NEC

MOS INTEGRATED CIRCUIT

μPD17016, 17017

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

The μPD17016 and 17017 are 4-bit single-chip CMOS microcontrollers equipped with hardware for digital tuning systems.

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

As the peripheral hardware, a prescaler that can operate at up to 150 MHz, PLL frequency synthesizer, and frequency counter for digital tuning systems, as well as many I/O ports, an LCD controller/driver, A/D converter, and D/A converter (PWM output) are provided.

Therefore, a high-performance, state-of-the-art digital tuning system can be organized with a single chip.

In addition to the μPD17016 and 17017, a one-time PROM model for program evaluation, the μPD17P005, is also available.

FEATURES

- 17K architecture: general-purpose register method
- Program memory (ROM)
  8 KB (3836 × 16 bits) : μPD17016
  16 KB (7932 × 16 bits) : μPD17017
- General-purpose data memory (RAM)
  320 × 4 bits: μPD17016
  432 × 4 bits: μPD17017
- Instruction execution time
  4.44 μs (with 4.5-MHz crystal resonator)
- Decimal operation
- Table reference
- Hardware for PLL frequency synthesizer
  Dual modulus prescaler (150 MHz max.), programmable divider, charge pump
- A wealth of peripheral hardware
  General-purpose I/O ports, LCD controller/driver, serial interface, A/D converter, D/A converter (PWM output), BEEP output, frequency counter
- Interrupt
  External: 1
  Internal: 1
- Power-ON reset, reset by CE pin, and power failure detection circuit
- Power-saving CMOS
- Supply voltage: V_{DD} = 5 V±10%

Unless otherwise specified, the μPD17017 is explained as the representative model in this document.

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

Document No. U10664EJ2V0DS00 (2nd edition)
Date Published September 1996 P
Printed in Japan

The mark ★ shows major revised points.

ORDERING INFORMATION

Part Number	Package
μPD17016GF-×××-3B9	80-pin plastic QFP (14 × 20 mm)
μPD17017GF-×××-3B9	80-pin plastic QFP (14 × 20 mm)

Remark ××× indicates a ROM code number.

FUNCTIONAL OUTLINE OF μPD17016 AND 17017

Item	Function
Program memory (ROM)	• 8 KB (3836 × 16 bits) : μPD17016 • 16 KB (7932 × 16 bits) : μPD17017 All internal ROM areas can be referenced as a table.
General-purpose data memory (RAM)	• 320 × 4 bits : μPD17016 • 432 × 4 bits : μPD17017 General register: 16 × 4 bits
Instruction execution time	• 4.44 μs (with 4.5-MHz crystal resonator)
Stack level	• 7 levels (stack can be manipulated)
General-purpose port	• I/O port : 16 pins • Input port : 8 pins • Output port : 9 pins (+8: LCD segment pin)
BEEP output	• 1 pin Selectable frequency (200 Hz, 1 kHz, 3 kHz)
LCD controller/driver	• 30 segments, 2 commons 1/2 duty, 1/2 bias, frame frequency: 125 MHz, drive voltage: V_{DD}, segment pins multiplexed with key source pins: 16 8 pins can be used as output port pins (4 pins can be set in output mode at one time)
Serial interface	• 1 channel 3-wire (serial I/O)
D/A converter	• 8 bits × 2 channels (PWM output, output voltage: 16 V MAX.)
A/D converter	• 6 bits × 6 channels (successive approximation by software)
Interrupt	• 2 sources (maskable interrupt) External : 1 source (INT_0 pin) Internal : 1 source (timer)
Timer	• 2 channels Timer carry (1, 5, 100, 250 ms) Timer interrupt (1, 5, 100, 250 ms)
Reset	• Power-ON reset (on power application) • Reset by CE pin (CE pin low level → high level) • Power failure detection function

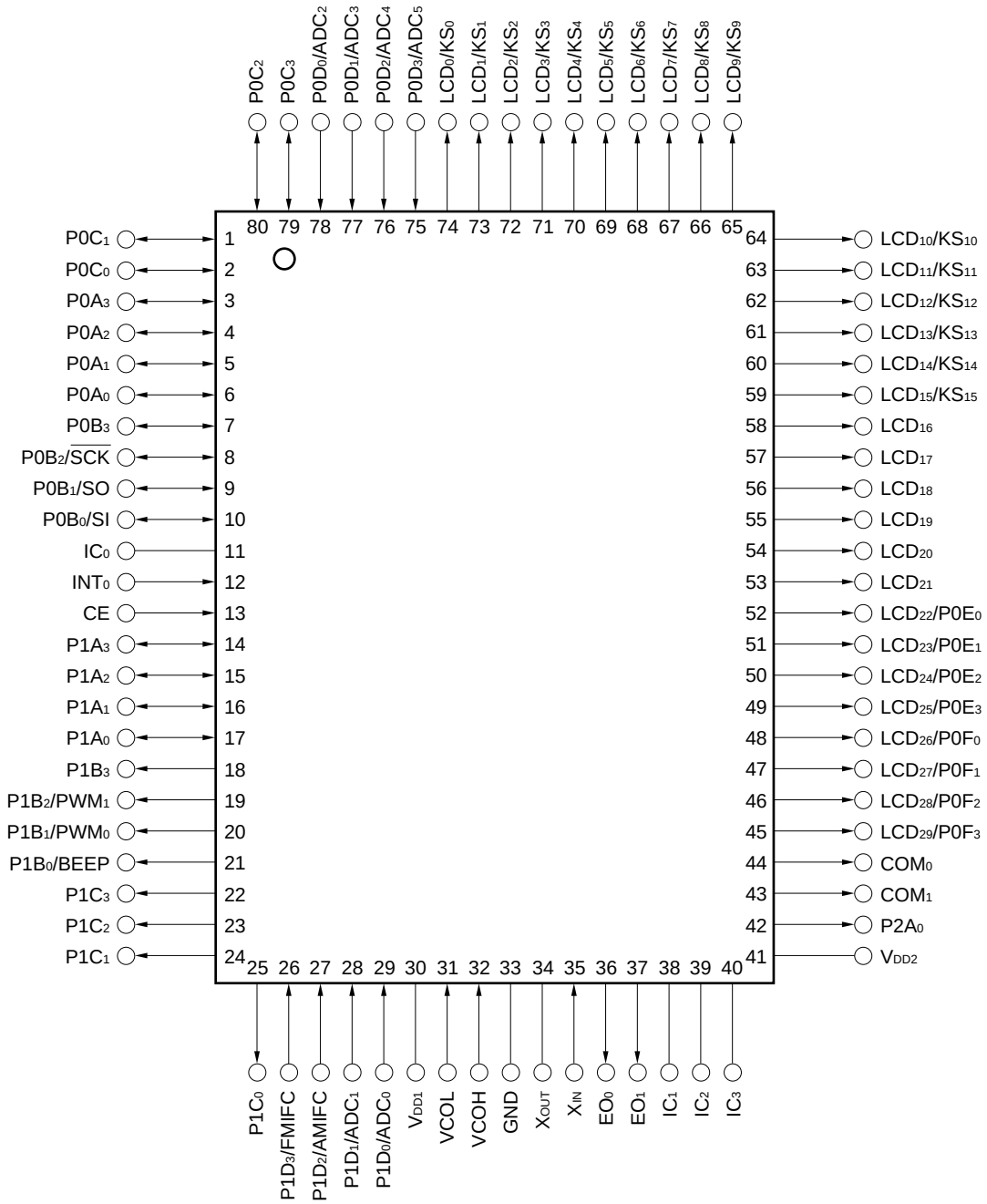
Item		Function
PLL frequency synthesizer	Division mode	• Two types Direct division mode (VCOL pin : 30 MHz MAX.) Pulse swallow mode (VCOL pin : 40 MHz MAX.) (VCOH pin : 150 MHz MAX.)
	Reference frequency	• 12 programmable frequencies 1, 1.25, 2.5, 3, 5, 6.25, 9, 10, 12.5, 25, 50, 100 kHz
	Charge pump	• Two independent error out outputs
	Phase comparator	• Unlock detection by program
Frequency counter		• Frequency measurement $P1D_3$/FMIFC pin : 5 to 15 MHz $P1D_2$/AMIFC pin : 0.1 to 1 MHz
Supply voltage		V_{DD} = 5 V±10%
Package		80-pin plastic QFP (14 × 20 mm)

PIN CONFIGURATION (Top View)

80-pin plastic QFP (14 × 20 mm, 0.8 mm pitch)

μPD17016GF-×××-3B9

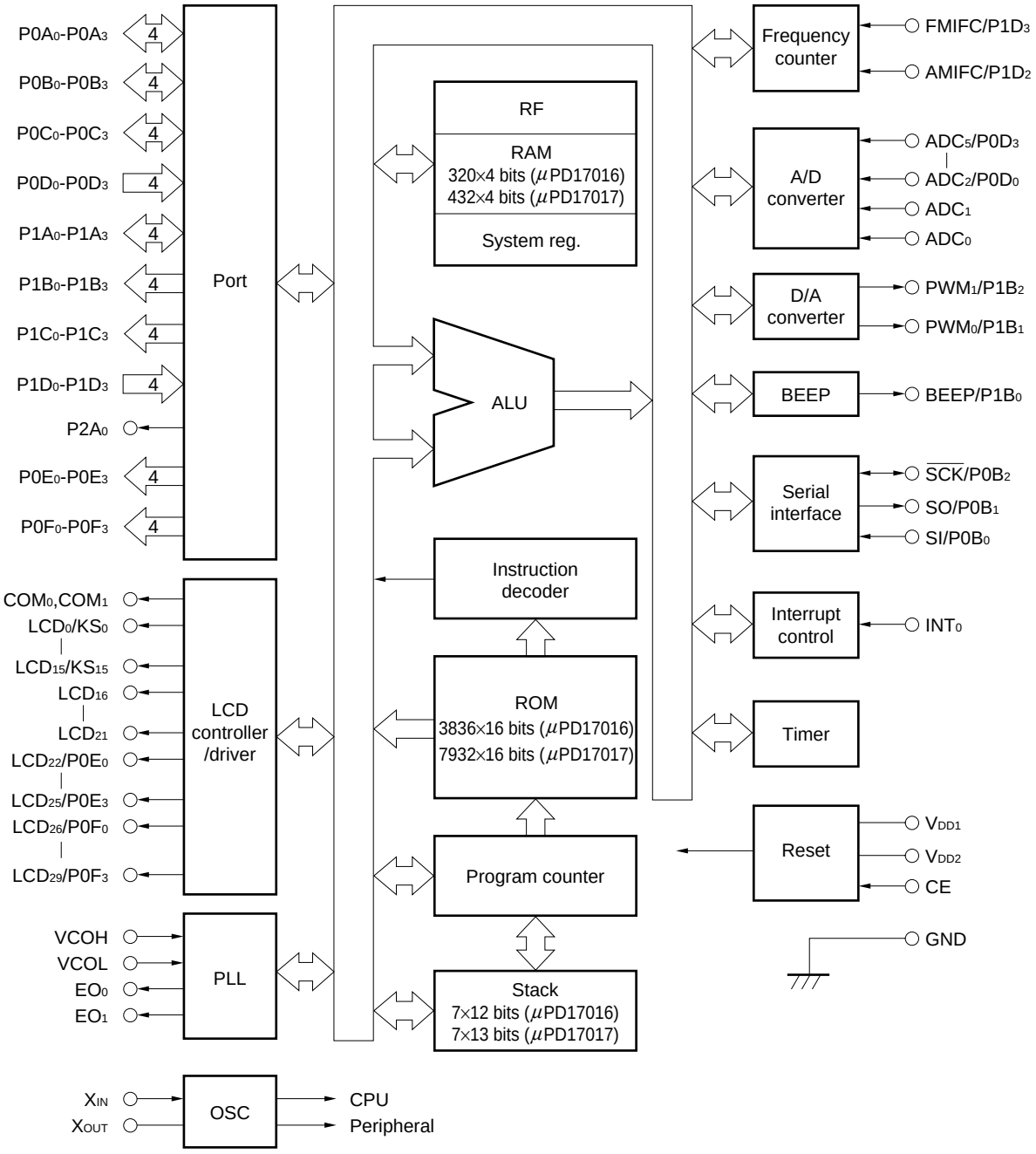
μPD17017GF-×××-3B9



PIN NAME

ADC_0-ADC_5	:	A/D converter input
AMIFC	:	AM intermediate frequency counter input
BEEP	:	BEEP output
CE	:	Chip enable input
COM_0, COM_1	:	LCD common signal output
EO_0, EO_1	:	Error out output
FMIFC	:	FM intermediate frequency counter input
GND	:	Ground
IC_0-IC_3	:	Internal connection
INT_0	:	External interrupt input
KS_0-KS_{15}	:	Key source signal output
LCD_0-LCD_{29}	:	LCD segment signal output
$P0A_0$-$P0A_3$	:	Port 0A
$P0B_0$-$P0B_3$	:	Port 0B
$P0C_0$-$P0C_3$	:	Port 0C
$P0D_0$-$P0D_3$	:	Port 0D
$P0E_0$-$P0E_3$	:	Port 0E
$P0F_0$-$P0F_3$	:	Port 0F
$P1A_0$-$P1A_3$	:	Port 1A
$P1B_0$-$P1B_3$	:	Port 1B
$P1C_0$-$P1C_3$	:	Port 1C
$P1D_0$-$P1D_3$	:	Port 1D
$P2A_0$	:	Port 2A
PWM_0, PWM_1	:	D/A converter output
$\overline{SCK}$	:	Serial clock I/O
SI	:	Serial data input
SO	:	Serial data output
VCOH	:	Local oscillation high input
VCOL	:	Local oscillation low input
V_{DD1}, V_{DD2}	:	Power supply
X_{IN}, X_{OUT}	:	Crystal resonator connection

BLOCK DIAGRAM



Contents

1. PIN FUNCTIONS

1.1 Pin Function List

Pin No.	Symbol	Function	Output Form	On Power-ON Reset
79 80 1 2	$P0C_3$ $P0C_2$ $P0C_1$ $P0C_0$	4-bit I/O port. Can be set in input or output mode in 4-bit units.	CMOS push-pull	Input
3 4 5 6	$P0A_3$ Note $P0A_2$ Note $P0A_1$ $P0A_0$	4-bit I/O port. Can be set in input or output mode in 1-bit units.	N-ch open drain, 5 V ($P0A_3$, $P0A_2$) CMOS push-pull ($P0A_1$, $P0A_0$)	Input
7 8 9 10	$P0B_3$ $P0B_2/\overline{SCK}$ $P0B_1$/SO $P0B_0$/SI	Port 0B and serial interface I/O. • $P0B_3$-$P0B_0$ • 4-bit CMOS I/O port • Can be set in input or output mode in 1-bit units • $\overline{SCK}$, SO, SI • $\overline{SCK}$: serial clock I/O • SO : serial data output • SI : serial data input	CMOS push-pull ($P0B_3$, $P0B_2/\overline{SCK}$, $P0B_1$/SO $P0B_0$)	Input ($P0B_3$-$P0B_0$)
11	IC_0	Internally connected. Connect this pin to V_{DD} or GND via resistor.	–	–
12	INT_0	Edge-detectable vectored interrupt input (rising edge detection). This pin is a Schmitt trigger input pin with hysteresis characteristics. **Do not apply a voltage higher than V_{DD} to INT_0 pin on power application; otherwise, normal operation will not be performed.**	–	Input

Note The $P0A_3$ and $P0B_2$ pins are N-ch open-drain output pins and must be connected to an external pull-up resistor.

Pin No.	Symbol	Function	Output Form	On Power-ON Reset
13	CE	Selects operation of μPD17017 and inputs reset signal. (1) Device operation selection PLL frequency synthesizer can operate when CE pin is high; it is automatically disabled (operation prohibited) while CE pin is low. (2) Reset signal input When CE pin goes high, device is reset in synchronization with internal basic timer 0 carry FF (CE reset). This pin does not accept low or high level of less than 110 to 165 μs to prevent malfunctioning due to noise. Input signal level of this pin can be detected by CE pin level judge register of register file (address 07H). Contents of CE pin level judge register are not changed by low or high level of less than 110 to 165 μs. This pin is a Schmitt trigger input pin with hysteresis characteristics. **Do not apply voltage higher than VDD to this pin on power application; otherwise, normal operation will not be performed.**	–	Input
14 | 17	$P1A_3$ | $P1A_0$	4-bit CMOS I/O port. Can be set in input or output mode in 1-bit units.	CMOS push-pull	Input
18 19 20 21	$P1B_3$ Note $P1B_2/PWM_1$ Note $P1B_1/PWM_0$ Note $P1B_0$/BEEP	Port 1B also serving as D/A converter and buzzer output pins. • $P1B_3$-$P1B_0$ • 4-bit output port • PWM_1, PWM_0 • D/A converter output with 8-bit resolution • BEEP • Buzzer output	N-ch open drain, 16 V ($P1B_3$, $P1B_2/PWM_1$, $P1B_1/PWM_0$) CMOS push-pull ($P1B_0$/BEEP)	Outpus undefined data ($P1B_3$-P1-B_0)
22 | 25	$P1C_3$ | $P1C_0$	4-bit CMOS output port.	CMOS push-pull	Outputs undefined data

Note The $P1B_3$, $P1B_2/PWM_1$, and $P1B_1/PWM_0$ pins are N-ch open-drain output pins and must be connected to an external pull-up resistor.

<table>
<tr><th>Pin No.</th><th>Symbol</th><th>Function</th><th>Output Form</th><th>On Power-ON Reset</th></tr>
<tr><td>26
27
28
29</td><td>P1D3/FMIFC
P1D2/AMIFC
P1D1/ADC1
P1D0/ADC0</td><td>Port 1D, frequency counter input, and analog input to A/D converter.
• P1D3-P1D0
• 4-bit input port
• FMIFC, AMIFC
• Input to FM and AM intermediate frequency counter
Measurable frequency
<table>
<tr><th>Input Pin</th><th>Input Frequency (MHz)</th><th>Input Amplitude (V_{p-p})</th></tr>
<tr><td rowspan="2">P1D3/FMIFC</td><td>5-15</td><td>0.3</td></tr>
<tr><td>10.5-10.9</td><td>0.06</td></tr>
<tr><td rowspan="2">P1D2/AMIFC</td><td>0.1-1</td><td>0.3</td></tr>
<tr><td>0.44-0.46</td><td>0.05</td></tr>
</table>
Because these pins are input pins of AC amplifier, cut DC components of input signals by using capacitor.
• ADC1, ADC0
• Analog input to A/D converter with 6-bit resolution.</td><td>–</td><td>Input
(P1D3-P1D0)</td></tr>
<tr><td>30
41</td><td>VDD1
VDD2</td><td>Positive power supply pins. Supply 5 V±10% to these pins when CPU and peripheral functions operate. Data can be retained at 2.2 V when clock is stopped. When VDD rises, μPD17017 is reset by internal power-ON reset circuit.
Do not apply voltage higher than VDD pins (VDD1 and VDD2) to all pins other than VDD pins. Especially, exercise care in raising VDD and CE pins simultaneously because latch up may occur.
Apply same voltage to VDD1 and VDD2 pins.
VDD2 supplies power to crystal oscillation circuit (XIN and XOUT pins) and error out circuit (EO0 and EO1 pins), and VDD1 pin supplies power to other circuits.</td><td>–</td><td>–</td></tr>
</table>

<table>
<tr><th>Pin No.</th><th>Symbol</th><th>Function</th><th>Output Form</th><th>On Power-ON Reset</th></tr>
<tr><td>31
32</td><td>VCOL
VCOH</td><td>Inputs local oscillation frequency to PLL. Two types of division modes are available: direct division (MF mode) and pulse swallow (HF and VHF modes).
<table><tr><th>Division Mode</th><th>Input Pin</th><th>Input Frequency (MHz)</th><th>Input Voltage (V_{p-p})</th></tr><tr><td>Direct division (MF)</td><td>VCOL</td><td>0.5-30</td><td>0.2</td></tr><tr><td>Pulse swallow (HF)</td><td>VCOL</td><td>5-40</td><td>0.2</td></tr><tr><td>Pulse swallow (VHF)</td><td>VCOH</td><td>9-150</td><td>0.2</td></tr></table>Because these pins are input pins of AC amplifier, cut DC components of input signal by using capacitor.</td><td>–</td><td>Input</td></tr>
<tr><td>33</td><td>GND</td><td>Ground</td><td>–</td><td>–</td></tr>
<tr><td>34</td><td>X_{OUT}Note</td><td rowspan="2">Crystal resonator connecting pins.
Connect crystal resonator of 4.5 MHz to these pins.</td><td>CMOS push-pull</td><td rowspan="2">–</td></tr>
<tr><td>35</td><td>X_{IN}Note</td><td>–</td></tr>
<tr><td>36
37</td><td>EO_0
EO_1</td><td>Output from charge pump of PLL frequency synthesizer.
If value resulting from dividing local oscillation (VCO) frequency input to VCOL (pin 31) or VCOH (pin 32) pin is higher than reference frequency, EO_0 and EO_1 pins output high level; if it is lower, these pins output low level. If two frequencies coincide, these pins are floated.
Because EO_0 and EO_1 pins output same signals, both pins can be used.</td><td>CMOS 3-state</td><td>High impedance</td></tr>
<tr><td>38</td><td>IC_1</td><td>Internally connected pin. Connect this pin to V_{DD} via resistor.</td><td>–</td><td>–</td></tr>
<tr><td>39
40</td><td>IC_2
IC_3</td><td>Internally connected pins. Leave these pins unconnected.</td><td>–</td><td>–</td></tr>
<tr><td>42</td><td>IC_2</td><td>1-bit CMOS output port.</td><td>CMOS push-pull</td><td>Outputs undefined data.</td></tr>
<tr><td>43
44</td><td>COM_1
COM_0</td><td>Output common signals of LCD controller/driver.
These pins output low level on execution of power-ON reset or clock stop instruction in display off mode.</td><td>CMOS3 ternary output</td><td>Low-level output</td></tr>
</table>

Note Refer to **APPENDIX A. NOTES ON CONNECTING CRYSTAL RESONATOR**.

Pin No.	Symbol	Function	Output Form	On Power-ON Reset
45 | 48	$LCD_{29}/P0F_3$ | $LCD_{26}/P0F_0$	Port 0F, port 0E, segment signal output pins of LCD controller/driver, and key source signal output pins of key matrix. • $P0F_3$-$P0F_0$ • 4-bit CMOS output port • $P0E_3$-$P0E_0$ • 4-bit CMOS output port • LCD_{29}-LCD_0 • Segment signal output of LCD controller/driver. • KS_{15}-KS_0 • Key source signal output of key matrix.	CMOS push-pull	Low-level output (LCD_{29}-LCD_0)
49 | 52 53 | 58 59 | 74	$LCD_{25}/P0E_3$ | $LCD_{22}/P0E_0$ LCD_{21} | LCD_{16} LCD_{15}/KS_{15} | LCD_0/KS_0			
75 | 78	$P0D_3/ADC_5$ | $P0D_0/ADC_2$	Port 0D, analog input to A/D converter, and key source signal return input of LCD segment. • $P0D_3$-$P0D_0$ • 4-bit input port • Internal pull-down resistor is always ON. • ADC_5-ADC_2 • Analog input to A/D converter with 6-bit resolution. • Internal pull-down resistor is OFF. • Key source return input • Internal pull-down resistor is ON only while key source is output (220 μs) if LCD segment pin is used as key source, and OFF while LCD segment signal is output.	–	Input with pull-down resistor ($P0D_3$-$P0D_0$)

1.2 Notes on Using General-Purpose Port

1.2.1 Data bit of port register

Input data of ports 0A, 0B, 0C, 0D, 1A, 1B, 1C, 1D, and 2A are read, and output data is set to these ports via the respective port registers (P0A through P2A registers) on data memory.

The $P0A_3$ pin of port 0A correspond to the most significant bit of port register P0A, and the $P0A_0$ pins correspond to the least significant bit.

Likewise, the pins of the other ports, ports 0B, 0C, 0D, 1A, 1B, 1C, 1D, and 2A, correspond to the bits of the respective port registers.

The output data of ports 0E and 0F are set by the LCD segment register on data memory.

1.2.2 I/O ports (ports 0A, 0B, 0C, and 1A)

(1) When each port is set in input mode

The status of each port pin is used as the value of the corresponding port register when an instruction that reads the contents of each port register on data memory is executed (when the address of a port register is specified as m of an instruction such as SKT m, #n4 and ADD r, m).

When an instruction that writes data to each port register is executed (when a port register address is specified as r of an instruction such as MOV m, #n4 and ADD r, m), the value of the corresponding port is written to the output data latch circuit.

(2) When each port is set in output mode

When an instruction that writes data to each port register is executed, the value of the data is written to the data latch circuit, and output from the corresponding port pins.

When an instruction that reads the contents of each port register is executed, the contents of the output data latch are used as the value of the port register. When an instruction that reads the contents of the port register corresponding to the $P0A_3$ and $P0A_2$ pins is executed, the statuses of the pins, which may be different from the output data, are read as is.

All these port pins are set in the input mode on power-ON reset, CE reset, and execution of the clock stop instruction.

Because the contents of the output data latch circuit are undefined on power-ON reset, undefined data may be output unless an instruction that writes data to the port register is executed before a port is set in the output mode. The contents of the output data latch circuit are not affected by CE reset or execution of the clock stop instruction.

1.2.3 Output ports (ports 1B, 1C, 0F, 0E, and 2A)

An output port writes the value of the corresponding port register to the output data latch circuit and outputs it from each port pin when an instruction that writes data to the port register is executed.

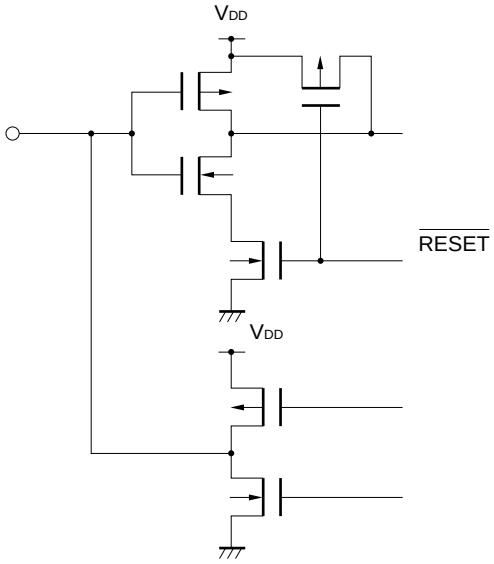
When an instruction that reads the value of the port register is executed, the status of the output data latch circuit is used as the value of the port register.

Undefined data is output on power-ON reset.

The current output data is retained on CE reset and execution of the clock stop instruction. However, ports 0E and 0F automatically output low levels on power-ON reset and execution of the clock stop instruction.

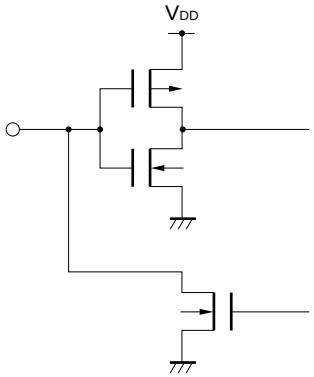
1.3 Equivalent Circuits of Pins

(1) P0A (P0A$_1$, P0A$_0$)
P0B (P0B$_3$, P0B$_2$/$\overline{\text{SCK}}$, P0B$_1$/SO, P0B$_0$/SI)
P0C (P0C$_3$, P0C$_2$, P0C$_1$, P0C$_0$)[Note]
P1A (P1A$_3$, P1A$_2$, P1A$_1$, P1A$_0$) **(I/O)**

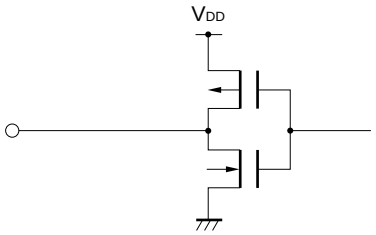


Note The $\overline{\text{RESET}}$ signal is not supplied to P0C.

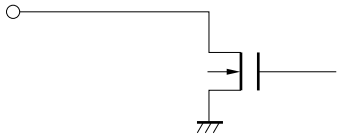
(2) P0A (P0A$_3$, P0A$_2$) (I/O)



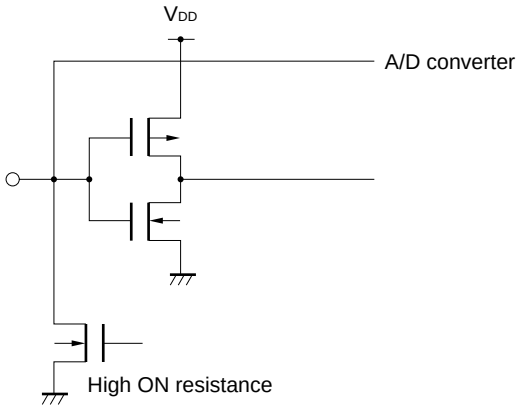
(3) P1B (P1B$_0$/BEEP)
P1C (P1C$_3$, P1C$_2$, P1C$_1$, P1C$_0$)
P2A (P2A$_0$)
LCD$_0$/KS$_0$-LCD$_{29}$/P0F$_3$
(Output)



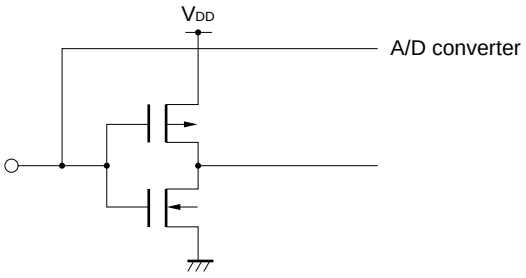
(4) P1B (P1B$_3$, P1B$_2$/PWM$_1$, P1B$_1$/PWM$_0$) (Output)



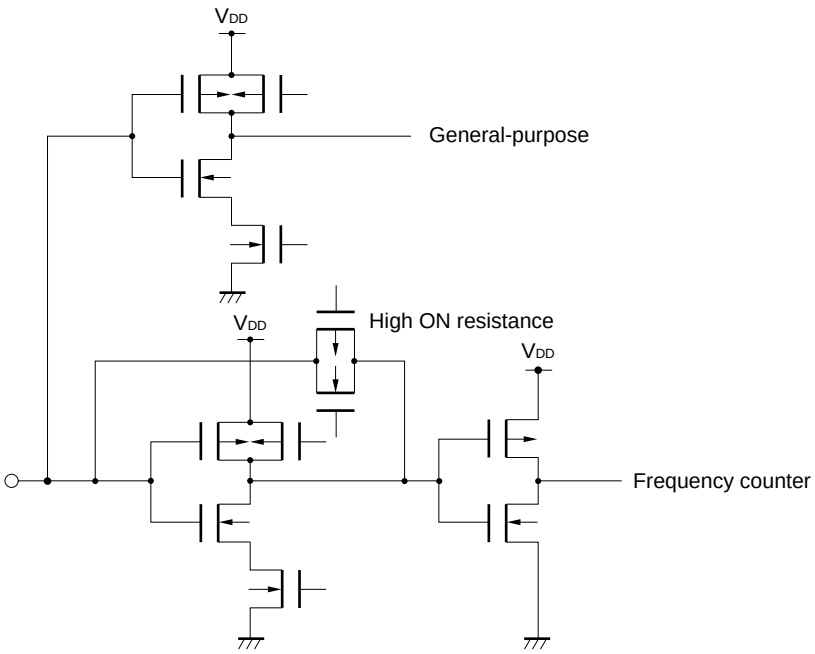
(5) P0D (P0D$_3$/ADC$_5$, P0D$_2$/ADC$_4$, P0D$_1$/ADC$_3$, P0D$_0$/ADC$_2$) (Input)



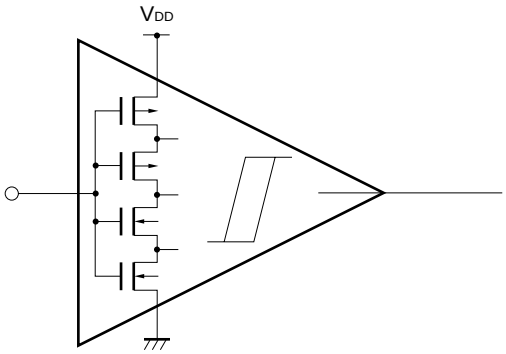
(6) P1D (P1D$_1$/ADC$_1$, P1D$_0$/ADC$_0$) (Input)



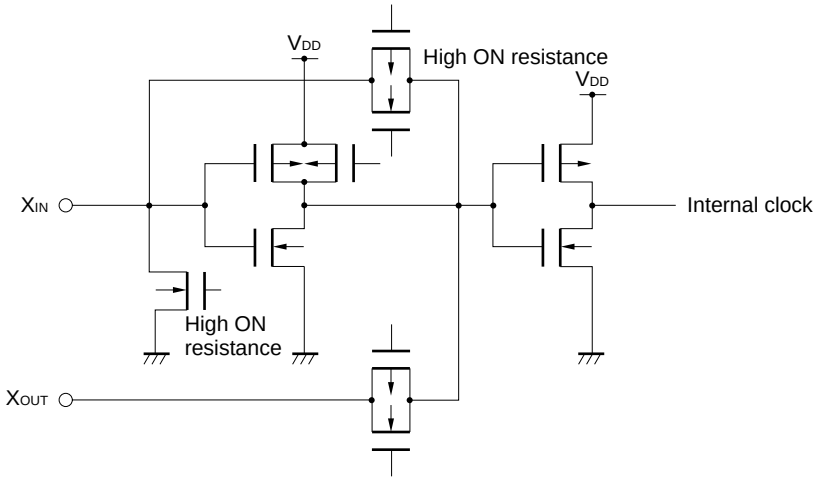
(7) P1D (P1D3/FMIFC, P1D2/AMIFC) (Input)



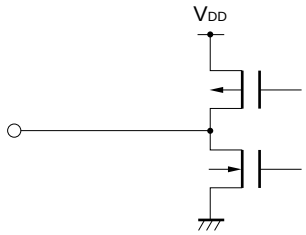
(8) CE, INT0 (Schmitt trigger input)



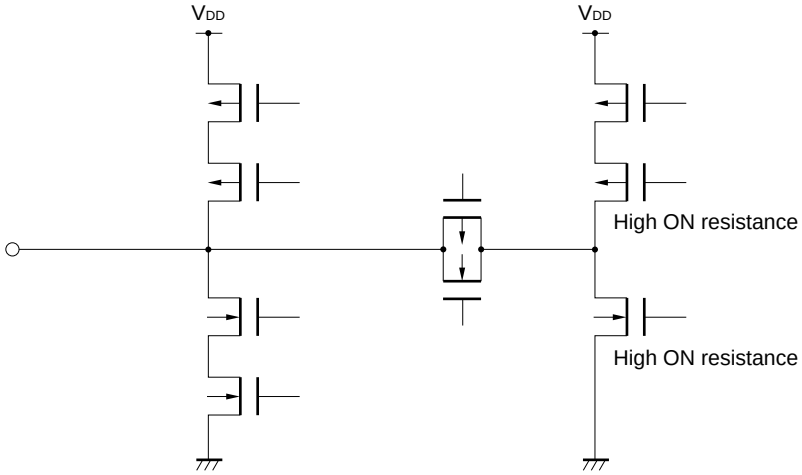
(9) XOUT (output), XIN (input)



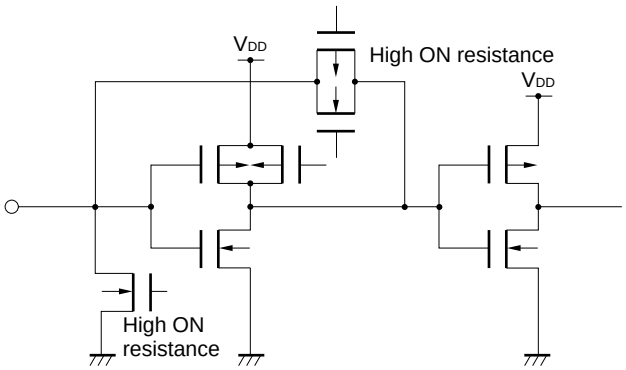
(10) EO_1, EO_0 (output)



(11) COM_1, COM_0 (output)



(12) VCOH, VCOL (input)



1.4 Processing of Unused Pins

It is recommended that unused pins be processed as follows:

Table 1-1. Processing of Unused Pins

Pin Name		I/O Mode	Recommended Processing
Port pins	$P0A_0$-$P0A_3$	I/O[Note 1]	Set general-purpose input port mode by software, and connect each port pin to V_{DD} or GND via resistor[Note 2].
	$P0B_0$/SI $P0B_1$/SO $P0B_2$/$\overline{SCK}$ $P0B_3$		
	$P0C_0$-$P0C_3$		
	$P0D_0$/ADC_2-$P0D_3$/ADC_5	Input	Connect each port pin to GND via resistor[Note 2].
	$P0E_0$/LCD_{22}-$P0E_3$/LCD_{25}	CMOS push-pull output	Open
	$P0F_0$/LCD_{26}-$P0F_3$/LCD_{29}		
	$P1A_0$-$P1A_3$	I/O[Note 1]	Set general-purpose input port mode by software, and connect each port pin to V_{DD} or GND via resistor[Note 2].
	$P1B_0$/BEEP	CMOS push-pull output	Open
	$P1B_1$/PWM_0, $P1B_2$/PWM_1 $P1B_3$	N-ch open drain output	Set low-level output by software and leave port pins open.
	$P1C_0$-$P1C_3$	CMOS push-pull output	Open
	$P1D_0$/ADC_0, $P1D_1$/ADC_1	Input	Connect each pin to V_{DD} or GND via resistor[Note 2].
	$P1D_2$/AMIFC[Note 3] $P1D_3$/FMIFC[Note 3]		Set port mode and connect each pin to V_{DD} or GND via resistor[Note 2].
	$P2A_0$	CMOS push-pull output	Open
Pins other than port pins	CE	Input	Connect to V_{DD} via resistor[Note 2].
	COM_0, COM_1	Output	Open
	EO_0, EO_1	Output	Open
	IC_0	–	Connect to V_{DD} or GND via resistor[Note 2].
	IC_1	–	Connect to V_{DD} via resistor[Note 2].
	IC_2, IC_3	–	Open
	INT_0	Input	Connect to V_{DD} or GND via resistor[Note 2].
	LCD_0/KS_0-LCD_{15}/KS_{15} LCD_{16}-LCD_{21}	Output	Open
	VCOH, VCOL	Input	Disable PLL by software and leave pins open.

Notes 1. The I/O port pins are set in the input mode on power application, execution of the clock stop instruction, or CE reset.

2. If a port pin is externally pulled up (connect to V_{DD} via resistor) or down (connect to GND via resistor) with a high resistance, the pin almost goes into a high-impedance state, and the current consumption (inrush current) of the port pin increases. Generally, the pull-up and pull-down resistances are several 10 kΩ, although they vary depending on the application circuit.

3. Do not set these pins as AMIFC and FMIFC modes; otherwise, the current consumption will increase.

1.5 Notes on Using CE and INT_0 Pins

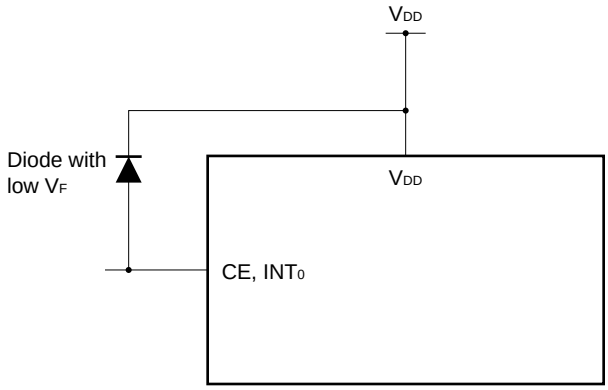
The CE and INT_0 pins have a function to set a test mode (for IC test) in which the internal operations of the μPD17017 are tested, in addition to the functions listed in **1.1 Pin Function List**.

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

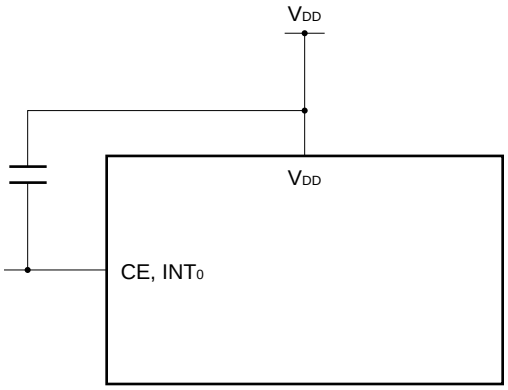
Noise may be superimposed on these pins if the length of the wiring of these pins is too long.

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

- **Connect diode with low V_F between CE or INT_0 and V_{DD}**



- **Connect capacitor between CE or INT_0 and V_{DD}**



2. PROGRAM MEMORY (ROM)

2.1 Outline of Program Memory

Figure 2-1 outlines the program memory.

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

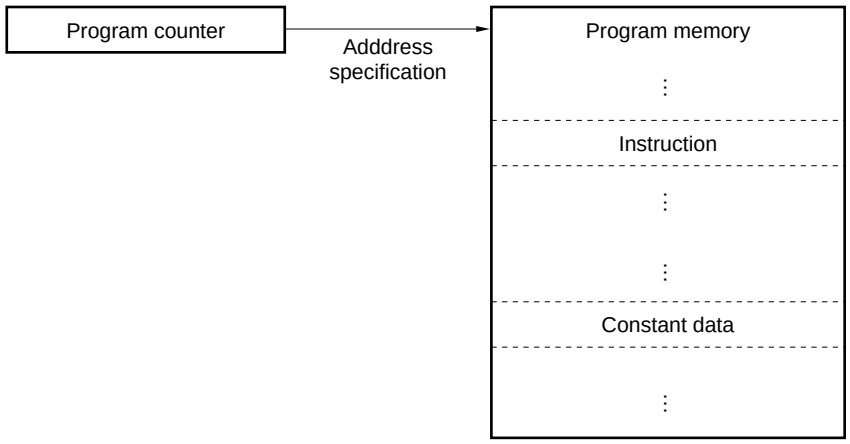
The addresses of the program memory are specified by the program counter.

The program memory has the following two major functions.

(1) Stores programs.

(2) Stores 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 is configured as follows:

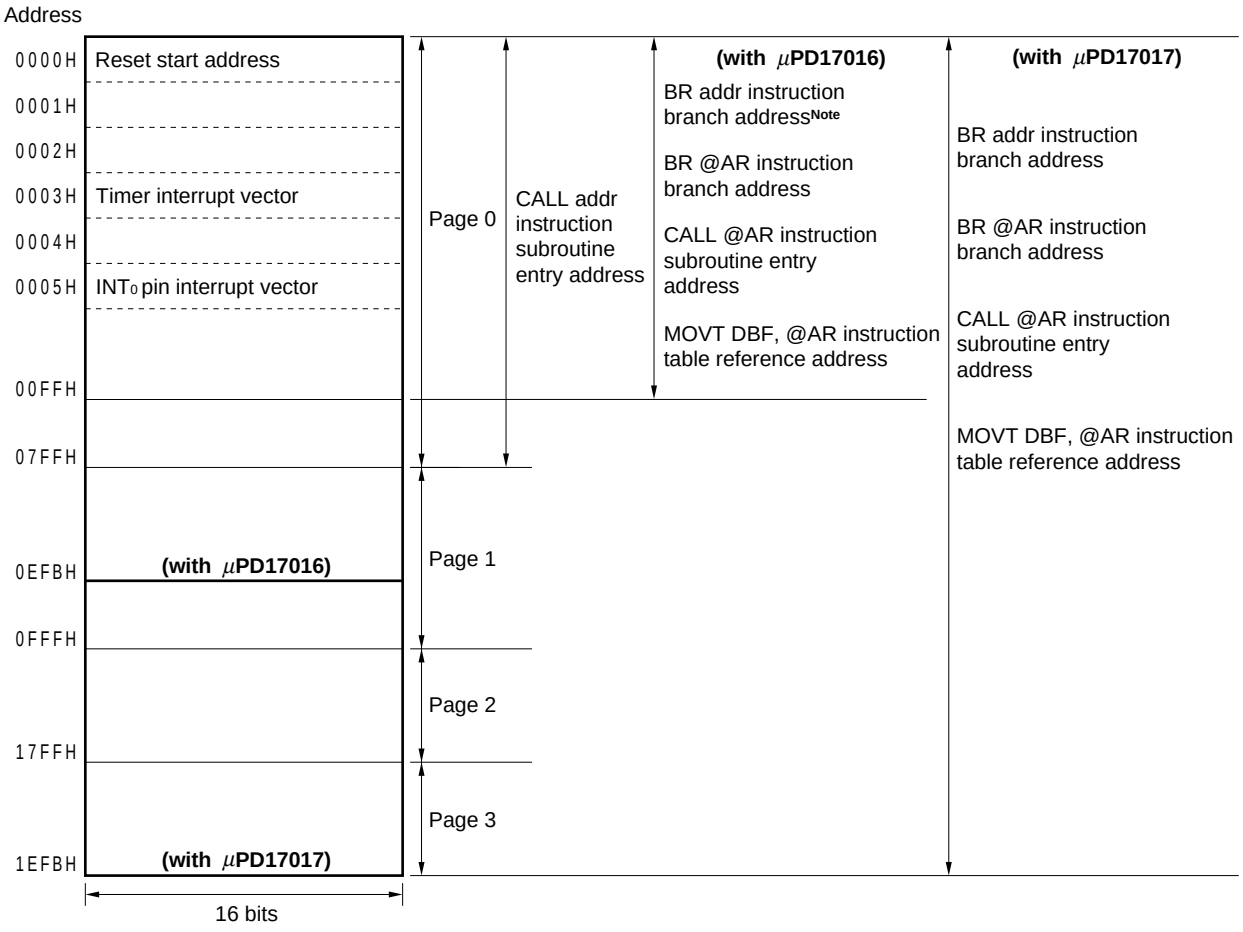
μPD17016: 3836 × 16 bits (0000H through 0EFBH)

μPD17017: 7932 × 16 bits (0000H through 1EFBH)

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

Constant data reads the contents of the program memory to the data buffer by using a table reference instruction.

Figure 2-2. Configuration of Program Memory ★



Note Valid in the address 0000H through 0EFBH.

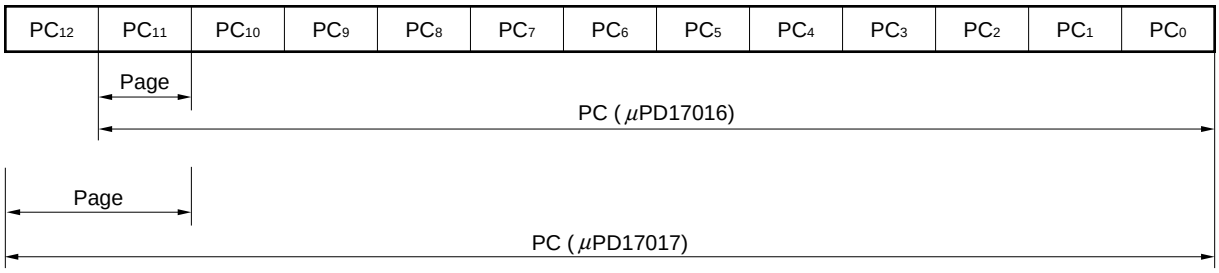
2.3 Program Counter

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

The program counter specifies an address of the program memory.

Bit PC_{11} of the μPD17016, and PC_{11} and PC_{12} of the μPD17017 indicate a page.

Figure 2-3. Configuration of Program Counter



2.4 Program Flow

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

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

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

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

(a) μPD17016

Instruction \ Program counter		Contents of program counter (PC)											
		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	Instruction operand (addr)										
	Page 1	1	←										→
CALL addr		0	Instruction operand (addr)										
			←										→
BR @AR CALL @AR MOVT DBF, @AR		0	0	0	0	Adddress register contents							
			←										→
RET RETSK RETI		Contents of address stack register (ASR) specified by stack pointer (SP) (Return address)											
		←											→
When interrupt is acknowledged		Vector address of each interrupt											
		←											→
Power-ON reset, CE reset		0	0	0	0	0	0	0	0	0	0	0	0

(b) μPD17017

Instruction \ Program counter		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	Instruction operand (addr)										
	Page 1	0	1											
	Page 2	1	0	←										→
	Page 3	1	1											
CALL addr		0	0	Instruction operand (addr)										
				←										→
BR @AR CALL @AR MOVT DBF, @AR		Adddress register contents												
		←												→
RET RETSK RETI		Contents of address stack register (ASR) specified by stack pointer (SP) (Return address)												
		←												→
When interrupt is acknowledged		Vector address of each interrupt												
		←												→
Power-ON reset, CE reset		0	0	0	0	0	0	0	0	0	0	0	0	0

Table 2-1. Interrupt Vector Address

Priority	Internal/External	Interrupt Source	Vector address
1	Internal	INT_0 pin	0005H
2	External	Timer	0003H

2.5 Notes on Using Program Memory

The program memory of the μPD17016 does not have addresses 0EFCH through 0FFFH, and that of the μPD17017 does not have addresses 1EFCH through 1FFFH. Therefore, do not use an instruction that causes the value of the program counter to be set to these addresses.

3. ADDRESS STACK (ASK)

3.1 Outline of Address Stack

Figure 3-1 outlines the address stack.

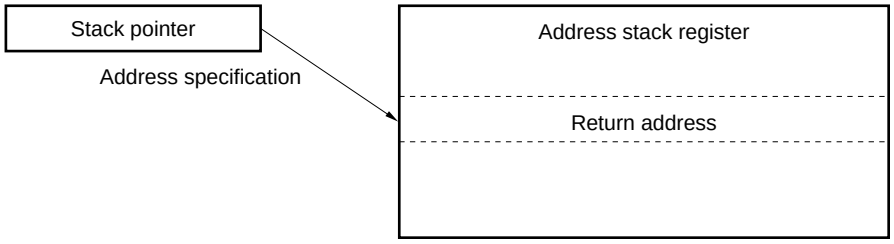
The address stack consists of a stack pointer and address stack registers.

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

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

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

Figure 3-1. Outline of Address Stack



3.2 Address Stack Register (ASR)

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

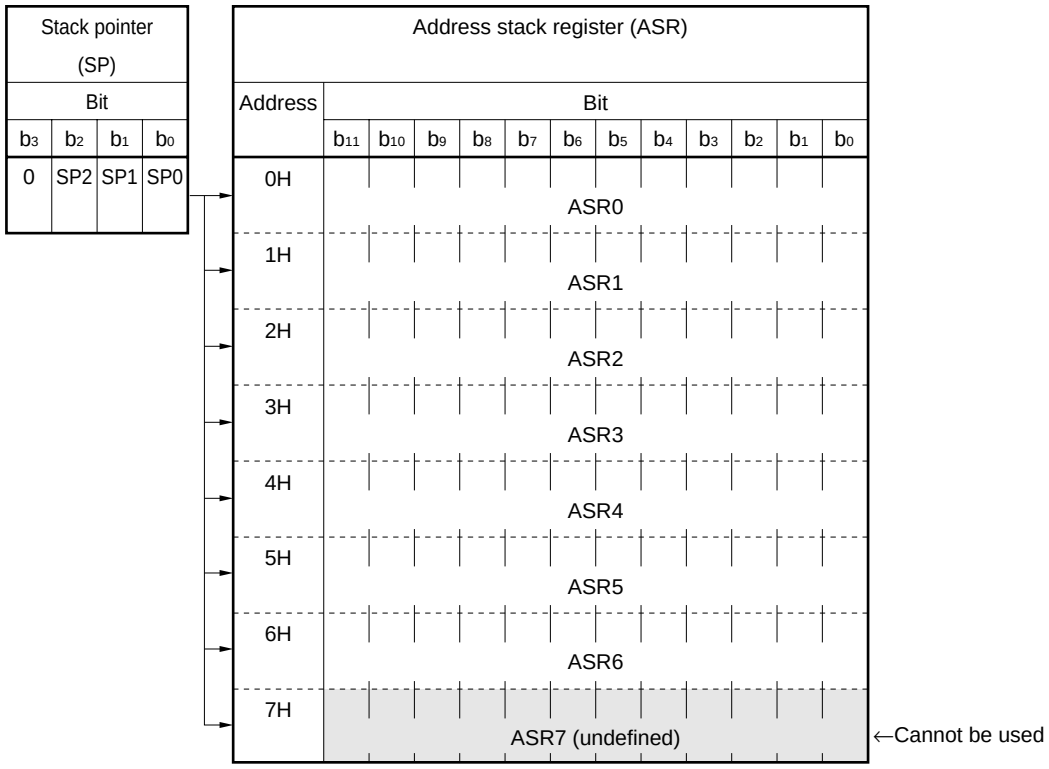
There are eight 16-bit address stack registers: ASR0 through ASR7. Actually, however, no register is assigned to ASR7, and seven registers, ASR0 through ASR6, are used.

The high-order 4 bits of ASR0 through ASR6 of the μPD17016 are fixed to “0”, and the high-order 3 bits of the address stack registers of the μPD17017 are fixed to “0”.

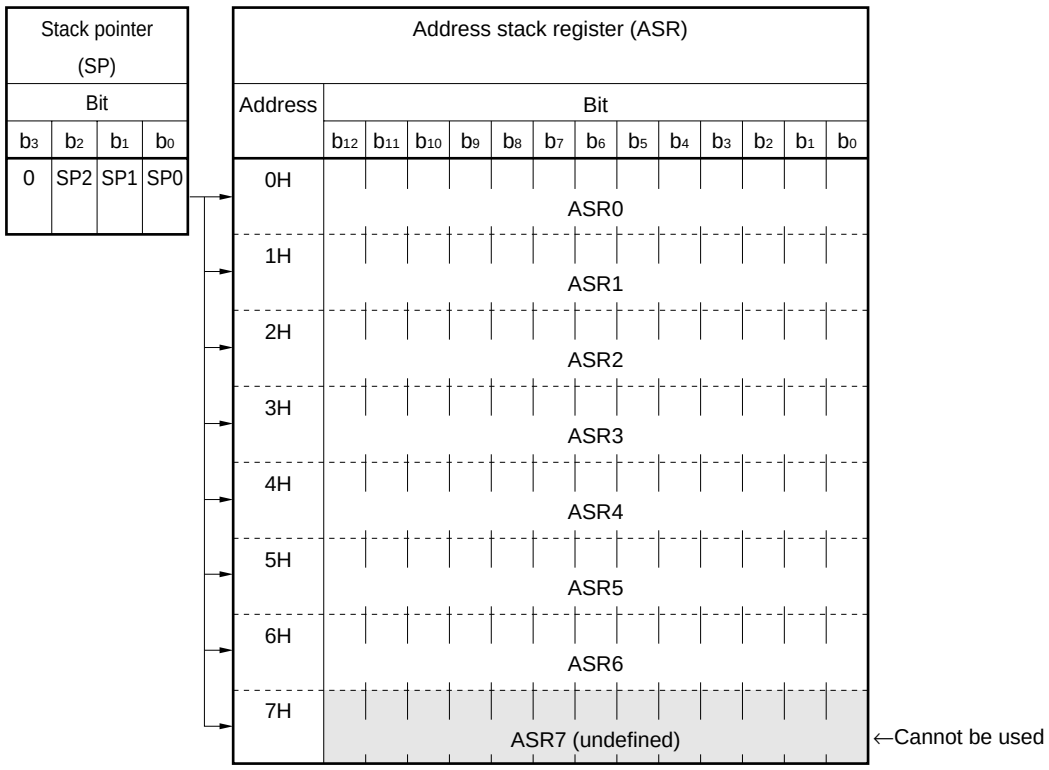
The address stack saves a return address when a subroutine call instruction or table reference instruction is executed, or when an interrupt is acknowledged.

Figure 3-2. Configuration of Address Stack Registers

(a) μPD17016



(b) μPD17017



3.3 Stack Pointer (SP)

3.3.1 Configuration and function of stack pointer

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

The stack pointer is a 4-bit binary counter.

It specifies the address of an address stack register.

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

Figure 3-3. Configuration and Function of Stack Pointer

Name	Flag Symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Stack pointer SP	0	SP2	SP1	SP0	01H	R/W

SP2	SP1	SP0	Specifies address of address stack register (ASR)
0	0	0	Address 0 (ASR0)
0	0	1	Address 1 (ASR1)
0	1	0	Address 2 (ASR2)
0	1	1	Address 3 (ASR3)
1	0	0	Address 4 (ASR4)
1	0	1	Address 5 (ASR5)
1	1	0	Address 6 (ASR6)
1	1	1	Address 7 (ASR7)

Fixed to “0”

On reset					
	Power-ON	0	1	1	1
	Clock stop		1	1	1
	CE		1	1	1

3.4 Operation of Address Stack

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

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

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

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

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

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

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

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

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

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

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

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

3.5 Notes on Using Address Stack

3.5.1 Nesting level

The value of the address stack register (ASR7) is “undefined” when the value of the stack pointer is 07H.

Do not use a subroutine call or interrupt exceeding level 7 without manipulating the stack; otherwise, execution returns to an “undefined address”.

4. DATA MEMORY (RAM)

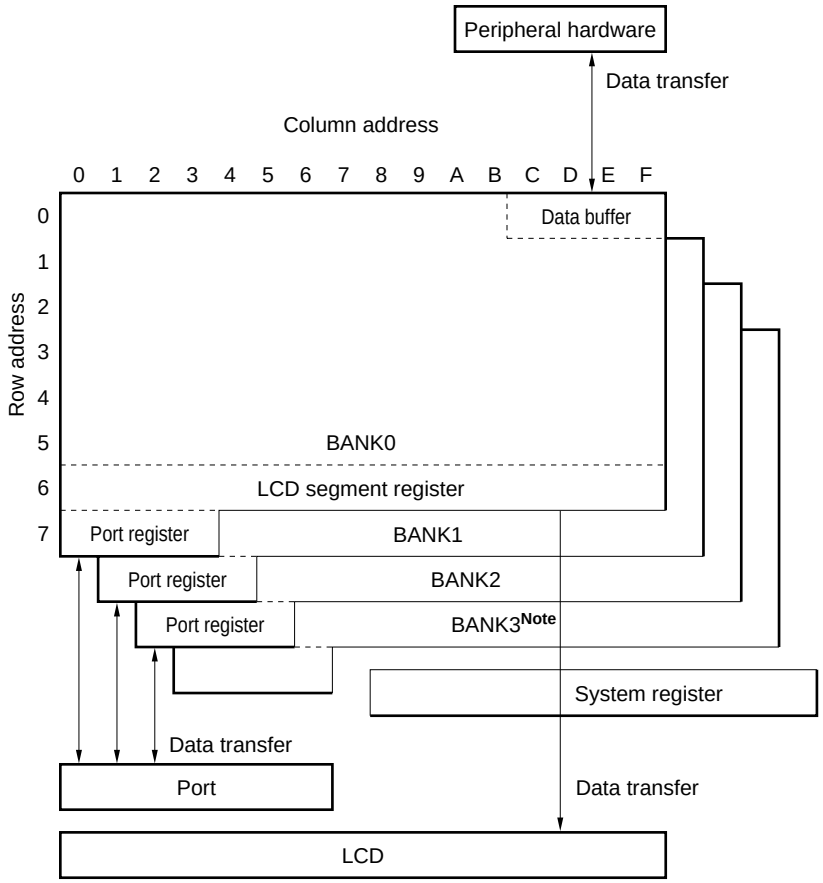
4.1 Outline of Data Memory

Figure 4-1 outlines the data memory.

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

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

Figure 4-1. Outline of Data Memory



Note BANK3 is not provided on the μPD17016.

4.2 Configuration and Function of Data Memory

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

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

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

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

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

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

4.2.1 System register (SYSREG)

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

Because the system register is allocated to every bank, the identical system register exists 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 BANK0.

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

4.2.3 LCD segment data register (LCD segment register)

The LCD segment register is allocated to addresses 60H through 6FH of BANK0.

For details, refer to **21. LCD CONTROLLER/DRIVER**.

4.2.4 Port data registers (port registers)

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

For details, refer to **15. GENERAL-PURPOSE PORT**.

4.2.5 General-purpose data memory

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

(a) μPD17016

The general-purpose data memory of the μPD17016 consists of a total of 320 nibbles (320 × 4 bits): 96 nibbles of BANK0, and 112 nibbles each of BANK1 and BANK2.

(b) μPD17017

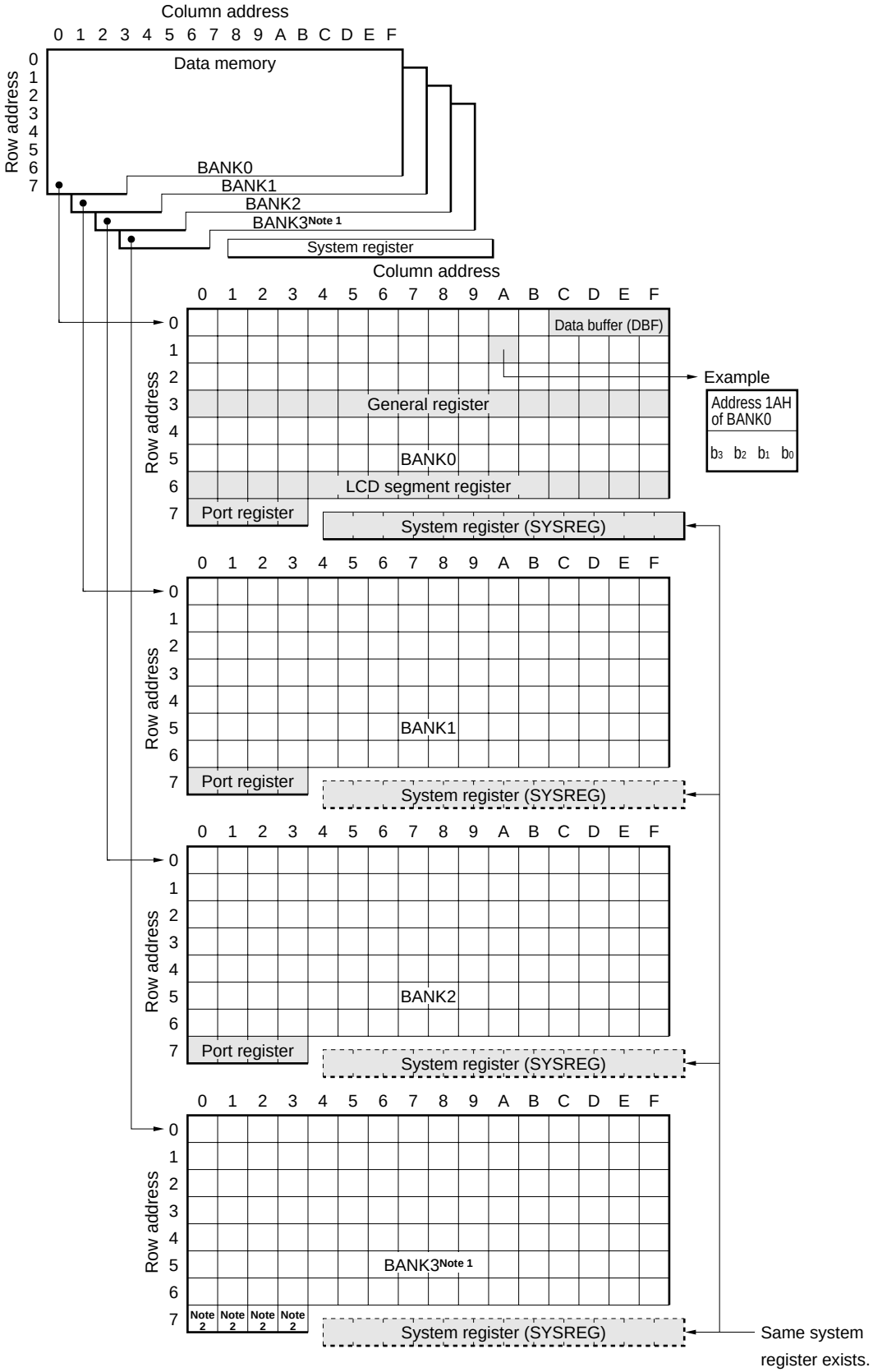
The general-purpose data memory of the μPD17017 consists of a total of 432 nibbles (432 × 4 bits): 96 nibbles of BANK0, and 112 nibbles each of BANK1 through BANK3.

4.2.6 Data memory not provided

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

For the details of these data memory areas, refer to **4.4.2 Notes on data memory area not provided**, **15. GENERAL-PURPOSE PORT**, and **21. LCD CONTROLLER/DRIVER**.

Figure 4-2. Configuration of Data Memory



Notes 1. BANK3 is not provided on the μPD17016.

2. BANK3 does not have addresses 70H through 73H.

Table 4-1. Data Memory Manipulation Instructions

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

4.3 Addressing of Data Memory

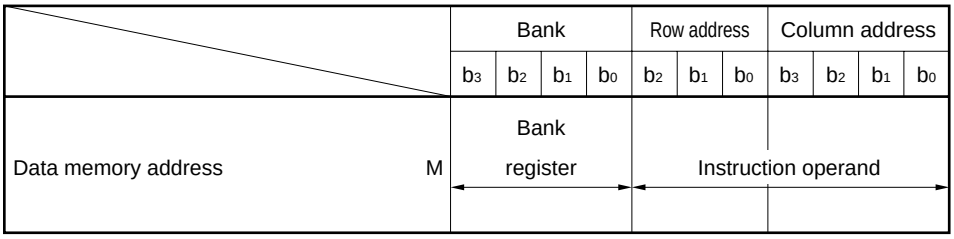
Figure 4-3 shows addressing of the data memory.

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

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

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

Figure 4-3. Addressing of Data Memory



4.4 Notes on Using Data Memory

4.4.1 On power-ON reset

The contents of the general-purpose data memory are “undefined” on power-ON reset.

Initialize the general-purpose data memory as necessary.

4.4.2 Notes on data memory area not provided

If a data memory manipulation instruction is executed to an address of the data memory area not provided, the following operations are performed.

(1) Device operation

If a read instruction is executed, “0” is read.

Nothing is affected even if a write instruction is executed.

(2) Assembler operation

Assembly is performed normally.

An “error” does not occur.

(3) In-circuit emulator operation

If a read instruction is executed, “0” is read.

Nothing is affected even if a write instruction is executed.

An “error” does not occur.

5. SYSTEM REGISTER (SYSREG)

5.1 Outline of System Register

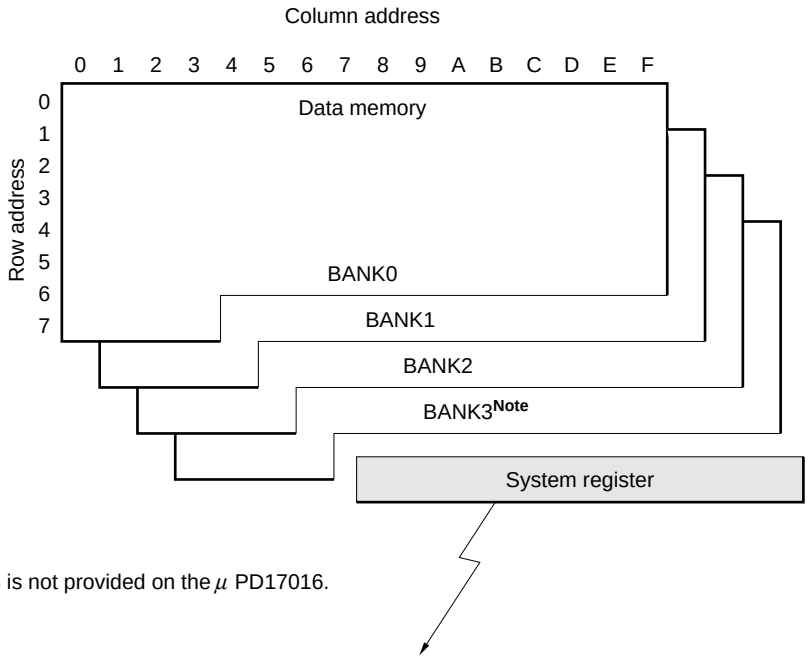
Figure 5-1 shows the location of the system memory on the data memory and the outline of the system register.

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

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

The system register consists of seven registers.

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



Note BANK3 is not provided on the μ PD17016.

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

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

5.2 System Register List

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

Figure 5-2. Configuration of System Register

Address	74H				75H				76H				77H				78H				79H			
Name	System register																							
	Address register (AR)																Window register (WR)				Bank register (BANK)			
Symbol	AR3				AR2				AR1				AR0				WR				BANK			
Bit	b_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	0	0	0	0	0	0	0	0	(μPD17016)												0	0		
	0	0	0						(μPD17017)												0	0		

Address	7AH				7BH				7CH				7DH				7EH				7FH			
Name	System register																							
	Index register (IX)												General register pointer (RP)								Program status word (PSWORD)			
	Data memory row address pointer (MP)																							
Symbol	IXH				IXM				IXL				RPH				RPL				PSW			
	MPH				MPL																			
Bit	b_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	MPE	0	0						(IX)				0	0	0		(RP)			BCD	CMP	CY	Z	IXE
					(MP)																			

5.3 Address Register (AR)

★ 5.3.1 Configuration of address register

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

As shown in this figure, the address register consists of 16 bits, or 74H through 77H (AR3 through AR0) of the system register. The high-order 8 bits of this register of the μPD17016 are fixed to "0", and the high-order 3 bits of the address register of the μPD17017 are fixed to "0".

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[Note]		0	0	0	0	0 (MSB)	0	0	0	(μPD17016)							(LSB)
		0	0	0	(MSB)					(μPD17017)							(LSB)
On reset	Power-ON	0				0				0				0			
	Clock stop	0				0				0				0			
	CE	0				0				0				0			

Remark Power-ON : power-ON reset
Clock stop : execution of clock stop instruction
CE : CE reset

Note Bits marked "0" are fixed to 0.

5.3.2 Function of address register

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

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

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

The address register can set the following range of the program memory addresses.

μPD17016 : 0000H through 00FFH
μPD17017 : 0000H through 1EFBH

Do not set an address other than above to the address register.

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

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

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

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

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

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

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

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

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

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

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

(a) With μPD17016

Because the address register of the μPD17016 consists of 8 bits, its contents are cleared to “0000H” if the “INC AR” instruction is executed when the contents of the address register are “00FFH”.

(b) With μPD17017

Because the address register of the μPD17017 consists of 13 bits, its contents are cleared to “0000H” if the “INC AR” instruction is executed when the contents of the address register are “1FFFH”.

5.3.3 Address register and data buffer

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

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

5.4 Window Register (WR)

5.4.1 Configuration of window register

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

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

Figure 5-4. Configuration of Window Register

Address		78H			
Name		Window register (WR)			
Symbol		WR			
Bit		b_3	b_2	b_1	b_0
Data		MSB			LSB
On reset	Power-ON	Undefined			
	Clock stop	Holds previous status			
	CE				

5.4.2 Function of window register

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

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

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

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

(1) “PEEK WR, rf” instruction

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

(2) “POKE rf, WR” instruction

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

5.5 Bank Register (BANK)

5.5.1 Configuration of bank register

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

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

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		0	0	MSB	LSB
On reset	Power-ON	0			
	Clock stop	0			
	CE	0			

5.5.2 Function of bank register

The bank register specifies a bank of the data memory.

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

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

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

Table 5-1. Specifying Bank of Data Memory

Bank register (BANK)				Data memory bank
b_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[Note]

Note BANK3 is not provided on the μPD17016. Do not set this bank with the μPD17016.

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

5.6.1 Configuration of index register and data memory row address pointer

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

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

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

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

Note, however, that the high-order 2 bits of the index register and data memory row address pointer (bits b_2 and b_1 of address 7AH) are always fixed to “0”.

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

Address		7AH				7BH				7CH				7EH				7FH			
Name		Index register (IX) / Memory pointer (MP)															Program status word (PSWORD)				
Symbol		IXH / MPH				IXM / MPL				IXL								PSW			
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		$\overbrace{MSB}$	0	0	$\overbrace{MSB}$				IX / MP				$\overbrace{LSB}$								IXE
On reset	Power-ON	0				0				0											0
	Clock stop	0				0				0											0
	CE	0				0				0											0

5.6.2 Functions of index register and data memory row address pointer

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

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

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

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

(1) Index register

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

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

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

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

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

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

(2) Data memory row address pointer

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

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

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

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

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

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

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

5.7 General Register Pointer (RP)

5.7.1 Configuration of general register pointer

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

As shown in this figure, the general register pointer consists of 7 bits: 4 bits of address 7DH (RPH) of the system register and the high-order 3 bits of address 7EH (RPL). However, because the high-order 3 bits of address 7E are always fixed to 0, actually the low-order 4 bits of this register (the lowest 1 bit of address 7DH and the high-order 3 bits of address 7EH) are valid.

Figure 5-7. Configuration of General Register Pointer

Address		7DH				7EH			
Name		General register pointer (RP)							
Symbol		RPH				RPL			
Bit		b_3	b_2	b_1	b_0	b_3	b_2	b_1	b_0
Data		0	0	0	MSB			LSB	BCD
On reset	Power-ON	0				0			
	Clock stop	0				0			
	CE	0				0			

5.7.2 Function of general register pointer

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

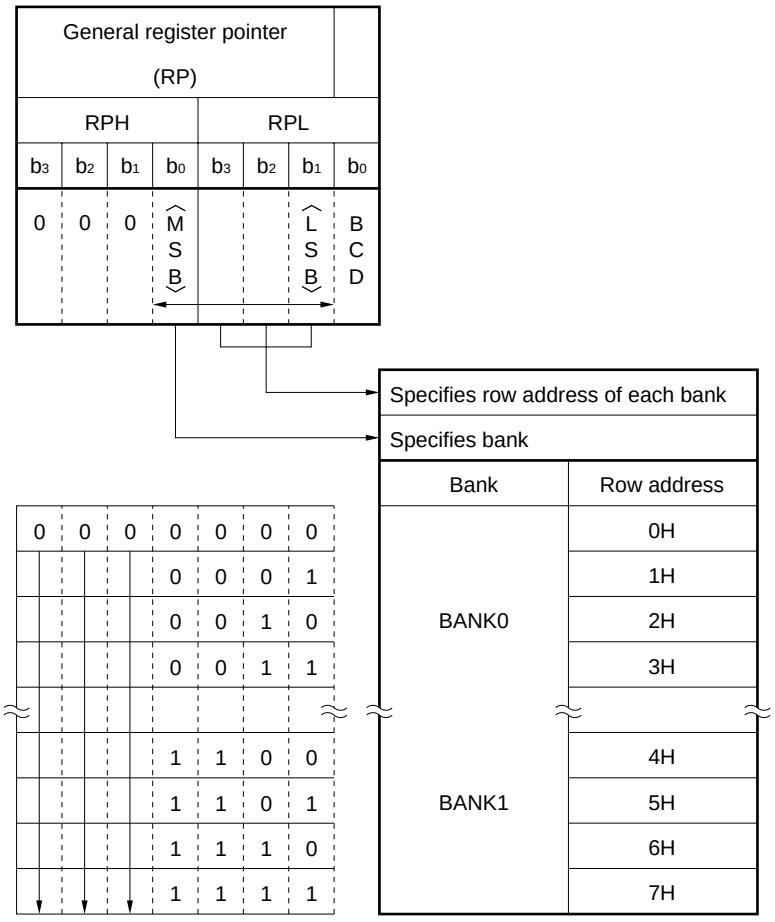
Figure 5-8 shows the address of the general register specified by the general register pointer.

