

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

The μ PD17933 and 17934 are 4-bit single-chip CMOS microcontrollers with hardware for digital tuning systems (DTSs).

- ★ These microcontrollers integrate a prescaler that operates at a voltage as low as 1.05 V and up to 220 MHz, a PLL frequency synthesizer, an intermediate frequency (IF) counter, and an LCD controller/driver on a single chip.

Therefore, a high-performance digital tuning system for a portable set can be organized with a single chip.

FEATURES

- Program memory (ROM)
 - μ PD17933 : 12K bytes (6144×16 bits)
 - μ PD17934 : 16K bytes (8192×16 bits)
- General-purpose data memory (RAM))
448 \times 4 bits
- Instruction execution time
53.3 μ s (with 75-kHz crystal resonator)
- PLL frequency synthesizer
Dual modulus prescaler (220 MHz MAX.),
programmable divider, phase comparator,
charge pump
- Peripheral hardware
General-purpose I/O ports, LCD controller/driver,
serial interface, A/D converter, BEEP output,
frequency counter
- Interrupt
External: 1 source
Internal : 3 sources
- Reset by $\overline{\text{RESET}}$ pin
- Low power consumption
- Supply voltage: $V_{DD} = 1.05$ to 1.8 V

ORDERING INFORMATION

Part Number	Package
μ PD17933GK-xxx-BE9	80-pin plastic TQFP (12 \times 12 mm, 0.5 mm pitch)
μ PD17934GK-xxx-BE9	80-pin plastic TQFP (12 \times 12 mm, 0.5 mm pitch)

Remark xxx indicates ROM code suffix.

Unless otherwise specified, the μ PD17934 is explained as the representative model in this document.

FUNCTIONAL OUTLINE

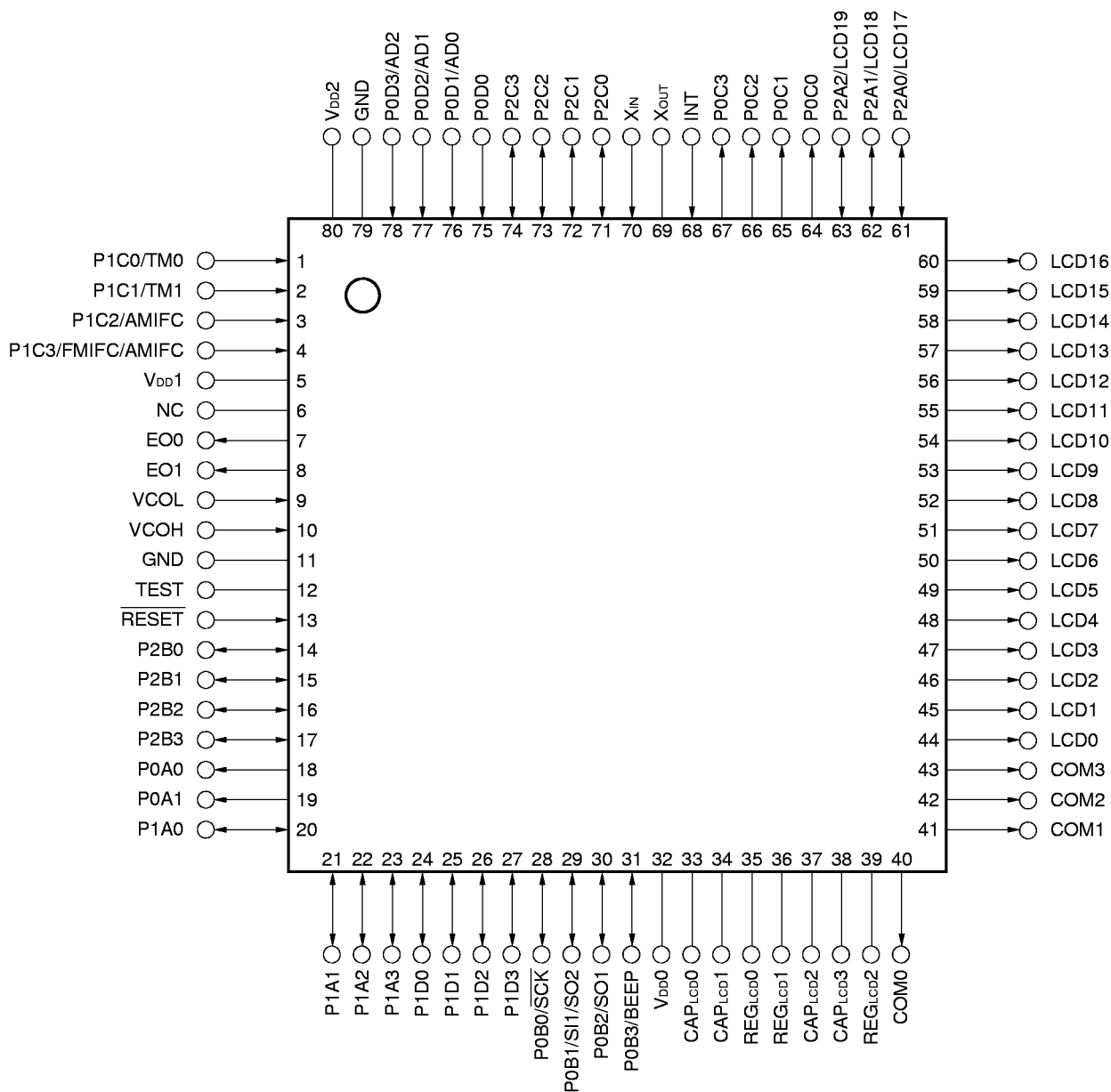
Part Number		μ PD17933	μ PD17934
Program memory (ROM)		12K bytes (6144 x 16 bits)	16K bytes (8192 x 16 bits)
General-purpose data memory (RAM)		448 x 4 bits	
Instruction execution time		53.3 μ s (with 75-MHz crystal oscillator)	
General-purpose port		37 pins <ul style="list-style-type: none"> • I/O port : 20 pins • Input port : 11 pins (of which 3 are muxed with LCD segment pins) • Output port : 6 pins 	
Stack level		<ul style="list-style-type: none"> • Address stack : 15 levels (stack can be manipulated) • Interrupt stack: 4 levels (stack can be manipulated) 	
Vector interrupt (maskable interrupt)		<ul style="list-style-type: none"> • External : 1 source (INT) • Internal : 3 sources (basic timer 0, 8-bit timer, serial interface) 	
Timer		3 channels <ul style="list-style-type: none"> • Basic timer 0 (125 ms) • Basic timer 1 (8 ms, 32 ms) • 8-bit timer (with event counter) 	
A/D converter		8-bits resolution x 3 channels	
LCD controller/driver		<ul style="list-style-type: none"> • 20 segments, 4 commons • 1/4 duty, 1/2 bias, frame frequency: 62.5 Hz, drive voltage $V_{LCD1}=3.0$ V TYP. • Muxed segment pins: 3 (Each can be used as general-purpose input port pin.) 	
Serial interface		1 channel (3-wire/2-wire modes selectable)	
PLL frequency synthesizer	Division mode	2 types <ul style="list-style-type: none"> • Direct division mode (VCOL pin) • Pulse swallow mode (VCOL pin/VCOH pin) 	
	Reference frequency	6 types selectable (1, 3, 5, 6.25, 12.5, 25 kHz)	
	Charge pump	Error out output: 2 pins (EO0 and EO1 pins)	
	Phase comparator	Unlock detectable by program	
★	Intermediate frequency (IF) counter	Frequency measurement <ul style="list-style-type: none"> • AMIFC pin: 400 to 500 kHz • FMIFC pin: 10 to 11 MHz 	
BEEP output		1 pin (1.5 kHz, 3 kHz)	
Reset		<ul style="list-style-type: none"> • Reset by $\overline{\text{RESET}}$ pin • Watchdog timer reset Can be set only once on power application: 4096 or 8192 instructions selectable • Stack pointer overflow/underflow reset Can be set only once on power application: Interrupt stack and address stack selectable 	
★	Supply voltage	$V_{DD} = 1.05$ to 1.8 V	
Package		80-pin plastic TQFP (12 x 12 mm, 0.5 mm pitch)	

PIN CONFIGURATION (Top View)

80-pin plastic TQFP (12 × 12 mm, 0.5 mm pitch)

μPD17933GK-xxx-BE9

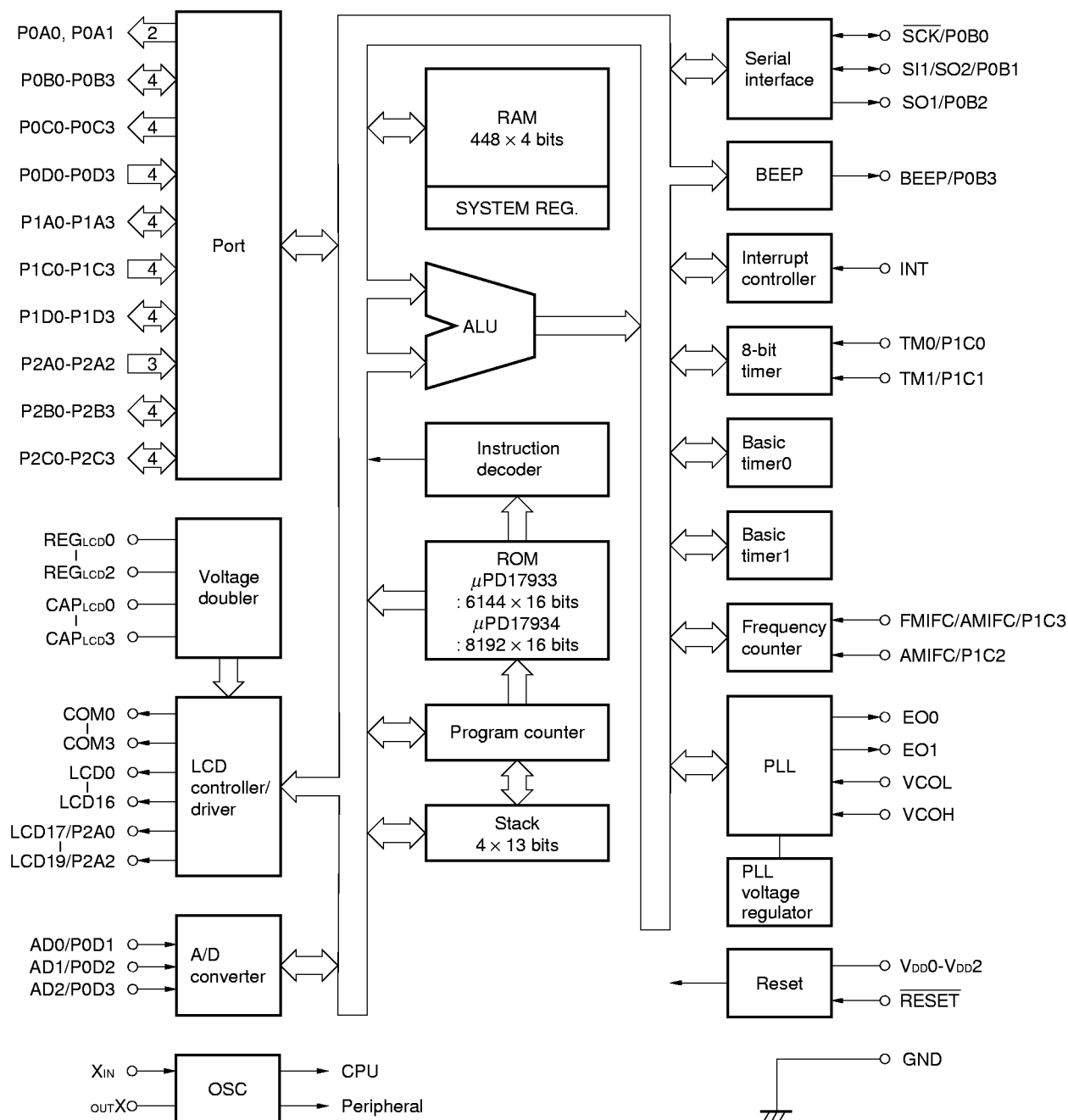
μPD17934GK-xxx-BE9



PIN NAME

AD0-AD2	: A/D converter inputs	P0D0-P0D3	: Port 0D
AMIFC	: Intermediate frequency counter input for AM	P1A0-P1A3	: Port 1A
BEEP	: BEEP output	P1C0-P1C3	: Port 1C
CAP _{LCD0} -CAP _{LCD3}	: Capacitor connection for LCD drive voltage	P1D0-P1D3	: Port 1D
COM0-COM3	: LCD common output	P2A0-P2A2	: Port 2A
EO0, EO1	: Error output	P2B0-P2B3	: Port 2B
FMIFC	: Intermediate frequency counter input for FM	P2C0-P2C3	: Port 2C
GND	: Ground	REG _{LCD0} -REG _{LCD2}	: Regulator output for LCD drive
INT	: External interrupt input	RESET	: Reset input
LCD0-LCD19	: LCD segment outputs	SCK	: 3-wire serial clock I/O
★ NC	: No connection	SI1	: 3-wire serial data input
P0A0, P0A1	: Port 0A	SO1, SO2	: 3-wire serial data outputs
P0B0-P0B3	: Port 0B	TEST	: Test input
P0C0-P0C3	: Port 0C	TM0, TM1	: Timer event inputs
		VCOH, VCOL	: Local oscillation inputs for PLL
		V _{DD0} -V _{DD2}	: Power supply
		X _{IN} , X _{OUT}	: Crystal resonator connection

BLOCK DIAGRAM



CONTENTS

1. PIN FUNCTIONS	11
1.1 Pin Function List	11
1.2 Equivalent Circuits of Pins	14
1.3 Recommended Connections of Unused Pins	18
1.4 Cautions on Using TEST Pin	19
2. PROGRAM MEMORY (ROM)	20
2.1 Outline of Program Memory	20
2.2 Program Memory	21
2.3 Program Counter	22
2.4 Flow of Program	22
2.5 Cautions on Using Program Memory	25
3. ADDRESS STACK (ASK)	26
3.1 Outline of Address Stack	26
3.2 Address Stack Register (ASR)	26
3.3 Stack Pointer (SP)	28
3.4 Operation of Address Stack	29
3.5 Cautions on Using Address Stack	30
4. DATA MEMORY (RAM)	31
4.1 Outline of Data Memory	31
4.2 Configuration and Function of Data Memory	33
4.3 Data Memory Addressing	35
4.4 Cautions on Using Data Memory	36
5. SYSTEM REGISTERS (SYSREG)	37
5.1 Outline of System Registers	37
5.2 System Register List	38
5.3 Address Register (AR)	39
5.4 Window Register (WR)	41
5.5 Bank Register (BANK)	42
5.6 Index Register (IX) and Data Memory Row Address Pointer (MP: memory pointer)	43
5.7 General Register Pointer (RP)	45
5.8 Program Status Word (PSWORD)	47
6. GENERAL REGISTER (GR)	49
6.1 Outline of General Register	49
6.2 General Register	49
6.3 Generating Address of General Register by Each Instruction	50
6.4 Cautions on Using General Register	50

7. ALU (Arithmetic Logic Unit) BLOCK	51
7.1 Outline of ALU Block	51
7.2 Configuration and Function of Each Block	52
7.3 ALU Processing Instruction List	52
7.4 Cautions on Using ALU	56
8. REGISTER FILE (RF) AND CONTROL REGISTERS	57
8.1 Outline of Register File	57
8.2 Configuration and Function of Register File	58
8.3 Control Registers and Input/Output Selection Registers	59
8.4 LCD Segment Registers	69
8.5 Cautions on Using Control Register	69
9. DATA BUFFER (DBF)	70
9.1 Outline of Data Buffer	70
9.2 Data Buffer	71
9.3 Relationships between Peripheral Hardware and Data Buffer	72
9.4 Cautions on Using Data Buffer	76
10. DATA BUFFER STACK	77
10.1 Outline of Data Buffer Stack	77
10.2 Data Buffer Stack Register	77
10.3 Data Buffer Stack Pointer	79
10.4 Operation of Data Buffer Stack	80
10.5 Using Data Buffer Stack	81
10.6 Cautions on Using Data Buffer Stack	81
11. GENERAL-PURPOSE PORT	82
11.1 Outline of General-purpose Port	82
11.2 General-Purpose I/O Port (P0B, P1A, P1D, P2B, P2C)	84
11.3 General-Purpose Input Port (P0D, P1C, P2A)	93
11.4 General-Purpose Output Port (P0A, P0C)	96
12. INTERRUPT	98
12.1 Outline of Interrupt Block	98
12.2 Interrupt Control Block	100
12.3 Interrupt Stack Register	106
12.4 Stack Pointer, Address Stack Registers, and Program Counter	110
12.5 Interrupt Enable Flip-Flop (INTE)	110
12.6 Accepting Interrupt	111
12.7 Operations after Interrupt Has Been Accepted	116
12.8 Returning from Interrupt Routine	116
12.9 External Interrupts (INT pin)	117
12.10 Internal Interrupts	119

13. TIMERS	120
13.1 Outline of Timers	120
13.2 Basic Timer 0	121
13.3 Basic Timer 1	125
13.4 Timer 0	131
14. A/D CONVERTER	138
14.1 Outline of A/D Converter	138
14.2 Input Selection Block	139
14.3 Compare Voltage Generation and Compare Blocks	141
14.4 Comparison Timing Chart	143
14.5 Using A/D Converter	144
14.6 Cautions on Using A/D Converter	148
14.7 Status at Reset	148
15. SERIAL INTERFACE	149
15.1 General	149
15.2 Clock Input/Output Control Block and Data Input/Output Control Block	150
15.3 Clock Control Block	153
15.4 Clock Counter	153
15.5 Presetable Shift Register	154
15.6 Wait Control Block	154
15.7 Serial Interface Operation	155
15.8 Notes on Setting and Reading Data	159
15.9 Operation Mode and Operational Outline of Each Blocks	160
15.10 Status on Reset	162
16. PLL FREQUENCY SYNTHESIZER	163
16.1 Outline of PLL Frequency Synthesizer	163
16.2 Input Selection Block and Programmable Divider	164
16.3 Reference Frequency Generator	168
16.4 Phase Comparator (f-DET), Charge Pump, and Unlock FF	170
16.5 PLL Disabled Status	174
16.6 Using PLL Frequency Synthesizer	175
16.7 Status at Reset	179
17. INTERMEDIATE FREQUENCY (IF) COUNTER	180
17.1 Outline of Frequency Counter	180
17.2 IF Counter Input Selection Block and Gate Time Control Block	181
17.3 Start/Stop Control Block and IF Counter	183
17.4 Using IF Counter	189
17.5 Status at Reset	191
18. BEEP	192
18.1 Outlines of BEEP	192
18.2 Output Wave Form of BEEP	194
18.3 Status at Reset	195

19. LCD CONTROLLER/DRIVER	196
19.1 Outline of LCD Controller/Driver	196
19.2 LCD Drive Voltage Generation Block	197
19.3 LCD Segment Register	198
19.4 Segment Signal/General-Purpose Input Port Select Block	200
19.5 Common Signal Output and Segment Signal Output Timing Control Blocks	202
19.6 Common Signal and Segment Signal Output Waves	203
19.7 Using LCD Controller/Driver	205
19.8 Status at Reset	207
20. STANDBY	208
20.1 Outline of Standby Function	208
20.2 Halt Function	209
20.3 Clock Stop Function	215
20.4 Device Operation in Halt and Clock Stop Status	217
20.5 Cautions on Processing of Each Pin in Halt and Clock Stop Status	217
21. RESET	220
21.1 Outline of Reset	220
21.2 Reset by RESET Pin	221
21.3 WDT&SP Reset	222
22. INSTRUCTION SET	228
22.1 Outline of Instruction Set	228
22.2 Legend	229
22.3 Instruction List	230
22.4 Assembler (RA17K) Embedded Macro Instruction	232
23. RESERVED SYMBOLS	233
23.1 Data Buffer (DBF)	233
23.2 System Registers (SYSREG)	233
23.3 LCD Segment Register	234
23.4 Port Register	235
23.5 Register File (Control Register)	236
23.6 Peripheral Hardware Register	238
23.7 Others	238
24. ELECTRICAL CHARACTERISTICS	239
25. PACKAGE DRAWING	243
★ 26. RECOMMENDED SOLDERING CONDITIONS	244
APPENDIX A. CAUTIONS ON CONNECTING CRYSTAL RESONATOR	245
APPENDIX B. DEVELOPMENT TOOLS	246

MEMO

1. PIN FUNCTIONS

1.1 Pin Function List

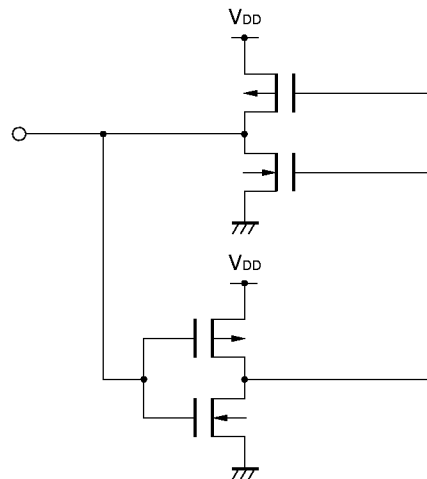
Pin No.	Symbol	Function	Output Form
1 2 3 4	PIC0/TM0 P1C1/TM1 P1C2/AMIFC P1C3/FMIFC/ AMIFC	<div>Port 1C, timer event input pins, IF counter (for frequency count) input pin for the AM/FM.</div> <div><div><div>• PIC0 through P1C3</div><div>4-bit input port</div></div><div><div>• TM0 and TM1</div><div>Timer event input pins</div></div><div><div>• AMIFC</div><div>IF counter input pin for AM</div></div><div><div>• FMIFC</div><div>IF counter input pin for FM</div></div></div> <div><div>At reset</div><div>Reset by $\overline{\text{RESET}}$ pin</div><div>Input (P1C0-P1C3)</div></div> <div><div>On clock stop</div><div>WDT&SP reset</div><div>Input (P1C0-P1C3)</div></div>	
80 5 32	V _{DD2} V _{DD1} V _{DD0}	<div>Power supply pins. Supply the same voltage to these pins.</div> <div>V_{DD2}: Supplies power to the comparator and peripherals of the A/D converter, and to the IF counter.</div> <div>V_{DD1}: Supplies power to the PLL.</div> <div>V_{DD0}: Supplies power to all the internal circuits except the above.</div>	—
6	NC	No connection	—
7 8	EO0 EO1	<div>Output from the charge pump of the PLL frequency synthesizer.</div> <div>These pins output the result of comparing the phases of the divided frequency of local oscillation with the reference frequency.</div> <div><div>At reset</div><div>Reset by $\overline{\text{RESET}}$ pin</div><div>High-impedance output</div></div> <div><div>On clock stop</div><div>WDT&SP reset</div><div>High-impedance output</div></div>	CMOS 3-state
9 10	VCOL VCOH	<div>Input of the local oscillation (VCO) frequency of the PLL.</div> <div><div>• VCOL</div><div>• Active when HF or MF mode selected by program; otherwise, pulled down.</div></div> <div><div>• VCOH</div><div>• Active when VHF mode selected by program; otherwise, pulled down.</div></div> <div>Because these input pins are connected to an internal AC amplifier, cut the DC component of the input signal with a capacitor.</div>	—
11 79	GND	Ground	—
12	TEST	Test input pin. Be sure to connect this pin directly to GND.	—
13	$\overline{\text{RESET}}$	Reset input	—

Pin No.	Symbol	Function			Output Form
14 17	P2B0 P2B3	These pins form a 4-bit I/O port which can be set in input or output mode in 1-bit units.			CMOS push-pull
	At reset		On clock stop		
	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset			
	Input	Input	Retained		
18 19	P0A0 P0A1	These pins form a 2-bit output port.			N-ch open- drain
	At reset		On clock stop		
	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset			
	Output low level	Output low level	Retained		
20 23	P1A0 P1A3	These pins form a 4-bit I/O port which can be set in input or output mode in 1-bit units.			N-ch open- drain
	At reset		On clock stop		
	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset			
	Input	Input	Retained		
24 27	P1D0 P1D3	These pins form a 4-bit I/O port which can be set in input or output mode in 1-bit units.			N-ch open- drain
	At reset		On clock stop		
	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset			
	Input	Input	Retained		
28 31	P0B0/ $\overline{\text{SCK}}$ P0B1/SI1/SO2 P0B2/SO1 P0B3/BEEP	Port P0B, the I/O pins of the serial interface and BEEP output pin. <ul style="list-style-type: none">P0B3 through P0B0<ul style="list-style-type: none">4-bit I/O portCan be set in input or output mode in 1-bit units.BEEP<ul style="list-style-type: none">BEEP outputSO1, SO2, SI1 Serial data output pins and serial data input pin when the 3-wire or 2-wire serial I/O mode of serial interface 1 is selected.$\overline{\text{SCK}}$<ul style="list-style-type: none">Serial clock I/O			CMOS push-pull
	At reset		On clock stop		
	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset			
	Input (P0B3-P0B0)	Input (P0B3-P0B0)	Retained		
33 34 37 38	CAP _{LCD0} CAP _{LCD1} CAP _{LCD2} CAP _{LCD3}	These pins can be used to connect capacitors for a double circuit to generate power to drive an LCD. Connect 0.33 μ F capacitors between CAP _{LCD0} and CAP _{LCD1} pins and between CAP _{LCD2} and CAP _{LCD3} pins.			—
35 36 39	REG _{LCD0} REG _{LCD1} REG _{LCD2}	Regulator output pins of power supply for LCD drive. Connect to GND with a 0.1 μ F capacitor.			—

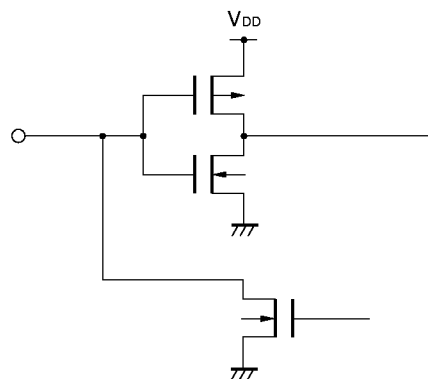
Pin No.	Symbol	Function			Output Form
40	COM0	These pins output the common signals of the LCD controller/driver.			CMOS
		At reset		On clock stop	3-state
43	COM3	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset		
		Output low level	Output low level	Output low level	
44	LCD0	These pins are the segment signal output pins of the LCD controller/driver.			CMOS
					push-pull
60	LCD16				
61	P2A0/LCD17	Port 2A input port, the segment signal output pins of the LCD controller/driver.			–
	P2A1/LCD18	• P2A0 through P2A2			
63	P2A2/LCD19	3-bit input port			
		• LCD17 through LCD19			
		LCD segment output			
		At reset		On clock stop	
		Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset		
		Input (P2A2-P2A0)	Input (P2A2-P2A0)	Retained	
64	P0C0	4-bit output			CMOS
		At reset		On clock stop	push-pull
67	P0C3	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset		
		Output low level	Output low level	Retained	
68	INT	Edge-detected vector interrupt input. Rising or falling edge is selectable.			–
69	X _{OUT}	Crystal resonator connection pin.			–
70	X _{IN}				
71	P2C0	These pins form a 4-bit I/O port which can be set in the input or output mode in 1-bit units.			CMOS
		At reset		On clock stop	push-pull
74	P2C3	Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset		
		Input	Input	Retained	
75	P0D0	Port 0D input port, A/D converter input pins and to release the HALT and STOP modes.			–
	P0D1/AD0	• P0D0 through P0D3			
78	P0D2/AD1	4-bit input port			
	P0D3/AD2	• AD0 through AD2			
		Analog input of A/D converter			
		• HALT and STOP mode releasing			
		Releases HALT or STOP mode when a high level is input.			
		At reset		On clock stop	
		Reset by $\overline{\text{RESET}}$ pin	WDT&SP reset		
		Input with pull-down resistor (P0D3-P0D0)	Input with pull-down resistor (P0D3-P0D0)	Retained	

1.2 Equivalent Circuits of Pins

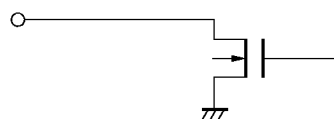
- (1) P0B (P0B3/BEEP, P0B2/SO1, P0B1/SI1/SO2, P0B0/ $\overline{\text{SCK}}$) } (I/O)
 P2B (P2B3, P2B2, P2B1, P2B0)
 P2C (P2C3, P2C2, P2C1, P2C0)



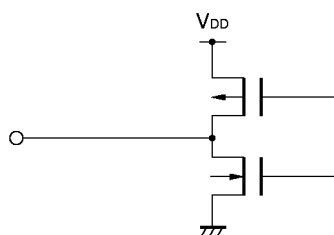
- (2) P1A (P1A3, P1A2, P1A1, P1A0) } (I/O)
 P1D (P1D3, P1D2, P1D1, P1D0)



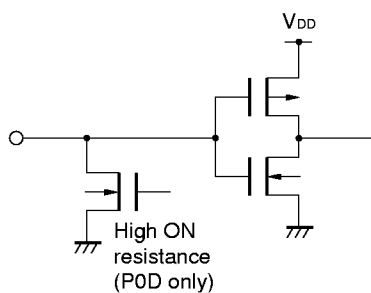
- (3) P0A (P0A1, P0A0) (Output)



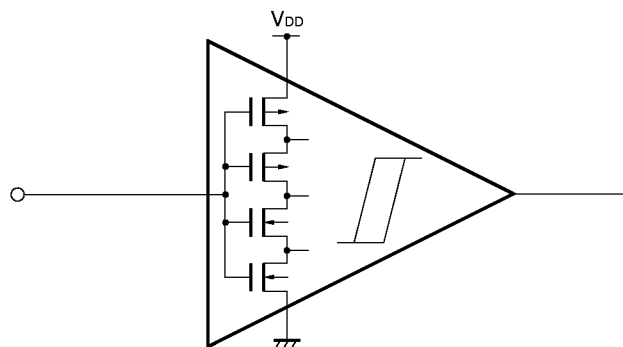
(4) P0C (P0C3, P0C2, P0C1, P0C0) (Output)



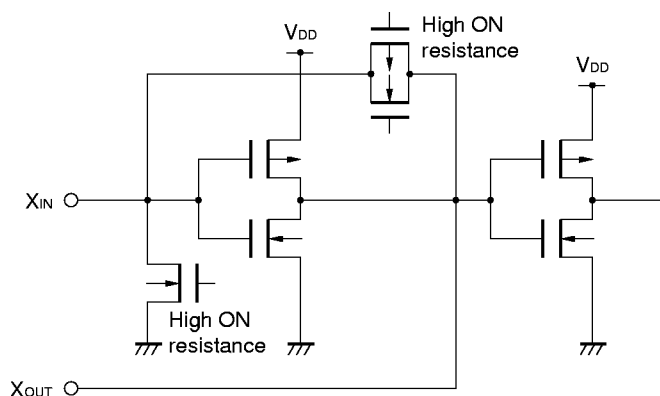
★ (5) P0D (P0D3/AD2, P0D2/AD1, P0D1/AD0, P0D0)
P1C (P1C3/FMIFC/AMIFC, P1C2/AMIFC, P1C1/TM1, PIC0/TM0) } (Input)
P2A (P2A2/LCD19, P2A1/LCD18, P2A0/LCD17)



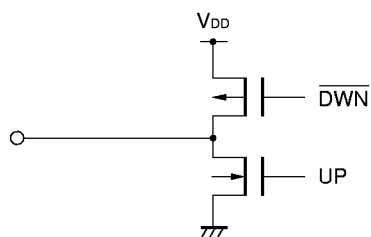
(6) $\overline{\text{INT}}$ } (Schmitt trigger input)
 $\overline{\text{RESET}}$



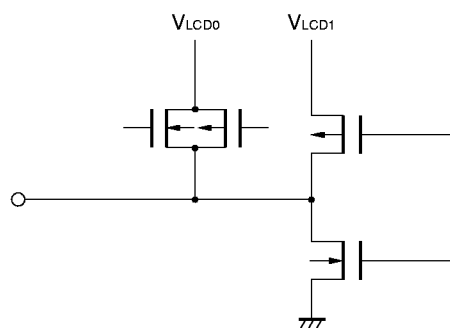
(7) X_{OUT} (Output), X_{IN} (Input)



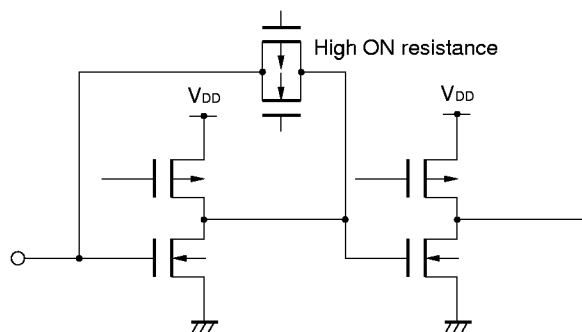
(8) EO1, EO0 (Output)



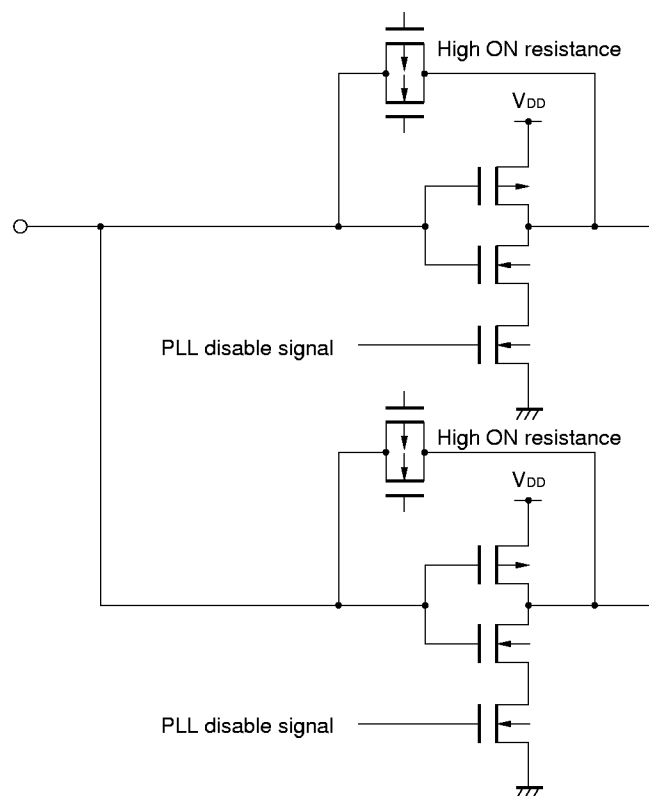
(9) COM3 through COM0 (Output)



(10) VCOH (Input)



(11)VCOL (Input)



1.3 Recommended Connections of Unused Pins

It is recommended to connect the unused pins as follows:

Table 1-1. Recommended Connections of Unused Pins

	Pin Name	I/O Form	Recommended Connection
★	Port pins		
	P0D3/AD2-P0D1/AD0,P0D0	Input	Connect each pin to GND via resistor ^{Note 1} .
	P1C3/FMIFC/AMIFC ^{Note 2}		
	P1C2/AMIFC ^{Note 2}		
	P1C1/TM1		Set each pin in port mode and connect it to V _{DD} or GND via resistor.
	P1C0/TM0		
	P0A1, P0A0	Output	Set in low-level output mode via software and leave unconnected.
	P0C3-P0C0		
	P0B3/BEEP	I/O ^{Note 3}	Set in general-purpose input port mode via software, and connect each pin to V _{DD} or GND via resistor.
	P0B2/SO1		
	P0B1/SI1/SO2		
	P0B0/ $\overline{\text{SCK}}$		
	P1A3-P1A0		
	P1D3-P1D0		
	P2A2/LCD19		
	P2A1/LCD18		
	P2A0/LCD17		
	P2B3-P2B0		
	P2C3-P2C0		
Pins other than port pins	EO1	Output	Leave unconnected.
	EO0		
	INT	Input	Connect to GND via resistor ^{Note 1} .
	NC	—	Leave unconnected.
	TEST	—	Directly connect to GND.
	VCOH	Input	Set to PLL disable via software and leave unconnected.
	VCOL		

Notes 1. If these pins are externally pulled up (connected to V_{DD} via resistor) or down (connected to GND via resistor) with a high resistor, the pins almost go into a high-impedance state, increasing the current consumption (through current) of the port. Although the value of the pull-up or pull-down resistor varies depending on the application circuit, it generally is several 10 kΩ.

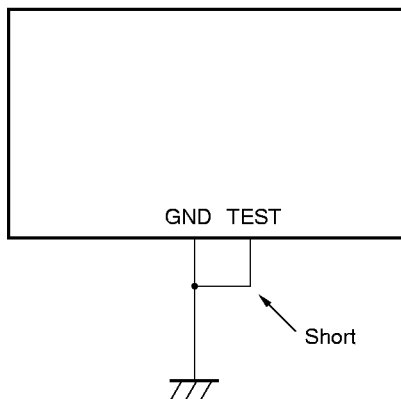
2. The circuit of the general-purpose input port is designed not to increase the current consumption even in the high-impedance state. Do not set these pins in the AMIFC and FMIFC modes; otherwise, the current consumption will increase.

3. The I/O ports are set in the general-purpose input port mode at reset by the $\overline{\text{RESET}}$ pin and reset due to overflow/underflow of the watchdog timer or stack, and on execution of the clock stop instruction.

1.4 Cautions on Using TEST Pin

When V_{DD} is applied to the TEST pin, the device is set in the test mode. Therefore, be sure to keep the wiring length of this pin as short as possible, and directly connect it to the GND pin.

If the wiring length between the TEST pin and GND pin is too long, or if external noise is superimposed on the TEST pin, generating a potential difference between the TEST pin and GND pin, your program may not run normally.



2. PROGRAM MEMORY (ROM)

2.1 Outline of Program Memory

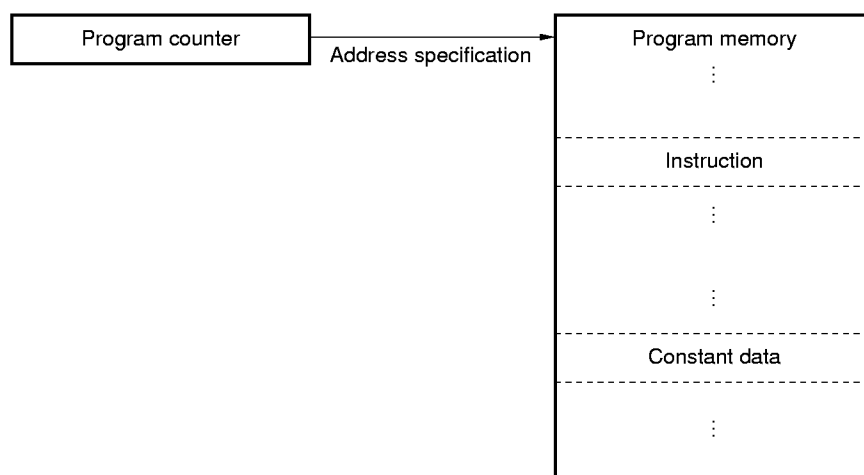
Figure 2-1 outlines the program memory.

As shown in this figure, the addresses of the program memory are specified by the program counter.

The program memory has the following two major functions.

- To store programs
- To store 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 consists of:

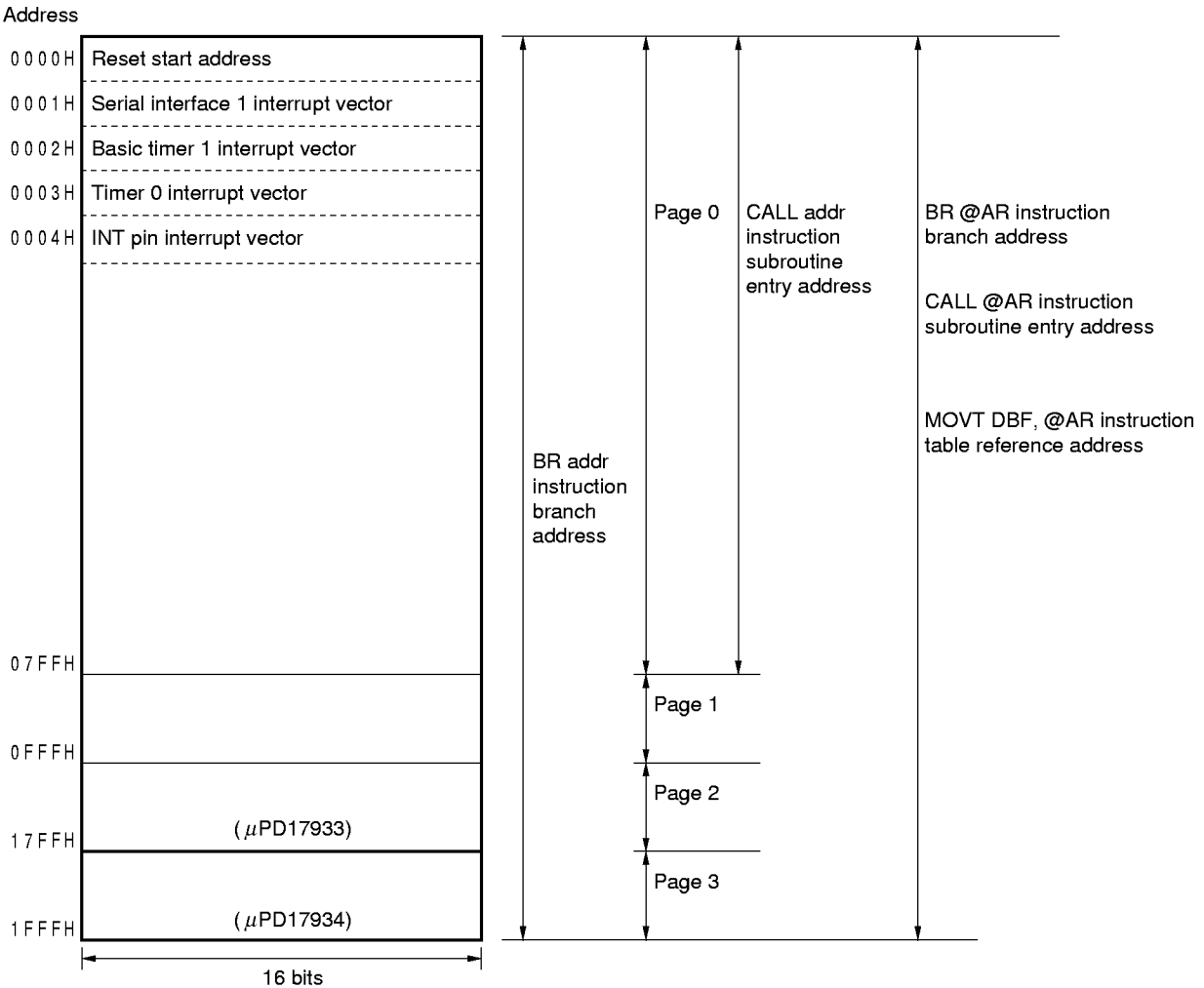
μPD17933: 6144 × 16 bits (0000H-17FFH)

μPD17934: 8192 × 16 bits (0000H-1FFFH)

Because all “instructions” are “one-word instructions”, one instruction can be stored to one address of the program memory.

As constant data, the contents of the program memory are read to the data buffer by using a table reference instruction.

Figure 2-2. Configuration of Program Memory



2.3 Program Counter

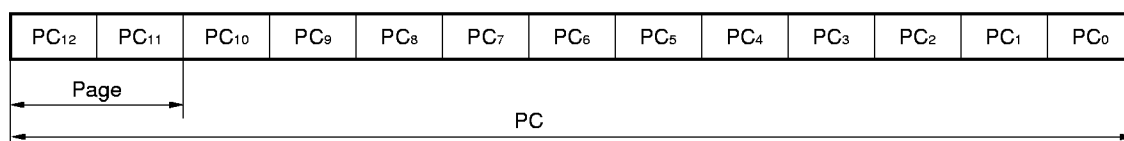
2.3.1 Configuration of program counter

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

As shown in this figure, the program counter consists of a 13-bit binary counter. Bits 11 and 12 of the program counter indicate a page.

The program counter specifies an address of the program memory.

Figure 2-3. Configuration of Program Counter



2.4 Flow of Program

The flow of the program is controlled by the program counter that specifies an address of the program memory.

The program flow when each instruction is executed is described below.

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

Table 2-4 shows the vector address when an interrupt is accepted.

2.4.1 Branch instruction

(1) Direct branch ("BR addr")

The branch destination addresses of the direct branch instruction are all the addresses of the program memory.

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

The branch destination addresses of the indirect branch instruction are all the addresses of the program memory. The addresses are 0000H through 17FFH in the μ PD17933, and 0000H through 1FFFH in the μ PD17934.

Refer to **5.3 Address Register (AR)**.

2.4.2 Subroutine

(1) **Direct subroutine call (“CALL addr”)**

The first address of a subroutine that can be called by the direct subroutine instruction is in page 0 (addresses 0000H through 07FFH).

(2) **Indirect subroutine call (CALL @AR)**

The first addresses of a subroutine that can be called by the indirect subroutine call instruction are all the addresses of the program memory. The addresses are 0000H through 17FFH in the μ PD17933, and 0000H through 1FFFH in the μ PD17934.

Refer to **5.3 Address Register (AR)**.

2.4.3 Table reference

The addresses that can be referenced by the table reference instruction (“MOVT DBF, @AR”) are all the addresses of the program memory. The addresses are 0000H through 17FFH in the μ PD17933, and 0000H through 1FFFH in the μ PD17934.

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

Figure 2-4. Value of Program Counter Upon Execution of Instruction

Program counter Instruction		Contents of Program Counter (PC)													
		b ₁₂	b ₁₁	b ₁₀	b ₉	b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀	
BR addr	Page 0	0	0	← Operand of instruction (addr)											
	Page 1	0	1												
	Page 2	1	0												
	Page 3	1	1												
CALL addr		0	0	← Operand of instruction (addr)											
BR @AR CALL @AR MOVT DBF, @AR		← Contents of address register													
RET RETSK RETI		← Contents of address stack register (ASR) (return address) specified by stack pointer (SP)													
Other instructions (including skip instruction)		Increment													
When interrupt is accepted		← Vector address of each interrupt													
Watchdog timer reset, reset by <u>RESET</u> pin		0	0	0	0	0	0	0	0	0	0	0	0	0	

Table 2-1. Interrupt Vector Address

Order	Internal/External	Interrupt Source	Vector Address
1	External	INT pin	0004H
2	Internal	Timer 0	0003H
3	Internal	Basic timer 1	0002H
4	Internal	Serial interface 1	0001H

2.5 Cautions on Using Program Memory

(1) μ PD17933

The program memory of the μ PD17933 is assigned to addresses 0000H through 17FFH. However, addresses that can be specified by the program counter (PC) are 0000H through 1FFFH, therefore, be careful on the following points when specifying the program memory addresses.

- Write a branch instruction when writing an instruction to address 17FFH.
- Do not write anything to addresses 1800H through 1FFFH.
- Do not branch to addresses 1800H through 1FFFH.

(2) μ PD17934

The program memory of the μ PD17934 is assigned to addresses 0000H through 1FFFH. Be careful on the following points.

- Write a branch instruction when writing an instruction to address 1FFFH.

3. ADDRESS STACK (ASK)

3.1 Outline of Address Stack

Figure 3-1 outlines the address stack.

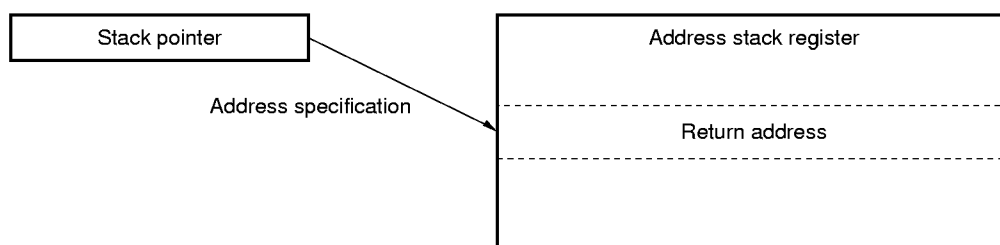
The address stack consists of a stack pointer and address stack registers.

The address of an address stack register is specified by the stack pointer.

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

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

Figure 3-1. Outline of Address Stack



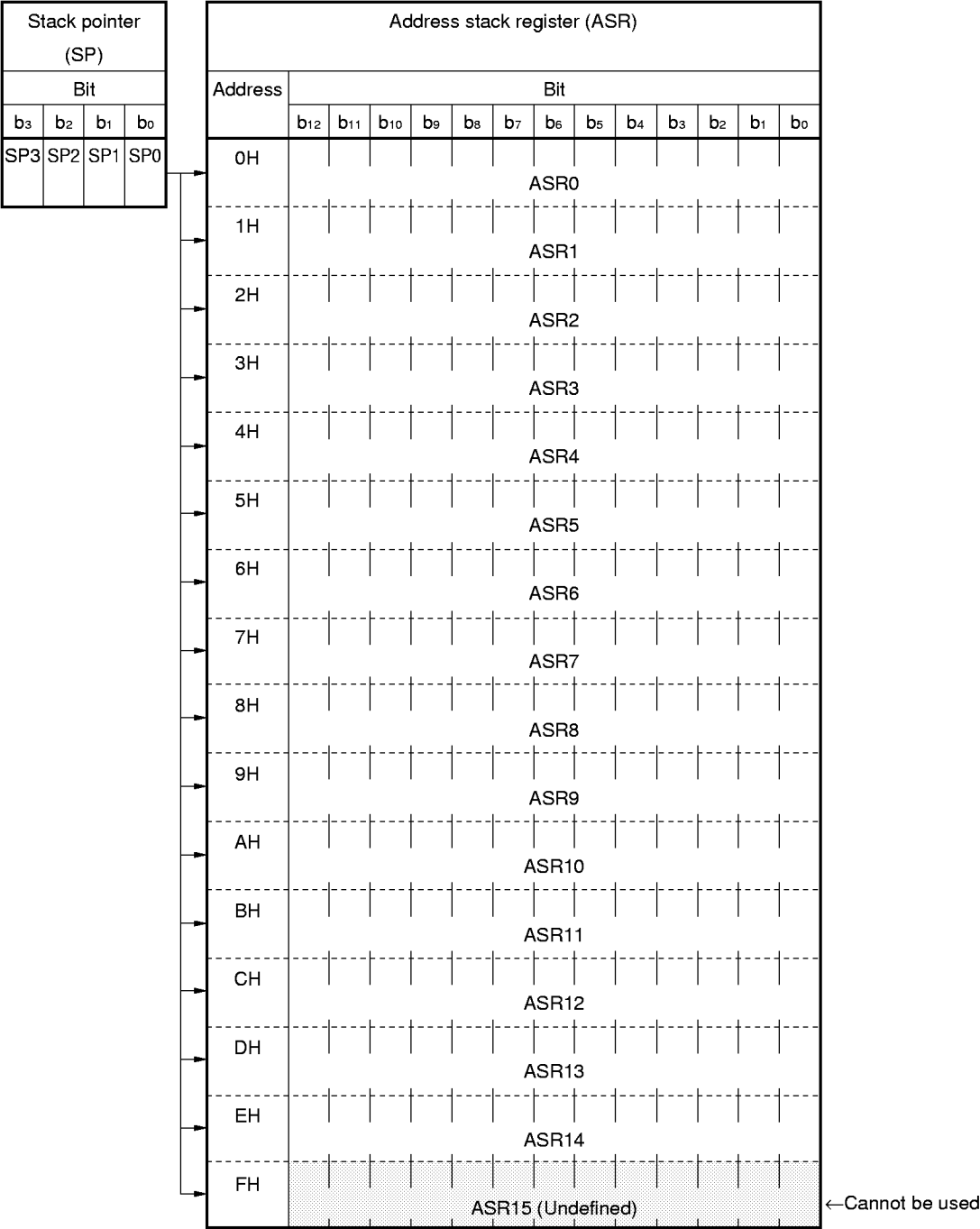
3.2 Address Stack Register (ASR)

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

The address stack register consists of sixteen 13-bit registers ASR0 through ASR15. Actually, however, it consists of fifteen 13-bit registers (ASR0 through ASR14) because no register is allocated to ASR15.

The address stack saves a return address when a subroutine is called, when an interrupt is accepted, and when the table reference instruction is executed.

Figure 3-2. Configuration of Address Stack Register



3.3 Stack Pointer (SP)

3.3.1 Configuration and function of stack pointer

Figure 3-3 shows the configuration and functions of the stack pointer.
 The stack pointer consists of a 4-bit binary counter.
 It specifies the address of an address stack register.
 A value can be directly read from or written to the stack pointer by using a register manipulation instruction.

Figure 3-3. Configuration and Function of Stack Pointer

Name	Flag symbol				Address	Read/Write
	b ₃	b ₂	b ₁	b ₀		
Stack pointer (SP)	(S P 3)	(S P 2)	(S P 1)	(S P 0)	01H	R/W

Specifies address of address stack register (ASR)				
0	0	0	0	Address 0 (ASR0)
0	0	0	1	Address 1 (ASR1)
0	0	1	0	Address 2 (ASR2)
0	0	1	1	Address 3 (ASR3)
0	1	0	0	Address 4 (ASR4)
0	1	0	1	Address 5 (ASR5)
0	1	1	0	Address 6 (ASR6)
0	1	1	1	Address 7 (ASR7)
1	0	0	0	Address 8 (ASR8)
1	0	0	1	Address 9 (ASR9)
1	0	1	0	Address 10 (ASR10)
1	0	1	1	Address 11 (ASR11)
1	1	0	0	Address 12 (ASR12)
1	1	0	1	Address 13 (ASR13)
1	1	1	0	Address 14 (ASR14)
1	1	1	1	Setting prohibited

At reset	Reset by RESET pin	1	1	1	1
	WDT&SP reset	1	1	1	1
Clock stop		Retained			

Reset by RESET pin : Reset by RESET pin
 WDT&SP reset : Reset by watchdog timer and stack pointer
 Clock stop : Upon execution of clock stop instruction

3.4 Operation of Address Stack

3.4.1 Subroutine call instruction (“CALL addr”, “CALL @AR”) and return instruction (“RET”, “RETSK”)

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

When the return instruction is executed, the contents of the address stack register (return address) 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 Table reference instruction (“MOVT DBF, @AR”)

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

3.4.3 When interrupt is accepted and on execution of return instruction (“RETI”)

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

When the return instruction is executed, the contents of an address stack register (return value) 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 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 an address stack register specified by the stack pointer.

When the “POP” instruction is executed, the contents of an 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 Cautions on Using Address Stack

3.5.1 Nesting level and operation on overflow

The value of address stack register (ASR15) is “undefined” when the value of the stack pointer is 0FH.

Accordingly, if a subroutine call exceeding 15 levels, or an interrupt is used without manipulating the stack, execution returns to an “undefined” address.

3.5.2 Reset on detection of overflow or underflow of address stack

Whether the device is reset on detection of overflow or underflow of the address stack can be specified by program. At reset, the program is started from address 0, and some control registers are initialized.

This reset function is valid at reset by the $\overline{\text{RESET}}$ pin. For details, refer to **21. RESET**.

4. DATA MEMORY (RAM)

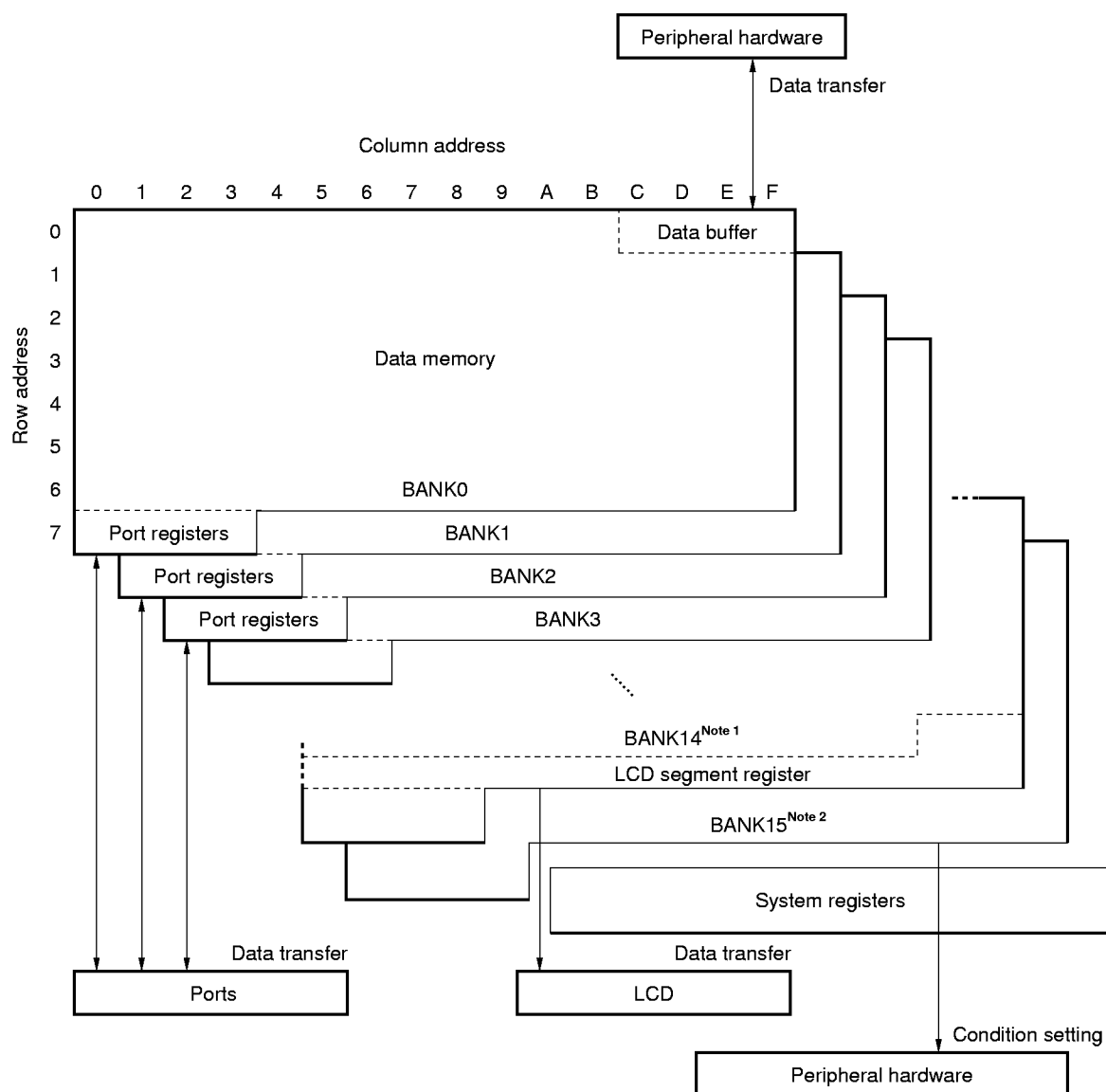
4.1 Outline of Data Memory

Figure 4-1 outlines the data memory.

As shown in the figure, system registers, a data buffer, port registers, LCD segment registers, and control registers are located on the data memory.

The data memory stores data, transfers data with the peripheral hardware or ports, and controls the CPU.

Figure 4-1. Outline of Data Memory



Notes 1. LCD segment registers are allocated to addresses 5CH through 6FH of BANK14.

2. Control registers are allocated to addresses 00H through 6FH of BANK15. Port input/output select registers are allocated to 60H through 6FH.

Cautions 1. Never write anything to address 31H of BANK15 because this address is a test mode area.

2. Address 5BH is not provided to BANK4 through BANK14.

4.2 Configuration and Function of Data Memory

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

As shown in this figure, the data memory is divided into several banks with each bank made up of a total of 128 nibbles with 7H row addresses and 0FH column addresses.

The data memory can be divided into five functional blocks. Each block is described in 4.2.1 through 4.2.6 below.

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

Table 4-1 lists the data memory manipulation instructions.

4.2.1 System registers (SYSREG)

The system registers are allocated to addresses 74H through 7FH.

Because the system registers are allocated to all banks, the same system registers exist at addresses 74H through 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 through 0FH of BANK 0.

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

4.2.3 Port registers

The port registers are allocated to addresses 70H through 73H of BANKs 0 through 2.

For details, refer to **11. GENERAL-PURPOSE PORTS**.

4.2.4 Control registers and port input/output select registers

The control registers are allocated to addresses 00H through 6FH of BANK 15. Of these registers, the port input/output select registers are allocated to addresses 60H through 6FH of BANK15. Addresses 00H through 3FH of BANK15, to which control registers are allocated, also overlap addresses 00H through 3FH of the register file.

For details, refer to **8. REGISTER FILE (RF) AND CONTROL REGISTERS**.

4.2.5 LCD segment registers

The LCD segment registers consists of a total of 20 nibbles at addresses 5CH through 6FH of BANK15 of the data memory.

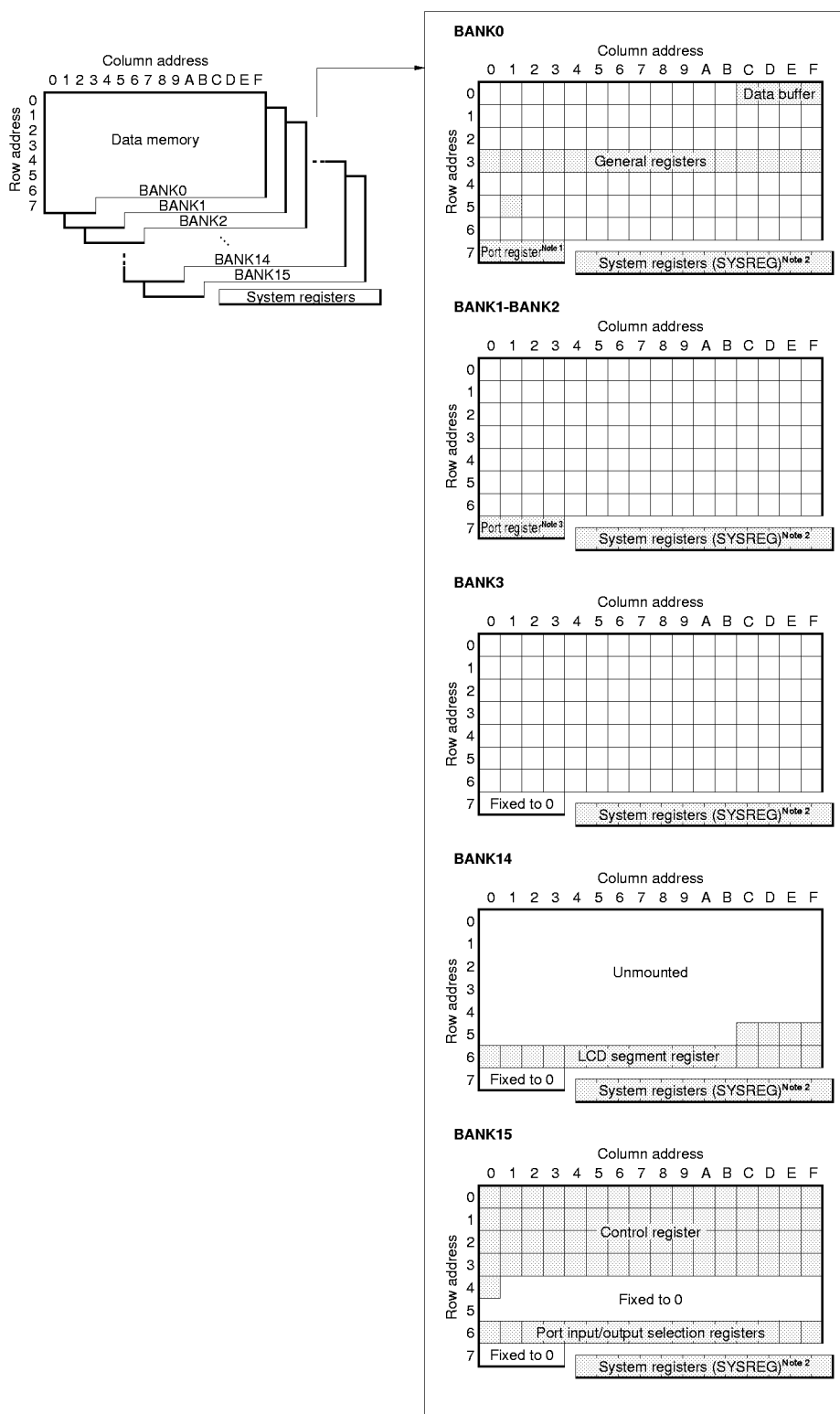
For details, refer to **8.4 LCD Segment Registers** and **19. LCD CONTROLLER/DRIVER**.

4.2.6 General-purpose data memory

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

The general-purpose data memory consists of a total of 448 nibbles of the 112 nibbles each of BANKs 0 through 3.

Figure 4-2. Configuration of Data Memory



Notes 1. The high-order 2 bits of 70H are fixed to 0.

2. The same system register exists.

3. Address 71H of BANK1, and the high-order 1 bit and address 73H of BANK2 are fixed to 0.

Cautions 1. Never write anything to address 31H of BANK15 because this address is a test mode area.

2. Address 5BH is not provided to BANK4 through BANK14.

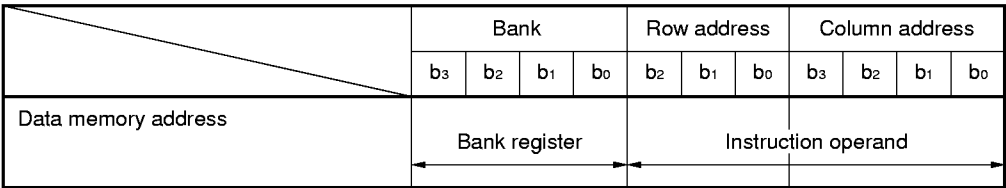
Table 4-1. Data Memory Manipulation Instructions

Function		Instruction
Operation	Add	ADD ADDC
	Subtract	SUB SUBC
	Logic	AND OR XOR
Compare		SKE SKGE SKLT SKNE
Transfer		MOV LD ST
Judge		SKT SKF

4.3 Data Memory Addressing

Figure 4-3 shows address specification of the data memory.
An address of the data memory is specified by a bank, row address, and column address.
A row address and a column address are directly specified by a data memory manipulation instruction.
However, a bank is specified by the contents of a bank register.
For the details of the bank register, refer to 5. SYSTEM REGISTER (SYSREG).

Figure 4-3. Address Specification of Data Memory



4.4 Cautions on Using Data Memory

4.4.1 At reset by $\overline{\text{RESET}}$ pin

The contents of the general-purpose data memory are “undefined” at reset by $\overline{\text{RESET}}$ pin.

Initialize the data memory as necessary.

4.4.2 Cautions on data memory not provided

If a data memory manipulation instruction that reads the data memory is executed to a data memory address not provided, undefined data is read.

Nothing is changed even if data is written to such an address.

