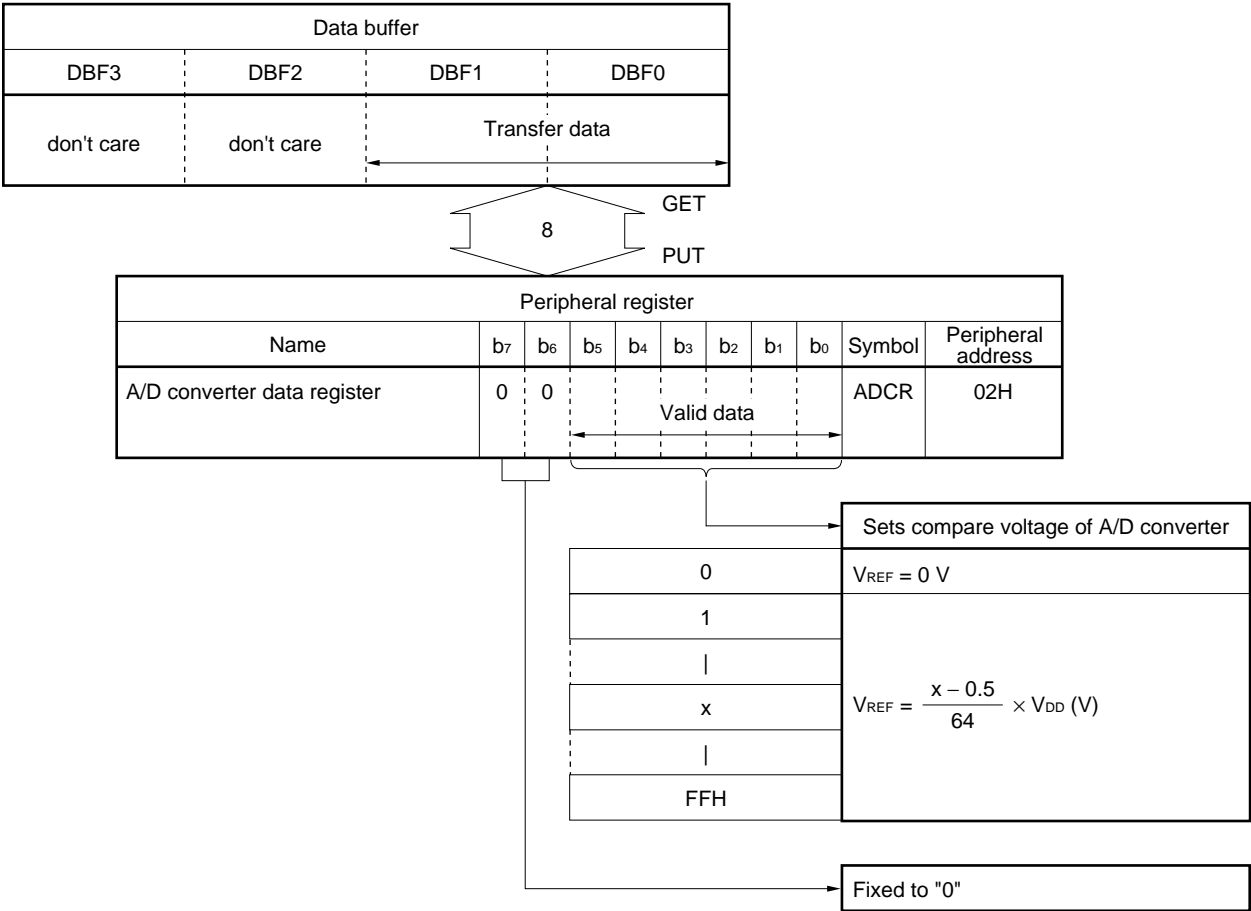


Table 13-1. Set Values of A/D Converter Data Register and Compare Voltages

Set Data of ADCR		Compare Voltage		Set Data of ADCR		Compare Voltage	
Decimal (DEC)	Hexadecimal (HEX)	Logic Voltage Unit: $\times V_{DD}$ V	At $V_{DD} = 5$ V Unit: V	Decimal (DEC)	Hexadecimal (HEX)	Logic Voltage Unit: $\times V_{DD}$ V	At $V_{DD} = 5$ V Unit: V
0	00H	0	0	32	20H	31.5/64	2.461
1	01H	0.5/64	0.039	33	21H	32.5/64	2.539
2	02H	1.5/64	0.117	34	22H	33.5/64	2.617
3	03H	2.5/64	0.195	35	23H	34.5/64	2.695
4	04H	3.5/64	0.273	36	24H	35.5/64	2.773
5	05H	4.5/64	0.352	37	25H	36.5/64	2.852
6	06H	5.5/64	0.430	38	26H	37.5/64	2.930
7	07H	6.5/64	0.508	39	27H	38.5/64	3.008
8	08H	7.5/64	0.586	40	28H	39.5/64	3.086
9	09H	8.5/64	0.664	41	29H	40.5/64	3.164
10	0AH	9.5/64	0.742	42	2AH	41.5/64	3.242
11	0BH	10.5/64	0.820	43	2BH	42.5/64	3.320
12	0CH	11.5/64	0.898	44	2CH	43.5/64	3.398
13	0DH	12.5/64	0.977	45	2DH	44.5/64	3.477
14	0EH	13.5/64	1.055	46	2EH	45.5/64	3.555
15	0FH	14.5/64	1.133	47	2FH	46.5/64	3.633
16	10H	15.5/64	1.211	48	30H	47.5/64	3.711
17	11H	16.5/64	1.289	49	31H	48.5/64	3.789
18	12H	17.5/64	1.367	50	32H	49.5/64	3.867
19	13H	18.5/64	1.445	51	33H	50.5/64	3.945
20	14H	19.5/64	1.523	52	34H	51.5/64	4.023
21	15H	20.5/64	1.602	53	35H	52.5/64	4.102
22	16H	21.5/64	1.680	54	36H	53.5/64	4.180
23	17H	22.5/64	1.758	55	37H	54.5/64	4.258
24	18H	23.5/64	1.836	56	38H	55.5/64	4.336
25	19H	24.5/64	1.914	57	39H	56.5/64	4.414
26	1AH	25.5/64	1.992	58	3AH	57.5/64	4.492
27	1BH	26.5/64	2.070	59	3BH	58.5/64	4.570
28	1CH	27.5/64	2.148	60	3CH	59.5/64	4.648
29	1DH	28.5/64	2.227	61	3DH	60.5/64	4.727
30	1EH	29.5/64	2.305	62	3EH	61.5/64	4.805
31	1FH	30.5/64	2.383	63	3FH	62.5/64	4.883

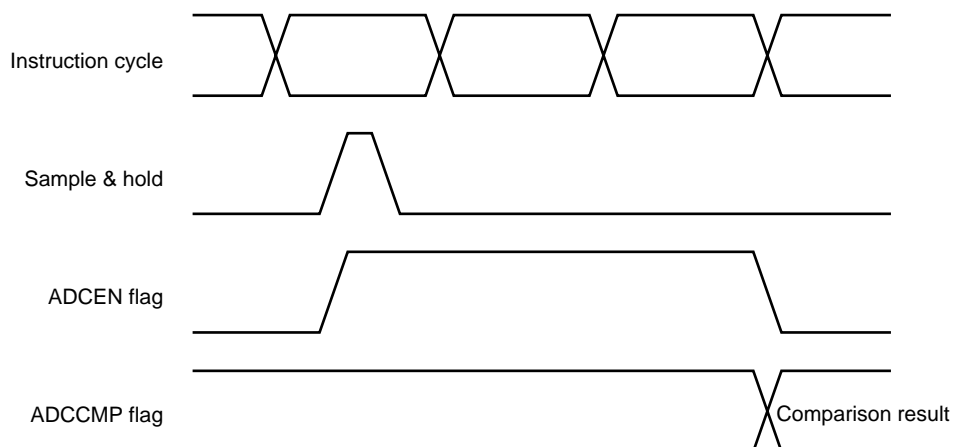
### 13.4 Comparison Timing Chart

The ADCEN flag is automatically cleared to 0 when the comparison operation has been completed.

Therefore, because the ADCEN flag is detected after it has been set, and the comparison result (ADCCMP flag) is read when the ADCEN flag has been cleared, one compare time is equal to three instruction execution times (6 μs).

Figure 13-7 shows the timing chart.

**Figure 13-7. Timing Chart of A/D Converter's Compare Operation**



### 13.5 Performance of A/D Converter

The performance of the A/D converter is as follows.

Parameter	Performance
Resolution	6 bits
Input voltage range	0-V <sub>DD</sub>
Quantization error	$\pm \frac{1}{2}$ LSB
Over range	$\frac{62.5}{64} \times V_{DD}$
Offset, gain, and non linearity errors	$\pm \frac{3}{2}$ LSB <sup>Note</sup>

**Note** Including quantization error.

## 13.6 Using A/D Converter

### 13.6.1 Comparing one compare voltage

Here is a program example.

**Example** To compare input voltage  $V_{ADCIN}$  of the  $ADC_0$  pin with compare voltage  $V_{REF}$  ( $31.5/64 V_{DD}$ ) and branch to AAA if  $V_{ADCIN} > V_{REF}$  or to BBB if  $V_{ADCIN} < V_{REF}$

```

INIT:
    ADCR7    FLG    0.0EH.3        ; Dummy
    ADCR6    FLG    0.0EH.2        ; Dummy
    ADCR5    FLG    0.0EH.1        ; Defines each bit of data buffer as ADCR data setting
    ADCR4    FLG    0.0EH.0        ; flag
    ADCR3    FLG    0.0FH.3
    ADCR2    FLG    0.0FH.2
    ADCR1    FLG    0.0FH.1
    ADCR0    FLG    0.0FH.0

    CLR2     ADCCH1, ADCCH0
                                     ; Sets P1B0/ADC0 pin for the A/D converter

START:
    INITFLG  NOT  ADCR3, NOT  ADCR2, NOT  ADCR1, NOT  ADCR0
    INITFLG  NOT  ADCR7, NOT  ADCR6,      ADCR5, NOT  ADCR4
    PUT      ADCR, DBF              ; Sets compare voltage  $V_{REF}$  to  $31.5/64 V_{DD}$ 
    SKT1     ADCCMP                 ; Detects ADCCMP flag, and
    BR       AAA                   ; branches to AAA if False (0)
    BR       BBB                   ; branches to BBB if True (1)

```

### 13.6.2 Successive approximation by binary search method

The A/D converter can compare only one voltage at one time.

To convert an input voltage into a digital signal, therefore, successive approximation must be executed by program.

If the processing time of the successive approximation program differs depending on the input voltage, the relationship with the other processing programs may be undesirable.

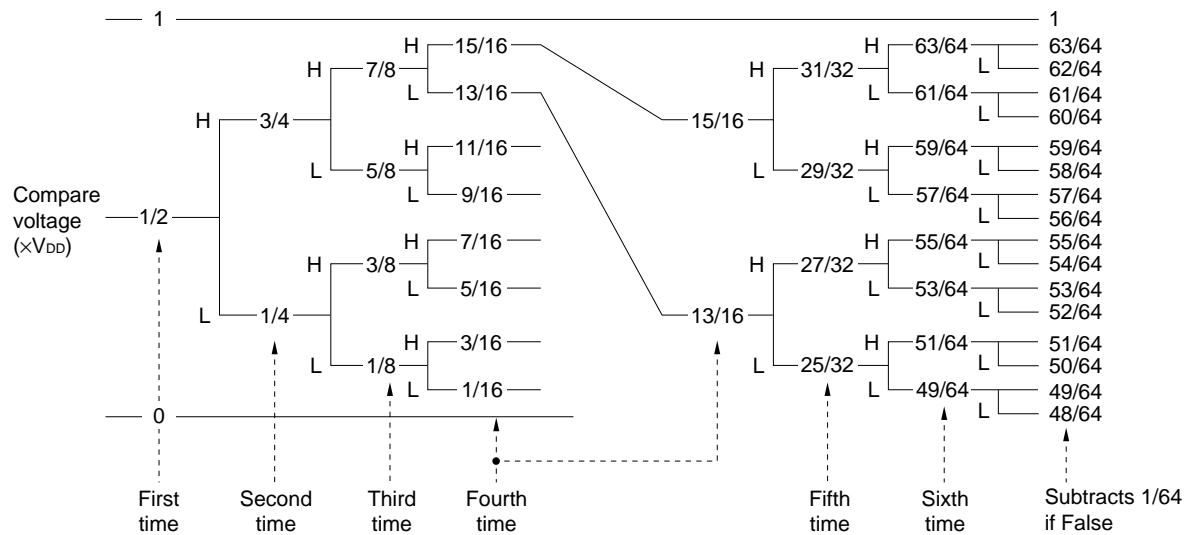
Therefore, use of the binary search method as explained in (1) through (3) below is recommended.

#### (1) Concept of binary search

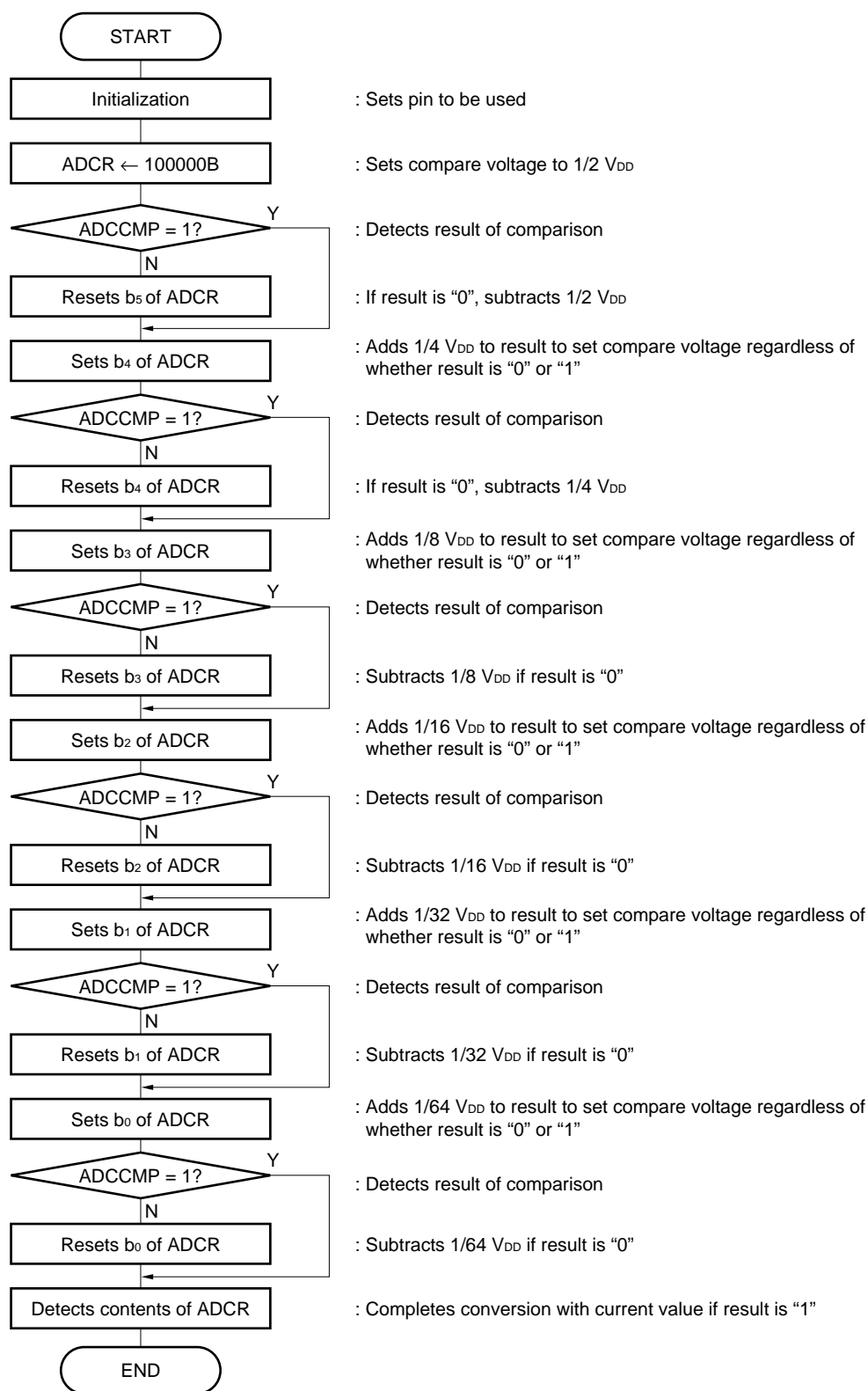
The concept of binary search is explained below.

First, the compare voltage is set to  $1/2 V_{DD}$ . If the result of comparison is True (a high level is input), a voltage of  $1/4 V_{DD}$  is added to the result; if the result of comparison is False (a low level is input), a voltage of  $1/4 V_{DD}$  is subtracted from the result and compared.

Subsequently, the compare voltage is sequentially compared with  $1/8 V_{DD}$  and  $1/16 V_{DD}$  to  $1/64 V_{DD}$ . If the result of comparison is False after comparison has been executed six times,  $1/64 V_{DD}$  is subtracted from the result and comparison is completed.



## (2) Flowchart of binary search method



### (3) Program example of binary search method

#### (a) Method with short conversion time

```

INIT:
    ADCR7    FLG    0.0EH.3    ; Dummy
    ADCR6    FLG    0.0EH.2    ; Dummy
    ADCR5    FLG    0.0EH.1    ; Defines each bit of data buffer as ADCR data setting flag
    ADCR4    FLG    0.0EH.0
    ADCR3    FLG    0.0FH.3
    ADCR2    FLG    0.0FH.2
    ADCR1    FLG    0.0FH.1
    ADCR0    FLG    0.0FH.0

    CLR2     ADCCH1, ADCCH0
                                ; Sets P1B0/ADC0 pin for the A/D converter

START:
    INITFLG  NOT ADCR3, NOT ADCR2, NOT ADCR1, NOT ADCR0
    INITFLG  NOT ADCR7, NOT ADCR6,      ADCR5, NOT ADCR4
    PUT      ADCR, DBF          ; Sets compare voltage to 31.5/64 VDD
    SKT1     ADCCMP             ; Detects ADCCMP and subtracts
    CLR1     ADCR5              ; 32/64 VDD if "0" and adds
    SET1     ADCR4              ; 16/64 VDD
    PUT      ADCR, DBF
    SKT1     ADCCMP             ; Detects ADCCMP and subtracts
    CLR1     ADCR4              ; 16/64 VDD if "0" and adds
    SET1     ADCR3              ; 8/64 VDD
    PUT      ADCR, DBF
    SKT1     ADCCMP             ; Detects ADCCMP and subtracts
    CLR1     ADCR3              ; 8/64 VDD if "0" and adds
    SET1     ADCR2              ; 4/64 VDD
    PUT      ADCR, DBF
    SKT1     ADCCMP             ; Detects ADCCMP and subtracts
    CLR1     ADCR2              ; 4/64 VDD if "0" and adds
    SET1     ADCR1              ; 2/64 VDD
    PUT      ADCR, DBF
    SKT1     ADCCMP             ; Detects ADCCMP and subtracts
    CLR1     ADCR1              ; 2/64 VDD if "0" and adds
    SET1     ADCR0              ; 1/64 VDD
    PUT      ADCR, DBF
    SKT1     ADCCMP             ; Detects ADCCMP and subtracts
    CLR1     ADCR0              ; 1/64 VDD if "0"

    END:

```

} A/D conversion

Number of program steps: 31 steps

Number of execution steps: 31 steps

A/D conversion time: 137.8 μs



**(b) Method with fewer program steps**

```
ADWORK1  MEM      0.00H          ; Work area for changing compare voltage
ADWORK0  MEM      0.01H
```

```

INITFLG NOT ADCCH1, NOT ADCCH0
                                ; Sets P1B0/ADC0 pin for the A/D converter

START:
MOV      DBF1, #0010B          ; Sets compare voltage to initial value of 31.5/64 VDD
MOV      DBF0, #0000B
MOV      ADWORK1, #0001B
MOV      ADWORK0, #0000B

AD_CHECK:
PUT      ADCR, DBF              ; Sets compare voltage VREF
SKT1     ADCCMP                  ; Detects ADCCMP flag
BR       ADIN_L
ADD      DBF0, ADWORK0          ; Increases compare voltage if "1"
ADDC     DBF1, ADWORK1
BR       NEXT_AD

ADIN_L:
SUB      DBF0, ADWORK0          ; Decreases compare voltage if "0"
SUBC     DBF1, ADWORK1

;      NOP                      ; Described to keep A/D conversion time constant

NEXT_AD:
RORC     ADWORK1
RORC     ADWORK0
SKT1     CY                    ; 6 bits completed?
BR       AD_CHECK
PUT      ADCR, DBF
SKT1     ADCCMP
AND      DBF0, #1110B
:

```

} A/D conversion

Number of program steps: 22 steps  
 Number of execution steps: 58 to 63 steps  
 A/D conversion time: 257.8 to 280  $\mu$ s

Where the A/D conversion time is held constant

Number of program steps: 23 steps  
 Number of execution steps: 63 steps  
 A/D conversion time: 280  $\mu$ s

### 13.7 Status on Reset

#### 13.7.1 On power-on reset

All the P1B<sub>1</sub>/ADC<sub>1</sub> and P1B<sub>0</sub>/ADC<sub>0</sub> pins are set in the general-purpose input port mode.

#### 13.7.2 On execution of clock stop instruction

All the P1B<sub>1</sub>/ADC<sub>1</sub> and P1B<sub>0</sub>/ADC<sub>0</sub> pins are set in the general-purpose input port mode.

#### 13.7.3 On CE reset

All the P1B<sub>1</sub>/ADC<sub>1</sub> and P1B<sub>0</sub>/ADC<sub>0</sub> pins are set in the general-purpose input port mode.

## 14. D/A CONVERTER (DAC)

The D/A converter (DAC) outputs its signal by means of PWM (Pulse Width Modulation), which varies the duty factor.

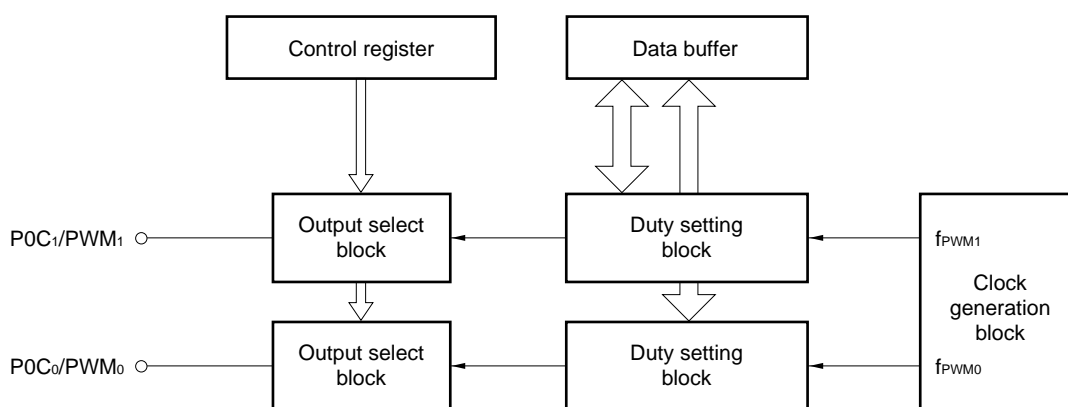
By connecting an external lowpass filter to the D/A converter, digital signals can be converted into analog signals.

### 14.1 Configuration of D/A Converter

Figure 14-1 shows the block diagram of the D/A converter.

As shown in the figure, the D/A converter consists of an output select block and a duty setting block for each pin, and a clock generation block.

Figure 14-1. Block Diagram of D/A Converter



### 14.2 Functional Outline of D/A Converter

The D/A converter outputs a variable-duty signal to each output pin.

The output frequency is 4.4 kHz, and the duty factor can be changed in 256 steps.

The following subsections 14.2.1 through 14.2.3 outline the function of each block of the D/A converter.

#### 14.2.1 Output select blocks

The output select blocks specify whether each pin is used as a general-purpose output port pin or a D/A converter pin.

The mode of each pin is selected by PWM1SEL and PWM0SEL of the PWM mode select register (refer to 14.3).

#### 14.2.2 Duty setting blocks

The duty setting blocks output a signal whose duty factor can be changed in 256 steps.

The duty factor of each output pin is independently set by the PWM data register (PWMR0 or PWMR1: peripheral address 04H or 05H) via the data buffer (refer to 14.4).

#### 14.2.3 Clock generation block

The clock generation block generates a basic clock that is used to set the duty factor of the output signal.

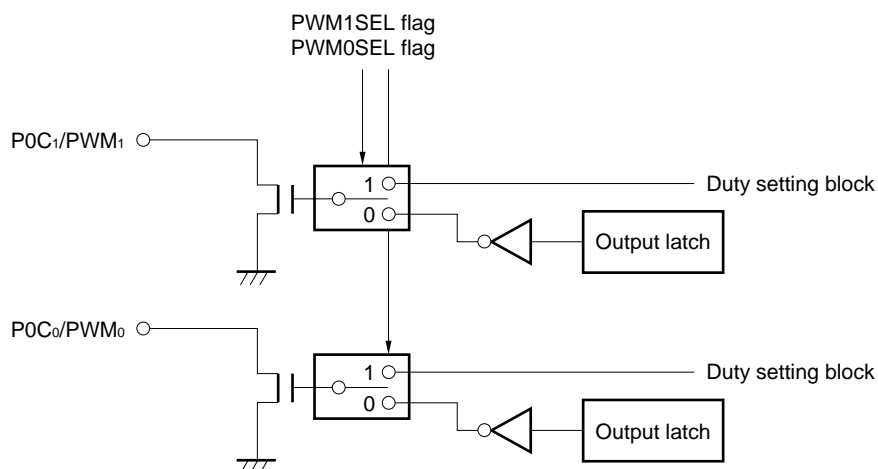
The generated clock frequency  $f_{PWM}$  is 1.125 MHz (refer to 14.4).

### 14.3 Output Select Blocks

#### 14.3.1 Configuration of output select blocks

Figure 14-2 shows the configuration of the output select blocks.

**Figure 14-2. Configuration of Output Select Blocks**



#### 14.3.2 Function of output select blocks

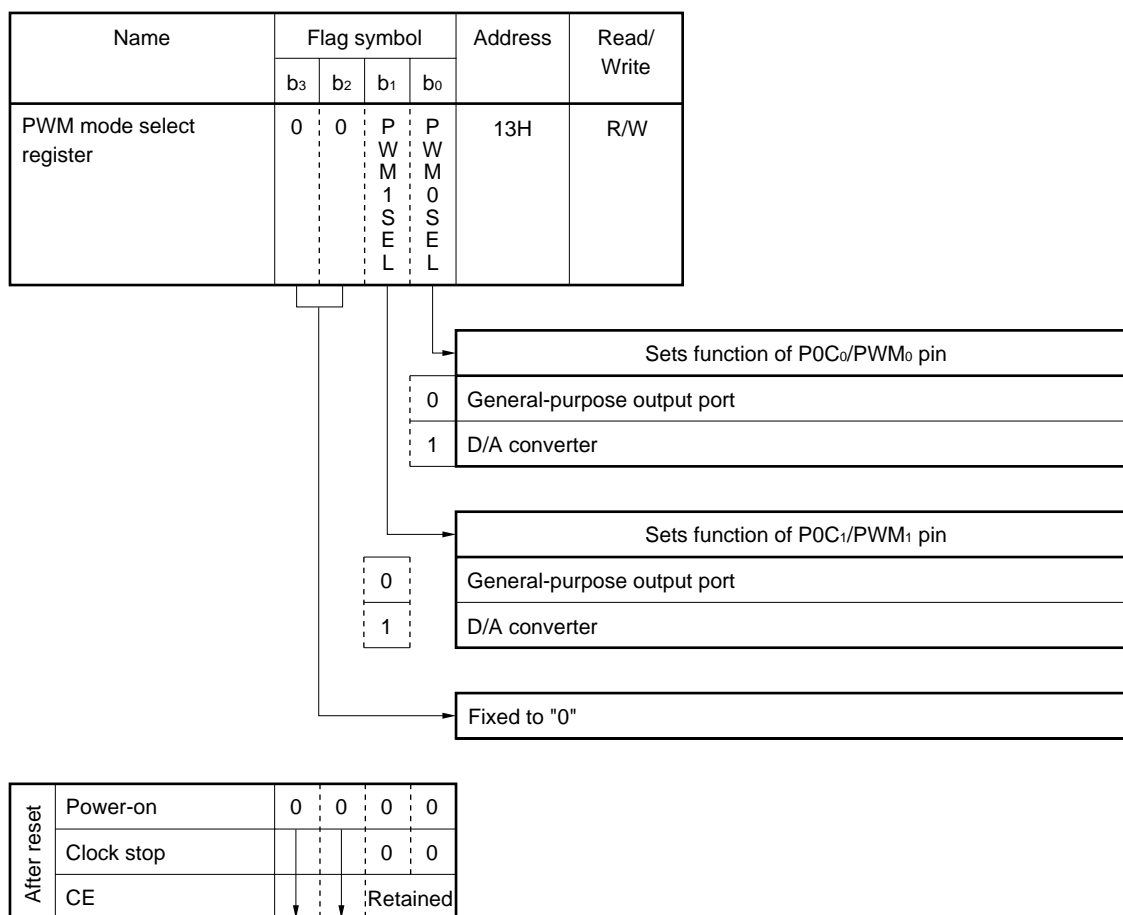
The output select blocks select whether the P1B2/PWM1 and P1B1/PWM0 pins are used as general-purpose output port pins or D/A converter pins.

This selection can be made by the PWM1SEL and PWM0SEL flags of the PWM mode select register. Each pin can be set in the port mode or D/A converter mode independently.

The P0C1/PWM1 and P0C0/PWM0 pins are N-ch open-drain output pins and must be connected with an external pull-up resistor.

The configuration and function of the PWM mode select register is shown in Figure 14-3.

Figure 14-3. Configuration of PWM Mode Select Register

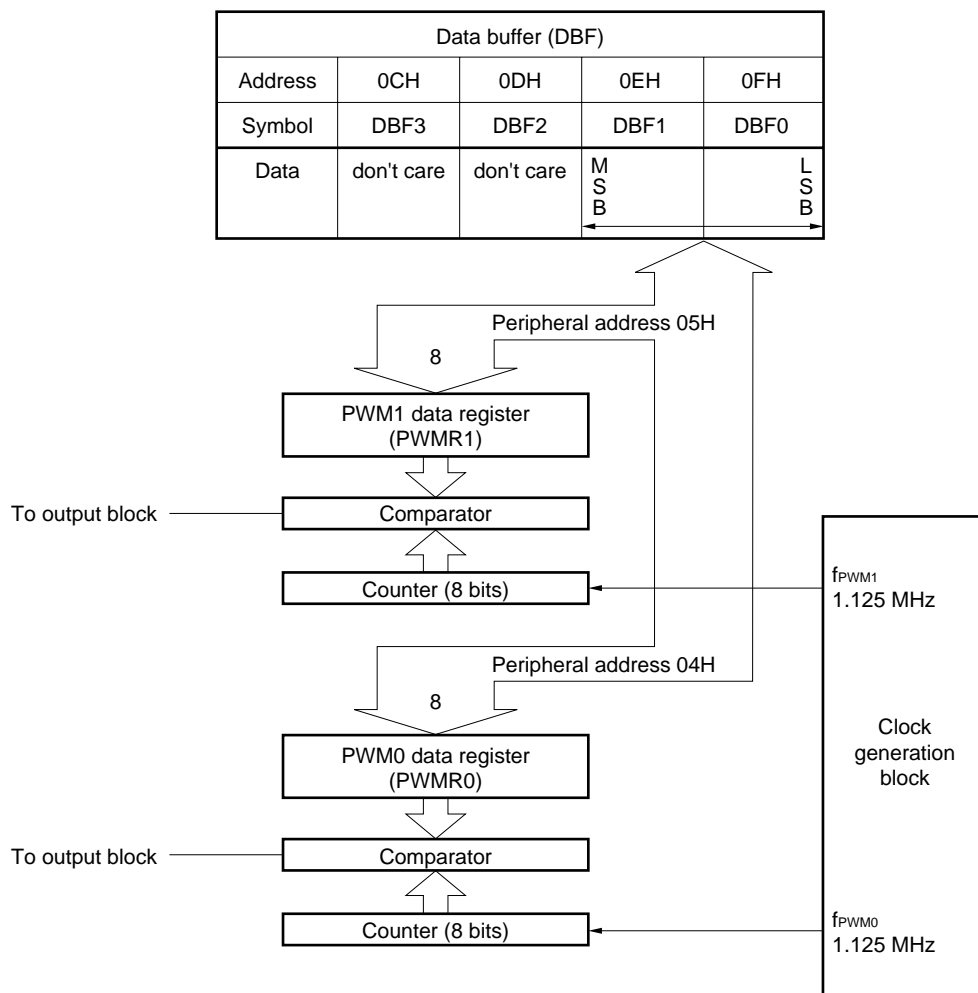


## 14.4 Duty Setting Blocks and Clock Generation Block

### 14.4.1 Configuration of duty setting blocks and clock generation block

Figure 14-4 shows the configuration of the duty setting blocks and clock generation block.

**Figure 14-4. Configuration of Duty Setting Blocks and Clock Generation Block**

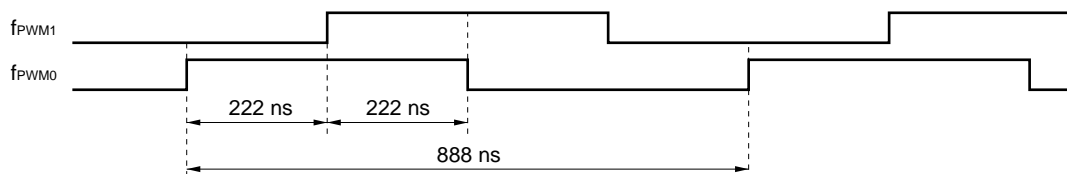


### 14.4.2 Function and configuration of clock generation blocks

The clock generation block outputs the basic clocks ( $f_{PWM1}$  and  $f_{PWM0}$ ) that set the duty factor of each output signal (of PWM<sub>1</sub> and PWM<sub>0</sub> pins).

The output frequency is 1.125 MHz (0.89  $\mu$ s) for both  $f_{PWM1}$  and  $f_{PWM0}$ .

However,  $f_{PWM1}$  and  $f_{PWM0}$  have the following phase difference.



### 14.4.3 Function and operation of duty setting blocks

The duty setting blocks compare the value set to each PWM data register (PWM1 and PWM0) with the value of each basic clock ( $f_{\text{PWM1}}$  and  $f_{\text{PWM0}}$ ) counted by an 8-bit counter, and output a high level if the value of the PWM data register is greater, and a low level if the value of PWM data register is less.

Where the value set to the PWM data register is “x”, the duty factor is as follows.

$$\text{Duty factor: } D = \frac{x + 0.25}{256} \times 100\%$$

0.25 is an offset. A high level is output even when  $x = 0$ .

Because the basic clock is 1.125 MHz, the frequency and cycle of the output signal are as follows.

$$\text{Frequency: } f = \frac{1.125 \text{ MHz}}{256} = 4.3945 \text{ kHz}$$

$$\text{Cycle: } T = \frac{256}{1.125 \text{ MHz}} = 227.6 \mu\text{s}$$

An independent value can be set to each PWM data register via the data buffer.

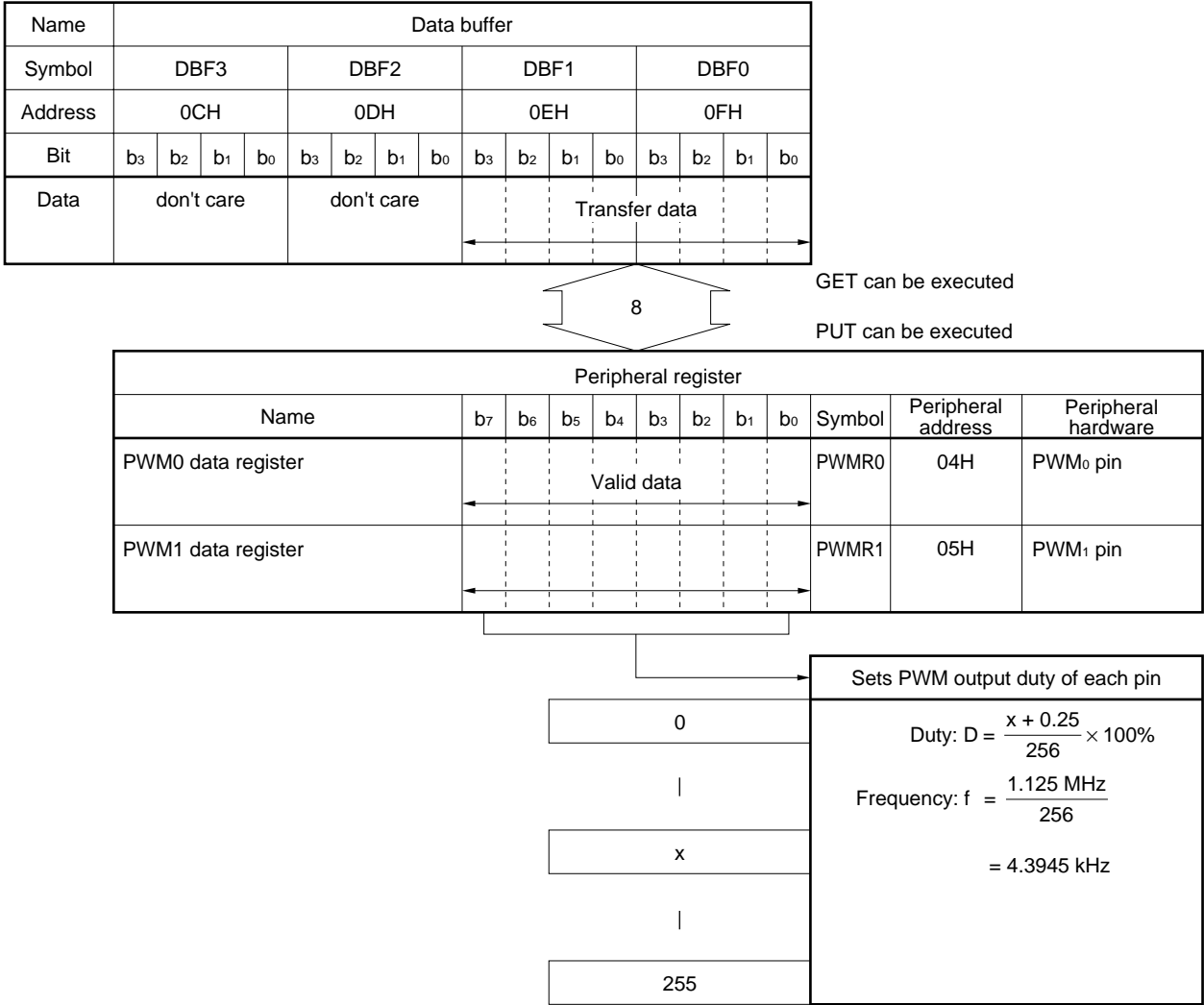
In other words, each pin can output a signal with an independent duty factor.

The following subsections 14.4.4 and 14.4.5 explain the configuration and function of each PWM data register, and the relationship between the output waveform and duty factor of each pin.

14.4.4 Configuration and function of each PWM data register

The function of each PWM data register is illustrated below.

The PWM data register sets the duty factor of a D/A converter (PWM output) output signal.

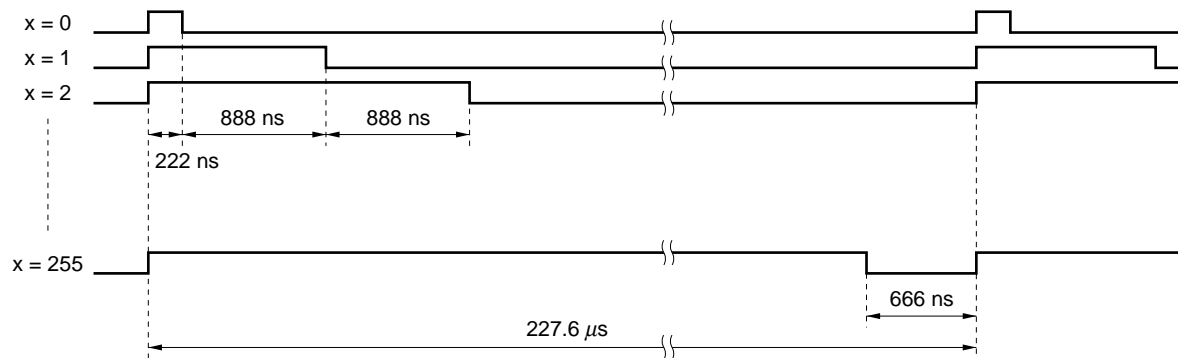




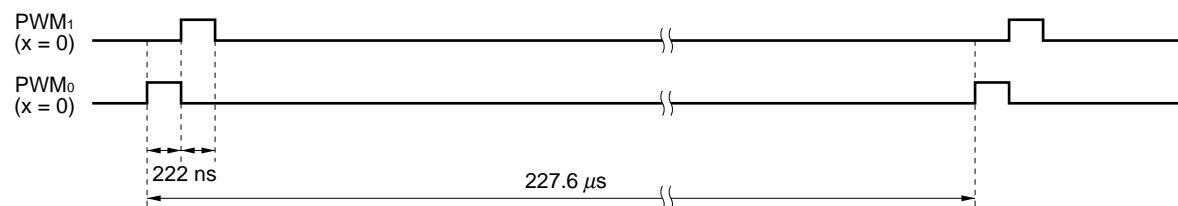
#### 14.4.5 Relationship of output waveform and each pin of D/A converter

(1) shows the relationship between the output waveform and duty factor. (2) shows the relationship of the output waveform of each pin.

##### (1) Duty factor and output waveform



##### (2) Output waveform of each pin



#### 14.5 Cautions When Using D/A Converter

- (1) The initial PWM output setting following the power on application is made in the following procedure. This is because the PWM data register is undefined so that data should be set beforehand.
  - <1> Set the value of PWM data register
  - <2> Set the PWMnSEL flag
- (2) Do not overwrite the data of PWM data register during PWM operation. The output of the correct duty for one cycle (227.6 μs) cannot be obtained.

## 14.6 Status on Reset

### 14.6.1 On power-on reset

The P0C<sub>1</sub>/PWM<sub>1</sub> and P0C<sub>0</sub>/PWM<sub>0</sub> pins are set in the general-purpose output port mode.

The output value is undefined.

The value of each PWM data register is undefined.

### 14.6.2 On execution of clock stop instruction

The P0C<sub>1</sub>/PWM<sub>1</sub> and P0C<sub>0</sub>/PWM<sub>0</sub> pins are set in the general-purpose output port mode.

The output value is the previous contents of the output latch.

Each PWM data register retains the previous value.

### 14.6.3 On CE reset

The P0C<sub>1</sub>/PWM<sub>1</sub> and P0C<sub>0</sub>/PWM<sub>0</sub> pins retain the previous output status.

Therefore, the pin used for the D/A converter retains the current PWM output.

### 14.6.4 In halt status

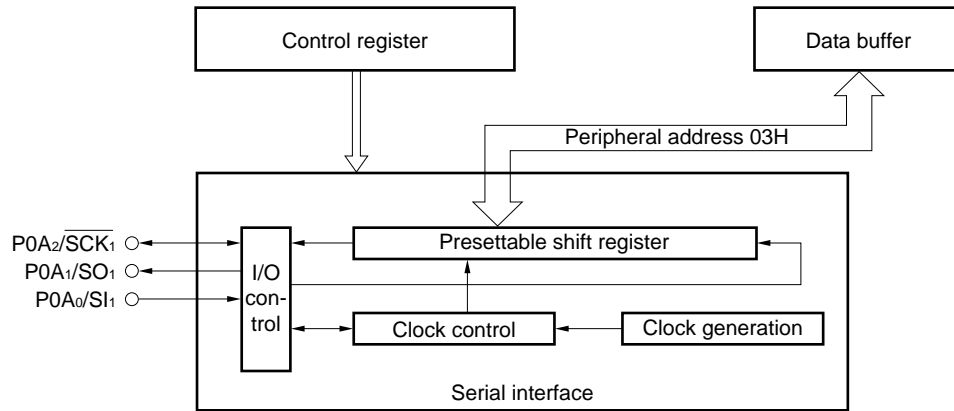
The P0C<sub>1</sub>/PWM<sub>1</sub> and P0C<sub>0</sub>/PWM<sub>0</sub> pins retain the previous output status.

Therefore, the pin used for the D/A converter retains the current PWM output.

## ★ 15. SERIAL INTERFACE

The serial interface is used to transfer 8-bit serial data with an external device.

**Figure 15-1. Block Diagram of Serial Interface**



## ★ 15.1 Configuration of Serial Interface

Figure 15-2 shows the configuration of the serial interface.

As shown in the figure, the shift clock control block of the serial interface consists of a clock I/O pin control block, clock generation block, wait control block, and clock count block.

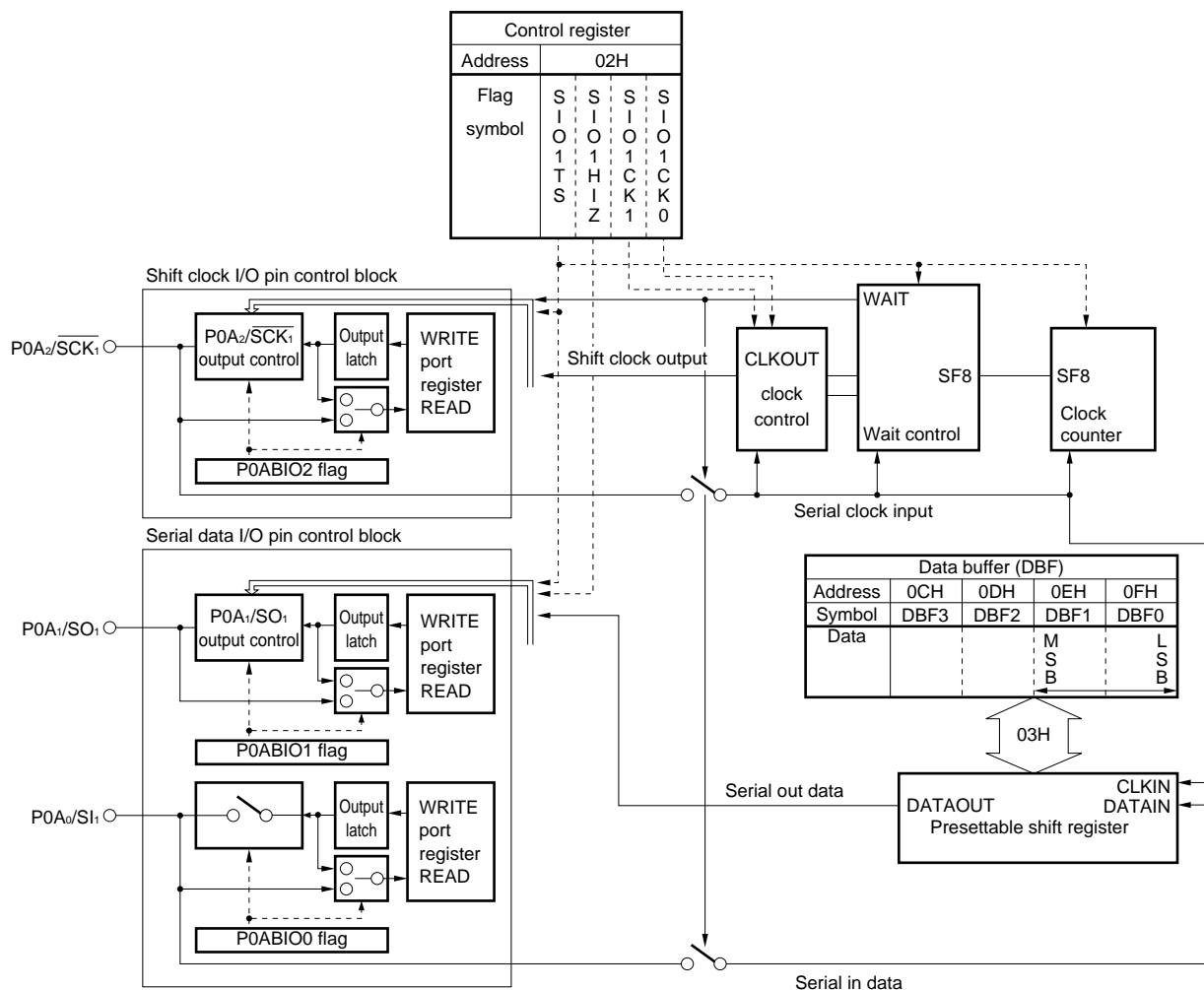
The serial data control block consists of a serial data I/O pin control block and a presettable shift register.

These blocks are controlled by the corresponding flags of the control registers.

Data is written to or read from the presettable shift register via the data buffer.

The following section 15.2 outlines each block.

★ **Figure 15-2. Configuration of Serial Interface**



## 15.2 Functional Outline of Serial Interface

The serial interface uses the P0A<sub>2</sub>/ $\overline{\text{SCK}}_1$ , P0A<sub>1</sub>/SO<sub>1</sub>, and P0A<sub>0</sub>/SI<sub>1</sub> pins.

The serial interface can select the internal clock or an external clock, and can execute receive and transmit

The following subsections 15.2.1 to 15.2.6 outline the functions of the respective blocks of the serial interface.

For details of each block, refer to 15.3 to 15.7.

### 15.2.1 Shift clock I/O pin control block

This block selects the shift clock I/O pin.

The shift clock I/O pin is selected by the serial I/O mode select register.

Refer to 15.3.

### 15.2.2 Serial data I/O pin control block

This block selects the serial data I/O pin.

The serial data I/O pin is selected by the serial I/O mode select register.

Refer to 15.3.

### 15.2.3 Clock generation block

This block selects the clock frequency of the shift clock and controls the shift clock output timing.

The shift clock frequency is selected by the serial I/O mode select register.

Refer to 15.4.

### 15.2.4 Clock counter

The clock counter counts the number of rising edges of the clock output by the shift clock output pin and outputs a signal at the eighth clock (SF8 signal).

The SF8 signal is used to make serial communication wait (pause).

Refer to 15.5.

### 15.2.5 Presettable shift register (SIO1SFR)

This shift register sets serial out data and stores serial in data.

It performs a shift operation by using the clock of the shift clock I/O pin and inputs/outputs data.

The output data is set and the input data is read via the data buffer.

Refer to 15.6.

### 15.2.6 Wait control block

This block places or releases serial communication in or from the wait status.

Serial communication is placed in or released from the wait status by the serial I/O mode select register.

Refer to 15.7.

### 15.3 Shift Clock and Serial Data I/O Pin Control Blocks

The shift clock and serial data I/O pin control blocks set the pins of the serial interface and control the transmit/receive operations.

These control operations are specified by the serial I/O mode select register.

15.3.1 shows the configuration and function of the serial I/O mode select register.

15.3.2 indicates the status of each pin set by the serial I/O mode select register.

#### 15.3.1 Configuration and function of serial I/O mode select register

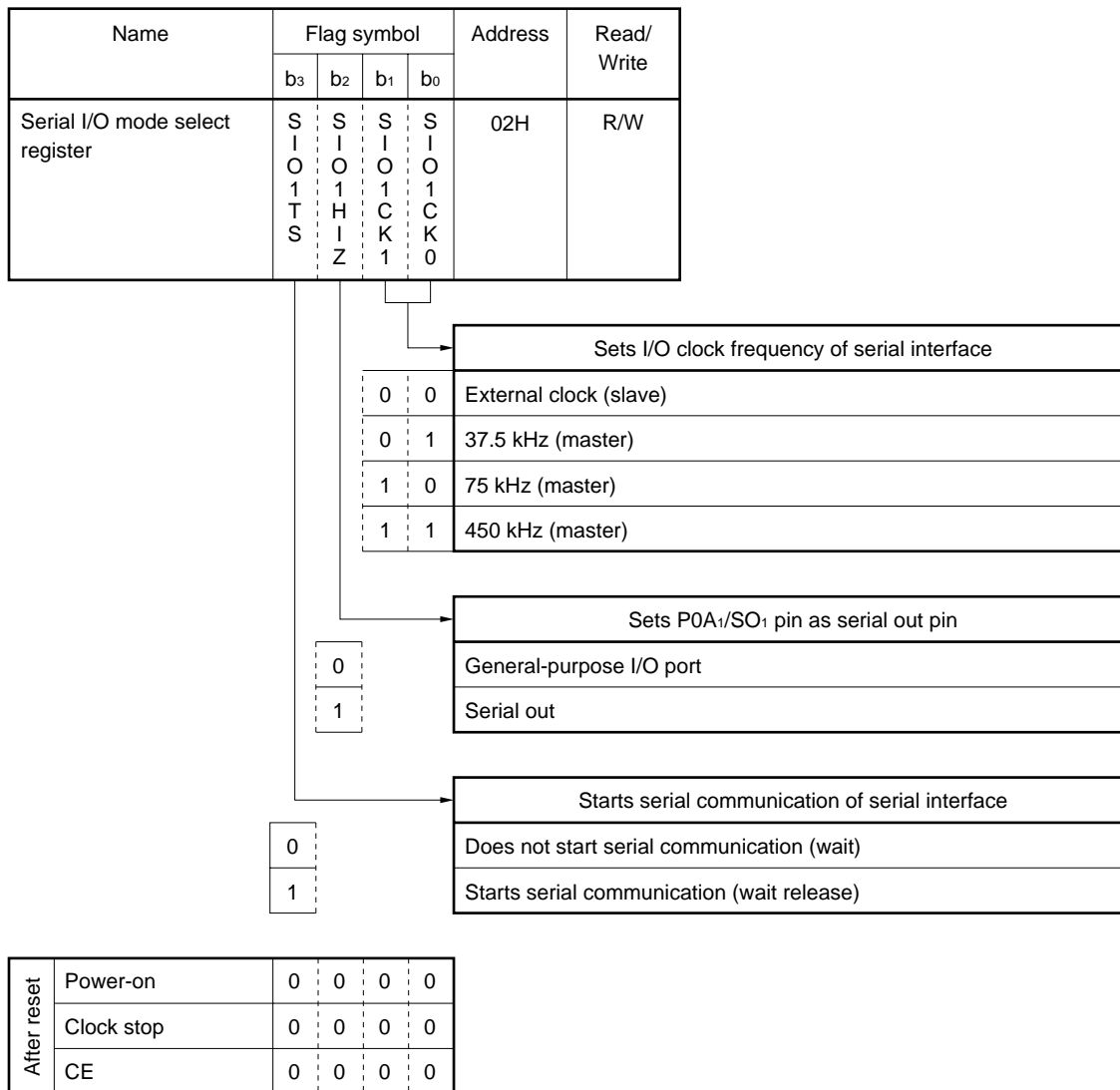
The configuration and function of the serial I/O mode select register are illustrated below.

The SIO1CK1 and SIO1CK0 flags are used to select the internal clock or an external clock and to set the frequency of the internal clock.

For details of the clock, refer to 15.4.

The SIO1TS flag places or releases the serial interface in or from the wait status.

For the wait operation, refer to 15.7.



### 15.3.2 Pin status setting by serial I/O mode select register

Table 15-1 shows the status of each pin set by the serial I/O mode select register.

As shown in this table, the I/O select flag of each pin must also be manipulated to set each pin.

For the details of the I/O select flag, refer to **10. GENERAL-PURPOSE PORTS**.

**Table 15-1. Pin Status Setting by Serial I/O Mode Select Register**

SIO1MODE					Pin					
Communication mode	b <sub>2</sub>	Setting of serial output	b <sub>1</sub>	b <sub>0</sub>	Clock direction	Pin symbol	I/O select flag of each pin			Pin setting status
	S I O 1 H I Z		S I O 1 C K 1	S I O 1 C K 0			P 0 A B I O 2	P 0 A B I O 1	P 0 A B I O 0	
3-wire serial I/O			0	0	External clock	P0A <sub>2</sub> /SCK <sub>1</sub>	0			Wait : General-purpose input port Wait released : External clock input
							1			Wait : General-purpose output port Wait released : External clock input
			0	1	Internal clock		0			Wait : General-purpose input port Wait released : Internal clock output
			1	0						
			1	1			1			Wait : General-purpose output port Wait released : General-purpose input port
			0	General-purpose port				P0A <sub>1</sub> /SO <sub>1</sub>		0
	1	Serial output		1					General-purpose output port	
				0					General-purpose input port	
				1					Serial output	
						P0A <sub>0</sub> /SI <sub>1</sub>		0		Serial input
								1		General-purpose output port

## 15.4 Clock Generation Block

The clock generation block generates the clock when the internal clock is used (i.e., when a master operation is performed) and controls the clock output timing.