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

Because the valid number of bits of the general register pointer is 4, the row addresses (0H through 7H) of BANK0 and BANK1 can be specified as general registers.

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

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



Caution Because the valid number of bits of the general register pointer is 4, BANK2 and BANK3 cannot be specified.

5.7.3 Notes on using general register pointer

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

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

5.8 Program Status Word (PSWORD)

5.8.1 Configuration of program status word

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

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

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

Figure 5-9. Configuration of Program Status Word

Address		7EH				7EH			
Name		(RP)			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					BCD	CMP	CY	Z	IXE
On reset	Power-ON	0				0			
	Clock stop	0				0			
	CE	0				0			

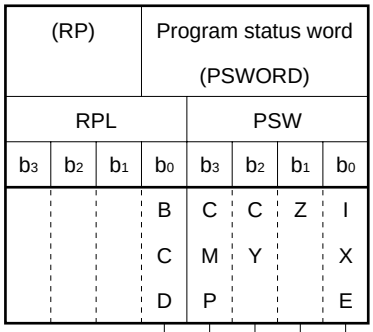
5.8.2 Function of program status word

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

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

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

Table 5-2. Functional Outline of Program Status Word



Flag Name	Function
Index enable flag (IXE)	This flag modifies address of data memory when data memory manipulation instruction is executed. 0 : Does not modify 1 : Modifies
Zero flag (Z)	This flag indicates that result of arithmetic operation executed is 0. Note that status of 0 or 1 differs depending on content of compare flag.
Carry flag (CY)	This flag indicates occurrence of carry or borrow as result of execution of addition or subtraction instruction. It is reset to “0” if carry/borrow does not occur; it is set to “1” if carry/borrow occurs. This flag is also used as sift bit of “RORC r” instruction.
Compare flag (CMP)	This flag is used to not store result of arithmetic operation to data memory or general register. 0 : Result stored 1 : Result not stored
BCD flag (BCD)	This flag is used to execute arithmetic operation in decimal. 0 : Executes binary operation 1 : Executes decimal operation

5.8.3 Notes on using program status word

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

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

5.9 Notes on Using System Register

The data of the system register fixed to “0” is not affected even if a write instruction is executed to it.

This data is always “0” when it is read.

6. GENERAL REGISTER (GR)

6.1 Outline of General Register

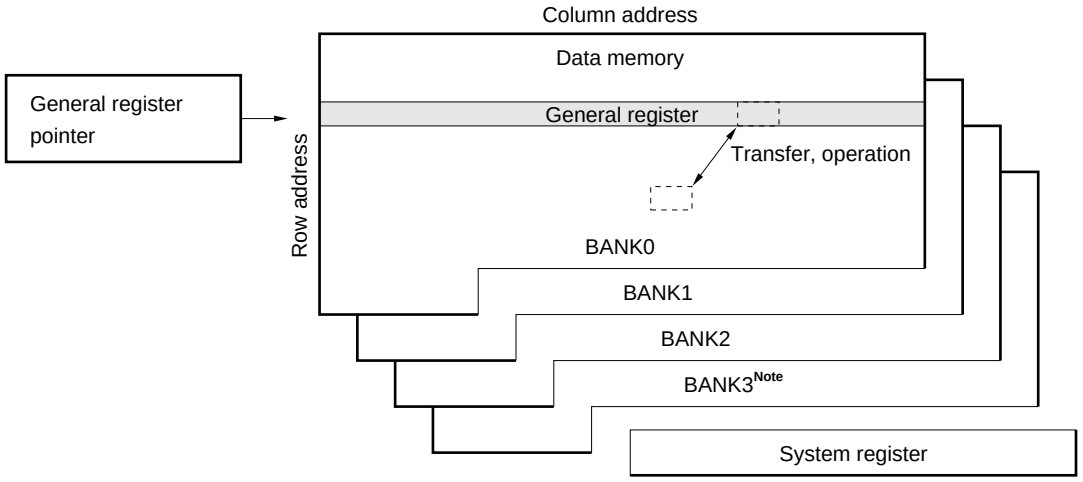
Figure 6-1 outlines the general register.

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

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

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

Figure 6-1. Outline of General Register



Note BANK3 is not provided to the μPD17016.

Caution Because the valid number of bits of the general register pointer is 4, BANK2 and BANK3 cannot be specified.

6.2 General Register Body

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

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

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

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

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

6.3 Address Generation of General Register by Instructions

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

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

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

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

Table 6-1. Address Generation 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	R	Contents of general register pointer							r			

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

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

Table 6-2. Address Generation of General Register

		Bank				Row address			Column address			
		b_3	b_2	b_1	b_0	b_2	b_1	b_0	b_3	b_2	b_1	b_0
General register address	R	Contents of general register pointer							r			
Indirect transfer address	@R	Same as data memory							Contents of R			

6.4 Notes on Using General Register

6.4.1 Row address of general register

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

6.4.2 Operation between general register and immediate data

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

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

7. ALU (Arithmetic Logic Unit) BLOCK

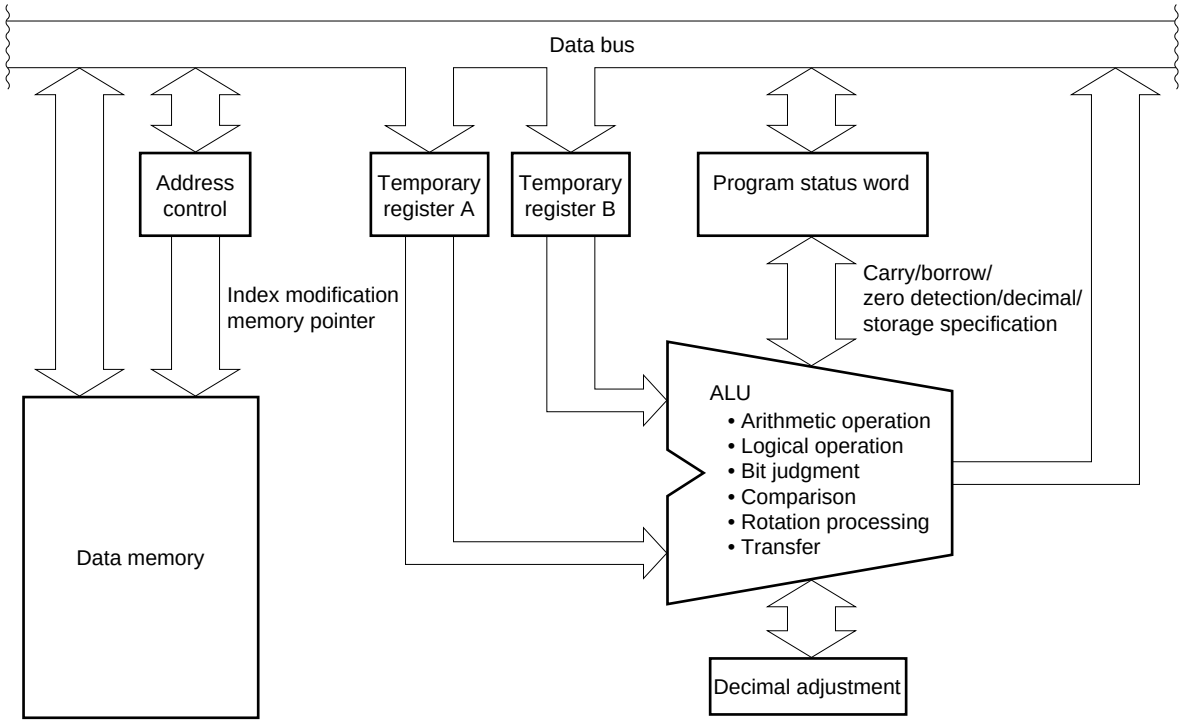
7.1 Outline of ALU Block

Figure 7-1 outlines the ALU block.

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

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

Figure 7-1. Outline of ALU Block



7.2 Configuration and Function of Each Block

7.2.1 ALU

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

7.2.2 Temporary registers A and B

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

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

7.2.3 Program status word

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

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

7.2.4 Decimal adjustment circuit

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

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 modification of data memory addresses by the index register and data memory row address pointer.

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

Table 7-1. List of ALU Processing Instruction Operations

ALU Function	Instruction		Difference in Operation due to Program Status Word (PSWORD)					Address Modification	
			Value of BCD flag	Value of CMP flag	Arithmetic operation	Operation of CY flag	Operation of Z flag	Index	Memory pointer
Addition	ADD	r, m	0	0	Stores result of binary operation.	Set if carry or borrow occurs; otherwise, reset.	Set if result of operation is 0000B; otherwise, reset.	Executed	Not executed
		m, #n4							
	ADDC	r, m	0	1	Does not store result of binary operation.		Holds status if result of operation is 0000B; otherwise, reset.		
		m, #n4							
Subtraction	SUB	r, m	1	0	Stores result of decimal operation.		Set if result of operation is 0000B; otherwise, reset.		
		m, #n4							
	SUBC	r, m	1	1	Does not store result of decimal operation.		Holds status if result of operation is 0000B; otherwise, reset.		
		m, #n4							
Logical operation	OR	r, m	Any (held)	Any (held)	Not affected	Holds previous status.	Holds previous status.	Executed	Not executed
		m, #n4							
	AND	r, m							
		m, #n4							
	XOR	r, m							
		m, #n4							
Judgment	SKT	m, #n	Any (held)	Any (reset)	Not affected	Holds previous status.	Holds previous status.	Executed	Not executed
	SKF	m, #n							
Compare	SKE	m, #n4	Any (held)	Any (held)	Not affected	Holds previous status.	Holds previous status.	Executed	Not executed
	SKNE	m, #n4							
	SKGE	m, #n4							
	SKLT	m, #n4							
Transfer	LD	r, m	Any (held)	Any (held)	Not affected	Holds previous status	Holds previous status	Executed	Not executed
	ST	m, r							
	MOV	m, #n4							
		@r, m							Executed
		m, @r							
Rotation	RORC	r	Any (held)	Any (held)	Not affected	Value of general register b_0	Holds previous value	Not executed	Not executed

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

IXE	MPE	General register address specified by r			Data memory address specified by m			Indirect transfer address specified by @r		
		Bank	Row address	Column address	Bank	Row address	Column address	Bank	Row address	Column address
		b_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	RP		r	BANK		m	BANK	m_R	(r)
0	1		ditto			ditto		MP		(r)
1	0		ditto		BANK		m	BANK	m_R	
					Logical IX		OR	Logical IXH, IXM		OR (r)
1	1		ditto			ditto		MP		(r)

BANK : bank register
IX : index register
IXE : index enable flag
IXH : bits 10 through 8 of index register
IXM : bits 7 through 4 of index register
IXL : bits 3 through 0 of index register
m : data memory address indicated by m_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 Result	Hexadecimal Addition		Decimal Addition	
	CY	Operation Result	CY	Operation Result
0	0	0000B	0	0000B
1	0	0001B	0	0001B
2	0	0010B	0	0010B
3	0	0011B	0	0011B
4	0	0100B	0	0100B
5	0	0101B	0	0101B
6	0	0110B	0	0110B
7	0	0111B	0	0111B
8	0	1000B	0	1000B
9	0	1001B	0	1001B
10	0	1010B	1	0000B
11	0	1011B	1	0001B
12	0	1100B	1	0010B
13	0	1101B	1	0011B
14	0	1110B	1	0100B
15	0	1111B	1	0101B
16	1	0000B	1	0110B
17	1	0001B	1	0111B
18	1	0010B	1	1000B
19	1	0011B	1	1001B
20	1	0100B	1	1110B
21	1	0101B	1	1111B
22	1	0110B	1	1100B
23	1	0111B	1	1101B
24	1	1000B	1	1110B
25	1	1001B	1	1111B
26	1	1010B	1	1100B
27	1	1011B	1	1101B
28	1	1100B	1	1010B
29	1	1101B	1	1011B
30	1	1110B	1	1100B
31	1	1111B	1	1101B

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

Remark The operation results in the shaded portion are not correctly adjusted by the decimal adjustment circuit.

7.4 Notes on Using ALU

7.4.1 Notes on using operation to program status word

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

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

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

7.4.2 Notes on using decimal operation

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

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

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

8. REGISTER FILE (RF)

8.1 Outline of Register File

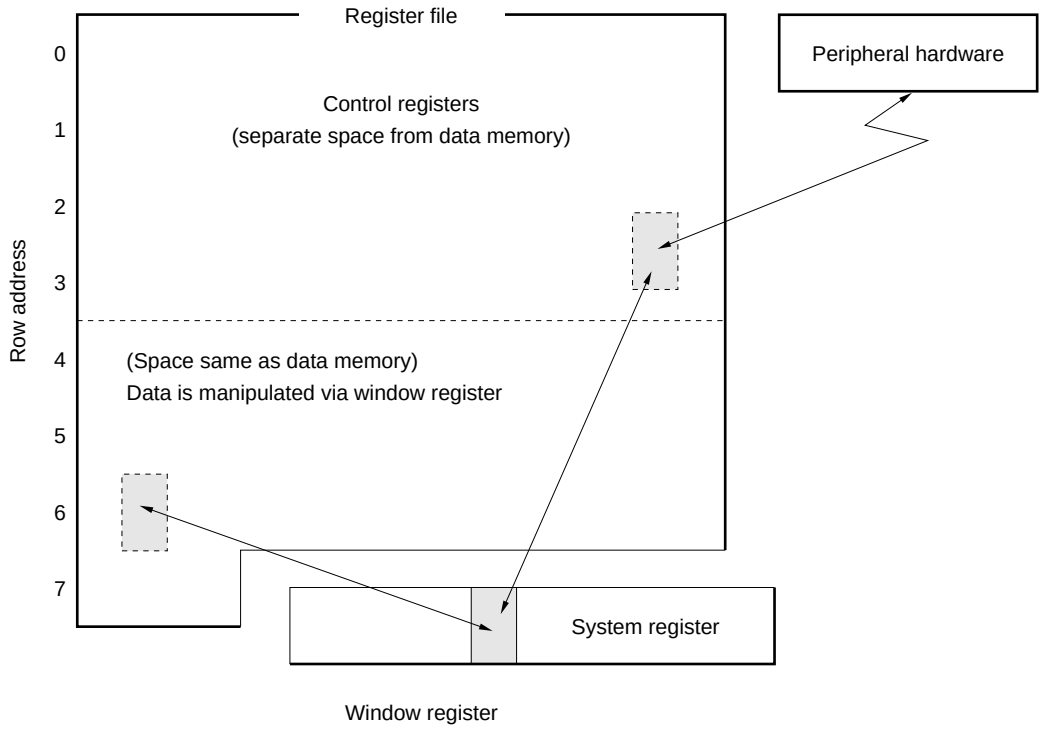
Figure 8-1 outlines the register file.

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

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

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

Figure 8-1. Outline of Register File



8.2 Configuration and Function of Register File

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

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

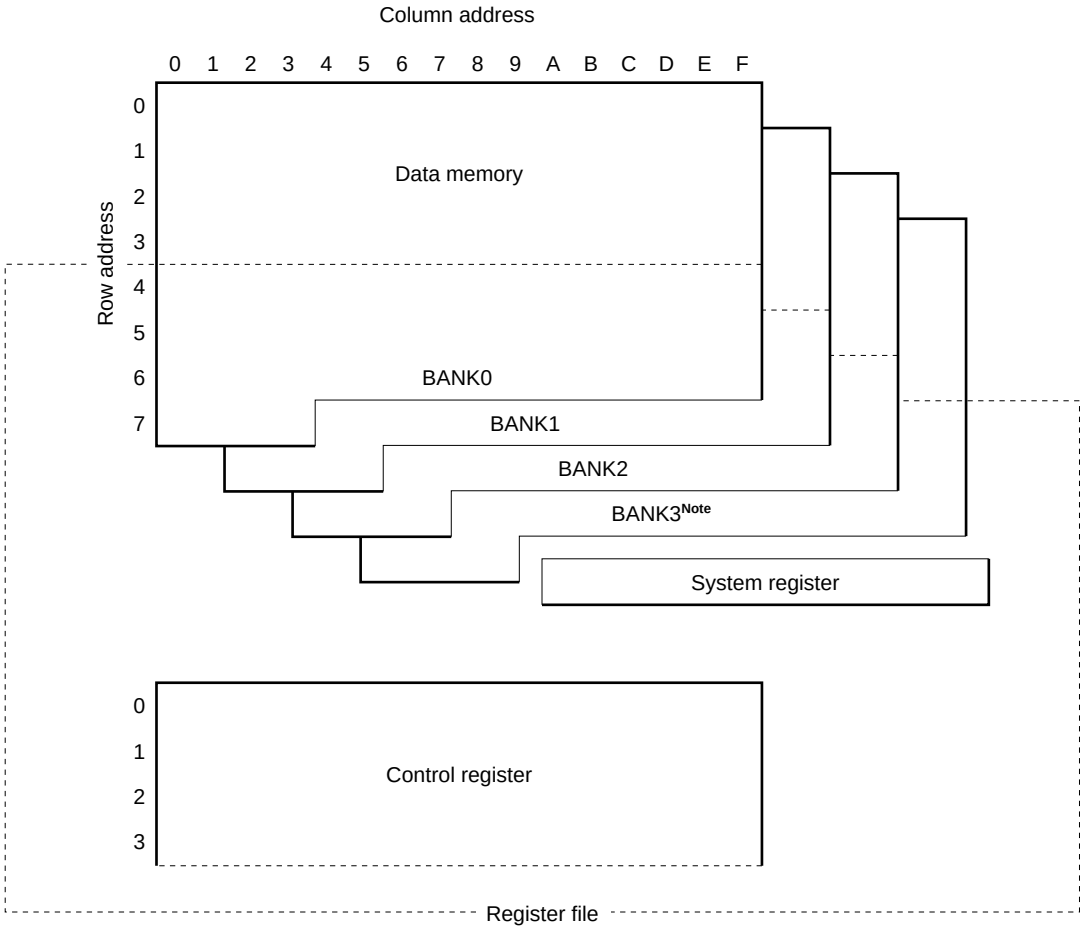
Control registers that set the conditions of the peripheral hardware units are allocated to addresses 00H through 3FH.

Addresses 40H through 7FH overlap the data memory.

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

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

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



Note BANK3 is not provided to the μPD17016.

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

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

(1) “PEEK WR, rf”

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

(2) “POKE rf, WR”

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

8.3 Control Registers

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

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

Of these nibbles, however, 19 nibbles are actually used. The remaining 45 nibbles are unused registers that are prohibited from being read or written.

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

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

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

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

The 45 nibbles of unused registers are undefined when they are read, and nothing is changed when data is written to them.

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

Colum address / Row address	Item	0	1	2	3	4	5	6	7
0 (8)	Name		Stack pointer (SP)	Serial I/O mode select		I/F count gate judge	PLL unlock FF judge	A/D converter compare judge	CE pin level judge
	Symbol		(SP3) (SP2) (SP1) (SP0)	SIO1TS SIO1HIZ SIO1CK1 SIO1CK0		0 0 0 IFCGOSTT	0 0 0 PLLUL	0 0 0 ADCCMP	0 0 0 CE
	Read/ Write		R/W	R/W		R	R & Reset	R	R
1 (9)	Name	LCD mode select		IF counter mode select		A/D converter channel select		Key input judge	Timer carry FF judge
	Symbol	0 0 KSEN LCDEN		IFCMD1 IFCMD0 IFCCK1 IFCCK0		ADCCH3 ADCCH2 ADCCH1 ADCCH0		0 0 0 KEYJ	0 0 0 BTM0CY
	Read/ Write	R/W		R/W		R & Reset		R & Reset	R & Reset
2 (A)	Name		PLL mode select		IF counter control				Port 0C group I/O select
	Symbol		PLLMD3 PLLMD2 PLLMD1 PLLMD0		0 0 IFCSTRT IFCRES				0 0 0 P0CGIO
	Read/ Write		R/W		W				R/W
3 (B)	Name		PLL reference mode select				Port 1A bit I/O select	Port 0B bit I/O select	Port 0A bit I/O select
	Symbol		PLLRFCK3 PLLRFCK2 PLLRFCK1 PLLRFCK0				P1ABIO3 P1ABIO2 P1ABIO1 P1ABIO0	P0BBIO3 P0BBIO2 P0BBIO1 P0BBIO0	P0ABIO3 P0ABIO2 P0ABIO1 P0ABIO0
	Read/ Write		R/W				R/W	R/W	R/W

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

8	9	A	B	C	D	E	F
	Timer mode select						
	BTM1CK1, BTM1CK0, BTM0CK1, BTM0CK0						
	R/W						

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

Peripheral hardware	Control register				Peripheral hardware control function		On reset		
	Name	Address	Read/ Write	b3 b2 Symbol b1 b0	Functional outline	Set value	Power ON	Clock stop	CE
Stack	Stack pointer (SP)	01H	R/W	(SP3) (SP2) (SP1) (SP0)	0 Stack pointer		7	7	7
Timer	Timer mode select	09H	R/W	BTM1CK1 BTM1CK0	Sets timer interrupt time.	0 0 1 1 100 ms 250 ms 5 ms 1 ms 0 1 1 1	0	0	Retained
				BTM0CK1 BTM0CK0	Sets time of timer carry FF.	0 0 1 1 100 ms 250 ms 5 ms 1 ms 0 1 1 1			
	Timer carry FF judge	17H	R & Reset	0 0 0 BTM0CY	Detects timer carry FF	0: FF reset, 1: FF set	0	1	1
Pin	CE pin level judge	07H	R	0 0 0 CE	Detects CE pin status	0: low level, 1: high level	–	–	–
PLL frequency synthesizer	PLL unlock FF judge	05H	R & Reset	0 0 0 PLLUL	Detects unlock FF status	0: lock, 1: unlock Read & reset	Undefined	Retained	Retained
	PLL mode select	21H	R/W	PLLMD3 PLLMD2 PLLMD1 PLLMD0	0 0 Sets division mode of PLL	0 0 1 1 disable MF VHF HF 0 1 0 1	0	0	Retained
	PLL reference mode select	31H	R/W	PLLRFCK3 PLLRFCK2 PLLRFCK1 PLLRFCK0	Sets reference frequency of PLL	0:1.25 kHz 1:2.5 kHz 2:5 kHz 3:10 kHz 4:6.25 kHz 5:12.5 kHz 6:25 kHz 7:50 kHz 8:3 kHz 9:A:B: setting prohibited C:1 kHz D:9 kHz E:100 kHz F: off	F	F	Retained
A/D converter	A/D converter channel select	14H	R/W	ADCCH3 ADCCH2 ADCCH1 ADCCH0	0 Selects pin used for A/D converter	0:AD_0 1:AD_1 2:AD_2 3:AD_3 4:AD_4 5:AD_5 6:7:Input port	7	7	7
	A/D converter compare judge	06H	R	0 0 0 ADCCMP	Detects result of comparison by A/D converter	0:V_{REF}>V_{ADCIN} 1:V_{REF}<V_{ADCIN}	Undefined	Retained	Retained

– : Determined by the status of the pin.

Table 8-1. Outline of Peripheral Hardware Control Functions of Control Registers (2/3)

Peripheral hardware	Control register				Peripheral hardware control function		On reset		
	Name	Address	Read/ Write	b3 b2 Symbol b1 b0	Functional outline	Set value	Power ON	Clock stop	CE
General-purpose port	Port 0C group I/O select	27H	R/W	0 0 0 P0CGIO	Sets synchronously input/output mode of $P0C_3$-$P0C_0$ pins	0: Input, 1: Output	0	0	0
	Port 1A bit I/O select	35H	R/W	P1ABIO3 P1ABIO2 P1ABIO1 P1ABIO0	$P1A_3$ pin $P1A_2$ pin $P1A_1$ pin $P1A_0$ pin	Sets input/output mode of each port pin 0: Input, 1: Output	0	0	0
	Port 0B bit I/O select	36H	R/W	P0BBIO3 P0BBIO2 P0BBIO1 P0BBIO0	$P0B_3$ pin $P0B_2$ pin $P0B_1$ pin $P0B_0$ pin				
	Port 0A bit I/O select	37H	R/W	P0ABIO3 P0ABIO2 P0ABIO1 P0ABIO0	$P0A_3$ pin $P0A_2$ pin $P0A_1$ pin $P0A_0$ pin				
Serial interface	Serial I/O mode select	02H	R/W	SIO1TS SIO1HIZ SIO1CK1 SIO1CK0	Sets start of serial interface. Sets $P0B_1$/SO pin. Sets clock of serial interface.	0: Does not operate, 1: Starts 0: General-purpose port, 1: Serial out 0 0 1 1 External 75 kHz 150 kHz 450 kHz 0 1 0 1	0	0	0
Frequency counter	IF counter gate judge	04h	R	0 0 0 IFCGOSTT	Detects opening/closing of gate of frequency counter	0: Close, 1: Open	0	Undefined	Undefined
	IF counter mode select	12H	R/W	IFCMD1 IFCMD0 IFCCK1 IFCCK0	Selects frequency counter/BEEP mode. Sets gate time of frequency counter.	0 0 1 1 BEEP FMIF AMIF Setting prohibited 0 1 0 1 0 0 1 1 1 ms 4 ms 8 ms Open 0 1 1 1	0	0	Retained
	IF counter control	23H	R/W	0 0 IFCSTRT IFCRES	Specifies start of frequency counter. Specifies data reset of frequency counter.	0: NOP instruction, 1: Start 0: NOP instruction, 1: Reset	0	0	Retained

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

Peripheral hardware	Control register				Peripheral hardware control function		On reset		
	Name	Address	Read/ Write	b3 b2 Symbol b1 b0	Functional outline	Set value	Power ON	Clock stop	CE
LCD driver	LCD mode select	10H	R/W	0					
				0					
				KSEN	Sets key source signal output.	0: Key source off, 1: Key source on	0	0	Retained
				LCDEN	Sets LCD display output.	0: Display off, 1: Display on			
	Key input judge	16H	R & Reset	0					
				0					
				0					
				KEYJ	Detects key input latch of LCD key source	0: No latch, 1: Latch	0	Undefined	Undefined

8.4 Notes on Using Register File

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

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

9. DATA BUFFER (DBF)

9.1 Outline of Data Buffer

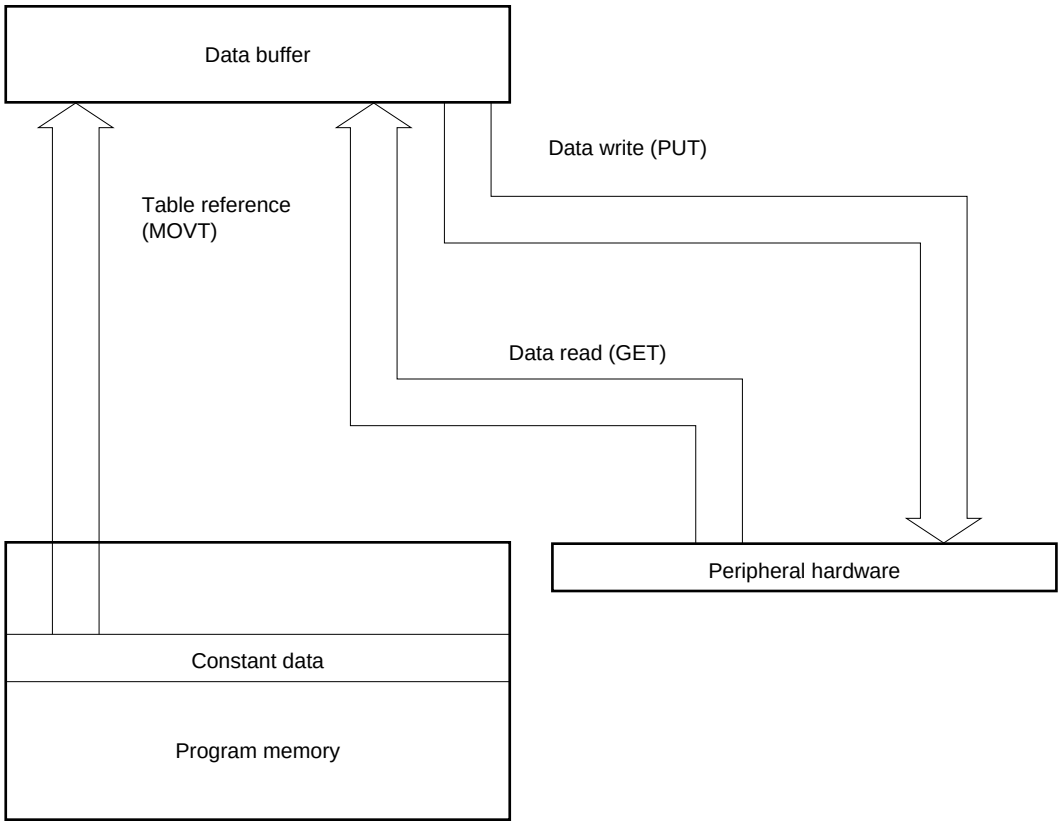
Figure 9-1 outlines the data buffer.

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

(1) Reads constant data on program memory (table reference)

(2) Transfers data with peripheral hardware unit

Figure 9-1. Outline of Data Buffer



9.2 Data Buffer

9.2.1 Configuration of data buffer

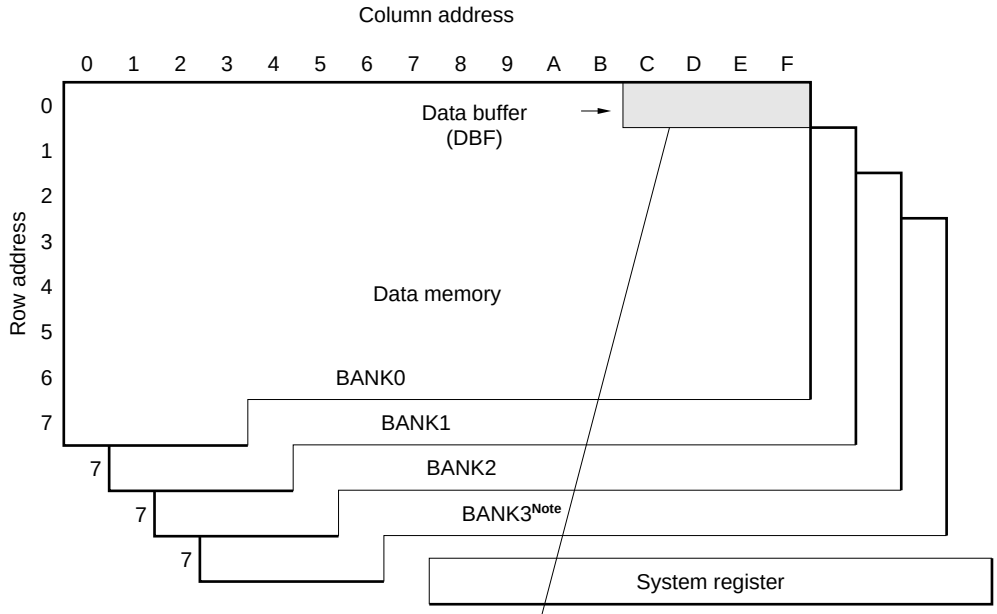
Figure 9-2 shows the configuration of the data buffer.

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

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

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

Figure 9-2. Configuration of Data Buffer



Note BANK3 is not provided on the μPD17016

Data memory	Address	0CH				0DH				0ED				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	MSB				Data								LSB			

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

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

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

With the μPD17017 all the addresses of the program memory can be referenced by this instruction, but with the μPD17016 only the address 0000H through 00FFH can be referenced. ★

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

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

(1) GET DBF, p

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

(2) PUT p, DBF

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

9.3 Peripheral Hardware and Data Buffer List

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

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

Peripheral hardware		Peripheral register that transfers data with data buffer			
		Name	Symbol	Peripheral address	Execution of PUT/GET instruction
A/D converter		A/D converter data register	ADCR	02H	PUT/GET
Serial interface	Serial interface 1 (SIO1)	Presettable shift register	SIO1SFR	03H	PUT/GET
D/A converter (PWM output)	PWM_0 pin	PWM data register 0	PWMR0	05H	PUT/GET
	PWM_1 pin	PWM data register 1	PWMR1	06H	
Address register (AR)		Address register	AR	40H	PUT/GET
PLL frequency synthesizer		PLL data register	PLLR	41H	PUT/GET
Key source controller/decoder		Key source data register	KSR	42H	PUT/GET
Frequency counter		IF counter data register	IFC	43H	GET

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

Function		
Number of I/O bits of data buffer	Number of bits actually used	Outline
8	6	Sets compare voltage V_{REF} data of A/D converter $V_{REF} = \frac{x - 0.5}{64} \times V_{DD}$, $1 \leq x \leq 63$
8	8	Sets serial out data and reads serial in data.
8	8	Sets duty factor of output signal of D/A converter. Duty $D = \frac{x + 0.25}{256} \times 100\ \%$, $0 \leq x \leq 255$ Frequency $f = 878.9$ Hz
16	13[Note]	Transfers data with address register.
16	16	Sets division ratio (N value) of PLL.
16	16	Sets output data of key source signal.
16	16	Reads count value of frequency counter.

Note 8 bits in the case of the μPD17016.

9.4 Notes on Using Data Buffer

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

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

10. INTERRUPT

10.1 Outline of Interrupt Block

Figure 10-1 outlines the interrupt block.

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

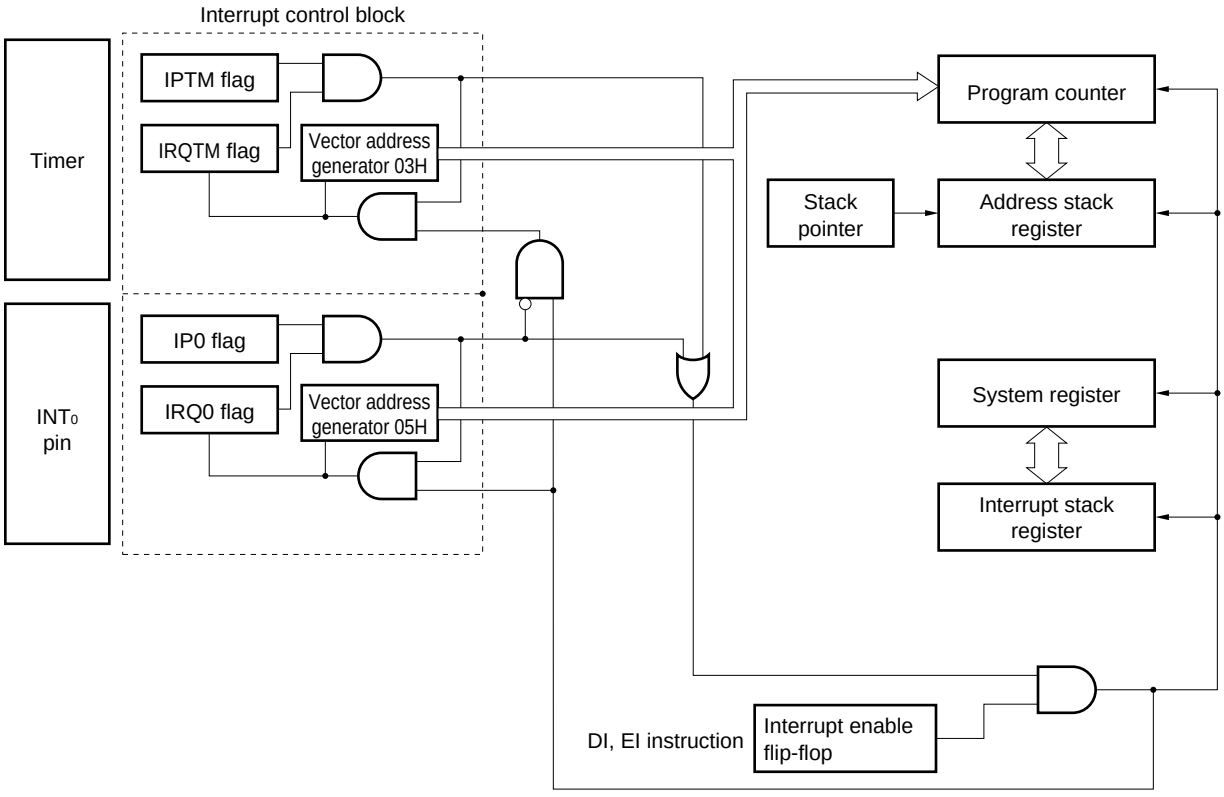
The interrupt block consists of an “interrupt control block” for each peripheral hardware unit, “interrupt enable flip-flop” that enables all the interrupts, “stack pointer” that is controlled when an interrupt is acknowledged, “address stack register”, “program counter”, and “system register stack”.

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

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

- INT_0 pin (rising-edge detection)
- Timer

Figure 10-1. Outline of Interrupt Block



10.2 Interrupt Control Block

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

10.2.1 Configuration and function of interrupt request flag (IRQ×××)

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

Once this flag has been set to "1", it is not reset until the corresponding interrupt is acknowledged or an interrupt request reset macro is executed.

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

Table 10-1 shows the software macros that reset interrupt requests.

These macros are defined in the device file and can be used by INCLUDEing the files supplied with the device file (D17016.INC for the μPD17016 and D17017.INC for the μPD17017) on the source program.

Table 10-1. Software Macros That Reset Interrupt Requests

Function	Macro Name
Resets timer interrupt request.	CLR1_IRQTM
Resets INT_0 pin interrupt request.	CLR1_IRQ0

Caution If the above macros are used, the contents of the window register are destroyed.
If an embedded macro instruction is used immediately before the above macro instructions, an "object error" occurs when the source file is assembled and then loaded to the in-circuit emulator. To use an embedded macro instruction immediately before the above macro instructions, insert a comment statement between them.
Moreover, while either of the above macro instructions is executed, the other interrupt request cannot be acknowledged. For details, refer to 10.11 Notes on Using Interrupt.

10.2.2 Configuration and function of interrupt permission flag (IP×××)

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

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

Enabling or disabling an interrupt is controlled by a software macro.
Table 10-2 lists the available software macros.
These macros are defined in the device file and can be used by INCLUDEing the files supplied with the device file (D17016.INC for the μPD17016 and D17017.INC for the μPD17017) on the source program.

Table 10-2. Software Macros Enabling/Disabling Interrupts

Function	Macro Format
Enables timer interrupt	SET1_IPTM
Enables INT_0 pin interrupt	SET1_IP0
Enables timer interrupt and INT_0 pin interrupt	SET2_IPTM_IP0
Disables timer interrupt	CLR1_IPTM
Disables INT_0 pin interrupt	CLR1_IP0
Disables timer interrupt and $INTP_0$ pin interrupt	CLR2_IPTM_IP0
Disables timer interrupt and enables INT_0 pin interrupt	INIT_NOT_IPTM_IP0
Enables timer interrupt and disables INT_0 pin interrupt	INIT_IPTM_NOT_IP0

Caution If the above macros are used, the contents of the window register are destroyed.
If an embedded macro instruction is used immediately before the above macro instructions, an "object error" occurs when the source file is assembled and then loaded to the in-circuit emulator. To use an embedded macro instruction immediately before the above macro instructions, insert a comment statement between them.

The status of the interrupt at reset is as follows:

- **On power-ON reset**
  The interrupt is disabled.

- **On execution of clock stop instruction**
  The interrupt is disabled.

- **On CE reset**
  The interrupt is disabled.

- **In halt status**
  The interrupt permission flag holds the previous status.

10.2.3 Vector address generator (VAG)

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

Table 10-3 shows the vector address of each interrupt source.

Table 10-3. Vector Address of Each Interrupt Source

Interrupt Source	Vector Address
INT_0 pin	05H
Timer	03H

10.3 Interrupt Stack Register

10.3.1 Configuration and function of interrupt stack register

Figure 10-2 shows the configuration of the interrupt stack register and the system register whose contents are saved to the interrupt stack register.

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

- Bank register (BANK)
- Index enable flag (IXE)

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

The interrupt stack register can save up to 4 levels of the contents of the above system registers.

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

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

Figure 10-2. Configuration of Interrupt Stack Register

Interrupt stack register (INTSK)								
Name	Bank stack				Status stack			
Bit	b_3	b_2	b_1	b_0	b_3	b_2	b_1	b_0
0H	–	–			–	–	–	
1H	–	–			–	–	–	
2H	–	–			–	–	–	
3H	–	–			–	–	–	

Remark –: Bit not saved

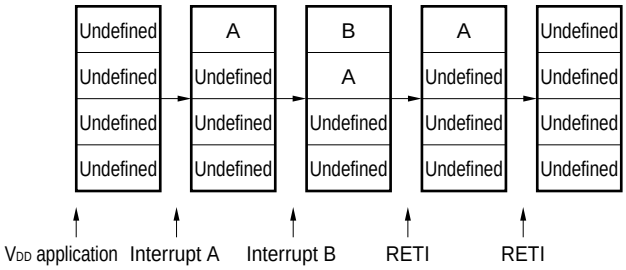
10.3.2 Interrupt stack register operation

Figure 10-3 illustrates the operation of the interrupt stack register.

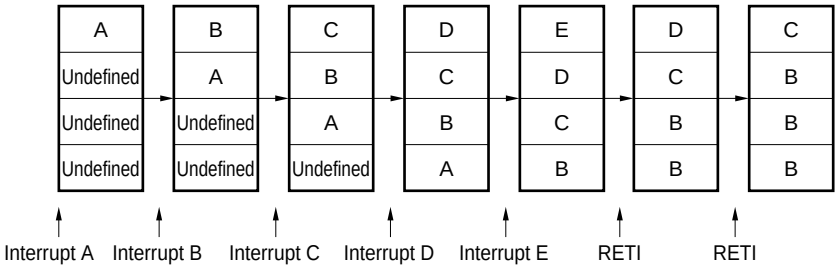
If multiple interrupts exceeding 4 levels are acknowledged, the first saved contents are discarded and therefore, must be saved by program.

Figure 10-3. Operation of Interrupt Stack Register

(a) If interrupt does not exceed level 4



(b) If interrupt exceeds level 4



10.4 Stack pointer, address stack register, and program counter

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

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

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

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

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

10.5 Interrupt Enable Flip-Flop (INTE)

The interrupt enable flip-flop enables all the interrupts.

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

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

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

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

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

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

10.6 Acknowledging Interrupts

10.6.1 Acknowledging interrupts and priority

An interrupt is acknowledged in the following procedure:

(1) Each peripheral hardware unit outputs an interrupt request signal to the corresponding interrupt control block if a given interrupt condition is satisfied (e.g., if a rising signal is input to the INT_0 pin).
(2) When the interrupt control block has received the interrupt request signal from the peripheral hardware unit, it sets the corresponding interrupt request flag (e.g., IRQ0 flag if the peripheral unit is the INT_0 pin) to "1".
(3) If the interrupt permission flag corresponding to the interrupt request flag (e.g., IP0 flag for IRQ0 flag) is set to "1" when the interrupt request flag is set to "1", the interrupt control block outputs "1".
(4) The signal output by the interrupt control block is ANDed with the output of the interrupt enable flip-flop, and an interrupt acknowledge signal is output.
This interrupt enable flip-flop is set to "1" by the "EI" instruction and reset to "0" by the "DI" instruction.
If the interrupt control block outputs "1" while the interrupt enable flip-flop is "1", the interrupt is acknowledged.

As shown in Figure 10-1, the interrupt acknowledge signal is input to each interrupt control block when the interrupt has been acknowledged.

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

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

INT_0 pin > timer

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

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

10.6.2 Timing chart for acknowledging interrupt

Figure 10-4 shows the timing chart illustrating acknowledging interrupts.

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

(a) in (1) shows the case where the interrupt request flag is lastly set to “1”, and (b) in (1) shows the case where the interrupt permission flag is lastly set to “1”.

In either case, the interrupt is acknowledged when all the interrupt request flag, interrupt enable flip-flop, and interrupt permission flag are set to “1”.

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

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

(2) in Figure 10-4 illustrates how more than one interrupt is used.

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

“Interrupt cycle” shown in Figure 10-4 is a special cycle in which the interrupt request flag is reset, a vector address is specified, and the contents of the program counter are saved after an interrupt has been acknowledged, and lasts for 4.44 μs, which is equivalent to the execution time of one instruction.

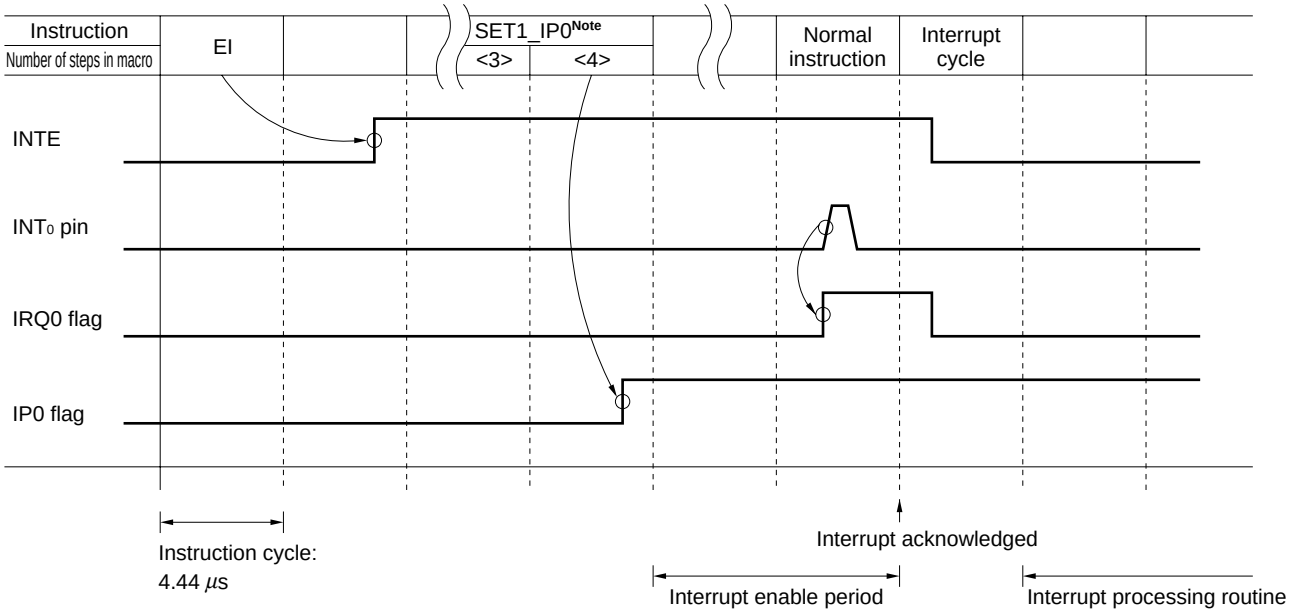
For further information, refer to **10.7 Operations after Interrupt Has Been Acknowledged**.

Figure 10-4. Timing Chart of Acknowledging Interrupt (1/3)

(1) When one interrupt (e.g., rising of INT0 pin) is used

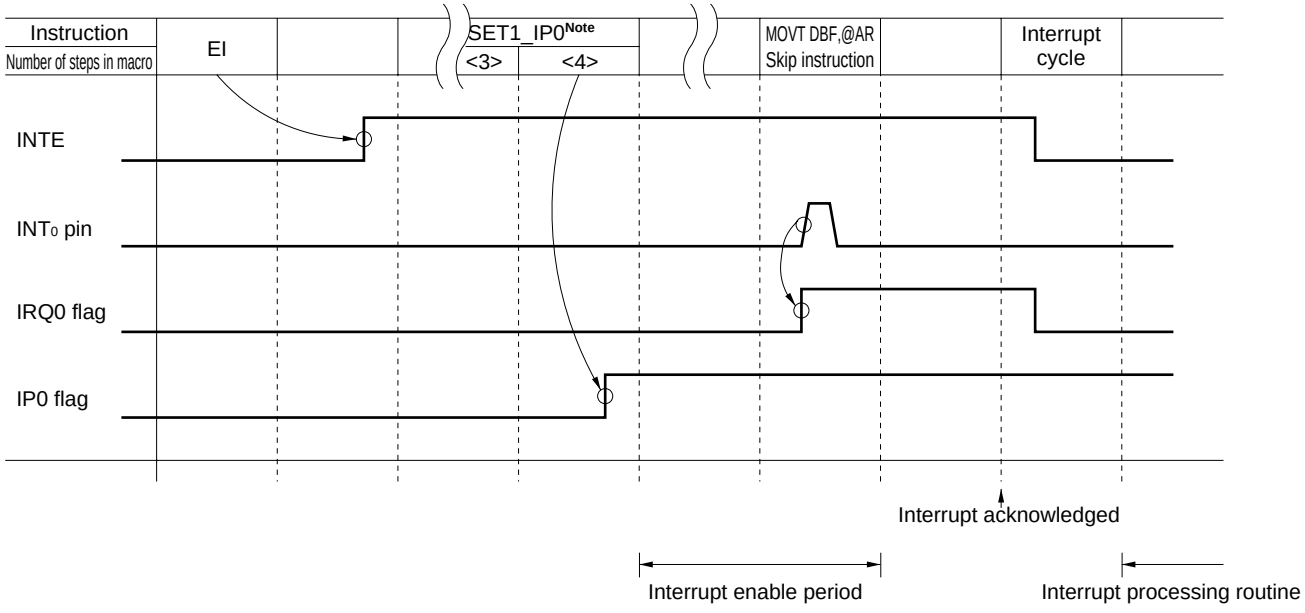
(a) If interrupt is not masked by interrupt permission flag (IP×××)

<1> If "MOVT" instruction or normal instruction that satisfies skip condition is not executed when interrupt is acknowledged



Note Software macro SET1_IP0 consists of four instructions.

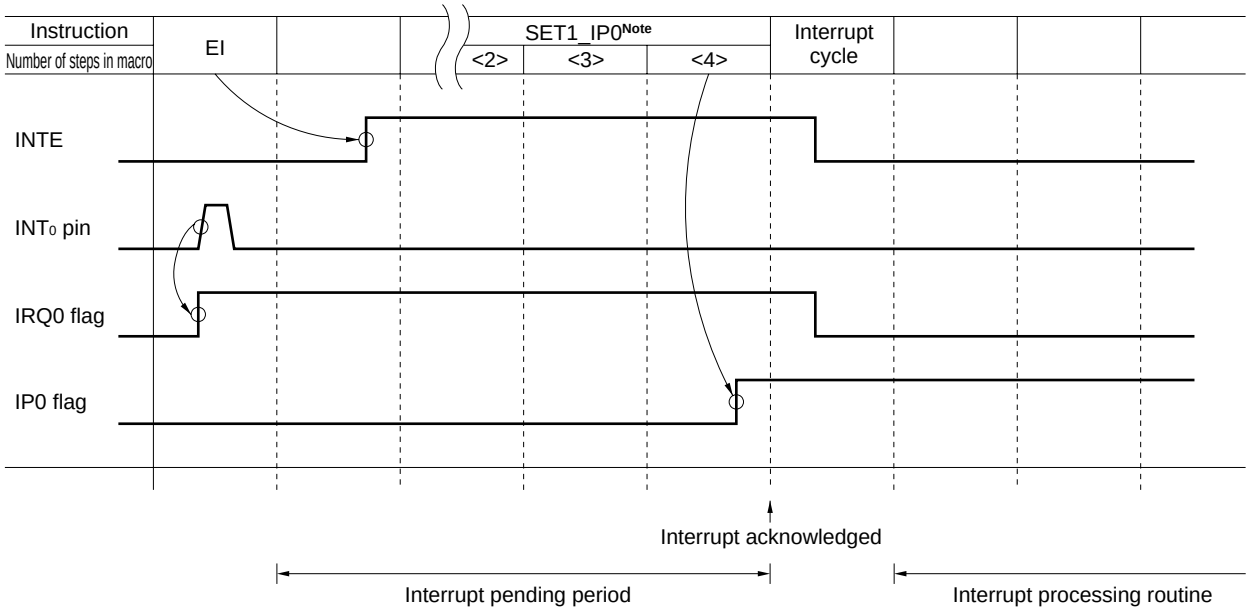
<2> If "MOVT" instruction or "instruction satisfying skip condition" is executed when interrupt is acknowledged



Note Software macro SET1_IP0 consists of four instructions.

Figure 10-4. Timing Chart of Acknowledging Interrupt (2/3)

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

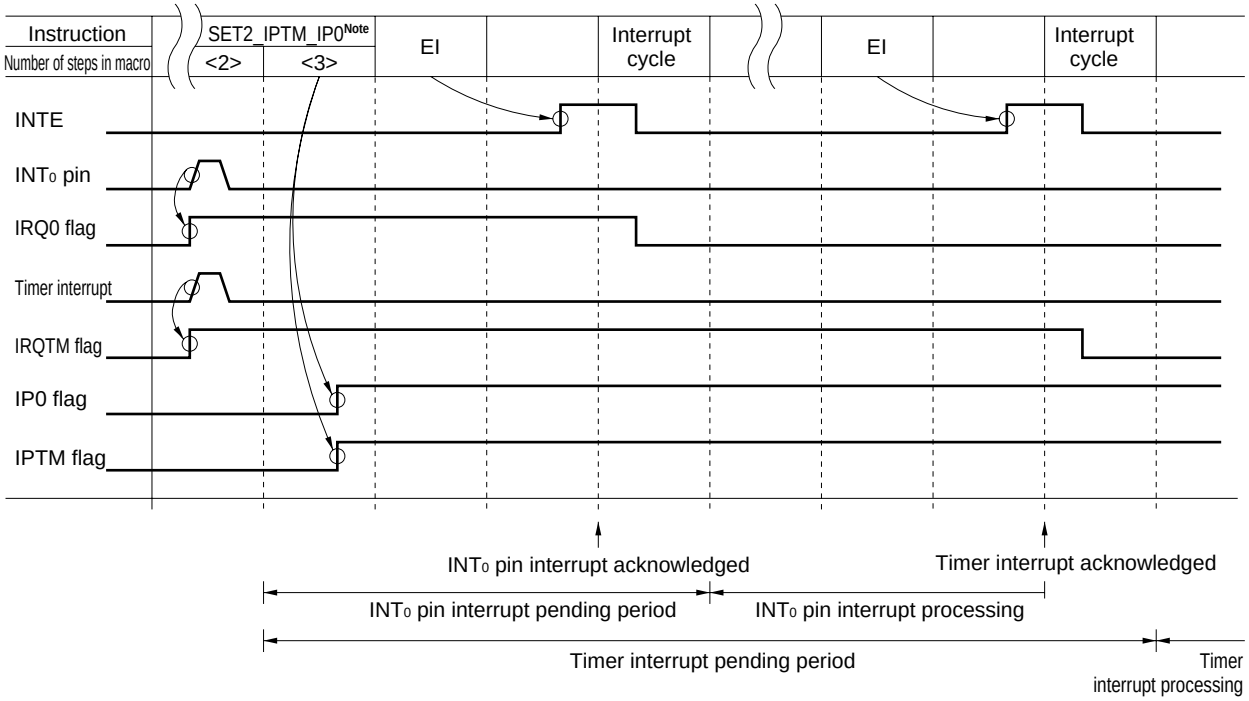


Note Software macro SET1_IP0 consists of four instructions.

Figure 10-4. Timing Chart of Acknowledging Interrupt (3/3)

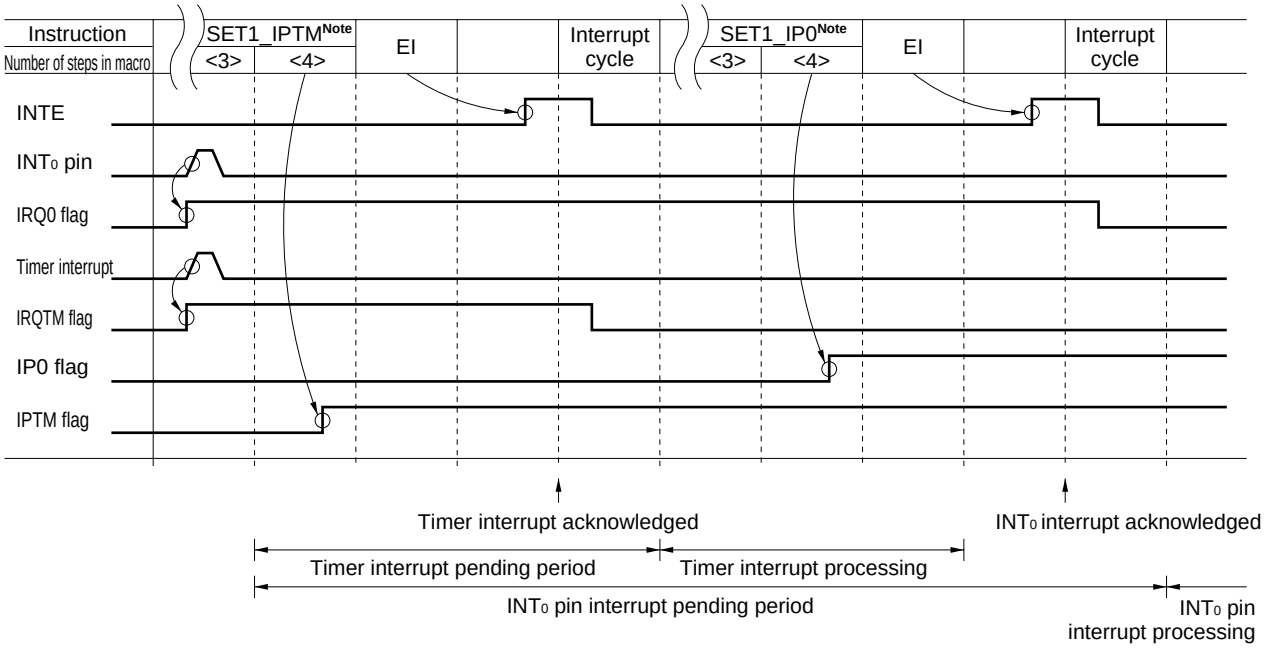
(2) When two interrupts (INT0 pin and timer) are used

(a) Hardware priority



Note Software macro SET2_IPTM_IP0 consists of three instructions.

(b) Software priority



Note Software macro SET1_IPTM, SET1_IP0 consists of four instructions.

10.7 Operation After Interrupt Has been Acknowledged

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

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

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

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

10.8 Restoring from Interrupt Processing Routine

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

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

(1) The contents of the address stack register specified by the stack pointer are saved to the program counter.
(2) The contents of the interrupt stack are restored to the low-order 2 bits of the bank register (BANK) and index enable flag (IXE).
(3) The contents of the stack pointer are incremented by one.

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

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

10.9 External (INT_0 Pin) Interrupt

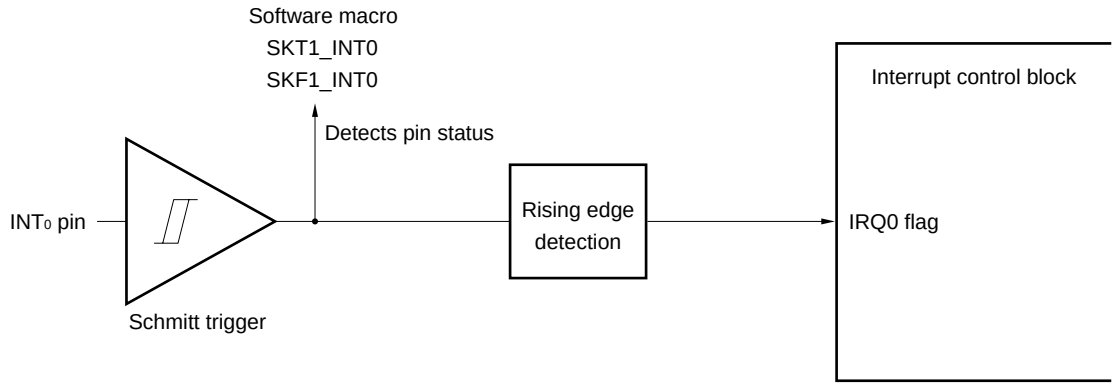
10.9.1 Outline of external interrupt

Figure 10-5 outlines the external interrupt.

As shown in the figure, the external interrupt issues an interrupt request at the rising edge of the signal input to the INT_0 pin.

The INT_0 pin is a Schmitt trigger input pin to prevent malfunctioning due to noise, and does not accept a pulse less than 1 μs wide.

Figure 10-5. Outline of External Interrupt



10.9.2 Edge detection block

The edge detection block detects the rising edge that issues the interrupt request of the INT_0 pin.

10.9.3 Pin status detection function

The level of the signal input to the INT_0 pin can be detected by using software macros.

Therefore, the INT_0 pin can be used as a 1-bit general-purpose input port when it is not used as an interrupt input pin.

Table 10-4 shows the software macros.

These macros are defined in the device file and can be used by INCLUDEing the file supplied with the device file (D17016.INC for the μPD17016 and D17017.INC for the μPD17017) on the source program.

Table 10-4. Software Macros Detecting INT_0 Pin Level

Function	Macro Format
Skips next instruction if high level is input to INT_0 pin	SKT1_INT0
Skips next instruction if low level is input to INT_0 pin	SKF1_INT0

Caution If the above macros are used, the contents of the window register are destroyed.
If an embedded macro instruction is used immediately before the above macro instructions, an "object error" occurs when the source file is assembled and then loaded to the in-circuit emulator. To use an embedded macro instruction immediately before the above macro instructions, insert a comment statement between them.

10.10 Internal Interrupt

The timer interrupt is used as the internal interrupt and can issue an interrupt request at fixed time intervals (refer to **11. TIMER FUNCTION**).

10.11 Notes on Using Interrupt

Other interrupts are not acknowledged while a software macro that resets an interrupt request is being executed.

Example 1.

```
  ⋮
  Processing A
  ⋮
  ⋮

  CLR1_IRQTM      ← Interrupt request of INT0 pin is not acknowledged nor kept pending even if it is
                    issued during this period.
  ⋮
  ⋮

  Processing B
  ⋮
```

Example 2.

```
  ⋮
  Processing A
  ⋮
  ⋮

  CLR1_IRQ0       ← Timer interrupt request is not acknowledged nor kept pending even if it is issued
                    during this period.
  ⋮
  ⋮

  Processing B
  ⋮
```

11. TIMER FUNCTION

The timer function is used for time management in developing a program.

11.1 Configuration of Timer

Figure 11-1 shows the configuration of the timer block.

As shown in the figure, the timer consists of a timer carry block and a timer interrupt block.

The clock generation circuit that sets the time of each timer divides the system clock by using clock select blocks A and B, and the timer mode select register (RF address 09H) of the control registers.

The following subsections 11.1.1 and 11.1.2 respectively explain the configuration of the timer carry block and that of the timer interrupt block.

Because the clock of each timer is created by dividing the system clock (4.5 MHz), if the frequency of the crystal resonator is not 4.5 MHz, the clock of each timer varies accordingly.

11.1.1 Configuration of timer carry block

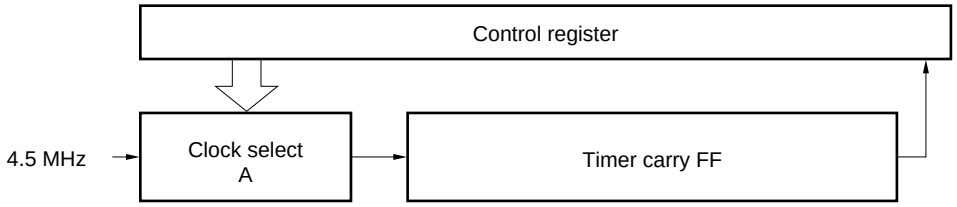
The timer carry block consists of clock select block A and timer carry FF block.

11.1.2 Configuration of timer interrupt block

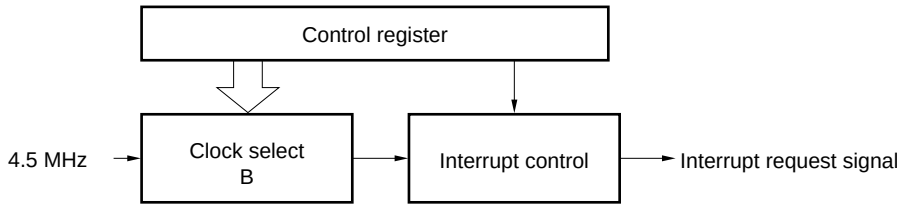
The timer interrupt block consists of clock select block B and interrupt control block.

Figure 11-1. Configuration of Timer Block

- **Timer carry block**



- **Timer interrupt block**



11.2 Functional Outline of Timer

The timer can be used in two ways: by detecting the carry FF of the timer carry and by using the interrupt of the timer.

The following subsections 11.2.1 and 11.2.2 respectively outline the functions of the timer carry and timer interrupt.

11.2.1 Functional outline of timer carry

The timer carry manages time by detecting by program the status of the timer carry FF that is set at fixed intervals.

Refer to **11.3 Timer Carry**.

11.2.2 Functional outline of timer interrupt

The timer interrupt manages time by issuing an interrupt request at fixed time intervals.

Refer to **11.4 Timer Interrupt**.

11.3 Timer Carry

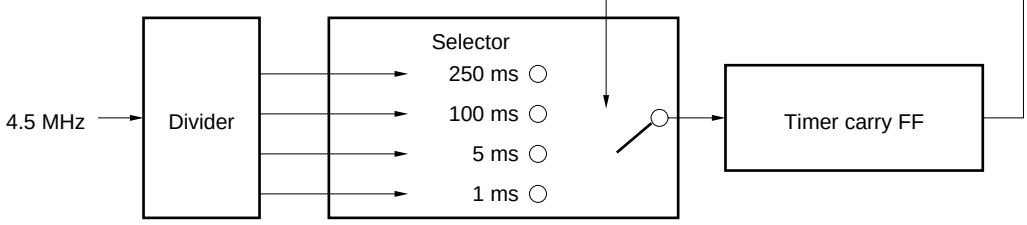
11.3.1 Configuration of timer carry

Figure 11-2 shows the configuration of the timer carry.

As shown in the figure, the timer carry consists of a divider, selector, and timer carry FF block.

Figure 11-2. Configuration of Timer Carry

Control register								
Name	Timer mode select				Timer carry FF judge			
Address	09H				17H			
Bit	b_3	b_2	b_1	b_0	b_3	b_2	b_1	b_0
Flag symbol	BTM1CK1	BTM1CK0	BTM0CK1	BTM0CK0	0	0	0	BTM0CY



11.3.2 Function of timer carry

The timer carry is set to 1 at the rising edge of the timer carry FF setting pulse that is set by the low-order 2 bits (BTM0CK1 and BTM0CK0 flags) of the timer mode select register.

The content of the timer carry FF corresponds to the least significant bit (BTM0CY flag) of the timer carry FF judge register (RF address 17H) on a one-to-one basis. If the timer carry FF is set to "1", therefore, the BTM0CY flag is also set to "1".

The BTM0CY flag is reset to "0" when data is written to the window register by the "PEEK" instruction (Read & Reset).

When the BTM0CY flag is reset to "0", the timer carry FF is also reset to "0".

Therefore, a timer that is set via the timer mode select register can be used by reading the contents of the BTM0CY flag by program.

When using the timer carry, note the following.

Caution The timer carry is disabled from being set on application of the supply voltage V_{DD} (on power-ON reset), and is not set to "1" until the content of the BTM0CY flag is once read by the "PEEK" instruction.

This means that "0" is always read from the BTM0CY flag immediately after power-ON reset, and that the flag is set to "1" at time intervals set by the timer mode select register.

The timer carry also controls the timing of the reset operation by the CE pin (CE reset). If the CE pin goes high, CE reset is effected in synchronization with the timing that sets the timer carry FF to "1".

Therefore, a power failure can be detected by reading the content of the BTM0CY flag at system reset (power-ON reset and CE reset).

For details, refer to **11.3.7 Notes on using timer carry**.

Because the BTM0CY flag is a read-only flag, the device operation is not affected in any way even if data is written to this flag by using the "POKE" instruction. However, an error occurs if the 17K Series assembler is used.

For details, refer to **8.4 Notes on Using Register File**.

11.3.3 Configuration and function of timer mode select register

The timer mode select register sets the time interval of the timer carry and the timer interrupt of the internal timer.

These two time intervals can be set independently.

Figure 11-3 shows the waveform of a timer time setting pulse.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Timer mode select register	BTM1CK1	BTM1CK0	BTM0CK1	BTM0CK0	09H	R/W

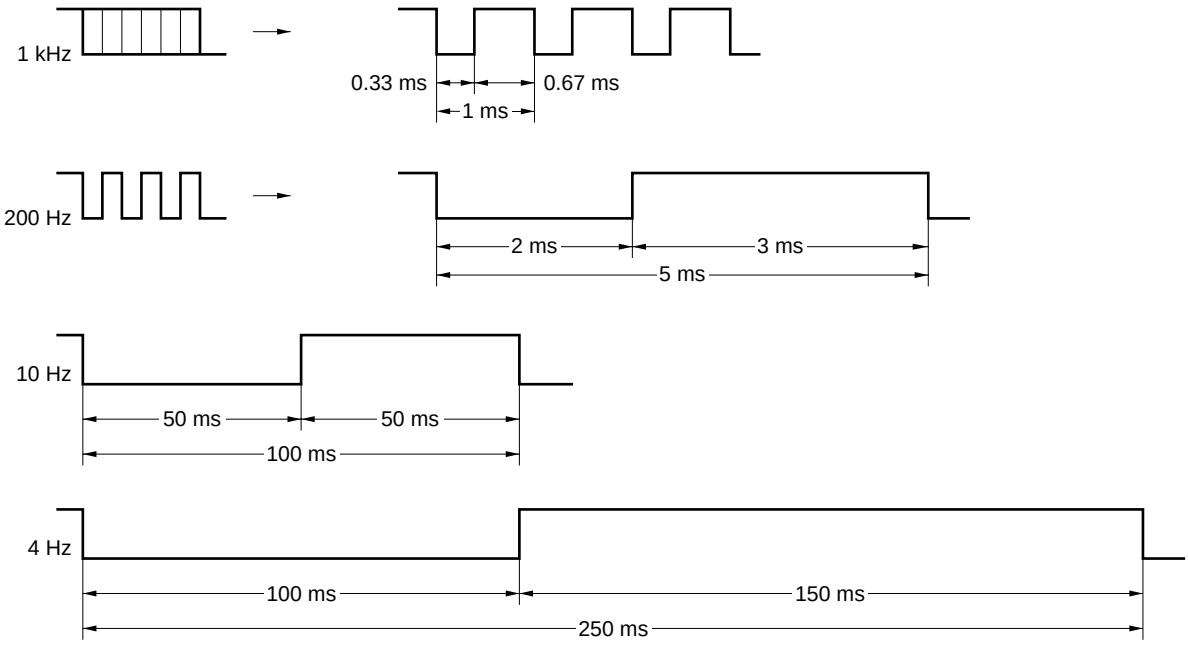
BTM0CK1	BTM0CK0	Sets time interval at which timer carry is set
0	0	100 ms
0	1	250 ms
1	0	5 ms
1	1	1 ms

BTM1CK1	BTM1CK0	Sets time interval at which timer interrupt is set[Note]
0	0	100 ms
0	1	250 ms
1	0	5 ms
1	1	1 ms

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop	0	0	0	0
	CE reset	Holds status			

Note For the details on the timer interrupt, refer to **11.4 Timer Interrupt**.

Figure 11-3. Waveform of Timer Time Setting Pulse



11.3.4 Configuration and function of timer carry FF judge register

The timer carry FF judge register detects the status of the timer carry FF of the internal timer. The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Timer carry FF judge register	0	0	0	BTM0CY	17H	R & Reset

BTM0CY	Detects status of timer carry FF
0	Timer carry FF is not set
1	Timer carry FF is set

Fixed to "0"

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop				1
	CE reset				1

The BTM0CY flag is set to “1” at time intervals set by the timer mode select register.

The BTM0CY flag is detected by the “PEEK” instruction via the window register.

If the BTM0CY flag is set to “1” at this time, the value of the flag is transferred to the window register, and then the BTM0CY flag is reset to “0” (Read & Reset).

The BTM0CY flag is “0” on power-ON reset, but “1” on CE reset (including CE reset that is executed in the clock stop status); therefore, it can be used as a power failure detection flag.

The BTM0CY flag is not set to “1” until it is read once by the “PEEK” instruction after power-ON reset. Once the “PEEK” instruction has been executed, it is set to “1” at the time interval set by the timer mode select register.

11.3.5 Example of using timer with BTM0CY flag

Here is an example program.

Example

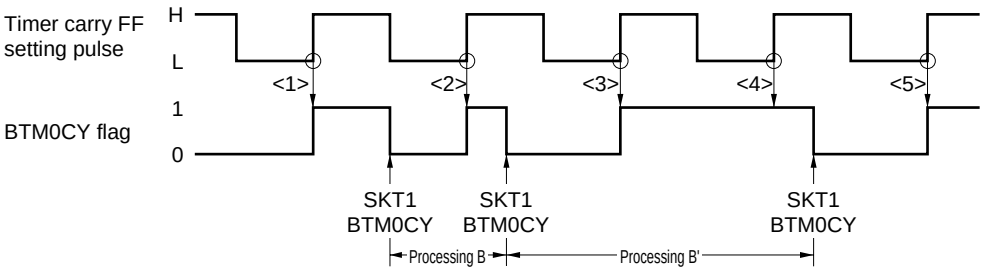
```
 M1       MEM 0.10H      ; 1-second counter
 INITFLG  NOT BTM0CK1, BTM0CK0
                         ; Embedded macro
                         ; Sets set time of timer carry FF to 250 ms
LOOP:
 SKT1     BTM0CY         ; Embedded macro
                         ; Tests BTM0CY flag; if “0”, branches to NEXT
 BR       NEXT
 ADD      M1, #0100B     ; Adds 4 to data memory M1
 SKT1     CY             ; Embedded macro
                         ; Tests CY flag and,
 BR       NEXT           ; if “0”, branches to NEXT
                         ; if “1”, executes processing A
  Processing A

NEXT

  Processing B           ; Executes processing B and branches to LOOP

 BR       LOOP
```

This program executes processing A every 1 second.
In developing a program, the following point must be noted.