5. SYSTEM REGISTERS (SYSREG)

5.1 Outline of System Registers

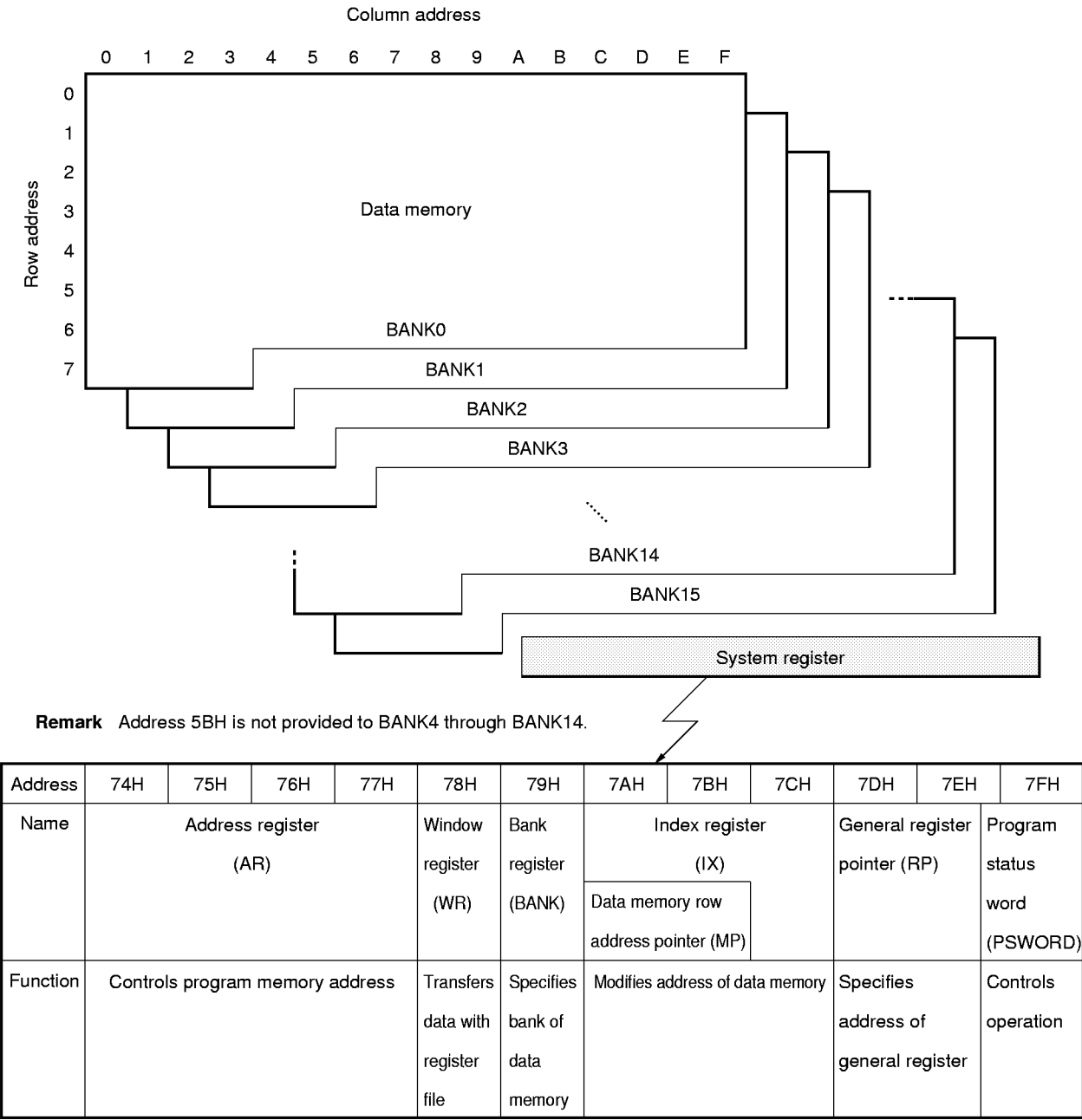
Figure 5-1 shows the location of the system registers on the data memory and their outline.

As shown in the figure, the system registers are allocated to addresses 74H through 7FH of all the banks of the data memory. Therefore, identical system registers exist at addresses 74H through 7FH of any bank.

Because the system registers are located on the data memory, they can be manipulated by all data memory manipulation instructions.

Seven types of system registers are available depending on function.

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



5.2 System Register List

Figure 5-2 shows the configurations of the system registers.

Figure 5-2. Configuration of System Registers

Address	74H	75H	76H	77H	78H	79H	7AH	7BH	7CH	7DH	7EH	7FH
Name	System registers											
	Address register (AR)				Window register (WR)	Bank register (BANK)	Index register (IX) Data memory row address pointer (MP)			General register pointer (RP)		Program status word (PSWORD)
Symbol	AR3	AR2	AR1	AR0	WR	BANK	IXH	IXM	IXL	RPH	RPL	PSW
							MPH	MPL				
Bit	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀	b ₃ b ₂ b ₁ b ₀
Data							(IX)			(RP)		B C Z I
												C M Y X
							(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 the figure, the address register consists of 16 bits of system register addresses 74H through 77H (AR3 through AR0).

Figure 5-3. Configuration of Address Register

Address		74H				75H				76H				77H			
Name		Address register (AR)															
Symbol		AR3				AR2				AR1				AR0			
Bit		b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
Data		⌋ M S B ⌋															⌋ L S B ⌋
At reset	Reset by RESET pin	0				0				0				0			
	WDT&SP reset	0				0				0				0			
Clock stop		Retained				Retained				Retained				Retained			

Reset by $\overline{\text{RESET}}$ pin : Reset by $\overline{\text{RESET}}$ pin

WDT&SP reset : Reset by watchdog timer and stack pointer

Clock stop : On execution of clock stop instruction

5.3.2 Function of address register

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

A dedicated instruction ("INC AR") is available that can increment the contents of the address instruction by one.

The following paragraphs (1) through (5) describe the operation of the address register when the respective instructions are executed.

(1) Table reference instruction ("MOVT DBF, @AR")

When the table reference instruction is executed, the constant data (16 bits) of a program memory address specified by the contents of the address register are read to the data buffer.

The constant data that can be specified by the address register is stored to address 0000H to 17FFH in the case of the μ PD17933, and address 0000H to 1FFFH in the case of the μ PD17934.

(2) Stack manipulation instruction ("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 transferred to an address stack register specified by the stack pointer whose value has been decremented by one.

When the "POP AR" instruction is executed, the contents of an 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")

When this instruction is executed, the program branches to a program memory address specified by the contents of the address register.

The branch address that can be specified by the address register is 0000H to 17FFH in the case of the μ PD17933, and 0000H to 1FFFH in the case of the μ PD17934.

(4) Indirect subroutine call instruction ("CALL @AR")

The subroutine at a program memory address specified by the contents of the address register can be called.

The first address of the subroutine that can be specified by the address register is 0000H to 17FFH in the case of the μ PD17933, and 0000H to 1FFFH in the case of the μ PD17934.

(5) Address register increment instruction ("INC AR")

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

5.3.3 Address register and data buffer

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

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

5.3.4 Cautions on Using Address Register

Because the address register is configured in 16 bits, it can specify an address up to FFFFH.

However, the program memory exists at addresses 0000H through 17FFH in the case of the μ PD17933 and addresses 0000H through 1FFFH in the case of the μ PD17934.

Therefore, the maximum value that can be set to the address register of the μ PD17933 is address 17FFH. In the case of the μ PD17934, it is address 1FFFH.

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 of system register address 78H (WR).

Figure 5-4. Configuration of Window Register

Address	78H			
Name	Window register (WR)			
Symbol	WR			
Bit	b ₃	b ₂	b ₁	b ₀
Data	$\begin{matrix} \wedge \\ M \\ S \\ B \\ \vee \end{matrix}$			$\begin{matrix} \wedge \\ L \\ S \\ B \\ \vee \end{matrix}$
At reset	Reset by RESET pin			Undefined
	WDT&SP reset			Retained
	Clock stop			

5.4.2 Function of window register

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

Data transfer between the window register and register file is manipulated by using dedicated instructions "PEEK WR, rf" and "POKE, rf WR" (rf: address of register file).

The following paragraphs (1) and (2) describe the operation of the window register when these instructions are executed.

For further information, also refer to **8. REGISTER FILE (RF) AND CONTROL REGISTERS**.

(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 of system register address 79H (BANK).

Figure 5-5. Configuration of Bank Register

Address		79H			
Name		Bank register (BANK)			
Symbol		BANK			
Bit		b ₃	b ₂	b ₁	b ₀
Data		⌈ M S B ⌋			⌈ L S B ⌋
At reset	Reset by RESET pin	0			
	WDT&SP reset	0			
	Clock stop	Retained			

5.5.2 Function of bank register

The bank register specifies a bank of the data memory.

Table 5-1 shows the relationships between the value of the bank register and a bank of the data memory that is specified.

Because the bank register is one of the system registers, its contents can be rewritten regardless of the bank currently specified.

When manipulating a bank register, therefore, the status of the bank at that time is irrelevant.

Table 5-1. Data Memory Bank Specification

Bank Register (BANK)				Bank of Data Memory
b ₃	b ₂	b ₁	b ₀	
0	0	0	0	BANK0
0	0	0	1	BANK1
0	0	1	0	BANK2
0	0	1	1	BANK3
1	1	1	0	BANK14
1	1	1	1	BANK15

Remark Addresses 00H through 5BH are not provided to BANK4 through BANK14.

Caution The area to which the data memory is allocated differs depending on the model. For details, refer to Figure 4-2 Configuration of Data Memory.

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) made up of 11 bits (the low-order 3 bits (IXH) of system register address 7AH, and 7BH and 7CH (IXM, IXL)) and an index enable flag (IXE) at the lowest bit position of 7FH (PSW).

The data memory row address pointer (memory pointer) consists of a data memory row address pointer (MP) that is made up of 7 bits of the low-order 3 bits of 7AH (MPH) and 7BH (MPL), and a data memory row address pointer enable flag (memory pointer enable flag: MPE) at the lowest bit position of 7AH (MPH).

In other words, the high-order 7 bits of the index register are shared with the data memory row address pointer

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 ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
Data		M P E	⌢ M S B ⌢										⌢ L S B ⌢								I X E
			⌢ M S B ⌢																		
			MP																		
At reset	Reset by $\overline{\text{RESET}}$ pin	0				0				0								0			
	WDT&SP reset	0				0				0								0			
Clock stop		Retained				Retained				Retained								R			

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) describe their functions.

A dedicated instruction ("INC IX") that increments the value of the index register by one is available.

For the details of address modification, refer to **7. ALU (Arithmetic Logic Unit) BLOCK**.

(1) Index register (IX)

When a data memory manipulation instruction is executed, the data memory address is modified by the contents of the index register.

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

To modify the 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 a data memory address (called real address) specified by the result of this OR operation.

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

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

INC	AR	RORC	r
INX	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 (MP)

When the general register indirect transfer instruction ("MOV @r,m" or "MOV m,@r") is executed, the indirect transfer destination address is modified.

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

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

Instructions other than the general register indirect transfer instruction 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 of 10 bits, its contents are incremented to "000H" if the "INC IX" instruction is executed when the contents of the index register are "3FFH".

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 the figure, the general register pointer consists of 7 bits including 4 bits of system register address 7DH (RPH) and the high-order 3 bits of address 7EH (RPL).

Figure 5-7. Configuration of General Register Pointer

Address		7DH				7EH			
Name		General register pointer (RP)							
Symbol		RPH				RPL			
Bit		b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
Data		⌈ M S B ⌋						⌈ L S B ⌋	⌈ B C D ⌋
At reset	Reset by RESET pin	0				0			
	WDT&SP reset	0				0			
Clock stop		Retained				Retained			

5.7.2 Function of general register pointer

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

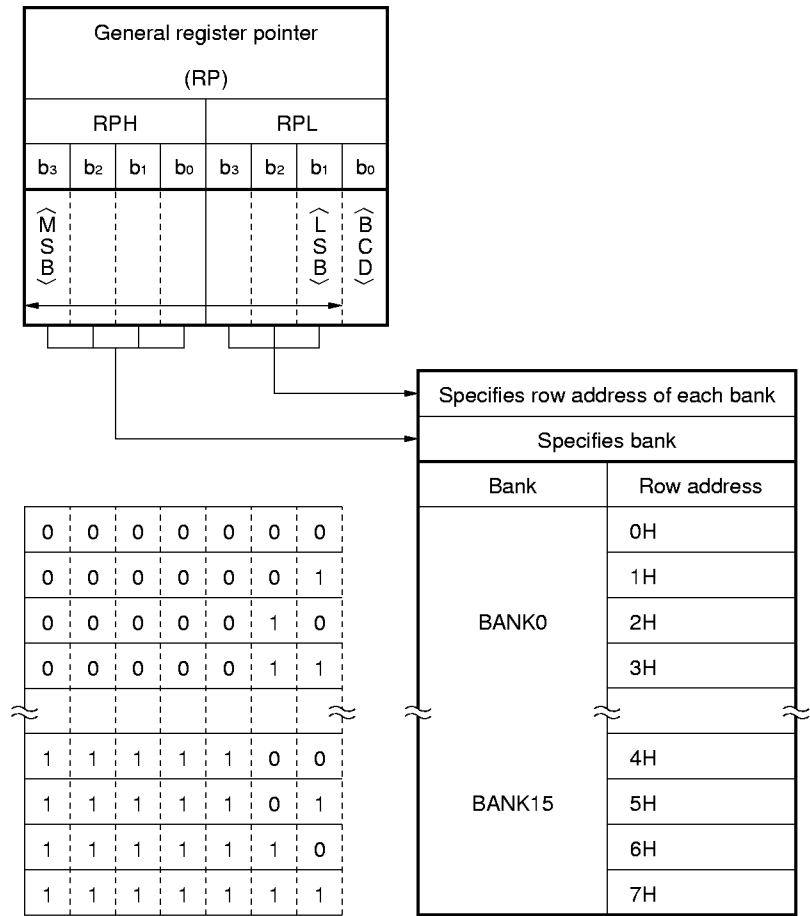
Figure 5-8 shows the addresses of the general registers specified by the general register pointer.

As shown in the figure, a bank is specified by the high-order 4 bits (RPH: address 7DH) of the general register pointer, and a row address is specified by the low-order 3 bits (RPL: address 7EH).

Because the valid number of bits of the general register pointer is 7, all the row addresses (0H through 7FH) of all the banks can be specified as general registers.

For the details of the operation of the general register, refer to 6. GENERAL REGISTER (GR).

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



Remark Address 5BH is not provided to BANK4 through BANK14.

5.7.3 Cautions on using general register pointer

The lowest bit of address 7EH (RPL) of the general register pointer is allocated as the BCD flag of the program status word.

When rewriting 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 a total of 5 bits including the lowest bit of system register address 7EH (RPL) and 4 bits of address 7FH (PSW).

Each bit of the program status word has its own function. The 5 bits of the program status word are BCD flag (BCD), compare flag (CMP), carry flag (CY), zero flag (Z), and index enable flag (IXE).

Figure 5-9. Configuration of Program Status Word

Address		7EH				7FH			
Name						Program status word (PSWORD)			
Symbol		RPL				PSW			
Bit		b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
Data					B C D	C M P	C Y	Z	I X E
At reset	Reset by RESET pin	0				0			
	WDT&SP reset	0				0			
	Clock stop	Retained				Retained			

5.8.2 Function of program status word

The program status word is a register that sets the conditions under which the ALU (Arithmetic Logic Unit) executes an operation or data transfer, or indicates the result of an 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. Outline of Function of Each Flag of Program Status Word

(RP)				Program Status Word (PSWORD)			
RPL				PSW			
b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
			B	C	C	Z	I
			C	M	Y		X
			D	P			E

Flag Name	Function
Index enable flag (IXE)	Modifies address of data memory when data memory manipulation instruction is executed. 0 : Does not modify 1 : Modifies
Zero flag (Z)	Indicates result of arithmetic operation is zero. Status of this flag differs depending on contents of compare flag.
Carry flag (CY)	Indicates occurrence of carry or borrow as result of execution of addition or subtraction instruction. This flag is reset to 0 if no carry or borrow occurs. It is set to 1 if carry or borrow occurs. This flag is also used as shift bit of "RORC r" instruction.
Compare flag (CMP)	Indicates whether result of arithmetic operation is stored to data memory or general register. 0 : Stores result. 1 : Does not store result.
BCD flag (BCD)	Indicates whether arithmetic operation is performed in decimal or binary. 0 : Binary operation 1 : Decimla operation

5.8.3 Cautions on using program status word

When an arithmetic operation (addition or subtraction) is executed to the program status word, the "result" of the arithmetic operation is stored.

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

6. GENERAL REGISTER (GR)

6.1 Outline of General Register

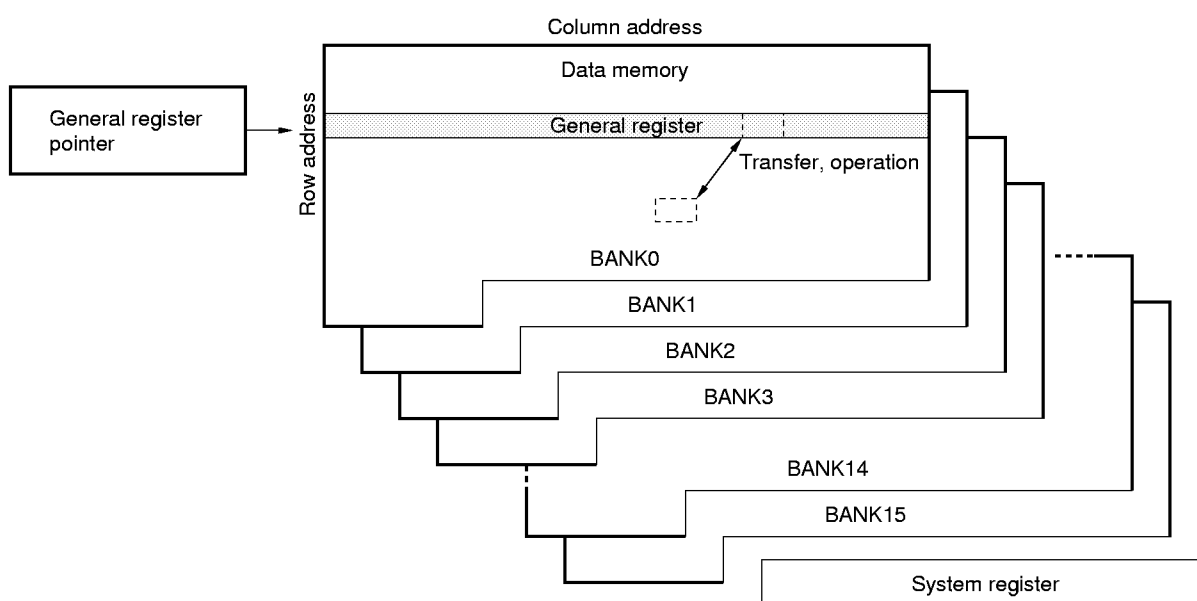
Figure 6-1 outlines the general register.

As shown in the figure, the general register is specified in the data memory by the general register pointer.

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

The general register is used to transfer or operate data between data memory addresses.

Figure 6-1. Outline of General Register



Remark Address 5BH is not provided to BANK4 through BANK14.

6.2 General Register

The general register consists of 16 nibbles (16×4 bis) of the same row address on the data memory.

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

The 16 nibbles of the same row address specified as a general register operate or transfer data with the data memory by a single instruction.

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

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

6.3 Generating Address of General Register by Each Instruction

The following sections 6.3.1 and 6.3.2 explain how the address of the general register is 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 Add (“ADD r, m”, “ADDC r, m”), subtract (“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 (“RORC r”) instructions

Table 6-1 shows the address of the general register specified by operand “r” of an instruction. Operand “r” of an instruction specifies only a column address.

Table 6-1. Generating Address of General Register

	Bank				Row address			Column address			
	b ₃	b ₂	b ₁	b ₀	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
General register address	Contents of general register pointer							r			

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

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

Table 6-2. Generating Address of General Register

	Bank				Row address			Column address			
	b ₃	b ₂	b ₁	b ₀	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
General register address	Contents of general register pointer							r			
Indirect transfer address	Same as data memory							Contents of “r”			

6.4 Cautions on Using General Register

6.4.1 Row address of general register

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

6.4.2 Operation between general register and immediate data

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

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

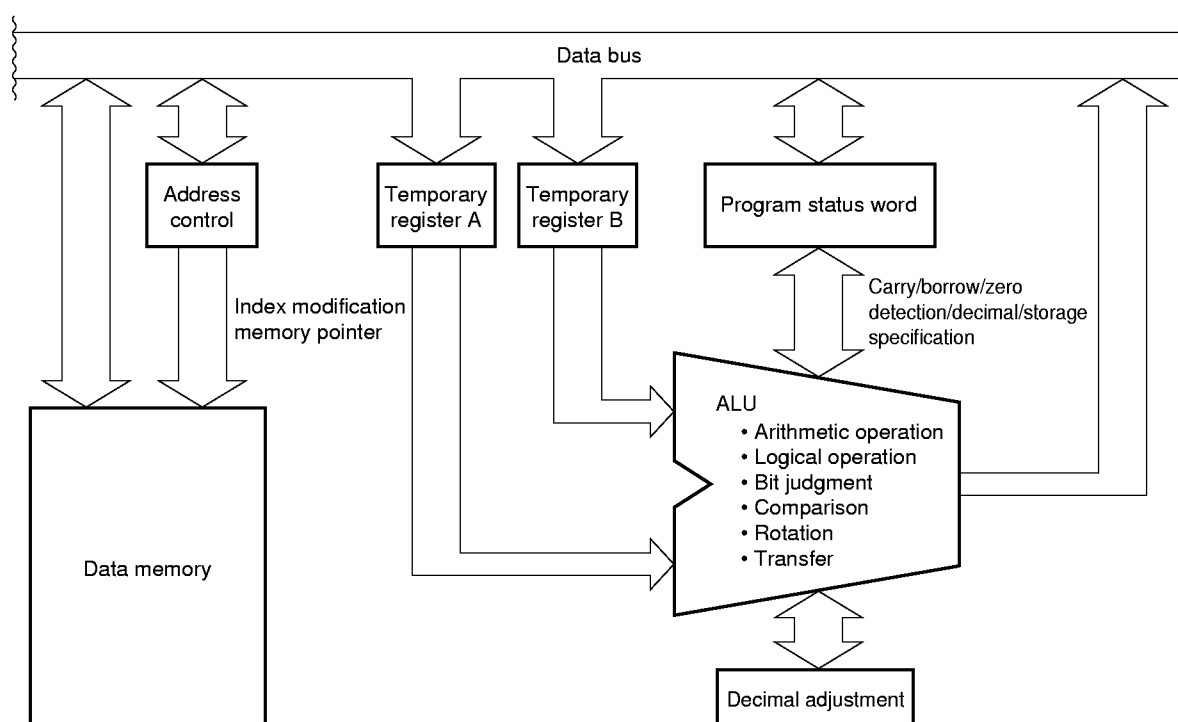
7.1 Outline of ALU Block

Figure 7-1 outlines the ALU block.

As shown in the figure, the ALU block consists of an ALU, temporary registers A and B, program status word, decimal adjustment circuit, and memory address control circuit.

The ALU operates on, 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 performs arithmetic operation, logical operation, bit judgment, comparison, rotation, and transfer of 4-bit data according to instructions 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 cannot be controlled by program.

7.2.3 Program status word

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

For further information on the program status word, also refer to **5.8 Program Status Word (PSWORD)**.

7.2.4 Decimal adjustment circuit

The decimal adjustment circuit converts the result of an arithmetic operation into a decimal number if the BCD flag of the program status word is set to "1" during arithmetic operations.

7.2.5 Address control circuit

The address control circuit 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 how data memory addresses are modified by the index register and data memory row address pointer.

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

Table 7-1. List of ALU Processing Instruction Operations

ALU Function	Instruction		Difference in Operation Depending on Program Status Word (PSWORD)					Address Modification	
			Value of BCD flag	Value of CMP flag	Operation	Operation of CY flag	Operation of Z flag	Index	Memory pointer
Add	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	Modifies	Does not modify
		m, #n4							
	ADDC	r, m	0	1	Does not store result of binary operation		Retains status if result of operation is 0000B; otherwise, reset		
		m, #n4							
Subtract	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		Retains status if result of operation is 0000B; otherwise, reset		
		m, #n4							
Logical operation	OR	r, m	Don't care	Don't care	Not affected	Retains previous status	Retains previous status	Modifies	Does not modify
		m, #n4	(retained)	(retained)					
	AND	r, m							
		m, #n4							
	XOR	r, m							
		m, #n4							
Judge	SKT	m, #n	Don't care	Don't care	Not affected	Retains previous status	Retains previous status	Modifies	Does not modify
	SKF	m, #n	(retained)	(reset)					
Compare	SKE	m, #n4	Don't care	Don't care	Not affected	Retains previous status	Retains previous status	Modifies	Does not modify
	SKNE	m, #n4	(retained)	(retained)					
	SKGE	m, #n4							
	SKLT	m, #n4							
Transfer	LD	r, m	Don't care	Don't care	Not affected	Retains previous status	Retains previous status	Modifies	Does not modify
	ST	m, r	(retained)	(retained)					
	MOV	m, #n4							
		@r, m							
		m, @r							
Rotate	RORC	r	Don't care (retained)	Don't care (retained)	Not affected	Value of b ₀ of general register	Retains previous status	Does not modify	Does not modify

Table 7-2. Modification of Data Memory Address and 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 ₃	b ₂	b ₁	b ₀	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀			
0	0	<div><div>RP</div><div>r</div></div>												<div><div>BANK</div><div>m</div></div>												<div><div>BANK</div><div>m_R</div><div>(r)</div></div>											
0	1	ditto												ditto												<div><div>MP</div><div>(r)</div></div>											
1	0	ditto												<div><div>BANK</div><div>m</div><div>Logical IX</div><div>OR</div></div>												<div><div>BANK</div><div>m_R</div><div>Logical IXH, IXM</div><div>OR</div><div>(r)</div></div>											
1	1	ditto												ditto												<div><div>MP</div><div>(r)</div></div>											

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_R, m_C

m_R : data memory row address (high-order)

m_C : data memory column address (low-order)

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

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

Remark Decimal adjustment is not correctly carried out in the shaded area in the above table.

7.4 Cautions on Using ALU

7.4.1 Cautions on execution operation to program status word

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

The CY and Z flags in the program status word are usually set or reset by the result of the arithmetic operation. If an arithmetic operation is executed to the program status word itself, the result of the operation is stored to the program status word, and consequently, it cannot be judged whether a carry or borrow occurs or whether the result of the operation is zero.

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

7.4.2 Cautions on executing decimal operation

The decimal operation can be executed only when the result of the operation falls within the following ranges:

- (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 or falling below the above ranges, the result is a value greater than 1010B (0AH).

8. REGISTER FILE (RF) AND CONTROL REGISTERS

8.1 Outline of Register File

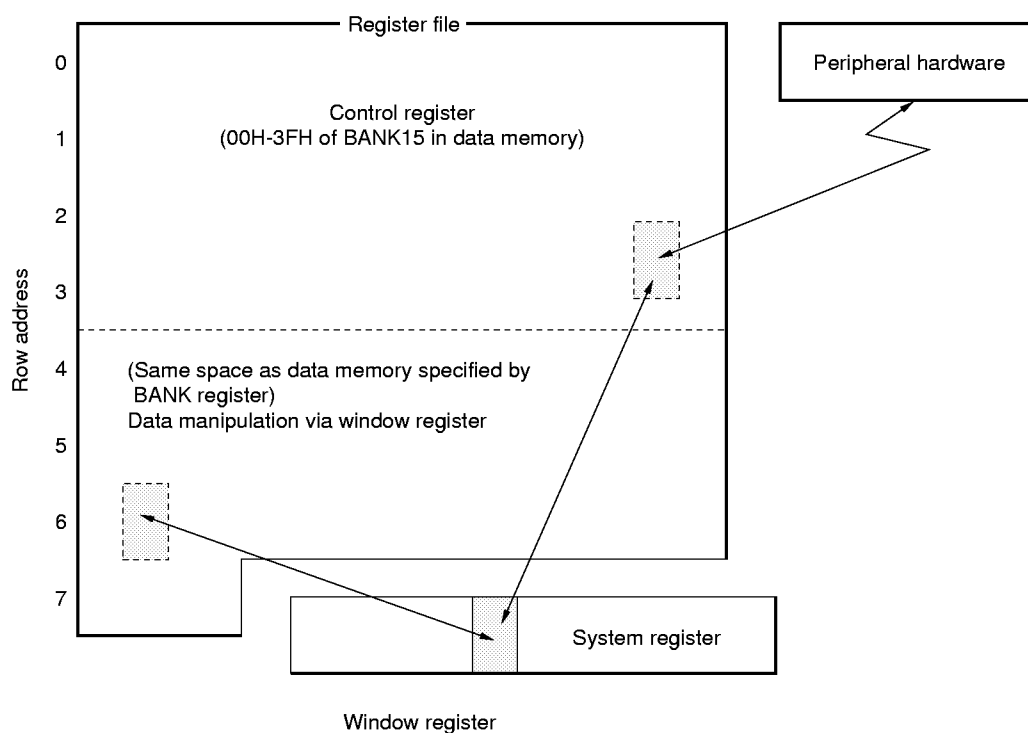
Figure 8-1 outlines the register file.

As shown in the figure, the register file consists of the control registers existing on addresses 00H through 3FH of BANK15 in the data memory, and a portion overlapping the data memory specified by the BANK register.

The control registers set conditions of the peripheral hardware units.

The data on the register file can be read or written via 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 the relationships between the register file and data memory.

The register file is assigned addresses in 4-bit units, like the data memory, and consists of a total of 128 nibbles with row addresses 0H through 7FH and column addresses 0H through 0FH.

Addresses 00H through 3FH are overlapping addresses 00H through 3FH of BANK15 and called control registers that sets the conditions of the peripheral hardware units.

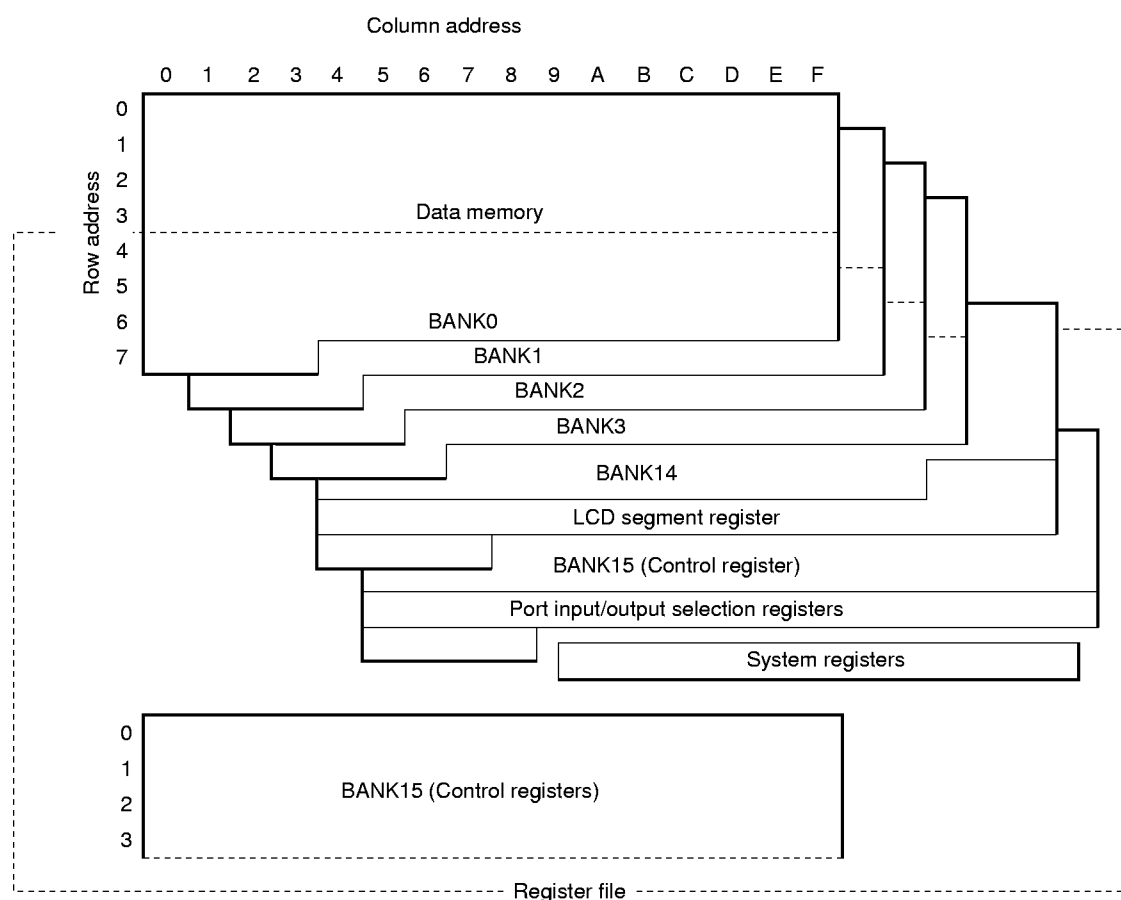
Addresses 40H through 7FH overlap the data memory specified by the BANK register.

In other words, addresses 40H through 7FH of the register file are addresses 40H through 7FH of the currently-selected bank of data memory.

Addresses 40H through 7FH of the register file can be manipulated in the same manner as the data memory, except that the addresses of the register file can also be manipulated by using register file manipulation instructions ("PEEK WR, rf" and "POKE rf, WR"). Note, however, that addresses 40H through 6FH of BANK15 are assigned control register including port input/output selection registers (for details refer to **8.3 Control Registers and Input/Output Selection Registers**).

Addresses 5CH through 6FH of BANK14 are assigned LCD segment register (for details, refer to **8.4 LCD Segment Register**).

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



Caution Never write anything to address 31H of BANK15 because this address is a test mode area.

Remark Address 5BH is not provided to BANK4 through BANK14.

8.2.1 Register file manipulation instructions (“PEEK WR, rf”, “POKE rf, WR”)

Data is read from or written to the register file via the window register of the system registers, by using the following instructions.

(1) “PEEK WR, rf”

Reads data of the register file addressed by “rf” to the window register.

(2) “POKE rf, WR”

Writes the data of the window register to the register file addressed by “rf”.

8.3 Control Registers and Input/Output Selection Registers

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

The control registers, including the input/output select registers, consist of a total of 112 nibbles (112 x 4 bits) at addresses 00H through 6FH of BANK15 of the data memory. Of these addresses, 00H through 3FH overlap addresses 00H through 3FH of the register file. Addresses 60H through 6FH are assigned the input/output select registers.

However, only 38 nibbles of the control registers are actually used. The remaining 14 nibbles are unused registers and prohibited from being written or read.

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

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.

Among the 4-bit data in 1 nibble, the bit fixed to “0” is always “0” when it is read, and is also “0” when it is written.

The 74 nibbles of unused registers are undefined when their contents are read, and nothing changes even when they are written.

Never write anything to address 31H of BANK15 because this address is a test mode area.

Table 8-1 lists the peripheral hardware control functions of the control registers.

Figure 8-3. Configuration of Control Registers (Addresses 00H-3FH) (1/4)

(BANK15) Column Address									
Row Address	Item	0	1	2	3	4	5	6	7
0 (8) ^{Note1}	Name		Stack pointer	Watchdog timer clock selection	Watchdog timer counter reset	Data buffer stack pointer	Stack overflow/ underflow reset selection		MOVT bit selection
	Symbol		(SP3) (SP2) (SP1) (SP0)	0 0 WDTCK1 WDTCK0	WDTRES 0 0 0	0 0 (DBFSPP1) (DBFSPP0)	0 0 SPRSEL1 SPRSEL0		0 0 MOVTSEL1 MOVTSEL0
	Read/ Write		R/W	R/W	W & Reset	R	R/W		R/W
1 (9) ^{Note1}	Name	PLL mode selection	PLL reference frequency selection	PLL unlock FF		BEEP clock selection		Watchdog timer/stack pointer reset status detection	Basic timer 0 carry
	Symbol	PLLSCNF 0 0 PLLMD1 PLLMD0	0 PLLRFCK2 PLLRFFCK1 PLLRFFCK0	0 0 0 PLLUL		0 BEEP0SEL BEEP0CK1 BEEP0CK0		0 0 0 WDTSPRES	0 0 0 BTMOCY
	Read/ Write	R/W	R/W	R & Reset		R/W		R & Reset	R & Reset
2 (A) ^{Note1}	Name		IF counter gate status detection	IF counter mode selection	IF counter control	A/D converter channel selection	A/D converter mode selection		
	Symbol	0 0 0 FCGCH0 Note2	0 0 0 IFCGOSTT IFCGMD1 IFCGMD0 IFCGCK1 IFCGCK0	0 0 IFCSTRT IFCRES	0 0 IFCSTRT IFCRES	0 0 ADCCH1 ADCCH0	0 0 ADCSTRT ADCCMP		
	Read/ Write		R	R/W	W	R/W	R	R/W	R/W
3 (B) ^{Note1}	Name		Note 3						
	Symbol								
	Read/ Write								

Notes 1. () indicates an address that is used when the assembler is used.

2. Never write anything to address 20H of BANK15.

3. Never write anything to address 31H of BANK15 because this address is a test mode area.

Figure 8-3. Configuration of Control Register (Address 00H-3FH) (2/4)

8	9	A	B	C	D	E	F
System register interrupt stack pointer							
0 (SYSRSP2) (SYSRSP1) (SYSRSP0)							
R							
Basic timer 1 clock selection				Serial I/O1 clock	Serial I/O1 mode selection		Interrupt edge selection
0 0 0 BTM1CK0				0 0 S S O O 1 1 C C K K 1 0	0 S S S O O O 1 1 1 M H T O D Z		0 0 0 I E G O
R/W				R/W	R/W		R/W
			Timer 0 counter clock selection	Timer 0 mode selection	Interrupt enable 1	Interrupt enable 2	Interrupt enable 3
			T T T T M M M M O O O O O O O O V V F	T 0 0 0 M O O O O O O O V V F	I P P P P S S T I O O M 1 0 3 2	I P P P P T T M M M M M 1 0 4 3	I P P P P 2 1 0 C E
			R/W	R/W	R/W	R/W	R/W
				Serial interface 1 interrupt request	Basic interval timer 1 interrupt request	Timer 0 interrupt request	INT0 pin interrupt request
				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 0	I N T 0 0 0 0 I R Q 1
				R/W	R/W	R/W	R/W

Note () indicates an address that is used when the assembler is used.

Figure 8-3. Configuration of Control Register (Addresses 40H-6FH) (3/4)

(BANK15) Column Address									
Row Address	Item	0	1	2	3	4	5	6	7
★ 4	Name	LCD driver display start							
	Symbol	<div> <div>L C D B C K</div> <div>0 0</div> <div>L O C K</div> </div> <small>Note</small>							
	Read/ Write	R/W							
5	Name								
	Symbol								
	Read/ Write								
6	Name								
	Symbol								
	Read/ Write								

★ **Note** Bit 3 of address 40H of BANK15 is a test mode area, therefore, do not write "1" to bit 3.

Figure 8-3. Configuration of Control Register (Addresses 40H-6FH) (4/4)

8				9				A				B				C				D				E				F			

Table 8-1. Peripheral Hardware Control Functions of Control Registers (1/5)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				At Reset		Clock Stop							
	Name	Address (BANK15)	Read/ Write	b ₃ b ₂ b ₁ b ₀ Symbol	Function	Set value		Reset by <div>RESET</div> pin	WDT & SP reset									
						0	1											
Stack	Stack pointer	01H	R/W	(SP3) ----- (SP2) ----- (SP1) ----- (SP0)				F	F	Retained								
				Interrupt stack pointer of system register	08H	R	0 ----- (SYSRSP2) ----- (SYSRSP1) ----- (SYSRSP0)				5	5	Retained					
							Data buffer stack pointer	04H	R		0 ----- 0 ----- (DBFSP1) ----- (DBFSP0)	Fixed to "0"				0	0	Retained
											Detects nesting level of data buffer stack	0 Level 0		0 Level 1	1 Level 2	1 Level 3		
	Stack overflow/ underflow reset selection	05H	R/W							0 ----- 0 ----- SPRSEL1 ----- SPRSEL0	Fixed to "0"				3	Retained	Retained	
Selects interrupt stack overflow/underflow reset (can be set only once following power application)				Reset prohibited	Reset valid													
Selects address stack overflow/underflow reset (can be set only once following power application)																		
Watchdog timer	Watchdog timer clock selection	02H	R/W	0 ----- 0 ----- WDTCK1 ----- WDTCK0	Fixed to "0"			3	Retained	Retained								
				Selects clock of watchdog timer (can be set only once following power application)	0 Not used instruction	0 4096	1 Setting 8192	1 prohibited instruction										
				Watchdog timer counter reset	03H	W & Reset	WDTRES ----- 0 ----- 0 ----- 0 ----- WDTSPRES	Resets watchdog timer counter	Invalid		Reset if written	Undefined	Undefined	Undefined				
							Fixed to "0"											
	WDT&SP reset status detection	16H	R & Reset				0 ----- 0 ----- 0 ----- WDTSPRES				0	1	Retained					
Detects resetting of watchdog timer/stack pointer				No reset request	Reset request													

Table 8-1. Peripheral Hardware Control Functions of Control Registers (2/5)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				At Reset		Clock Stop				
	Name	Address (BANK15)	Read/ Write	b ₃ b ₂ b ₁ b ₀ Symbol	Function	Set value		Reset by RESET pin	WDT & SP reset						
						0	1								
MOV _T	MOV _T bit selection	07H	R/W	0	Fixed to "0"			0	0	Retained					
				0											
				MOV _{TSEL} 1							Sets bit transferred by MOV _T (transferred to 0EH and 0FH only during 8-bit transfer)	00 16-bit transfer	0 High-order 8-bit transfer	1 Low-order 8-bit transfer	
				MOV _{TSEL} 0							01 1	0			
Serial interface	Serial I/O1 clock selection	1CH	R/W	0	Fixed to "0"			0	0	0					
				0											
				SIO0CK1							Sets internal clk of serial	0 External clock (75 kHz)	0 125 kHz	1 18.75 kHz	1 37.5 kHz
				SIO0CK0							interface 1	0 1	0 0	1 1	
	Serial I/O1 mode selection	1DH	R/W	0	Fixed to "0"			0	0	0					
				SIO1MOD							Sets SI1/SO2 pin switching	SI1	SO2		
				SIO1HIZ							Sets P0B2/SO1 pin status	General I/O port	Serial data output pin		
				SIO1TS							Starts or stops operation	Stops operation	Starts operation		
PLL frequency synthesizer	PLL mode selection	10H	R/W	PLLSCNF	Sets low-order bits of swallow counter	LSB is 0	LSB is 1	Undefined	Undefined	Retained					
				0	Fixed to "0"										
				PLLMD1	Sets division mode of PLL	0 Disabled	0 MF				1 VHF	1 HF			
				PLLMD0	0 1	0 0	1 1								
	PLL reference frequency selection	11H	R/W	0	Fixed to "0"			7	7	7					
				PLLRFCK2							Sets reference frequency of PLL	0: 1 kHz 3: 6.25 kHz 6: Setting prohibited 7: Setting prohibited	1: 3 kHz 4: 12.5 kHz 5: 25 kHz		
				PLLRFCK1											
				PLLRFCK0											
	PLL unlock FF	12H	R & Reset	0	Fixed to "0"			Undefined	Undefined	Retained					
				0											
				0											
	PLLUL	Detects status of unlock FF	Locked	Unlocked											
BEEP	BEEP/general- purpose port pin function selection	14H	R/W	0	Fixed to "0"			0	0	0					
				BEEP0SEL							Selects function of P0B3/BEEP pin	General-purpose I/O port	BEEP		
				BEEP0CK1							Sets BEEP pin	0 Low level output	0 High level output	1 1.5 kHz	1 3 kHz
				BEEP0CK0							0 1	0 0	1 1		
Timer	Basic timer 0 carry	17H	R & Reset	0	Fixed to "0"			0	Retained	Retained					
				0											
				0											
				BTM0CY							Detects basic timer 0 carry FF	FF reset	FF set		

Table 8-1. Peripheral Hardware Control Functions of Control Registers (3/5)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				At Reset		Clock Stop
	Name	Address (BANK15)	Read/ Write	b ₃ b ₂ b ₁ b ₀ Symbol	Function	Set value		Reset by <div>RESET</div> pin	WDT & SP reset		
						0	1				
Timer	Basic timer 1 clock selection	18H	R/W	0	Fixed to "0"			0	0	Retained	
				0							
				0							
				BTM1CK0	Selects clock of basic timer 0	31.25 Hz (32 ms)	125 Hz (8 ms)				
	Timer 0 counter clock selection	2BH	R/W	TM0EN	Starts or stops timer 0 counter	Stops	Starts	0	0	0	
				TM0RES	Resets timer 0 counter	Not affected	Reset				
				TM0CK1	Sets basic clock of timer	0 0 1 1 TM0 TM1 75 kHz 25 kHz (13.3 μs) (40 μs)					
				TM0CK0	0 counter	0 1 0 1					
	Timer 0 mode selection	2CH	R/W	TM0OVF	Detects timer 0 overflow	No overflow	Overflow	0	0	0	
				TM0GCEG	Sets edge of gate close input signal	Rising edge	Falling edge				
				TM0GOEG	Sets edge of gate open input signal						
				TM0MD	Selects modulo counter/gate counter of timer 0	Modulo counter	Gate counter				
Interrupt	Interrupt edge selection	1FH	R/W	0	Fixed to "0"			0	0	Retained	
				0							
				0							
				IEG0	Sets interrupt issuance edge (INT pin)	Rising edge	Falling edge				
	Interrupt enable	2FH	R/W	IPSIO1	Enables serial interface 1 interrupt	Disables interrupt	Enables interrupt	0	0	Retained	
				IPBTM1	Enables basic interface timer 1 interrupt						
				IPTM0	Enables timer 0 interrupt						
				IP0	Enables INT pin interrupt						
	Serial interface 1 interrupt request	3CH	R/W	0	Fixed to "0"			0	0	Retained	
				0							
				0							
				IRQSIO1	Detects serial interface 1 interrupt request	No interrupt request	Interrupt request				

Table 8-1. Peripheral Hardware Control Functions of Control Registers (4/5)


Peripheral Hardware	Control Register				Peripheral Hardware Control Function			At Reset		Clock			
	Name	Address (BANK15)	Read/ Write	b ₃ b ₂ b ₁ b ₀ Symbol	Function	Set value		Reset by	WDT & SP reset	Stop			
						0	1	 pin					
Interrupt	Basic interval timer 1 interrupt request	3DH	R/W	0	Fixed to “0”			0	0	Retained			
				0									
				0									
				IRQBTM1							Detects basic interval timer 1 interrupt request	No interrupt request	Interrupt request
	Timer 0 interrupt request	3EH	R/W	0	Fixed to “0”			0	0	Retained			
				0									
				0									
				IRQTM0							Detects timer 0 interrupt request	No interrupt request	Interrupt request
	INT0 pin interrupt request	3FH	R/W	INT0	Detects INT0 pin status	Low level	High level	Undefined	Undefined	Undefined			
				0	Fixed to “0”			0	0	Retained			
				0									
				IRQ0	Detects INT0 pin interrupt request	No interrupt request	Interrupt request						
IF counter		20H	R/W	0	Fixed to “0”			0	0	0			
				0									
				0									
				FCGCH0							Note 1	Note 1	
	IF counter gate status detection	21H	R	0	Fixed to “0”			0	0	0			
				0									
				0									
				IFCGOSTT							Detects IF counter gate status	Closed	Open
	IF counter mode selection	22H	R/W	IFCMD1	Sets IF counter mode	0 0 1 1 general-AMIFC FMIFC AMIFC2	0	0	0				
				IFCMD0		purpose input port							
				IFCCK1	Sets IF counter gate time	0 0 1 1 1ms 4 ms 8 ms Open							
				IFCCK0		0 1 0 1							
	IF counter control	23H	W	0	Fixed to “0”			0	0	0			
				0									
				IFCSTRT							Starts or stops IF counter	Nothing affected	Starts counter
				IFCRES							Resets IF counter data	Nothing affected	Resets counter
A/D converter channel selection	24H	R/W	0	Fixed to “0”			0	0	Retained				
			0										
			ADCCH1							Selects pin used for A/D converter	0: A/D converter not used 1: P0D1/AD0 pin 2: P0D2/AD1 pin 3: P0D3/AD2 pin		
			ADCCH0										

Table 8-1. Peripheral Hardware Control Functions of Control Registers (5/5)

Peripheral Hardware	Control Register				Peripheral Hardware Control Function				At Reset		Clock		
	Name	Address (BANK15)	Read/ Write	b ₃ b ₂ b ₁ b ₀ Symbol	Function	Set value		Power- ON reset	WDT & SP reset	Stop			
						0	1						
A/D converter	A/D converter mode selection	25H	R	0	Fixed to "0"			0	0	0			
				0									
				ADCSTR							Detects operating status of A/D converter	Conversion ends	Converting
				ADCCMP							Detects comparison result of A/D converter	V _{ADCREf} > V _{ADCI}	V _{ADCREf} < V _{ADCI}
LCD driver	LCD driver display start	40H	R/W	LCDDBC	Note 2	Note 2		0	0	0			
				0									
				0									
				LCDEN							Sets ON/OFF of all the LCD display	Display OFF	Display ON
	LCD port selection	69H	R/W	0	Fixed to "0"			0	0	0			
				LCDD19SEL							Selects function of P2A2/LCD19 pin	General- purpose input port	LCD Segment
				LCDD18SEL							Selects function of P2A2/LCD18 pin		
				LCDD17SEL							Selects function of P2A2/LCD17 pin		
	Input/ output port	Port 0D pull- down resistor selection	6AH	R/W	P0DPLD3	Selects pull-down resistor of P0D3 pin	Pull-down resistor used	Pull-down resistor not used	0	0	Retained		
					P0DPLD2	Selects pull-down resistor of P0D2 pin							
P0DPLD1					Selects pull-down resistor of P0D1 pin								
P0DPLD0					Selects pull-down resistor of P0D0 pin								
Port 2C bit I/O selection		6BH	R/W	P2CBIO3	Selects I/O of port P2C3	Input	Output	0	0	Retained			
				P2CBIO2	Selects I/O of port P2C2								
				P2CBIO1	Selects I/O of port P2C1								
				P2CBIO0	Selects I/O of port P2C0								
Port 2B bit I/O selection		6CH	R/W	P2BBIO3	Selects I/O of port P2B3	Input	Output	0	0	Retained			
				P2BBIO2	Selects I/O of port P2B2								
				P2BBIO1	Selects I/O of port P2B1								
				P2BBIO0	Selects I/O of port P2B0								
Port 1D bit I/O selection		6DH	R/W	P1DBIO3	Selects I/O of port P1D3	Input	Output	0	0	Retained			
				P1DBIO2	Selects I/O of port P1D2								
				P1DBIO1	Selects I/O of port P1D1								
				P1DBIO0	Selects I/O of port P1D0								
Port 1A bit I/O selection		6EH	R/W	P1ABIO3	Selects I/O of port P1A3	Input	Output	0	0	Retained			
				P1ABIO2	Selects I/O of port P1A2								
				P1ABIO1	Selects I/O of port P1A1								
				P1ABIO0	Selects I/O of port P1A0								
Port 0B bit I/O selection	6FH	R/W	P0BBIO3	Selects I/O of port P0B3	Input	Output	0	0	Retained				
			P0BBIO2	Selects I/O of port P0B2									
			P0BBIO1	Selects I/O of port P0B1									
			P0BBIO0	Selects I/O of port P0B0									

★ Notes 1. Never write anything to address 20H of BANK15.

★ 2. Bit 3 of address 40H of BANK15 is a test mode area, therefore, do not write "1" to bit 3.

8.4 LCD Segment Registers

The LCD segment registers consist of a total of 20 nibbles (20 x 4 bits) of addresses 5CH through 6FH of BANK14 of the data memory. For details, refer to **19. LCD CONTROLLER/DRIVER**.

8.5 Cautions on Using Control Register

Keep in mind the following points (1) through (4) when using the write-only (W), read-only (R), and unused registers of the control registers (addresses 00H through 6FH of BANK15).

- (1) An "undefined value" is read if a write-only register is read.
- (2) Nothing is affected even if a read-only register is written.
- (3) An "undefined value" is read if an unused register is read. Nor is anything affected if this register is written.
- (4) Never write anything to address 31H of BANK15 because this address is a test mode area.

9. DATA BUFFER (DBF)

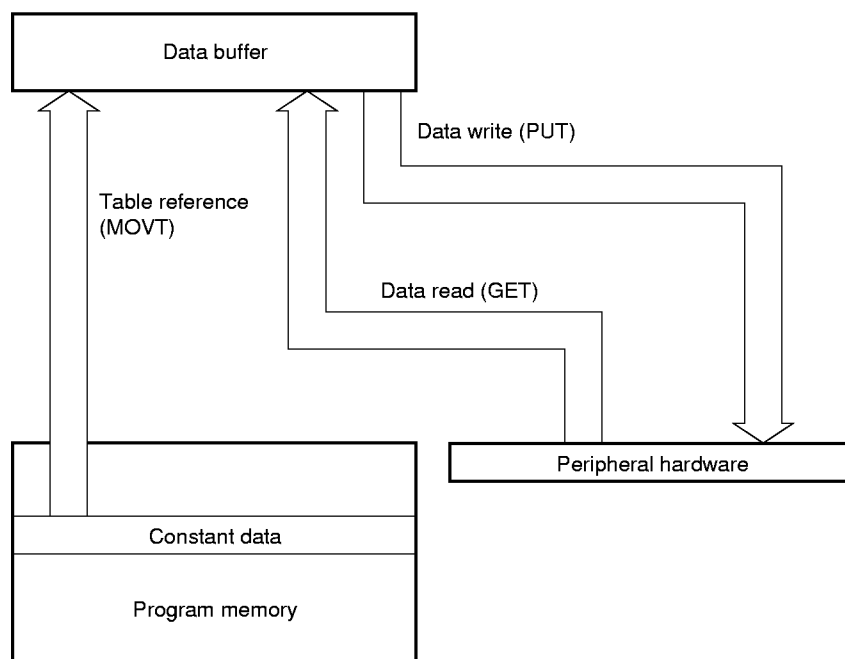
9.1 Outline of Data Buffer

Figure 9-1 outlines the data buffer.

The data buffer is located on the data memory and has the following two functions.

- Reads constant data on the program memory (table reference)
- Transfers data with the peripheral hardware units

Figure 9-1. Outline of Data Buffer



9.2 Data Buffer

9.2.1 Configuration of data buffer

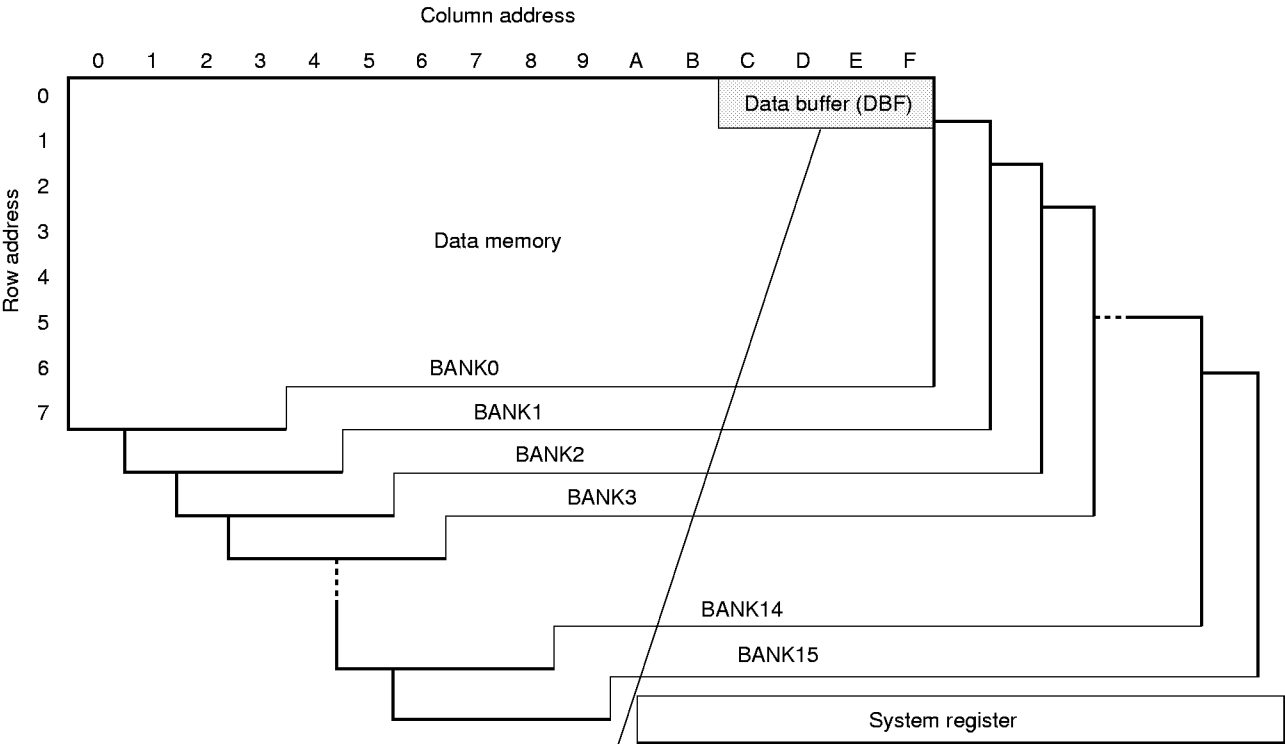
Figure 9-2 shows the configuration of the data buffer.

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

The 16-bit data is configured with bit 3 of address 0CH as the MSB and bit 0 of address 0FH as the LSB.

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

Figure 9-2. Configuration of Data Buffer



Remark Address 5BH is not provided to BANK4 through BANK14.

Data memory	Address	0CH				0DH				0EH				0FH			
	Bit	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀	b ₃	b ₂	b ₁	b ₀
Data buffer	Bit	b ₁₅	b ₁₄	b ₁₃	b ₁₂	b ₁₁	b ₁₀	b ₉	b ₈	b ₇	b ₆	b ₅	b ₄	b ₃	b ₂	b ₁	b ₀
	Signal	DBF3				DBF2				DBF1				DBF0			
	Data	⌈ M S B ⌋				Data								⌈ L S B ⌋			

9.2.2 Table reference instruction (“MOV_T DBF, @AR”)

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

The number of bits transferred by the table reference instruction can be specified by MOV_T selection register (address 07H) of the control registers.

When 8-bit data is transferred, it is read to DBF1 and 0.

When the table reference instruction is used, one stack level is used.

All the addresses of the program memory can be referenced by the table reference instruction.

9.2.3 Peripheral hardware control instructions (“PUT” and “GET”)

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

(1) GET DBF, p

Reads the data of a peripheral register addressed by “p” to the data buffer.

(2) PUT p, DBF

Sets the data of the data buffer to a peripheral register addressed by “p”.

9.3 Relationships between Peripheral Hardware and Data Buffer

Table 9-1 shows the relationships between the peripheral hardware and the data buffer.

MEMO

Table 9-1. Relationships between Peripheral Hardware and Data Buffer (1/2)

Peripheral Hardware	Peripheral Register Transferring Data with Data Buffer					
	Name	Symbol	Peripheral address	Execution of PUT/GET instruction	I/O bit	Actual bit
A/D converter	A/D converter data register	ADCR	02H	PUT/GET	8	8
Serial interface 1	Presetable shift register 1	SIO1SFR	04H	PUT/GET	8	8
Timer 0	Timer 0 modulo register	TM0M	1AH	PUT/GET	8	8
	Timer 0 counter	TM0C	1BH	GET	8	8
Address register	Address register	AR	40H	PUT/GET	16	16
Data buffer stack	DBF stack	DBFSTK	41H	PUT/GET	16	16
PLL frequency synthesizer ^{Note}	PLL data register	PLLRL	42H	PUT/GET	16	16
Frequency counter	IFC data register	IFC	43H	GET	16	16

Note The programmable counter of the PLL frequency synthesizer is configured of 17 bits, of which the high-order 16 bits indicate the PLL data register (PLLRL) and the low-order bits are allocated to the PLLSCNF flag (the third bit of address 10H).

For details, refer to **16. PLL FREQUENCY SYNTHESIZER**.

Table 9-1. Relationships between Peripheral Hardware and Data Buffer (2/2)

At Reset		Clock Stop	Function
Reset by RESET pin	WDT&SP reset		
0	0	Retained	Sets compare voltage V_{ADCREf} of A/D converter
Undefined	Undefined	Retained	Sets serial-out data and reads serial-in data
FF	FF	FF	Sets modulo register value of timer 0
0	0	0	Reads count value of timer 0 counter
0	0	Retained	Transfers data with address register
Undefined	Undefined	Retained	Saves data of data buffer
Undefined	Undefined	Retained	Sets division value (N value) of PLL
0	0	0	Reads count value of frequency counter

9.4 Cautions on Using Data Buffer

Keep the following points in mind concerning the unused peripheral addresses, write-only peripheral register (PUT only), and read-only peripheral register (GET only) when transferring data with the peripheral hardware via data buffer.

- An “undefined value” is read if a write-only register is read.
- Nothing is affected even if a read-only register is written.
- An “undefined value” is read if an unused address is read. Nor is anything affected if this address is written.

10. DATA BUFFER STACK

10.1 Outline of Data Buffer Stack

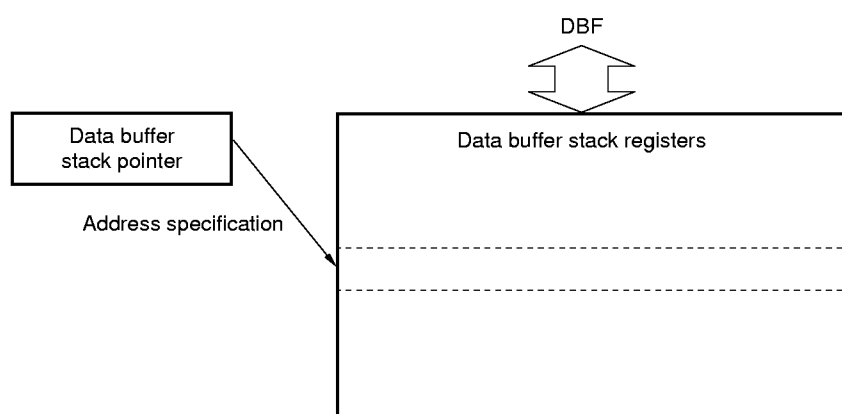
Figure 10-1 outlines the data buffer stack.

As shown in the figure, the data buffer stack consists of a data buffer stack pointer and data buffer stack registers.

The data buffer stack saves or restores the contents of the data buffer when the “PUT” or “GET” instruction is executed.

Therefore, the contents of the data buffer can be saved by one instruction when an interrupt is accepted.

Figure 10-1. Outline of Data Buffer Stack



10.2 Data Buffer Stack Register

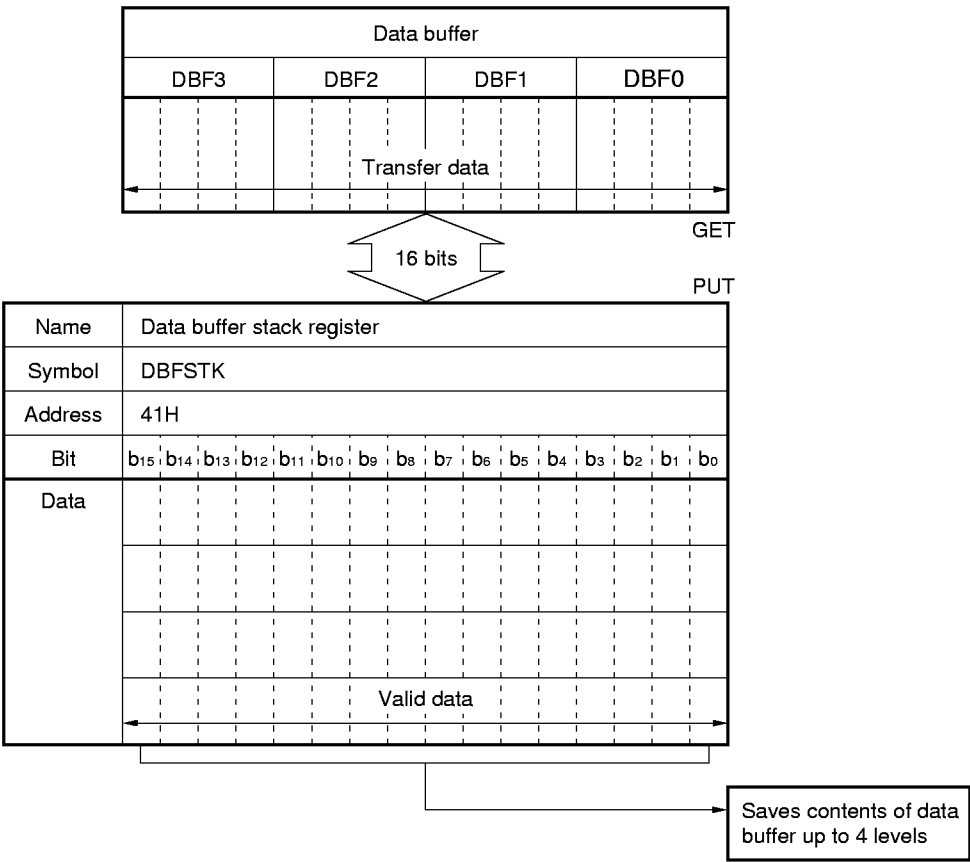
Figure 10-2 shows the configuration of the data buffer stack registers.

As shown in the figure, the data buffer stack registers consist of four 16-bit registers.

The contents of the data buffer are saved by executing the “PUT” instruction, and the saved data is restored by executing the “GET” instruction.

The data buffer contents can be successively saved up to 4 levels.

Figure 10-2. Configuration of Data Buffer Stack Register



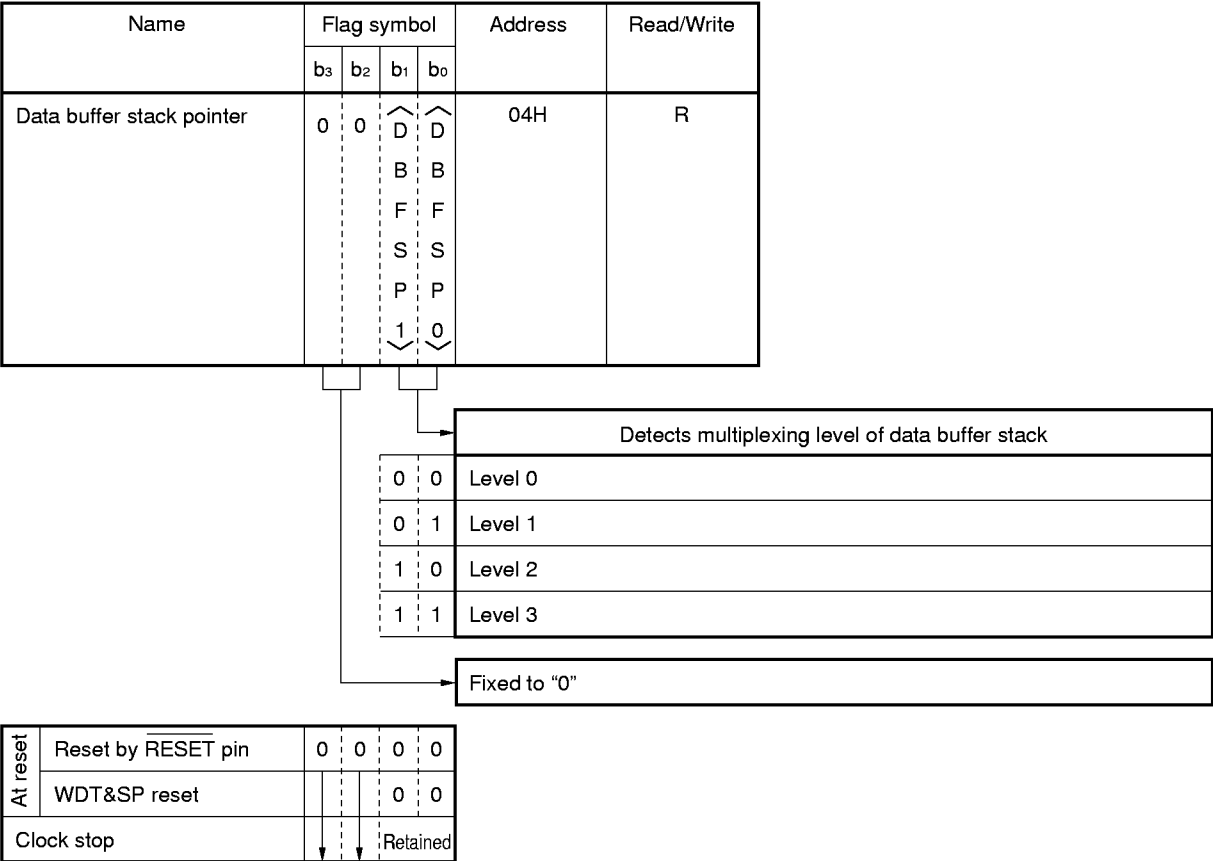
10.3 Data Buffer Stack Pointer

The data buffer stack pointer detects the multiplexing level of the data buffer stack registers.

When the “PUT” instruction is executed to the data buffer stack, the value of the data buffer stack pointer is incremented by one; when the “GET” instruction is executed, the value of the pointer is decremented by one.

The data buffer stack pointer can be only read and cannot be written.

The configuration and function of the data buffer stack pointer are illustrated below.