The frequency  $f_{sc}$  of the internal clock is set by using the SIO1CK1 and SIO1CK0 flags of the serial I/O mode select register.

The shift clock is successively output until the value of the clock counter, which is explained in 15.5, reaches "8".

The following subsection 15.4.1 explains the clock output waveform and clock generation timing.

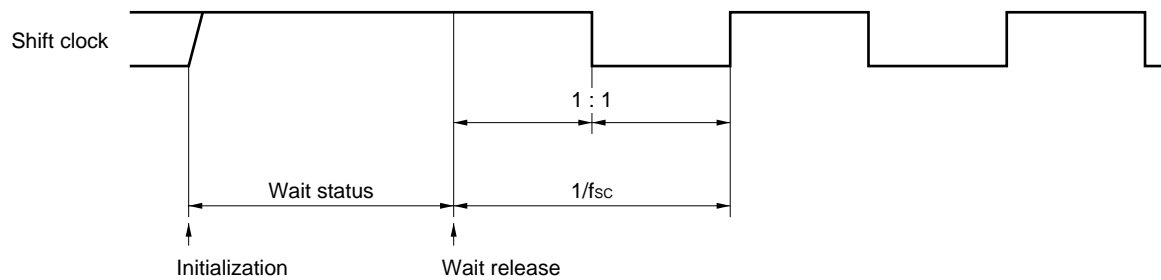
### 15.4.1 Internal shift clock generation timing

#### (1) On releasing wait status from initial status

The initial status is the status in which the internal clock is selected and the  $P0A_2/\overline{SCK}_1$  pin is set in the output mode.

A high level is output to the  $P0A_2/\overline{SCK}_1$  pin in the wait status.

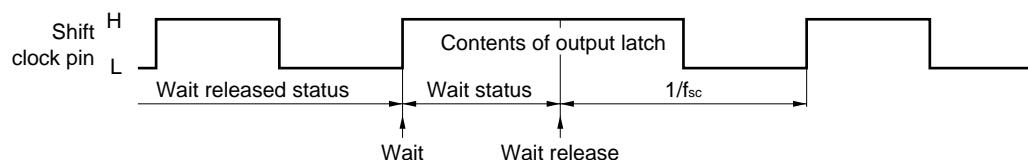
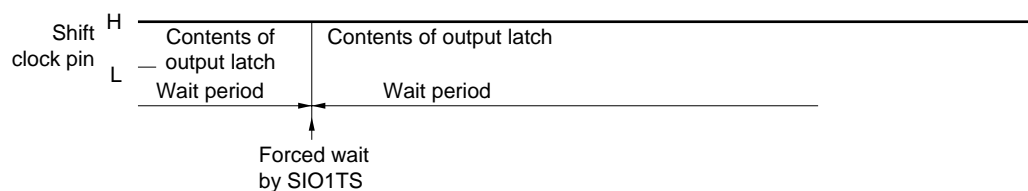
Wait release and clock selection can be made simultaneously.



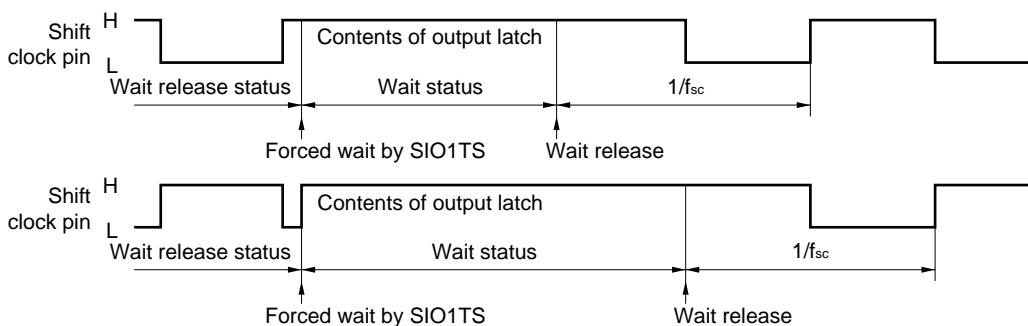


**(2) When wait operation is performed**

For details of the wait operation, refer to 15.7.

**(a) Wait status with value of clock counter reaching "8" (normal operation)****(b) If forced wait is executed in wait status****(c) If forced wait is executed when wait status is released**

At this time, the clock counter is reset.

**(d) If wait status is released in wait release status**

The clock output waveform is not changed at this time.

The clock counter is not reset. However, do not change the clock frequency during wait release.

## 15.5 Clock Counter

The clock counter is a wrap-around counter that counts the number of the shift clocks output from or input to the shift clock (P0A2/ $\overline{\text{SCK}}_1$ ) pin.

The clock counter directly reads the status of the shift clock pin. At this time, whether the clock is the internal clock or an external clock is not identified.

The clock counter does not operate in the wait status of serial communication.

When the value of the clock counter is "8", serial communication is placed in the wait status at the rising edge of the shift clock.

The contents of the clock counter cannot be directly read by program.

The following subsections 15.5.1 and 15.5.2 explain the operation of the clock counter and the conditions under which the clock counter is reset.

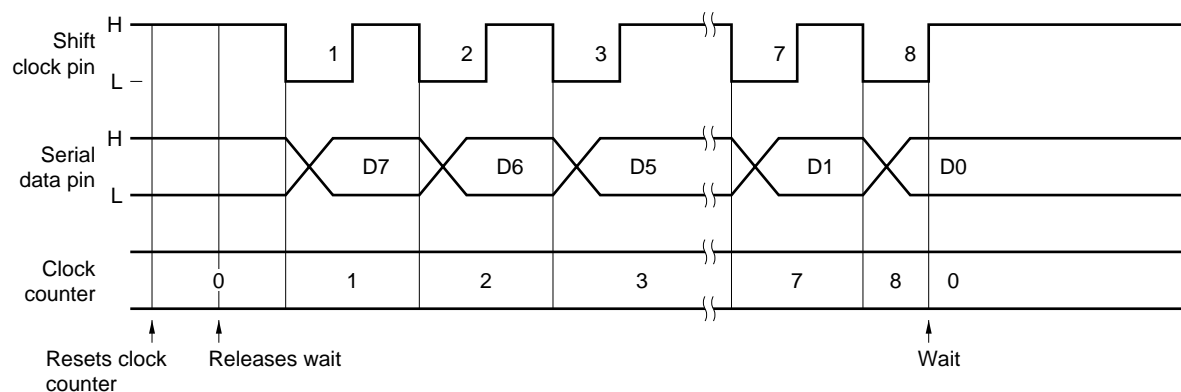
### 15.5.1 Operation of clock counter

Figure 15-3 shows the operation of the clock counter.

The initial value of the clock counter is "0". The value of the clock counter is incremented by one each time the falling of the shift clock pin is detected. When the value of the clock counter has been incremented to "8", the clock counter is reset to "0" at the next rising edge of the shift clock pin.

Serial communication is placed in the wait status when the clock counter has been reset to "0".

**Figure 15-3. Operation of Clock Counter**



### 15.5.2 Clock counter reset condition

The clock counter is reset to 0 when any of the following conditions (1) through (5) is satisfied.

- (1) On power-on reset
- (2) On execution of the clock stop instruction
- (3) When "0" is written to the SIO1TS flag (forced wait)
- (4) When the shift clock rises while the value of the clock counter is "8" with the wait status released
- (5) On CE reset

## 15.6 Presetable Shift Register (SIO1SFR)

The presetable shift register is an 8-bit shift register that writes serial out data and reads serial in data.

Data is written to or read from the presetable shift register via the data buffer by using the PUT or GET instruction.

15.6.1 shows the configuration of the presetable shift register and its relationship with the data buffer.

The presetable shift register performs its shift operation in synchronization with the clock applied to the shift clock ( $P0A_2/\overline{SCK_1}$ ) pin.

At this time, the contents of the most significant bit (MSB) of the presetable shift register are output to the serial data output pin in synchronization with the fall of the shift clock, and the least significant bit (LSB) of the presetable shift register is read in synchronization with the rise of the clock.

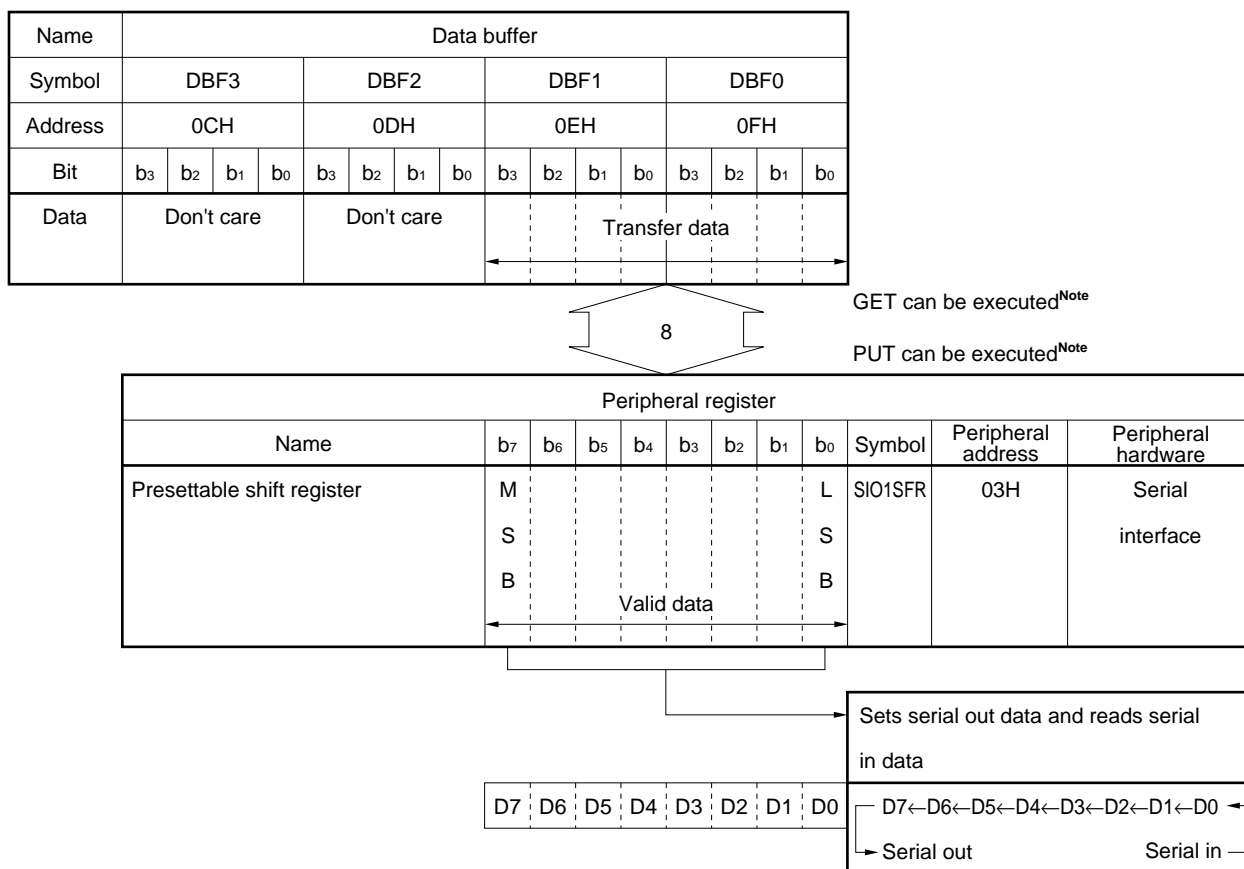
15.6.2 explains the points to be noted when writing or reading data to or from the presetable shift register.

The presetable shift register does not shift data in the wait status.

For details of the operation in each serial communication mode, refer to **15.8**.

### 15.6.1 Configuration of presetable shift register and its relationship with data buffer

The configuration of the presetable shift register and its relationship with the data buffer are shown below.



**Note** If PUT or GET is executed in serial communication mode, data may be corrupted. For details, refer to **15.6.2 Notes on setting and reading data**.

### 15.6.2 Notes on setting and reading data

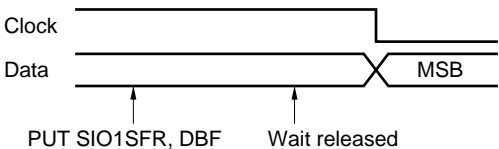
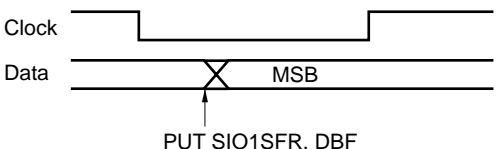
Data is written to the presettable shift register by the PUT SIO1SFR, DBF instruction.

Data is read from the register by the GET DBF, SIO1SFR instruction.

Set or read data to or from the register in the wait status. While the wait status is released, the data may not be correctly set or read depending on the status of the shift clock pin.

Table 15-2 indicates the timing of setting and reading data and points to be noted.

**Table 15-2. Reading (GET) and Writing (PUT) Data from/to Presettable Shift Register and Notes**

Status on Execution of PUT/GET		Status of Shift Clock Pin	Operation of Presettable Shift Register (SIO1SFR)
Wait status	Read (GET)	With external clock: Floated	Normally read.
	Write (PUT)	With internal clock: Normally the output latch value is used at high level.	Normally written. Content of MSB is output at falling edge of shift clock when wait status is released next time (during transfer operation). 
Wait released status	Read (GET)	Low level	Normally read.
		High level	Cannot be read normally. Contents of SIO1SFR are destroyed.
	Write (PUT)	Low level	Normally written. Contents of MSB are output when PUT instruction is executed. Clock counter is not reset. 
		High level	Cannot be written normally. Contents of SIO1SFR are destroyed.

## 15.7 Wait Control Block

The wait control block controls communication of the serial interface by placing or releasing communication in or from the wait status.

The wait control block is controlled by the SIO1TS flag of the serial I/O mode select register.

The following subsection 15.7.1 explains the wait operation and points to be noted.

### 15.7.1 Wait operation and notes

In the wait status, the clock generation block and presettable shift register stop operation, and serial communication pauses.

Therefore, serial communication can be started when the wait status is released.

The wait status is released when "1" is written to the SIO1TS flag.

When "1" is written to this flag, the internal clock is output to the shift clock output pin (during master operation), and presettable shift register and clock counter start operating.

If the shift clock rises when the value of the clock counter is "8", the wait status is set. At this time, the SIO1TS flag is automatically reset to 0.

The operating status of serial communication can be checked by detecting the content of the SIO1TS flag while the wait status is released.

Therefore, data is read or set after "1" has been written to the SIO1TS flag, serial communication has been started, and then clearing of the SIO1TS flag to 0 has been detected.

If data is written to (by PUT instruction) or read from (by GET instruction) the presettable shift register while the wait status is released, the correct data may not be written or read.

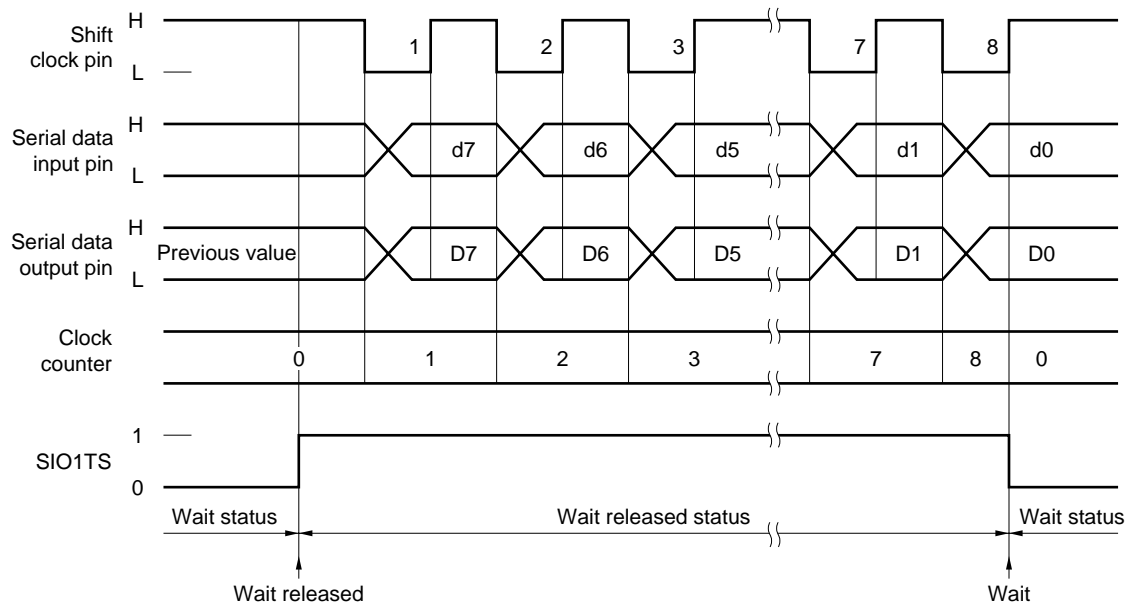
For details, refer to **15.6.2 Notes on setting and reading data**.

If "0" is written to the SIO1TS flag while the wait status is released, the wait status is set. This is called forced wait. When forced wait is executed, the clock counter is reset to 0.

Figure 15-4 shows an example of the wait operation.

★

Figure 15-4. Example of Wait Operation



When the wait is released, the serial data is output at the next falling edge of the clock, and the status becomes the wait released status.

When the shift clock has been input eight times, the shift clock pin outputs a high level, and the clock counter and presetable shift register stop operation.

If data is written to or read from the presetable shift register while the wait status is released and the shift clock pin is high, the correct data may not be set or read.

If data is written to the presetable shift register while the wait status is released and the shift clock pin is low, the contents of the MSB are output to the serial data output pin as soon as the PUT instruction has been executed.

If a forced wait is executed while the wait status is released, the wait status is set and the clock counter is reset to 0 as soon as "0" has been written to the SIO1TS flag.

## 15.8 Outline of Serial Interface Operation

Table 15-3 shows an outline of the serial interface operation in each mode.

**Table 15-3. Outline of Serial Interface Operation in Each Mode**

Operation Mode Item		3-Wire Serial I/O Mode			
		Slave Operation SIO1CK1 = SIO1CK0 = 0		Master Operation SIO1CK1 = SIO1CK0 = other than 0	
Setting status of each pin	P0A <sub>2</sub> /SCK <sub>1</sub>	Wait	Wait released	Wait	Wait released
		When P0ABIO2 = 0 Floating General-purpose input port When P0ABIO2 = 1 General-purpose output port Outputs contents of output latch	Regardless of P0ABIO2 Floating Wait external clock input	When P0ABIO2 = 0 Floating General-purpose input port When P0ABIO2 = 1 General-purpose output port Outputs contents of output latch	Regardless of P0ABIO2 Outputs internal clock
	P0A <sub>1</sub> /SO <sub>1</sub>	When SIO1HIZ = 0	When SIO1HIZ = 1	When SIO1HIZ = 0	When SIO1HIZ = 1
		When P0ABIO1 = 0 General-purpose input port Floating When P0ABIO1 = 1 General-purpose output port Outputs contents of output latch	When P0ABIO1 = 0 General-purpose input port Floating When P0ABIO1 = 1 Outputs serial data	When P0ABIO1 = 0 General-purpose input port Floating When P0ABIO1 = 1 General-purpose output port Outputs contents of output latch	When P0ABIO1 = 0 General-purpose input port Floating When P0ABIO1 = 1 Outputs serial data
	P0A <sub>0</sub> /SI <sub>1</sub>	When P0ABIO0 = 0 Floating Waits for external data When P0ABIO0 = 1 General-purpose output port. Outputs contents of output latch			
Operation of clock counter		Incremented at falling edge of $\overline{\text{SCK}}_1$ pin			
Operation of presetable shift register (SIO1SFR)		Output When SIO1HIZ = 1 Shifts data from MSB and outputs it to SO <sub>1</sub> pin at falling edge of $\overline{\text{SCK}}_1$ pin When SIO1HIZ = 0 Does not output data Input Shifts data of SI <sub>1</sub> pin from LSB and inputs it at rising edge of $\overline{\text{SCK}}_1$ pin regardless of P0ABIO0 However, the contents of output latch are output to SI <sub>1</sub> pin when P0ABIO0 = 1			
Wait operation		Serial communication is started when "1" is written to SIO1TS. SIO1TS is reset to 0 at rising edge of shift clock when value of clock counter is "8". For the operation of each pin, refer to above.			

## 15.9 Status of Serial Interface on Reset

### 15.9.1 On power-on reset

Each pin is set in the general-purpose input port mode (floating output).  
The value of the presetable shift register is undefined.

### 15.9.2 On execution of clock stop instruction

Each pin is set in the general-purpose input port mode (floating output).  
The presetable shift register retains the previous value.

### 15.9.3 On CE reset

Each pin is set in the general-purpose input port mode (floating output).  
The presetable shift register retains the previous value.

### 15.9.4 In halt status

Each pin retains the current status.

If the internal clock is used (master operation) at this time, the clock is not output after the HALT instruction has been executed.

To use the internal clock, therefore, the HALT instruction must be executed after communication has been completed.

If an external clock is forcibly input, the serial interface functions even when the internal clock is used.

If the external clock is used (slave operation), the operation continues even when the HALT instruction has been executed.



## 16. PLL FREQUENCY SYNTHESIZER

The PLL (Phase Locked Loop) frequency synthesizer is used to lock the frequency in the MF (Medium Frequency), HF (High Frequency), and VHF (Very High Frequency) bands to a specific frequency by comparing phase differences.

### 16.1 Configuration of PLL Frequency Synthesizer

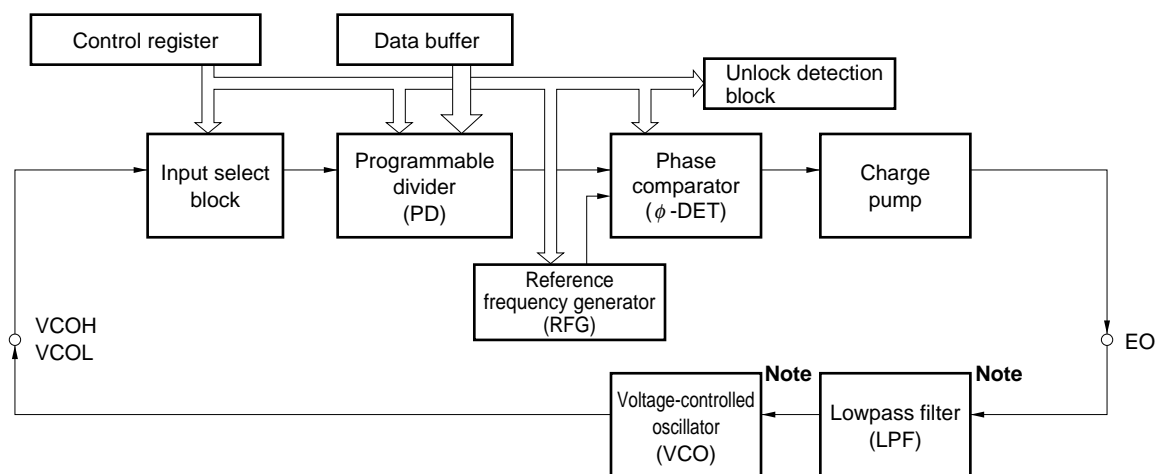
Figure 16-1 shows the block diagram of the PLL frequency synthesizer.

As shown in the figure, the PLL frequency synthesizer consists of an input select block, programmable divider (PD), phase comparator ( $\phi$ -DET), reference frequency generator (RFG), and charge pump.

By connecting these blocks with an external lowpass filter (LPF) and voltage-controlled oscillator (VCO), a PLL frequency synthesizer is organized.

★

**Figure 16-1. Block Diagram of PLL Frequency Synthesizer**



**Note** External circuit

## 16.2 Functional Outline of PLL Frequency Synthesizer

The PLL frequency synthesizer divides a signal input from the VCOH or VCOL pin by using the programmable divider and outputs a phase difference from the reference frequency from the EO pin.

The PLL frequency synthesizer operates only when the CE pin is high, and is disabled when the CE pin is low.

For the details on the disabled status of the PLL frequency synthesizer, refer to **16.6**.

The following subsections 16.2.1 through 16.2.6 outline the function of each block of the PLL frequency synthesizer.

### 16.2.1 Input select block

This block selects the pin from which a signal output from an external voltage-controlled oscillator is input.

As the input pin, the VCOH or VCOL pin is selected by the PLL mode select register (RF address 21H).

For details, refer to **16.3**.

### 16.2.2 Programmable divider

The programmable divider divides the signal input from the VCOH or VCOL pin at the division ratio set by the program.

Two types of division modes can be selected: direct division and pulse swallow modes.

The division mode is selected by the PLL mode select register.

The division ratio is set by the PLL data register (PLLR: peripheral address 41H) via the data buffer.

For details, refer to **16.3**.

### 16.2.3 Reference frequency generator

This generator generates a reference frequency to be compared by the phase comparator.

Twelve types of reference frequencies can be selected by using the PLL reference clock select register (RF address 31H).

For details, refer to **16.4**.

### 16.2.4 Phase comparator and unlock detection block

The phase comparator compares the division signal output by the programmable divider with the signal from the reference frequency generator, and outputs a phase difference.

The unlock detection block detects the unlock status of the PLL.

The unlock status of the PLL is detected by the PLL unlock FF judge register (RF address 05H).

For details, refer to **16.5**.

### 16.2.5 Charge pump

The charge pump outputs the signal output by the phase comparator to the EO pin as a high-level, low-level, or floating signal.

For details, refer to **16.5**.

## 16.3 Input Select Block and Programmable Divider

### 16.3.1 Configuration of input select block and programmable divider

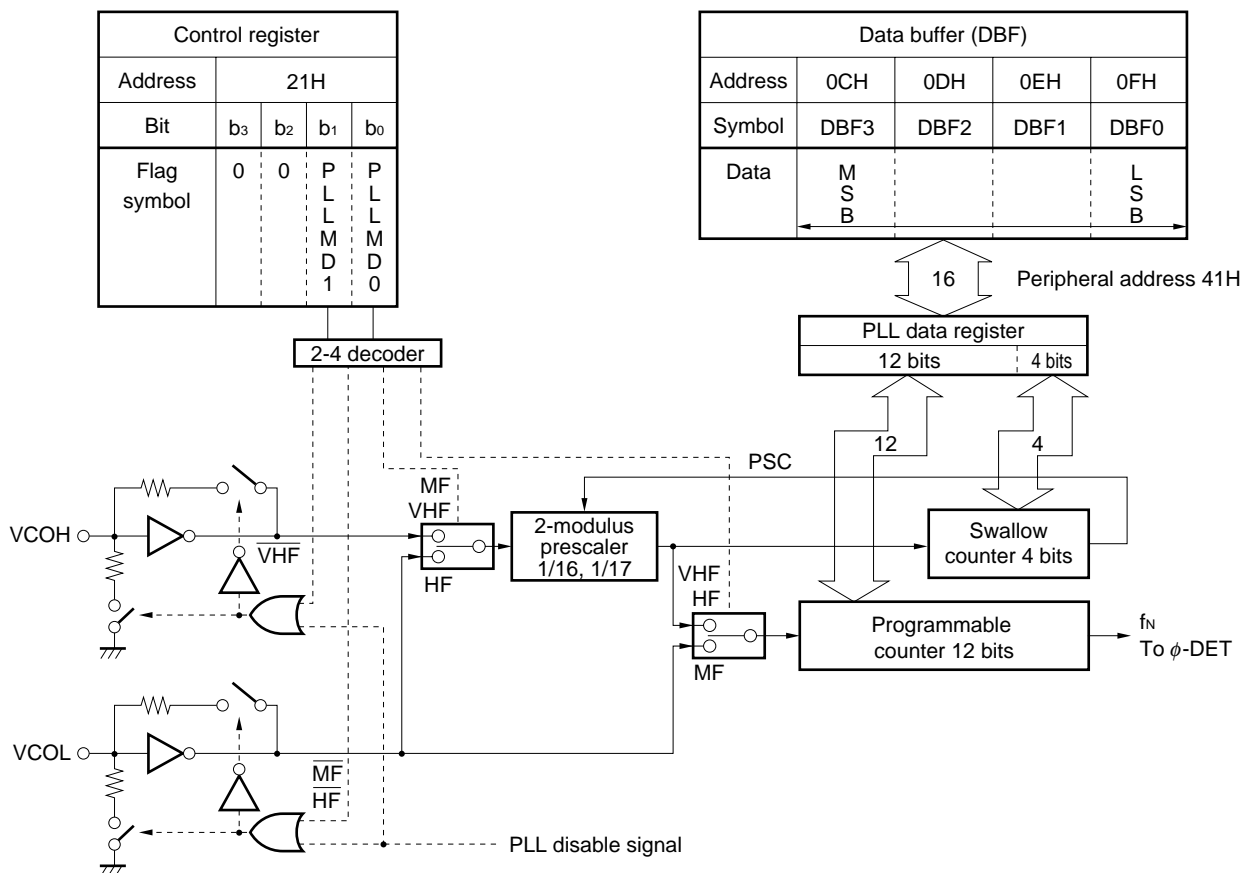
Figure 16-2 shows the configuration of the input select block and programmable divider.

As shown in the figure, the input select block consists of the VCOH and VCOL pins, and the amplifiers of the respective pins.

The programmable divider consists of a 2-modulus prescaler, swallow counter, programmable counter, and division mode select switch.

★

**Figure 16-2. Configuration of Input Select Block and Programmable Divider**



### 16.3.2 Functions of input select block and programmable divider

The input select block and programmable divider select the input pin and division mode of the PLL frequency synthesizer.

As the input pin, the VCOH or VCOL pin can be selected.

The selected pin goes into an intermediate-potential state (approx.  $1/2 V_{DD}$ ). The pin not selected is internally pulled down.

These pins input signals via an AC amplifier, and the DC component of the input signal must be cut off by connecting a capacitor to the pin in series.

Either the direct division mode or pulse swallow mode can be selected as the division mode.

The programmable counter divides the signal input from the VCOH or VCOL pin in a specified division mode according to the values set to the swallow counter and programmable counter.

Table 16-1 show the input pins (VCOH and VCOL) and division modes.

The input pin and division mode to be used are selected by the PLL mode select register.

16.3.3 explains the configuration and function of the PLL mode select register.

The division ratio is set to the programmable divider by the PLL data register via the data buffer.

16.3.4 explains the programmable divider and PLL data register.

**Table 16-1. Input Pins and Division Modes**

Division Mode	Pin	Input Frequency (MHz)	Input Amplitude ( $V_{P-P}$ )	Settable Division Ratio	Division Ratio Settable in Data Buffer
Direct division (MF)	VCOL	0.5 to 20	0.3	16 to $2^{12} - 1$	010xH-FFFxH (x: lower 4 bits are arbitrary)
Pulse swallow (HF)	VCOL	5 to 30	0.3	256 to $2^{16} - 1$	0100H-FFFFH
Pulse swallow (VHF)	VCOH	50 to 150	0.3	256 to $2^{16} - 1$	0100H-FFFFH
		30 to 250			

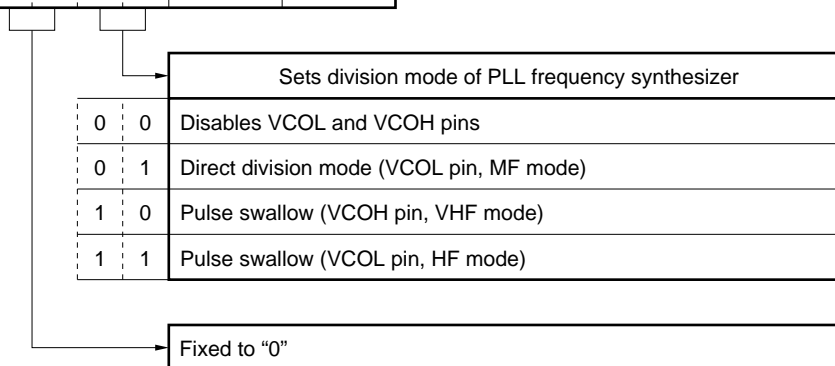
### 16.3.3 Configuration and function of PLL mode select register

The PLL mode select register specifies the division mode of the PLL frequency synthesizer and the pin to be used.

The configuration and function of the PLL mode select register are shown below.

The paragraphs (1) through (4) below outline the respective division modes.

Name	Flag symbol				Address	Read/ Write
	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>		
PLL mode select register	0	0	P L L M D 1	P L L M D 0	21H	R/W



After reset	Power-on	0	0	0	0
	Clock stop			0	0
	CE			Retained	

### (1) Direct division mode (MF)

In this mode, the VCOL pin is used.

The VCOH pin is pulled down.

In the direct division mode, the frequency of the input signal is divided only by the programmable counter.

## (2) Pulse swallow mode (HF)

The VOL pin is used in this mode.

The VCOH pin is pulled down.

In this mode, the frequency of the input signal is divided by the swallow counter and programmable counter.

### (3) Pulse swallow mode (VHF)

The VCOH pin is used in this mode.

The VCOL pin is pulled down.

In this mode, the frequency of the input signal is divided by the swallow counter and programmable counter.

#### (4) Disabling VCOL and VCOH pins

The VCOH and VCOL pins are internally pulled down.

However, the phase comparator, reference frequency generator, and charge pump operate.

Therefore, the operation is different from that in the PLL disable status to be explained later.

### 16.3.4 Programmable divider and PLL data register

The programmable divider divides the signal input from the VCOH or VCOL pin by the value set to the swallow counter and programmable counter.

The swallow counter and programmable counter are 4-bit binary down counters.

The division ratio is set to the swallow counter and programmable counter by the PLL data register (PLLR: peripheral address 41H) via data buffer.

Data is set to or read from the PLL data register by using the PUT PLLR, DBF or GET DBF, PLLR instruction.

The value to be divided is called N value.

For how to set the N value in each division mode, refer to **16.7**.

#### (1) PLL data register and data buffer

The relationship between the PLL data register and data buffer is explained next.

In the direct division mode, the higher 12 bits are valid, and all 16 bits are valid in the pulse swallow mode.

In the direct division mode, all the higher 12 bits are set to the programmable counter.

In the pulse swallow mode, the higher 12 bits are set to the programmable counter, and the lower 4 bits are set to the swallow counter.

#### (2) Relationship between division value N and divided output frequency

The relationship between the value “N” set to the PLL data register and the frequency “f<sub>N</sub>” of the signal divided and output by the programmable divider is as follows.

For details, refer to **16.7**.

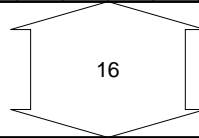
##### (a) In direct division mode (MF)

$$f_N = \frac{f_{IN}}{N} \quad N: 12 \text{ bits}$$

##### (b) In pulse swallow mode (HF and VHF)

$$f_N = \frac{f_{IN}}{N} \quad N: 16 \text{ bits}$$

Name	Data buffer															
Symbol	DBF3				DBF2				DBF1				DBF0			
Address	0CH				0DH				0EH				0FH			
Bit	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
Data																



GET can be executed

PUT can be executed

Peripheral register																		
Name	b <sub>15</sub>	b <sub>14</sub>	b <sub>13</sub>	b <sub>12</sub>	b <sub>11</sub>	b <sub>10</sub>	b <sub>9</sub>	b <sub>8</sub>	b <sub>7</sub>	b <sub>6</sub>	b <sub>5</sub>	b <sub>4</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>	Symbol	Peripheral address
PLL data register																		PLL R
																		41H
																		PLL frequency synthesizer



Sets division value of PLL frequency synthesizer

Direct division mode	0	don't care	Setting prohibited
	15 (00FH)	don't care	
	16 (010H)	don't care	Division value N: N = x
	x	don't care	
	2 <sup>12</sup> −1 (FFFH)	don't care	
Pulse swallow mode	0		Setting prohibited
	255 (00FFH)		
	256 (0100H)		Division value N: N = x
	x		
	2 <sup>16</sup> −1 (FFFFH)		

## 16.4 Reference Frequency Generator

### 16.4.1 Configuration and function of reference frequency generator

Figure 16-3 shows the configuration of the reference frequency generator.

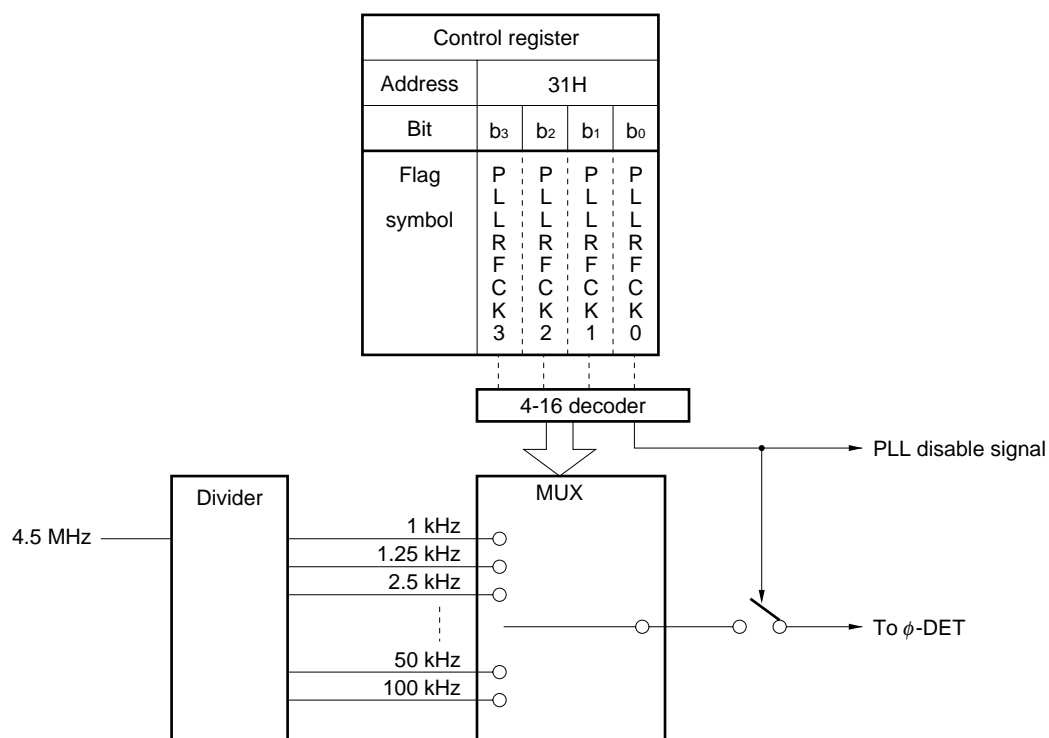
As shown in the figure, the reference frequency generator divides the crystal oscillator's 4.5 MHz to generate the reference frequency "f<sub>r</sub>" of the PLL frequency synthesizer.

Twelve reference frequencies can be selected: 1, 1.25, 2.5, 3, 5, 6.25, 9, 10, 12.5, 25, 50, and 100 kHz.

Reference frequency f<sub>r</sub> is selected by the PLL reference clock select register.

16.4.2 shows the configuration and function of the PLL reference clock select register.

**Figure 16-3. Configuration of Reference Frequency Generator (RFG)**





### 16.4.2 Configuration and function of PLL reference clock select register

The configuration and function of the PLL reference clock select register are shown below.

Name	Flag symbol				Address	Read/ Write
	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>		
PLL reference clock select register	P L L R F C K 3	P L L R F C K 2	P L L R F C K 1	P L L R F C K 0	31H	R/W

Sets reference frequency f <sub>r</sub> of PLL frequency synthesizer				
0	0	0	0	1.25 kHz
0	0	0	1	2.5 kHz
0	0	1	0	5 kHz
0	0	1	1	10 kHz
0	1	0	0	6.25 kHz
0	1	0	1	12.5 kHz
0	1	1	0	25 kHz
0	1	1	1	50 kHz
1	0	0	0	3 kHz
1	0	0	1	Setting prohibited
1	0	1	0	Setting prohibited
1	0	1	1	Setting prohibited
1	1	0	0	1 kHz
1	1	0	1	9 kHz
1	1	1	0	100 kHz
1	1	1	1	PLL disabled

After reset	Power-on	1	1	1	1
	Clock stop	1	1	1	1
	CE	Retained			

When the PLL is disabled by the PLL reference clock select register, the VCOH and VCOL pins are internally pulled down.

The EO pin is floated.

For disabling the PLL, refer to **16.6**.

## 16.5 Phase Comparator ( $\phi$ -DET), Charge Pump, and Unlock Detection Block

### 16.5.1 Configuration of phase comparator, charge pump, and unlock detection block

Figure 16-4 shows the configuration of the phase comparator, charge pump, and unlock detection block.

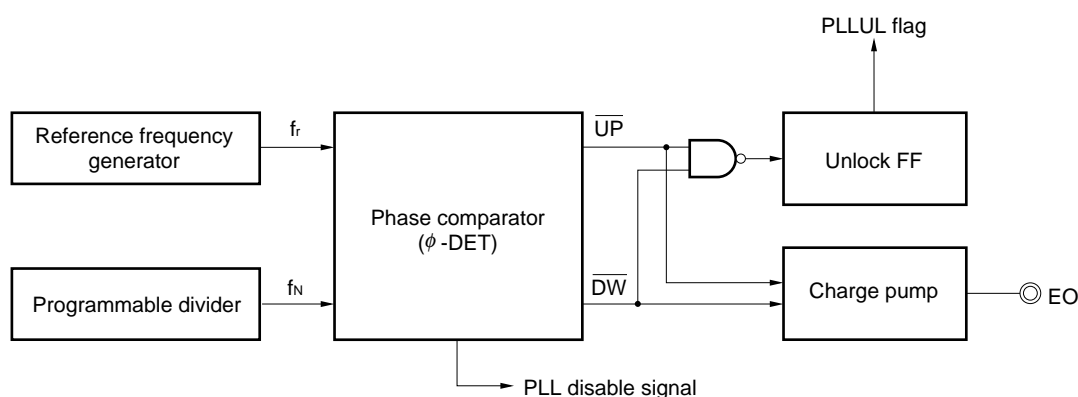
The phase comparator compares the divided frequency output “ $f_N$ ” of the programmable divider with the reference frequency output “ $f_r$ ” of the reference frequency generator, and outputs an up request ( $\overline{UP}$ ) or down request ( $\overline{DW}$ ) signal.

The charge pump outputs the output of the phase comparator from the error out (EO) pin.

The unlock detection block detects the unlock status of the PLL frequency synthesizer.

The following subsections 16.5.2 to 16.5.4 explain the operations of the phase comparator, charge pump, and unlock detection block respectively.

**Figure 16-4. Configurations of Phase Comparator, Charge Pump, and Unlock Detection Block**



### 16.5.2 Function of phase comparator

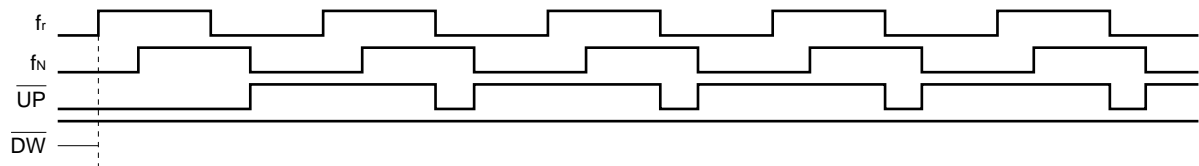
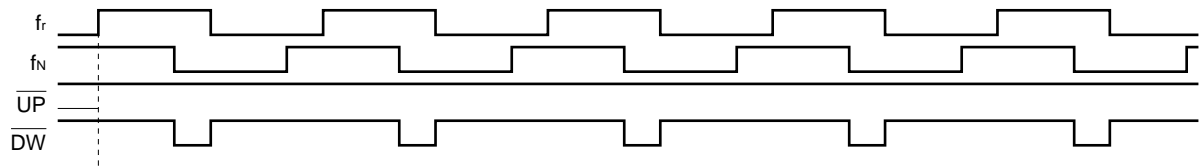
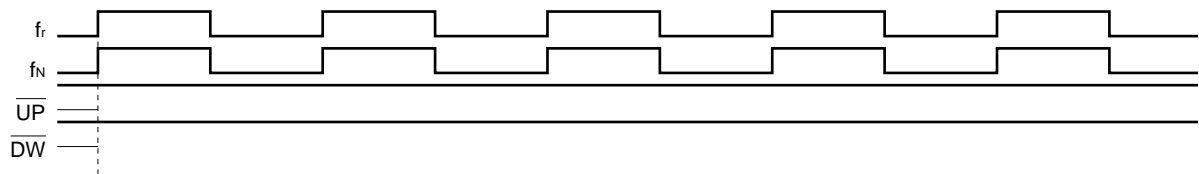
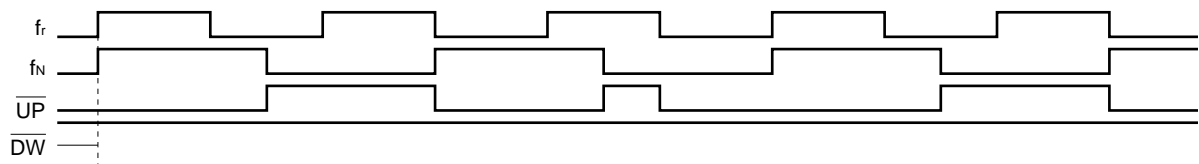
As shown in Figure 16-4, the phase comparator compares the divided frequency output “ $f_N$ ” of the programmable divider with the reference frequency output “ $f_r$ ” of the reference frequency generator, and outputs an up request or down request signal.

If the divided frequency  $f_N$  is lower than the reference frequency  $f_r$ , the phase comparator outputs the up request signal; if  $f_N$  is higher than  $f_r$ , it outputs the down request signal.

Figure 16-5 shows the relationship among the reference frequency  $f_r$ , divided frequency  $f_N$ , up request signal, and down request signal.

When the PLL is disabled, neither the up request nor down request signal is output.

The up request and down request signals are respectively input to the charge pump and unlock detection block.

Figure 16-5. Relationship Between  $f_r$ ,  $f_N$ ,  $\overline{UP}$ , and  $\overline{DW}$ (a) If  $f_N$  is behind  $f_r$  in phase(b) If  $f_N$  leads  $f_r$  in phase(c) If  $f_N$  and  $f_r$  are in phase(d) If  $f_N$  is lower than  $f_r$  in frequency

### 16.5.3 Charge pump

As shown in Figure 16-4, the charge pump outputs the up request signal or down request signal from the phase comparator to the error out (EO) pin.

Therefore, the relationship between the output of the error out pin, divided frequency  $f_N$ , and reference frequency  $f_r$  is as follows.

When reference frequency  $f_r >$  divided frequency  $f_N$ : Low-level output

When reference frequency  $f_r <$  divided frequency  $f_N$ : High-level output

When reference frequency  $f_r =$  divided frequency  $f_N$ : Floating

### 16.5.4 Unlock detection block

As shown in Figure 16-4, the unlock detection block detects the unlock status of the PLL frequency synthesizer by using the up request or down request signal from the phase comparator.

Because either of the up request or down request signal outputs a low level in the unlock status, this low-level signal is used to detect the unlock status.

In the unlock status, the unlock flip-flop (FF) is set to 1.

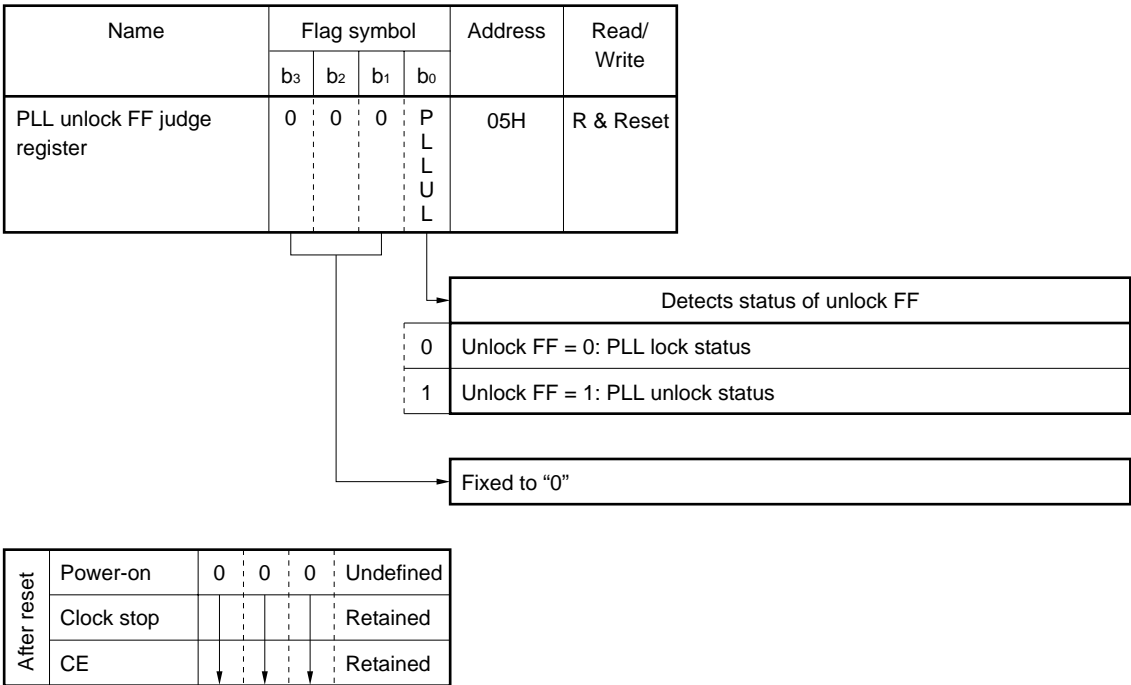
The unlock status is detected by the PLL unlock FF judgement register (refer to 16.5.5).

The unlock FF is set at the cycle of reference frequency  $f_r$  selected at that time.

When the contents of the PLL unlock FF judge register are read (by the PEEK instruction), the unlock FF is reset (Read & Reset).

Therefore, the unlock FF must be detected at a cycle longer than the cycle  $1/f_r$  of the reference frequency  $f_r$ .

### 16.5.5 Configuration and function of unlock FF judge register



This register is a read-only register and is reset when its contents are read to the window register by the PEEK instruction.

Because the unlock FF is set at the cycle of reference frequency  $f_r$ , the contents of the PLL unlock FF judge register must be written to the window register at a cycle longer than the cycle  $1/f_r$  of the reference frequency  $f_r$ .

The delay of the phase comparator up/down request signal is fixed to between 0.8 μs and 1.0 μs.

## 16.6 PLL Disabled Status

The PLL frequency synthesizer stops operation (is disabled) while the CE pin (pin 7) is low.

When the PLL is disabled by the PLL reference clock select register, the PLL frequency synthesizer also stops operation.

Table 16-2 shows the operation of each block under each PLL disable condition.

When the VCOL and VCOH pins are disabled by the PLL mode select register, only the VCOL and VCOH pins are internally pulled down, and the other blocks operate.

Because the PLL reference clock select register and PLL mode select register are not initialized (but hold the previous status) on CE reset, they are restored to the original status when the CE pin has once gone low and then back high again after the PLL has been disabled.

To disable the PLL on CE reset, therefore, initialize these registers in the program.

The PLL is disabled at power-on reset.

**Table 16-2. Operation of Blocks Under PLL Disable Conditions**

Condition Blocks	CE Pin = Low Level (PLL Disabled)	CE Pin = High Level	
		PLL Reference Clock Select Register = 1111B (PLL Disabled)	PLL Mode Select Register = 0000B (VCOH, VCOL Disabled)
VCOL and VCOH pins	Internally pulled down	Internally pulled down	Internally pulled down
Programmable counter	Stops division	Stops division	Operates
Reference frequency generator	Stops output	Stops output	Operates
Phase comparator	Stops output	Stops output	Operates
Charge pump	Floats error out pins	Floats error out pins	Operates However, usually outputs low level because there is no input.

## 16.7 Using PLL Frequency Synthesizer

To control the PLL frequency synthesizer, the following data is necessary.

- (1) Division mode: Direct division (MF), pulse swallow (HF, VHF)
- (2) Pin used: VCOL, VCOH
- (3) Reference frequency:  $f_r$
- (4) Division ratio: N

The following subsections 16.7.1 to 16.7.3 explain how to set the PLL data in each division mode (MF, HF, and VHF).

16.7.1 Direct division mode (MF)

(1) Selecting division mode

Select the direct division mode by using the PLL mode select register.

(2) Pin used

When the direct division mode is selected, the VCOL pin is enabled to operate.

(3) Setting reference frequency  $f_r$

Set the reference frequency by using the PLL reference clock select register.

(4) Calculating division value N

Calculate as follows:

$$N = \frac{f_{\text{VCOL}}}{f_r}$$

where,

$f_{\text{VCOL}}$ : Input frequency of VCOL pin

$f_r$ : Reference frequency

(5) Example of setting PLL data

How to set the data to receive broadcasting in the following MW band is explained below.

Reception frequency: 1,422 kHz (MW band)

Reference frequency: 9 kHz

Intermediate frequency: +450 kHz

Division value N:

$$N = \frac{f_{\text{VCOL}}}{f_r} = \frac{1,422 + 450}{9} = 208 \text{ (decimal)}$$
$$= 0D0H \text{ (hexadecimal)}$$

Set data to the PLL data register (PLLr: peripheral address 41H), PLL mode select register (RF address 21H), and PLL reference clock select register (RF address 31H) as follows.

PLL data register (RLLR)															
0	0	0	0	1	1	0	1	0	0	0	0	don't care			
0				D				0							

PLL mode select register	PLL reference clock select register
0 0 0 1	1 1 0 1
MF	9 kHz

### 16.7.2 Pulse swallow mode (HF)

#### (1) Selecting division mode

Select the pulse swallow mode by using the PLL mode select register.

#### (2) Pin used

When the pulse swallow mode is selected, the VCOL pin is enabled to operate.

#### (3) Setting reference frequency $f_r$

Set the reference frequency by using the PLL reference clock select register.

#### (4) Calculating division value N

Calculate as follows:

$$N = \frac{f_{\text{VCOL}}}{f_r}$$

where,

$f_{\text{VCOL}}$ : Input frequency of VCOL pin

$f_r$ : Reference frequency

#### (5) Example of setting PLL data

How to set the data to receive broadcasting in the following SW band is explained below.

Reception frequency: 25.50 MHz (SW band)

Reference frequency: 5 kHz

Intermediate frequency: +450 kHz

Division value N:

$$N = \frac{f_{\text{VCOL}}}{f_r} = \frac{25,500 + 450}{5} = 5190 \text{ (decimal)}$$

$$= 1446\text{H (hexadecimal)}$$

Set data to the PLL data register (PLLR: peripheral address 41H), PLL mode select register (RF address 21H), and PLL reference clock select register (RF address 31H) as follows.

PLL data register (RLLR)															
0	0	0	1	0	1	0	0	0	1	0	0	0	1	1	0
1				4				4				6			

PLL mode select register	PLL reference clock select register
0 0 1 1	0 0 1 0
HF	5 kHz

16.7.3 Pulse swallow mode (VHF)

(1) Selecting division mode

Select the pulse swallow mode by using the PLL mode select register.

(2) Pin used

When the pulse swallow mode is selected, the VCOH pin is enabled to operate.

(3) Setting reference frequency  $f_r$

Set the reference frequency by using the PLL reference clock select register.

(4) Calculating division value N

Calculate as follows:

$$N = \frac{f_{VCOH}}{f_r}$$

where,

$f_{VCOH}$ : Input frequency of VCOH pin

$f_r$ : Reference frequency

(5) Example of setting PLL data

How to set the data to receive broadcasting in the following FM band is explained below.

Reception frequency: 100.0 MHz (FM band)

Reference frequency: 25 kHz

Intermediate frequency: +10.7 MHz

Division value N:

$$N = \frac{f_{VCOH}}{f_r} = \frac{100.0 + 10.7}{0.025} = 4428 \text{ (decimal)}$$
$$= 114CH \text{ (hexadecimal)}$$

Set data to the PLL data register (PLLR: peripheral address 41H), PLL mode select register (RF address 21H), and PLL reference clock select register (RF address 31H) as follows.

PLL data register (RLLR)															
0	0	0	1	0	0	0	1	0	1	0	0	1	1	1	0
1				1				4				C			

PLL mode select register	PLL reference clock select register
0 0 1 0	0 1 1 0
VHF	25 kHz



## **16.8 Status on Reset**

### **16.8.1 On power-on reset**

The PLL is disabled on power-on reset because the PLL reference clock select register is initialized to 1111B.