Caution Keep the time interval at which the BTM0CY flag is detected shorter than the time interval at which the timer carry FF is set to 1.

This is because, in the above example, if the execution time of processing B is longer than 250 ms, setting of the timer carry FF is not detected.

Figure 11-4. Detection of BTM0CY Flag and Timer Carry FF



Because the execution time of processing B' is long, after the BTM0CY flag set in <2> has been detected, the status of BTM0CY flag that is set in <3> is not detected.

11.3.6 Error of timer caused by BTM0CY flag

The error of the timer caused by the BTM0CY flag includes an error caused by the detection time of the BTM0CY flag and an error that occurs when the set time of the timer carry FF is changed.

(1) Error caused by detection time of BTM0CY flag

As explained in 11.3.5, the time interval at which the BTM0CY flag is detected must be shorter than the time interval at which the timer carry FF is set to 1.

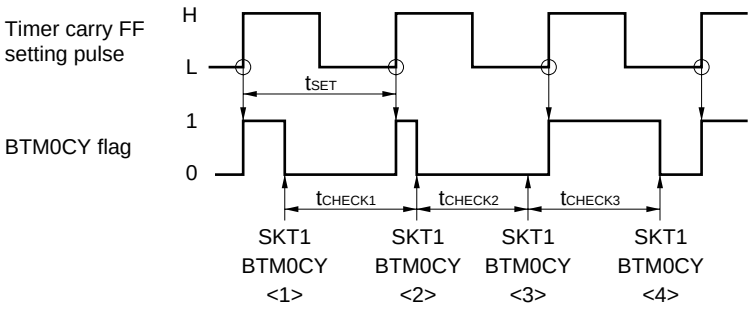
Where the time interval at which the BTM0CY flag is detected is t_{CHECK} and the time interval at which the timer carry FF is set is t_{SET} (250, 100, 5, or 1 ms), the following relation must be satisfied.

$t_{CHECK} < t_{SET}$

The error of the timer when the BTM0CY flag is detected as shown in Figure 11-5 is as follows.

$0 < \text{error} < t_{CHECK}$

Figure 11-5. Error Caused by Detection Time of BTM0CY Flag



As shown in Figure 11-5, the timer is updated when the BTM0CY flag is detected in <2> because it is “1”. When the flag is 0 when it is detected in <3> next. Therefore, the timer is not updated until the flag is detected again in <4>.

At this time, the time of the timer is prolonged by the time of t_{CHECK3}.

(2) Error when changing set time of timer carry FF

The set time of the timer carry FF is set by the BTM0CK1 and BTM0CK0 flags of the timer mode select register.

Four types of timer time setting pulses, 1 kHz, 200 Hz, 10 Hz, and 4 Hz, can be selected as shown in Figures 11-2 and 11-3.

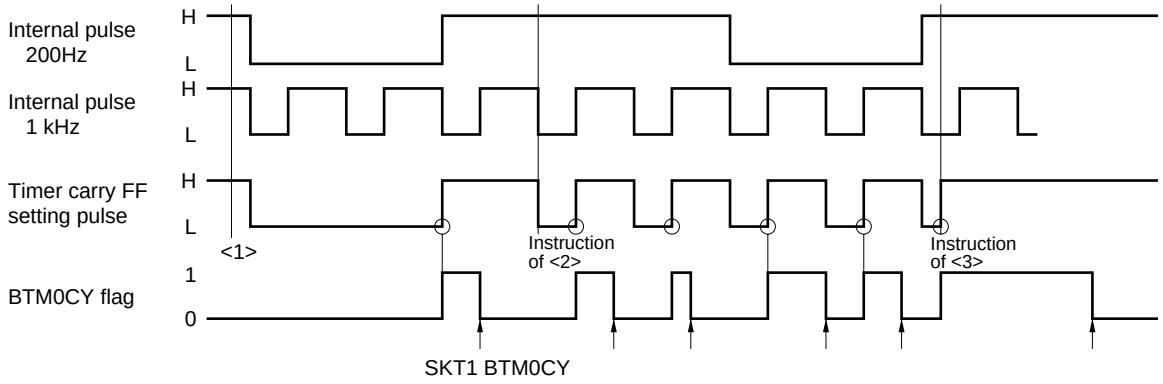
These four pulses operate independently.

If the timer time setting pulse is changed by using the BTM0CK1 and BTM0CK0 flags, an error occurs as shown in the following example.

Example

```
;<1>
 INITFLG  BTM0CK1, NOT BMT0CK0    ; Embedded macro
                                  ; Sets timer carry FF setting pulse to
                                  ; 200 Hz (5 ms)
 Processing A

;<2>
 SET2   BTM0CK1, BTM0CK0          ; Embedded macro
                                  ; Sets timer carry FF setting pulse to
                                  ; 1 kHz (1 ms)
 Processing A

;<3>
 INITFLG  BTM0CK1,NOT BTM0CK0     ; Embedded macro
                                  ; Sets timer carry FF setting pulse to
                                  ; 200 Hz (5 ms)
```

The timer carry FF setting pulse is changed as follows at this time.



When the set time of the timer carry FF is changed as shown above, the BTM0CY flag holds the previous status (<2> in the figure) if the changed pulse falls, but is set to 1 (<3> in the figure) if the pulse rises.

In the previous example, the timer time setting pulse is changed between 200 Hz (5 ms) and 1 kHz (1 ms). The same applies when the pulse is changed between 4 Hz (250 ms) and 10 Hz (100 ms).

Therefore, an error until the BTM0CY flag is set first when the timer carry FF setting time is changed as shown in Figure 11-6 is as follows.

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

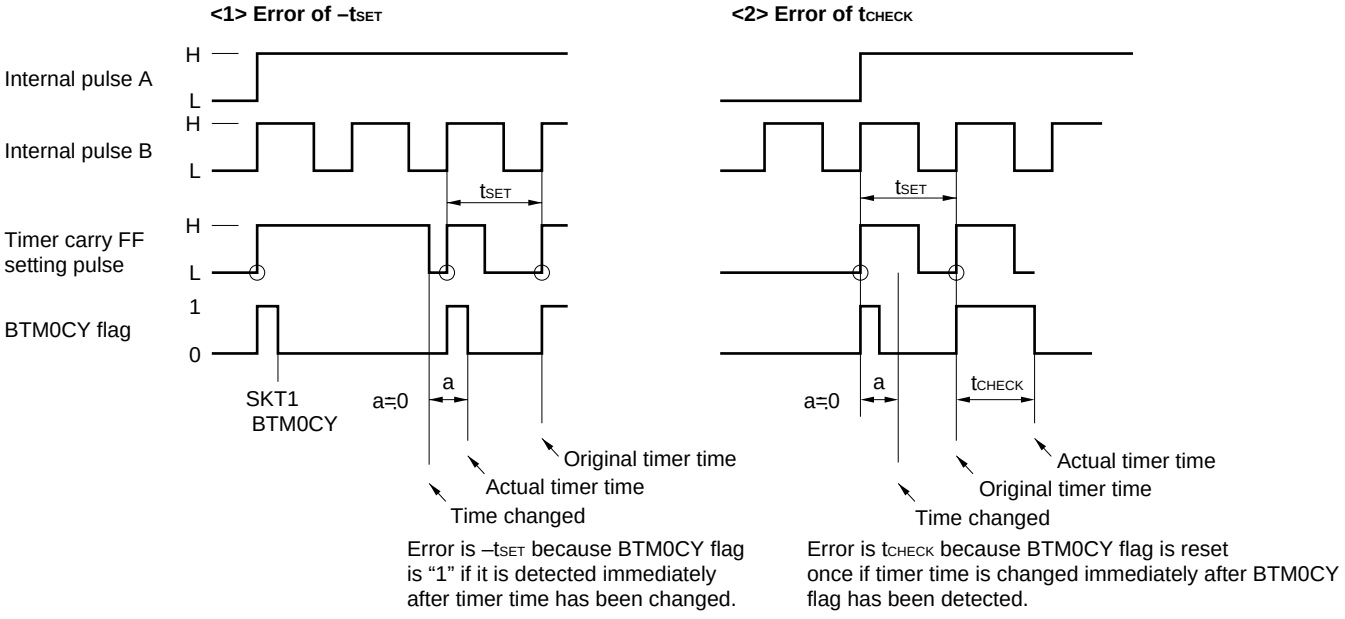
where,

t_{SET} : new timer carry FF setting time
t_{CHECK} : time to detect BTM0CY flag

The internal pulses of 4 Hz, 10 Hz, 200 Hz, and 1 kHz have a phase difference. This phase difference is included in the above error because it is shorter than the time of the new pulse.

For the phase difference of each pulse, refer to **11.4.5 Notes on using timer interrupt**.

Figure 11-6. Error When Timer Carry FF Setting Time Is Changed from A to B



11.3.7 Notes on using timer carry

The timer carry is also used as a reset synchronization signal when reset is effected by the CE pin (CE reset), not only for the timer function.

The CE reset is effected if the timer carry FF setting pulse rises the next time after the CE pin has gone high. At this time, the following points must be noted.

(1) The sum of the timer updating processing time and the BTM0CY flag detection time must be shorter than the timer carry FF setting time.

(2) If a program is created in which the timer always operates at a fixed interval regardless of CE reset after application of the supply voltage (power-ON reset), correction must be made each time CE reset is effected.

(3) Detection of the BTM0CY flag takes precedence over the reset synchronization signal on CE reset. If these two coincide, therefore, CE reset is delayed once.

(1) through (3) above are explained in more detail in the following (a) through (c).

(a) Timer updating processing time and detection time interval of BTM0CY flag

As explained in 11.3.6, the time interval t_{SET} at which the BTM0CY flag is detected must be shorter than the time interval at which the timer carry FF is set.

If the updating processing time of the timer is long even if the time interval to detect the BTM0CY flag is short, the timer processing may not be executed normally if CE reset is effected.

Therefore, the following condition must be satisfied.

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

where,

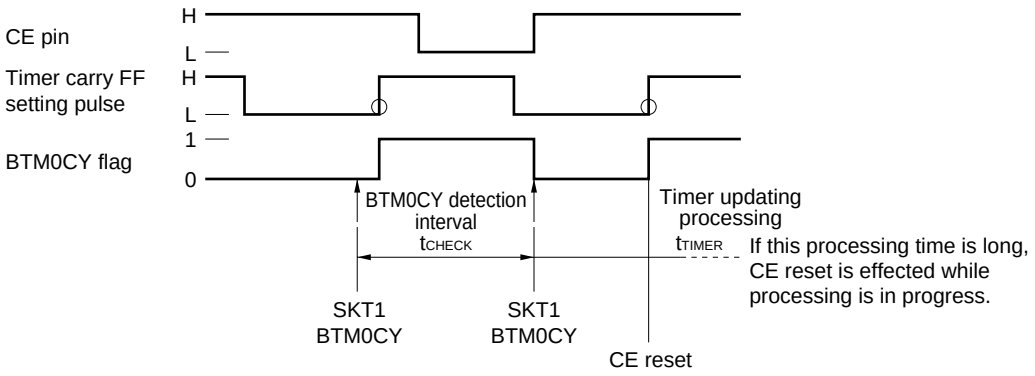
t_{CHECK} : time interval to detect BTM0CY flag
t_{TIMER} : timer updating processing time
t_{SET} : time to set timer carry FF

Here is an example.

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

```
START:                          ; Program address 0000H
      CLR2   BTM0CK1, BTM0CK0
                                ; Embedded macro
                                ; Sets set time of timer carry FF to 100 ms
TIMER:
     ;<1>
           SKT1  BTM0CY         ; Embedded macro
           BR    AAA            ; Tests BTM0CY flag; if “0”, branches to AAA

           [ Timer update ]

           BR    TIMER
AAA:
           [ Processing A ]

           BR    TIMER
```

The timing chart of the above program is shown below.



(b) Correction of timer carry on CE reset

An example of correcting the timer at CE reset is shown below.

As shown in this example, the timer must be corrected on CE reset if "the timer carry is used to detect a power failure and the timer carry is also used as a watch timer".

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

If the CE pin goes high, CE reset is effected in synchronization with the rising edge of the timer carry FF setting pulse. At this time, the BTM0CY flag is set to "1" and the program is started.

Therefore, it is judged that power-ON reset has been effected if the BTM0CY flag is "0" and that CE reset has been effected if the BTM0CY flag is "1", by detecting the status of the BTM0CY flag on system reset (power-ON reset or CE reset). A power failure can be detected in this way.

At this time, the watch timer must continue operating even on CE reset.

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

Therefore, the watch timer must be updated if it is judged by means of power failure detection that CE reset has been effected.

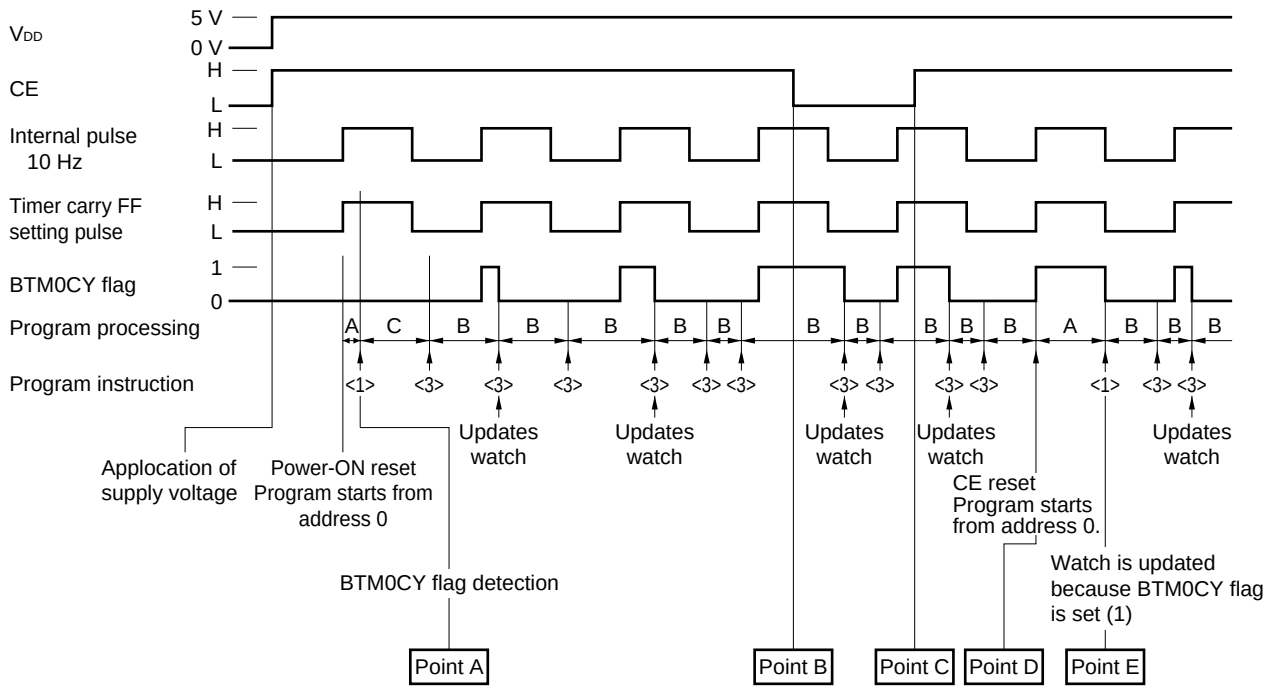
For the details on power failure detection, refer to **13.6 Power Failure Detection**.

Example Example of correcting timer on CE reset (to detect power failure and update watch by timer carry)

```
START:                          ; Program address 0000H

      [ Processing A ]

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

      [ Updates watch           ; Updates watch because back up (CE reset) is
        by 100 ms ]             ; performed

LOOP:
     ;<3>

      [ Processing B ]          ; While performing processing B,

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

      BR      LOOP
```

Figure 11-7 shows the timing chart of the above program.

Figure 11-7. Timing Chart



As shown in Figure 11-7, the program is started from address 0000H at the rising edge of the internal 10-MHz pulse when supply voltage V_{DD} is applied first.

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

Therefore, “processing C” is executed and the timer carry FF setting pulse is set to 100 ms.

Because the contents of the BTM0CY flag are read once at point A, the BTM0CY flag is set to “1” every 100 ms.

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

Because the CE pin goes high at point C, CE reset is effected at point D where the pulse for setting the next timer carry FF rises and the program is started from address 0000H.

Because the BTM0CY flag is set to “1” when it is detected at point E at this time, it is judged that back up (CE reset) has been performed.

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

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

The same applies when the timer carry FF setting pulse is set to 250, 5, or 1 ms.

Therefore, the BTM0CY flag must be detected to detect a power failure within the timer carry FF setting time after the program has started from address 0000H.

(c) If BTM0CY flag is detected at same time as CE reset

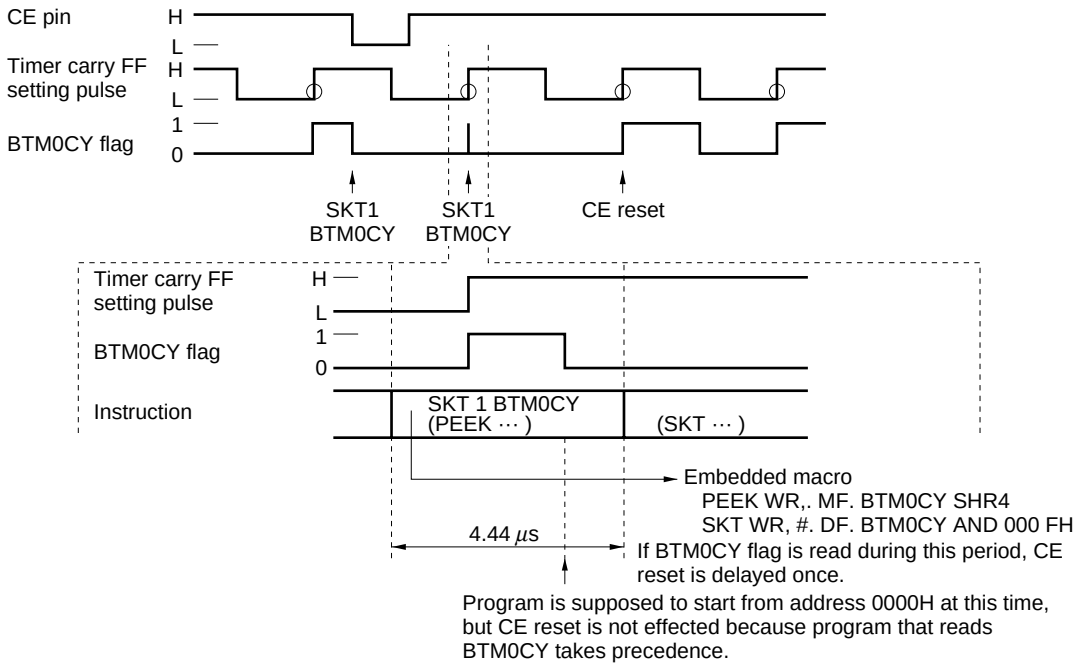
As explained in (b), CE reset is effected if the BTM0CY flag is set to "1".

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

Therefore, if the next setting of the BTM0CY flag (rising of the timer carry FF setting pulse) after the CE pin has gone high is performed at the same time as the instruction that reads the BTM0CY flag, CE reset is effected "when the BTM0CY flag is set next".

This operation is illustrated in Figure 11-8.

Figure 11-8. Operation When CE Reset Is Executed At Same Time As Instruction That Reads BTM0CY Flag



Therefore, a program that periodically detects the BTM0CY flag never executes a CE reset if the detection interval of the BTM0CY flag is the same as the setting time of the TMCY flag.

This must be prevented as follows.

Because one instruction cycle is 4.44 μs (1/225 kHz), a program, for example, that detects the BTM0CY flag once every 255 instructions reads the BTM0CY flag every 1 ms = 4.44 μs × 225.

Once detection and setting of the BTM0CY flag coincide, CE reset is never effected no matter which of 1, 5, 100, or 250 ms of the timer time setting pulse is selected.

Therefore, **do not develop a program that has a cycle satisfying the following condition**.

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

where,

t_{SET} : setting time of BTM0CY flag

X : period x number of steps of instruction that reads BTM0CY flag

An example of a program that satisfies this condition is shown below. Do not develop such a program.

Example

```
            Processing A

      SET2   BTM0CK1, BTM0CK0
                                ; Embedded macro
                                ; Timer carry FF setting pulse is set to 1 ms
LOOP:
    ;<1>
         SKT1    BTM0CY       ; Embedded macro
         BR      BBB
AAA:

           221 steps

         BR      LOOP
BBB:

           221 steps

         BR      LOOP
```

In this example, the instruction <1> that reads the BTM0CY flag is repeatedly executed every 225 instructions. If the BTM0CY flag is set at the timing of the instruction <1>, therefore, CE reset is not effected.

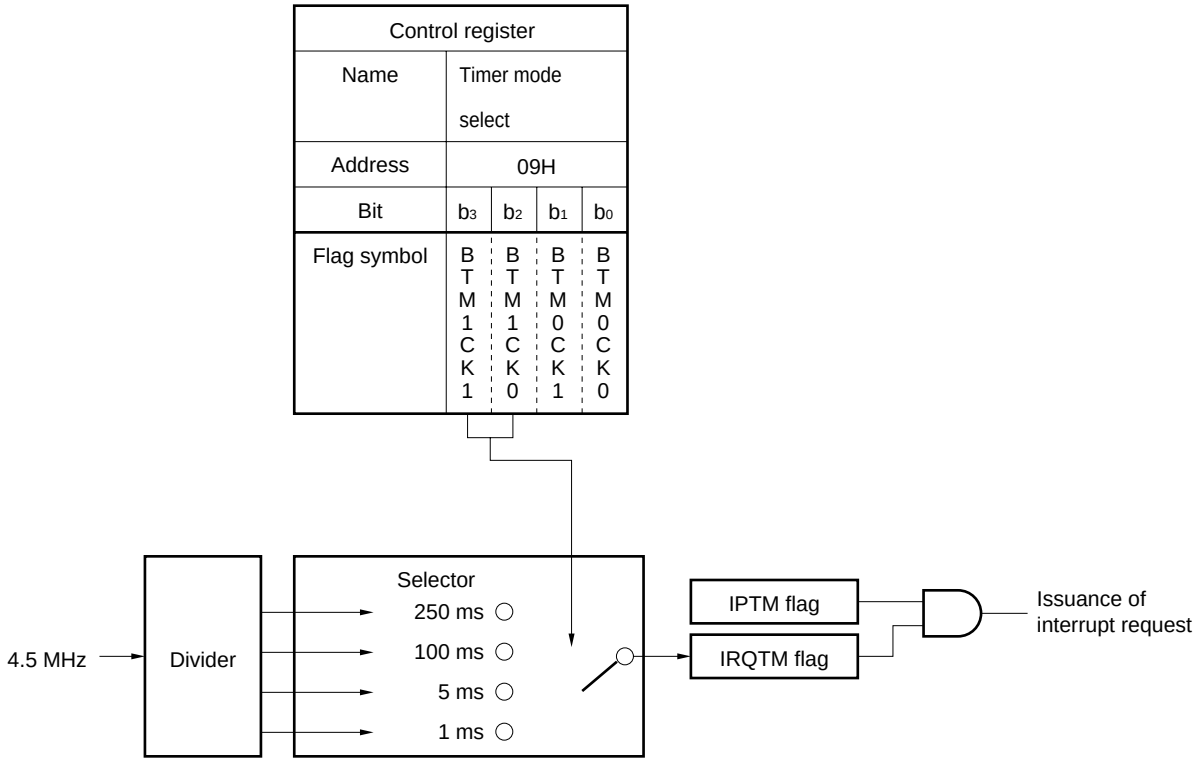
11.4 Timer Interrupt

11.4.1 Configuration of timer interrupt

Figure 11-9 shows the configuration of the timer interrupt.

As shown in the figure, the timer interrupt consists of a divider, selector, and interrupt control flag.

Figure 11-9. Configuration of Timer Interrupt



Remark The interrupt permission flag (IPTM) is set by a software macro. The interrupt request flag (IRQTM) can be reset by software. For details, refer to **10. INTERRUPT**.

11.4.2 Function of timer interrupt

The timer interrupt request is issued at the falling edge of a timer interrupt pulse set by the high-order 2 bits (BTM1CK1 and BTM1CK0 flags) of the timer mode select register (refer to **11.3.3**).

The timer interrupt request corresponds to the IRQTM flag on a one-to-one basis. When the timer interrupt request is issued, the IRQTM flag is set to "1".

In other words, the IRQTM flag is set to "1" if the timer interrupt pulse falls.

So that the timer interrupt is acknowledged, the "EI" instruction that enables all the interrupts must be executed and the timer interrupt must be enabled, in addition to issuance of the interrupt request as explained in **10. Interrupt**.

The timer interrupt is enabled by setting the IPTM flag to "1".

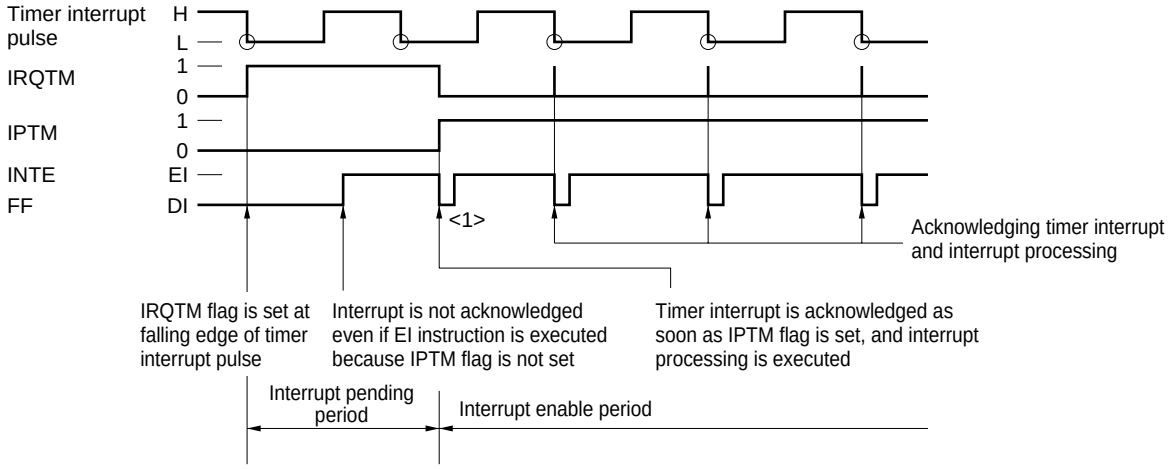
Therefore, the timer interrupt is acknowledged if the IRQTM flag is set to "1" when the "EI" instruction is executed and the IPTM flag is set to "1".

The IPTM flag is set by a software macro. For the macro that enables or disables an interrupt, refer to **Table 10-2. Software Macros Enabling/Disabling Interrupts**. The IRQTM flag cannot be set by software.

When the timer interrupt is acknowledged, the program execution branches to program memory address 0003H. The IRQTM flag is reset to "0" as soon as the interrupt has been acknowledged.

Figure 11-10 shows the relation between the timer interrupt pulse and IRQTM flag.

Figure 11-10. Relation between Timer Interrupt Pulse and IRQTM Flag



A point to be noted is that, as shown in point <1> in the above figure, the timer interrupt is acknowledged when the "EI" instruction is executed next and the IPTM flag is set once the IRQTM flag has been set when the timer interrupt is disabled by the "DI" instruction or IPTM flag.

In this case, the interrupt request can be cleared by executing a software macro that resets the interrupt request (refer to **Table 10-1. Software Macros That Reset Interrupt Requests**).

One level of the stack is used when the timer interrupt is acknowledged.

At this time, the contents of the register bank (BANK: address 79H) and the content of the index enable flag (IXE: bit 0 of address 7FH) are automatically saved.

To return from the interrupt processing routine, use a dedicated instruction "RETI".

For details, refer to **3. ADDRESS STACK (ASK)** and **10. INTERRUPT**.

The following subsections 11.4.3 and 11.4.4 explain an example of using the timer interrupt and an error of the timer interrupt.

For the relation between the timer interrupt and the other interrupt (INT_0 pin), refer to **10. INTERRUPT**.

11.4.3 Example of using timer with timer interrupt

Example

```
M1      MEM     0.10H       ; 80-ms counter
TIMER   DAT     0003H       ; Symbol definition of timer interrupt vector address

        BR      START       ; Branches to START
ORG     TIMER               ; Program address (0003H)
        ADD     M1, #0001B  ; Adds 1 to M1
        SKT1    CY          ; Tests CY flag
        BR      EI_RETI     ; Return if carry does not occur

        Processing A

EI_RETI:
        EI
        RETI
START:
        INITFLG BTM1CK1, NOT BTM1CK0
                            ; Embedded macro
                            ; Sets timer interrupt pulse to 5 ms
        MOV     M1, #0000B  ; Clears contents of M1
        SET1_IPTM           ; Enables timer interrupt (software macro)
        EI                  ; Enables all interrupts
LOOP:

        Processing B

        BR      LOOP
```

The above program executes processing A every 80 ms.

The points to be noted at this time are that the DI status is automatically set when the interrupt is acknowledged, and that the IRQTM flag is set to “1” even in the DI status.

If processing A takes longer than 5 ms, therefore, the interrupt is acknowledged even if the “RETI” instruction is executed.

Consequently, processing B is not executed.

11.4.4 Error of timer interrupt

As explained in 11.4.2, the timer interrupt is acknowledged each time the timer interrupt pulse falls if the EI instruction is executed and the timer interrupt is enabled.

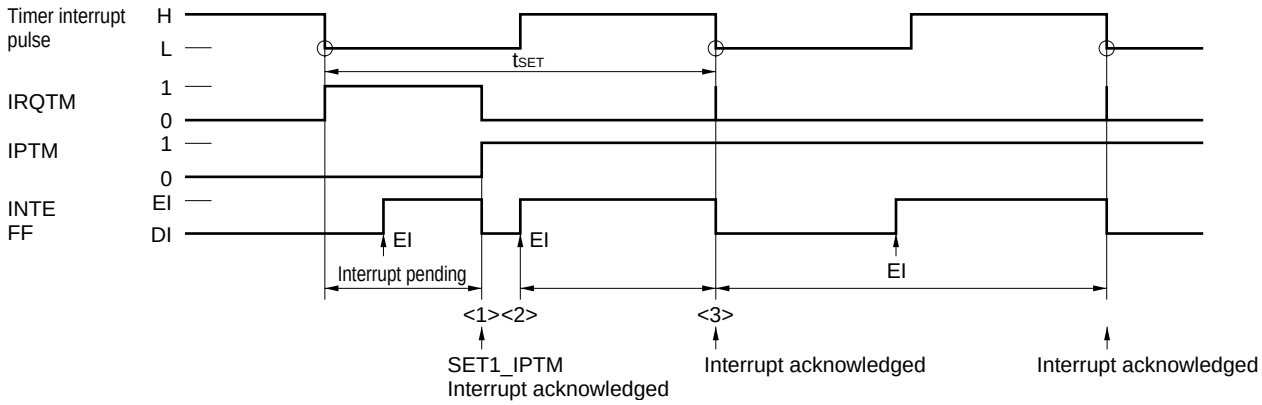
Therefore, an error of the timer when the timer interrupt is used occurs only in the following cases (1) through (3).

(1) When the first interrupt is acknowledged with the timer interrupt enabled
(2) When the first interrupt is acknowledged with the time of the timer interrupt pulse changed
(3) If the software macro that resets the timer interrupt request is executed at the same time as the falling of the timer interrupt pulse (the interrupt is not acknowledged)

Figure 11-11 shows the errors that may occur in each of the above cases.

Figure 11-11. Error of Timer Interrupt (1/2)

(a) When timer interrupt is enabled



At point <1> in the above figure, the timer interrupt is acknowledged immediately if the interrupt is enabled by setting the IPTM flag.

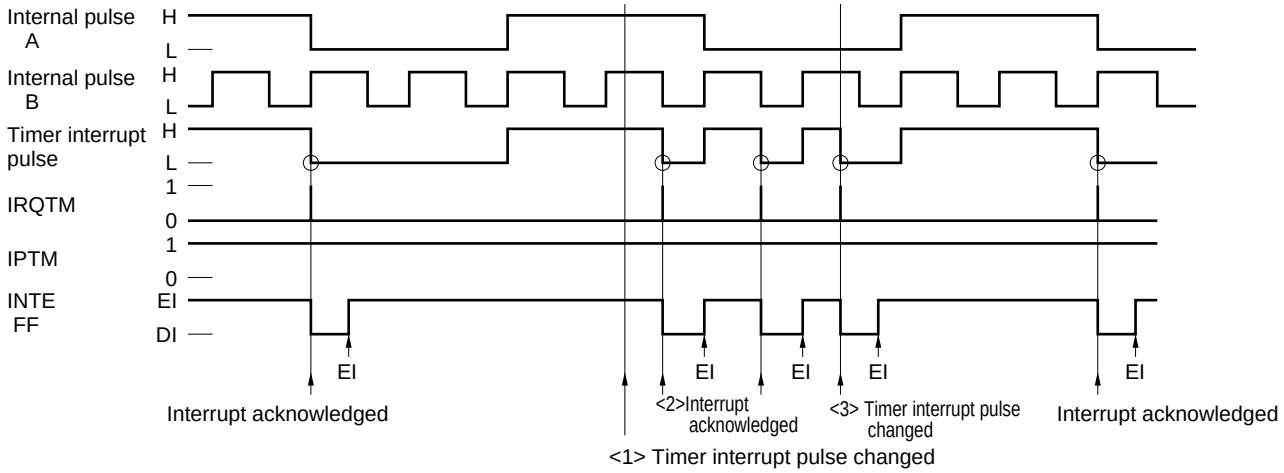
At this time, the error is $-t_{SET}$.

If the interrupt is enabled by the “EI” instruction at point <2>, the interrupt occurs at the falling edge of the timer interrupt pulse at point <3>.

At this time, the error is $-t_{SET} < \text{error} < 0$.

Figure 11-11. Error of Timer Interrupt (2/2)

(b) If timer interrupt pulse is changed



Because the timer interrupt pulse does not fall even if the timer interrupt pulse is changed to B at point <1>, the interrupt is acknowledged at the next point <2>.

If the timer interrupt pulse is changed to A at point <3>, the timer interrupt pulse falls, and the interrupt is immediately acknowledged.

11.4.5 Notes on using timer interrupt

When developing a program in which the timer always operates after the supply voltage V_{DD} has been once applied (power-ON reset) by using the timer interrupt, such as a watch program, the timer interrupt processing must be completed within specific time.

Take the following program for example.

Example

```
M1      MEM     0.10H          ; 1-ms counter
TIMER   DAT     0003H          ; Symbol definition of interrupt vector address

        BR      START          ; Branches to START
ORG     TIMER                  ; Program address (0003H)
        ADD     M1, #0100B     ; Adds 0100B to contents of M1
        SKT1    CY             ; Watch processing if carry occurs
        BR      EI_RETI        ; Return if carry does not occur
      ;<1>

        Watch processing

EI_RETI:
        EI
        RETI
START:
        INITFLG NOT BTM1CK1, BTM1CK0, NOT BTM0CK1, NOT BTM0CK0
                               ; Embedded macro
                               ; Sets timer interrupt time to 250 ms and timer carry FF
                               ; setting time to 100 ms
        SET1_IPTM              ; Enables timer interrupt (software macro)
        EI                     ; Enables all interrupts
LOOP:

        Processing A

        BR      LOOP
```

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

If the CE pin goes high, CE reset is effected in synchronization with the rising of the timer carry FF setting pulse, as shown in Figure 11-12 (a).

If the timer interrupt request (IRQTM flag) is issued at the same time as setting of the timer carry FF, CE reset takes precedence.

When CE reset is effected, the IRQTM flag is automatically reset to “0”.

In other words, timer processing is not executed once if the CE reset is effected while the IRQTM flag is set to “1”.

To prevent this, a delay time is provided between “rising of the timer carry FF setting pulse” and “falling of the timer interrupt pulse” as shown in Figure 11-12(b).

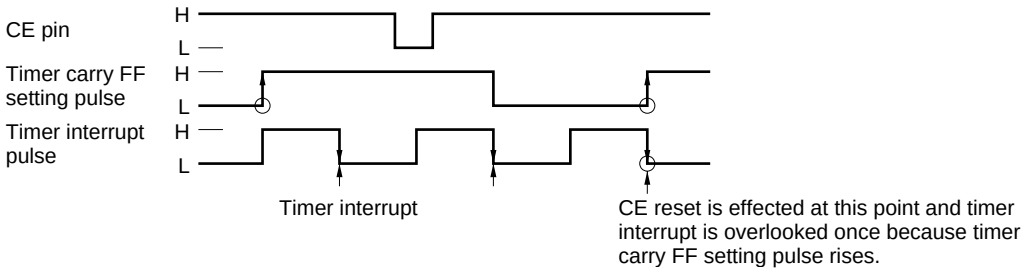
In the above example, the timer interrupt is not overlooked even if the CE reset is effected if the watch processing is executed within 10 ms as shown in Figure 11-12(b).

Because the timer carry FF and timer interrupt time setting pulse can be set to 4 Hz (250 ms), 10 Hz (100 ms), 200 Hz (5 ms), or 1 kHz (1 ms) independently, time differences shown in Figure 11-13 and Table 11-1 are provided.

To validate the timer interrupt at CE reset, therefore, the processing of the timer interrupt must be completed within the delay time of the pulse as shown in Figure 11-13.

Figure 11-12. Timing Chart

(a)



(b)

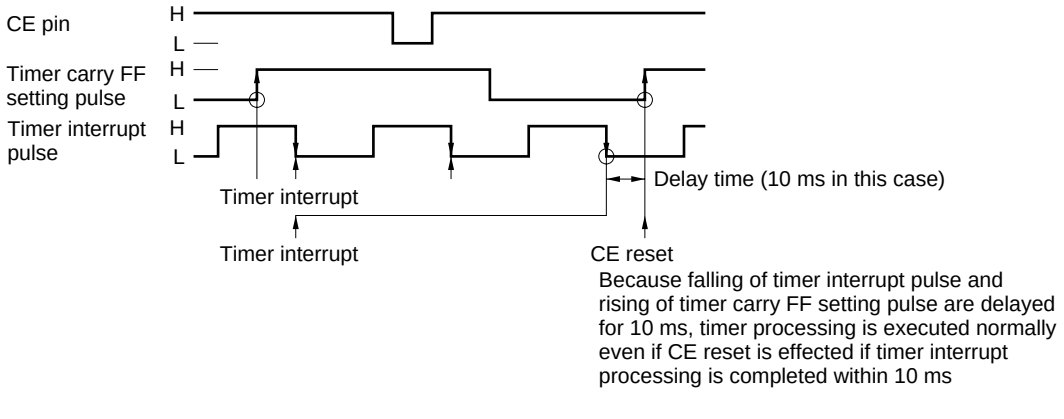


Figure 11-13. Time Difference between Timer Carry FF Setting Pulse and Timer Interrupt Pulse

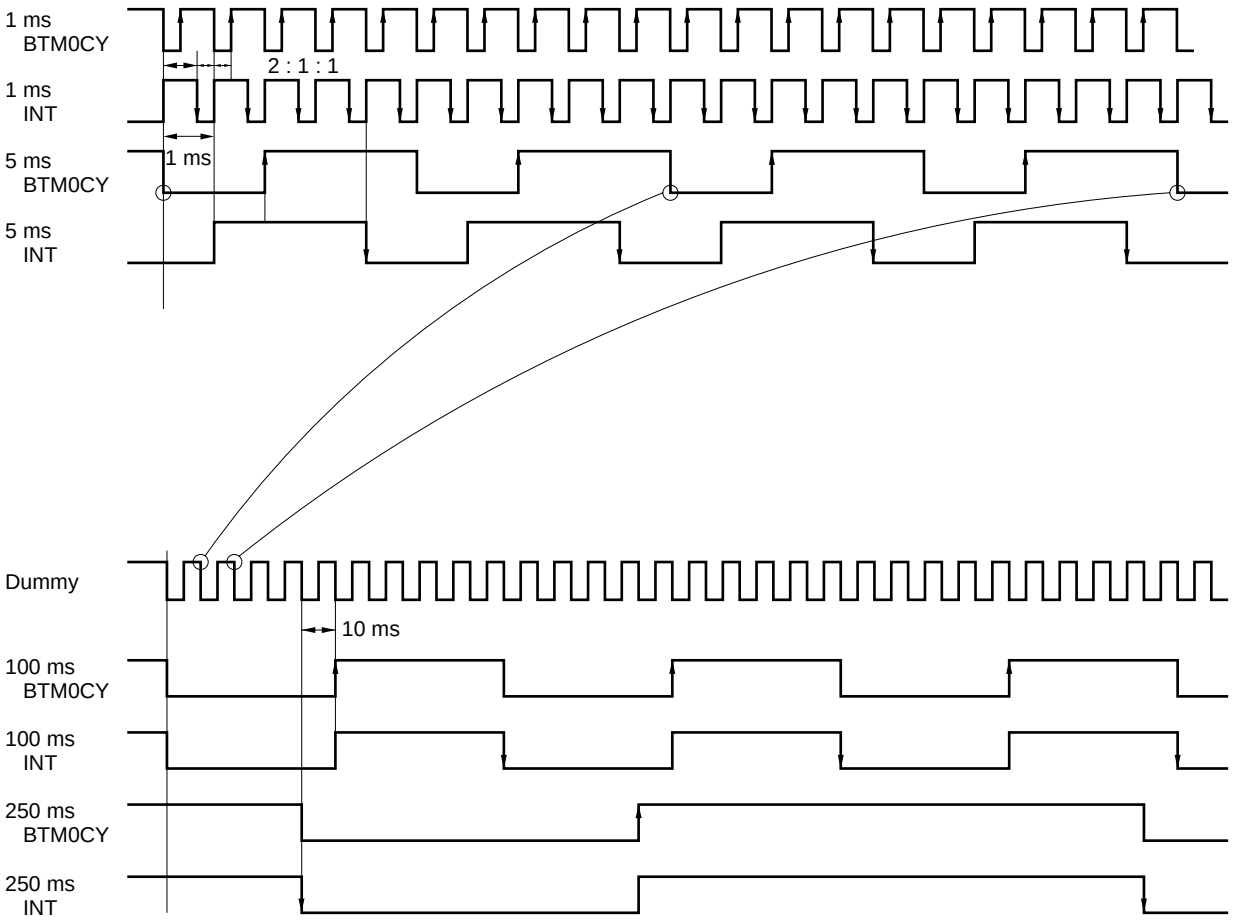
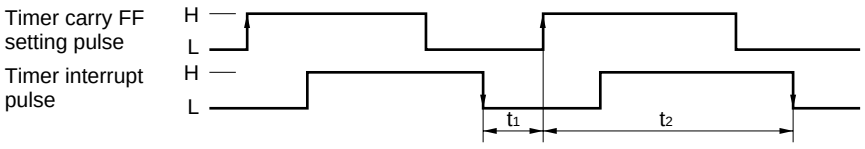


Table 11-1. Time Difference between Rising Edge of Timer Carry FF Pulse and Falling Edge of Timer Interrupt Pulse

Internal Pulse		Minimum Value of Time Difference (Refer to figure below.)	
Timer carry	Timer interrupt	t_1	t_2
1 ms	1 ms	666 μs	333 μs
1 ms	5 ms	333 μs	666 μs
1 ms	100 ms	333 μs	666 μs
1 ms	250 ms	333 μs	666 μs
5 ms	1 ms	333 μs	666 μs
5 ms	5 ms	3 ms	2 ms
5 ms	100 ms	2 ms	3 ms
5 ms	250 ms	2 ms	3 ms
100 ms	1 ms	333 μs	666 μs
100 ms	5 ms	1 ms	4 ms
100 ms	100 ms	50 ms	50 ms
100 ms	250 ms	10 ms	40 ms
250 ms	1 ms	333 μs	666 μs
250 ms	5 ms	1 ms	4 ms
250 ms	100 ms	40 ms	10 ms
250 ms	250 ms	100 ms	150 ms



12. STANDBY

The standby function is used to reduce the current consumption of the device during back up.

12.1 Configuration of Standby Block

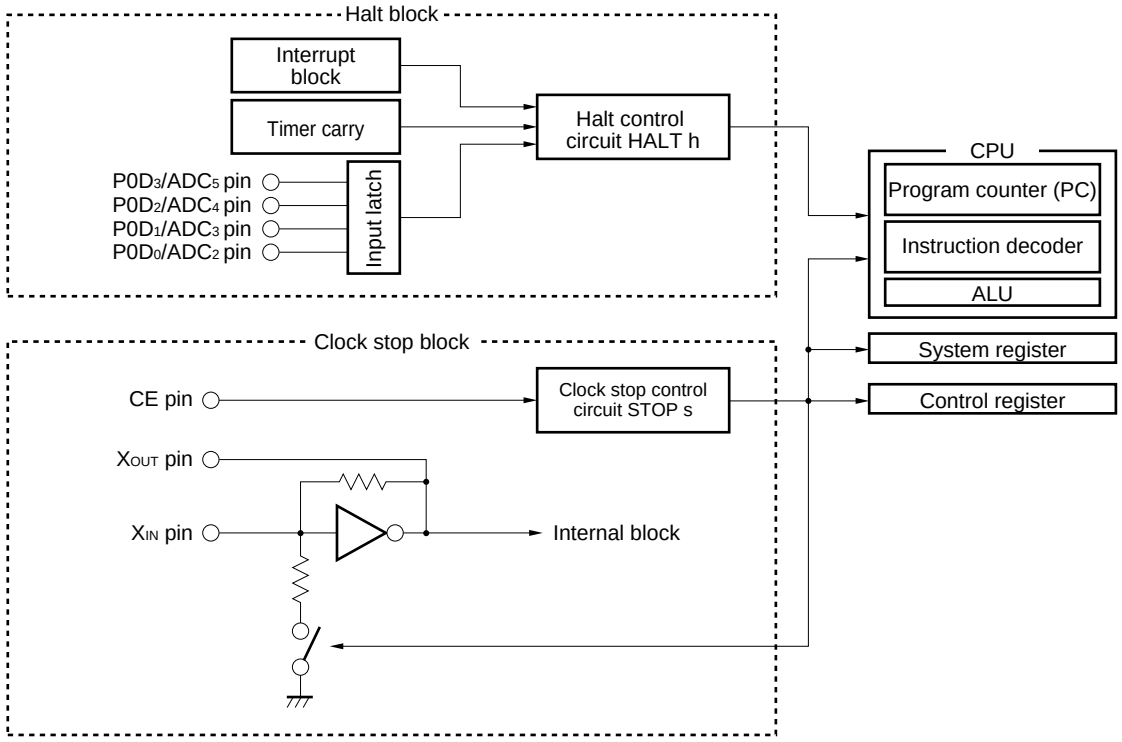
Figure 12-1 shows the configuration of the standby block.

As shown in the figure, the standby block is divided into two blocks: halt control block and clock stop control block.

The halt control block consists of a halt control circuit, interrupt control block, timer carry, and key input pins P0D$_0$/ADC$_2$ through P0D$_3$/ADC$_5$, and controls the operation of the CPU (program counter, instruction decoder, and ALU block).

The clock stop control block controls the 4.5-MHz crystal oscillation circuit, CPU, system register, and control registers, by using the clock stop control circuit.

Figure 12-1. Configuration of Standby Block



12.2 Standby Function

The standby function reduces the current consumption of the device by stopping some or all the operations of the device.

The standby function can be used in two modes: halt and clock stop.

The halt mode is to reduce the current consumption of the device by executing a dedicated instruction “HALT h” and stopping the operation of the CPU.

The clock stop mode is to reduce the current consumption of the device by executing a dedicated instruction “STOP s” and stopping the 4.5-MHz crystal oscillation circuit.

In addition to the halt and clock stop modes, the operation mode of the device can be also set by the CE pin.

The CE pin is used to control the operation of the PLL frequency synthesizer and reset the device, and can be said to be a type of the standby function in that it controls the operation of the PLL frequency synthesizer.

The following section 12.3 explains how to set the operation mode of the device by using the CE pin.

Sections 12.4 and 12.5 respectively explain the halt and clock stop modes.

12.3 Selecting Device Operation Mode with CE Pin

The CE pin controls the following functions (1) through (3) by using the level and rising edge of an externally input signal.

(1) Controls operation of PLL frequency synthesizer
(2) Enables or disables clock stop instruction
(3) Resets device

12.3.1 Controlling operation of PLL frequency synthesizer

The PLL frequency synthesizer can operate only when the CE pin is high.

The PLL frequency synthesizer is automatically disabled when the CE pin is low.

At this time, the VCOH and VCOL pins are internally pulled down, and the EO_0 and EO_1 pins are floated.

The PLL frequency synthesizer can be disabled by program at any time when the CE pin is high.

12.3.2 Enabling and disabling clock stop instruction

The clock stop instruction “STOP s” is enabled only when the CE pin is low.

The “STOP s” instruction is executed as a no-operation (NOP) instruction if it is executed when the CE pin is high.

12.3.3 Resetting device

The device can be reset (CE reset) by raising the CE pin.

The device can also be reset through power application (power-ON reset).

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

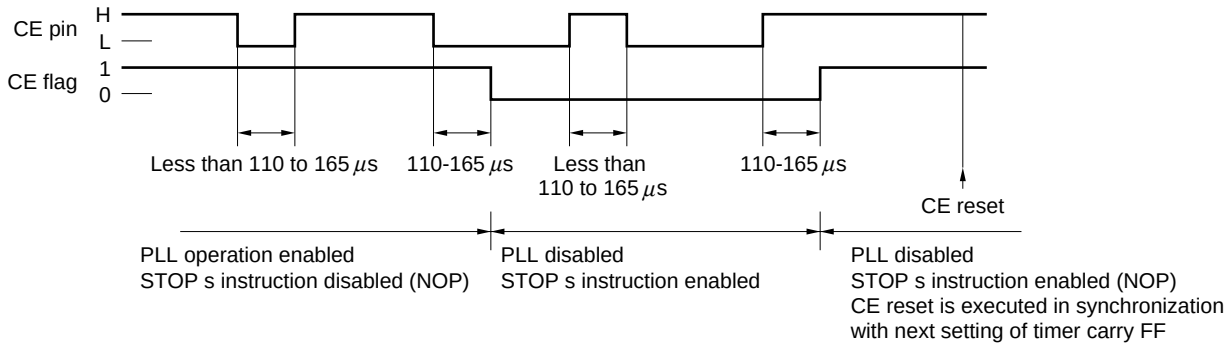
12.3.4 Inputting signal to CE pin

The CE pin does not accept a low or high level of less than 110 to 165 μs to prevent malfunctioning due to noise.

The level of the signal input to the CE pin can be detected by using the CE flag of the CE pin level judge register (RF address 07H).

Figure 12-2 shows the relation between the input signal and CE flag.

Figure 12-2. Relation between Signal Input to CE Pin and CE Flag



12.3.5 Configuration and function of CE pin level judge register

The CE pin level judge register detects the level of the signal input to the CE pin.

The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
CE pin level judge register	0	0	0	CE	07H	R

Detects level input to CE pin	
0	Low level
1	High level

Fixed to “0”

On reset	Power-ON	0	0	0	–
	Clock stop				–
	CE reset				–

– : Determined depending on pin status

The CE flag is not affected by a low or high level of less than 110 to 165 μs.

12.4 Halt Function

The halt function stops the operation clock of the CPU by executing the “HALT h” instruction.

When the “HALT h” instruction is executed, the program stops at the “HALT h” instruction, until the halt status is released later.

Therefore, the current consumption of the device can be reduced in the halt status by the operating current of the CPU.

The halt status can be released by key input, timer carry, or interrupt.

The releasing condition of the key input, timer carry, and interrupt is specified by the operand “h” of the “HALT h” instruction.

The “HALT h” instruction is valid regardless of the input level of the CE pin.

The following subsections 12.4.1 through 12.4.6 explain the halt status, halt release condition, and each halt release condition.

12.4.1 Halt status

All the operations of the CPU are stopped in the halt status.

In other words, program execution is stopped at the “HALT h” instruction.

However, the peripheral hardware units continue the operations set before the “HALT h” instruction is executed.

For the operations of the peripheral hardware units, refer to **12.6 Device Operations in Halt and Clock Stop Status**.

12.4.2 Halt release condition

Figure 12-3 shows the halt release conditions.

As shown in this figure, the halt release conditions are set by 4-bit data specified by operand “h” of the “HALT h” instruction.

The halt status is released when the condition specified as “1” by operand “h” is satisfied.

When the halt status is released, the execution starts from the instruction next to the “HALT h” instruction.

If two or more release conditions are specified, and if any one of the specified conditions is satisfied, the halt condition is released.

If the device is reset (power-ON reset or CE reset), the halt status is released, and each reset operation is performed.

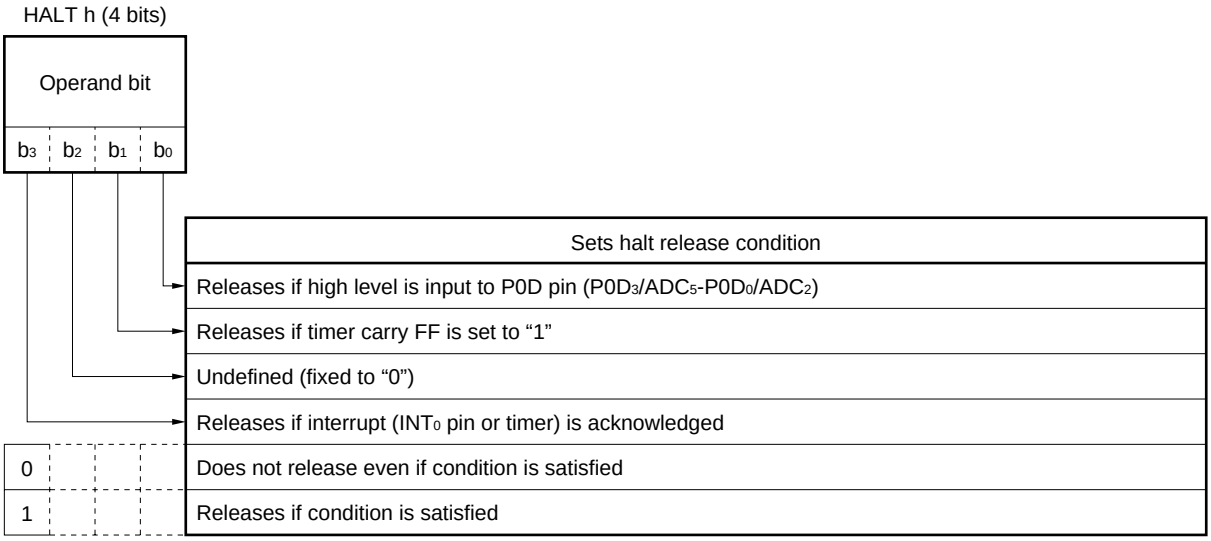
If 0000B is set as the halt release condition “h”, no release condition is set.

At this time, the halt status is released if the device is reset (power-ON reset or CE reset).

The following subsections 12.4.3 through 12.4.5 explains halt release conditions set by key input, timer carry, and interrupt.

12.4.6 shows an example when two or more release conditions are specified.

Figure 12-3. Halt Release Condition



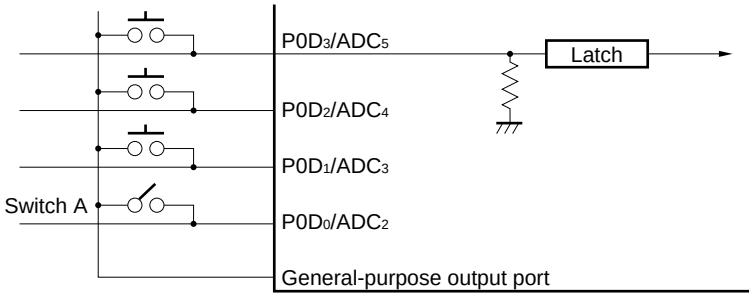
12.4.3 Releasing halt status by key input

Releasing the halt status by key input is specified by the “HALT 0001B” instruction.

If releasing the halt status by key input is specified, the halt status is released when a high level is input to any of the four pins P0D0/ADC2 through P0D3/ADC5.

The following paragraphs (1) through (4) explain the points to be noted when using a general-purpose output port for a key source signal, when multiplexing LCD segment signal output with key source signal output, and when using one of the P0D0/ADC2 through P0D3/ADC5 pins as an A/D converter pin.

(1) Notes on using general-purpose output port for key source signal



The “HALT 0001B” instruction is executed after a general-purpose output port for key source signal goes high.

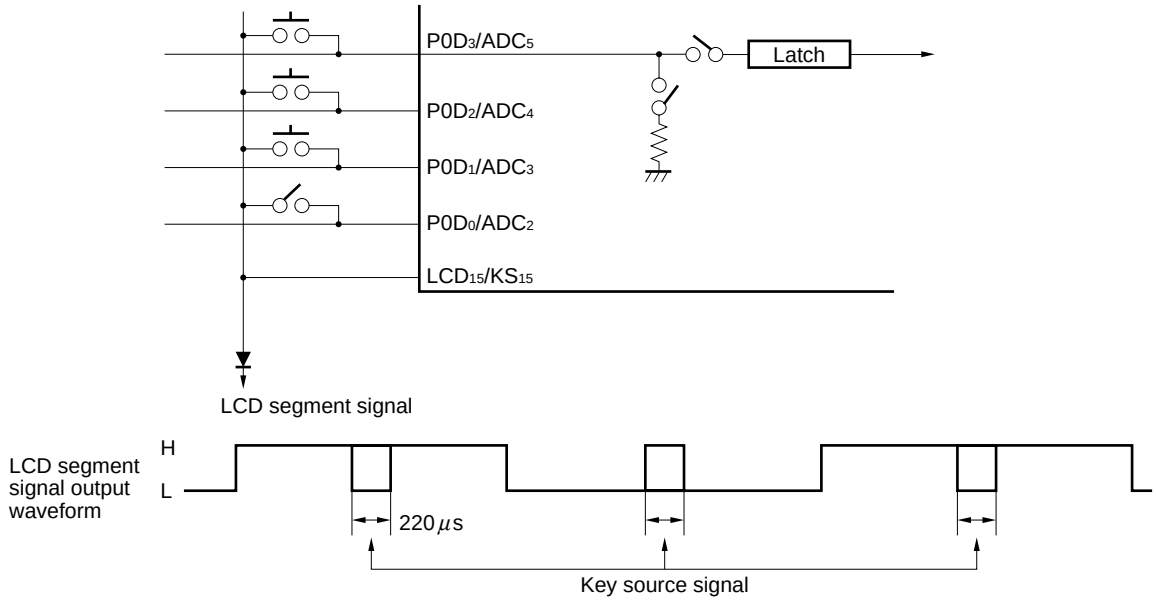
If an alternate switch such as switch A in the above figure is used at this time, a high level is always applied to the P0D0/ADC2 pin while switch A is closed, and the halt status is immediately released.

Therefore, care must be execised in using the alternate switch.

When using a general-purpose output port for key source signal, reset the KSEN flag of the LCD mode select register (RF address 10H) to “0”.

At this time, the P0D0/ADC2 through P0D3/ADC5 pins are automatically pulled down.

(2) Notes on multiplexing LCD segment signal output and key source signal output



Execute the “HALT 0001B” instruction after setting key source signal output data.

At this time, the halt status is not released even if a high level of the LCD segment signal is input to the pin whose key source signal output data is “0”.

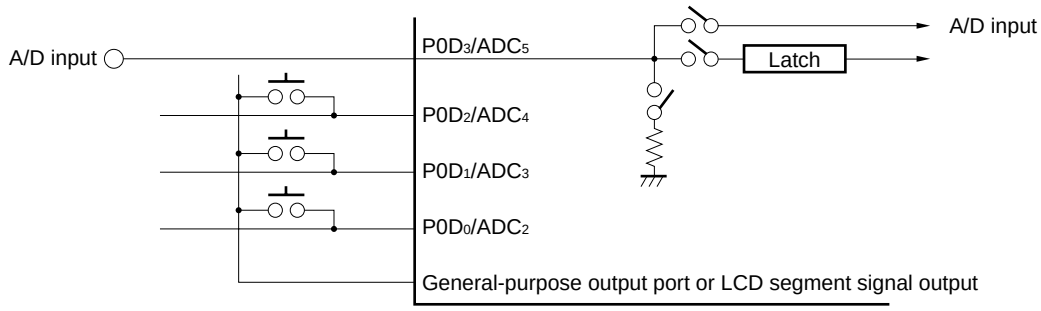
To multiple LCD segment signal output with key source signal output, set the KSEN flag of the LCD mode select register to 1.

The key source signal data (setting the pin that outputs a key source) is set by the key source data register (KSR: peripheral address 42H) via the data buffer.

The internal key latch circuit when LCD segment signal output is multiplexed with key source signal output latches data only while the key source signal is output, and is disconnected from the external source while the LCD segment signal is output.

The internal pull-down resistor is ON only when the key source signal is output.

(3) Notes on using P0D0/ADC2 to P0D3/ADC5 pin as A/D converter pin



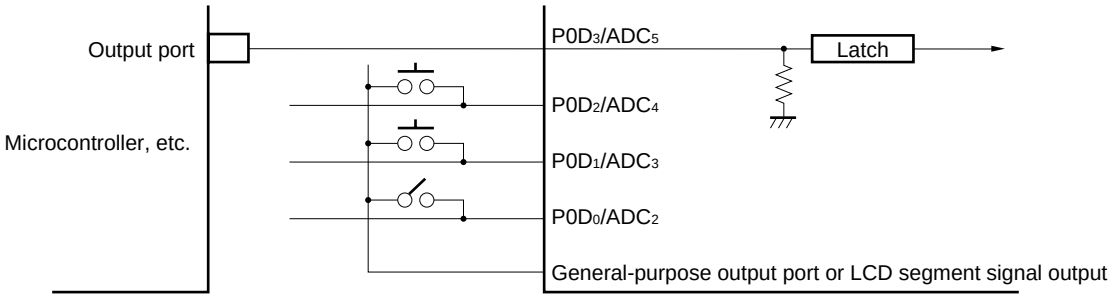
When any of the P0D0/ADC2 through P0D3/ADC5 pins is used as an A/D converter pin, the selected pin (only one pin can be selected at one time) is disconnected from the input latch and is connected to the internal A/D converter input.

If a high level is input to the pin selected as an A/D converter pin at this time, the latch circuit holds the high level.

If the “HALT 0001H” instruction is executed in this status, the halt status is immediately released even if the instruction can be executed because the input latch is at high level.

To prevent this, set the pin in the input port mode before executing the “HALT 0001H” instruction, and set a low level for A/D converter input.

(4) Others



The P0D0/ADC2 through P0D3/ADC5 pins can also be used as general-purpose input port pins with pull-down resistors.

Therefore, the halt status can also be controlled by using an external microcontroller as shown above.

12.4.4 Releasing halt status by timer carry

Releasing the halt status by the timer carry is set by the “HALT 0010B” instruction.

When the release of the halt status is set by the timer carry, the halt status is released as soon as the timer carry FF has been set to “1”.

The timer carry FF corresponds to the BTM0CY flag of the timer carry FF judge register on a one-to-one basis, as explained in **11. TIMER FUNCTION**, and is set to “1” at fixed time intervals (1 ms, 5 ms, 100 ms, or 250 ms).

Therefore, the halt status can be released at fixed time intervals.

Example

```
M1      MEM     0.10H       ; 1-second counter
HLTTMR  DAT     0010B       ; Symbol definition
        CLR2    BTM0CK1, BTM0CK0
                            ; Embedded macro
                            ; Sets timer carry FF setting time to 250 ms
LOOP:
        HALT    HLTTMR      ; Sets release condition by timer carry FF and halt status
        SKT1    BTM0CY      ; Embedded macro
        BR      LOOP        ; Branches to LOOP if BTM0CY flag is not set
        ADD     M1, #0100B  ; Adds 0100B to contents of M1
        SKT1    CY          ; Embedded macro
        BR      LOOP        ; Executes processing A if carry occurs

        Processing A

        BR      LOOP
```

In this example, the halt status is released every 250 ms and processing A is executed every 1 second.

12.4.5 Releasing halt status by interrupt

Releasing the halt status by an interrupt is set by the “HALT 1000B” instruction.

If releasing the halt status by an interrupt is set, the halt status is released as soon as the interrupt has been acknowledged.

Two interrupt sources, INT_0 pin and timer, are available as explained in **10. INTERRUPT**.

Therefore, the interrupt source to be used to release the halt status must be specified by program in advance.

So that the interrupt is acknowledged, all the interrupts must be enabled (by the EI instruction), each interrupt is enabled (by setting the corresponding interrupt permission flag), in addition that the interrupt request must be issued from each interrupt source.

Even if an interrupt request is issued, if that interrupt is not enabled, the interrupt is not acknowledged and the halt status is not released.

When the halt status has been released because the interrupt has been acknowledged, the program flow branches to the vector address of the interrupt.

If the “RETI” instruction is executed after the interrupt processing, the program flow returns to the instruction next to the “HALT” instruction.

Here is an example.

Example

```
HLTINT  DAT     1000B   ; Symbol definition of halt condition
INTTM   DAT     0003H   ; Interrupt vector address symbol definition
INT0PIN DAT     0005H   ;

START:                  ; Program address 0000H
        BR      MAIN
ORG     INTTM           ; Timer interrupt vector address (0003H)
        BR      INTTIMER
ORG     INT0PIN         ; INT0 pin interrupt vector address (0005H)

        Processing A    ; Interrupt processing by INT0 pin

        BR      EI_RETI
INTTIMER:

        Processing B    ; Interrupt processing by timer

EI_RETI:
        EI
        RETI
MAIN:
        SET2_IPTM_IP0   ; Software macro
        SET2    BTM1CK1, BTM1CK0
                        ; Embedded macro
                        ; Sets time interval of timer interrupt to 1 ms
LOOP:

        Processing C    ; Main routine processing

        EI              ; Enables all interrupts
        HALT    HLTINT  ; Specifies releasing halt by interrupt
    ; <1>
        BR      LOOP
```

In this example, the halt status is released when the timer interrupt is acknowledged, and processing B is executed. When the INT_0 pin interrupt is acknowledged, processing A is executed.

Each time the halt status is released, processing C is executed.

If the INT_0 pin interrupt request and timer interrupt request are issued at the same time in the halt status, processing A of the INT_0 pin, which has the higher hardware priority, is executed.

If “RETI” is executed after execution of processing A, execution restores to the “BR LOOP” instruction in <1>, but the “BR LOOP” instruction is not executed, and the timer interrupt is immediately acknowledged.

If “RETI” is executed after processing B of the timer interrupt has been executed, the “BR LOOP” instruction is executed.

Caution When executing the HALT instruction which is to be released if the interrupt request flag (IRQ×××) for which the interrupt permission flag (IP×××) is set is set, describe a NOP instruction immediately before the HALT instruction.

If a NOP instruction is described immediately before the HALT instruction, a time of one instruction is generated in between the IRQ××× manipulation instruction and HALT instruction. In the case of the CLR1 IRQ××× instruction, for example, clearing IRQ××× is correctly reflected on the HALT instruction (refer to Example 1 below). If a NOP instruction is not described immediately before the HALT instruction, the CLR1 IRQ××× instruction is not correctly reflected on the HALT instruction, and the HALT mode is not set (refer to Example 2 below).

Example 1. Program that correctly executes HALT instruction

```
      ⋮
      ⋮                ; Sets IRQ×××
      ⋮

CLR1    IRQ×××
NOP                    ; Describes NOP instruction immediately before
                       ; HALT instruction
                       ; (clearing IRQxxx is correctly reflected on HALT
                       ; instruction)
HALT    1000B          ; Correctly executes HALT instruction
                       ; (HALT mode is set)
      ⋮
      ⋮
```

2. Program that does not set HALT mode

```
      ⋮
      ⋮                ; Sets IQR×××
      ⋮

CLR1    IRQ×××         ; Clearing IRQxxx is not reflected on HALT instruction
                       ; (but on instruction next to HALT)
HALT    1000B          ; HALT instruction is ignored (HALT mode is not set)
      ⋮
      ⋮
```

12.4.6 If two or more release conditions are simultaneously set

If two or more release conditions are simultaneously set, and if even one of the conditions is satisfied, the halt status is released.

The method to identify the release condition that is satisfied when two or more release conditions are specified is shown below.

Example 1.

```
        HLTINT   DAT  1000B
        HLTTMR   DAT  0010B
        HLTKEY   DAT  0001B
        INT0PIN  DAT  0005H        ; INT0 pin interrupt vector address symbol
                                   ; definition
START:
        BR    MAIN
ORG     INT0PIN
        Processing A               ; INT0 pin interrupt processing
        EI
        RETI
TMRUP                              ; Timer carry processing
        Processing B
        RET
KEYDEC:                            ; Key input processing
        Processing C
        RET
MAIN:
        MOVT   DBF, @AR            ; Sets key source output data (table reference)
                                   ; to key source data register (KSR)
        PUT    KSR, DBF
        SET2   KSEN, LCDEN         ; Embedded macro
                                   ; Multiplexes LCD segment signal output with
                                   ; key source signal output
        SET2   BTM0CK1, BTM0CK0    ; Embedded macro
                                   ; Sets timer carry FF setting time to 1 ms
        SET1_IP0                   ; Software macro
                                   ; Enables INT0 pin interrupt
        EI
LOOP:
        HALT   HLTINT OR HLTTMR OR HLTKEY
                                   ; Specifies interrupt, timer carry, and key input
                                   ; as halt release conditions
        SKF1   BTM0CY              ; Embedded macro
                                   ; Detects TMCY flag
        CALL   TMRUP               ; Timer carry processing if set to “1”
        SKF1   KEYJ                ; Embedded macro
                                   ; Detects key input latch
        CALL   KEYDEC              ; Key input processing if latched
        BR     LOOP
```

In example 1 above, three halt release conditions are specified: INT_0 pin interrupt, 1-ms timer carry, and key input.

To detect which condition has caused the halt status to be released, the vector address, BTM0CY flag, and KEYJ flag are detected to identify the interrupt, timer carry, and key input, respectively.

When using two or more release conditions, the following two points must be noted.

(1) All the specified release conditions must be detected when the halt status has been released.
(2) The conditions must be sequentially detected starting from the one with the highest priority.

For example, if the program below "MAIN:" in example 1 above is as shown in example 2 below, care must be exercised. Do not develop the program as shown in example 2 if the timer carry has a high priority.

Example 2.

```
MAIN:
     SET4  P1C3, P1C2, P1C1, P1C0  ; Uses general-purpose output port as key
     SET2  BTM0CK1, BTM0CK0        ; source signal
     ;
     SET1_IP0
     EI
 LOOP:
     HALT  HLTINT OR HLTTMR OR HLTKEY
     SKF4  P0D3, P0D2, P0D1, P0D0  ; Detects key input
     BR    KEYDEC
     SKF1  BTM0CY
     CALL  TMRUP
     BR    LOOP
 KEYDEC:                           ; Key input processing
     Processing C
     BR    LOOP
```

In example 2 above, suppose the timer carry FF is set to "1" immediately after the halt status has been released by key input.

Then the program executes the "HALT" instruction again after executing the key input processing.

Because the timer carry FF remains set at this time, the halt status is immediately released.

Usually, however, a high level is input for about 100 ms as key input. Consequently, execution further branches to the key input processing.

As a result, the timer carry FF is not correctly detected.

12.5 Clock Stop Function

The clock stop function stops the 4.5-MHz crystal oscillation circuit by executing the “STOP s” instruction (clock stop status).

Therefore, the current consumption of the device is decreased to the minimum value of 15 μA.

For the details on the current consumption, refer to **12.7 Current Consumption in Halt Status and Clock Stop Status**.

Specify “0000B” as operand “s” of the “STOP s” instruction.

The “STOP s” instruction is valid only while the CE pin is low.

It is executed as a no-operation (NOP) instruction even when executed while the CE pin is high.

In other words, the “STOP s” instruction must be executed while the CE pin is low.

The clock stop status is released by raising the level of the CE pin from low to high (CE reset).

The following subsections 12.5.1 through 12.5.3 explain the clock stop status, how to release the clock stop status, and notes on using the clock stop instruction.

12.5.1 Clock stop status

Because the crystal oscillation circuit is stopped in the clock stop status, all the device operations, such as those of the CPU and peripheral hardware, are stopped.

For the operations of the CPU and peripheral hardware, refer to **12.6 Device Operations in Halt and Clock Stop Status**.

The power failure detection circuit does not operate in the clock stop status even if the supply voltage V_{DD} of the device is lowered to 2.2 V. Therefore, the data memory can be backed up at a low voltage. For the details on the power failure detection circuit, refer to **13. RESET**.

12.5.2 Releasing clock stop status

The clock stop status is released either by raising the level of the CE pin from low to high (CE reset), or by lowering the supply voltage V_{DD} of the device to 2.2 V or less once, and then increasing it to 4.5 V (power-ON reset).

Figures 12-4 and 12-5 respectively show how the clock stop is released on CE reset and power-ON reset.

If the clock stop status is released by power-ON reset, the power failure detection circuit operates.

For the details on power-ON reset, refer to **13.4 Power-ON Reset**.