10.4 Operation of Data Buffer Stack

Figure 10-3 shows the operation of the data buffer stack.

As shown in the figure, when the PUT instruction is executed, the contents of the data buffer are transferred to a data buffer stack register specified by the stack pointer, and the stack pointer is incremented by one.

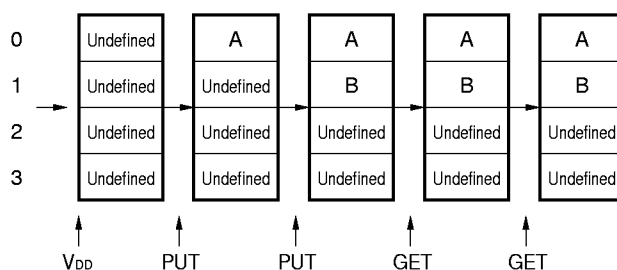
When the GET instruction is executed, the contents of a data buffer stack register specified by the stack pointer are transferred to the data buffer, and the stack pointer is decremented by one.

Therefore, note that the value of the stack pointer is set to 1 if data has been written once because its initial value is 0, and that the stack pointer is set to 0 when data has been written four times.

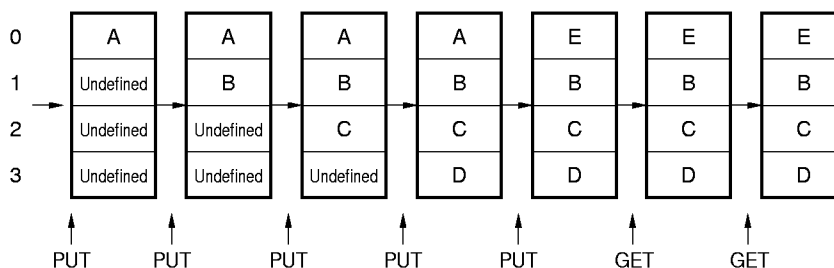
Note that when writing (PUT) exceeding four levels, the first data are discarded.

Figure 10-3. Operation of Data Buffer Stack

(a) If writing does not exceed level 4



(b) If writing exceeds level 4



10.5 Using Data Buffer Stack

A program example is shown below.

Example To save the contents of the data buffer and address register by using INT0 interrupt routine (the contents of the data buffer and address register are not automatically saved when an interrupt occurs).

```

START:
    BR    INITIAL      ; Reset address
                    ; Interrupt vector address
    NOP                    ; SIO
    NOP                    ; Basic timer 1
    NOP                    ; TM0

INTINT:
                    ; INT pin interrupt vector address (0004H)
    PUT    DBFSTK, DBF  ; Saves contents of DBF to first level of data buffer
                    ; stack (DBFSTK)
    GET    DBF, AR      ; Transfers contents of address register (AR) to DBF
    PUT    DBFSTK, DBF  ; Saves contents of AR to second level of data buffer
                    ; stack

|              |
|--------------|
| Processing B |
|--------------|


                    ; INT interrupt processing

    GET    DBF, DBFSTK  ; Restores second level of data buffer stack to data buffer,
    PUT    AR, DBF      ; and restores contents of data buffer to address register
    GET    DBF, DBFSTK  ; Restores first level of data buffer stack to data buffer
    EI
    RETI

INITIAL:
    SET1   IP0
    EI

LOOP:
    

|              |
|--------------|
| Processing A |
|--------------|



    BR     LOOP

END

```

10.6 Cautions on Using Data Buffer Stack

The contents of the data buffer stack are not automatically saved when an interrupt is accepted, and therefore, must be saved by software.

Even when a bank of the data memory other than BANK0 is specified, the contents of the data buffer (existing in BANK0) can be saved or restored by using the "PUT" and "GET" instructions.

11. GENERAL-PURPOSE PORT

The general-purpose ports output high-level, low-level, or floating signals to external circuits, and read high-level or low-level signals from external circuits.

11.1 Outline of General-purpose Port

Table 11-1 shows the relationships between each port and port register.

The general-purpose ports are classified into I/O, input, and output ports.

The I/O ports can be set in the input or output mode in 1-bit (1-pin) units. The input or output mode of each I/O port is specified by the port input/output selection registers (addresses 60H through 6FH) of BANK15.

Table 11-1. Relationships between Port (Pin) and Port Register (1/2)

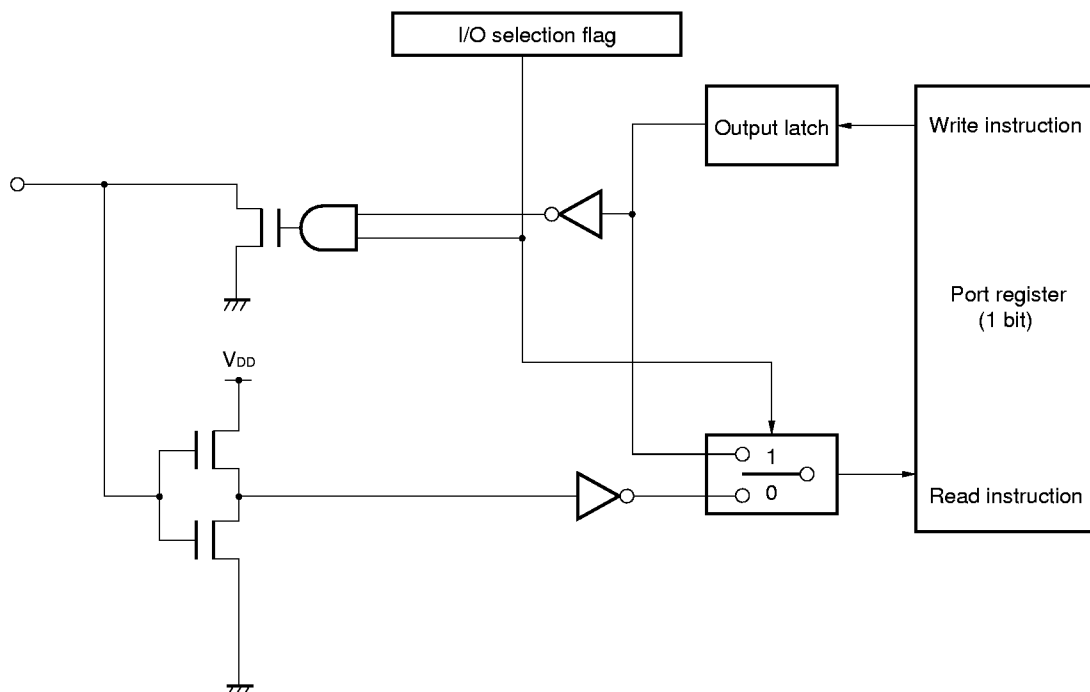
Port	Pin			Data Setting Method				
	No.	Symbol	I/O	Port register (data memory)				
				Bank	Address	Symbol	Bit symbol (reserved word)	
Port 0A	No pin		Output	BANK0	70H	P0A	b ₃	—
	No pin						b ₂	—
	19	P0A1					b ₁	P0A1
	18	P0A0					b ₀	P0A0
Port 0B	31	P0B3	I/O (bit I/O)		71H	P0B	b ₃	P0B3
	30	P0B2					b ₂	P0B2
	29	P0B1					b ₁	P0B1
	28	P0B0					b ₀	P0B0
Port 0C	67	P0C3	Output		72H	P0C	b ₃	P0C3
	66	P0C2					b ₂	P0C2
	65	P0C1					b ₁	P0C1
	64	P0C0					b ₀	P0C0
Port 0D	78	P0D3	Input		73H	P0D	b ₃	P0D3
	77	P0D2					b ₂	P0D2
	76	P0D1					b ₁	P0D1
	75	P0D0					b ₀	P0D0

Table 11-1. Relationships between Port (Pin) and Port Register (2/2)

Port	Pin			Data Setting Method						
	No.	Symbol	I/O	Port register (data memory)						
				Bank	Address	Symbol	Bit symbol (reserved word)			
Port 1A	23	P1A3	I/O (bit I/O)	BANK1	70H	P1A	b ₃	P1A3		
	22	P1A2					b ₂	P1A2		
	21	P1A1					b ₁	P1A1		
	20	P1A0					b ₀	P1A0		
Port 1C	4	P1C3	Input		BANK1	72H	P1C	b ₃	P1C3	
	3	P1C2						b ₂	P1C2	
	2	P1C1						b ₁	P1C1	
	1	P1C0						b ₀	P1C0	
Port 1D	27	P1D3	I/O (bit I/O)			BANK1	73H	P1D	b ₃	P1D3
	26	P1D2							b ₂	P1D2
	25	P1D1							b ₁	P1D1
	24	P1D0							b ₀	P1D0
Port 2A	No pin		Input	BANK2			70H	P2A	b ₃	—
	63	P2A2							b ₂	P2A2
	62	P2A1							b ₁	P2A1
	61	P2A0							b ₀	P2A0
Port 2B	17	P2B3	I/O (bit I/O)		BANK2		71H	P2B	b ₃	P2B3
	16	P2B2							b ₂	P2B2
	15	P2B1							b ₁	P2B1
	14	P2B0							b ₀	P2B0
Port 2C	74	P2C3	I/O (bit I/O)			BANK2	72H	P2C	b ₃	P2C3
	73	P2C2							b ₂	P2C2
	72	P2C1							b ₁	P2C1
	71	P2C0							b ₀	P2C0
—	No pin		—	BANK3 BANK15 ^{Note}			70H-73H	—	Fixed to 0	

Note Address 5BH is not provided to BANK4 through BANK14.

- ★ (2) P1A (P1A3, P1A2, P1A1, P1A0)
P1D (P1D3, P1D2, P1D1, P1D0)



11.2.2 Using I/O port

The input or output mode of the I/O ports is set by I/O selection register P0B, P1A, P1D, P2B or P2C of the control registers.

Because these are bit I/O ports, they can be set in the input or output mode in 1-bit units.

Setting the output data of or reading the input data of a port is carried out by executing an instruction that writes data to or reads data from the port.

11.2.3 shows the configuration of the I/O selection register of each port.

11.2.4 and 11.2.5 describe how each port is used as an input or output port.

11.2.6 describes the reset status of the I/O ports.

11.2.3 I/O port I/O selection register

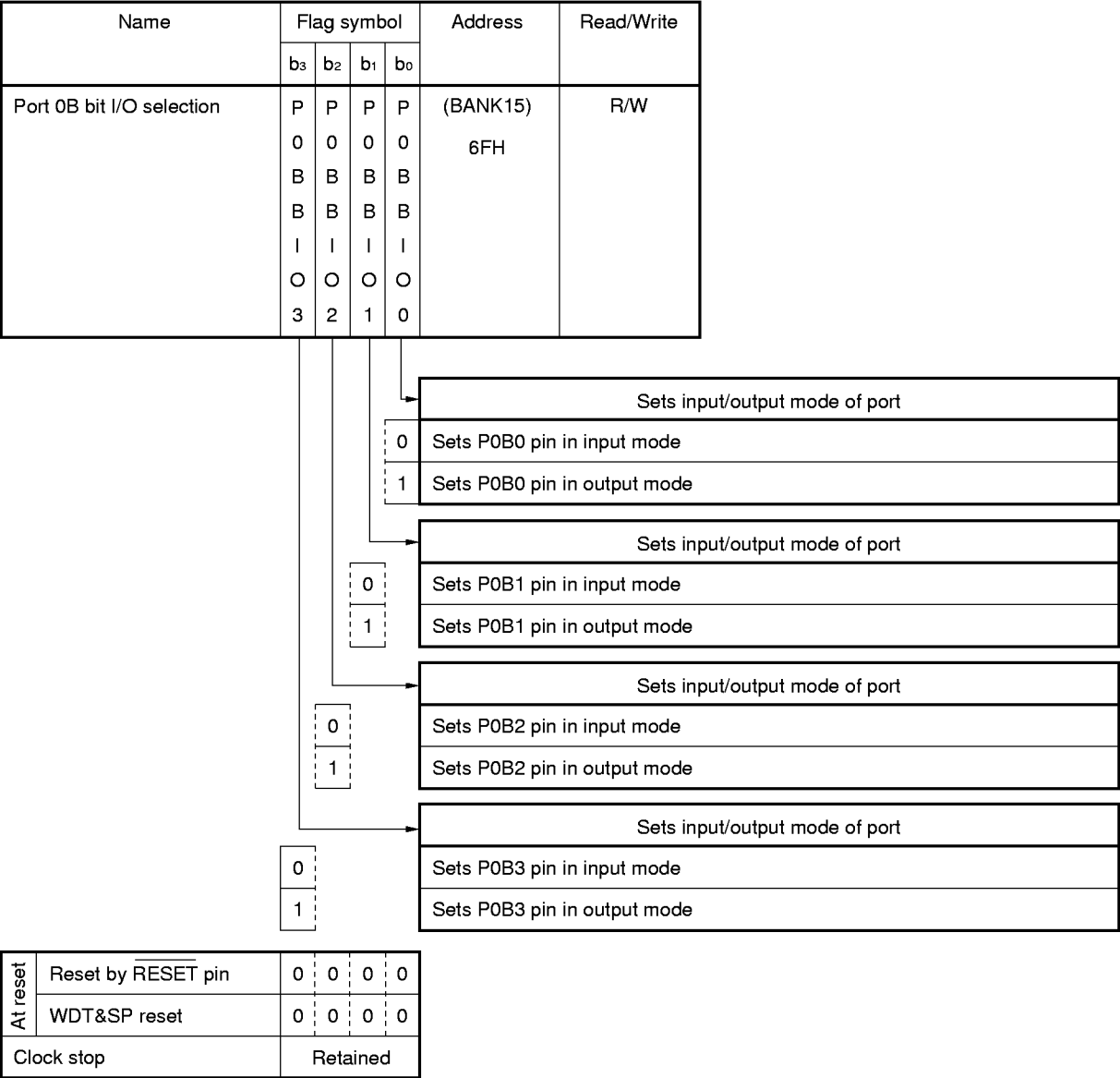
The following I/O selection registers of the I/O ports are available.

- Port 0B bit I/O selection register
- Port 1A bit I/O selection register
- Port 1D bit I/O selection register
- Port 2B bit I/O selection register
- Port 2C bit I/O selection register

Each I/O selection register sets the input or output mode of the corresponding port pin.

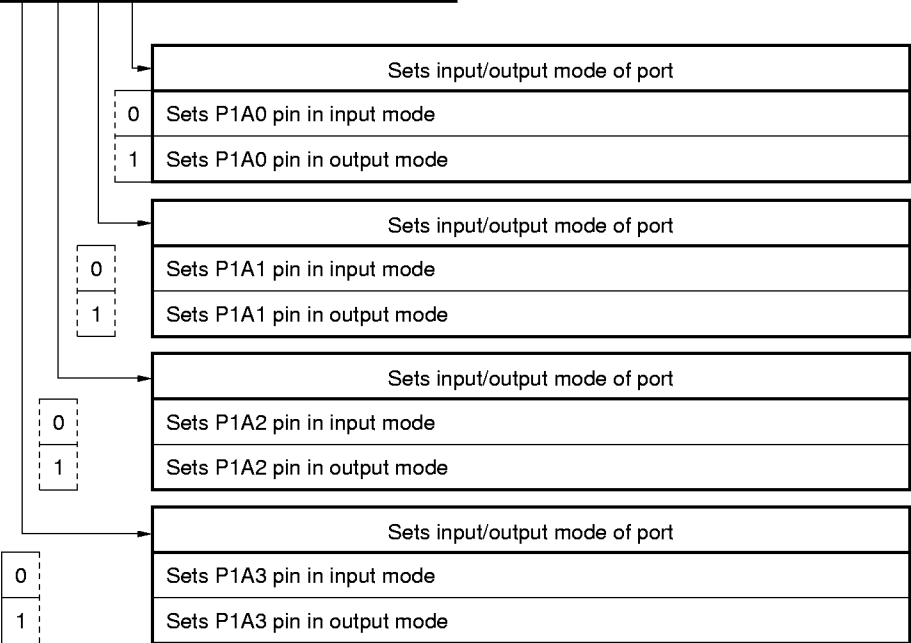
The following paragraphs (1) through (5) describe the configuration and functions of the above I/O selection registers.

(1) Port 0B bit I/O selection register



(2) Port 1A bit I/O selection register

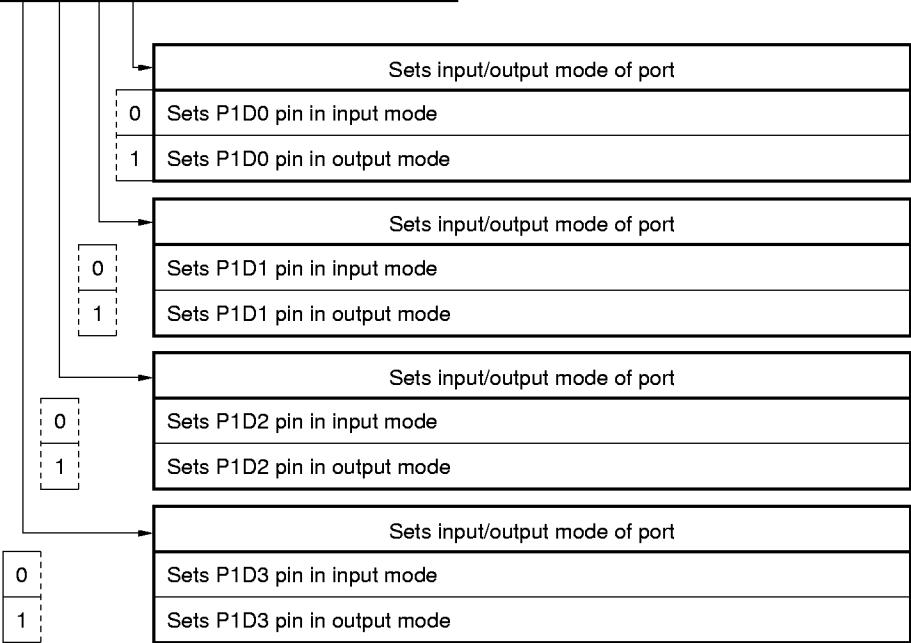
Name	Flag symbol				Address	Read/Write
	b ₃	b ₂	b ₁	b ₀		
Port 1A bit I/O selection	P	P	P	P	(BANK15) 6EH	R/W
	1	1	1	1		
	A	A	A	A		
	B	B	B	B		
	I	I	I	I		
	O	O	O	O		
	3	2	1	0		



At reset	Reset by RESET pin	0	0	0	0
	WDT&SP reset	0	0	0	0
Clock stop		Retained			

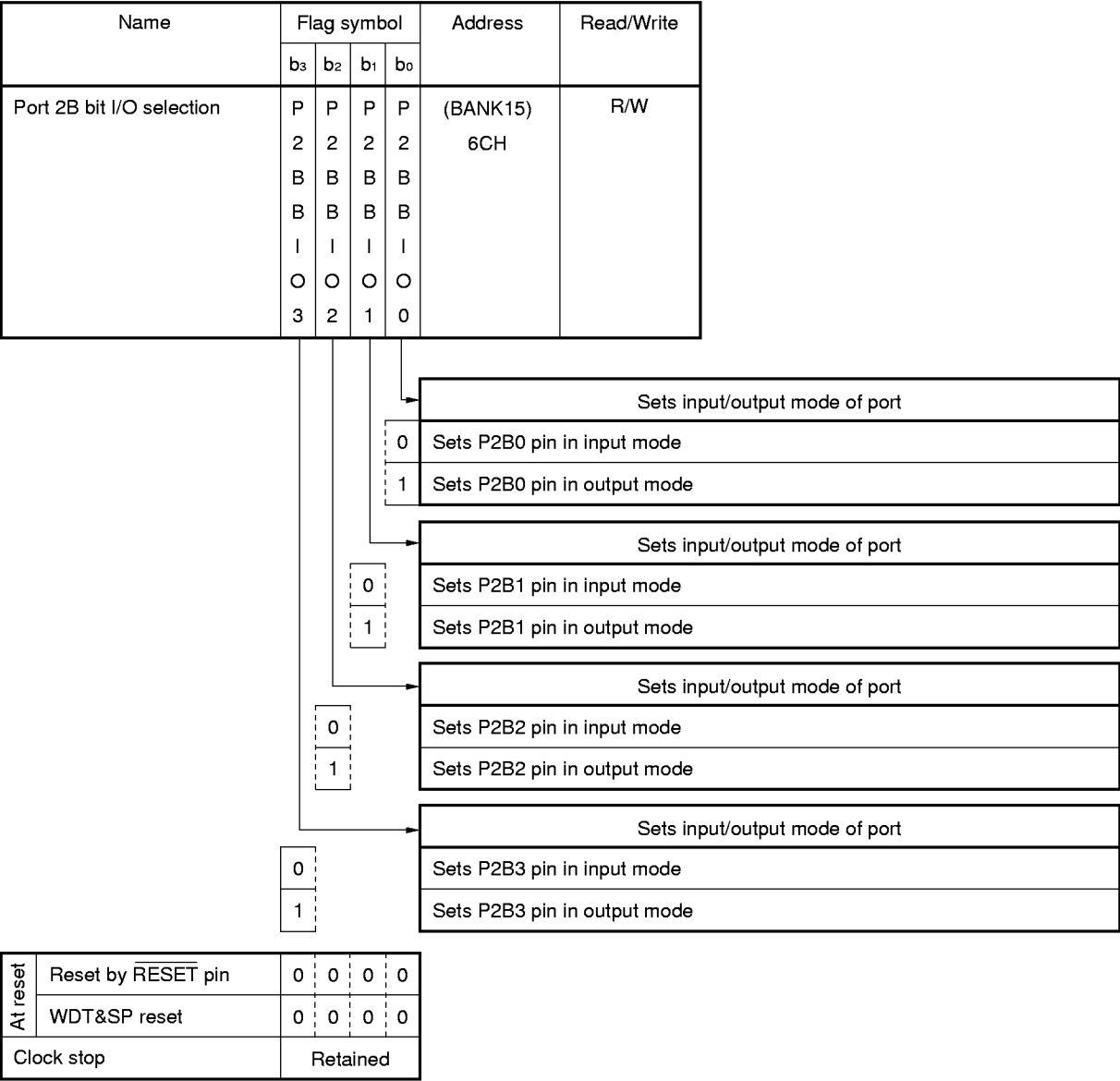
(3) Port 1D bit I/O selection register

Name	Flag symbol				Address	Read/Write
	b ₃	b ₂	b ₁	b ₀		
Port 1D bit I/O selection	P	P	P	P	(BANK15) 6DH	R/W
	1	1	1	1		
	D	D	D	D		
	B	B	B	B		
	I	I	I	I		
	O	O	O	O		
	3	2	1	0		

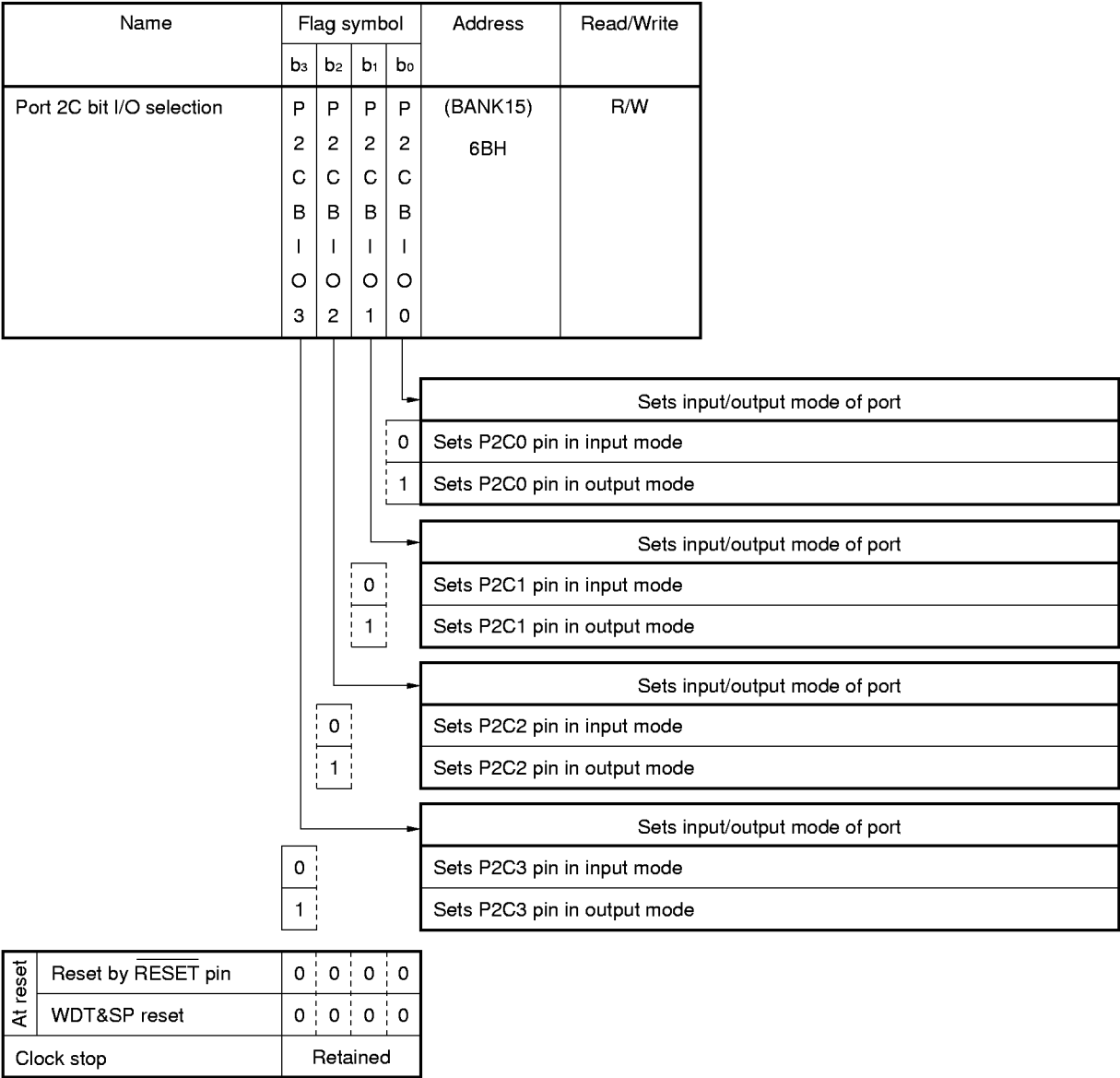


At reset	Reset by RESET pin	0	0	0	0
	WDT&SP reset	0	0	0	0
Clock stop		Retained			

(4) Port 2B bit I/O selection register



(5) Port 2C bit I/O selection register



11.2.4 When using I/O port as input port

The port pin to be set in the input mode is selected by the I/O selection register corresponding to the port. Ports P0B, P1A, P1D, P2B and P2C can be set in the input or output mode in 1-bit units.

The pin set in the input mode is floated (Hi-Z) and waits for input of an external signal.

The input data is read by executing a read instruction (such as SKT) to the port register corresponding to the port pin.

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

When a write instruction (such as MOV) is executed to the port register corresponding to the pin set in the input mode, the contents of the output latch are rewritten.

11.2.5 When using I/O port as output port

The port pin to be set in the output mode is selected by the I/O selection register corresponding to the port. Ports P0B, P1A, P1D, P2B and P2C can be set in the input or output mode in 1-bit units.

The pin set in the output mode outputs the contents of the output latch.

The output data is set by executing a write instruction (such as MOV) to the port register corresponding to the port pin.

Write "1" to the port register to output a high level to the port pin; write "0" to output a low level. The port pin can be also floated (Hi-Z) if it is set in the input mode.

If a read instruction (such as SKT) is executed to the port register corresponding to a port pin set in the output mode, the contents of the output latch are read.

11.2.6 Status of I/O port at reset

(1) At reset by $\overline{\text{RESET}}$ pin

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

The contents of the output latch are reset to "0".

(2) At WDT&SP reset

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

The contents of the output latch are reset to "0".

(3) On execution of clock stop instruction

The setting of the input or output mode is retained.

The contents of the output latch are also retained.

(4) In halt status

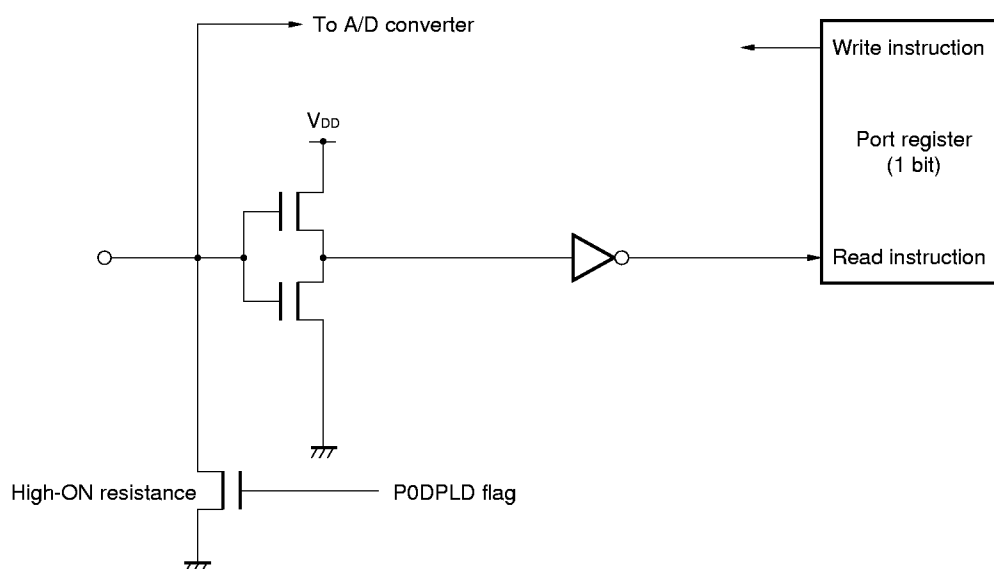
The previous status is retained.

11.3 General-Purpose Input Port (P0D, P1C, P2A)

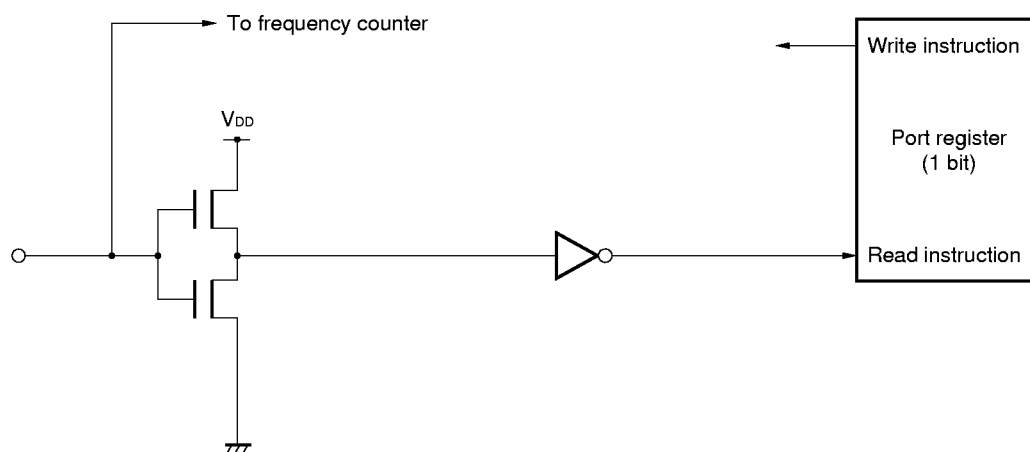
11.3.1 Configuration of input port

The following paragraphs (1) and (2) show the configuration of the input port.

(1) P0D (P0D3, P0D2, P0D1, P0D0)



(2) P1C (P1C3, P1C2, P1C1, P1C0) P2A (P2A2, P2A1, P2A0)



11.3.2 Using input port

The input data is read by executing a read instruction (such as SKT) to the port register corresponding to the port pin.

“1” is read from the port register when a high level is input to the corresponding port pin; when a low level is input to the port pin, “0” is read from the register.

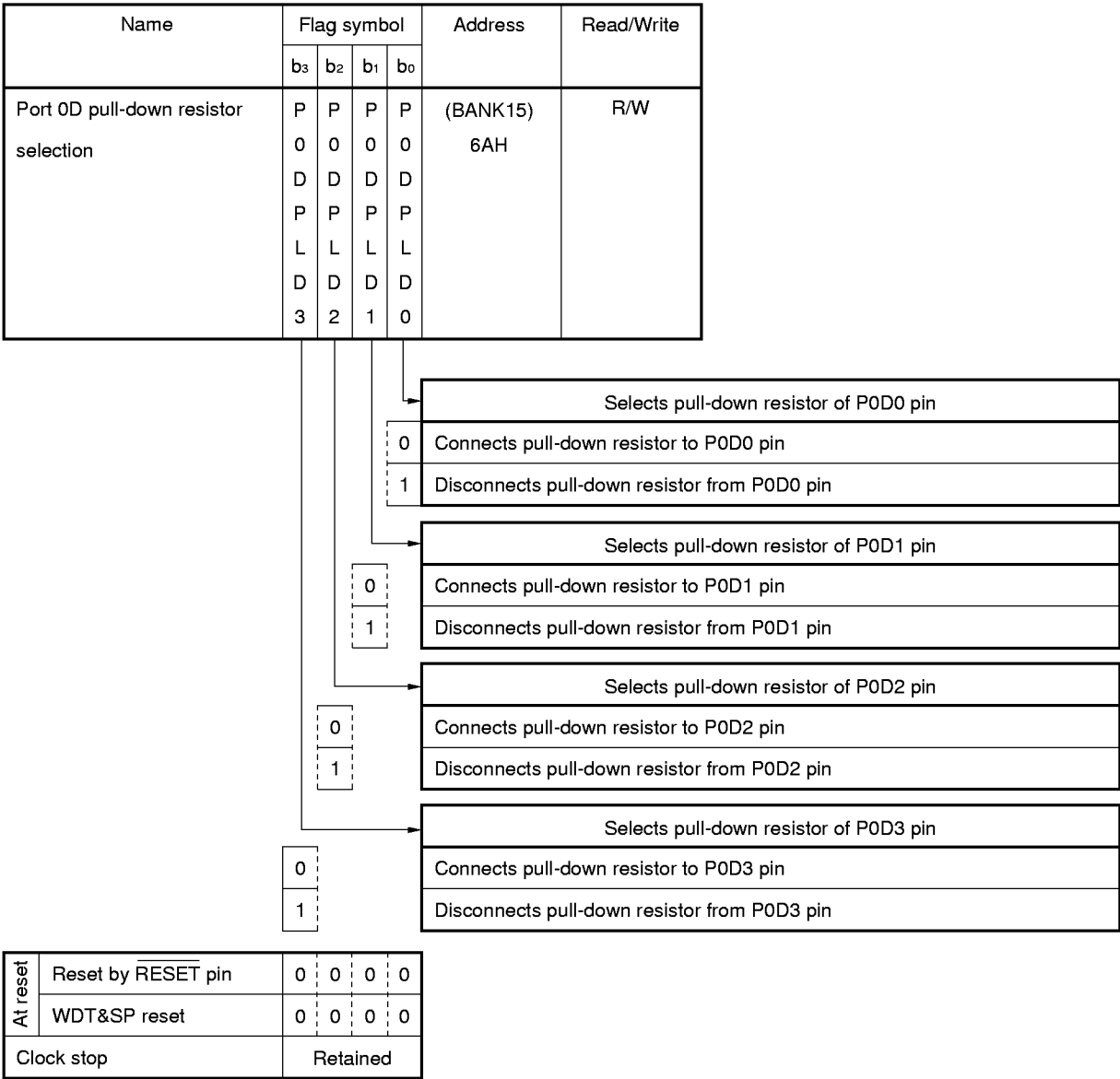
Nothing is affected even if a write instruction (such as MOV) is executed to the port register.

P0D has a pull-down resistor that can be connected or disconnected by software in 1-bit units. The pull-down resistor is connected when “0” is written to the corresponding bit of the port 0D pull-down resistor selection register. When “1” is written to the corresponding bit of this register, the pull-down resistor is disconnected.

11.3.3 Port 0D pull-down resistor selection register

The port 0D pull-down resistor selection register specifies whether a pull-down resistor is connected to P0D3 through P0D0 pins. The configuration and function of this register are illustrated below.

- Port 0D pull-down resistor selection register



11.3.4 Status of input port at reset

(1) At reset by $\overline{\text{RESET}}$ pin

All the input ports are set in the input mode.

All the pull-down resistors of P0D are connected.

(2) At WDT&SP reset

All the input ports are set in the input mode.

All the pull-down resistors of P0D are connected.

(3) On execution of clock stop instruction

All the input ports are set in the input mode.

The pull-down resistors of P0D retain the previous status.

(4) In halt status

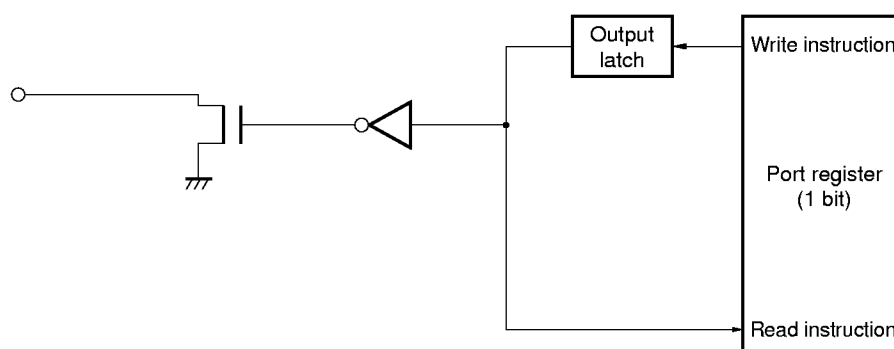
The previous status is retained.

11.4 General-Purpose Output Port (P0A, P0C)

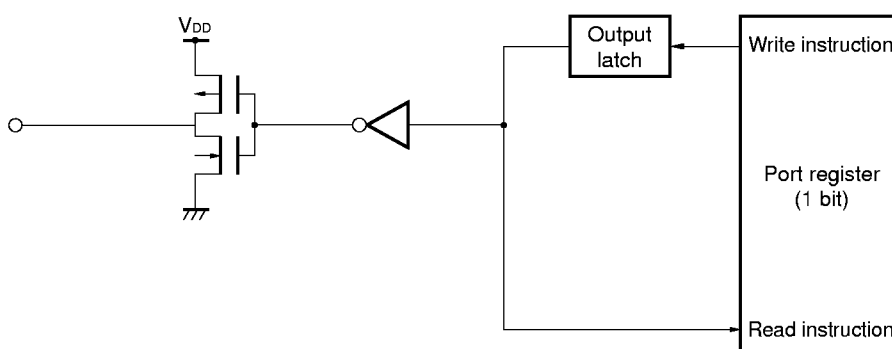
11.4.1 Configuration of output port

The configuration of the output port is shown below.

(1) P0A (P0A1, P0A0)



(2) P0C (P0C3, P0C2, P0C1, P0C0)



11.4.2 Using output port

The output port outputs the contents of the output latch to each pin.

The output data is set by executing a write instruction (such as MOV) to the port register corresponding to the port pin.

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

However, because P0A is an N-ch open-drain output port, it is floated when it outputs a high level. Therefore, an external pull-up resistor must be connected to this port.

If a read instruction (such as SKT) is executed to the port register, the contents of the output latch are read.

11.4.3 Status of output port at reset

(1) At reset by $\overline{\text{RESET}}$ pin

The contents of the output latch are output.

The contents of the output latch are reset to "0".

(2) At WDT&SP reset

The contents of the output latch are output.

The contents of the output latch are reset to "0".

(3) On execution of clock stop instruction

The contents of the output latch are output.

The contents of the output latch are retained.

(4) In halt status

The contents of the output latch are output.

The contents of the output latch are retained.

12. INTERRUPT

12.1 Outline of Interrupt Block

Figure 12-1 outlines the interrupt block.

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

The interrupt block consists of an "interrupt request servicing block" corresponding to each peripheral hardware unit, "interrupt enable flip-flop" that enables all interrupts, "stack pointer" that is controlled when an interrupt is accepted, "address stack registers", "program counter", and "interrupt stack".

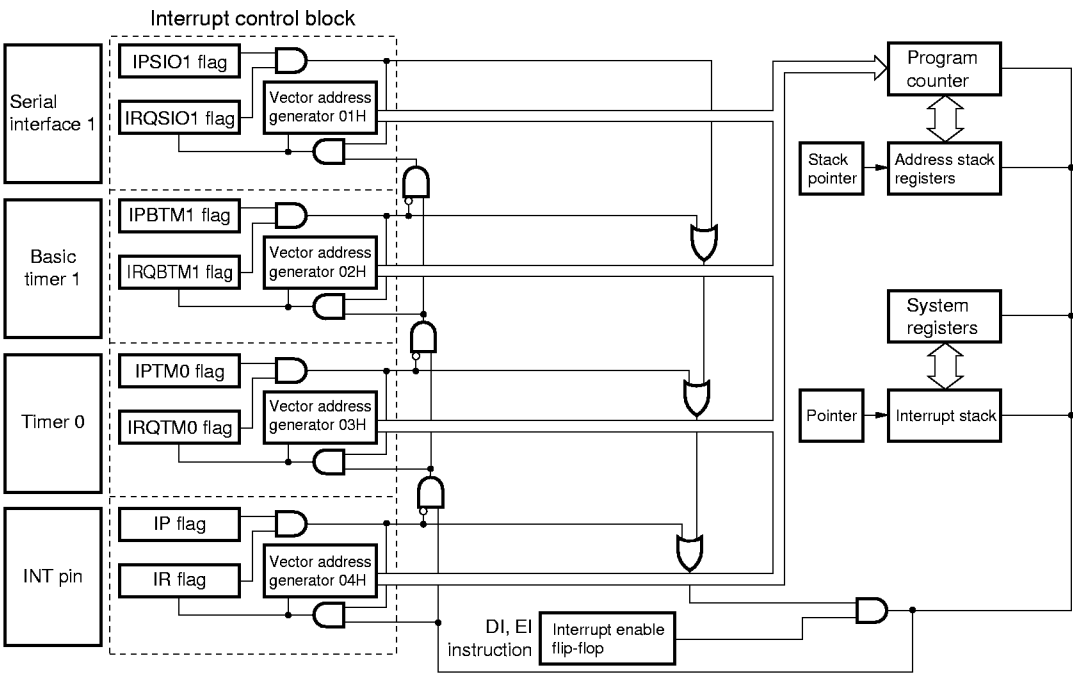
The "interrupt control block" of each peripheral hardware unit consists of an "interrupt request flag (IRQ $\times\times\times$)" that detects the corresponding interrupt request, "interrupt enable flag (IP $\times\times\times$)" that enables the interrupt, and "vector address generator (VAG)" that specifies a vector address when the interrupt is accepted.

The μ PD17934 has the following three types of maskable interrupts.

- INT interrupts
- Timer 0 and basic interval timer 1 interrupts
- Serial interface 1 interrupts

When an interrupt is accepted, execution branches to a predetermined address, and the interrupt is serviced.

Figure 12-1. Outline of Interrupt Block



12.2 Interrupt Control Block

An interrupt control block is provided for each peripheral hardware unit. This block detects issuance of an interrupt request, enables the interrupt, and generates a vector address when the interrupt is accepted.

12.2.1 Configuration and function of interrupt request flag (IRQ_{xxx})

Each interrupt request flag is set to 1 when an interrupt request is issued by the corresponding peripheral hardware unit, and is reset to 0 when the interrupt is accepted.

Writing the interrupt request flag to "1" directly is equivalent to issuance of the interrupt request.

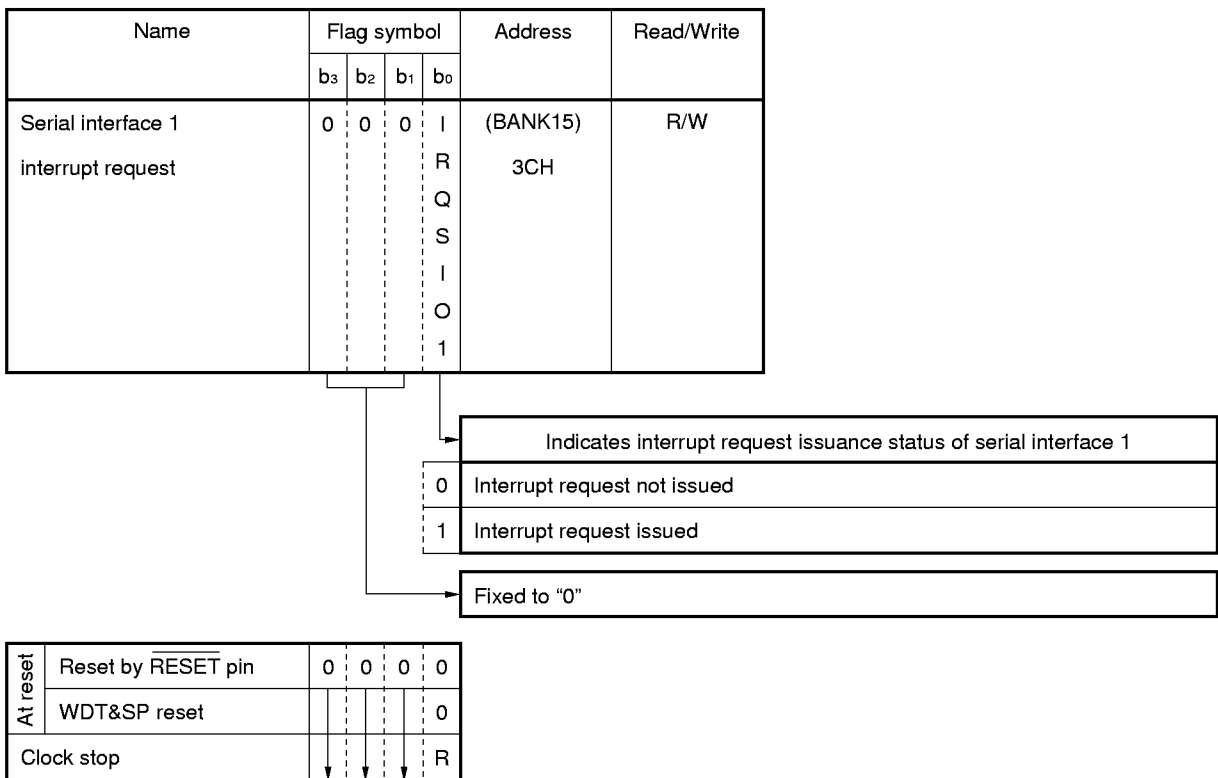
By detecting the interrupt request flag when an interrupt is not enabled, issuance status of each interrupt request can be detected.

Once the interrupt request flag has been set, it is not reset until the corresponding interrupt is accepted, or until "0" is written to the flag via a window register.

Even if two or more interrupt requests are issued at the same time, the interrupt request flag corresponding to the interrupt that has not been accepted is not reset.

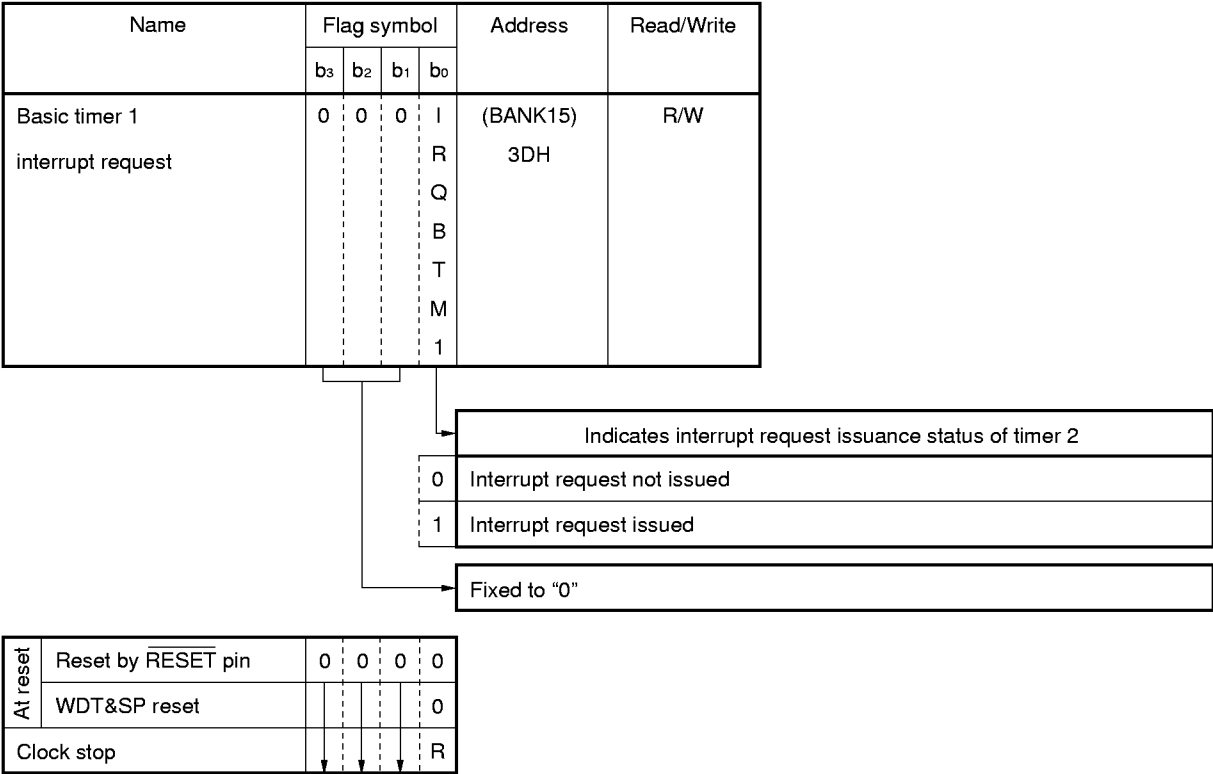
Figures 12-2 through 12-15 show the configuration and function of the respective interrupt request registers.

Figure 12-2. Configuration of Serial Interface 1 Interrupt Request Register



R: Retained

Figure 12-3. Configuration of Basic Timer 1 Interrupt Request Register



R: Retained

Figure 12-4. Configuration of Timer 0 Interrupt Request Register

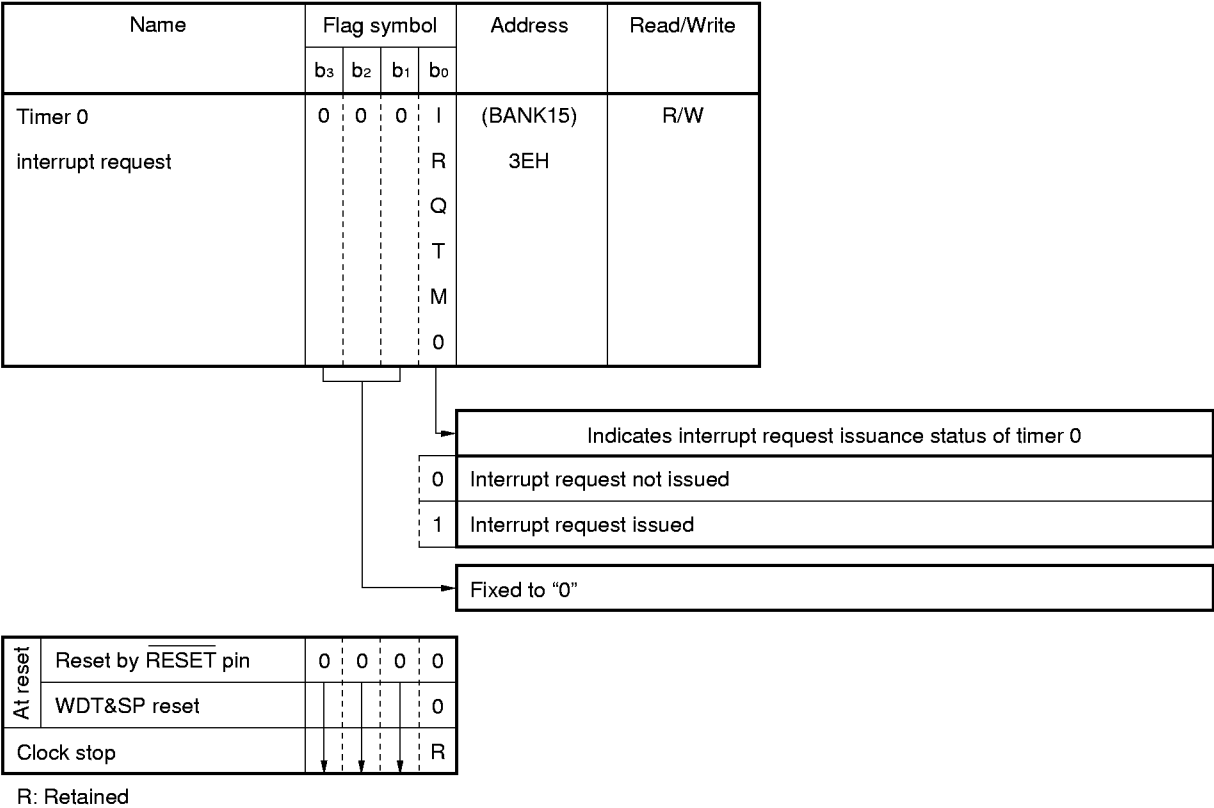
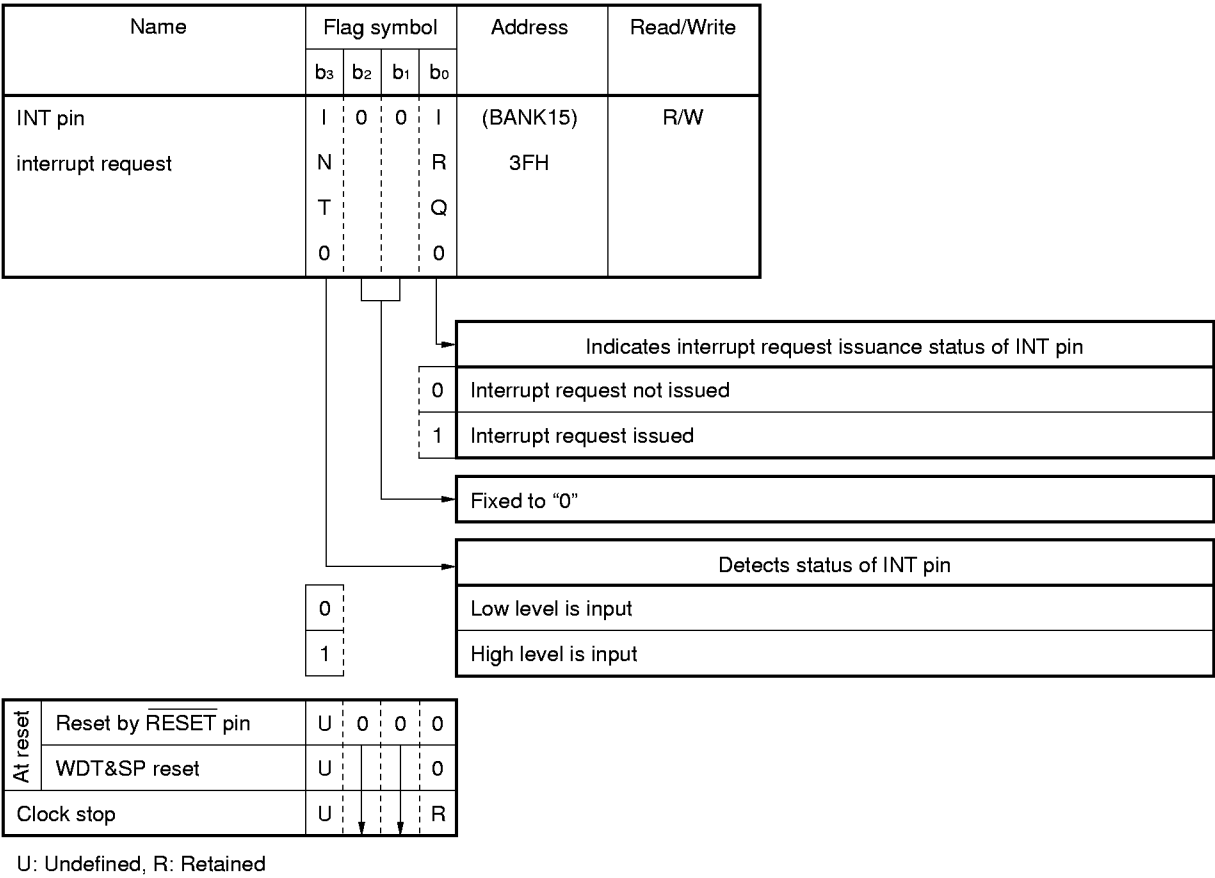


Figure 12-5. Configuration of INT Pin Interrupt Request Register



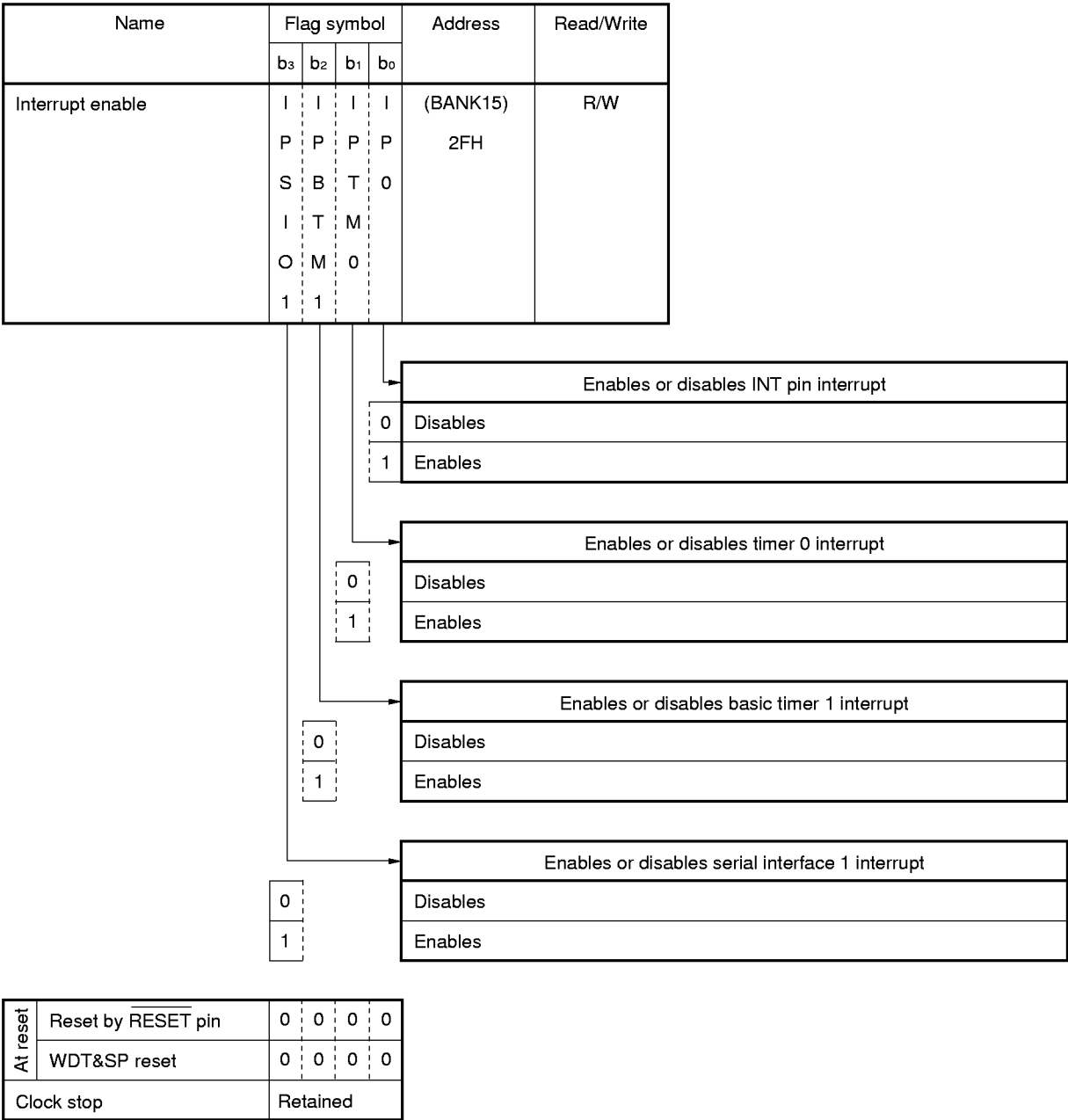
12.2.2 Function and configuration of interrupt request flag (IP_{xxx})

Each interrupt request flag enables the interrupt of the corresponding peripheral hardware unit. In order for an interrupt to be accepted, all the following conditions must be satisfied.

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

The interrupt enable flags are located on the interrupt enable register on the register file. Figures 12-6 shows the configuration of each interrupt enable register.

Figure 12-6. Configuration of Interrupt Enable Register



12.2.3 Vector address generator (VAG)

The vector address generator generates a branch address (vector address) of the program memory corresponding to an interrupt source that has been accepted from the corresponding peripheral hardware.

Table 12-1 shows the vector addresses of the respective interrupt sources.

Table 12-1. Interrupt Sources and Vector Addresses

Interrupt Source	Vector Address
INT pin	0004H
Timer 0	0003H
Basic timer 1	0002H
Serial interface 1	0001H

12.3 Interrupt Stack Register

12.3.1 Configuration and function of interrupt stack register

Figure 12-7 shows the configuration of the interrupt stack register.

The interrupt stack register saves the contents of the following system registers (except the address register (AR)) when an interrupt is accepted.

- Window register (WR)
- Bank register (BANK)
- Index register (IX)
- General pointer (RP)
- Program status word (PSWORD)

When an interrupt is accepted and the contents of the above system registers are saved to the interrupt stack, the contents of the above system registers, except the window register, are reset to “0”.

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

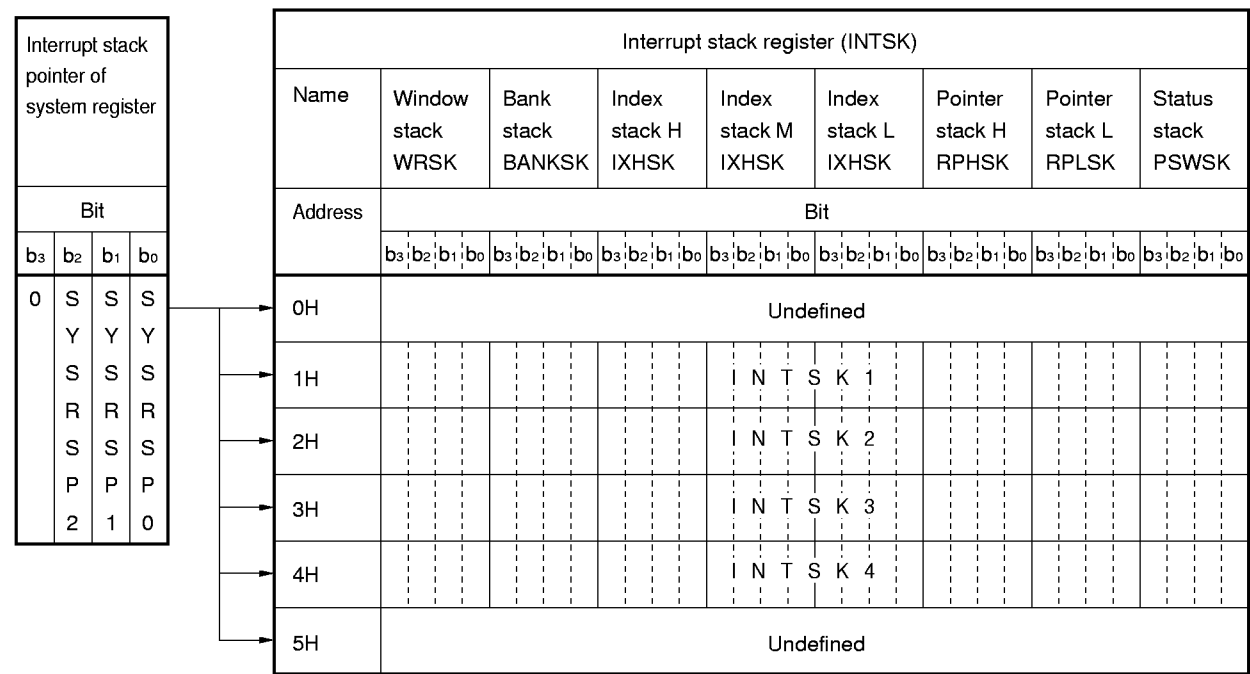
Therefore, interrupts can be nested up to four levels.

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

The contents of the interrupt stack register are undefined at reset by $\overline{\text{RESET}}$ pin.

The previous contents are retained on execution of the clock stop instruction.

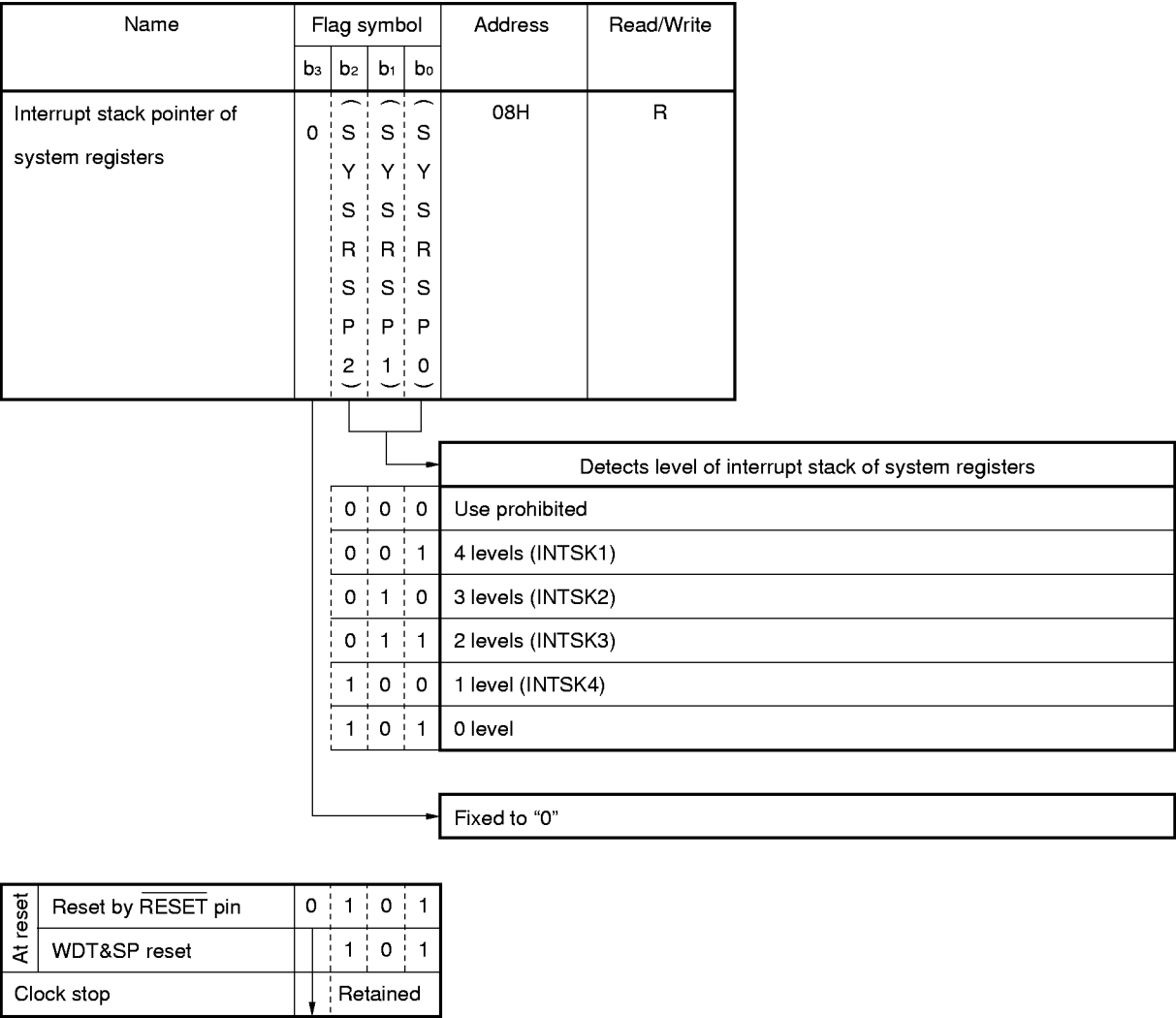
Figure 12-7. Configuration of Interrupt Stack Register



12.3.2 Interrupt stack pointer of system register

The interrupt stack pointer of the system register detects the nesting level of interrupts. The interrupt stack pointer can be only read and cannot be written.