### **16.8.2 On execution of clock stop instruction**

The PLL is disabled when the CE pin goes low.

### **16.8.3 On CE reset**

#### **(1) CE reset after execution of clock stop instruction**

The PLL is disabled because the PLL reference clock select register is initialized to 1111B by the clock stop instruction.

#### **(2) CE reset without clock stop instruction executed**

Because the PLL reference clock select register retains the previous status, the previous status is restored as soon as the CE pin has gone high.

### **16.8.4 In halt status**

The set status is retained if the CE pin is high.

## 17. FREQUENCY COUNTER

### 17.1 Outline of Frequency Counter

Figure 17-1 illustrates the frequency counter.

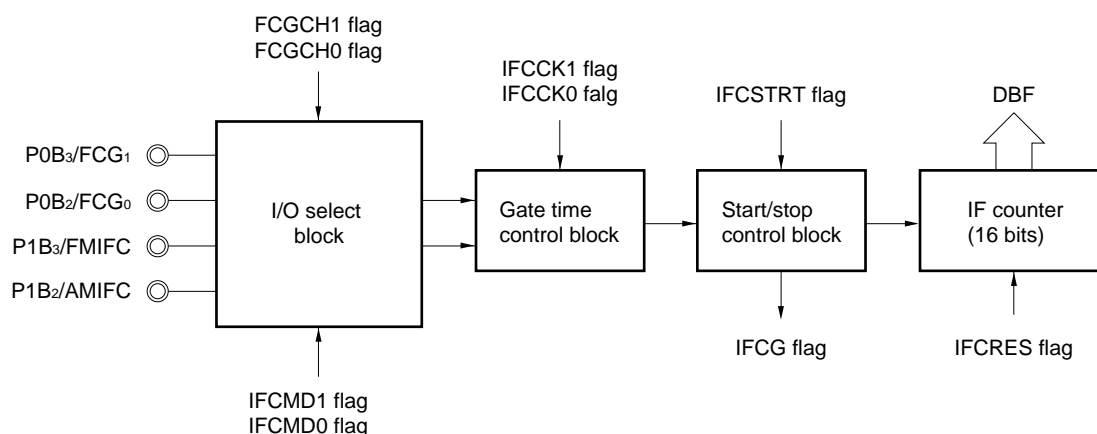
The frequency counter has an IF counter function to count the intermediate frequency (IF) of an external input signal and an external gate counter (FCG: Frequency Counter for external Gate signal) to detect the pulse width of an external input signal.

The IF counter function counts the frequency input to the P1B<sub>3</sub>/FMIFC or P1B<sub>2</sub>/AMIFC pin at fixed intervals (1 ms, 4 ms, 8 ms, or open) by using a 16-bit counter.

The external gate counter function counts the frequency of the internal clock (1 kHz, 100 kHz, 900 kHz) from the rising to the falling of the signal input to the P0B<sub>3</sub>/FCG<sub>1</sub> or P0B<sub>2</sub>/FCG<sub>0</sub> pin.

The IF counter and external gate counter functions cannot be used at the same time.

**Figure 17-1. Outline of Frequency Counter**



- Remarks 1.** FCGCH1 and FCGCH0 (bits 1 and 0 of the FCG channel select register; refer to **Figure 17-4**) select the pin used for the external gate counter function.
- 2.** IFCMD1 and IFCMD0 (bits 3 and 2 of the IF counter mode select register; refer to **Figure 17-3**) select the IF counter or external gate counter function.
- 3.** IFCKK1 and IFCKK0 (bits 1 and 0 of the IF counter mode select register; refer to **Figure 17-3**) select the gate time of the IF counter function and the reference frequency of the external gate counter function.
- 4.** IFCSTRT (bit 1 of the IF counter control register; refer to **Figure 17-6**) control starting of the IF counter and external gate counter functions.
- 5.** IFCG (bit 0 of the IF counter gate judge register; refer to **Figure 17-7**) detects opening/closing the gate of the IF counter function.
- 6.** IFCRES (bit 0 of the IF counter control register; refer to **Figure 17-6**) reset the count value of the IF counter.

## 17.2 Input/Output Select Block and Gate Time Control Block

Figure 17-2 shows the configuration of the input/output select block and gate time control block.

The input/output select block consists of an IF counter input select block and FCG I/O select block.

The IF counter input select block selects whether the frequency counter is used as an IF counter or an external gate counter, by using the IF counter mode select register. When the frequency counter is used as the IF counter, either P1B<sub>3</sub>/FMIFC or P1B<sub>2</sub>/AMIFC pin and a count mode are selected. The pin not used for the IF counter is used as a general-purpose input port pin.

The FCG I/O select block selects a pin to be used from either the P0B<sub>3</sub>/FCG<sub>1</sub> or P0B<sub>2</sub>/FCG<sub>0</sub> pin by using the FCG channel select register, when the frequency counter is used as the external gate counter. The pin not used is used as a general-purpose I/O port pin.

When using the frequency counter as the external gate counter, the pin to be used must be set in the input mode by using the port 0B bit I/O select register. This is because the pin is set in the general-purpose output port mode if it is set in the output mode even if the external gate counter function is selected by the IF counter mode select register and FCG channel select register.

The gate time control block selects gate time by using the IF counter mode select register when the frequency counter is used as the IF counter, or a count frequency when the frequency counter is used as the external gate counter.

Figure 17-3 shows the configuration of the IF counter mode select register.

Figure 17-4 shows the configuration of the FCG channel select register.

**Figure 17-2. Configuration of I/O Select Block and Gate Time Control Block**

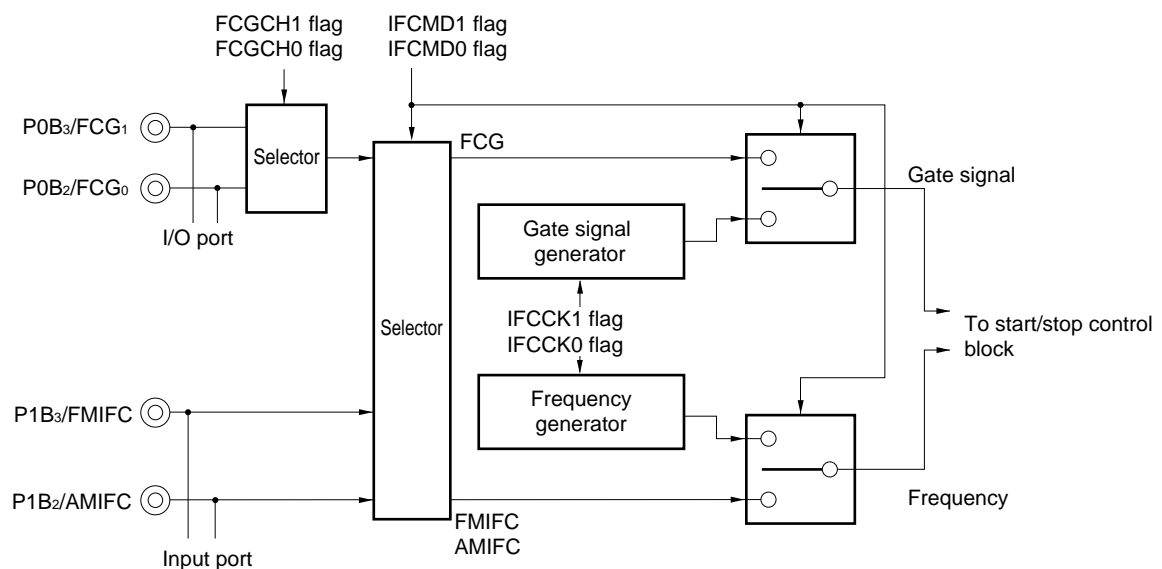
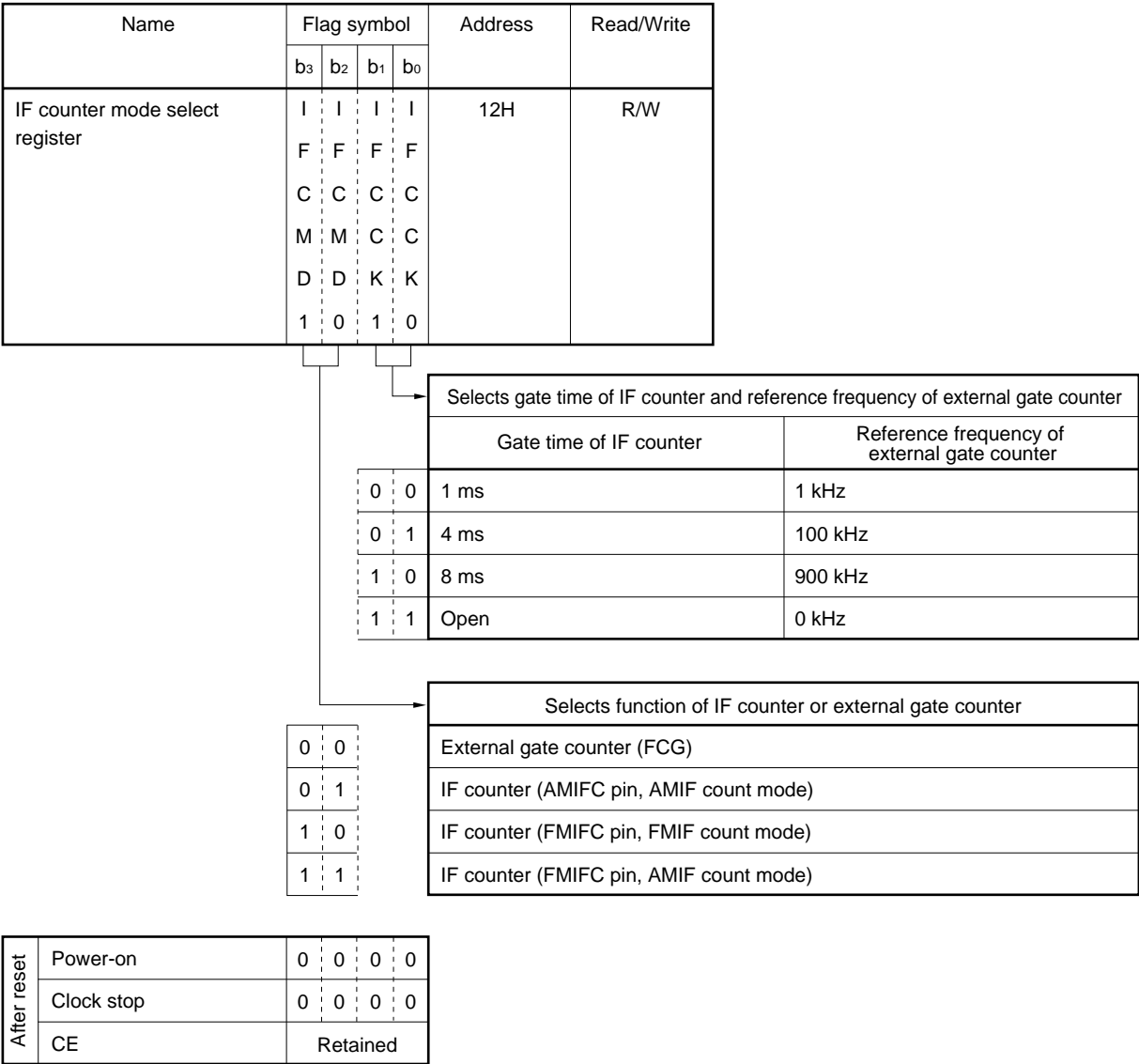
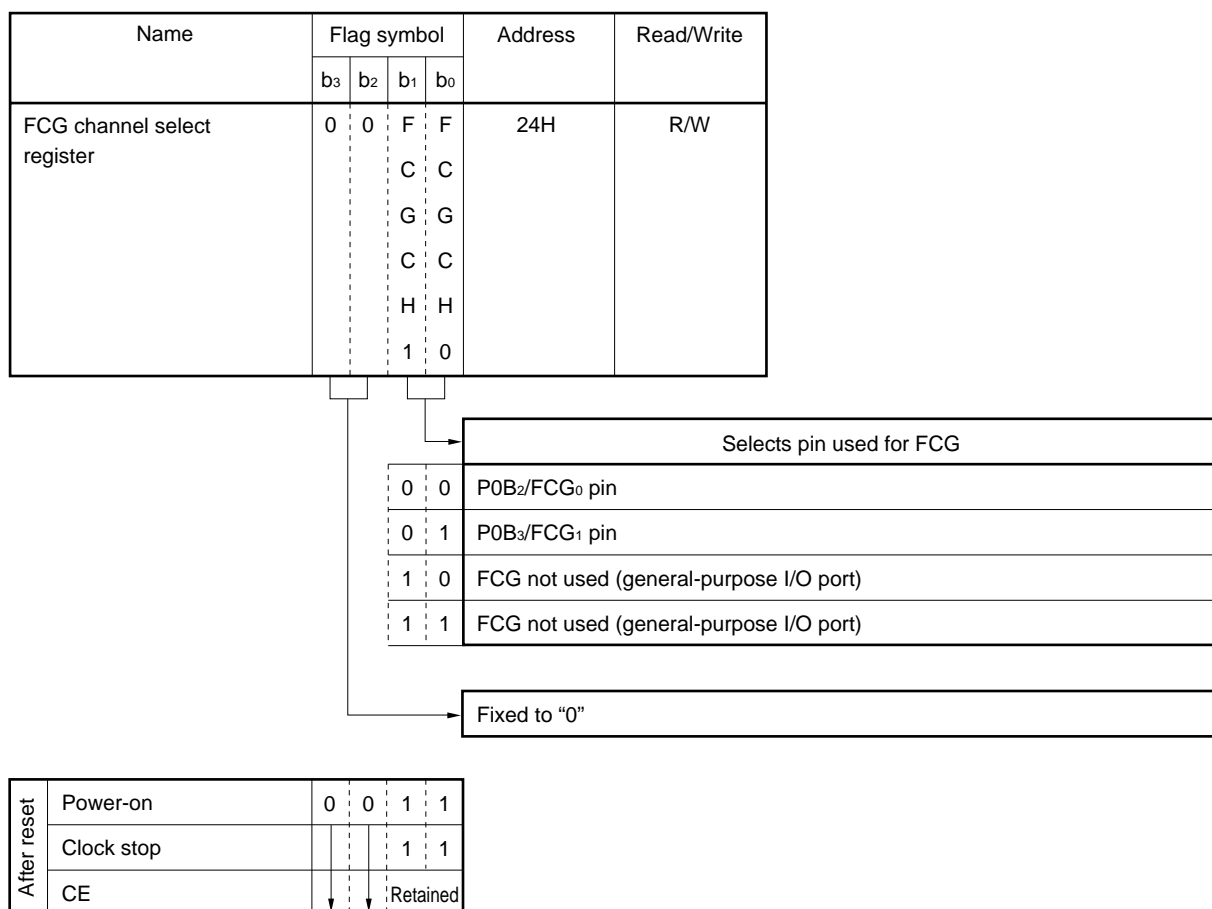


Figure 17-3. Configuration of IF Counter Mode Select Register



**Caution** The IF counter and external gate counter functions cannot be used at the same time.

Figure 17-4. Configuration of FCG Channel Select Register



## 17.3 Start/Stop Control Block and IF Counter

### 17.3.1 Configuration of start/stop control block and IF counter

Figure 17-5 shows the configuration of the start/stop control block and IF counter.

The start/stop control block starts the frequency counter or detects the end of counting.

The counter is started by the IF counter control register.

The end of counting is detected by the IF counter gate judge register. When the external gate counter function is used, however, the end of counting cannot be detected by the IF counter gate judge register.

Figure 17-6 shows the configuration of the IF counter control register.

Figure 17-7 shows the configuration of the IF counter gate judge register.

17.3.2 and 17.3.3 describe the gate operation when the IF counter function is selected and that when the external gate counter function is selected.

The IF counter is a 16-bit binary counter that counts up the input frequency when the IF counter function or external gate counter function is selected.

When the IF counter function is selected, the frequency input to a selected pin is counted while the gate is opened by an internal gate signal. The frequency count is counted without alteration in the AMIF count mode. In the FMIF counter mode, however, the frequency input to the pin is halved and counted.

When the external gate counter function is selected, the internal frequency is counted while the gate is opened by the signal input to the pin.

When the IF counter counts up to FFFFH, the following input becomes 0000H, and then counting continues.

The count value is read by the IF counter data register (IFC) via data buffer.

The count value is reset by the IF counter control register.

Figure 17-8 shows the configuration of the IF counter data register.

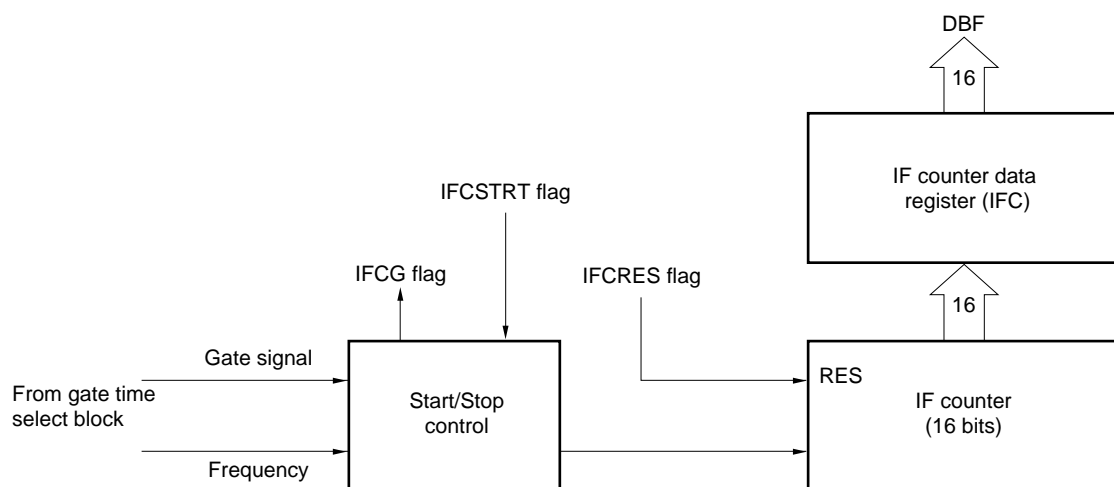
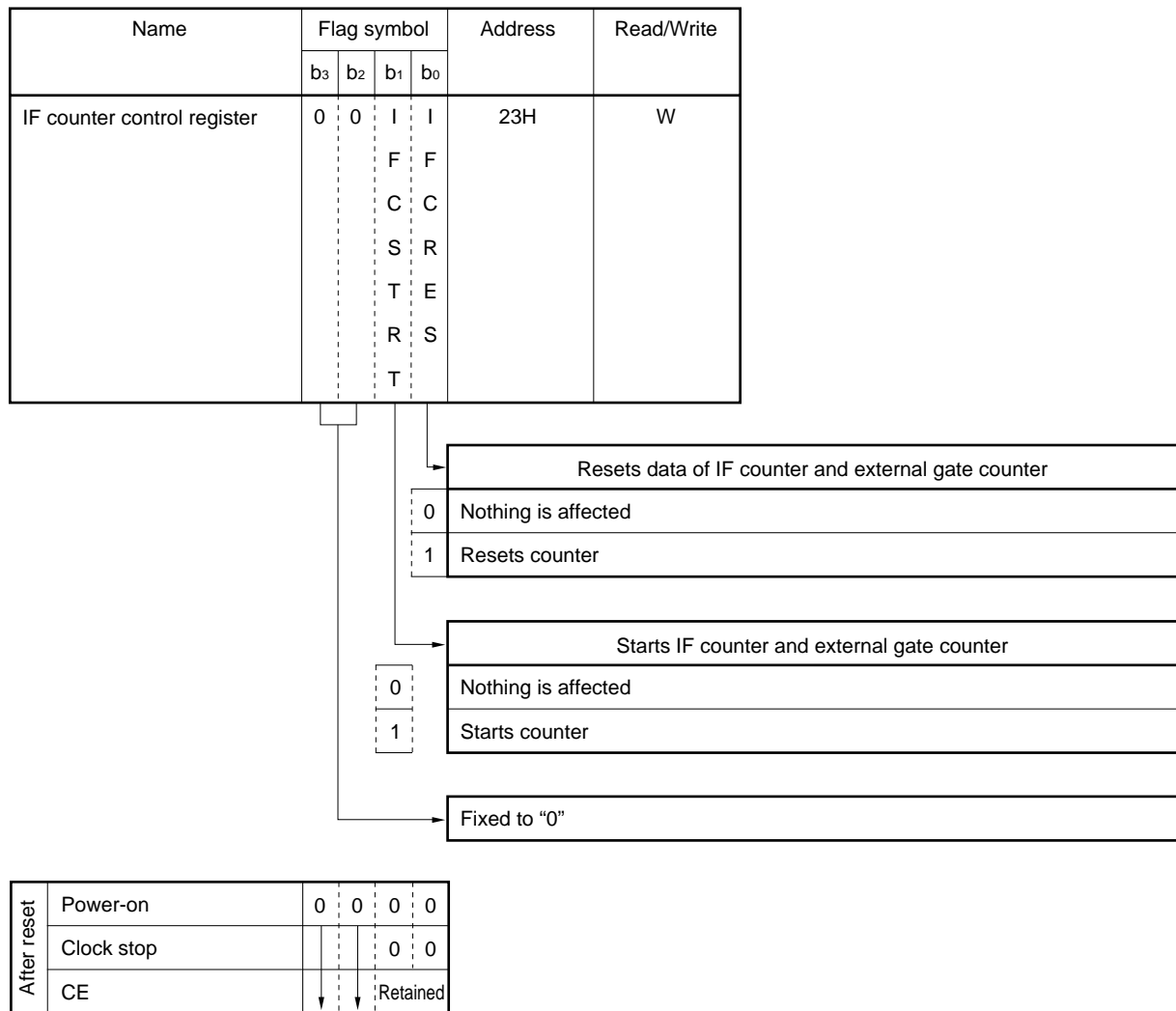


Figure 17-5. Configuration of Start/Stop Control Block and IF Counter

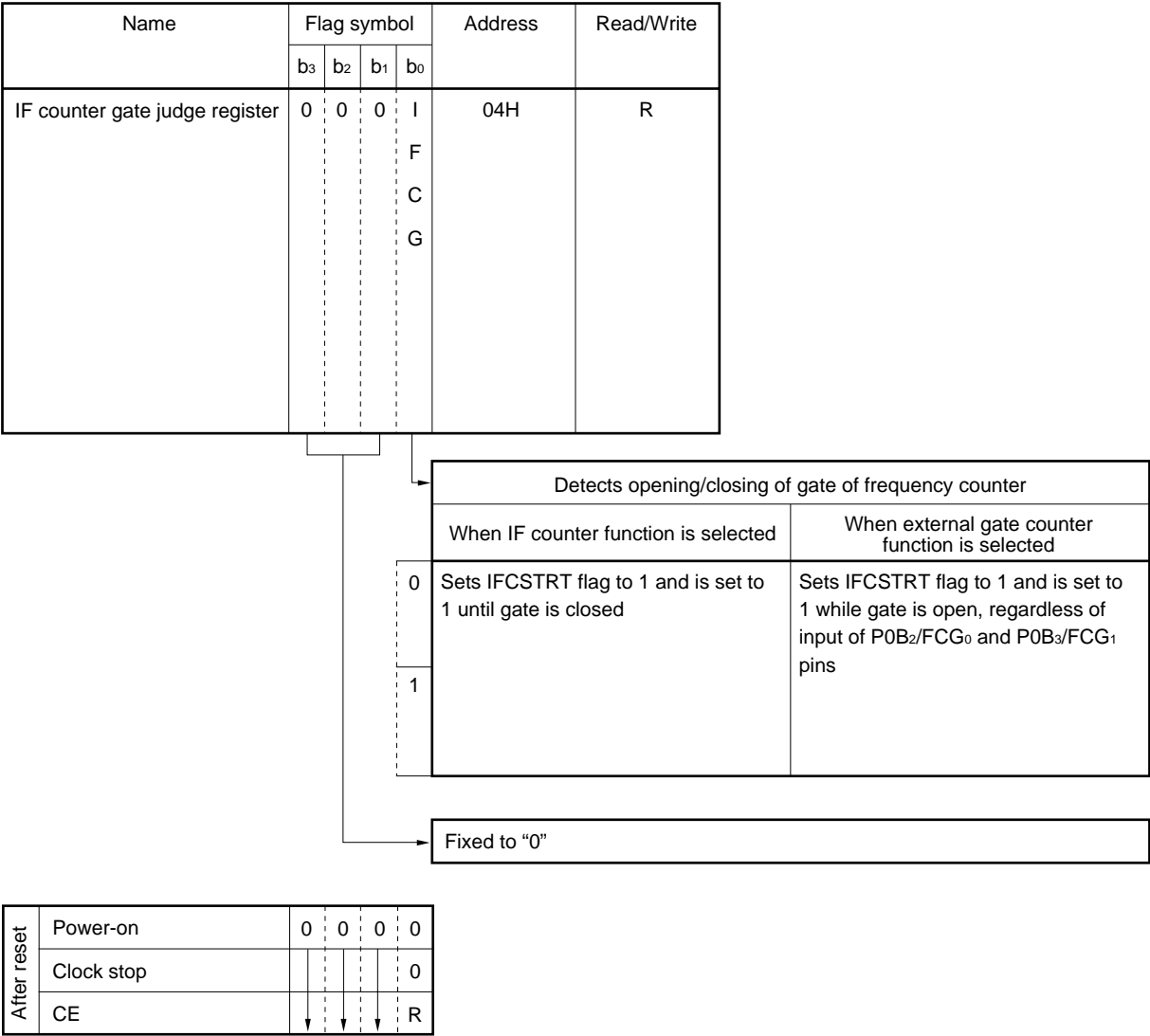
Figure 17-6. Configuration of IF Counter Control Register



The IF counter control register is controlled by writing the contents of the window register to it using the POKE instruction.

When the contents of the IF counter are read by the PEEK instruction, 0 is read in the window register.

Figure 17-7. Configuration of IF Counter Gate Judge Register



- Cautions**
- 1. Do not read the contents of the IF counter data register (IFC) to the data buffer while the IFCG flag is set to 1.
  - 2. The gate of the external gate counter cannot be opened or closed by the IFCG flag. Use the IFCSTRT flag to open or close the gate.

**Remark** R: Retained



### 17.3.2 Operation of gate when IF counter function is selected

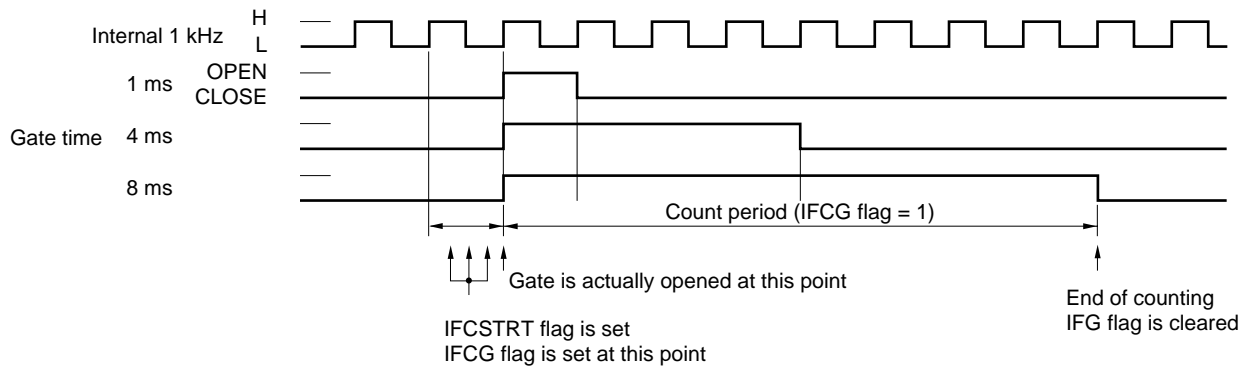
#### (1) When gate time of 1, 4, or 8 ms is selected

The gate is opened for 1, 4, or 8 ms from the rising of the internal 1 kHz signal after the IFCSTRT flag has been set to 1, as illustrated below.

While this gate is open, the frequency input from a selected pin is counted by a 16-bit counter.

When the gate is closed, the IFCG flag is cleared to 0.

The IFCG flag is automatically set to 1 when the IFCSTRT flag is set.



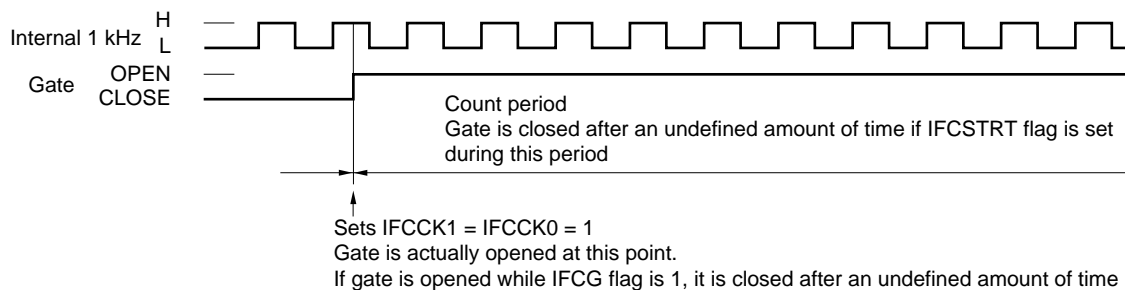
#### (2) When gate is open

If opening of the gate is selected by the IFCKK1 and IFCKK0 flags, the gate is opened as soon as its opening has been selected, as illustrated below.

If the counter is started by using the IFCSTRT flag while the gate is open, the gate is closed after an undefined amount of time.

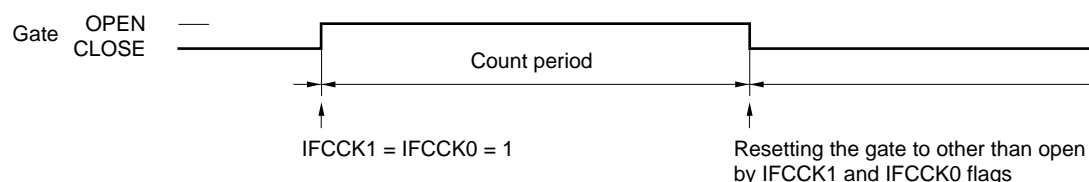
To open the gate, therefore, do not set the IFCSTRT flag to 1.

However, the counter can be reset by the IFCRES flag.



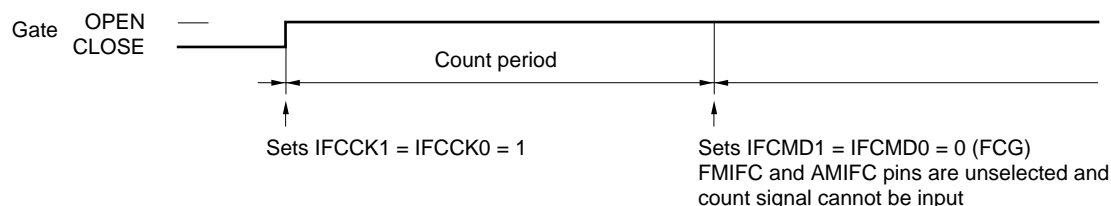
The gate is opened or closed in the following two ways when opening the gate is selected as the gate time.

(a) Resetting the gate to other than open by using IFCKK1 and IFCKK0 flags



(b) Unselect pin used by using IFCMD1 and IFCMD0 flags

In this way, the gate remains open, and counting is stopped by disabling input from the pin.



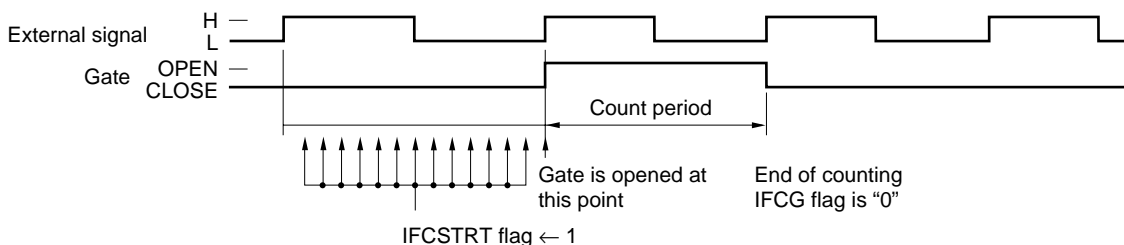
### 17.3.3 Gate operation when external gate counter function is selected

The gate is opened from the rising to the next rising of the signal input to a selected pin after the IFCSTRT flag has been set to 1, as illustrated below.

While the gate is open, the internal frequency (1 kHz, 100 kHz, 900 kHz) is counted by a 16-bit counter.

The IFCG flag is set to 1 from the rising to the next rising of the external signal after the IFCSTRT flag has been set.

In other words, the opening or closing of the gate cannot be detected by the IFCG flag when the external gate counter function is selected.



If reset and started while gate is open

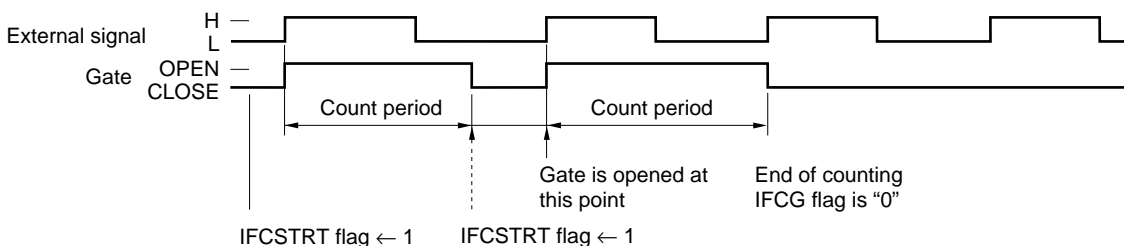
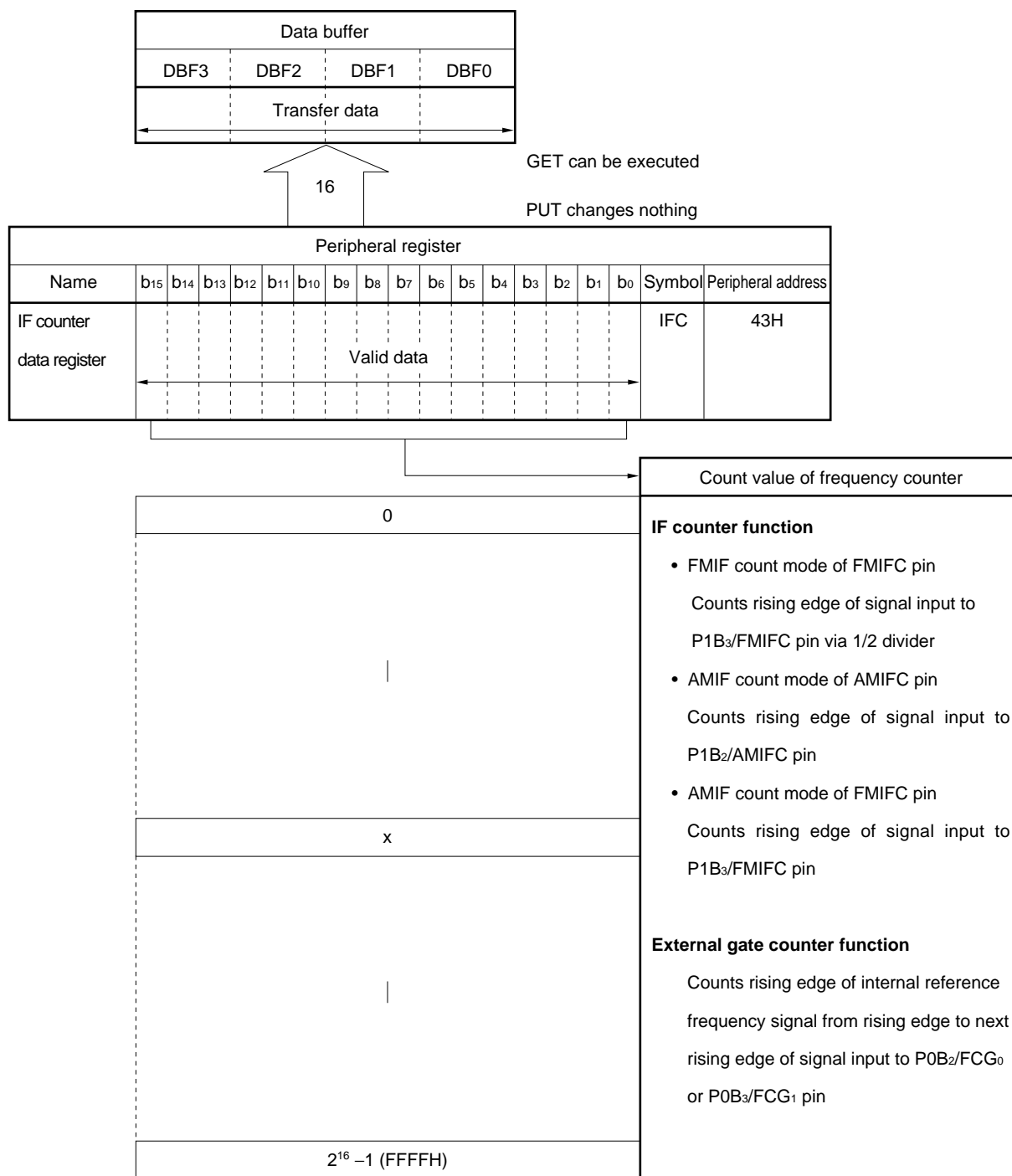


Figure 17-8. Configuration of IF Counter Data Register



The IF counter is cleared to 0000H when its count value has reached FFFFH, and then continues counting.

## 17.4 Using IF Counter Function

The following subsections 17.4.1 through 17.4.3 explain how to use the hardware of the IF counter function, program example, and count error.

### 17.4.1 Using hardware of IF counter

Figure 17-9 shows the block diagram illustrating how the P1B<sub>3</sub>/FMIFC and P1B<sub>2</sub>/AMIFC pins are used.

Table 17-1 shows the range of the frequencies that can be input to the P1B<sub>3</sub>/FMIFC and P1B<sub>2</sub>/AMIFC pins.

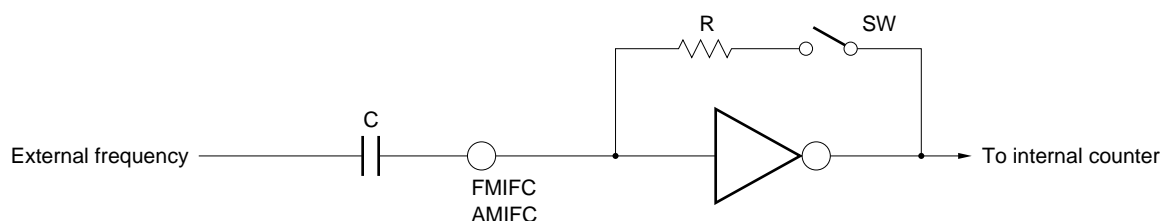
Because the input pins of the IF counter have an internal amplifier, cut off the DC component of the input signal by using capacitor C as shown in Figure 17-9.

When the P1B<sub>3</sub>/FMIFC or P1B<sub>2</sub>/AMIFC pin is selected as the IF counter pin, switch SW turns ON, applying a voltage of about  $1/2V_{DD}$  to each pin.

If the voltage has not risen to a sufficient intermediate level at this time, the AC amplifier does not operate normally, and consequently, the IF counter does not correctly operate.

Therefore, make sure that a sufficiently long wait time elapses from the time each pin is selected as an IF counter until counting is started.

**Figure 17-9. Function of Each IF Counter Pin**



**Table 17-1. Input Frequency Range of IF Counter**

Input Pin	Input Frequency (MHz)	Input Amplitude ( $V_{p-p}$ )
P1B <sub>3</sub> /FMIFC FMIF mode	5 to 15	0.3
	10.5 to 10.9	0.1
P1B <sub>3</sub> /FMIFC AMIF mode	0.3 to 1	0.3
P1B <sub>2</sub> /AMIFC AMIF mode	0.3 to 1	0.3
	0.44 to 0.46	0.1

### 17.4.2 Program example of IF counter function

A program example of the IF counter function is shown below.

As shown in this example, a wait time must elapse after an instruction that selects the P1B<sub>3</sub>/FMIFC or P1B<sub>2</sub>/AMIFC pin as the IF counter pin has been executed before counting is started.

This is because the internal AC amplifier may not operate normally immediately after each pin has been selected, as explained in 17.4.1.

#### Example To count frequency on P1B<sub>3</sub>/FMIFC pin (gate time: 8 ms)

```

INITFLG  IFCMD1, NOT IFCMD0, IFCK1, NOT IFCK0
                                         ; Selects FMIFC pin and sets gate time to 8 ms.
                                         ; Internal AC amplifier stabilization time



Wait



LOOP:
    SKT1   IFCG           ; Detects opening/closing of gate.
    BR     READ          ; Branches to READ: when gate is closed.



Processing A


                                         ; Do not read data of IF counter by this processing A.

    BR     LOOP

READ:
    GET    DBF, IFC       ; Reads value of IF counter data register to data buffer.

```

### 17.4.3 Error of IF counter

The IF counter may have a gate time error and a count error.

These errors are explained in (1) and (2) below.

#### (1) Error of gate time

The gate time of the IF counter is created by dividing the 4.5 MHz system clock.

Therefore, if the system clock deviates "+x" ppm, the gate time deviates "-x" ppm.

#### (2) Count error

The IF counter counts the frequency at the rising edge of an input signal.

If a high level is input to the pin when the gate is opened, therefore, one excess pulse is counted.

However, counting is not performed because of the status of the pin when the gate is closed.

Therefore, a count error of "+1, -0" may occur.

## 17.5 Error of External Gate Counter

The external gate counter has an internal frequency error and count error, as described in (1) and (2) below.

### (1) Internal frequency error

The internal frequency of the external gate counter is created by dividing the 4.5 MHz system clock frequency. Therefore, if this frequency has an error of “+x” ppm, the internal frequency accordingly has an error of “+x” ppm.

### (2) Count error

The external gate counter counts the frequency at the rising edge of the internal frequency.

Therefore, if the internal frequency is at low level when the gate is opened (when the input signal of the pin rises), one extra pulse is counted.

However, this extra pulse may not be counted, depending on the count level of the internal frequency, when the gate is closed (when the input signal of the pin rises next time).

Therefore, the count error is “+1, -0”.

## 17.6 Status on Reset

### 17.6.1 On power-on reset

The P1B<sub>3</sub>/FMIFC and P1B<sub>2</sub>/AMIFC pins are set in the general-purpose input port mode.

The P0B<sub>3</sub>/FCG<sub>1</sub> and P0B<sub>2</sub>/FCG<sub>0</sub> pins are set in the general-purpose I/O port mode.

### 17.6.2 On execution of clock stop instruction

The P1B<sub>3</sub>/FMIFC and P1B<sub>2</sub>/AMIFC pins are set in the general-purpose input port mode.

The P0B<sub>3</sub>/FCG<sub>1</sub> and P0B<sub>2</sub>/FCG<sub>0</sub> pins are set in the general-purpose I/O port mode.

### 17.6.3 On CE reset

The P1B<sub>3</sub>/FMIFC, P1B<sub>2</sub>/AMIFC, P0B<sub>3</sub>/FCG<sub>1</sub>, and P0B<sub>2</sub>/FCG<sub>0</sub> pins retain the previous status.

### 17.6.4 In halt status

The P1B<sub>3</sub>/FMIFC, P1B<sub>2</sub>/AMIFC, P0B<sub>3</sub>/FCG<sub>1</sub>, and P0B<sub>2</sub>/FCG<sub>0</sub> pins retain the status immediately before the halt status was set.

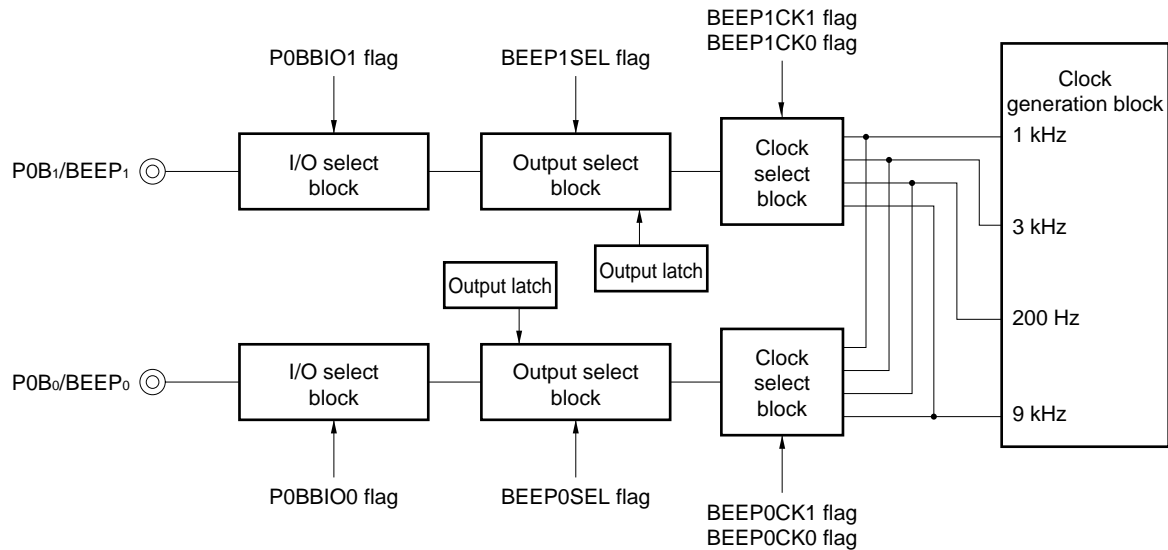
## 18. BEEP

### 18.1 General

Figure 18-1 shows the outline of BEEP.

BEEP outputs 1 kHz, 3 kHz, 200 Hz, or 9 kHz clock from the P0B<sub>0</sub>/BEEP<sub>0</sub> and P0B<sub>1</sub>/BEEP<sub>1</sub> pins.

Figure 18-1. Outline of BEEP



- Remarks**
1. P0BBIO1 and P0BBIO0 (bits 1 and 0 of the port 0B bit I/O select register; refer to **Figure 18-2**) set the P0B<sub>1</sub>/BEEP<sub>1</sub> and P0B<sub>0</sub>/BEEP<sub>0</sub> pins in the input/output mode.
  2. BEEP1SEL and BEEP0SEL (bits 1 and 0 of the BEEP select register; refer to **Figure 18-3**) set the P0B<sub>1</sub>/BEEP<sub>1</sub> and P0B<sub>0</sub>/BEEP<sub>0</sub> pin in the general-purpose output port or BEEP output mode.
  3. BEEP1CK1, BEEP1CK0, BEEP0CK1, and BEEP0CK0 (bits 3 to 0 of the BEEP clock select register; refer to **Figure 18-4**) set the output frequencies of BEEP<sub>1</sub> and BEEP<sub>0</sub>.

# 18.2 I/O Select Block and Output Select Block

The I/O select block sets the P0B<sub>1</sub>/BEEP<sub>1</sub> and P0B<sub>0</sub>/BEEP<sub>0</sub> pins in the input or output mode by using the port 0B bit I/O select register. These pins must be set in the output mode when they are used as the BEEP pins.

The output select block sets the P0B<sub>1</sub>/BEEP<sub>1</sub> and P0B<sub>0</sub>/BEEP<sub>0</sub> pins in the general-purpose output port or BEEP output mode by using the BEEP select register.

Figure 18-2 shows the configuration and function of the port 0B bit I/O select register.

Figure 18-3 shows the configuration and function of the BEEP select register.

**Figure 18-2. Configuration of Port 0B Bit I/O Select Register**

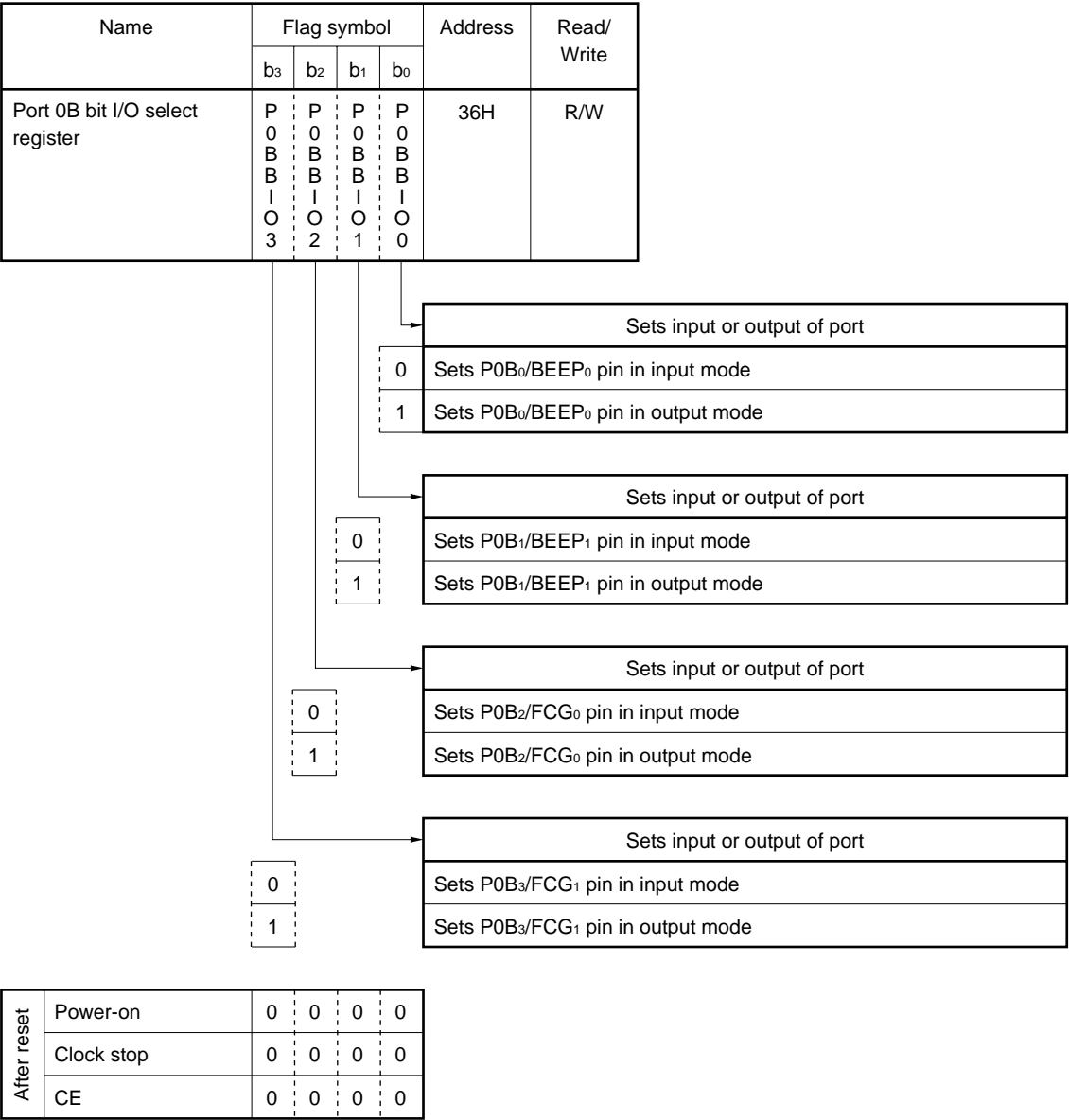
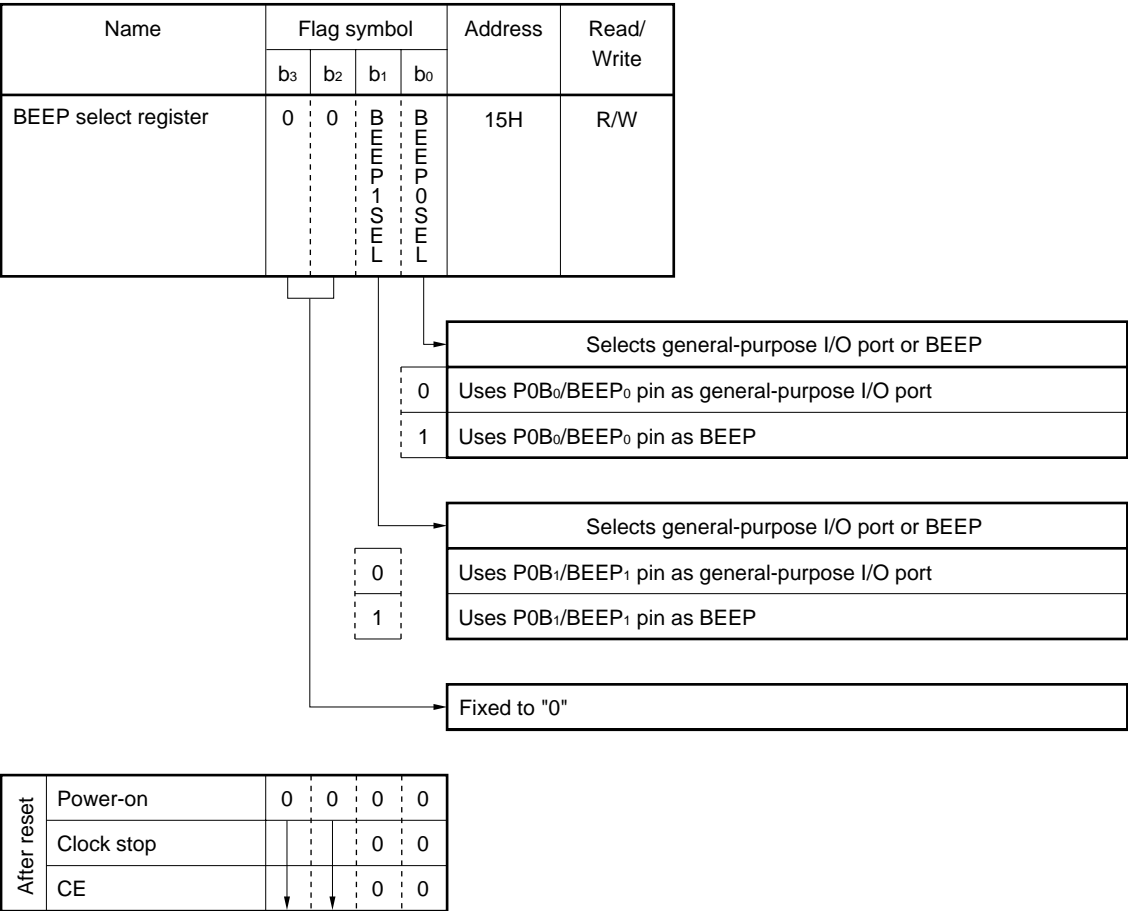




Figure 18-3. Configuration of BEEP Select Register



18.3 Clock Select Block and Clock Generator Block

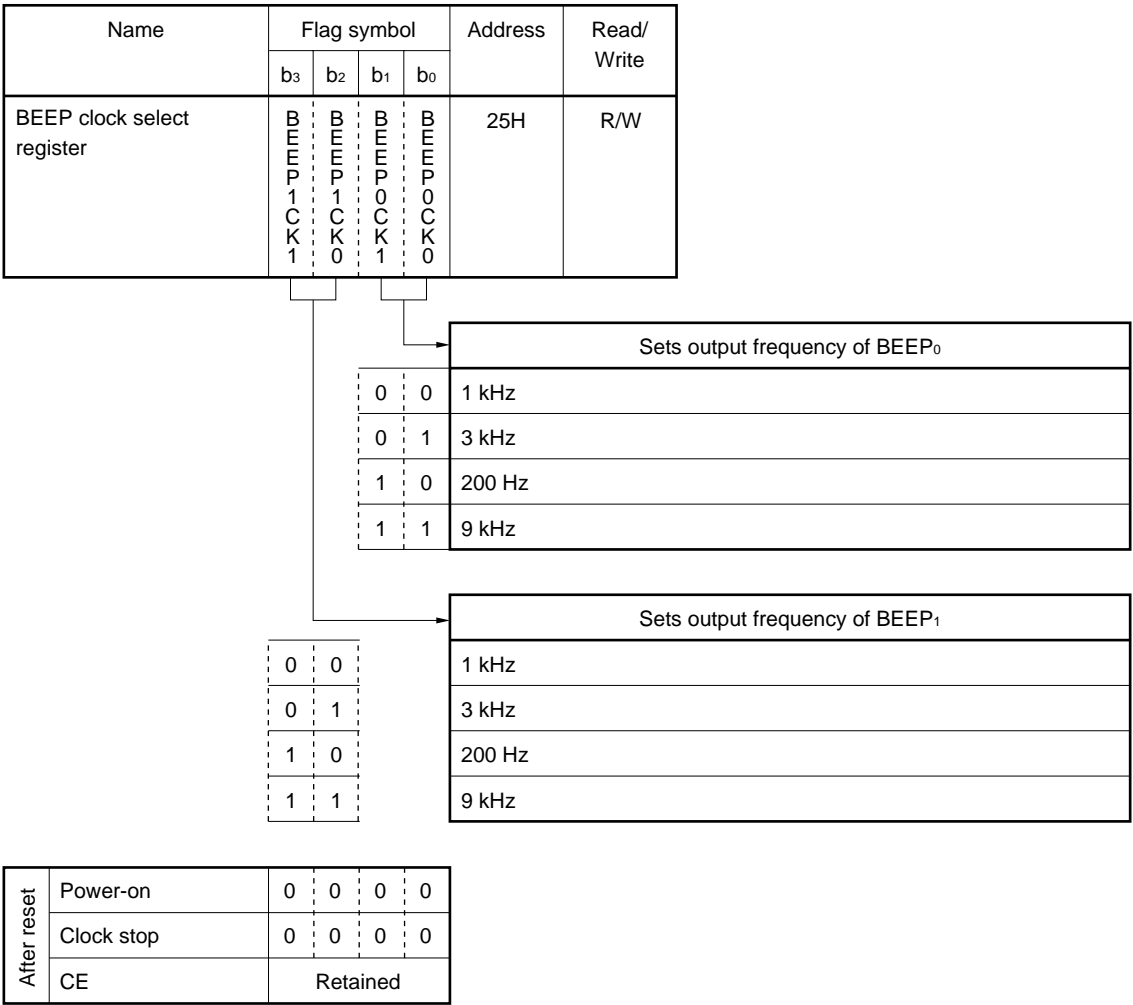
The clock select block selects the output frequencies of the BEEP<sub>1</sub> and BEEP<sub>0</sub> pins by using the BEEP clock select register.