Figure 12-4. Releasing Clock Stop Status by CE Reset

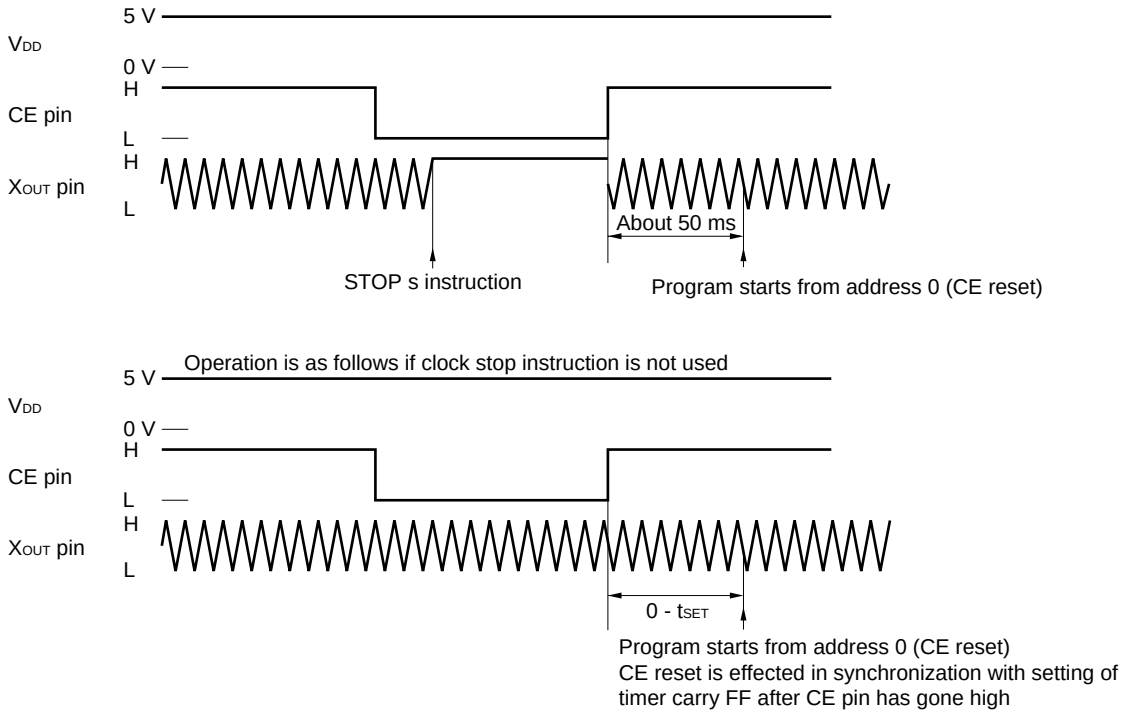
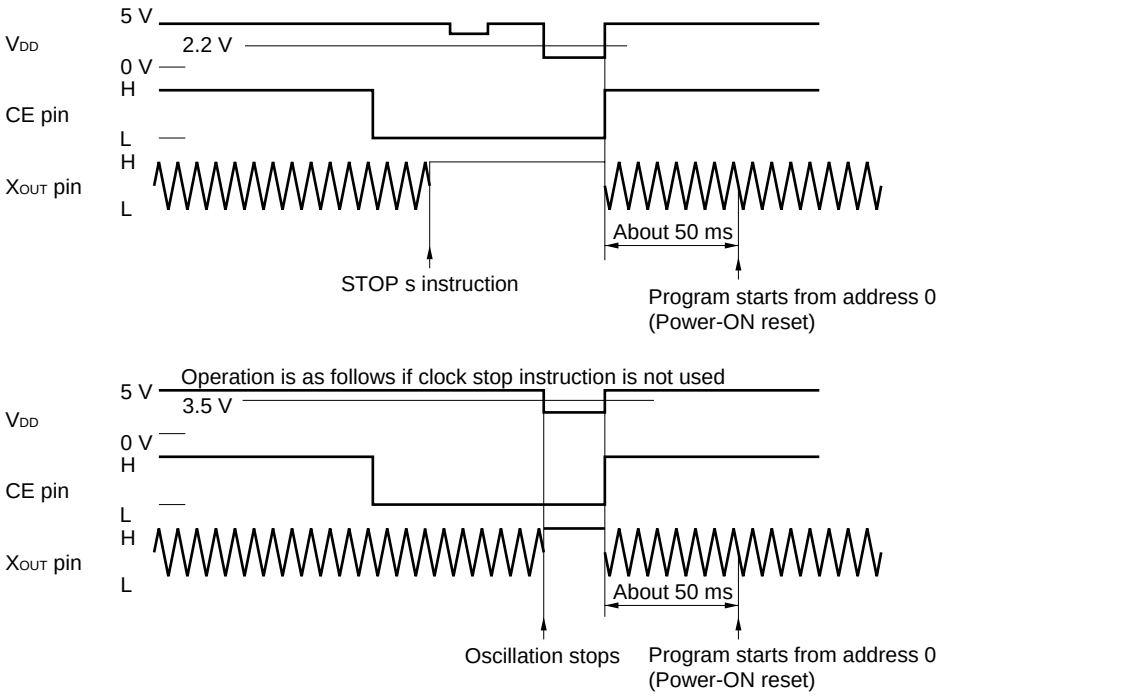


Figure 12-5. Releasing Clock Stop Status by Power-ON reset



12.5.3 Notes on using clock stop instruction

The clock stop (STOP s) instruction is valid only while the CE pin is low.

Therefore, processing to be performed if the CE pin happens to be high must be taken into consideration.

Take the following program as an example.

Example

```
        XTAL    DAT   0000B   ; Symbol definition of clock stop condition
CEJDG:
 ; <1>
        SKF1    CE            ; Embedded macro
                              ; Detects input level of CE pin
        BR      MAIN          ; Branches to main processing if CE = high
        Processing A          ; Processing of CE = low
 ; <2>
        STOP    XTAL          ; Clock stop
 ; <3>
        BR      $ – 1
MAIN:
        Main processing
        BR      CEJDG
```

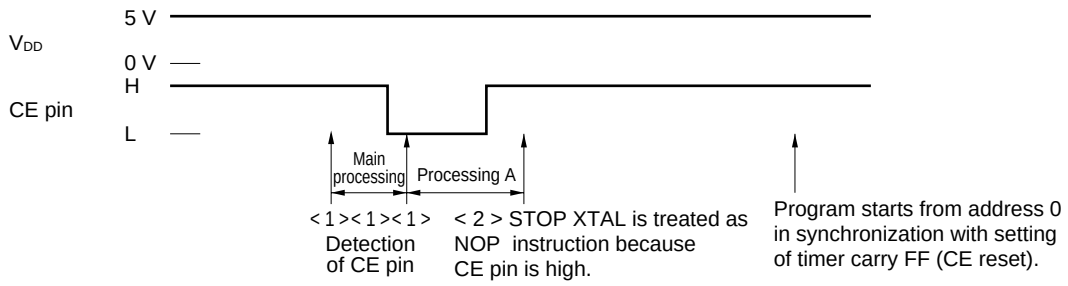
In the above example, the status of the CE pin is detected in <1>. If the CE pin is low, processing A is performed and then the clock stop instruction “STOP XTAL” in <2> is executed.

If the CE pin goes high while the “STOP XTAL” instruction in <2> is executed, however, the “STOP XTAL” instruction is treated as a no-operation (NOP) instruction.

Should branch instruction “BR$-1” in <3> be missing at this time, the program would execute the main processing, causing malfunctioning.

Therefore, either a branch instruction must be inserted as in <3>, or the program must be designed in the manner that malfunctioning does not occur even if the main processing is executed.

If a branch instruction is used as in <3>, CE reset is executed in synchronization with the next setting of the timer carry FF even while the CE pin is high.



12.6 Device Operations in Halt and Clock Stop Status

Table 12-1 shows the operations of the CPU and peripheral hardware in the halt status and clock stop status.

As shown in this table, all the peripheral hardware units continue the normal operation in the halt status, except that instruction execution is stopped.

All the peripheral hardware units stop operation in the clock stop status.

The control registers that control the operations of the peripheral hardware units operate normally in the halt status (i.e., are not initialized), but are initialized to specific values in the clock stop status (as soon as the STOP s instruction has been executed).

To put in another way, the peripheral hardware units continue the operations set by the control registers in the halt status, and operate in accordance with the control registers that are initialized to specific values in the clock stop status.

For the values to which the control registers are initialized, refer to **8. REGISTER FILE (RF)**.

Table 12-1. Device Operations in Halt Status and Clock Stop Status

Peripheral Hardware	Status			
	CE Pin = High		CE Pin = Low	
	Halt	Clock Stop	Halt	Clock Stop
Program counter	Stops at address of HALT instruction	STOP instruction is invalid (NOP)	Stops at address of HALT instruction	Initialized to 0000H and stops
System register	Retained		Retained	Initialized[Note]
Peripheral register	Retained		Retained	Retained
Control register	Retained		Retained	Initialized[Note]
Timer	Normal operation		Normal operation	Stops operation
PLL frequency synthesizer	Normal operation		Disabled	Disabled
A/D converter	Normal operation		Normal operation	Stops operation
D/A converter	Normal operation		Normal operation	Stops operation
BEEP output	Normal operation		Normal operation	Stops operation
Serial interface	Normal operation		Normal operation	Stops operation
Frequency counter	Normal operation		Normal operation	Stops operation
LCD controller/driver	Normal operation		Normal operation	Stops operation
Key source controller/ decoder	Normal operation		Normal operation	Stops operation
General-purpose I/O port	Normal operation		Normal operation	Input port
General-purpose input port	Normal operation		Normal operation	Input port
General-purpose output port	Normal operation		Normal operation	Retained

Note For the value to which these registers are initialized, refer to **5. SYSTEM REGISTER (SYSREG)** and **8. REGISTER FILE (RF)**.

12.7 Current Consumption in Halt Status and Clock Stop Status

12.7.1 Device current consumption in halt status

Figure 12-6 shows the device current consumption I_{DD} in the halt status.

Current consumption curves <1> through <4> in this figure were drawn by using the programs shown below.

As shown in Figure 12-6, the less the halt status is released, the lower the current consumption.

(1) Program 1

The HALT instruction is not used.

Example

```
NOP
BR   $ – 1
```

(2) Program 2

A 5-ms timer interrupt is specified as the halt release condition, and 20 instructions (about 90 μs) are executed each time the halt status has been released.

Example

```
HLTINT   DAT   1000B
INTTM    DAT   0003H

         BR    LOOP
ORG      TMINT
REPT     17
         NOP
ENDR
         EI
         RETI
LOOP:
         INITFLG BTM1CK1, NOT BTM1CK0
         ;
         SET_IPTM
         EI
         HALT  HLTINT
         BR    $ – 1
```

(3) Program 3

A 100-ms timer interrupt is specified as the halt release condition, and 20 instructions are executed each time the halt status has been released.

Example

```
HLTINT   DAT   1000B
INTTM    DAT   0003H

         BR    LOOP
ORG      TMINT
REPT     17
         NOP
ENDR
         EI
         RETI
LOOP:
         CLR2  BTM1CK1, BTM1CK0
         ;
         SET1_IPTM
         EI
         HALT  HLTINT
         BR    $ – 1
```

(4) Program 4

Nothing is set as the halt release condition.

Example

```
HLTNORLS DAT  0000B
HALT HLTNORLS
```

The device current consumption I_{DD} shown in Figure 12-6 were measured under the following conditions.

- PLL is disabled.
- The frequency counter is disabled.
- A sine wave with a frequency f_{IN} = 4.5 MHz and input amplitude $V_{IN} = V_{DD}$ is input to the X_{IN} pin from a standard signal generator.
- All the pins set in the output mode are open.
- All the pins set in the input mode are pulled down by a 47-kΩ resistor (except the X_{IN} pin).

12.7.2 Device current consumption in clock stop status

Figure 12-7 shows the device current consumption I_{DD} in the clock stop status.

The current consumption shown in this figure were measured under the following conditions.

- All the pins set in the output mode are open.
- All the pins set in the input mode are pulled down by a 47-kΩ resistor (except the X_{IN} pin).
- A crystal resonator is connected (however, oscillation is stopped).

Figure 12-6. Device Current Consumption in Halt Status (Reference Value)

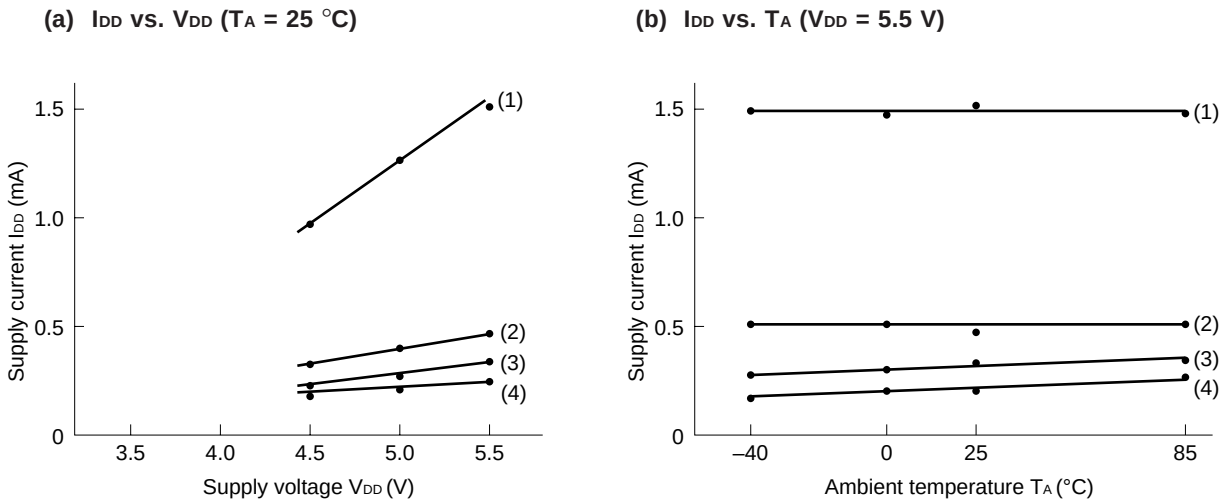
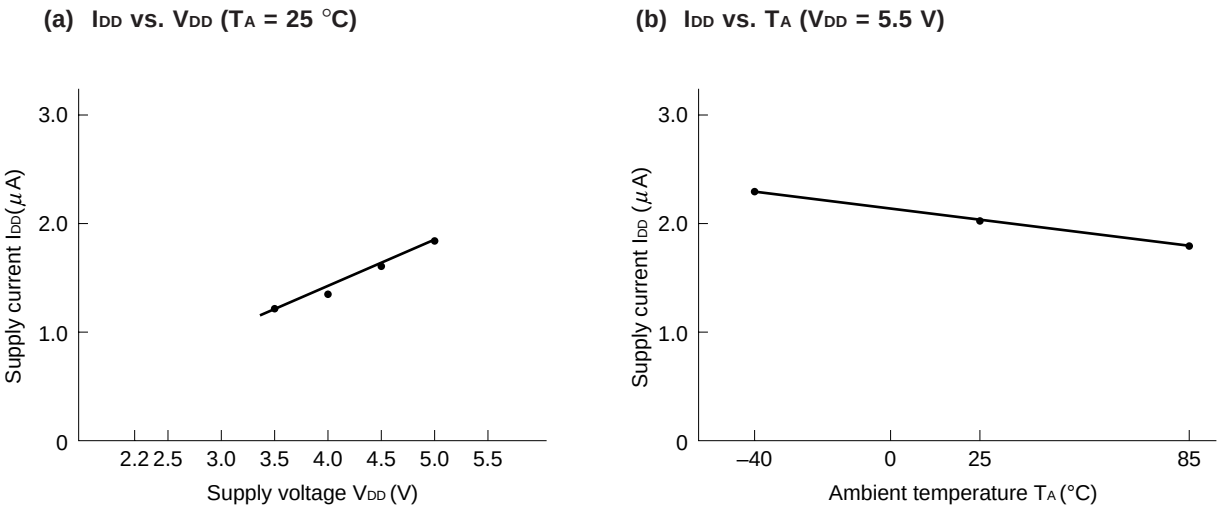


Figure 12-7. Device Current Consumption in Clock Stop Status (Reference Value)



12.7.3 Notes on processing each pin in halt and clock stop status

The halt status is used to reduce the current consumption, for example, when only the watch operates.

The clock stop function is used to reduce the current consumption when only the contents of the data memory are to be retained.

Therefore, the current consumption must be reduced as much as possible in the halt and clock stop statuses.

At this time, the current consumption may increase depending on the status of each pin, and therefore the points shown in Table 12-2 must be noted.

Table 12-2. Notes on Status of Each Pin in Halt and Clock Stop Statuses (1/2)

Pin Function		Pin Symbol	Status of Each Pin and Notes on Processing	
			Halt	Clock stop
General-purpose port	Port 0A	$P0A_3$ $P0A_2$ $P0A_1$ $P0A_0$	Previous status before halt status is set is retained as is. (1) In output mode Current consumption increases if these pins are externally pulled down while they output high level, or externally pulled up while they output low level. Pay attention to N-ch open-drain output pins ($P0A_3$, $P0A_2$, and $P1B_3$-$P1B_0$). (2) In input mode (except ports 1A and 1D) Current consumption increases due to noise if these pins are floated. (3) Port 0D ($P0D_3$/ADC_5-$P0D_0$/ADC_2) Current consumption increases if these pins are externally pulled up because they have pull-down resistor. However, pull-down resistor of pin selected as A/D converter pin is OFF. (4) Ports 1D ($P1D_3$/FMIFC-$P1D_0$/ADC_0) and 1A ($P1A_3$-$P1A_0$) Current consumption increases when $P1D_3$/FMIFC and $P1D_2$/AMIFC pins are used for IF counter because internal amplifier operates. Because IF counter is not automatically disabled even if CE pin goes low, it must be initialized by program as necessary. Ports 1D and 1A are designed to prevent increase in current consumption due to noise even if they are set in general purpose input port mode and floated.	All these pins are set in general-purpose input port mode. All input ports, except port 0C ($P0C_3$-$P0C_0$), are designed to prevent increase in current consumption due to noise even if they are externally floated. Port 0C ($P0C_3$-$P0C_0$) must be externally pulled down or up so that current consumption does not increase due to noise. Port 0D ($P0D_3$/ADC_5-$P0D_0$/ADC_2) is internally pulled down.
	Port 0B	$P0B_3$ $P0B_2$/$\overline{SCK}$ $P0B_1$/SO $P0B_0$/SI		
	Port 0C	$P0C_3$ $P0C_2$ $P0C_1$ $P0C_0$		
	Port 1A	$P1A_3$ $P1A_2$ $P1A_1$ $P1A_0$		
General-purpose input port	Port 0D	$P0D_3$/ADC_5 $P0D_2$/ADC_4 $P0D_1$/ADC_3 $P0D_0$/ADC_2		
	Port 1D	$P1D_3$/FMIFC $P1D_2$/AMIFC $P1D_1$/ADC_1 $P1D_0$/ADC_0		
General-purpose output port	Port 1B	$P1B_3$ $P1B_2$/PWM_1 $P1B_1$/PWM_0 $P1B_0$/BEEP		These pins are set in general-purpose output port pins. Output contents are retained as is. Therefore, current consumption increases if these pins are externally pulled down while they output high level, or pulled up while they output low level.
	Port 1C	$P1C_3$ $P1C_2$ $P1C_1$ $P1C_0$		
	Port 2A	$P2A_0$		
Interrupt		INT_0	Current consumption increase due to external noise if this pin is floated.	

Table 12-2. Notes on Status of Each Pin in Halt and Clock Stop Statuses (2/2)

Pin Function	Pin Symbol	Status of Each Pin and Notes on Processing	
		Halt	Clock stop
LCD segment	$LCD_{29}/P0F_3$ | $LCD_{26}/P0F_0$ $LCD_{25}/P0E_3$ | $LCD_{22}/P0E_0$ LCD_{21} | LCD_{16} LCD_{15}/KS_{15} | LCD_0/KS_0	Same as above general-purpose output ports applies if these pins are used in general-purpose output port mode. If they output key source signals, current consumption increases via port 0D (with pull-down resistor) if there is switch that is always ON such as transistor switch and if "1" is output as key source data.	All pins are set in LCD segment signal output mode and output low level (display off).
PLL frequency synthesizer	VCOL VCOH EO_0 EO_1	Current consumption increases during PLL operation. These pins are as follows when PLL is disabled. VCOL and VCOH : internally pulled down EO_0 and EO_1 : floated PLL is automatically disabled when CE pin goes low.	PLL is disabled. These pins are as follows. VCOL and VCOH : internally pulled down EO_0 and EO_1 : floated
Crystal oscillation circuit	X_{IN} X_{OUT}	Current consumption changes due to oscillation waveform of crystal oscillation circuit. Current consumption decreases as oscillation amplitude increases. Because oscillation amplitude is influenced by crystal resonator and load capacitor used, evaluation must be performed.	X_{IN} pin is internally pulled down, and X_{OUT} pin outputs high level.

13. RESET

The reset function is used to initialize the device operation.

13.1 Configuration of Reset Block

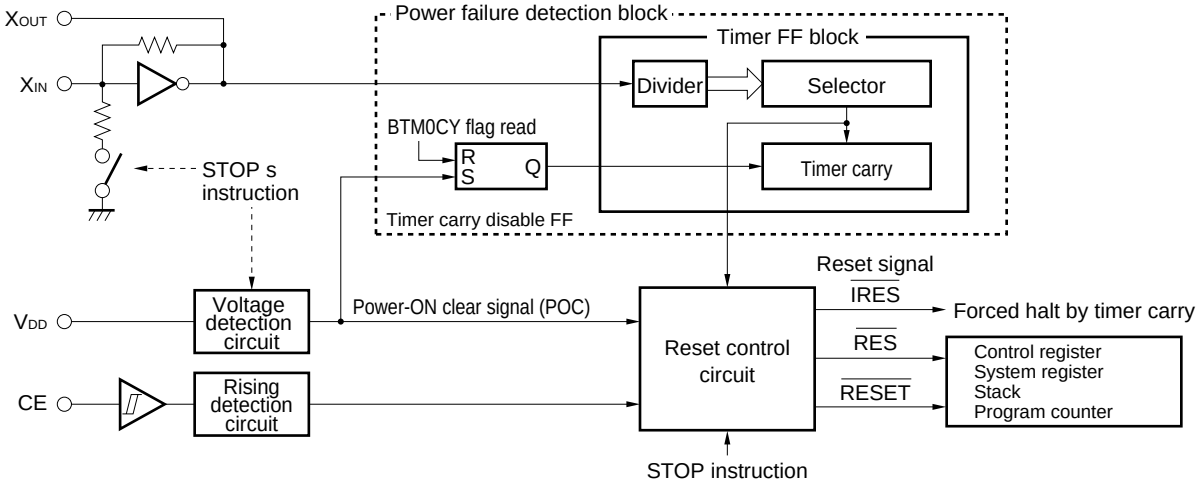
Figure 13-1 shows the configuration of the reset block.

The device is reset in two ways: by applying supply voltage V_{DD} (power-ON reset or V_{DD} reset) and by using the CE pin (CE reset).

The power-ON reset block consists of a voltage detection circuit that detects a voltage input to the V_{DD} pin, a power failure detection circuit, and a reset control circuit.

The CE reset block consists of a circuit that detects the rising of a signal input to the CE pin, and a reset control circuit.

Figure 13-1. Configuration of Reset Block



13.2 Reset Function

Power-ON reset is effected when supply voltage V_{DD} rises from a specific level, and CE reset is effected when the CE pin goes high.

Power-ON reset initializes the program counter, stack, system register, and control registers, and executes the program from address 0000H.

CE reset initializes the program counter, stack, system register, and some control registers, and executes the program from address 0000H.

The major differences between power-ON reset and CE reset are the control registers that are initialized and the operation of the power failure detection circuit that is explained in 13.6.

Both power-ON reset and CE reset are controlled by the reset signals $\overline{IRES}$, $\overline{RES}$, and $\overline{RESET}$ output from the reset control circuit shown in Figure 13-1.

Table 13-1 shows the relation among the $\overline{IRES}$, $\overline{RES}$, and $\overline{RESET}$ signals, and power-ON reset, and CE reset.

The reset control circuit also operates when the clock stop instruction (STOP s) explained in **12. STANDBY** is executed.

The following sections 13.3 and 13.4 respectively explain CE reset and power-ON reset.

Section 13.5 explains the relation between CE reset and power-ON reset.

Table 13-1. Relation between Internal Reset Signals and Each Reset Operation

Internal Reset Signal	Output Signal			Control Operation by Each Reset Signal
	CE Reset	Power-ON Reset	Clock Stop	
$\overline{IRES}$	×	○	○	Forcibly sets device in halt status. Halt status is released when timer carry FF is set.
$\overline{RES}$	×	○	○	Initializes some control registers.
$\overline{RESET}$	○	○	○	Initializes program counter, stack, system register, and some control registers.

13.3 CE Reset

CE reset is effected when the CE pin goes high.

When the CE pin goes high, the $\overline{\text{RESET}}$ signal is output in synchronization with the rising edge of the next timer carry FF setting pulse, and the device is reset.

When CE reset is effected, the $\overline{\text{RESET}}$ signal initializes the program counter, stack, system register, and some control registers, and the program is executed starting from address 0000H.

For the value to which each of the above registers is initialized, refer to the description of each register.

The operation of CE reset differs depending on whether the clock stop instruction is used.

The differences in operation are explained in the following subsections 13.3.1 and 13.3.2.

Subsection 13.3.3 explains the points to be noted on using CE reset.

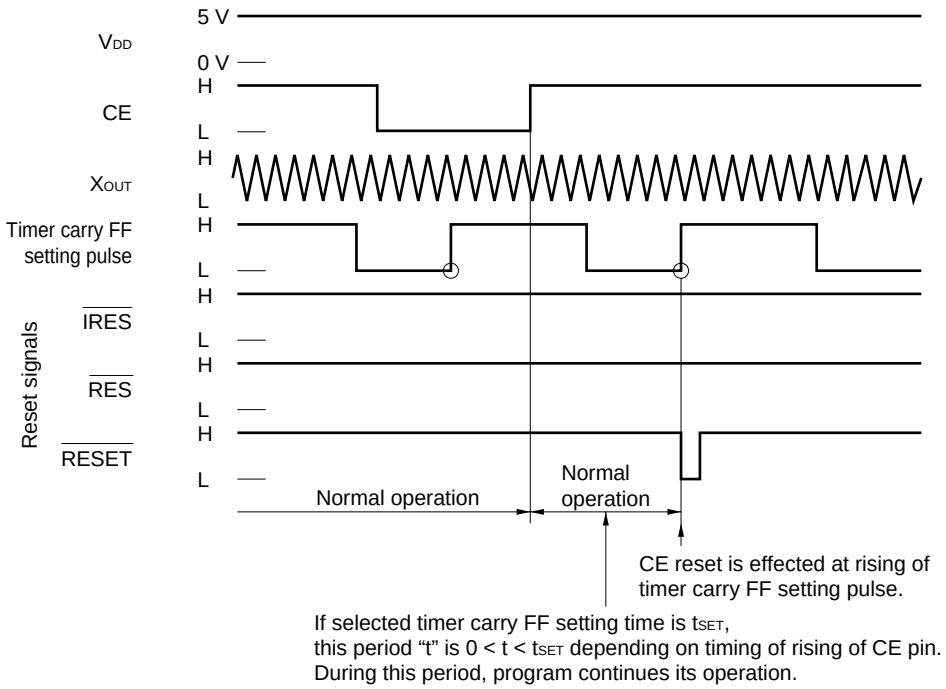
13.3.1 CE reset when clock stop (STOP s) instruction is not used

Figure 13-2 shows the operation of CE reset when the clock stop (STOP s) instruction is not used.

When the STOP s instruction is not used, the timer mode select register of the control registers is not initialized.

After the CE pin has gone high, therefore, the $\overline{\text{RESET}}$ signal is output at the rising edge of the timer carry FF setting pulse (1 ms, 5 ms, 100 ms, 250 ms) selected at that time, and the device is reset.

Figure 13-2. CE Reset Operation When Clock Stop Instruction Is Not Used



13.3.2 CE reset when clock stop (STOP s) instruction is used

Figure 13-3 shows the operation of CE reset when the clock stop (STOP s) instruction is used.

When the STOP s instruction is used, the $\overline{\text{IRES}}$, $\overline{\text{RES}}$, and $\overline{\text{RESET}}$ signals are output as soon as the STOP s instruction has been executed.

At this time, the timer mode select register of the control registers is initialized to 0000B by the $\overline{\text{RES}}$ signal, the timer carry FF setting signal is set to 100 ms.

Because the $\overline{\text{IRES}}$ signal is output while the CE pin is low, the halt status, which can be released by the timer carry, is forcibly set.

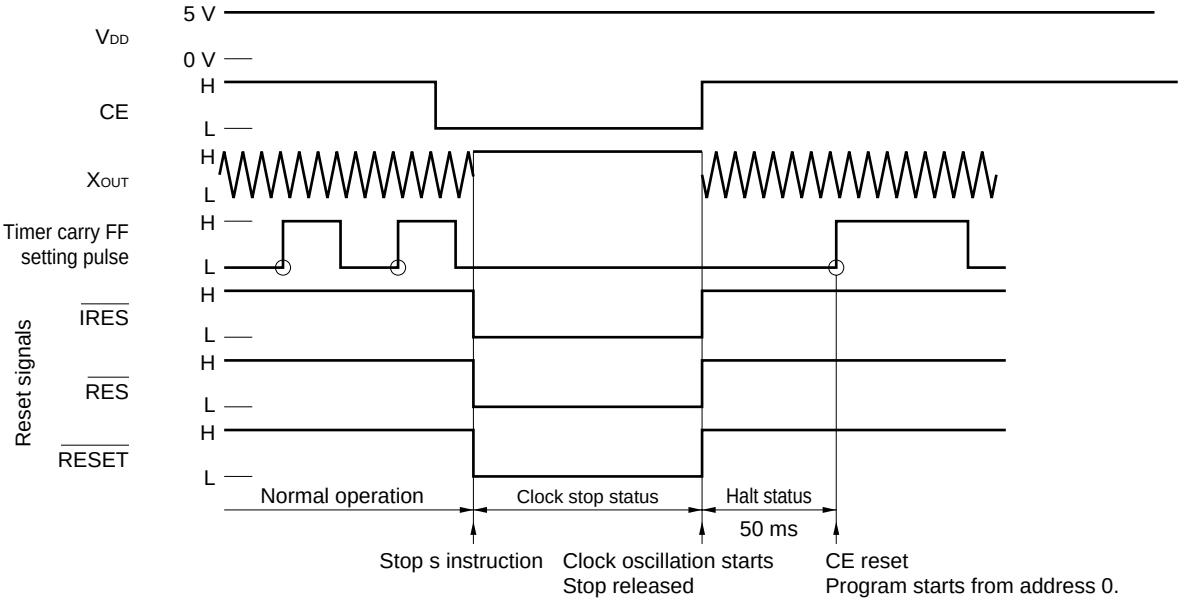
However, the device stops operation because the clock is stopped.

When the CE pin goes high, the clock stop status is released, and oscillation starts.

Because the halt status that can be released by the timer carry is set at this time by the $\overline{\text{IRES}}$ signal, the program starts from address 0 when the CE pin goes high and then the timer carry FF setting pulse rises.

Because the timer carry FF setting pulse is initialized to 100 ms, CE reset is effected 50 ms after the CE pin has gone high.

Figure 13-3. CE Reset Operation When Clock Stop Instruction Is Used



13.3.3 Notes on CE reset

Because CE reset is effected regardless of the instruction under execution, the following points <1> and <2> must be noted.

(1) Time to execute timer processing such as watch

When developing a watch program by using the timer carry or timer interrupt, the processing of that program must be completed in specific time.

For details, refer to **11.3.7 Notes on using timer carry** and **11.4.5 Notes on using timer interrupt**.

(2) Processing of data and flag used for program

Care must be exercised in rewriting the contents of data or flag that cannot be processed with one instruction and whose contents must not change even when CE reset is effected, such as a security code. This is explained in detail by using the following examples.

Example 1.

```
R1      MEM     0.01H       ; First digit of key input data of security code
R2      MEM     0.02H       ; Second digit of key input data of security code
R3      MEM     0.03H       ; First digit data for changing security code
R4      MEM     0.04H       ; Second digit data for changing security code
M1      MEM     0.11H       ; First digit of current security code
M2      MEM     0.12H       ; Second digit of current security code

START:

      Key input processing
      R1 ← contents of key A   ; Security code input wait mode
      R2 ← contents of key B   ; Substitutes contents of pressed key into R1 and R2.

        SET2    CMP, Z ;<1> ; Compares security code with input data.
        SUB     R1, M1
        SUB     R2, M2
        SKT1    Z
        BR      ERROR       ; Input data is different from security code.
MAIN:

      Key input processing
      R3 ← contents of key C   ; Security code rewriting mode
      R4 ← contents of key D   ; Substitutes contents of pressed key into R3 and R4.

        ST      M1, R3 ;<2> ; Rewrites security code.
        ST      M2, R4 ;<3>
        BR      MAIN
ERROR:

      Must not operate
```

Suppose the current security code is “12H” in the above program, the contents of data memory areas M1 and M2 are “1H” and “2H”, respectively.

If CE reset is effected, the contents of key input are compared with security code “12H” in <1>. If they coincide, the normal processing is performed.

If the security code is changed by the main processing, the new code is written to M1 and M2 in <2> and <3>.

Suppose the security code is changed to “34H”, “3H” and “4H” are written to M1 and M2, respectively, in <2> and <3>.

If CE reset is effected at the point where <2> is executed, the program is executed from address 0000H without <3> executed.

Consequently, the security code is changed to “32H”, making impossible to clear the security.

In this case, use the program shown in following Example 2.

Example 2.

```
R1      MEM     0.01H       ; First digit of key input data of security code
R2      MEM     0.02H       ; Second digit of key input data of security code
R3      MEM     0.03H       ; First digit data for changing security code
R4      MEM     0.04H       ; Second digit data for changing security code
M1      MEM     0.11H       ; First digit of current security code
M2      MEM     0.12H       ; Second digit of current security code
CHANGE FLG      0.13H.0     ; "1" while security code is changed

START:

      Key input processing
      R1 ← contents of key A  ; Security code input wait mode
      R2 ← contents of key B  ; Substitutes contents of pressed key into R1 and R2.

        SKT1    CHANGE ;<4> ; If CHANGE flag is "1"
        BR      SECURITY_CHK
        ST      M1, R3      ; rewrites M1 and M2.
        ST      M2, R4
        CLR1    CHANGE
SECURITY_CHK:
        SET2    CMP, Z ;<1> ; Compares security code with input data.
        SUB     R1, M1
        SUB     R2, M2
        SKT1    Z
        BR      ERROR       ; Input data is different from security code.
MAIN:

      Key input processing
      R3 ← contents of key C  ; Security code rewriting mode
      R4 ← contents of key D  ; Substitutes contents of pressed key into R3 and R4.

        SET1    CHANGE ;<5> ; Until security code is changed
                            ; Sets CHANGE flag to "1".
        ST      M1, R3 ;<2> ; Rewrites security code
        ST      M2, R4 ;<3>
        CLR1    CHANGE      ; When security code has been changed, sets
                            ; CHANGE flag to "0".
        BR      MAIN
ERROR:

      Must not operate
```

In the program in Example 2, the CHANGE flag is set to "1" in <5> before the security code is changed in <2> and <3>.

Therefore, the security code is rewritten in <4> even if CE reset is effected before <3> is executed.

13.4 Power-ON Reset

Power-ON reset is effected when the supply voltage V_{DD} of the device rises from a specific level (called power-ON clear voltage).

If the supply voltage V_{DD} is lower than the power-ON clear voltage, a power-ON clear signal (POC) is output from the voltage detection circuit shown in Figure 13-1.

When the power-ON clear voltage is output, the crystal oscillation circuit is stopped, and the device operation is stopped.

While the power-ON clear signal is output, the $\overline{IRES}$, $\overline{RES}$, and $\overline{RESET}$ signals are output.

If supply voltage V_{DD} exceeds the power-ON clear voltage, the power-ON clear signal is cleared, and crystal oscillation is started. At the same time, the $\overline{IRES}$, $\overline{RES}$, and $\overline{RESET}$ signals are also cleared.

At this time, the halt status is set to be released by the timer carry due to the $\overline{IRES}$ signal. Therefore, power-ON reset is effected at the rising edge of the next timer carry FF setting signal.

The timer carry FF setting signal is initialized to 100 ms by the $\overline{RESET}$ signal. For this reason, reset is effected 50 ms after supply voltage V_{DD} has exceeded the power-ON clear voltage, and the program is started from address 0.

This operation is illustrated in Figure 13-4.

The program counter, stack, system register, and control registers are initialized as soon as the power-ON clear signal has been output.

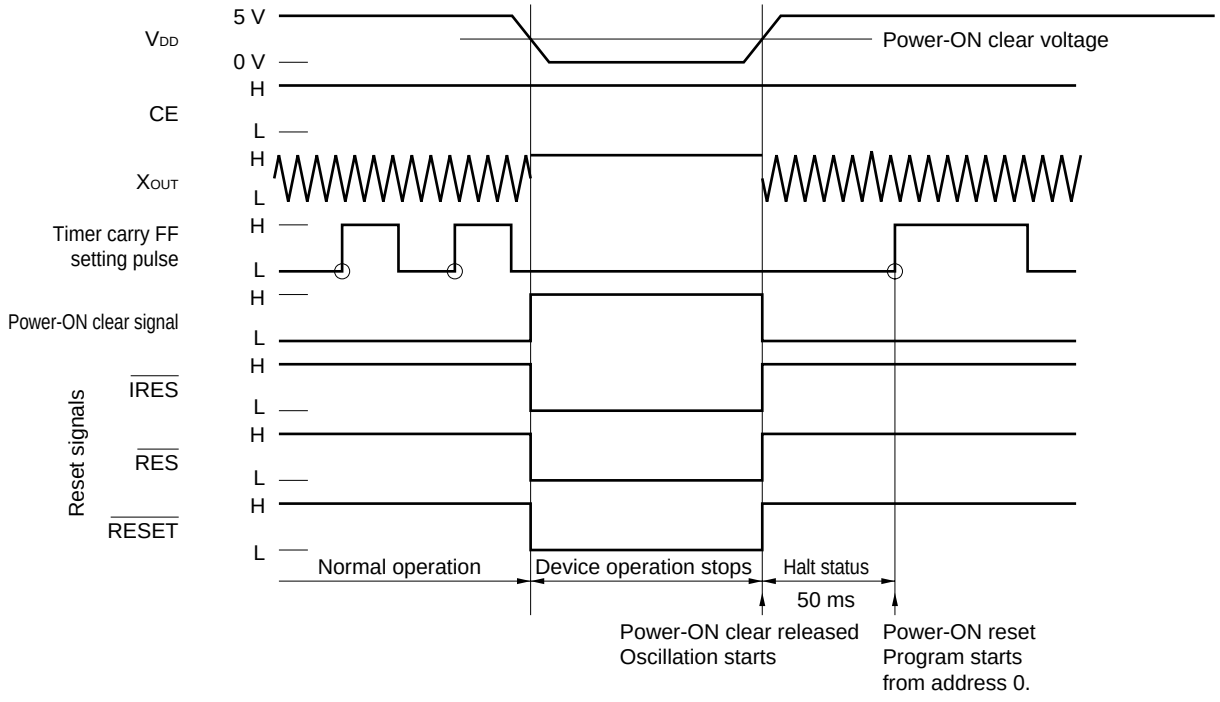
For the value to which each of the above registers is to be initialized, refer to the description of each register.

The power-ON clear voltage is 3.5 V (rated value) during normal operation, and 2.2 V (rated value) in the clock stop status.

The operations performed when the power-ON clear voltage is at the respective levels are explained in 13.4.1 and 13.4.2.

The operation to be performed if the supply voltage V_{DD} rises from 0 V is explained in 13.4.3.

Figure 13-4. Operation of Power-ON Reset



13.4.1 Power-ON reset during normal operation

Figure 13-5 (a) shows the operation.

As shown in the figure, the power-ON clear signal is output and the device operation stops regardless of the input level of the CE pin, if the supply voltage V_{DD} drops below 3.5 V.

If V_{DD} rises beyond 3.5 V again, the program starts from address 0000H after 50 ms of halt status.

The “normal operation” is when the clock stop instruction is not used and includes the halt status that is set by the halt instruction.

13.4.2 Power-ON reset in clock stop status

Figure 13-5 (b) shows the operation.

As shown in the figure, the power-ON clear signal is output and the device operation stops if supply voltage V_{DD} drops below 2.2 V.

However, it seems as if the device operation were not changed because the device is in the clock stop status.

When supply voltage V_{DD} rise beyond 3.5 V next time, the program starts from address 0000H after 50 ms of halt status.

13.4.3 Power-ON reset when supply voltage V_{DD} rises from 0 V

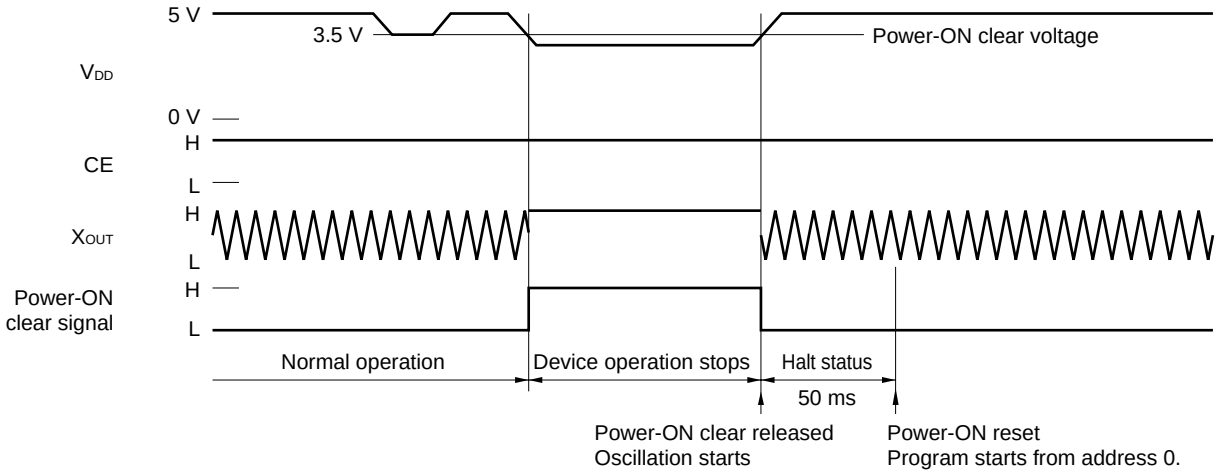
Figure 13-5 (c) shows the operation.

As shown in the figure, the power-ON clear signal is output until supply voltage V_{DD} rises from 0 V to 3.5 V.

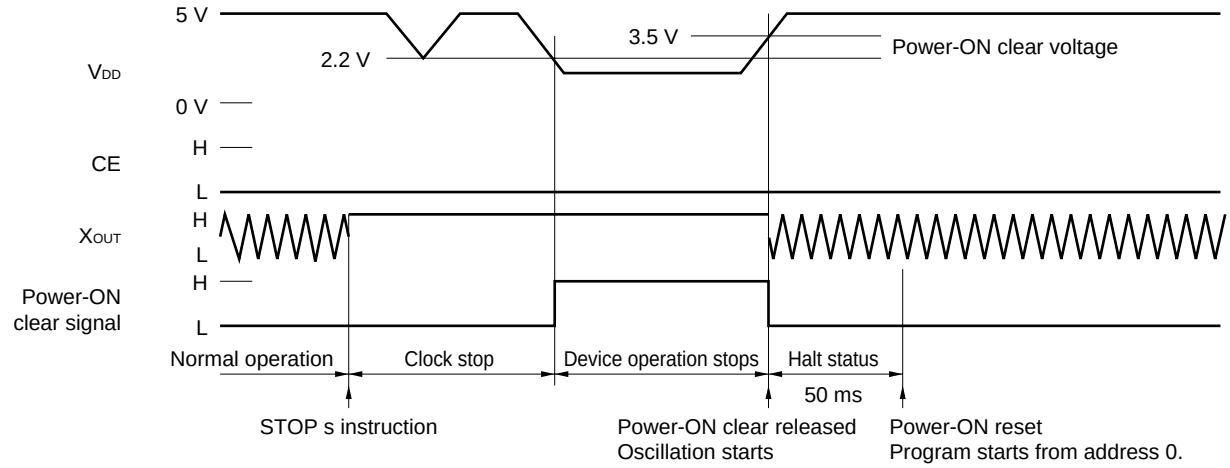
When V_{DD} rises beyond the power-ON clear voltage, the crystal oscillation circuit starts operating, and the program starts from address 0000H after 50 ms of halt status.

Figure 13-5. Power-ON Reset and Supply Voltage V_{DD}

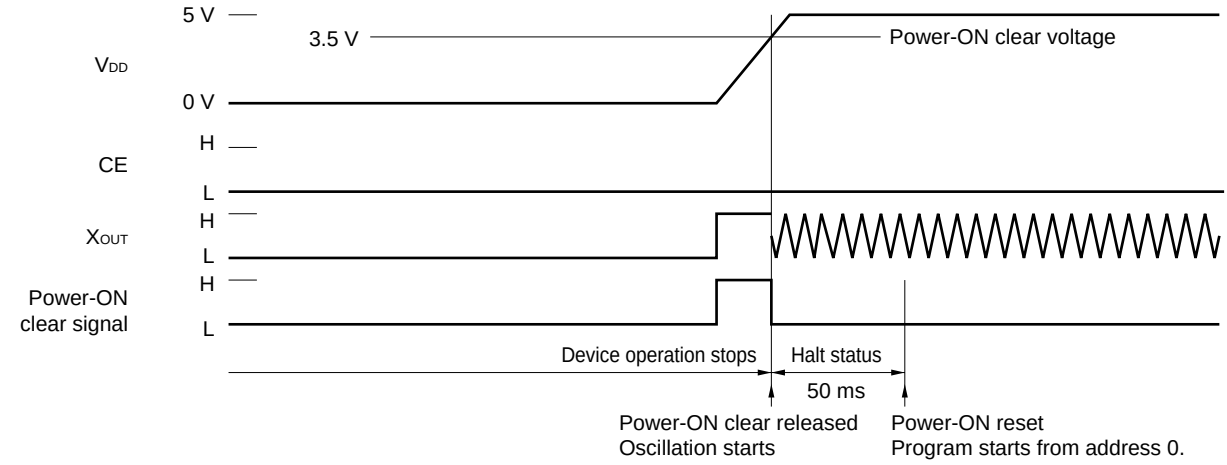
(a) During normal operation (including halt status)



(b) In clock stop status



(c) When supply voltage V_{DD} rises from 0 V



13.5 Relation between CE Reset and Power-ON Reset

There is a possibility that power-ON reset and CE reset are effected at the same time when power is first applied.

The reset operations performed at this time are explained in 13.5.1 through 13.5.3.

13.5.4 explains the points to be noted in raising supply voltage V_{DD}.

13.5.1 If V_{DD} pin and CE pin rise simultaneously

Figure 13-6 (a) shows the operation.

At this time, the program starts from address 0000H due to power-ON reset.

13.5.2 If CE pin rises in forced halt status of power-ON reset

Figure 13-6 (b) shows the operation.

At this time, the program starts from address 0000H due to power-ON reset in the same manner as in 13.5.1.

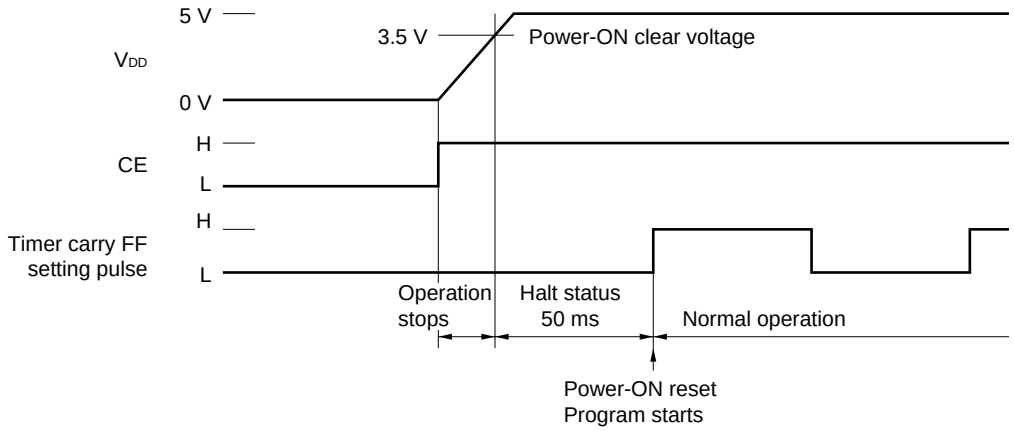
13.5.3 If CE pin rises after power-ON reset

Figure 13-6 (c) shows the operation.

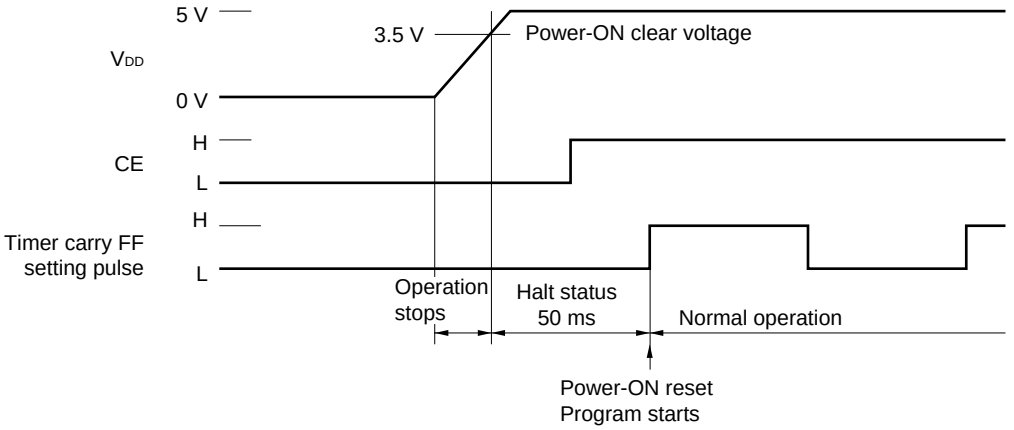
At this time, the program starts from address 0000H due to power-ON reset, and the program starts from address 0000H again at the rising of the next timer carry FF setting signal because of CE reset.

Figure 13-6. Relation between Power-ON Reset and CE Reset

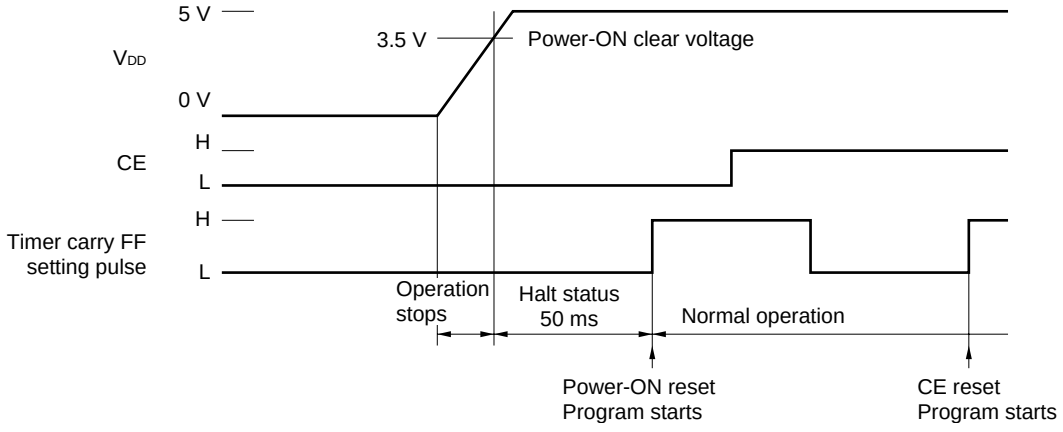
(a) If V_{DD} and CE pins rise simultaneously



(b) If CE pin rises in halt status



(c) If CE pin rises after power-ON reset



13.5.4 Notes on raising supply voltage V_{DD}

When raising supply voltage V_{DD}, keep in mind the following points (1) and (2).

(1) When raising supply voltage V_{DD} from power-ON clear voltage

It is necessary to raise supply voltage V_{DD} to higher than 3.5 V at least once.

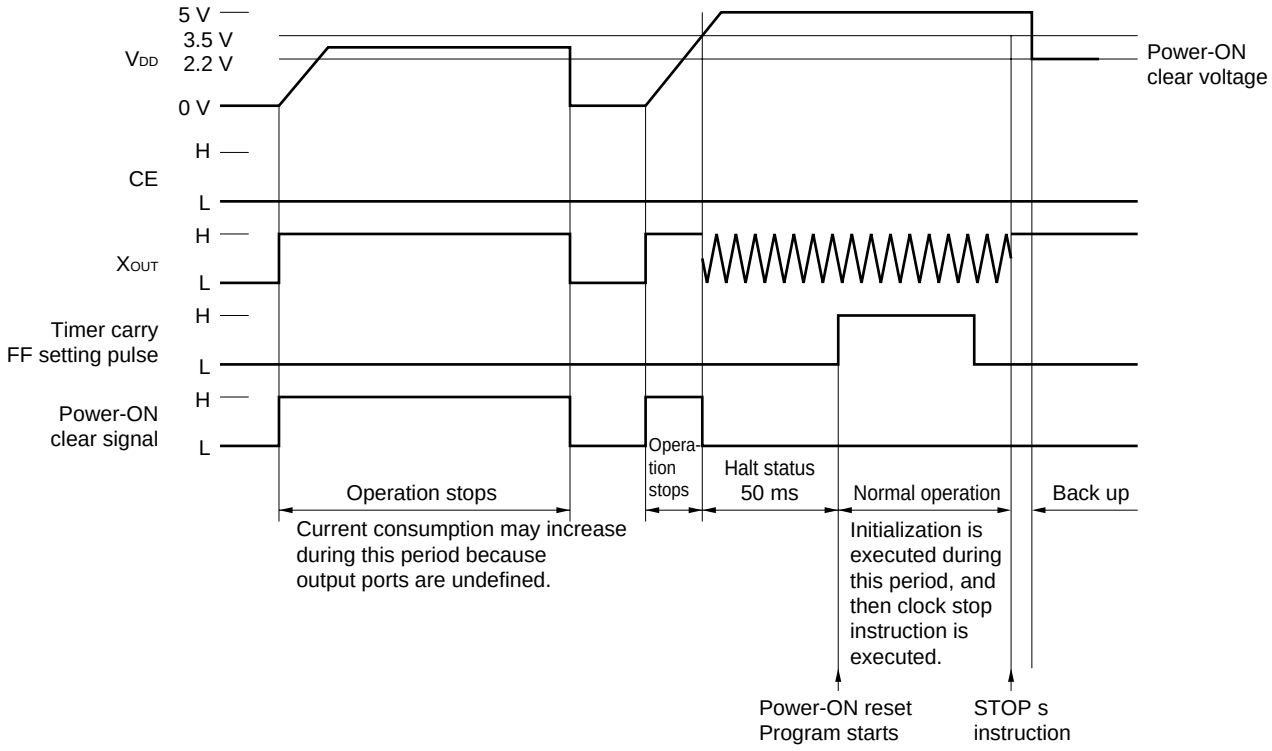
This is illustrated in Figure 13-7.

Suppose, for example, only a voltage less than 3.5 V is applied on application of V_{DD} with a program that backs up V_{DD} at 2.2 V by using the clock stop instruction, as shown in Figure 13-7, the power-ON clear signal is continuously output, and the program does not operate.

Because the output ports of the device output undefined values, the current consumption increases in some cases.

If the device is backed up by batteries, therefore, the back-up time is substantially shortened.

Figure 13-7. Notes on Raising V_{DD}



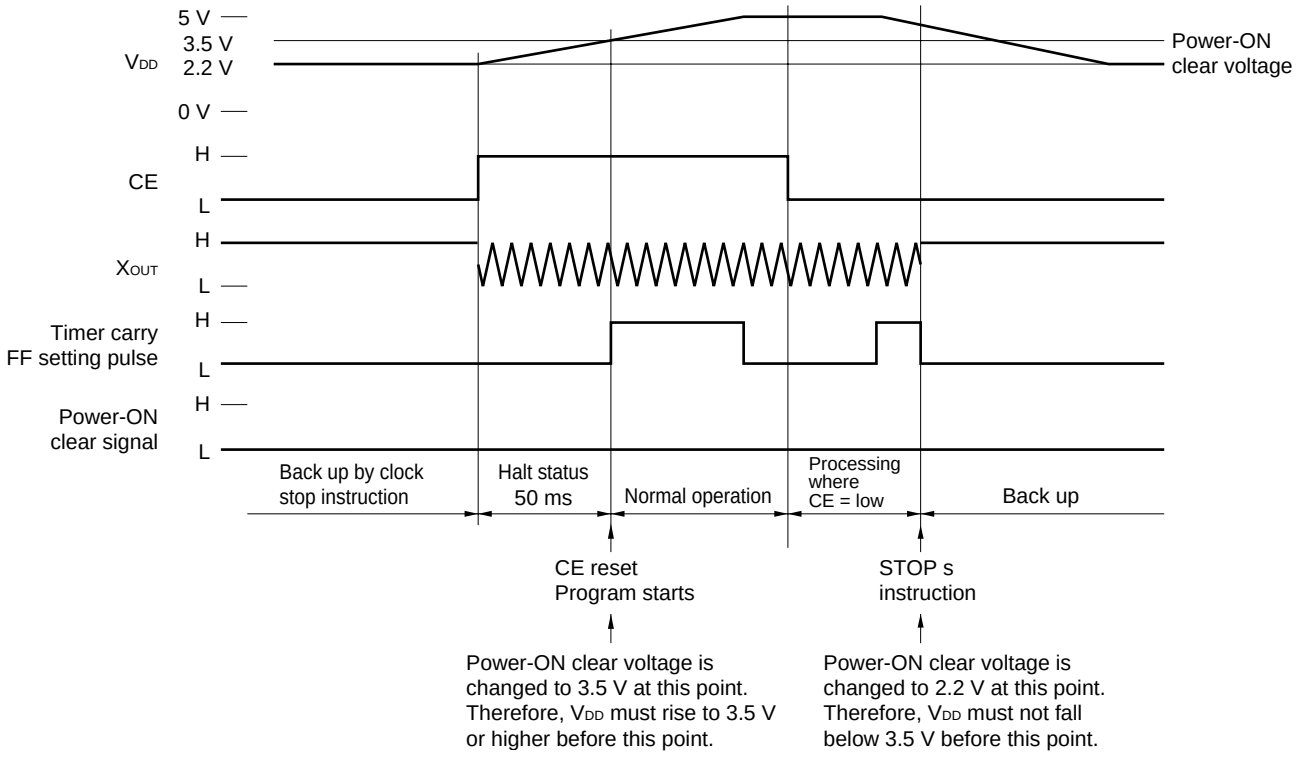
(2) Restoring from clock stop status

To restore the device from the back-up status while supply voltage V_{DD} is backed up at 2.2 V by using the clock stop instruction, V_{DD} must be raised to 3.5 V or higher within 50 ms after the CE pin has gone high.

As shown in Figure 13-8, the device is restored from the clock stop status by means of CE reset. Because the power-ON clear voltage is changed to 3.5 V 50 ms after the CE pin has gone high, power-ON reset is effected unless V_{DD} is 3.5 V or higher at this point.

The same applies when V_{DD} is lowered.

Figure 13-8. Restoring from Clock Stop Status



13.6 Power Failure Detection

Power failure detection is used to judge whether power-ON reset by application of supply voltage V_{DD}, or CE reset has been effected when the device is reset, as shown in Figure 13-9.

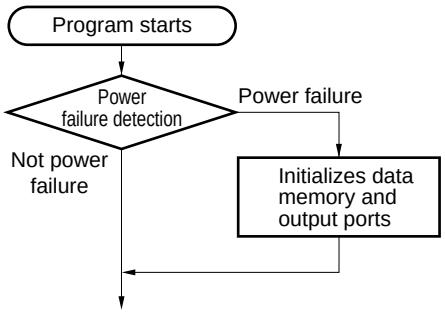
Because the contents of the data memory and ports are “undefined” on power application, these contents are initialized by means of power failure detection.

A power failure can be detected in two ways: by using the power failure detection circuit to detect the BTM0CY flag, and by detecting the contents of the data memory (RAM judgement).

13.6.1 and 13.6.2 explain how a power failure is detected by using the power failure detection circuit and BTM0CY flag.

13.6.3 and 13.6.4 explain how a power failure is detected by RAM judgement method.

Figure 13-9. Power Failure Detection Flow Chart



13.6.1 Power failure detection circuit

The power failure detection circuit consists of a voltage detection circuit, a timer carry disable flip-flop that is set by the output (power-ON clear signal) of the voltage detection circuit, and a timer carry, as shown in Figure 13-1.

The timer carry disable FF is set to “1” by the power-ON clear signal, and is reset to “0” when an instruction that reads the BTM0CY flag is executed.

When the timer carry disable FF is set to “1”, the BTM0CY flag is not set to “1”.

When the power-ON clear signal is output (at power-ON reset), the program is started with the BTM0CY flag reset, and the BTM0CY flag is disabled from being set until an instruction that reads the BTM0CY flag is executed.

Once the instruction that reads the BTM0CY flag has been executed, the BTM0CY flag is set each time the timer carry FF setting pulses has risen. It can be judged whether power-ON reset (power failure) or CE reset (not power failure) has been effected by detecting the contents of the BTM0CY flag when the device is reset. Power-ON reset has been effected if the BTM0CY flag is reset to “0”; CE reset has been effected if it is set to “1”.

The voltage at which a power failure can be detected is the same as the voltage at which power-ON reset is effected, or V_{DD} = 3.5 V during crystal oscillation, or V_{DD} = 2.2 V in the clock stop status.

Figure 13-10 shows the transition of the status of the BTM0CY flag.

Figures 13-11 and 13-10 show the timing chart and the operation of the BTM0CY flag.

Figure 13-10. Status Transition of BTM0CY Flag

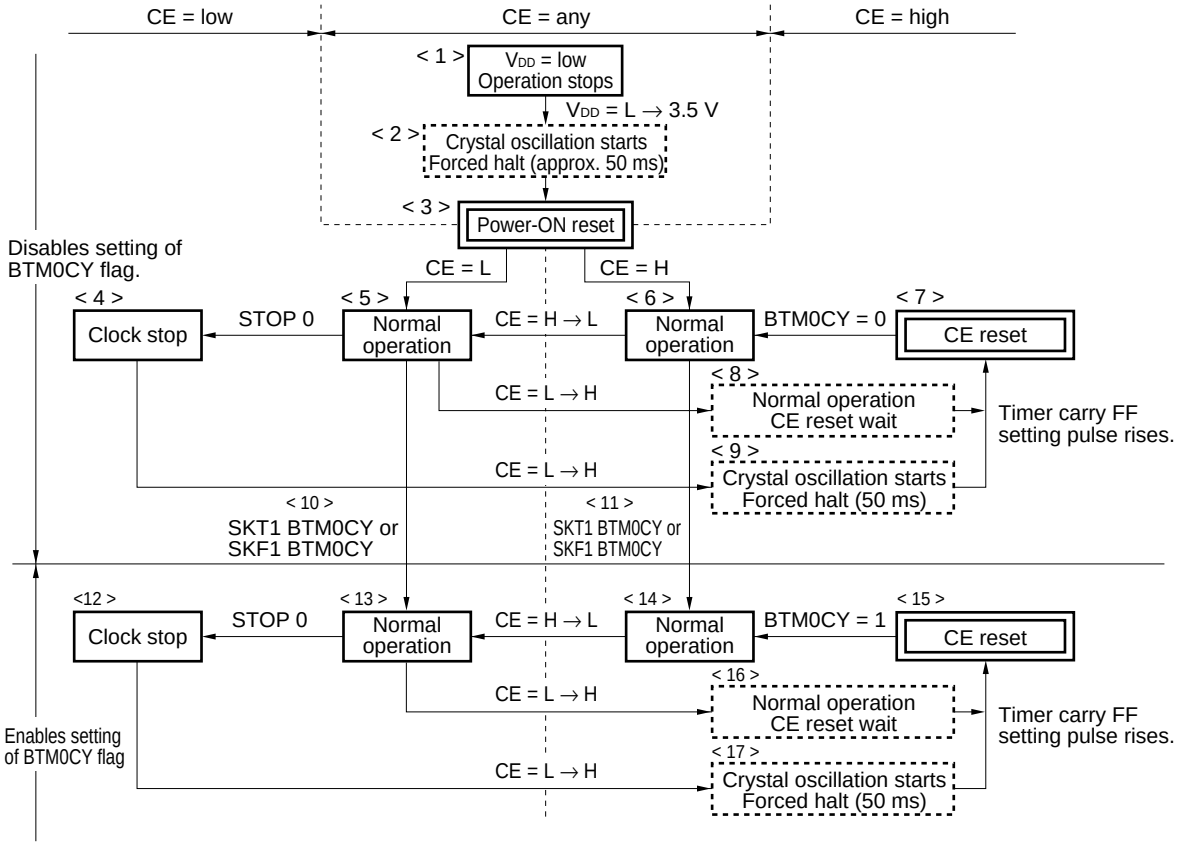
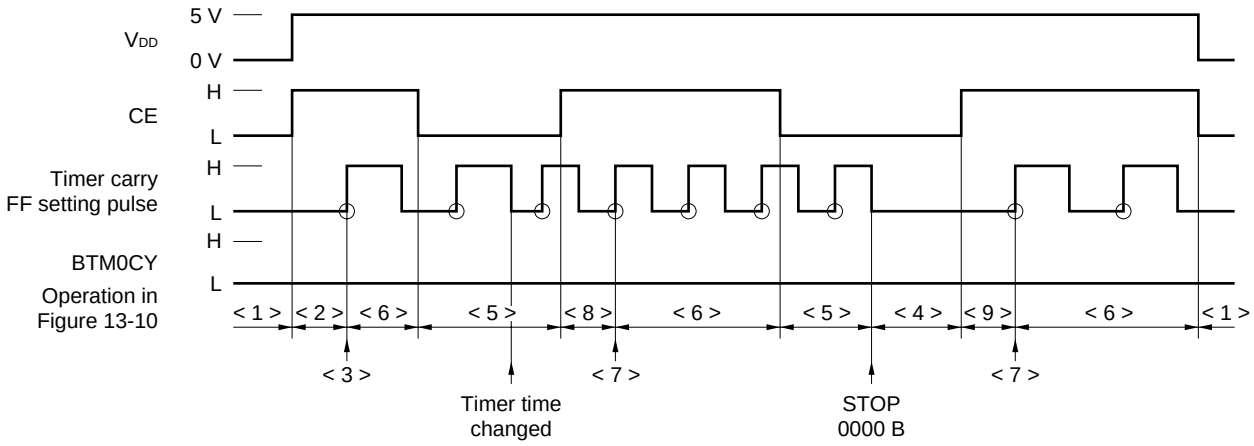
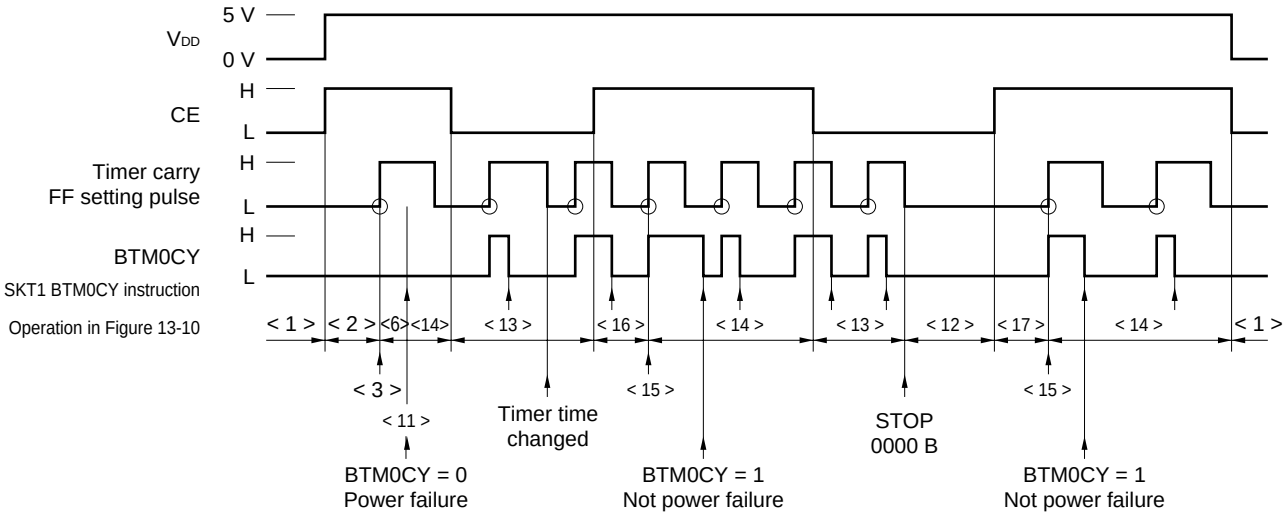


Figure 13-11. Operation of BTM0CY Flag

(a) When BTM0CY flag never detected (SKT1 BTM0CY or SKF1 BTM0CY is not executed)



(b) When detecting power failure by BTM0CY flag



13.6.2 Notes on detecting power failure by BTM0CY flag

The following points must be noted when using the BTM0CY flag for watch counting.

(1) Updating watch

When developing a watch program by using the timer carry, the watch must be updated after a power failure has been detected.

This is because counting of the watch is skipped once because the BTM0CY flag is reset to “0” when the BTM0CY flag is read on detection of a power failure.

(2) Watch updating processing time

The processing to update the watch must be completed before the next timer carry FF setting pulse rises. This is because, if the CE pin goes high during the watch updating processing, CE reset is effected without the watch updating processing completed.

For further information on (1) and (2) above, refer to **11.3.7 (b) Correction of timer carry on CE reset**.

When detecting a power failure, the following points must be noted.

(3) Timing of power failure detection

To count the watch by using the BTM0CY flag, the BTM0CY flag must be read to detect a power failure within the time since the program has started from address 0000H until the next timer carry FF setting pulse rises.

For example if the timer carry FF setting time is set to 5 ms, and a power failure is detected 6 ms after the program has been started, the BTM0CY flag is overlooked once.

For details, refer to **11.3.7 (b) Correction of timer carry on CE reset**.

Power failure detection and initial processing must be completed within the timer carry FF setting time as shown in the following example.

This is because, if the CE pin goes high and CE reset is effected during power failure detection processing and initial processing, these processing may be stopped in midway, and thus problems may occur.

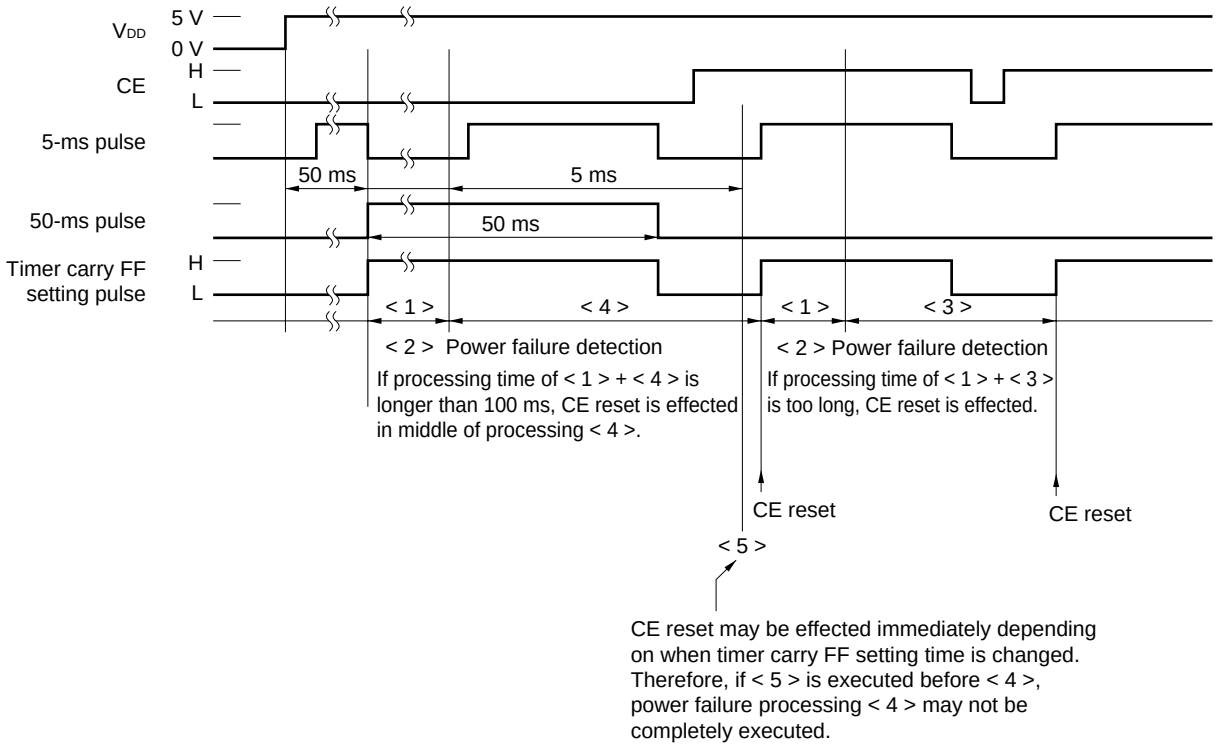
To change the timer carry FF setting time by the initial processing, the instruction that changes the time must be executed at the end of the initial processing, and the instruction must be one instruction.

This is because the initial processing may not be completely executed because of CE reset if the timer carry FF setting time is changed before the initial processing is executed, as shown in the following example.

Example

```
START:                                  ; Program address 0000H
;<1>

     Processing on reset

;<2>
  SKT1     BTM0CY                       ; Power failure detection
  BR       INITIAL
BACKUP:
;<3>

     Watch updating

  BR       MAIN
INITIAL:
;<4>

     Initial processing

;<5>
  INITFLG BTM0CK1, NOT BTM0CK0          ; Embedded macro
                                        ; Sets timer carry FF setting time to 5 ms

MAIN:

     Main processing

  SKT1     BTM0CY
  BR       MAIN

     Watch updating

  BR       MAIN
```

Example of operation



13.6.3 Power failure detection by RAM judgement method

The RAM judgement method is to detect a power failure by judging whether the contents of the data memory at a specific address are the specified value.