The configuration and function of the interrupt stack pointer are illustrated below.



12.3.3 Interrupt stack operation

Figure 12-8 shows the operation of the interrupt stack.

When nested interrupts exceeding four levels are accepted, since the contents saved first are discarded they therefore must be saved by the program.

Figure 12-8. Operation of Interrupt Stack (1/2)

(a) Where interrupt nesting level is 4 or less

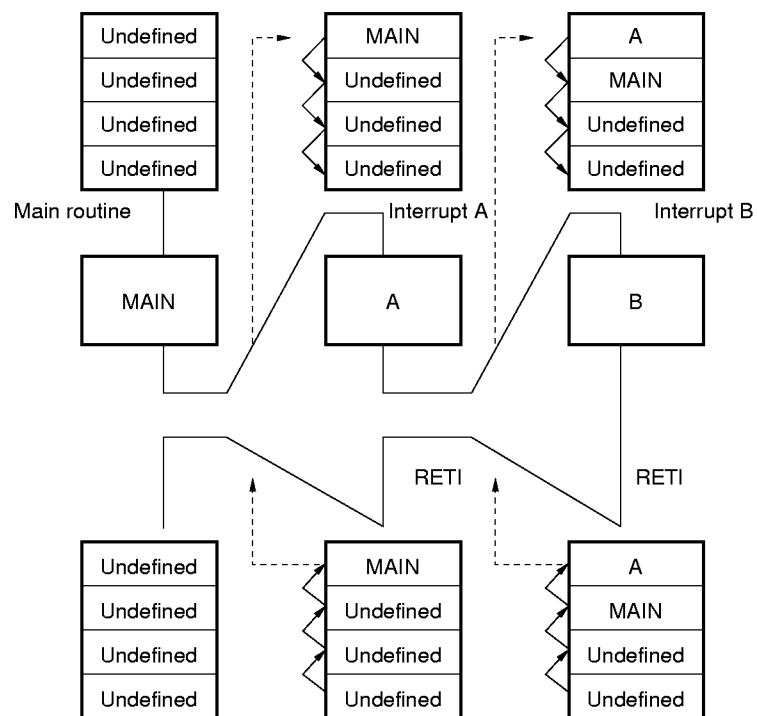
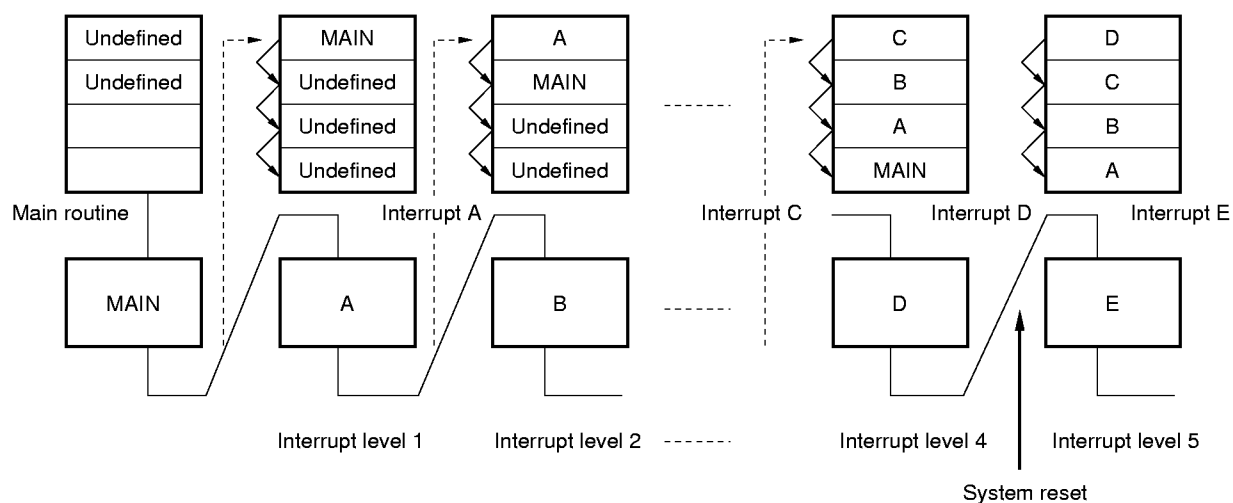


Figure 12-8. Operation of Interrupt Stack (2/2)

(b) Where interrupt nesting level is 5 or more



Caution The system is reset when an interrupt of level 5 is accepted.

However, the ISPRES flag, which resets the non-maskable interrupt if the interrupt stack overflows or underflows, must be set to "1". This flag is "1" after system reset, and can then be written only once.

12.4 Stack Pointer, Address Stack Registers, and Program Counter

The address stack registers save a return address when execution returns from an interrupt routine.

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

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

Next, the interrupt routine is executed. When the interrupt return (RETI) instruction is executed after that, the contents of an 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)**.

12.5 Interrupt Enable Flip-Flop (INTE)

The interrupt enable flip-flop enables or disables the four types of maskable 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 dedicated instructions EI (to set) and DI (to reset).

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

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

This flip-flop is also reset at power-ON reset, at a reset by the $\overline{\text{RESET}}$ pin, at a watchdog timer, overflow or underflow of the stack. The flip-flop retains the previous status on execution of the clock stop instruction.

12.6 Accepting Interrupt

12.6.1 Accepting Interrupt and priority

The following operations are performed before an interrupt is accepted.

- (1) Each peripheral hardware unit outputs an interrupt request signal to the corresponding interrupt request block if a given interrupt condition (for example, input of the falling signal to the INT pin) is satisfied.
- (2) When each interrupt request block accepts an interrupt request signal from the corresponding peripheral hardware unit, it sets the corresponding interrupt request flag (for example, IRQ0 flag if it is the INT pin that has issued the interrupt request) to "1".
- (3) The interrupt enable flag corresponding to each interrupt request flag (for example, IP0 flag if the interrupt request flag is IRQ0) is set to "1" when each interrupt request flag is set to "1", and each interrupt request block outputs "1".
- (4) The signal output by the interrupt request block is ORed with the output of the interrupt enable flip-flop, and an interrupt accept 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 "1" is output by each interrupt request processing block while the interrupt enable flip-flop is set to "1", the interrupt is accepted.

As shown in Figure 12-1, the output of the interrupt enable flip-flop is input to each interrupt request block via an AND circuit when an interrupt is accepted.

The signal input to each interrupt request block causes the interrupt request flag corresponding to each interrupt request flag to be reset to "0" and the vector address corresponding to each interrupt to be output.

If the interrupt request block outputs "1" at this time, the interrupt accept signal is not transferred to the next stage. If two or more interrupt requests are issued at the same time, therefore, the interrupts are accepted according to the priority shown in Table 12-2.

Unless the interrupt request enable flag is set to "1", the corresponding interrupt is not accepted.

Therefore, by resetting the interrupt enable flag to "0", the interrupt with a high hardware priority can be disabled.

Table 12-2. Interrupt Priority

Interrupt Source	Priority
INT pin	1
Timer 0	2
Basic timer 1	3
Serial interface 1	4

12.6.2 Timing chart when interrupt is accepted

The timing charts in Figure 12-9 illustrate the operations performed when an interrupt or interrupts are accepted.

Figure 12-9 (1) is the timing chart when one interrupt is accepted.

(a) in (1) is the timing chart where the interrupt request flag is set to "1" after all the others, and (b) is the timing chart where the interrupt enable flag is set to "1" after all the others.

In either case, the interrupt is accepted when the interrupt request flag, interrupt enable-flip flop, and interrupt enable flag all have been set to "1".

If the flag or flip-flop that has been set last is set in the first instruction cycle of the "MOVT DBF, @AR" instruction or by an instruction that satisfies a given skip condition, the interrupt is accepted in the second instruction cycle of the "MOVT DBF, @AR" instruction or after the instruction that is skipped (this instruction is treated as NOP) has been executed.

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

Therefore, the interrupt is accepted after the instruction next to the EI instruction has been executed even when the interrupt request flag is set in the execution cycle of the EI instruction.

(2) in Figure 12-9 is the timing chart where two or more interrupts are used.

When two or more interrupts are used, the interrupts are accepted according to the hardware priority if all the interrupt enable flags are set. However, the hardware priority can be changed by setting the interrupt enable flags by the program.

"Instruction cycle" shown in Figure 12-9 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 accepted. It takes 53.3 μ s, which is equivalent to one instruction execution time, to be completed.

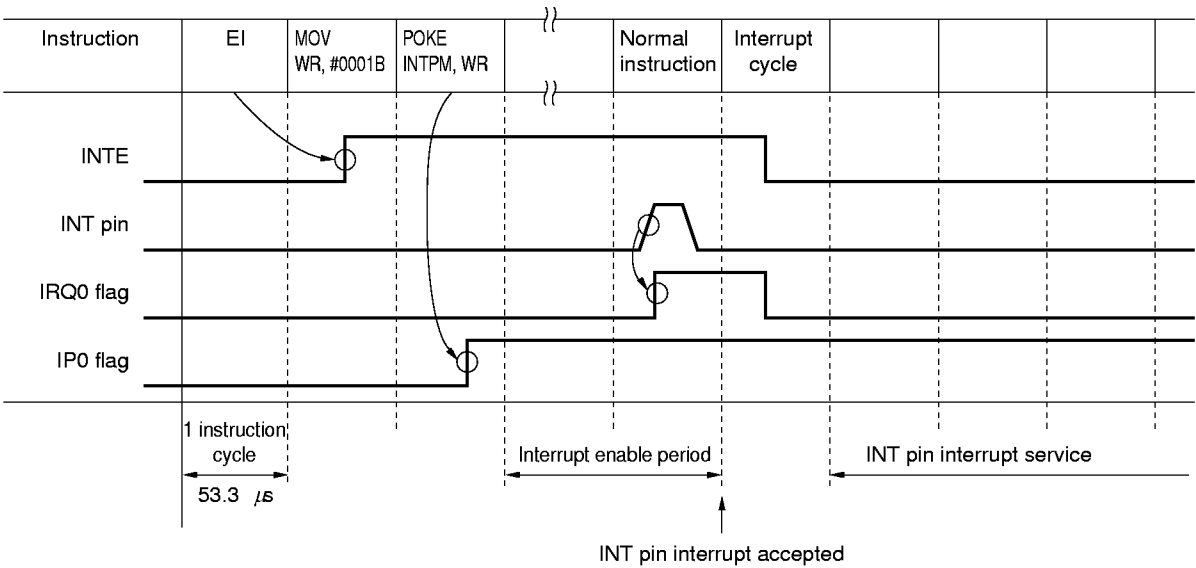
For details, refer to **12.7 Operations after Interrupt Has Been Accepted**.

Figure 12-9. Timing Charts When Interrupt Is Accepted (1/3)

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

(a) If there is no interrupt mask time by the interrupt flag (IP_{xxx})

<1> If a normal instruction which is not “MOVT” or an instruction that satisfies a skip condition is executed when interrupt is accepted



<2> If “MOVT” or an instruction that satisfies a skip condition is executed when interrupt is accepted

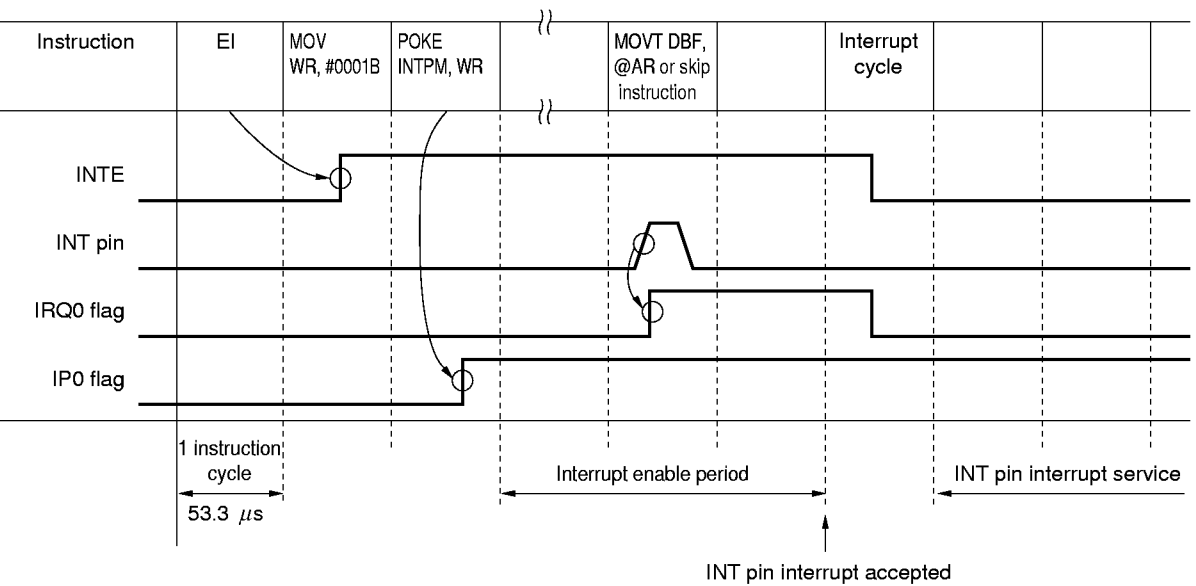
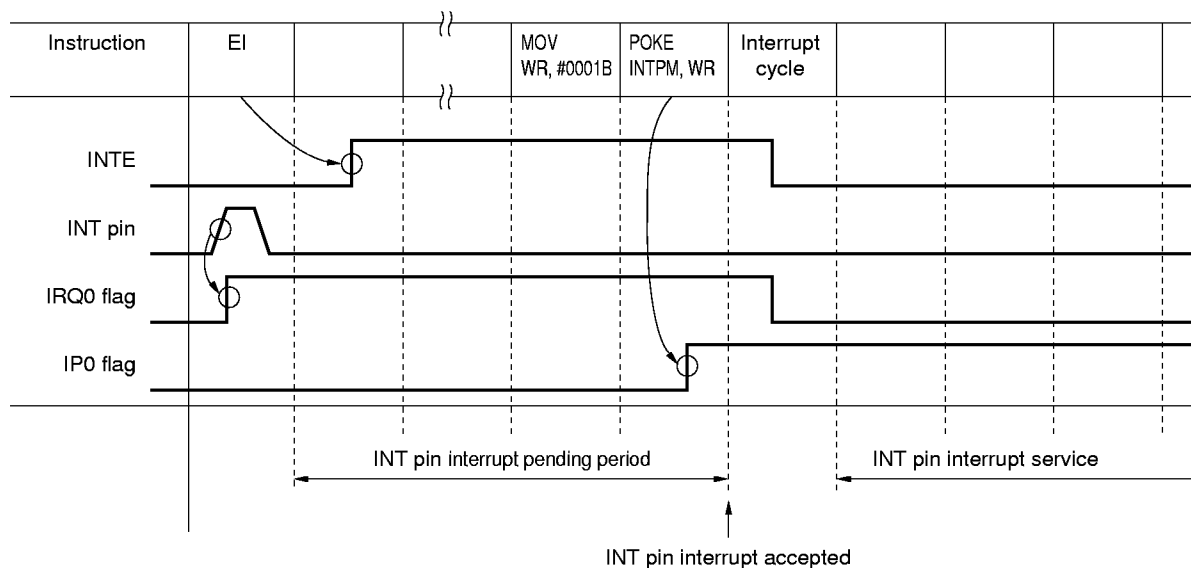


Figure 12-9. Timing Charts When Interrupt Is Accepted (2/3)

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



(2) If two or more interrupts (e.g., INT pin and basic timer 1) are used

(a) Hardware priority

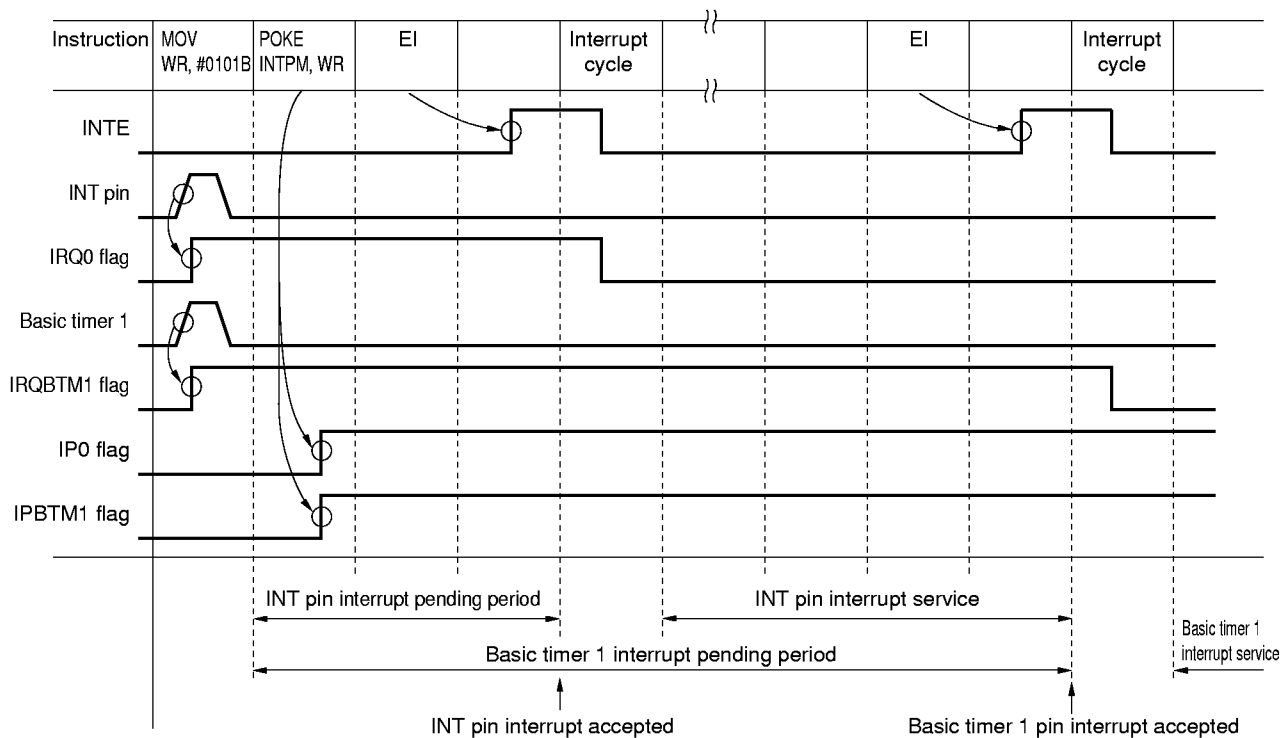
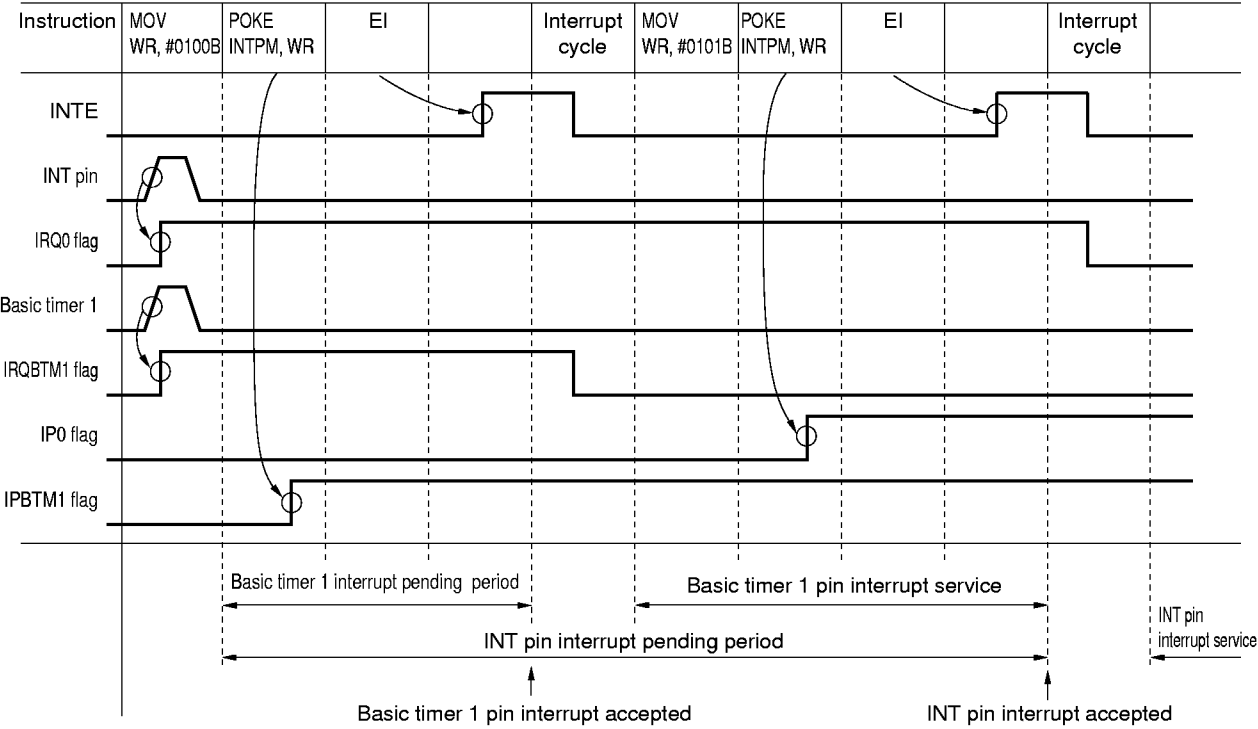


Figure 12-9. Timing Charts When Interrupt Is Accepted (3/3)

(b) Software priority



12.7 Operations after Interrupt Has Been Accepted

When an interrupt is accepted, the following operations are sequentially performed automatically.

- (1) The interrupt enable flip-flop and the interrupt request flag corresponding to the accepted interrupt request are reset to "0". As a result, the other interrupts are disabled.
- (2) The contents of the stack pointer are decremented by one.
- (3) The contents of the program counter are saved to an address stack register specified by the stack pointer. At this time, the contents of the program counter are the program memory address after the address at which the interrupt has been accepted.
For example, if a branch instruction is executed when the interrupt has been accepted, the contents of the program counter are the branch destination address. If a subroutine call instruction is executed, the contents of the program counter are the call destination address. If the skip condition of a skip instruction is satisfied, the next instruction is executed as NOP and then the interrupt is accepted. Consequently, the contents of the program counter are the address after that of the instruction that is skipped.
- (4) The contents of the system registers (except the address register) are saved to the interrupt stack.
- (5) The contents of the vector address generator corresponding to the interrupt that has been accepted are transferred to the program counter. In other words, execution branches to the interrupt routine.

The operations (1) through (5) above require the time of one special instruction cycle (53.3 μ s) in which normal instruction execution is not performed.

This instruction cycle is called an "interrupt cycle".

In other words, the time of one instruction cycle (53.3 μ s) is required after an interrupt has been accepted until execution branches to the corresponding vector address.

12.8 Returning from Interrupt Routine

The interrupt return (RETI) instruction is used to return from an interrupt routine to the processing during which an interrupt was accepted.

When the RETI instruction is executed, the following operations are sequentially performed automatically.

- (1) The contents of an address stack register specified by the stack pointer are restored to the program counter.
- (2) The contents of the interrupt stack are restored to the system registers.
- (3) The contents of the stack pointer are incremented by one.

The operations (1) through (3) above require one instruction cycle (53.3 μ s) in which the RETI instruction is executed.

The only difference between the RETI instruction and the RET and RETSK instructions, which are subroutine return instructions, is the restoration of the bank register and index register in step (2) above.

12.9 External Interrupts (INT pin)

12.9.1 Outline of external Interrupts

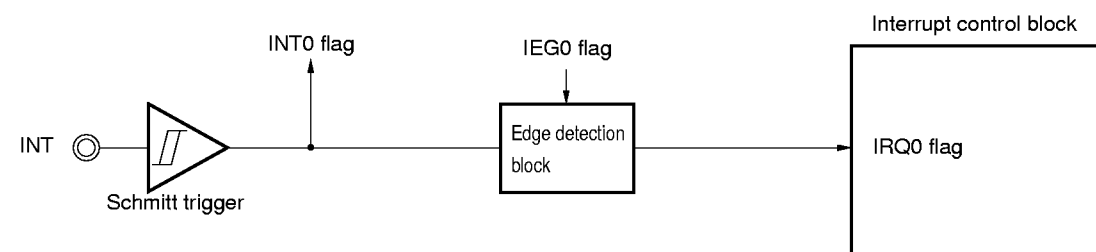
Figure 12-10 outlines the external interrupts.

As shown in the figure, external interrupt requests are issued at the rising or falling edges of signals input to the INT pin.

Whether an interrupt request is issued at the rising or falling edge of an INT pin is independently specified by the program.

The INT pin is a Schmitt trigger input pin to prevent malfunctioning due to noise. This pin does not accept a pulse input of less than 100 ns.

Figure 12-10. Outline of External Interrupts

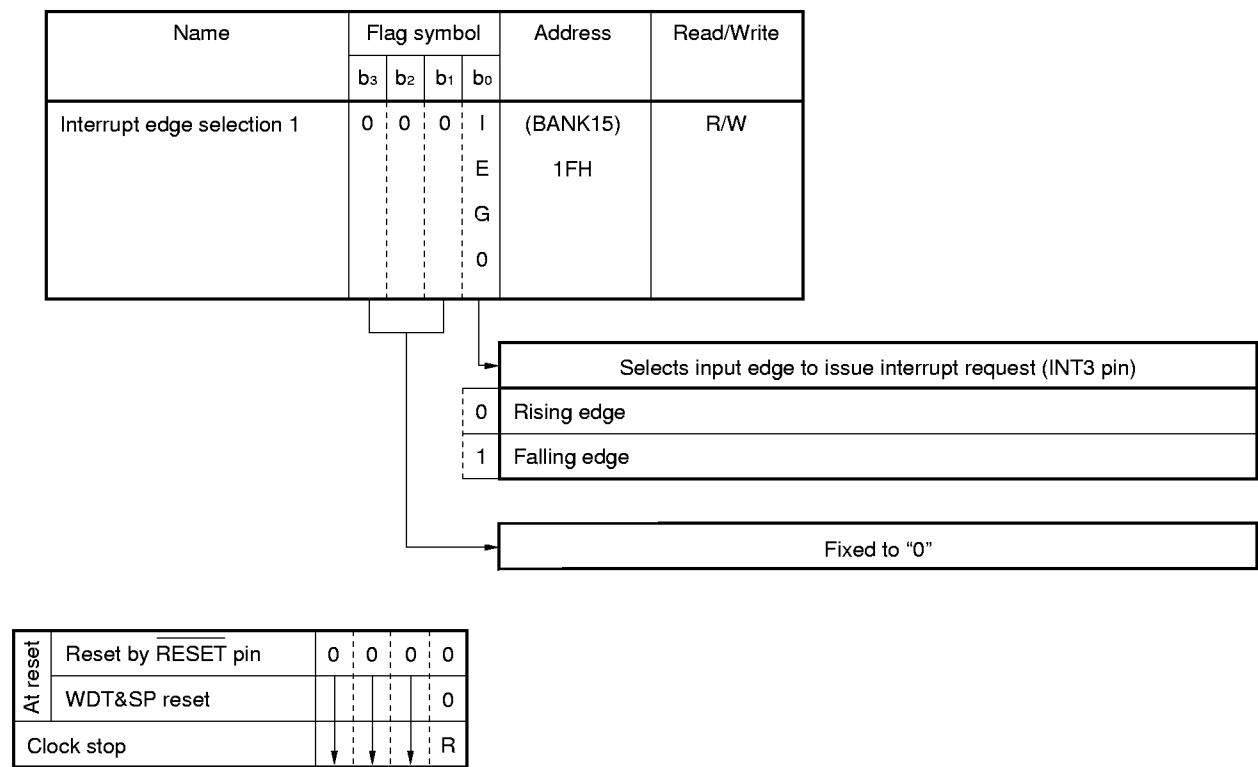


12.9.2 Edge detection block

The edge detection block specifies the valid edge (rising or falling edge) of an input signal that issues the interrupt request of INT pin, by using an interrupt edge selection register.

Figure 12-11 shows the configuration and function of the interrupt edge selection register.

Figure 12-11. Configuration of Interrupt Edge Selection Register



R: Retained

Caution The external input delays about 100 ns.

Table 12-3. Issuance of Interrupt Request by Changing IEG Flag

Changes in IEG0 Flag	Status of INT Pin	Issuance of Interrupt Request	Status of Interrupt Request Flag
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

12.9.3 Interrupt control block

The signal levels that are input to the INT pin can be detected by using the INT0 flag.

Because INT0 flag is reset independently of interrupts, when the interrupt function is not used the INT pin can be used as a 1-bit input port.

If the interrupts are not enabled, these ports can be used as general-purpose port pins whose rising or falling edge can be detected by reading the corresponding interrupt request flags.

At this time, however, the interrupt request flags are not automatically reset and must be reset by the program.

For further information, also refer to **12.2.1 Configuration and function of interrupt request flag (IRQ_{xxx})**.

12.10 Internal Interrupts

The following three internal interrupts are available.

- Timer 0
- Basic timer 1
- Serial interface 1

12.10.1 Timer 0 and basic timer 1 interrupts

Interrupt requests are issued at fixed intervals.

For details, refer to **13. TIMER**.

12.10.2 Serial interface 1 interrupts

Interrupt requests can be issued at the end of a serial output or serial input operation.

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

13. TIMERS

Timers are used to manage the program execution time.

13.1 Outline of Timers

Figure 13-1 outlines the timers.

The following three timers are available.

- Basic timer 0, 1
- Timer 0

Basic timer 0 and 1 detect the status of a flip-flop that is set at fixed time intervals in software.

Timer 0 is a modulo timer and can use interrupts.

The clock of each timer is created by dividing the system clock (75 kHz).

Figure 13-1. Outline of Timers

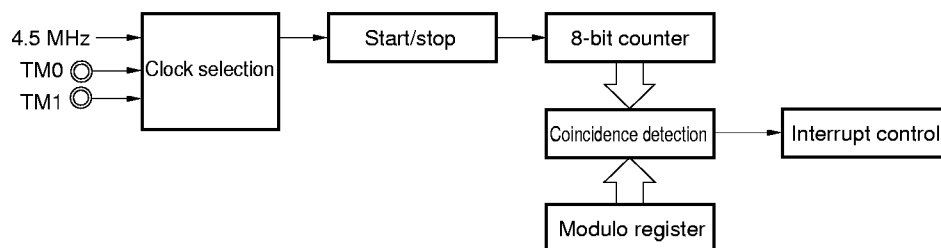
(1) Basic timer 0



(2) Basic timer 0



(3) Timer 0



13.2 Basic Timer 0

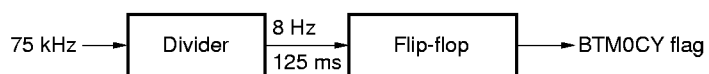
13.2.1 Outline of basic timer 0

Figure 13-2 outlines basic timer 0.

Basic timer 0 is used as a timer by detecting in software the BTM0CY flag that is set at fixed intervals (125 ms).

- ★ If the BTM0CY flag is read first after reset by $\overline{\text{RESET}}$ pin, "1" is always read. After that, the flag is set to "0" at 125 ms intervals.

Figure 13-2. Outline of Basic Timer 0



Remark BTM0CY (bit 0 of basic timer 0 carry register: refer to **Figure 13-3**) detects the status of the flip-flop.

The flip-flop is set at fixed intervals (125 ms) and its status is detected by the BTM0CY flag of the basic timer 0 carry register.

The BTM0CY flag is always set to “1” at reset by $\overline{\text{RESET}}$ pin instruction.

Figure 13-3 shows the configuration of the basic timer 0 carry register.

Figure 13-3. Configuration of Basic Timer 0 Carry Register

The diagram shows a signal line that is initially high. It then transitions to a low state, which is labeled "Fixed to '0'". After a period of being low, it transitions back to a high state. This high state is labeled "Detects status of flip-flop". Below this, a table shows the status of the flip-flop for two cases:

Value	Description
0	Flip-flop is not set
1	Flip-flop is set

At reset	Reset by $\overline{\text{RESET}}$ pin	0	0	0	1
	WDT&SP reset				R
Clock stop		↓	↓	↓	R

R: Retained

13.2.3 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

```

LOOP:
SKT1  BTM0CY           ; Branches to NEXT if BTM0CY flag is "0"
BR    NEXT
ADD   M1, #1           ; Adds 1 to M1
SKE   M1, #08H         ; Executes processing A if M1 is "8" (1 second has elapsed)
BR    NEXT
MOV   M1, #0
    
```

Processing A

NEXT:

Processing B

; Executes processing B and branches to LOOP

```
BR    LOOP
```

13.2.4 Errors of basic timer 0

Errors of basic timer 0 include an error due to the detection time of the BTM0CY flag.

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.

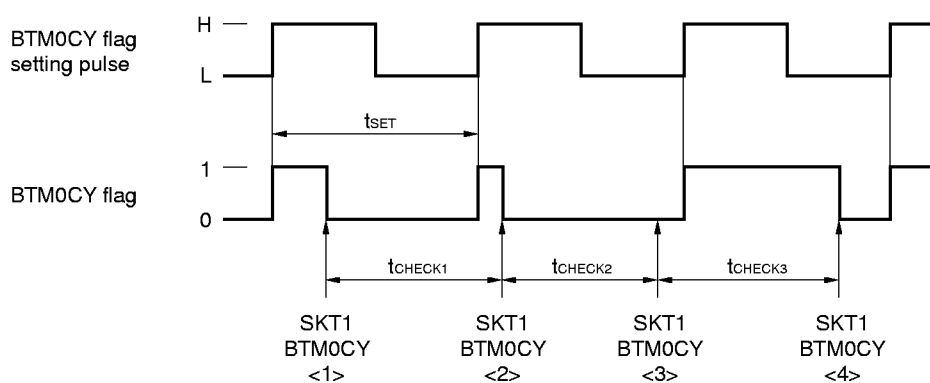
Where the time interval at which the BTM0CY flag is detected is t_{CHECK} and the time interval at which the flag is set is t_{SET} (125 ms), t_{CHECK} and t_{SET} must relate as follows.

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

At this time, the error of the timer when the BTM0CY flag is detected is as follows, as shown in Figure 13-4.

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

Figure 13-4. Error of Basic Timer 0 due to Detection Time of BTM0CY Flag



As shown in Figure 13-4, 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} .

13.3 Basic Timer 1

13.3.1 General

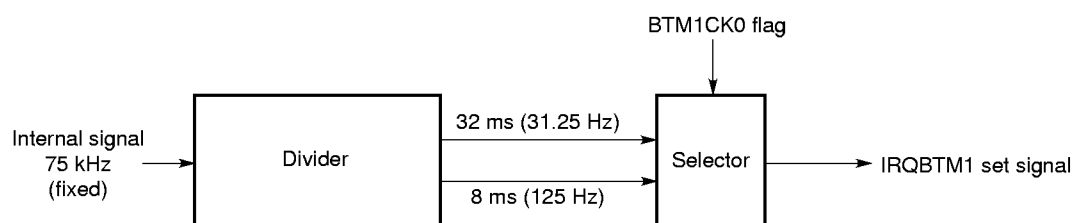
Figure 13-5 outlines the basic timer 1.

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

The time interval of the IRQBTM1 flag is set by the BTM1CK0 flag of the basic timer 1 clock select register. Figure 13-6 shows the configuration of the basic timer 1 clock select register.

The interrupt generated by the basic timer 1 is accepted when the IRQBTM1 flag is set, if the EI instruction has been issued and the IPBTM1 flag has been set (refer to **12. INTERRUPT**).

Figure 13-5. Outline of Basic Timer 1



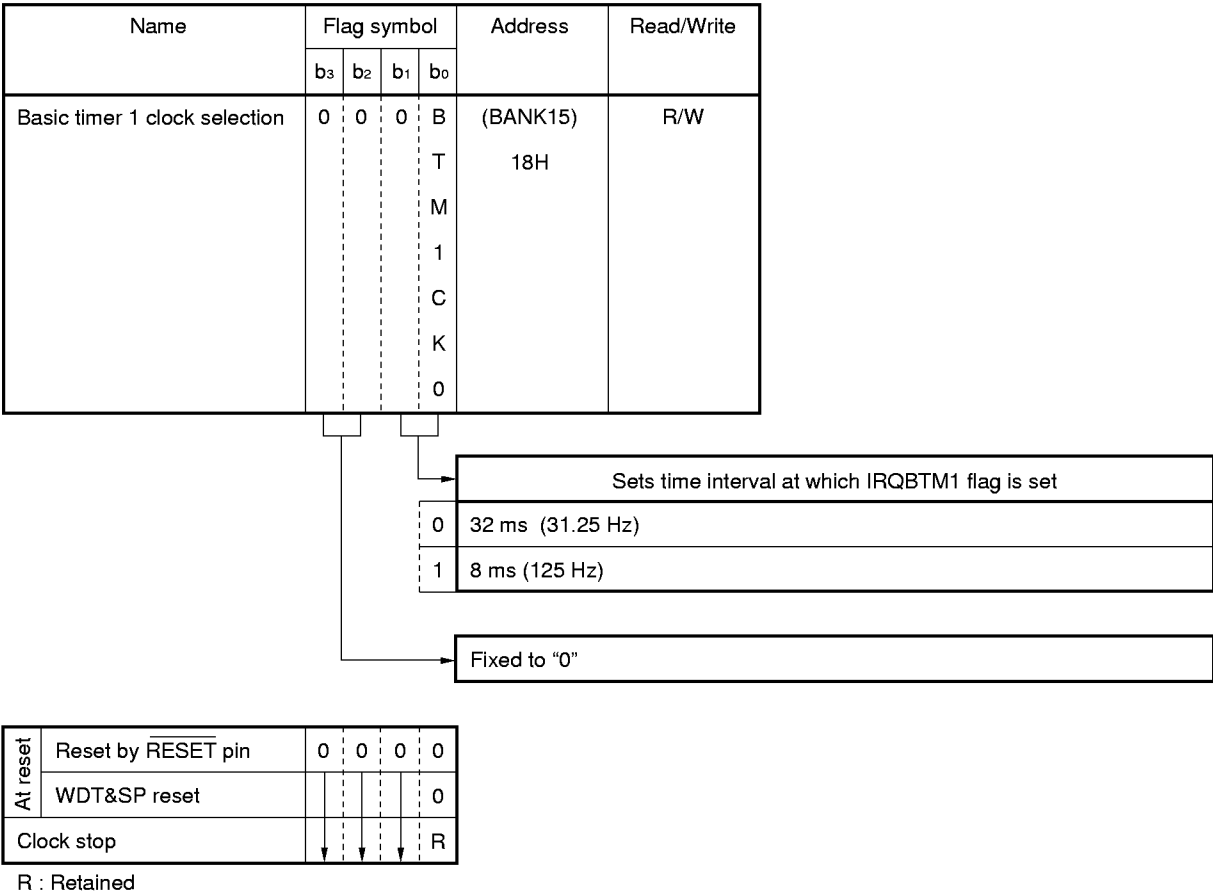
Remark BTM1CK0 (bit 1 of interrupt edge select register. Refer to **Figure 13-6**) set the time interval at which the IRQBTM1 flag is set.

13.3.2 Clock selection block

The clock selection block divides the system clock (75 kHz) and sets the time interval at which the IRQBTM1 flag is to be set, by using the BTM1CK0 flags.

Figure 13-6 shows the configuration of the basic timer 0 clock selection register.

Figure 13-6. Configuration of Basic Timer 1 Clock Selection Register



13.3.3 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

ORG      BR      START      ; Branches to START
          BTIMER1          ; Program address (0002H)
          ADD      M1, #0001B ; Adds 1 to M1
          SKT1     CY        ; Tests CY flag
          BR      EI_RETI    ; Returns if no carry
          MOV      M1, #0110B
          Processing A
EI_RETI: EI
          RETI
START:    MOV      M1, #0110B ; Initializes contents of M1 to 6
          BANK1
          SET1     BTM1CK     ; Embedded macro
                                ; Sets basic timer 1 interrupt pulse to 8 ms
          SET1     IPBTM1     ; Enables basic timer 1 interrupt
          EI        ; Enables all interrupts
LOOP:    BANK0
          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 accepted, and that the IRQBTM1 flag is set to 1 even in the DI status.

This means that the interrupt is accepted even if execution exits from an interrupt service routine by execution of the "RETI" instruction, if processing A takes longer than 8 ms.

Consequently, processing B is not executed.

13.3.4 Error of basic timer 1

As described in 13.3.3, the interrupt generated by basic timer 1 is accepted each time the basic timer 1 interrupt pulse falls, if the EI instruction has been executed, and if the interrupt has been enabled.

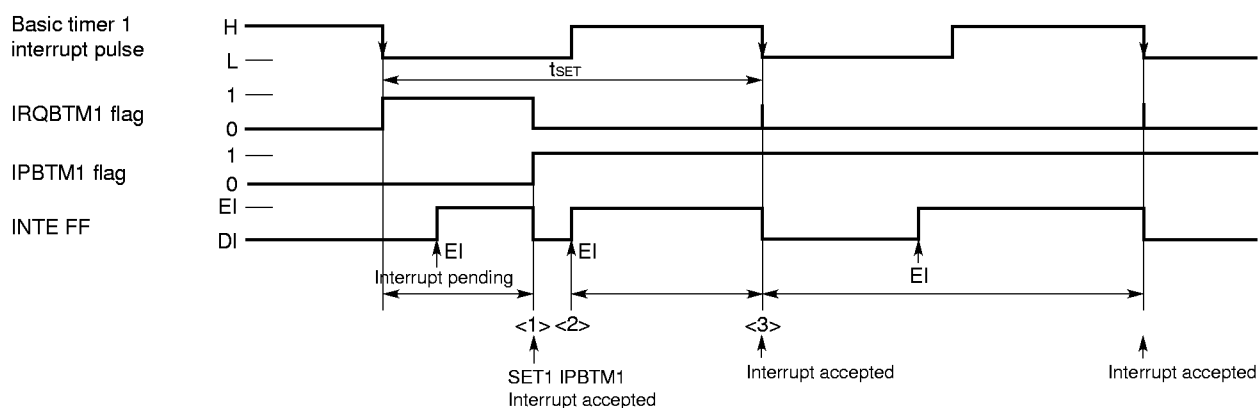
Therefore, an error of basic timer 1 occurs only when any of the following operations are performed:

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

Figure 13-7 shows an error in each of the above operations.

Figure 13-7. 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 accepted as soon as the interrupt is enabled.

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

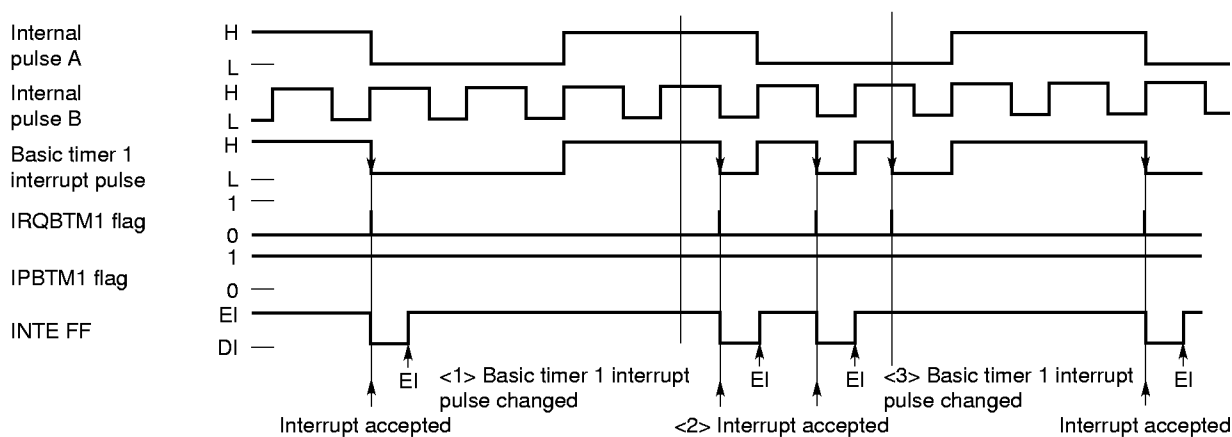
If an interrupt is enabled by the "EI" instruction at the next point <3>, the interrupt occurs at the falling edge of the basic timer 1 interrupt pulse.

At this time, the error is:

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

Figure 13-7. Error of Basic Timer 1 (2/2)

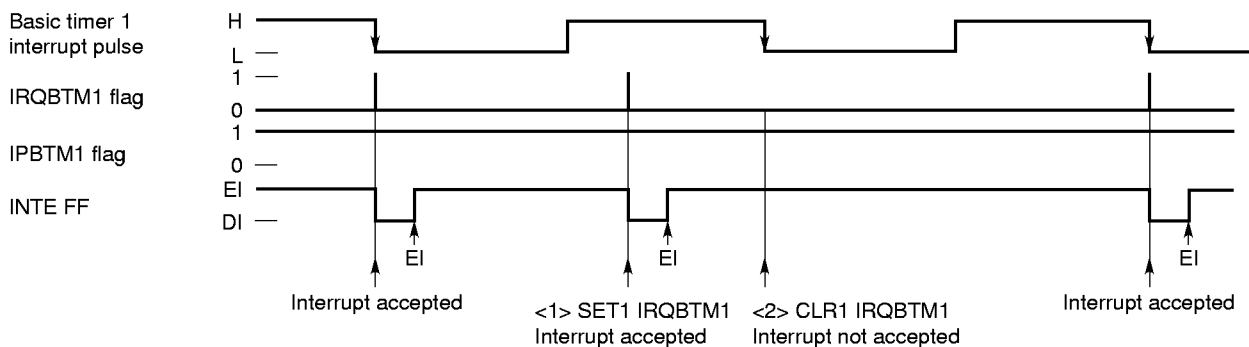
(b) When basic timer 1 interrupt pulse is changed



Even if the basic timer 1 interrupt pulse is changed to B at point <1> in the above figure, the interrupt is accepted 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 accepted because the basic timer 1 interrupt pulse falls.

(c) When IRQBTM1 flag is manipulated



The interrupt is immediately accepted 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 accepted.

13.3.5 Notes on using basic timer 1

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

Let's take the following example:

Example

```

M1      MEM      0.10H      ; 80-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, #0001B  ; Adds 1 to M1
        SKT1     CY          ; Watch processing if carry occurs
        BR       EI_RET1    ; Restores if no carry occurs
        MOV      M1, #0110B
; <1>


Processing B


EI_RET1:
        EI
        RETI

START:
        MOV      M1, #0110B  ; Initializes contents of M1 to 6
        BANK1
        SET1     BTM1CK
                                ; Embedded macro
                                ; Sets time of interrupt by basic timer 1 to 8 ms
        SET1     IPBTM1      ; Embedded macro
                                ; Enables interrupt by basic timer 1
        EI           ; Enables all interrupts

LOOP:


Processing A


        BR       LOOP

```

In this example, processing B is executed every 80 ms while processing A is executed.

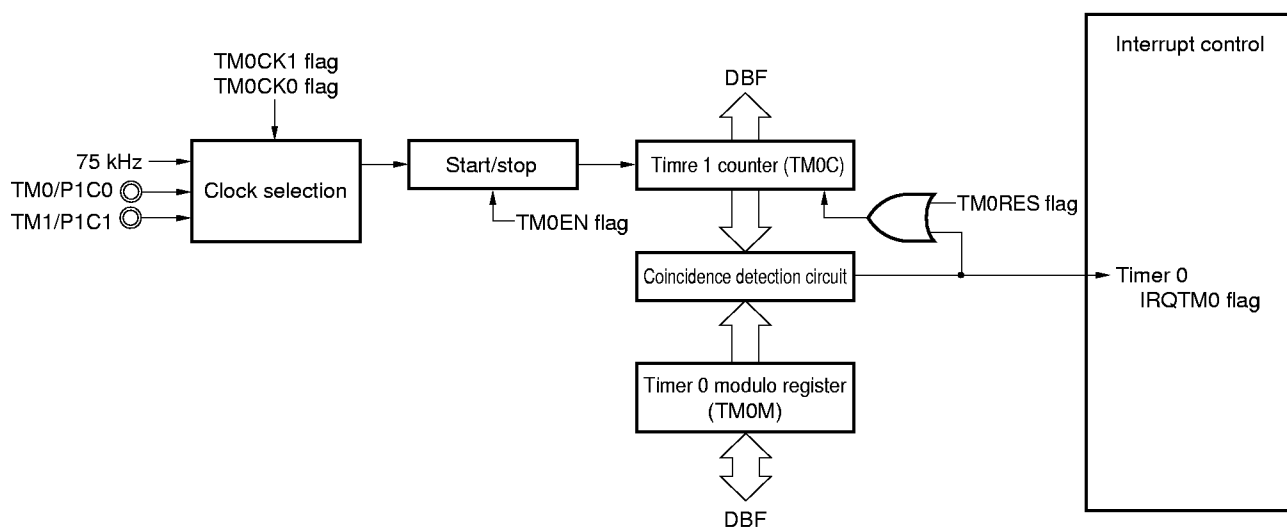
13.4 Timer 0

13.4.1 Outline of timer 0

Figure 13-8 outlines timer 0.

Timer 0 counts the basic clock (75, 25, or external clock (TM0, TM1)) with an 8-bit counter, and compares the count value with a value set in advance.

Figure 13-8. Outline of Timer 0



- Remarks**
1. TM0CK1 and TM0CK0 (bits 1 and 0 of timer 0 counter clock selection register: refer to **Figure 13-9**) set the basic clock frequency.
 2. TM0EN (bit 3 of timer 0 counter clock selection register: refer to **Figure 13-9**) starts or stops timer 0.
 3. TM0RES (bit 2 of timer 0 counter clock selection register: refer to **Figure 13-9**) resets timer 0 counter.

13.4.2 Clock selection and start/stop control blocks

The clock selection block selects a basic clock to operate timer 0 counter.

Four types of basic clocks can be selected by using the TMOCK1 and TMOCK0 flags.

The start/stop block starts or stops the basic clock input to timer 0 by using the TM0EN flag.

Figure 13-9 shows the configuration and function of each flag.

13.4.3 Count block

The count block counts the basic clock with timer 0 counter, reads the count value, and issues an interrupt request when its count value coincides with the value of the timer 0 modulo register.

The timer 0 counter can be reset by the TM0RES flag.

The timer 0 counter is automatically reset when its value coincides with the value of the timer 0 modulo register.

The value of the timer 0 counter can be read via data buffer.

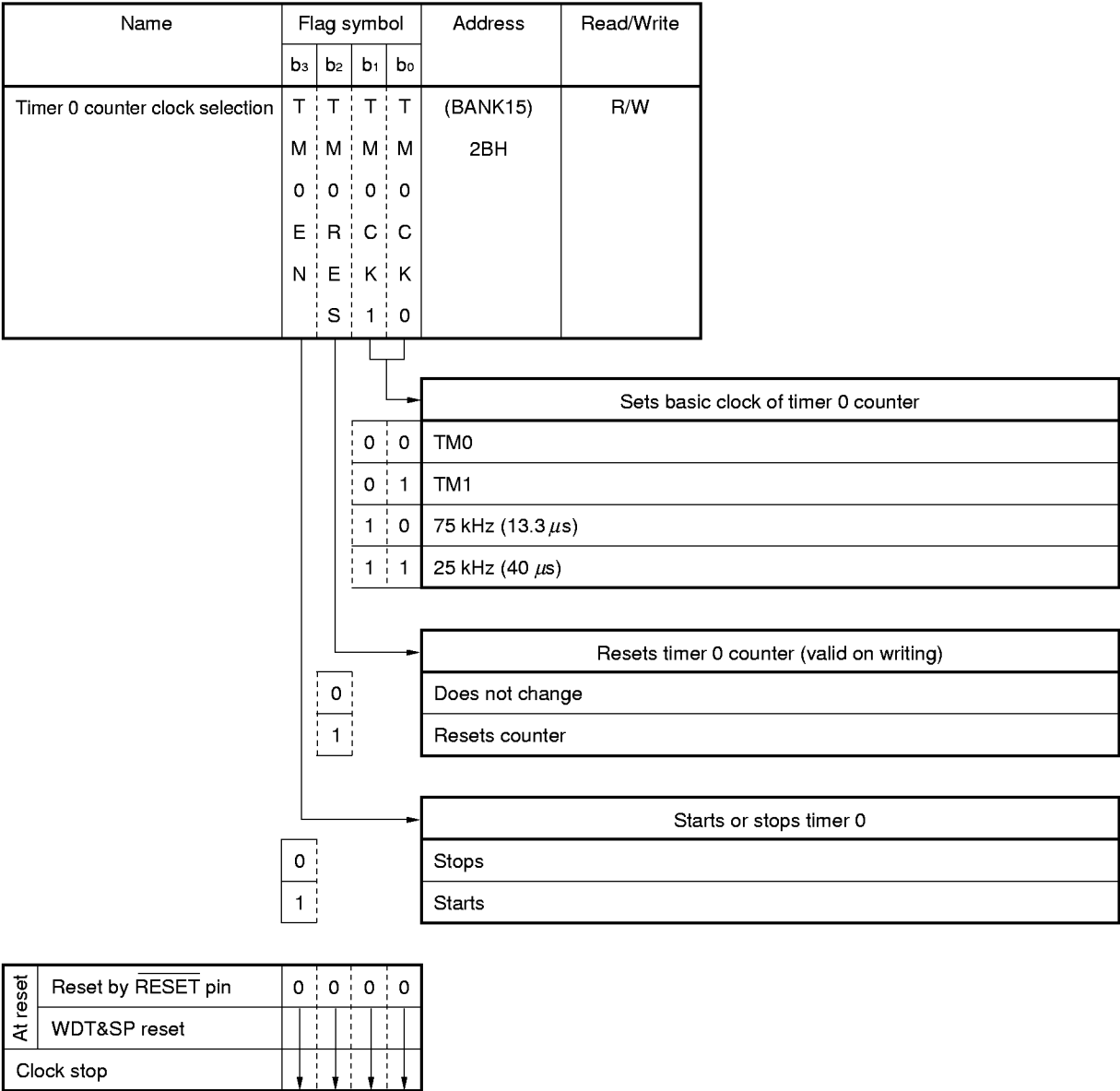
Data can be written to the value of the timer 0 modulo register via data buffer.

Figure 13-9 shows the configuration of timer 0 counter clock selection register.

Figure 13-10 shows the configuration of the timer 0 counter.

Figure 13-11 shows the configuration of the timer 0 modulo register.

Figure 13-9. Configuration of Timer 0 Counter Clock Selection Register



1

0

1

1

0

1

0

1

Caution When the TM0RES flag is read, 0 is always read.

Figure 13-10. Configuration of Timer 0 Counter

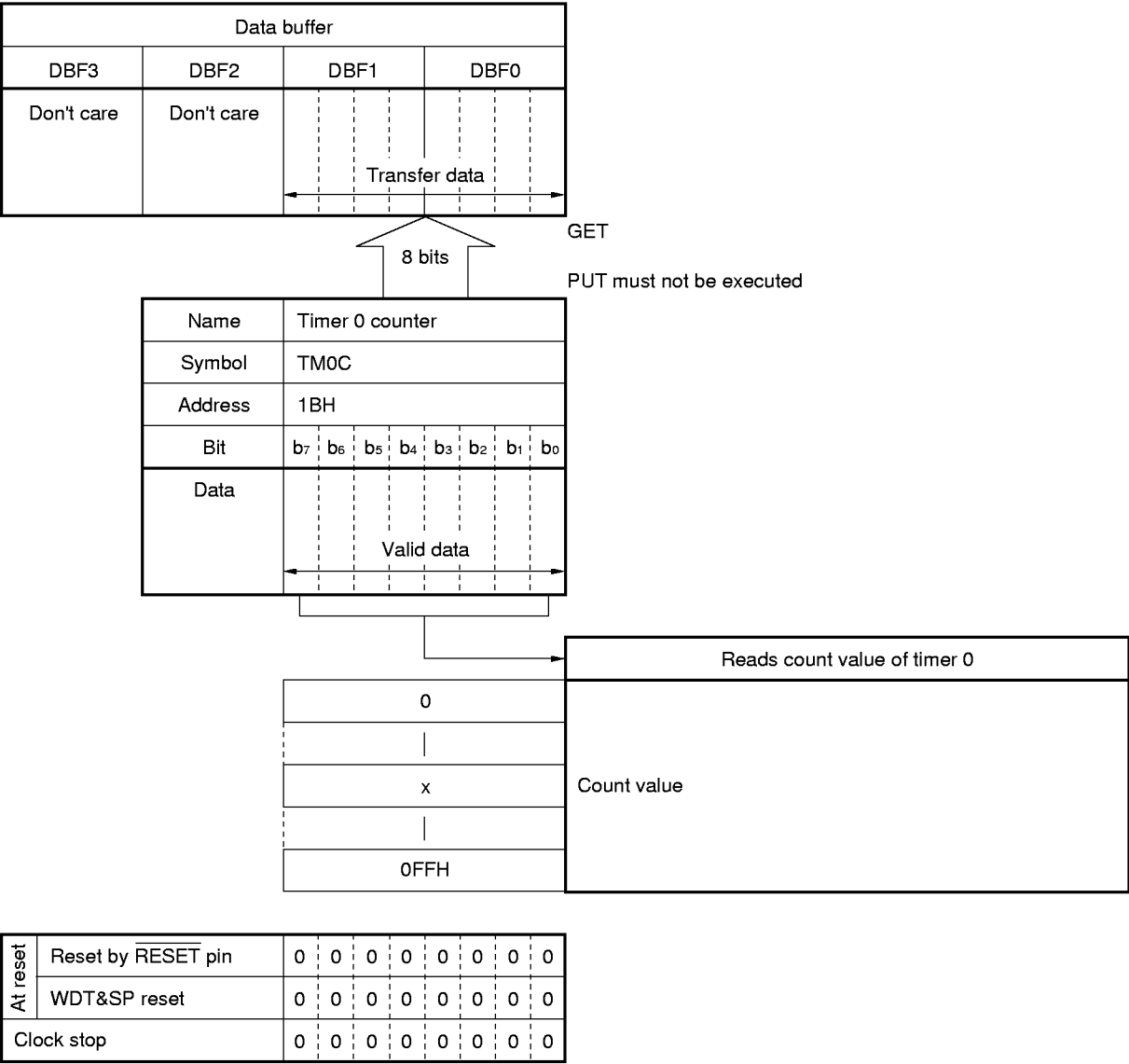
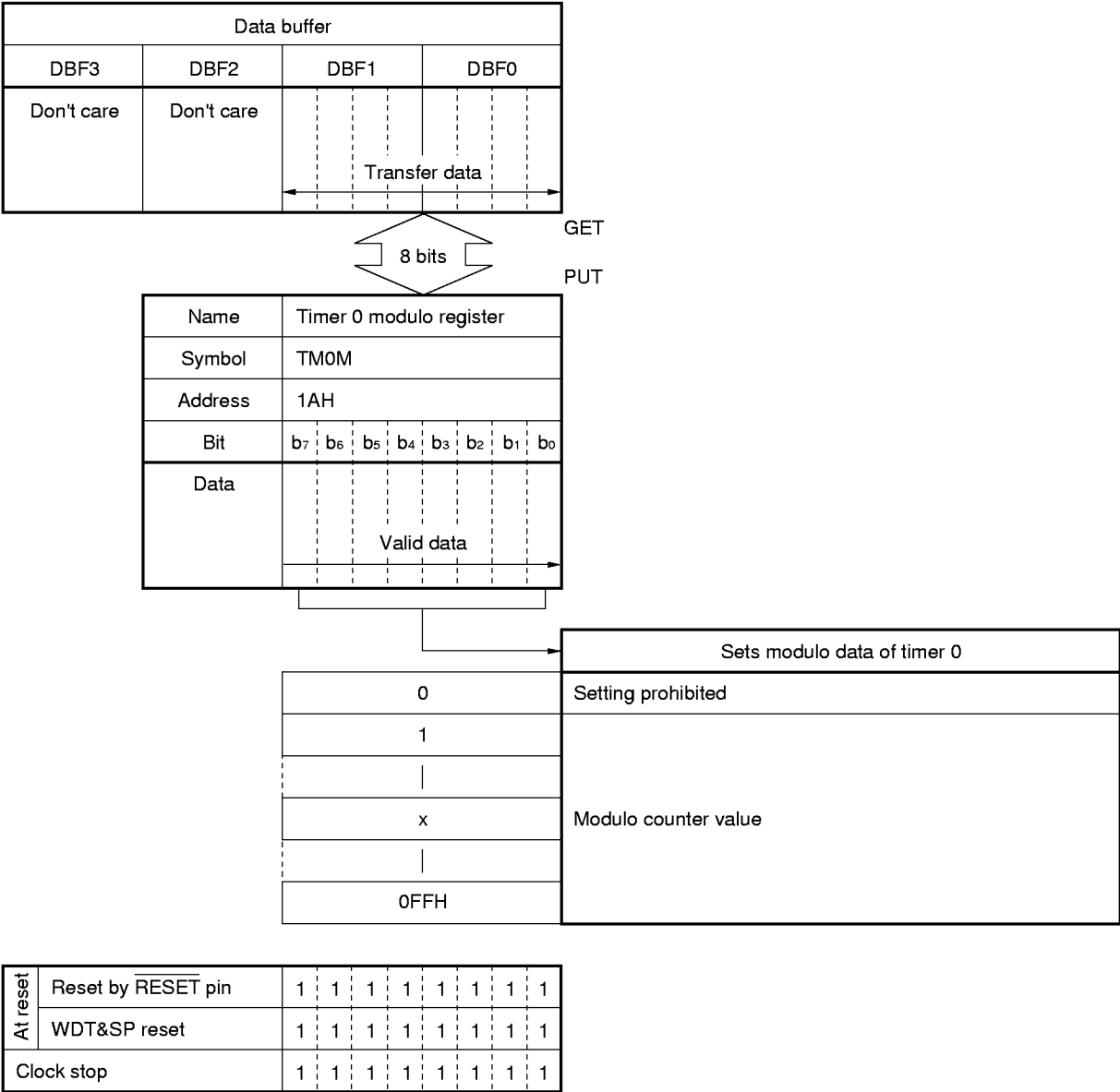


Figure 13-11. Configuration of Timer 0 Modulo Register



13.4.4 Example of using timer 0

(1) Modulo timer

The modulo timer is used for time management by generating timer 0 interrupt at fixed intervals.

An example of a program is shown below.

This program executes processing B every 400 μs.

```

TM0DATA   DAT      0009H           ; Count data = 10

START:
    BR      INITIAL                ; Reset address
    ; Interrupt vector address
    NOP                     ; SIO1
    NOP                     ; BTM1
    BR      INT_TM0            ; TM0
    NOP                     ; INT

INITIAL:
    INITFLG   NOT TM0EN, TM0RES, TM0CK1, TM0CK0
    ;          (Stop) , (Reset) , (Basic clock = 40 μs)
    MOV       DBF0, #TM0DATA
    MOV       DBF1, #TM0DATA SHR4 AND 0FH
    PUT       TM0, DBF
    SET1      TM0EN            ; START
    SET1      IPTM0            ; Enables timer 0 interrupt
    EI

LOOP:
    

Processing A


    BR      LOOP

INT_TM0:
    PUT       DBFSTK, DBF       ; Saves data buffer
    

Processing B


    GET       DBF, DBFSTK
    EI
    RETI                        ; Return

END
    
```

13.4.5 Error of timer 0

Timer 0 has an error of up to 1 basic clock in the following cases.

(1) On starting/stopping counter

The counter is started or stopped by setting the TM0EN flag.

Therefore, an error of 0 to +1 clocks occurs when the TM0EN flag is set, and an error of -1 to 0 clocks occurs when the flag is reset.

In all, an error of ± 1 count occurs.

(2) On resetting counter operation

An error of 0 to +1 clocks occurs when the counter is reset.

(3) On selecting basic clock during counter operation

An error of 0 to +1 clocks of the newly selected clock occurs.

14. A/D CONVERTER

14.1 Outline of A/D Converter

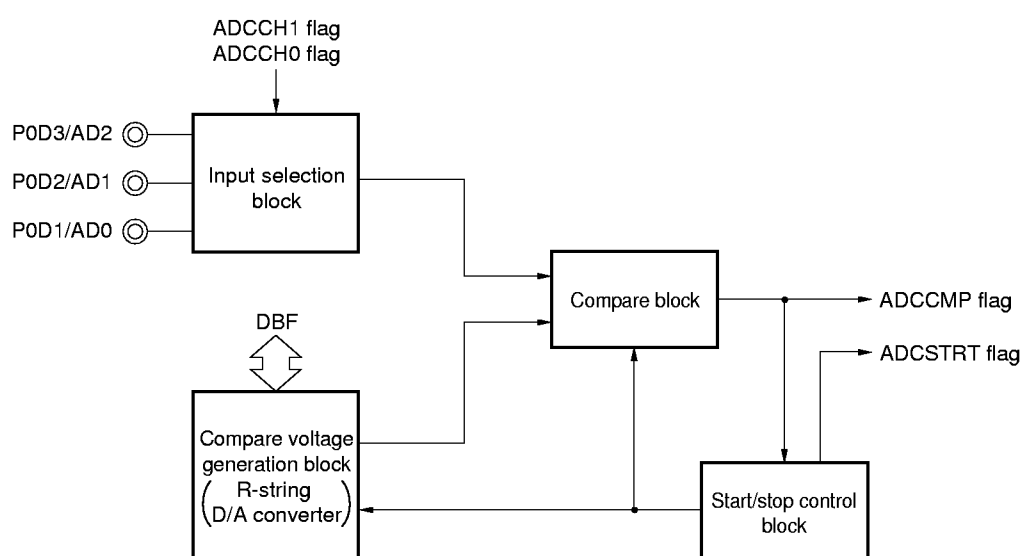
Figure 14-1 outlines the A/D converter.

The A/D converter compares the analog voltages input to the AD2 through AD0 pins with the internal compare voltage and converts them into 4-bit digital signals by judging the result of the comparison by software.

The result of the comparison is detected by the ADCCMP flag.

This converter is of successive approximation type.

Figure 14-1. Outline of A/D Converter



- Remarks**
1. ADCCH1 and ADCCH0 (bits 1 and 0 of A/D converter channel selection register: refer to **Figure 14-3**) select pins used for the A/D converter.
 2. ADCCMP (bit 0 of A/D converter mode selection register: refer to **Figure 14-5**) detects the result of comparison.
 3. ADCSTRT (bit 1 of A/D converter mode selection register: refer to **Figure 14-5**) detects the operating status.

14.2 Input Selection Block

Figure 14-2 shows the configuration of the input selection block.

The input selection block selects a pin to be used by using the ADCCH1 and ADCCH0 flags. Only one pin can be used for the A/D converter. When one of the P0D1/AD0 through P0D3/AD2 pins is selected, the other two pins are forcibly set in the input port mode.

The P0D1/AD0 through P0D3/AD2 pins can be connected to a pull-down resistor if so specified by the P0DPL1 through P0DPL3 flags. To use the P0D1/AD0 through P0D3/AD2 pins for the A/D converter, therefore, disconnect their pull-down resistors to correctly detect an external input analog voltage. (For details, refer to **11.3.3 Port 0D pull-down resistor selection register.**)

Figure 14-3 shows the configuration of the A/D converter channel selection register.