The clock generator block generates the clock to be output to the BEEP<sub>1</sub> and BEEP<sub>0</sub> pins.

The clock frequency to be generated is 1 kHz, 3 kHz, 200 Hz, or 9 kHz.

Figure 18-4 shows the configuration and function of the BEEP clock select register.

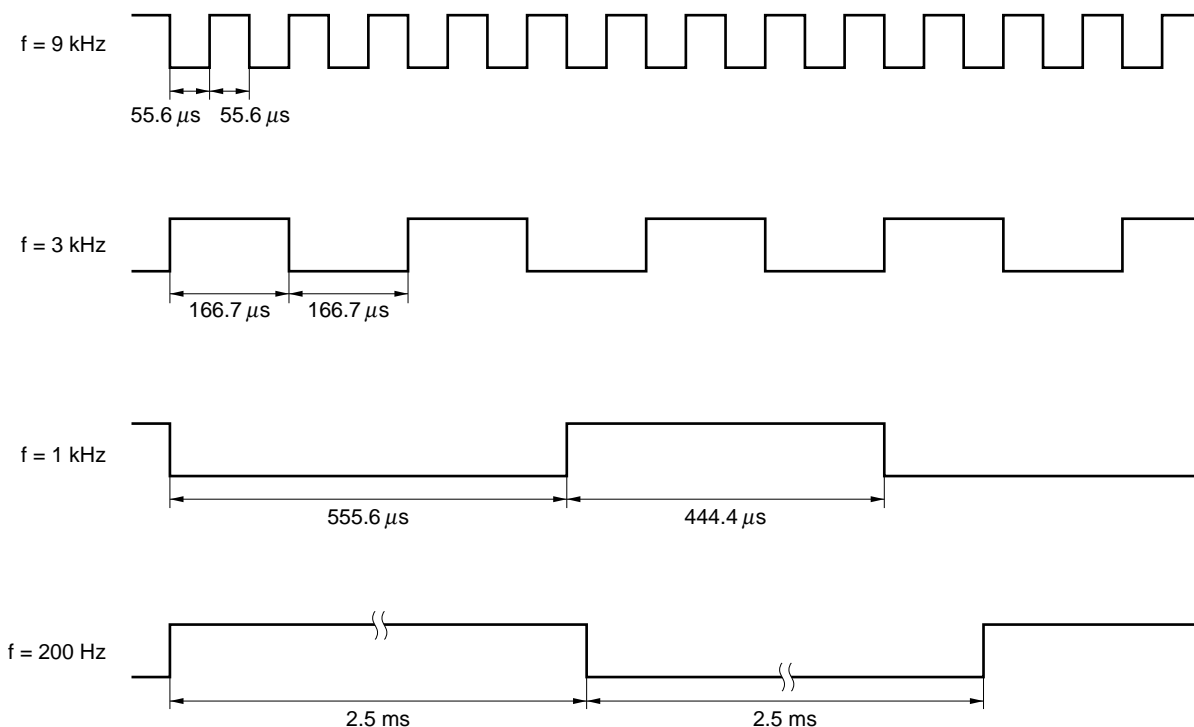
Figure 18-4. Configuration of BEEP Clock Select Register



## 18.4 Output Waveform of BEEP

The duty cycle of the output waveform of BEEP is 50% except when  $f = 1$  kHz.

The output waveform is shown below.



**Remark** f: Output frequency of BEEP

## 18.5 Status on Reset

### 18.5.1 At power-on reset

The P0B<sub>0</sub>/BEEP<sub>0</sub> and P0B<sub>1</sub>/BEEP<sub>1</sub> pins are set in the general-purpose input port mode.

### 18.5.2 At clock stop

The P0B<sub>0</sub>/BEEP<sub>0</sub> and P0B<sub>1</sub>/BEEP<sub>1</sub> pins are set in the general-purpose input port mode.

### 18.5.3 At CE reset

The P0B<sub>0</sub>/BEEP<sub>0</sub> and P0B<sub>1</sub>/BEEP<sub>1</sub> pins are set in the general-purpose input port mode.

## ★ 19. LCD CONTROLLER/DRIVER

The LCD (Liquid Crystal Display) controller/driver can display an LCD of up to 60 dots by outputting segment and common signals in combination.

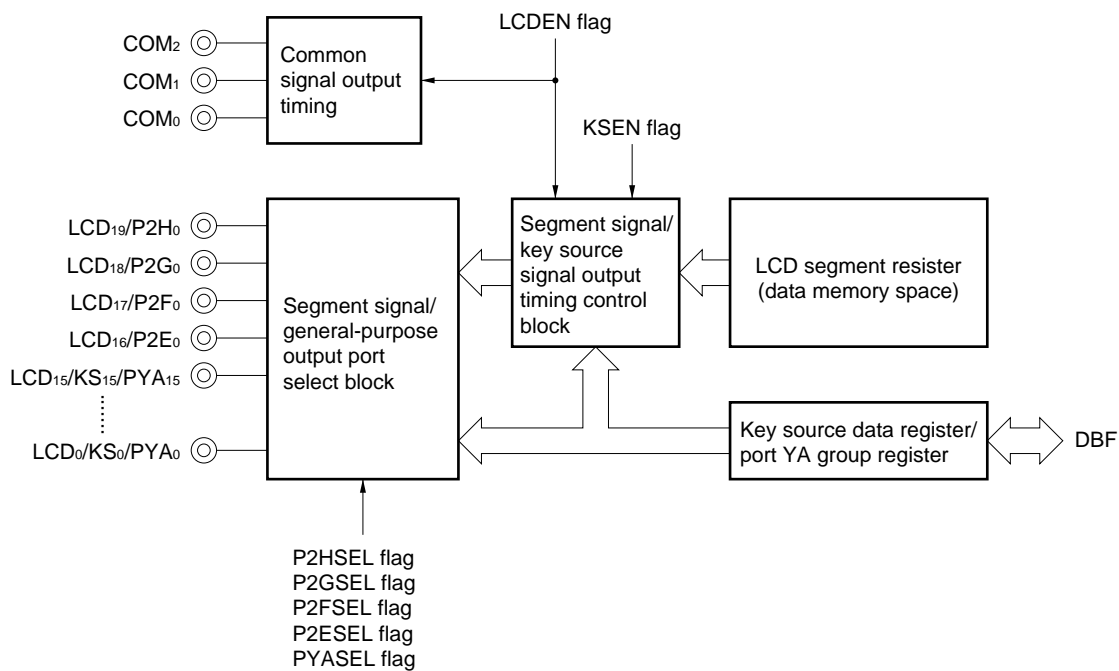
### 19.1 Configuration of LCD Controller/Driver

Figure 19-1 shows the block diagram of the LCD controller/driver.

As shown in the figure, the LCD controller/driver consists of a common signal output timing control block, segment signal/key source signal output timing control block, segment signal/general-purpose output port select block, LCD segment register, and key source data register/port YA group register.

The following section 19.2 outlines the function of each block.

**Figure 19-1. Outline of LCD Controller/Driver**



- Remarks**
1. P2HSEL, P2GSEL, P2FSEL, and P2ESEL (bits 3 to 0 of LCD port select register; refer to **Figure 19-7**) set the output of the LCD<sub>19</sub>/P2H<sub>0</sub>, LCD<sub>18</sub>/P2G<sub>0</sub>, LCD<sub>17</sub>/P2F<sub>0</sub>, and LCD<sub>16</sub>/P2E<sub>0</sub> in the LCD segment signal output or general-purpose output port mode.
  2. PYASEL (bit 0 of LCD mode select register; refer to **Figure 19-9**) sets the output of the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> through LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins in the LCD segment signal output or general-purpose output port mode.
  3. LCDEN (bit 1 of LCD mode select register; refer to **Figure 19-9**) turns ON/OFF all LCD displays.
  4. KSEN (bit 2 of LCD mode select register; refer to **Figure 19-9**) sets the output of the key source signal.

## 19.2 Functional Outline of LCD Controller/Driver

The LCD controller/driver can display up to 60 dots by using a combination of common signal output pins (COM<sub>2</sub> to COM<sub>0</sub>) and segment signal output pins (LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub>).

Figure 19-2 shows the relationship between common signal output pins, segment signal output pins, and display dots.

As shown in this figure, three dots can be displayed at the intersections between one segment line and the COM<sub>2</sub> to COM<sub>0</sub> pins.

The driving mode is 1/3 duty, 1/2 bias, and the drive voltage is supply voltage V<sub>DD</sub>.

The segment signal output pins (LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub>) can also be used as general-purpose output port pins.

When these pins are used as general-purpose output port pins, ports 2H (LCD<sub>19</sub>/P2H<sub>0</sub>), 2G (LCD<sub>18</sub>/2G<sub>0</sub>), 2F (LCD<sub>17</sub>/P2F<sub>0</sub>), 2E (LCD<sub>16</sub>/P2E<sub>0</sub>), and YA (LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub>) can be independently used.

Of the segment signal output pins, the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins are also used as key source signal output pins.

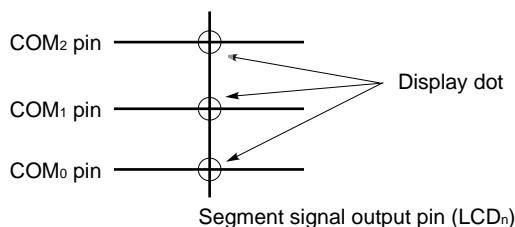
The key source signals and LCD segment signals are output by means of time-division multiplexing.

For details of the general-purpose output ports, refer to **10. GENERAL-PURPOSE PORTS**.

For details of the key source signals, refer to **20. KEY SOURCE CONTROLLER/DECODER**.

The following subsections 19.2.1 through 19.2.5 outline the function of each block of the LCD controller/driver.

**Figure 19-2. Common Signal Output, Segment Signal Output, and Display Dots**



### 19.2.1 LCD segment register

The LCD segment register sets dot data that is used to turn ON/OFF the LCD.

Because this register is mapped in the data memory, it can be controlled by any data memory manipulation instruction.

When the segment signal output pins are used as general-purpose output port pins, this register sets output data.

For details, refer to **19.3**.

### 19.2.2 Common signal output timing control block

The common signal output timing control block controls the common signal output timing of the COM<sub>2</sub>, COM<sub>1</sub>, and COM<sub>0</sub> pins.

These pins output a low level when the LCD is not displayed.

Whether the LCD is displayed or not is selected by the LCD mode select register (RF address 10H).

For details, refer to **19.4**.

### 19.2.3 Segment signal/key source signal output timing control block

The segment signal/key source signal output timing control block controls the segment signal output timing of the LCD<sub>19</sub>/P2H<sub>0</sub> through LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins.

These pins output a low level when the LCD is not displayed.

Whether the LCD is displayed or not is selected by the LCD mode select register.

The segment signal/key source signal output timing control block controls the timing of the segment and key source signals output from the LCD<sub>15</sub>/KS<sub>15</sub> through LCD<sub>0</sub>/KS<sub>0</sub> pins.

Whether the key source signals are used or not is selected by the LCD mode select register.

For details, refer to **19.5**.

### 19.2.4 Segment signal/general-purpose output port select block

The segment signal/general-purpose output port select block selects whether each segment signal output pin is used for LCD display (to output a segment signal) or as a general-purpose output port pin.

This selection is made by using the P2HSEL to P2ESEL flags of LCD port select register and PYASEL flag of LCD mode select register.

For details, refer to **19.4** and **19.5**.

### 19.2.5 Key source data register/port YA group register

The key source data register/port YA group register sets the key source output data that is output from the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins.

The key source signal output data is set by the key source data register (KSR: peripheral address 42H) via the data buffer.

For details, refer to **20. KEY SOURCE CONTROLLER/DECODER**.

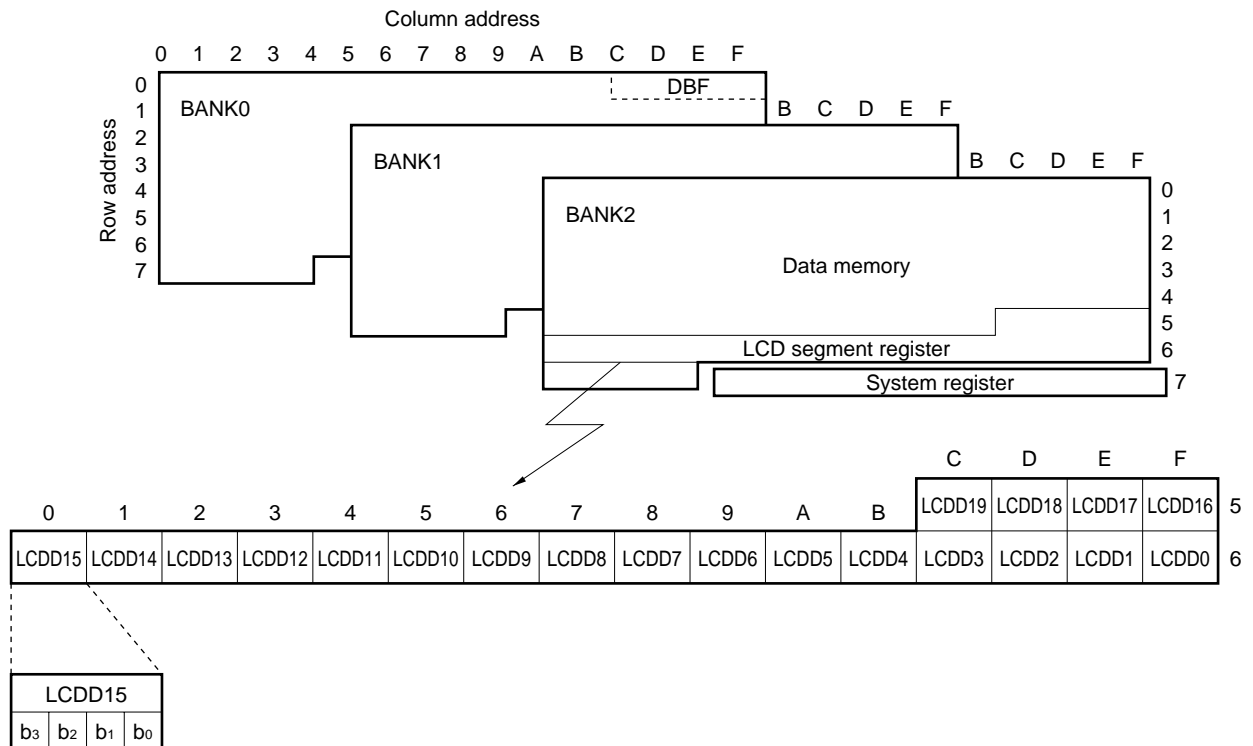
### 19.3 LCD Segment Register

The LCD segment register specifies whether each dot on the LCD is turned on or off.

#### 19.3.1 Configuration of LCD segment register

Figure 19-3 shows the location and configuration of the LCD segment register in the data memory.

**Figure 19-3. Location and Configuration of LCD Segment Register in Data Memory**



### 19.3.2 Function of LCD segment register

Figure 19-4 shows the relation of 1 nibble (4 bits) of the LCD segment register and LCD display dots.

As shown in this figure, the lower 3 bits of display data (ON/OFF data) in 1 nibble of the LCD segment register can be set.

The LCD display dot corresponding to a bit that is set to 1 is turned on, and the dot corresponding to a bit that is reset to 0 is turned off.

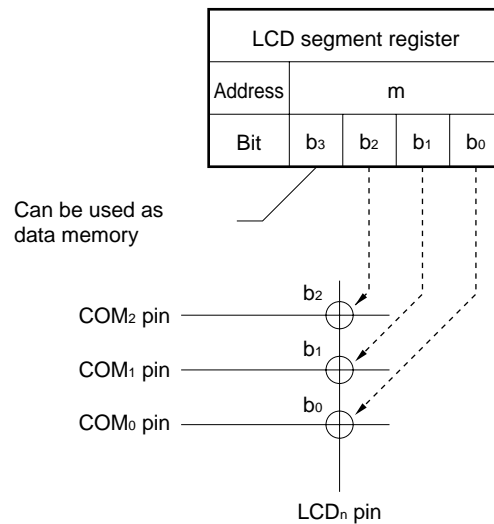
The highest one bit can be used as data memory, however data should be set carefully because the address is the same.

LCDD19 to LCDD16 of the LCD segment register also set output data when the LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>16</sub>/P2E<sub>0</sub> pins are used as output port pins. In this case, output data is set to the least significant bit. The higher 3 bits can be used as data memory, however data setting requires caution because the addresses are the same.

When LCD display is not used, LCDD15 to LCDD0 can be used as normal data memory.

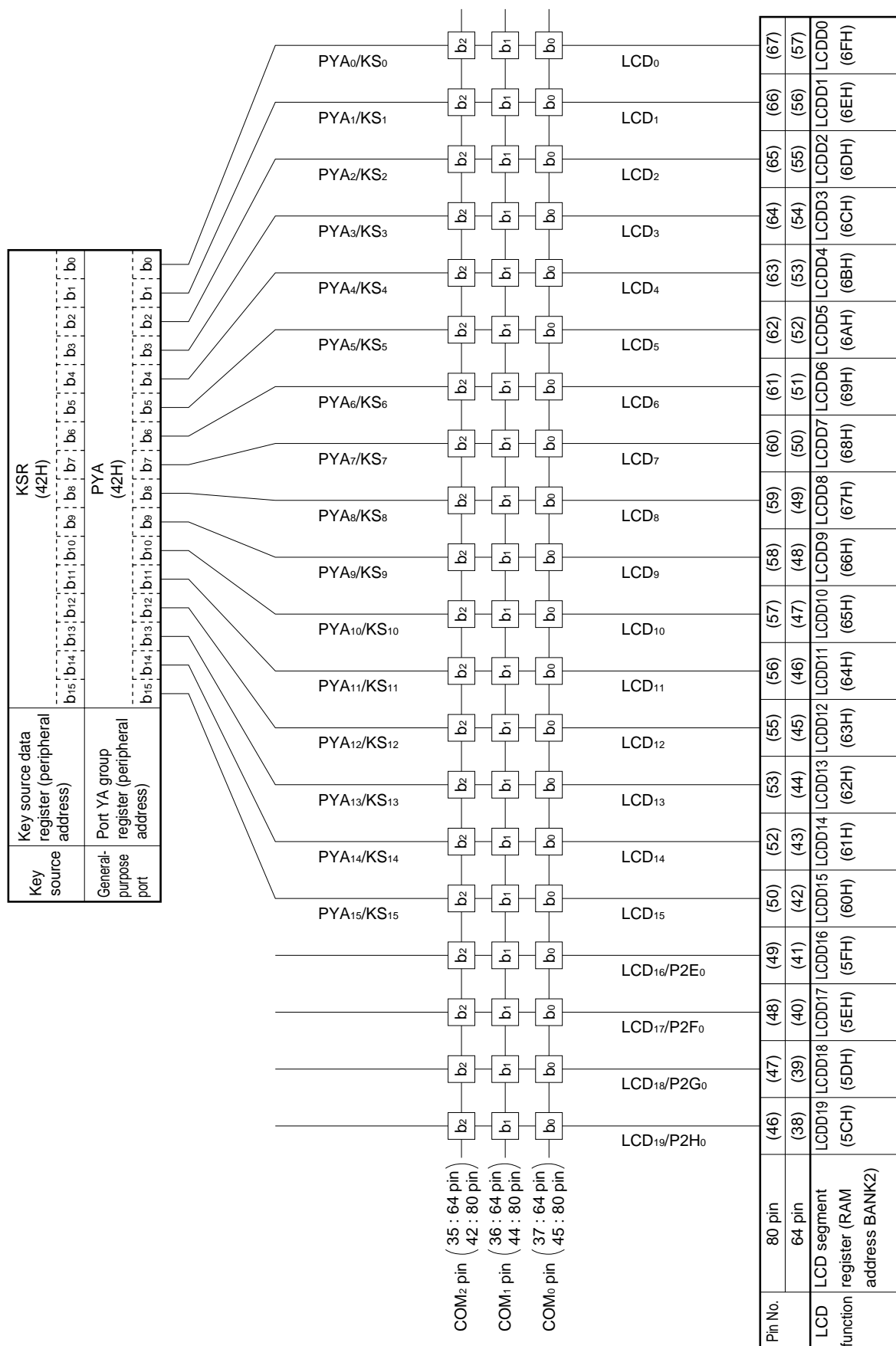
Figure 19-5 shows the relationship between each LCD segment register and LCD display dots that are turned ON/OFF.

**Figure 19-4. Relationship of 1 Nibble of LCD Segment Register and LCD Display Dots**





★ Figure 19-5. Relationship Between LCD Display Dot, Output of Each Pin and Each Data Setting Register



### 19.4 Segment Signal/General-Purpose Output Port Select Block

Figure 19-6 shows the configuration of the segment signal/general-purpose output port select block.

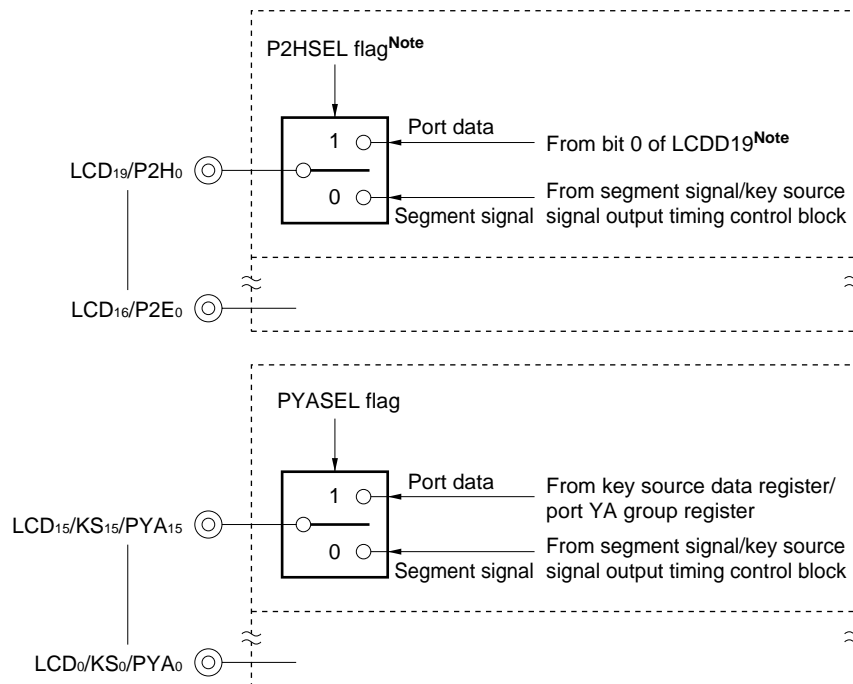
This block specifies whether each pin is used as a segment signal output pin or a general-purpose output port pin, by using the P2HSEL through P2ESEL flags of the LCD port select register and the PYASEL flag of the LCD mode select register. When each flag is “1”, the corresponding pin is set in the general-purpose output port mode; when the flag is “0”, the pin is set in the segment signal output mode.

The LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins can simultaneously output segment signals and key source signals. When port YA is selected, however, port output takes precedence.

Figure 19-7 shows the configuration and function of the LCD port select register.

Figure 19-9 shows the configuration and function of the LCD mode select register.

**Figure 19-6. Configuration of Segment Signal/General-Purpose Output Port Select Block**

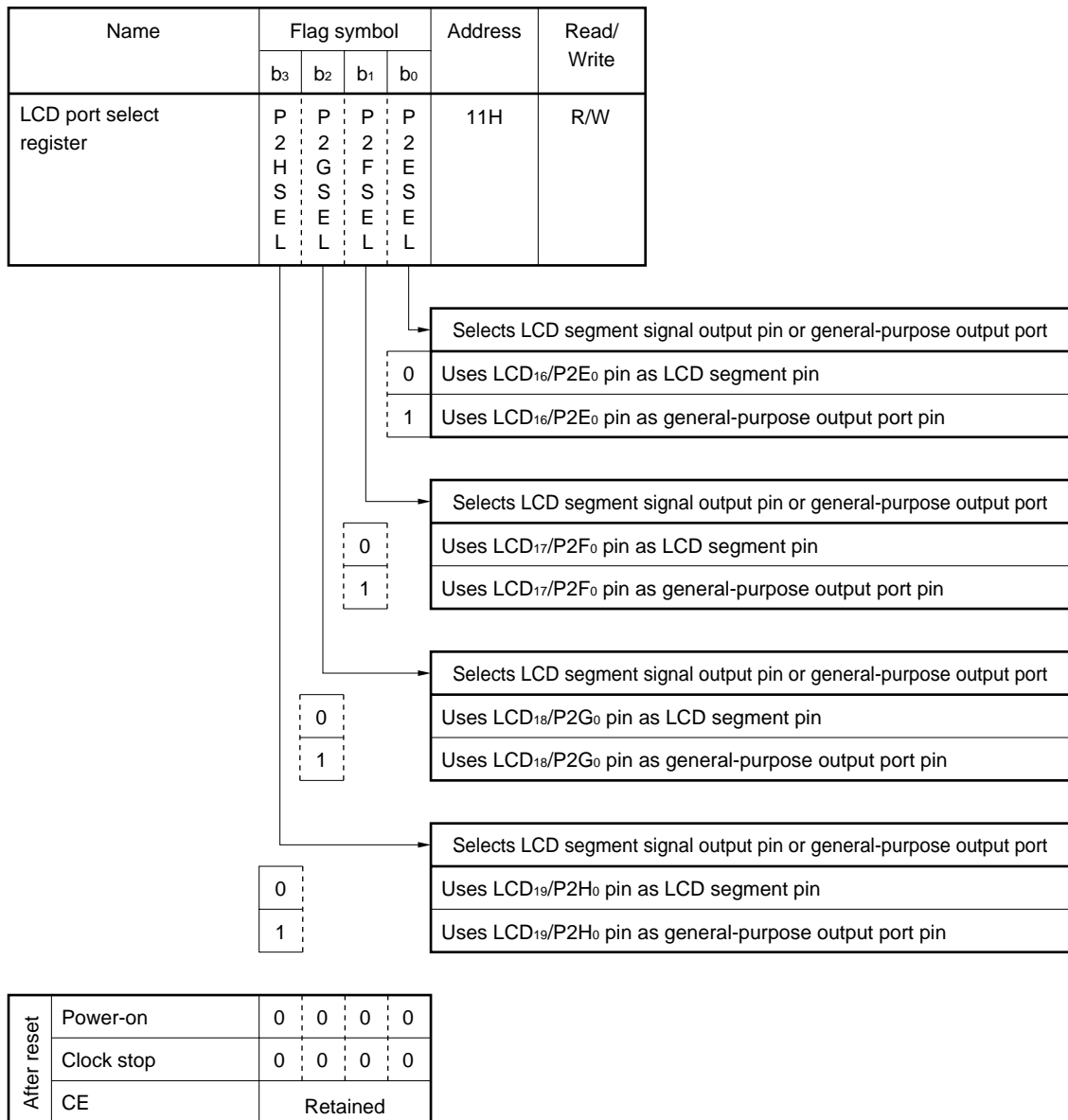


**Note** P2GSEL flag and LCDD18 for LCD<sub>18</sub>/P2G<sub>0</sub> pin.

P2FSEL flag and LCDD17 for LCD<sub>17</sub>/P2F<sub>0</sub> pin.

P2ESEL flag and LCDD16 for LCD<sub>16</sub>/P2E<sub>0</sub> pin.

Figure 19-7. Configuration of LCD Port Select Register



### 19.5 Common Signal Output Timing Control Block and Segment Signal/Key Source Signal Output Timing Control Block

Figure 19-8 shows the configuration of the common signal output timing control block and segment signal/key source signal output timing control block.

The common signal output timing control block controls the output timing of the COM<sub>2</sub> to COM<sub>0</sub> signals.

The segment signal/key source signal output timing control block controls the output timing of the segment signals and key source signals of the LCD<sub>19</sub>/P2H<sub>0</sub> through LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins.

The common and segment signals are output when the LCDEN flag of the LCD mode select register is "1". By clearing the LCDEN flag to "0", therefore, all LCD displays can be turned OFF.

The key source signal is output when the KSEN flag of the LCD mode select register is "1".

When LCD display is not performed, the COM<sub>2</sub> to COM<sub>0</sub> and LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins output a low level.

Figure 19-9 shows the configuration and function of the LCD mode select register.

**Figure 19-8. Configuration of the Common Signal Output Timing Control Block and Segment Signal/Key Source Signal Output Timing Control Block**

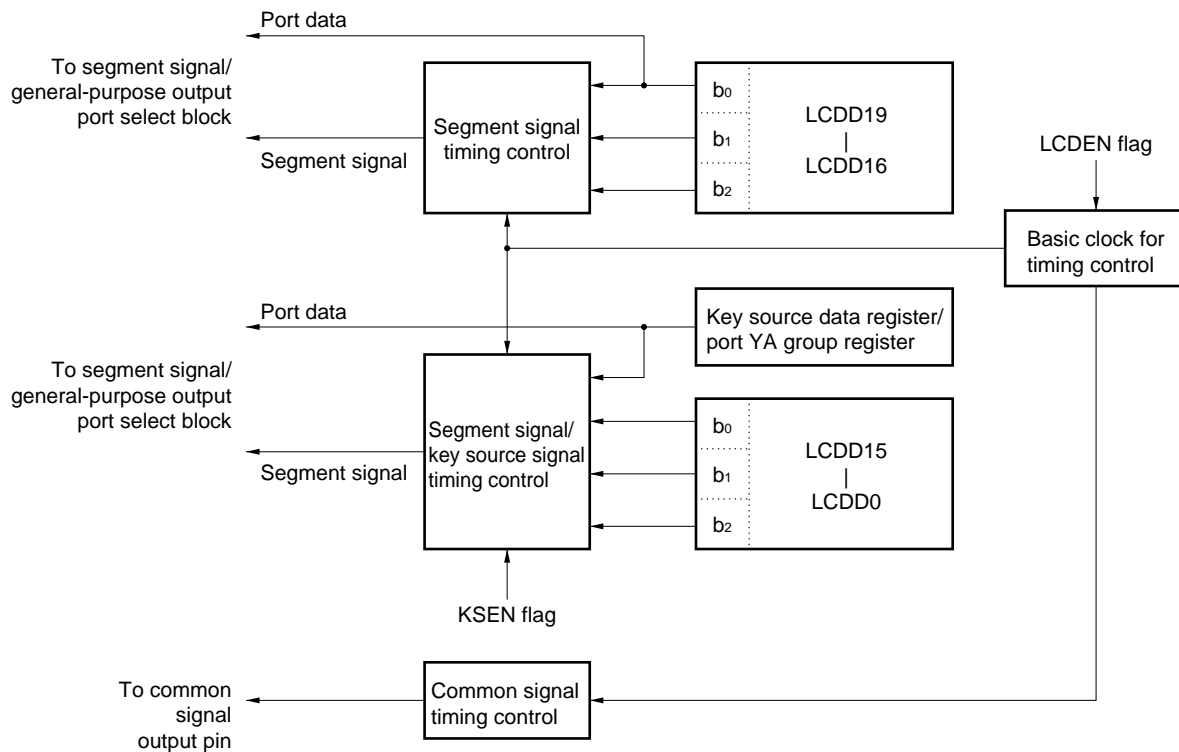
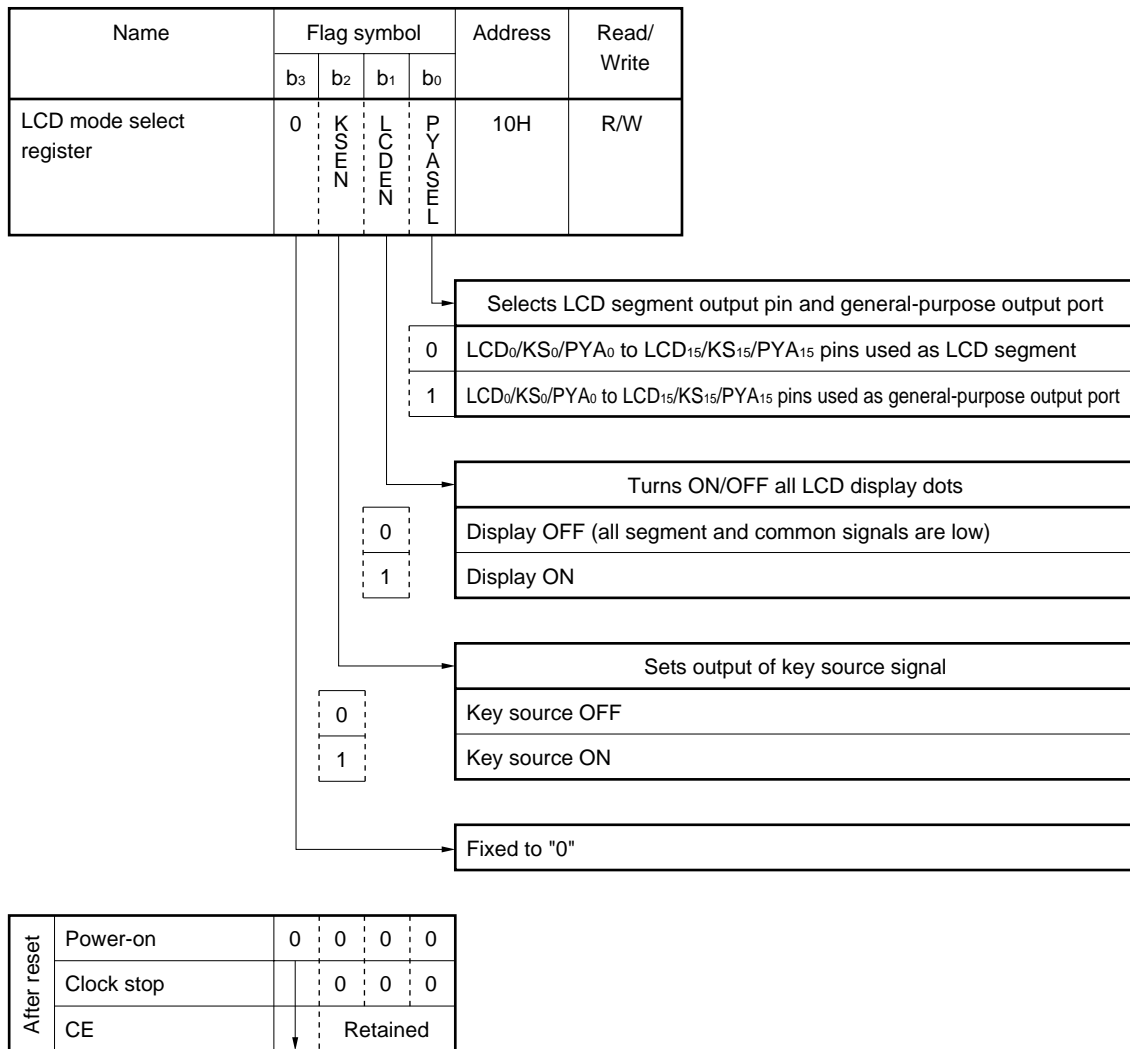


Figure 19-9. Configuration of LCD Mode Select Register



## 19.6 Output Waveforms of Common and Segment Signals

Figures 19-10 to 19-12 show the output waveforms of the common and segment signals.

Figure 19-10 shows the output waveform with the key source signals not output, and Figure 19-11 and 19-12 shows the output waveform with the key source signals output.

As shown in Figure 19-10, the LCD driver outputs signals with a frame frequency of 83 Hz at 1/3 duty, 1/2 bias (voltage average mode).

As the common signals, three levels of voltages (GND,  $1/2 V_{DD}$ , and  $V_{DD}$ ) each having a phase difference of  $1/6$  from the others are output from the  $COM_1$  and  $COM_0$  pins.

Therefore, voltages in a range of  $1/2 V_{DD} \pm 1/2 V_{DD}$  are output. This display mode is called 1/2 bias drive mode.

As the segment signals, two levels (0,  $V_{DD}$ ) of voltages each having a phase corresponding to a display dot are output from each segment signal output pin.

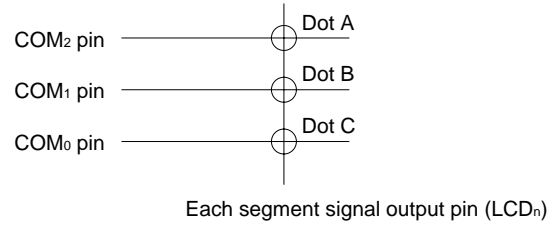
Because three display dots (A, B, and C) are turned ON/OFF by one segment pin as shown in Figure 19-10, eight types of phases <1> through <8> shown in Figure 19-10 are output by combination of each dot, and ON and OFF.

Each display dot is turned on when the potential difference between the common and segment signals reaches  $V_{DD}$ .

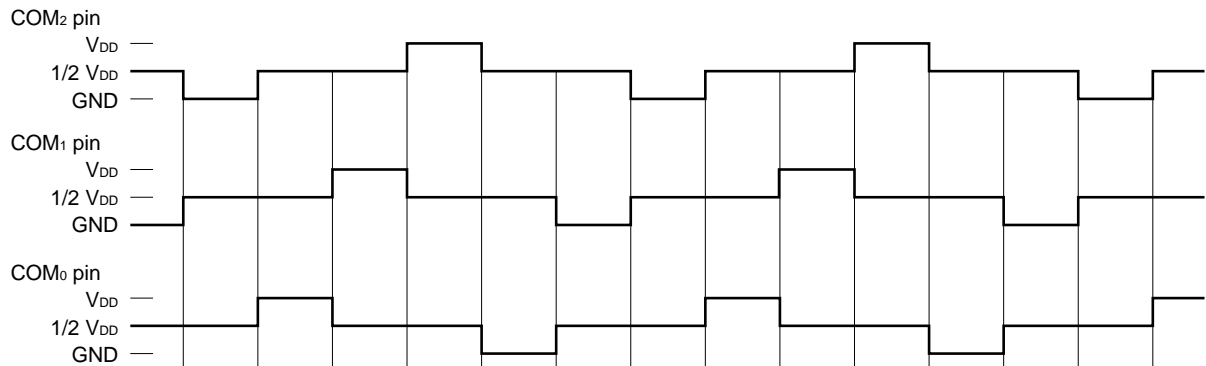
The duty factor at which each display dot is turned on is  $1/3$ , and the frequency of the LCD clock is 167 Hz.

This display mode is called 1/3 duty drive mode.

**Figure 19-10. Common Signal and Segment Signal Output Waveform  
(When Key Source Signal Is Not Output)**



### Common signal



### Each segment pin

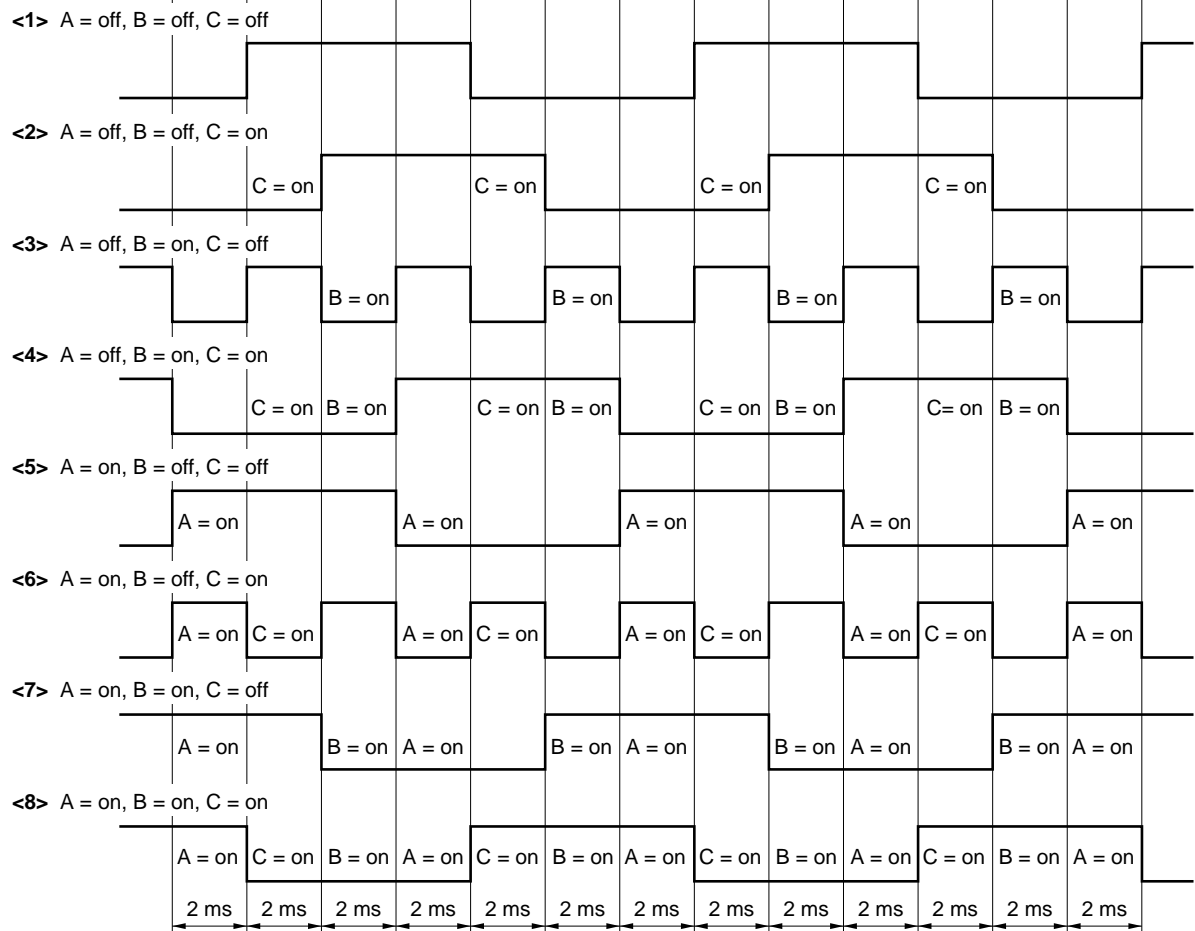
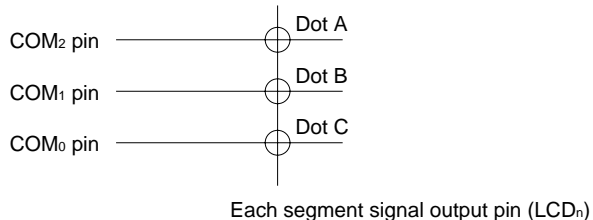
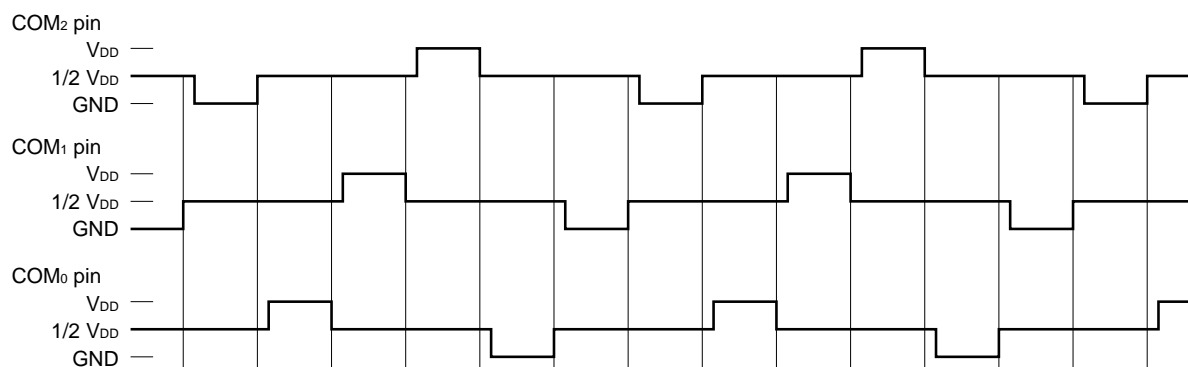


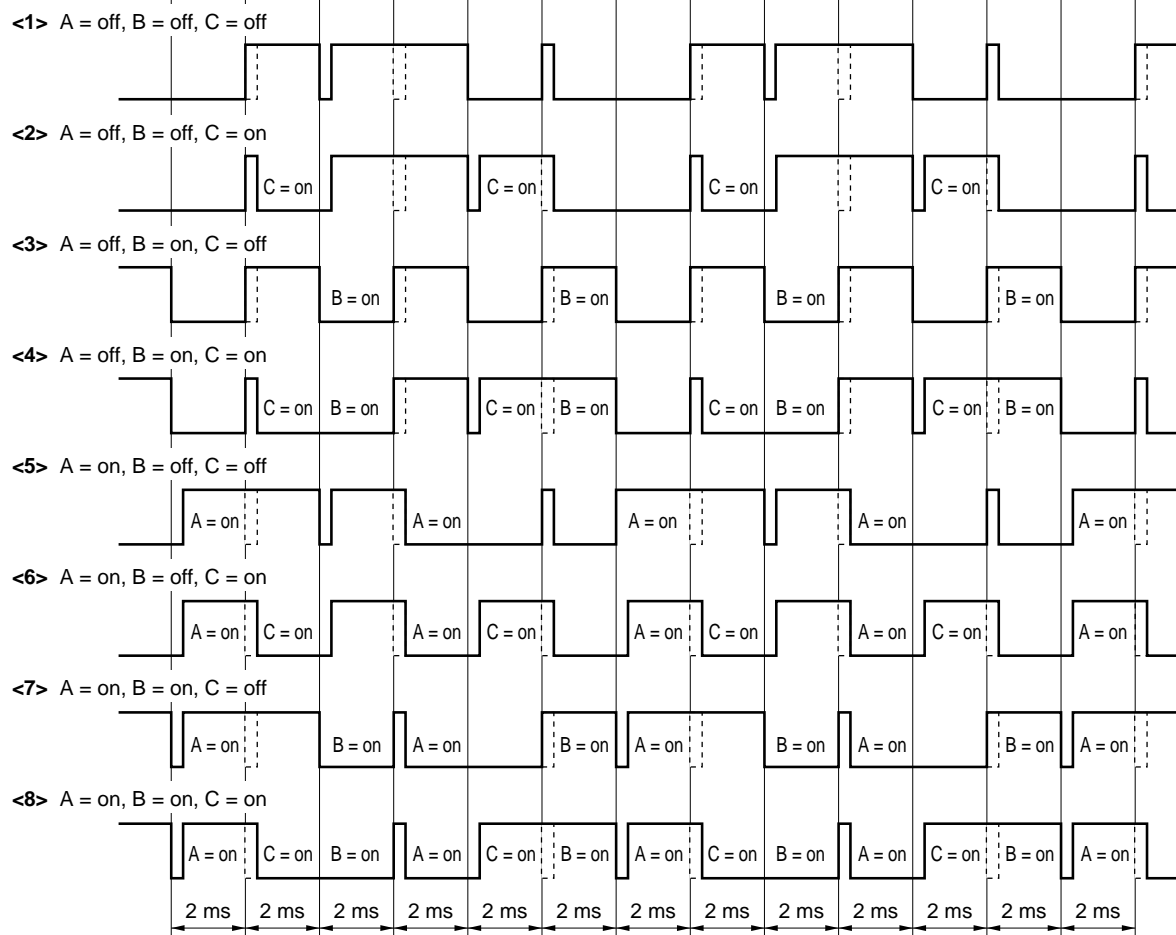
Figure 19-11. Common Signal and Segment Signal Output Waveform  
(When "1" Is Output as Key Source Signal)



### Common signal

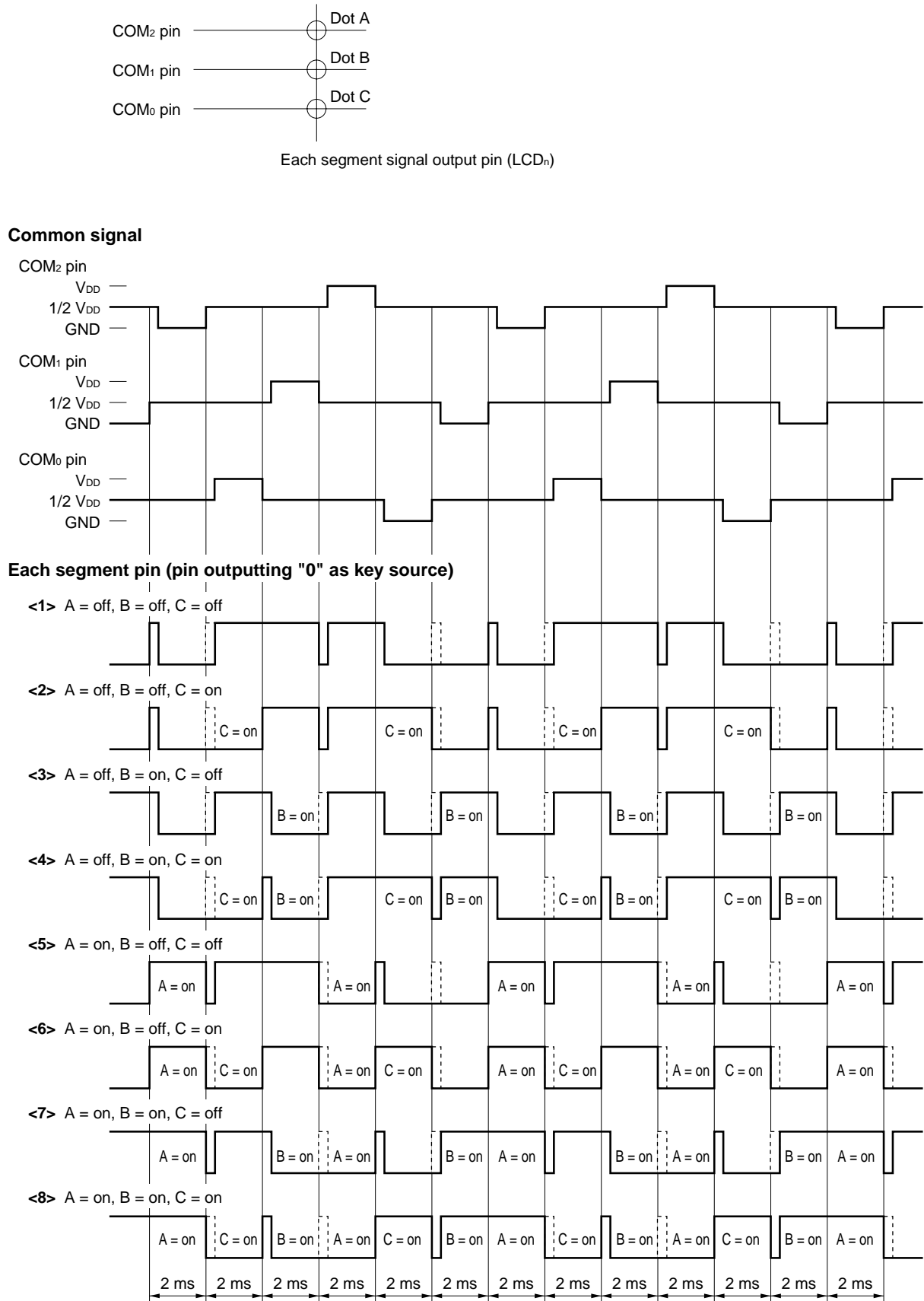


### Each segment pin (pin outputting "1" as key source)





**Figure 19-12. Common Signal and Segment Signal Output Waveform  
(When "0" Is Output as Key Source Signal)**



## 19.7 Using LCD Controller/Driver

Figure 19-13 shows an example of wiring an LCD panel.

An example of a program that turns on a 7-segment LCD panel by using the LCD<sub>0</sub> to LCD<sub>3</sub> pins as shown in Figure 19-13 is shown below.

### Example

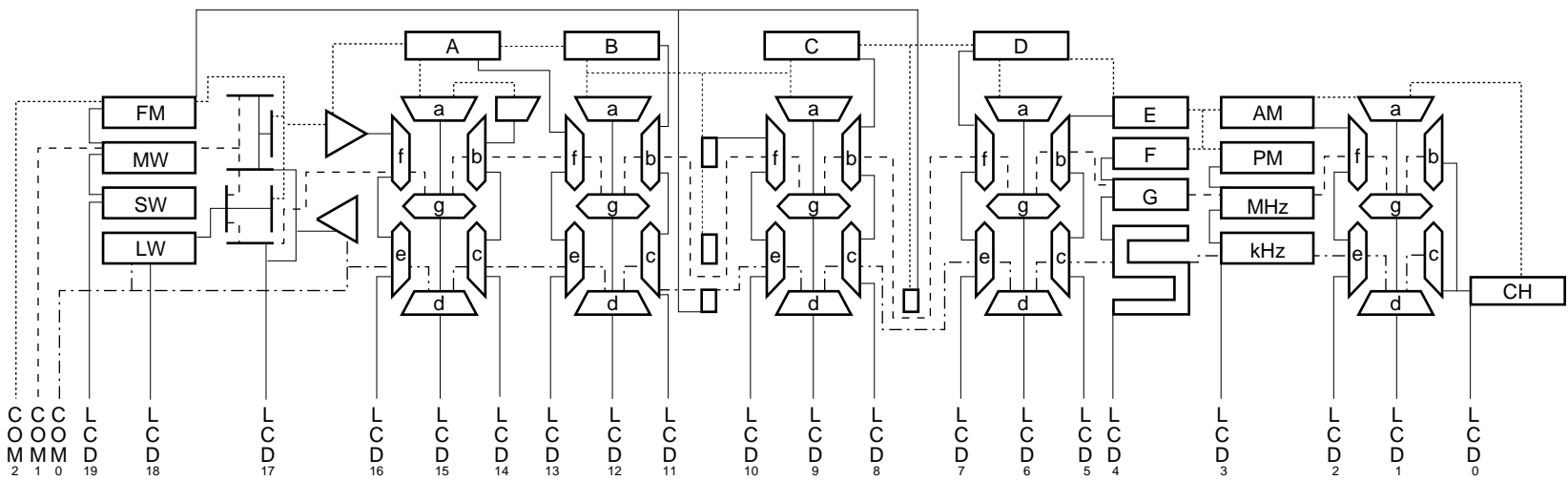
```

PMN0    MEM    0.01H    ; Preset memory number and BK data storage area
CH      FLG    DBF0.2    ; Symbol definition of least significant bit of DBF as "CH" display flag
LCDDATA:    ; Display table data
;          b3 b2 b1 b0 b3 b2 b1 b0 b3 b2 b1 b0 ; Corresponds to LCD segment register
;          - f e   a g d   - b c   ; Corresponds to LCD group register
DW      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 B ; BLANK
DW      0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 B ; 1
DW      0 0 0 0 0 0 0 0 1 0 1 1 1 0 0 1 0 B ; 2
DW      0 0 0 0 0 0 0 0 0 1 1 1 1 0 0 1 1 B ; 3
DW      0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 1 B ; 4
DW      0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 0 1 B ; 5
DW      0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 0 1 B ; 6
DW      0 0 0 0 0 0 0 1 0 0 1 0 0 0 0 1 1 B ; 7
DW      0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 1 1 B ; 8
DW      0 0 0 0 0 0 0 1 0 0 1 1 1 0 0 1 1 B ; 9
DW      0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 1 1 B ; A
DW      0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 1 1 B ; B
DW      0 0 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 B ; C
DW      0 0 0 0 0 0 0 1 1 0 1 0 1 0 0 1 1 B ; D
DW      0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 0 0 B ; E
DW      0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 1 0 B ; F

CLR1    PYASEL
MOV      RPL, #1110B
MOV      AR3, #.DL.LCDDATA SHR 12 AND 0FH
MOV      AR2, #.DL.LCDDATA SHR 8  AND 0FH
MOV      AR1, #.DL.LCDDATA SHR 4  AND 0FH
MOV      AR0, #.DL.LCDDATA          AND 0FH
ADD      AR0, PMN0
ADDC     AR1, #0
ADDC     AR2, #0
ADDC     AR3, #0
MOVT     DBF, @AR
MOV      RPH, #0
MOV      RPL, #0
SKGE     PMN0, #0AH
SET1     CH
BANK2
LD       LCDD0, DBF0
LD       LCDD1, DBF1
LD       LCDD2, DBF2
SET1     LCDEN

```

Figure 19-13. Example of Wiring LCD Panel



Correspondence between segment and common pins and LCD panel display

Segment pin \ Common pin	L C D 19	L C D 18	L C D 17	L C D 16	L C D 15	L C D 14	L C D 13	L C D 12	L C D 11	L C D 10	L C D 9	L C D 8	L C D 7	L C D 6	L C D 5	L C D 4	L C D 3	L C D 2	L C D 1	L C D 0
COM <sub>2</sub>	FM				a		A	a	B	:	a	C	D	a	E	F	PM	AM	a	CH
COM <sub>1</sub>	MW			f	g	b	f	g	b	f	g	b	f	g	b	G	MHz	f	g	b
COM <sub>0</sub>	SW	LW		e	d	c	e	d	c	e	d	c	e	d	c		kHz	e	d	c

## 19.8 Status on Reset

### 19.8.1 On power-on reset

The LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins are specified as LCD segment signal output pins, and output a low level.