An example of a program that detects a power failure by the RAM judgement method is shown below.

The RAM judgement method detects a power failure by comparing an “undefined” value with the “specified value” because the contents of the data memory are “undefined” on application of supply voltage V_{DD}.

Therefore, there is a possibility that a wrong judgment may be made as explained in **13.6.4 Notes on detecting power failure by RAM judgement method**.

When the RAM judgement method is used, however, the device can be backed up at a voltage lower than that at which a power failure is detected, by using the power failure detection circuit, as shown in Table 13-2.

Table 13-2. Comparing Power Failure Detection by Power Failure Detection Circuit and RAM Judgement Method

	Power Failure Detection Circuit		RAM Judgement Method	
Data hold voltage	Effective value	Rated value	Effective value	Rated value
(in clock stop status)	1-2 V	2.2 V	0-1 V	2.0 V
Operating status	No malfunctioning		Malfunctioning may occur	

Example Program to detect power failure by RAM judgement method

```
M012    MEM     0.12H
M034    MEM     0.34H
M056    MEM     0.56H
M107    MEM     1.07H
M128    MEM     1.28H
M16F    MEM     1.6FH
DATA0   DAT     1010B
DATA1   DAT     0101B
DATA2   DAT     0110B
DATA3   DAT     1001B
DATA4   DAT     1100B
DATA5   DAT     0011B

START:
        SET2    CMP, Z
        SUB     M012, #DATA0    ; If M012 = DATA0 and
        SUB     M034, #DATA1    ; M034 = DATA1 and
        SUB     M056, #DATA2    ; M035 = DATA2 and
        BANK1
        SUB     M107, #DATA3    ; M107 = DATA3 and
        SUB     M128, #DATA4    ; M128 = DATA4 and
        SUB     M16F, #DATA5    ; M16F = DATA5,
        BANK0
        SKF1    Z
        BR      BACKUP          ; branches to BACKUP
 ;INITIAL:

        Initial processing

        MOV     M012, #DATA0
        MOV     M034, #DATA1
        MOV     M056, #DATA2
        BANK1
        MOV     M107, #DATA3
        MOV     M128, #DATA4
        MOV     M16F, #DATA5
        BR      MAIN
BACKUP:

        Backup processing

MAIN:

        Main processing
```

13.6.4 Notes on detecting power failure by RAM judgement method

The value of the data memory on application of supply voltage V_{DD} is basically "undefined", and therefore, the following points (1) and (2) must be noted.

(1) Data to be compared

Where the number of bits of the data memory to be compared by the RAM judgement method is "n bits", the probability at which the value of the data memory coincides with the value to be compared on application of V_{DD} is $(1/2)^n$.

This means that backup is judged at a probability of $(1/2)^n$ when a power failure is detected by the RAM judgement method.

To lower this probability, as many bits as possible must be compared.

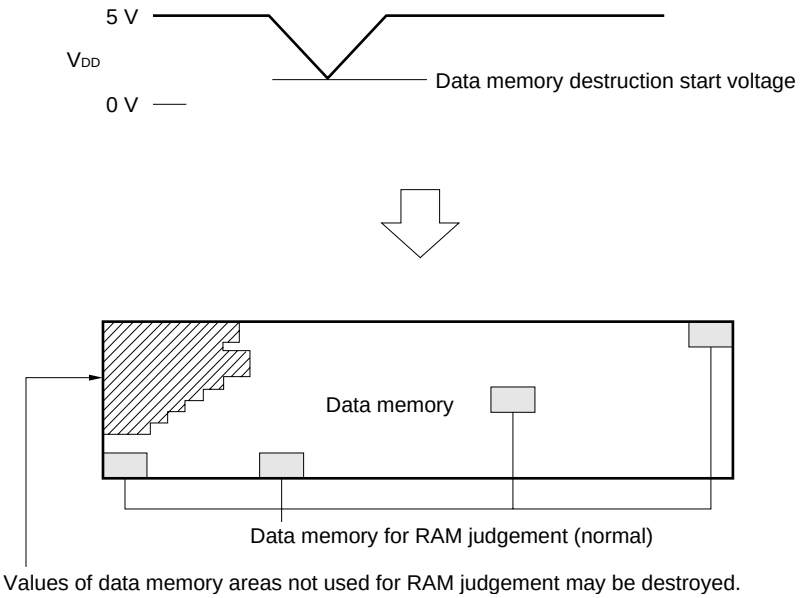
Because the contents of the data memory on application of V_{DD} are likely to be the same value such as "0000B" and "1111B", it is recommended to mix "0" and "1" as data to be compared, such a "1010B" and "0110B" to reduce the possibility of a wrong judgment.

(2) Notes on program

If V_{DD} rises from the level at which the data memory contents may be destroyed as shown in Figure 13-12, and even if the value of the data memory area to be compared is normal, the values of the other data memory areas may be destroyed.

This is judged as backup if a power failure is detected by the RAM judgement method. Therefore, consideration must be given so that the program does not hang up even if the contents of the data memory are destroyed.

Figure 13-12. V_{DD} and Destruction of Data Memory Contents



14. PLL FREQUENCY SYNTHESIZER

The PLL (Phase Locked Loop) frequency synthesizer is used to lock the frequency in the MF (Medium Frequency), HF (High Frequency), and VHF (Very High Frequency) bands to a specific frequency by comparing phase differences.

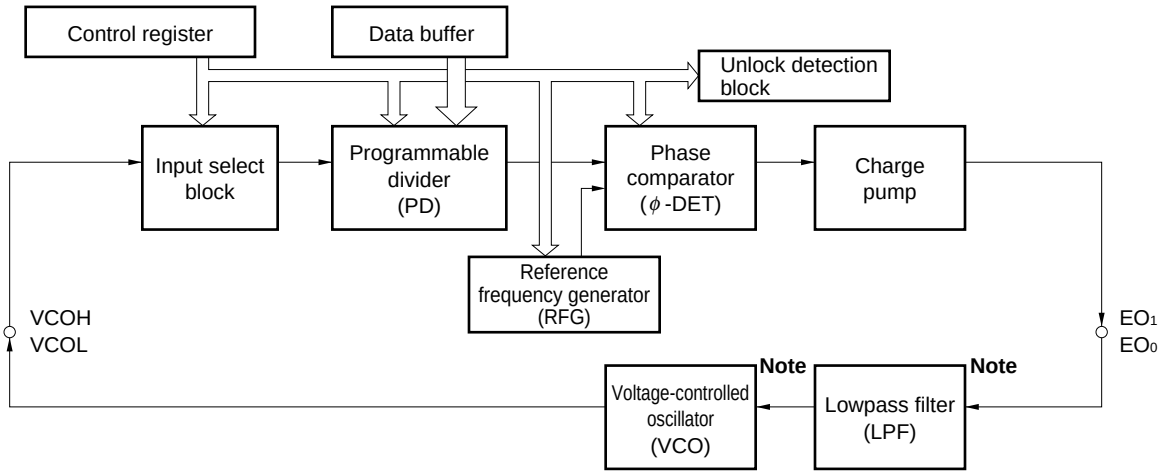
14.1 Configuration of PLL Frequency Synthesizer

Figure 14-1 shows the block diagram of the PLL frequency synthesizer.

As shown in the figure, the PLL frequency synthesizer consists of an input select block, programmable divider (PD), phase comparator (ϕ-DET), reference frequency generator (RFG), and charge pump.

By connecting these blocks with an external lowpass filter (LPF) and voltage-controlled oscillator (VCO), a PLL frequency synthesizer is organized.

Figure 14-1. Block Diagram of PLL Frequency Synthesizer



Note External circuit

14.2 Functional Outline of PLL Frequency Synthesizer

The PLL frequency synthesizer divides a signal input from the VCOH (pin 32) or VCOL (pin 31) pin by using the programmable divider and outputs a phase difference from the reference frequency from the EO_1 and EO_0 pins.

The PLL frequency synthesizer operates only when the CE pin is high, and is disabled when the CE pin is low.

For the details on the disable status of the PLL frequency synthesizer, refer to **14.6**.

The following subsections 14.2.1 through 14.2.6 outline the function of each block of the PLL frequency synthesizer.

14.2.1 Input select block

This block selects the pin from which a signal output from an external voltage-controlled oscillator is input.

As the input pin, the VCOH or VCOL pin is selected by the PLL mode select register (RF address 21H).

For details, refer to **14.3**.

14.2.2 Programmable divider

The programmable divider divides the signal input from the VCOH or VCOL pin at the division ratio set by the program.

Two types of division modes can be selected: direct division and pulse swallow modes.

The division mode is selected by the PLL mode select register.

The division ratio is set by the PLL data register (PLLR: peripheral address 41H) via the data buffer.

For details, refer to **14.3**.

14.2.3 Reference frequency generator

This generator generates a reference frequency to be compared by the phase comparator.

Twelve types of reference frequencies can be selected by using the PLL reference mode select register (RF address 31H).

For details, refer to **14.4**.

14.2.4 Phase comparator and unlock detection block

The phase comparator compares the division signal output by the programmable divider with the signal from the reference frequency generator, and outputs a phase difference.

The unlock detection block detects the unlock status of the PLL.

The unlock status of the PLL is detected by the PLL unlock FF judge register (RF address 05H).

For details, refer to **14.5**.

14.2.5 Charge pump

The charge pump outputs the signal output by the phase comparator to the EO_1 an EO_0 pins as a high-level, low-level, or floating signal.

For details, refer to **14.5**.

14.3 Input Select Block and Programmable Divider

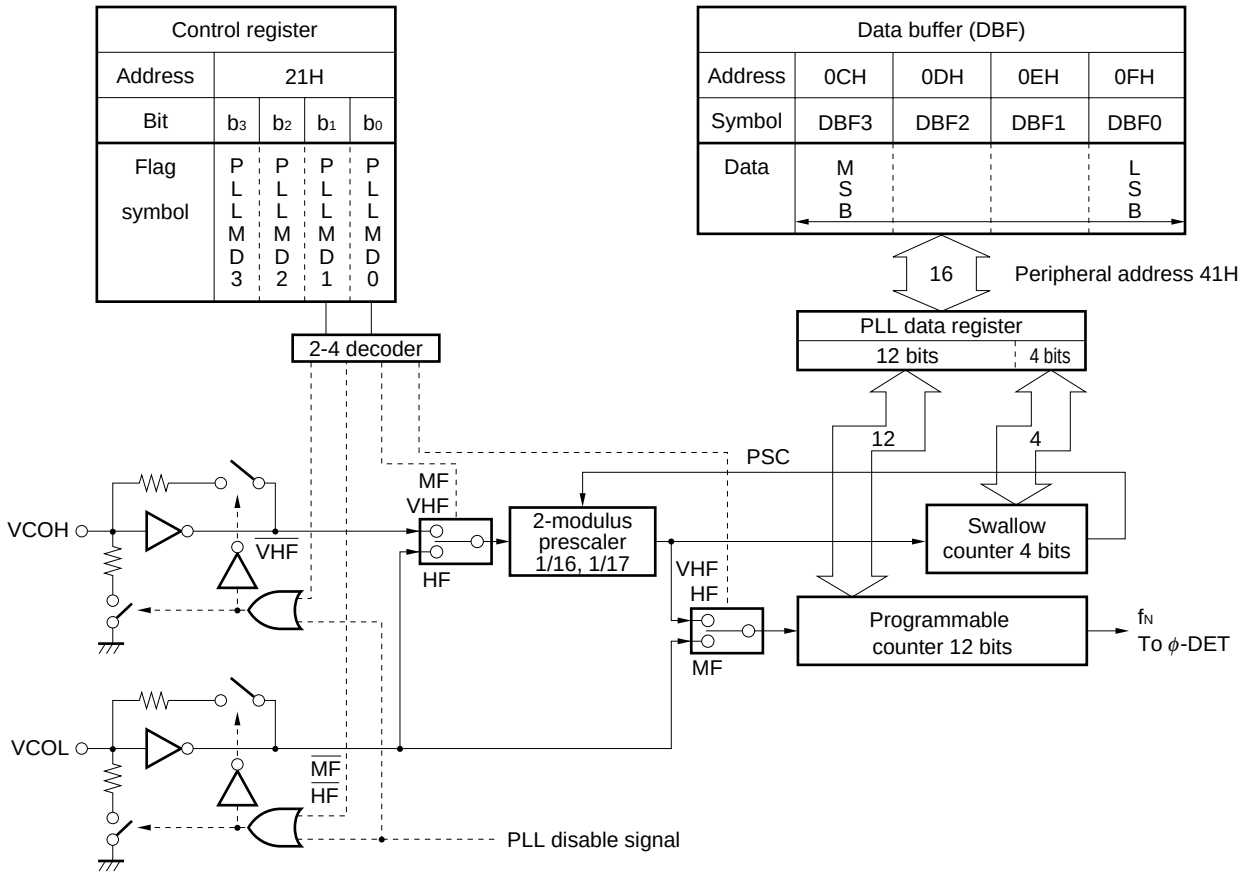
14.3.1 Configuration of input select block and programmable divider

Figure 14-2 shows the configuration of the input select block and programmable divider.

As shown in the figure, the input select block consists of the VCOH and VCOL pins, and the amplifiers of the respective pins.

The programmable divider consists of a 2-modulus prescaler, swallow counter, programmable counter, and division mode select switch.

Figure 14-2. Configuration of Input Select Block and Programmable Divider



14.3.2 Functions of input select block and programmable divider

The input select block and programmable divider selects the input pin and division mode of the PLL frequency synthesizer.

As the input pin, the VCOH or VCOL pin can be selected.

The selected pin goes into an intermediate-potential state (approx. 1/2 V_{DD}). The pin not selected is internally pulled down.

These pins input signals via an AC amplifier, and the DC component of the input signal must be cut off by connecting a capacitor to the pin in series.

Either the direct division mode or pulse swallow mode can be selected as the division mode.

The programmable counter divides the signal input from the VCOH or VCOL pin in a specified division mode according to the values set to the swallow counter and programmable counter.

Table 14-1 show the input pins (VCOH and VCOL) and division modes.

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

14.3.3 explains the configuration and function of the PLL mode select register.

The division ratio is set to the programmable divider by the PLL data register via the data buffer.

14.3.4 explains the programmable divider and PLL data register.

Table 14-1. Input Pins and Division Modes

Division Mode	Pin	Input Frequency (MHz)	Input Amplitude (V_{p-p})	Settable Division Ratio	Division Ratio Settable in Data Buffer
Direct division (MF)	VCOL	0.5-30	0.2	16 to $2^{12} - 1$	010×H-FFF×H (×: low-order 4 bits are don't care)
Pulse swallow (HF)	VCOL	5-40	0.2	256 to $2^{16} - 1$	0100H-FFFFH
Pulse swallow (VHF)	VCOH	9-150	0.2	256 to $2^{16} - 1$	0100H-FFFFH

14.3.3 Configuration and function of PLL mode select register

The PLL mode select register specifies the division mode of the PLL frequency synthesizer and the pin to be used.

The configuration and function of the PLL mode select register are shown below.

The paragraphs (1) through (4) below outlines the respective division modes.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
PLL mode select register	PLLMD3	PLLMD2	PLLMD1	PLLMD0	21H	R/W

PLLMD1	PLLMD0	Sets division mode of PLL frequency synthesizer
0	0	Disables VCOL and VCOH pins
0	1	Direct division mode (VCOL pin, MF mode)
1	0	Pulse swallow (VCOH pin, VHF mode)
1	1	Pulse swallow (VCOL pin, HF mode)

Fixed to "0"

		PLLMD3	PLLMD2	PLLMD1	PLLMD0
On reset	Power-ON	0	0	0	0
	Clock stop	↓	↓	0	0
	CE	↓	↓	Retained	

(1) Direct division mode (MF)

In this mode, the VCOL pin is used.

The VCOH pin is pulled down.

In the direct division mode, the frequency of the input signal is divided only by the programmable counter.

(2) Pulse swallow mode (HF)

The VOL pin is used in this mode.

The VCOH pin is pulled down.

In this mode, the frequency of the input signal is divided by the swallow counter and programmable counter.

(3) Pulse swallow mode (VHF)

The VCOH pin is used in this mode.

The VCOL pin is pulled down.

In this mode, the frequency of the input signal is divided by the swallow counter and programmable counter.

(4) Disabling VCOL and VCOH pins

The VCOH and VCOL pins are internally pulled down.

However, the phase comparator, reference frequency generator, and charge pump operate.

Therefore, the operation is different from that in the PLL disable status to be explained later.

14.3.4 Programmable divider and PLL data register

The programmable divider divides the signal input from the VCOH or VCOL pin by the value set to the swallow counter and programmable counter.

The swallow counter and programmable counter are 4-bit binary down counters.

The division ratio is set to the swallow counter and programmable counter by the PLL data register (PLLR: peripheral address 41H) via data buffer.

Data is set to or read from the PLL data register by using the “PUT PLLR, DBF” or “GET DBF, PLLR” instruction.

The value to be divided is called N value.

For how to set the N value in each division mode, refer to **14.7**.

(1) PLL data register and data buffer

The relation between the PLL data register and data buffer is explained next.

In the direct division mode, the high-order 12 bits are valid, and all the 16 bit are valid in the pulse swallow mode.

In the direct division mode, all the high-order 12 bits are set to the programmable counter.

In the pulse swallow mode, the high-order 12 bits are set to the programmable counter, and the low-order 4 bits are set to the swallow counter.

(2) Relation between division value N and divided output frequency

The relation between the value “N” set to the PLL data register and the frequency “f_N” of the signal divided and output by the programmable divider is as follows.

For details, refer to **14.7**.

(a) In direct division mode (MF)

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

(b) In pulse swallow mode (HF and VHF)

$$f_N = \frac{f_{IN}}{N} \quad N: \text{ 16 bits}$$

Name	Data buffer															
Symbol	DBF3				DBF2				DBF1				DBF0			
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	Transfer data															

GET can be executed

16

PUT can be executed

Peripheral register																			
Name	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	Symbol	Peripheral address	Peripheral address
PLL data register	Valid data																PLLR	41H	PLL frequency synthesizer

Sets division ratio of PLL frequency synthesizer

Direct division mode	0	Don't care
	15 (00FH)	Don't care
	16 (010H)	Don't care
	x	Don't care
	$2^{12}-1$ (FFFH)	Don't care
Pulse swallow mode	0	
	255 (00FFH)	
	256 (0100H)	
	x	
	$2^{16}-1$ (FFFFH)	

Direct division mode	Setting prohibited
	Division ratio N: N = x
Pulse swallow mode	Setting prohibited
	Division ratio N: N = x

14.4 Reference Frequency Generator

14.4.1 Configuration and function of reference frequency generator

Figure 14-3 shows the configuration of the reference frequency generator.

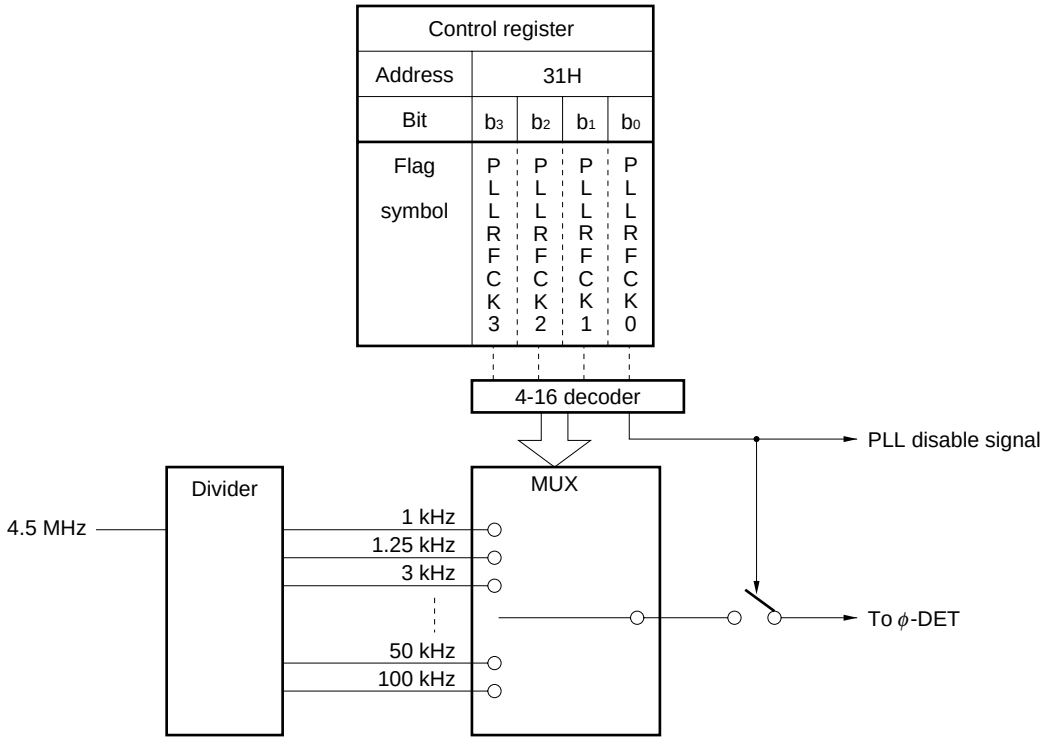
As shown in the figure, the reference frequency generator divides 4.5 MHz of the crystal oscillation circuit to generate the reference frequency "f_r" of the PLL frequency synthesizer.

Twelve reference frequencies can be selected: 1, 1.25, 2.5, 3, 5, 6.25, 9, 10, 12.5, 25, 50, and 100 kHz.

Reference frequency f_r is selected by the PLL reference mode select register.

14.4.2 shows the configuration and function of the PLL reference mode select register.

Figure 14-3. Configuration of Reference Frequency Generator (RFG)



14.4.2 Configuration and function of PLL reference mode select register

The configuration and function of the PLL reference mode select register are shown below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
PLL reference mode select register	PLLRFCK3	PLLRFCK2	PLLRFCK1	PLLRFCK0	31H	R/W

b_3	b_2	b_1	b_0	Sets reference frequency f_r of PLL frequency synthesizer
0	0	0	0	1.25 kHz
0	0	0	1	2.5 kHz
0	0	1	0	5 kHz
0	0	1	1	10 kHz
0	1	0	0	6.25 kHz
0	1	0	1	12.5 kHz
0	1	1	0	25 kHz
0	1	1	1	50 kHz
1	0	0	0	3 kHz
1	0	0	1	Setting prohibited
1	0	1	0	Setting prohibited
1	0	1	1	Setting prohibited
1	1	0	0	1 kHz
1	1	0	1	9 kHz
1	1	1	0	100 kHz
1	1	1	1	PLL disabled

		b_3	b_2	b_1	b_0
On reset	Power-ON	1	1	1	1
	Clock stop	1	1	1	1
	CE	Retained			

When the PLL is disabled by the PLL reference mode select register, the VCOH and VCOL pins are internally pulled down.

The EO_1 and EO_0 pins are floated.

For disabling the PLL, refer to **14.6**.

14.5 Phase Comparator (φ-DET), Charge Pump, and Unlock Detection Block

14.5.1 Configuration of phase comparator, charge pump, and unlock detection block

Figure 14-4 shows the configuration of the phase comparator, charge pump, and unlock detection block.

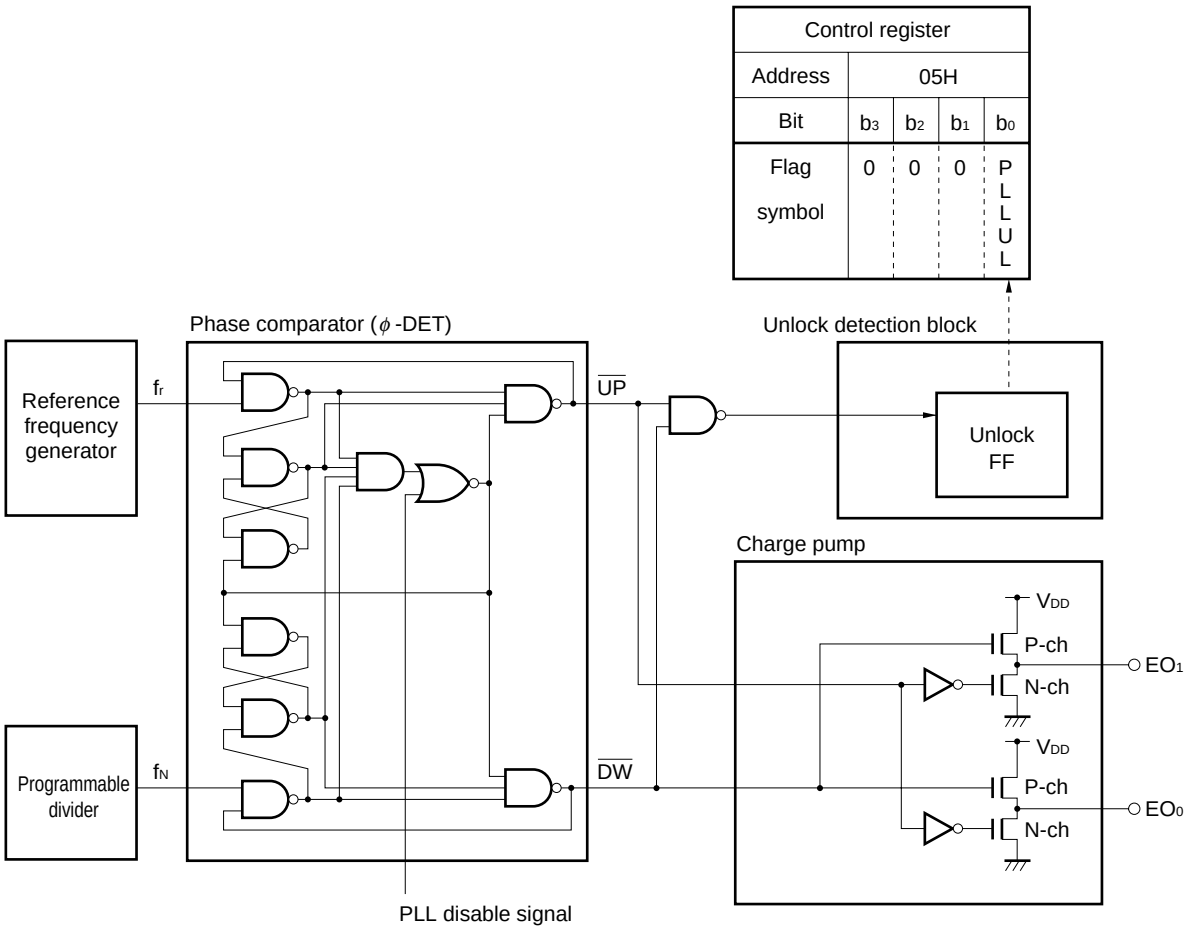
The phase comparator compares the divided frequency output "f_N" of the programmable divider with the reference frequency output "f_r" of the reference frequency generator, and outputs an up request ($\overline{UP}$) or down request ($\overline{DW}$) signal.

The charge pump outputs the output of the phase comparator from the error out (EO_1 and EO_0) pins.

The unlock detection block detects the unlock status of the PLL frequency synthesizer.

The following subsections 14.5.2 through 14.5.4 respectively explain the operations of the phase comparator, charge pump, and unlock detection block.

Figure 14-4. Configuration of Phase Comparator, Charge Pump, and Unlock Detection Block



14.5.2 Function of phase comparator

As shown in Figure 14-4, the phase comparator compares the divided frequency output "f_N" of the programmable divider with the reference frequency output "f_r" of the reference frequency generator, and outputs an up request or down request signal.

If the divided frequency f_N is lower than the reference frequency f_r, the phase comparator outputs the up request signal; if f_N is higher than f_r, it outputs the down request signal.

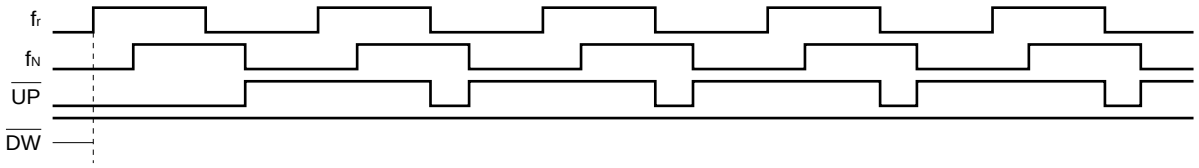
Figure 14-5 shows the relation among the reference frequency f_r, divided frequency f_N, up request signal, and down request signal.

When the PLL is disabled, neither the up request nor down request signal is output.

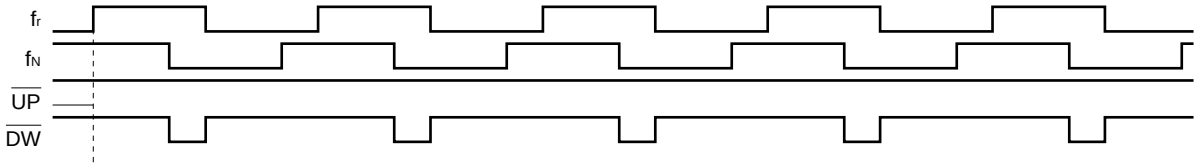
The up request and down request signals are respectively input to the charge pump and unlock detection block.

Figure 14-5. Relation among f_r, f_N, $\overline{UP}$, and $\overline{DW}$

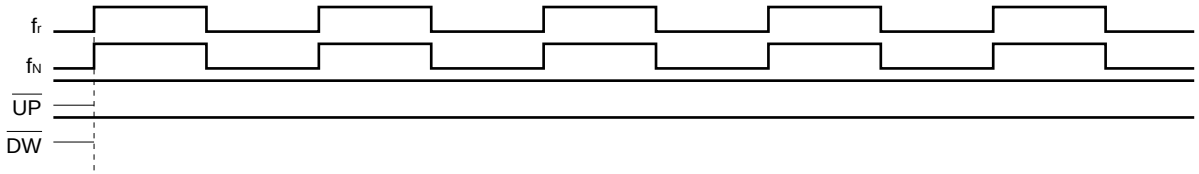
(a) If f_N is behind f_r in phase



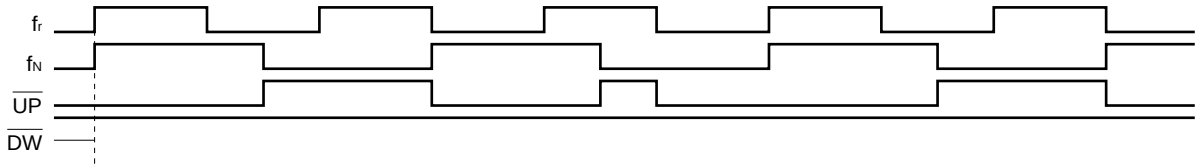
(b) If f_N leads f_r in phase



(c) If f_N and f_r are in phase



(d) If f_N is lower than f_r in frequency



14.5.3 Charge pump

As shown in Figure 14-4, the charge pump outputs the up request signal or down request signal from the phase comparator to the error out (EO_1 and EO_0) pins.

Therefore, the relation among the outputs of the error out pins, divided frequency f_N, and reference frequency f_r are as follows.

When reference frequency f_r > divided frequency f_N: low-level output
When reference frequency f_r < divided frequency f_N: high-level output
When reference frequency f_r = divided frequency f_N: floating output

14.5.4 Unlock detection block

As shown in Figure 14-4, the unlock detection block detects the unlock status of the PLL frequency synthesizer by using the up request or down request signal from the phase comparator.

Because either of the up request or down request signal outputs a low level in the unlock status, this low-level signal is used to detect the unlock status.

In the unlock status, the unlock flip-flop (FF) is set to 1.

The unlock status is detected by the PLL unlock FF judgement register (refer to **14.5.5**).

The unlock FF is set at the cycle of reference frequency f_r selected at that time.

When the contents of the PLL unlock FF judgement register are read (by the PEEK instruction), the unlock FF is reset (Read & Reset).

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

14.5.5 Configuration and function of unlock FF judgement register

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
PLL unlock FF judgement register	0	0	0	PLLUL	05H	R & Reset

Detects status of unlock FF	
0	Unlock FF = 0 : PLL lock status
1	Unlock FF = 1 : PLL unlock status

Fixed “0”

On reset					
	Power-ON	0	0	0	Undefined
	Clock stop	↓	↓	↓	Retained
	CE	↓	↓	↓	Retained

This register is a read-only register and is reset when its contents are read to the window register by the “PEEK” instruction.

Because the unlock FF is set at the cycle of reference frequency f_r, the contents of the PLL unlock FF judgement register must be written to the window register at a cycle longer than the cycle $1/f_r$ of the reference frequency f_r.

14.6 PLL Disabled Status

The PLL frequency synthesizer stops operation (is disabled) while the CE pin (pin 13) is low.

When the PLL is disabled by the PLL reference mode select register, the PLL frequency synthesizer also stops operation.

Table 14-2 shows the operation of each block under each PLL disable condition.

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

Because the PLL reference mode select register and PLL mode select register are not initialized (but hold the previous status) on CE reset, they are restored to the original status when the CE pin has once gone low and then back high again after the PLL has been disabled.

To disable the PLL on CE reset, therefore, initialize these registers in the program.

The PLL is disabled at power-ON reset.

Table 14-2. Operation of Blocks Under PLL Disable Conditions

Condition / Blocks	CE Pin = Low Level (PLL disabled)	CE Pin = High Level	
		PLL reference mode select register = 1111B (PLL disabled)	PLL mode select register = 0000B (VCOH, VCOL disabled)
VCOL and VCOH pins	Internally pulled down	Internally pulled down	Internally pulled down
Programmable counter	Stops division	Stops division	Operates
Reference frequency generator	Stops output	Stops output	Operates
Phase comparator	Stops output	Stops output	Operates
Charge pump	Floats error out pins	Floats error out pins	Operates. However, usually outputs low level because there is no input.

14.7 Using PLL Frequency Synthesizer

To control the PLL frequency synthesizer, the following data are necessary.

(1) Division mode : direct division (MF), pulse swallow (HF, VHF)
(2) Pin used : VCOL, VCOH
(3) Reference frequency : f_r
(4) Division ratio : N

The following subsections 14.7.1 through 14.7.3 explain how to set the PLL data in each division mode (MF, HF, and VHF).

14.7.1 Direct division mode

(1) Selecting division mode

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

(2) Pin used

When the direct division mode is selected, the VCOL pin is enabled to operate.

(3) Setting reference frequency f_r

Set the reference frequency by using the PLL reference mode select register.

(4) Calculating division value N

Calculate as follows:

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

where,

f_{VCOL} : input frequency of VCOL pin

f_r : reference frequency

(5) Example of setting PLL data

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

Reception frequency : 1422 kHz (MW band)

Reference frequency : 9 kHz

Intermediate frequency: +450 kHz

Division value N:

$$N = \frac{f_{VCOL}}{f_r} = \frac{1422 + 450}{9} = 208 \text{ (decimal)}$$

$$= 0D0H \text{ (hexadecimal)}$$

Set data to the PLL data register (PLLR: peripheral address 41H), PLL mode select register (RF address 21H), and PLL reference mode select register (RF address 31H) as follows.

PLL data register (RLLR)															
0	0	0	0	1	1	0	1	0	0	0	0	Don't care			
0				D				0							

PLL mode select register				PLL reference mode select register			
0	0	0	1	1	1	0	1
MF				9 kHz			

14.7.2 Pulse swallow mode (HF)

(1) Selecting division mode

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

(2) Pin used

When the pulse swallow mode is selected, the VCOL pin is enabled to operate.

(3) Setting reference frequency f_r

Set the reference frequency by using the PLL reference mode select register.

(4) Calculating division value N

Calculate as follows:

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

where,

f_{VCOL} : input frequency of VCOL pin

f_r : reference frequency

(5) Example of setting PLL data

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

Reception frequency : 25.50 MHz (SW band)

Reference frequency : 5 kHz

Intermediate frequency: +450 kHz

Division value N:

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

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

Set data to the PLL data register (PLLR: peripheral address 41H), PLL mode select register (RF address 21H), and PLL reference mode select register (RF address 31H) as follows.

PLL data register (RLLR)			
0 0 0 1	0 1 0 0	0 1 0 0	0 1 1 0
1	4	4	6

PLL mode select register	PLL reference mode select register
0 0 1 1	0 0 1 0
HF	5 kHz

14.7.3 Pulse swallow mode (VHF)

(1) Selecting division mode

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

(2) Pin used

When the pulse swallow mode is selected, the VCOH pin is enabled to operate.

(3) Setting reference frequency f_r

Set the reference frequency by using the PLL reference mode select register.

(4) Calculating division value N

Calculate as follows:

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

where,

f_{VCOH}: input frequency of VCOH pin

f_r : reference frequency

(5) Example of setting PLL data

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

Reception frequency : 100.0 MHz (FM band)

Reference frequency : 25 kHz

Intermediate frequency: +10.7 MHz

Division value N:

$$N = \frac{f_{VCOH}}{f_r} = \frac{100.0 + 10.7}{0.025} = 4428 \text{ (decimal)}$$

$$= 114CH \text{ (hexadecimal)}$$

Set data to the PLL data register (PLLR: peripheral address 41H), PLL mode select register (RF address 21H), and PLL reference mode select register (RF address 31H) as follows.

PLL data register (RLLR)			
0 0 0 1	0 0 0 1	0 1 0 0	1 1 1 0
1	1	4	C

PLL mode select register	PLL reference mode select register
0 0 1 0	0 1 1 0
VHF	25 kHz

14.8 Status on Reset

14.8.1 On power-ON reset

The PLL is disabled on power-ON reset because the PLL reference mode select register is initialized to 1111B.

14.8.2 On execution of clock stop instruction

The PLL is disabled when the CE pin goes low.

14.8.3 On CE reset

(1) CE reset after execution of clock stop instruction

The PLL is disabled because the PLL reference mode select register is initialized to 1111B by the clock stop instruction.

(2) CE reset without clock stop instruction executed

Because the PLL reference mode select register retains the previous status, the previous status is restored as soon as the CE pin has gone high.

14.8.4 In halt status

The set status is retained if the CE pin is high.

15. GENERAL-PURPOSE PORT

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

15.1 Configuration and Classification of General-Purpose Ports

Figure 15-1 shows the block diagram of the general-purpose ports.

Table 15-1 classifies the general-purpose ports.

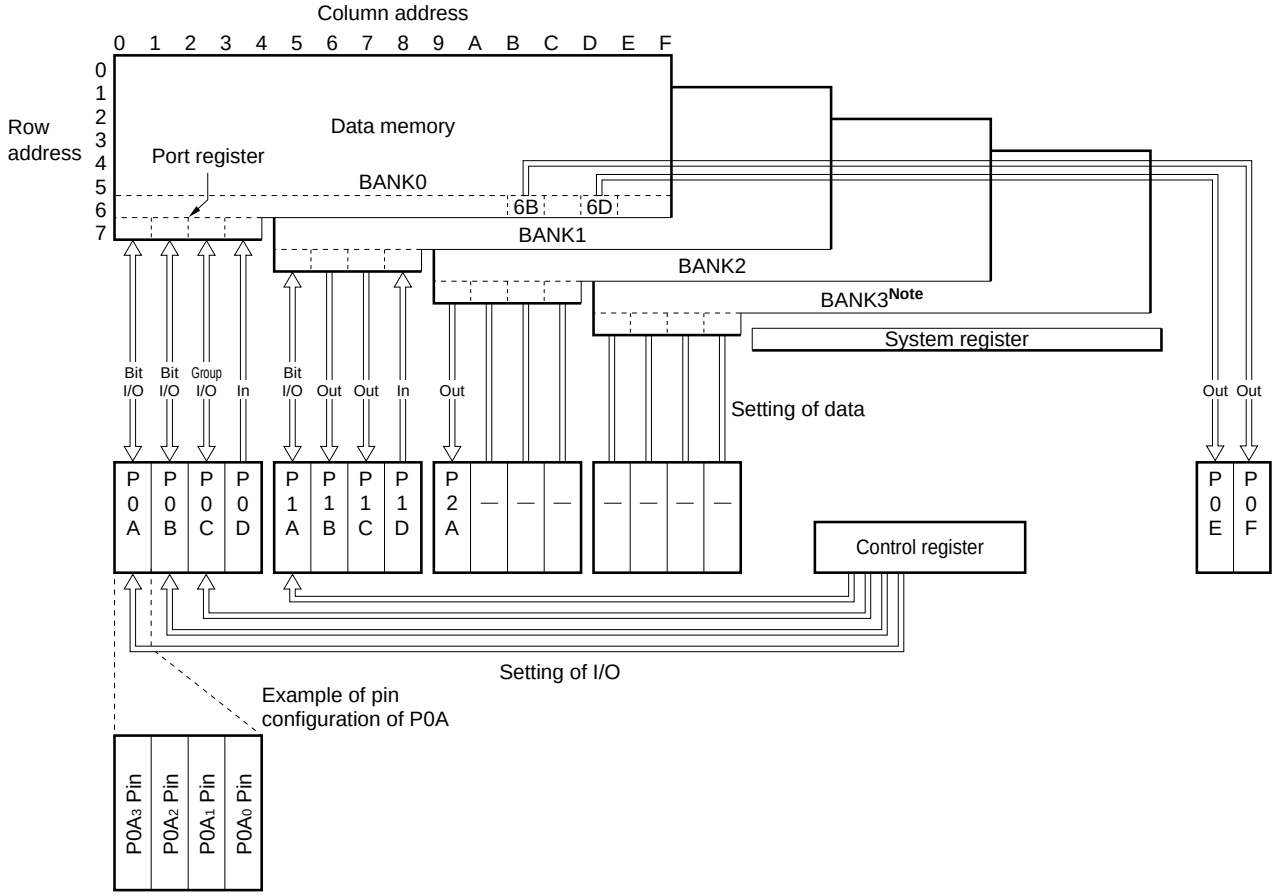
As shown in Figure 15-1, the general-purpose ports include ports 0A (P0A) through 2A (P2A) to which are set by addresses 70H through 73H (port registers) of each bank of the data memory, and ports 0E (P0E) and 0F (P0F) to which data are set by addresses 6BH and 6DH of bank 0 of the data memory.

Each port consists of general-purpose port pins (e.g., port 0A consists of $P0A_3$ through $P0A_0$ pins).

As shown in Table 15-1, the general-purpose ports are classified into input/output ports (I/O ports), input-only ports (input ports), and output-only ports (output ports).

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

Figure 15-1. Block Diagram of General-Purpose Port



Note BANK3 is not provided on the μPD17016.

Table 15-1. Classification of General-Purpose Ports

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

15.2 Functional Outline of General-Purpose Ports

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

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

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

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

P0A through P0D, P1A through P1D, and P2A are set in the general-purpose port mode on power-ON reset.

P0E and P0F are used as LCD segment signal output pins on power-ON reset. To use these ports as general-purpose output ports, the corresponding control registers must be set independently.

The following subsections 15.2.1 through 15.2.4 explain the port registers, the function of the port group register, and the functional outline of each port.

15.2.1 General-purpose port data register (port register)

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

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

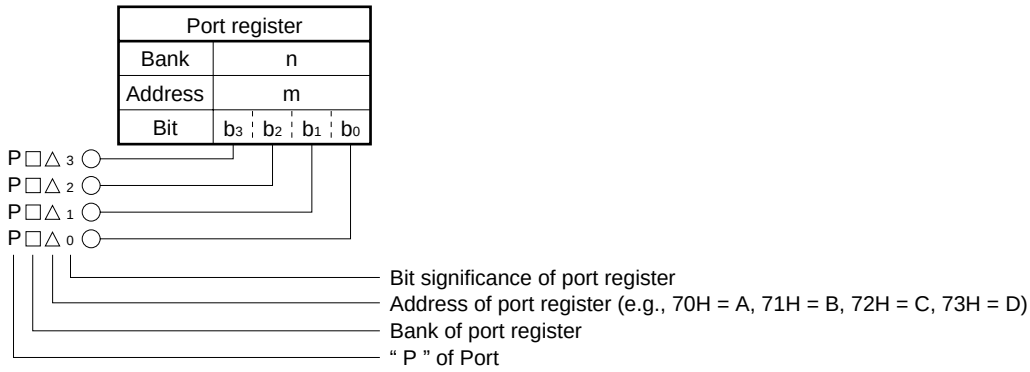
Figure 15-2 shows the relation between a port register and the corresponding port pins.

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

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

Table 15-2 shows the relation between each port (each pin) and port register.

Figure 15-2. Relation between Port Register and Pins



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

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

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

P0E and P0F are multiplexed with LCD segment signal output pins. The port registers of P0E and P0F are also multiplexed with LCD segment registers.

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

15.2.2 General-purpose I/O ports (P0A, P0B, P0C, and P1A)

P0A, P0B, P0C, and P1A can be set in the input or output mode by the P0A bit I/O select register (RF address 37H), P0B bit I/O select register (RF address 36H), P0C group I/O select register (RF address 27H), P1A bit I/O select register (RF address 35H), respectively.

The input/output data of the P0A, P0B, P0C, and P1A are set by port registers P0A (address 70H of BANK0), P0B (address 71H of BANK0), P0C (address 72H of BANK0), and P1A (address 70H of BANK1), respectively.

Refer to **Table 15-2**.

For details, refer to **15.3**.

15.2.3 General-purpose input ports (P0D and P1D)

The input data of P0D and P1D are read by port registers P0D (address 73H of BANK0) and P1D (address 73H of BANK1), respectively.

Refer to **Table 15-2**.

For details, refer to **15.4**.

15.2.4 General-purpose output ports (P1B, P1C, P2A, P0E, and P0F)

(1) P1B, P1C, and P2A

The output data of P1B, P1C, and P2A are set by port registers P1B (address 71H of BANK1), P1C (address 72H of BANK1), and P2A (address 70H of BANK2).

Refer to **Table 15-2**.

For details, refer to **15.5**.

(2) P0E and P0F

P0E and P0F usually operate as LCD segment signal output pins. These ports can be set in the output port mode by using a software macro supplied with the device file.

To set the output data of P0E and P0F, P0E register (multiplexed with LCD segment register LCDD13, address 6DH of BANK0) and P0F register (multiplexed with LCDD11, address 6BH of BANK0) are used, respectively.

Refer to **Table 15-2**.

For details, refer to **15.6**.

Table 15-2. Relation between Each Port (Pin) and Port Register (1/2)

Port	Pin			Setting of Data					
	No.	Symbol	I/O	Port register (data memory)					Remark
				Bank	Address	Symbol	Bit symbol (reserved word)		
Port 0A (P0A)	3	$P0A_3$	I/O (bit I/O)	BANK0	70H	P0A	b_3	P0A3	
	4	$P0A_2$					b_2	P0A2	
	5	$P0A_1$					b_1	P0A1	
	6	$P0A_0$					b_0	P0A0	
Port 0B (P0B)	7	$P0B_3$	I/O (bit I/O)		71H	P0B	b_3	P0B3	
	8	$P0B_2$					b_2	P0B2	
	9	$P0B_1$					b_1	P0B1	
	10	$P0B_0$					b_0	P0B0	
Port 0C (P0C)	79	$P0C_3$	I/O (group I/O)		72H	P0C	b_3	P0C3	
	80	$P0C_2$					b_2	P0C2	
	1	$P0C_1$					b_1	P0C1	
	2	$P0C_0$					b_0	P0C0	
Poer 0D (P0D)	75	$P0D_3$	Input		73H	P0D	b_3	P0D3	
	76	$P0D_2$					b_2	P0D2	
	77	$P0D_1$					b_1	P0D1	
	78	$P0D_0$					b_0	P0D0	
Port 1A (P1A)	14	$P1A_3$	I/O (bit I/O)	BANK1	70H	P1A	b_3	P1A3	
	15	$P1A_2$					b_2	P1A2	
	16	$P1A_1$					b_1	P1A1	
	17	$P1A_0$					b_0	P1A0	
Port 1B (P1B)	18	$P1B_3$	Output		71H	P1B	b_3	P1B3	
	19	$P1B_2$					b_2	P1B2	
	20	$P1B_1$					b_1	P1B1	
	21	$P1B_0$					b_0	P1B0	
Port 1C (P1C)	22	$P1C_3$	Output		72H	P1C	b_3	P1C3	
	23	$P1C_2$					b_2	P1C2	
	24	$P1C_1$					b_1	P1C1	
	25	$P1C_0$					b_0	P1C0	
Port 1D (P1D)	26	$P1D_3$	Input		73H	P1D	b_3	P1D3	
	27	$P1D_2$					b_2	P1D2	
	28	$P1D_1$					b_1	P1D1	
	29	$P1D_0$					b_0	P1D0	
Port 2A (P2A)	No target pins			BANK2	70H	P2A	b_3	P2A3	Nothing is allocated. Cannot be used as data memory.
							b_2	P2A2	
							b_1	P2A1	
	42	$P2A_0$	Output				b_0	P2A0	

Table 15-2. Relation between Each Port (Pin) and Port Register (2/2)

Port	Pin			Setting of Data					
	No.	Symbol	I/O	Port register (data memory)					Remark
				Bank	Address	Symbol	Bit symbol (reserved word)		
				BANK2	71H	–	b_3	–	Nothing is allocated. Cannot be used as data memory.
							b_2		
							b_1		
							b_0		
					72H	–	b_3	–	
							b_2		
							b_1		
							b_0		
					73H	–	b_3	–	
							b_2		
							b_1		
							b_0		
				BANK3[Note]	70H	–	b_3	–	Nothing is allocated. Cannot be used as data memory.
							b_2		
							b_1		
							b_0		
					71H	–	b_3	–	
							b_2		
							b_1		
							b_0		
					72H	–	b_3	–	
							b_2		
							b_1		
							b_0		
					73H	–	b_3	–	
							b_2		
							b_1		
							b_0		
Port 0E (P0E)	49	$P0E_3$	Output	BANK0	6BH (multiplexed with LCDD11)	P0E	b_3	P0E3	
	50	$P0E_2$					b_2	P0E2	
	51	$P0E_1$					b_1	P0E1	
	52	$P0E_0$					b_0	P0E0	
Port 0F (P0F)	45	$P0F_3$	Output		6DH (multiplexed with LCDD13)	P0F	b_3	P0F3	
	46	$P0F_2$					b_2	P0F2	
	47	$P0F_1$					b_1	P0F1	
	48	$P0F_0$					b_0	P0F0	

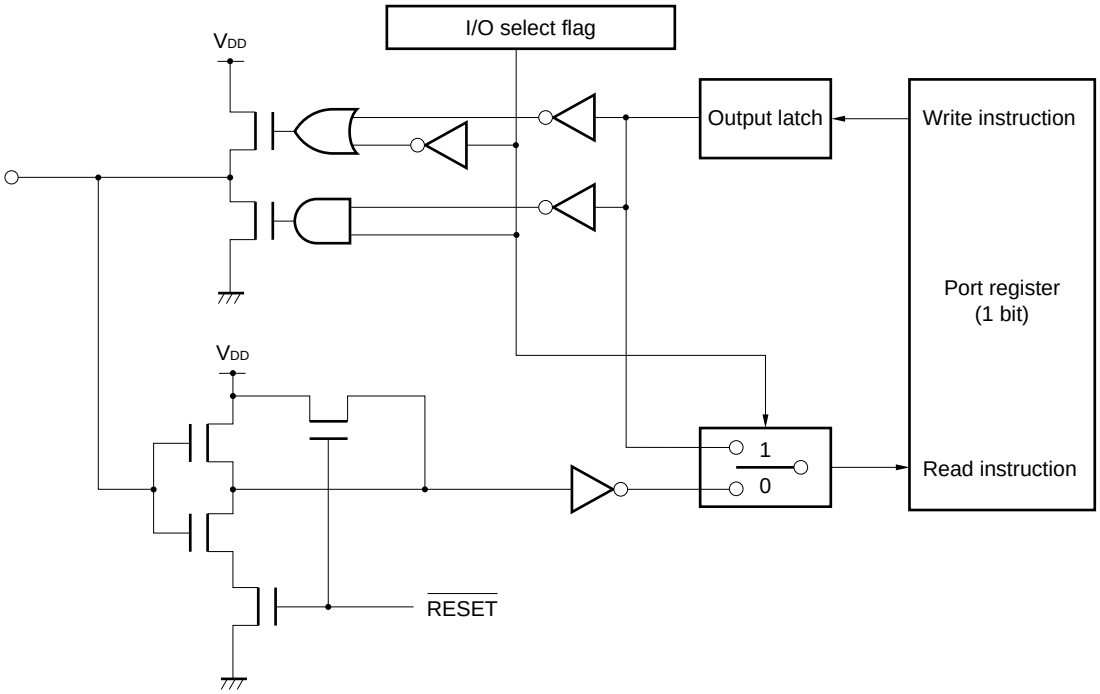
Note BANK3 is not provided on the μPD17016.

15.3 General-Purpose I/O Ports (P0A, P0B, P0C, and P1A)

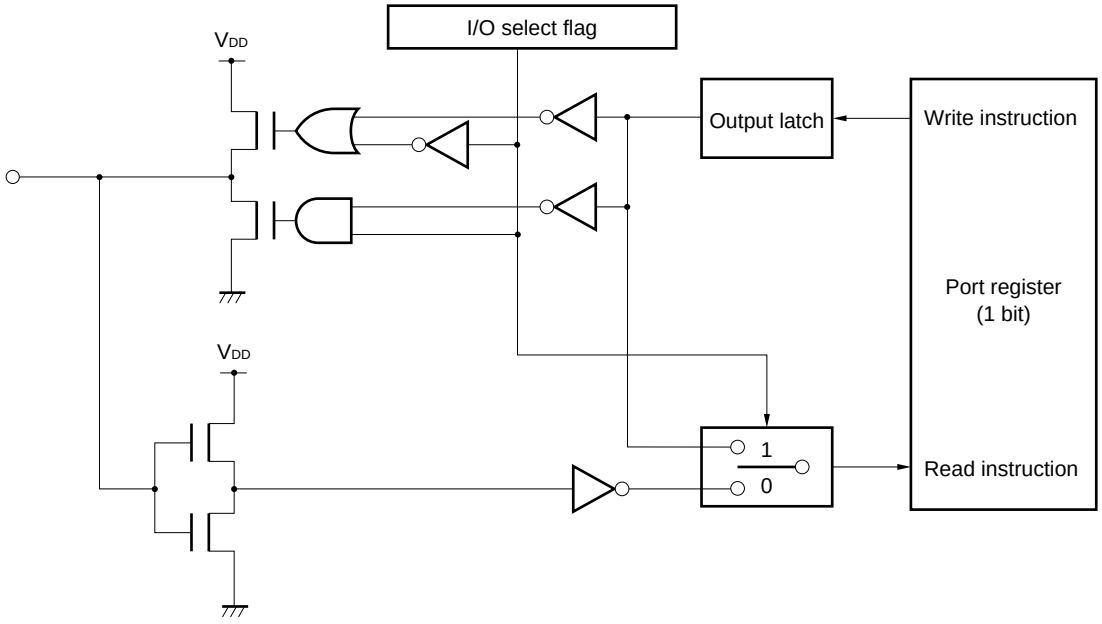
15.3.1 Configuration of I/O ports

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

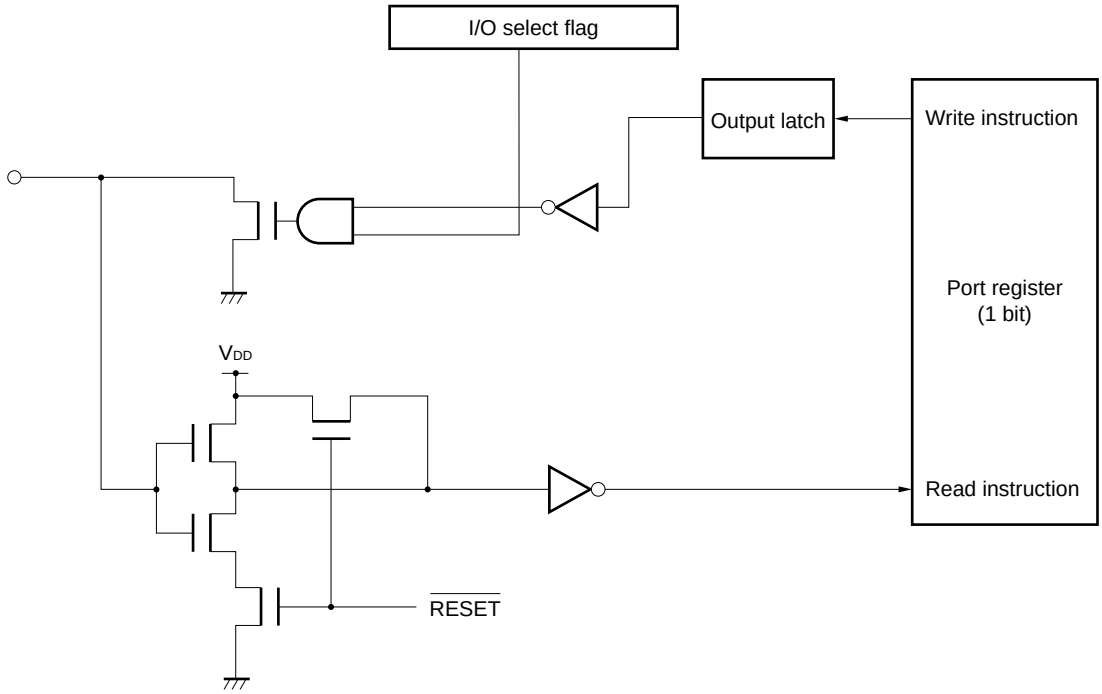
(1) P0A (P0A$_1$ and P0A$_0$ pins),
P0B (P0B$_3$, P0B$_2$, P0B$_1$, and P0B$_0$ pins),
P1A (P1A$_3$, P1A$_2$, P1A$_1$, and P1A$_0$ pins)



(2) P0C (P0C$_3$, P0C$_2$, P0C$_1$, and P0C$_0$ pins)



(3) P0A (P0A3 and P0A2 pins)



15.3.2 Using I/O ports

The I/O ports are set in the input or output mode by I/O select registers P0A P0B, P0C, and P1A of the control registers.

The bit I/O ports (P0A, P0B, and P1A) can be set in the input or output mode in 1-bit units, and group I/O port (P0C) can be set in the input or output mode in 4-bit units.

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

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

15.3.4 and 15.3.5 explain how to use the input and output ports.

15.3.6 explains the points to be noted in using the I/O ports.

15.3.3 Port 0A bit I/O select register
Port 0B bit I/O select register
Port 1A bit I/O select register
Port 0C group I/O select register

The port 0A bit I/O, port 0B bit I/O, port 1A bit I/O, and port 0C group I/O select registers sets each pin of the P0A, P0B, P1A, and P0C in the input or output mode.

The configuration and functions of these registers are shown below.

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

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Port 0A bit I/O select register	P0ABIO3	P0ABIO2	P0ABIO1	P0ABIO0	37H	R/W

Sets input or output mode of port	
0	Sets $P0A_0$ pin in input mode.
1	Sets $P0A_0$ pin in output mode.

Sets input or output mode of port	
0	Sets $P0A_1$ pin in input mode.
1	Sets $P0A_1$ pin in output mode.

Sets input or output mode of port	
0	Sets $P0A_2$ pin in input mode.
1	Sets $P0A_2$ pin in output mode.

Sets input or output mode of port	
0	Sets $P0A_3$ pin in input mode.
1	Sets $P0A_3$ pin in output mode.

On reset	Power-ON	0	0	0	0
	Clock stop	0	0	0	0
	CE	0	0	0	0

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

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Port 0B bit I/O select register	P0BBIO3	P0BBIO2	P0BBIO1	P0BBIO0	36H	R/W

P0BBIO0	Sets input or output mode of port
0	Sets $P0B_0$ pin in input mode.
1	Sets $P0B_0$ pin in output mode.

P0BBIO1	Sets input or output mode of port
0	Sets $P0B_1$ pin in input mode.
1	Sets $P0B_1$ pin in output mode.

P0BBIO2	Sets input or output mode of port
0	Sets $P0B_2$ pin in input mode.
1	Sets $P0B_2$ pin in output mode.

P0BBIO3	Sets input or output mode of port
0	Sets $P0B_3$ pin in input mode.
1	Sets $P0B_3$ pin in output mode.

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop	0	0	0	0
	CE	0	0	0	0

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

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Port 1A bit I/O select register	P1ABIO3	P1ABIO2	P1ABIO1	P1ABIO0	35H	R/W

P1ABIO0	Sets input or output mode of port
0	Sets $P1A_0$ pin in input mode.
1	Sets $P1A_0$ pin in output mode.

P1ABIO1	Sets input or output mode of port
0	Sets $P1A_1$ pin in input mode.
1	Sets $P1A_1$ pin in output mode.

P1ABIO2	Sets input or output mode of port
0	Sets $P1A_2$ pin in input mode.
1	Sets $P1A_2$ pin in output mode.

P1ABIO3	Sets input or output mode of port
0	Sets $P1A_3$ pin in input mode.
1	Sets $P1A_3$ pin in output mode.

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop	0	0	0	0
	CE	0	0	0	0

(4) Port 0C group I/O select register

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Port 0C group I/O select register	0	0	0	P0CGIO	27H	R/W

Sets input or output mode of port	
0	Sets $P0C_3$-$P0C_0$ pin in input mode.
1	Sets $P0C_3$-$P0C_0$ pin in output mode.

Fixed to “0”

On reset	Power-ON	0	0	0	0
	Clock stop				0
	CE				0

15.3.4 Using I/O ports (P0A, P0B, P0C, and P1A) as input ports

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

Note that P0C can be set in the input or output mode in 4-bit units only.

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

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

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

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

15.3.5 Using I/O ports (P0A, P0B, P0C, and P1A) as output ports

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

Note that P0C can be set in the input or output mode in 4-bit units only.

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

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

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

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

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

Note, however, that the statuses of the $P0A_3$ and $P0A_2$ pins are read as is, and the read contents may be different from the contents of the output latch.

Refer to **15.3.6**.

15.3.6 Notes on using I/O ports ($P0A_3$ and $P0A_2$ pins)

When using the $P0A_3$ and $P0A_2$ pins as output pins, the contents of the output latch may be rewritten, as shown in the example below.

Example To specify $P0A_3$ and $P0A_2$ pins as output port pins

```
 INITFLG P0ABI03, P0ABI02, NOT P0ABI01, NOT P0ABI00
                 ; Sets P0A3 and P0A2 pins in output mode
 INITFLG P0A3, P0A2, NOT P0A1, NOT P0A0
                 ; Outputs high level to P0A3 and P0A2 pins
;<1>
 CLR1   P0A3     ; Outputs low level to P0A3 pin
 Macro expansion
 AND    . MF. P0A3 SHR 4, # .DF. (NOT P0A3 AND 0FH)
```

If the $P0A_2$ pin happens to be lowered by an external device when the instruction in <1> above is executed, the contents of the output latch of the $P0A_2$ pin are rewritten to "0" by the "CLR1" instruction.

In other words, if an operation instruction (such as "ADD" and "OR") to the P0A port register when the $P0A_3$ or $P0A_2$ pin is set in the output mode, the contents of the output latch are rewritten to the current level of the pin, regardless of the previous status of the pin.

15.3.7 Status of I/O ports (P0A, P0B, P0C, and P1A) on reset

(1) On power-ON reset

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

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

(2) On CE reset

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

The contents of the output latch are retained.

(3) On execution of clock stop instruction

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

The contents of the output latch are retained.

The I/O ports other than P0C prevents an increase in the current dissipation due to the noise of the input buffer by using the $\overline{\text{RESET}}$ signal when the clock stop instruction is executed, as explained in 15.3.1.

If P0C is floated on execution of the clock stop instruction, the current dissipation may increase due to external noise. Externally pull this port down or up as necessary.

(4) In halt status

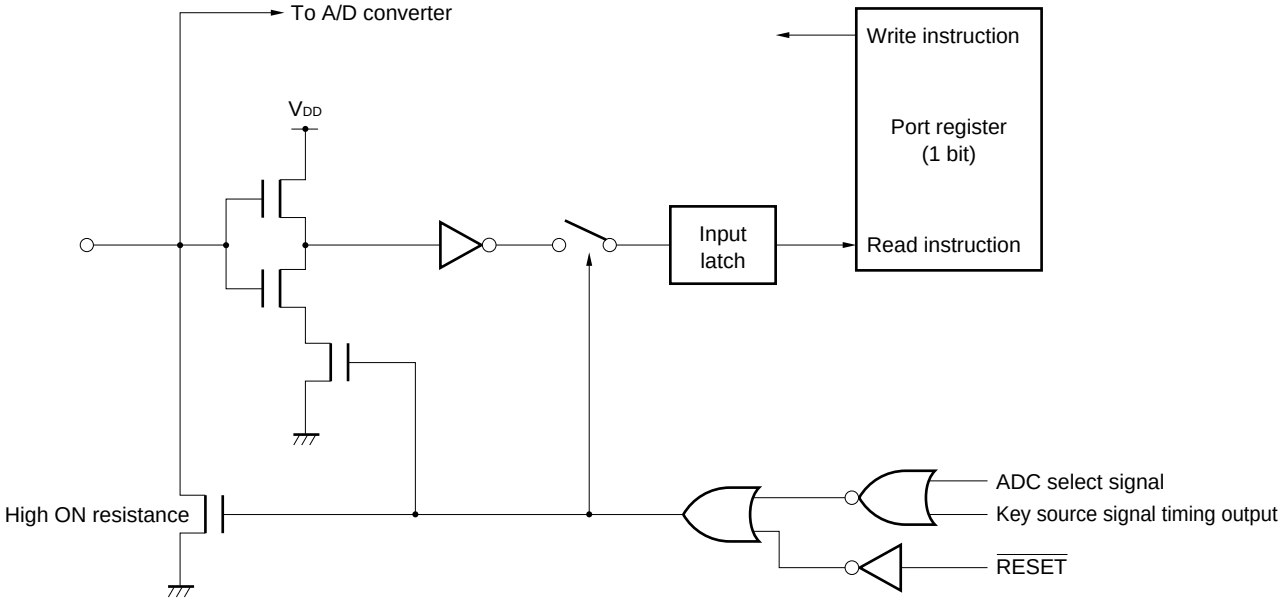
The previous status is retained.

15.4 General-Purpose Input Ports (P0D and P1D)

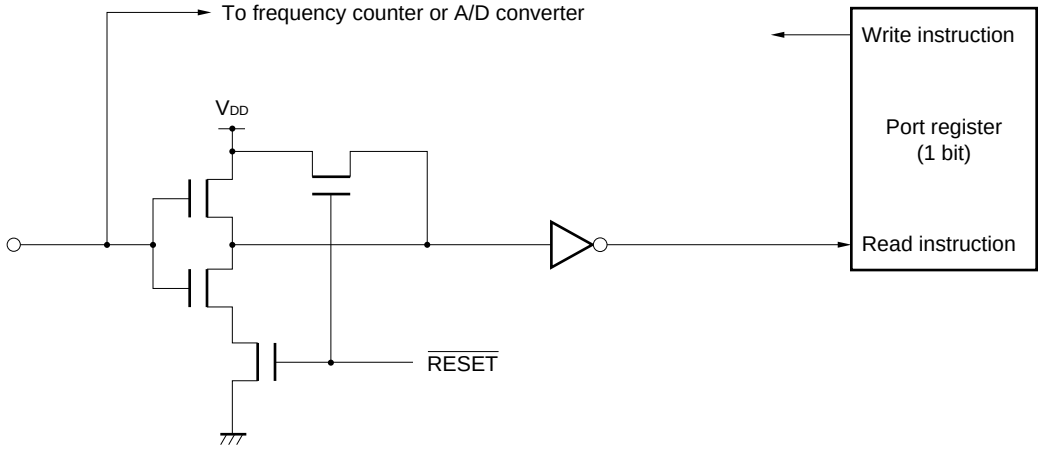
15.4.1 Configuration of input ports

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

(1) P0D ($P0D_3$, $P0D_2$, $P0D_1$, and $P0D_0$ pins)



(2) P1D ($P1D_3$, $P1D_2$, $P1D_1$, and $P1D_0$ pins)



15.4.2 Using input ports (P0D and P1D)

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

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

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

15.4.3 Notes on using input port (P0D)

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

15.4.4 Status of input ports (P0D and P1D) on reset

(1) On power-ON reset

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

(2) On CE reset

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

(3) On execution of clock stop instruction

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

Because the $\overline{\text{RESET}}$ signal is output when the clock stop instruction is executed, the P1D prevents an increase in the current dissipation due to the noise of the input buffer.

The P0D is internally pulled down.

(4) In halt status

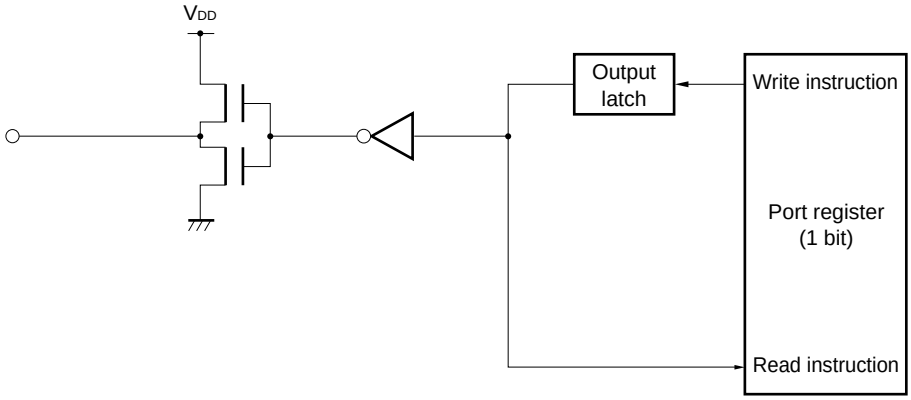
The previous status is retained.

15.5 General-Purpose Output Ports (P1B, P1C, and P2A)

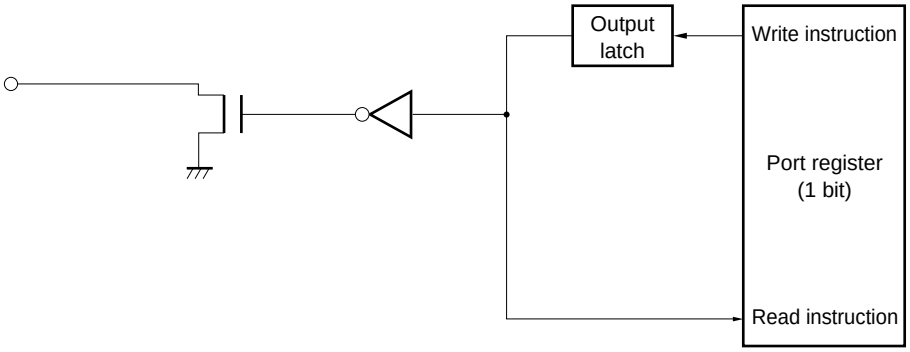
15.5.1 Configuration of output ports (P1B, P1C, and P2A)

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

(1) P1B ($P1B_0$ pin),
P1C ($P1C_3$, $P1C_2$, $P1C_1$, and $P1C_0$ pins),
P2A ($P2A_0$ pin)



(2) P1B ($P1B_3$, $P1B_2$, and $P1B_1$ pins)



15.5.2 Example of using output ports (P1B, P1C, and P2A)

The output ports output the contents of the output latches.

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

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

Note, however, that the $P1B_3$, $P1B_2$, and $P1B_1$ pins are N-ch open-drain output pins and are floated when a high level is output.

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

15.5.3 Status of output ports (P1B, P1C, and P2A) on reset

(1) On power-ON reset

The contents of the output latch are output.

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

(2) On CE reset

The contents of the output latch are output.

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

(3) On execution of clock stop instruction

The contents of the output latch are output.

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

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

(4) In halt status

The contents of the output latch are output.

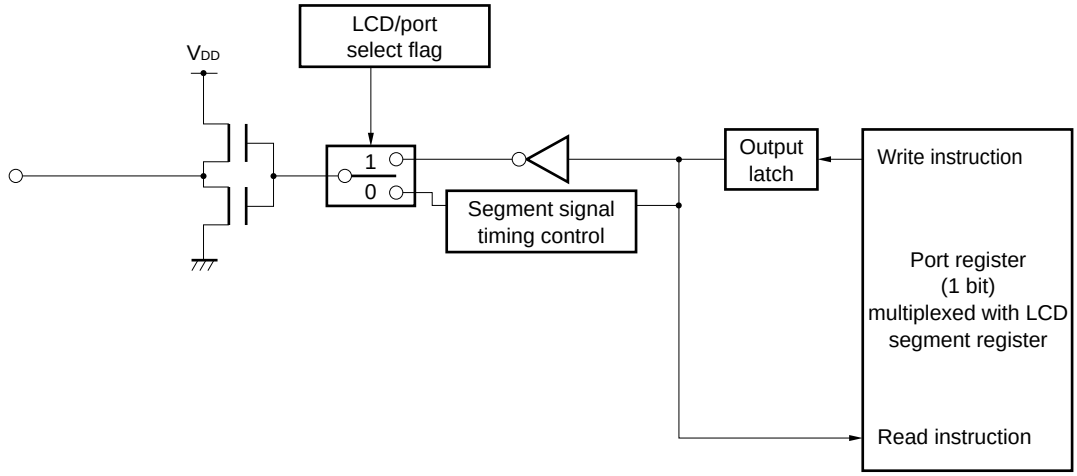
The contents of the output latch are retained and the output data are not changed in the halt status.

15.6 General-Purpose Output Ports (P0E and P0F)

15.6.1 Configuration of output ports (P0E and P0F)

The configuration of the output ports is shown below.

P0E (P0E$_3$, P0E$_2$, P0E$_1$, and P0E$_0$ pins)
P0F (P0F$_3$, P0F$_2$, P0F$_1$, and P0F$_0$ pins)



15.6.2 Example of using output ports (P0E and P0F)

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

To use it as an output port pin, select the port to be used by a software macro supplied with the device file.

The port to be used can be selected independently.

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

Table 15-3 shows the software macros that set the ports (P0E and P0F).

Table 15-3. Software Macros Setting Ports (P0E and P0F)

Function	Macro Format
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as general-purpose output port	SET1_P0EON
Uses $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port	SET1_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port	SET2_P0EON_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as LCD segment signal output pins	CLR1_P0EON
Uses $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins	CLR1_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins	CLR2_P0EON_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as LCD segment signal output pins and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port	INIT_NOT_P0EON_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as general-purpose output port and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins	INIT_P0EON_NOT_P0FON

Caution If the above macros are used, the contents of the window register are destroyed.
If the above macros are immediately follow an embedded macro instruction, an "object error" occurs when the source file is assembled and loaded to the in-circuit emulator. If an embedded macro is used before the above macros, insert a comment statement in between them.