Figure 14-2. Configuration of Input Selection Block

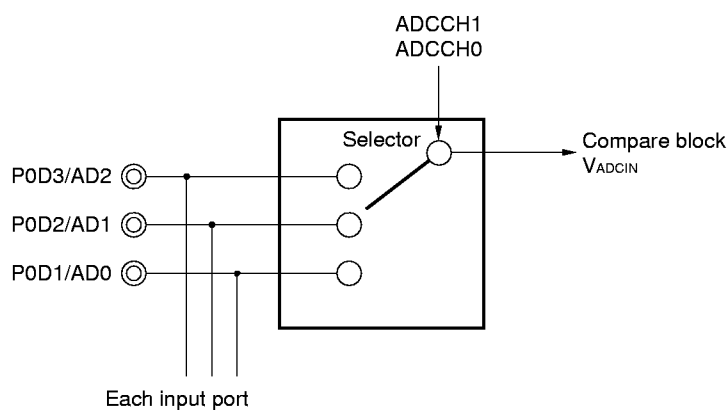
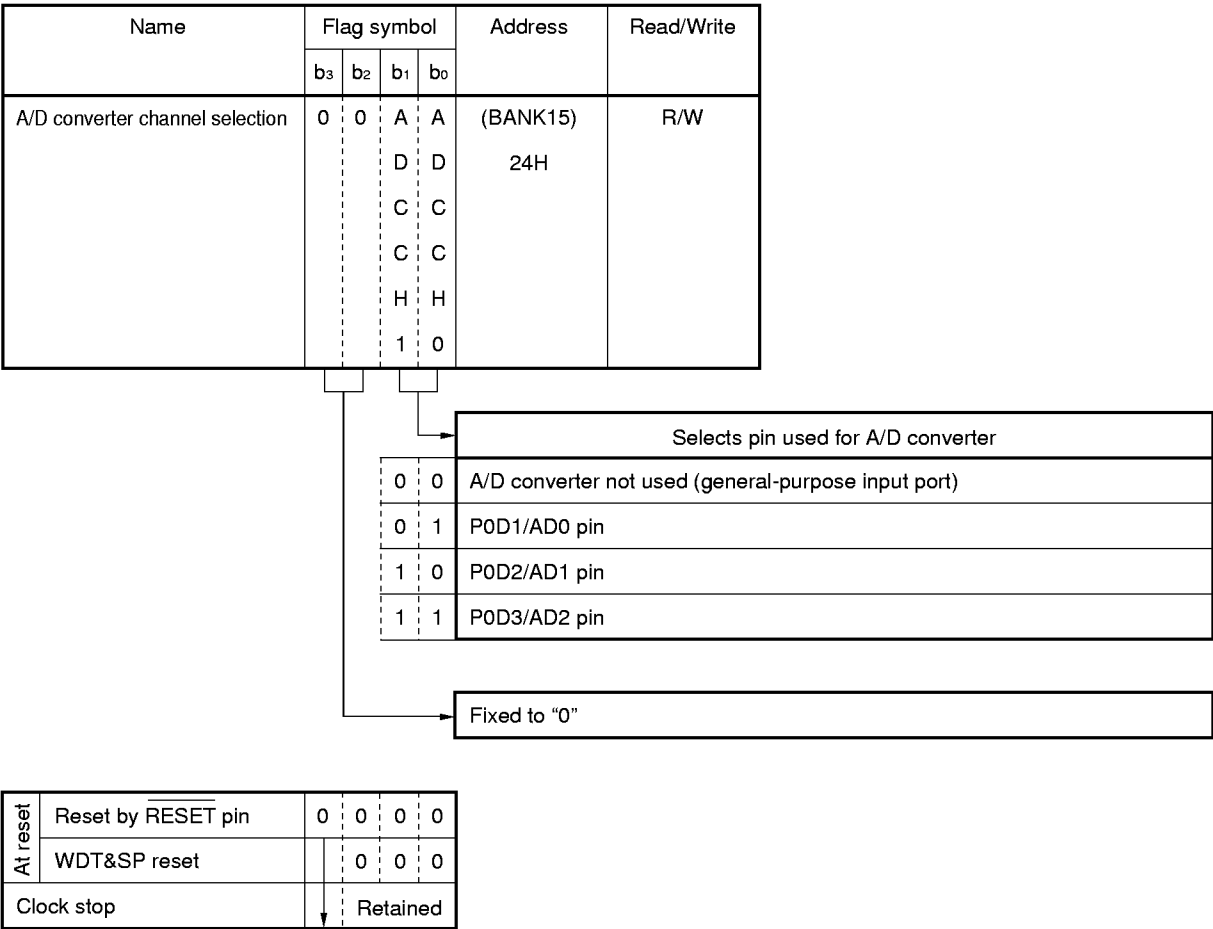


Figure 14-3. Configuration of A/D Converter Channel Selection Register



14.3 Compare Voltage Generation and Compare Blocks

Figure 14-4 shows the configuration of the compare voltage generation block and compare block.

The compare voltage generation block switches a tap decoder according to the 8-bit data set to the A/D converter reference voltage setting register and generates 256 different of compare voltages V_{ADCREf} .

In other words, this block is an R-string D/A converter.

The supply voltage to this R-string D/A converter is the same as the supply voltage V_{DD} of the device.

The compare block compares voltage V_{ADCIN} input from a pin with compare voltage V_{ADCREf} .

Comparison by the comparator is performed as soon as data has been written to the ADCSTRT flag. It takes the A/D converter the time of executing two instructions ($106.6 \mu s$) to perform comparison once.

The current operating status of the comparator can be checked by reading the content of the ADCSTRT flag.

The result of the comparison is detected by the ADCCMP flag.

Figure 14-5 shows the configuration of A/D converter mode selection register.

Figure 14-4. Configuration of Compare Voltage Generation and Compare Blocks

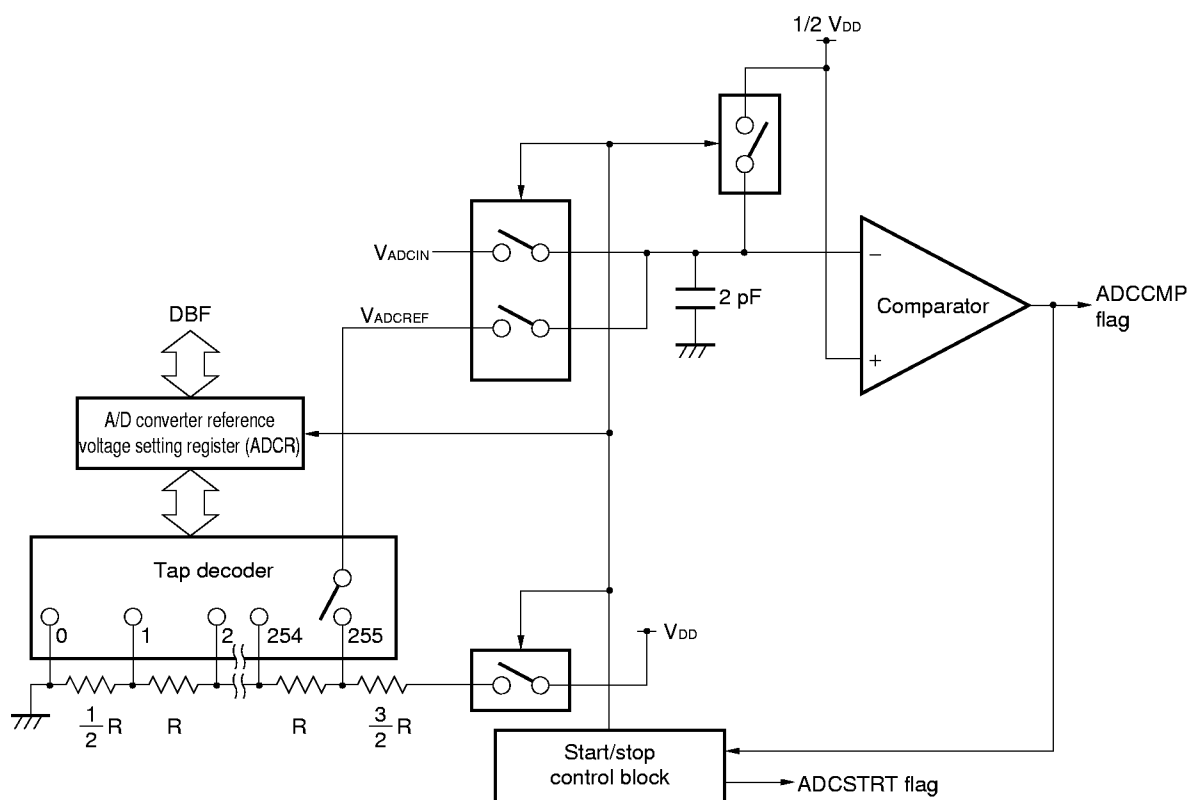
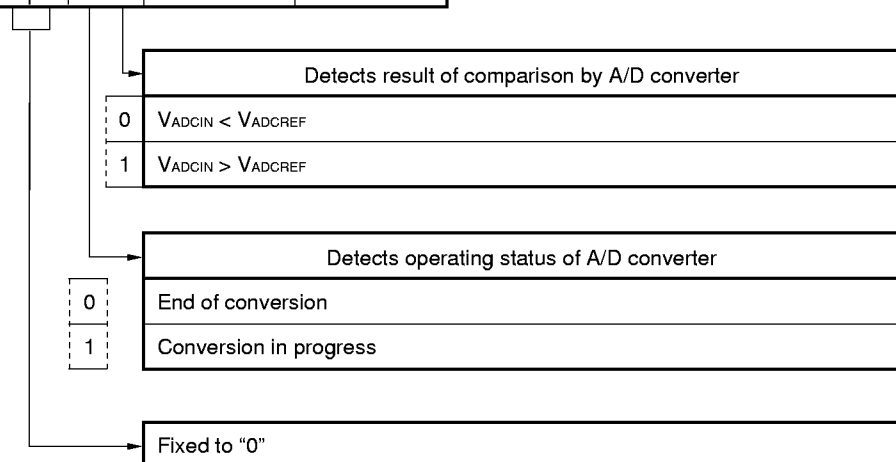


Figure 14-5. Configuration of A/D Converter Mode Selection Register

Name	Flag symbol				Address	Read/Write
	b ₃	b ₂	b ₁	b ₀		
A/D converter mode selection	0	0	A	A	(BANK15) 25H	R/W
			D	D		
			C	C		
			S	C		
			T	M		
			R	P		
			T			



At reset	Reset by $\overline{\text{RESET}}$ pin	0	0	0	0
	WDT&SP reser			0	0
Clock stop		↓	↓	0	R

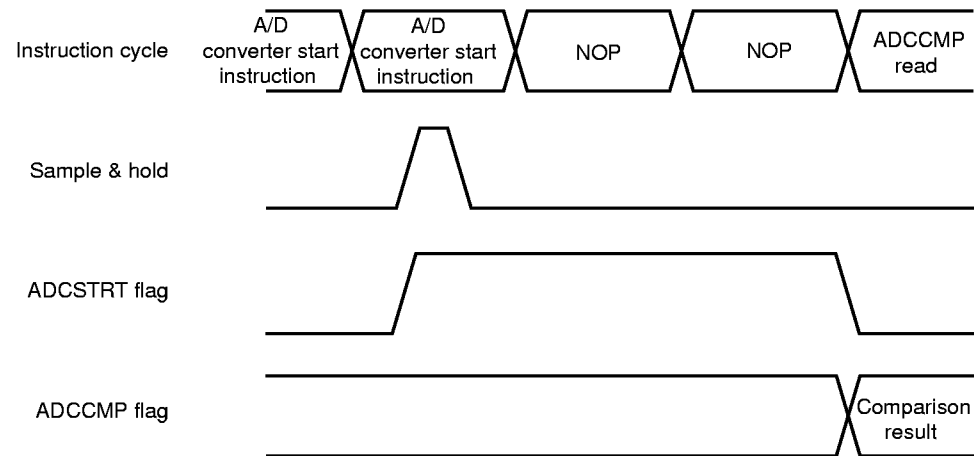
R:Retained

14.4 Comparison Timing Chart

The ADCSTRT flag is reset to 0 two instructions after the ADCSTRT flag has been set. At this point, the compare result (ADCCMP flag) can be read.

Figure 14-6 shows the timing chart.

Figure 14-6. Timing Chart of A/D Converter's Compare Operation



14.5 Using A/D Converter

14.5.1 Comparing with one compare voltage

An example of a program in this mode is shown below.

Example To compare input voltage V_{ADCIN} of AD0 pin with compare voltage V_{ADCREf} (127.5/256 V_{DD}), and branch to AAA if $V_{\text{ADCIN}} < V_{\text{ADCREf}}$, or to BBB if $V_{\text{ADCIN}} > V_{\text{ADCREf}}$

```

ADCR7 FLG          0.0EH.3 ; Defines each bit of DBF as ADCR data setting flag.
ADCR6 FLG          0.0EH.2
ADCR5 FLG          0.0EH.1
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

BANK15
INITFLG NOT P0DPLD3, NOT P0DPLD2, P0DPLD1, NOT P0DPLD0 ; Disconnects pull-down resistor of P0D1 pin.
BANK0
INITFLG NOT ADCCH1, ADCCH0 ; Selects AD0 pin for A/D converter.
INITFLG ADCR7, NOT ADCR6, NOT ADCR5, NOT ADCR4 ;
INITFLG NOT ADCR3, NOT ADCR2, NOT ADCR1, NOT ADCR0 ;
PUT ADCR, DBF ; Sets compare voltage  $V_{\text{ADCREf}}$ .
SET1 ADCSTRT ; Starts A/D conversion.
NOP ; Waits for duration of two instructions.
NOP ;
SKT1 ADCCMP ; Judges result of comparison.
BR AAA
BR BBB

```

14.5.2 Successive comparison by means of binary search

The A/D converter can compare only one compare voltage at a time.

Consequently, successive comparison must be executed through program in order to convert input voltages into digital signals.

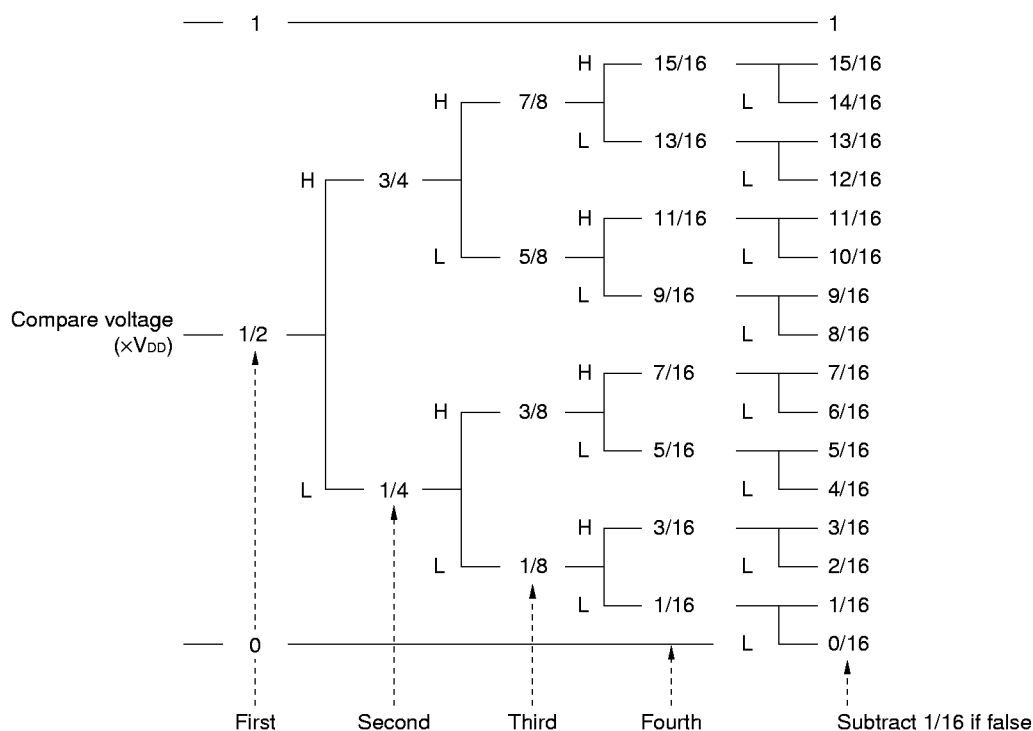
If the processing time of the successive comparison program is different depending on the input voltage, it is not desirable because of the relations with the other programs.

Therefore, the binary search method described in (1) through (3) below is useful.

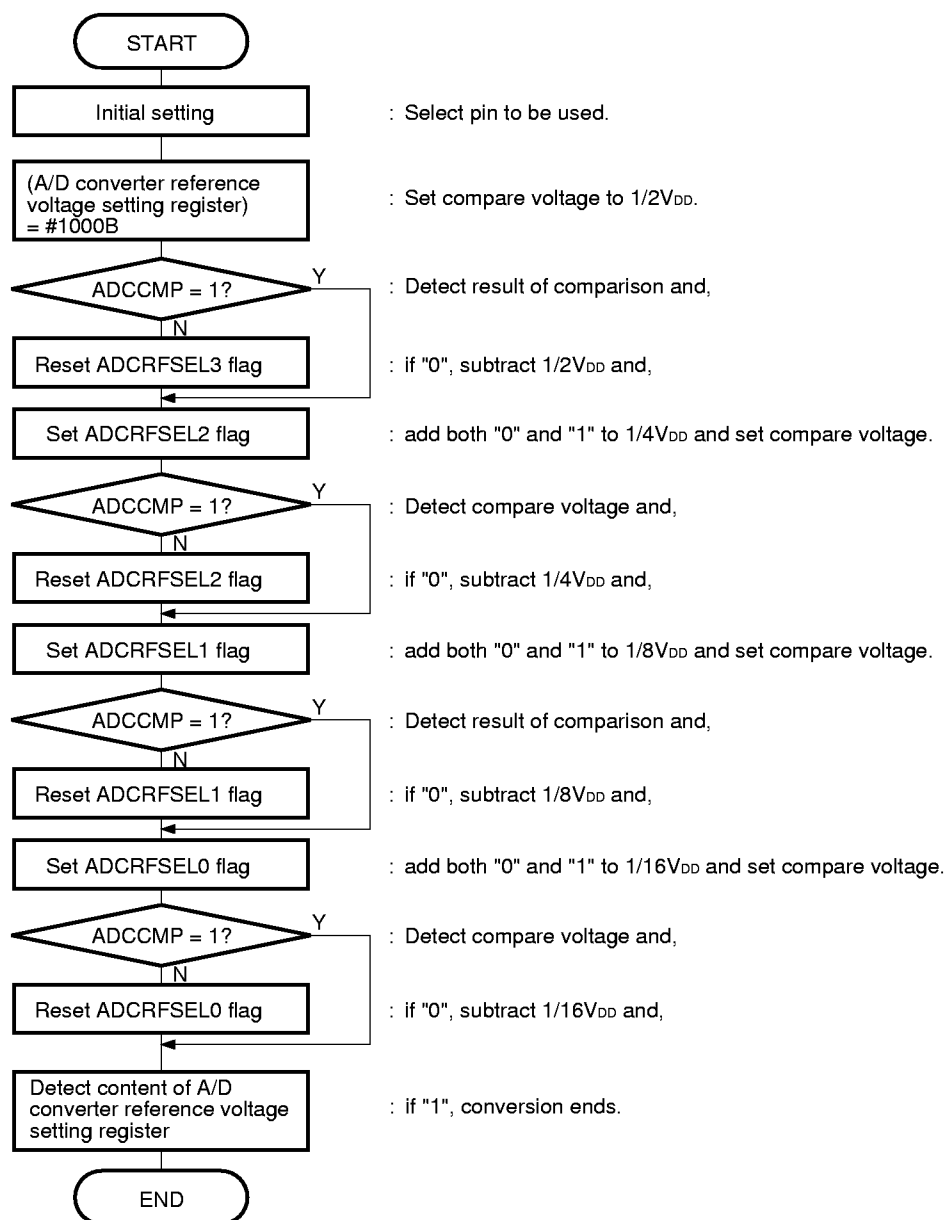
(1) Concept of binary search

The following figure illustrates the concept of binary search.

First, the compare voltage is set to $1/2V_{DD}$. If the result of comparison is True (high level), a voltage of $1/4V_{DD}$ is applied; if the result is False (low level), a voltage of $1/4V_{DD}$ is subtracted for comparison. Similarly, comparison is performed in sequence from $1/8V_{DD}$ to $1/16V_{DD}$. If the result is False after comparison has been executed four times, $1/16V_{DD}$ is subtracted, and the comparison ends.



(2) Flowchart of binary search



(3) Program example of binary search

START:

```

BANK1
INITFLG  NOT ADCCH1, ADCCH0                ; Selects AD0 pin.
INITFLG  P0DPLD1                          ; Sets pull-down resistor of AD0 pin OFF.
INITFLG  NOT ADCRFSEL3, ADCRFSEL2, ADCRFSEL1, ADRFSEL0 ; Sets compare voltage to 7.5/16 VDD.
SET1     ADCSTRT                          ; A/D comparator starts operating.
NOP                                           ; 2 wait
NOP                                           ;
SKF1     ADCCMP                          ; Detects ADCCMP.
SET1     ADCRFSEL3                      ; If 0, adds 7.5/16 VDD and,
CLR1     ADCRFSEL2                      ; subtracts 3.5/16 VDD.
SET1     ADCSTRT                          ; A/D comparator starts operating.
NOP                                           ; 2 wait
NOP                                           ;
SKF1     ADCCMP                          ; Detects ADCCMP.
SET1     ADCRFSEL2                      ; If 0, adds 3.5/16 VDD and,
CLR1     ADCRFSEL1                      ; subtracts 1.5/16 VDD.
SET1     ADCSTRT                          ; A/D comparator starts operating.
NOP                                           ; 2 wait
NOP                                           ;
SKF1     ADCCMP                          ; Detects ADCCMP.
SET1     ADCRFSEL1                      ; If 0, adds 1.5/16 VDD and,
CLR1     ADCRFSEL0                      ; subtracts 0.5/16 VDD.
SET1     ADCSTRT                          ; A/D comparator starts operating.
NOP                                           ; 2 wait
NOP                                           ;
SKF1     ADCCMP                          ; Detects ADCCMP.
SET1     ADCRFSEL0                      ; If 0, adds 0.5/16 VDD.

```

END:

14.6 Cautions on Using A/D Converter

14.6.1 Cautions on selecting A/D converter pin

When one of the P0D1/AD0 through P0D3/AD2 pins is selected, the other two pins are forcibly set in the input port mode. The P0D1/AD0 through P0D3/AD2 pins can be connected to a pull-down resistor if so specified by the P0DPL1 through P0DPLD3 flags in bank 15. To use the P0D1/AD0 through P0D3/AD2 pins for the A/D converter, therefore, disconnect their pull-down resistors to correctly detect an external input analog voltage.

14.7 Status at Reset

14.7.1 At reset by $\overline{\text{RESET}}$ pin

All the P0D1/AD0 through P0D3/AD2 pins are set in the general-purpose input port mode.

The P0D1 through P0D3 pins are connected with a pull-down resistor.

14.7.2 At WDT&SP reset

All the P0D1/AD0 through P0D3/AD2 pins are set in the general-purpose input port mode.

The P0D1 through P0D3 pins are connected with a pull-down resistor.

14.7.3 On execution of clock stop instruction

The status of the pin selected for the A/D converter is retained as is.

The previous status of the pull-down resistor of the P0D1 through P0D3 pins is retained.

14.7.4 In halt status

The status of the pin selected for the A/D converter is retained as is.

The previous status of the pull-down resistor of the P0D1 through P0D3 pins is retained.

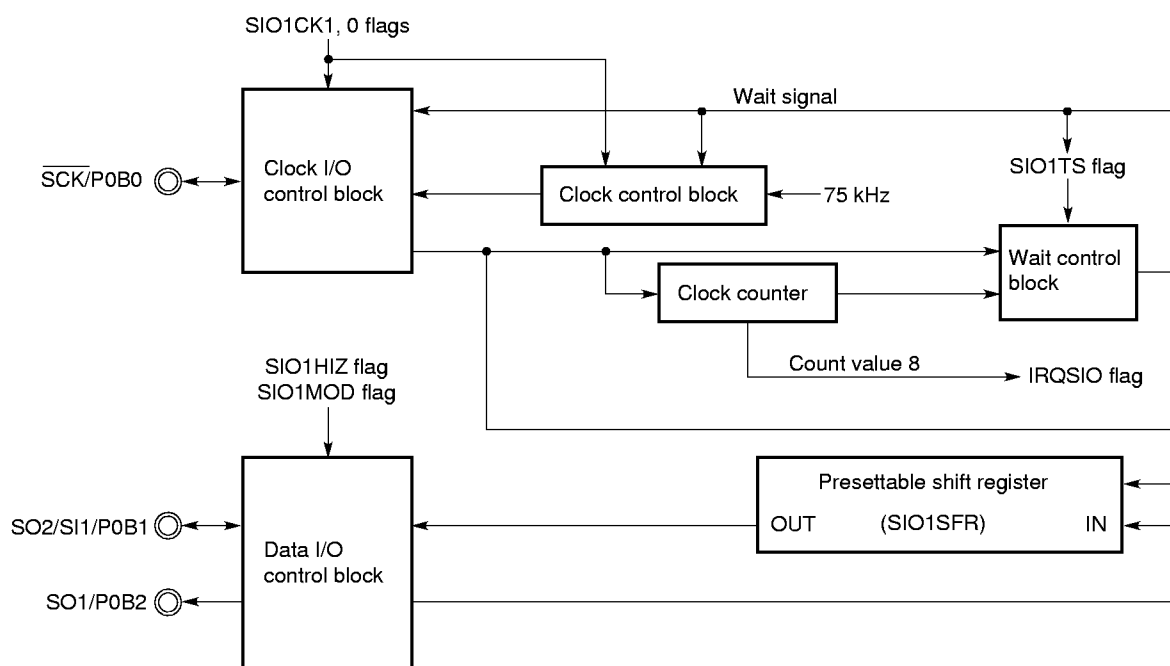
15. SERIAL INTERFACE

15.1 General

Figure 15-1 shows the outline of the serial interface.

This serial interface is of two-wire/three-wire serial I/O type. The former type uses \overline{SCK} and SO2/SI1 pins. The latter uses \overline{SCK} , SI1, and SO1 pins.

Figure 15-1. Outline of Serial Interface



- Remarks 1.** SIO1CK1 and 0 (bits 0 and 1 of serial I/O clock select register. Refer to **Figure 15-2**) set a shift clock.
- 2.** SIO1TS (bit 0 of serial I/O mode select register. Refer to **Figure 15-3**) starts/stops communication.
- 3.** SIO1HIZ (bit 1 of serial I/O mode select register. Refer to **Figure 15-3**) sets the function of the SO1/P0B2 pin.
- 4.** SIO1MOD (bit 3 of serial I/O mode select register. Refer to **Figure 15-3**) selects I/O of SO2/SI1/P0B1 pin.

15.2 Clock Input/Output Control Block and Data Input/Output Control Block

The clock input/output control block and data input/output control block select the operation mode of the serial interface (2-wire or 3-wire mode), control the transmit/receive operation, and select a shift clock.

The flags that control these blocks are allocated to the serial I/O clock select register and serial I/O mode select register.

Figure 15-2 shows the configuration and function of the serial I/O clock select register.

Figure 15-3 shows the configuration and function of the serial I/O mode select register.

Table 15-1 shows the setting status of each pin by the corresponding control flags. As shown in this table, the input/output setting flag of each pin must be also manipulated in addition to the control flag of the serial interface, to set each pin.

The SIO1CK1 and 0 flags select the internal clock (master) or external clock (slave) operation.

The SIO1HIZ flag selects whether the SO1/P0B2 pin is used as a serial data output pin.

The SIO1MOD flag selects whether the SO1/SI1/P0B1 pin is used as a serial data input (SI1 pin) or serial data output (SO2) pin.

Figure 15-2. Configuration of Serial I/O Clock Select Register

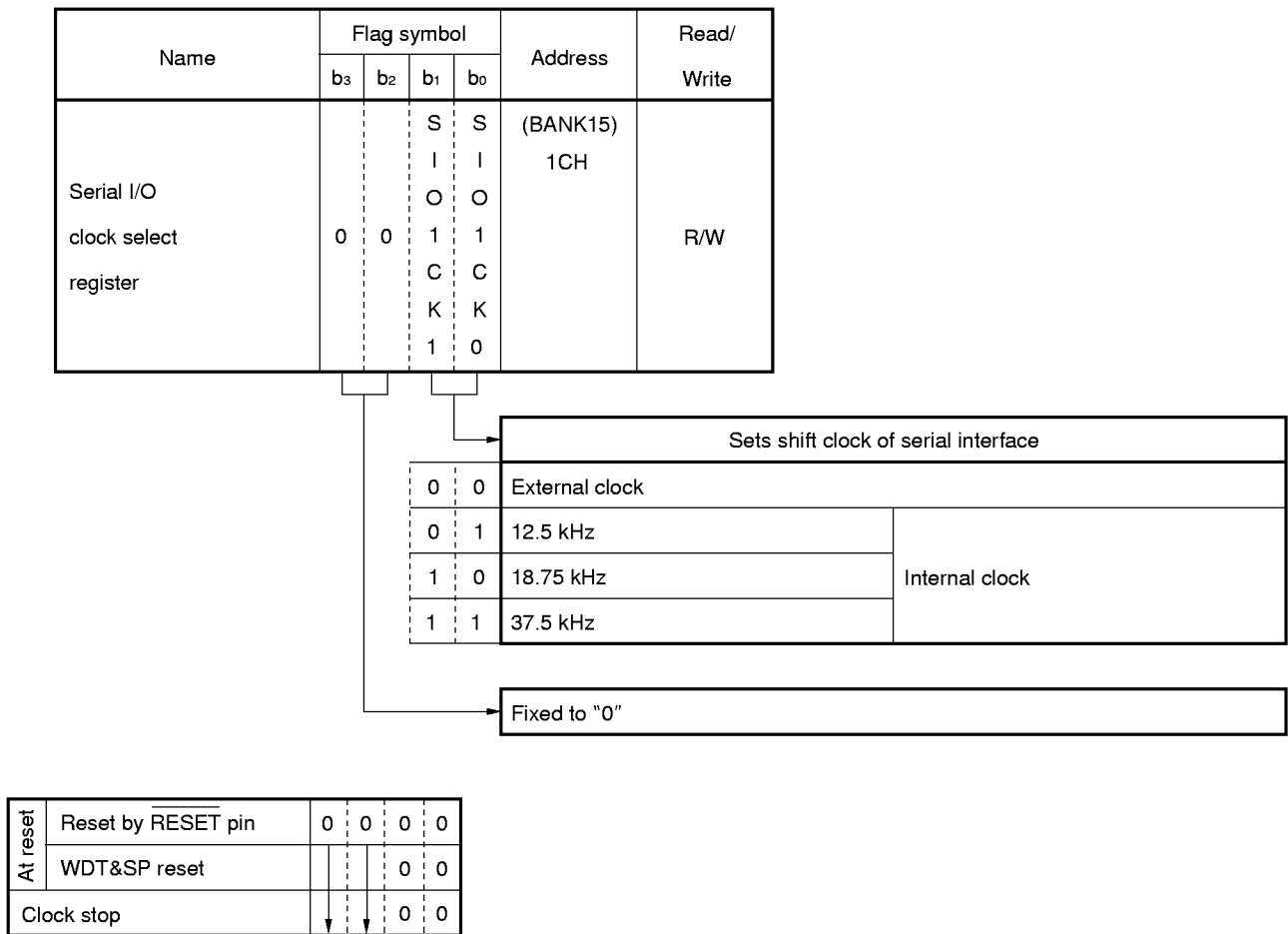
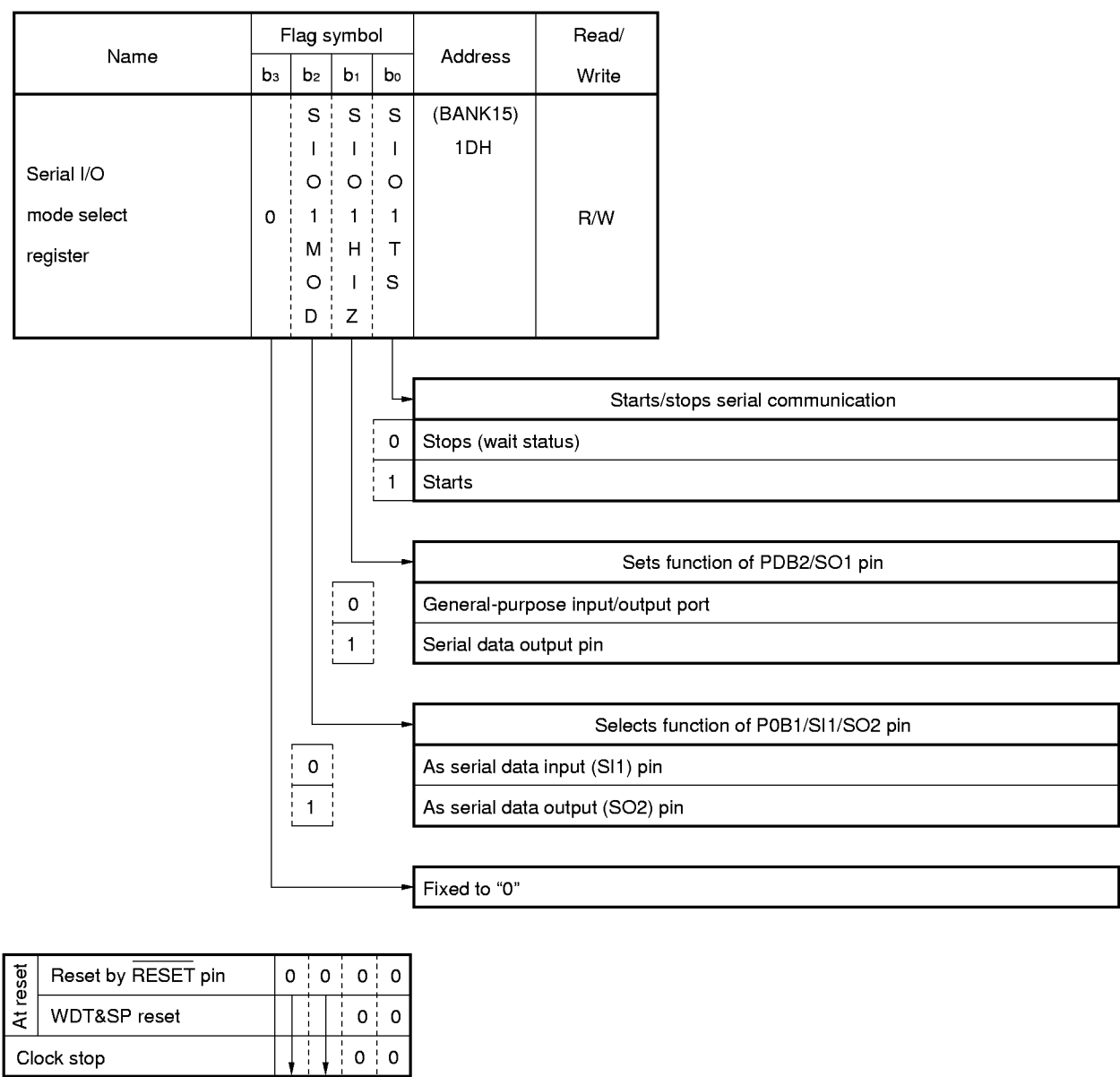


Figure 15-3. Configuration of Serial I/O Mode Select Register



★

Table 15-1. Set Status of Each Pin By Control Flags

Control flags of serial interface								I/O setting flag of each pin														
Communi- cation mode	S I O 1 M O D	Serial I/O select	S I O 1 H I Z	Serial interface pin setting	S I O 1 C K 1	S I O 1 C K 0	Clock setting	Pin name	P 0 B 0 B 1 O 2	P 0 B 0 B 1 O 1	P 0 B 0 B 1 O 0	Set status of pin										
3-wire serial I/O ^{Note 1} and 2-wire serial I/O ^{Note 2}					0	0	External clock	P0B0/ $\overline{\text{SCK}}$			0	During wait: general-purpose input port Wait released: external clock input										
											1	General-purpose output port										
											0	0	General-purpose input port									
													1	During wait: waits for internal clock output Wait released: internal clock output								
							0		Internal clock (reception)					P0B1/SI1/ SO2		0	During wait: general-purpose input port Wait released: serial input					
					1	Output (trans- mission)														1	General-purpose output port	
																					0	During wait: waits for serial output
																					1	Wait released: serial output
				0	General- purpose I/O			P0B2/SO1	0		General-purpose input port											
											1	Serial output				1	General-purpose output port					
																	0	During wait: waits for serial output				
																	1	Wait released: serial output				

Notes 1. To set the 3-wire serial I/O mode, be sure to reset SIO1MOD to 0 and set SIO1HIZ to 1.

2. To use the 2-wire serial I/O mode, be sure to reset SIO1HIZ to 0.

15.2.1 Setting 2-/3-wire mode

The serial interface uses two pins in the two-wire mode: $\overline{\text{SCK}}/\text{P0B0}$ and $\text{SO2}/\text{SI1}/\text{P0B1}$.

The $\overline{\text{SCK}}/\text{P0B0}$ pin is used as a shift clock input/output pin, and the $\text{SO2}/\text{SI1}/\text{P0B1}$ pin is used as a serial data input/output pin. The $\text{SO1}/\text{P0B2}$ pin is not used for the serial interface and is set in the general-purpose output port mode by the SIO1HIZ flag. In this way, the serial interface operates in the two-wire mode.

In the three-wire mode, three pins, $\overline{\text{SCK}}/\text{P0B0}$, $\text{SO1}/\text{P0B2}$, and $\text{SO2}/\text{SI1}/\text{P0B1}$ are used.

The $\overline{\text{SCK}}/\text{P0B0}$ is used as a shift clock input/output pin, the $\text{SO1}/\text{P0B2}$ pin is used as a serial data output pin, and the $\text{SO2}/\text{SI1}/\text{P0B1}$ pin is used as a serial data input pin.

Unlike in the two-wire mode, the $\text{SO1}/\text{P0B2}$ pin is used as a serial data output pin according to the setting of the SIO1HIZ flag. The $\text{SO2}/\text{SI1}/\text{P0B1}$ pin is used as a serial data input pin according to the setting of the SIO1MOD flag.

In this way, the serial interface operates in the three-wire mode.

15.2.2 Selecting data input/output using 2-wire serial interface

In the two-wire mode, the $\text{SO2}/\text{SI1}/\text{P0B1}$ pin is used as an input/output pin for serial data.

Whether the $\text{SO2}/\text{SI1}/\text{P0B1}$ pin is used as a serial data input pin (SI1 pin) or serial data output pin (SO2 pin) is specified by the SIO1MOD flag (refer to **Figure 15-3 Configuration of Serial I/O Mode Select Register**).

15.3 Clock Control Block

The clock control block generates a clock when the internal clock is used (master operation), and controls clock output timing.

The frequency f_{SC} of the internal clock is set by the SIO1CK0 and SIO1CK1 flags of the serial I/O clock select register.

Figure 15-2 shows the configuration and function of the serial I/O clock select register.

For the clock generation timing, refer to **15.7 Operation of Serial Interface**.

15.4 Clock Counter

The clock counter counts the shift clock output or input from the shift clock pin ($\overline{\text{SCK}}/\text{P0B0}$ pin).

Because the clock counter directly reads the status of the clock pin, it cannot identify whether the clock is an internal clock or an external clock.

The contents of the clock counter cannot be directly read by software.

For the operation and timing chart of the clock counter, refer to **15.7 Operation of Serial Interface**.

15.5 Presetable Shift Register

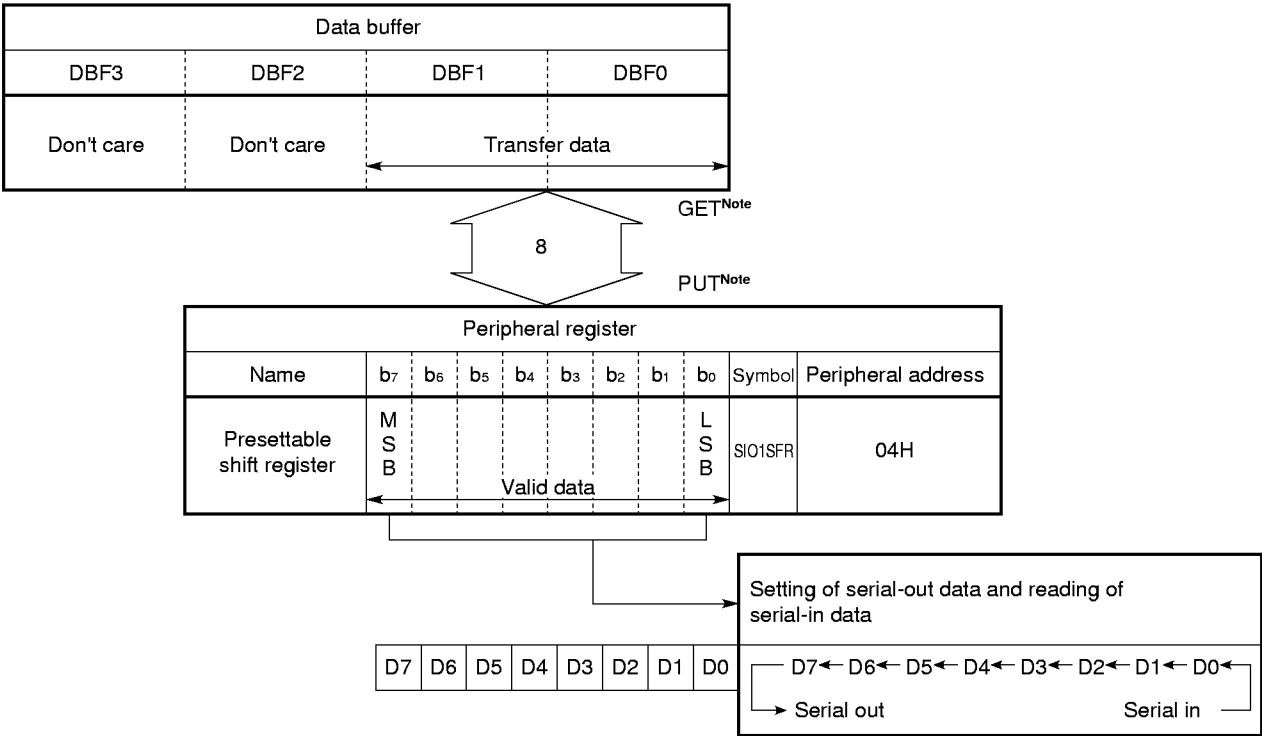
The presetable shift register is an 8-bit shift register that writes serial-out data and reads serial-in data.

Writing/reading data to/from the presetable shift register is performed by PUT and GET instructions via data buffer.

The presetable shift register outputs (transmits) the content of its most significant bit (MSB) from the serial data I/O pin in synchronization with the falling edge of the shift clock, and reads data to its least significant bit (LSB) in synchronization with the rising edge of the shift clock.

Figure 15-4 shows the configuration and function of the presetable shift register.

Figure 15-4. Configuration of Presetable Shift Register



Note If the PUT or GET instruction is executed during serial communication, the data may be lost. For details, refer to 15.8 Notes on Setting and Reading Data.

15.6 Wait Control Block

The wait control block performs wait (pause) control of communication.

By releasing the wait status by using the SIO1TS flag of the serial I/O mode select register, serial communication is started.

After the wait status has been released, and communication has been started, the wait status is resumed if shift clock rises at clock counter "8".

The communication status can be detected by the SIO1TS flag.

That is, the communication status can be detected by detecting the status of the SIO1TS flag after setting "1" to the SIO1TS flag.

If "0" is written to the SIO1TS flag while the wait status is released, the wait status is set. This is called a forced wait status.

For the configuration and function of the serial I/O mode select register, refer to Figure 15-3.

15.7 Serial Interface Operation

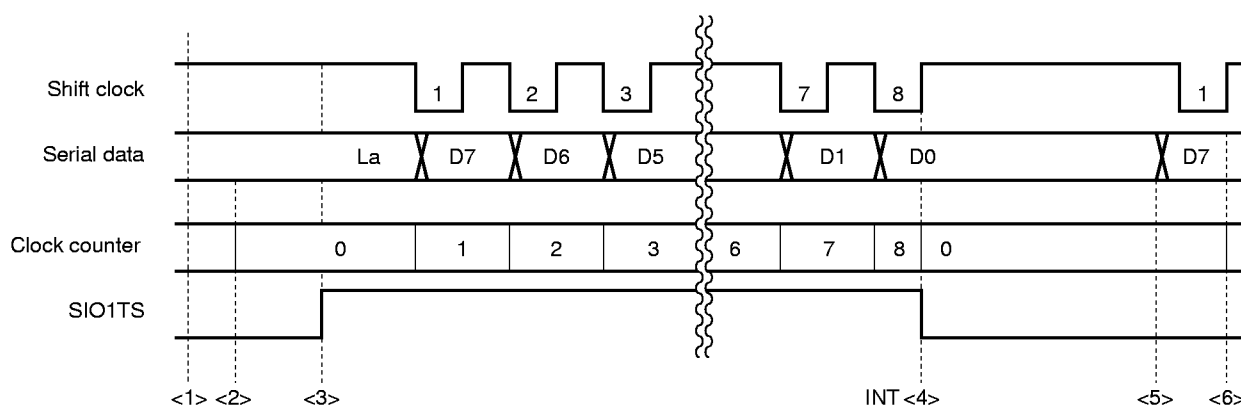
The timing of each operation of the serial interface is described below.

This timing is applicable to both 2-wire and 3-wire modes.

15.7.1 Timing chart

Figure 15-5 shows a timing chart.

Figure 15-5. Timing Chart of Serial Interface



- Remark**
- <1> Initial status (general-purpose input port)
 - <2> Start condition satisfied by general-purpose I/O port
 - <3> Wait released
 - <4> Wait timing
 - <5> General-purpose input port mode set
 - <6> Stop condition satisfied by general-purpose I/O port

15.7.2 Operation of clock counter

The initial value of the clock counter is "0". The value of the clock counter is incremented each time the falling edge of the clock pin has been detected. When the value of the clock counter reaches "8", it is reset to "0" at the next rising edge of the clock pin. After the clock counter has been reset to "0", the serial communication is placed in the wait status.

The conditions under which the clock counter is reset are as follows:

- (1) At power-ON reset
- (2) When clock stop instruction is executed
- (3) When "0" is written to SIO1TS flag
- (4) If shift clock rises while wait status is released and present value of clock counter is "8"

15.7.3 Wait operation and note

When the wait status has been released, serial data is output at the next falling edge of the clock (transmission operation), and the wait released status continues until eight clocks are counted.

After the eight clocks have been output, make the shift clock pin high and stop the operations of the clock counter and presetable shift register.

Note that, 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 is not set.

If data is written to the presetable shift register while the wait status is released and the shift clock pin is low, the content of the MSB of the data is output to the serial data output pin when the "PUT" instruction is executed.

If the forced wait status is set while the wait status is released, the wait status is immediately set when "0" is written to the SIO1TS flag.

15.7.4 Interrupt request issuance timing

An interrupt request is issued when eight clocks have been transmitted (received).

15.7.5 Shift clock generation timing

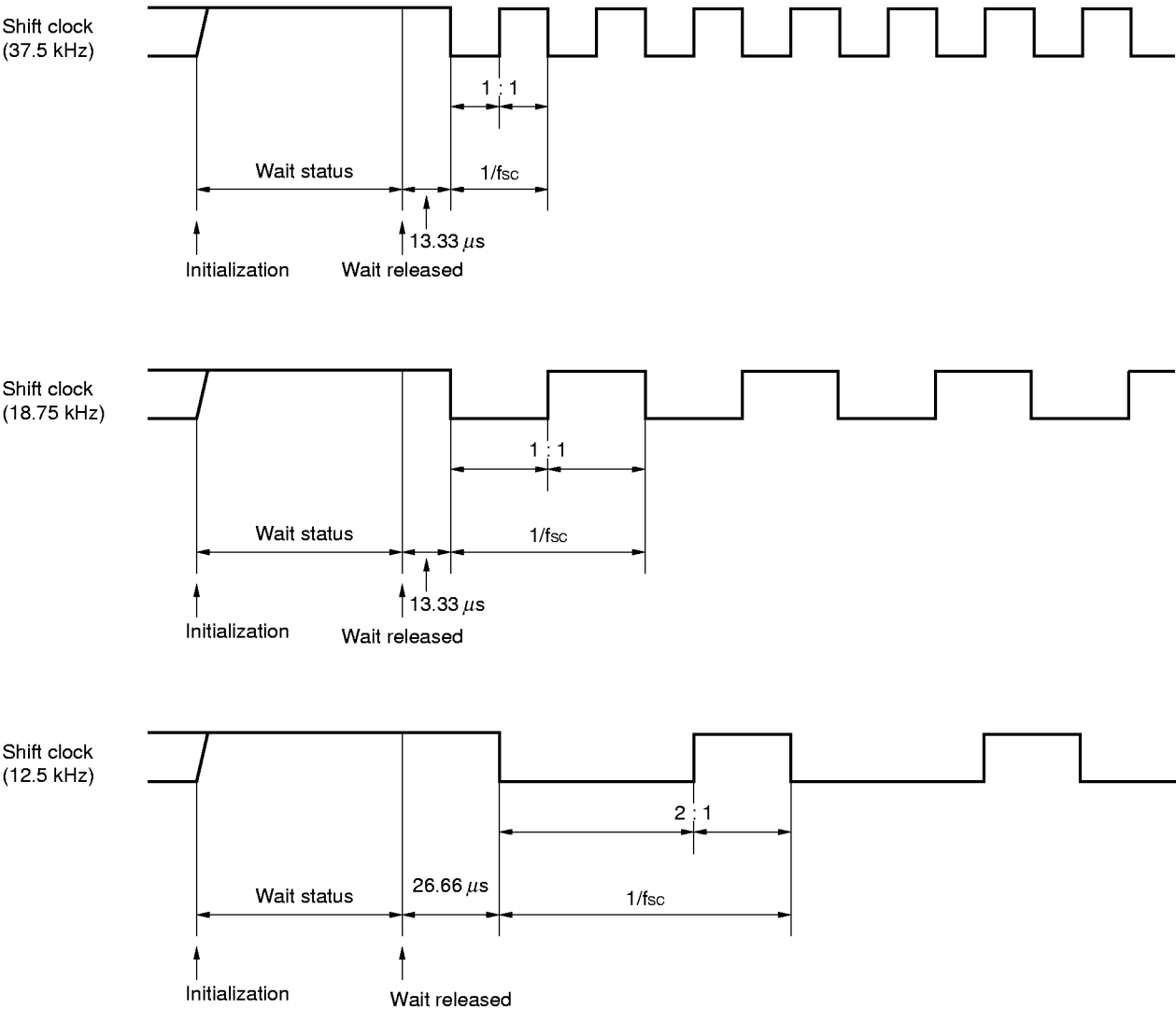
(1) When wait status is released from initial status

The "initial status" means the point at which the P0B0/ $\overline{\text{SCK}}$ pin has been made high with the internal clock operation selected.

During the wait status, a high level is output to the shift clock pin.

The wait status can be released and a clock can be selected at the same time.

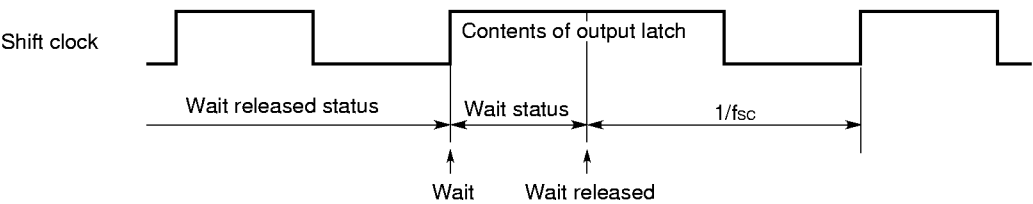
Figure 15-6. Shift Clock Generation Timing of Serial Interface (1/4)



(2) When wait operation is performed

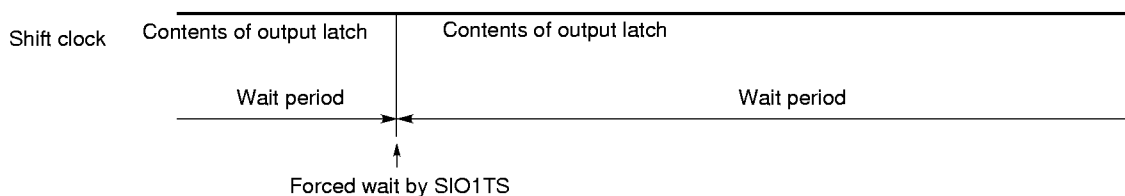
(a) When wait status is set at the 8th clock (normal operation)

Figure 15-6. Shift Clock Generation Timing of Serial Interface (2/4)



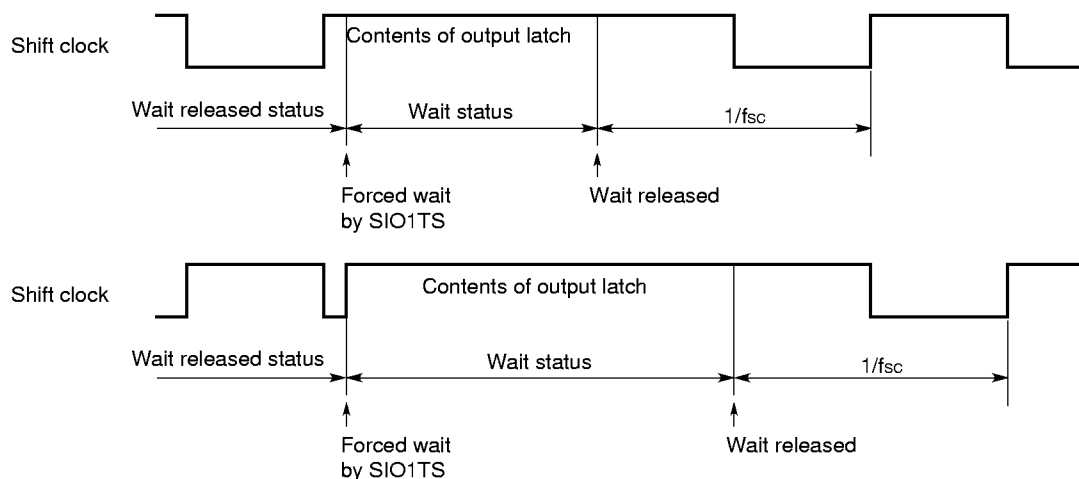
(b) When forced wait status is set during wait status

Figure 15-6. Shift Clock Generation Timing of Serial Interface (3/4)



(c) When forced wait status is set while wait status is released

Figure 15-6. Shift Clock Generation Timing of Serial Interface (4/4)



(d) When wait status is released while wait status is released

The clock output waveform does not change. Neither is the counter reset. However, do not change the clock frequency while the wait status is released.

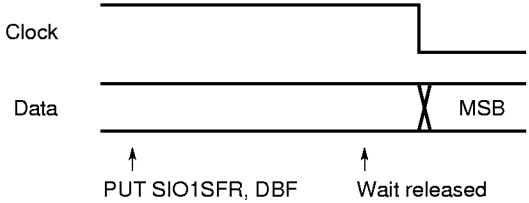
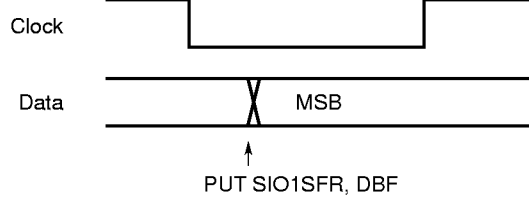
15.8 Notes on Setting and Reading Data

Use the "PUT SIO1SFR, DBF" instruction to set data to the presetable shift register. Use the "GET DBF, SIO1SFR" instruction to read data.

Set or read the data 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.

The following table describes the points to be noted in setting and reading data.

Table 15-2. Data Read and Write Operations of Presetable Shift Register and Notes

Status on execution of PUT/GET		Status of shift clock pin	Operation of presetable shift register
Wait status	Read (GET)	<ul style="list-style-type: none"> Floating with external clock Value of output latch with internal clock. Normally, used with high level 	Normal read
	Write (PUT)		Normal write Content of MSB is output as data at falling edge of shift clock after wait status is released next (transmission operation). 
Wait release status	Read (GET)	Low level	Normal read
		High level	Cannot be read normally. Contents of SIO1SFR are lost.
	Write (PUT)	Low level	Cannot be written normally. Contents of SIO1SFR are lost.
		High level	Normal write Content of MSB is output as data when PUT instruction is executed. Clock counter is not reset. 

15.9 Operation Mode and Operational Outline of Each Blocks

Tables 15-3 and 15-4 outline the operations of the serial interface.

Table 15-3. Operation in 3-wire Serial I/O Mode

Operation mode Item		Slave operation (SIO1CK1 = SIO1CK0 = 0)		Master operation (SIO1CK1 = SIO1CK0 = other than 0)	
		During wait (SIO1TS = 0)	Wait released (SIO1TS = 1)	During wait (SIO1TS = 0)	Wait released (SIO1TS = 1)
Status of each pin	$\overline{\text{SCK}}/\text{P0B0}$	<ul style="list-style-type: none">When P0BBIO0 = 0 General-purpose input portWhen P0BBIO0 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO0 = 0 External clock input portWhen P0BBIO0 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO0 = 0 General-purpose input portWhen P0BBIO0 = 1 Waits for internal clock output	<ul style="list-style-type: none">When P0BBIO0 = 0 General-purpose input portWhen P0BBIO0 = 1 Internal clock output
	SI1/SO2/P0B1	SIO1MOD = 0			
		<ul style="list-style-type: none">When P0BBIO1 = 0 General-purpose input portWhen P0BBIO1 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO1 = 0 Serial inputWhen P0BBIO1 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO1 = 0 General-purpose input portWhen P0BBIO1 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO1 = 0 Serial inputWhen P0BBIO1 = 1 General-purpose output port
	SO1/P0B2	SIO1HIZ = 1			
		Waits for serial output	Serial output	Waits for serial output	Serial output
Program counter		Incremented at falling edge of $\overline{\text{SCK}}$ pin			
Operation of presetable shift register	Output	<ul style="list-style-type: none">When SIO1HIZ = 0 Not outputWhen SIO1HIZ = 1 Shifted from MSB and output from SO1 pin at falling edge of $\overline{\text{SCK}}$ pin			
	Input	<ul style="list-style-type: none">When SIO1MOD = 0 Shifted from LSB and status of SI1 pin is input at rising edge of $\overline{\text{SCK}}$ pin. If SI1 pin is set in output mode, however, contents of output latch are input.			

Table 15-4. Operation in Two-Wire Serial I/O Mode

Operation mode		Slave operation (SIO1CK1 = SIO1CK0 = 0)		Master operation (SIO1CK1 = SIO1CK0 = other than 0)	
Item		During wait (SIO1TS = 0)	Wait released (SIO1TS = 1)	During wait (SIO1TS = 0)	Wait released (SIO1TS = 1)
Status of each pin	$\overline{\text{SCK}}/\text{P0B0}$	<ul style="list-style-type: none">When P0BBIO0 = 0 General-purpose input portWhen P0BBIO0 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO0 = 0 External clock input portWhen P0BBIO0 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO0 = 0 General-purpose input portWhen P0BBIO0 = 1 Waits for internal clock output	<ul style="list-style-type: none">When P0BBIO0 = 0 General-purpose input portWhen P0BBIO0 = 1 Internal clock output
	SI1/SO2/P0B1	SIO1MOD = 0			
		<ul style="list-style-type: none">When P0BBIO1 = 0 General-purpose input portWhen P0BBIO1 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO1 = 0 Serial inputWhen P0BBIO1 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO1 = 0 General-purpose input portWhen P0BBIO1 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO1 = 0 Serial inputWhen P0BBIO1 = 1 General-purpose output port
		SIO1MOD = 1			
		Waits for serial output regardless of P0BBIO1	Serial output regardless of P0BBIO1	Waits for serial output regardless of P0BBIO1	Serial output regardless of P0BBIO1
	SO1/P0B2	SIO1HIZ = 0			
		<ul style="list-style-type: none">When P0BBIO2 = 0 General-purpose input portWhen P0BBIO2 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO2 = 0 Serial inputWhen P0BBIO2 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO2 = 0 General-purpose input portWhen P0BBIO2 = 1 General-purpose output port	<ul style="list-style-type: none">When P0BBIO2 = 0 Serial inputWhen P0BBIO2 = 1 General-purpose output port
SIO1MOD = 1					
	Waits for serial output regardless of P0BBIO2	Serial output regardless of P0BBIO2	Waits for serial output regardless of P0BBIO2	Serial output regardless of P0BBIO2	
Program counter		Incremented at falling edge of $\overline{\text{SCK}}$ pin			
Operation of presettable shift register	Output	<ul style="list-style-type: none">When SIO1SEL = 1 Shifted from MSB and output from SO2 pin at falling edge of $\overline{\text{SCK}}$ pin			
	Input	<ul style="list-style-type: none">When SIO1SEL = 0 Shifted from LSB and status of SI1 pin is input at rising edge of $\overline{\text{SCK}}$ pin. If SI1 pin is set in output port mode, however, contents of output latch are input.			

15.10 Status on Reset

15.10.1 At reset by $\overline{\text{RESET}}$ pin

P0B0/ $\overline{\text{SCK}}$, P0B1/SI1/SO2, and P0B2/SO1 pins are set in the general-purpose input port mode.

The contents of the presetable shift register are undefined.

15.10.2 WDT&SP reset

P0B0/ $\overline{\text{SCK}}$ pin, P0B1/SI1/SO2, and P0B2/SO1 pins are set in the general-purpose input port.

The previous contents of the presetable shift register are retained.

15.10.3 At clock stop

All the pins hold the current status.

The previous contents of the presetable shift register are retained.

15.10.4 In halt status

All the pins hold the current status.

The internal clock stops output in the status in which the HALT instruction is executed.

When the external clock is used, the operation continued even if the HALT instruction is executed.

16. PLL FREQUENCY SYNTHESIZER

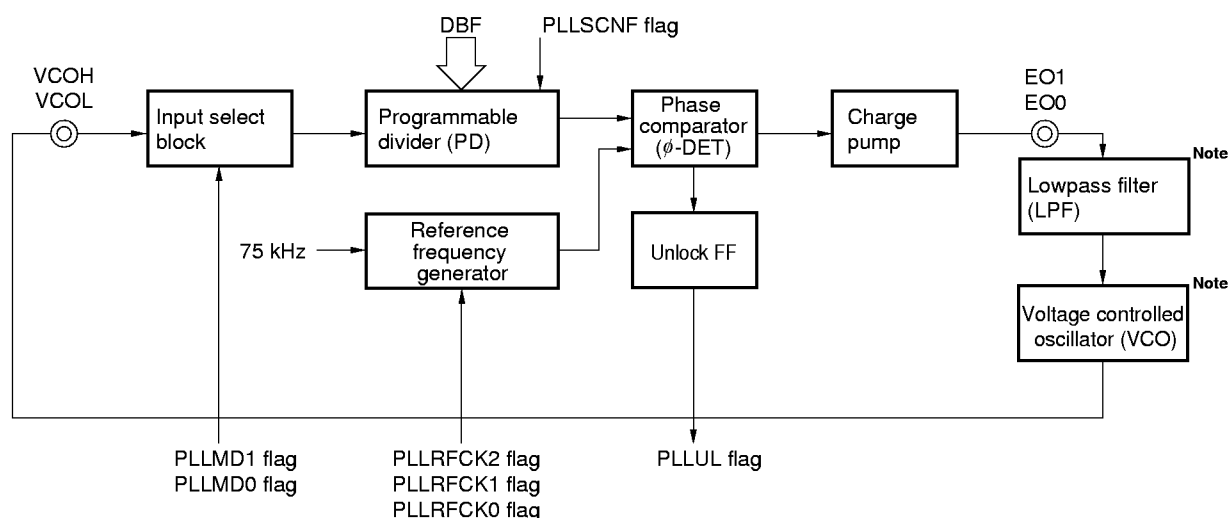
The PLL (Phase Locked Loop) frequency synthesizer is used to lock a frequency in the MF (Medium Frequency), HF (High Frequency), and VHF (Very High Frequency) to a constant frequency by means of phase difference comparison.

16.1 Outline of PLL Frequency Synthesizer

Figure 16-1 outlines the PLL frequency synthesizer. A PLL frequency synthesizer can be configured by connecting an external lowpass filter (LPF) and voltage controlled oscillator (VCO).

The PLL frequency synthesizer divides a signal input from the VCOH or VCOL pin by using a programmable divider and outputs a phase difference between this signal and a reference frequency from the EO0 and EO1 pins.

Figure 16-1. Outline of PLL Frequency Synthesizer



Note External circuit

- Remarks**
1. PLLMD1 and PLLMD0 (bits 1 and 0 of PLL mode selection register: refer to **Figure 16-3**) selects a division mode of the PLL frequency synthesizer.
 2. PLLSCNF (bit 3 of PLL mode selection register: refer to **Figure 16-3**) selects the least significant bit of the swallow counter.
 3. PLLRCK2 through PLLRCK0 (bits 3 through 0 of PLL reference frequency selection register: refer to **Figure 16-6**) selects a reference frequency f_r of the PLL frequency synthesizer.
 4. PLLUL (bit 0 of PLL unlock FF register: refer to **Figure 16-9**) detects the PLL unlock FF status.

16.2 Input Selection Block and Programmable Divider

16.2.1 Configuration and function of input selection block and programmable divider

Figure 16-2 shows the configuration of the input selection block and programmable divider.

The input selection block selects an input pin and division mode of the PLL frequency synthesizer.

The VCOH or VCOL pin can be selected as the input pin.

The voltage on the selected pin is at the intermediate level (approx. $1/2 V_{DD}$). The pin not selected is internally pulled down.

Because these pins are connected to an internal AC amplifier, cut the DC component of the input signal by connecting a capacitor in series to the pin.

Direct division mode and pulse swallow mode can be selected as division modes.

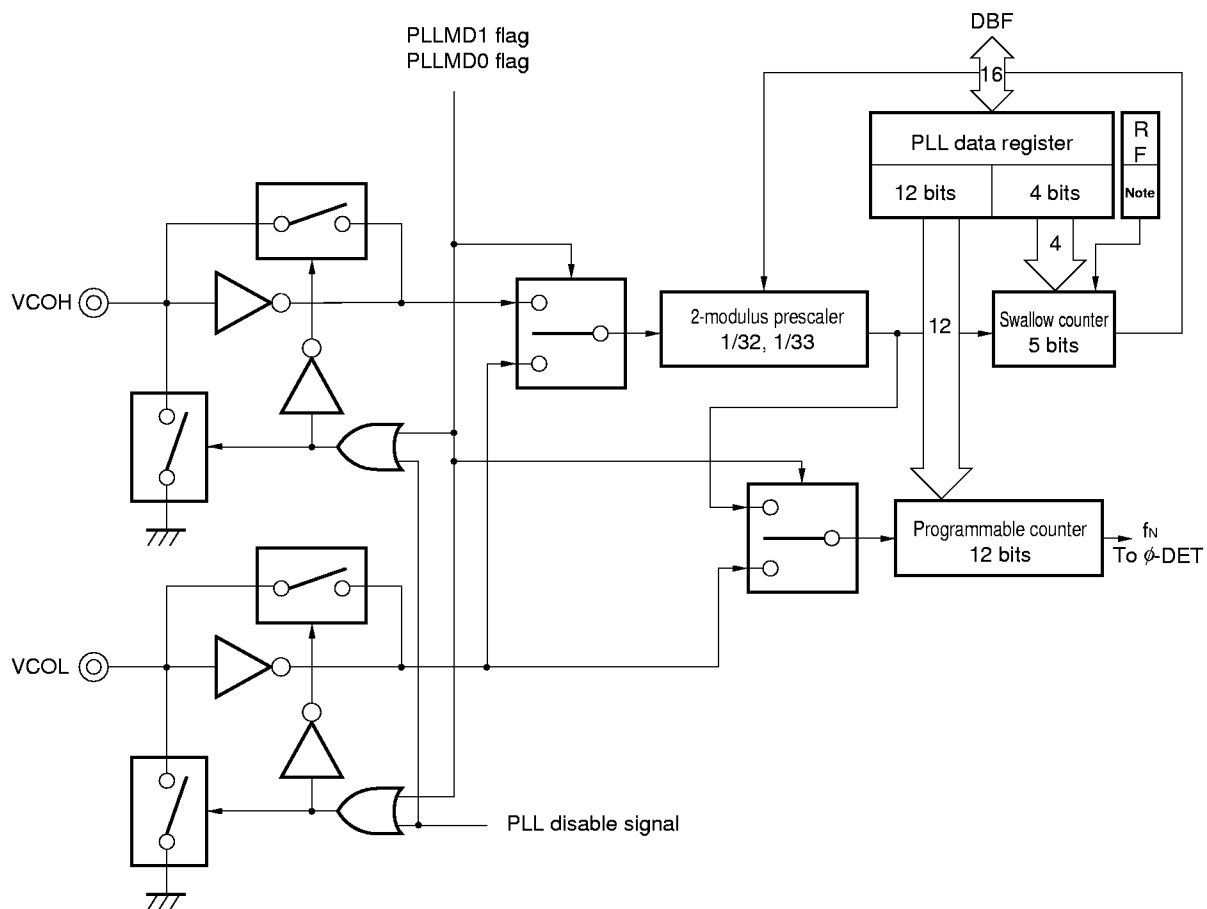
The programmable divider divides the frequency of the input signal according to the value set to the swallow counter and programmable counter.