The COM<sub>2</sub> to COM<sub>0</sub> pins output a low level.

Therefore, the LCD display is turned off.

### 19.8.2 On execution of clock stop instruction

The LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins are specified as LCD segment signal output pins, and output a low level.

The COM<sub>2</sub> to COM<sub>0</sub> pins output a low level.

Therefore, the LCD display is turned off.

### 19.8.3 On CE reset

Of the LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins, those that are specified as segment signal output pins output segment signals, and those that are specified as general-purpose output port pins retain the current output value.

The COM<sub>2</sub> and COM<sub>0</sub> pins output common signals.

### 19.8.4 In halt status

Of the LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins, those that are specified as segment signal output pins output segment signals, and those that are specified as general-purpose output port pins retain the current output value.

The COM<sub>2</sub> and COM<sub>0</sub> pins output common signals.

## 20. KEY SOURCE CONTROLLER/DECODER

The key source controller/decoder can configure a key matrix of up to 64 keys by outputting key source signals by means of the LCD segment signal output and time division.

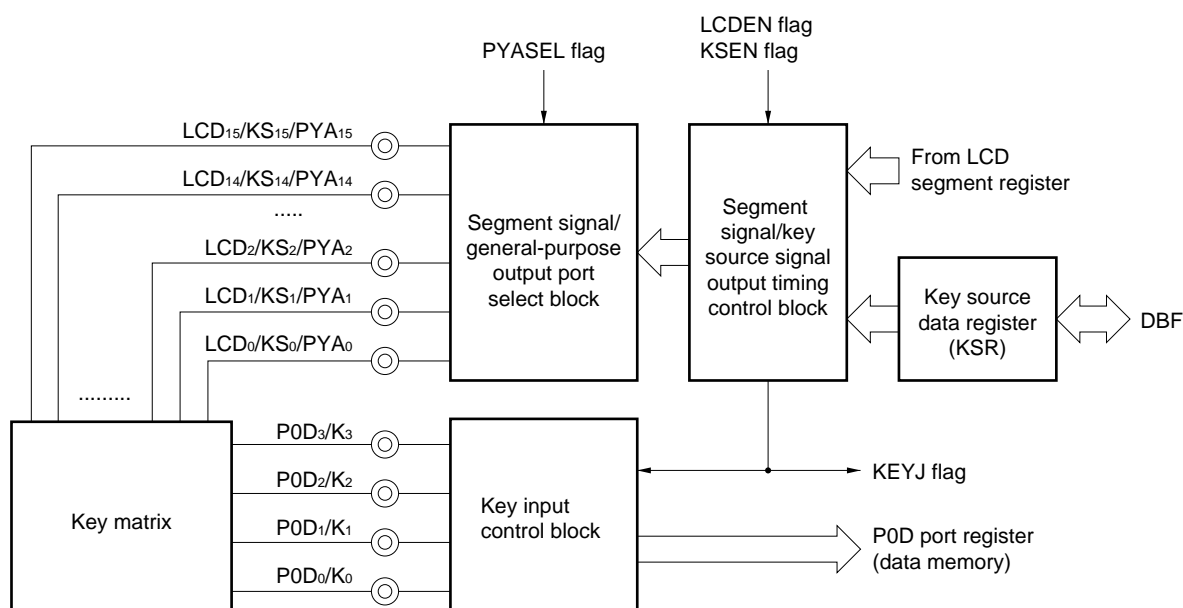
### 20.1 Configuration of Key Source Controller/Decoder

Figure 20-1 shows the configuration of the key source controller/decoder.

As shown in the figure, the key source controller/decoder consists of a segment signal/general-purpose output port select block, segment signal/key source signal output timing control block, key source data register, key input control block, and P0D port register.

The following section 20.2 outlines the function of each block.

**Figure 20-1. Outline of Key Source Controller/Decoder**



- Remarks**
1. PYASEL (bit 0 of the LCD mode select register; refer to **20.4.3 Configuration and function of LCD mode select register**) sets the  $LCD_{15}/KS_{15}/PYA_{15}$  to  $LCD_0/KS_0/PYA_0$  pins in the LCD segment signal output or general-purpose output port mode.
  2. LCDEN (bit 1 of the LCD mode select register; refer to **20.4.3 Configuration and function of LCD mode select register**) turns ON/OFF all LCD displays.
  3. KSEN (bit 1 of the LCD mode select register; refer to **20.4.3 Configuration and function of LCD mode select register**) sets output of the key source signal.
  4. KEYJ (bit 0 of the key input judge register; refer to **20.5.3 Configuration and function of key input judge register**) detects whether the latch contents of the key input pin are valid.

## 20.2 Functional Outline of Key Source Controller/Decoder

The key source controller/decoder can configure a key matrix of up to 64 keys by using key source signal output pins (LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub>) and key input pins (P0D<sub>3</sub>/K<sub>3</sub> to P0D<sub>0</sub>/K<sub>0</sub>).

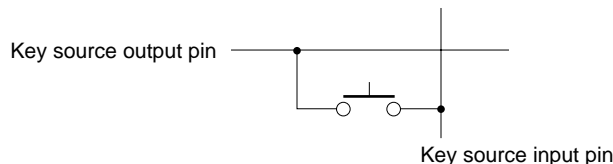
Figure 20-2 shows the example of key matrix configuration.

The LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins are multiplexed with LCD segment signal output pins.

Therefore, the key source signals and LCD segment signals are output by means of time-division multiplexing.

The following subsections 20.2.1 through 20.2.3 outline the function of each block of the key source controller/decoder.

**Figure 20-2. Example of Key Matrix Configuration**



### 20.2.1 Key source data register (KSR)

The key source data register sets the key source output data of the pin that outputs a key source signal.

Data is set to the key source data register via the data buffer.

When data is set to this register, the key source data is output from the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins.

For details, refer to **20.3**.

### 20.2.2 Segment signal/key source signal output timing control block

The segment signal/key source signal output timing block controls the output timing of the key source and segment signals of the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins.

Whether a key source signal is used or not is specified by the LCD mode select register.

The key source signal is not output when LCD display is not used. In this case, the above pins output a low level.

Whether LCD display is not used or not is specified by the LCD mode select register.

For details, refer to **20.4**.

### 20.2.3 Key input control block and P0D port register

The key input control block detects the key input signals input to the P0D<sub>3</sub>/K<sub>3</sub> to P0D<sub>0</sub>/K<sub>0</sub> pins in synchronization with key source signal output timing.

To output the key source signals from the LCD<sub>15</sub>/KS<sub>15</sub> through LCD<sub>0</sub>/KS<sub>0</sub> pins, therefore, the P0D<sub>3</sub>/K<sub>3</sub> to P0D<sub>0</sub>/K<sub>0</sub> pins are used as key input pins.

The key input data is read by the P0D port register (address 73H of BANK0) in the data memory.

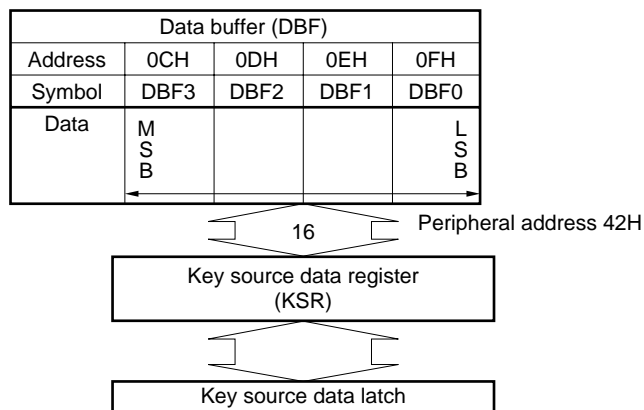
For details, refer to **20.5**.

## 20.3 Key Source Data Setting Block

### 20.3.1 Configuration of key source data setting block

Figure 20-3 shows the configuration of the key source data setting block.

**Figure 20-3. Configuration of Key Source Data Setting Block**



### 20.3.2 Function of key source data setting block

The key source data setting block sets the key source data to be output from the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins.

The key source data is set to the key source data register (KSR: peripheral address 42H) via the data buffer.

Each bit of the key source data register corresponds to the LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins, and sets the key source data of each pin.

When “1” is set to a bit of the key source data register, the pin corresponding to this bit outputs a high level as a key source signal; when the bit is reset to 0, the corresponding pin outputs a low level.

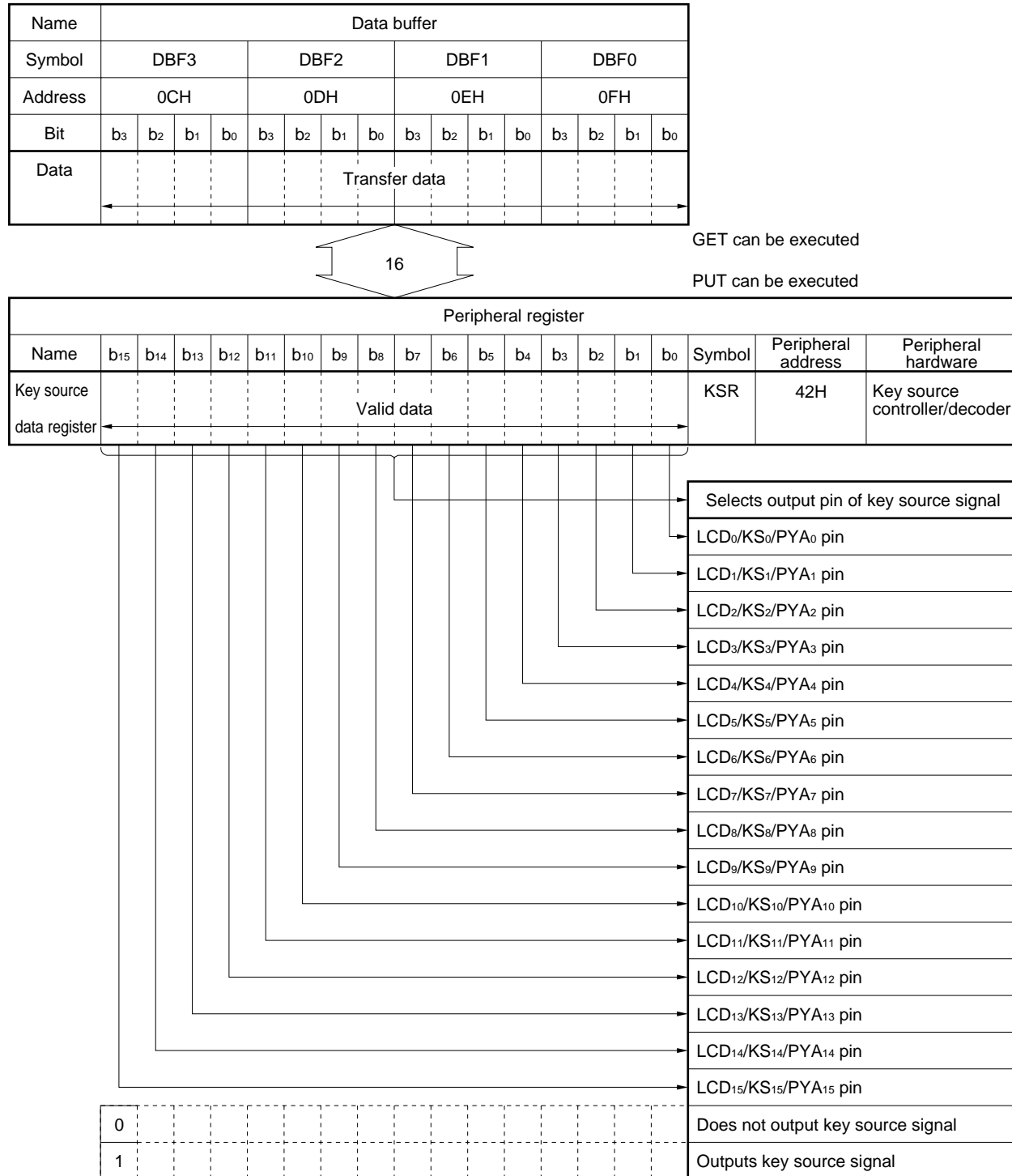
For the output timing, refer to **20.4**.

The following subsections 20.3.3 explains the configuration and function of the key source data register.

Also refer to **Figure 19-5** in **19. LCD CONTROLLER/DRIVER**.

### 20.3.3 Configuration and function of key source data register (KSR)

The configuration and function of the key source data register are illustrated below.



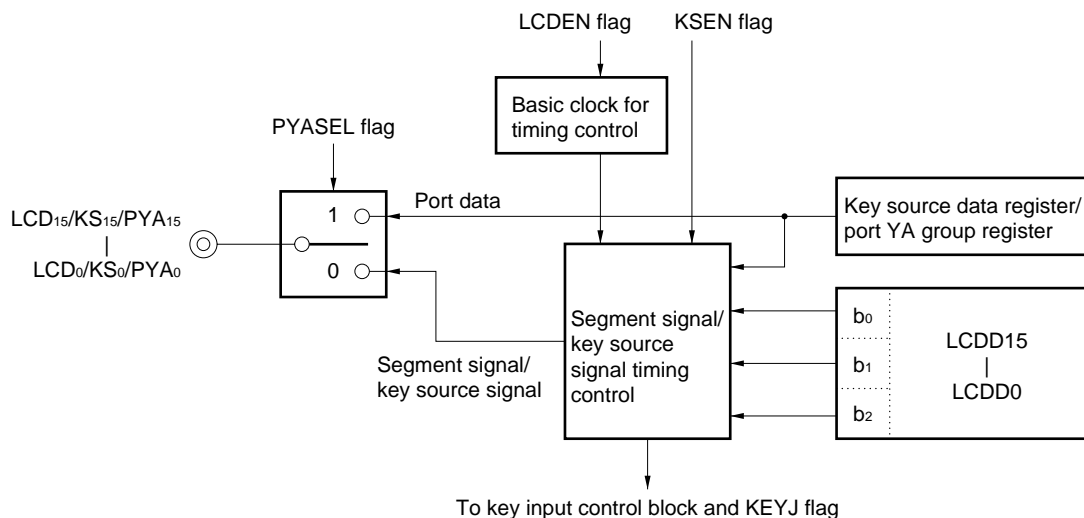


## 20.4 Output Timing Control Blocks and Segment/Port Select Block

### 20.4.1 Configuration of output timing control blocks and segment/port select block

Figure 20-4 shows the configuration of the common signal and segment signal/key source signal output timing control blocks and segment signal/general-purpose output port select block.

Figure 20-4. Configuration of Timing Control Blocks and Port Select Block



### 20.4.2 Function of output timing control block

The segment signal/key source signal output timing control block controls the output timing of the key source and segment signals.

The LCD segment signal is output when the LCDEN flag of the LCD mode select register is "1".

All the LCD display dots can be turned OFF by resetting the LCDEN flag to 0. At this time, a low level is output as the segment signal, and the key source signal is not output.

To output the key source signal, therefore, the LCDEN flag must be "1".

The key source signal is also output when the KSEN flag of the LCD mode select register is "1".

Therefore, the KSEN flag is used to specify whether the key source signal is used or not.

To output the key source signal, therefore, the LCDEN and KSEN flags must be "1".

The following subsection 20.4.3 indicates the configuration and function of the LCD mode select register.

Subsection 20.4.4 shows the output waveforms of the key source and segment signals.

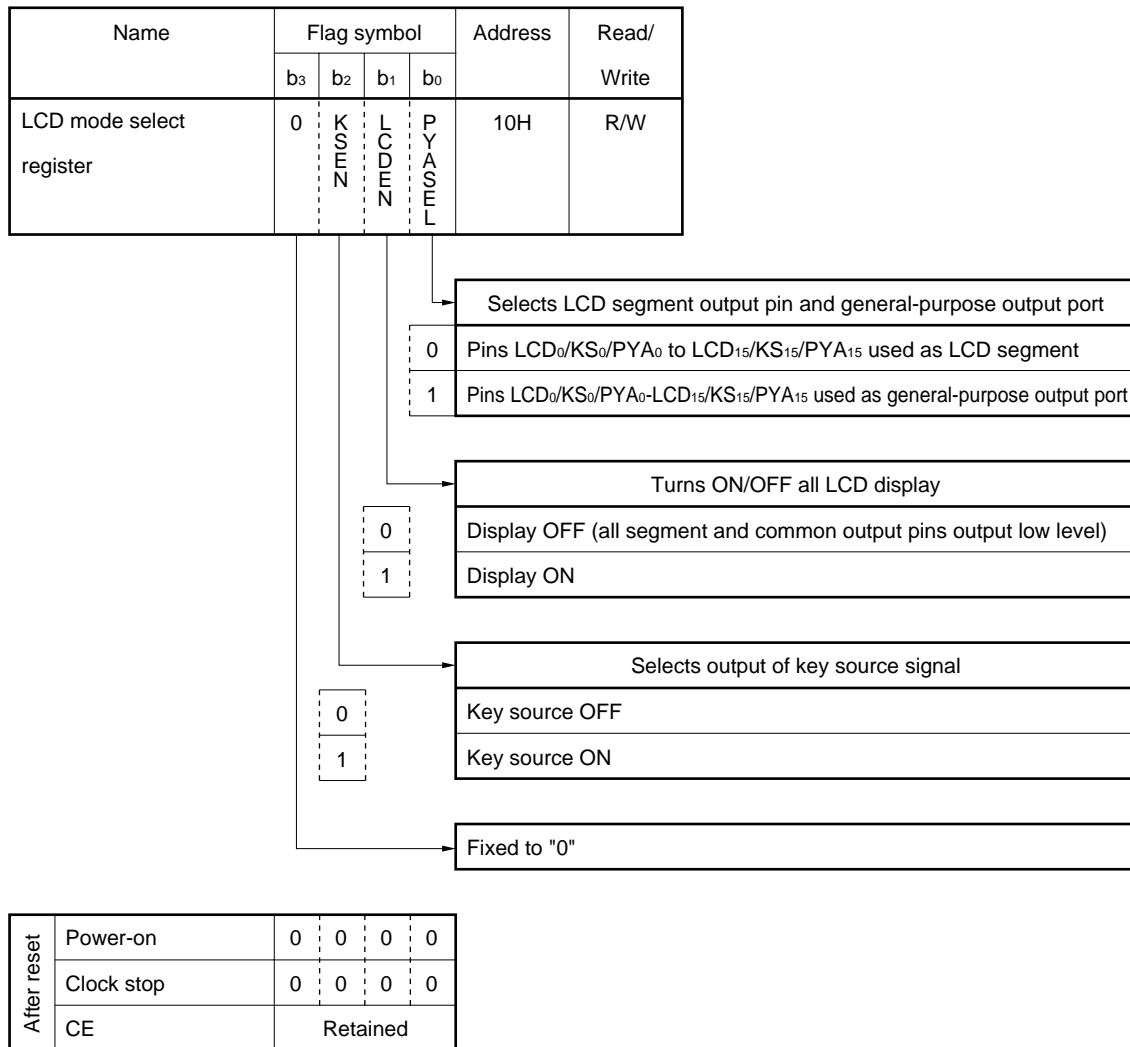
For the relationship between the common and segment signals of the LCD, and key source signal, refer to

## 19. LCD CONTROLLER/DRIVER.

### 20.4.3 Configuration and function of LCD mode select register

The LCD mode select register turns ON/OFF all the LCD display dots, and specifies output of the key source signal.

The configuration and function of this register are illustrated below.



### 20.4.4 Output waveforms of segment and key source signals

Figures 20-5 and 20-6 show the output waveforms of the key source and segment signals.

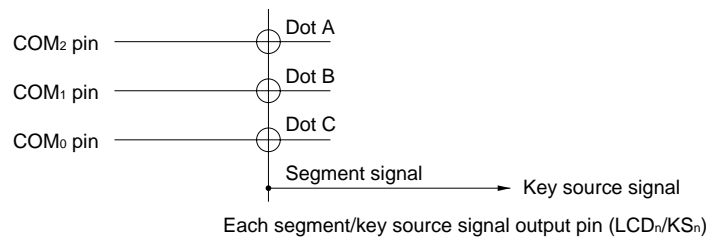
As shown in figures, the key source and segment signals are output by means of time-division multiplexing.

The key source signal is output for 220 μs at intervals of 4 ms.

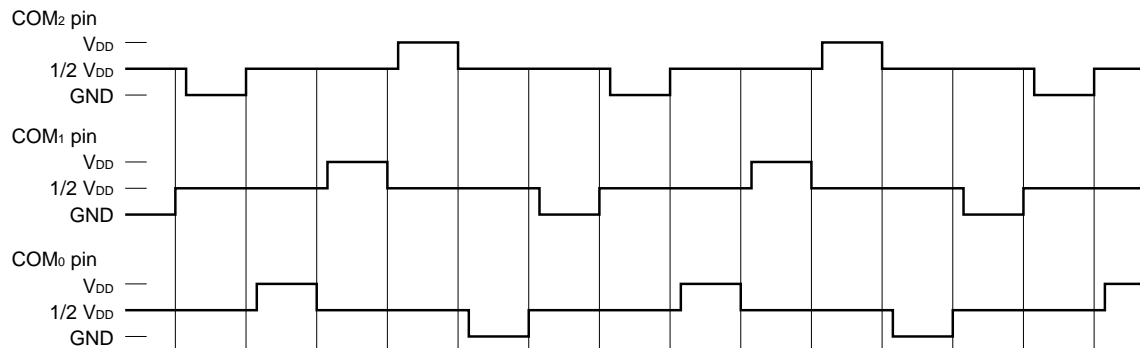
To put it in another way, a pin corresponding to a bit of the key source data register that is set to 1 outputs a high level for 220 μs every 4 ms, and a pin corresponding to a bit of the key source data register that is reset to 0 outputs a low level for 220 μs every 4 ms.

When output of the key source signal is selected (KSEN flag = 1), pins that do not output key source signals (LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>16</sub>/P2E<sub>0</sub>) output the waveform shown in Figure 20-6. However, a waveform of "0" is output as the key source data.

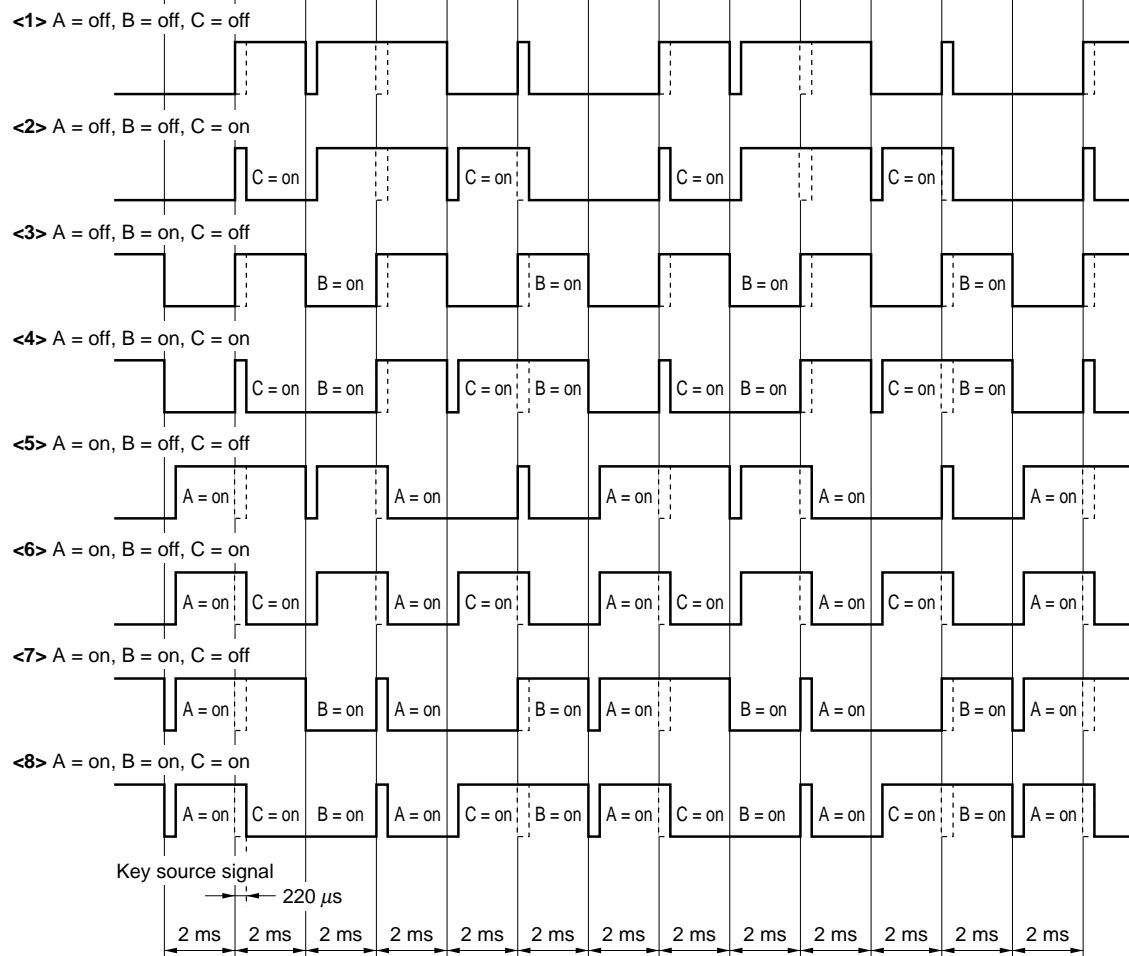
**Figure 20-5. Output Waveforms of Segment and Key Source Signals  
(When "1" Is Output as Key Source Signal)**



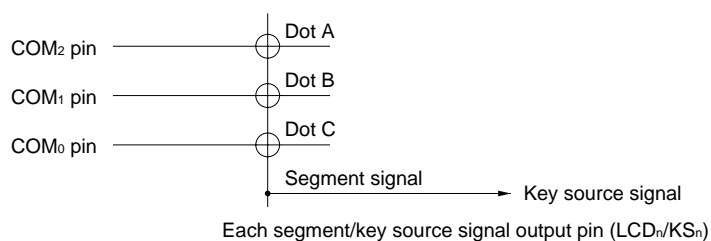
### Common signal



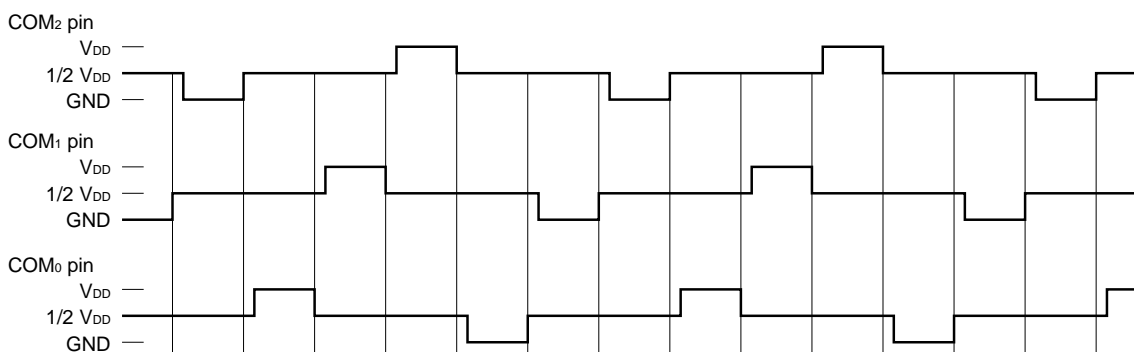
### Each segment pin (pin outputting "1" as key source)



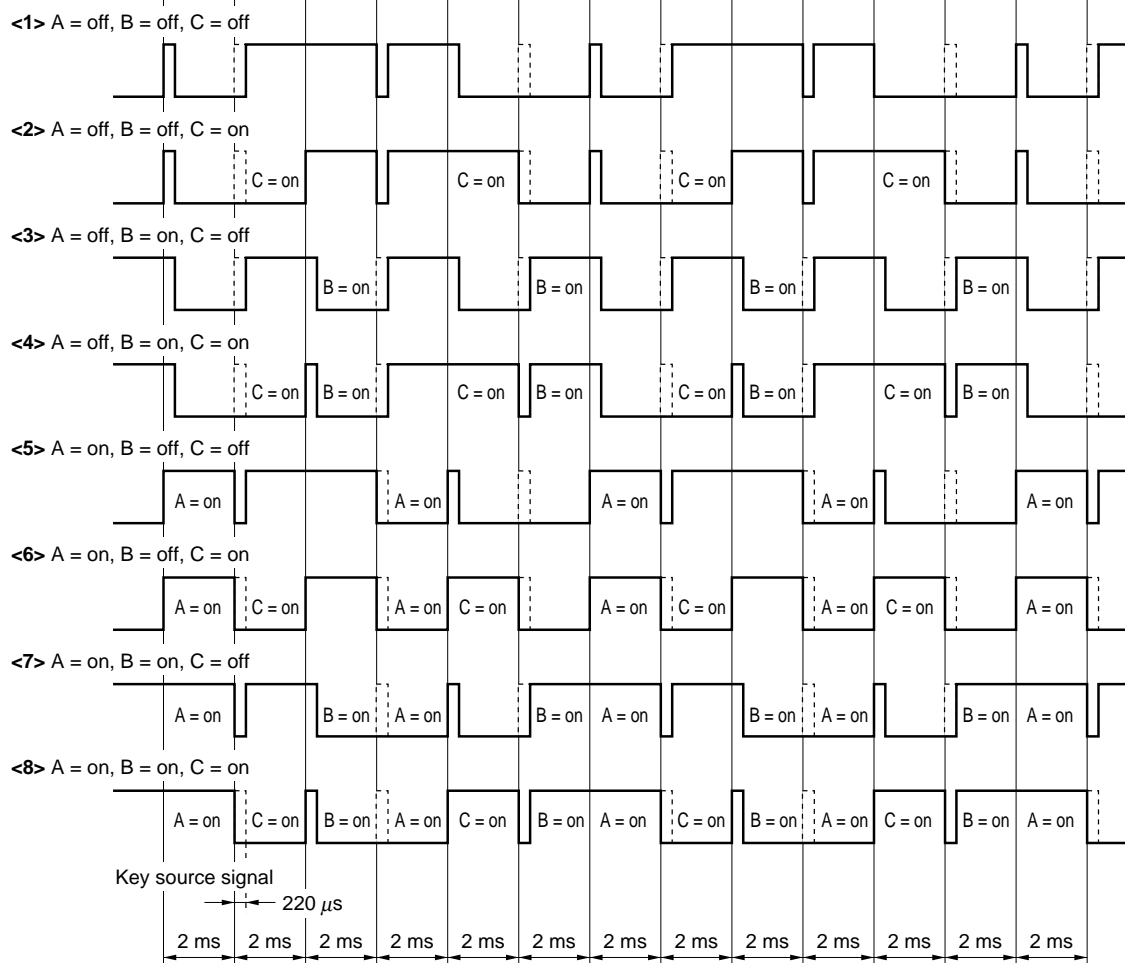
**Figure 20-6. Output Waveforms of Segment and Key Source Signals  
(When "0" Is Output as Key Source Signal)**



### Common signal



### Each segment pin (pin outputting "0" as key source)

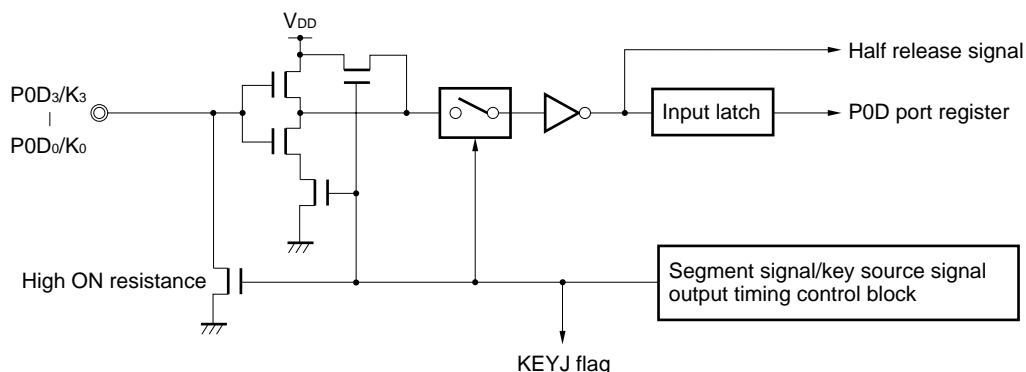


## 20.5 Key Input Control Block

### 20.5.1 Configuration of key input control block

Figure 20-7 shows the configuration of the key input control block.

Figure 20-7. Configuration of Key Input Control Block



### 20.5.2 Function of key input control block

The key input control block controls the timing to read the key input signals from the P0D3/K3 to P0D0/K0 pins and reads the key input data.

Figure 20-8 illustrates the key input signals and key input timing.

As shown in this figure, the internal-pull down resistors of the P0D3/K3 to P0D0/K0 pins are turned off while the display data of the LCD segment is output, and turned on only for 220  $\mu$ s while the key source signal is output.

For the duration of 220  $\mu$ s during which the key source signal is output, the input signal of each key input pin is connected to the input latch.

Therefore, the signal input to each key input pin can be detected in the 220  $\mu$ s during which the key source signal is output.

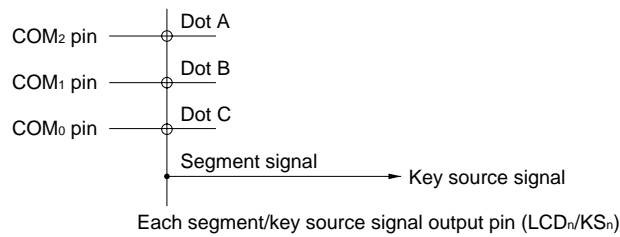
Figure 20-9 shows the timing chart of the key source signal, key input signal, and key input data (P0D port register).

Whether a key source signal is output or not is detected by the KEYJ flag of the key input judge register (RF address 16H).

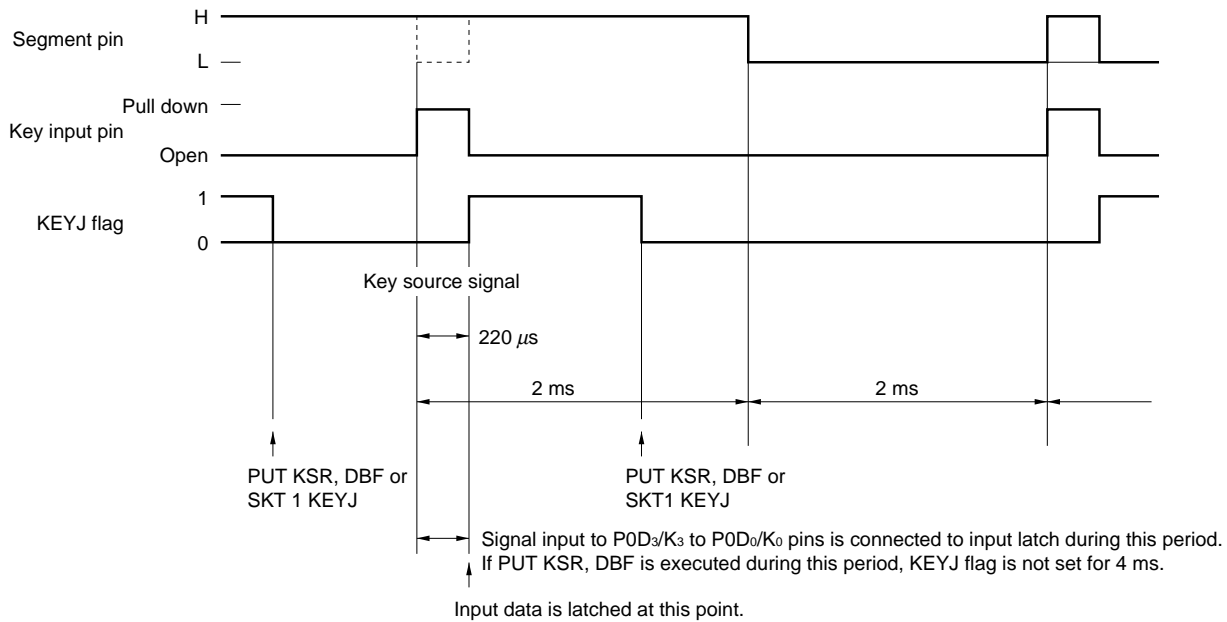
The KEYJ flag is set after the key source signal has been output for 220  $\mu$ s, and is reset when data has been set to the key source data register and when the content of the KEYJ flag has been read.

By detecting the KEYJ flag after the key source signal data has been output to the key source data register, and then detecting the status of each key input pin after the KEYJ flag has been set to 1, the key can be input.

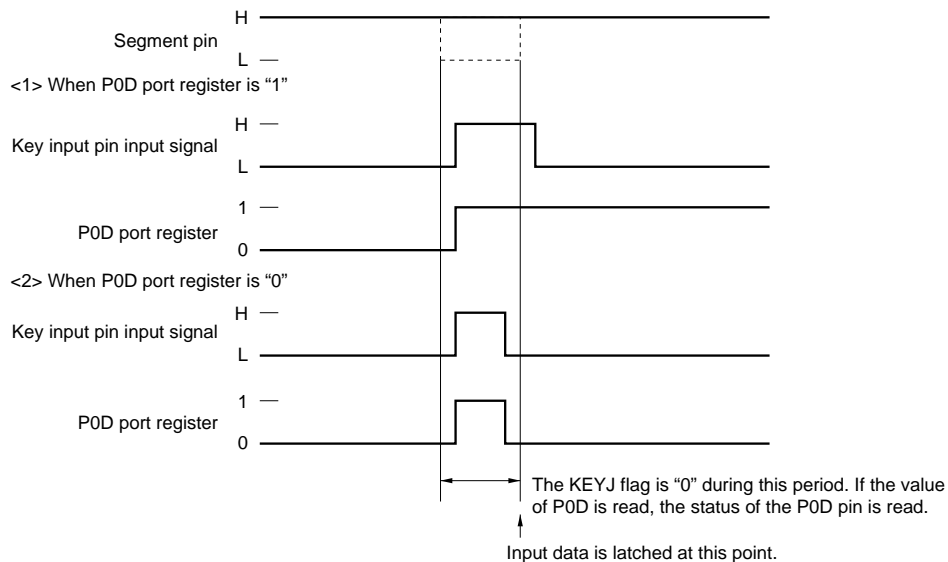
The following subsection 20.5.3 explains the configuration and function of the key input judge register.

**Figure 20-8. Key Source Signal and Key Input Timing**

Each segment pin (pin outputting "1" as key source, A = ON, B = ON, C = OFF)



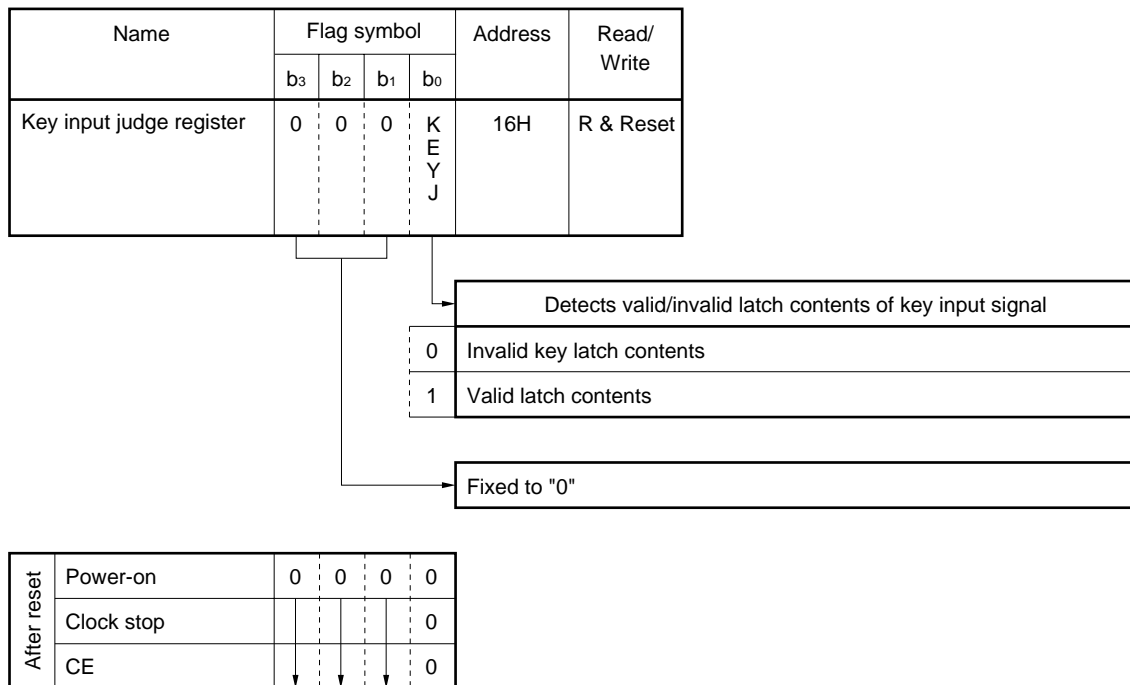
**Caution** The KEYJ flag is not set to 1 when in HALT mode.

**Figure 20-9. Timing Chart of Key Source Signal, Key Input Signal, and Key Input Data (P0D port register)**

### 20.5.3 Configuration and function of key input judge register

The key input judge register detects the presence or absence of the key input signal latch when the LCD segment signal output pins are shared with key source signal output.

The configuration and functions of this register are illustrated below.



**Caution** The KEYJ flag is not set to 1 when in HALT mode.  
The KEYJ flag retains the data prior to HALT instruction execution.

## 20.6 Using Key Source Controller/Decoder

### 20.6.1 Configuring key matrix

Figure 20-10 shows an example of configuring a key matrix.

As shown in this figure, the key matrix can be configured for up to 64 keys.

Because the key source signal output pins also output the LCD segment signals at the same time, diode "A" must be used to prevent the reverse flow of the LCD segment signal if a momentary switch is used.

Diodes "B" and "C" are used to prevent sneaking of the key source signal.

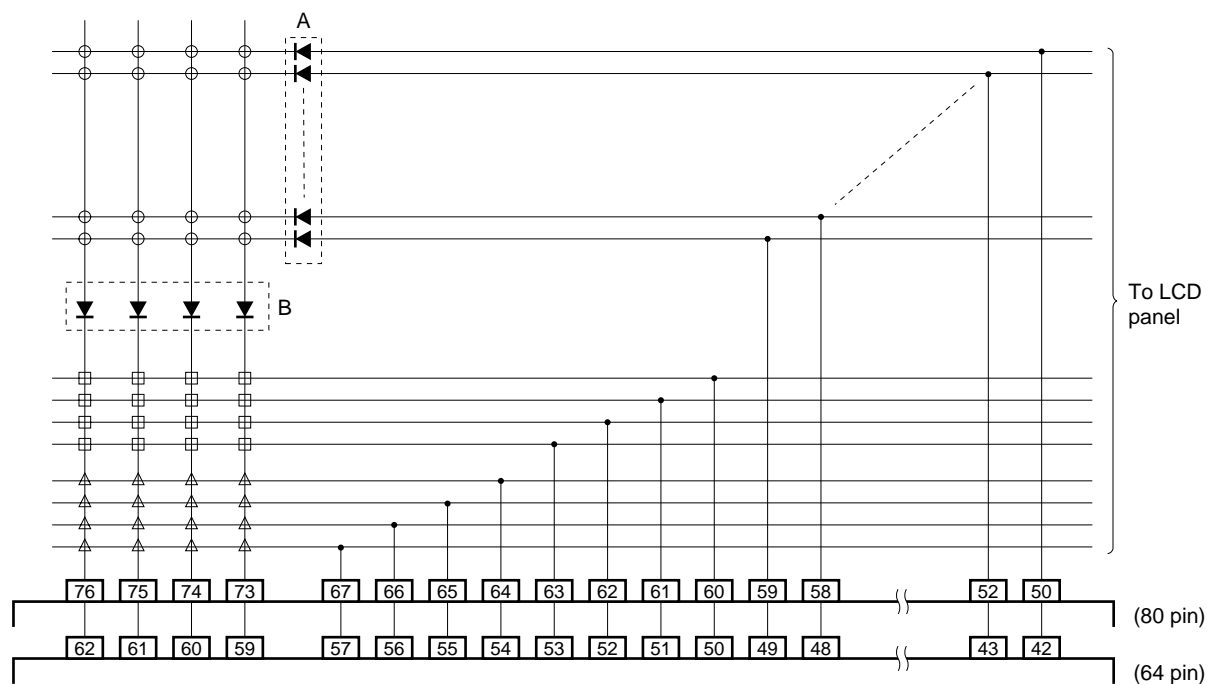
Use a PNP transistor as the transistor switch.

The following (1) explains the points to be noted when an NPN transistor is used.

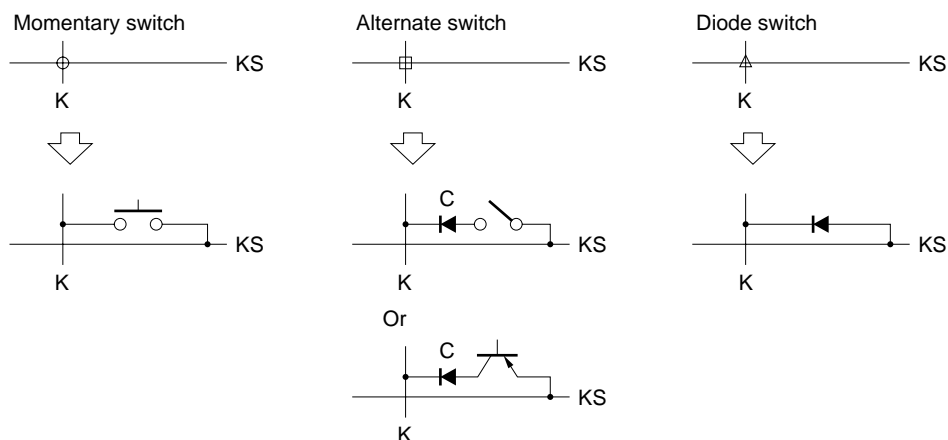
(2) through (4) explain the points to be noted if diodes A, B, and C are not used.

★

**Figure 20-10. Example of Key Matrix Configuration**



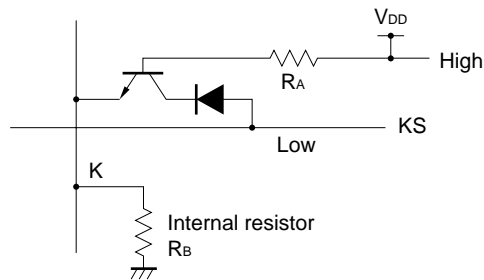
#### Configuration of each switch





**(1) Notes on using NPN transistor switch**

When an NPN transistor switch is used, the low level may not be accurately read as illustrated in the figure below.



If KS is low and a high level is input to the base of the transistor in the figure on the left, voltage  $V_K$  input to K is as follows.

$$V_K = \frac{R_B}{R_A + R_B} \times (V_{DD} - V_{BE})$$

A low level must be input to K at this time because KS is low. However, the voltage input to K changes depending on  $R_A$  and  $R_B$ , as indicated by the above expression.

Therefore, a low level may not be input depending on the values of  $R_A$  and  $R_B$ .

**(2) Notes when diode A is not used**

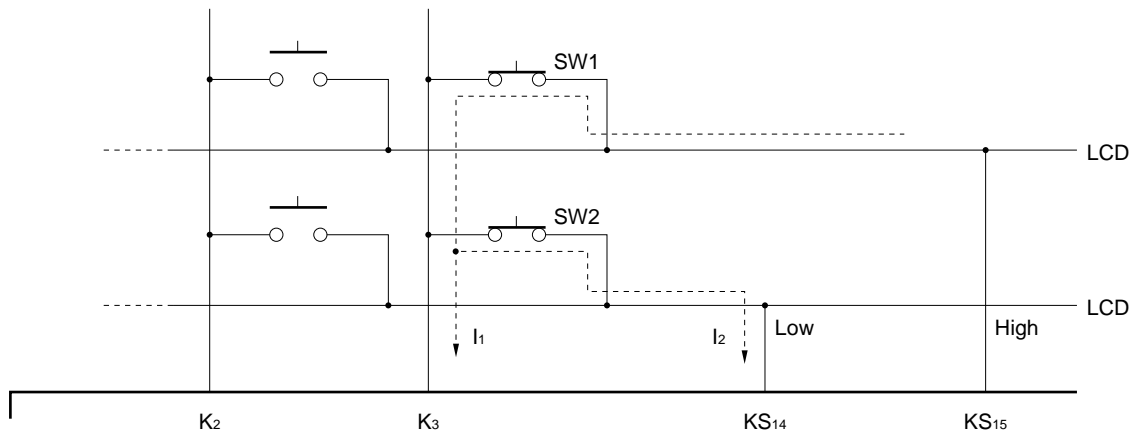
An example of a circuit where diode A is missing is shown below.

Suppose switches SW1 and SW2 are ON,  $KS_{15}$  outputs a high level, and  $KS_{14}$  outputs a low level, as shown below.

If diode A is missing, currents  $I_1$  and  $I_2$  indicated by the dotted lines flow.

Consequently, the high level of  $KS_{15}$  and low level of  $KS_{14}$  are not output correctly because of  $I_2$ , and the key input data of  $K_3$  cannot be accurately read.

If  $KS_{15}$  and  $KS_{14}$  are used to output LCD segment signals, the LCD cannot be turned ON/OFF correctly.



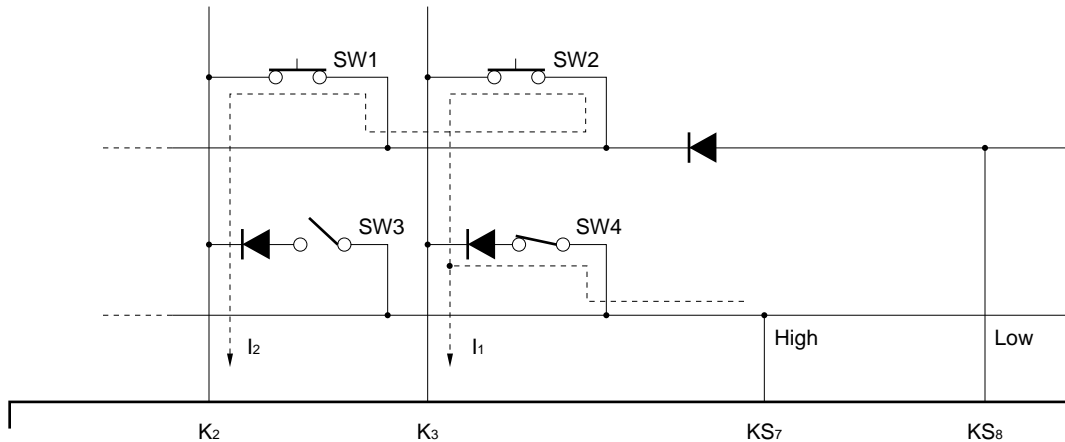
**(3) Notes when diode B is not used**

An example of a circuit where diode B is not used is shown below.

Suppose switches SW1, SW2, and SW4 are ON, and KS<sub>7</sub> outputs a high level, as shown below.

If diode B is missing, currents  $I_1$  and  $I_2$  flow as indicated by the dotted lines.

Consequently, a high level is input to K<sub>2</sub> because of  $I_2$  despite that switch SW3 is OFF, and it is judged that SW3 is ON.

**(4) Note when diode C is not used**

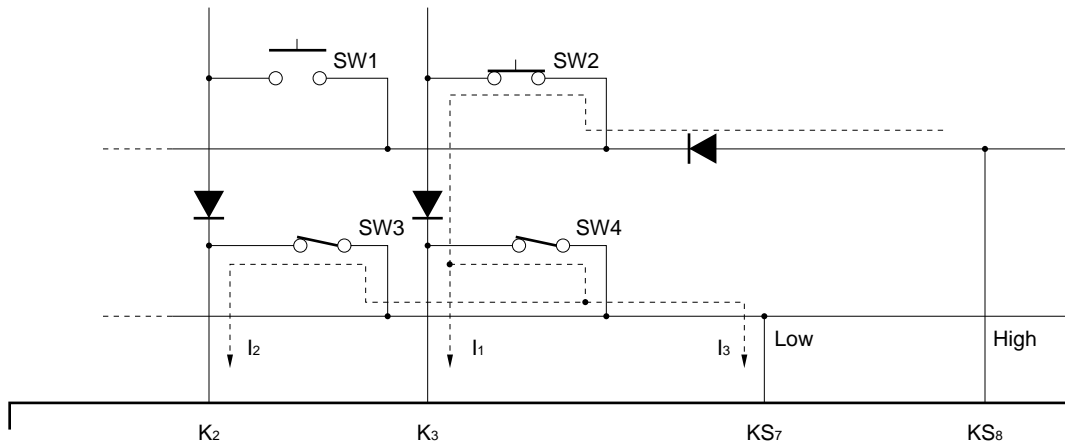
An example of a circuit where diode C is not used is shown below.

Suppose switches SW2, SW3, and SW4 are ON, and KS<sub>8</sub> outputs a high level, as shown below.

If diode C is missing, currents  $I_1$ ,  $I_2$ , and  $I_3$  flow as indicated by the dotted lines.

Consequently, a high level is input to K<sub>2</sub> because of  $I_2$  despite the fact that switch SW1 is OFF, and it is judged that SW1 is ON.

Moreover, KS<sub>8</sub> cannot output a high level correctly because of  $I_3$ .



## 20.6.2 Reading alternate switches and diode switches

Here is a program example.