15.6.3 Setting data to P0E and P0F

Output data are set to the P0E and P0F by executing an instruction, such as "MOV", that writes data to the port registers corresponding to the ports.

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

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

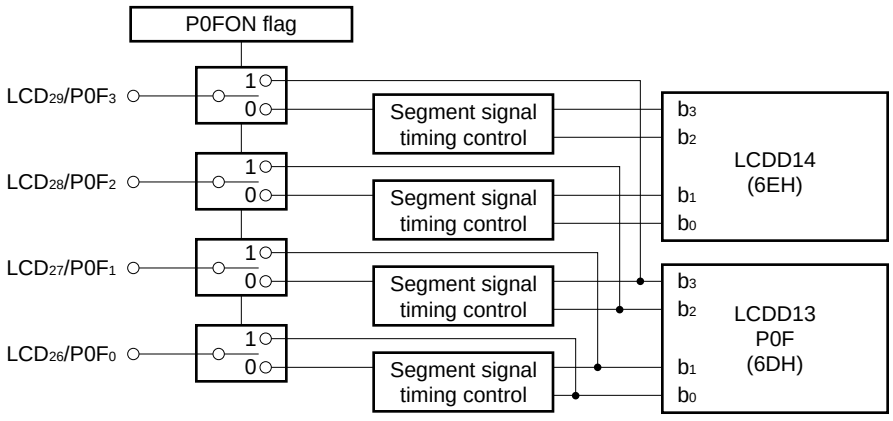
Figure 15-3 shows the relation between the P0F port register and LCD segment register.

As shown in this figure, LCD segment register LCDD14 can be used as a general-purpose data memory area when the P0F is used.

The same applies to the P0E.

Refer to **Figure 21-6. Relation among LCD Display Dots, Ports 0E and 0F, Key Source Output, and Each Data Setting Register** in **21. LCD CONTROLLER/DRIVER**.

Figure 15-3. Relation between P0F Port Register and LCD Segment Register



15.6.4 Status of output ports (P0E and P0F) on reset

(1) On power-ON reset

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

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

(2) On CE reset

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

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

(3) On execution of clock stop instruction

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

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

(4) In halt status

The contents of the output latch are output.

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

16. A/D CONVERTER (ADC)

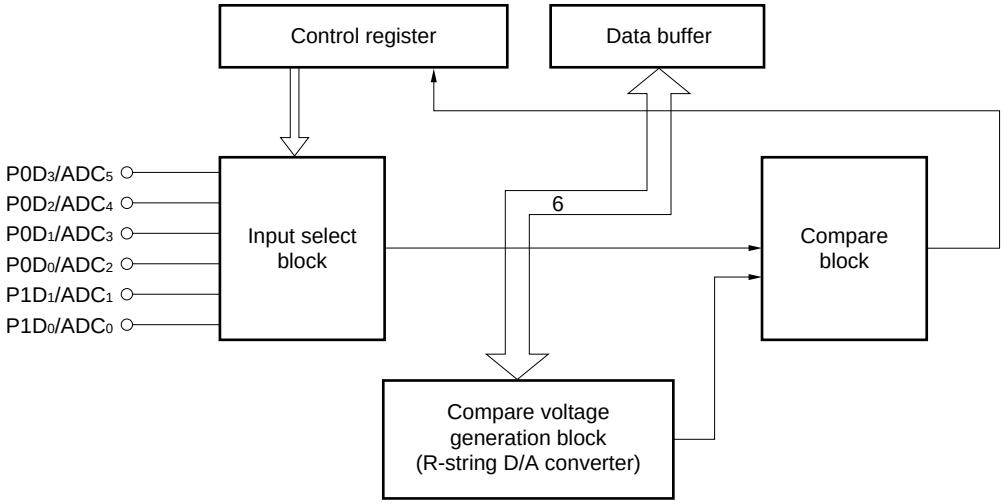
The A/D converter is used to input an external analog signal as a digital signal.

16.1 Configuration of A/D Converter

Figure 16-1 shows the block diagram of the A/D converter.

As shown in the figure, the A/D converter consists of an input select block, compare voltage generation block, and compare block.

Figure 16-1. Block Diagram of A/D Converter



16.2 Functional Outline of A/D Converter

The A/D converter compares the voltage input to the $P0D_3/ADC_5$ to $P1D_0/ADC_0$ pin with an internal compare voltage, and outputs the result of the comparison as “True (1)” or “False (0)”.

By judging this comparison result by software, the A/D converter can be used as a successive approximation converter.

The following subsections 16.2.1 through 16.2.3 outline the function of each block of the A/D converter.

For details, refer to **16.3** through **16.5**.

16.2.1 Input select block

This block selects which of the $P0D_3/ADC_5$ through $P1D_0/ADC_0$ pins is to be used.

The pin is selected by the A/D converter channel select register (RF address 14H).

Only one pin can be selected at a time.

Refer to **16.3**.

16.2.2 Compare voltage generation block

This block generates a compare voltage against which an input voltage is to be compared.

The compare voltage is generated by an R-string D/A converter.

Refer to **16.4**.

16.2.3 Compare block

This block compares an input voltage with an internal compare voltage.

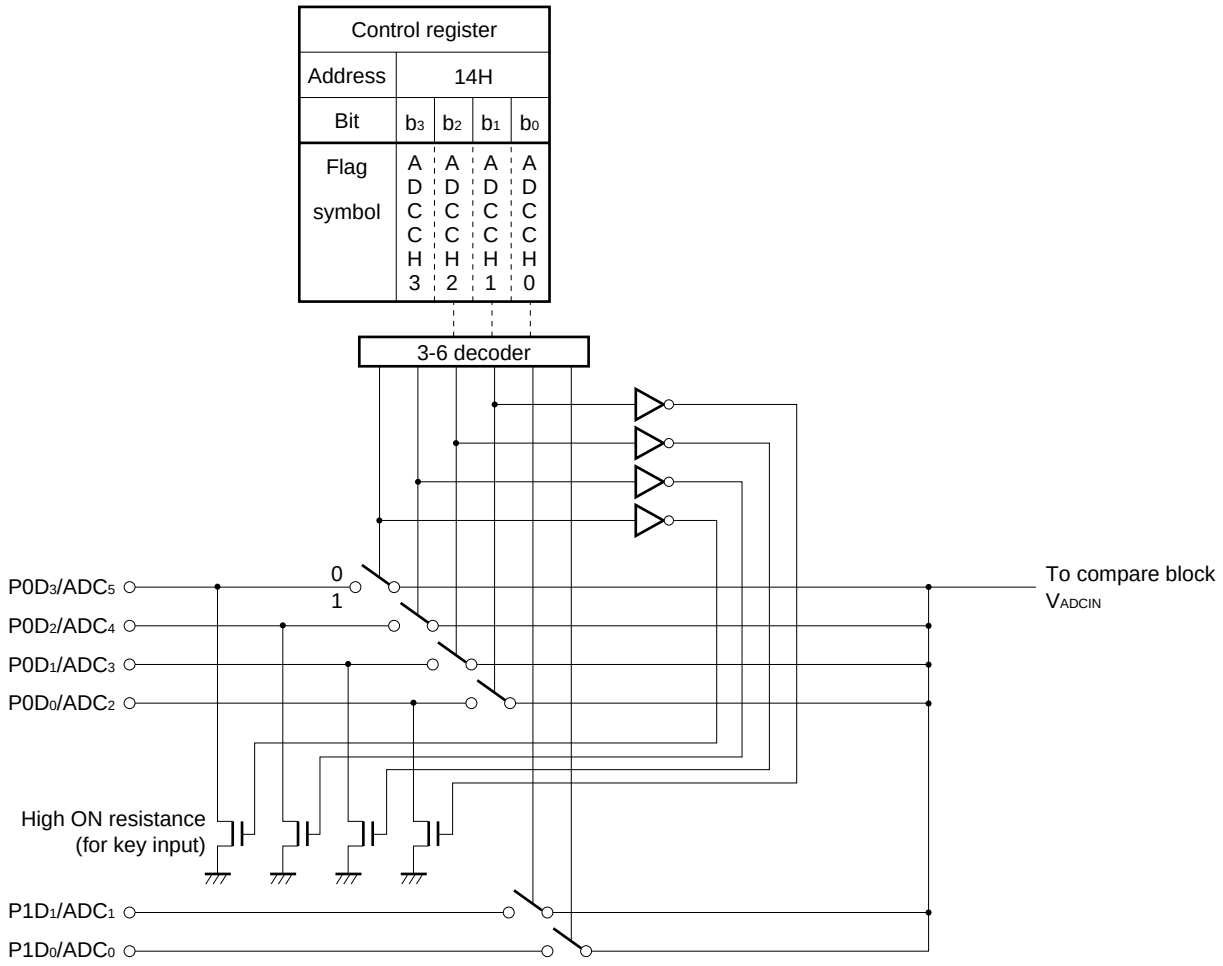
The result of the comparison is detected by the A/D converter compare judge register (RF address 06H). Refer to **16.5**.

16.3 Input Select Block

16.3.1 Configuration of input select block

Figure 16-2 shows the configuration of the input select block.

Figure 16-2. Configuration of Input Select Block



16.3.2 Function of input select block

The input select block selects a pin to be used by using the A/D converter channel select register.

Only one pin can be used as an A/D converter pin at a time.

The pin not selected for the A/D converter can be used as general-purpose input port.

Although port 0D ($P0D_3$/ADC_5 through $P0D_0$/ADC_2 pins) is internally connected to a pull-down resistor, the pin selected by the A/D converter channel select register (refer to **16.3.3**) is disconnected from the pull-down resistor as soon as it has been selected.

The pins not selected remain connected to the pull-down resistor.

16.3.3 Configuration and function of A/D converter channel select register

The A/D converter channel select register selects a pin to be used for the A/D converter.

The configuration and function of this register are shown below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
A/D converter channel select register	ADCCH3	ADCCH2	ADCCH1	ADCCH0	14H	R/W

			Selects pin to be used for A/D converter
0	0	0	$P1D_0$/ADC_0 pin
0	0	1	$P1D_1$/ADC_1 pin
0	1	0	$P0D_0$/ADC_2 pin
0	1	1	$P0D_1$/ADC_3 pin
1	0	0	$P0D_2$/ADC_4 pin
1	0	1	$P0D_3$/ADC_5 pin
1	1	0	A/D converter is not used (general-purpose input port)
1	1	1	A/D converter is not used (general-purpose input port)

Fixed to “0”

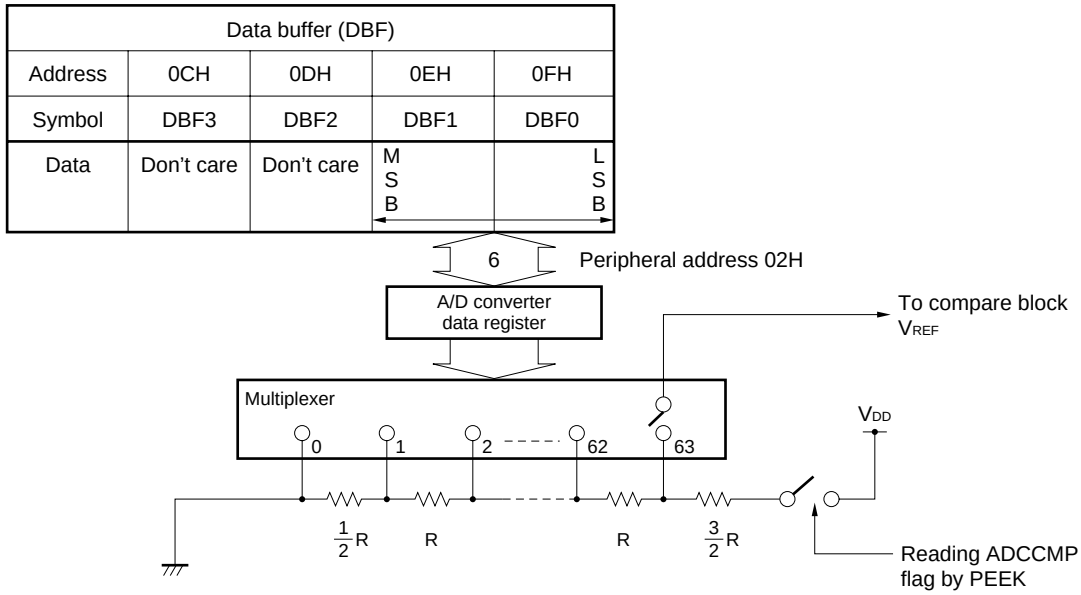
On reset	Power-ON	0	1	1	1
	Clock stop	↓	1	1	1
	CE	↓	1	1	1

16.4 Compare Voltage Generation Block

16.4.1 Configuration of compare voltage generator block

Figure 16-3 shows the configuration of the compare voltage generator block.

Figure 16-3. Configuration of Compare Voltage Generator Block



16.4.2 Function of compare voltage generator block

The compare voltage generator block switches the multiplexer according to the 6-bit data set to the A/D converter data register (ADCR: peripheral address 02H) and generates a compare voltage.

This block is therefore an R-string D/A converter.

The compare voltage can be adjusted in 64 steps by the R string (resistance division).

The voltage supplied to the R string is the same as the supply voltage V_{DD} of the device.

The voltage is supplied to the resistor of the R string only when the A/D converter compare judge register, which is to be explained later, is detected.

The compare voltage is compared with a voltage input to the compare block from a selected pin.

16.4.3 explains the configuration and function of the A/D converter data register.

Table 16-1 lists the compare voltages.

16.4.3 Configuration and function of A/D converter data register (ADCR)

The A/D converter data register sets the compare voltage of the A/D converter.

Because this register is 6 bits long, the low-order 6 bits of the data buffer are valid.

Name	Data buffer															
Symbol	DBF3				DBF2				DBF1				DBF0			
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	Don't care				Don't care				Transfer data							

8

GET can be executed

PUT can be executed

Peripheral register											
Name	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Symbol	Peripheral address	Peripheral hardware
A/D converter data register	0	0	Valid data						ADCR	02H	A/D converter

Sets compare voltage V_{REF} of A/D converter	
0	V_{REF} = 0 V
1	$V_{REF} = \frac{x - 0.5}{64} \times V_{DD}$ V
x	
63	

Fixed to "0"

Table 16-1. Set Values of A/D Converter Data Register and Compare Voltages

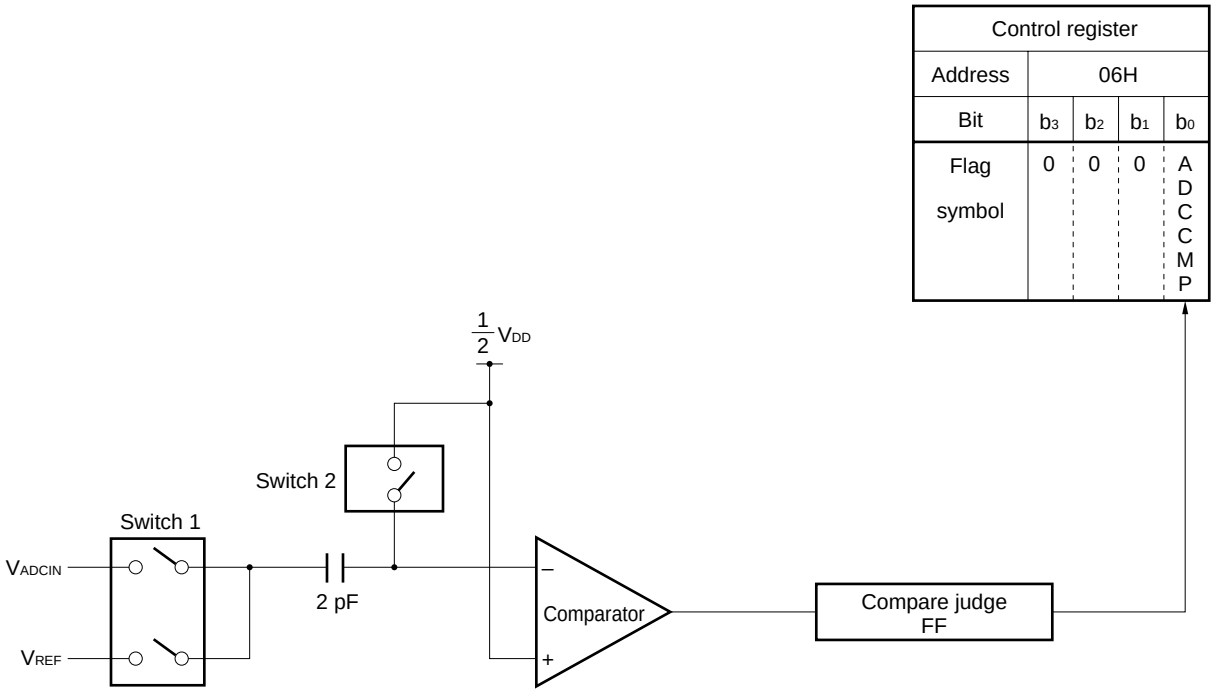
Set Data of ADCR		Compare Voltage		Set Data of ADCR		Compare Voltage	
Decimal (DEC)	Hexadecimal (HEX)	Logic voltage Unit: × V_{DD} V	At V_{DD} = 5 V Unit: V	Decimal (DEC)	Hexadecimal (HEX)	Logic voltage Unit: × V_{DD} V	At V_{DD} = 5 V Unit: V
0	00H	0	0	32	20H	31.5/64	2.461
1	01H	0.5/64	0.039	33	21H	32.5/64	2.539
2	02H	1.5/64	0.117	34	22H	33.5/64	2.617
3	03H	2.5/64	0.195	35	23H	34.5/64	2.695
4	04H	3.5/64	0.273	36	24H	35.5/64	2.773
5	05H	4.5/64	0.352	37	25H	36.5/64	2.852
6	06H	5.5/64	0.430	38	26H	37.5/64	2.930
7	07H	6.5/64	0.508	39	27H	38.5/64	3.008
8	08H	7.5/64	0.586	40	28H	39.5/64	3.086
9	09H	8.5/64	0.664	41	29H	40.5/64	3.164
10	0AH	9.5/64	0.742	42	2AH	41.5/64	3.242
11	0BH	10.5/64	0.820	43	2BH	42.5/64	3.320
12	0CH	11.5/64	0.898	44	2CH	43.5/64	3.398
13	0DH	12.5/64	0.977	45	2DH	44.5/64	3.477
14	0EH	13.5/64	1.055	46	2EH	45.5/64	3.555
15	0FH	14.5/64	1.133	47	2FH	46.5/64	3.633
16	10H	15.5/64	1.211	48	30H	47.5/64	3.711
17	11H	16.5/64	1.289	49	31H	48.5/64	3.789
18	12H	17.5/64	1.367	50	32H	49.5/64	3.867
19	13H	18.5/64	1.445	51	33H	50.5/64	3.945
20	14H	19.5/64	1.523	52	34H	51.5/64	4.023
21	15H	20.5/64	1.602	53	35H	52.5/64	4.102
22	16H	21.5/64	1.680	54	36H	53.5/64	4.180
23	17H	22.5/64	1.758	55	37H	54.5/64	4.258
24	18H	23.5/64	1.836	56	38H	55.5/64	4.336
25	19H	24.5/64	1.914	57	39H	56.5/64	4.414
26	1AH	25.5/64	1.992	58	3AH	57.5/64	4.492
27	1BH	26.5/64	2.070	59	3BH	58.5/64	4.570
28	1CH	27.5/64	2.148	60	3CH	59.5/64	4.648
29	1DH	28.5/64	2.227	61	3DH	60.5/64	4.727
30	1EH	29.5/64	2.305	62	3EH	61.5/64	4.805
31	1FH	30.5/64	2.383	63	3FH	62.5/64	4.883

16.5 Compare Block

16.5.1 Configuration of compare block

Figure 16-4 shows the configuration of the compare block.

Figure 16-4. Configuration of Compare Block



16.5.2 Function of compare block

The compare block compares the voltage V_{ADCIN} input from a pin with the internal compare voltage V_{REF} by using a comparator and outputs the result of the comparison to the compare judge FF.

The compare judge FF can be detected by reading the ADCCMP flag of the A/D converter compare judge register.

The ADCCMP flag is set when $V_{ADCIN} > V_{REF}$ and reset when $V_{ADCIN} < V_{REF}$.

The comparator performs comparison when the ADCCMP flag has been read.

In other words, when the ADCCMP flag has been read by using the “PEEK” instruction, comparison is performed by manipulating switches 1 and 2.

Therefore, the time required for the A/D converter to perform comparison once is equal to one instruction execution time (4.44 μs).

16.5.3 explains the configuration and function of the A/D converter compare judge register.

16.5.3 Configuration and function of A/D converter compare judge register

The A/D converter compare judge register detects the result of comparison between input voltage V_{ADCIN} and compare voltage V_{REF}.

The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
A/D converter compare judge register	0	0	0	ADCCMP	06H	R

ADCCMP	Detects result of comparison by A/D converter
0	$V_{ADCIN} < V_{REF}$
1	$V_{ADCIN} > V_{REF}$

Fixed to "0"

On reset	Power-ON	0	0	0	Undefined
	Clock stop	↓	↓	↓	Retained
	CE	↓	↓	↓	Retained

16.6 Performances of A/D Converter

The performances of the A/D converter are as follows.

Parameter	Performance
Resolution	1LSB
Input voltage range	0-V_{DD}
Quantization error	$\pm\frac{1}{2}$ LSB
Over range	$\frac{62.5}{64} \times V_{DD}$
Error of offset, gain, and non-linearity	$\pm\frac{3}{2}$ LSB**Note**

Note Including quantization error.

16.7 Using A/D Converter

16.7.1 Comparing one compare voltage

Here is a program example.

Example To compare input voltage V_{ADCIN} of ADC_0 pin with compare voltage V_{REF} (31.5/64 V_{DD}) and branch to AAA if $V_{ADCIN} > V_{REF}$ or to BBB if $V_{ADCIN} < V_{REF}$

```
INIT:
    ADCR7   FLG  0.0EH.3      ; Dummy
    ADCR6   FLG  0.0EH.2      ; Dummy
    ADCR5   FLG  0.0EH.1      ; Defines each bit of data buffer as ADCR data setting
    ADCR4   FLG  0.0EH.0      ; flag
    ADCR3   FLG  0.0FH.3
    ADCR2   FLG  0.0FH.2
    ADCR1   FLG  0.0FH.1
    ADCR0   FLG  0.0FH.0

    CLR3    ADCCH2, ADCCH1, ADCCH0
                              ; Sets P1D0/ADC0 pin for the A/D converter
START:
    INITFLG NOT ADCR3, NOT ADCR2, NOT ADCR1, NOT ADCR0
    INITFLG NOT ADCR7, NOT ADCR6,     ADCR5, NOT ADCR4
    PUT     ADCR, DBF         ; Sets compare voltage VREF to 31.5/64 VDD
    SKT1    ADCCMP            ; Detects ADCCMP flag, and
    BR      AAA               ; branches to AAA if False (0)
    BR      BBB               ; branches to BBB if True (1)
```

16.7.2 Successive approximation by binary search method

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

To convert an input voltage into a digital signal, therefore, successive approximation must be executed by program.

If the processing time of the successive approximation program differs depending on the input voltage, the relation with the other processing programs is undesirable.

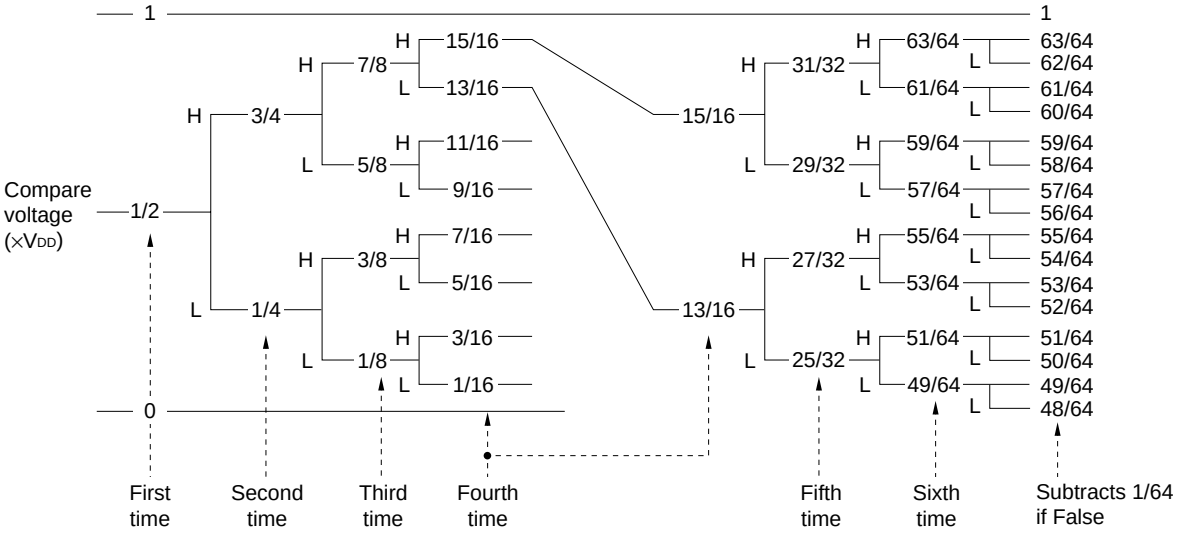
Therefore, use of the binary search method as explained in (1) through (3) below is recommended.

(1) Concept of binary search

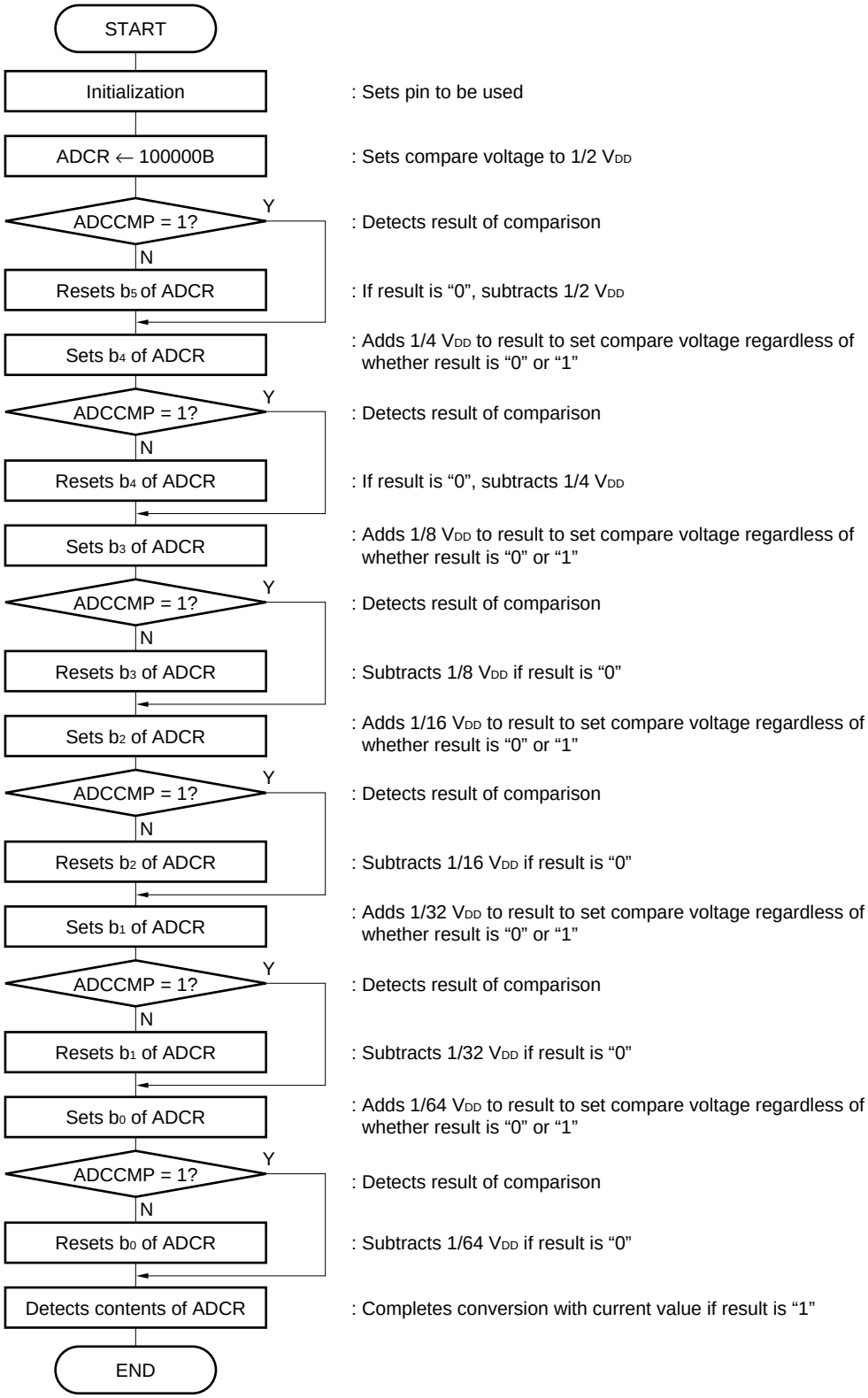
The concept of binary search is explained below.

First, the compare voltage is set to 1/2 V_{DD}. If the result of comparison is True (a high level is input), a voltage of 1/4 V_{DD} is added to the result; if the result of comparison is False (a low level is input), a voltage of 1/4 V_{DD} is subtracted from the result and compared.

Subsequently, the compare voltage is sequentially compared with 1/8 V_{DD} and 1/16 V_{DD} to 1/64 V_{DD}. If the result of comparison is False after comparison has been executed six times, 1/64 V_{DD} is subtracted from the result and comparison is completed.



(2) Flowchart of binary search method



(3) Program example of binary search method

(a) Method with short conversion time

```
INIT:
     ADCR7   FLG  0.0EH.3   ; Dummy
     ADCR6   FLG  0.0EH.2   ; Dummy
     ADCR5   FLG  0.0EH.1   ; Defines each bit of data buffer as ADCR data setting flag
     ADCR4   FLG  0.0EH.0
     ADCR3   FLG  0.0FH.3
     ADCR2   FLG  0.0FH.2
     ADCR1   FLG  0.0FH.1
     ADCR0   FLG  0.0FH.0

     CLR3    ADCCH2, ADCCH1, ADCCH0
                            ; Sets P1D0/ADC0 pin for the A/D converter
START:
     INITFLG NOT ADCR3, NOT ADCR2, NOT ADCR1, NOT ADCR0
     INITFLG NOT ADCR7, NOT ADCR6,     ADCR5, NOT ADCR4
     PUT     ADCR, DBF      ; Sets compare voltage to 31.5/64 VDD
     SKT1    ADCCMP         ; Detects ADCCMP and subtracts
     CLR1    ADCR5          ; 32/64 VDD if "0" and adds
     SET1    ADCR4          ; 16/64 VDD
     PUT     ADCR, DBF
     SKT1    ADCCMP         ; Detects ADCCMP and subtracts
     CLR1    ADCR4          ; 16/64 VDD if "0" and adds
     SET1    ADCR3          ; 8/64 VDD
     PUT     ADCR, DBF
     SKT1    ADCCMP         ; Detects ADCCMP and subtracts
     CLR1    ADCR3          ; 8/64 VDD if "0" and adds      } A/D conversion
     SET1    ADCR2          ; 4/64 VDD
     PUT     ADCR, DBF
     SKT1    ADCCMP         ; Detects ADCCMP and subtracts
     CLR1    ADCR2          ; 4/64 VDD if "0" and adds
     SET1    ADCR1          ; 2/64 VDD
     PUT     ADCR, DBF
     SKT1    ADCCMP         ; Detects ADCCMP and subtracts
     CLR1    ADCR1          ; 2/64 VDD if "0" and adds
     SET1    ADCR0          ; 1/64 VDD
     PUT     ADCR, DBF
     SKT1    ADCCMP         ; Detects ADCCMP and subtracts
     CLR1    ADCR0          ; 1/64 VDD if "0"
END:
```

Number of program steps : 31 steps
Number of execution steps: 31 steps
A/D conversion time : 137.8 μs

(b) Method with fewer program steps

```
ADWORK1  MEM     0.00H            ; Work area for changing compare voltage
ADWORK0  MEM     0.01H

         INITFLG NOT ADCCH2, NOT ADCCH1, NOT ADCCH0
                                  ; Sets P1D0/ADC0 pin for the A/D converter
START:                                                              }
         MOV     DBF1, #0010B     ; Sets compare voltage to initial value of 31.5/64 VDD
         MOV     DBF0, #0000B
         MOV     ADWORK1, #0001B
         MOV     ADWORK0, #0000B
AD_CHECK:
         PUT     ADCR, DBF        ; Sets compare voltage VREF
         SKT1    ADCCMP           ; Detects ADCCMP flag
         BR      ADIN_L
         ADD     DBF0, ADWORK0    ; Increases compare voltage if "1"
         ADDC    DBF1, ADWORK1
         BR      NEXT_AD                                            } A/D
ADIN_L:                                                             } conversion
         SUB     DBF0, ADWORK0    ; Decreases compare voltage if "0"
         SUBC    DBF1, ADWORK1
;        NOP                      ; Described to keep A/D conversion time constant
NEXT_AD:
         RORC    ADWORK1
         RORC    ADWORK0
         SKT1    CY               ; 6 bits completed?
         BR      AD_CHECK
         PUT     ADCR, DBF
         SKT1    ADCCMP
         AND     DBF0, #1110B                                       }
         ⋮
```

Number of program steps : 22 steps
Number of execution steps: 58 to 63 steps
A/D conversion time : 257.8 to 280 μs

Where the A/D conversion time is held constant

Number of program steps : 23 steps
Number of execution steps: 63 steps
A/D conversion time : 280 μs

16.8 Notes on Using A/D Converter

When the P0D3/ADC5 to P0D0/ADC2 pin is used for the A/D converter and if it is specified that the halt status can be released by key input, the halt status may not be set.

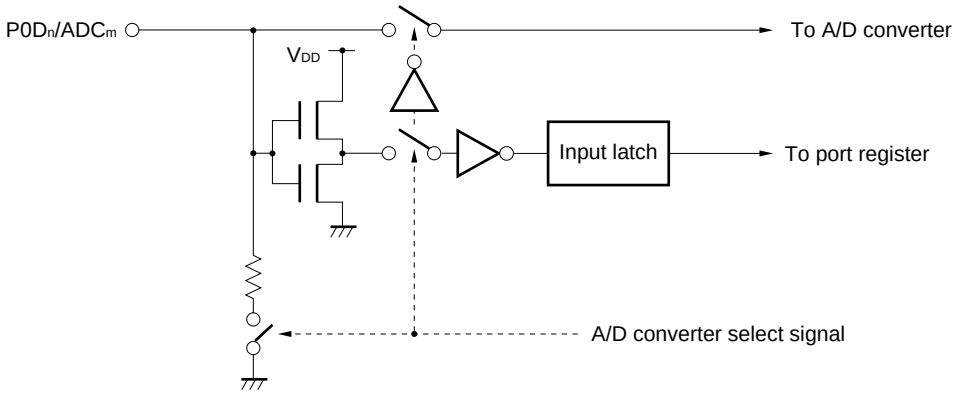
This happens because, as explained in 12.4 Halt Function, the pin selected for the A/D converter is disconnected from the latch of the input port.

Figure 16-5 shows the relation between the P0D3/ADC5 through P0D0/ADC2 pins and input latch.

As shown in this figure, if a high level is input to the input pin when the A/D converter is set by the A/D converter select signal, the status "1" of input latch is retained.

Even if an attempt is made to set the state where the halt status is released by key input at this time, therefore, since a high level is being input to this pin, the halt status is immediately released.

Figure 16-5. Relation between P0D3/ADC5 through P0D0/ADC2 Pins and Input Latch



16.9 Status on Reset

16.9.1 On power-ON reset

All the P0D3/ADC5 through P0D0/ADC2 pins, P1D1/ADC1, and P1D0/ADC0 pins are set in the general-purpose input port mode.

16.9.2 On execution of clock stop instruction

All the P0D3/ADC5 through P0D0/ADC2 pins, P1D1/ADC1, and P1D0/ADC0 pins are set in the general-purpose input port mode.

16.9.3 On CE reset

All the P0D3/ADC5 through P0D0/ADC2 pins, P1D1/ADC1, and P1D0/ADC0 pins are set in the general-purpose input port mode.

17. D/A CONVERTER (DAC)

The D/A converter (DAC) outputs its signal by means of PWM (Pulse Width Modulation) that varies the duty factor.

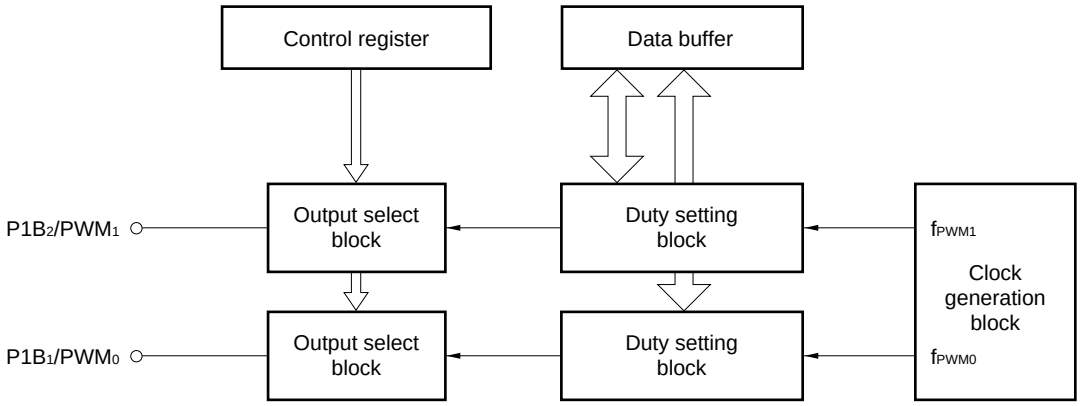
By connecting an external lowpass filter to the D/A converter, digital signals can be converted into analog signals.

17.1 Configuration of D/A Converter

Figure 17-1 shows the block diagram of the D/A converter.

As shown in the figure, the D/A converter consists of an output select block and a duty setting block for each pin, and a clock generation block.

Figure 17-1. Block Diagram of D/A Converter



17.2 Functional Outline of D/A Converter

The D/A converter outputs a variable-duty signal to each output pin.

The output frequency is 878.9 Hz, and the duty factor can be changed in 256 steps.

The following subsections 17.2.1 through 17.2.3 outlines the function of each block of the D/A converter.

17.2.1 Output select blocks

The output select blocks specify whether each pin is used as a general-purpose output port pin or a D/A converter pin.

The mode of each pin is selected by the PWM pin setting software macro supplied with the device file (refer to **17.3**).

17.2.2 Duty setting blocks

The duty setting blocks output a signal whose duty factor can be changed in 256 steps.

The duty factor of each output pin is independently set by the PWM data register (PWMR0 or PWMR1: peripheral address 05H or 06H) via the data buffer.

17.2.3 Clock generation block

The clock generation block generates a basic clock that is used to set the duty factor of the output signal.

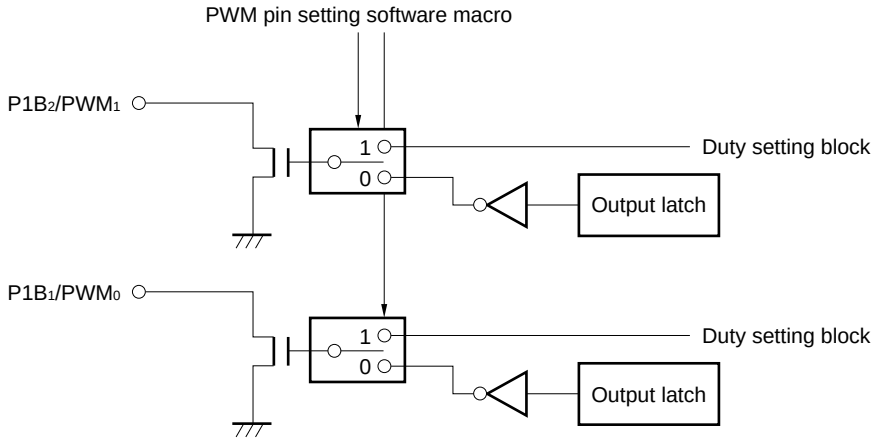
The generated clock frequency f_{PWM} is 225 kHz (refer to **17.4**).

17.3 Output Select Blocks

17.3.1 Configuration of output select blocks

Figure 17-2 shows the configuration of the output select blocks.

Figure 17-2. Configuration of Output Select Blocks



17.3.2 Function of output select blocks

The output select blocks select whether the P1B2/PWM1 and P1B1/PWM0 are used as general-purpose output port pins or D/A converter pins.

This selection can be made by the PWM pin setting software macro supplied with the device file. Each pin can be set in the port mode or D/A converter mode independently.

The P1B2/PWM1 and P1B1/PWM0 pins are N-ch open-drain output pins and must be connected with an external pull-up resistor.

Table 17-1 lists the PWM pin setting software macros.

Table 17-1. PWM Pin Setting Software Macros

Function	Macro Format
Uses P1B1/PWM0 pin as PWM pin	SET1_PWM0ON
Uses P1B2/PWM1 pin as PWM pin	SET1_PWM1ON
Uses P1B1/PWM0 and P1B2/PWM1 pins as PWM pins	SET2_PWM1ON_PWM0ON
Uses P1B1/PWM0 pin as general-purpose output port pin	CLR1_PWM0ON
Uses P1B2/PWM1 pin as general-purpose output port pin	CLR1_PWM1ON
Uses P1B1/PWM0 and P1B2/PWM1 pins general-purpose output port pins	CLR2_PWM1ON_PWM0ON
Uses P1B1/PWM0 pin as PWM pin and P1B2/PWM1 pin as general-purpose output port pin	INIT_NOT_PWM1ON_PWM0ON
Uses P1B1/PWM0 pin as general-purpose output port pin and P1B2/PWM1 pin as PWM pin	INIT_PWM1ON_NOT_PWM0ON

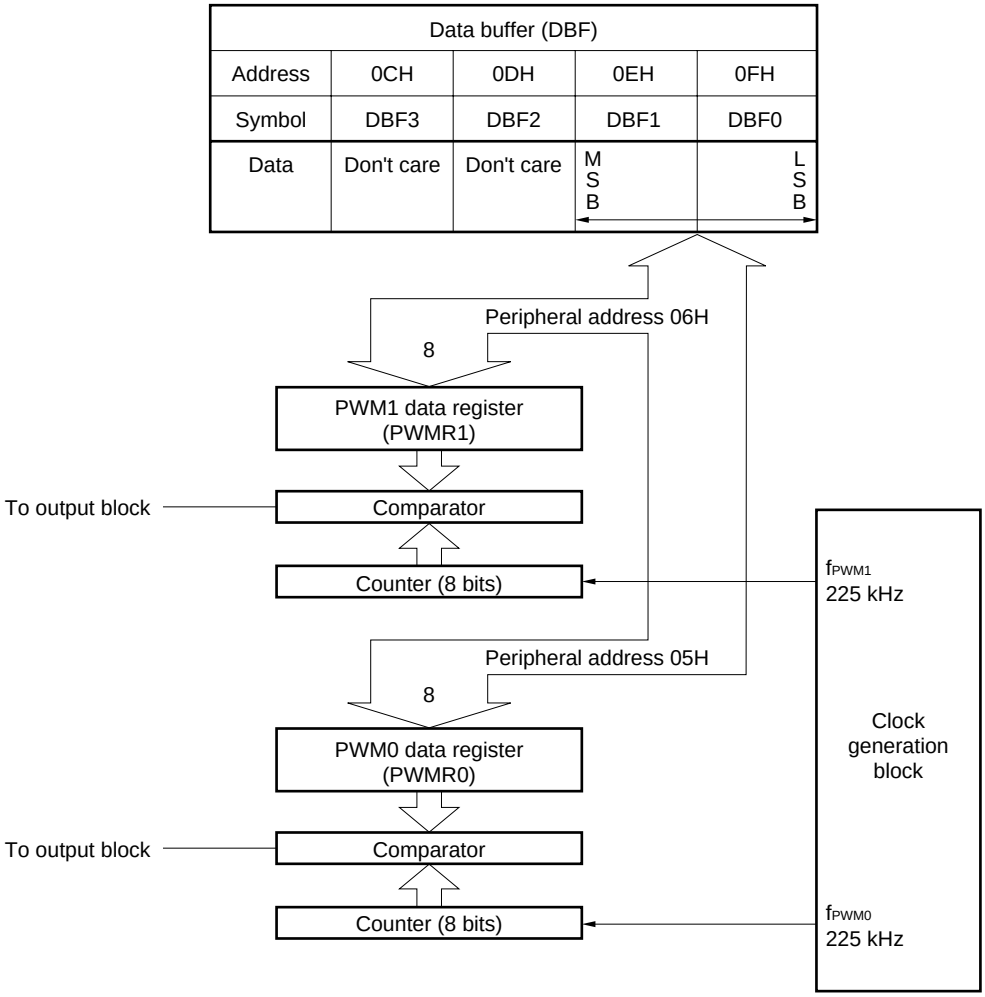
Caution If the above macros are used, the contents of the register window are destroyed.
If the above macro instructions immediately follow an embedded macro instruction, an "object error" occurs when the source file is assembled and loaded to the in-circuit emulator. If an embedded macro exists immediately before the above macro instructions, insert a comment statement in between.

17.4 Duty Setting Blocks and Clock Generation Block

17.4.1 Configuration of duty setting blocks and clock generation block

Figure 17-3 shows he configuration of the duty setting blocks and clock generation block.

Figure 17-3. Configuration of Duty Setting Blocks and Clock Generation Block

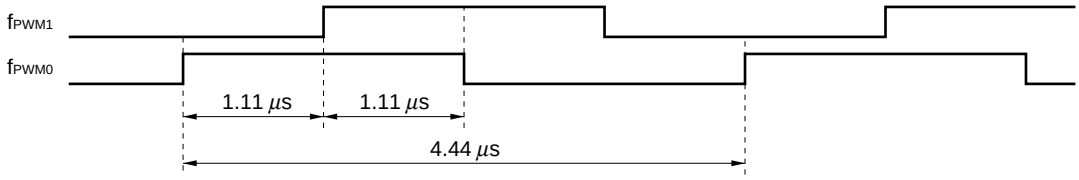


17.4.2 Function and configuration of clock generation blocks

The clock generation blocks outputs basic clocks (f_{PWM1} and f_{PWM0}) that set the duty factor of each output signal (of PWM_1 and PWM_0 pins).

The output frequency is 225 kHz (4.44 μs) for both f_{PWM1} and f_{PWM0}.

However, f_{PWM1} and f_{PWM0} have the following phase difference.



17.4.3 Function and operation of duty setting blocks

The duty setting blocks compare the value set to each PWM data register (PWM1 and PWM0) with the value of each basic clock (f_{PWM1} and f_{PWM0}) counted by an 8-bit counter, and output a high level if the value of the PWM data register is greater, and a low level if the value of PWM data register is less.

Where the value set to the PWM data register is “x”, the duty factor is as follows.

$$\text{Duty factor: } D = \frac{x + 0.25}{256} \times 100\ \%$$

0.25 is an offset. A high level is output even when x = 0.

Because the basic clock is 225 kHz, the frequency and cycle of the output signal are as follows.

$$\text{Frequency: } f = \frac{225\ \text{kHz}}{256} = 878.9\ \text{Hz}$$

$$\text{Cycle: } t = \frac{256}{225\ \text{kHz}} = 1137.8\ \mu\text{s}$$

An independent value can be set to each PWM data register via the data buffer.

In other words, each pin can output a signal with an independent duty factor.

The following subsections 17.4.4 and 17.4.5 explain the configuration and function of each PWM data register, and the relation between the output waveform and duty factor of each pin.

17.4.4 Configuration and function of each PWM data register

The function of each PWM data register is illustrated below.

The PWM data register sets the duty factor of a D/A converter (PWM output) output signal.

Name	Data buffer															
Symbol	DBF3				DBF2				DBF1				DBF0			
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	Don't care				Don't care				Transfer data							

8

GET can be executed

PUT can be executed

Peripheral register											
Name	b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	Symbol	Peripheral address	Peripheral hardware
PWM0 data register	Valid data								PWMR0	05H	PWM_0 pin
PWM1 data register									PWMR1	06H	PWM_1 pin

Sets PWM output duty of each pin
0
|
x
|
255

$$\text{Duty : } D = \frac{x + 0.25}{256} \times 100\ \%$$

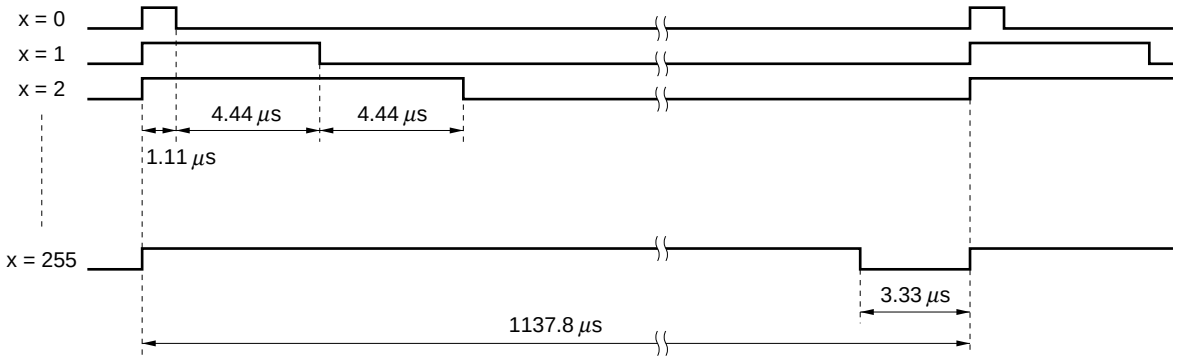
$$\text{Frequency : } f = \frac{225}{256}\ \text{kHz}$$

$$= 878.9\ \text{Hz}$$

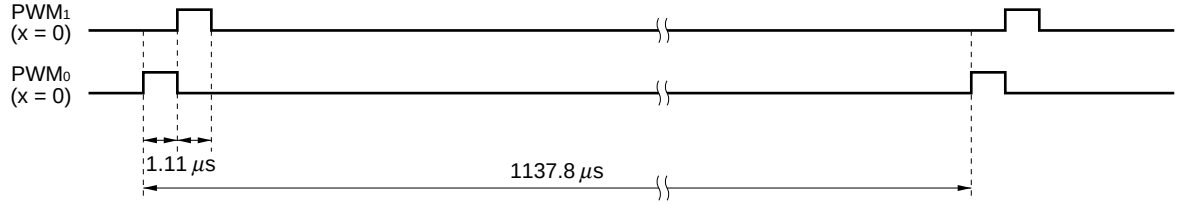
17.4.5 Relation of output waveform and each pin of D/A converter

(1) shows the relation between the output waveform and duty factor. (2) shows the relation of the output waveform of each pin.

(1) Duty factor and output waveform



(2) Output waveform of each pin



17.5 Status on Reset

17.5.1 On power-ON reset

The P1B2/PWM1 and P1B1/PWM0 pins are set in the general-purpose output port mode.

The output value is “undefined”.

The value of each PWM data register is “undefined”.

17.5.2 On execution of clock stop instruction

The P1B2/PWM1 and P1B1/PWM0 pins are set in the general-purpose output port mode.

The output value is the “previous contents of the output latch”.

Each PWM data register retains the previous value.

17.5.3 On CE reset

The P1B2/PWM1 and P1B1/PWM0 pins retain the previous output status.

Therefore, the pin used for the D/A converter retains the current PWM output.

17.5.4 In halt status

The P1B2/PWM1 and P1B1/PWM0 pins retain the previous output status.

Therefore, the pin used for the D/A converter retains the current PWM output.

18. BEEP OUTPUT

The BEEP output function outputs a clock of 200 Hz, 1 kHz, or 3 kHz from the P1B0/BEEP pin.

18.1 Method of BEEP Output

The BEEP output function is controlled by a software macro defined in the device file. To start or stop the BEEP output, describe this software macro in the source program.

(1) Software macro starting BEEP output

This macro sets the frequency of the BEEP output and the P1B0/BEEP pin in the BEEP output mode, to start the BEEP output.

(a) To output 200 Hz

BEEPON200

(b) To output 1 kHz

BEEPON1K

(c) To output 3 kHz

BEEPON3K

(2) Software macro stopping BEEP output

This macro stops the BEEP output. Consequently, the P1B0/BEEP pin can be used as a general-purpose output port pin, and the contents of the output latch at that time can be output.

(a) To stop BEEP output

BEEPOFF

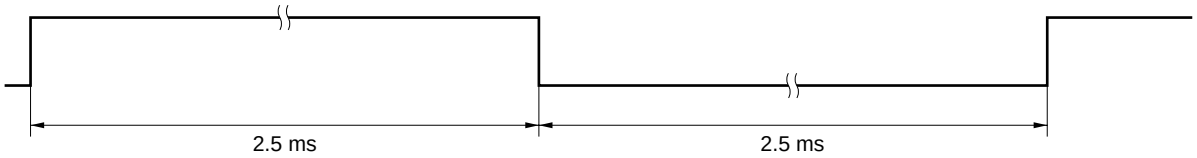
Caution If the above macros are used, the contents of the register window are destroyed.
If the above macro instructions immediately follow an embedded macro instruction, an "object error" occurs when the source file is assembled and loaded to the in-circuit emulator.
If an embedded macro exists immediately before the above macro instructions, insert a comment statement in between.

18.2 Output Waveform of BEEP

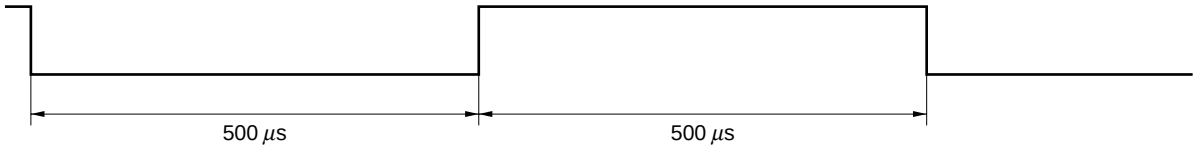
The duty factor of the output waveform of BEEP is 50 %.

The output waveform is shown below.

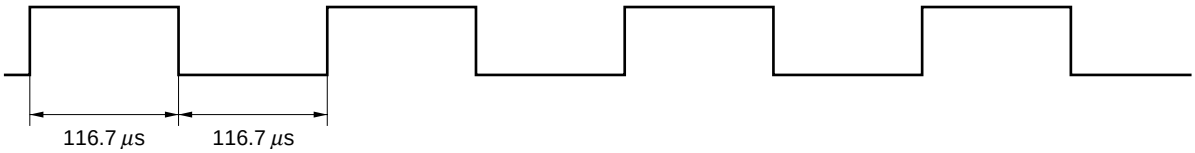
(1) 200-Hz output



(2) 1-kHz output



(3) 3-kHz output



18.3 Status on Reset

(1) On power-ON reset

The P1B0/BEEP pin is set in the general-purpose output port mode. The value of the output latch is undefined, and therefore undefined data is output.

(2) On execution of clock stop instruction

The P1B0/BEEP pin is set in the general-purpose output port mode. The value of the output latch before the clock stop instruction is executed is retained, and the value of the output latch at that time is output.

(3) On CE reset

The P1B0/BEEP pin operates as is with the previous status retained.

(4) In halt status

The P1B0/BEEP pin operates as is with the previous status retained.

18.4 Notes on Using BEEP Function

The BEEP function block shares part of the hardware with the frequency counter that is explained later.

Therefore, the BEEP function and frequency counter cannot be used at the same time.

If the data of the IF counter mode select register and IF counter data register (IFC: peripheral address 43H) are manipulated while the BEEP function is being used, the operation explained in 18.4.1 is performed.

If the BEEP output software macro is executed while the frequency counter is being used, the operation explained in 18.4.2 is performed.

18.4.1 When BEEP function is used

(1) When IFCMD1 and IFCMD0 flags of IF counter mode select register are manipulated

If a value other than "0" is written to the IFCMD1 and IFCMD0 flags, the $P1B_0$/BEEP pin retains the current output level, and stops the BEEP output.

When both the IFCMD1 and IFCMD0 flags are reset to "0", the BEEP output is started.

(2) When IF counter data register is manipulated

The BEEP output is not affected even if the IF counter data register is read (by GET) or written (by PUT).

An "undefined value" is read from the register. Nothing is changed even if data is written to the register.

However, do not write anything to the IF counter data register because this register is a read-only register.

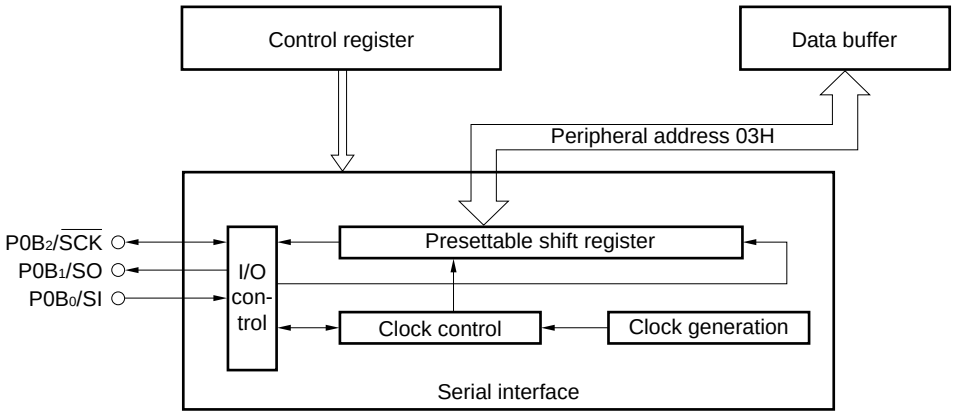
18.4.2 When frequency counter is being used

If the BEEP output software macro is executed while the frequency counter is being used, the frequency counter is forcibly stopped, and the BEEP output is started. At this time, the contents of the window register (WR) and data buffer (DBF) are destroyed. To operate the frequency counter again, stop the BEEP output once by using the BEEPOFF software macro, and start counting from the beginning.

19. SERIAL INTERFACE

The serial interface is used to transfer 8-bit serial data with an external device.

Figure 19-1. Block Diagram of Serial Interface



19.1 Configuration of Serial Interface

Figure 19-2 shows the configuration of the serial interface.

As shown in the figure, the shift clock control block of the serial interface consists of a clock I/O pin block, clock generation block, wait control block, and clock count block.

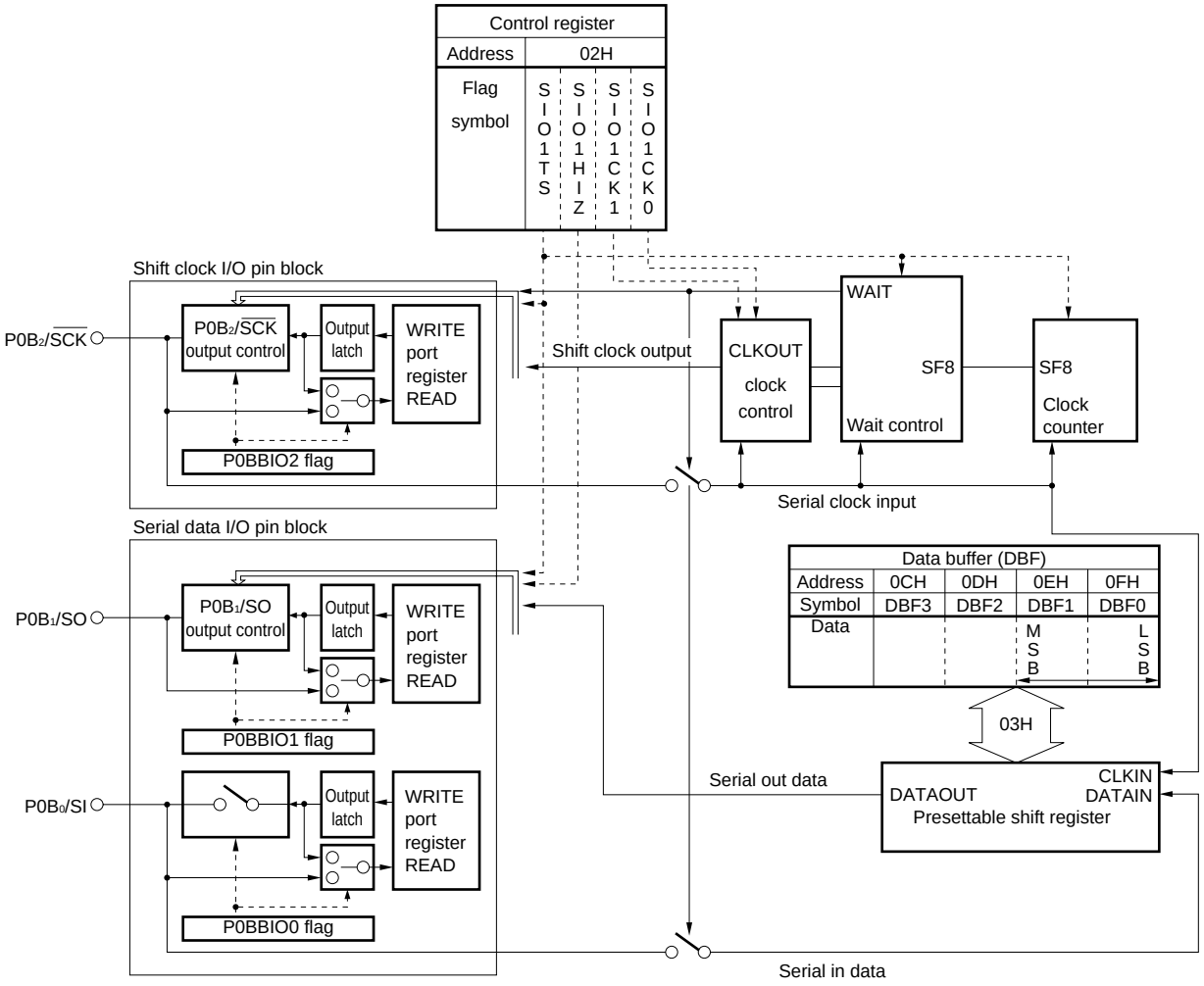
The serial data control block consists of a serial data I/O pin block and a presettable shift register.

These blocks are controlled by the corresponding flags of the control registers.

Data is written to or read from the presettable shift register via the data buffer.

The following section 19.2 outlines each block.

Figure 19-2. Configuration of Serial Interface



19.2 Functional Outline of Serial Interface

The serial interface uses the P0B2/$\overline{\text{SCK}}$, P0B1/SO, and P0B0/SI pins.

The serial interface can select the internal clock or an external clock, and can execute reception and transfer operations.

The following subsections 19.2.1 through 19.2.6 outline the functions of the respective blocks of the serial interface.

For details of each block, refer to **19.3** through **19.7**.

19.2.1 Shift clock I/O pin block

This block selects a shift clock I/O pin.

The shift clock I/O pin is selected by the serial I/O mode select register.

Refer to **19.3**.

19.2.2 Serial data I/O pin block

This block selects a serial data I/O pin.

The serial data I/O pin is selected by the serial I/O mode select register.

Refer to **19.3**.

19.2.3 Clock generation block

This block selects the clock frequency of the shift clock and controls the shift clock output timing.

The shift clock frequency is selected by the serial I/O mode select register.

Refer to **19.4**.

19.2.4 Clock counter

The clock counter counts the number of rising edges of the clock output by the shift clock output pin and outputs a signal at the eighth clock (SF8 signal).

The SF8 signal is used to make serial communication wait (pause).

Refer to **19.5**.

19.2.5 Presettable shift register (SIO1SFR)

This shift register sets serial out data and stores serial in data.

It performs a shift operation by using the clock of the shift clock I/O pin and inputs/outputs data.

The output data is set and the input data is read via the data buffer.

Refer to **19.6**.

19.2.6 Wait control block

This block controls places or releases serial communication in or from the wait status.

Serial communication is placed in or released from the wait status by the serial I/O mode select register.

Refer to **19.7**.

19.3 Shift Clock and Serial Data I/O Pin Control Blocks

The shift clock and serial data I/O pin control blocks sets the pins of the serial interface and controls the transfer/reception operation.

These control operations are specified by the serial I/O mode select register.

19.3.1 shows the configuration and function of the serial I/O mode select register.

19.3.2 indicates the status of each pin set by the serial I/O mode select register.

19.3.1 Configuration and function of serial I/O mode select register

The configuration and function of the serial I/O mode select register are illustrated below.

The SIO1CK1 and SIO1CK0 flags are used to select the internal clock or an external clock and to set the frequency of the internal clock.

For the details on the clock, refer to **19.4**.

The SIO1TS flag places or releases the serial interface in or from the wait status.

For the wait operation, refer to **19.7**.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Serial I/O mode select register	SIO1TS	SIO1HIZ	SIO1CK1	SIO1CK0	02H	R/W

Set I/O clock frequency of serial interface	
0	0	External clock (slave)
0	1	75 kHz (master)
1	0	150 kHz (master)
1	1	450 kHz (master)

Sets $P0B_1$/SO pin as serial out pin	
0	General-purpose I/O port
1	Serial out

Starts serial communication of serial interface	
0	Does not start serial communication (wait)
1	Starts serial communication (wait release)

On reset	Power-ON	0	0	0	0
	Clock stop	0	0	0	0
	CE	0	0	0	0

19.3.2 Pin status setting by serial I/O mode select register

Table 19-1 shows the status of each pin set by the serial I/O mode select register.

As shown in this table, the I/O select flag of each pin must be also manipulated to set each pin.

For the details on the I/O select flag, refer to **15. GENERAL-PURPOSE PORT**.

Table 19-1. Pin Status Setting by Serial I/O Mode Select Register

SIO1MODE						Pin				
Communication mode	b_2	Setting of serial output	b_1	b_0	Clock direction	Pin symbol	I/O select flag of each pin			Set pin status
	SIO1HIZ		SIO1CK1	SIO1CK0			P0BBIO2	P0BBIO1	P0BBIO0	
3-wire serial I/O			0	0	External clock	P0B2/$\overline{SCK}$	0			Wait : general-purpose input port Wait released : external clock input
							1			Wait : general-purpose output port Wait released : general-purpose output port
			0	1	Internal clock		0			Wait : general-purpose input port Wait released : general-purpose input port
			1	0						
			1	1			1			Wait : High level output Wait released : General-purpose input port
	0	General-purpose port				P0B1/SO		0		General-purpose input port
								1		General-purpose output port
	1	Serial output						0		General-purpose input port
								1		Serial output
						P0B0/SI			0	Serial input
									1	General-purpose output port

19.4 Clock Generation Block

The clock generation block generates the clock when the internal clock is used (i.e., when the master operation is performed) and controls the clock output timing.

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

The shift clock is successively output until the value of the clock counter, which is explained in 19.5, reaches "8".

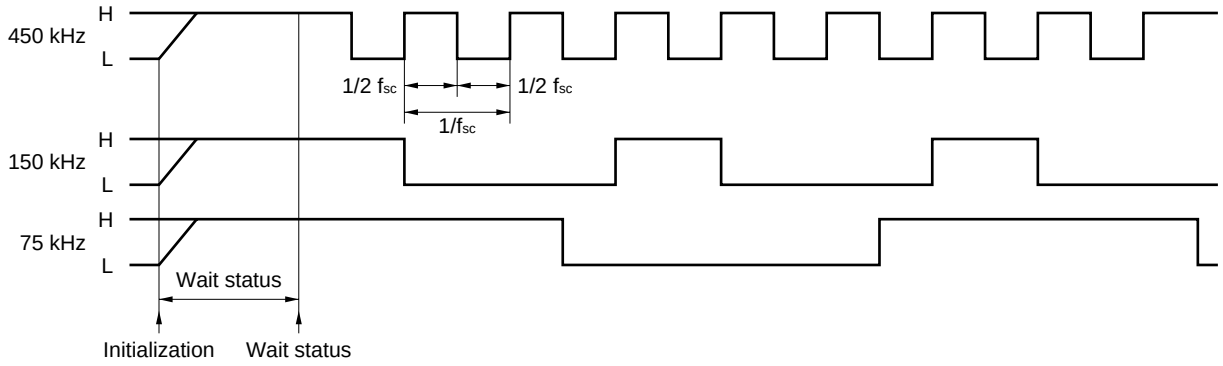
The following subsection 19.4.1 explains the clock output waveform and clock generation timing.

19.4.1 Internal shift clock generation timing

(1) On releasing wait status from initial status

The initial status is the status in which the internal clock is selected, the $P0B_2/\overline{SCK}$ pin is set in the output mode, and a high level is output to this pin.

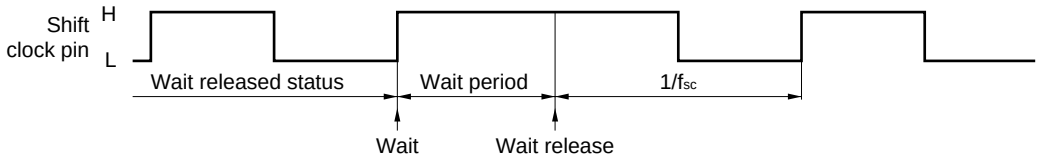
A high level is output to the $P0B_2/\overline{SCK}$ pin in the wait status.



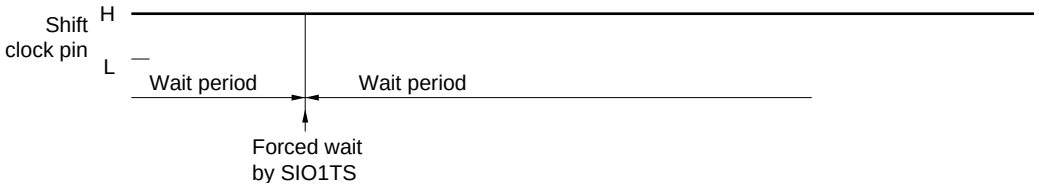
(2) When wait operation is performed

For the details on the wait operation, refer to **19.7**.

(a) Wait status with value of clock counter reaching "8" (normal operation)

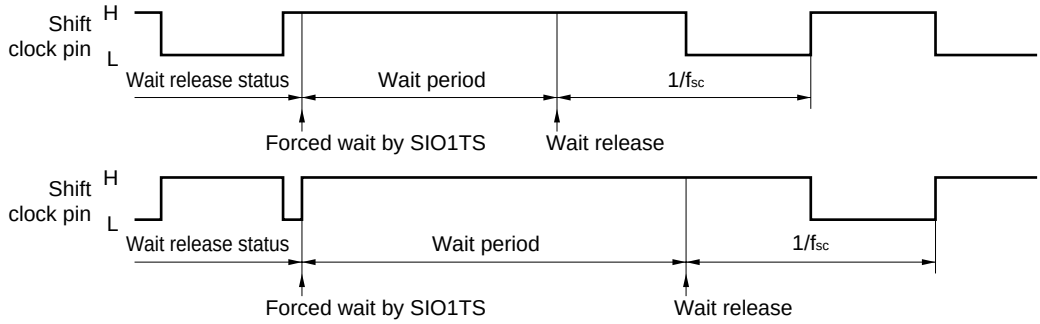


(b) If forced wait is executed in wait status



(c) If forced wait is executed when wait status is released

At this time, the clock counter is reset.



(d) If wait status is released in wait release status

The clock output waveform is not changed at this time.

The clock counter is not reset.

(e) If clock frequency is changed and wait status is released at the same time

The clock frequency can be changed and the wait status can be released by the serial I/O mode select register of the control registers.

Therefore, the clock frequency can be changed and the wait status can be released with a single instruction.

In this case, the operation is the same as releasing the wait status from the initial status as explained in (1).

19.5 Clock Counter

The clock counter is a wrap-around counter that counts the number of the shift clocks output from or input to the shift clock ($P0B_2/\overline{SCK}$) pin.

The clock counter directly reads the status of the shift clock pin. At this time, whether the clock is the internal clock or an external clock is not identified.

The clock counter does not operate in the wait status of serial communication.

When the value of the clock counter is “8”, serial communication is placed in the wait status at the rising edge of the shift clock.

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

The following subsections 19.5.1 and 19.5.2 explain the operation of the clock counter and the conditions under which the clock counter is reset.

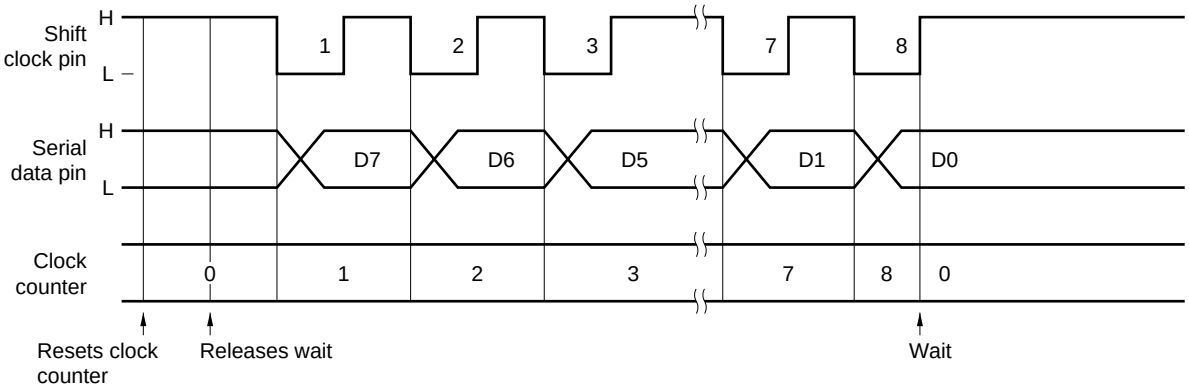
19.5.1 Operation of clock counter

Figure 19-3 shows the operation of the clock counter.

The initial value of the clock counter is “0”. The value of the clock counter is incremented by one each time the falling of the shift clock pin is detected. When the value of the clock counter has been incremented to “8”, the clock counter is reset to “0” at the next rising edge of the shift clock pin.