The pin and division mode to be used are selected by the PLL mode selection register.

Figure 16-3 shows the configuration of the PLL mode selection register.

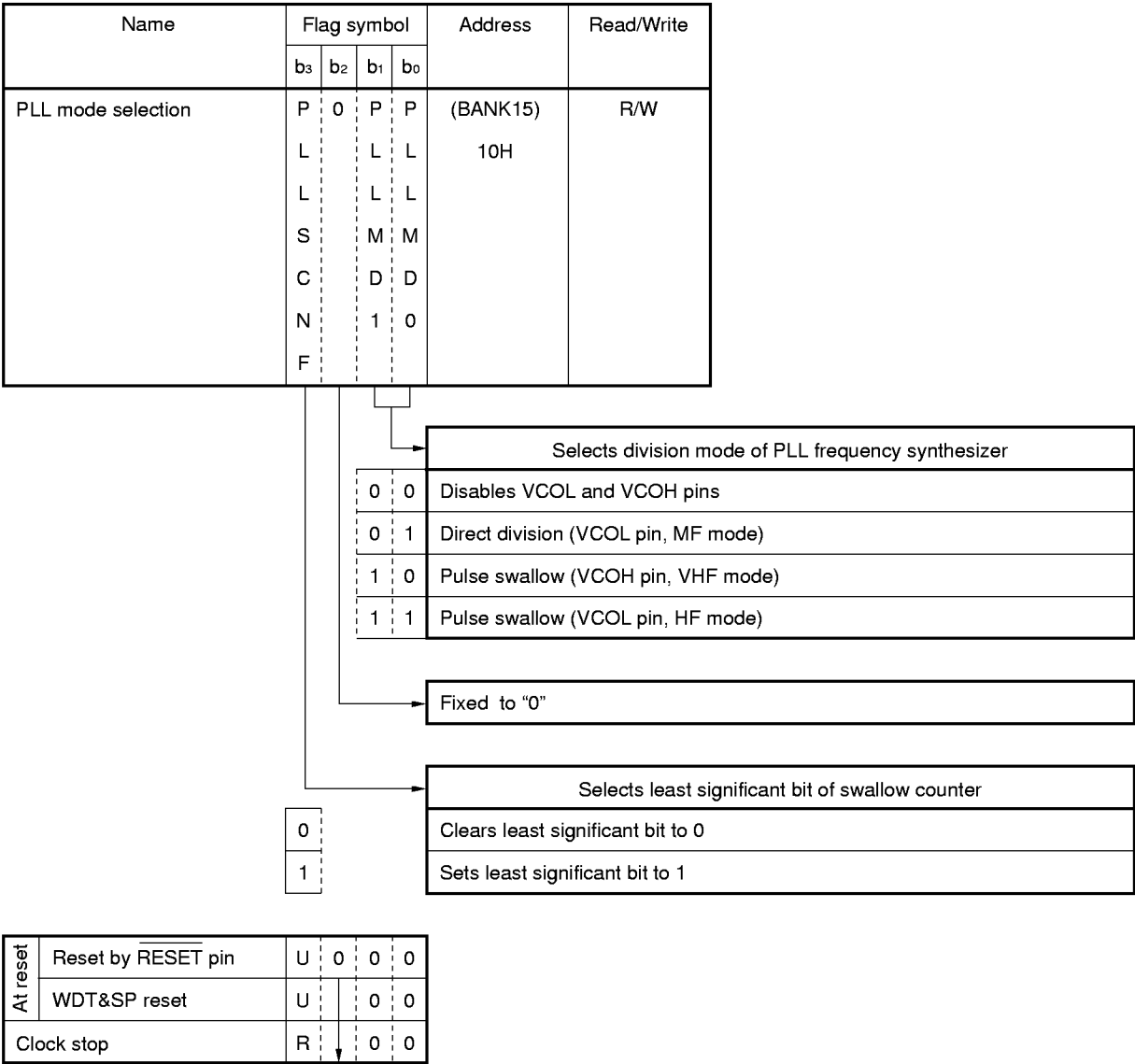
The value of the programmable divider is set by using the PLL data register via data buffer.

Figure 16-2. Configuration of Input Selection Block and Programmable Divider



Note PLLSCNF flag

Figure 16-3. Configuration of PLL Mode Selection Register



U: Undefined R: Retained

16.2.2 Outline of each division mode

(1) Direct division mode (MF)

In this mode, the VCOL pin is used.
The VCOH pin is pulled down.
In this mode, only the programmable counter is used for frequency division.

(2) Pulse swallow mode (HF)

In this mode, the VCOL pin is used.
The VCOH pin is pulled down.
In this mode, the swallow counter and programmable counter are used for frequency division.

(3) Pulse swallow mode (VHF)

In this mode, the VCOH pin is used.

The VCOL pin is pulled down.

In this mode, the swallow counter and programmable counter are used for frequency division.

(4) VCOL and VCOH pin disabled

In this mode, only the VCOL and VCOH pins are internally pulled down, but the other blocks operate.

16.2.3 Programmable divider and PLL data register

The programmable divider consists of a 5-bit swallow counter and a 12-bit programmable counter. Each counter is a 17-bit binary down counter.

The programmable counter is allocated to the high-order 12 bits of the PLL data register, and the swallow counter is allocated to the low-order 4 bits. Data are set to these counters via data buffer.

The least significant bit of the swallow counter sets data to the PLLSCNF flag of the control register.

The value by which the input signal frequency is to be divided is called "N value".

For how to set a division value (N value) in each division mode, refer to **16.6 Using PLL Frequency Synthesizer**.

(1) PLL data register and data buffer

Figure 16-4 shows the relationships between the PLL data register and data buffer.

In the direct division mode, the high-order 12 bits of the PLL data register are valid, and all 17 bits of the register are valid in the pulse swallow mode.

In the direct division mode, all 12 bits are used as a programmable counter.

In the pulse swallow mode, the high-order 12 bits are used as a programmable counter, and the low-order 5 bits are used as a swallow counter.

(2) Relationship between division value N of programmable divider and divided output frequency

The relationship between the value "N" set to the PLL data register and the signal frequency " f_{IN} " divided and output by the programmable divider is as shown below.

For details, refer to **16.6 Using PLL Frequency Synthesizer**.

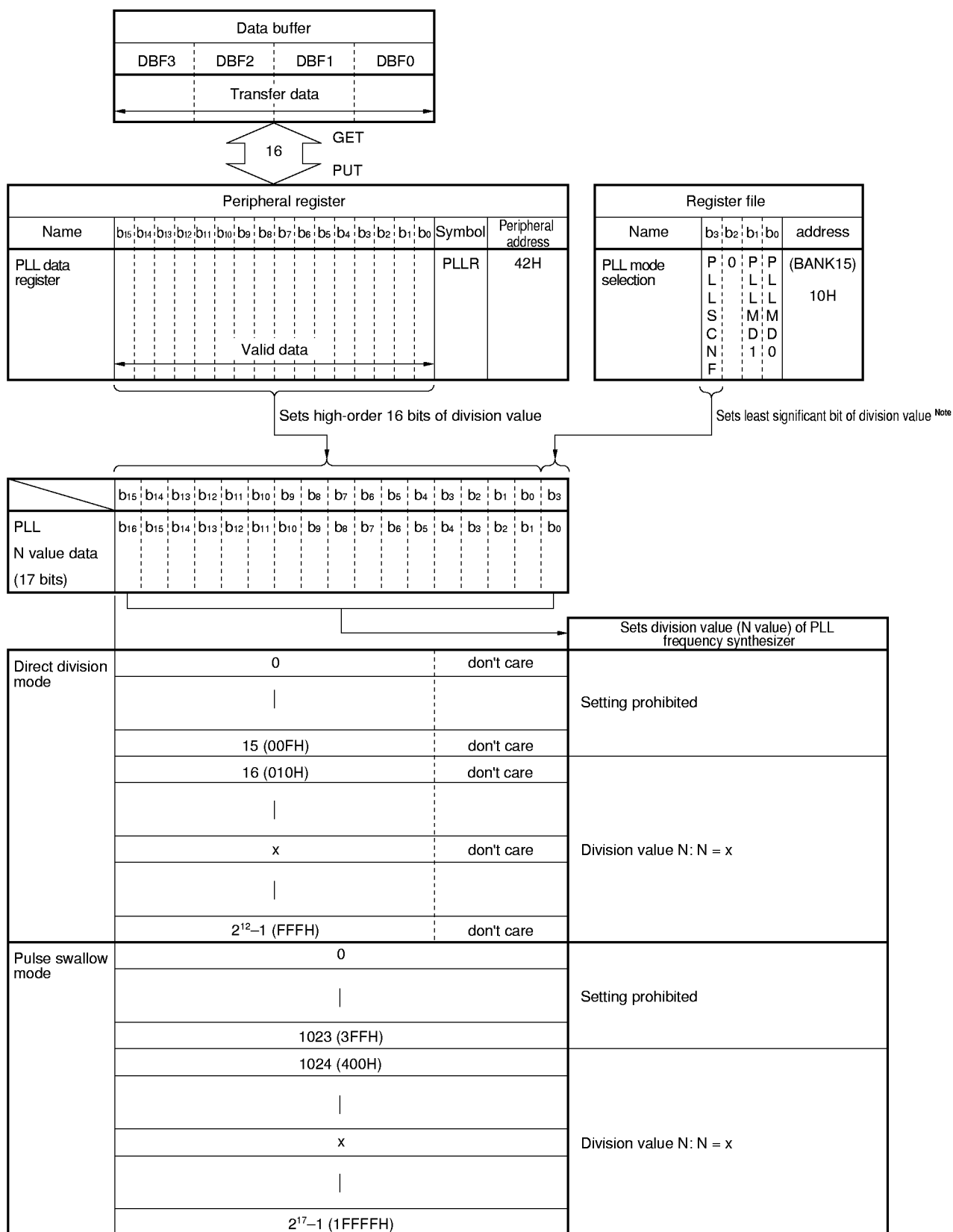
(a) Direct division mode (MF)

$$f_{IN} = \frac{f_{IN}}{N} \quad N: 12 \text{ bits}$$

(b) Pulse swallow mode (HF, VHF)

$$f_{IN} = \frac{f_{IN}}{N} \quad N: 17 \text{ bits}$$

Figure 16-4. Setting Division Value (N Value) of PLL Frequency Synthesizer



16.3 Reference Frequency Generator

Figure 16-5 shows the configuration of the reference frequency generator.

The reference frequency generator generates the reference frequency “fr” of the PLL frequency synthesizer by dividing the 75 kHz output of a crystal oscillator.

Six frequencies can be selected as reference frequency fr: 1, 3, 5, 6.25, 12.5, and 25 kHz.

The reference frequency fr is selected by the PLL reference frequency selection register.

Figure 16-6 shows the configuration and function of the PLL reference frequency selection register.

Figure 16-5. Configuration of Reference Frequency Generator

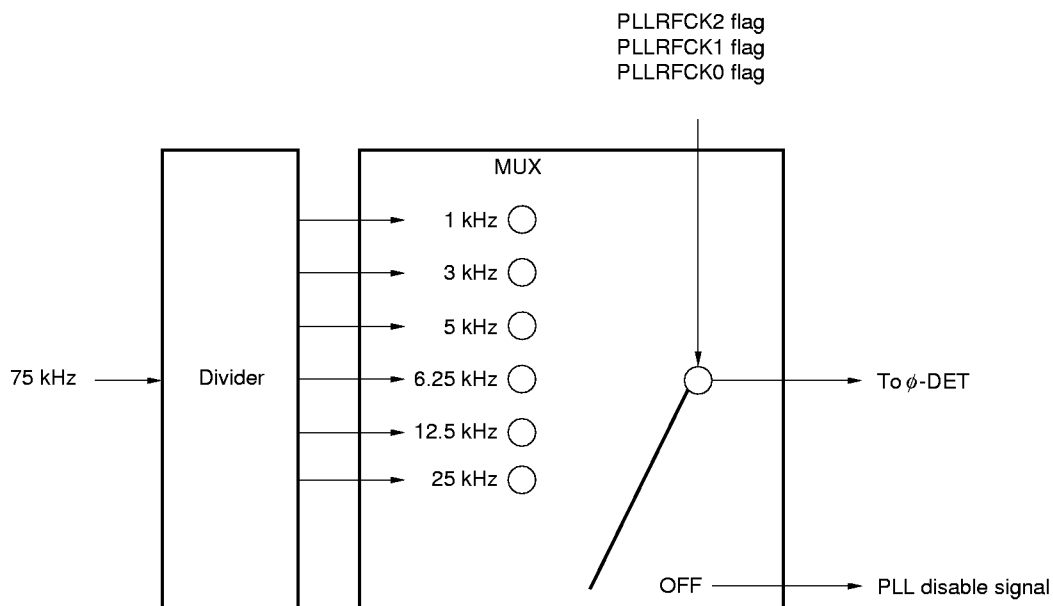
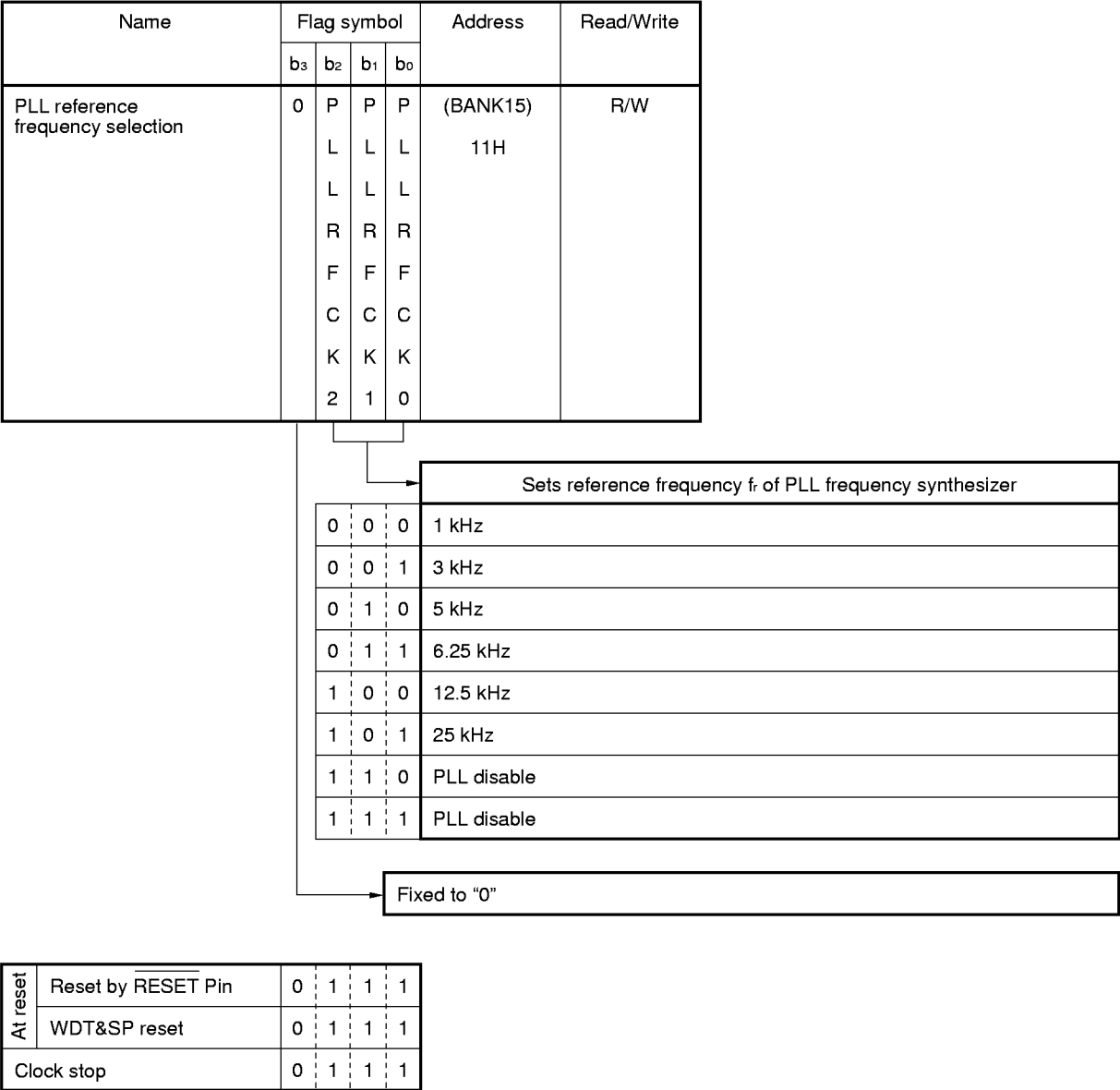


Figure 16-6. Configuration of PLL Reference Frequency Selection Register



Remark When the PLL frequency synthesizer is disabled by the PLL reference frequency selection register, the VCOH and VCOL pins are internally pulled down. The EO1 and EO0 pins are floated.

16.4 Phase Comparator (ϕ -DET), Charge Pump, and Unlock FF

16.4.1 Configuration of phase comparator, charge pump, and unlock FF

Figure 16-7 shows the configuration of the phase comparator, charge pump, and unlock FF.

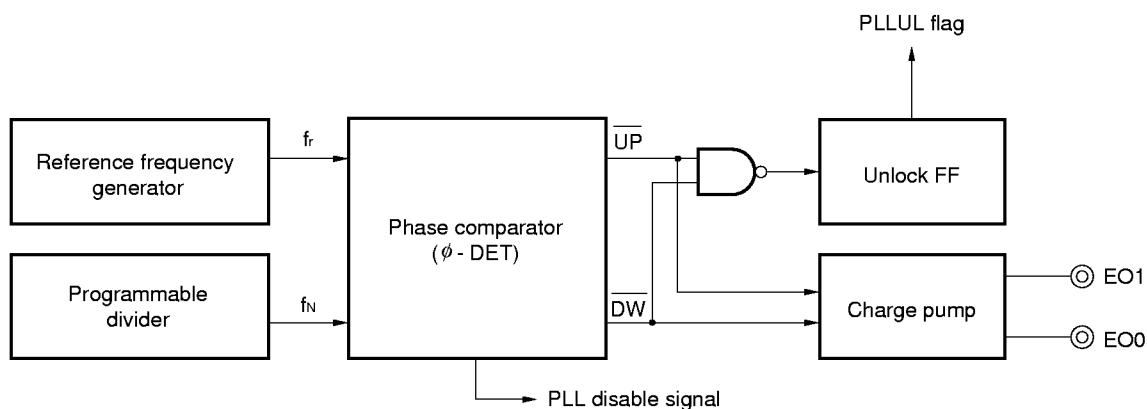
The phase comparator compares the phase of the divided frequency " f_N " output by the programmable divider with the phase of the reference frequency " f_r " output by the reference frequency generator, and outputs an up (\overline{UP}) or down (\overline{DW}) request signal.

The charge pump outputs the output of the phase comparator from an error out pin (EO1 and EO0 pins).

The unlock FF detects the unlock status of the PLL frequency synthesizer.

16.4.2 through 16.4.4 describe the operations of the phase comparator, charge pump, and unlock FF.

Figure 16-7. Configuration of Phase Comparator, Charge Pump, and Unlock FF



16.4.2 Function of phase comparator

As shown in Figure 16-7, the phase comparator compares the phases of the divided frequency " f_N " output by the programmable divider and the reference frequency " f_r ", and outputs an up or down request signal.

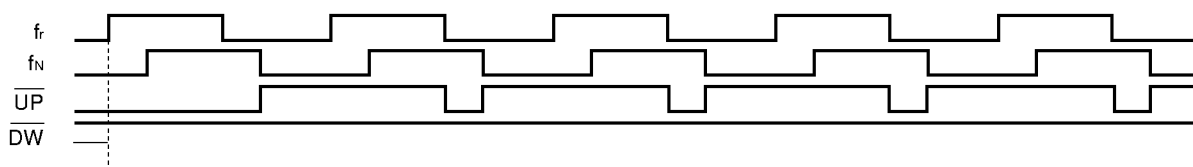
If the divided frequency f_N is lower than reference frequency f_r , the up request signal is output. If f_N is higher than f_r , the down request signal is output.

Figure 16-8 shows the relationship between reference frequency f_r , divided frequency f_N , up request signal, and down request signal.

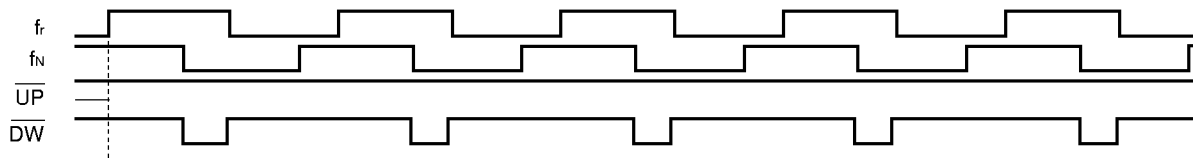
When the PLL frequency synthesizer is disabled, neither the up request nor the down request signal is output. The up and down request signals are input to the charge pump and unlock FF, respectively.

Figure 16-8. Relationship between f_r , f_N , \overline{UP} , and \overline{DW}

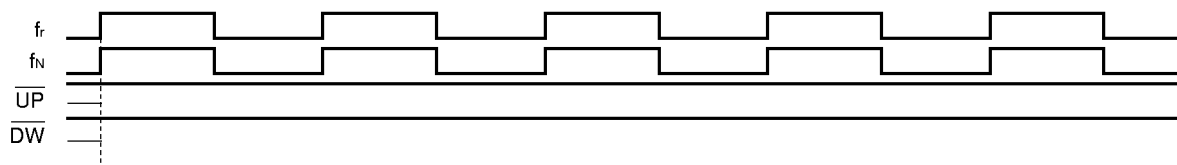
(a) If f_N lags behind f_r



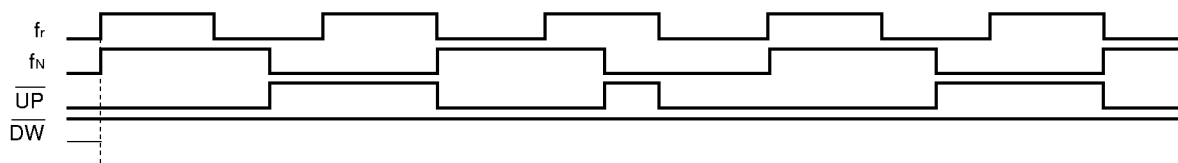
(b) If f_N leads f_r



(c) If f_N and f_r are in phase



(d) If f_N is lower than f_r



16.4.3 Charge pump

As shown in Figure 16-7, the charge pump outputs the up request and down request signals output by the phase comparator, from the error out pins (EO1 and EO0 pins).

Therefore, the relationship between the output of the error out pins, divided frequency f_N and reference frequency f_r is as follows:

Where reference frequency $f_r >$ divided frequency f_N : Low-level output

Where reference frequency $f_r <$ divided frequency f_N : High-level output

Where reference frequency $f_r =$ divided frequency f_N : Floating

16.4.4 Unlock FF

As shown in Figure 16-7, the unlock FF detects the unlock status of the PLL frequency synthesizer from the up request and down request signals of the phase comparator.

Because either the up request or down request signal is low in the unlock status, the unlock status is detected by this low-level signal.

In the unlock status, the unlock FF is set to 1.

The unlock FF is set in the cycle of the reference frequency f_r selected at that time. When the contents of the PLL unlock FF register are read (by the PEEK instruction), the unlock FF is reset (Read & Reset).

Therefore, the unlock FF must be detected in a cycle longer than cycle $1/f_r$ of the reference frequency f_r .

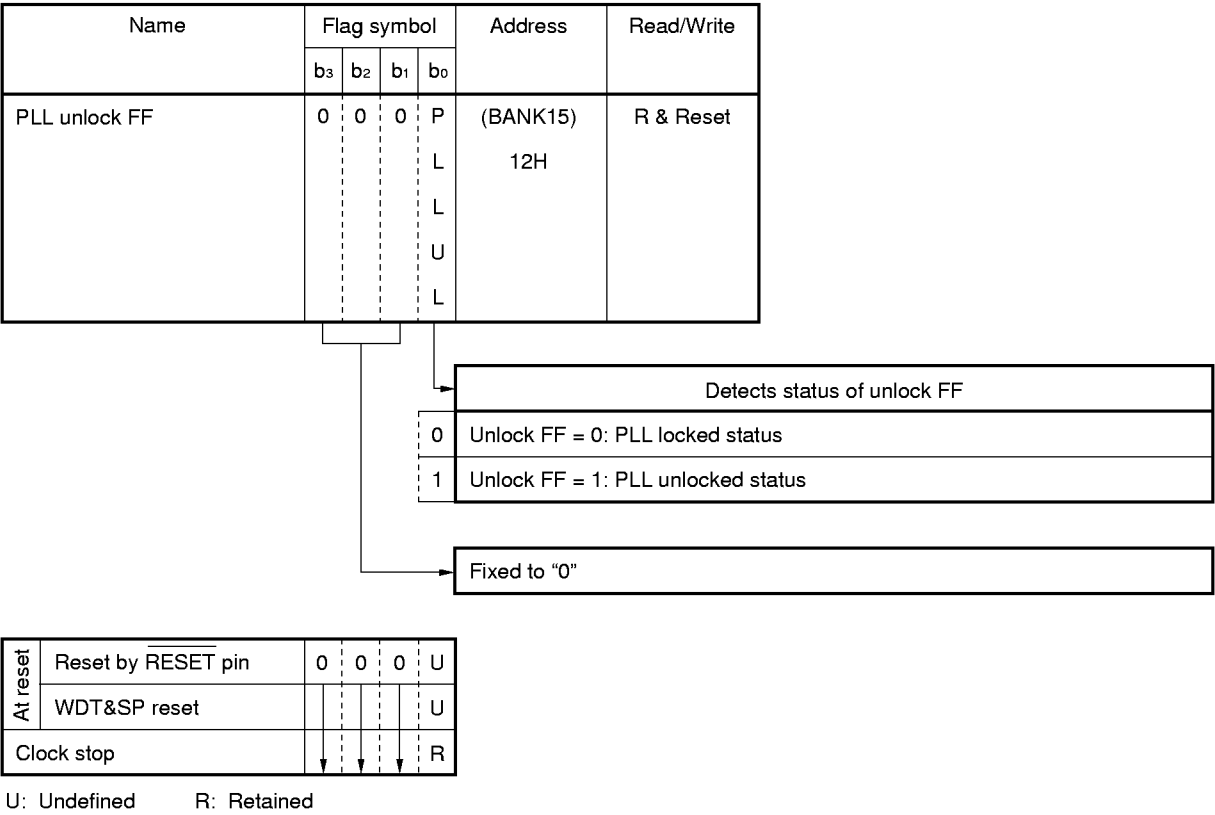
The status of the unlock FF is detected by the PLL unlock FF register. Figure 16-9 shows the configuration of the PLL unlock FF register.

Because this register is a read-only register, its contents can be read to the window register by the "PEEK" instruction.

Because the unlock FF is set in a cycle of the reference frequency f_r , the contents of the PLL unlock FF register are read to the window register in a cycle longer than cycle $1/f_r$ of the reference frequency.

The delay time of the up and down request signals of the phase comparator are fixed to 0.8 to 1.0 μ s.

Figure 16-9. Configuration of PLL Unlock FF Register



16.5 PLL Disabled Status

The PLL frequency synthesizer stops when PLL disabled status is selected by the PLL reference frequency register (RF address 11H).

Table 16-1 shows the operation of each block in the PLL disabled status.

When the VCOL and VCOH pins are disabled by the PLL mode selection register, only the VCOL and VCOH pins are internally pulled down, and the other blocks operate.

At reset by $\overline{\text{RESET}}$ pin, the PLL frequency synthesizer is disabled.

Table 16-1. Operation of Each Block under Each PLL Disable Condition

Condition Each Block	PLL reference frequency selection register = 1111B (PLL disabled)	PLL mode selection register = 0000B (VCOH and VCOL disabled)
VCOL, VCOH pins	Internally pulled down	Internally pulled down
Programmable divider	Division stopped	Operates
Reference frequency generator	Output stopped	Operates
Phase comparator	Output stopped	Operates
Charge pump	Error out pins are floated	Operates. However, usually outputs low level because no signal is input

16.6 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) Pins used : VCOL and VCOH pins
- (3) Reference frequency : fr
- (4) Division value : N

16.6.1 through 16.6.3 below describe how to set PLL data in each division mode (MF, HF, and VHF).

16.6.1 Direct division mode (MF)

(1) Selecting division mode

Select the direct division mode by using the PLL mode selection register.

(2) Pins used

The VCOL pin is enabled to operate when the direct division mode is selected.

(3) Selecting reference frequency fr

Select the reference frequency by using the PLL reference frequency selection register.

(4) Calculation of division value N

Calculate N as follows:

$$N = \frac{f_{\text{VCOL}}}{f_r}$$

f_{VCOL} : Input frequency of VCOL pin

f_r : Reference frequency

(5) Example of setting PLL data

How to set data to receive broadcasting in the following MW band is described below.

Reception frequency : 1422 kHz (MW band)

Reference frequency : 3 kHz

Intermediate frequency : +450 kHz

Division value N is calculated as follows:

$$N = \frac{f_{\text{VCOL}}}{f_r} = \frac{1422 + 450}{3} = 624 \text{ (decimal)} \\ = 270\text{H (hexadecimal)}$$

Set data to the PLL data register, PLL mode selection register, and PLL reference frequency selection register as follows:

PLL data register (PLLR)												
0	0	1	0	0	1	1	1	0	0	0	0	don't care
2				7				0				

	PLL mode selection register			PLL reference frequency selection register			
Note 1							
Note 2	0	0	1	0	0	0	1
	MF			3 kHz			

- Notes 1. PLLSCNF flag
2. don't care

16.6.2 Pulse swallow mode (HF)

(1) Selecting division mode

Select the pulse swallow mode by using the PLL mode selection register.

(2) Pins used

The VCOL pin is enabled to operate when the pulse swallow mode is selected.

(3) Selecting reference frequency fr

Select the reference frequency by using the PLL reference frequency selection register.

(4) Calculation of division value N

Calculate N as follows:

$$N = \frac{f_{\text{VCOL}}}{f_r}$$

f_{VCOL} : Input frequency of VCOL pin
 f_r : Reference frequency

(5) Example of setting PLL data

How to set data to receive broadcasting in the following SW band is described below.

Reception frequency : 25.50 MHz (SW band)

Reference frequency : 5 kHz

Intermediate frequency: +450 kHz

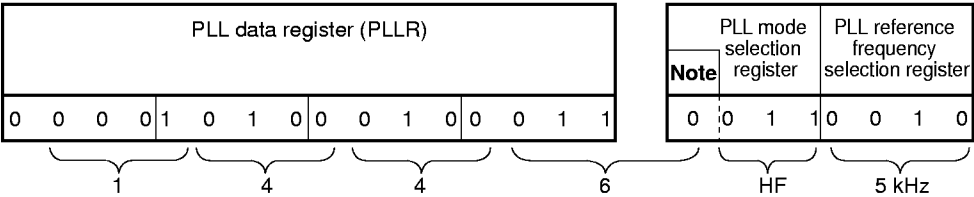
Division value N is calculated as follows:

$$N = \frac{f_{\text{VCOL}}}{f_r} = \frac{25500 + 450}{5} = 5190 \text{ (decimal)}$$

$$= 1446\text{H} \text{ (hexadecimal)}$$

Set data to the PLL data register, PLL mode selection register, and PLL reference frequency selection register as follows:

Caution The division value N is 17 bits long when the pulse swallow mode is selected, and the least significant bit of the swallow counter is the bit 3 of the PLL mode selection register (PLLSCNF). To set “1446H” as the division value N, the value to be actually set to the PLL data register is “0A23H”.



Note PLLSCNF flag

16.6.3 Pulse swallow mode (VHF)

(1) Selecting division mode

Select the pulse swallow mode by using the PLL mode selection register.

(2) Pins used

The VCOH pin is enabled to operate when the pulse swallow mode is selected.

(3) Selecting reference frequency fr

Select the reference frequency by using the PLL reference frequency selection register.

(4) Calculation of division value N

Calculate N as follows:

$$N = \frac{f_{VCOH}}{fr}$$

f_{VCOH} : Input frequency of VCOH pin
 fr : Reference frequency

(5) Example of setting PLL data

How to set data to receive broadcasting in the following FM band is described below.

- Reception frequency : 98.15 MHz (FM band)
- Reference frequency : 25 kHz
- Intermediate frequency : +10.7 MHz

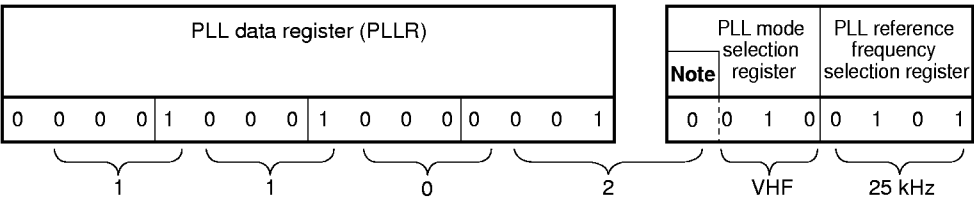
Division value N is calculated as follows:

$$N = \frac{f_{VCOH}}{fr} = \frac{98.15 + 10.7}{0.025} = 4354 \text{ (decimal)}$$

$$= 1102H \text{ (hexadecimal)}$$

Set data to the PLL data register, PLL mode selection register, and PLL reference frequency selection register as follows:

Caution The division value N is 17 bits long when the pulse swallow mode is selected, and the least significant bit of the swallow counter is the bit 3 of the PLL mode selection register (PLLSCNF). To set “1102H” as the division value N, the value to be actually set to the PLL data register is “0881H”.



Note PLLSCNF flag

Note that data must be set to the PLLSCNF flag before a write (PUT) instruction is executed to the PLL data register (PLLR).

Example

SET1	PLLSCNF
MOV	DBF0, #0
MOV	DBF1, #4
MOV	DBF2, #4
PUT	PLLR, DBF

16.7 Status at Reset

16.7.1 At reset by $\overline{\text{RESET}}$ pin

The PLL frequency synthesizer is disabled because the PLL reference frequency selection register is initialized to 0111B.

16.7.2 At WDT&SP reset

The PLL frequency synthesizer is disabled because the PLL reference frequency selection register is initialized to 0111B.

16.7.3 On execution of clock stop instruction

The PLL frequency synthesizer is disabled because the PLL reference frequency selection register is initialized to 0111B.

16.7.4 In halt status

The set status is retained.

17. INTERMEDIATE FREQUENCY (IF) COUNTER

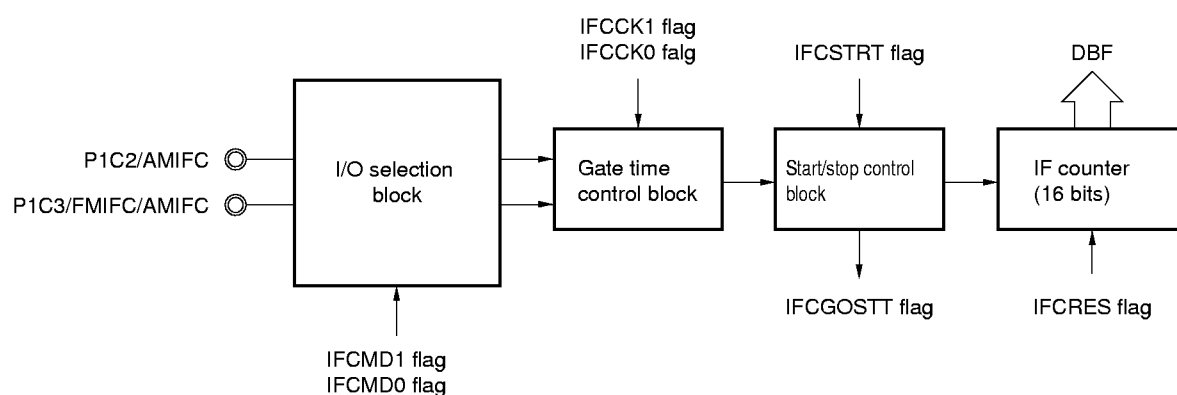
★ 17.1 Outline of Frequency Counter

Figure 17-1 outlines the frequency counter.

The IF counter is mainly used to count the intermediate frequency (IF) output from a tuner for detecting broadcasting stations.

The IF counter counts the frequency input to the P1C2/AMIFC or P1C3/FMIFC/AMIFC pin at fixed intervals (1 ms, 4 ms, 8 ms, or open) by using a 16-bit counter.

★ **Figure 17-1. Outline of IF Counter**



- Remarks**
1. IFCMD1 and IFCMD0 (bits 3 and 2 of IF counter mode selection register: refer to **Figure 17-3**) select the IF counter function.
 2. IFCK1 and IFCK0 (bits 1 and 0 of IF counter mode selection register: refer to **Figure 17-3**) select the gate time of the IF counter.
 3. IFCSTRT (bit 1 of IF counter control register: refer to **Figure 17-5**) control starting of the IF counter.
 4. IFCGOSTT (bit 0 of IF counter gate status detection register: refer to **Figure 17-6**) detects opening/closing the gate of the IF counter function.
 5. IFCRES (bit 0 of IF counter control register: refer to **Figure 17-5**) reset the count value of the IF counter.

★ 17.2 IF Counter Input Selection Block and Gate Time Control Block

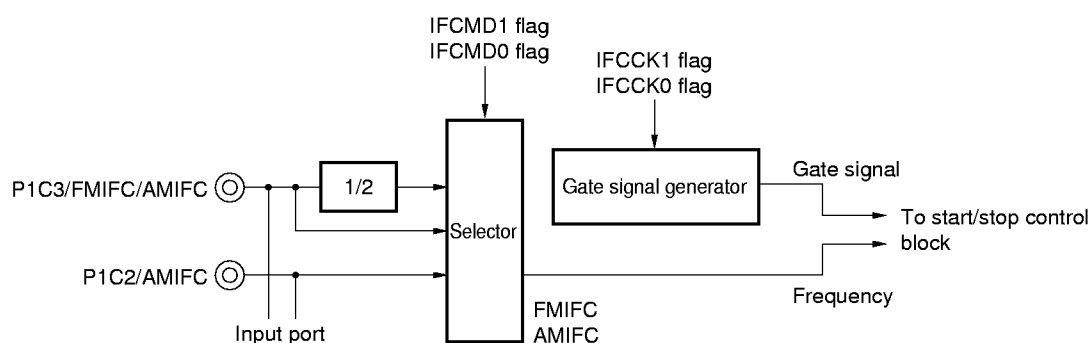
Figure 17-2 shows the configuration of the IF counter input selection block and gate time control block.

The IF counter selection block selects whether the P1C2/AMIFC or P1C3/FMIFC/AMIFC pin is used as an IF counter function or a general-purpose input port, by using the IF counter mode selection register.

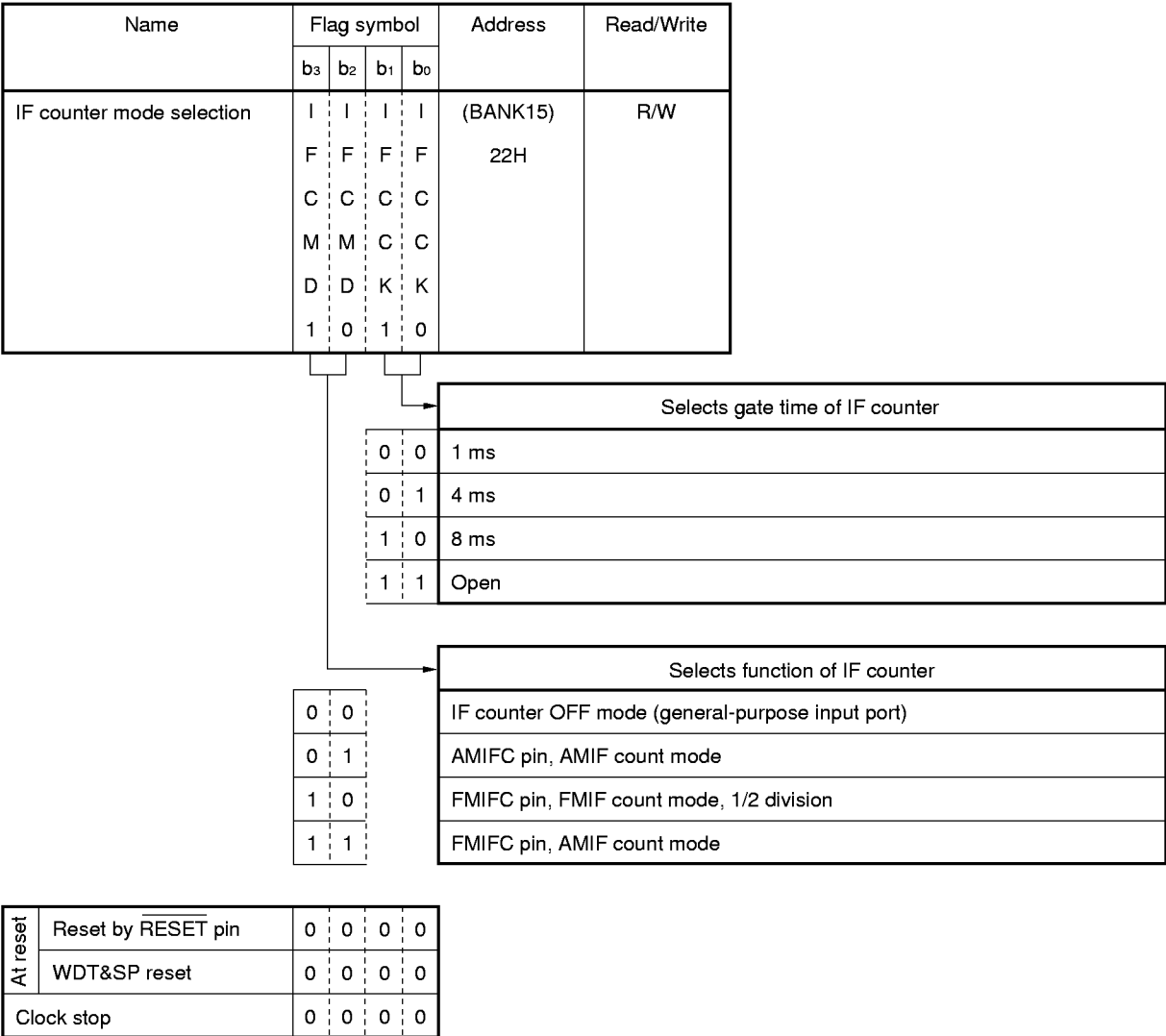
The gate time control block selects gate time by using the IF counter mode selection register when the frequency counter is used as the IF counter.

Figure 17-3 shows the configuration of the IF counter mode selection register.

★ Figure 17-2. Configuration of IF Counter Input Selection Block and Gate Time Control Block



★ Figure 17-3. Configuration of IF Counter Mode Selection Register



17.3 Start/Stop Control Block and IF Counter

★ 17.3.1 Configuration of start/stop control block and IF counter

Figure 17-4 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 status detection register.

Figure 17-6 shows the configuration of the IF counter control register.

Figure 17-7 shows the configuration of the IF counter gate status detection register.

17.3.2 describes the gate operation when the IF counter function is selected.

The IF counter is a 16-bit binary counter that counts up the input frequency when the IF 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 IF counter counts up to FFFFH, it remains at FFFFH until reset.

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 17-7 shows the configuration of the IF counter data register.

Figure 17-4. Configuration of Start/Stop Control Block and IF Counter

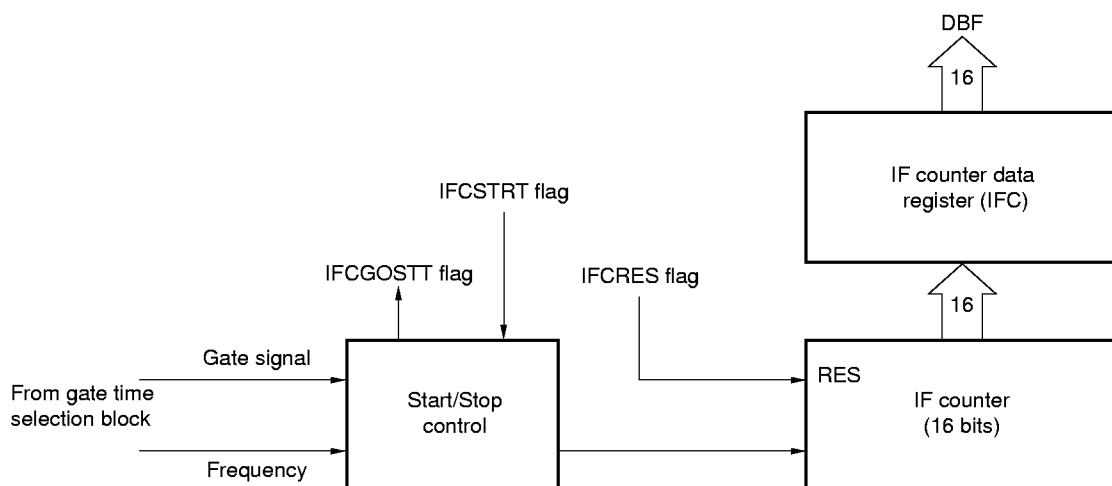
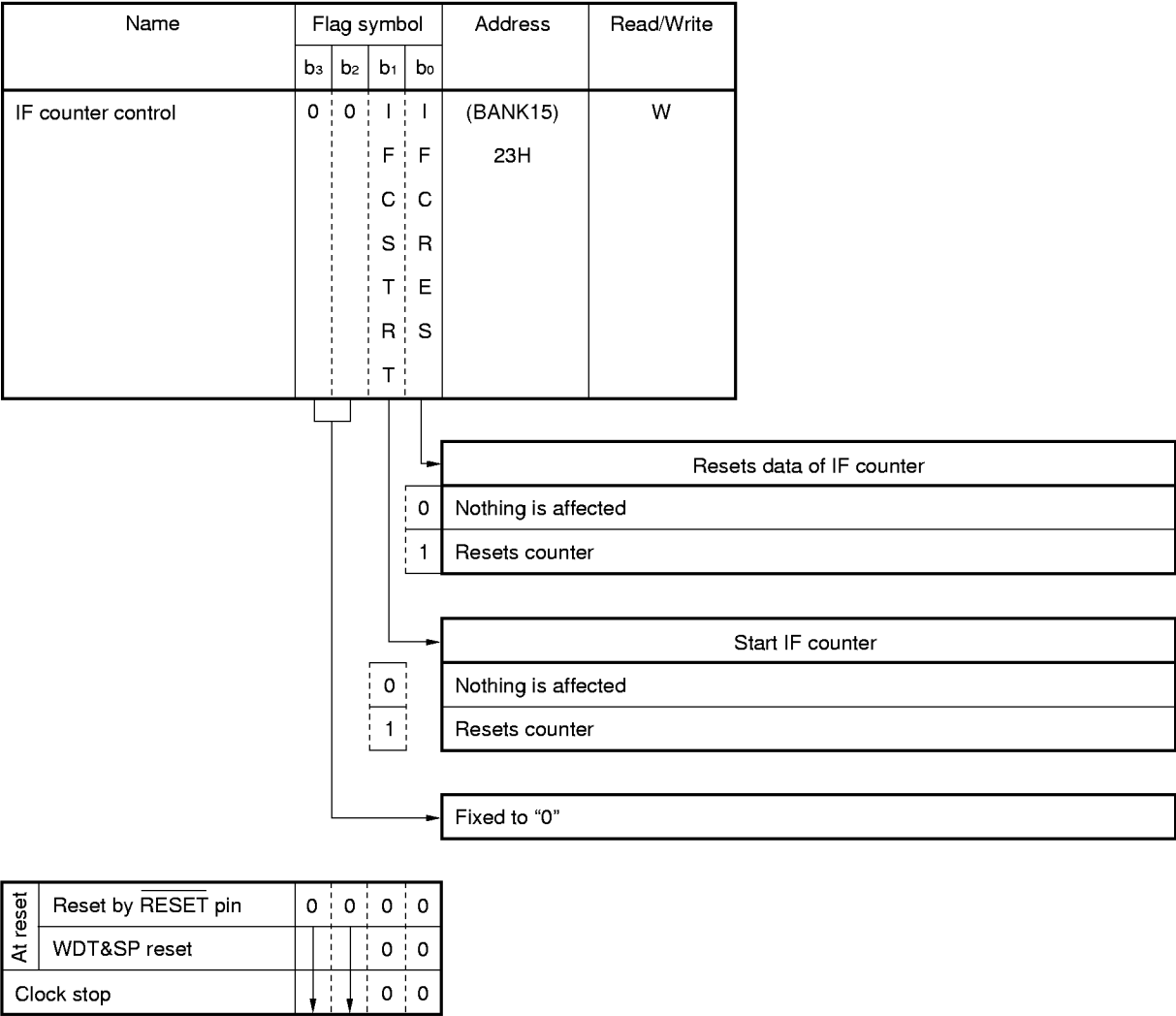
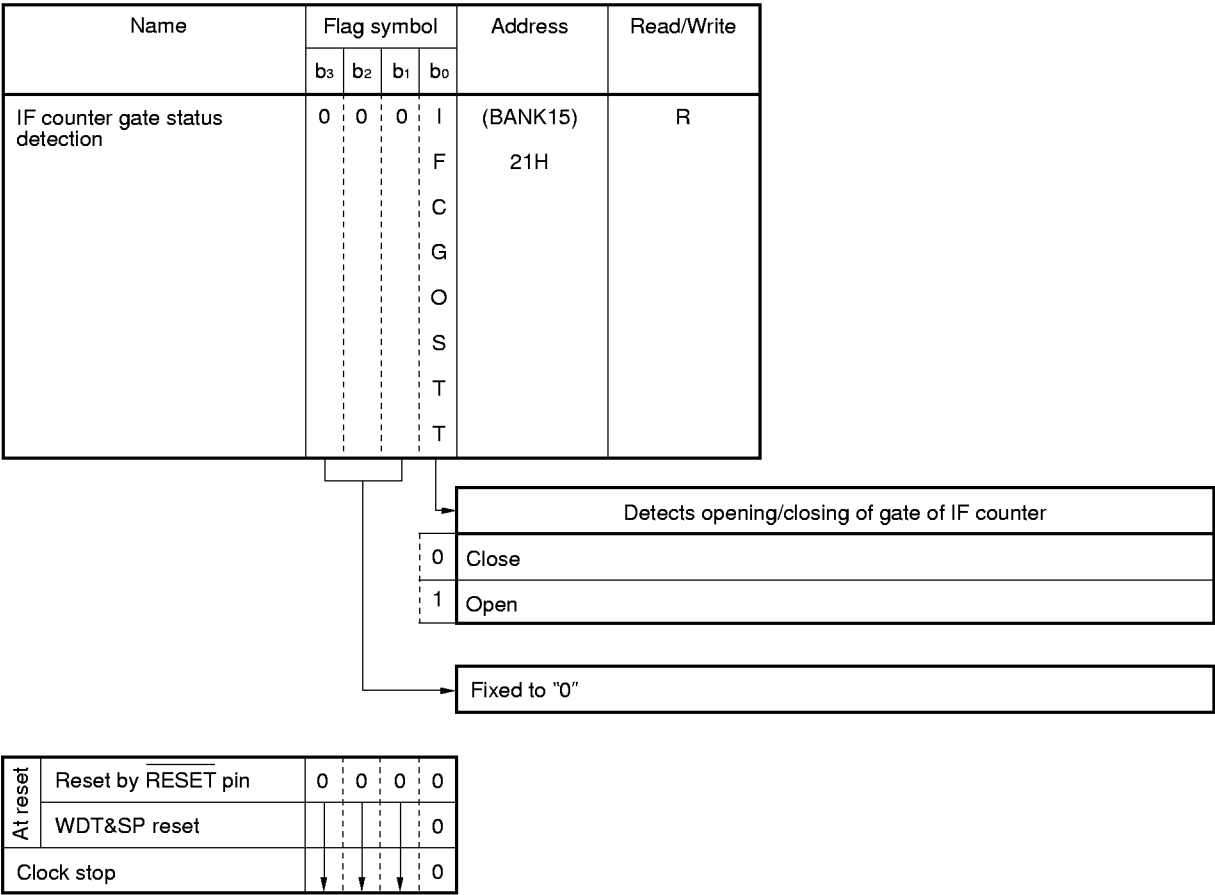


Figure 17-5. Configuration of IF Counter Control Register



★ Figure 17-6. Configuration of IF Counter Gate Status Detection Register



Caution Do not read the contents of the IF counter data register (IFC) to the data buffer while the IFCGOSTT flag is set to 1.

17.3.2 Operation of gate when IF counter function is selected

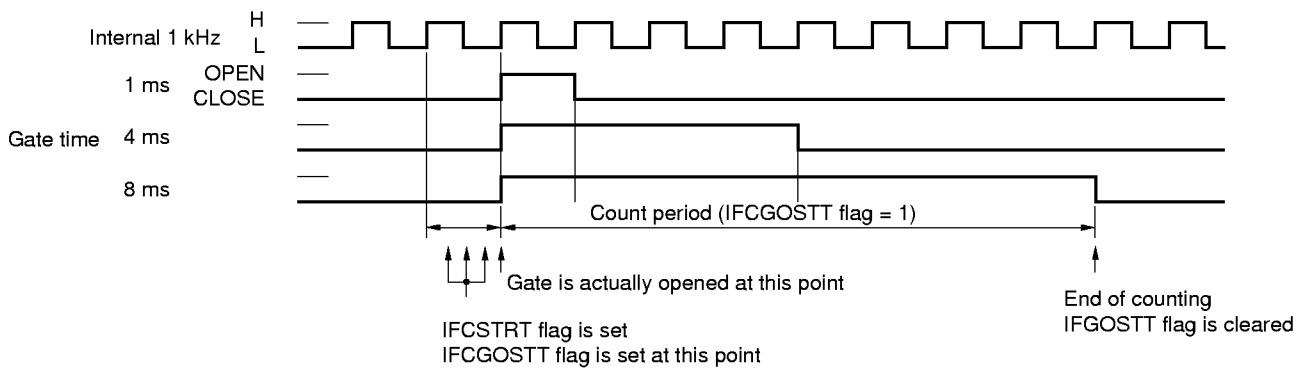
(1) When gate time of 1, 4, or 8 ms is selected

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 IFCGOSTT flag is cleared to 0.

The IFCGOSTT flag is automatically set to 1 when the IFCSTRT flag is set.



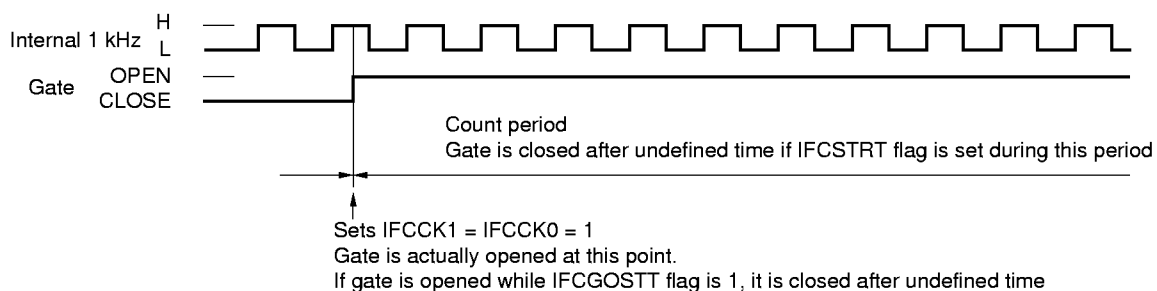
(2) When gate is open

If opening of the gate is selected by the IFCCK1 and IFCCK0 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 undefined time.

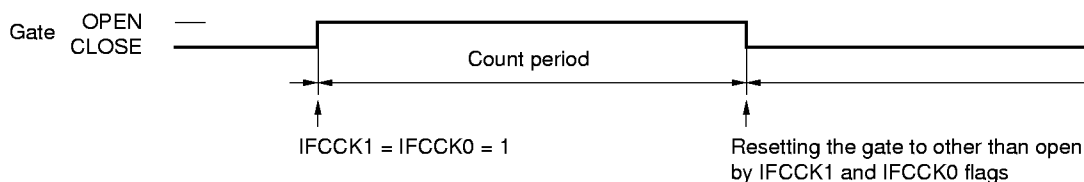
To open the gate, therefore, do not set the IFCSTRT flag to 1.

However, the counter can be reset by the IFCRES flag.



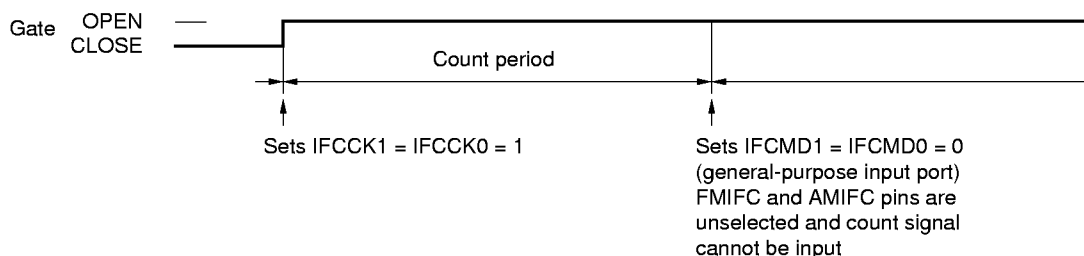
The gate is opened or closed in the following two ways when opening the gate is selected as the gate time.

(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 Function and operation of 16-bit counter

The 16-bit counter counts up the frequency input within selected gate time.

The 16-bit counter can be reset by writing "1" to the IFCRES flag of the IF counter control register.

Once the 16-bit counter has counted up to FFFFH, it remains at FFFFH until it is reset.

The IF counter counts the frequency input to the P1C2/AMIFC or P1C3/FMIFC/AMIFC pin while the gate is open.

Note, however, in the FMIF count mode input to the P1C3/FMIFC/AMIFC is divided by two and counted.

The relationship between count value "x (decimal)" and input frequencies (f_{FMIFC} and f_{AMIFC}) is shown below.

• FMIFC

$$f_{FMIFC} = \frac{x}{t_{GATE}} \times 2 \text{ (kHz)} \quad t_{GATE}: \text{gate time (1 ms, 4 ms, 8 ms)}$$

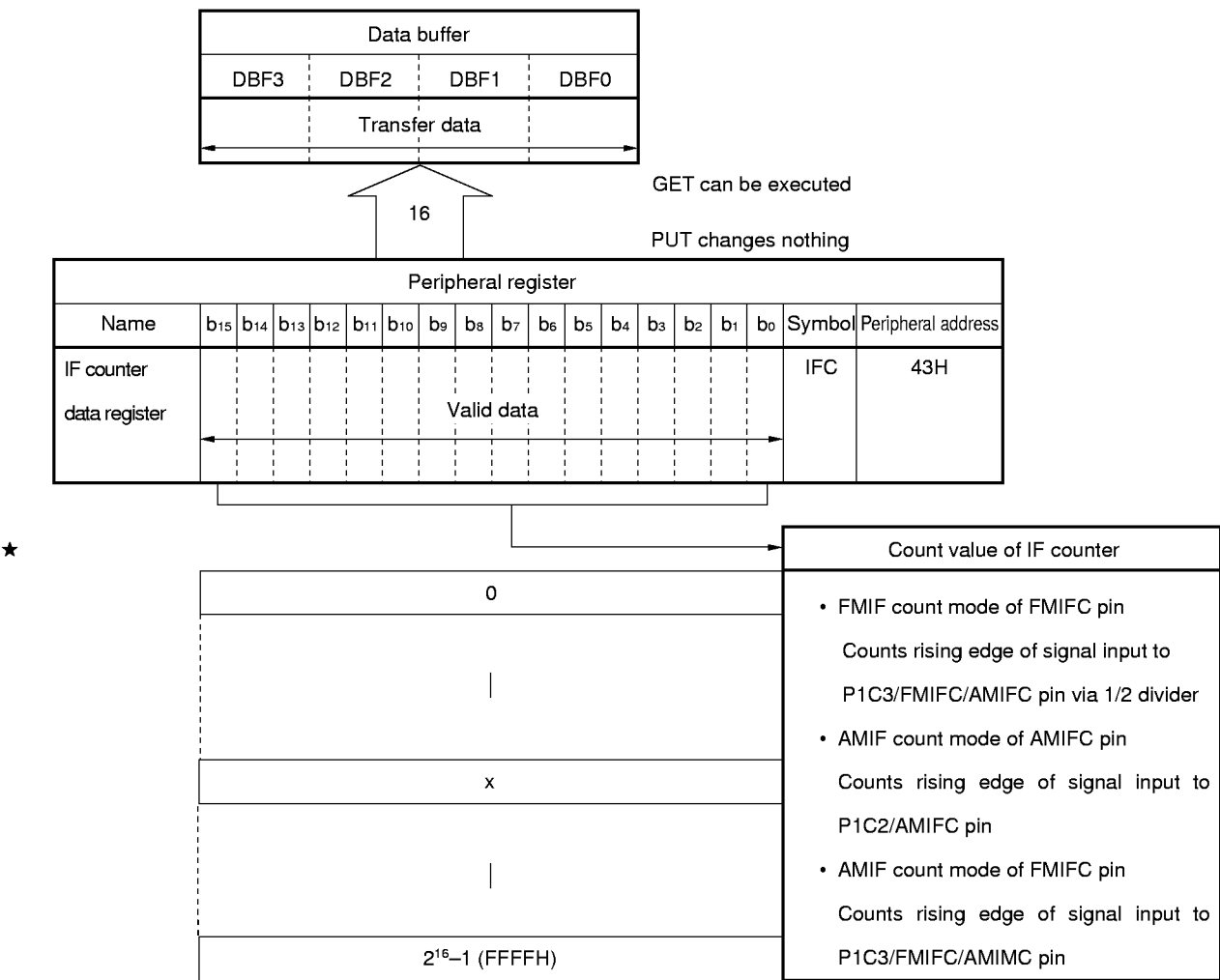
• AMIFC

$$f_{AMIFC} = \frac{x}{t_{GATE}} \text{ (kHz)} \quad t_{GATE}: \text{gate time (1 ms, 4 ms, 8 ms)}$$

The value of the IF counter data register is read via data buffer.

Figure 17-7 shows the configuration and function of the IF counter data register.

Figure 17-7. Configuration of IF Counter Data Register



Once the IF counter data register has counted up to FFFFH, it remains at FFFFH until the counter is reset.

17.4 Using IF Counter

The following sections 17.4.1 through 17.4.3 describe how to use the hardware of the IF counter, a program example, and count error.

★ 17.4.1 Using hardware of IF counter

Figure 17-8 shows the block diagram when the P1C2/AMIFC and P1C3/FMIFC/AMIFC pins.

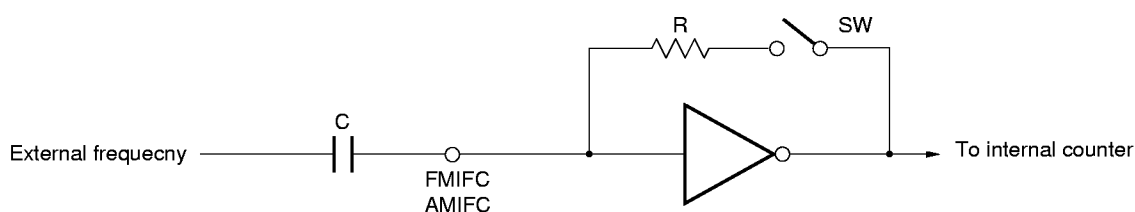
As shown in the figure, the IF counter uses an input pin with an AC amplifier, the DC component of the input signal must be cut with a capacitor.

When the P1C2/AMIFC or P1C3/FMIFC/AMIFC pin is selected for the IF counter function, switch SW turns ON, and the voltage level on each pin reaches about $1/2V_{DD}$.

If the voltage has not risen to a sufficient intermediate level at this time, the IF counter does not operate normally because the AC amplifier is not in the normal operating range.

Therefore, make sure that a sufficient wait time elapses after each pin has been specified to be used for the IF counter until counting is started.

Figure 17-8. IF Count Function Block Diagram of Each Pin



★ 17.4.2 Program example of IF counter

A program example of the IF counter is shown below.

As shown in this example, make sure that a wait time elapses after an instruction that selects the P1C2/AMIFC or P1C3/FMIFC/AMIFC pin for the IF counter function has been executed until counting is started.

This is because, as described in 17.4.1, the internal AC amplifier does not operate normally immediately after a pin has been selected for the IF counter.

Example To count the frequency input to the P1C3/FMIFC pin (FMIF count mode) (gate time: 8 ms)

```

INITFLG IFCMD1, NOT IFCMD0, IFCCK1, NOT IFCCK0
                                ; Selects FMIFC pin (FMIF count mode), and sets gate time to 8 ms

                                ; Internal AC amplifier stabilization time
                                Wait
                                ; Internal AC amplifier stabilization time

SET1  IFCRES                    ; Resets counter
SET1  IFCSTRT                   ; Starts counting
LOOP:
SKT1  IFCG0STT                  ; Detects opening or closing of gate
BR    READ                      ; Branches to READ: if gate is closed

                                Processing A

BR    LOOP                      ; Do not read data of IF counter with this processing A
READ:
GET   DBF, IFC                  ; Reads value of IF counter data register to data buffer

```

17.4.3 Error of IF counter

The errors of the IF counter include a gate time error and a count error. The following paragraphs (1) and (2) describe each of these errors.

(1) Gate time error

The gate time of the IF counter is created by dividing the 75-kHz clock. Therefore, if the system clock is shifted from 75 kHz by “+x” ppm, the gate time is shifted by “-x” ppm.

(2) Count error

The IF counter counts frequency by the rising edge of the input signal.

If a high level is input to the pin when the gate is open, therefore, one excess pulse is counted.

If the gate is closed, however, a count error due to the status of the pin does not occur.

Therefore, the count error is “+1, -0”.

★ **17.5 Status at Reset**

17.5.1 At reset by RESET pin

The P1C2/AMIFC and P1C3/FMIFC/AMIFC pins are set in the general-purpose input port mode.

17.5.2 At WDT&SP reset

The P1C2/AMIFC and P1C3/FMIFC/AMIFC pins are set in the general-purpose input port mode.

17.5.3 On execution of clock stop instruction

The P1C2/AMIFC and P1C3/FMIFC/AMIFC pins are set in the general-purpose input port mode.

17.5.4 In halt status

The P1C2/AMIFC and P1C3/FMIFC/AMIFC pins retain the status immediately before the halt mode is set.

18. BEEP

18.1 Outlines of BEEP

Figure 18-1 outlines BEEP.

BEEP outputs a clock of 1.5 kHz or 3 kHz from the BEEP pin.

The output select block selects, by using the BEEP0SEL flag, whether the P0B3/BEEP pin is used as a general-purpose I/O port pin or BEEP output pin. The BEEP0CK0 and BEEP0CK1 flags of the BEEP clock select register are used to select whether the output frequency of the P0B3/BEEP pin is 1.5 kHz or 3 kHz, or the output level of the BEEP pin. The clock generation block generates the 1.5-kHz or 3-kHz clock to be output to the P0B3/BEEP pin.

Figure 18-2 shows the configuration and function of the BEEP clock select register.