**Example** To read statuses of alternate and diode switches of LCD<sub>15</sub>/KS<sub>15</sub>/PYA<sub>15</sub> to LCD<sub>8</sub>/KS<sub>8</sub>/PYA<sub>8</sub> pins to addresses 20H to 27H of BANK0 of data memory

```

KS8      NIBBLE8 0.20H
KEY_IN   MEM      0.73H           ; POD port register

KEY_LOAD:
        CLR1     PYASEL           ; Sets LCD15/KS15/PYA15 to LCD8/KS8/PYA8 pins
                                   ; to LCD segment
        SET2     LCDEN, KSEN       ; LCD segment and key source signal output
        MOV      DBF3, #0000B      ; Sets key source data
        MOV      DBF2, #0001B      ; Outputs low level from KS8
        MOV      DBF1, #0000B
        MOV      DBF0, #0000B
        MOV      IXM, #0000B
        MOV      IXL, #0000B
        MOV      RPH, #0000B
        MOV      RPL, #0000B

KSCAN:
        PUT      KSR, DBF          ; Outputs signal of key source data

LOOP:
        SKF1     KEYJ              ; Determines if key input is latched
        BR       KCHECK

        Processing A              ; Waits until key input is latched

        BR       LOOP

KCHECK:
        MOV      RPL#.DM.KEY_IN SHR 3 AND 0EH
        SET1     IXE
        ST       KS8, KEY_IN       ; Stores key input data to data memory
        CLR1     IXE
        MOV      RPL, #0000B
        INC      IX
        ADD      DBF2, DBF2         ; Updates value of key source data and
        ADD      DBF3, DBF3         ; scans key again
        SKT1     CY                ; Determines if all key source lines are input
        BR       KSCAN

KEY_END:                                ; End of input

```

### 20.6.3 Inputting momentary switch by binary search

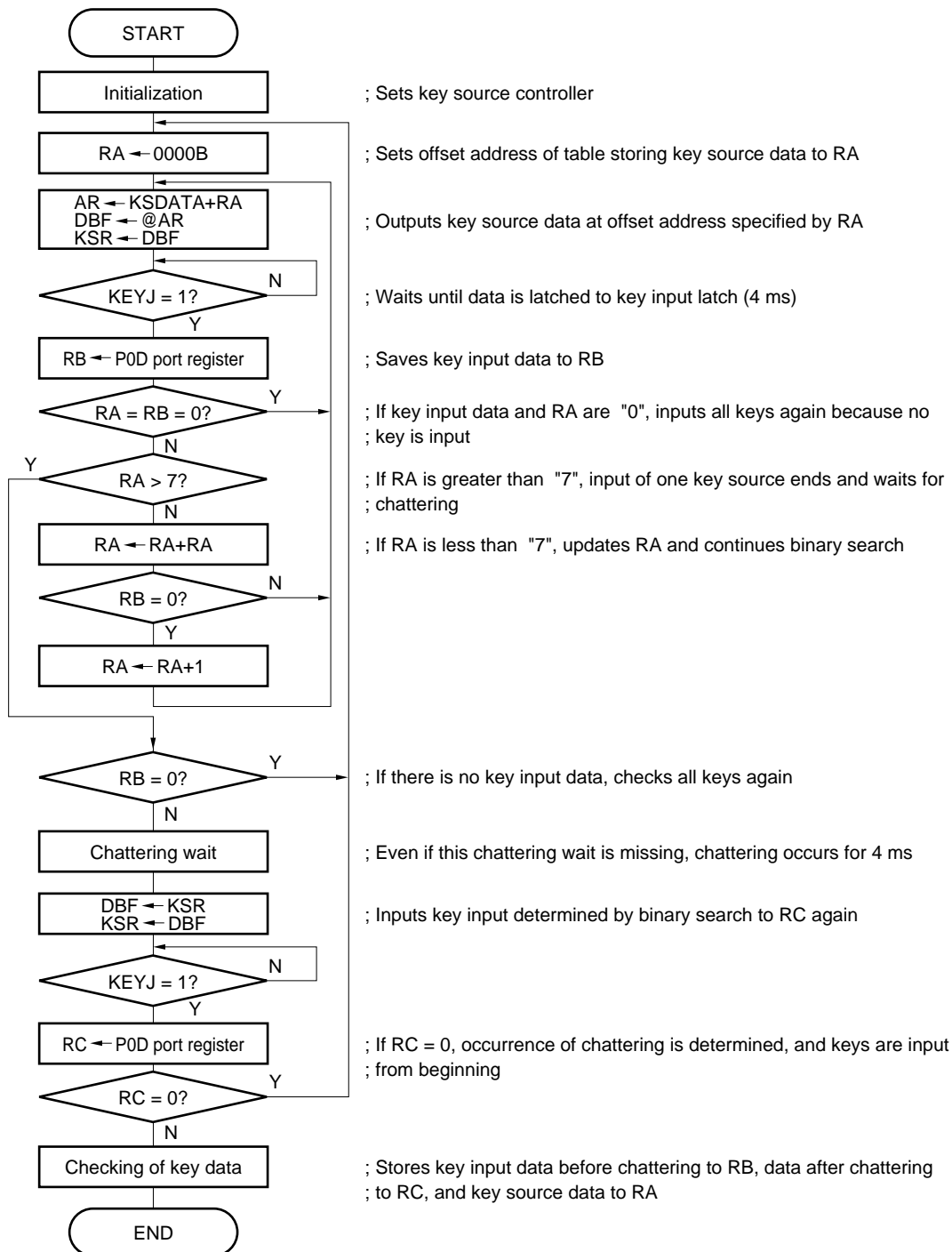
The key source controller/decoder requires 4 ms to input the key of one key source signal line.

To input the keys of 16 key source signals, therefore, it takes 64 ms.

It is therefore convenient if the binary search method explained in (1) and (2) below is used.

#### (1) Flowchart

When  $KS_7$  to  $KS_0$  are used as key source signals of momentary switch



## Example of table data for binary search

	Table Data (Key Source Data)															
	b <sub>15</sub>	b <sub>14</sub>	b <sub>13</sub>	b <sub>12</sub>	b <sub>11</sub>	b <sub>10</sub>	b <sub>9</sub>	b <sub>8</sub>	b <sub>7</sub>	b <sub>6</sub>	b <sub>5</sub>	b <sub>4</sub>	b <sub>3</sub>	b <sub>2</sub>	b <sub>1</sub>	b <sub>0</sub>
0000B																
0001B																
0010B																
0011B																
0100B																
0101B																
0110B																
0111B																
1000B																
1001B																
1010B																
1011B																
1100B																
1101B																
1110B																
1111B																

## (2) Program example

```

RA      MEM    0.1AH          ; General-purpose work register
RB      MEM    0.1BH          ; General-purpose work register
RC      MEM    0.1CH          ; General-purpose work register

```

```

KEY_IN MEM    0.73H          ; P0D port register

```

```

KSDATA:
;      KKKKKKKKKKKKKKKKK
;      SSSSSSSSSSSSSSSSS
;      1111119876543210
;      543210
DW     0000000011111111B    ; RA = 0
DW     0000000011110000B    ; RA = 1
DW     0000000000001100B    ; RA = 2
DW     0000000000110000B    ; RA = 3
DW     000000000000010B     ; RA = 4
DW     0000000000001000B    ; RA = 5
DW     0000000000100000B    ; RA = 6
DW     0000000010000000B    ; RA = 7
DW     000000000000001B     ; RA = 8
DW     000000000000010B     ; RA = 9
DW     000000000000100B     ; RA = 10
DW     000000000001000B     ; RA = 11
DW     000000000010000B     ; RA = 12
DW     000000000100000B     ; RA = 13
DW     000000001000000B     ; RA = 14
DW     000000010000000B     ; RA = 15

```

```

KEY_LOAD:
CLR1   PYASEL                ; Sets LCD15/KS15/PYA15 to LCD8/KS8/PYA8 pins
                                ; to LCD segment
SET2    LCDEN, KSEN           ; LCD segment and key source signal output

```

```

START:
    MOV    RA, #0000B

KSCAN:
    MOV    AR3, #.DL.KSDATA SHR 0CH AND 0FH
    MOV    AR2, #.DL.KSDATA SHR 8 AND 0FH
    MOV    AR1, #.DL.KSDATA SHR 4 AND 0FH
    MOV    AR0, #.DL.KSDATA AND 0FH
    MOV    RPL, #.DL.AR0 SHR 3 AND 0EH
    ADD    AR0, RA
    ADDC   AR1, #0
    ADDC   AR2, #0
    ADDC   AR3, #0
    MOV    RPL, #0
    MOVT   DBF, @AR                ; Reads table data

    PUT    KSR, DBF                ; Outputs signal of key source data

LOOP1:
    SKF1   KEYJ                    ; Determines if key input is latched
    BR     KCHECK

|              |
|--------------|
| Processing A |
|--------------|


    ; Waits until key input is latched

    BR     LOOP1

KCHECK:
    MOV    PRL, #.DM.RB SHR 3 AND 0EH
    LD      RB, KEY_IN              ; Stores key input data to RB
    SKNE   RA, #0000B              ; All keys are checked?
    SKE     RB, #0000B
    BR     Key input
    BR     START                    ; There is no key input

Key input:
    SKLT   RA, #1000B              ; Key sources are narrowed down to one?
    BR     LASTCHK
    ; If not, updates value of RA, and scans keys again

    ADD    RA, RA
    SKE     RB, #0000B
    ADD    RA, #0001B
    BR     KSCAN

LASTCHK:
    MOV    RPL, #0
    SKNE   RB, #0000B              ; Key input to one key source?
    BR     START                    ; If not, it is determined that chattering occurs

|                 |
|-----------------|
| Chattering wait |
|-----------------|


```

```
LOOP2:
    SKF1    KEYJ                ; Determines if key input is latched
    BR      KEYDEC

|              |
|--------------|
| Processing B |
|--------------|


    ; Waits until key input is latched

    BR      LOOP2
KEYDEC:
    MOV     RPL, #.DM.RC SHR 3 AND 0EH
    LD      RC, KEY_IN          ; Stores key input data to latch
    SET2    CAP, Z              ; Compares key input data after chattering with key input
    SUB     RC, RB              ; data before chattering wait
    SKT1    Z
    BR      START              ; If data differ

KEY_END:
    ; Stores key source data to RA, key input data before
    ; chattering to RB, and key input data after chattering to RC
```

## 20.7 Status on Reset

### 20.7.1 On power-ON reset

The LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins are specified as the LCD segment signal output pins and output a low level (display OFF). A low level is output as the key source signal.

### 20.7.2 On execution of clock stop instruction

The LCD<sub>19</sub>/P2H<sub>0</sub> to LCD<sub>0</sub>/KS<sub>0</sub>/PYA<sub>0</sub> pins are specified as the LCD segment signal output pins and output a low level (display OFF). A low level is output as the key source signal.

### 20.7.3 On CE reset

The output data is retained as is if the key source signal is being output.

### 20.7.4 In halt status

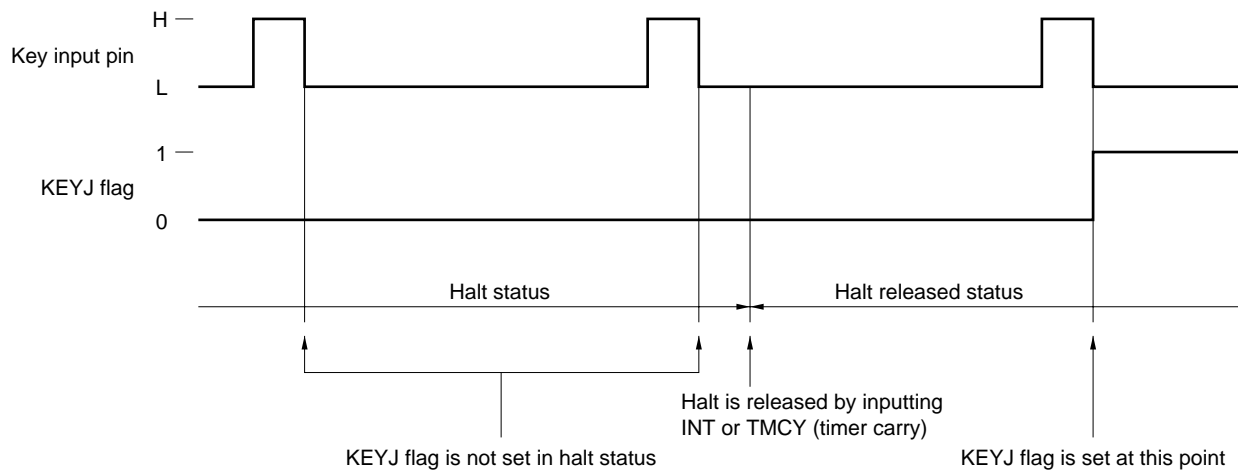
The output data is retained as is if the key source signal is being output.

If key input is specified as a halt status releasing condition, the halt status is released when a high level is input to the P0D<sub>3</sub>/K<sub>3</sub> to P0D<sub>0</sub>/K<sub>0</sub> pins.

If the key source controller is used, however, the halt status is released only by a high level that is input within 220  $\mu$ s during which the key source data is output.

For an explanation of how to release the halt status by key input, refer to **21.4 Halt Function**.

**Figure 20-11. KEYJ Flag Status in Halt Status**





## 21. STANDBY

The standby function is used to reduce the current consumption of the device during back up.

### 21.1 Configuration of Standby Block

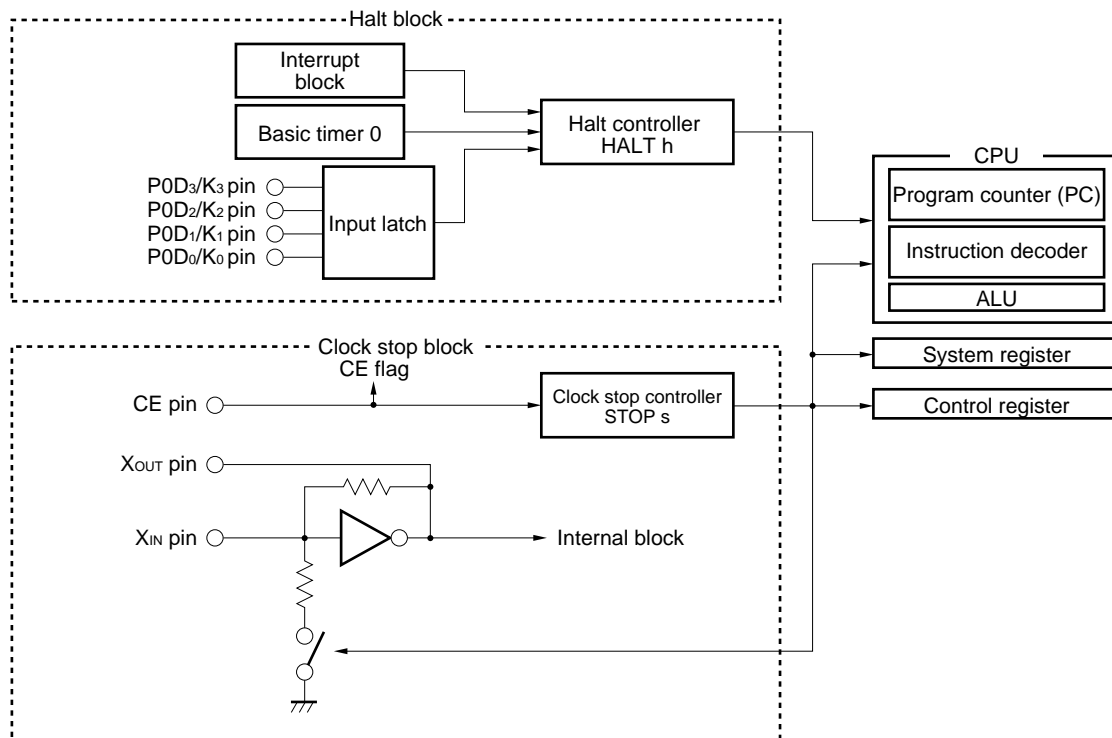
Figure 21-1 shows the configuration of the standby block.

As shown in the figure, the standby block is divided into two blocks: halt control block and clock stop control block.

The halt control block consists of a halt controller, interrupt control block, timer carry, and key input pins P0D<sub>3</sub>/K<sub>3</sub> pin to P0D<sub>0</sub>/K<sub>0</sub> pin, and controls the operation of the CPU (program counter, instruction decoder, and ALU block).

The clock stop control block controls the 4.5 MHz crystal oscillator, CPU, system register, and control registers, by using the clock stop controller.

**Figure 21-1. Configuration of Standby Block**



**Remark** CE (Bit 0 of the CE pin level judge register; refer to **21.3.5 Configuration and function of CE pin level judge register**)

Detects the CE pin status.

## 21.2 Standby Function

The standby function reduces the current consumption of the device by stopping some or all its operations.

The standby function can be used in two modes: halt and clock stop.

The halt mode is to reduce the current consumption of the device by executing a dedicated instruction "HALT h" and stopping the operation of the CPU.

The clock stop mode is to reduce the current consumption of the device by executing a dedicated instruction "STOP s" and stopping the 4.5 MHz crystal oscillator.

In addition to the halt and clock stop modes, the operation mode of the device can be also set by the CE pin.

The CE pin is used to control the operation of the PLL frequency synthesizer and reset the device, and can be said to be a type of the standby function in that it controls the operation of the PLL frequency synthesizer.

The following section 21.3 explains how to set the operation mode of the device by using the CE pin.

Sections 21.4 and 21.5 explain the halt and clock stop modes respectively.

## 21.3 Selecting Device Operation Mode with CE Pin

The CE pin controls the following functions (1) through (3) by using the level and rising edge of an externally input signal.

- (1) Controls operation of PLL frequency synthesizer
- (2) Enables or disables clock stop instruction
- (3) Resets device

### 21.3.1 Controlling operation of PLL frequency synthesizer

The PLL frequency synthesizer can operate only when the CE pin is high.

The PLL frequency synthesizer is automatically disabled when the CE pin is low.

At this time, the VCOH and VCOL pins are internally pulled down, and the EO pin is floated.

The PLL frequency synthesizer can be disabled by the PLL reference clock select register at any time when the CE pin is high.

### 21.3.2 Enabling and disabling clock stop instruction

The clock stop instruction "STOP s" is enabled only when the CE pin is low.

The STOP s instruction is executed as a no-operation (NOP) instruction if it is executed when the CE pin is high.

### 21.3.3 Resetting device

The device can be reset (CE reset) by raising the CE pin.

The device can also be reset through power application (power-on reset).

For details, refer to **22. RESET**.

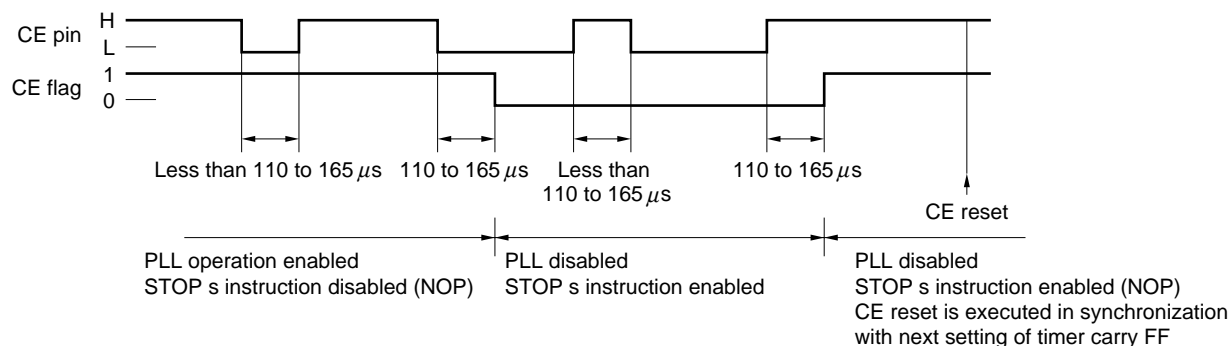
### 21.3.4 Inputting signal to CE pin

The CE pin does not accept a low or high level of less than 110 to 165  $\mu\text{s}$  to prevent malfunctioning due to noise.

The level of the signal input to the CE pin can be detected by using the CE flag of the CE pin level judge register (RF address 07H).

Figure 21-2 shows the relationship between the input signal and CE flag.

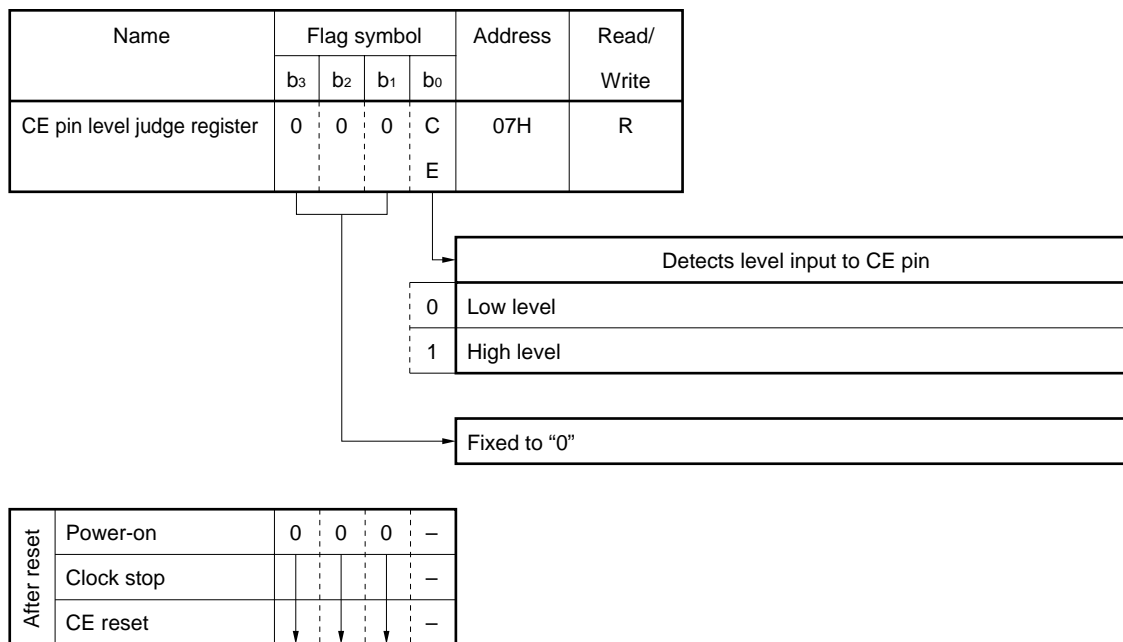
**Figure 21-2. Relationship Between Signal Input to CE Pin and CE Flag**



### 21.3.5 Configuration and function of CE pin level judge register

The CE pin level judge register detects the level of the signal input to the CE pin.

The configuration and function of this register are illustrated below.



— : Determined depending on pin status

The CE flag is not affected by a low or high level of less than 110 to 165  $\mu\text{s}$ .

## 21.4 Halt Function

The halt function stops the operation clock of the CPU by executing the HALT h instruction.

When the HALT h instruction is executed, the program stops at the HALT h instruction, until the halt status is released later.

Therefore, the current consumption of the device can be reduced in the halt status by the operating current of the CPU.

The halt status can be released by key input, timer carry, or interrupt.

The releasing condition of the key input, timer carry, and interrupt is specified by the operand “h” of the HALT h instruction.

The HALT h instruction is valid regardless of the input level of the CE pin.

The following subsections 21.4.1 through 21.4.6 explain the halt status, halt release condition, and each halt release condition.

### 21.4.1 Halt status

All the operations of the CPU are stopped in the halt status.

In other words, program execution is stopped at the HALT h instruction.

However, the peripheral hardware units continue the operations set before the HALT h instruction is executed.

For the operations of the peripheral hardware units, refer to **21.6 Device Operations in Halt and Clock Stop Status**.

### 21.4.2 Halt release condition

Figure 21-3 shows the halt release conditions.

As shown in this figure, the halt release conditions are set by 4-bit data specified by operand “h” of the HALT h instruction.

The halt status is released when the condition specified as “1” by operand “h” is satisfied.

When the halt status is released, the execution starts from the instruction next to the HALT h instruction.

If two or more release conditions are specified, and if any one of the specified conditions is satisfied, the halt condition is released.

If the device is reset (power-on reset or CE reset), the halt status is released, and each reset operation is performed.

If 0000B is set as the halt release condition “h”, no release condition is set.

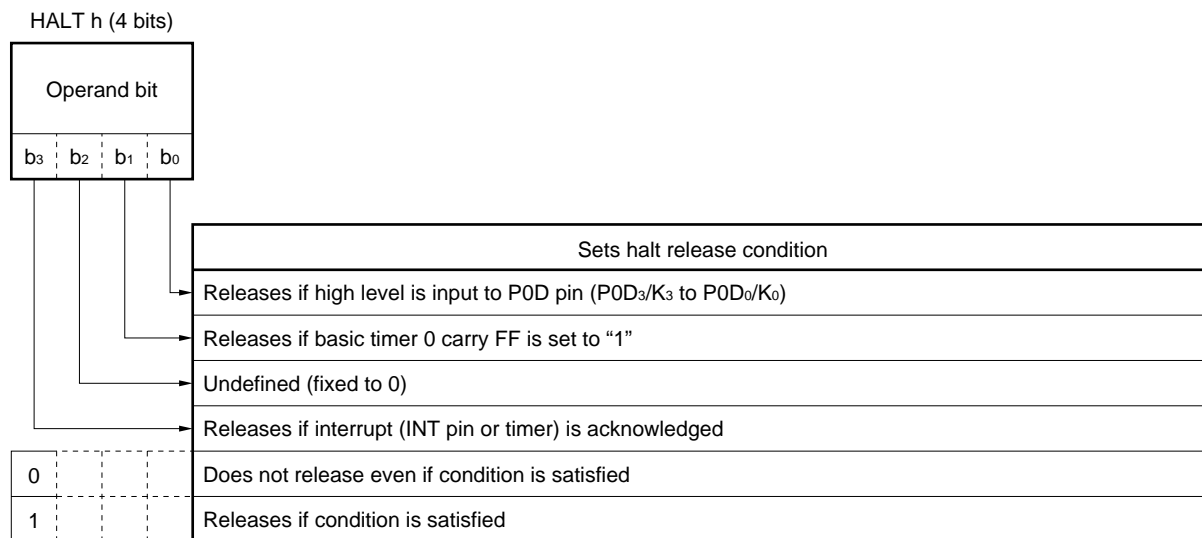
At this time, the halt status is released if the device is reset (power-on reset or CE reset).

The following subsections 21.4.3 through 21.4.5 explains halt release conditions set by key input, basic timer 0, and interrupt.

21.4.6 shows an example when two or more release conditions are specified.

★

**Figure 21-3. Halt Release Condition**



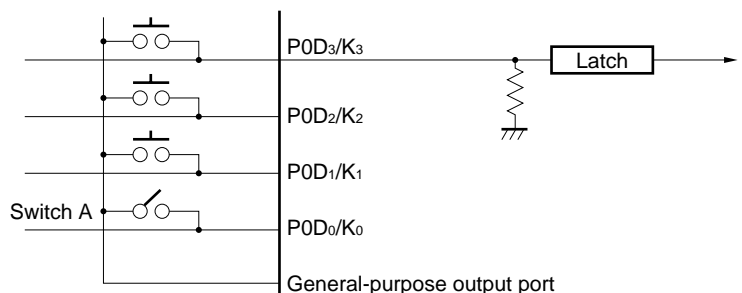
### 21.4.3 Releasing halt status by key input

Releasing the halt status by key input is specified by the HALT 0001B instruction.

If releasing the halt status by key input is specified, the halt status is released when a high level is input to any of the four pins P0D<sub>0</sub>/K<sub>0</sub> to P0D<sub>3</sub>/K<sub>3</sub>.

The following paragraphs (1) through (3) explain the points to be noted when using a general-purpose output port for a key source signal and when multiplexing LCD segment signal outputs with key source signal outputs.

#### (1) Notes on using general-purpose output port for key source signal



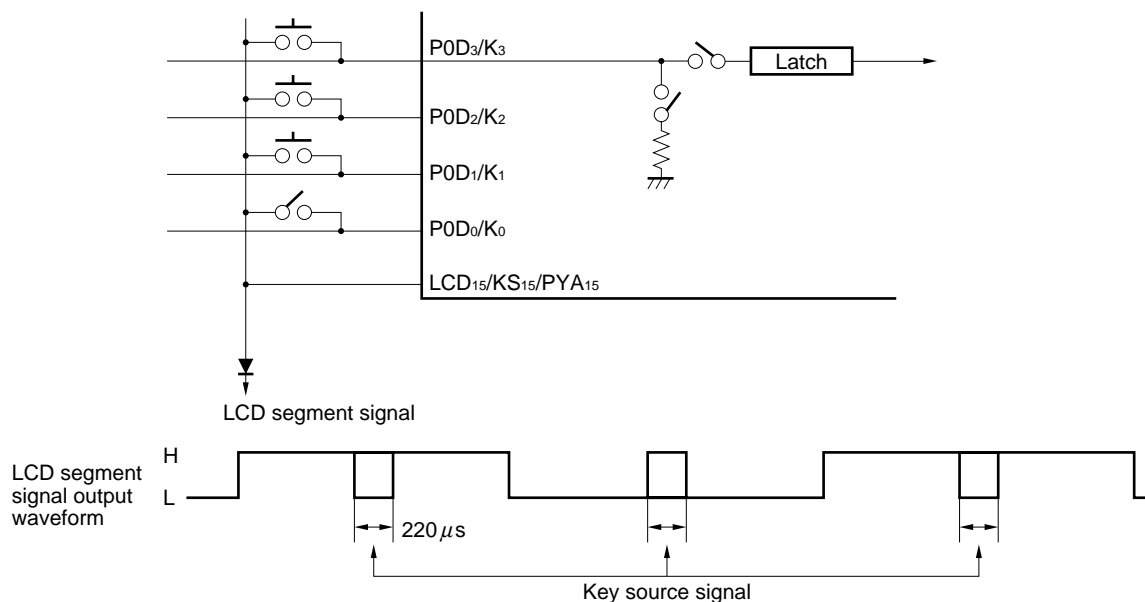
The HALT 0001B instruction is executed after a general-purpose output port for key source signal goes high.

If an alternate switch such as switch A in the above figure is used at this time, a high level is always applied to the P0D<sub>0</sub>/K<sub>0</sub> pin while switch A is closed, and the halt status is immediately released.

Therefore, care must be exercised in using the alternate switch.

When using a general-purpose output port for key source signal, reset the KSEN flag of the LCD mode select register (RF address 10H) to 0.

At this time, the P0D<sub>0</sub>/K<sub>0</sub> to P0D<sub>3</sub>/K<sub>3</sub> pins are automatically pulled down.

**(2) Notes on multiplexing LCD segment signal and key source signal outputs**

Execute the HALT 0001B instruction after setting key source signal output data.

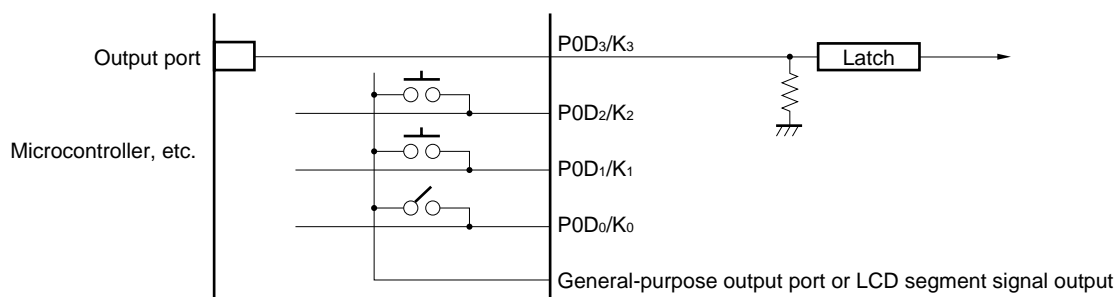
At this time, the halt status is not released even if a high level of the LCD segment signal is input to the pin whose key source signal output data is "0".

To multiplex an LCD segment signal output with a key source signal output, set the KSEN flag of the LCD mode select register to 1.

The key source signal data (setting the pin that outputs a key source) is set by the key source data register (KSR: peripheral address 42H) via the data buffer.

The internal key latch circuit when an LCD segment signal output is multiplexed with a key source signal output latches data only while the key source signal is output, and is disconnected from the external source while the LCD segment signal is output.

The internal pull-down resistor is ON only when the key source signal is output.

**(3) When releasing from halt status using other microcontrollers**

The P0D<sub>0</sub>/K<sub>0</sub> to P0D<sub>3</sub>/K<sub>3</sub> pins can also be used as general-purpose input port pins with pull-down resistors.

Therefore, the halt status can also be released by using other microcontrollers as shown above.

#### 21.4.4 Releasing halt status by basic timer 0

Releasing the halt status by basic timer 0 is set by the HALT 0010B instruction.

When the release of the halt status is set by basic timer 0, the halt status is released as soon as the basic timer 0 carry FF has been set to 1.

The basic timer 0 carry FF corresponds to the BTM0CY flag of the basic timer 0 carry FF judge register on a one-to-one basis, as explained in **12. TIMER**, and is set to 1 at fixed time intervals (1 ms, 5 ms, 100 ms, or 250 ms).

Therefore, the halt status can be released at fixed time intervals.

#### Example

```

M1      MEM      0.10H      ; 1-second counter
HLTTMR  DAT      0010B      ; Symbol definition
                INITFLG  NOT  BTM0CK1, BTM0CK0
                                ; Embedded macro
                                ; Sets basic timer 0 carry FF setting time to 250 ms

LOOP:
                HALT      HLTTMR      ; Sets release condition by basic timer 0 carry FF and
                                halt status
                SKT1      BTM0CY      ; Embedded macro
                BR        LOOP      ; Branches to LOOP if BTM0CY flag is not set
                ADD      M1, #0100B  ; Adds 0100B to contents of M1
                SKT1      CY         ; Embedded macro
                BR        LOOP      ; Executes processing A if carry occurs



Processing A



                BR        LOOP

```

In this example, the halt status is released every 250 ms and processing A is executed every 1 second.

#### 21.4.5 Releasing halt status by interrupt

Releasing the halt status by an interrupt is set by the HALT 1000B instruction.

If releasing the halt status by an interrupt is set, the halt status is released as soon as the interrupt has been acknowledged.

Four interrupt sources are available as explained in **11. INTERRUPTS**.

Therefore, the interrupt source to be used to release the halt status must be specified by program in advance.

So that the interrupt is acknowledged, all the interrupts must be enabled (by the EI instruction), each interrupt is enabled (by setting the corresponding interrupt enable flag), in addition that the interrupt request must be issued from each interrupt source.

Even if an interrupt request is issued, if that interrupt is not enabled, the interrupt is not acknowledged and the halt status is not released.

When the halt status has been released because the interrupt has been acknowledged, the program flow branches to the vector address of the interrupt.

If the RETI instruction is executed after the interrupt processing, the program flow returns to the instruction next to the HALT instruction.

Here is an example.



## Example

```

HLTINT  DAT      1000B  ; Symbol definition of halt condition
INTTM   DAT      0003H  ; Interrupt vector address symbol definition
INTPIN  DAT      0004H  ; Interrupt vector address symbol definition

START:                                     ; Program address 0000H
      BR        MAIN
ORG     INTTM                                     ; 12-bit timer interrupt vector address (0003H)
      BR        INTTIMER
ORG     INTPIN                                    ; INT pin interrupt vector address (0004H)

      Processing A                               ; Interrupt processing by INT pin

      BR        EI_RETI
INTTIMER:

      Processing B                               ; Interrupt processing by 12-bit timer

EI_RETI:
      EI
      RETI
MAIN:
      SET2      IPTM, IP0
                                   ; Embedded macro
      SET2      BTM1CK1, BTM1CK0
                                   ; Embedded macro
                                   ; Sets time interval of 12-bit timer to 1 ms
LOOP:

      Processing C                               ; Main routine processing

      EI                                     ; Enables all interrupts
      HALT      HLTINT ; Specifies releasing halt by interrupt
; <1>
      BR        LOOP

```

In this example, the halt status is released when the 12-bit timer interrupt is acknowledged, and processing B is executed. When the INT pin interrupt is acknowledged, processing A is executed.

Each time the halt status is released, processing C is executed.

If the INT pin interrupt request and 12-bit timer interrupt request are issued at the same time in the halt status, processing A of the INT pin, which has the higher hardware priority, is executed.

If "RETI" is executed after execution of processing A, execution restores to the BR LOOP instruction in <1>, but the BR LOOP instruction is not executed, and the 12-bit timer interrupt is immediately acknowledged.

If "RETI" is executed after processing B of the 12-bit timer interrupt has been executed, the BR LOOP instruction is executed.

**Caution** When executing the HALT instruction that will set the release condition where by the halt status is released by the setting of the interrupt request flag (IRQ<sub>xxx</sub>) when the interrupt enable flag (IP<sub>xxx</sub>) is set, describe a NOP instruction immediately before the HALT instruction. If a NOP instruction is described immediately before the HALT instruction, a time of one instruction is generated in between the IRQ<sub>xxx</sub> manipulation instruction and HALT instruction. In the case of the CLR1 IRQ<sub>xxx</sub> instruction, for example, clearing IRQ<sub>xxx</sub> is correctly reflected on the HALT instruction (refer to Example 1 below). If a NOP instruction is not described immediately before the HALT instruction, the CLR1 IRQ<sub>xxx</sub> instruction is not correctly reflected on the HALT instruction, and the HALT mode is not set (refer to Example 2 below).

**Example 1. Program that correctly executes HALT instruction**

```

      :
      :
      : ; Sets IRQxxx
      :
CLR1  IRQxxx
NOP      ; Describes NOP instruction immediately before
          ; HALT instruction
          ; (clearing IRQxxx is correctly reflected on HALT
          ; instruction)
HALT  1000B ; Correctly executes HALT instruction
          ; (HALT mode is set)
      :
      :

```

**2. Program that does not set HALT mode**

```

      :
      :
      : ; Sets IQRxxx
      :
CLR1  IRQxxx ; Clearing IRQxxx is not reflected on HALT instruction
          ; (but on instruction next to HALT)
HALT  1000B ; HALT instruction is ignored (HALT mode is not set)
      :
      :

```

#### 21.4.6 If two or more release conditions are simultaneously set

If two or more release conditions are simultaneously set, and if even one of the conditions is satisfied, the halt status is released.

The method to identify the release condition that is satisfied when two or more release conditions are specified is shown below.

##### Example

```

          HLTINT    DAT    1000B
          HLTTMR    DAT    0010B
          HLTKEY    DAT    0001B
          INTPIN    DAT    0004H          ; INT pin interrupt vector address symbol
                                          ; definition

START:
          BR        MAIN
ORG      INTPIN
          Processing A                    ; INT pin interrupt processing
          EI
          RETI
TMRUP    ; Basic timer 0 processing
          Processing B
          RET
KEYDEC:   ; Key input processing
          Processing C
          RET
MAIN:
          MOVT      DBF, @AR              ; Sets key source output data (table reference)
                                          ; to key source data register (KSR)
          PUT       KSR, DBF
          SET2      KSEN, LCDEN           ; Embedded macro
                                          ; Multiplexes LCD segment signal output with
                                          ; key source signal output
          SET2      BTM0CK1, BTM0CK0     ; Embedded macro
                                          ; Sets basic timer 0 carry FF setting time to 1 ms
          SET1      IP                   ; Embedded macro
                                          ; Enables INT pin interrupt
          EI
LOOP:
          HALT      HLTINT OR HLTTMR OR HLTKEY ; Specifies interrupt, basic timer 0, and key input
                                          ; as halt release conditions
          SKF1      BTM0CY               ; Embedded macro
                                          ; Detects BTM0CY flag
          CALL      TMRUP                ; Basic timer 0 processing if set to 1
          SKF1      KEYJ                 ; Embedded macro
                                          ; Detects key input latch Note
          CALL      KEYDEC               ; Key input processing if latched
          BR        LOOP

```

**Note** If the target key source output data is not output, the KEYJ flag is not set (1).

## 21.5 Clock Stop Function

The clock stop function stops the 4.5 MHz crystal oscillator by executing the STOP s instruction (clock stop status).

Therefore, the current consumption of the device is decreased to the minimum value of 10  $\mu$ A.

Specify "0000B" as operand "s" of the STOP s instruction.

The STOP s instruction is valid only while the CE pin is low.

It is executed as a no-operation (NOP) instruction even when executed while the CE pin is high.

In other words, the STOP s instruction must be executed while the CE pin is low.

The clock stop status is released by raising the level of the CE pin from low to high (CE reset).

The following subsections 21.5.1 through 21.5.3 explain the clock stop status, how to release the clock stop status, and notes on using the clock stop instruction.

### 21.5.1 Clock stop status

Because the crystal oscillator is stopped in the clock stop status, all the device operations, such as those of the CPU and peripheral hardware, are stopped.

For the operations of the CPU and peripheral hardware, refer to **21.6 Device Operations in Halt and Clock Stop Status**.

The power failure detector does not operate in the clock stop status even if the supply voltage  $V_{DD}$  of the device is lowered to 2.3 V. Therefore, the data memory can be backed up at a low voltage. For the details on the power failure detector, refer to **22. RESET**.

### 21.5.2 Releasing clock stop status

The clock stop status is released either by raising the level of the CE pin from low to high (CE reset), or by lowering the supply voltage  $V_{DD}$  of the device to 2.3 V or less once, and then increasing it to 3.5 V (power-on reset).

Figures 21-4 and 21-5 show how the clock stop is released on CE reset and power-on reset respectively.

If the clock stop status is released by power-on reset, the power failure detector operates.

For the details on power-on reset, refer to **22.4 Power-on Reset**.

Figure 21-4. Releasing Clock Stop Status by CE Reset

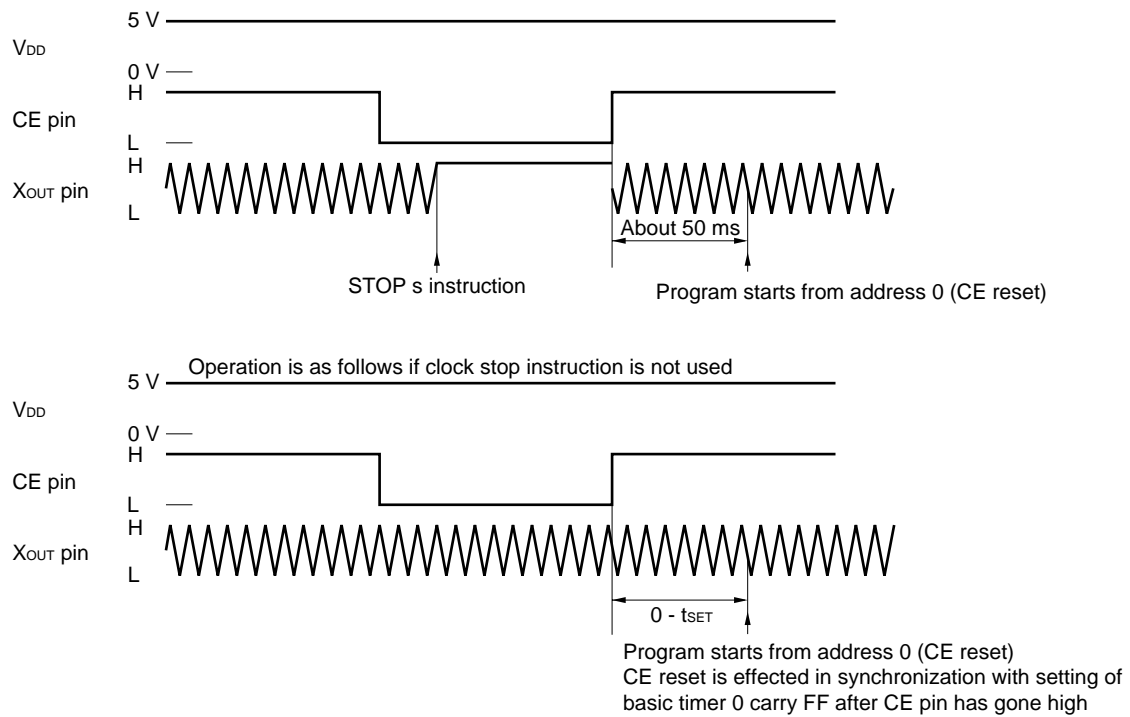
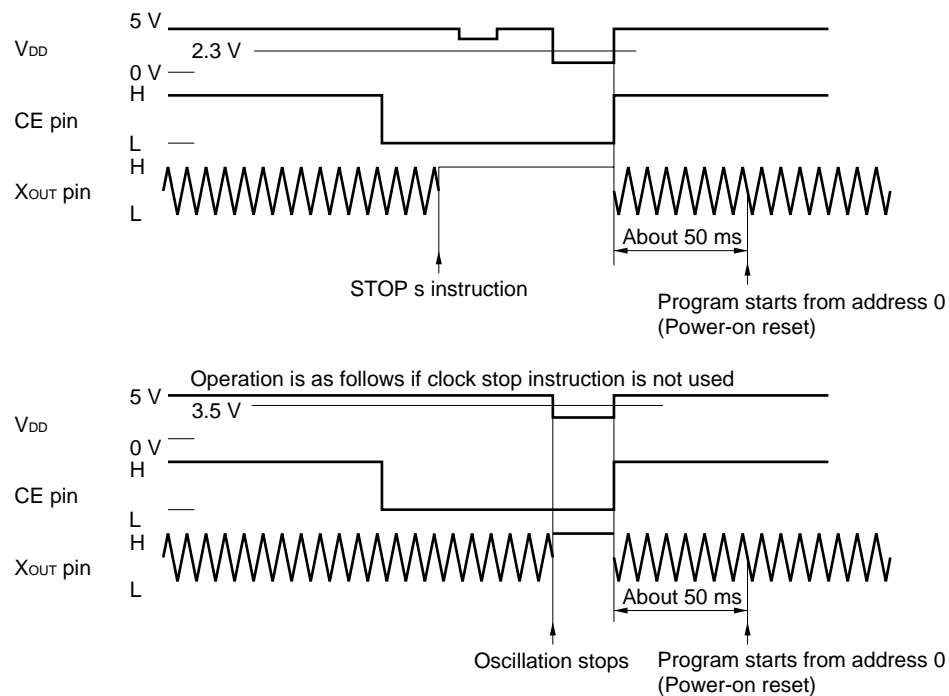


Figure 21-5. Releasing Clock Stop Status by Power-on reset



### 21.5.3 Notes on using clock stop instruction

The clock stop (STOP s) instruction is valid only while the CE pin is low.

Therefore, processing to be performed if the CE pin happens to be high must be taken into consideration.

Take the following program as an example.

#### Example

```

        XTAL    DAT    0000B    ; Symbol definition of clock stop condition
CEJDG:
    ; <1>
        SKF1    CE                ; Embedded macro
                                   ; Detects input level of CE pin
        BR      MAIN              ; Branches to main processing if CE = high
        [Processing A]              ; Processing of CE = low
    ; <2>
        STOP    XTAL              ; Clock stop
    ; <3>
        BR      $ - 1
MAIN:
        [Main processing]
        BR      CEJDG

```

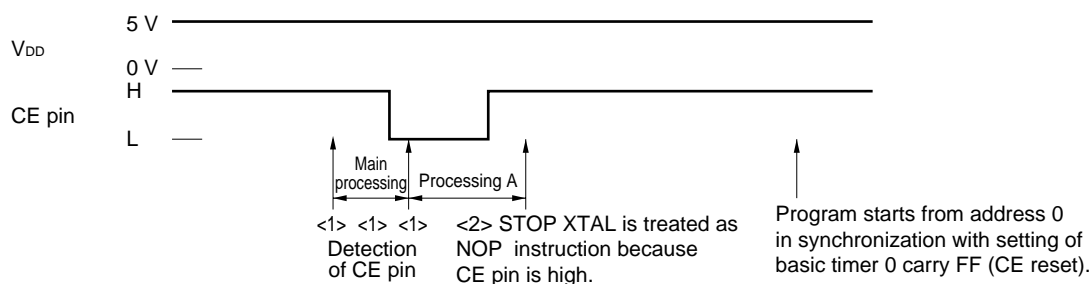
In the above example, the status of the CE pin is detected in <1>. If the CE pin is low, processing A is performed and then the clock stop instruction “STOP XTAL” in <2> is executed.

If the CE pin goes high while the STOP XTAL instruction in <2> is executed, however, the STOP XTAL instruction is treated as a no-operation (NOP) instruction.

Should branch instruction “BR\$-1” in <3> be missing at this time, the program would execute the main processing, causing malfunctioning.

Therefore, either a branch instruction must be inserted as in <3>, or the program must be designed in the manner that malfunctioning does not occur even if the main processing is executed.

If a branch instruction is used as in <3>, CE reset is executed in synchronization with the next setting of the timer carry FF even while the CE pin is high.



## 21.6 Device Operations in Halt and Clock Stop Status

Table 21-1 shows the operations of the CPU and peripheral hardware in the halt status and clock stop status.

As shown in this table, all the peripheral hardware units continue the normal operation in the halt status, except that instruction execution is stopped.

All the peripheral hardware units stop operation in the clock stop status.

The control registers that control the operations of the peripheral hardware units operate normally in the halt status (i.e., are not initialized), but are initialized to specific values in the clock stop status (as soon as the STOP instruction has been executed).

To put in another way, the peripheral hardware units continue the operations set by the control registers in the halt status, and operate in accordance with the control registers that are initialized to specific values in the clock stop status.

For the values to which the control registers are initialized, refer to **8. REGISTER FILE (RF)**.

The following shows an example.

**Example** When the P0A<sub>0</sub>/SI<sub>1</sub> pin of port 0A is specified as an output port pin, and the P0A<sub>1</sub>/SO<sub>1</sub> and P0A<sub>2</sub>/ $\overline{\text{SCK}}_1$  pins are used for the serial interface

```

HLTINT  DAT    1000B
XTAL     DAT    0000B
INITFLG  P0ABI02, P0ABI01, P0ABI00
; <1>
SET3     P0A2, P0A1, P0A0
; <2>
INITFLG  SI01HIZ, SI01CK1, SI01CK0
CLR1     IRQSI01
SET1     IPSI01
EI
; <3>
SET1     SI01TS
; <4>
HALT     HLTINT
; <5>
STOP     XTAL

```

In the above example, the P0A<sub>2</sub> through P0A<sub>0</sub> pins output a high level in <1>, the condition of serial interface 1 is set in <2>, and serial communication is started in <3>.

When the HALT instruction is executed in <4>, serial communication continues and the halt status is released when the interrupt by serial interface 1 is acknowledged.

If the STOP instruction in <5> is executed instead of the HALT instruction in <4>, the contents of all the control registers set in <1>, <2>, and <3> are initialized. Consequently, serial communication is stopped, and all the pins of port 0A are set in the general-purpose input port mode.

Table 21-1. Device Operations in Halt Status and Clock Stop Status

Peripheral Hardware	Status			
	CE Pin = High		CE Pin = Low	
	Halt	Clock Stop	Halt	Clock Stop
Program counter	Stops at address of HALT instruction	STOP instruction is invalid (NOP)	Stops at address of HALT instruction	Initialized to 0000H and stops
System register	Retained		Retained	Initialized <sup>Note</sup>
Peripheral register	Retained		Retained	Retained
Control register	Retained		Retained	Initialized <sup>Note</sup>
Timer	Normal operation		Normal operation	Stops operation
PLL frequency synthesizer	Normal operation		Disabled	Disabled
A/D converter	Normal operation		Normal operation	Stops operation
D/A converter	Normal operation		Normal operation	Stops operation
BEEP output	Normal operation		Normal operation	Stops operation
Serial interface	Normal operation		Normal operation	Stops operation
Frequency counter	Normal operation		Normal operation	Stops operation
LCD controller/driver	Normal operation		Normal operation	Stops operation
Key source controller/decoder	Normal operation		Normal operation	Stops operation
General-purpose I/O port	Normal operation		Normal operation	Input port
General-purpose input port	Normal operation		Normal operation	Input port
General-purpose output port	Normal operation		Normal operation	Retained

**Note** For the values to which these registers are initialized, refer to **5. SYSTEM REGISTER (SYSREG)** and **8. REGISTER FILE (RF)**.

### 21.7 Notes on Processing Each Pin in Halt and Clock Stop Status

The halt status is used to reduce the current consumption, when only the watch operates, for example.

The clock stop function is used to reduce the current consumption when only the contents of the data memory are to be retained.

Therefore, the current consumption must be reduced as much as possible in the halt and clock stop statuses.

At this time, the current consumption may increase depending on the status of each pin, and therefore the points shown in Table 21-2 must be noted.



Table 21-2. Notes on Status of Each Pin in Halt and Clock Stop Statuses (1/2)

Pin Function		Pin Symbol	Status of Each Pin and Notes on Processing	
			Halt	Clock Stop
General-purpose I/O port	Port 0A	P0A2/ $\overline{\text{SCK}}_1$ P0A1/SO <sub>1</sub> P0A0/SI <sub>1</sub>	<p>Previous status before halt status is set is retained as is.</p> <p>(1) In output mode</p> <p>Current consumption increases if these pins are externally pulled down while they output high level, or externally pulled up while they output low level.</p> <p>Pay attention to N-ch open-drain output pins (P0C<sub>3</sub> to P0C<sub>0</sub>/PWM<sub>0</sub>).</p> <p>(2) In input mode (except P0B<sub>3</sub>/FCG<sub>1</sub>, P0B<sub>2</sub>/FCG<sub>0</sub>, P1B<sub>3</sub>/FMIFC, P1B<sub>2</sub>/AMIFC)</p> <p>Current consumption increases due to noise if these pins are floated.</p>	<p>All these pins are set in general-purpose input port mode.</p> <p>All input ports, except port 1D (P1D<sub>3</sub> to P1D<sub>0</sub>), are designed to prevent increase in current consumption due to noise even if they are externally floated. Port 1D (P1D<sub>3</sub> to P1D<sub>0</sub>) must be externally pulled down or up so that current consumption does not increase due to noise.</p> <p>Port 0D (P0D<sub>3</sub>/K<sub>3</sub> to P0D<sub>0</sub>/K<sub>0</sub>) is internally pulled down.</p>
	Port 0B	P0B <sub>3</sub> /FCG <sub>1</sub> P0B <sub>2</sub> /FCG <sub>0</sub> P0B <sub>1</sub> /BEEP <sub>1</sub> P0B <sub>0</sub> /BEEP <sub>0</sub>		
	Port 1A	P1A <sub>2</sub> P1A <sub>1</sub> P1A <sub>0</sub>		
	Port 1D	P1D <sub>3</sub> P1D <sub>2</sub> P1D <sub>1</sub> P1D <sub>0</sub>		
General-purpose input port	Port 0D	P0D <sub>3</sub> /K <sub>3</sub> P0D <sub>2</sub> /K <sub>2</sub> P0D <sub>1</sub> /K <sub>1</sub> P0D <sub>0</sub> /K <sub>0</sub>	<p>(3) Port 0D (P0D<sub>3</sub>/K<sub>3</sub> to P0D<sub>0</sub>/K<sub>0</sub>)</p> <p>Current consumption increases if these pins are externally pulled up because they have pull-down resistors.</p> <p>(4) P0B<sub>3</sub>/FCG<sub>1</sub>, P0B<sub>2</sub>/FCG<sub>0</sub>, P1B<sub>3</sub>/FMIFC, P1B<sub>2</sub>/AMIFC</p> <p>Current consumption increases when P0B<sub>3</sub>/FCG<sub>1</sub>, P0B<sub>2</sub>/FCG<sub>0</sub>, P1B<sub>3</sub>/FMIFC, P1B<sub>2</sub>/AMIFC pins are used for IF counter because internal amplifier operates.</p>	<p>These pins are set in general-purpose output port mode.</p> <p>Output contents are retained as is.</p> <p>Therefore, current consumption increases if these pins are externally pulled down while they output high level, or pulled up while they output low level.</p>
	Port 1B	P1B <sub>3</sub> /FMIFC P1B <sub>2</sub> /AMIFC P1B <sub>1</sub> /ADC <sub>1</sub> P1B <sub>0</sub> /ADC <sub>0</sub>		
General-purpose output port	Port 0C	P0C <sub>3</sub> P0C <sub>2</sub> P0C <sub>1</sub> /PWM <sub>1</sub> P0C <sub>0</sub> /PMW <sub>0</sub>	<p>Because IF counter is not automatically disabled even if CE pin goes low, it must be initialized by program as necessary.</p> <p>P0B<sub>3</sub>/FCG<sub>1</sub>, P0B<sub>2</sub>/FCG<sub>0</sub>, P1B<sub>3</sub>/FMIFC, P1B<sub>2</sub>/AMIFC are designed to prevent increase in current consumption due to noise even if they are set in general purpose input port mode and floated.</p>	
	Port 1C	P1C <sub>3</sub> P1C <sub>2</sub> P1C <sub>1</sub> P1C <sub>0</sub>		
Interrupt		INT	Current consumption increase due to external noise if this pin is floated.	