Serial communication is placed in the wait status when the clock counter has been reset to “0”.

Figure 19-3. Operation of Clock Counter



19.5.2 Clock counter reset condition

The clock counter is reset to 0 when any of the following conditions (1) through (5) is satisfied.

(1) On power-ON reset
(2) On execution of the clock stop instruction
(3) When “0” is written to the SIO1TS flag (forced wait)
(4) When the shift clock rises while the value of the clock counter is “8” with the wait status released
(5) On CE reset

19.6 Presettable Shift Register (SIO1SFR)

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

Data is written to or read from the presettable shift register via the data buffer by using the “PUT” or “GET” instruction.

19.6.1 shows the configuration of the presettable shift register and its relation with the data buffer.

The presettable shift register performs its shift operation in synchronization with the clock applied to the shift clock (P0B$_2$/$\overline{SCK}$) pin.

At this time, the content of the most significant bit (MSB) of the presettable shift register is output to the serial data output pin in synchronization with the fall of the shift clock, and the least significant bit (LSB) of the presettable shift register is read in synchronization with the rise of the clock.

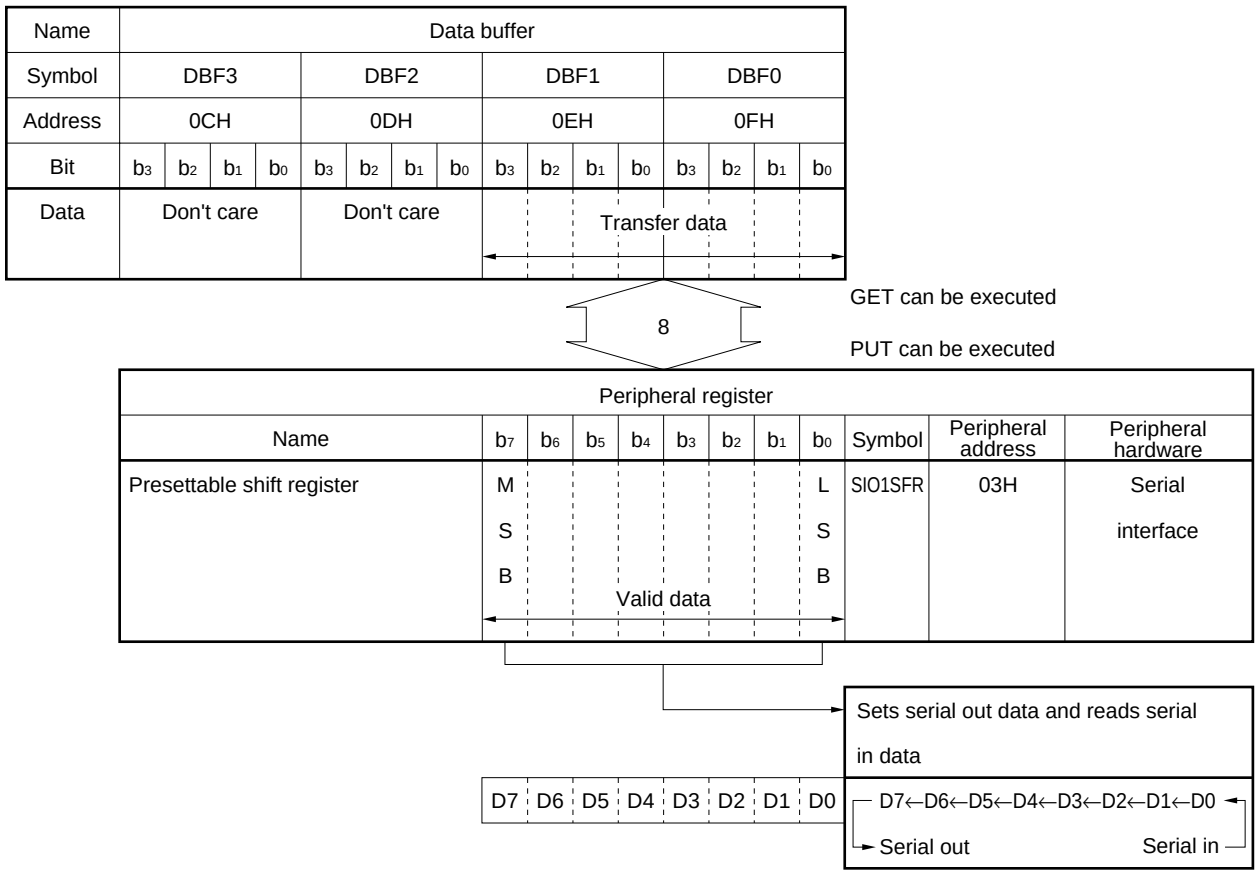
19.6.2 explains the operation of the presettable shift register. 19.6.3 explains the points to be noted in writing or reading data to or from the presettable shift register.

The presettable shift register does not shift data in the wait status.

For the details on the operation in each serial communication mode, refer to **19.8**.

19.6.1 Configuration of presettable shift register and its relation with data buffer

The configuration of the presettable shift register and its relation with the data buffer are shown below.



19.6.2 Operation of presettable shift register

Figure 19-4 illustrates the data shift operation of the presettable shift register.

Table 19-2 shows the data shift operation during transfer and reception.

Figure 19-4. Data Shift Operation of Presettable Shift Register

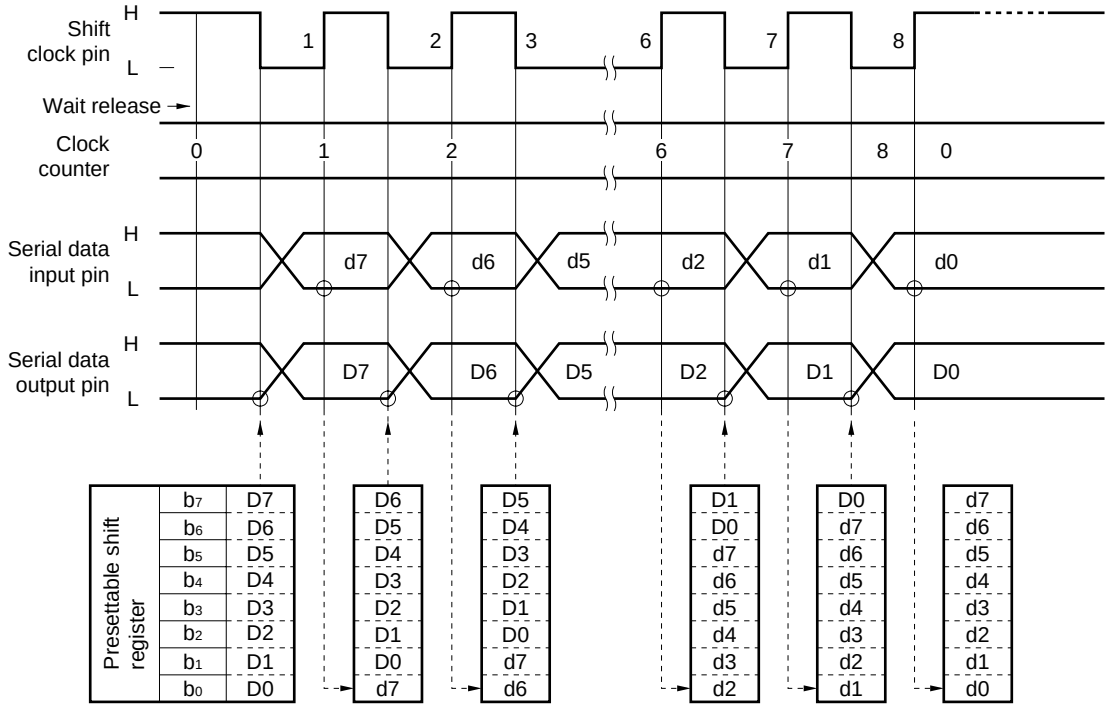


Table 19-2. Data Shift Operation during Transfer and Reception

Serial I/O mode	
Serial input operation	Serial output operation
Shifts and inputs status of P0B$_0$/SI pin at rising edge of shift clock starting from LSB. If P0BBIO0 flag is “0”, contents of output latch are input. This operation is not performed in wait status.	Data is shifted and output, starting from MSB, to P0B$_1$/SO pin at falling edge of shift clock. If P0BBIO1 flag is “1”, or if SIO1HIZ flag is “0”, data is not output. This operation is not performed in wait status.

19.6.3 Notes on setting and reading data

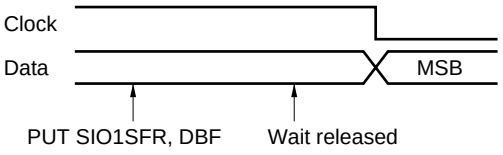
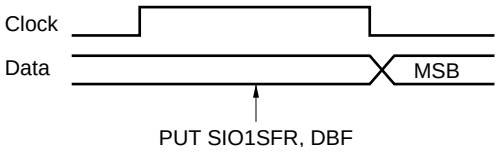
Data is written to the presettable shift register by the “PUT SIO1SFR, DBF” instruction.

Data is read from the register by the “GET DBF, SIO1SFR” instruction.

Set or read data to or from the register in the wait status. While the wait status is released, the data may not be correctly set or read depending on the status of the shift clock pin.

Table 19-3 indicates the timing of setting and reading data and points to be noted.

Table 19-3. Reading (GET) and Writing (PUT) Data from/to Presettable Shift Register and Notes

Status on Execution of PUT/GET		Status of Shift Clock Pin	Presettable Shift Register (SIO1SFR)
Wait status	Read (GET)	With external clock: floated	Normally read.
	Write (PUT)	With internal clock: high level	Normally written. Content of MSB is output at falling edge of shift clock when wait status is released next time (during transfer operation). 
Wait released status	Read (GET)	Low level	Normally read.
		High level	Cannot be read normally. Contents of SIO1SFR are destroyed.
	Write (PUT)	High level	Normally written. Contents of MSB are output when PUT instruction is executed. Clock counter is not reset. 
		Low level	Cannot be written normally. Contents of SIO1SFR are destroyed.

19.7 Wait Block

The wait block controls communication of the serial interface by placing or releasing communication in or from the wait status.

The wait block is controlled by the SIO1TS flag of the serial I/O mode select register.

The following subsection 19.7.1 explains the wait operation and points to be noted.

19.7.1 Wait operation and notes

In the wait status, the clock generation block and presettable shift register stop operation, and serial communication pauses.

Therefore, serial communication can be started when the wait status is released.

The wait status is released when “1” is written to the SIO1TS flag.

When “1” is written to this flag, the internal clock is output to the shift clock output pin (during master operation), and presettable shift register and clock counter start operating.

If the shift clock rises when the value of the clock counter is “8”, the wait status is set. At this time, the SIO1TS flag is automatically reset to “0”.

The operating status of serial communication can be checked by detecting the content of the SIO1TS flag while the wait status is released.

Therefore, data is read or set after “1” has been written to the SIO1TS flag, serial communication has been started, and then clearing of the SIO1TS flag to “0” has been detected.

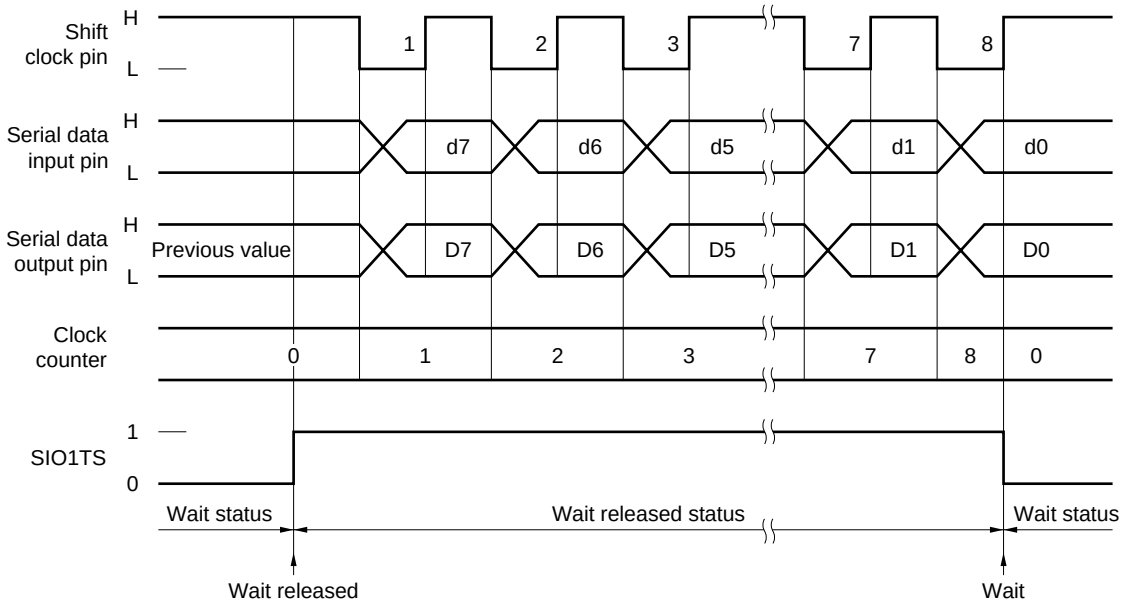
If data is written to (by PUT instruction) or read from (by GET instruction) the presettable shift register while the wait status is released, the correct data may not be written or read.

For details, refer to **19.6.3 Notes on setting and reading data**.

If “0” is written to the SIO1TS flag while the wait status is released, the wait status is set. This is called forced wait. When forced wait is executed, the clock counter is reset to “0”.

Figure 19-5 shows an example of the wait operation.

Figure 19-5. Example of Wait Operation



When the wait status is released, the serial data is output at the next falling edge of the clock, and the wait status is released.

When the shift clock has been input eight times, the shift clock pin outputs a high level, and the clock counter and presettable shift register stop operation.

If data is written to or read from the presettable shift register while the wait status is released and the shift clock pin is high, the correct data may not be set or read.

If data is written to the presettable shift register while the wait status is released and the shift clock pin is low, the content of the MSB is output to the serial data output pin as soon as the “PUT” instruction has been executed.

If forced wait is executed while the wait status is released, the wait status is set and the clock counter is reset to “0” as soon as “0” has been written to the SIO1TS flag.

19.8 Using Serial Interface

Figure 19-6 illustrates the I/O block and communication method of the serial interface.

Table 19-4 indicates the operations of the serial interface in each mode.

As shown in Figure 19-6 and Table 19-4, the serial interface operates on the internal clock (master operation) or external clock (slave operation), and can perform transfer and reception on both the clocks.

The master or slave operation is selected by the SIO1CK1 and SIO1CK0 flags.

Reception or transfer is selected by the SIO1HIZ flag.

During the master operation, the internal shift clock is output from the $P0B_2/\overline{SCK}$ pin. However, the $P0B_2/\overline{SCK}$ pin must be set in the output port mode (P0BBIO2 flag = 1).

During the slave operation, the $P0B_2/\overline{SCK}$ pin is floated, and the serial interface waits for an external clock. At this time, the $P0B_2/\overline{SCK}$ pin must be set in the input port mode (P0BBIO2 flag = 0).

Serial data is output from the $P0B_1$/SO pin at the falling edge of the shift clock, regardless of whether the internal clock or external clock is used. However, the $P0B_1$/SO pin must be set in the output port mode (P0BBIO1 flag = 1) and the SIO1HIZ flag must be set.

As the serial input data, the status of the $P0B_1$/SO pin is input to the presettable shift register at the rising edge of the shift clock, regardless of whether the internal clock or external clock is used. However, the $P0B_0$/SI pin must be set in the input port mode (P0BBIO0 flag = 0).

The $P0B_2/\overline{SCK}$ pin reads the “current status of the output latch” when the value output to the pin is read, and “the current status of the pin” while the wait status is released.

Paragraphs (1) through (4) below Table 19-4 show program examples of transfer and reception during master and slave operations.

Figure 19-6. I/O Block and Communication Method of Serial Interface (1/2)

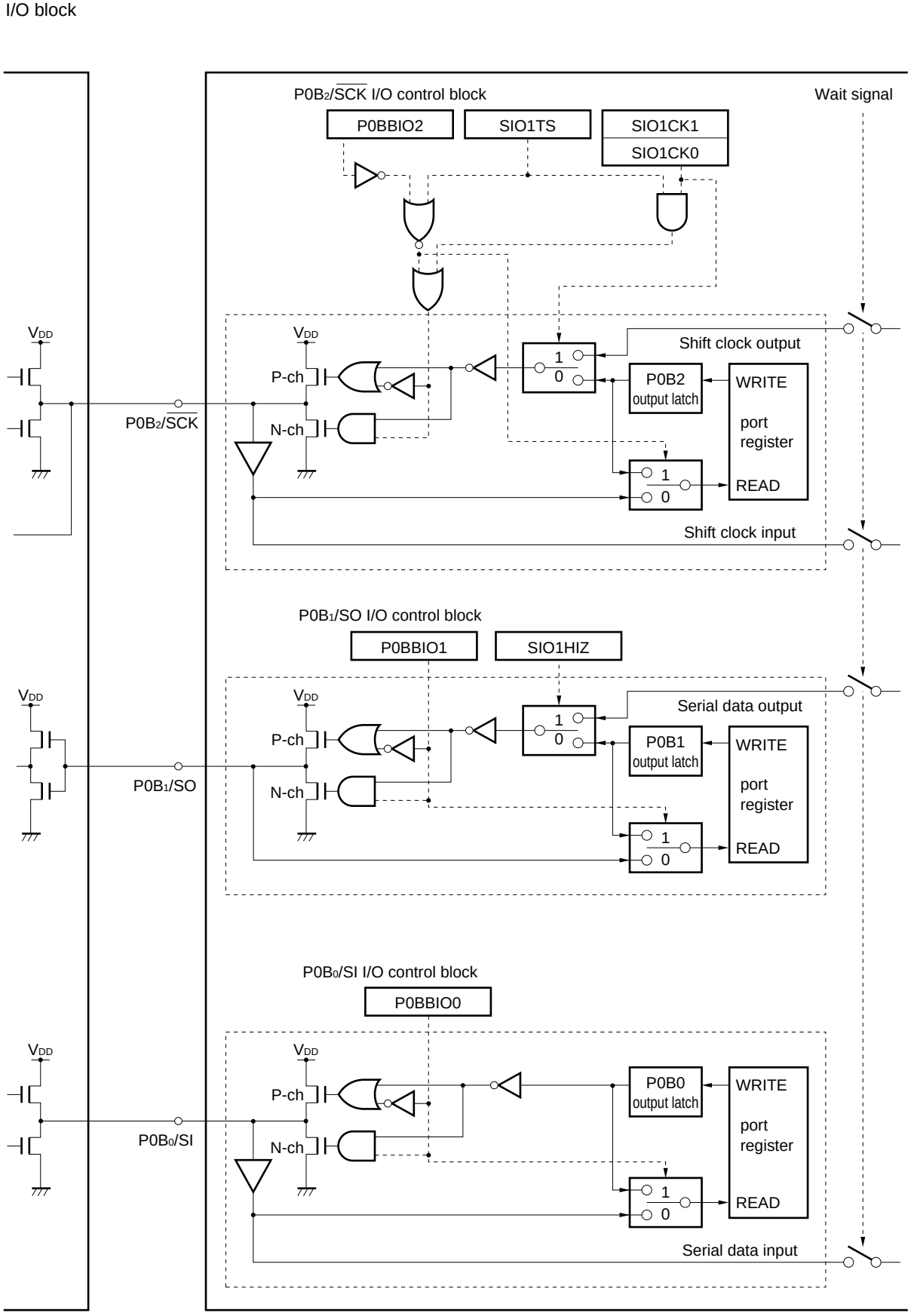


Figure 19-6. I/O Block and Communication Method of Serial Interface (2/2)

Communication

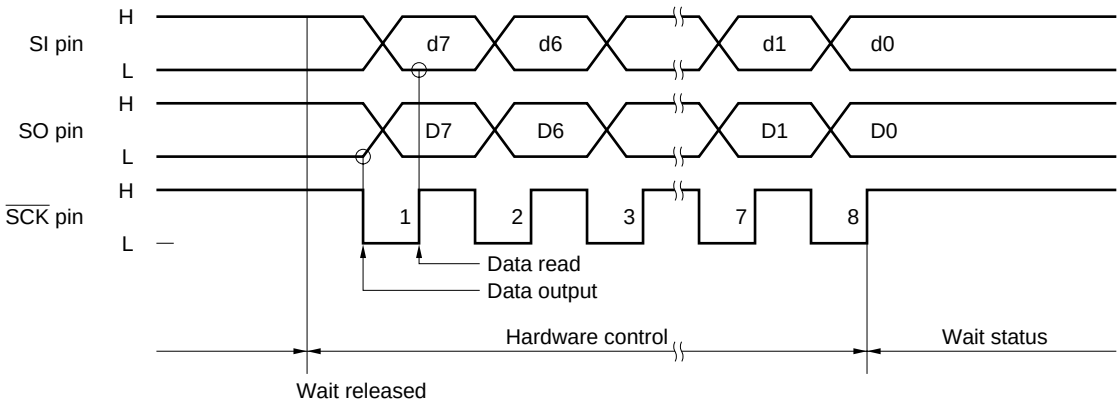


Table 19-4. Operation of Serial Interface in Each Mode

Item \ Operation Mode		3-Wire Serial I/O Mode			
		Slave operation SIO1CK1 = SIO1CK0 = 0		Master operation SIO1CK1 = SIO1CK0 = other than 0	
		Wait	Wait released	Wait	Wait released
Setting status of each pin	$P0B_2/\overline{SCK}$	When P0BBIO2 = 0 Floating General-purpose input port When P0BBIO2 = 1 General-purpose output port Outputs contents of output. Usually, P0BBIO2 is cleared to 0.	When P0BBIO2 = 0 Floating External clock input When P0BBIO2 = 1 General-purpose output port Outputs contents of output latch.	When P0BBIO2 = 0 Floating General-purpose input port When P0BBIO2 = 1 Outputs high level. Usually, P0BBIO2 is set to 1.	When P0BBIO2 = 0 Floating General-purpose input port When P0BBIO2 = 1 Outputs internal clock.
	$P0B_1/SO$	When SIO1HIZ = 0	When SIO1HIZ = 1	When SIO1HIZ = 0	When SIO1HIZ = 1
		When P0BBIO1 = 0 General-purpose input port Floating When P0BBIO1 = 1 General-purpose output port Outputs contents of output latch.	When P0BBIO1 = 0 General-purpose input port Floating When P0BBIO1 = 1 Outputs serial data.	When P0BBIO1 = 0 General-purpose input port Floating When P0BBIO1 = 1 General-purpose output port Outputs contents of output latch.	When P0BBIO1 = 0 General-purpose input port Floating When P0BBIO1 = 1 Outputs serial data.
	$P0B_0/SI$	When P0BBIO0 = 0 Floating Waits for external data When P0BBIO0 = 1 General-purpose output port. Outputs contents of output latch. Usually, P0BBIO0 is cleared to 0.			
Operation of clock counter		Incremented at falling edge of $\overline{SCK}$ pin.			
Operation of presettable shift register (SIO1SFR)		Output When SIO1HIZ = 1 Shifts data from MSB and outputs it to SO_2 pin at falling edge of $\overline{SCK}$ pin. When SIO1HIZ = 0 Does not output data. Input Shifts data of SI pin from LSB and inputs it at rising edge of $\overline{SCK}$ pin regardless of P0BBIO0. However, the contents of output latch are output to SI pin when P0BBIO0 = 1.			
Wait operation		Serial communication is started when “1” is written to SIO1TS. SIO1TS is reset to “0” at rising edge of shift clock when value of clock counter is “8”. For operation of each pin, refer to above.			

(1) Program example of serial interface (in master transfer mode)

Example To transfer 2-byte data “A596H”

```
        SCKBIO  FLG     P0BBIO2
        SOBIO   FLG     P0BBIO1

        MOV     DBF1,  #0AH         ; Sets transfer data
        MOV     DBF0,  #5
        PUT     SIO1SFR, DBF
        SET2    SCKBIO, S0BIO
        INITFLG SIO1TS, SIO1HIZ, NOT SIO1CK1, SIO1CK0
                                    ; Releases wait, and outputs serial data
                                    ; Master (fSC = 75 kHz)
LOOP1:
        SKF1    SIO1TS              ; Wait until wait status is released
        BR      LOOP1
        MOV     DBF1, #9            ; Sets transfer data
        MOV     DBF0, #6
        PUT     SIO1SFR, DBF
        SET1    SIO1TS              ; Releases wait
LOOP2:
        SKF1    SIO1TS              ; Waits until wait status is released
        BR      LOOP2
        ⋮
```

(2) Program example of serial interface (in master reception mode)

Example To receive 2-byte data and store it to addresses 00H through 03H of BANK0

```
        SCKBIO  FLG     P0BBIO2
        SIBIO   FLG     P0BBIO0

        DATA1H  MEM     0.00H       ; Stores high-order 4 bits of first byte
        DATA1L  MEM     0.01H       ; Stores low-order 4 bits of first byte
        DATA2H  MEM     0.02H       ; Stores high-order 4 bits of second byte
        DATA2L  MEM     0.03H       ; Stores low-order 4 bits of second byte

        INITFLG SCKBIO, NOT SIBIO
        INITFLG SIO1TS, NOT SIO1HIZ, SIO1CK1, SIO1CK0
                                    ; Releases wait, and does not output serial data
                                    ; Master (fSC = 450 kHz)
LOOP1:
        SKF1    SIO1TS              ; Wait until wait status is released
        BR      LOOP1
        GET     DBF, SIO1SFR        ; Reads receive data
        ST      DATA1H, DBF1        ; Stores read data
        ST      DATA1L, DBF0
        SET1    SIO1TS              ; Releases wait
LOOP2:
        SKF1    SIO1TS              ; Waits until wait status is released
        BR      LOOP2
        GET     DBF, SIO1SFR        ; Reads receive data
        ST      DATA2H, DBF1        ; Stores read data
        ST      DATA2L, DBF0
        ⋮
```

(3) Program example of serial interface (in slave transfer mode)

Example To transfer 2-byte data “A596H”

```
        SCKBIO  FLG     P0BBIO2
        SOBIO   FLG     P0BBIO1

        INITFLG NOT SCKBIO, SOBIO
        MOV     DBF1, #0AH          ; Sets transfer data
        MOV     DBF0, #5
        PUT     SIO1SFR, DBF
        INITFLG SIO1TS, SIO1HIZ, NOT SIO1CK1, NOT SIO1CK0
                                    ; Releases wait, outputs serial data, slave
LOOP1:
        SKF1    SIO1TS              ; Wait until wait status is released
        BR      LOOP1
        MOV     DBF1, #9            ; Sets transfer data
        MOV     DBF0, #6
        PUT     SIO1SFR, DBF
        SET1    SIO1TS              ; Releases wait
LOOP2:
        SKF1    SIO1TS              ; Waits until wait status is released
        BR      LOOP2
        ⋮
```

(4) Program example of serial interface (in slave reception mode)

Example To receive 2-byte data and store it to addresses 00H through 03H of BANK0

```
        SCKBIO  FLG     P0BBIO2
        SIBIO   FLG     P0BBIO0

        DATA1H  MEM     0.00H       ; Stores high-order 4 bits of first byte
        DATA1L  MEM     0.01H       ; Stores low-order 4 bits of first byte
        DATA2H  MEM     0.02H       ; Stores high-order 4 bits of second byte
        DATA2L  MEM     0.03H       ; Stores low-order 4 bits of second byte

        CLR2    SCKBIO, SIBIO
        INITFLG SIO1TS, NOT SIO1HIZ, NOT SIO1CK1, NOT SIO1CK0
                                    ; Releases wait, does not output serial
                                    ; data, slave
LOOP1:
        SKF1    SIO1TS              ; Wait until wait status is released
        BR      LOOP1
        GET     DBF, SIO1SFR        ; Reads receive data
        ST      DATA1H, DBF1        ; Stores read data
        ST      DATA1L, DBF0
        SET1    SIO1TS              ; Releases wait
LOOP2:
        SKF1    SIO1TS              ; Waits until wait status is released
        BR      LOOP2
        GET     DBF, SIO1SFR        ; Reads receive data
        ST      DATA2H, DBF1        ; Stores read data
        ST      DATA2L, DBF0
        ⋮
```

19.9 Status of Serial Interface on Reset

19.9.1 On power-ON reset

All of the pins P0B2/$\overline{\text{SCK}}$ through P0B0/SI are set in the general-purpose input port mode (floating output).

The value of the presettable shift register is undefined.

19.9.2 On execution of clock stop instruction

All of the pins P0B2/$\overline{\text{SCK}}$ through P0B0/SI are set in the general-purpose input port mode (floating output).

The presettable shift register retains the previous value.

19.9.3 On CE reset

All of the pins P0B2/$\overline{\text{SCK}}$ through P0B0/SI are set in the general-purpose input port mode (floating output).

The presettable shift register retains the previous value.

19.9.4 In halt status

The I/O pins retain the current status.

If the internal clock is used (master operation) at this time, the clock is not output after the “HALT” instruction has been executed.

To use the internal clock, therefore, the “HALT” instruction must be executed after communication has been completed.

If an external clock is forcibly input, the serial interface functions even when the internal clock is used.

If the external clock is used (slave operation), the operation continues even when the “HALT” instruction has been executed.

20. FREQUENCY COUNTER (FC)

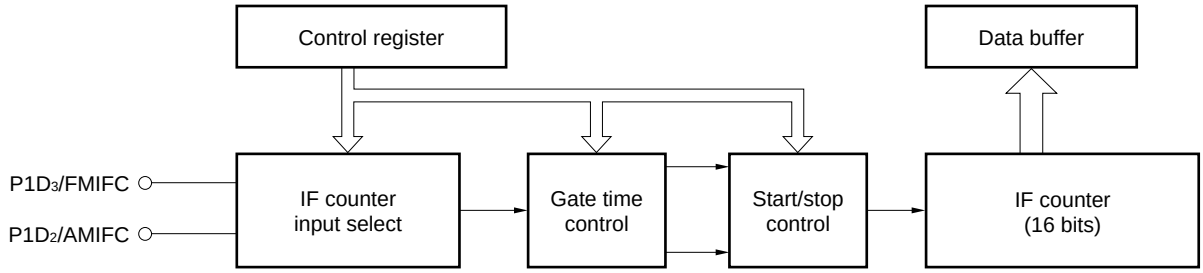
The frequency counter (FC) is used to count the intermediate frequency (IF) of a tuner.

20.1 Configuration of Frequency Counter

Figure 20-1 shows the block diagram of the frequency counter.

As shown in the figure, the IF counter consists of an input select block, gate time control block, start/stop control block, and count block.

Figure 20-1. Block Diagram of Frequency Counter



20.2 Functional Outline of Frequency Counter

The frequency counter has an IF counter function that is used to count the frequency of an externally input signal.

The IF counter function is to count the frequency input to the P1D3/FMIFC or P1D2/AMIFC pin by using a 16-bit counter at fixed time intervals (1 ms, 4 ms, 8 ms, or open).

The following subsections 20.2.1 through 20.2.4 outline each function block of the frequency counter.

Because the frequency counter shares the hardware with the BEEP function explained in **18. BEEP OUTPUT**, it cannot be used with the BEEP function at the same time. For details, refer to **20.7 Notes on Using Frequency Counter**.

20.2.1 IF counter input select block

The IF counter input select block specifies whether the P1D3/FMIFC and P1D2/AMIFC pins are used as general-purpose input port pins or IF counter pins.

This selection is made by the IF counter mode select register (RF address 12H).

For details, refer to **20.3**.

20.2.2 Gate time control block

The gate time control block controls the time during which the frequency is counted by the IF counter mode select register.

The IF counter mode select register sets the internal gate time (1 ms, 4 ms, 8 ms, or open) to count the frequency applied to the $P1D_3$/FMIFC or $P1D_2$/AMIFC pin.

For details, refer to **20.3**.

20.2.3 Start/stop control block

The start/stop control block starts or stops the frequency counter.

Starting or stopping the frequency counter is controlled by the IF counter control register (RF address 23H) and IF counter gate judge register (RF address 04H).

For details, refer to **20.4**.

20.2.4 IF counter

The input frequency is counted by a 16-bit binary counter.

The count value is read by the IF counter data register (IFC: peripheral address 43H) via the data buffer.

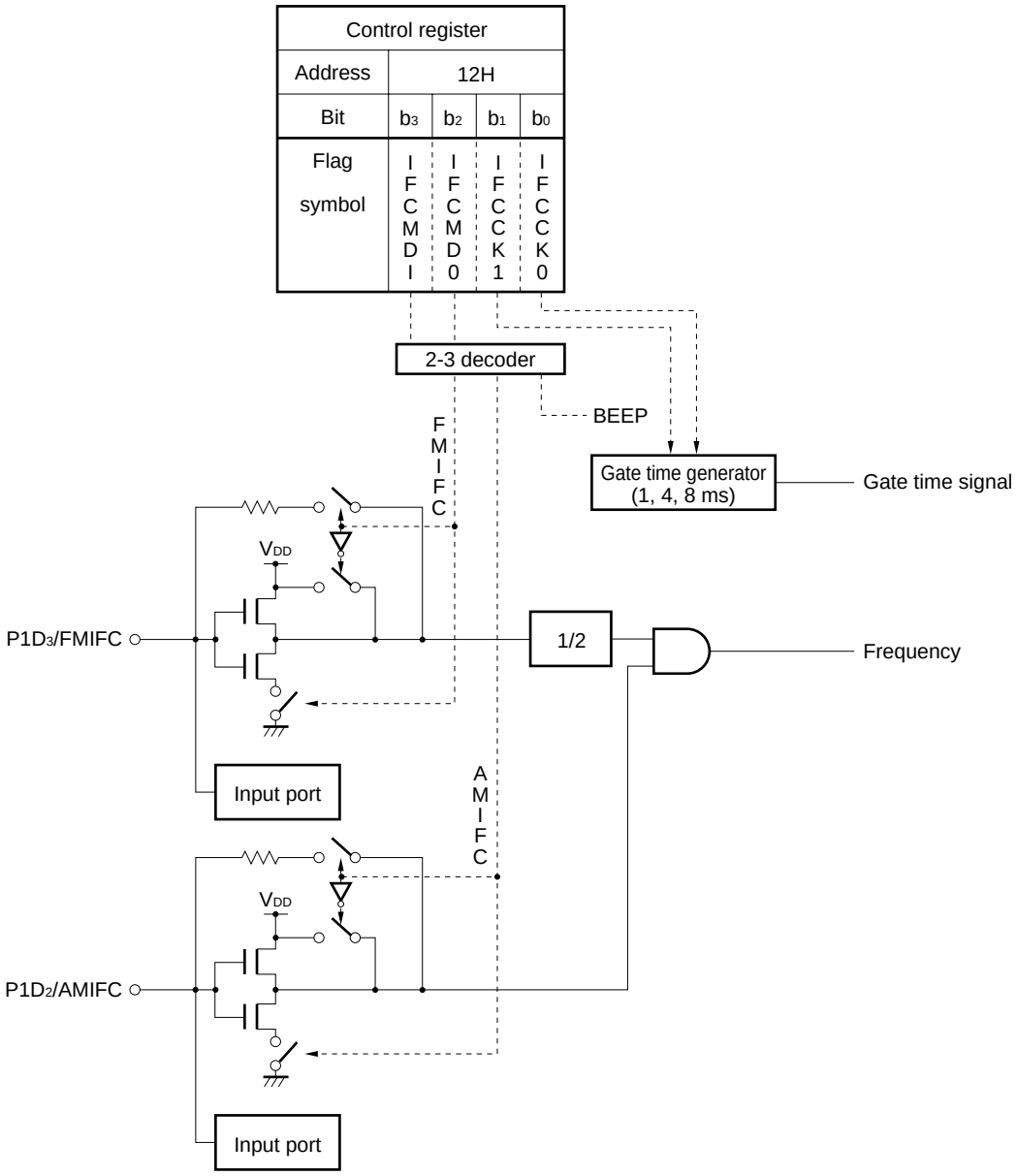
For details, refer to **20.4**.

20.3 Input Select Block and Gate Time Control Block

20.3.1 Configuration of input select block and gate time control block

Figure 20-2 shows the configuration of the input select block and gate time control block.

Figure 20-2. Configuration of Input Select Block and Gate Time Control Block



20.3.2 Function of input select block

The input select block selects whether the P1D3/FMIFIC and P1D2/AMIFC pins are used as general-purpose output port pins or frequency counter pins.

This selection is made by the IFCMD1 and IFCMD0 flags of the IF counter mode select register (refer to **20.3.4**).

20.3.3 Function of gate time control block

The gate time control block sets a gate time (count time).

The gate time is set by the IFCCK1 and IFCCK0 flags of the IF counter mode select register (refer to **20.3.4**).

20.3.4 Configuration and function of IF count mode select register

The IF counter mode select register selects the IF counter function and gate time.

The configuration and function of this register are illustrated below.

Because the frequency counter is shared with the BEEP function, this register is also used to select the BEEP output.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
IF counter mode select register	IFCMD1	IFCMD0	IFCCK1	IFCCK0	12H	R/W

IFCCK1	IFCCK0	Gate time of IF counter
0	0	1 ms
0	1	4 ms
1	0	8 ms
1	1	Open

IFCMD1	IFCMD0	Selects IF counter or BEEP function
0	0	BEEP output
0	1	IF counter (FMIFC)
1	0	IF counter (AMIFC)
1	1	Setting prohibited

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop	0	0	0	0
	CE	Retained			

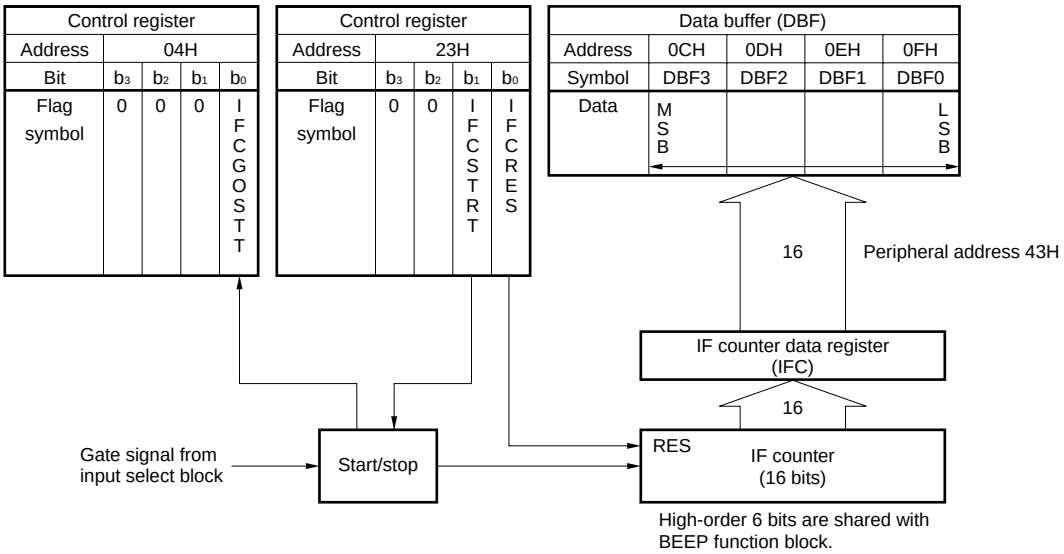
Caution The IF counter function and BEEP function cannot be used at the same time.

20.4 Start/Stop Control Block and IF Counter

20.4.1 Configuration of start/stop control block and counter

Figure 20-3 shows the configuration of the start/stop control block and counter.

Figure 20-3. Configuration of Start/Stop Control Block and Counter



20.4.2 Function of start/stop control block

The start/stop control block detects the start and end of counting by the frequency counter.

The counting is started by using the IFCSTRT flag of the IF counter control register.

To detect the end of the counting, the IFCGOSTT flag of the IF counter gate judge register is used.

The following subsections 20.4.6 and 20.4.7 explain the configuration and functions of the IF counter control register and IF counter gate judge register.

20.4.3 Gate operation of IF counter

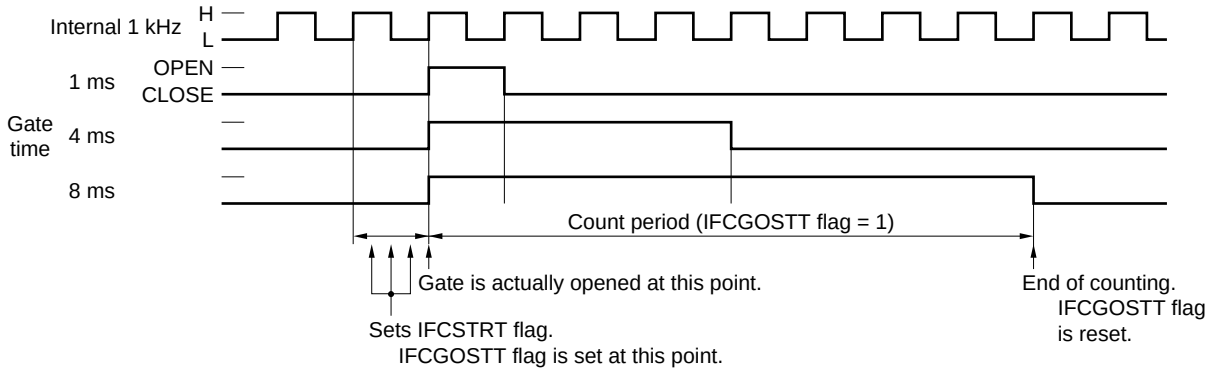
(1) When 1, 4, or 8 ms is selected as gate time

The gate is opened for 1, 4, or 8 ms since the internal 1-kHz signal has risen after the IFCSTRT flag was set to “1”, as shown below.

While this gate is open, the frequency input from the pin is counted by the 16-bit counter.

When the gate is closed, the IFCGOSTT flag is reset.

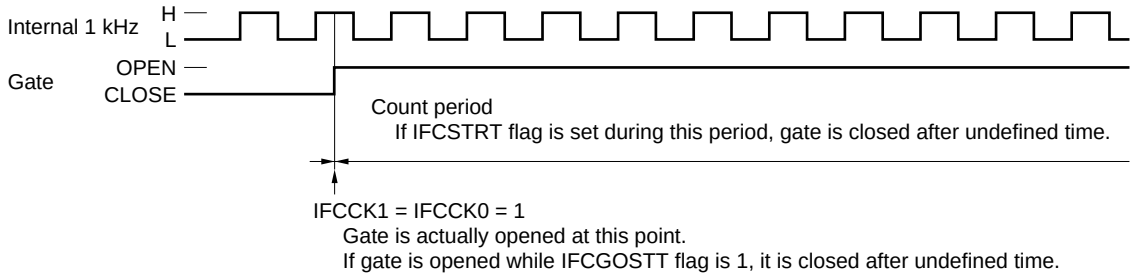
The IFCGOSTT flag is automatically set to “1” when the IFCSTRT flag has been set.



(2) When "open" is selected as gate time

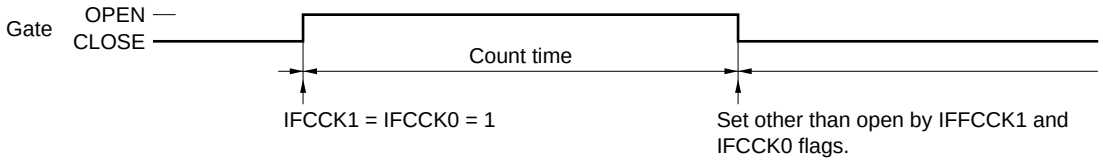
As soon as it has been specified by the IFCCK1 and IFCCK0 flags that the gate is "opened", the gate is opened. If counting is started by the IFCSTRT flag while the gate is open, the gate is closed after an undefined time. Therefore, do not set the IFCSTRT flag to "1" when selecting "open" as the gate time.

However, the counter can be reset by the IFCRES flag.



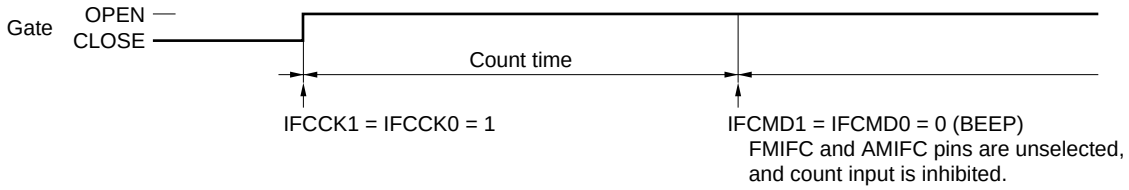
When "open" is selected as the gate time, the gate can be opened or closed in two ways as explained in (a) and (b) below.

(a) Set gate time other than "open" by IFCCK1 and IFCCK0 flags



(b) Unselect pin used by using IFCMD1 and IFCMD0 flags

In this case, the gate is remains open, and counting is stopped by inhibiting input from the pin.



20.4.4 Function and operation of 16-bit counter

The 16-bit counter counts up the frequency input within the specified gate time.

This counter is reset by writing “1” to the IFCRES flag of the IF counter control register.

When the 16-bit counter counts up to FFFFH, it is cleared to 0000H and continues counting when the next input is applied.

Because the high-order 6 bits of this counter are shared with the BEEP function, the frequency counter and BEEP function cannot be used at the same time.

The function of the IF counter is to count the frequency input to the $P1D_3$/FMIFC or $P1D_2$/AMIFC pin while the gate is open.

Note, however, that the frequency input to the $P1D_3$/FMIFC pin is divided by two and is counted.

The relation between count value “x (HEX)” and input frequency (f_{FMIFC} or f_{AMIFC}) is as follows.

FMIFC

$$f_{FMIFC} = \frac{x\ (DEC)}{t_{GATE}} \times 2\ (kHz) \qquad t_{GATE} = \text{gate time (1, 4, or 8 ms)}$$

AMIFC

$$f_{AMIFC} = \frac{x\ (DEC)}{t_{GATE}}\ (kHz) \qquad t_{GATE} = \text{gate time (1, 4, or 8 ms)}$$

The value of the IF counter data register is read via the data buffer.

The following subsection 20.4.5 explains the configuration and function of the IF counter data register.

20.4.5 Configuration and function of IF counter data register (IFC)

The configuration and function of the IF counter data register are illustrated below.

The IF counter data register reads the count value of the frequency counter.

This register is cleared to 0000H at the next input when its value has reached FFFFH, and continues counting.

Name	Data buffer															
Symbol	DBF3				DBF2				DBF1				DBF0			
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	Transfer data															

16

GET can be executed

Nothing is changed even if PUT is executed.

Peripheral register																		
Name	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	Symbol	Peripheral address	Peripheral hardware
IF counter data register	Valid data																IFC	43H	Frequency counter

Count value of frequency counter
0
⋮
x
⋮
$2^{16}-1$ (FFFFH)

IF counter function

- FMIF counter

  Counts rising edge of signal input to $P1D_3$/FMIFC pin by using 1/2 divider

- AMIF counter

  Counts rising edge of signal input to $P1D_2$/AMIFC pin.

The high-order 6 bits of the IF counter are shared with the BEEP function.

Therefore, the frequency counter function and BEEP function cannot be used at the same time.

For details, refer to **20.7 Notes on Using Frequency Counter**.

20.4.6 Configuration and function of IF counter control register

The IF counter control register starts the frequency counter (IF counter) function and controls resetting of the 16-bit counter.

The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
IF counter control register	0	0	IFCSTRT	IFCRES	23H	W

IFCRES	Controls count value of IF counter
0	Nothing is changed.
1	Resets counter.

IFCSTRT	Starts counting by IF counter
0	Nothing is changed.
1	Starts counting.

Fixed to "0"

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop			0	0
	CE			Retained	

The IF counter control register is controlled by writing the contents of the window register by using the “POKE” instruction.

When the contents of this register are read to the window register by the “PEEK” instruction, “0” is read.

If either of the IFCRES or IFCSTRT flag is set by using the assembler (AS17K) embedded macroinstruction SETn, the flag that is not set becomes “undefined”, and an error occurs. For details, refer to **20.7.3 Notes on using embedded macroinstruction SETn to IFCRES and IFCSTRT flags**.

20.4.7 Configuration and function of IF counter gate judge register

The IF counter gate judge register detects the opening and closing of the gate of the IF counter.

The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
IF counter gate judge register	0	0	0	IFCGOSTT	04H	R

IFCGOSTT: Detects opening/closing of gate of frequency counter

- Set to 1 until gate time expires after IFCSTRT flag has been set to "1".

Fixed to "0"

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop	↓	↓	↓	Undefined
	CE	↓	↓	↓	Undefined

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 (i.e., while the gate is open).

20.5 Using IF Counter Function

The following subsections 20.5.1 through 20.5.3 explain how to use the hardware of the IF counter function, program example, and count error.

20.5.1 Using hardware of IF counter

Figure 20-4 shows the block diagram illustrating how the $P1D_3$/FMIFC and $P1D_2$/AMIFC pins are used.

Table 20-1 shows the range of the frequencies that can be input to the $P1D_3$/FMIFC and $P1D_2$/AMIFC pins.

Because the input pins of the IF counter have an internal amplifier, cut off the DC component of the input signal by using capacitor C as shown in Figure 20-4.

When the $P1D_3$/FMIFC or $P1D_2$/AMIFC pin is selected as the IF counter pin, switch SW turns ON, applying a voltage of about $1/2V_{DD}$ to each pin.

If the voltage has not risen to a sufficient intermediate level at this time, the AC amplifier does not operate normally, and consequently, the IF counter does not correctly operate.

Therefore, make sure that a sufficiently long wait time elapses from the time each pin is selected as an IF counter until counting is started.

Figure 20-4. Function of Each IF Counter Pin

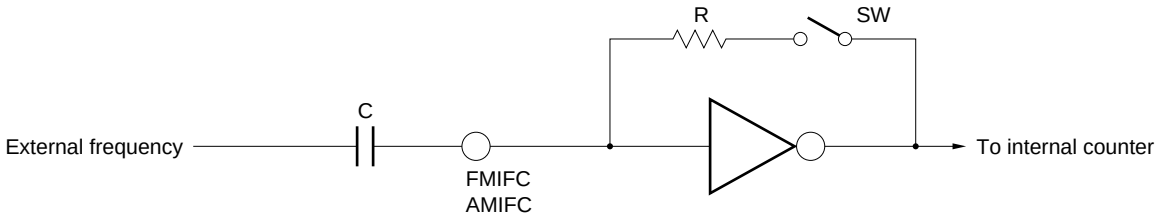


Table 20-1. Input Frequency Range of IF Counter

Input Pin	Input Frequency (MHz)	Input Amplitude (V_{p-p})
$P1D_3$/FMIFC	5-15	0.3
	10.5-10.9	0.06
$P1D_2$/AMIFC	0.1-1	0.3
	0.44-0.46	0.05

20.5.2 Program example of IF counter function

A program example of the IF counter function is shown below.

As shown in this example, wait time must elapse after an instruction that selects the P1D3/FMIFC or P1D2/AMIFC pin as the IF counter pin has been executed before counting is started.

This is because the internal AC amplifier may not operate normally immediately after each pin has been selected, as explained in 20.5.1.

Example To count frequency on P1D3/FMIFC pin (gate time: 8 ms)

```
INITFLG  NOT  IFCMD1, IFCMD0, IFCCK1, NOT IFCCK0
                              ; Selects FMIFC pin and sets gate time to 8 ms.
    Wait                      ; Internal AC amplifier stabilization time

    IFC_RES_AND_START         ; Resets and starts counter (sets IFCRES and
                              ; IFCSTRT flags)Note.
                              ; Macro definition is used because SET2 instruction cannot
                              ; be used.
LOOP:
    SKT1     IFCGOSTT         ; Detects opening/closing of gate.
    BR       READ             ; Branches to READ: when gate is closed.

    Processing A              ; Do not read data of IF counter by this processing A.
                              ; Do not select BEEP function.
    BR       LOOP
READ:
    GET      DBF, IFC         ; Reads value of IF counter data register to data buffer.
```

Note Refer to **20.7.3 Notes on using embedded macroinstruction SETn to IFCRES and IFCSTRT flags**.

20.5.3 Error of IF counter

The IF counter may have a gate time error and a count error.

These errors are explained in (1) and (2) below.

(1) Error of gate time

The gate time of the IF counter is created by dividing the 4.5-MHz system clock.

Therefore, if the system clock deviates “+x” ppm, the gate time deviates “–x” ppm.

(2) Count error

The IF counter counts the frequency at the rising edge of an input signal.

If a high level is input to the pin when the gate is opened, therefore, one excess pulse is counted.

However, counting is not performed because of the status of the pin when the gate is closed.

Therefore, a count error of “+1, –0” may occur.

20.6 Status on Reset

20.6.1 On power-ON reset

The P1D3/FMIFC and P1D2/AMIFC pins are set in the general-purpose input port mode.

20.6.2 On execution of clock stop instruction

The P1D3/FMIFC and P1D2/AMIFC pins are set in the general-purpose input port mode.

20.6.3 On CE reset

The P1D3/FMIFC and P1D2/AMIFC pins retain the previous status.

20.6.4 In halt status

The P1D3/FMIFC and P1D2/AMIFC pins retain the status immediately before the halt status is set.

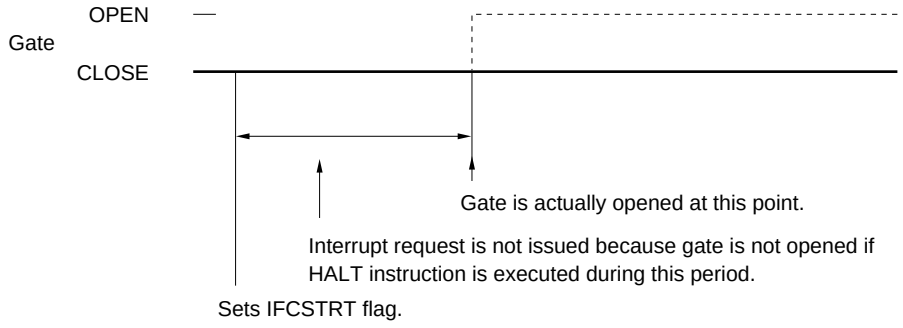
To release the halt status by using the interrupt of the frequency counter, the following point must be noted.

Caution If the "HALT" instruction is executed after counting has been started by the IFCSTRT flag and before the gate is actually opened, the gate will not be opened.
Therefore, wait at least 1 ms after starting counting before executing the "HALT" instruction.

Figure 20-5 shows the gate operation when the "HALT" instruction is executed.

Because the closing of the gate cannot be detected unless the gate is opened, the interrupt request is not issued. If interrupts other than that of the IF counter are not enabled with halt release conditions other than the interrupt not specified, the HALT status will not be released.

Figure 20-5. Gate Operation When "HALT" Instruction Is Used



20.7 Notes on Using Frequency Counter

The frequency counter shares its hardware with the BEEP function explained earlier.

Therefore, the BEEP function and frequency counter cannot be used at the same time.

If the data of the IF counter mode select register and IF counter data register are manipulated while the BEEP function is used, the operation explained in 20.7.1 is performed.

If the BEEP output software macro is executed when the frequency counter is used, the operation explained in 20.7.2 is performed.

The assembler embedded macroinstruction SETn cannot be used to manipulate the IFCRES and IFCSTRT flags, as explained in 20.4.6. This is explained in more detail in 20.7.3.

20.7.1 When BEEP output is used

(1) When IFCMD1 and IFCMD0 flags of IF counter mode select register are manipulated

If a value other than “0” is written to the IFCMD1 and IFCMD0 flags, the $P1B_0$/BEEP pin retain the current output level, and stops the BEEP output.

When both the IFCMD1 and IFCMD0 flags are reset to “0” again, the BEEP output is started.

(2) When IF counter data register is manipulated

The BEEP output is not affected even if a read (GET) or write (PUT) instruction is executed.

When the IF counter data register is read, an “undefined” value is read. Nothing is changed even if data is written to the register.

Note, however, that the IF counter data register is a read-only peripheral register. Do not write data to this register. If the write instruction “PUT IFC, DBF” is executed, the 17 series assembler outputs an error.

20.7.2 When frequency counter is used

If the BEEP output software macro is executed while the frequency counter is used, the frequency counter is forcibly stopped, and the BEEP output is started. At this time, the contents of the window register (WR) and data buffer (DBF) are destroyed. To operate the frequency counter again, once stop the BEEP output by using the BEEPOFF macro, and start counting from the beginning.

20.7.3 Notes on using embedded macroinstruction SETn to IFCRES and IFCSTRT flags

Because the IFCRES and IFCSTRT flags are in a write-only register, an “undefined” value is read if these flags are read.

If either of these flags is set by assembler (AS17K) embedded macroinstruction “SETn”, an “undefined” value is written to the other flag.

```
SET1  IFCRES
        PEEK   WR,.MF.IFCRES SHR4        ; Value of WR is “undefined” at this time.
        OR     WR,#.DF.IFCRES AND 0FH    ; Only bit 0 of WR is set.
        POKE   .MF.IFCRES SHR 4, WR      ; Undefined value is written to IFCSTRT (bit 1).
```

If the SETn instruction is executed to the IFCRES and IFCSTRT flag, therefore, the assembler outputs an error.

Therefore, use macro definition to manipulate the IFCRES and IFCSTRT flags as follows.

```
IFCW  MEM 0.0A3H
;
IFC_RES_AND_START     MACR0
      MOV    WR,  #0011B
      POKE   IFCW, WR
ENDM
;
IFC_RES               MACR0
      MOV    WR,  #0001B
      POKE   IFCW, WR
ENDM
;
IFC_START             MACR0
      MOV    WR,  #0010B
      POKE   IFCW, WR
ENDM
```

The above macroinstruction is supplied with the device file as “IFCSET.LIB” file and “D17016.INC/D17017.INC” file.

Therefore, the above macroinstruction can be used as is by INCLUDEing these files on the source program.

For how to use each file, refer to **AS17016 Device File User’s Manual**.

21. LCD CONTROLLER/DRIVER

The LCD (Liquid Crystal Display) controller/driver can display an LCD of up to 60 dots by outputting segment and common signals in combination.

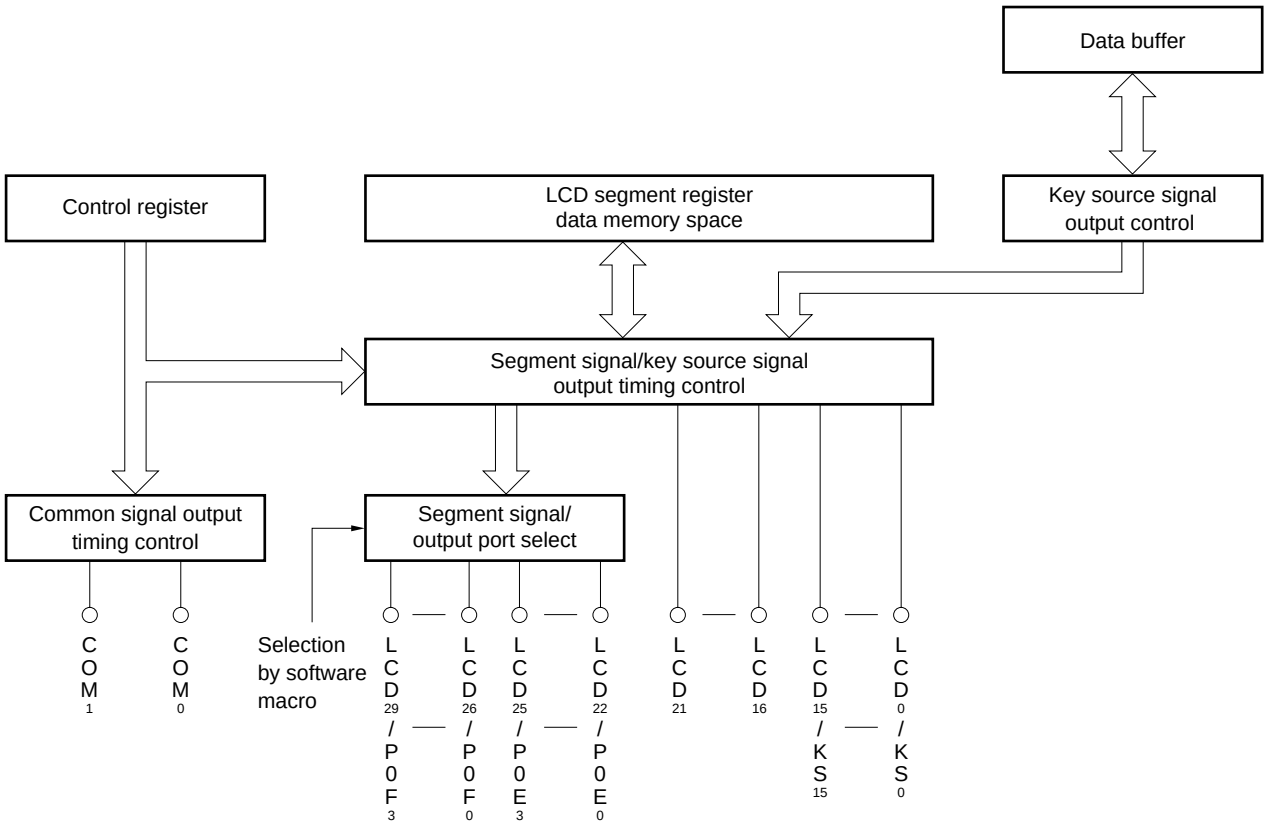
21.1 Configuration of LCD Controller/Driver

Figure 21-1 shows the block diagram of the LCD controller/driver.

As shown in the figure, the LCD controller/driver consists of a common signal output timing control block, segment signal/key source signal output timing control block, segment signal/output port select block, LCD segment register, and key source signal output control block.

The following section 21.2 outlines the function of each block.

Figure 21-1. Block Diagram of LCD Controller/Driver



21.2 Functional Outline of LCD Controller/Driver

The LCD controller/driver can display up to 60 dots by using the combination of common signal output pins (COM_1 and COM_0) and segment signal output pins ($LCD_{29}/P0F_3$ through LCD_0/KS_0).

Figure 21-2 shows the relation among common signal output pins, segment signal output pins, and display dots.

As shown in this figure, two dots at the intersections between one segment line with COM_1 and COM_0 pins.

The driving mode is 1/2 duty, 1/2 bias, and the drive voltage is supply voltage V_{DD}.

The segment signal output pins ($LCD_{29}/P0F_3$ through $LCD_{22}/P0E_0$) pins can also be used as general-purpose output port pins.

When these pins are used as general-purpose output port pins, ports 0F ($LCD_{29}/P0F_3$ through $LCD_{26}/P0F_0$) and 0E ($LCD_{25}/P0E_3$ through $LCD_{22}/P0E_0$) can be independently used.

Of the segment signal output pins, the LCD_{15}/KS_{15} through LCD_0/KS_0 pins are also used as key source signal output pins.

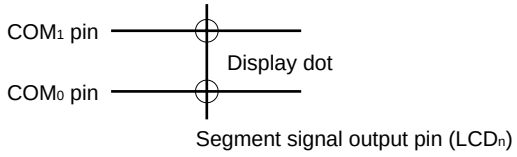
The key source signals and LCD segment signals are output by means of time-division multiplexing.

For the details on the general-purpose output ports, refer to **15. GENERAL-PURPOSE PORT**.

For the details on the key source signals, refer to **22. KEY SOURCE CONTROLLER/DECODER**.

The following subsections 21.2.1 through 21.2.5 outlines the function of each block of the LCD controller/driver.

Figure 21-2. Common Signal Output, Segment Signal Output, and Display Dot



21.2.1 LCD segment register

The LCD segment register sets dot data that is used to turn ON/OFF the LCD.

Because this register is mapped on the data memory, it can be controlled by any data memory manipulation instruction.

When the segment signal output pins are used as general-purpose output port pins, this registers sets output data.

For details, refer to **21.3**.

21.2.2 Common signal output timing control block

The common signal output timing control block controls the common signal output timing of the COM_1 and COM_0 pins.

These pins output a low level when the LCD is not displayed.

Whether the LCD is displayed or not is selected by the LCD mode select register (RF address 10H).

For details, refer to **21.4**.

21.2.3 Segment signal/key source signal output timing control block

The segment signal/key source signal output timing control block controls the segment signal output timing of the $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins.

These pins output a low level when the LCD is not displayed.

Whether the LCD is displayed or not is selected by the LCD mode select register.

The segment signal/key source signal output timing control block controls the timing of the segment and key source signals output from the LCD_{15}/KS_{15} through LCD_0/KS_0 pins.

Whether the key source signals are used or not is selected by the LCD mode select register.

For details, refer to **21.4**.

21.2.4 Segment signal/general-purpose port select block

The segment signal/general-purpose port select block selects whether each segment signal output pin is used for LCD display (to output a segment signal) or as a general-purpose output port pin.

This selection is made by using the software macro supplied with the device file.

For details, refer to **21.4**.

21.2.5 Key source signal output control block

The key source signal output control block sets the key source output data that are output from the LCD_{15}/KS_{15} through LCD_0/KS_0 pins and detects key input timing.

The key source signal output data are set by the key source data register (KSR: peripheral address 42H) via the data buffer.

To use the key source signals, the $P0D_3/ADC_5$ through $P0D_0/ADC_2$ pins are used as key input pins.

For details, refer to **22. KEY SOURCE CONTROLLER/DECODER**.

21.3 LCD Segment Register

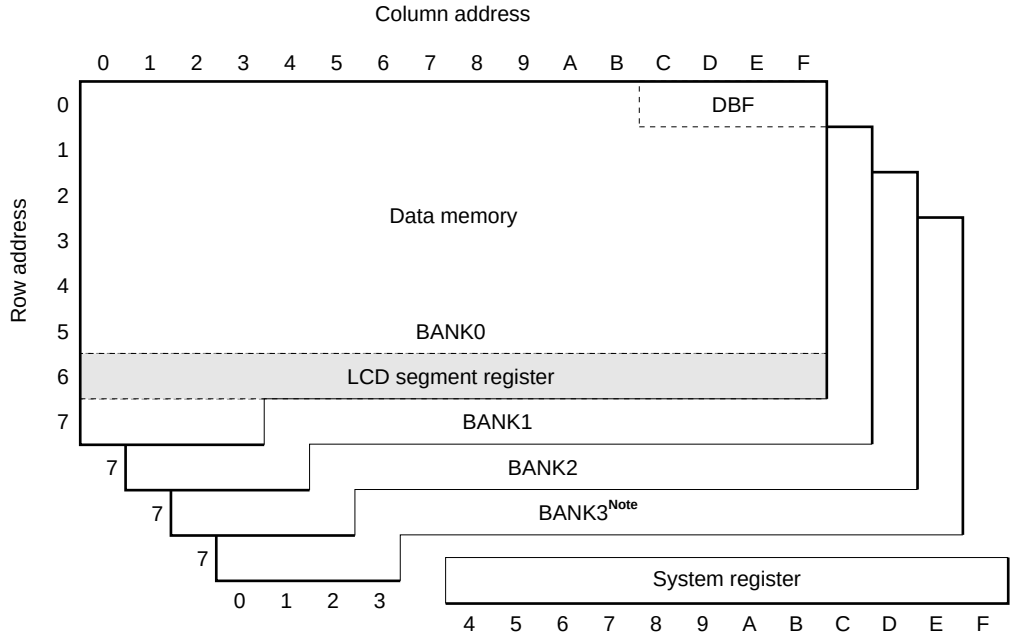
The LCD segment register specifies whether each dot on the LCD is turned ON/OFF.

21.3.1 Configuration of LCD segment register

Figure 21-3 shows the location of the LCD segment register on the data memory.

Figure 21-4 shows the configuration of the LCD segment register.

Figure 21-3. Location of LCD Segment Register on Data Memory



Note BANK3 is not provided on the μPD17016.

Figure 21-4. Configuration of LCD Segment Register

LCD segment register																
Address	60H	61H	62H	63H	64H	65H	66H	67H	68H	69H	6AH	6BH	6CH	6DH	6EH	6FH
Symbol	LCDD0	LCDD1	LCDD2	LCDD3	LCDD4	LCDD5	LCDD6	LCDD7	LCDD8	LCDD9	LCDD10	LCDD11	LCDD12	LCDD13	LCDD14	–

Nothing is allocated to this address.
This address cannot be used as a data memory area.

LCDD0			
b_3	b_2	b_1	b_0

21.3.2 Function of LCD segment register

Figure 21-5 shows the relation of 1 nibble (4 bits) of the LCD segment register and LCD display dots.

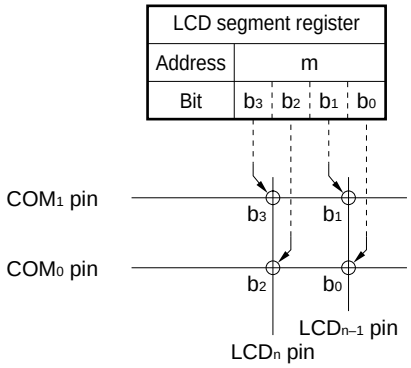
As shown in this figure, 1 nibble of the LCD segment register can set display data (ON/OFF data) of 4 dots.

The LCD display dot corresponding to a bit that is set to “1” is turned ON, and the dot corresponding to a bit that is reset to “0” is turned OFF.

The LCD segment register also sets output data when the segment signal output pins are used as output port pins.

Figure 21-6 shows the relation between each LCD segment register and LCD display dots that are turned ON/OFF.