Figure 18-1. Outline of BEEP

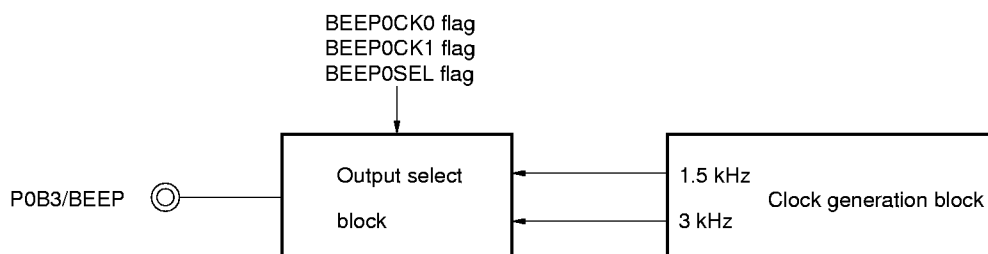
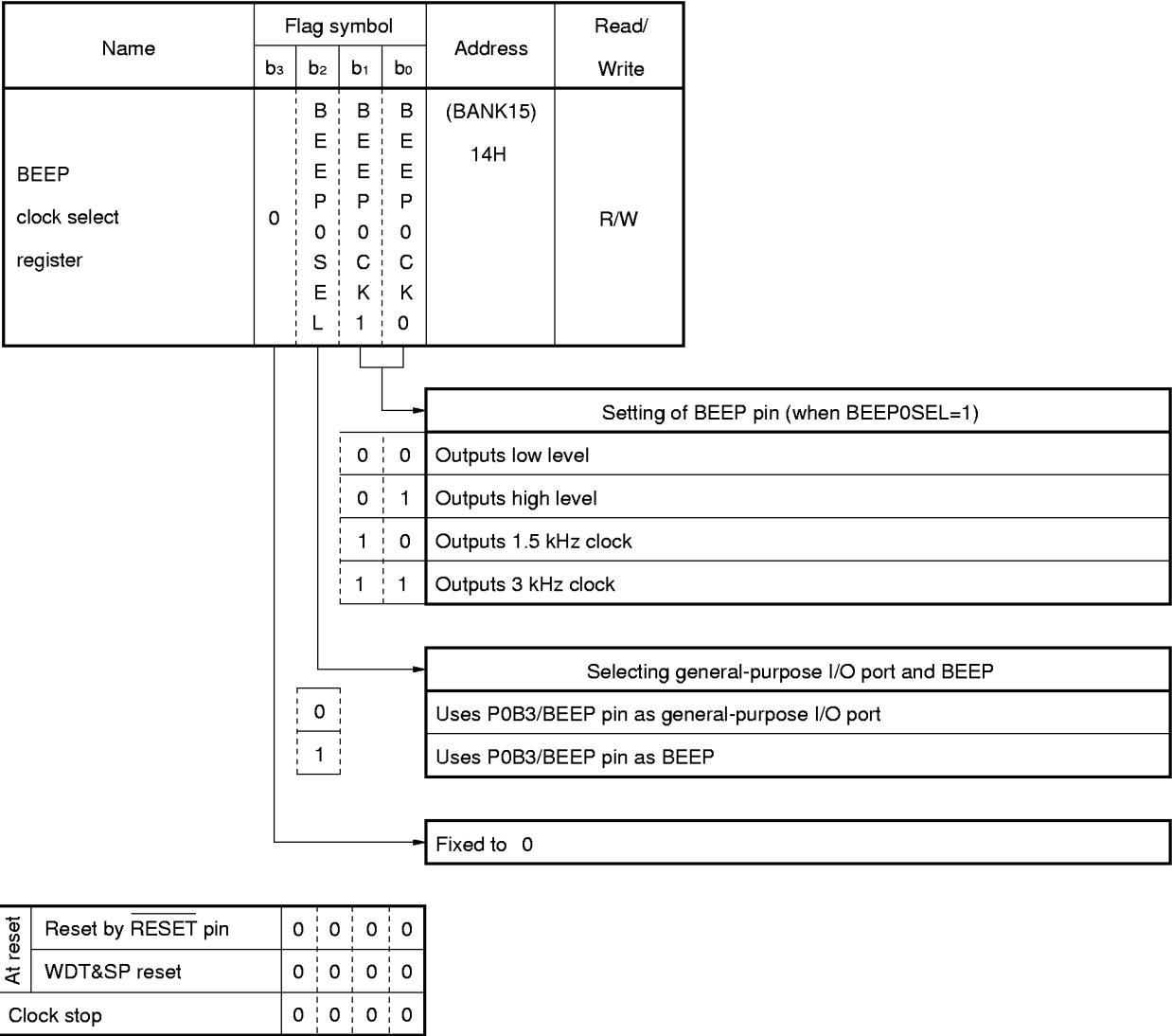
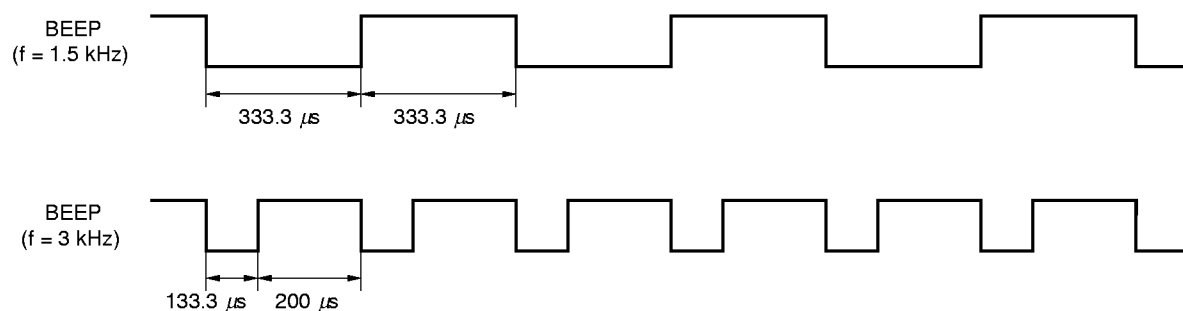


Figure 18-2. Configuration and Function of BEEP Clock Select Register



18.2 Output Wave Form of BEEP

(1) Output wave of $f = 1.5$ kHz and $f = 3$ kHz

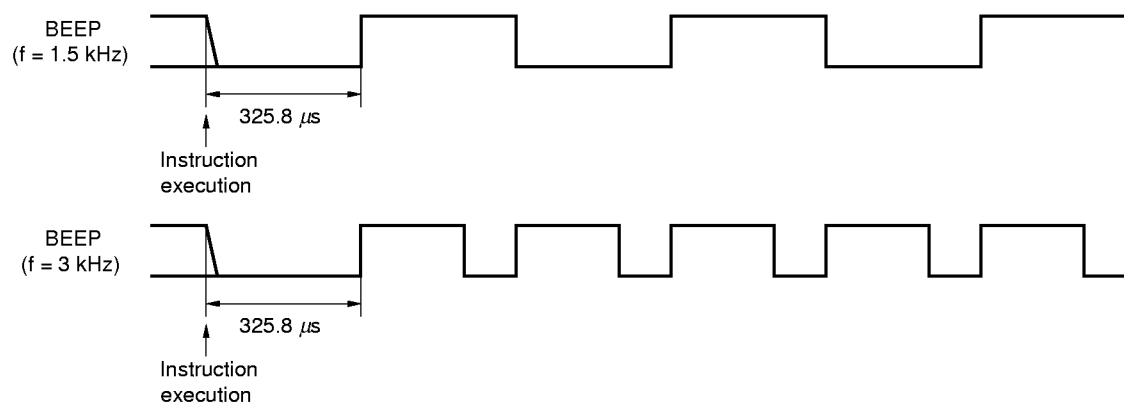


Example Program to output 3-kHz clock from P0B3/BEEP pin

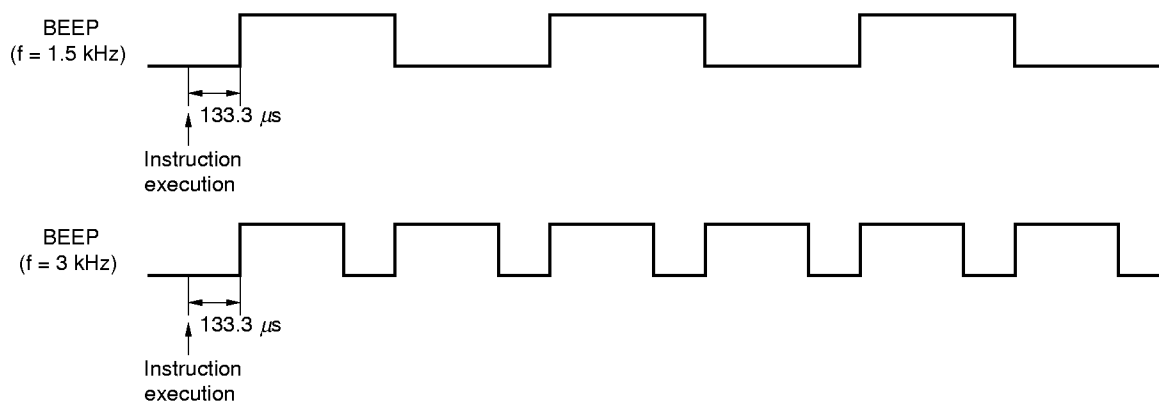
```

BANK1          ; Same as MOV BANK, #0001B
MOV    14H, #0011B ; Writes 0011B to data memory address 14H
                ; Outputs 3 kHz from BEEP pin
    
```

(2) Maximum time until clock is output from P0B3/BEEP pin after instruction execution



(3) Minimum time until clock is output from P0B3/BEEP pin after instruction execution



18.3 Status at Reset

18.3.1 At reset by $\overline{\text{RESET}}$ pin

The P0B3/BEEP pin is set in the general-purpose input port mode.

18.3.2 WDT&SP reset

P0B3/BEEP pin is set in the general-purpose input port mode.

18.3.3 On execution of clock stop instruction

The P0B3/BEEP pin is set in the general-purpose input/output port mode.

18.3.4 In halt status

The P0B3/BEEP output pin retains the previous status.

19. LCD CONTROLLER/DRIVER

The LCD (Liquid Crystal Display) controller/driver can display an LCD of up to 60 dots by a combination of command signal and segment signal outputs.

19.1 Outline of LCD Controller/Driver

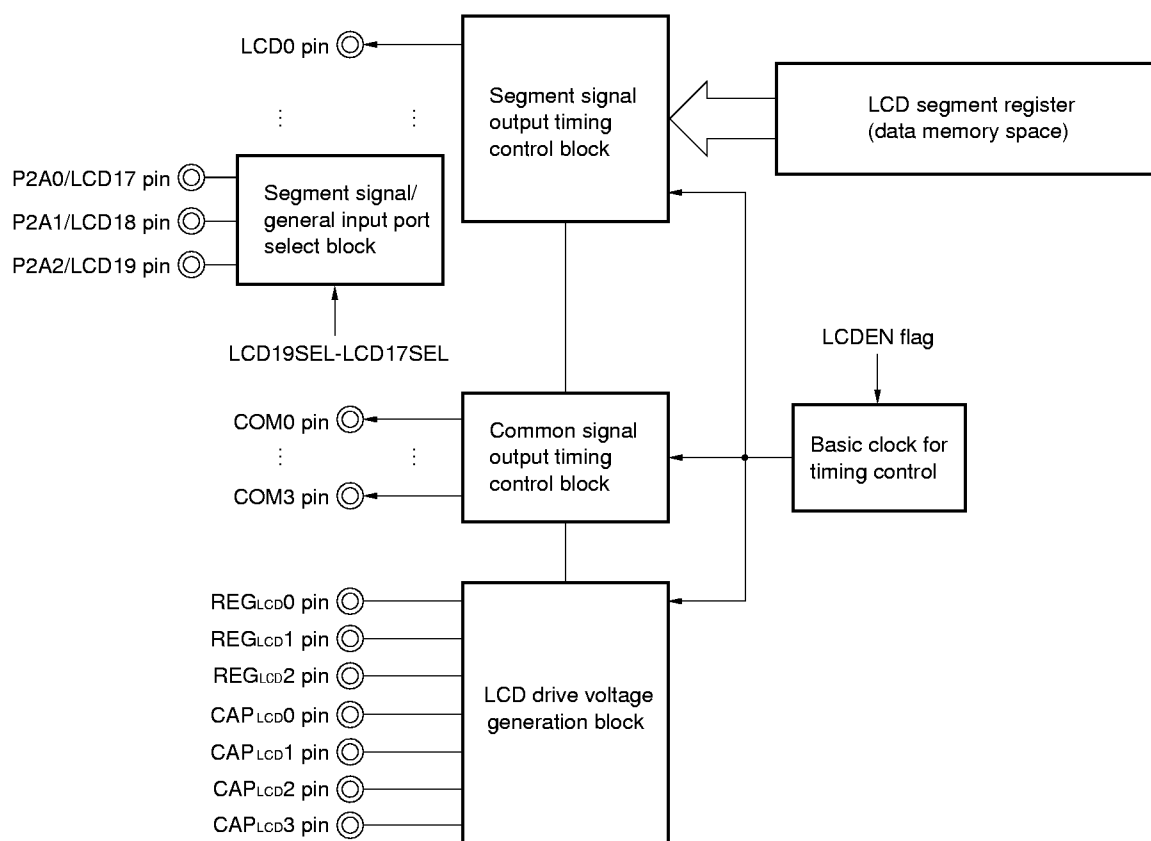
Figure 19-1 outlines the LCD controller/driver.

The LCD controller/driver can be used to display up to 60 dots by using a combination of common signal output pins (COM0 through COM3) and segment signal output pins (LCD0 through LCD19).

The drive mode is 1/4 duty, 1/2 bias, the frame frequency is 62.5 Hz, and drive voltage is V_{LCD} .

Among the segment signal output pins, LCD17 through LCD19 can be used as a general-purpose input port. For details of the general-purpose input port, refer to **11.3 GENERAL-PURPOSE INPUT PORT (P0D, P1C, P2A)**.

Figure 19-1. Outline of LCD Controller/Driver



- Remarks**
1. LCDEN (bit 0 of LCD driver display start register: refer to **Figure 19-8**) turns ON/OFF all LCD display.
 2. LCD19SEL-LCD17SEL (bits 2-0 of LCD port select register: refer to **Figure 19-6**)

19.2 LCD Drive Voltage Generation Block

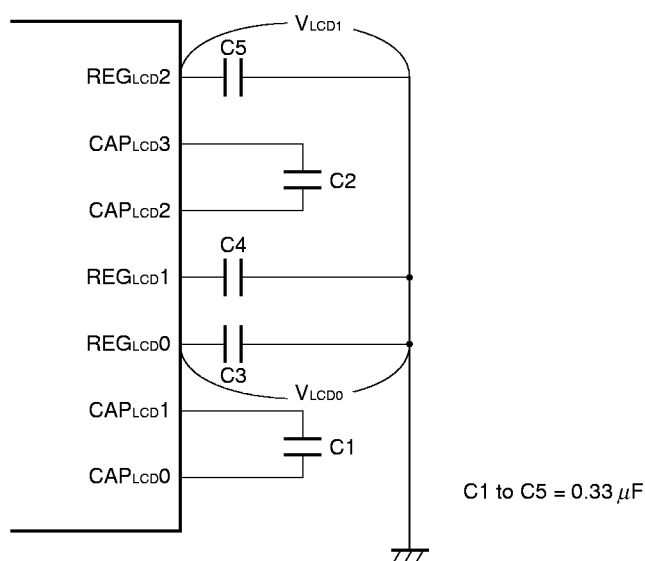
The LCD drive voltage generation block generates a voltage to drive the LCD.

The μPD17934 supplies the LCD drive voltage from an external doubler circuit. To configure a doubler circuit, connect a capacitor to the CAP_{LCD0}, CAP_{LCD1}, CAP_{LCD2}, CAP_{LCD3}, REG_{LCD0}, REG_{LCD1}, and REG_{LCD2} pins.

Figure 19-2 shows an example of configuration of the doubler circuit. To use a voltage of 3.0 V (TYP.), connect as shown in Figure 19-2.

To operate the doubler circuit, the LCDEN flag of the LCD display start register must be set to "1". Unless this flag is set to "1", the LCD drive voltage generation block does not operate. For the LCDEN flag, refer to **19.5 Common Signal Output and Segment Signal Output Timing Control Blocks**.

★ **Figure 19-2. Configuration of Doubler Circuit**



Remark (): pin number

Note that, because of the configuration of the doubler circuit, the values of the LCD drive voltages (V_{LCD1} and V_{LCD0}) differ if the values of C1, C2, C3, C4, and C5 are changed.

19.3 LCD Segment Register

The LCD segment register sets dot data to turn on or turn off dots on the LCD.

Figure 19-3 shows the location in the data memory and configuration of the LCD segment register.

Because the LCD segment register is located in data memory, it can be controlled by all the data memory manipulation instructions.

One nibble of the LCD segment register can set display data of 4 dots (data to turn dots on or off). If the LCD segment register is set to “1” at this time, the LCD display dot is on; the dot goes off if the register is set to “0”.

Figure 19-4 shows the relation between the LCD segment register and LCD display dot.

Figure 19-3. Location on Data Memory and Configuration of LCD Segment Register

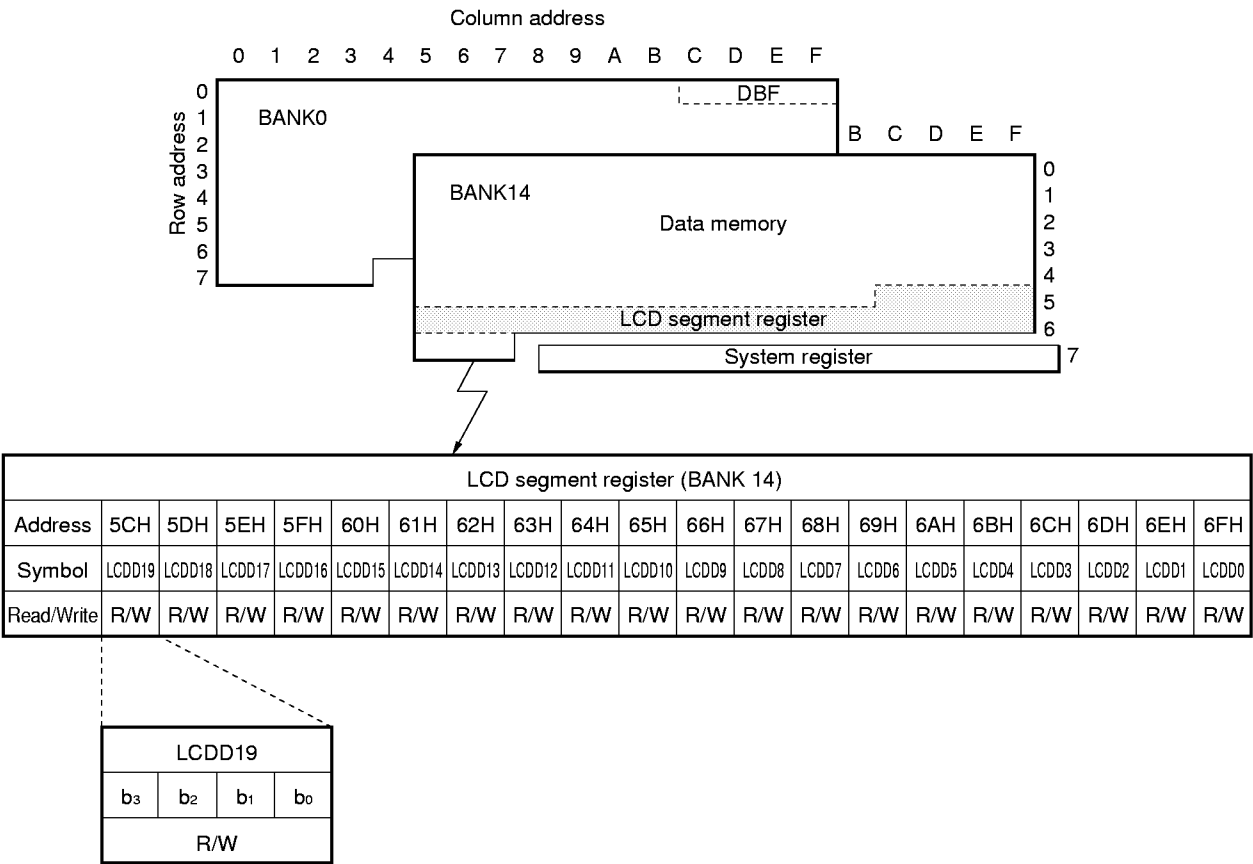
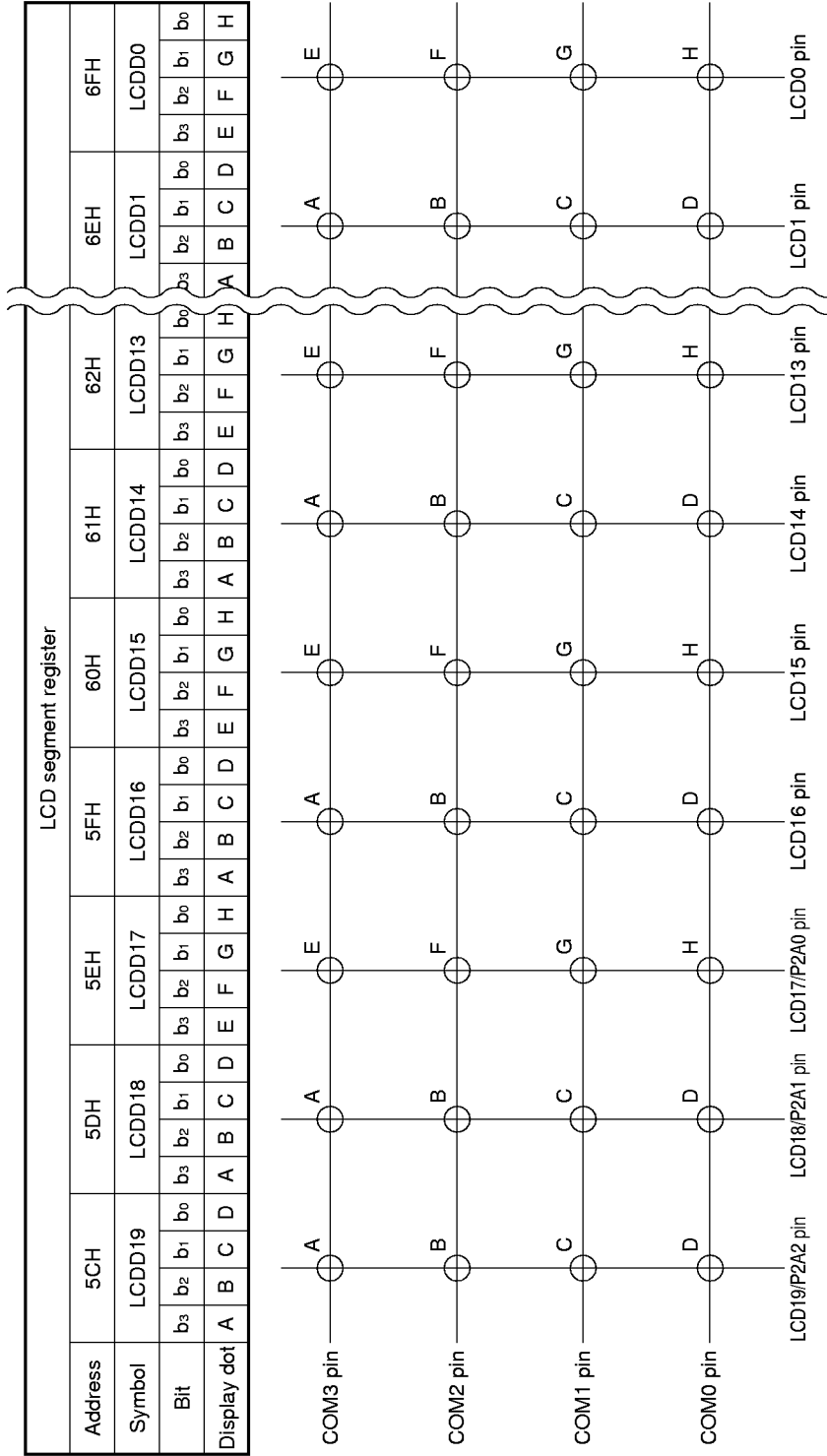


Figure 19-4. Relation between LCD Segment Register and LCD Display Dot



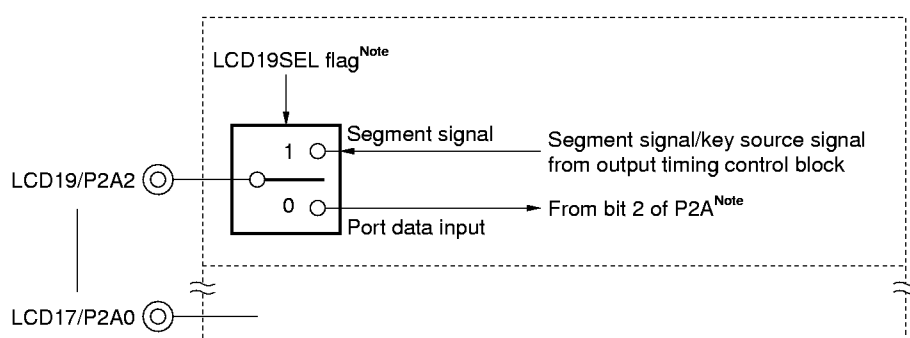
19.4 Segment Signal/General-Purpose Input Port Select Block

Figure 19-5 shows the configuration of the segment signal/general-purpose input port select block.

This block specifies, by using the LCD19SEL through LCD17SEL flags of the LCD port select register, whether each pin is used as a segment signal output pin or a general-purpose input port pin. When each flag is "1", the corresponding pin is used as a segment signal output pin; when it is "0", the pin is used as a general-purpose input port pin.

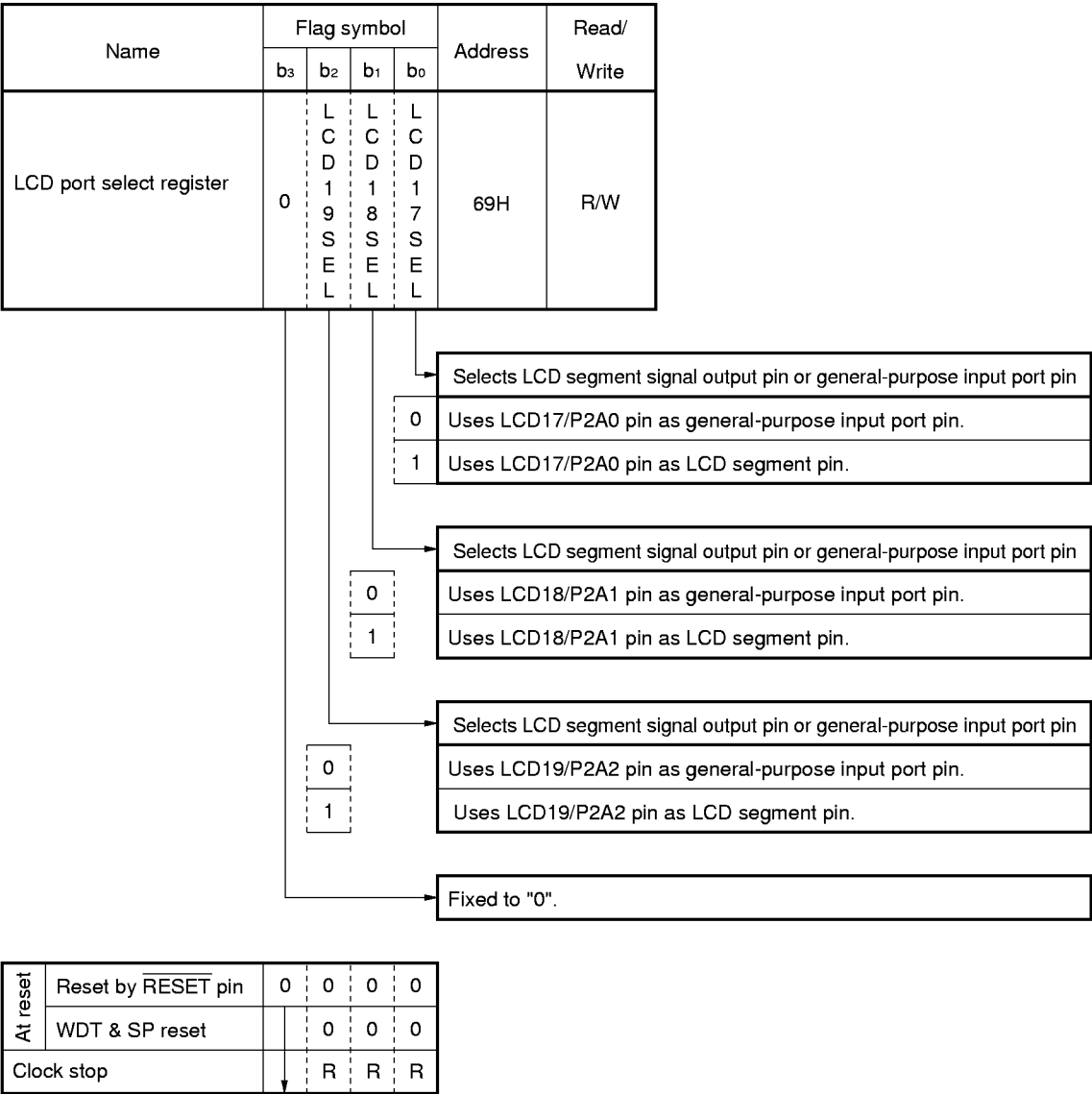
Figure 19-6 shows the configuration of the LCD port select register.

Figure 19-5. Configuration of Segment Signal/General-Purpose Port Select Block



Note The LCD19/P2A2 pin corresponds to the LCD19SEL flag and bit 2 of P2A, the LCD18/P2A1 pin, to the LCD18SEL flag and bit 1 of P2A, and the LCD17/P2A0 pin, to the LCD17SEL flag and bit 0 of P2A, respectively.

Figure 19-6. Configuration of LCD Port Select Register



R: Retained

19.5 Common Signal Output and Segment Signal Output Timing Control Blocks

Figure 19-7 shows the common signal output and segment signal output timing control blocks.

The common signal output timing control block controls the common signal output timing of the COM0 through COM3 pins.

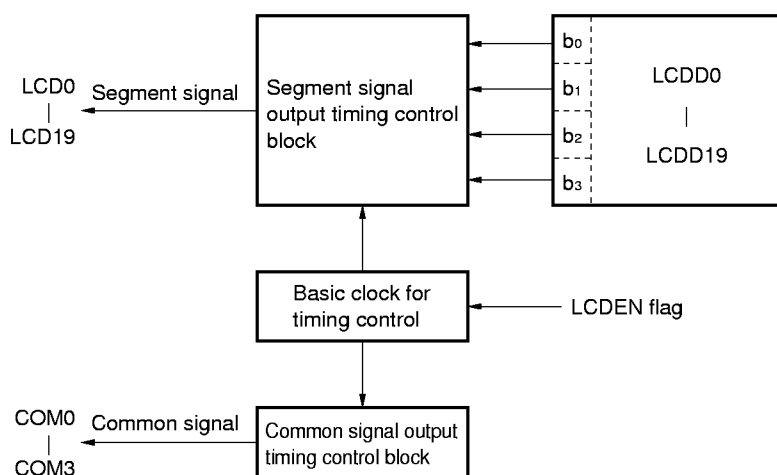
The segment signal output timing control block controls the segment signal output timing of the LCD0 through LCD19 pins.

The common and segment signals are output when the LCDEN flag of the LCD driver display start register is set to "1".

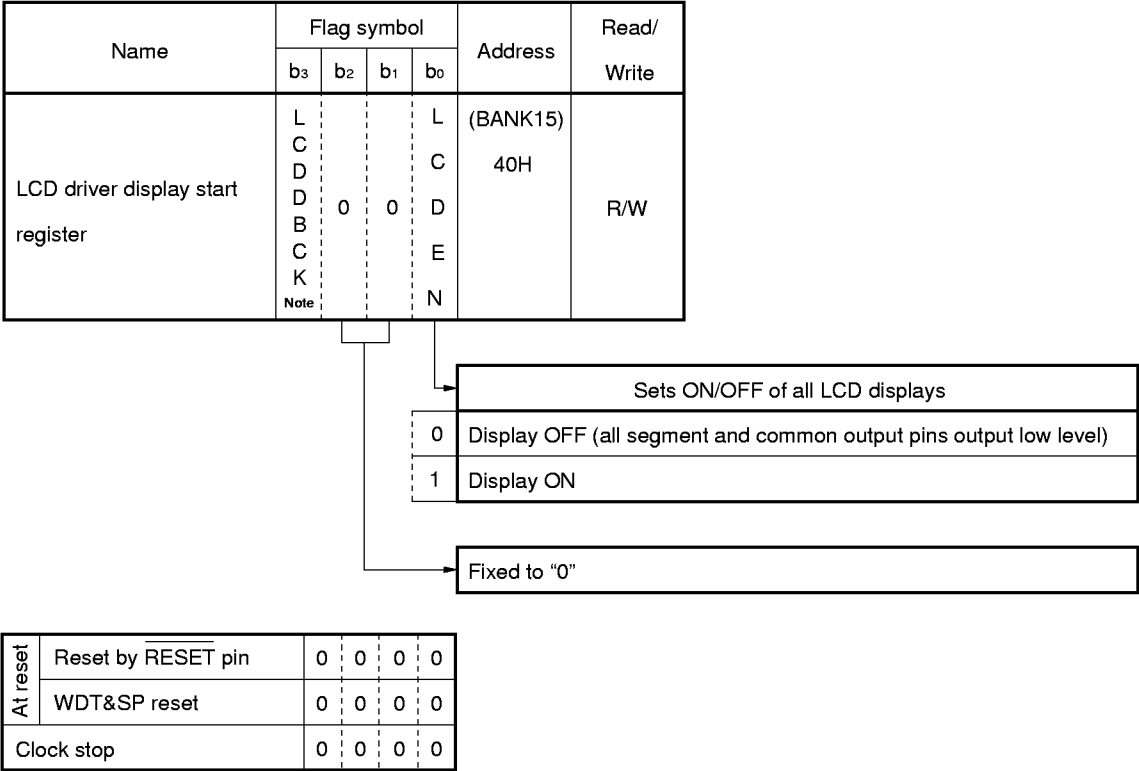
When this flag is reset to "0", all the LCD display dots can be extinguished (refer to **Figure 19-8**).

When LCD display is not carried out, the COM0 through COM3 and LCD0 through LCD19 pins output low level.

Figure 19-7. Configuration of Common Signal Output and Segment Signal Output Timing Control Blocks



★ Figure 19-8. Configuration of LCD Driver Display Start Register



Remark R: Retained

Note Bit 3 of the LCD display start register is a test mode area. Therefore, do not write "1" to this bit.

19.6 Common Signal and Segment Signal Output Waves

Figure 19-9 shows an example of the common signal and segment signal output waves.

The μPD17934 outputs a signal with a frame frequency of 62.5 Hz using a 1/4 duty, 1/2 bias (voltage average method) drive mode.

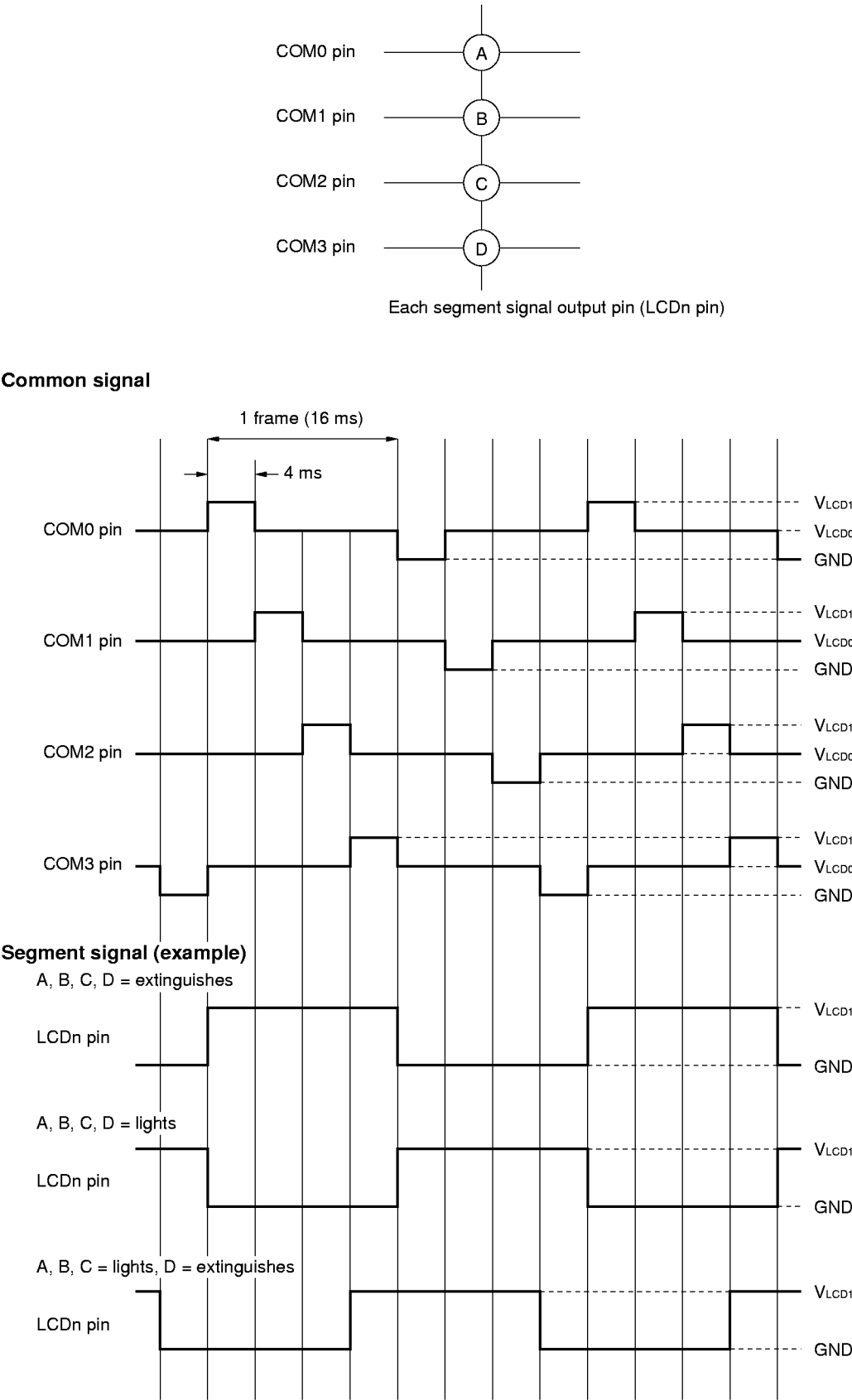
As the common signals, the COM0 through COM3 pins output three levels of voltages (GND, V_{LCD0}, and V_{LCD1}) each having a phase difference of 1/8 from the others. In other words, voltages of ±1/2V_{DD} are output with the V_{LCD0} as the reference. This display method is called the 1/2 bias drive method.

As the segment signals, the segment signal output pins output voltages of two levels (GND and V_{LCD1}) having a phase corresponding to each display dot. Because one segment pin can turn on or off four display dots (A, B, C, and D) as shown in Figure 19-9, sixteen phases can be output by combining lighting and extinguishing of each dot.

Each display dot turned on when the potential difference between a common signal and a segment signal is V_{LCD1}. In other words, the duty factor at which each display dot turns on is 1/4.

This display method is called the 1/4 duty display method, and the frame frequency is 62.5 Hz.

Figure 19-9. Common Signal and Segment Signal Output Waveforms



19.7 Using LCD Controller/Driver

Figure 19-10 shows an example of wiring of an LCD panel using the pins LCD0 through LCD14.

An example of a program that lights the 7 segments connected to LCD0 and LCD1 pins shown in Figure 19-10 is given below.

Example

```

PMN0    MEM    0.01H                ; Preset number storage area
CH      FLG    LCDD0.3              ; Defines symbol with high-order 1 bit of LCD0 register for 'CH' display

LCDDATA:                                ; LCD segment table data

        DW     0000000000000000B    ; BLANK
        DW     0000000000000110B    ; 1
        DW     0000000010110101B    ; 2
        DW     0000000010100111B    ; 3
        DW     000000001100110B     ; 4
        DW     0000000011100011B    ; 5
        DW     0000000011110011B    ; 6
        DW     0000000010000110B    ; 7
        DW     0000000011110111B    ; 8
        DW     0000000011100111B    ; 9

        MOV     AR0, #.DL.LCDDATA SHR 12 AND 0FH
        MOV     AR1, #.DL.LCDDATA SHR 8  AND 0FH
        MOV     AR2, #.DL.LCDDATA SHR 4  AND 0FH
        MOV     AR3, #.DL.LCDDATA        AND 0FH

        LD      DBF0, AR0
        LD      DBF1, AR1
        LD      DBF2, AR2
        LD      DBF3, AR3

        ADD     DBF0, PMN0
        ADDC    DBF1, #0
        ADDC    DBF2, #0
        ADDC    DBF3, #0

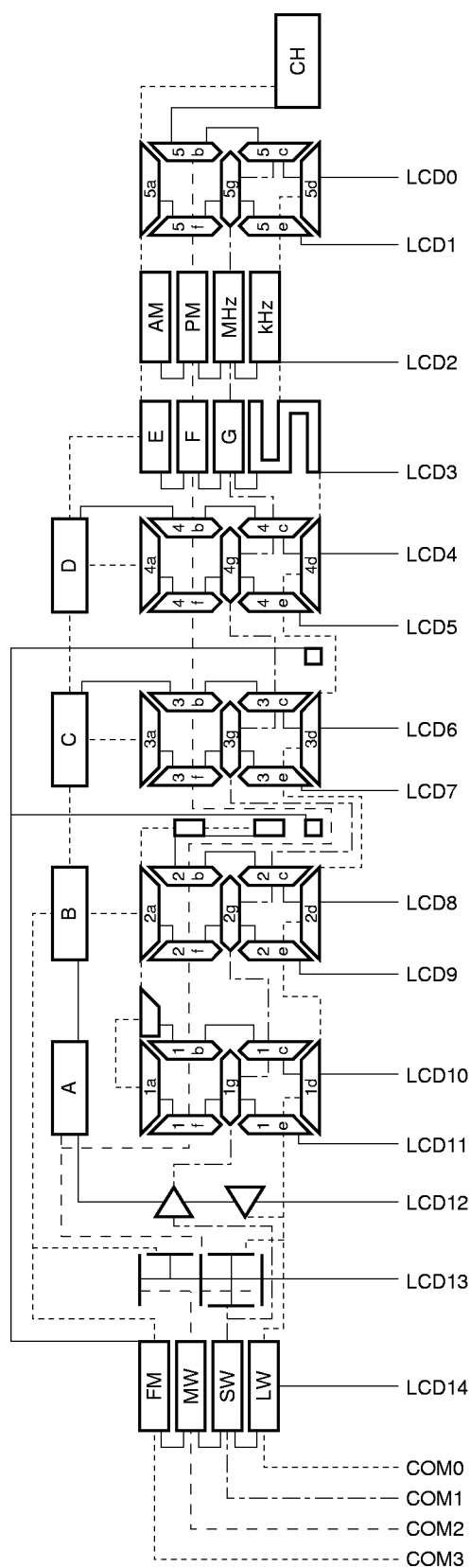
        ST      AR0, DBF0
        ST      AR1, DBF1
        ST      AR2, DBF2
        ST      AR3, DBF3

        MOVT    DBF, @AR              ; Table reference instruction









BANK1
        ST      LCDD0, DBF0
        ST      LCDD1, DBF1
        SET1    CH
        SET1    LCDEN                  ; LCD ON

```

Figure 19-10. Example of Wiring of LCD Panel (When LCD0-LCD14 Pins Are Used)



Correspondence of Segment and Common Pins, and LCD Panel Display (When LCD0-LCD14 Pins Are Used)

Segment Pin	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
	Common														
COM3	FM		B	1a		2a	:	3a	C	4a	D	E	AM	5a	CH
COM2	MW		A	1f	1b	2f	2b	3f	3b	4f	4b	F	PM	5f	5b
COM1	SW			1g	1c	2g	2c	3g	3c	4g	4c	G	MHz	5g	5c
COM0	LW			1e	1d	2e	2d	3e	3d	4e	4d		kHz	5e	5d

19.8 Status at Reset

19.8.1 At reset by RESET pin

The LCD0 through LCD16 pins output a low level.

The LCD17/P2A0 through LCD19/P2A2 pins are set in the general purpose input port.

The COM0 through COM3 pins also output a low level.

Therefore, the LCD display is OFF.

The contents of the LCD segment register are undefined.

19.8.2 WDT&SP reset

The LCD0 through LCD16 pins output a low level.

The LCD17/P2A0 through LCD19/P2A2 pins are set in the general-purpose input port.

The COM0 through COM3 pins output a low level.

Therefore, the LCD display is OFF.

The contents of the LCD segment register are undefined.

19.8.3 On execution of clock stop instruction

The LCD0 through LCD16 pins output a low level.

The LCD17/P2A0 through LCD19/P2A2 pins are set in the general purpose input port.

The COM0 through COM3 pins also output a low level.

Therefore, the LCD display is OFF.

The LCD segment register retains the previous contents.

19.8.4 In halt status

The LCD0 through LCD19 pins output segment signals.

The COM0 through COM3 pins output common signals.

The LCD segment register retains the previous contents.

20. STANDBY

The standby function is used to reduce the current consumption of the device while the device is backed up.

20.1 Outline of Standby Function

Figure 20-1 outlines the standby block.

The standby function reduces the current consumption of the device by partly or totally stopping the device operation.

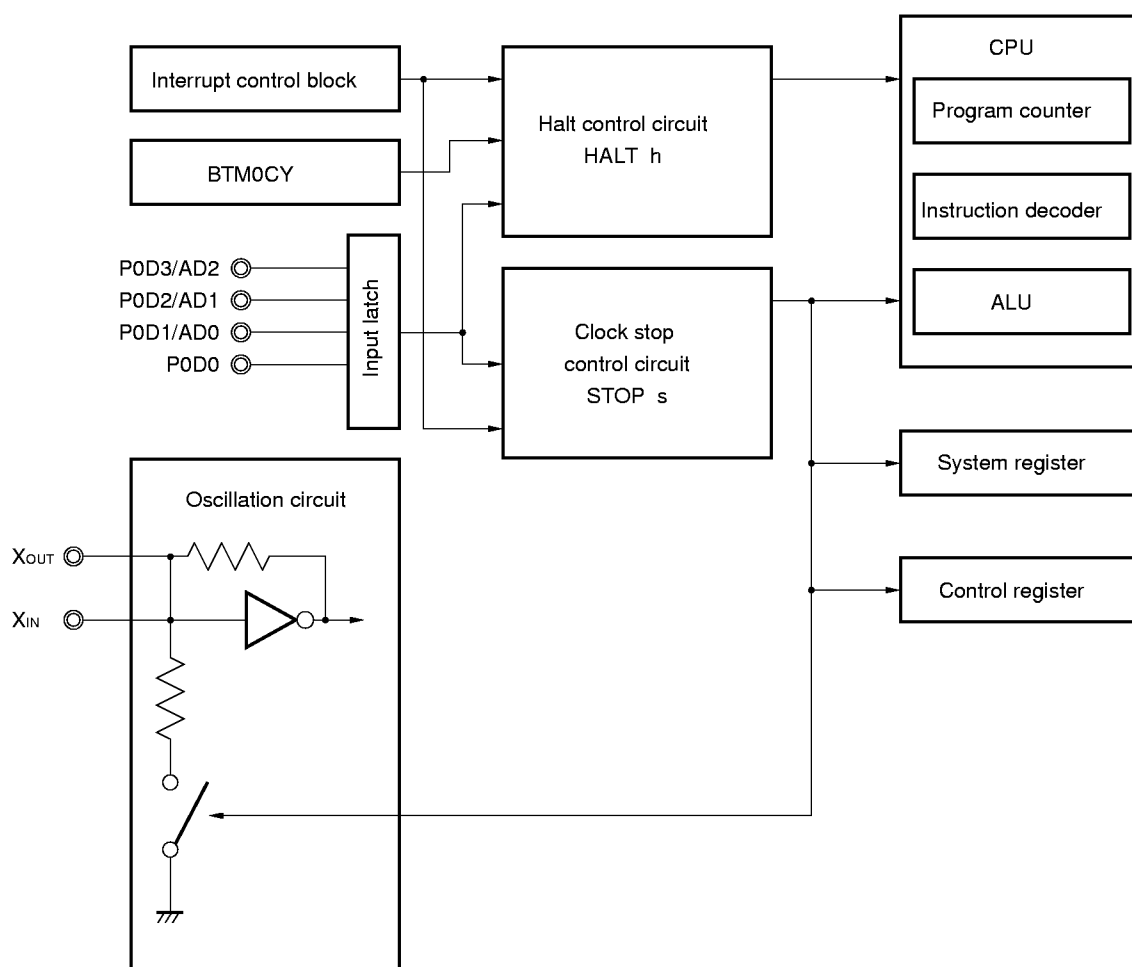
The following two types of standby functions are available for selection as the application requires.

- Halt function
- Clock stop function

The halt function reduces the current consumption of the device by stopping the CPU operation by using a dedicated instruction "HALT h".

The clock stop function reduces the current consumption of the device by stopping the oscillation of the oscillation circuit by using a dedicated instruction "STOP s".

Figure 20-1. Outline of Standby Block



20.2 Halt Function

20.2.1 Outline of halt function

The halt function stops the operating clock of the CPU by executing the “HALT h” instruction.

When this instruction is executed, the program is stopped until the halt status is later released. Therefore, the current consumption of the device in the halt status is reduced by the operating current of the CPU.

The halt status is released by using basic timer 0 carry FF, interrupt, or port input (P0D).

The release condition is specified by operand “h” of the “HALT h” instruction.

20.2.2 Halt status

In the halt status, all the operations of the CPU are stopped. In other words, execution of the program is stopped at the “HALT h” instruction. However, the peripheral hardware units continue the operation specified before execution of the “HALT h” instruction.

For the operation of each peripheral hardware unit, refer to **20.4 Device Operation in Halt and Clock Stop Status**.

20.2.3 Halt release condition

Figure 20-2 shows the halt release condition.

The halt release condition is specified by 4-bit data specified by operand “h” of the “HALT h” instruction.

The halt status is released when the condition specified by “1” in operand “h”.

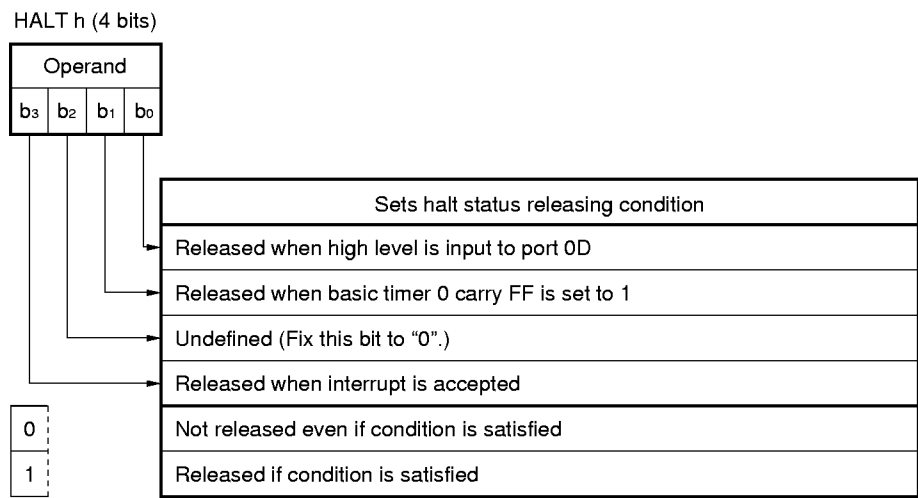
When the halt status is released, program execution is started from the instruction after the “HALT h” instruction.

If the halt status is released by an interrupt, the operation to be performed after the halt status has been released differs depending on whether the interrupts are enabled (EI status) or disabled (DI status) when an interrupt source (IRQxxx = 1) is issued with the interrupt (IPxxx = 1) enabled.

If two or more releasing conditions are specified, the halt status is released when one of the specified condition is satisfied.

If 0000B is set as halt release condition “h”, no releasing condition is set. If the device is reset at this time, the halt status is released.

Figure 20-2. Halt Release Condition



20.2.4 Releasing halt by input port (P0D)

The halt releasing condition using an input port is specified by the "HALT 0001B" instruction.

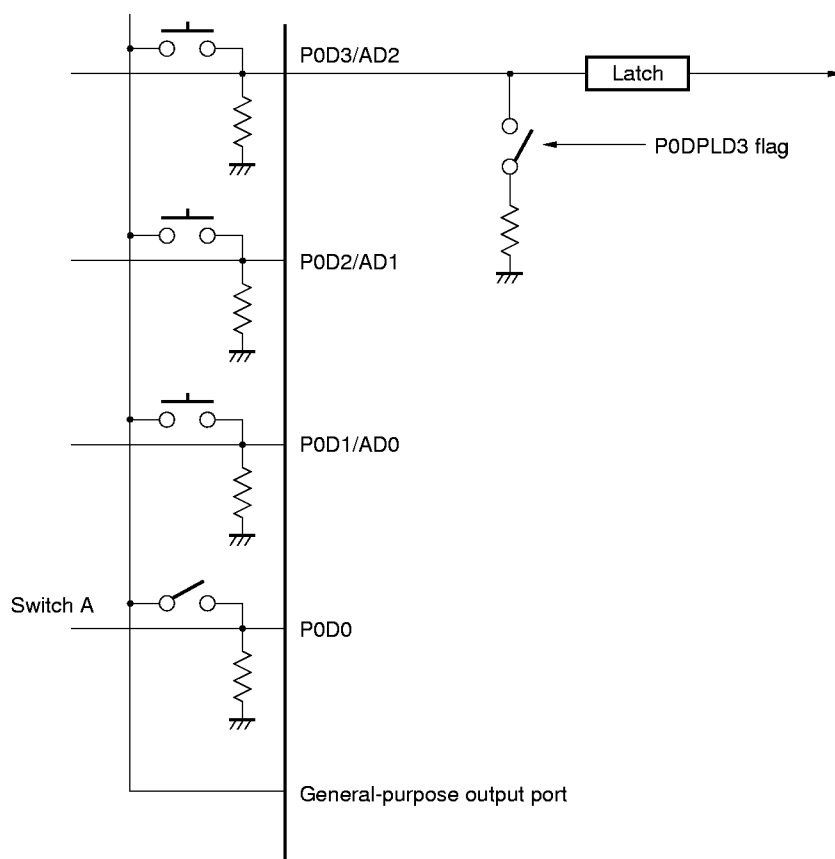
When the halt releasing condition using an input port is specified, the halt status is released if a high level is input to one of the P0D0 through P0D3 pins.

The P0D0 through P0D3 pins are multiplexed with the A/D converter input pins AD0 through AD2 (except P0D0) and the halt status is not released when these pins are used as A/D converter input pins.

An example is given below.

- **To use as key matrix**

The P0D0 through P0D3 pins are general-purpose input port pins which can be set in the input or output mode in 1-bit units and can be connected to an internal pull-down resistor. If connection of the internal pull-down resistor is specified by software, an external resistor can be eliminated as shown in this example (the internal pull down resistor is connected at reset by the $\overline{\text{RESET}}$ pin.)



The "HALT 0001B" instruction is executed after the general-purpose output ports for key source signal are made high. Note that if an alternate switch is used as shown by switch A in the above figure, the halt status is released immediately because a high level is input to the P0D0 pin while switch A is closed.

20.2.5 Releasing halt status by basic timer 0 carry FF

Releasing the halt status by using the basic timer 0 carry FF is specified by the "HALT 0010B" instruction.

When releasing the halt status by the basic timer 0 carry FF is specified, 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 on a one-to-one basis and is set at fixed time intervals (125 ms). Therefore, the halt status can be released at fixed time intervals.

Example To release halt status every 125 ms to execute processing A

```

HLTTMR  DAT      0010B                      ; Symbol definition
LOOP:
    HALT    HLTTMR                      ; Specifies setting of basic timer 0 carry FF as halt releasing condition
    SKT1    BTM0CY                      ; Embedded macro
    BR      LOOP                        ; Branches to LOOP if BTM0CY flag is not set

Processing A

          ; Executes processing A if carry occurs

    BR      LOOP
    
```

20.2.6 Releasing halt status by interrupt

Releasing the halt status by an interrupt is specified by the "HALT 1000B" instruction.

When releasing the halt status by an interrupt is specified, the halt status is released as soon as the interrupt has been accepted.

Many interrupt sources are available as described in 12. INTERRUPTS. Which interrupt source is used to release the halt status must be specified in advance in software.

To accept an interrupt, each interrupt request must be issued from each interrupt source and each interrupt must be enabled (by setting the corresponding interrupt enable flag).

Therefore, the interrupt is not accepted even if the interrupt request is issued, and the halt status is not released.

When the halt status is released by accepting an interrupt, the program flow branches to the vector address of the interrupt.

When the RETI instruction is executed after interrupt servicing, the program flow is restored to the instruction after the HALT instruction.

If all the interrupts are disabled (DI status), the halt status is released by enabling an interrupt (IPxxx = 1) and issuing an interrupt source (IRQxxx = 1), and the flow of the program goes to the instruction after the HALT instruction.

Example Releasing halt status by timer 0 and INT pin interrupts

In this example, the halt status is released and processing B is executed when timer 0 interrupt is accepted. And processing A is executed when INT pin interrupt is accepted.

Each time the halt status has been released, processing C is executed.

```

HLTINT    DAT        1000B                ; Symbol definition
START:                                          ; Address 0000H
          BR          MAIN
;*** Interrupt vector address ***
          NOP                ; SIO2
          NOP                ; Basic timer 1
          BR          INTTMO        ; Branches to timer 0 interrupt processing
INTP:      ; Branches to INT pin interrupt processing
          ; INT pin interrupt vector address (0004H)

          Processing A                ; INT pin interrupt processing
          EI
          RETI
INTTMO:
          Processing B                ; Timer 0 interrupt processing
          EI
          RETI
MAIN:
          INITFLG    TMOCK1, TMOCK0        ; Sets timer 0 count clock to 40 μs
          MOV        DBF1, #0
          MOV        DBF0, #32H
          PUT        TM0M,DBF              ; Sets time interval of timer 0 interrupt to 2 ms
          SET2       TM0RES, TM0EN          ; Resets and starts timer 0
          SET2       IPTMO, IP0             ; Enables INT pin and timer 0 interrupts
LOOP:
          Processing C                ; Main routine processing
          EI                ; Enables all interrupts
          HALT        HLTINT              ; Specifies releasing halt status by interrupt
          ;<1>
          BR          LOOP

```

If the INT pin interrupt request and timer 0 interrupt request are issued simultaneously in the halt status, processing A for the INT pin, which has the higher hardware priority, is executed.

After execution of processing A and when “RETI” is executed, the program branches to the “BR LOOP” instruction of <1>. However, the “BR LOOP” instruction is not executed, and timer 0 interrupt is immediately accepted.

When the “RETI” instruction is executed after processing B of timer 0 interrupt has been executed, the “BR LOOP” instruction is executed.

Caution To reset the interrupt request flag (IRQxxx) once before the halt instruction is executed, insert a NOP instruction (or one or more other instructions) between the HALT instruction and the instruction that resets the interrupt request flag (IRQxxx) as shown below. If a NOP instruction (or one or more other instructions) is not inserted, the interrupt request flag is not reset, and therefore, the halt status is released immediately.

Example

```
:
:                               ; IRQxxx is set at certain timing
:
CLR1  IRQxxx    ; Resets IRQxxx flag once
NOP    ; Resets IRQxxx flag at this timing
        ; Unless this period is missing, the IRQxxx flag is not reset,
        ; and the next HALT instruction is immediately released
HALT   1000B    ;
```

20.2.7 If two or more releasing conditions are specified at same time

If two or more halt releasing conditions are specified at same time, the halt status is released when one of the conditions is satisfied.

The following program example shows how the releasing conditions are identified if two or more conditions are satisfied at the same time.

Example

```

      HLTINTP  DAT    1000B
      HLTBTM   DAT    0010B
      HLTP0D   DAT    0001B
      P0D      MEM    0.73H

START:
      BR      MAIN
;*** Interrupt vector address ***
      NOP                    ; SIO
      NOP                    ; Basic timer 1
      NOP                    ; TM0
      NOP                    ; INT

INTP:                                ; INT pin interrupt vector address (0004H)

      Processing A           ; INT pin interrupt processing

      EI
      RETI

BTMOUP:                                ; Timer carry FF processing

      Processing B

      RET

P0DP:                                ; P0D input processing

      Processing C

      RET

MAIN:
      SET1    IP0            ; Enables INT pin interrupt
      EI

LOOP:
      HALT    HLTINT OR HLTBTM OR HLTP0C
                                ; Selects interrupt, timer carry FF (125 ms), and P0D input as halt releasing
                                ; conditions
      SKF1    BTM0CY          ; Detects BTM0CY flag
      CALL    BTM0UP          ; Timer carry FF processing if flag is set to 1
      SKF     P0D, 1111B      ; Detects P0D input
      CALL    P0DP            ; Port input processing if P0D is high
      BR      LOOP

```

In the above example, three halt status releasing conditions, INT pin interrupt, 125-μs basic timer 0 carry FF, and port 0D input, are specified.

To identify which condition has released the halt status, a vector address (interrupt), BTM0CY flag (timer carry FF), and port register (port input) are detected.

To use two or more releasing conditions, the following two points must be noted.

- When the halt status is released, all the specified releasing conditions must be detected.
- The releasing condition with the higher priority must be detected first.

20.3 Clock Stop Function

20.3.1 Outline of clock stop function

The clock stop function stops the oscillation circuit of a 75-kHz crystal resonator by executing the “STOP s” instruction (clock stop status).

★ Therefore, the current consumption of the device is reduced to 10 μA MAX ($T_A = -10$ to $+50$ °C, $V_{DD} = 1.05$ to 1.8 V)

20.3.2 Clock stop status

In the clock stop status, all the device operations of the CPU and peripheral hardware units are stopped because the generation circuit of the crystal resonator is stopped.

For the operations of the CPU and peripheral hardware units, refer to **20.4 Device Operation in Halt and Clock Stop Status**.

20.3.3 Releasing clock stop status

Figure 20-3 shows the stop status releasing conditions.

The stop status releasing condition is specified by 4-bit data specified by operand “s” of the “STOP s” instruction.

The stop status is released when the condition specified by “1” in operand “s” is satisfied.

When the stop status has been released, a halt period which is half the time ($t_{SET}/2$) specified by the basic timer 0 clock selection register as oscillation circuit stabilization wait time has elapsed, and the program execution is started from the instruction next to the “STOP s” instruction. If releasing the stop status by an interrupt is specified, however, the program operation after the stop status has been released differs depending on whether the interrupt is enabled (EI status) or disabled (DI status) when an interrupt source is issued ($IRQ_{xxx} = 1$) with the interrupt enabled ($IP_{xxx} = 1$).

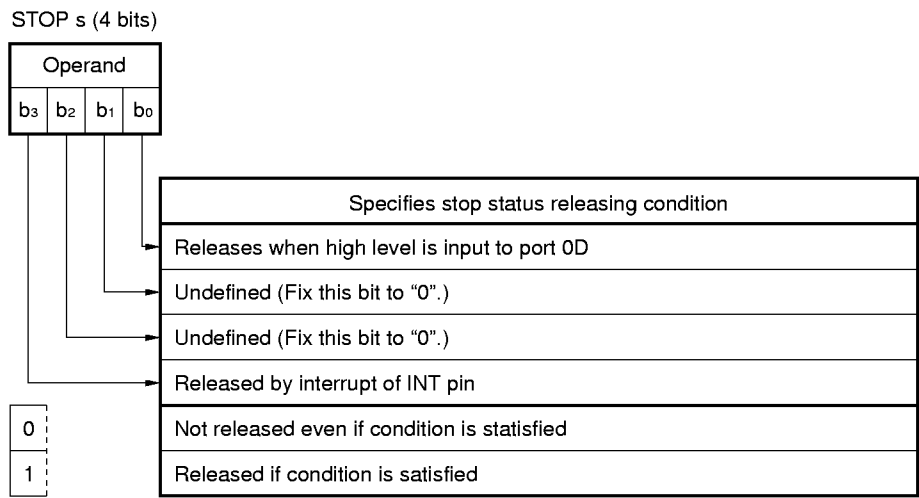
If all the interrupts are enabled (EI status), the stop status is released when the interrupt is enabled ($IP_{xxx} = 1$) and the interrupt source is issued ($IRQ_{xxx} = 1$), and the program flow returns to the instruction next to the STOP instruction.

If all the interrupts are disabled (DI status), the stop status is released when the interrupt is enabled ($IP_{xxx} = 1$) and the interrupt resource is issued ($IRQ_{xxx} = 1$), and the program flow returns to the instruction next to the STOP instruction.

If two or more releasing conditions are specified at one time, and if one of the conditions is satisfied, the stop status is released.

If 0000B is specified as stop releasing condition “s”, no releasing condition is satisfied. If the device is reset at this time the stop status is released.

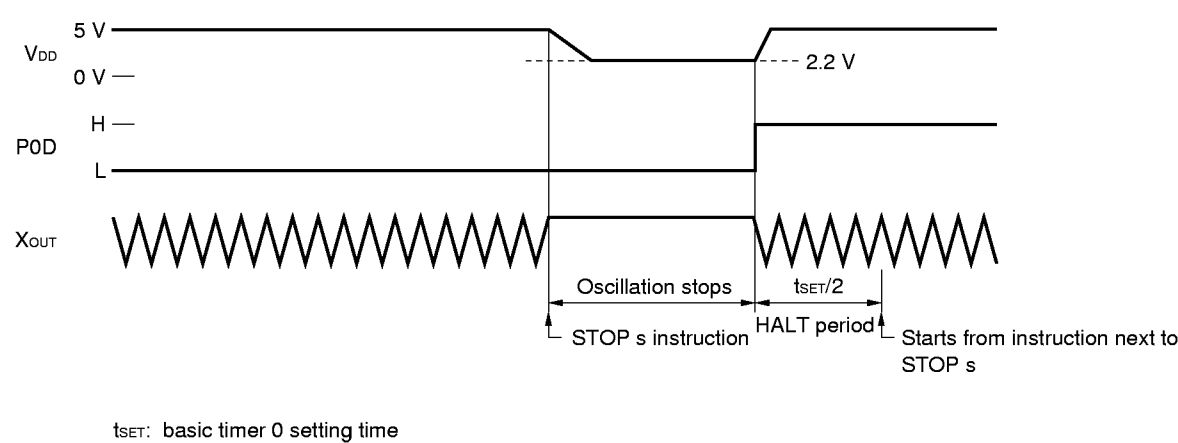
Figure 20-3. Stop Releasing Conditions



20.3.4 Releasing clock stop status by high level input of port 0D

Figure 20-4 illustrates how the clock stop status is released by the high level input to port 0D.

Figure 20-4. Releasing Clock Stop Status By High Level Input of Port 0D



20.4 Device Operation in Halt and Clock Stop Status

Table 20-1 shows the operations of the CPU and peripheral hardware units in the halt and clock stop status.

In the halt status, all the peripheral hardware units continue the normal operation until instruction execution is stopped.

In the clock stop status, all the peripheral hardware units stop operation.

The control registers that control the operations of the peripheral hardware units operate normally (not initialized) in the halt status, but are initialized to specified values when the clock stop instruction is executed.

In other words, all peripheral hardware continues the operation specified by the control register in the halt status, and the operation is determined by the initialized value of the control register in the clock stop status.

For the values of the control registers in the clock stop status, refer to **8. REGISTER FILE (RF) AND CONTROL REGISTER**.

Table 20-1. Device Operation in Halt and Clock Stop Status

Peripheral Hardware	Status	
	Halt	Clock stop
Program counter	Stops at address of HALT instruction	Stops at address of STOP instruction
System register	Retained	Retained
Peripheral register	Retained	Partly initialized ^{Note}
Control register	Retained	Partly initialized ^{Note}
Timer	Normal operation	Operation stops
PLL frequency synthesizer	Normal operation	Operation stops
A/D converter	Normal operation	Operation stops
Serial interface	Stops operation when internal clock (master) is selected and continues operation when external clock (slave) is selected	Stops operation and used as general-purpose I/O port
Frequency counter	Normal operation	Stops operation and used as general-purpose input port
BEEP output	Normal operation	Stops operation and used as general-purpose I/O port
LCD controller/driver	Normal operation	Stops operation
General-purpose I/O port	Normal operation	Retained
General-purpose input port	Normal operation	Input port
General-purpose output port	Normal operation	Retains output latch

Note For the value to which these registers are initialized, refer to **5. SYSTEM REGISTER (SYSREG)** and **8. REGISTER FILE (RF) AND CONTROL REGISTER**.