Table 21-2. Notes on Status of Each Pin in Halt and Clock Stop Statuses (2/2)

Pin Function	Pin Symbol	Status of Each Pin and Notes on Processing	
		Halt	Clock Stop
LCD segment	LCD <sub>19</sub> /P2H <sub>0</sub> LCD <sub>18</sub> /P2G <sub>0</sub> LCD <sub>17</sub> /P2F <sub>0</sub> LCD <sub>16</sub> /P2E <sub>0</sub> LCD <sub>15</sub> /KS <sub>15</sub> /PYA <sub>15</sub>   LCD <sub>0</sub> /KS <sub>0</sub> /PYA <sub>0</sub>	Same as above general-purpose output ports applies if these pins are used in general-purpose output port mode.  If they output key source signals, current consumption increases via port 0D (with pull-down resistor) if there is switch that is always ON such as transistor switch and if "1" is output as key source data.	All pins are set in LCD segment signal output mode and output low level (display off).
PLL frequency synthesizer	VCOL VCOH EO	Current consumption increases during PLL operation.  These pins are as follows when PLL is disabled.  VCOL and VCOH: Internally pulled down  EO: Floated  PLL is automatically disabled when CE pin goes low.	PLL is disabled.  These pins are as follows.  VCOL and VCOH: Internally pulled down  EO: Floated
Crystal oscillator	X <sub>IN</sub> X <sub>OUT</sub>	Current consumption changes due to oscillation waveform of crystal oscillator.  Current consumption decreases as oscillation amplitude increases.  Because oscillation amplitude is influenced by crystal resonator and load capacitor used, evaluation must be performed.	X <sub>IN</sub> pin is internally pulled down, and X <sub>OUT</sub> pin outputs high level.

## 22. RESET

The reset function is used to initialize the device operation.

### 22.1 Configuration of Reset Block

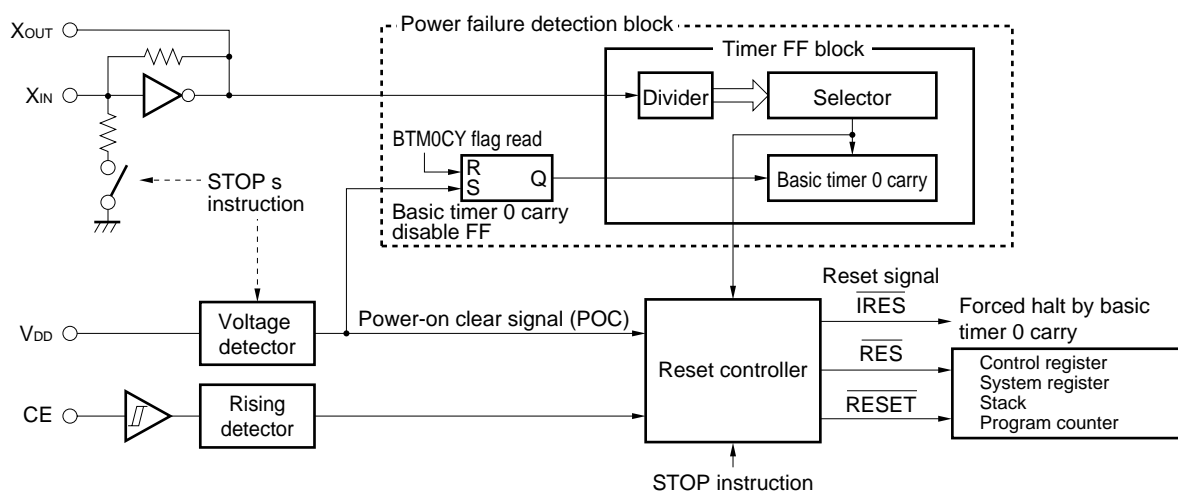
Figure 22-1 shows the configuration of the reset block.

The device is reset in two ways: by applying supply voltage  $V_{DD}$  (power-on reset or  $V_{DD}$  reset) and by using the CE pin (CE reset).

The power-on reset block consists of a voltage detector that detects a voltage input to the  $V_{DD}$  pin, a power failure detector, and a reset controller.

The CE reset block consists of a circuit that detects the rising of a signal input to the CE pin, and a reset controller.

**Figure 22-1. Configuration of Reset Block**



## 22.2 Reset Function

Power-on reset is effected when supply voltage  $V_{DD}$  rises from a specific level, and CE reset is effected when the CE pin goes high.

Power-on reset initializes the program counter, stack, system register, and control registers, and executes the program from address 0000H.

CE reset initializes the program counter, stack, system register, and some control registers, and executes the program from address 0000H.

The major differences between power-on reset and CE reset are the control registers that are initialized and the operation of the power failure detector that is explained in 22.6.

Both power-on reset and CE reset are controlled by the reset signals  $\overline{IRES}$ ,  $\overline{RES}$ , and  $\overline{RESET}$  output from the reset controller shown in Figure 22-1.

Table 22-1 shows the relationship between the  $\overline{IRES}$ ,  $\overline{RES}$ , and  $\overline{RESET}$  signals, and power-on reset, and CE reset.

The reset controller also operates when the clock stop instruction (STOP s) explained in 21. **STANDBY** is executed.

The following sections 22.3 and 22.4 explain CE reset and power-on reset respectively.

Section 22.5 explains the relationship between CE reset and power-on reset.

**Table 22-1. Relationship Between Internal Reset Signals and Each Reset Operation**

Internal Reset Signal	Output Signal			Control Operation by Each Reset Signal
	CE Reset	Power-on Reset	Clock Stop	
$\overline{IRES}$	×	○	○	Forcibly sets device in halt status. Halt status is released when basic timer 0 carry FF is set.
$\overline{RES}$	×	○	○	Initializes some control registers.
$\overline{RESET}$	○	○	○	Initializes program counter, stack, system register, and some control registers.

## 22.3 CE Reset

CE reset is effected when the CE pin goes high.

When the CE pin goes high, the  $\overline{\text{RESET}}$  signal is output in synchronization with the rising edge of the next basic timer 0 carry FF setting pulse, and the device is reset.

When CE reset is effected, the  $\overline{\text{RESET}}$  signal initializes the program counter, stack, system register, and some control registers, and the program is executed starting from address 0000H.

For the value to which each of the above registers is initialized, refer to the description of each register.

The operation of CE reset differs depending on whether the clock stop instruction is used.

The differences in operation are explained in the following subsections 22.3.1 and 22.3.2.

Subsection 22.3.3 explains the points to be noted on using CE reset.

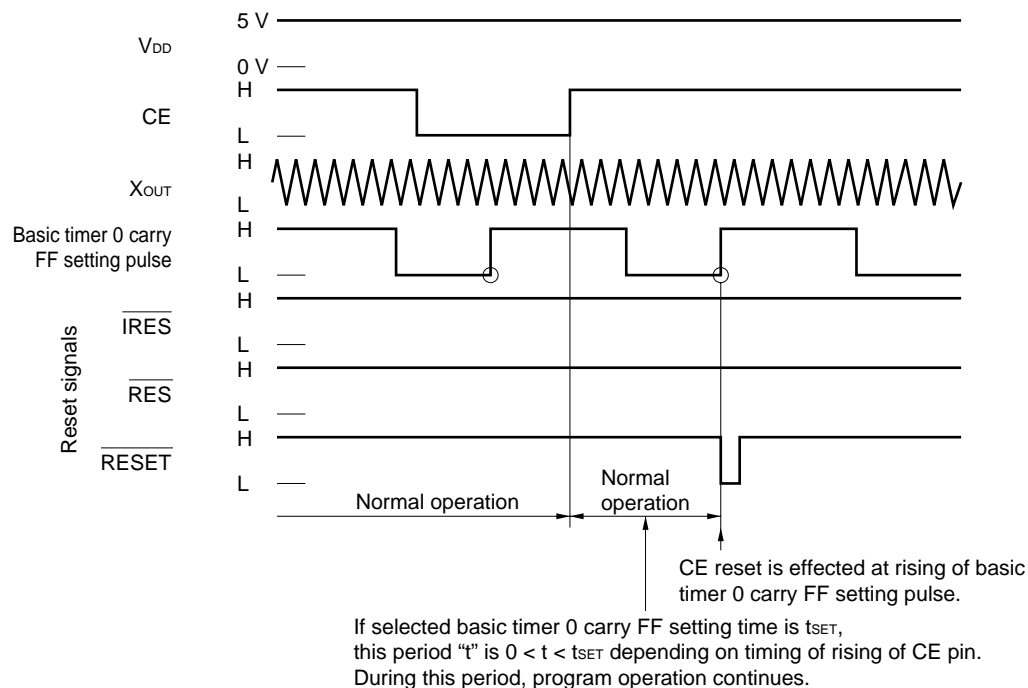
### 22.3.1 CE reset when clock stop (STOP s) instruction is not used

Figure 22-2 shows the operation of CE reset when the clock stop (STOP s) instruction is not used.

When the STOP s instruction is not used, the basic timer clock select register of the control registers is not initialized.

After the CE pin has gone high, therefore, the  $\overline{\text{RESET}}$  signal is output at the rising edge of the basic timer 0 carry FF setting pulse (1 ms, 5 ms, 100 ms, 250 ms) selected at that time, and the device is reset.

**Figure 22-2. CE Reset Operation When Clock Stop Instruction Is Not Used**



### 22.3.2 CE reset when clock stop (STOP s) instruction is used

Figure 22-3 shows the operation of CE reset when the clock stop (STOP s) instruction is used.

When the STOP s instruction is used, the  $\overline{\text{IRES}}$ ,  $\overline{\text{RES}}$ , and  $\overline{\text{RESET}}$  signals are output as soon as the STOP s instruction has been executed.

At this time, the basic timer clock select register of the control registers is initialized to 0000B by the  $\overline{\text{RES}}$  signal, the basic timer 0 carry FF setting signal is set to 100 ms.

Because the  $\overline{\text{IRES}}$  signal is output while the CE pin is low, the halt status, which can be released by the basic timer 0 carry, is forcibly set.

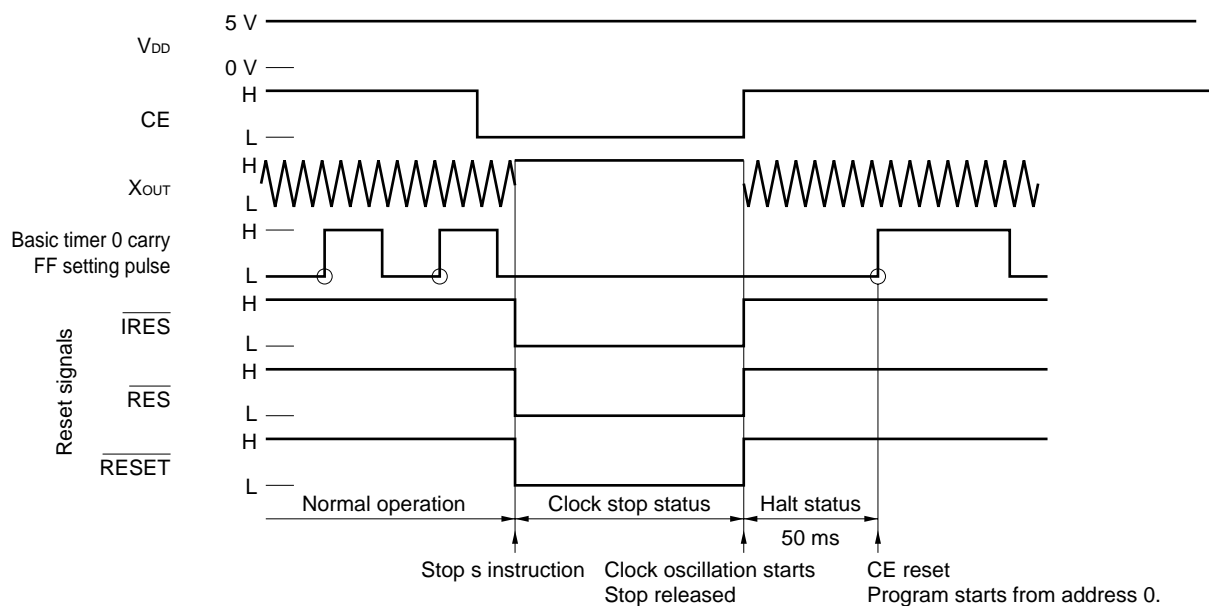
However, the device stops operation because the clock is stopped.

When the CE pin goes high, the clock stop status is released, and oscillation starts.

Because the halt status that can be released by the basic timer 0 carry is set at this time by the  $\overline{\text{IRES}}$  signal, the program starts from address 0 when the CE pin goes high and then the basic timer 0 carry FF setting pulse rises.

Because the basic timer 0 carry FF setting pulse is initialized to 100 ms, CE reset is effected 50 ms after the CE pin has gone high.

**Figure 22-3. CE Reset Operation When Clock Stop Instruction Is Used**



### 22.3.3 Notes on CE reset

Because CE reset is effected regardless of the instruction under execution, the following points <1> and <2> must be noted.

#### (1) Time to execute timer processing such as watch

When developing a watch program by using basic timer 0 or basic timer 1, the processing of that program must be completed within a specific time.

For details, refer to **12.2.6 Notes on using basic timer 0** and **12.3.5 Notes on using basic timer 1**.

#### (2) Processing of data and flag used for program

Care must be exercised in rewriting the contents of data or a flag that cannot be processed with one instruction and whose contents must not change even when CE reset is effected, such as a security code. This is explained in detail using the following examples.

**Example 1.**

R1	MEM	0.01H	; First digit of key input data of security code
R2	MEM	0.02H	; Second digit of key input data of security code
R3	MEM	0.03H	; First digit data for changing security code
R4	MEM	0.04H	; Second digit data for changing security code
M1	MEM	0.11H	; First digit of current security code
M2	MEM	0.12H	; Second digit of current security code

START:

Key input processing R1 ← contents of key A ; Security code input wait mode R2 ← contents of key B ; Substitutes contents of pressed key into R1 and R2.
--

```

SET2  CMP, Z ;<1> ; Compares security code with input data.
SUB   R1, M1
SUB   R2, M2
SKT1  Z
BR    ERROR ; Input data is different from security code.
    
```

MAIN:

Key input processing R3 ← contents of key C ; Security code rewriting mode R4 ← contents of key D ; Substitutes contents of pressed key into R3 and R4.
---

```

ST     M1, R3 ;<2> ; Rewrites security code.
ST     M2, R4 ;<3>
BR     MAIN
    
```

ERROR:

Must not operate
------------------

Suppose the current security code is "12H" in the above program, the contents of data memory areas M1 and M2 are "1H" and "2H", respectively.

If CE reset is effected, the contents of the key input are compared with security code "12H" in <1>. If they match, normal processing is performed.

If the security code is changed by the main processing, the new code is written to M1 and M2 in <2> and <3>.

Suppose the security code is changed to "34H", "3H", and "4H" are written to M1 and M2, respectively, in <2> and <3>.

If a CE reset is effected at the point where <2> is executed, the program is executed from address 0000H without <3> being executed.

Consequently, the security code is changed to "32H", making it impossible to clear the security.

In this case, use the program shown in Example 2 below.

**Example 2.**

```

R1      MEM      0.01H      ; First digit of key input data of security code
R2      MEM      0.02H      ; Second digit of key input data of security code
R3      MEM      0.03H      ; First digit data for changing security code
R4      MEM      0.04H      ; Second digit data for changing security code
M1      MEM      0.11H      ; First digit of current security code
M2      MEM      0.12H      ; Second digit of current security code
CHANGE FLG  0.13H.0      ; "1" while security code is changed

```

START:

Key input processing
R1 ← contents of key A ; Security code input wait mode
R2 ← contents of key B ; Substitutes contents of pressed key into R1 and R2.

```

SKT1    CHANGE ;<4> ; If CHANGE flag is "1"
BR      SECURITY_CHK
ST      M1, R3      ; rewrites M1 and M2.
ST      M2, R4
CLR1    CHANGE

```

SECURITY\_CHK:

```

SET2    CMP, Z ;<1> ; Compares security code with input data.
SUB      R1, M1
SUB      R2, M2
SKT1    Z
BR      ERROR      ; Input data is different from security code.

```

MAIN:

Key input processing
R3 ← contents of key C ; Security code rewriting mode
R4 ← contents of key D ; Substitutes contents of pressed key into R3 and R4.

```

SET1    CHANGE ;<5> ; Until security code is changed
          ; Sets CHANGE flag to "1".
ST      M1, R3 ;<2> ; Rewrites security code
ST      M2, R4 ;<3>
CLR1    CHANGE      ; When security code has been changed, sets
          ; CHANGE flag to "0".
BR      MAIN

```

ERROR:

Must not operate
------------------

In the program in Example 2, the CHANGE flag is set to "1" in <5> before the security code is changed in <2> and <3>.

Therefore, the security code is rewritten in <4> even if a CE reset is effected before <3> is executed.



## 22.4 Power-on Reset

Power-on reset is effected when the supply voltage  $V_{DD}$  of the device rises from a specific level (called power-on clear voltage).

If the supply voltage  $V_{DD}$  is lower than the power-on clear voltage, a power-on clear signal (POC) is output from the voltage detector shown in Figure 22-1.

When the power-on clear voltage is output, the crystal oscillator is stopped, and the device operation is stopped.

While the power-on clear signal is output, the  $\overline{IRES}$ ,  $\overline{RES}$ , and  $\overline{RESET}$  signals are output.

If supply voltage  $V_{DD}$  exceeds the power-on clear voltage, the power-on clear signal is cleared, and crystal oscillation is started. At the same time, the  $\overline{IRES}$ ,  $\overline{RES}$ , and  $\overline{RESET}$  signals are also cleared.

At this time, the halt status is set to be released by the basic timer 0 carry due to the  $\overline{IRES}$  signal. Therefore, power-on reset is effected at the rising edge of the next basic timer 0 carry FF setting signal.

The basic timer 0 carry FF setting signal is initialized to 100 ms by the  $\overline{RESET}$  signal. For this reason, reset is effected 50 ms after supply voltage  $V_{DD}$  has exceeded the power-on clear voltage, and the program is started from address 0.

This operation is illustrated in Figure 22-4.

The program counter, stack, system register, and control registers are initialized as soon as the power-on clear signal has been output.

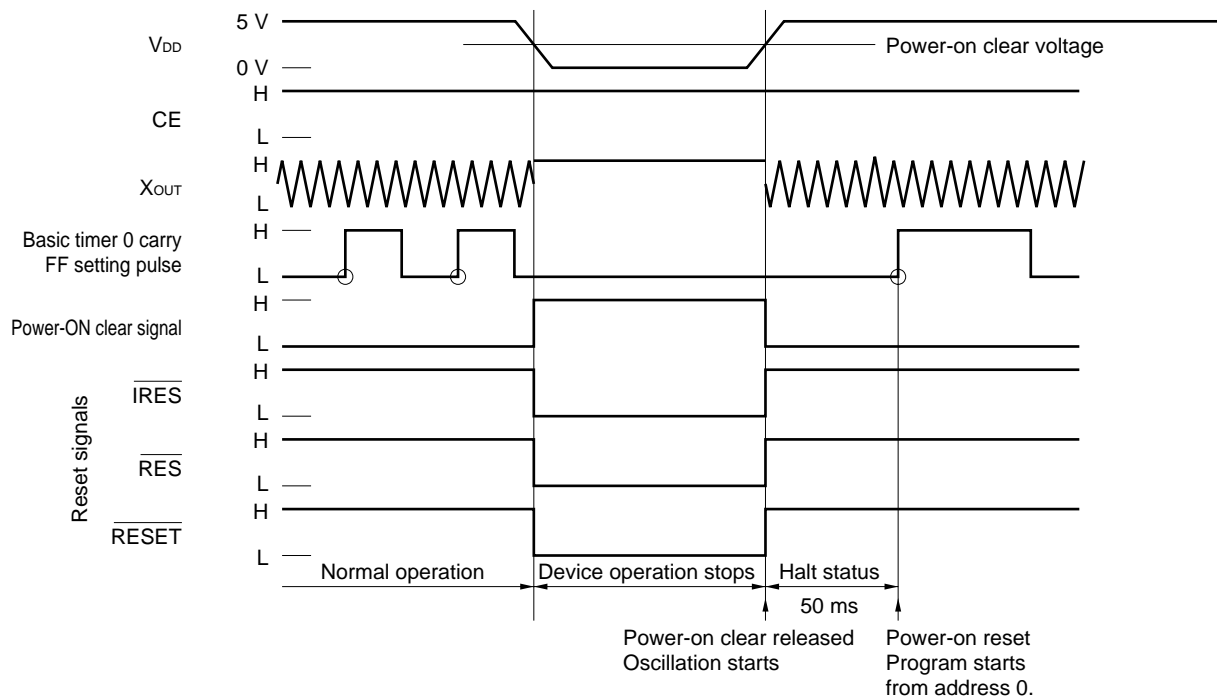
For the value to which each of the above registers is to be initialized, refer to the description of each register.

The power-on clear voltage is 3.5 V (rated value) during normal operation, and 2.3 V (rated value) in the clock stop status.

The operations performed when the power-on clear voltage is at the respective levels are explained in 22.4.1 and 22.4.2.

The operation to be performed if the supply voltage  $V_{DD}$  rises from 0 V is explained in 22.4.3.

**Figure 22-4. Operation of Power-on Reset**



#### 22.4.1 Power-on reset during normal operation

Figure 22-5 (a) shows the operation.

As shown in the figure, the power-on clear signal is output and the device operation stops regardless of the input level of the CE pin, if the supply voltage  $V_{DD}$  drops below 3.5 V.

If  $V_{DD}$  rises beyond 3.5 V again, the program starts from address 0000H after 50 ms of halt status.

“Normal operation” is when the clock stop instruction is not used and includes the halt status that is set by the halt instruction.

#### 22.4.2 Power-on reset in clock stop status

Figure 22-5 (b) shows the operation.

As shown in the figure, the power-on clear signal is output and the device operation stops if supply voltage  $V_{DD}$  drops below 2.3 V.

However, it seems as if the device operation has not changed because the device is in the clock stop status.

When supply voltage  $V_{DD}$  rises beyond 3.5 V next time, the program starts from address 0000H after a 50 ms halt.

#### 22.4.3 Power-on reset when supply voltage $V_{DD}$ rises from 0 V

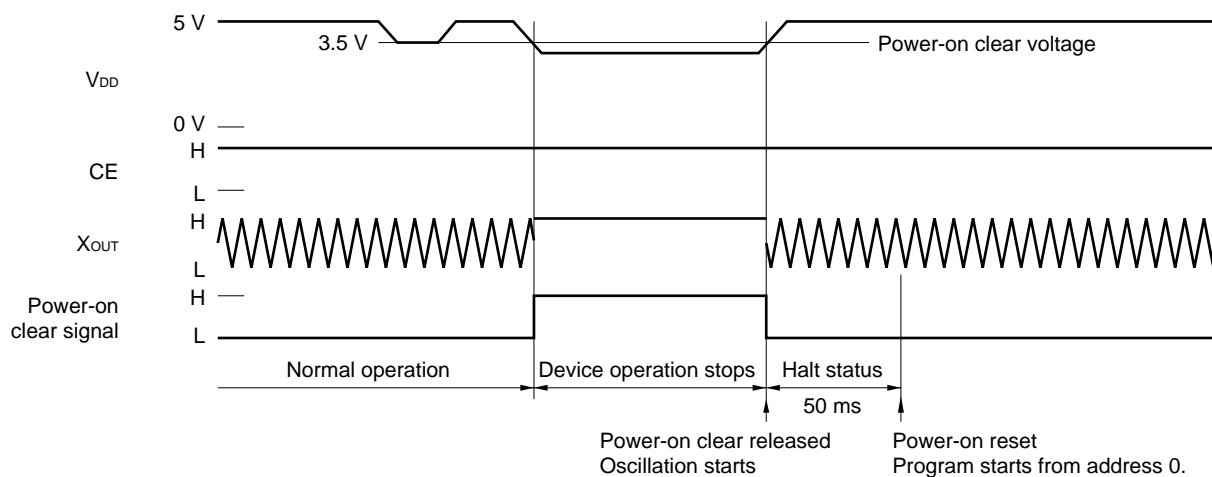
Figure 22-5 (c) shows the operation.

As shown in the figure, the power-on clear signal is output until supply voltage  $V_{DD}$  rises from 0 V to 3.5 V.

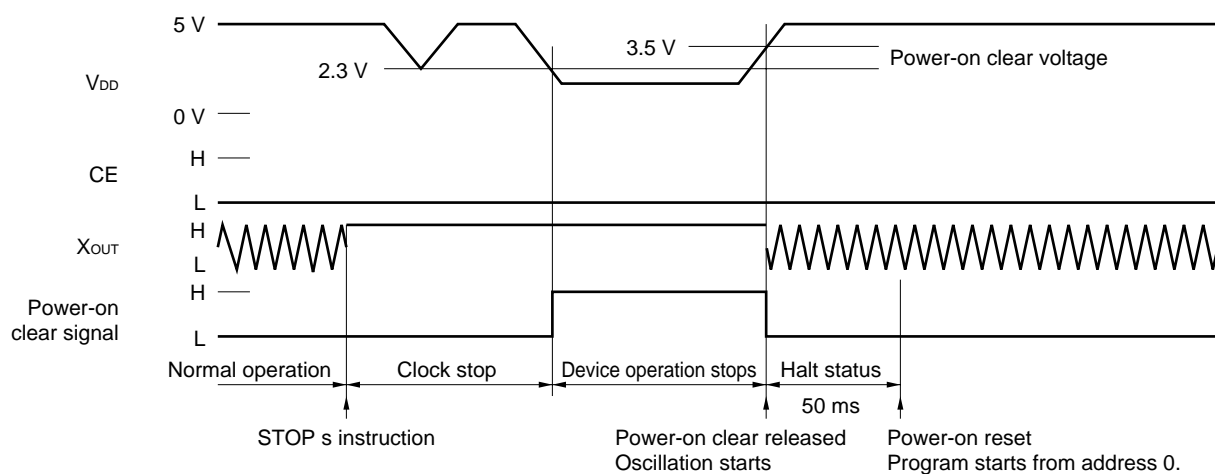
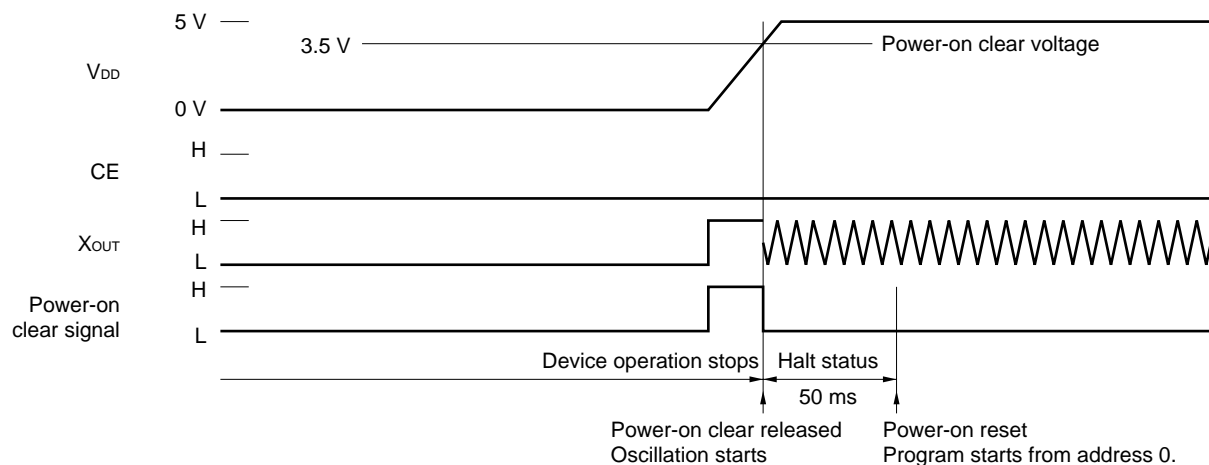
When  $V_{DD}$  rises beyond the power-on clear voltage, the crystal oscillator starts operating, and the program starts from address 0000H after a 50 ms halt.

Figure 22-5. Power-on Reset and Supply Voltage  $V_{DD}$ 

## (a) During normal operation (including halt status)



## (b) In clock stop status

(c) When supply voltage  $V_{DD}$  rises from 0 V

## 22.5 Relationship Between CE Reset and Power-on Reset

There is a possibility that power-on reset and CE reset are effected at the same time when power is first applied.

The reset operations performed at this time are explained in 22.5.1 through 22.5.3.

22.5.4 explains the points to be noted in raising supply voltage  $V_{DD}$ .

### 22.5.1 If $V_{DD}$ pin and CE pin rise simultaneously

Figure 22-6 (a) shows the operation.

At this time, the program starts from address 0000H due to power-on reset.

### 22.5.2 If CE pin rises in forced halt status of power-on reset

Figure 22-6 (b) shows the operation.

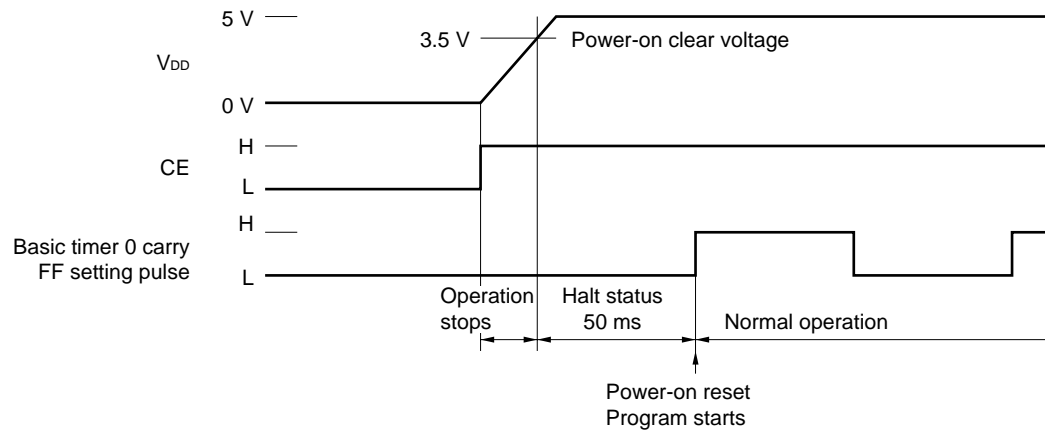
At this time, the program starts from address 0000H due to power-on reset in the same manner as in 22.5.1.

### 22.5.3 If CE pin rises after power-on reset

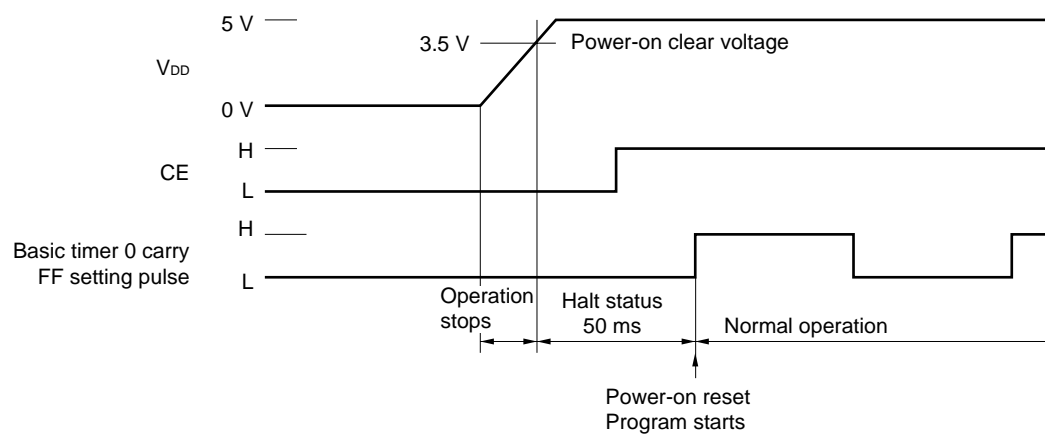
Figure 22-6 (c) shows the operation.

At this time, the program starts from address 0000H due to power-on reset, and the program starts from address 0000H again at the rising of the next basic timer 0 carry FF setting signal because of CE reset.

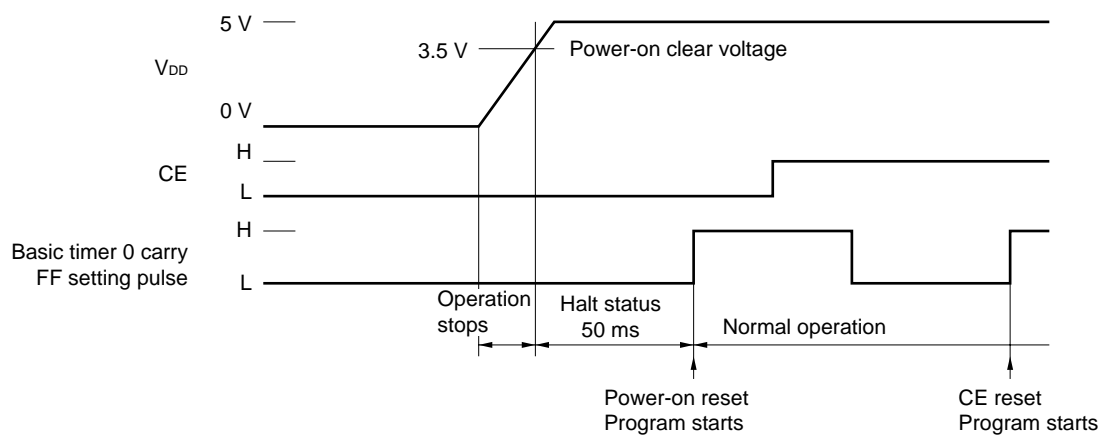
Figure 22-6. Relationship Between Power-on Reset and CE Reset

(a) If  $V_{DD}$  and CE pins rise simultaneously

## (b) If CE pin rises in halt status



## (c) If CE pin rises after power-on reset



### 22.5.4 Notes on raising supply voltage $V_{DD}$

When raising supply voltage  $V_{DD}$ , keep in mind the following points (1) and (2).

#### (1) When raising supply voltage $V_{DD}$ from power-on clear voltage

It is necessary to raise supply voltage  $V_{DD}$  to higher than 3.5 V at least once.

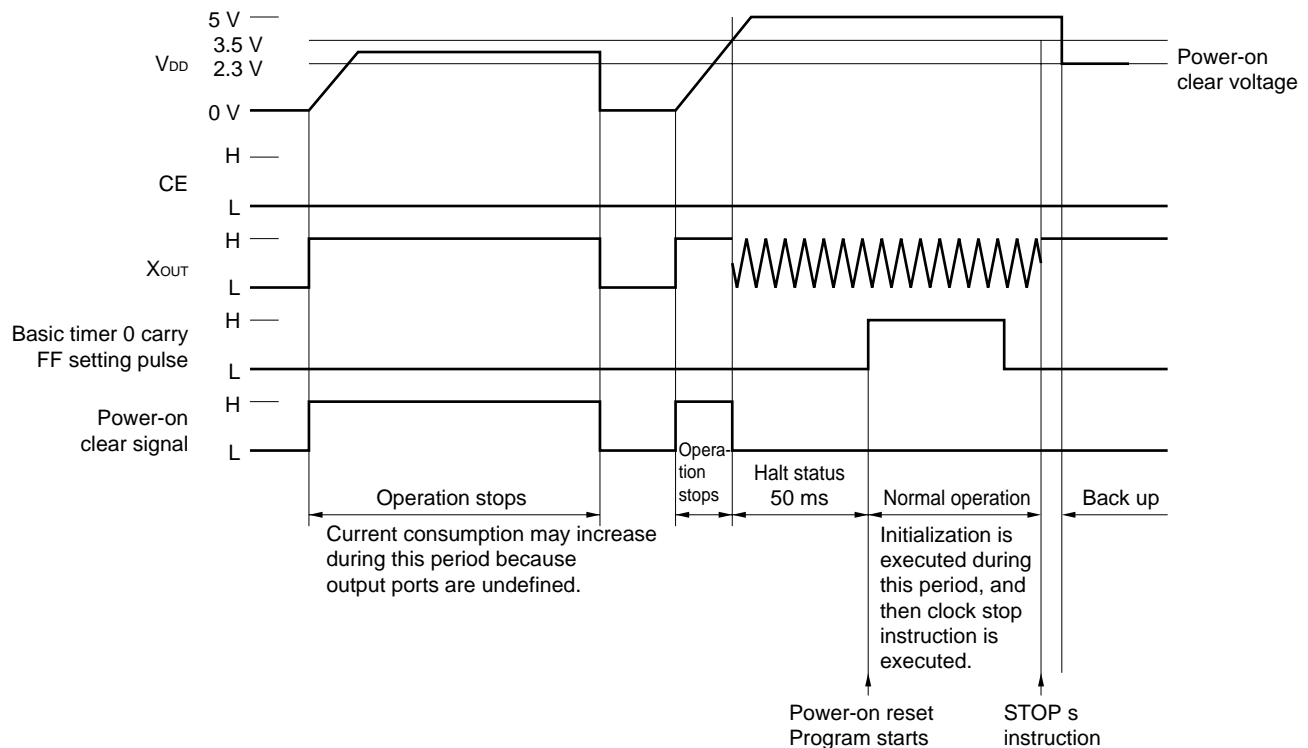
This is illustrated in Figure 22-7.

Suppose, for example, only a voltage less than 3.5 V is applied on application of  $V_{DD}$  with a program that backs up  $V_{DD}$  at 2.3 V by using the clock stop instruction, as shown in Figure 22-7, the power-on clear signal is continuously output, and the program does not operate.

Because the output ports of the device output undefined values, the current consumption increases in some cases.

If the device is backed up by batteries, therefore, the back-up time is substantially shortened.

Figure 22-7. Notes on Raising  $V_{DD}$



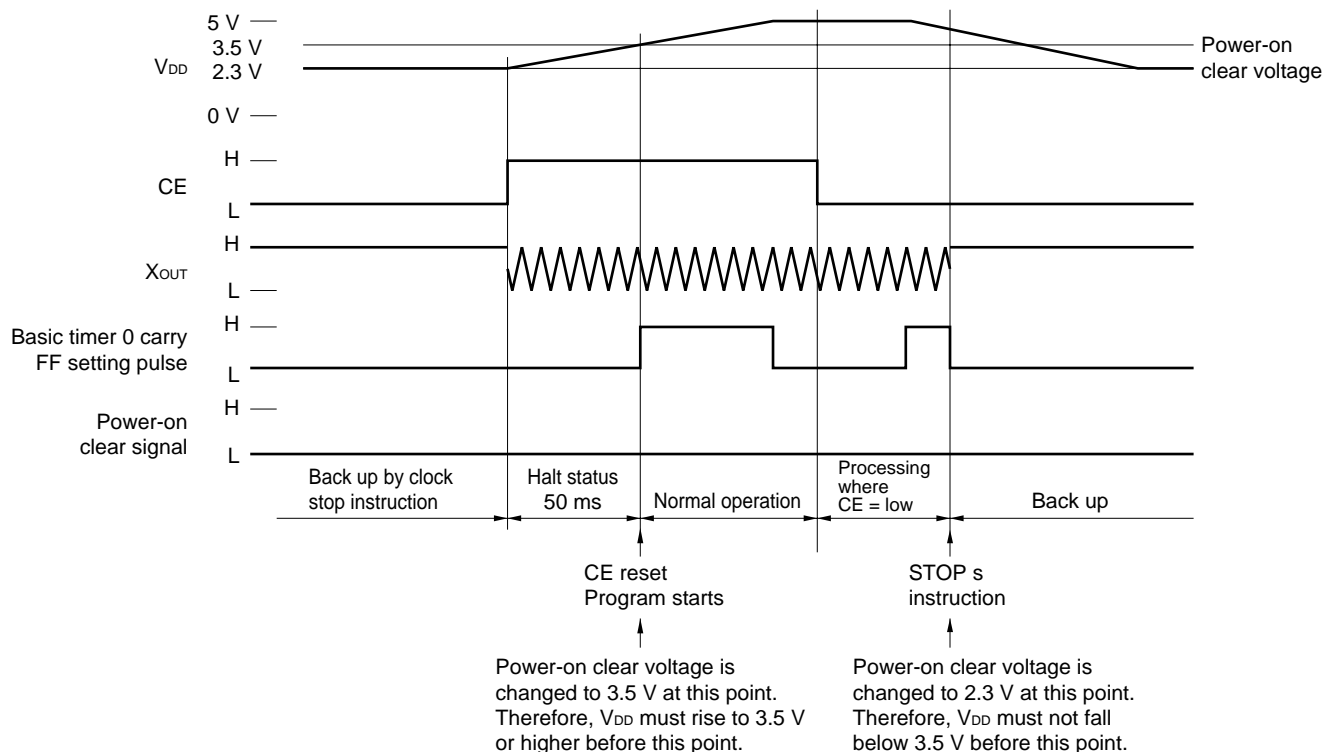
**(2) Restoring from clock stop status**

To restore the device from the back-up status while supply voltage  $V_{DD}$  is backed up at 2.3 V by using the clock stop instruction,  $V_{DD}$  must be raised to 3.5 V or higher within 50 ms after the CE pin has gone high.

As shown in Figure 22-8, the device is restored from the clock stop status by means of CE reset. Because the power-on clear voltage is changed to 3.5 V 50 ms after the CE pin has gone high, power-on reset is effected unless  $V_{DD}$  is 3.5 V or higher at this point.

The same applies when  $V_{DD}$  is lowered.

**Figure 22-8. Restoring from Clock Stop Status**



## 22.6 Power Failure Detection

Power failure detection is used to judge whether power-on reset by application of supply voltage  $V_{DD}$ , or CE reset has been effected when the device is reset, as shown in Figure 22-9.

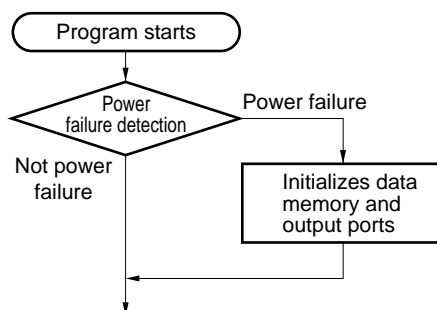
Because the contents of the data memory and ports are undefined on power application, these contents are initialized by means of power failure detection.

A power failure can be detected in two ways: by using the power failure detector to detect the BTM0CY flag, and by detecting the contents of the data memory (RAM judgement).

22.6.1 and 22.6.2 explain how a power failure is detected by using the power failure detector and BTM0CY flag.

22.6.3 and 22.6.4 explain how a power failure is detected by RAM judgement method.

**Figure 22-9. Power Failure Detection Flow Chart**



### 22.6.1 Power failure detector

The power failure detector consists of a voltage detector, a basic timer 0 carry disable flip-flop that is set by the output (power-on clear signal) of the voltage detector, and a basic timer 0 carry, as shown in Figure 22-1.

The basic timer 0 carry disable FF is set to 1 by the power-on clear signal, and is reset to 0 when an instruction that reads the BTM0CY flag is executed.

When the basic timer 0 carry disable FF is set to 1, the BTM0CY flag is not set to 1.

When the power-on clear signal is output (at power-on reset), the program is started with the BTM0CY flag reset, and the BTM0CY flag is disabled from being set until an instruction that reads the BTM0CY flag is executed.

Once the instruction that reads the BTM0CY flag has been executed, the BTM0CY flag is set each time the basic timer 0 carry FF setting pulses has risen. It can be judged whether power-on reset (power failure) or CE reset (not power failure) has been effected by detecting the contents of the BTM0CY flag when the device is reset. Power-on reset has been effected if the BTM0CY flag is reset to 0; CE reset has been effected if it is set to 1.

The voltage at which a power failure can be detected is the same as the voltage at which power-on reset is effected, or  $V_{DD} = 3.5$  V during crystal oscillation, or  $V_{DD} = 2.3$  V in the clock stop status.

Figure 22-10 shows the transition of the status of the BTM0CY flag.

Figures 22-11 and 22-10 show the timing chart and the operation of the BTM0CY flag.



Figure 22-10. Status Transition of BTM0CY Flag

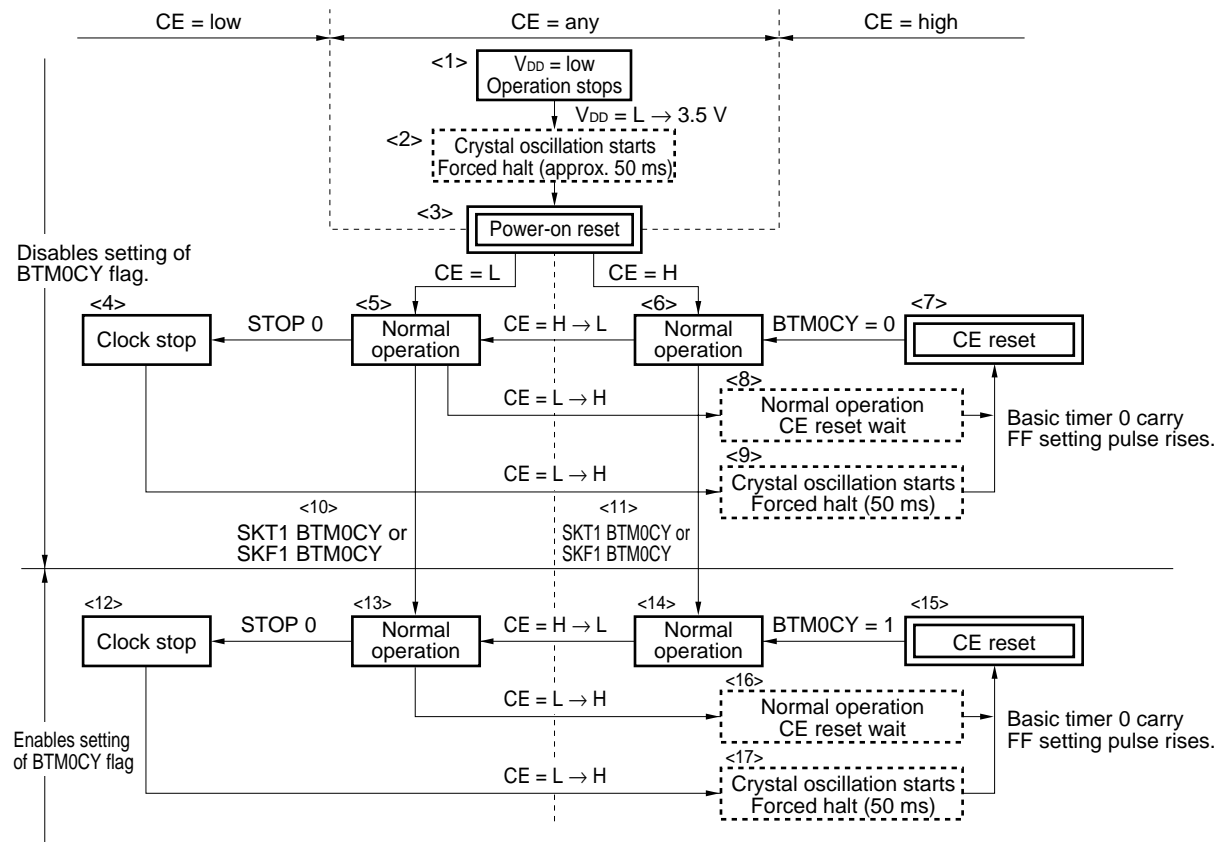
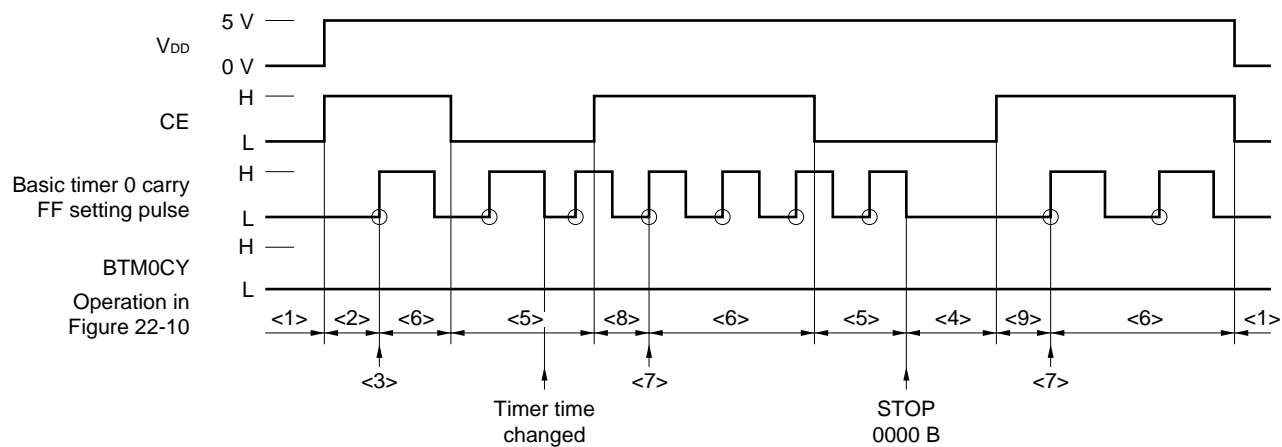
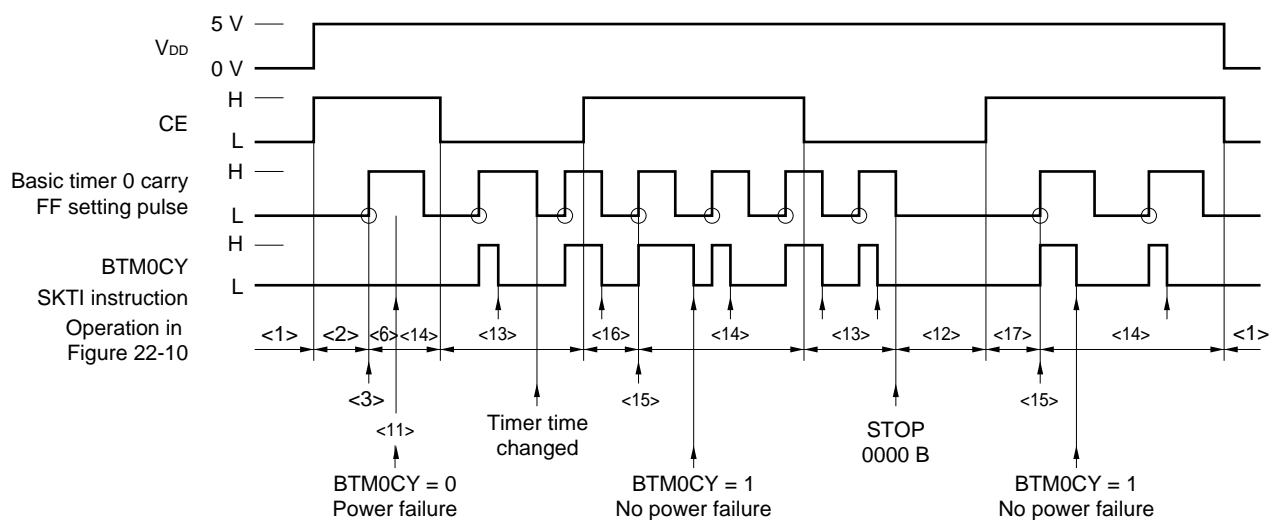


Figure 22-11. Operation of BTM0CY Flag

(a) When BTM0CY flag never detected (SKT1 BTM0CY or SKF1 BTM0CY is not executed)



(b) When detecting power failure by BTM0CY flag



### 22.6.2 Notes on detecting power failure by BTM0CY flag

The following points must be noted when using the BTM0CY flag for watch counting.

#### (1) Updating watch

When developing a watch program by using the basic timer 0 carry, the watch must be updated after a power failure has been detected.

This is because counting of the watch is skipped once because the BTM0CY flag is reset to 0 when the BTM0CY flag is read on detection of a power failure.

#### (2) Watch updating processing time

The processing to update the watch must be completed before the next basic timer 0 carry FF setting pulse rises.

This is because, if the CE pin goes high during the watch updating processing, CE reset is effected without the watch updating processing completed.

For further information on (1) and (2) above, refer to **12.2.6 (3) Correction of basic timer 0 carry on CE reset.**

When detecting a power failure, the following points must be noted.

#### (3) Timing of power failure detection

To count the watch by using the BTM0CY flag, the BTM0CY flag must be read to detect a power failure within the time since the program has started from address 0000H until the next basic timer 0 carry FF setting pulse rises.

For example if the basic timer 0 carry FF setting time is set to 5 ms, and a power failure is detected 6 ms after the program has been started, the BTM0CY flag is overlooked once.

For details, refer to **12.2.6 (3) Correction of basic timer 0 carry on CE reset.**

Power failure detection and initial processing must be completed within the basic timer 0 carry FF setting time as shown in the following example.

This is because, if the CE pin goes high and CE reset is effected during power failure detection processing and initial processing, these processing may be stopped in midway, and thus problems may occur.

To change the basic timer 0 carry FF setting time by the initial processing, the instruction that changes the time must be executed at the end of the initial processing, and the instruction must be one instruction.

This is because the initial processing may not be completely executed because of CE reset if the basic timer 0 carry FF setting time is changed before the initial processing is executed, as shown in the following example.

# Example

```

START:                                ; Program address 0000H
;<1>

Processing on reset

;<2>
    SKT1    BTM0CY                    ; Power failure detection
    BR      INITIAL
BACKUP:
;<3>

Watch updating

    BR      MAIN
INITIAL:
;<4>

Initial processing

;<5>
    INITFLG BTM0CK1, NOT BTM0CK0      ; Embedded macro
                                        ; Sets basic timer 0 carry FF setting time to 5 ms

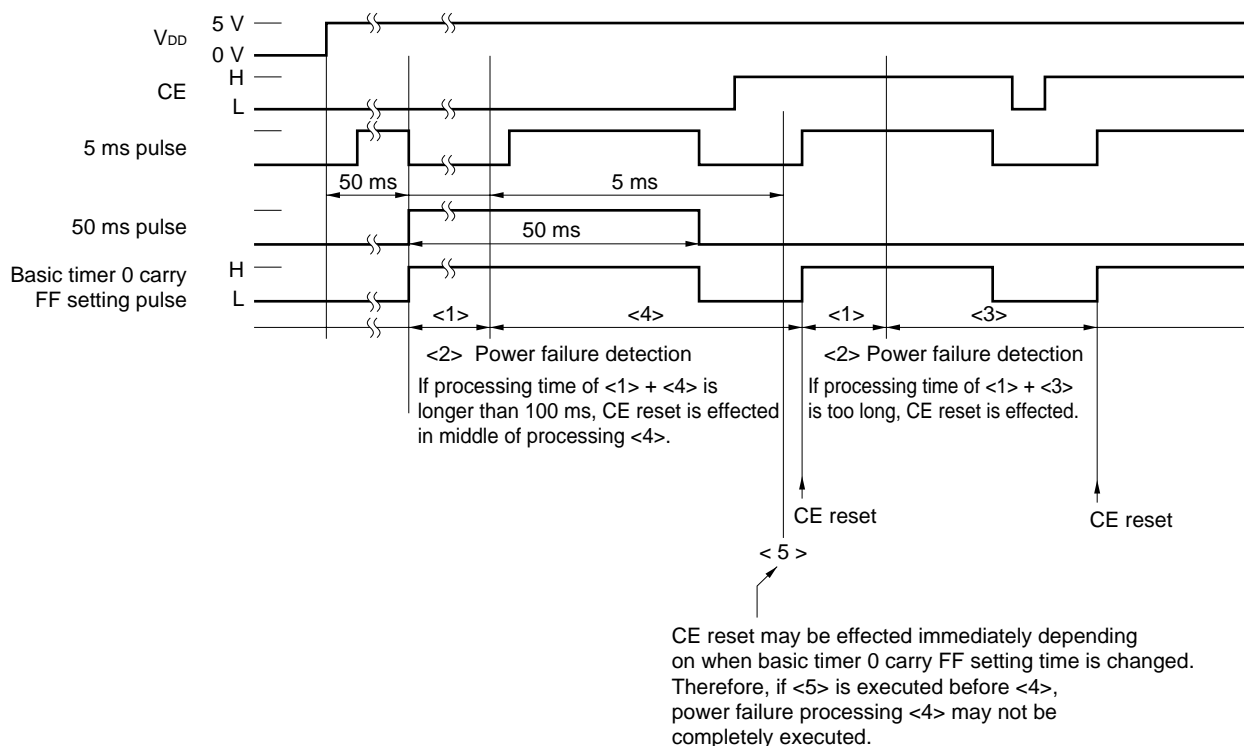
MAIN:

Main processing

    SKT1    BTM0CY
    BR      MAIN

Watch updating

    BR      MAIN
    
```

**Example of operation****22.6.3 Power failure detection by RAM judgement method**

The RAM judgement method is to detect a power failure by judging whether the contents of the data memory at a specific address are the specified value.

An example of a program that detects a power failure by the RAM judgement method is shown below.

The RAM judgement method detects a power failure by comparing an undefined value with the specified value because the contents of the data memory are undefined on application of supply voltage  $V_{DD}$ .

Therefore, there is a possibility that a wrong judgment may be made as explained in **22.6.4 Notes on detecting power failure by RAM judgement method.**

**Example Program to detect power failure by RAM judgement method**

```

M012    MEM    0.12H
M034    MEM    0.34H
M056    MEM    0.56H
M107    MEM    1.07H
M128    MEM    1.28H
M16F    MEM    1.6FH
DATA0    DAT    1010B
DATA1    DAT    0101B
DATA2    DAT    0110B
DATA3    DAT    1001B
DATA4    DAT    1100B
DATA5    DAT    0011B

START:
    SET2      CMP, Z
    SUB       M012, #DATA0    ; If M012 = DATA0 and
    SUB       M034, #DATA1    ; M034 = DATA1 and
    SUB       M056, #DATA2    ; M056 = DATA2 and
    BANK1
    SUB       M107, #DATA3    ; M107 = DATA3 and
    SUB       M128, #DATA4    ; M128 = DATA4 and
    SUB       M16F, #DATA5    ; M16F = DATA5,
    BANK0
    SKF1      Z
    BR        BACKUP          ; branches to BACKUP
; INITIAL:

```

Initial processing

```

MOV      M012, #DATA0
MOV      M034, #DATA1
MOV      M056, #DATA2
BANK1
MOV      M107, #DATA3
MOV      M128, #DATA4
MOV      M16F, #DATA5
BR       MAIN
BACKUP:

```

Backup processing

```

MAIN:

```

Main processing

#### 22.6.4 Notes on detecting power failure by RAM judgement method

The value of the data memory on application of supply voltage  $V_{DD}$  is basically undefined, and therefore, the following points (1) and (2) must be noted.

##### (1) Data to be compared

Where the number of bits of the data memory to be compared by the RAM judgement method is “n bits”, the probability at which the value of the data memory matches the value to be compared on application of  $V_{DD}$  is  $(1/2)^n$ .

This means that backup is judged at a probability of  $(1/2)^n$  when a power failure is detected by the RAM judgement method.

To lower this probability, as many bits as possible must be compared.

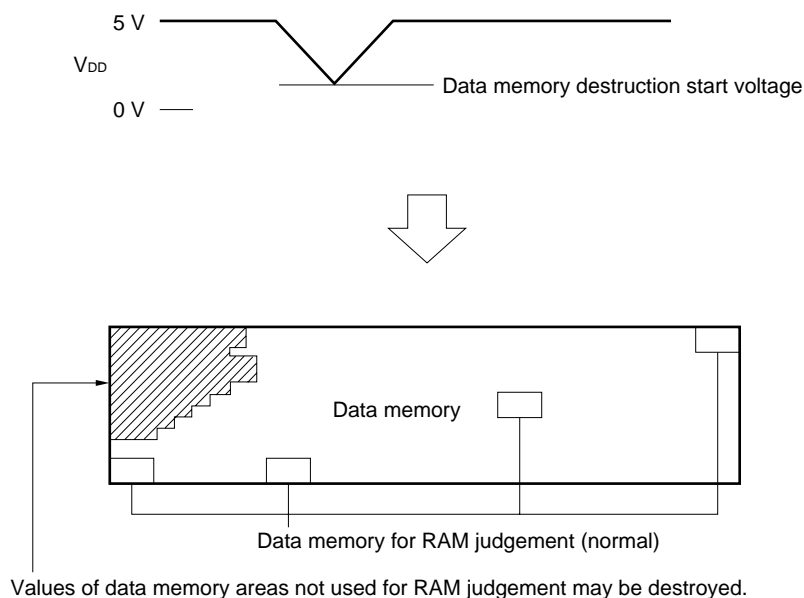
Because the contents of the data memory on application of  $V_{DD}$  are likely to be the same value such as “0000B” and “1111B”, it is recommended to mix “0” and “1” as data to be compared, such as “1010B” and “0110B” to reduce the possibility of a wrong judgment.

##### (2) Notes on program

If  $V_{DD}$  rises from the level at which the data memory contents may be destroyed as shown in Figure 22-12, and even if the value of the data memory area to be compared is normal, the values of the other data memory areas may be destroyed.

This is judged as backup if a power failure is detected by the RAM judgement method. Therefore, consideration must be given so that the program does not hang up even if the contents of the data memory are destroyed.

Figure 22-12.  $V_{DD}$  and Destruction of Data Memory Contents



## 23. INSTRUCTION SET

## 23.1 Outline of Instruction Set

b <sub>14</sub> to b <sub>11</sub> \ b <sub>15</sub>		0		1	
BIN	HEX				
0000	0	ADD	r, m	ADD	m, #n4
0001	1	SUB	r, m	SUB	m, #n4
0010	2	ADDC	r, m	ADDC	m, #n4
0011	3	SUBC	r, m	SUBC	m, #n4
0100	4	AND	r, m	AND	m, #n4
0101	5	XOR	r, m	XOR	m, #n4
0110	6	OR	r, m	OR	m, #n4
0111	7	INC	AR		
		INC	IX		
		RORC	r		
		MOVT	DBF, @AR		
		PUSH	AR		
		POP	AR		
		GET	DBF, p		
		PUT	p, DBF		
		PEEK	WR, rf		
		POKE	rf, WR		
		BR	@AR		
		CALL	@AR		
		RET			
		RETSK			
		RETI			
		EI			
		DI			
		STOP	s		
		HALT	h		
		NOP			
1000	8	LD	r, m	ST	m, r
1001	9	SKE	m, #n4	SKGE	m, #n4
1010	A	MOV	@r, m	MOV	m, @r
1011	B	SKNE	m, #n4	SKLT	m, #n4
1100	C	BR	addr (page 0)	CALL	addr (page 0)
1101	D	BR	addr (page 1)	MOV	m, #n4
1110	E			SKT	m, #n
1111	F			SKF	m, #n



## 23.2 Legend

AR:	Address register
ASR:	Address stack register indicated by stack pointer
addr:	Program memory address (lower 11 bits)
BANK:	Bank register
CMP:	Compare flag
CY:	Carry flag
DBF:	Data buffer
h:	Halt release condition
INTEF:	Interrupt enable flag
INTR:	Register automatically saved to stack when interrupt occurs
INTSK:	Interrupt stack register
IX:	Index register
MP:	Data memory row address pointer
MPE:	Memory pointer enable flag
m:	Data memory address indicated by m <sub>R</sub> , m <sub>C</sub>
m <sub>R</sub> :	Data memory row address (higher)
m <sub>C</sub> :	Data memory column address (lower)
n:	Bit position (4 bits)
n4:	Immediate data (4 bits)
PAGE:	Page (bit 11 of program counter)
PC:	Program counter
p:	Peripheral address
p <sub>H</sub> :	Peripheral address (higher 3 bits)
p <sub>L</sub> :	Peripheral address (lower 4 bits)
r:	General register column address
rf:	Register file address
r <sub>fR</sub> :	Register file address (higher 3 bits)
r <sub>fC</sub> :	Register file address (lower 4 bits)
SP:	Stack pointer
s:	Stop release condition
WR:	Window register
(x):	Contents addressed by x

## 23.3 Instruction Set List

Instructions	Mnemonic	Operand	Operation	Instruction Code			
				op Code	Operand		
Add	ADD	r, m	$(r) \leftarrow (r) + (m)$	00000	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) + n4$	10000	m <sub>R</sub>	m <sub>C</sub>	n4
	ADDC	r, m	$(r) \leftarrow (r) + (m) + CY$	00010	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) + n4 + CY$	10010	m <sub>R</sub>	m <sub>C</sub>	n4
	INC	AR	$AR \leftarrow AR + 1$	00111	000	1001	0000
		IX	$IX \leftarrow IX + 1$	00111	000	1000	0000
Subtract	SUB	r, m	$(r) \leftarrow (r) - (m)$	00001	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) - n4$	10001	m <sub>R</sub>	m <sub>C</sub>	n4
	SUBC	r, m	$(r) \leftarrow (r) - (m) - CY$	00011	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) - n4 - CY$	10011	m <sub>R</sub>	m <sub>C</sub>	n4
Logical operation	OR	r, m	$(r) \leftarrow (r) \vee (m)$	00110	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) \vee n4$	10110	m <sub>R</sub>	m <sub>C</sub>	n4
	AND	r, m	$(r) \leftarrow (r) \wedge (m)$	00100	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) \wedge n4$	10100	m <sub>R</sub>	m <sub>C</sub>	n4
	XOR	r, m	$(r) \leftarrow (r) \nabla (m)$	00101	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow (m) \nabla n4$	10101	m <sub>R</sub>	m <sub>C</sub>	n4
Judge	SKT	m, #n	$CMP \leftarrow 0$ , if $(m) \wedge n = n$ , then skip	11110	m <sub>R</sub>	m <sub>C</sub>	n
	SKF	m, #n	$CMP \leftarrow 0$ , if $(m) \wedge n = 0$ , then skip	11111	m <sub>R</sub>	m <sub>C</sub>	n
Compare	SKE	m, #n4	$(m) - n4$ , skip if zero	01001	m <sub>R</sub>	m <sub>C</sub>	n4
	SKNE	m, #n4	$(m) - n4$ , skip if not zero	01011	m <sub>R</sub>	m <sub>C</sub>	n4
	SKGE	m, #n4	$(m) - n4$ , skip if not borrow	11001	m <sub>R</sub>	m <sub>C</sub>	n4
	SKLT	m, #n4	$(m) - n4$ , skip if borrow	11011	m <sub>R</sub>	m <sub>C</sub>	n4
Rotate	RORC	r	$\rightarrow CY \rightarrow (r)_{b3} \rightarrow (r)_{b2} \rightarrow (r)_{b1} \rightarrow (r)_{b0} \leftarrow$	00111	000	0111	r
Transfer	LD	r, m	$(r) \leftarrow (m)$	01000	m <sub>R</sub>	m <sub>C</sub>	r
	ST	m, r	$(m) \leftarrow (r)$	11000	m <sub>R</sub>	m <sub>C</sub>	r
	MOV	@r, m	if MPE = 1: $(MP, (r)) \leftarrow (m)$ if MPE = 0: $(BANK, m_R, (r)) \leftarrow (m)$	01010	m <sub>R</sub>	m <sub>C</sub>	r
		m, @r	if MPE = 1: $(m) \leftarrow (MP, (r))$ if MPE = 0: $(m) \leftarrow (BANK, m_R, (r))$	11010	m <sub>R</sub>	m <sub>C</sub>	r
		m, #n4	$(m) \leftarrow n4$	11101	m <sub>R</sub>	m <sub>C</sub>	n4
	MOVT	DBF, @AR	$SP \leftarrow SP - 1$ , $ASR \leftarrow PC$ , $PC \leftarrow AR$ , $DBF \leftarrow (PC)$ , $PC \leftarrow ASR$ , $SP \leftarrow SP + 1$	00111	000	0001	0000
	PUSH	AR	$SP \leftarrow SP - 1$ , $ASR \leftarrow AR$	00111	000	1101	0000
	POP	AR	$AR \leftarrow ASR$ , $SP \leftarrow SP + 1$	00111	000	1100	0000
	PEEK	WR, rf	$WR \leftarrow (rf)$	00111	rf <sub>R</sub>	0011	rf <sub>C</sub>

Instructions	Mnemonic	Operand	Operation	Instruction Code			
				op Code	Operand		
Transfer	POKE	rf, WR	$(rf) \leftarrow WR$	00111	r <sub>fR</sub>	0010	r <sub>fC</sub>
	GET	DBF, p	$DBF \leftarrow (p)$	00111	p <sub>H</sub>	1011	p <sub>L</sub>
	PUT	p, DBF	$(p) \leftarrow DBF$	00111	p <sub>H</sub>	1010	p <sub>L</sub>
Branch	BR	addr	$PC_{10-0} \leftarrow addr, PAGE \leftarrow 0$	01100	addr		
			$PC_{10-0} \leftarrow addr, PAGE \leftarrow 1$	01101			
	@AR		$PC \leftarrow AR$	00111	000	0100	0000
Subroutine	CALL	addr	$SP \leftarrow SP - 1, ASR \leftarrow PC$ $PC_{10-0} \leftarrow addr, PAGE \leftarrow 0$	11100	addr		
		@AR	$SP \leftarrow SP - 1, ASR \leftarrow PC$ $PC \leftarrow AR$	00111	000	0101	000
	RET		$PC \leftarrow ASR, SP \leftarrow SP + 1$	00111	000	1110	0000
	RETSK		$PC \leftarrow ASR, SP \leftarrow SP + 1$ and skip	00111	001	1110	0000
	RETI		$PC \leftarrow ASR, INTR \leftarrow INTSK, SP \leftarrow SP + 1$	00111	010	1110	0000
Interrupt	EI		$INTEF \leftarrow 1$	00111	000	1111	0000
	DI		$INTEF \leftarrow 0$	00111	001	1111	0000
Others	STOP	s	STOP	00111	010	1111	s
	HALT	h	HALT	00111	011	1111	h
	NOP		No operation	00111	100	1111	0000

## 23.4 Assembler (RA17K) Embedded Macro Instructions

### Legend

flag n: FLG symbol

n: Bit number

< >: Can be omitted

	Mnemonic	Operand	Operation	n
Embedded macro	SKTn	flag 1, ... flag n	if (flag 1) to (flag n) = all "1", then skip	$1 \leq n \leq 4$
	SKFn	flag 1, ... flag n	if (flag 1) to (flag n) = all "0", then skip	$1 \leq n \leq 4$
	SETn	flag 1, ... flag n	(flag 1) to (flag n) $\leftarrow$ 1	$1 \leq n \leq 4$
	CLRn	flag 1, ... flag n	(flag 1) to (flag n) $\leftarrow$ 0	$1 \leq n \leq 4$
	NOTn	flag 1, ... flag n	if (flag n) = "0", then (flag n) $\leftarrow$ 1 if (flag n) = "1", then (flag n) $\leftarrow$ 0	$1 \leq n \leq 4$
	INITFLG	<NOT> flag 1, ... <<NOT> flag n>	if description = NOT flag n, then (flag n) $\leftarrow$ 0 if description = flag n, then (flag n) $\leftarrow$ 1	$1 \leq n \leq 4$
	BANKn		(BANK) $\leftarrow$ n	$0 \leq n \leq 2$

## 24. RESERVED SYMBOLS

### 24.1 Data Buffer (DBF)

Symbol Name	Attribute	Value	R/W	Description
DBF3	MEM	0.0CH	R/W	Bits 15 through 12 of DBF
DBF2	MEM	0.0DH	R/W	Bits 11 through 8 of DBF
DBF1	MEM	0.0EH	R/W	Bits 7 through 4 of DBF
DBF0	MEM	0.0FH	R/W	Bits 3 through 0 of DBF

### 24.2 System Register (SYSREG)

Symbol Name	Attribute	Value	R/W	Description
AR3	MEM	0.74H	R	Bits 15 through 12 of address register
AR2	MEM	0.75H	R	Bits 11 through 8 of address register
AR1	MEM	0.76H	R/W	Bits 7 through 4 of address register
AR0	MEM	0.77H	R/W	Bits 3 through 0 of address register
WR	MEM	0.78H	R/W	Window register
BANK	MEM	0.79H	R/W	Bank register
IXH	MEM	0.7AH	R/W	Index register, high
MPH	MEM	0.7AH	R/W	Memory pointer, high
MPE	FLG	0.7AH.3	R/W	Memory pointer enable flag
IXM	MEM	0.7BH	R/W	Index register, middle
MPL	MEM	0.7BH	R/W	Memory pointer, low
IXL	MEM	0.7CH	R/W	Index register, low
RPH	MEM	0.7DH	R/W	General register pointer, high
RPL	MEM	0.7EH	R/W	General register pointer, low
PSW	MEM	0.7FH	R/W	Program status word
BCD	FLG	0.7EH.0	R/W	BCD operation flag
CMP	FLG	0.7FH.3	R/W	Compare flag
CY	FLG	0.7FH.2	R/W	Carry flag
Z	FLG	0.7FH.1	R/W	Zero flag
IXE	FLG	0.7FH.0	R/W	Index enable flag

### 24.3 LCD Segment Register

Symbol Name	Attribute	Value	R/W	Description
LCDD0	MEM	2.6F	R/W	LCD segment register
LCDD1	MEM	2.6E	R/W	LCD segment register
LCDD2	MEM	2.6D	R/W	LCD segment register
LCDD3	MEM	2.6C	R/W	LCD segment register
LCDD4	MEM	2.6B	R/W	LCD segment register
LCDD5	MEM	2.6A	R/W	LCD segment register
LCDD6	MEM	2.69	R/W	LCD segment register
LCDD7	MEM	2.68	R/W	LCD segment register
LCDD8	MEM	2.67	R/W	LCD segment register
LCDD9	MEM	2.66	R/W	LCD segment register
LCDD10	MEM	2.65	R/W	LCD segment register
LCDD11	MEM	2.64	R/W	LCD segment register
LCDD12	MEM	2.63	R/W	LCD segment register
LCDD13	MEM	2.62	R/W	LCD segment register
LCDD14	MEM	2.61	R/W	LCD segment register
LCDD15	MEM	2.60	R/W	LCD segment register
LCDD16	MEM	2.5FH	R/W	LCD segment register
LCDD17	MEM	2.5EH	R/W	LCD segment register
LCDD18	MEM	2.5DH	R/W	LCD segment register
LCDD19	MEM	2.5CH	R/W	LCD segment register

## 24.4 Port Register

Symbol Name	Attribute	Value	R/W	Description
P0A2	FLG	0.70H.2	R/W	Bit 2 of port 0A
P0A1	FLG	0.70H.1	R/W	Bit 1 of port 0A
P0A0	FLG	0.70H.0	R/W	Bit 0 of port 0A
P0B3	FLG	0.71H.3	R/W	Bit 3 of port 0B
P0B2	FLG	0.71H.2	R/W	Bit 2 of port 0B
P0B1	FLG	0.71H.1	R/W	Bit 1 of port 0B
P0B0	FLG	0.71H.0	R/W	Bit 0 of port 0B
P0C3	FLG	0.72H.3	R/W	Bit 3 of port 0C
P0C2	FLG	0.72H.2	R/W	Bit 2 of port 0C
P0C1	FLG	0.72H.1	R/W	Bit 1 of port 0C
P0C0	FLG	0.72H.0	R/W	Bit 0 of port 0C
P0D3	FLG	0.73H.3	R	Bit 3 of port 0D
P0D2	FLG	0.73H.2	R	Bit 2 of port 0D
P0D1	FLG	0.73H.1	R	Bit 1 of port 0D
P0D0	FLG	0.73H.0	R	Bit 0 of port 0D
P1A2	FLG	1.70H.2	R/W	Bit 2 of port 1A
P1A1	FLG	1.70H.1	R/W	Bit 1 of port 1A
P1A0	FLG	1.70H.0	R/W	Bit 0 of port 1A
P1B3	FLG	1.71H.3	R/W	Bit 3 of port 1B
P1B2	FLG	1.71H.2	R/W	Bit 2 of port 1B
P1B1	FLG	1.71H.1	R/W	Bit 1 of port 1B
P1B0	FLG	1.71H.0	R/W	Bit 0 of port 1B
P1C3	FLG	1.72H.3	R/W	Bit 3 of port 1C
P1C2	FLG	1.72H.2	R/W	Bit 2 of port 1C
P1C1	FLG	1.72H.1	R/W	Bit 1 of port 1C
P1C0	FLG	1.72H.0	R/W	Bit 0 of port 1C
P1D3	FLG	1.73H.3	R	Bit 3 of port 1D
P1D2	FLG	1.73H.2	R	Bit 2 of port 1D
P1D1	FLG	1.73H.1	R	Bit 1 of port 1D
P1D0	FLG	1.73H.0	R	Bit 0 of port 1D
P2E0	FLG	2.5FH.0	R/W	Bit 0 of port 2E
P2F0	FLG	2.5EH.0	R/W	Bit 0 of port 2F
P2G0	FLG	2.5DH.0	R/W	Bit 0 of port 2G
P2H0	FLG	2.5CH.0	R/W	Bit 0 of port 2H

## 24.5 Register File (Control Register)

Symbol Name	Attribute	Value	R/W	Description
SP	MEM	0.81H	R/W	Stack pointer
SIO1TS	FLG	0.82H.3	R/W	Serial interface start flag
SIO1HIZ	FLG	0.82H.2	R/W	P0A <sub>1</sub> /SO <sub>1</sub> pin select flag
SIO1CK1	FLG	0.82H.1	R/W	Serial interface clock select flag
SIO1CK0	FLG	0.82H.0	R/W	Serial interface clock select flag
IFCG	FLG	0.84H.0	R	IF counter gate status flag
PLLUL	FLG	0.85H.0	R	PLL unlock FF flag
ADCCMP	FLG	0.86H.0	R	ADC judge flag
CE	FLG	0.87H.0	R	CE pin status flag
BTM1CK1	FLG	0.89H.3	R/W	Basic timer 0 clock select flag
BTM1CK0	FLG	0.89H.2	R/W	Basic timer 0 clock select flag
BTM0CK1	FLG	0.89H.1	R/W	Basic timer 1 clock select flag
BTM0CK0	FLG	0.89H.0	R/W	Basic timer 1 clock select flag
TMCK	FLG	0.8CH.0	R/W	12-bit timer clock select flag
TMOVF	FLG	0.8DH.0	R	Timer/counter overflow detector flag
TMRPT	FLG	0.8EH.2	R/W	12-bit timer mode select flag
TMRES	FLG	0.8EH.1	R/W	Timer/counter reset flag
TMEN	FLG	0.8EH.0	R/W	Timer/counter start/stop flag
KSEN	FLG	0.90H.2	R/W	Key source latch enable flag
LCDEN	FLG	0.90H.1	R/W	LCD enable flag
PYASEL	FLG	0.90H.0	R/W	Port YA select flag
P2HSEL	FLG	0.91H.3	R/W	Port 2H select flag
P2GSEL	FLG	0.91H.2	R/W	Port 2G select flag
P2FSEL	FLG	0.91H.1	R/W	Port 2F select flag
P2ESEL	FLG	0.91H.0	R/W	Port 2E select flag
IFCMD1	FLG	0.92H.3	R/W	IF counter mode select flag
IFCMD0	FLG	0.92H.2	R/W	IF counter mode select flag
IFCCK1	FLG	0.92H.1	R/W	IF counter clock select flag
IFCCK0	FLG	0.92H.0	R/W	IF counter clock select flag
PWM1SEL	FLG	0.93H.1	R/W	P0C <sub>1</sub> /PWM <sub>1</sub> pin select flag
PWM0SEL	FLG	0.93H.0	R/W	P0C <sub>0</sub> /PWM <sub>0</sub> pin select flag
ADCCH1	FLG	0.94H.1	R/W	A/D converter channel select flag
ADCCH0	FLG	0.94H.0	R/W	A/D converter channel select flag
BEEP1SEL	FLG	0.95H.1	R/W	P0B <sub>1</sub> /BEEP <sub>1</sub> pin select flag
BEEP0SEL	FLG	0.95H.0	R/W	P0B <sub>0</sub> /BEEP <sub>0</sub> pin select flag
KEYJ	FLG	0.96H.0	R	Key input judge flag



Symbol Name	Attribute	Value	R/W	Description
BTM0CY	FLG	0.97H.0	R	Basic timer 0 carry flag
IEG	FLG	0.9FH.0	R/W	INT pin interrupt edge select flag
PLLMD1	FLG	0.0A1H.1	R/W	PLL mode select flag
PLLMD0	FLG	0.0A1H.0	R/W	PLL mode select flag
IFCSTRT	FLG	0.0A3H.1	R/W	IF counter start flag
IFCRES	FLG	0.0A3H.0	R/W	IF counter reset flag
FCGCH1	FLG	0.0A4H.1	R/W	External gate counter channel select flag
FCGCH0	FLG	0.0A4H.0	R/W	External gate counter channel select flag
BEEP1CK1	FLG	0.0A5H.3	R/W	BEEP <sub>1</sub> clock select flag
BEEP1CK0	FLG	0.0A5H.2	R/W	BEEP <sub>1</sub> clock select flag
BEEP0CK1	FLG	0.0A5H.1	R/W	BEEP <sub>0</sub> clock select flag
BEEP0CK0	FLG	0.0A5H.0	R/W	BEEP <sub>0</sub> clock select flag
P1DGIO	FLG	0.0A7H.0	R/W	Port 1D group I/O select flag
IPSI01	FLG	0.0AFH.3	R/W	Serial interface interrupt enable flag
IPBTM1	FLG	0.0AFH.2	R/W	Basic timer 1 interrupt enable flag
IPTM	FLG	0.0AFH.1	R/W	12-bit timer interrupt enable flag
IP	FLG	0.0AFH.0	R/W	INT pin interrupt enable flag
PLLRFCCK3	FLG	0.0B1H.3	R/W	PLL reference clock select flag
PLLRFCCK2	FLG	0.0B1H.2	R/W	PLL reference clock select flag
PLLRFCCK1	FLG	0.0B1H.1	R/W	PLL reference clock select flag
PLLRFCCK0	FLG	0.0B1H.0	R/W	PLL reference clock select fla
P1ABIO2	FLG	0.0B5H.2	R/W	I/O select flag of P1A <sub>2</sub> pin
P1ABIO1	FLG	0.0B5H.1	R/W	I/O select flag of P1A <sub>1</sub> pin
P1ABIO0	FLG	0.0B5H.0	R/W	I/O select flag of P1A <sub>0</sub> pin
P0BBIO3	FLG	0.0B6H.3	R/W	I/O select flag of P0B <sub>3</sub> pin
P0BBIO2	FLG	0.0B6H.2	R/W	I/O select flag of P0B <sub>2</sub> pin
P0BBIO1	FLG	0.0B6H.1	R/W	I/O select flag of P0B <sub>1</sub> pin
P0BBIO0	FLG	0.0B6H.0	R/W	I/O select flag of P0B <sub>0</sub> pin
P0ABIO2	FLG	0.0B7H.2	R/W	I/O select flag of P0A <sub>2</sub> pin
P0ABIO1	FLG	0.0B7H.1	R/W	I/O select flag of P0A <sub>1</sub> pin
P0ABIO0	FLG	0.0B7H.0	R/W	I/O select flag of P0A <sub>0</sub> pin
IRQSIO1	FLG	0.0BCH.0	R/W	Serial interface interrupt request flag
IRQBTM1	FLG	0.0BDH.0	R/W	Basic timer 1 interrupt request flag
IRQTM	FLG	0.0BEH.0	R/W	12-bit timer interrupt request flag
INT	FLG	0.0BFH.3	R	INT pin interrupt status flag
IRQ	FLG	0.0BFH.0	R/W	INT pin interrupt request flag

**24.6 Peripheral Hardware Register**

Symbol Name	Attribute	Value	R/W	Description
ADCR	DAT	02H	R/W	A/D converter reference voltage setting register
SIO1SFR	DAT	03H	R/W	Serial interface presettable shift register
PWMR0	DAT	04H	R/W	PWM <sub>0</sub> data register
PWMR1	DAT	05H	R/W	PWM <sub>1</sub> data register
AR	DAT	40H	R/W	Address register
PLLr	DAT	41H	R/W	PLL data register
KSR	DAT	42H	R/W	Key source data register
PYA	DAT	42H	R/W	PYA group register
IFC	DAT	43H	R	IF counter data register
TMM	DAT	46H	R/W	Timer modulo register
TMC	DAT	47H	R	Timer counter

**24.7 Others**

Symbol Name	Attribute	Value	Description
DBF	DAT	0FH	Fixed operand value of PUT, GET, and MOVt instructions
IX	DAT	01H	Fixed operand value of INC instruction

## 25. ELECTRICAL SPECIFICATIONS

### Absolute Maximum Ratings (T<sub>A</sub> = 25°C)

Parameter	Symbol	Condition	Rating	Unit
Supply voltage	V <sub>DD</sub>		−0.3 to +6.0	V
Input voltage	V <sub>I</sub>		−0.3 to V <sub>DD</sub> + 0.3	V
Output voltage	V <sub>O</sub>	Other than P0C <sub>0</sub> to P0C <sub>3</sub>	−0.3 to V <sub>DD</sub> + 0.3	V
Output breakdown voltage	V <sub>BDS</sub>	P0C <sub>0</sub> to P0C <sub>3</sub>	14.0	V
Output current, high	I <sub>OH</sub>	Per pin	−12	mA
		Total for all pins	−20	mA
Output current, low	I <sub>OL</sub>	Per pin	15	mA
		Total for all pins	30	mA
Overall error	P <sub>t</sub>		200	mW
Operating ambient temperature	T <sub>A</sub>	When overall function operates	−40 to +85	°C
Storage temperature	T <sub>stg</sub>		−55 to +125	°C

**Caution** Product quality may suffer if the absolute maximum rating is exceeded even momentarily for any parameter. That is, the absolute maximum ratings are rated values at which the product is on the verge of suffering physical damage, and therefore the product must be used under conditions that ensure that the absolute maximum ratings are not exceeded.

### Recommended Operating Conditions

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Supply voltage	V <sub>DD1</sub>	When overall function operate	4.5	5.0	5.5	V
	V <sub>DD2</sub>	When PLL stops and CPU operates	3.5	5.0	5.5	V
Data retention voltage	V <sub>DDR</sub>	When crystal oscillation stops	2.3		5.5	V
Output breakdown voltage	V <sub>BDS</sub>	P0C <sub>0</sub> to P0C <sub>3</sub>			12.0	V
Operating ambient temperature	T <sub>A</sub>		−40		+85	°C
Supply voltage rise time	t <sub>rise</sub>	V <sub>DD</sub> : 0 → 4.5 V			500	ms

DC Characteristics (T<sub>A</sub> = -40 to +85°C, V<sub>DD</sub> = 4.5 to 5.5 V)

Parameter	Symbol	Conditions		MIN.	TYP.	MAX.	Unit
Supply current	I <sub>DD1</sub>	With CPU operating, PLL stopped, sine wave input to X <sub>IN</sub> pin (f <sub>IN</sub> = 4.5 MHz, V <sub>IN</sub> = V <sub>DD</sub> )			1.0	2.0	mA
	I <sub>DD2</sub>	With CPU operating, PLL stopped, sine wave input to X <sub>IN</sub> pin (f <sub>IN</sub> = 4.5 MHz, V <sub>IN</sub> = V <sub>DD</sub> ) HALT instruction used			0.5	1.0	mA
Data retention voltage	V <sub>DDR1</sub>	With crystal oscillation	Power failure detection by timer FF	3.5			V
	V <sub>DDR2</sub>	With crystal oscillation stopped	Power failure detection by timer FF	2.3			V
	V <sub>DDR3</sub>		Data memory retained	2.0			V
Data retention current	I <sub>DDR1</sub>	With crystal oscillation stopped	V <sub>DD</sub> = 5 V, T <sub>A</sub> = 25°C		2.0	4.0	μA
	I <sub>DDR2</sub>				2.0	20.0	μA
	I <sub>DDR3</sub>		V <sub>DD</sub> = 2.3 V, T <sub>A</sub> = 25°C		1.0	2.0	μA
	I <sub>DDR4</sub>		V <sub>DD</sub> = 2.3 V		1.0	10.0	μA
Intermediate-level output voltage	V <sub>OM</sub>	COM <sub>0</sub> to COM <sub>2</sub> V <sub>DD</sub> = 5.0 V		2.3		2.7	V
Input voltage, high	V <sub>IH1</sub>	P0A <sub>1</sub> , P0B <sub>0</sub> to P0B <sub>3</sub> , P1A <sub>0</sub> to P1A <sub>2</sub> , P1B <sub>0</sub> to P1B <sub>3</sub> , P1D <sub>0</sub> to P1D <sub>3</sub>		0.7V <sub>DD</sub>		V <sub>DD</sub>	V
	V <sub>IH2</sub>	P0A <sub>0</sub> , P0A <sub>2</sub> , CE, INT		0.8V <sub>DD</sub>		V <sub>DD</sub>	V
	V <sub>IH3</sub>	P0D <sub>0</sub> to P0D <sub>3</sub>		0.6V <sub>DD</sub>		V <sub>DD</sub>	V
Input voltage, low	V <sub>IL1</sub>	P0A <sub>1</sub> , P0B <sub>0</sub> to P0B <sub>3</sub> , P0D <sub>0</sub> to P0D <sub>3</sub> , P1A <sub>0</sub> to P1A <sub>2</sub> , P1B <sub>0</sub> to P1B <sub>3</sub> , P1D <sub>0</sub> to P1D <sub>3</sub>		0		0.2V <sub>DD</sub>	V
	V <sub>IL2</sub>	P0A <sub>0</sub> , P0A <sub>2</sub> , CE, INT		0		0.2V <sub>DD</sub>	V
Output current, high	I <sub>OH1</sub>	P0A <sub>0</sub> to P0A <sub>2</sub> , P0B <sub>0</sub> to P0B <sub>3</sub> , P1A <sub>0</sub> to P1A <sub>2</sub> , P1C <sub>0</sub> to P1C <sub>3</sub> , P1D <sub>0</sub> to P1D <sub>3</sub> V <sub>OH</sub> = V <sub>DD</sub> - 1 V		-1.0			mA
	I <sub>OH2</sub>	LCD <sub>0</sub> to LCD <sub>19</sub> , EO V <sub>OH</sub> = V <sub>DD</sub> - 1 V		-1.0			mA
Output current, low	I <sub>OL1</sub>	P0A <sub>0</sub> to P0A <sub>2</sub> , P0B <sub>0</sub> to P0B <sub>3</sub> , P1A <sub>0</sub> to P1A <sub>2</sub> , P1C <sub>0</sub> to P1C <sub>3</sub> , P1D <sub>0</sub> to P1D <sub>3</sub> V <sub>OL</sub> = 1 V		1.0			mA
	I <sub>OL2</sub>	LCD <sub>0</sub> to LCD <sub>19</sub> , EO V <sub>OL</sub> = 1 V		1.0			mA
	I <sub>OL3</sub>	P0C <sub>0</sub> to P0C <sub>3</sub> V <sub>OL</sub> = 1 V		1.0			mA
Input current, high	I <sub>IH1</sub>	VCOH pin pulled down V <sub>IH</sub> = V <sub>DD</sub>		10			mA
	I <sub>IH2</sub>	VCOL pin pulled down V <sub>IH</sub> = V <sub>DD</sub>		0.1			mA
	I <sub>IH3</sub>	X <sub>IN</sub> pin pulled down V <sub>IH</sub> = V <sub>DD</sub>		0.1			mA
	I <sub>IH4</sub>	P0D <sub>0</sub> to P0D <sub>3</sub> pin pulled down V <sub>IH</sub> = V <sub>DD</sub>		10		150	μA
Output off leakage current	I <sub>L1</sub>	P0C <sub>0</sub> to P0C <sub>3</sub> V <sub>OH</sub> = 12 V				1.0	μA
	I <sub>L2</sub>	EO V <sub>OH</sub> = V <sub>DD</sub> , V <sub>OL</sub> = 0 V				±1.0	μA

**AC Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 4.5$  to  $5.5$  V)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Operating frequency	$f_{IN1}$	VCOL pin, MF mode, sine wave input $V_{IN} = 0.15 V_{P-P}$	0.90		3.0	MHz
		VCOL pin, MF mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	0.50		20	MHz
	$f_{IN2}$	VCOL pin, HF mode, sine wave input $V_{IN} = 0.15 V_{P-P}$	5		25	MHz
		VCOL pin, HF mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	5		40	MHz
	$f_{IN3}$	VCOH pin, VHF mode, sine wave input $V_{IN} = 0.15 V_{P-P}$	60		130	MHz
		VCOH pin, VHF mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	30		250	MHz
	$f_{IN4}$	AMIFC pin, AMIF count mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	0.3		1.0	MHz
	$f_{IN5}$	AMIFC pin, AMIF count mode, sine wave input $V_{IN} = 0.1 V_{P-P}$	0.44		0.46	MHz
	$f_{IN6}$	FMIFC pin, FMIF count mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	5		15	MHz
	$f_{IN7}$	FMIFC pin, FMIF count mode, sine wave input $V_{IN} = 0.1 V_{P-P}$	10.5		10.9	MHz

**AD Converter Characteristics ( $T_A = -40$  to  $+85^\circ\text{C}$ ,  $V_{DD} = 4.5$  to  $5.5$  V)**

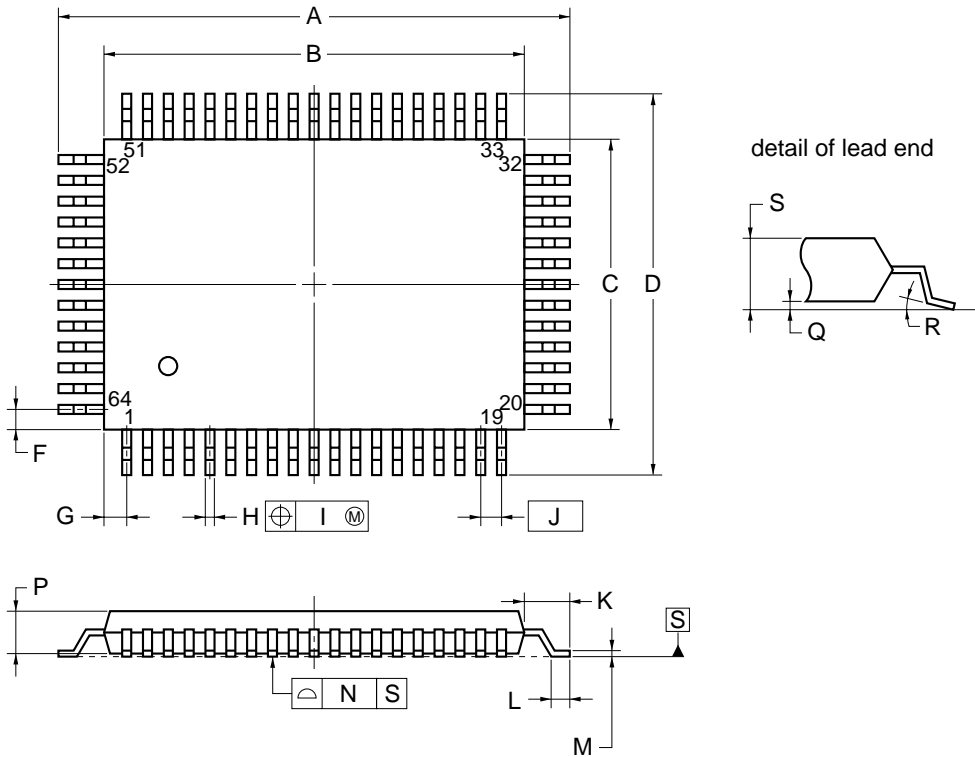
Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
A/D conversion resolution					6	bit
A/D conversion total error				$\pm 10$	$\pm 1.5$	LSB

**Reference Characteristics ( $T_A = +25^\circ\text{C}$ ,  $V_{DD} = 5.0$  V)**

Parameter	Symbol	Conditions	MIN.	TYP.	MAX.	Unit
Supply current	$I_{DD3}$	With CPU and PLL operating, sine wave input to VCOH pin ( $f_{IN} = 130$ MHz, $V_{IN} = 0.3 V_{P-P}$ )		12		mA
	$I_{DD4}$	With CPU and PLL operating, sine wave input to VCOH pin ( $f_{IN} = 250$ MHz, $V_{IN} = 0.3 V_{P-P}$ )		13		mA
Output current, high	$I_{OH3}$	COM <sub>0</sub> to COM <sub>2</sub> $V_{OH} = V_{DD} - 1$ V		-300		$\mu$ A
Output current, low	$I_{OL4}$	COM <sub>0</sub> to COM <sub>2</sub> $V_{OL} = 1$ V		300		$\mu$ A
Output current, intermediate	$I_{OM1}$	COM <sub>0</sub> to COM <sub>2</sub> $V_{OH} = V_{DD} - 1$ V		-25		$\mu$ A
	$I_{OM2}$	COM <sub>0</sub> to COM <sub>2</sub> $V_{OL} = 1$ V		25		$\mu$ A

26. PACKAGE DRAWINGS

64-PIN PLASTIC QFP (14x20)



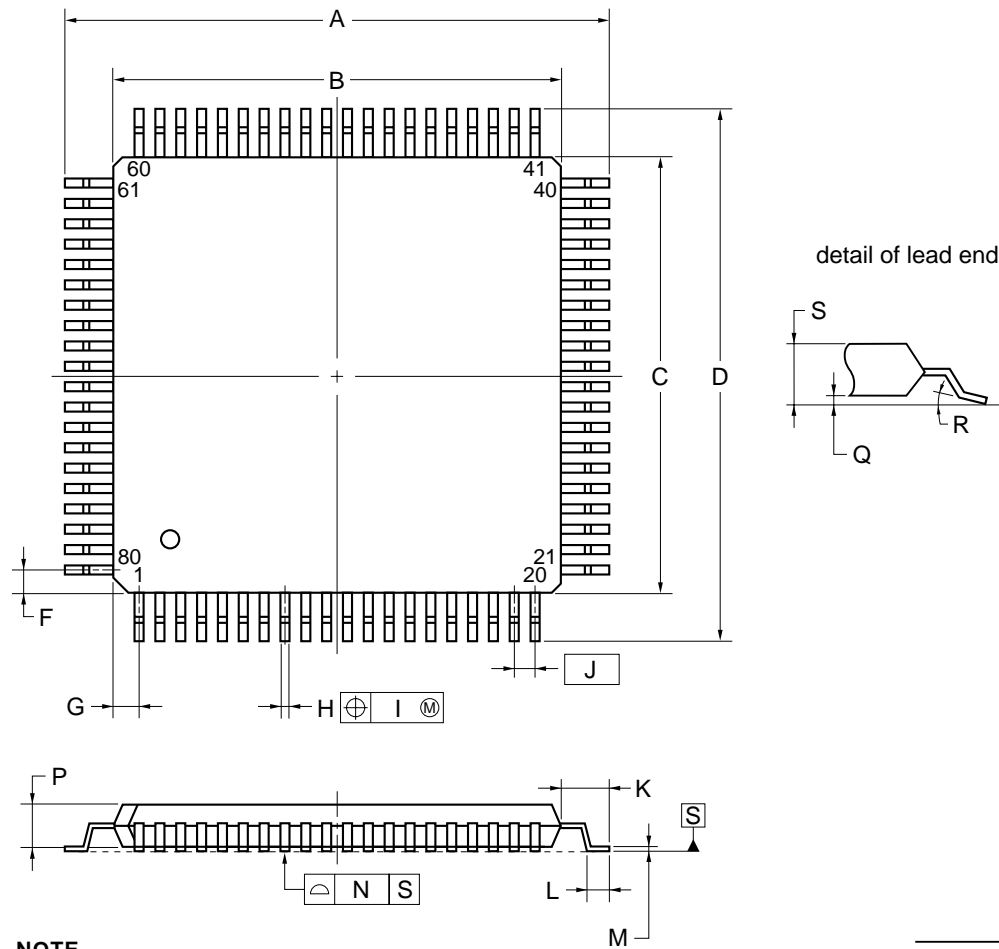
NOTE

Each lead centerline is located within 0.20 mm of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS
A	23.6±0.4
B	20.0±0.2
C	14.0±0.2
D	17.6±0.4
F	1.0
G	1.0
H	0.40±0.10
I	0.20
J	1.0 (T.P.)
K	1.8±0.2
L	0.8±0.2
M	0.15 <sup>+0.10</sup> <sub>-0.05</sub>
N	0.10
P	2.7±0.1
Q	0.1±0.1
R	5°±5°
S	3.0 MAX.

P64GF-100-3B8,3BE,3BR-4

## ★ 80-PIN PLASTIC QFP (14x14)



ITEM	MILLIMETERS
A	17.20±0.20
B	14.00±0.20
C	14.00±0.20
D	17.20±0.20
F	0.825
G	0.825
H	0.32±0.06
I	0.13
J	0.65 (T.P.)
K	1.60±0.20
L	0.80±0.20
M	0.17 <sup>+0.03</sup> <sub>-0.07</sub>
N	0.10
P	1.40±0.10
Q	0.125±0.075
R	3° <sup>+7°</sup> <sub>-3°</sub>
S	1.70 MAX.

P80GC-65-8BT-1

## 27. RECOMMENDED SOLDERING CONDITIONS

The μPD17012 should be soldered and mounted under the following recommended conditions.

For the details of the recommended soldering conditions, refer to the document **Semiconductor Device Mounting Technology Manual (C10535E)**.

For soldering methods and conditions other than those recommended, contact your NEC sales representative.

**Table 27-1. Surface Mounting Type Soldering Conditions (1/2)**

### (1) μPD17012GF-xxx-3BE: 64-pin plastic QFP (14 × 20 mm)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Twice or less, Exposure limit: 7 days <sup>Note</sup> (After 7 days, prebake at 125°C for 20 hours)	IR35-207-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Twice or less, Exposure limit: 7 days <sup>Note</sup> (After 7 days, prebake at 125°C for 20 hours)	VP15-207-2
Wave soldering	Soldering bath temperature: 260°C max., Time: 10 seconds max., Count: Once, Preheating temperature: 120°C max. (package surface temperature), Exposure limit: 7 days <sup>Note</sup> (After 7 days, prebake at 125°C for 20 hours)	WS60-207-1
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per pin row)	—

**Note** After opening the dry pack, store it below 25 °C and 65% RH for the allowable storage period.

**Caution** Do not use different soldering methods together (except for partial heating).

★

**Table 27-1. Surface Mounting Type Soldering Conditions (2/2)**

### (2) μPD17012GC-xxx-8BT: 80-pin plastic QFP (14 × 14 mm)

Soldering Method	Soldering Conditions	Recommended Condition Symbol
Infrared reflow	Package peak temperature: 235°C, Time: 30 seconds max. (at 210°C or higher), Count: Twice or less	IR35-00-2
VPS	Package peak temperature: 215°C, Time: 40 seconds max. (at 200°C or higher), Count: Twice or less	VP15-00-2
Wave soldering	Soldering bath temperature: 260°C max., Time: 10 seconds max., Count: Once, Preheating temperature: 120°C max. (package surface temperature)	WS60-00-1
Partial heating	Pin temperature: 300°C max., Time: 3 seconds max. (per pin row)	—

**Caution** Do not use different soldering methods together (except for partial heating).



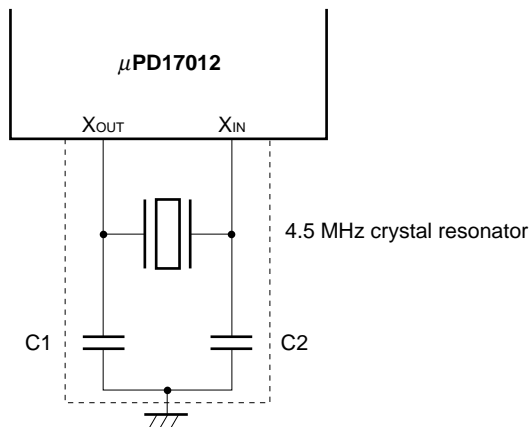
## APPENDIX A. NOTES ON CONNECTING CRYSTAL RESONATOR

When using the system clock oscillator, wire as follows in the area enclosed by the broken lines in the figure below to avoid an adverse effect from wiring capacitance.

- Keep the wiring length as short as possible.
- Do not cross the wiring with the other signal lines. Do not route the wiring near a signal line through which a high fluctuating current flows.
- Always make the ground point of the oscillator capacitor the same potential as GND. Do not ground the capacitor to a ground pattern through which a high current flows.
- Do not fetch signals from the oscillator.

Also caution is required for the following three points when connecting the capacitor and adjusting the operating frequency.

- <1> If capacitances C1 and C2 are too high, the oscillation startup characteristic may be degraded or the current consumption may rise.
- <2> The trimmer capacitor for adjusting the oscillation frequency is generally connected to the X<sub>IN</sub> pin. However, depending on the crystal resonator used, the oscillation stabilization may be affected (in this case, connect the trimmer capacitor to the X<sub>OUT</sub> pin). Therefore, oscillation should be evaluated by the crystal resonator that is actually being used.
- <3> Adjust the oscillation frequency while measuring the LCD drive waveform (83.3 Hz) or VCO oscillation frequency. If the probe is connected to the X<sub>OUT</sub> or X<sub>IN</sub> pin, the oscillation frequency cannot be measured correctly due to the probe capacitance.



## APPENDIX B. DEVELOPMENT TOOLS

The following development tools are available for program development of the μPD17012.

## Hardware

Name	Description
In-circuit emulator ( IE-17K, IE-17K-ET <sup>Note 1</sup> )	IE-17K and IE-17K-ET are in-circuit emulators that can be commonly used with any model in 17K Series. IE-17K and IE-17K-ET are connected to a host machine, which is a PC-9800 series or IBM PC/AT™, with RS232-C. When these in-circuit emulators are used in combination with the system evaluation board (SE board) dedicated to each model, they operate as emulators corresponding to that model. When human interface software <i>SIMPLEHOST</i> ® is used, a more sophisticated debugging environment can be created.
SE board (SE-17012)	SE-17012 is SE board for μPD17012 and 17P012. This SE board can be used alone to evaluate the system (SE-17012) or in combination with an in-circuit emulator for debugging.
Emulation probe (EP-17202GF)	EP-17202GF is an emulation probe for the 64-pin plastic QFP (GF-3BE type) of the μPD17012 and 17P012. The SE board and target system are connected when the EP-17202GF is used in combination with the EV-9200G-64 <sup>Note 2</sup> .
★ Emulation probe (EP-17K80GC)	EP-17K80GC is an emulation probe for the 80-pin plastic QFP (GC-8BT type) of the μPD17012 and 17P012. The SE board and target system are connected when the EP-17K80GC is used in combination with the EV-9200GC-80 <sup>Note 2</sup> .
Conversion socket (EV-9200G-64 <sup>Note 2</sup> )	EV-9200G-64 is a conversion socket for a 64-pin plastic QFP (GF-3BE type). It is used to connect the EP-17202GF to the target system.
★ Conversion socket (EV-9200GC-80 <sup>Note 2</sup> )	EV-9200GC-80 is a conversion socket for an 80-pin plastic QFP (GC-8BT type). It is used to connect the EP-17K80GC to the target system.
PROM programmer ( AF-9703 <sup>Note 3</sup> , AF-9704 <sup>Note 3</sup> , AF-9705 <sup>Note 3</sup> , AF-9706 <sup>Note 3</sup> )	AF-9703, AF-9704, AF-9705, and AF-9706 are PROM programmers corresponding to μPD17P012. They can program the μPD17P012 when connected to program adapters AF-9776B and PA-17P012GC.
Program adapter (AF-9776B <sup>Note 3</sup> )	AF-9776B is an adapter for programming the 64-pin plastic QFP (GF-3BE type) of the μPD17P012. It is used in combination with the AF-9703, AF9704, AF-9705, or AF-9706.
★ Program adapter (PA-17P012GC)	PA-17P012GC is an adapter for programming the 80-pin plastic QFP (GC-8BT type) of the μPD17P012. It is used in combination with the AF-9703, AF9704, AF-9705, or AF-9706.

**Notes** 1. Low-cost model: external power supply type

2. One EV-9200G-80 is provided with the EP-17003GF. Five EV-9200G-80s are also available as a set. One EV-9200GC-80 is provided with the EP-17K80GC. Five EV-9200GC-80s are also available as a set.

3. These are products of Ando Electric Co., Ltd. For details, consult Ando Electric Co, Ltd. (TEL: +81-3-3733-1166).

## Software

Name	Outline	Host Machine	OS	Supply Media	Order Code
17K assembler (RA17K)	RA17K is an assembler common to the 17K Series products. To develop the program of the $\mu$ PD17012, the RA17K is used in combination with the device file.	PC-9800 series	Japanese Windows™	3.5"2HD	$\mu$ SAA13RA17K
		IBM PC/AT-compatible	Japanese Windows	3.5"2HC	$\mu$ SAB13RA17K
			English Windows		$\mu$ SBB13RA17K
Device file (AS17012)	AS17012 is a device file for $\mu$ PD17012 and $\mu$ PD17P012. It is used in combination with an assembler common to the 17K Series (RA17K).	PC-9800 series	Japanese Windows	3.5"2HD	$\mu$ SAA13AS17012
		IBM PC/AT-compatible	Japanese Windows	3.5"2HC	$\mu$ SAB13AS17012
			English Windows		$\mu$ SBB13AS17012
Support software (SIMPLEHOST)	SIMPLEHOST is software that serves as a human interface on Windows for program development using an in-circuit emulator and personal computer.	PC-9800 series	Japanese Windows	3.5"2HD	$\mu$ SAA13ID17K
		IBM PC/AT-compatible	Japanese Windows	3.5"2HC	$\mu$ SAB13ID17K
			English Windows		$\mu$ SBB13ID17K

**NOTES FOR CMOS DEVICES****① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS**

Note:

Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

**② HANDLING OF UNUSED INPUT PINS FOR CMOS**

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to  $V_{DD}$  or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

**③ STATUS BEFORE INITIALIZATION OF MOS DEVICES**

Note:

Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

## Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

### **NEC Electronics Inc. (U.S.)**

Santa Clara, California  
Tel: 408-588-6000  
800-366-9782  
Fax: 408-588-6130  
800-729-9288

### **NEC Electronics (Germany) GmbH**

Duesseldorf, Germany  
Tel: 0211-65 03 02  
Fax: 0211-65 03 490

### **NEC Electronics (UK) Ltd.**

Milton Keynes, UK  
Tel: 01908-691-133  
Fax: 01908-670-290

### **NEC Electronics Italiana s.r.l.**

Milano, Italy  
Tel: 02-66 75 41  
Fax: 02-66 75 42 99

### **NEC Electronics (Germany) GmbH**

Benelux Office  
Eindhoven, The Netherlands  
Tel: 040-2445845  
Fax: 040-2444580

### **NEC Electronics (France) S.A.**

Velizy-Villacoublay, France  
Tel: 01-30-67 58 00  
Fax: 01-30-67 58 99

### **NEC Electronics (France) S.A.**

Spain Office  
Madrid, Spain  
Tel: 91-504-2787  
Fax: 91-504-2860

### **NEC Electronics (Germany) GmbH**

Scandinavia Office  
Taeby, Sweden  
Tel: 08-63 80 820  
Fax: 08-63 80 388

### **NEC Electronics Hong Kong Ltd.**

Hong Kong  
Tel: 2886-9318  
Fax: 2886-9022/9044

### **NEC Electronics Hong Kong Ltd.**

Seoul Branch  
Seoul, Korea  
Tel: 02-528-0303  
Fax: 02-528-4411

### **NEC Electronics Singapore Pte. Ltd.**

United Square, Singapore 1130  
Tel: 65-253-8311  
Fax: 65-250-3583

### **NEC Electronics Taiwan Ltd.**

Taipei, Taiwan  
Tel: 02-2719-2377  
Fax: 02-2719-5951

### **NEC do Brasil S.A.**

Electron Devices Division  
Rodovia Presidente Dutra, Km 214  
07210-902-Guarulhos-SP Brasil  
Tel: 55-11-6465-6810  
Fax: 55-11-6465-6829

J99.1

**SIMPLEHOST is a trademark of NEC Corporation.**

**Windows is either a registered trademark or a trademark of Microsoft Corporation in the United States and/or other countries.**

**PC/AT is a trademark of International Business Machines Corporation.**

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

- **The information in this document is subject to change without notice. Before using this document, please confirm that this is the latest version.**
  - 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.
  - Descriptions of circuits, software, and other related information in this document are provided for illustrative purposes in semiconductor product operation and application examples. The incorporation of these circuits, software, and information in the design of the customer's equipment shall be done under the full responsibility of the customer. NEC Corporation assumes no responsibility for any losses incurred by the customer or third parties arising from the use of these circuits, software, and information.
  - While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.
  - NEC devices are classified into the following three quality grades:  
"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.
    - Standard: Computers, office equipment, communications equipment, test and measurement equipment, audio and visual equipment, home electronic appliances, machine tools, personal electronic equipment and industrial robots
    - Special: Transportation equipment (automobiles, trains, ships, etc.), traffic control systems, anti-disaster systems, anti-crime systems, safety equipment and medical equipment (not specifically designed for life support)
    - Specific: Aircraft, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.
- The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.