20.5 Cautions on Processing of Each Pin in Halt and Clock Stop Status

The halt status is used to reduce the current consumption when, say, only the watch is used.

The clock stop function is used to reduce the current consumption of the device to only use the data memory.

Therefore, the current consumption must be reduced as much as possible in the halt status or clock stop status.

At this time, the current consumption significantly varies depending on the status of each pin, and the points shown in Table 20-2 must be noted.

Table 20-2. Status of Each Pin in Halt and Clock Stop Status and Cautions (1/2)

Pin Function		Pin Symbol	Status of Each Pin and Cautions on Processing	
			Halt status	Clock stop status
General-purpose I/O port	Port 0B	P0B3/BEEP P0B2/SO1 P0B1/SI1/SO2 P0B0/ $\overline{\text{SCK}}$	Retains status before halt (1) When specified as output pin Current consumption increases if pin is externally pulled down while it outputs high level, or externally pulled up while it outputs low level. Exercise care in using N-ch open-drain output (P1A3-P1A0, P1D3-P1D0) (2) When specified as input pin Current consumption increases due to noise if pin is floated	All port pins are set in general-purpose port mode (except P0D3/AD2- P0D1/AD0, P2A2/LCD19-P2A0/LCD17) Input or output mode of general-purpose I/O port set before clock stop status is retained. (1) When specified as general-purpose output port Current consumption increases due to noise if pin is floated (2) When specified as general-purpose input port Current consumption does not increase due to noise even if pin is floated
	Port 1A	P1A3 P1A2 P1A1 P1A0		
	Port 1D	P1D3 P1D2 P1D1 P1D0		
	Port 2B	P2B3-P2B0		
	Port 2C	P2C3-P2C0		
General-purpose input port	Port 0D	P0D3/AD2 P0D2/AD1 P0D1/AD0 P0D0	(3) Port 0D (P0D3/AD1-P0D1/AD0, P0D0) Current consumption increases if pin is externally pulled up because it is provided with pull-down resistor selectable by software (4) Port 1C (P1C3/FMIFC/AMIFC, P1C2/AMIFC, P1C1/TM1, P1C0/TM0)	(3) P0D3/AD2-P0D1/AD0 Pin used for A/D converter is retained as is. Pull-down resistor of P0D3- P0D0 pin retains previous status (4) P2A2/LCD19-P2A0/LCD17 Pin used for LCD segment is retained as is.
	Port 1C	P1C3/FMIFC/AMIFC P1C2/AMIFC P1C1/TM1 P1C0/TM0		
	Port 2A	P2A2/LCD19 P2A1/LCD18 P2A0/LCD17		
General-purpose output port	Port 0A	P0A1 P0A0	When P1C2/AMIFC or P1C3/FMIFC/AMIFC pin is used for IF counter, current consumption increases because internal amplifier operates	Specified as general-purpose output port. Output contents are retained as is. If pin is externally pulled down while it outputs high level or externally pulled up while it outputs low level, current consumption increases
	Port 0C	P0C3-P0C0		

Table 20-2. Status of Each Pin in Halt and Clock Stop Status and Cautions (2/2)

Pin Function	Pin Symbol	Status of Each Pin and Cautions on Processing	
		Halt status	Clock stop status
External interrupt	INT	Current consumption increases due to noise if pin is floated	
PLL frequency synthesizer	VCOL VCOH EO0 EO1	Current consumption increases during PLL operation. When PLL is disabled, pin is in following status: VCOH, VCOL : internally pulled down EO1, EO0 : floated	PLL is disabled VCOH, VCOL : internally pulled down EO1, EO0 : floated
Crystal oscillation circuit	X _{IN} X _{OUT}	Current consumption changes due to oscillation waveform of crystal oscillation circuit. The higher oscillation amplitude, the lower current consumption. Oscillation amplitude must be evaluated because it is influenced by crystal resonator or load capacitor used	X _{IN} pin is internally pulled down, and X _{OUT} pin outputs high level
LCD controller/driver	LCD19/P2A2 LCD18/P2A1 LCD17/P2A0 LCD16 LCD0 COM3 COM0	(1) When LCD19/P2A2-LCD17/P2A0 are used as general-purpose input port pins When these pins are used as general-purpose input port pins, the same points as described above must be noted. (2) When LCD controller/driver is used (LCDEN = 1) LCD19-LCD0 : Output segment signals COM3-COM0 : Output common signals (3) LCD display off (LCDEN = 0) LCD19-LCD0 : Output segment signals COM3-COM0 : Output common signals	(1) When LCD19/P2A2-LCD17/P2A0 are used as general-purpose input port pins When these pins are used as general-purpose input port pins, the same points as described above must be noted. (2) When LCD controller/driver is used LCDEN = 0 LCD16-LCD0 : Output low level COM3-COM0 : Output low level

21. RESET

21.1 Outline of Reset

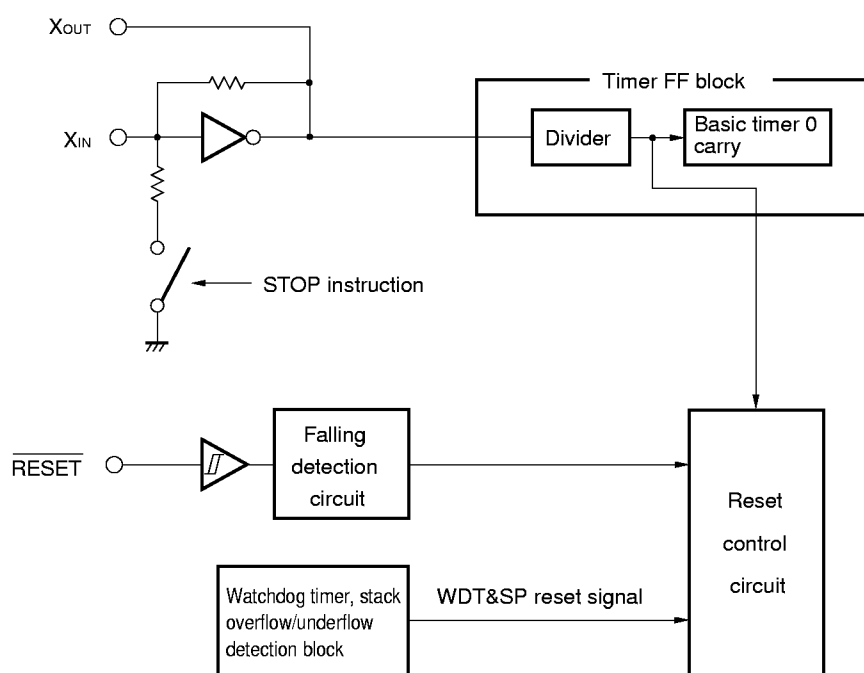
The reset function is used to initialize the device.

The μPD17934 can be reset in the following ways:

- Reset by $\overline{\text{RESET}}$ pin
- WDT&SP reset

★

Figure 21-1. Configuration of Reset Block



★ 21.2 Reset by $\overline{\text{RESET}}$ Pin

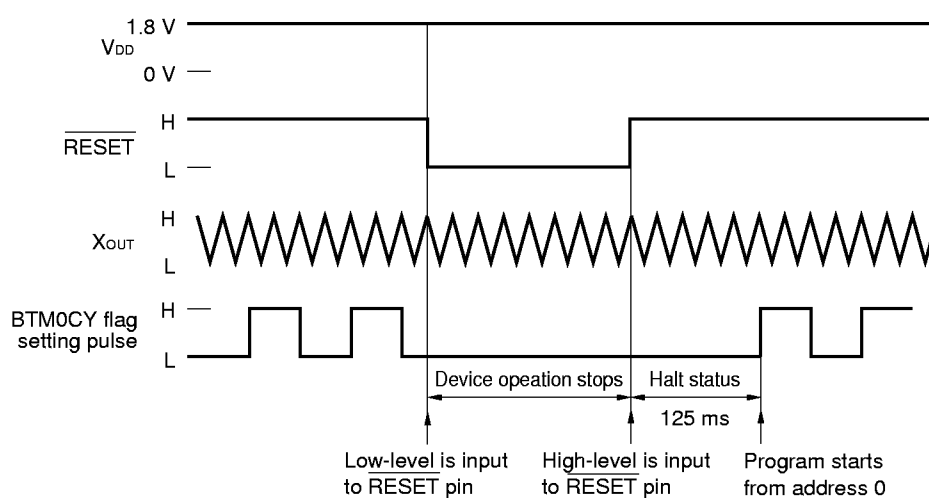
When a low level is input to the $\overline{\text{RESET}}$ pin, an internal reset signal is generated.

At this point, the program counter, stack, system registers, and control registers are initialized (for the initial value, refer to the description of each register).

When the $\overline{\text{RESET}}$ pin is raised next time, the program starts from address 0 at the rising edge of the basic timer 0 carry FF setting signal 125 ms after a high level has been input to the $\overline{\text{RESET}}$ pin.

If reset is executed by the $\overline{\text{RESET}}$ pin during program execution, the data in the data memory may be lost.

★ **Figure 21-2. Reset Operation by $\overline{\text{RESET}}$ Pin**

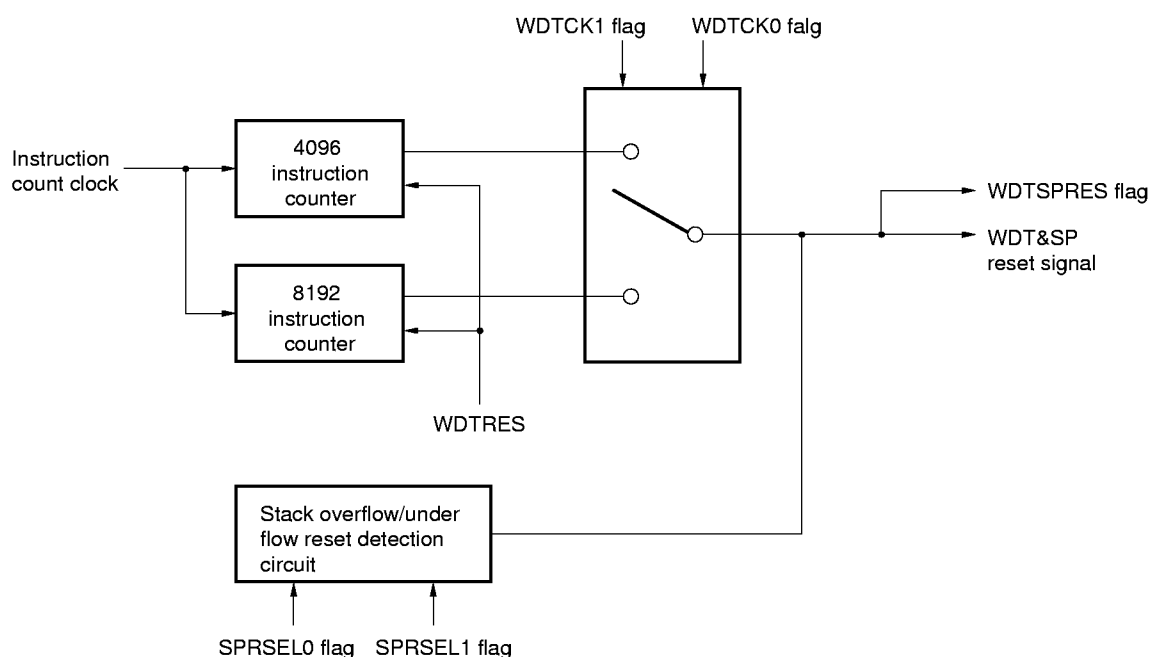


21.3 WDT&SP Reset

WDT&SP reset includes the following:

- Watchdog timer reset
- Stack pointer overflow/underflow reset

Figure 21-3. Outline of WDT&SP Reset



21.3.1 Watchdog timer reset

The watchdog timer is a circuit that generates a reset signal when the execution sequence of the program is abnormal (hung-up).

Hanging-up means that the program jumps to an unexpected routine due to external noise, entering a specific infinite loop and causing the system to be deadlocked. By using the watchdog timer, the program can be restored from this hang-up status because a reset signal is generated from the watchdog timer at fixed time intervals and program execution is started from address 0.

The watchdog timer does not function in the clock stop mode and halt mode.

Resetting by the watchdog timer initializes all the registers except the stack overflow selection register, watchdog timer counter reset register, and basic timer 0 carry register.

The watchdog timer reset is detected by the WDTSPRES flag (R&Reset).

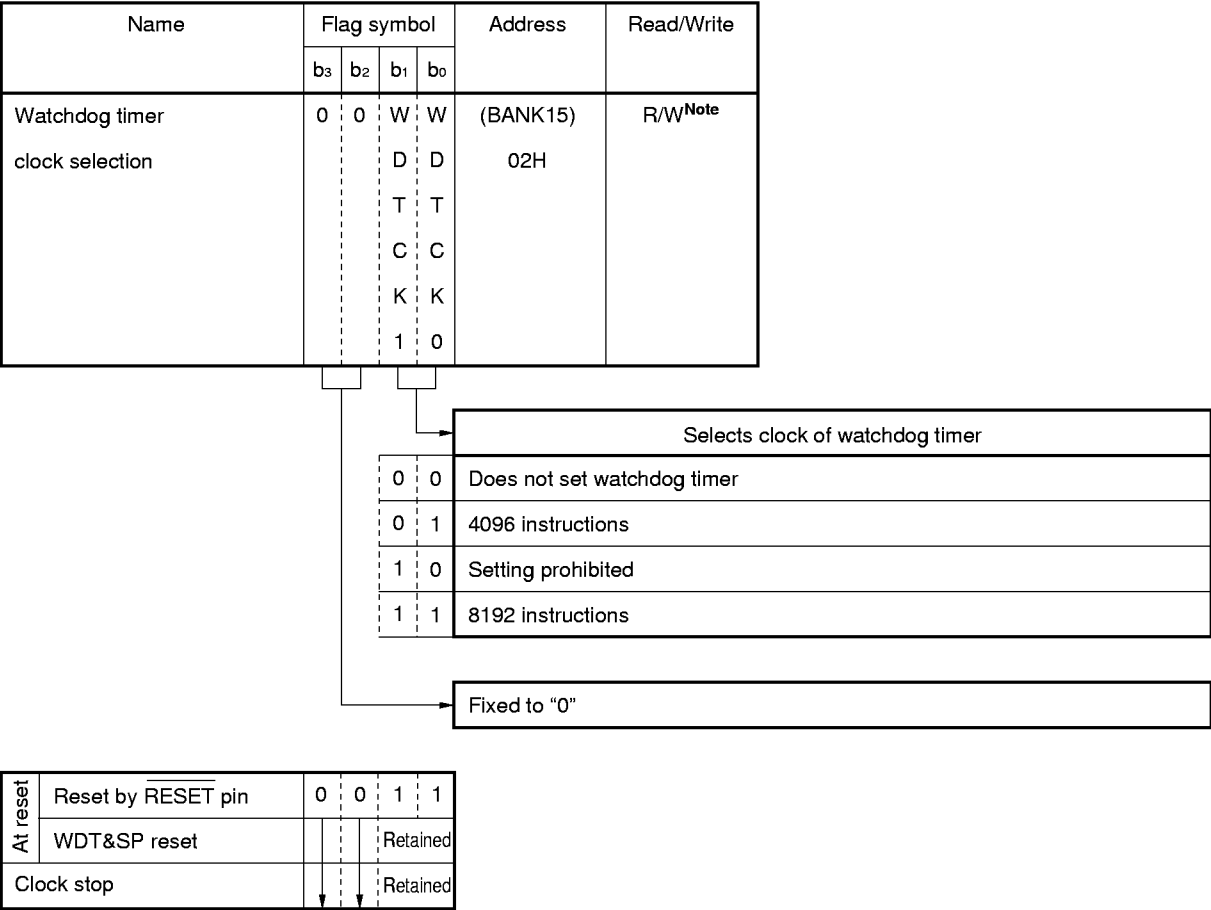
21.3.2 Watchdog timer setting flags

These flags can be set only once after power-ON reset on power application or reset by the $\overline{\text{RESET}}$ pin. The WDTCK0 and WDTCK1 flags select an interval at which the reset signal is output. The reference time can be selected to the following three conditions:

- 4096 instructions
- 8192 instructions
- Watchdog timer not set

On power application, 8192 instructions are selected. If the reset signal generation interval is specified to be 8192 instructions, the watchdog timer FF must be reset at intervals not exceeding 8192 instructions. The valid reset period is from 1 to 8192 instructions. If the reset signal generation interval is 4096 instructions, the watchdog timer FF must be reset at intervals not exceeding 4096 instructions. The valid reset period is from 1 to 4096 instructions.

Figure 21-4. Configuration of Watchdog Timer Clock Selection Register



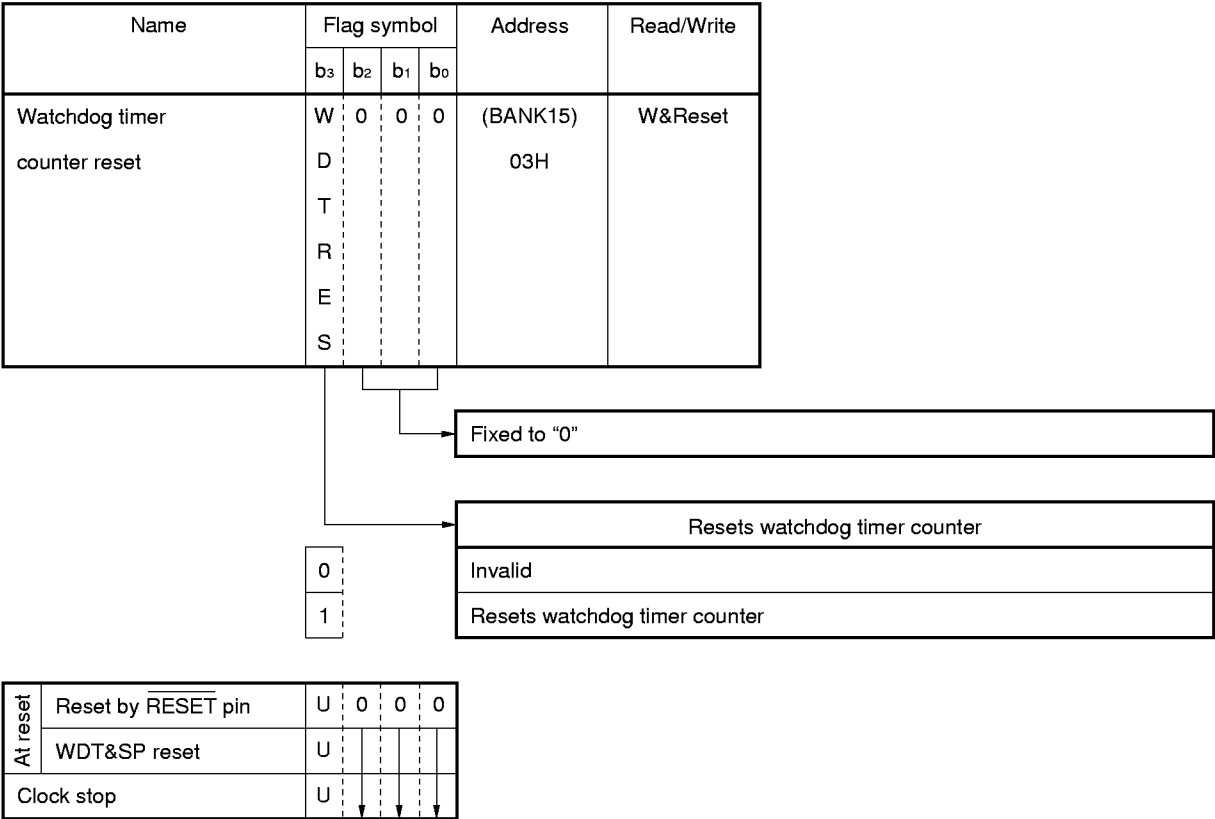
Note Can be written only once.

The WDTRES flag is used to reset the watchdog timer counter.

When this flag is set to 1, the watchdog timer counter is automatically reset.

If the WDTRES flag is set to 1 once within a reference time in which the WDTCK0 and WDTCK1 flags are set, the reset signal is not output by the watchdog timer.

Figure 21-5. Configuration of Watchdog Timer Counter Reset Register



U: Undefined

21.3.3 Stack pointer overflow/underflow reset

A reset signal is generated if the address or interrupt stack overflows or underflows.

Stack pointer overflow/underflow reset can be used to detect a program hang-up in the same manner as watchdog timer reset.

The reset signal is generated under the following conditions:

- Interrupt due to overflow or underflow of interrupt stack (4 levels)
- Interrupt due to overflow or underflow of address stack (15 levels)

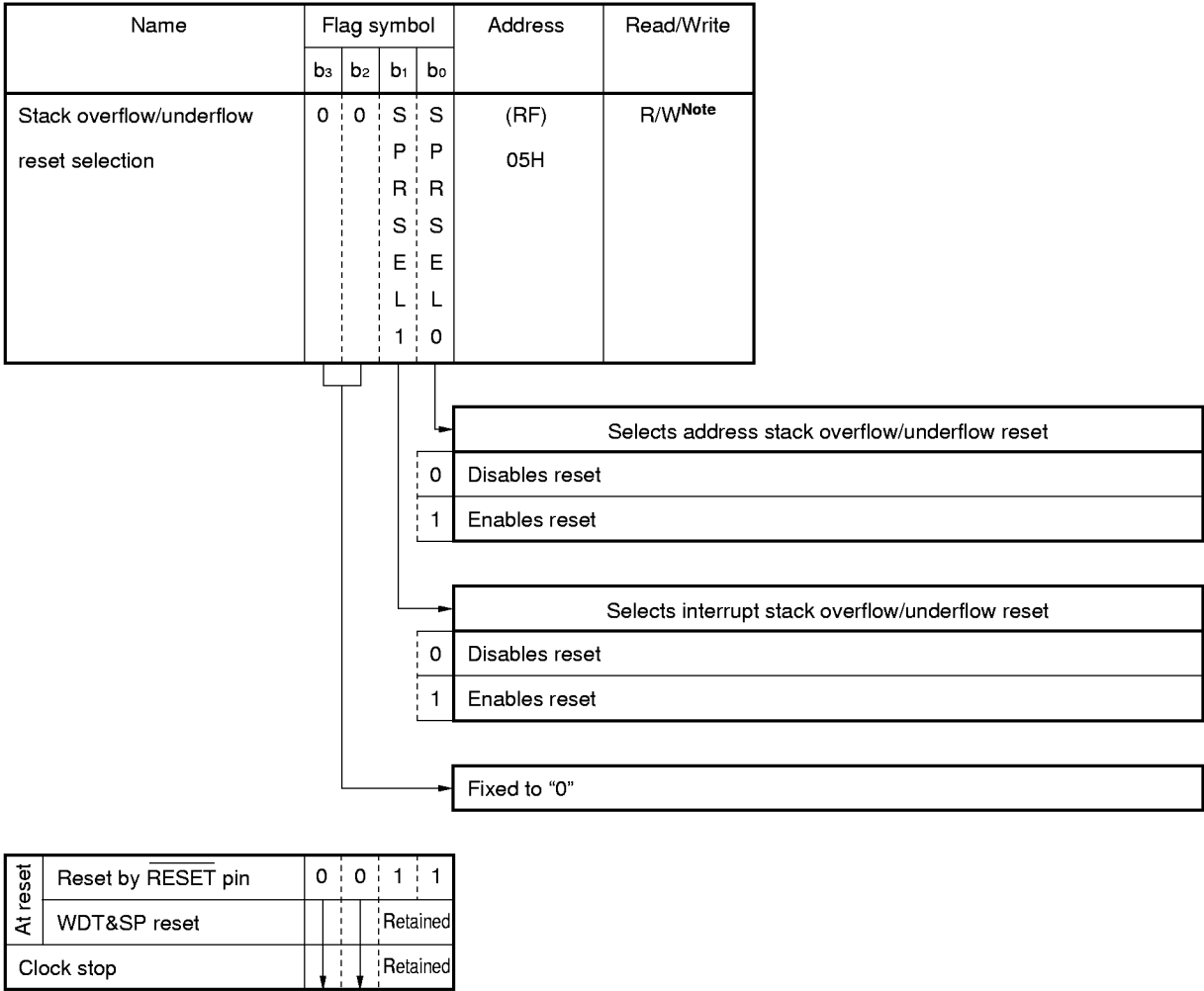
Reset by stack pointer overflow or underflow initializes all the registers, except the stack overflow selection register, watchdog timer counter reset register, and basic timer 0 carry register.

Generation of stack pointer overflow or underflow reset is detected by the WDTSPRES flag (R&Reset).

21.3.4 Stack pointer setting flag

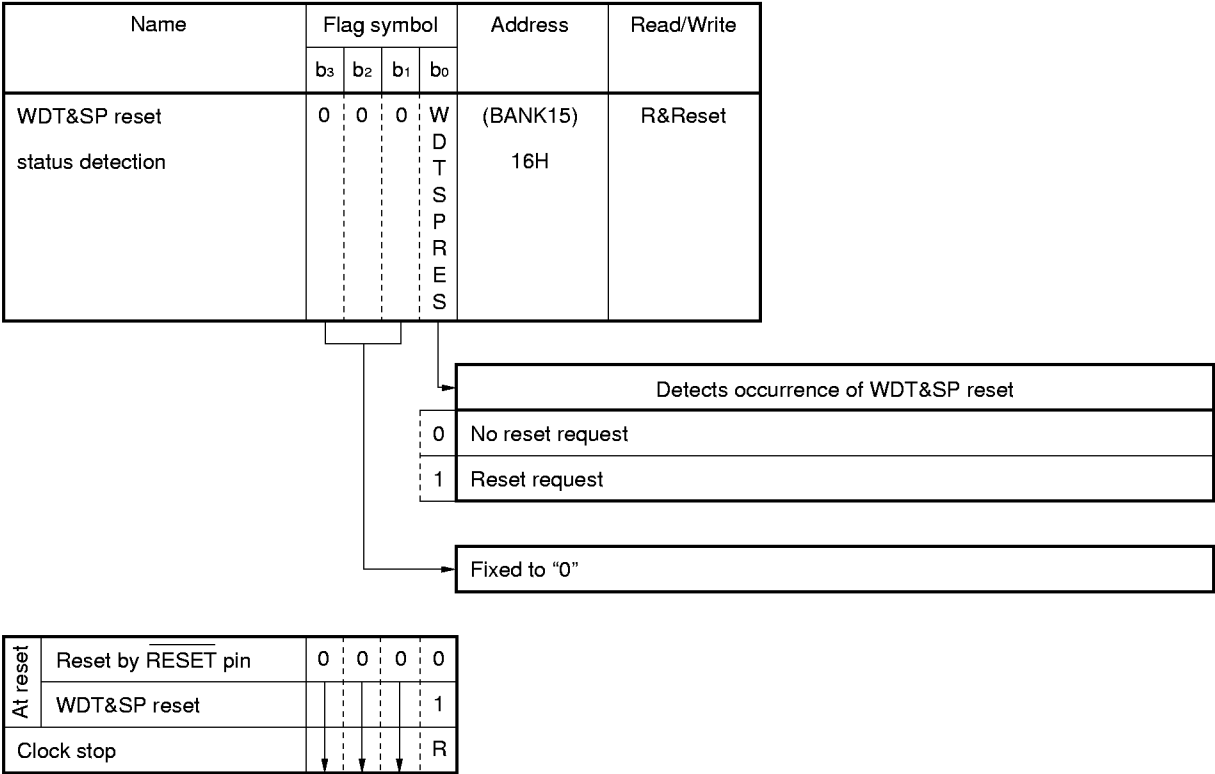
The stack overflow/underflow reset selection register can be set only once after reset by the $\overline{\text{RESET}}$ pin. This register specifies whether reset by address stack overflow or underflow and reset by interrupt stack overflow or underflow are enabled or disabled.

Figure 21-6. Configuration of Stack Overflow/Underflow Reset Selection Register



Note Can be written only once.

Figure 21-7. Configuration of WDT&SP Reset Selection Register



R: Retained

22. INSTRUCTION SET

22.1 Outline of Instruction Set

b ₁₄ -b ₁₁ \ b ₁₅		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	BR	addr (page 2)	SKT	m,#n4
1111	F	BR	addr (page 3)	SKF	m,#n

22.2 Legend

AR	: Address register
ASR	: Address stack register indicated by stack pointer
addr	: Program memory address (low-order 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 _R , m _C
m _R	: Data memory row address (high-order)
m _C	: Data memory column address (low-order)
n	: Bit position (4 bits)
n4	: Immediate data (4 bits)
PAGE	: Page (bits 12 and 11 of program counter)
PC	: Program counter
P	: Peripheral address
p _H	: Peripheral address (high-order 3 bits)
p _L	: Peripheral address (low-order 4 bits)
r	: General register column address
rf	: Register file address
rf _R	: Register file row address (high-order 3 bits)
rf _C	: Register file column address (low-order 4 bits)
SP	: Stack pointer
s	: Stop release condition
WR	: Window register
(x)	: Contents addressed by x

22.3 Instruction List

Instructions	Mnemonic	Operand	Operation	Instruction Code			
				Op code	Operand		
Add	ADD	r,m	$(r) \leftarrow (r) + (m)$	00000	m _R	m _C	r
		m,#n4	$(m) \leftarrow (m) + n4$	10000	m _R	m _C	n4
	ADDC	r,m	$(r) \leftarrow (r) + (m) + CY$	00010	m _R	m _C	r
		m,#n4	$(m) \leftarrow (m) + n4 + CY$	10010	m _R	m _C	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 _R	m _C	r
		m,#n4	$(m) \leftarrow (m) - n4$	10001	m _R	m _C	n4
	SUBC	r,m	$(r) \leftarrow (r) - (m) - CY$	00011	m _R	m _C	r
		m,#n4	$(m) \leftarrow (m) - n4 - CY$	10011	m _R	m _C	n4
Logical operation	OR	r,m	$(r) \leftarrow (r) \vee (m)$	00110	m _R	m _C	r
		m,#n4	$(m) \leftarrow (m) \vee n4$	10110	m _R	m _C	n4
	AND	r,m	$(r) \leftarrow (r) \wedge (m)$	00100	m _R	m _C	r
		m,#n4	$(m) \leftarrow (m) \wedge n4$	10100	m _R	m _C	n4
	XOR	r,m	$(r) \leftarrow (r) \oplus (m)$	00101	m _R	m _C	r
		m,#n4	$(m) \leftarrow (m) \oplus n4$	10101	m _R	m _C	n4
Judge	SKT	m,#n	$CMP \leftarrow 0$, if $(m) \wedge n = n$, then skip	11110	m _R	m _C	n
	SKF	m,#n	$CMP \leftarrow 0$, if $(m) \wedge n = 0$, then skip	11111	m _R	m _C	n
Compare	SKE	m,#n4	$(m) - n4$, skip if zero	01001	m _R	m _C	n4
	SKNE	m,#n4	$(m) - n4$, skip if not zero	01011	m _R	m _C	n4
	SKGE	m,#n4	$(m) - n4$, skip if not borrow	11001	m _R	m _C	n4
	SKLT	m,#n4	$(m) - n4$, skip if borrow	11011	m _R	m _C	n4
Rotate	RORC	r	$\rightarrow CY \leftarrow (r) b_3 \leftarrow (r) b_2 \leftarrow (r) b_1 \leftarrow (r) b_0 \leftarrow$	00111	000	0111	r
Transfer	LD	r,m	$(r) \leftarrow (m)$	01000	m _R	m _C	r
	ST	m,r	$(m) \leftarrow (r)$	11000	m _R	m _C	r
	MOV	@r,m	if MPE = 1 : $(MP, (r)) \leftarrow (m)$ if MPE = 0 : $(BANK, m_R, (r)) \leftarrow (m)$	01010	m _R	m _C	r
		m, @r	if MPE = 1 : $(m) \leftarrow (MP, (r))$ if MPE = 0 : $(m) \leftarrow (BANK, m_R, (r))$	11010	m _R	m _C	r
		m,#n4	$(m) \leftarrow n4$	11101	m _R	m _C	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
	GET	DBF,p	$DBF \leftarrow (p)$	00111	p _H	1011	p _L
	PUT	p,DBF	$(p) \leftarrow DBF$	00111	p _H	1010	p _L
	PEEK	WR,rf	$WR \leftarrow (rf)$	00111	rf _R	0011	rf _C
	POKE	rf,WR	$(rf) \leftarrow WR$	00111	rf _R	0010	rf _C

Instructions	Mnemonic	Operand	Operation	Instruction Code			
				Op code	Operand		
Branch	BR	addr	$PC_{10-0} \leftarrow \text{addr}, \text{PAGE} \leftarrow 0$	01100	addr		
			$PC_{10-0} \leftarrow \text{addr}, \text{PAGE} \leftarrow 1$	01101			
			$PC_{10-0} \leftarrow \text{addr}, \text{PAGE} \leftarrow 2$	01110			
			$PC_{10-0} \leftarrow \text{addr}, \text{PAGE} \leftarrow 3$	01111			
	@AR		$PC \leftarrow AR$	00111	000	0100	0000
Subroutine	CALL	addr	$SP \leftarrow SP - 1, ASR \leftarrow PC$ $PC_{11} \leftarrow 0, PC_{10-0} \leftarrow \text{addr}$	11100	addr		
		@AR	$SP \leftarrow SP - 1, ASR \leftarrow PC$ $PC \leftarrow AR$	00111	000	0101	0000
	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

22.4 Assembler (RA17K) Embedded Macro Instruction

Legend

- flag n : FLG symbol
n : Bit number
< > : Can be omitted

	Mnemonic	Operand	Operation	n
Embedded macro	SKTn	flag 1, ... flag n	if (flag1) ~ (flag n) = all "1", then skip	1 ≤ n ≤ 4
	SKFn	flag 1, ... flag n	if (flag 1) ~ (flag n) = all "0", then skip	1 ≤ n ≤ 4
	SETn	flag 1, ... flag n	(flag 1) ~ (flag n) ← 1	1 ≤ n ≤ 4
	CLRn	flag 1, ... flag n	(flag 1) ~ (flag n) ← 0	1 ≤ n ≤ 4
	NOTn	flag 1, ... flag n	if (flag n) = "0", then (flag n) ← 1 if (flag n) = "1", then (flag n) ← 0	1 ≤ n ≤ 4
	INITFLG	<NOT> flag 1, ... <<NOT> flag n>	if description = NOT flag n, then (flag n) ← 0 if description = flag n, then (flag n) ← 1	1 ≤ n ≤ 4
	BANKn		(BANK) ← n	0 ≤ n ≤ 15
Expanded instruction	BRX	Label	Jump Label	—
	CALLX	function-name	CALL sub-routine	—
	SYSCALX	function-name or expression	CALL system sub-routine	—
	INITFLGX	<NOT/INV> flag 1, ... <NOT/INV> flag n	if description = NOT (or INV) flag, (flag) ← 0 if description = flag, (flag) ← 1	n ≤ 4

23. RESERVED SYMBOLS

23.1 Data Buffer (DBF)

Symbol Name	Attribute	Value	R/W	Description
DBF3	MEM	0.0CH	R/W	Bits 15 through 12 of data buffer
DBF2	MEM	0.0DH	R/W	Bits 11 through 8 of data buffer
DBF1	MEM	0.0EH	R/W	Bits 7 through 4 of data buffer
DBF0	MEM	0.0FH	R/W	Bits 3 through 0 of data buffer

23.2 System Registers (SYSREG)

Symbol Name	Attribute	Value	R/W	Description
AR3	MEM	0.74H	R/W	Bits 15 through 12 of address register
AR2	MEM	0.75H	R/W	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	Bits 10 through 8 of index register
MPH	MEM	0.7AH	R/W	Bits 6 through 4 of memory pointer
MPE	FLG	0.7AH.3	R/W	Memory pointer enable flag
IXM	MEM	0.7BH	R/W	Bits 7 through 4 of index register
MPL	MEM	0.7BH	R/W	Bits 3 through 0 of memory pointer
IXL	MEM	0.7CH	R/W	Bits 3 through 0 of index register
RPH	MEM	0.7DH	R/W	Bits 6 through 3 of general register pointer
RPL	MEM	0.7EH	R/W	Bits 2 through 0 of general register pointer
BCD	FLG	0.7EH.0	R/W	BCD operation flag
PSW	MEM	0.7FH	R/W	Program status word
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

23.3 LCD Segment Register

Symbol Name	Attribute	Value	R/W	Description
LCDD19	MEM	14.5CH	R/W	LCD segment register
LCDD18	MEM	14.5DH	R/W	LCD segment register
LCDD17	MEM	14.5EH	R/W	LCD segment register
LCDD16	MEM	14.5FH	R/W	LCD segment register
LCDD15	MEM	14.60H	R/W	LCD segment register
LCDD14	MEM	14.61H	R/W	LCD segment register
LCDD13	MEM	14.62H	R/W	LCD segment register
LCDD12	MEM	14.63H	R/W	LCD segment register
LCDD11	MEM	14.64H	R/W	LCD segment register
LCDD10	MEM	14.65H	R/W	LCD segment register
LCDD9	MEM	14.66H	R/W	LCD segment register
LCDD8	MEM	14.67H	R/W	LCD segment register
LCDD7	MEM	14.68H	R/W	LCD segment register
LCDD6	MEM	14.69H	R/W	LCD segment register
LCDD5	MEM	14.6AH	R/W	LCD segment register
LCDD4	MEM	14.6BH	R/W	LCD segment register
LCDD3	MEM	14.6CH	R/W	LCD segment register
LCDD2	MEM	14.6DH	R/W	LCD segment register
LCDD1	MEM	14.6EH	R/W	LCD segment register
LCDD0	MEM	14.6FH	R/W	LCD segment register

23.4 Port Register

Symbol Name	Attribute	Value	R/W	Description
P0A1	FLG	0.70H.1	R/W	Port 0A bit 1
P0A0	FLG	0.70H.0	R/W	Port 0A bit 0
P0B3	FLG	0.71H.3	R/W	Port 0B bit 3
P0B2	FLG	0.71H.2	R/W	Port 0B bit 2
P0B1	FLG	0.71H.1	R/W	Port 0B bit 1
P0B0	FLG	0.71H.0	R/W	Port 0B bit 0
P0C3	FLG	0.72H.3	R/W	Port 0C bit 3
P0C2	FLG	0.72H.2	R/W	Port 0C bit 2
P0C1	FLG	0.72H.1	R/W	Port 0C bit 1
P0C0	FLG	0.72H.0	R/W	Port 0C bit 0
P0D3	FLG	0.73H.3	R Note	Port 0D bit 3
P0D2	FLG	0.73H.2	R Note	Port 0D bit 2
P0D1	FLG	0.73H.1	R Note	Port 0D bit 1
P0D0	FLG	0.73H.0	R Note	Port 0D bit 0
P1A3	FLG	1.70H.3	R/W	Port 1A bit 3
P1A2	FLG	1.70H.2	R/W	Port 1A bit 2
P1A1	FLG	1.70H.1	R/W	Port 1A bit 1
P1A0	FLG	1.70H.0	R/W	Port 1A bit 0
P1C3	FLG	1.72H.3	R Note	Port 1C bit 3
P1C2	FLG	1.72H.2	R Note	Port 1C bit 2
P1C1	FLG	1.72H.1	R Note	Port 1C bit 1
P1C0	FLG	1.72H.0	R Note	Port 1C bit 0
P1D3	FLG	1.73H.3	R/W	Port 1D bit 3
P1D2	FLG	1.73H.2	R/W	Port 1D bit 2
P1D1	FLG	1.73H.1	R/W	Port 1D bit 1
P1D0	FLG	1.73H.0	R/W	Port 1D bit 0
P2A2	FLG	2.70H.2	R/W	Port 2A bit 2
P2A1	FLG	2.70H.1	R/W	Port 2A bit 1
P2A0	FLG	2.70H.0	R/W	Port 2A bit 0
P2B3	FLG	2.71H.3	R/W	Port 2B bit 3
P2B2	FLG	2.71H.2	R/W	Port 2B bit 2
P2B1	FLG	2.71H.1	R/W	Port 2B bit 1
P2B0	FLG	2.71H.0	R/W	Port 2B bit 0
P2C3	FLG	2.72H.3	R/W	Port 2C bit 3
P2C2	FLG	2.72H.2	R/W	Port 2C bit 2
P2C1	FLG	2.72H.1	R/W	Port 2C bit 1
P2C0	FLG	2.72H.0	R/W	Port 2C bit 0

Note These ports are input-only ports. The assembler and in-circuit emulator will not output error messages even if an instruction to output from these ports is written. Also, if the instruction is actually executed on a device, the operation will have no change.

★ 23.5 Register File (Control Register)

Symbol Name	Attribute	Value	R/W	Description
SP	MEM	0.81H	R/W	Stack pointer
DBFSP	MEM	0.84H	R	DBF stack pointer
SPRSEL	MEM	0.85H	R/W	Stack overflow select flag (Settable only once after power application)
MOVTSEL1	FLG	0.87H.1	R/W	MOVT bit select flag
MOVTSEL0	FLG	0.87H.0	R/W	MOVT bit select flag
SYSRSP	MEM	0.88H	R	System register stack pointer
WDTCK	MEM	15.02H	R/W	Watchdog timer clock select flag
WDTRES	FLG	15.03H.3	R/W	Watchdog timer count reset
PLLSCNF	FLG	15.10H.3	R/W	Swallow counter MSB set flag
PLLM1D1	FLG	15.10H.1	R/W	PLL mode select flag
PLLM1D0	FLG	15.10H.0	R/W	PLL mode select flag
PLLR1CK3	FLG	15.11H.3	R/W	PLL reference frequency select flag
PLLR1CK2	FLG	15.11H.2	R/W	PLL reference frequency select flag
PLLR1CK1	FLG	15.11H.1	R/W	PLL reference frequency select flag
PLLR1CK0	FLG	15.11H.0	R/W	PLL reference frequency select flag
PLLUL	FLG	15.12H.0	R	PLL unlock FF flag
BEEP0SEL	FLG	15.14H.2	R/W	BEEP0 enable flag
BEEP0CK1	FLG	15.14H.1	R/W	BEEP0 clock select flag
BEEP0CK0	FLG	15.14H.0	R/W	BEEP0 clock select flag
WDTCY	FLG	15.16H.0	R	Watchdog timer/stack pointer reset status detection flag
BTM0CY	FLG	15.17H.0	R	Basic timer 0 carry flag
BTM1CK0	FLG	15.18H.0	R/W	Basic timer 1 clock select flag
SIO1CK1	FLG	15.1CH.1	R/W	Serial interface 1 I/O clock select flag
SIO1CK0	FLG	15.1CH.0	R/W	Serial interface 1 I/O clock select flag
SIO1MOD	FLG	15.1DH.2	R/W	Serial interface 1 SI1/SO2 select flag
SIO1HIZ	FLG	15.1DH.1	R/W	Serial interface 1 general-purpose port select flag
SIO1TS	FLG	15.1DH.0	R/W	Serial interface 1 transmit/receive start flag
IEG0	FLG	15.1FH.0	R/W	INT0 pin interrupt request detection edge direction select flag
IFCG0STT	FLG	15.21H.0	R	IF counter gate status detection flag (1: open, 0: close)
IFCMD1	FLG	15.22H.3	R/W	IF counter mode select flag (10: FMIFC, 11: AMIFC2)
IFCMD0	FLG	15.22H.2	R/W	IF counter mode select flag (00: FCG, 01: AMIFC)
IFCCK1	FLG	15.22H.1	R/W	IF counter clock select flag
IFCCK0	FLG	15.22H.0	R/W	IF counter clock select flag
IFCSTRT	FLG	15.23H.1	W	IF counter count start
IFCRES	FLG	15.23H.0	R/W	IF counter reset

Symbol Name	Attribute	Value	R/W	Description
ADCCH3	FLG	15.24H.3	R/W	A/D converter channel select flag
ADCCH2	FLG	15.24H.2	R/W	A/D converter channel select flag
ADCCH1	FLG	15.24H.1	R/W	A/D converter channel select flag
ADCCH0	FLG	15.24H.0	R/W	A/D converter channel select flag
ADCSTRT	FLG	15.25H.1	R/W	A/D converter compare start flag
ADCCMP	FLG	15.25H.0	R	A/D converter compare result detection flag
TM0EN	FLG	15.2BH.3	R/W	Modulo timer 0 count start flag
TM0RES	FLG	15.2BH.2	R/W	Modulo timer 0 reset flag (The value is 0 when read)
TM0CK1	FLG	15.2BH.1	R/W	Modulo timer 0 clock select flag (10: TM10, 11: TM11)
TM0CK0	FLG	15.2BH.0	R/W	Modulo timer 0 clock select flag (00: 75 kHz, 01: 25 kHz)
TM0OVF	FLG	15.2CH.3	R	Modulo timer 0 overflow detection flag
IPSIO1	FLG	15.2FH.3	R/W	Serial interface 1 interrupt enable flag
IPBTM1	FLG	15.2FH.2	R/W	Basic timer 1 interrupt enable flag
IPTM0	FLG	15.2FH.1	R/W	Modulo timer 0 interrupt enable flag
IP0	FLG	15.2FH.0	R/W	INT0 pin interrupt enable flag
IRQSIO1	FLG	15.3CH.0	R/W	Serial interface 1 interrupt request detection flag
IRQBTM1	FLG	15.3DH.0	R/W	Basic timer 1 interrupt request detection flag
IRQTM0	FLG	15.3EH.0	R/W	Modulo timer 0 interrupt request detection flag
INT0	FLG	15.3FH.3	R/W	INT0 pin status detection flag
IRQ0	FLG	15.3FH.0	R/W	INT0 pin interrupt request detection flag
LCDEN	FLG	15.40H.0	R/W	LCD driver display start flag
LCD19SEL	FLG	15.69H.2	R/W	P2A2/LCD19 switching flag
LCD18SEL	FLG	15.69H.1	R/W	P2A1/LCD18 switching flag
LCD17SEL	FLG	15.69H.0	R/W	P2A0/LCD17 switching flag
PODPLD3	FLG	15.6AH.3	R/W	POD3 pin pull-down resistor switching flag
PODPLD2	FLG	15.6AH.2	R/W	POD2 pin pull-down resistor switching flag
PODPLD1	FLG	15.6AH.1	R/W	POD1 pin pull-down resistor switching flag
PODPLD0	FLG	15.6AH.0	R/W	POD0 pin pull-down resistor switching flag
P2CBIO3	FLG	15.6BH.3	R/W	P2C3 I/O select flag
P2CBIO2	FLG	15.6BH.2	R/W	P2C2 I/O select flag
P2CBIO1	FLG	15.6BH.1	R/W	P2C1 I/O select flag
P2CBIO0	FLG	15.6BH.0	R/W	P2C0 I/O select flag
P2BBIO3	FLG	15.6CH.3	R/W	P2B3 I/O select flag
P2BBIO2	FLG	15.6CH.2	R/W	P2B2 I/O select flag
P2BBIO1	FLG	15.6CH.1	R/W	P2B1 I/O select flag
P2BBIO0	FLG	15.6CH.0	R/W	P2B0 I/O select flag

Symbol Name	Attribute	Value	R/W	Description
P1DBIO3	FLG	15.6DH.3	R/W	P1D3 I/O select flag
P1DBIO2	FLG	15.6DH.2	R/W	P1D2 I/O select flag
P1DBIO1	FLG	15.6DH.1	R/W	P1D1 I/O select flag
P1DBIO0	FLG	15.6DH.0	R/W	P1D0 I/O select flag
P1ABIO3	FLG	15.6EH.3	R/W	P1A3 I/O select flag
P1ABIO2	FLG	15.6EH.2	R/W	P1A2 I/O select flag
P1ABIO1	FLG	15.6EH.1	R/W	P1A1 I/O select flag
P1ABIO0	FLG	15.6EH.0	R/W	P1A0 I/O select flag
P0BBIO3	FLG	15.6FH.3	R/W	P0B3 I/O select flag
P0BBIO2	FLG	15.6FH.2	R/W	P0B2 I/O select flag
P0BBIO1	FLG	15.6FH.1	R/W	P0B1 I/O select flag
P0BBIO0	FLG	15.6FH.0	R/W	P0B0 I/O select flag

23.6 Peripheral Hardware Register

Symbol Name	Attribute	Value	R/W	Description
ADCR	DAT	02H	R/W	A/D converter reference voltage set register
SIO1SFR	DAT	04H	R/W	Serial interface 1 presetable shift register
TM0M	DAT	1AH	R/W	Timer modulo 0 register
TM0C	DAT	1BH	R	Timer modulo 0 counter
AR	DAT	40H	R/W	Address register
DBFSTK	DAT	41H	R/W	DBF stack register
PLL	DAT	42H	R/W	PLL data register
IFC	DAT	43H	R	IF counter data register

23.7 Others

Symbol Name	Attribute	Value	Description
DBF	DAT	0FH	Operand (DBF) for GET/PUT/MOVT/MOVTH/MOVTI instruction
IX	DAT	01H	Operand (IX) for INC instruction
AR_EPA1	DAT	8040H	Operand (EPA bit ON) for CALL/BR/MOVT/MOVTH/MOVTI instruction
AR_EPA0	DAT	4040H	Operand (EPA bit OFF) for CALL/BR/MOVT/MOVTH/MOVTI instruction

★ 24. ELECTRICAL CHARACTERISTICS

Absolute Maximum Ratings (T_A = 25 °C)

Parameter	Symbol	Condition	Rating	Unit
Supply voltage	V _{DD0}		−0.3 to +2.0	V
	V _{DD1}		−0.3 to +2.0	V
	V _{DD2}		−0.3 to +2.0	V
Input voltage	V _{I1}	P0D0-P0D3, P1C0-P1C3 pins	−0.3 to V _{DD2} +0.3	V
	V _{I2}	VCOH and VCOL pins	−0.3 to V _{DD1} +0.3	V
	V _{I3}	P0B0-P0B3, P1A0-P1A3, P1D0-P1D3, P2A0-P2A2, P2B0-P2B3, P2C0-P2C3, <u>RESET</u> , and INT pins	−0.3 to V _{DD0} +0.3	V
Output voltage	V _{O1}	P0B0-P0B3, P0C0-P0C3, P2B0-P2B3, P2C0-P2C3	−0.3 to V _{DD0} +0.3	V
	V _{O2}	EO1 and EO2 pins	−0.3 to REG _{LCD1} +0.3	V
	V _{O3}	LCD0-LCD19, COM0-COM3 pins	−0.3 to REG _{LCD2} +0.3	V
High-level output current	I _{OH}	1 pin	−3.0	mA
		Total of P0B0-P0B3, P0C0-P0C3, P2B0-P2B3, and P2C0-P2C3	−30.0	mA
Low-level output current	I _{OL1}	1 pin of P0A0, P0A1, and P1D0-P1D3	10.0	mA
	I _{OL2}	1 pin other P0A0, P0A1, and P1D0-P1D3	3.0	mA
		Total of P0A0, P0A1, P0B0-P0B3, P0C0-P0C3, P1A0-P1A3, P1D0-P1D3, P2B0-P2B3, and P2C0-P2C3	45.0	mA
Output voltage	V _{BDS}	P0A0, P0A1, P1A0-P1A3, P1D0-P1D3	−0.3 to +4.0	V
Operating ambient temperature	T _A		−10 to +50	°C
Storage temperature	T _{stg}		−55 to +125	°C

Caution If the absolute maximum rating of even one of the above parameters is exceeded even momentarily, the quality of the product may be degraded. The absolute maximum ratings, therefore, specify the value exceeding which the product may be physically damaged. Be sure to use the product with these ratings never being exceeded.

Recommended Supply Voltage Range (T_A = -10 to +50 °C)

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Supply voltage	V _{DD0}		1.05		1.8	V
	V _{DD1}		1.05		1.8	V
	V _{DD2}		1.05		1.8	V

Recommended Output Voltage (T_A = -10 to +50 °C)

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Output voltage	V _{BDS}	P0A0, P0A1, P1A0-P1A3, P1D0-P1D3	-0.3		+4.0	V

★ DC Characteristics (T_A = -10 to +50 °C, V_{DD} = V_{DD0} = V_{DD1} = V_{DD2} = 1.05 to 1.8 V)

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Supply current	I _{DD11}	CPU operation (V _{DD0} = 1.8 V, T _A = 25 °C)		120	320	μA
	I _{DD12}	HALT operation (V _{DD0} = 1.8 V, T _A = 25 °C)		70	240	μA
	I _{DD2}	PLL operation (VCOH pin, f _{IN} = 130 MHz, V _{IN} = 0.5 V _{P-P} , V _{DD} = 1.8 V)		7.0	10.0	mA
	I _{DD3}	A/D converter, IF counter operation (f _{IN} = 10.7 MHz, V _{IN} = 0.3 V _{P-P} , V _{DD2} = 1.8 V)		3.0	7.0	mA
High-level input voltage	V _{IH1}	P0B0-P0B3, P1A0-P1A3, P1C0-P1C3, P1D0-P1D3, P2A0-P2A2, P2B0-P2B3, P2C0-P2C3, $\overline{\text{RESET}}$, INT	0.8 V _{DD}		V _{DD}	V
	V _{IH2}	P0D0-P0D3	0.8 V _{DD}		V _{DD}	V
Low-level input voltage	V _{IL1}	P0B0-P0B3, P1A0-P1A3, P1C0-P1C3, P1D0-P1D3, P2A0-P2A2, P2B0-P2B3, P2C0-P2C3, $\overline{\text{RESET}}$, INT	0		0.1 V _{DD}	V
	V _{IL2}	P0D0-P0D3	0		0.1 V _{DD}	V
High-level output voltage	V _{OH1}	P0B0-P0B3, P0C0-P0C3, P2B0-P2B3, P2C0-P2C3			V _{DD0}	V
	V _{OH2}	EO0, EO1			REG _{LCD1}	V
	V _{OH3}	LCD0-LCD19, COM0-COM3			REG _{LCD2}	V
High-level input current	I _{IH1}	With P0D0-P0D3 pulled down	V _{IH} = V _{DD0} = 1.1 V	2	30	μA
			V _{IH} = V _{DD0}	2		μA
Low-level input current	I _{IL1}	With X _{IN} pin pulled up	V _{IH} = V _{DD0} = 1.1 V	-2	-30	μA
			V _{IH} = V _{DD0}	-2		μA
High-level output current	I _{OH1}	P0B0-P0B3, P0C0-P0C3, P2B0-P2B3, P2C0-P2C3 (V _{OH} = V _{DD1} - 0.2 V)	-0.13			mA
	I _{OH2}	EO0, EO1 (V _{OH} = V _{DD2} - 0.2 V)	-0.13			mA
	I _{OH3}	LCD0-LCD19 (V _{OH} = LCD _{REG2} - 0.2 V)	-1			μA
	I _{OH4}	COM0-COM3 (V _{OH} = LCD _{REG2} - 0.2 V)	-10			μA
Low-level output current	I _{OL1}	P0B0-P0B3, P0C0-P0C3, P2B0-P2B3, P2C0-P2C3 (V _{OL} = 0.2 V)	0.2			mA
	I _{OL2}	EO0, EO1 (V _{OL} = 0.2 V)	0.2			mA
	I _{OL3}	P1A0-P1A3 (V _{OL} = 0.2 V)	0.4			mA
	I _{OL4}	P0A0, P1A1, P1D0-P1D3 (V _{OL} = 0.2 V)	6.0			mA
	I _{OL5}	LCD0-LCD19 (V _{OL} = 0.2 V)	1			μA
	I _{OL6}	COM0-COM3 (V _{OL} = 0.2 V)	10			μA
LCD drive voltage	V _{LCD0}	LCD0-LCD19 output open, C1-C5 = 0.1 μF	1.4	1.5	1.6	V
	V _{LCD1}	T _A = 25 °C	2.8	3.0	3.2	V
Output off leakage current	I _{L1}	P0A0, P0A1, P1A0-P1A3, P1D0-P1D3 (V _{OH} = 1.8 V)			1	μA
	I _{L2}	EO0, EO1 (V _{OH} = 1.8 V, V _{OL} = 0 V)			±1	μA
COM intermediate potential output voltage	V _M	COM0-COM3 (output open)	V _{LCD0} -0.2	V _{LCD0}	V _{LCD0} +0.2	V
COM intermediate potential output current	I _{OM}	COM0-COM3 (V _{OM} = V _M ± 0.2 V)	±1			μA

Remark Unless otherwise specified, the characteristics of muxed pins are the same as those of the port pins.

AC Characteristics ($T_A = -10$ to $+50$ °C, $V_{DD} = V_{DD0} = V_{DD1} = V_{DD2} = 1.05$ to 1.8 V)

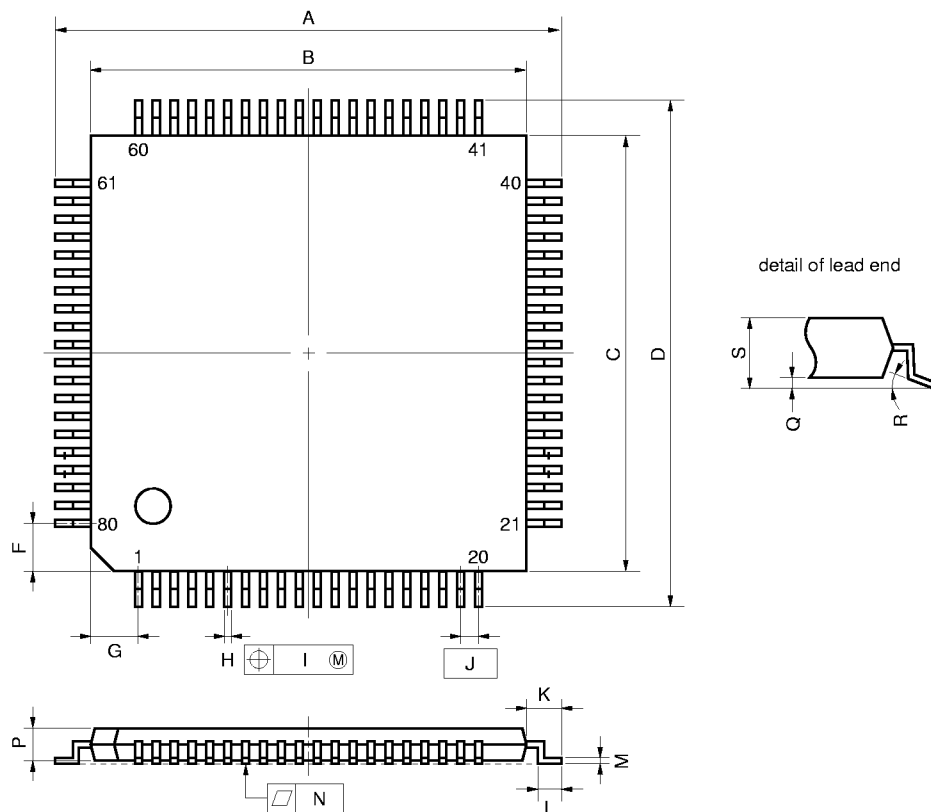
Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Operating frequency	f _{IN1}	VHF mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	35		220	MHz
	f _{IN2}	HF mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	5		40	MHz
	f _{IN3}	MF mode, sine wave input $V_{IN} = 0.3 V_{P-P}$	0.8		3.5	MHz
	f _{IN4}	AMIFC pin, sine wave input $V_{IN} = 0.3 V_{P-P}$	0.4		2	MHz
	f _{IN5}	AMIFC pin, sine wave input $V_{IN} = 0.3 V_{P-P}$	0.4		2	MHz
	f _{IN6}	FMIFC pin, sine wave input $V_{IN} = 0.3 V_{P-P}$	10		11	MHz
SCK input frequency	f _{IN7}	External clock			75	kHz

A/D Converter Characteristics ($T_A = -10$ to $+50$ °C, $V_{DD} = V_{DD0} = V_{DD1} = V_{DD2} = 1.05$ to 1.8 V)

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Total conversion error		$V_{DD2} = 1.05$ to 1.8 V			±3	LSB

25. PACKAGE DRAWING

80 PIN PLASTIC TQFP (FINE PITCH) (□12)



NOTE

Each lead centerline is located within 0.10 mm (0.004 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
A	14.0±0.2	0.551±0.008
B	12.0±0.2	0.472 ^{+0.009} _{-0.008}
C	12.0±0.2	0.472 ^{+0.009} _{-0.008}
D	14.0±0.2	0.551±0.008
F	1.25	0.049
G	1.25	0.049
H	0.22±0.05	0.009 ^{+0.002} _{-0.003}
I	0.10	0.004
J	0.5 (T.P.)	0.020 (T.P.)
K	1.0±0.2	0.039 ^{+0.009} _{-0.008}
L	0.5±0.2	0.020 ^{+0.008} _{-0.009}
M	0.145±0.05	0.006 ^{+0.002} _{-0.003}
N	0.10	0.004
P	1.0±0.05	0.040 ^{+0.002} _{-0.003}
Q	0.1±0.05	0.004±0.002
R	3 [°] _{-3°}	3 [°] _{-3°}
S	1.2 MAX.	0.048 MAX.

S80GK-50-9EU

★ 26. RECOMMENDED SOLDERING CONDITIONS

Solder the μPD17933 and μPD17934 under the following recommended conditions.

For details of the recommended soldering conditions, refer to information document **Semiconductor Device Mounting Technology Manual (C10535E)**.

For soldering methods and conditions other than those recommended, consult NEC.

Table 26-1. Soldering conditions for surface-mount type

μPD17933GK-xxx-BE9: 80-pin plastic TQFP (12 mm × 12 mm, 0.5-mm pitch)

μPD17934GK-xxx-BE9: 80-pin plastic TQFP (12 mm × 12 mm, 0.5-mm pitch)

Soldering Method	Soldering Conditions	Recommended Conditions Symbol
Infrared reflow	Package peak temperature: 235 °C; Time: 30 secs. max. (210 °C min.); Number of times: twice max.; Number of days: 7 ^{Note} (after that, prebaking is necessary at 125 °C for 10 hours) <Precaution> Products other than in heat-resistance trays (such as those packaged in a magazine, taping, or non-heat-resistance tray) cannot be baked while they are in their package.	IR35-107-2
VPS	Package peak temperature: 215 °C; Time: 40 secs. max. (200 °C min.); Number of times: twice max.; Number of days: 7 ^{Note} (after that, prebaking is necessary at 125 °C for 10 hours) <Precaution> Products other than in heat-resistance trays (such as those packaged in a magazine, taping, or non-heat-resistance tray) cannot be baked while they are in their package.	VP15-107-2
Partial heating	Pin temperature: 300 °C max.; Time: 3 secs. max. (per device side)	—

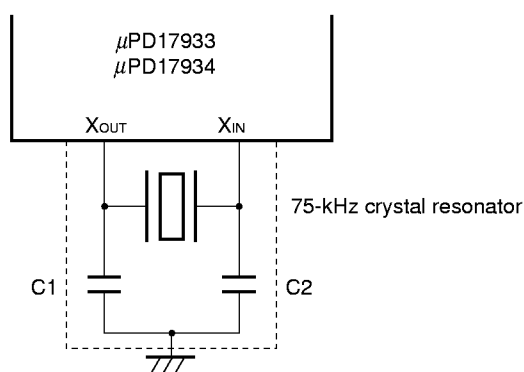
Note Number of days in storage after the dry pack has been opened. The storage conditions are at 25 °C, 65 % RH max.

Caution Do not use two or more soldering methods in combination (except partial heating).

APPENDIX A. CAUTIONS ON CONNECTING CRYSTAL RESONATOR

When using the system clock oscillation circuit, wire the portion enclosed by the dotted line in the figure below as follows to prevent adverse influence from wiring capacity.

- Keep the wiring length as short as possible.
- If capacitances C1 and C2 are too high, the oscillation start characteristics may be degraded or current consumption may increase.
- Generally, connect a trimmer capacitor for adjusting the oscillation frequency to the X_{IN} pin. Depending on the crystal resonator to be used, however, the oscillation stability differs. Therefore, evaluate the crystal resonator actually used.
- The crystal oscillation frequency cannot be accurately adjusted when an emulation probe is connected to the X_{OUT} and X_{IN} pin, because of the capacitance of the probe. Adjust the frequency while measuring the VCO oscillation frequency.



APPENDIX B. DEVELOPMENT TOOLS

The following development tools are available for development of programs for the μPD17933 and 17934.

Hardware

Name	Outline
In-circuit emulator (IE-17K IE-17K-ET ^{Note 1} EMU-17K ^{Note 2})	IE-17K, IE-17K-ET, and EMU-17K are in-circuit emulators that can be used with any model in the 17K series. IE-17K and IE-17K-ET are connected to a host machine, which is PC-9800 series or IBM PC/AT TM , with RS-232C. EMU-17K is mounted to the expansion slot of a host machine, PC-9800 series. By using these in-circuit emulators with a system evaluation board (SE board) corresponding to each model, these emulators operate dedicated to the model. When man-machine interface software <i>SIMPLEHOST</i> TM is used, a more sophisticated debugging environment can be created. EMU-17K also has a function to allow you to check the contents of the data memory real-time.
SE board (SE-17934)	SE-17934 is an SE board for the μPD17933 and 17934. This board can be used alone to evaluate a system, or in combination with an in-circuit emulator for debugging.
Emulation probe (EP-17K80GK)	EP-17K80GK is an emulation probe for the μPD17933 and 17934. By using this probe with TGK-080SDP ^{Note 3} , the SE board and target system are connected.
Conversion socket (TGK-080SDP ^{Note 3})	TGK-080SDP is a conversion socket for 80-pin plastic TQFP (12 × 12 mm). It is used to connect EP-17K80GK and target system.

- Notes**
1. Low-price model: external power supply type
 2. This is a product of I.C Corp. For details, consult I.C Corp. (03-3447-3793).
 3. A product of TOKYO Eletech Corp. (03-5295-1661). Consult your NEC distributor when purchasing the product.

Remark Third party PROM programmers AF-9703, AF-9704, AF-9705, and AF-9706 are available from Ando Electric Co., Ltd. Use these programmers with programmer adapter PA-17P709GC. For details, consult Ando Electric Co., Ltd. (03-3733-1163).

★ Software

Name	Outline	Host Machine	OS		Parts Number
17K series assembler (RA17K)	RA17K is a relocatable assembler that can be commonly used with 17K series. To develop programs for the μPD17933 and 17934, this RA17K and a device file (AS17934) are used in combination. The RA17K is provided with linker (LK17K) and document processor (DOC17K) utility.	PC-9800 series	MS-DOS™		μS5A13RA17K
		IBM PC/AT	PC DOS™		μS7B13RA17K
17K series C-like compiler (em1C-17K™)	em1C-17K is a C-like compiler that can be used with all models on the 17K series. It is used together with RA17K.	PC-9800 series	MS-DOS		μS5A13CC17K
		IBM PC/AT	PC DOS		μS7B13CC17K
Device file (AS17934)	AS17934 is a device file for the μPD17933 and 17934. It is used together with the assembler common to the 17K series (RA17K).	PC-9800 series	MS-DOS		μS5A13AS17707
		IBM PC/AT	PC DOS		μS7B13AS17707
Support software (SIMPLEHOST)	SIMPLEHOST is man-machine interface software that runs on Windows™ when a program is developed by using an in-circuit emulator and personal computer.	PC-9800 series	MS-DOS	Windows	μS5A13IE17K
		IBM PC/AT	PC DOS		μS7B13IE17K

Remark The version of the supported OS is as follows:

OS	Version
MS-DOS	Ver.3.30 to Ver.5.00A ^{Note}
PC DOS	Ver.3.1 to Ver.5.0 ^{Note}
Windows	Ver.3.0 to Ver.3.1

Note MS-DOS Ver. 5.00/5.00A and PC DOS Ver. 5.0 have a task swap function, but this function cannot be used with this software.

[MEMO]

[MEMO]

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 device 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: 01-504-2787
Fax: 01-504-2860

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: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.

Cumbica-Guarulhos-SP, Brasil
Tel: 011-6465-6810
Fax: 011-6465-6829

emIC-17K and SIMPLEHOST are trademarks of NEC Corporation.

MS-DOS and Windows are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

PC/AT and PC DOS are trademarks of IBM 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.

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.

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: Aircrafts, 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.

Anti-radioactive design is not implemented in this product.