Figure 21-5. Relation of 1 Nibble of LCD Segment Register and LCD Display Dots



[MEMO]

Figure 21-6. Relation among LCD Display Dots, Ports 0E and 0F, Key Source Output, and Each Data Setting Register (1/2)

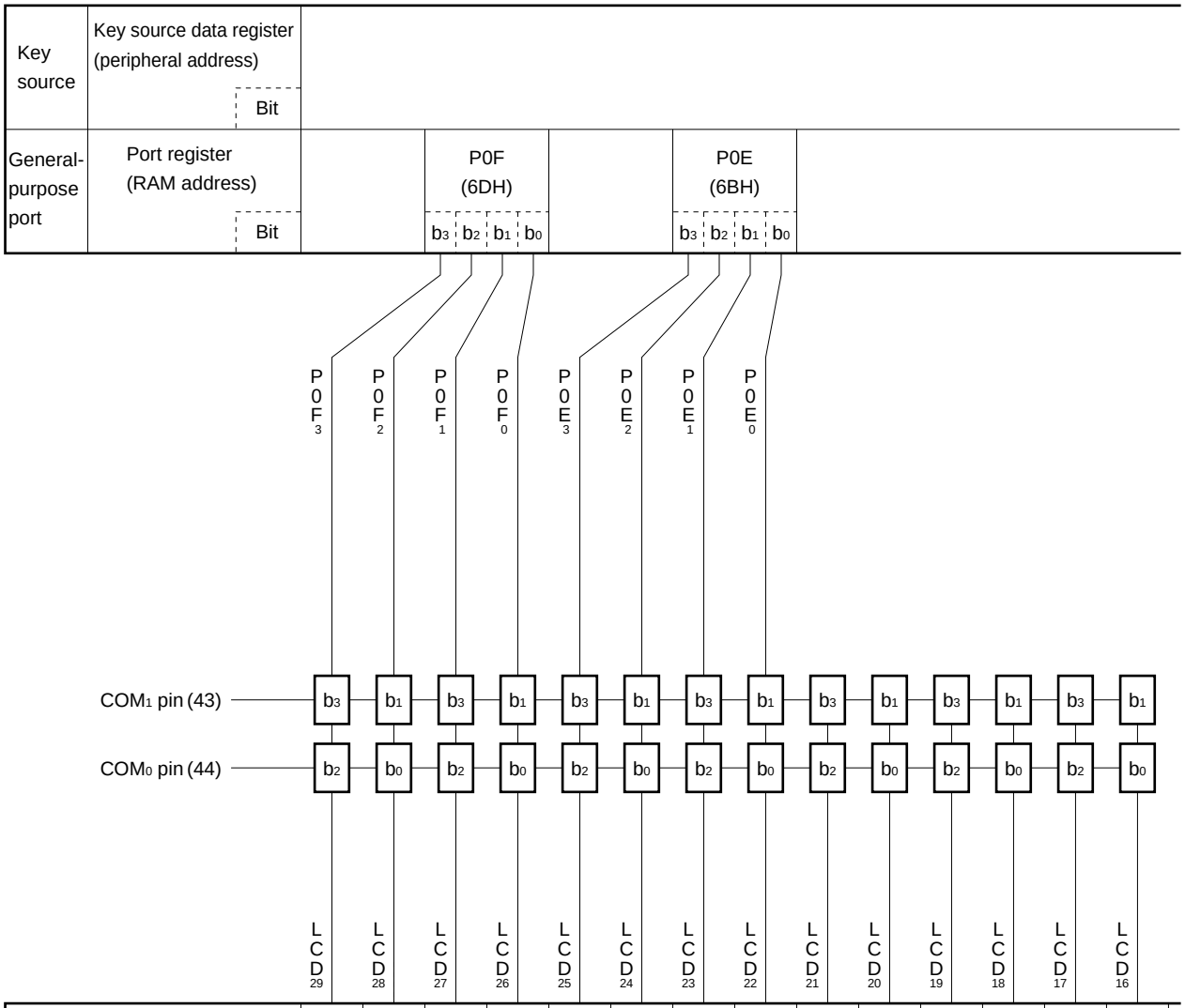
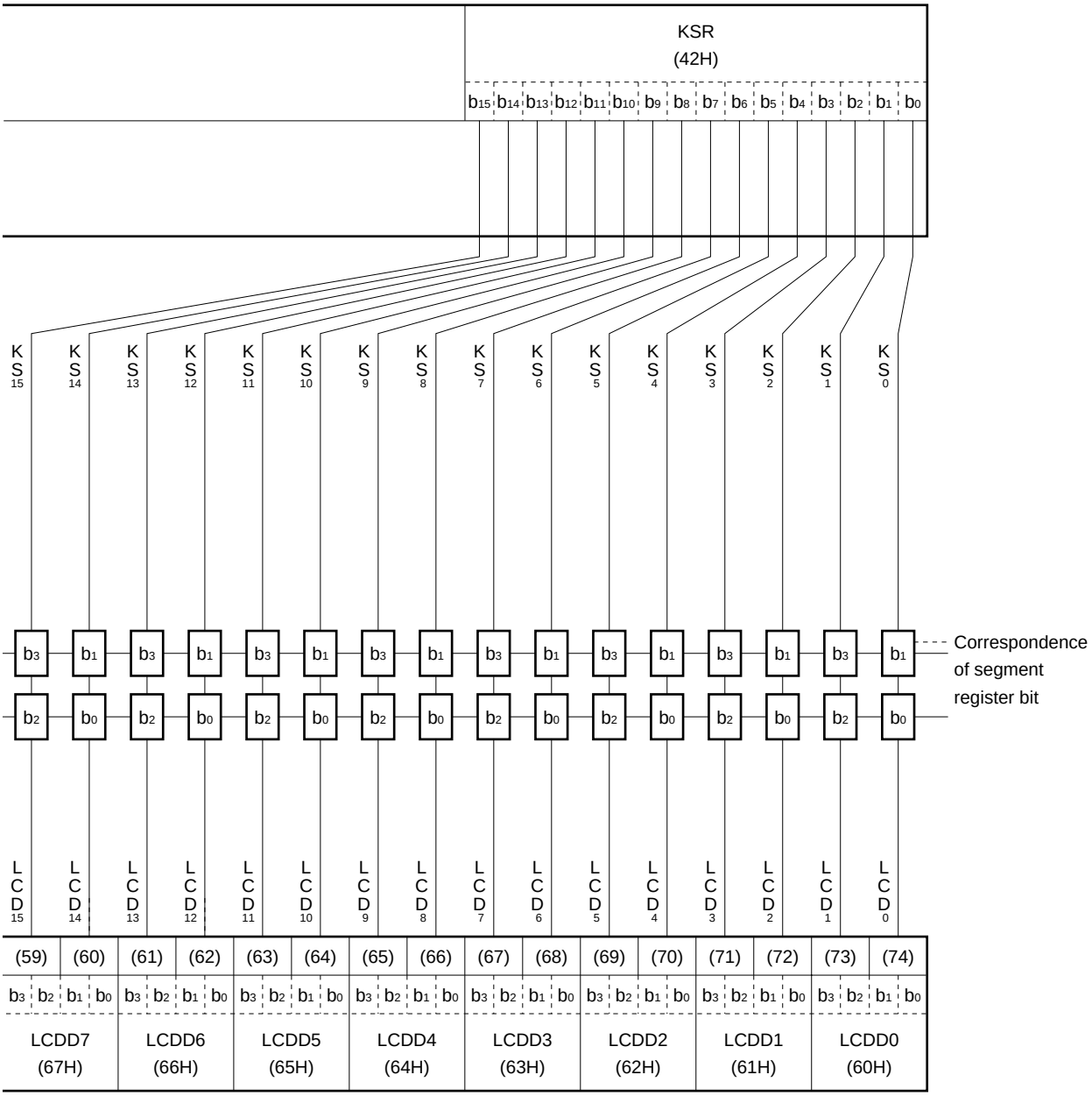


Figure 21-6. Relation among LCD Display Dots, Ports 0E and 0F, Key Source Output, and Each Data Setting Register (2/2)

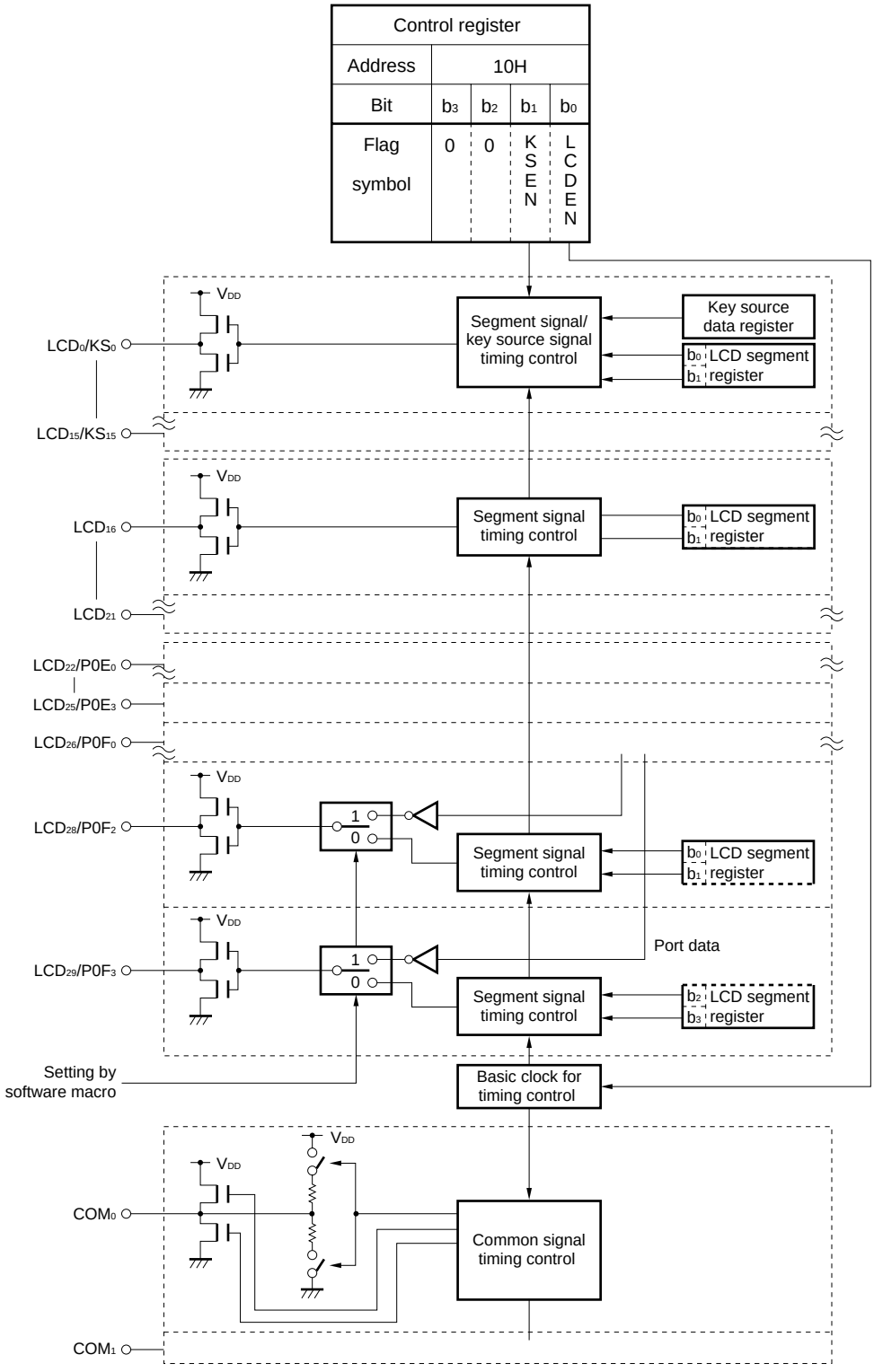


21.4 Output Timing Control Blocks and Segment/Port Select Block

21.4.1 Configuration of output timing control blocks and segment/port select block

Figure 21-7 shows the configuration of the common signal and segment signal/key source signal output timing control block and segment signal/general-purpose output port select block.

Figure 21-7. Configuration of Timing Control Blocks and Port Select Block



21.4.2 Function of segment signal/general-purpose output port select block

The segment signal/general-purpose output port select block specifies whether each pin is used as a segment signal output pin or a general-purpose output port pin, by using the software macro supplied with the device file (refer to **Table 21-1. Software Macros Setting Ports (P0E and P0F)**).

For the details on the general-purpose output port, refer to **15. GENERAL-PURPOSE PORT**.

Table 21-1. Software Macros Setting Ports (P0E and P0F)

Function	Macro Format
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as general-purpose output port	SET1_P0EON
Uses $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port	SET1_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port	SET2_P0EON_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as LCD segment signal output pins	CLR1_P0EON
Uses $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins	CLR1_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins	CLR2_P0EON_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as LCD segment signal output pins and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port	INIT_NOT_P0EON_P0FON
Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as general-purpose output port and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins	INIT_P0EON_NOT_P0FON

Caution If the above macros are used, the contents of the window register are destroyed.
If the above macros are immediately follow an embedded macro instruction, an "object error" occurs when the source file is assembled and loaded to the in-circuit emulator. If an embedded macro is used before the above macros, insert a comment statement in between them.

21.4.3 Functions of output timing control blocks

The output timing control blocks control the timing of the common and segment signals for LCD display and the timing of the key source and segment signals when the key source controller/decoder is used.

The common and segment signals are output when the LCDEN flag of the LCD mode select register is "1". In other words, all the LCD display dots can be turned OFF by the LCDEN flag.

When all the LCD display dots have been turned OFF, the common and segment signals output a low level.

The key source signals are output when the KSEN flag of the LCD mode select register is "1".

Therefore, use of the key source signal can be specified by the KSEN flag.

21.4.4 explains the configuration and function of the LCD mode select register.

21.4.5 shows the output waveforms of the common and segment signals.

For the details on the key source controller/decoder, refer to **22. KEY SOURCE CONTROLLER/DECODER**.

21.4.4 Configuration and function of LCD mode select register

The LCD mode select register turns ON/OFF all the LCD display dots, and specifies output of the key source signals.

The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
LCD mode select register	0	0	KSEN	LCDEN	10H	R/W

Turns ON/OFF all LCD display dots	
0	Display OFF (all segment and common signals are low)
1	Display ON

Sets output of key source signal	
0	Key source OFF
1	Key source ON

Fixed to "0"

On reset	Power-ON	0	0	0	0
	Clock stop			0	0
	CE			Retained	

21.4.5 Output Waveforms of Common and Segment Signals

Figures 21-8 and 21-9 show the output waveforms of the common and segment signals.

Figure 21-8 shows the output waveform with the key source signals not output, and Figure 21-9 shows the output waveform with the key source signals output.

As shown in Figure 21-8, the LCD driver outputs signals with a frame frequency of 125 Hz at 1/2 duty, 1/2 bias (voltage average mode).

As the common signals, three levels of voltages (0, 1/2 V_{DD}, and V_{DD}) each having a phase difference of 1/4 from the others are output from the COM_1 and COM_0 pins.

Therefore, voltages in a range of $1/2V_{DD} \pm 1/2\ V_{DD}$ are output. This display mode is called 1/2 bias drive mode.

As the segment signals, two levels (0, V_{DD}) of voltages each having a phase corresponding to a display dot are output from each segment signal output pin.

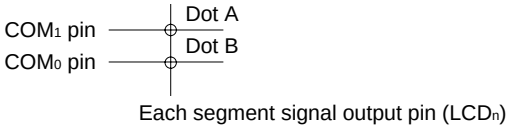
Because two display dots (A and B) are turned ON/OFF by one segment pin as shown in Figure 21-8, four types of phases <1> through <4> shown in Figure 21-8 are output by combination of dots A and B, and ON and OFF.

Dots A and B are turned ON when the potential difference between the common and segment signals reaches V_{DD}.

The duty factor at which each of dots A and B is turned ON is 1/2, and the frequency of the LCD clock is 250 Hz.

This display mode is called 1/2 duty drive mode, and the frame frequency is 125 Hz.

Figure 21-8. Output Waveforms of Common and Segment Signals (with key source signals not output)



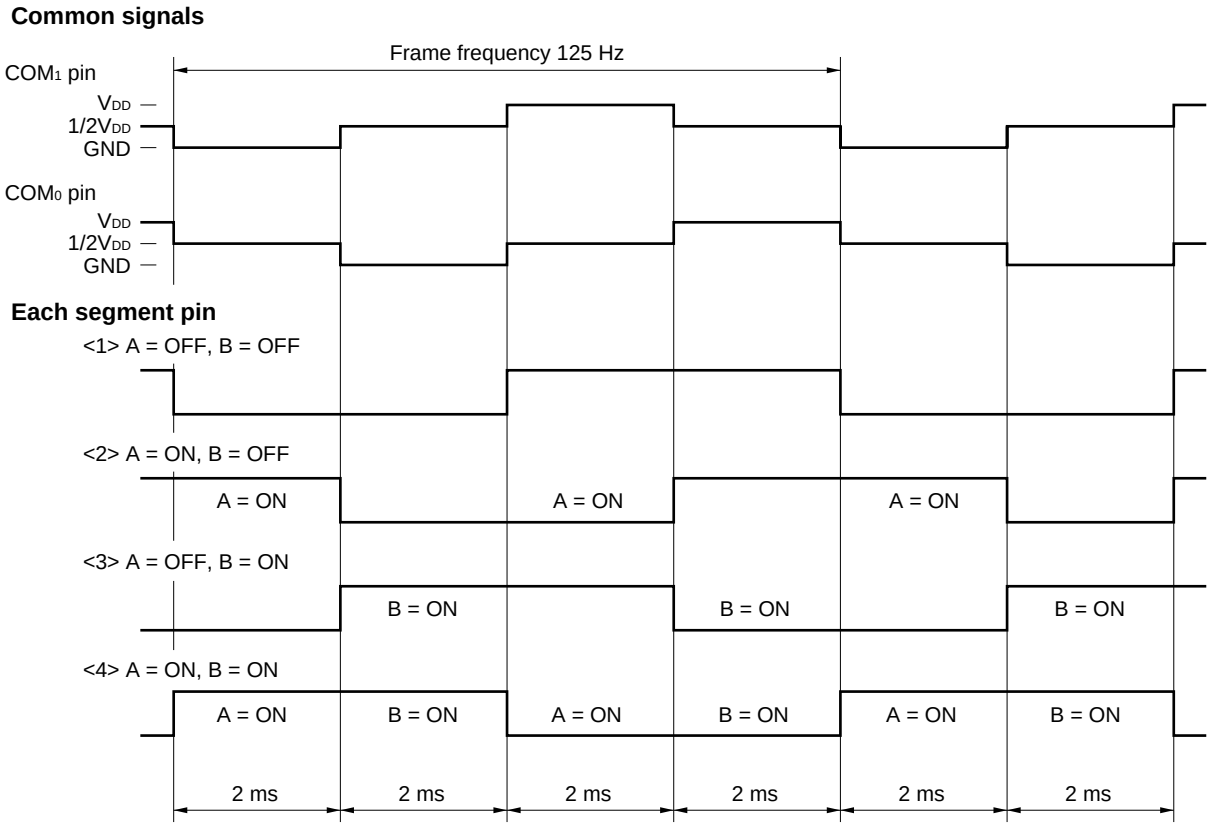
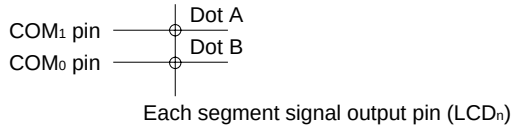
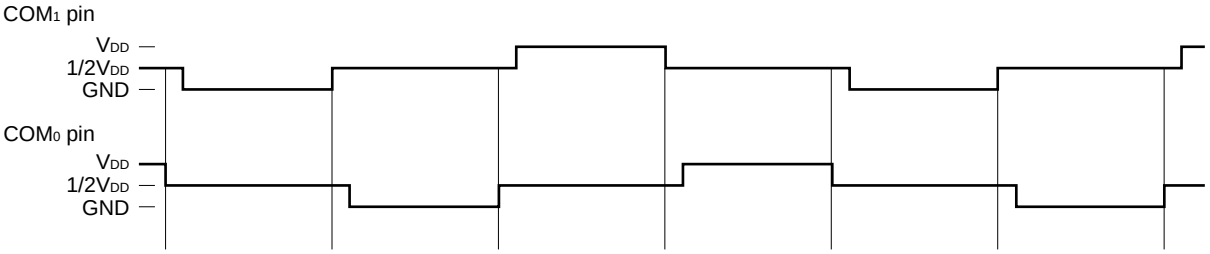


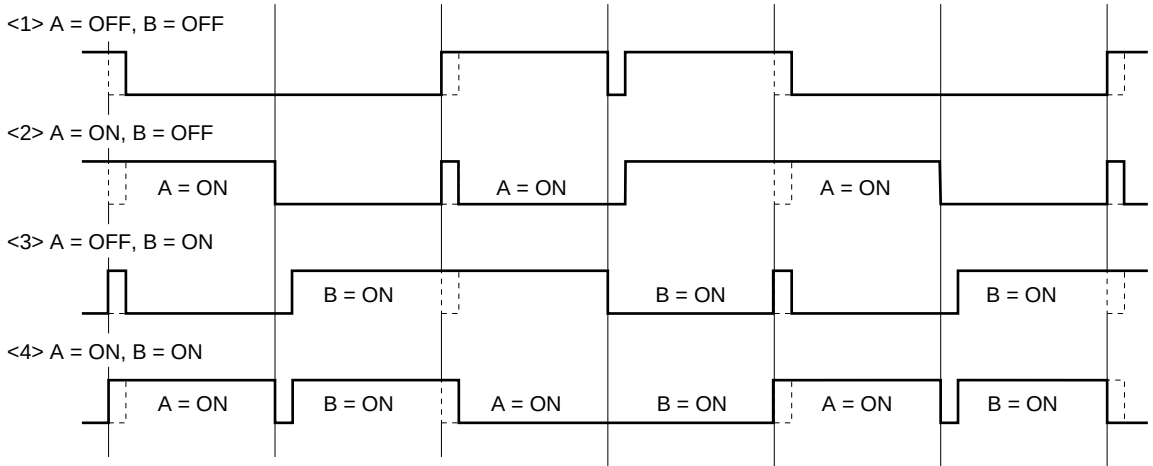
Figure 21-9. Output Waveforms of Common and Segment Signals (with key source signals output)



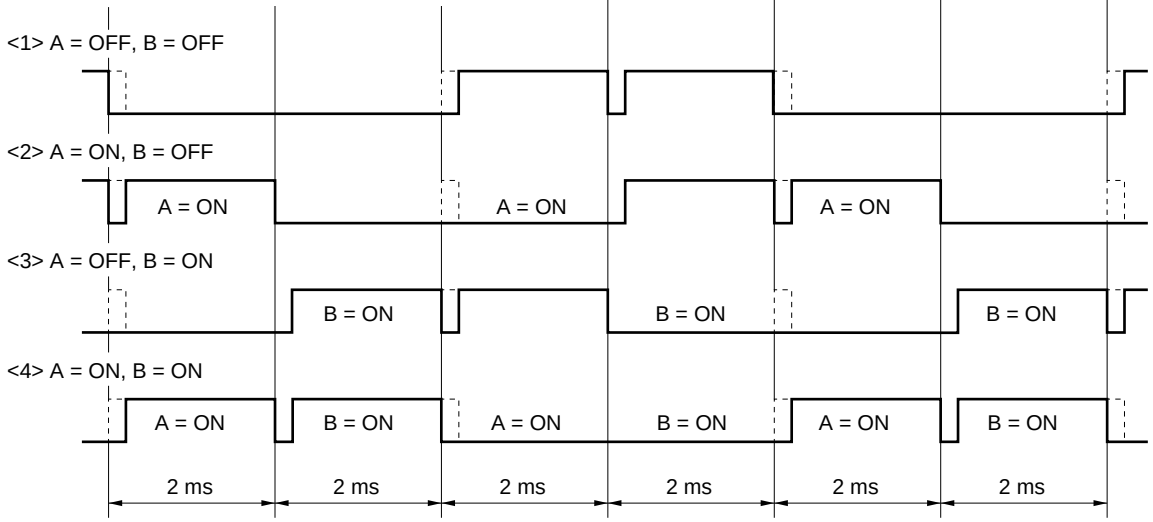
Common signals



Each segment pin (pin outputting "1" as key source)



Each segment pin (pin outputting "0" as key source)



21.5 Using LCD Controller/Driver

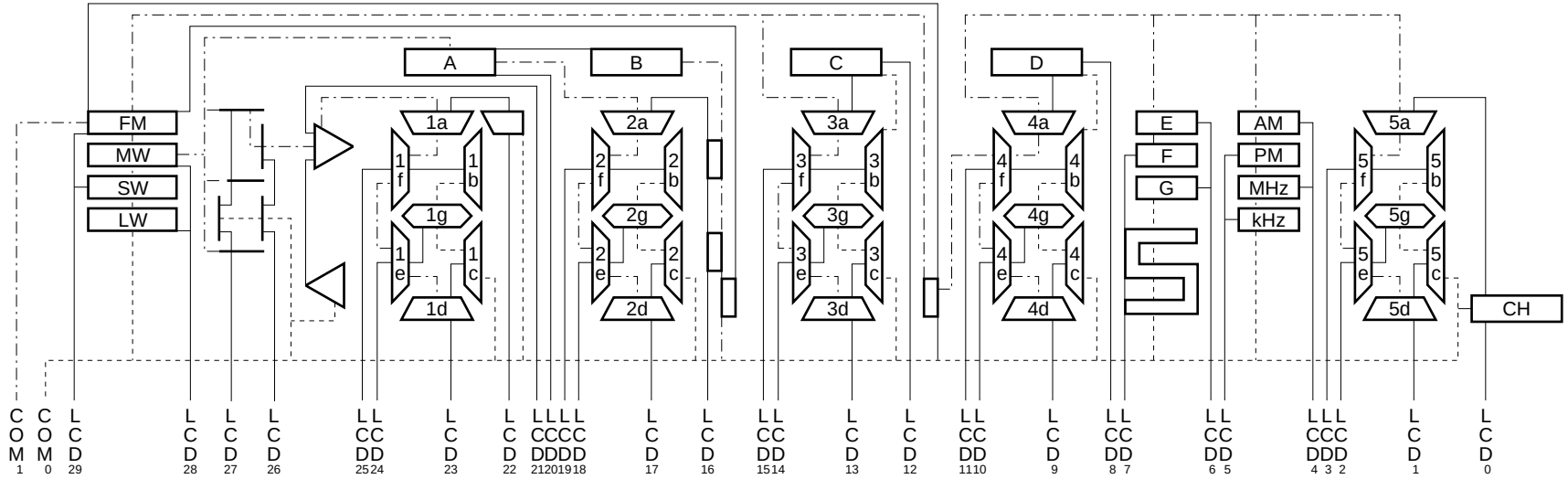
Figure 21-10 shows an example of wiring an LCD panel.

An example of a program that turns ON a 7-segment LCD panel by using the LCD_0 through LCD_3 pins as shown in Figure 21-10 is shown below.

Example

```
    PMN0    MEM  0.01H     ; Preset memory number and BK data storage area
    CH      FLG  DBF0.1    ; Symbol definition of least significant bit of DBF as "CH" display flag
LCDDATA:                   ; Display table data
;                          b3 b2 b1 b0 b3 b2 b1 b0      ; Corresponds to LCD segment register
;                          f  b  e  g  d  c  a  –       ; Corresponds to LCD group register
    DW      0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 B    ; BLANK
    DW      0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 B    ; 1
    DW      0 0 0 0 0 0 0 0 0 1 1 1 1 0 1 0 B    ; 2
    DW      0 0 0 0 0 0 0 0 0 1 0 1 1 1 1 0 B    ; 3
    DW      0 0 0 0 0 0 0 0 1 1 0 1 0 1 0 0 B    ; 4
    DW      0 0 0 0 0 0 0 0 1 0 0 1 1 1 1 0 B    ; 5
    DW      0 0 0 0 0 0 0 0 1 0 1 1 1 1 1 0 B    ; 6
    DW      0 0 0 0 0 0 0 0 1 1 0 0 0 1 1 0 B    ; 7
    DW      0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 B    ; 8
    DW      0 0 0 0 0 0 0 0 1 1 0 1 1 1 1 0 B    ; 9
    DW      0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 B    ; A
    DW      0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 0 B    ; B
    DW      0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 B    ; C
    DW      0 0 0 0 0 0 0 0 1 1 1 0 1 1 1 0 B    ; D
    DW      0 0 0 0 0 0 0 0 1 0 1 1 1 0 1 0 B    ; E
    DW      0 0 0 0 0 0 0 0 1 0 1 1 0 0 1 0 B    ; F

    CLR2_P0EON_P0FON    ; Software macro
    MOV     RPL, #1110B
    MOV     AR3, #.DL.LCDDATA SHR 12 AND 0FH
    MOV     AR2, #.DL.LCDDATA SHR 8  AND 0FH
    MOV     AR1, #.DL.LCDDATA SHR 4  AND 0FH
    MOV     AR0, #.DL.LCDDATA        AND 0FH
    ADD     AR0, PMN0
    ADDC    AR1,   #0
    ADDC    AR2,   #0
    ADDC    AR3,   #0
    MOVT    DBF,   @AR
    MOV     RPH, #0000B
    MOV     RPL, #0000B
    SKGE    PMN0,  #0AH
    SET1    CH
    LD      LCDD1, DBF1
    LD      LCDD0, DBF0
    SET1    LCDEN
```

Figure 21-10. Example of Wiring of LCD Panel



Correspondence between segment and common pins on LCD panel

Segment Pin / Common Pin		LCD 29	LCD 28	LCD 27	LCD 26	LCD 25	LCD 24	LCD 23	LCD 22	LCD 21	LCD 20	LCD 19	LCD 18	LCD 17	LCD 16	LCD 15	LCD 14	LCD 13	LCD 12	LCD 11	LCD 10	LCD 9	LCD 8	LCD 7	LCD 6	LCD 5	LCD 4	LCD 3	LCD 2	LCD 1	LCD 0
		45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74
COM_1	43	FM	MW	⌐	⌐	1f	1e	1d	1a	▷	A	2f	2e	2d	2a	3f	3e	3d	3a	4f	4e	4d	4a	F	E	PM	AM	5f	5e	5d	5a
COM_0	44	SW	LW	⌐	⌐	1b	1g	1c	⊢	◁	B	2b	2g	2c	:	3b	3g	3c	C	4b	4g	4c	D	5	G	kHz	MHz	5b	5g	5c	CH

21.6 Status on Reset

21.6.1 On power-ON reset

The $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins are specified as LCD segment signal output pins, and output a low level.

The COM_1 and COM_0 pins output a low level.

Therefore, the LCD display is turned OFF.

21.6.2 On execution of clock stop instruction

The $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins are specified as LCD segment signal output pins, and output a low level.

The COM_1 and COM_0 pins output a low level.

Therefore, the LCD display is turned OFF.

21.6.3 On CE reset

Of the $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins, those that are specified as segment signal output pins output segment signals, and those that are specified as general-purpose output port pins retain the current output value.

The COM_1 and COM_0 pins output common signals.

21.6.4 In halt status

Of the $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins, those that are specified as segment signal output pins output segment signals, and those that are specified as general-purpose output port pins retain the current output value.

The COM_1 and COM_0 pins output common signals.

22. KEY SOURCE CONTROLLER/DECODER

The key source controller/decoder can configure a key matrix of up to 64 keys by outputting key source signals from the pins that also output LCD segment signals. Whether key source signals or LCD segment signals are output from the pins is controlled by means of time-division multiplexing.

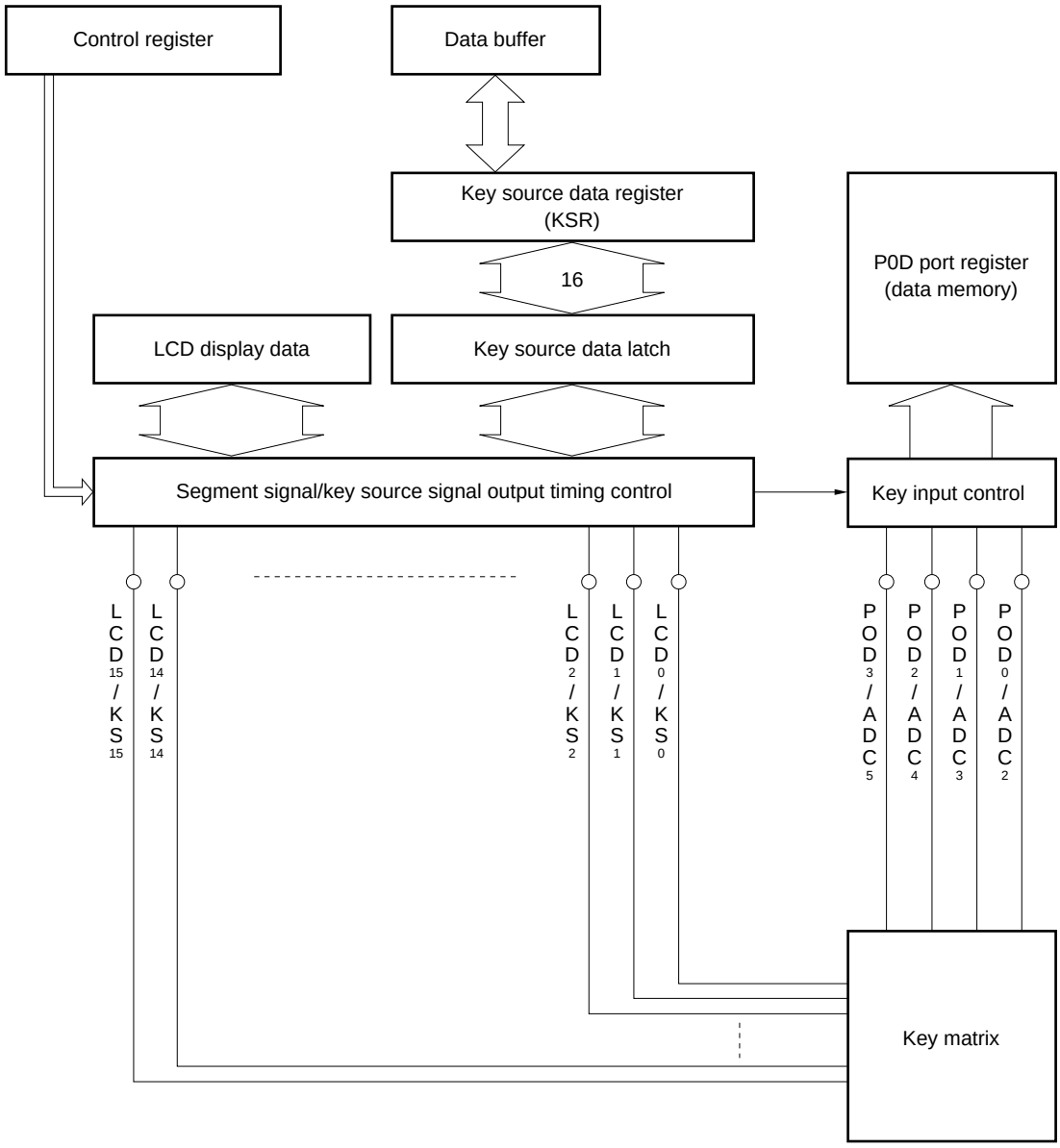
22.1 Configuration of Key Source Controller/Decoder

Figure 22-1 shows the configuration of the key source controller/decoder.

As shown in the figure, the key source controller/decoder consists of a segment signal/key source signal timing control block, key source data register, key input control block, and P0D port register.

The following section 22.2 outlines the function of each block.

Figure 22-1. Block Diagram of Key Source Controller/Decoder



22.2 Functional Outline of Key Source Controller/Decoder

The key source controller/decoder can configure a key matrix of up to 64 keys by using key source signal output pins (LCD_{15}/KS_{15} through LCD_0/KS_0) and key input pins ($P0D_3/ADC_5$ through $P0D_0/ADC_2$).

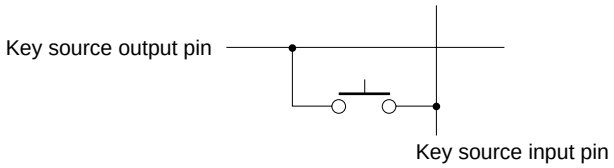
Figure 22-2 shows the example of key matrix configuration.

The LCD_{15}/KS_{15} through LCD_0/KS_0 pins are multiplexed with LCD segment signal output pins.

Therefore, the key source signals and LCD segment signals are output by means of time-division multiplexing.

The following subsections 21.2.1 through 22.2.3 outline the function of each block of the key source controller/decoder.

Figure 22-2. Example of Key Matrix Configuration



22.2.1 Key source data register (KSR)

The key source data register sets the key source output data of the pin that outputs a key source signal.

Data are set to the key source data register via the data buffer.

When data are set to this register, the key source data are output from the LCD_{15}/KS_{15} through LCD_0/KS_0 pins.

For details, refer to **22.3**.

22.2.2 Segment signal/key source signal output timing control block

The segment signal/key source signal output timing block controls the output timing of the key source and segment signals of the $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins.

Whether a key source signal is used or not is specified by the LCD mode select register.

The key source signal is not output when LCD display is not used. In this case, the above pins output a low level.

Whether LCD display is not used or not is specified by the LCD mode select register.

For details, refer to **22.4**.

22.2.3 Key input control block and P0D port register

The key input control block detects the key input signals input to the $P0D_3/ADC_5$ through $P0D_0/ADC_2$ pins in synchronization with key source signal output timing.

To output the key source signals from the LCD_{15}/KS_{15} through LCD_0/KS_0 pins, therefore, the $P0D_3/ADC_5$ through $P0D_0/ADC_2$ pins are used as key input pins.

The key input data are read by the P0D port register (address 73H of BANK0) on the data memory.

Because the $P0D_3/ADC_5$ through $P0D_0/ADC_2$ pins are multiplexed with the A/D converter pins, care must be exercised when using these pins as the A/D converter pins.

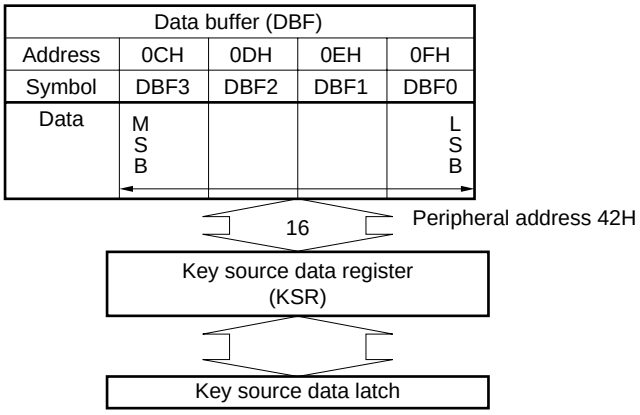
For details, refer to **22.5**.

22.3 Key Source Data Setting Block

22.3.1 Configuration of key source data setting block

Figure 22-3 shows the configuration of the key source data setting block.

Figure 22-3. Configuration of Key Source Data Setting Block



22.3.2 Function of key source data setting block

The key source data setting block sets the key source data to be output from the LCD_{15}/KS_{15} through LCD_0/KS_0 pins.

The key source data are set to the key source data register (KSR: peripheral address 42H) via the data buffer.

Each bit of the key source data register corresponds to the LCD_{15}/KS_{15} through LCD_0/KS_0 pins, and sets the key source data of each pin.

When "1" is set to a bit of the key source data register, the pin corresponding to this bit outputs a high level as a key source signal; when the bit is reset to "0", the corresponding pin outputs a low level.

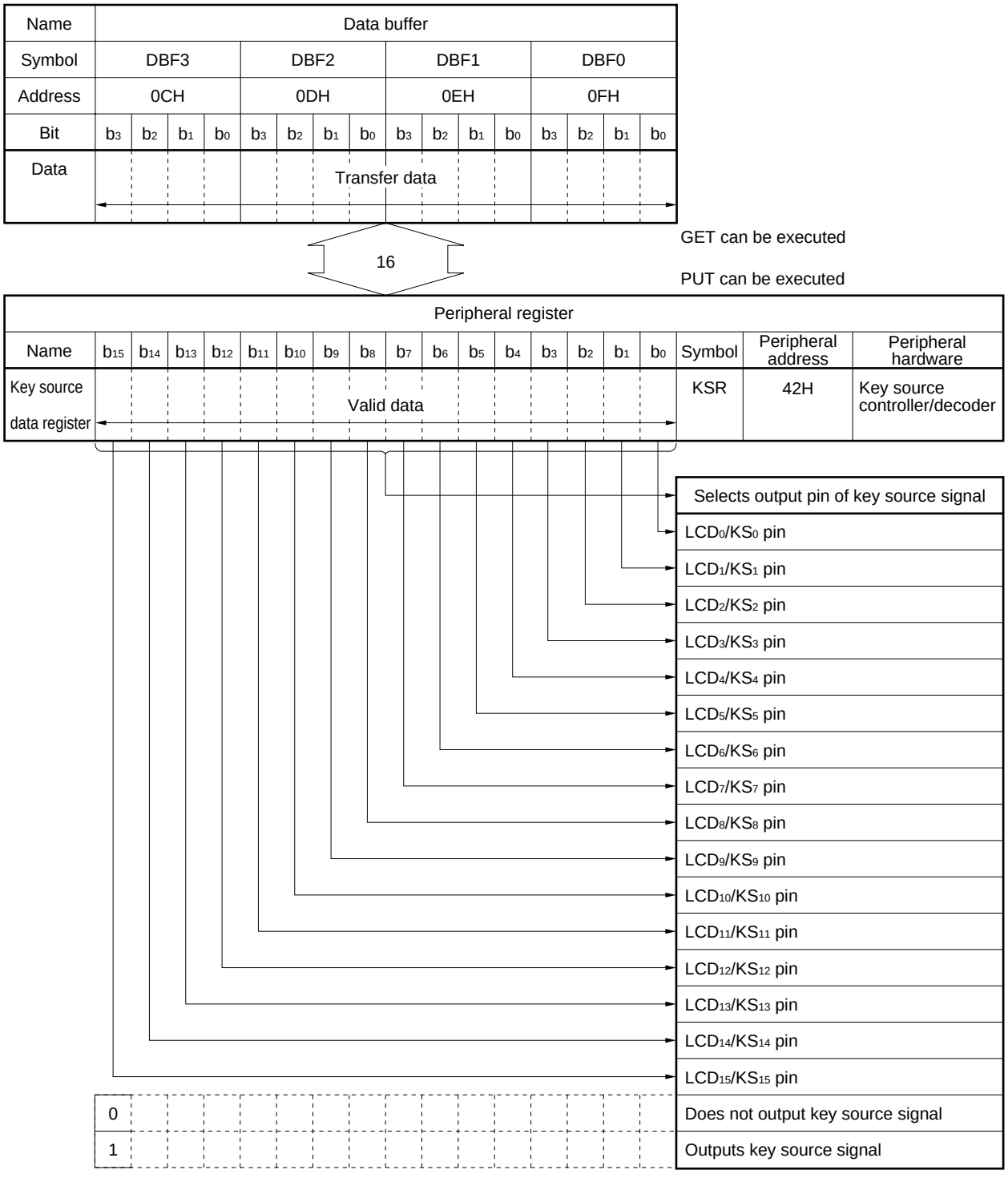
For the output timing, refer to **22.4**.

The following subsections 22.3.3 explains the configuration and function of the key source data register.

Also refer to **Figure 21-6** in **21. LCD CONTROLLER/DRIVER**.

22.3.3 Configuration and function of key source data register (KSR)

The configuration and function of the key source data register are illustrated below.

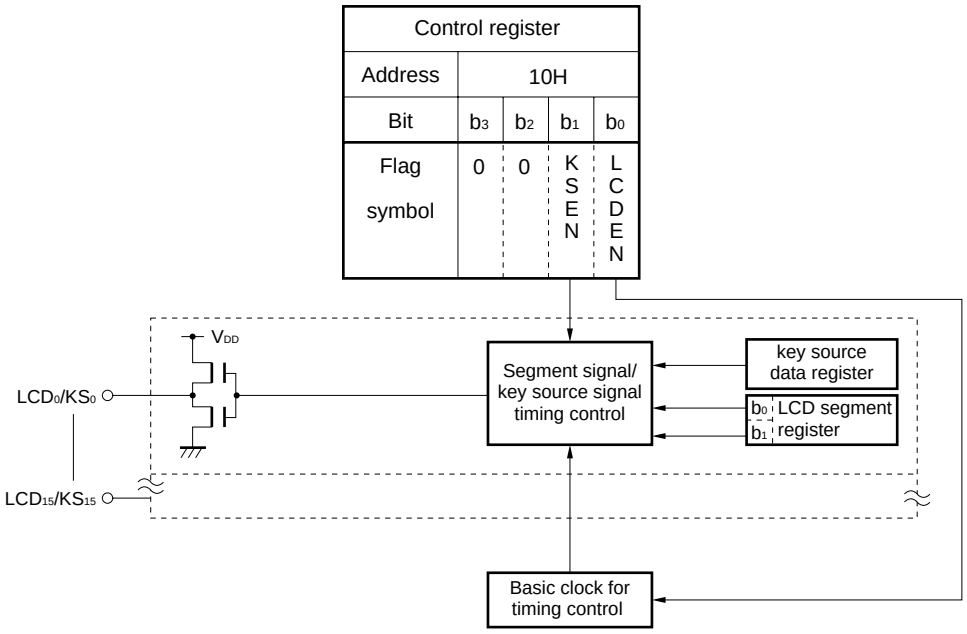


22.4 Output Timing Control Blocks and Segment/Port Select Block

22.4.1 Configuration of output timing control blocks and segment/port select block

Figure 22-4 shows the configuration of the common signal and segment signal/key source signal output timing control blocks and segment signal/general-purpose output port select block.

Figure 22-4. Configuration of Timing Control Blocks and Port Select Block



22.4.2 Function of output timing control blocks

The output timing control blocks controls the output timing of the key source and segment signals.

The LCD segment signal is output when the LCDEN flag of the LCD mode select register is “1”.

All the LCD display dots can be turned OFF by resetting the LCDEN flag to “0”. At this time, a low level is output as the segment signal, and the key source signal is not output.

To output the key source signal, therefore, the LCDEN flag must be “1”.

The key source signal is also output when the KSEN flag of the LCD mode select register is “1”.

Therefore, the KSEN flag is used to specify whether the key source signal is used or not.

To output the key source signal, therefore, the LCDEN and KSEN flags must be “1”.

The following subsection 22.4.3 indicate the configuration and function of the LCD mode select register.

Subsection 22.4.4 shows the output waveforms of the key source and segment signals.

For the relation among the common and segment signals of the LCD, and key source signal, refer to **21. LCD CONTROLLER/DRIVER**.

22.4.3 Configuration and function of LCD mode select register

The LCD mode select register turns ON/OFF all the LCD display dots, and specifies output of the key source signal.

The configuration and function of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
LCD mode select register	0	0	KSEN	LCDEN	10H	R/W

LCDEN	Turns ON/OFF all LCD display dots
0	Display OFF (all segment and common output pins output low level)
1	Display ON

KSEN	Selects output of key source signal
0	Key source OFF
1	Key source ON

Fixed to "0"

		b_3	b_2	b_1	b_0
On reset	Power-ON	0	0	0	0
	Clock stop			0	0
	CE			Retained	

22.4.4 Output waveforms of segment and key source signals

Figure 22-5 shows the output waveforms of the key source and segment signals.

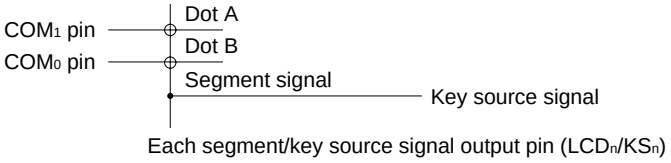
As shown in the figure, the key source and segment signals are output by means of time-division multiplexing.

The key source signal is output for 220 μs at intervals of 4 ms.

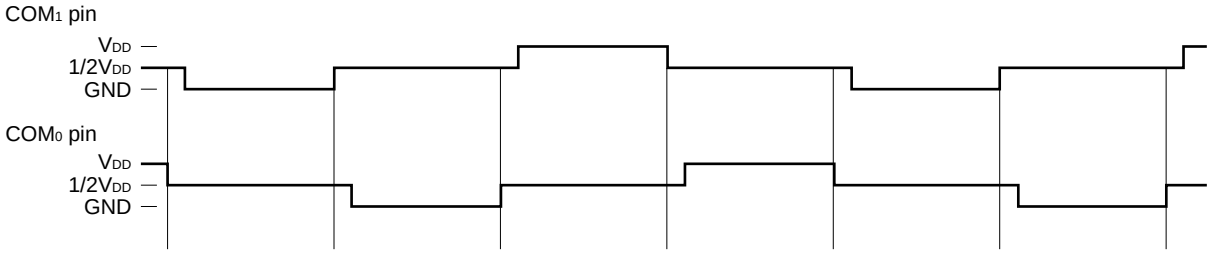
To put it in another way, a pin corresponding to a bit of the key source data register that is set to “1” outputs a high level for 220 μs every 4 ms, and a pin corresponding to a bit of the key source data register that is reset to “0” outputs a low level for 220 μs every 4 ms.

When output of the key source signal is selected (KSEN flag = 1), pins that do not output key source signals (LCD_{29}/$P0F_3$ through LCD_{16}) output the waveform shown in Figure 22-5. However, a waveform of “0” is output as the key source data.

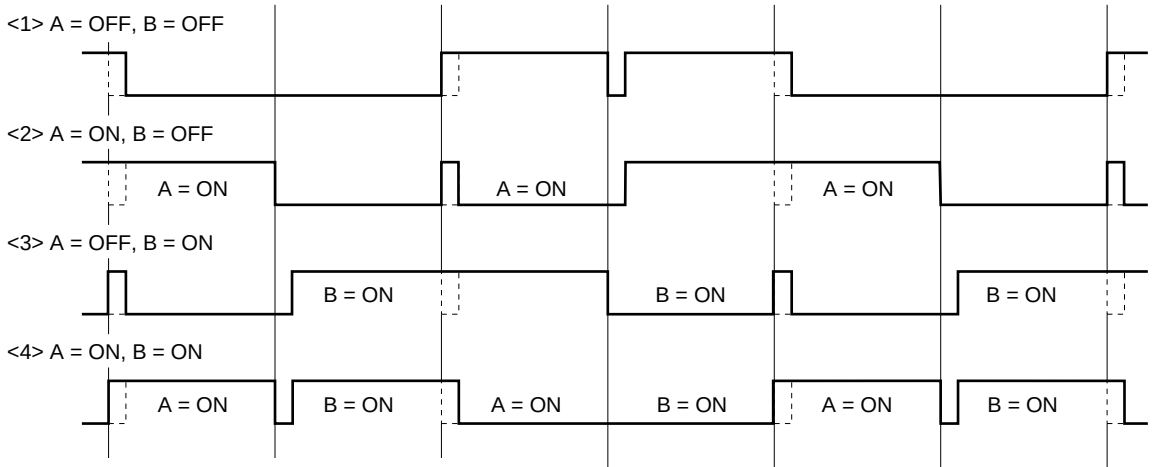
Figure 22-5. Output Waveforms of Key Source and Segment Signals



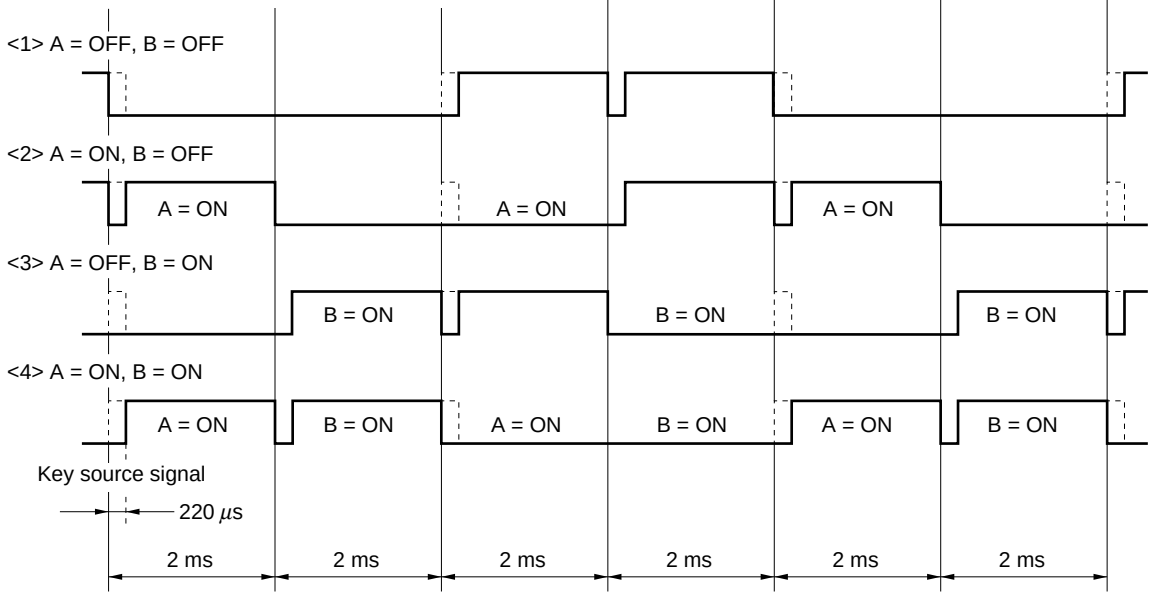
Connon signals



Each segment pin (pin outputting "1" as key source)



Each segment pin (pin outputting "0" as key source)

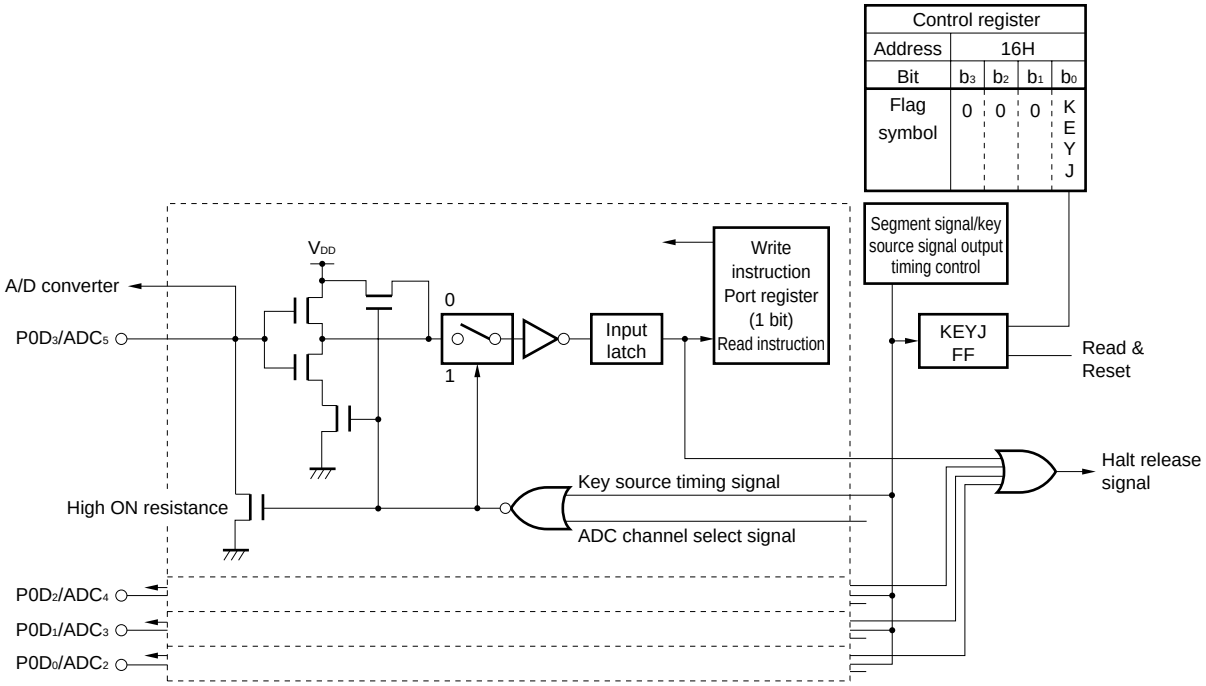


22.5 Key Input Control Block

22.5.1 Configuration of key input control block

Figure 22-6 shows the configuration of the key input control block.

Figure 22-6. Configuration of Key Input Control Block



22.5.2 Function of key input control block

The key input control block controls the timing to read the key input signals from the P0D3/ADC5 through P0D0/ADC2 pins and reads the key input data.

Figure 22-7 illustrates the key input signals and key input timing.

As shown in this figure, the internal-pull down resistors of the P0D3/ADC5 through P0D0/ADC2 pins are turned OFF while the display data of the LCD segment is output, and turned ON only for 220 μs while the key source signal is output.

For the duration of 220 μs during which the key source signal is output, the input signal of each key input pin is connected to the input latch.

Therefore, the signal input to each key input pin can be detected in the 220 μs during which the key source signal is output.

Figure 22-8 shows the timing chart of the key source signal, key input signal, and key input data (P0D port register).

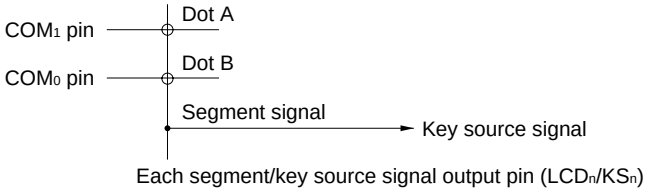
Whether a key source signal is output or not is detected by the KEYJ flag of the key input judge register (RF address 16H).

The KEYJ flag is set after the key source signal has been output for 220 μs, and is reset when data has been set to the key source data register and when the content of the KEYJ flag has been read.

By detecting the KEYJ flag after the key source signal data has been output to the key source data register, and then detecting the status of each key input pin after the KEYJ flag has been set to “1”, the key can be input.

The following subsection 22.5.3 explains the configuration and function of the key input judge register.

Figure 22-7. Key Source Signal and Key Input Timing



Each segment pin (pin outputting "1" as key source, A = ON, B = OFF)

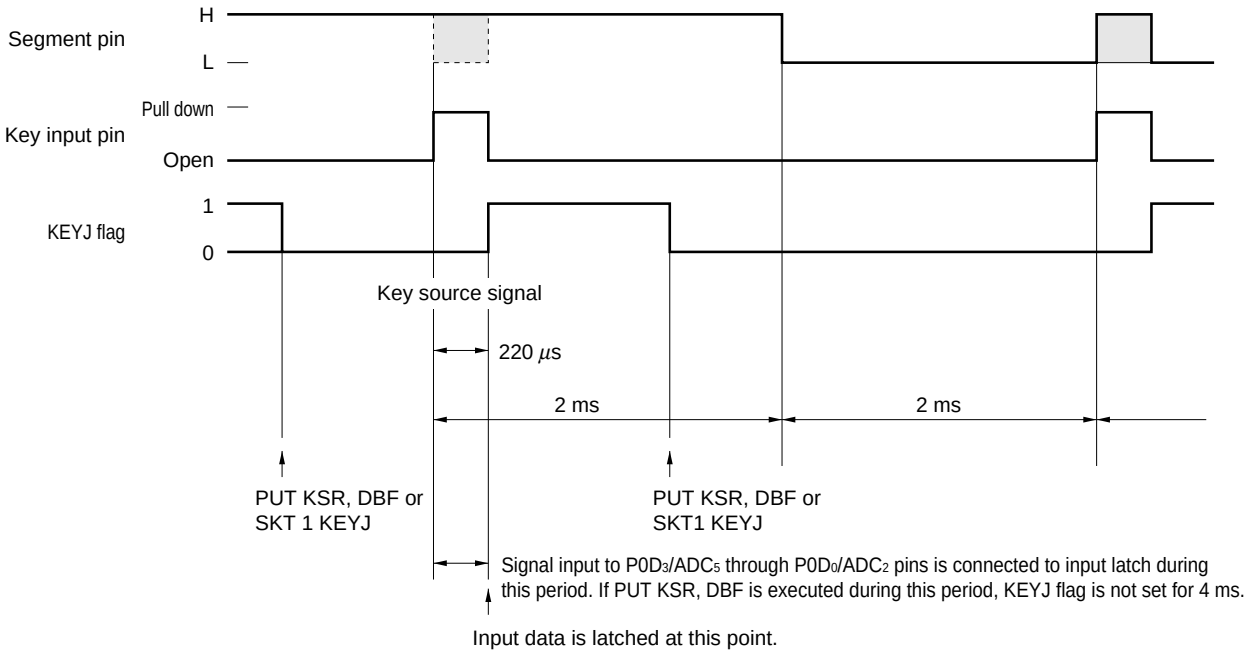
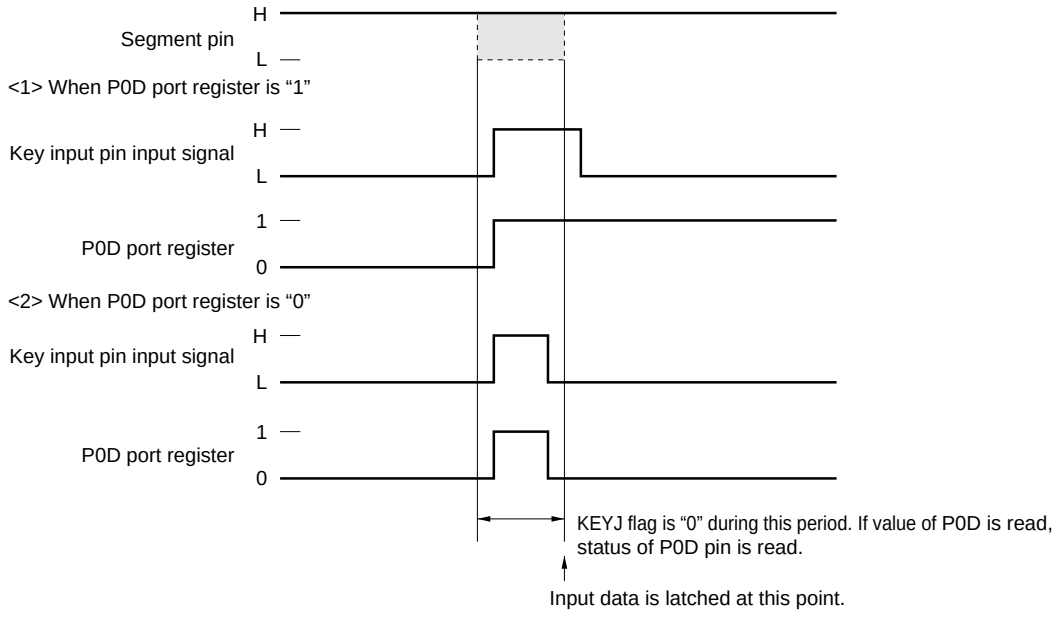


Figure 22-8. Timing Chart of Key Source Signal, Key Input Signal, and Key Input Data (P0D port register)



22.5.3 Configuration and function of key input judge register

The key input judge register detects the presence or absence of the key input signal latch when the LCD segment signal output pins are shared with key source signal output.

The configuration and functions of this register are illustrated below.

Name	Flag symbol				Address	Read/ Write
	b_3	b_2	b_1	b_0		
Key input judge register	0	0	0	KEYJ	16H	R & Reset

Detects presence/absence of latch of key input signal	
0	No key latch
1	Key latch

Fixed to "0"

On reset	Power-ON	0	0	0	0
	Clock stop				Unde-fined
	CE				Unde-fined

The key source data is set by setting the contents of the data buffer to the key source data register by using the “PUT” instruction.

When the key source signal output data is set by the “PUT” instruction via the data buffer, the KEYJ flag is reset to “0”.

The KEYJ flag is also reset to “0” (Read & Reset) when the contents of the window register are read by the “PEEK” instruction.

22.6 Using Key Source Controller/Decoder

22.6.1 Configuring key matrix

Figure 22-9 shows an example of configuring a key matrix.

As shown in this figure, the key matrix can be configured for up to 64 keys.

Because the key source signal output pins also output the LCD segment signals at the same time, diode “A” must be used to prevent reverse flow of the LCD segment signal if a momentary switch is used.

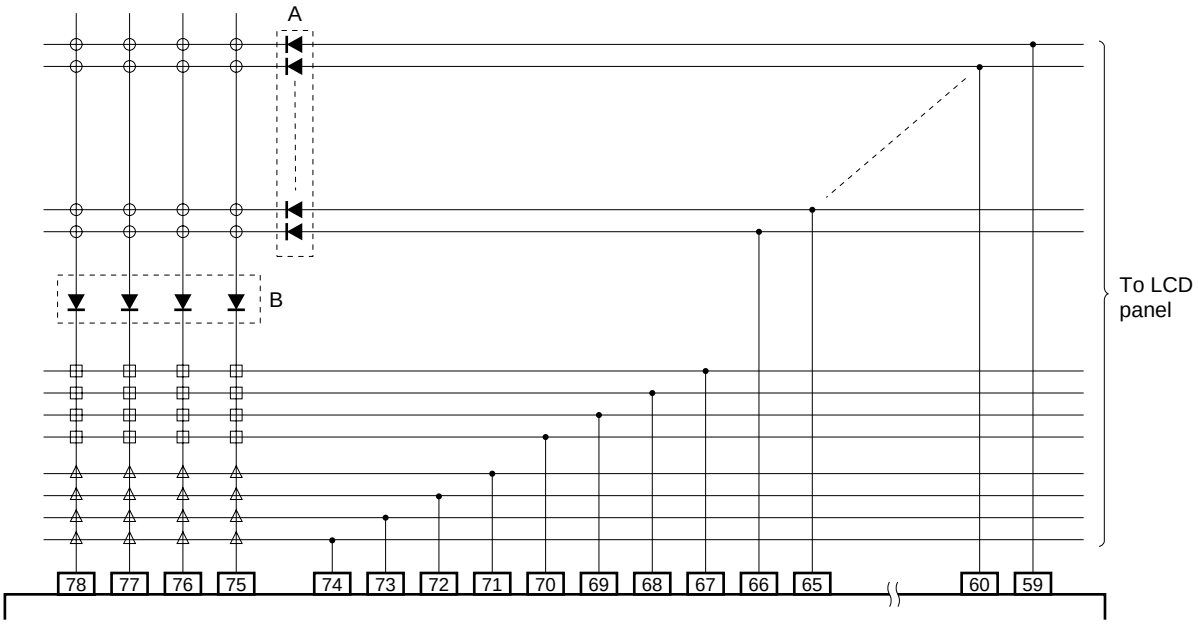
Diodes “B” and “C” are used to prevent sneaking of the key source signal.

Use a PNP transistor as the transistor switch.

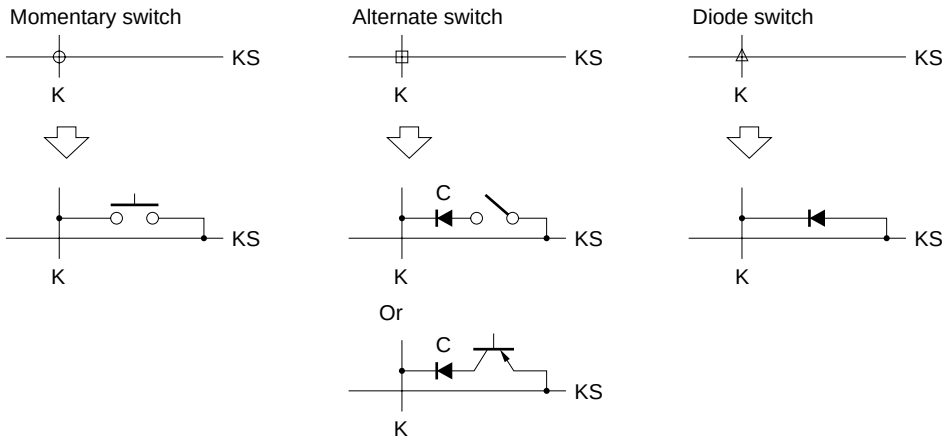
The following (1) explains the points to be noted when an NPN transistor is used.

(2) through (4) explain the points to be noted if diodes A, B, and C are not used.

Figure 22-9. Example of Key Matrix Configuration

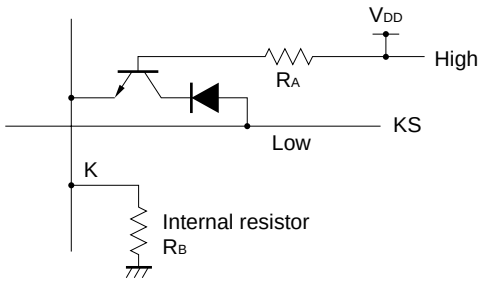


Configuration of each switch



(1) Notes on using NPN transistor switch

When an NPN transistor switch is used, the low level may not be accurately read as illustrated in the figure below.



If KS is low and a high level is input to the base of the transistor in the figure on the left, voltage V_K input to K is as follows.

$$V_K = \frac{R_B}{R_A + R_B} \times (V_{DD} - V_{BE})$$

A low level must be input to K at this time because KS is low. However, the voltage input to K changes depending on R_A and R_B, as indicated by the above expression.

Therefore, a low level may not be input depending on the values of R_A and R_B.

(2) Notes when diode A is not used

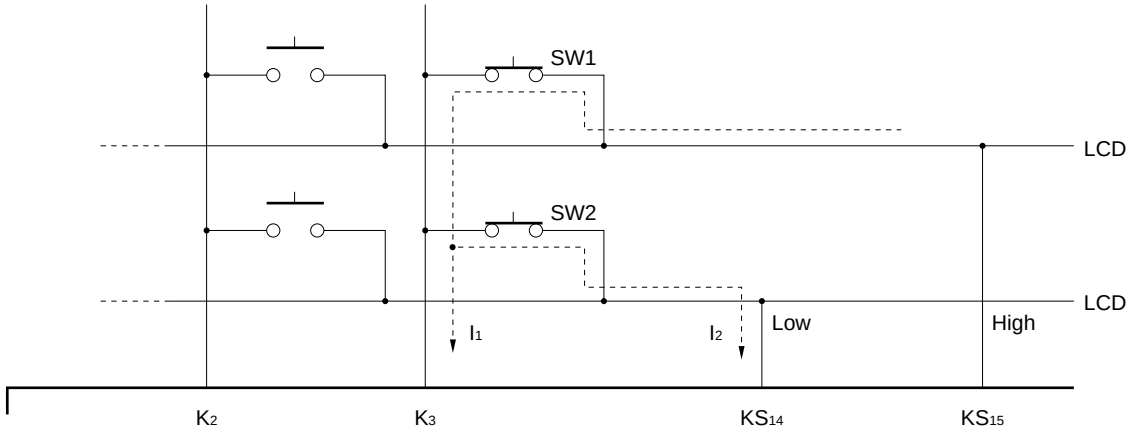
An example of a circuit where diode A is missing is shown below.

Suppose switches SW1 and SW2 are ON, KS_{15} outputs a high level, and KS_{14} outputs a low level, as shown below.

If diode A is missing, currents I_1 and I_2 indicated by the dotted lines flow.

Consequently, the high level of KS_{15} and low level of KS_{14} are not output correctly because of I_2, and the key input data of K_3 cannot be accurately read.

If KS_{15} and KS_{14} are used to output LCD segment signals, the LCD cannot be turned ON/OFF correctly.



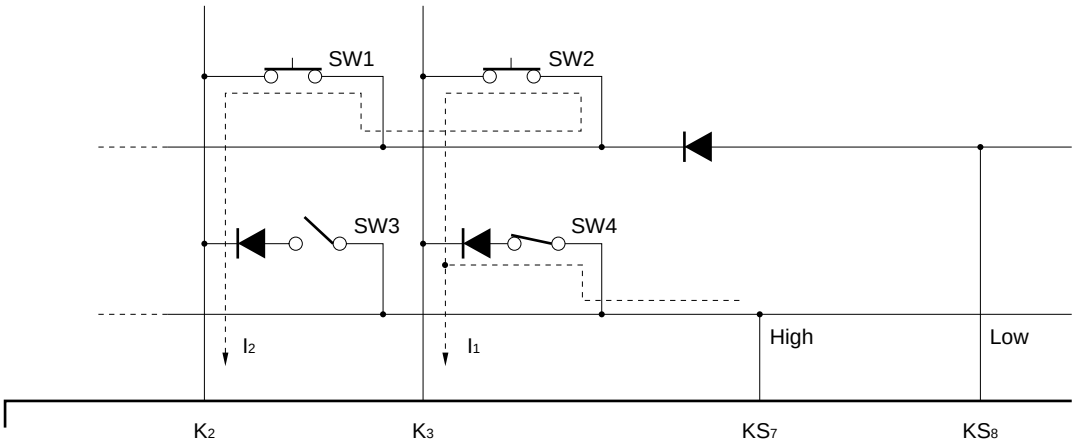
(3) Notes when diode B is not used

An example of a circuit where diode B is not used is shown below.

Suppose switches SW1, SW2, and SW4 are ON, and KS_7 outputs a high level, as shown below.

If diode B is missing, currents I_1 and I_2 flow as indicated by the dotted lines.

Consequently, a high level is input to K_2 because of I_2 despite that switch SW3 is OFF, and it is judged that SW3 is ON.



(4) Note when diode C is not used

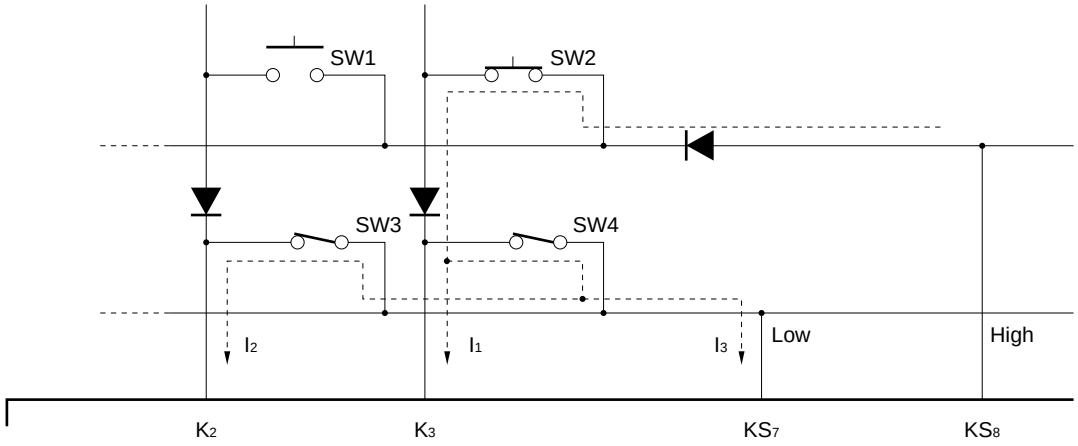
An example of a circuit where diode C is not used is shown below.

Suppose switches SW2, SW3, and SW4 are ON, and KS_8 outputs a high level, as shown below.

If diode C is missing, currents I_1, I_2, and I_3 flow as indicated by the dotted lines.

Consequently, a high level is input to K_2 because of I_2 despite the fact that switch SW1 is OFF, and it is judged that SW1 is ON.

Moreover, KS_8 cannot output a high level correctly because of I_3.



22.6.2 Reading alternate switches and diode switches

Here is a program example.

Example To read statuses of alternate and diode switches of LCD_{15}/KS_{15} through LCD_8/KS_8 pins to addresses 20H through 27H of BANK0 of data memory

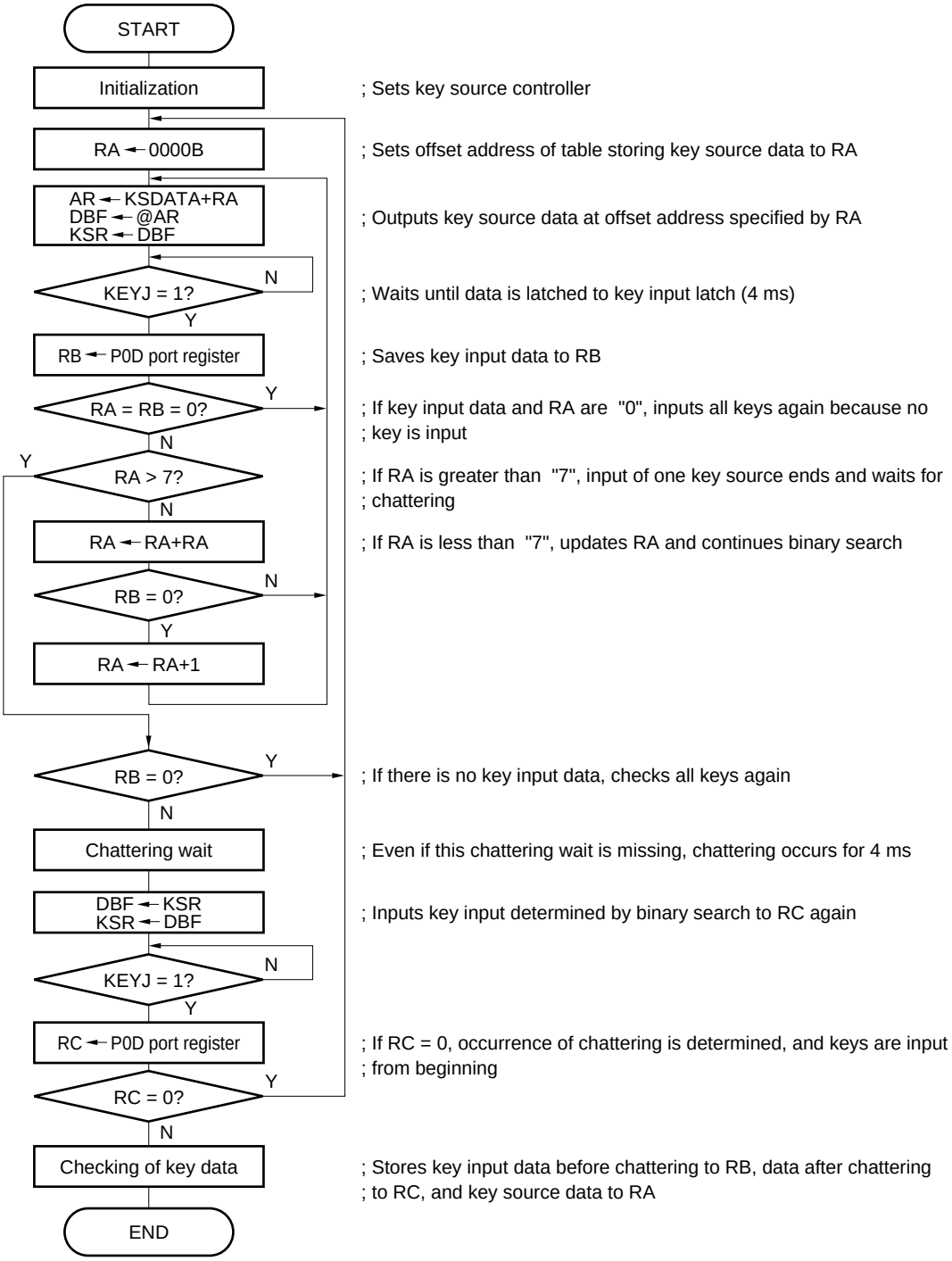
```
KS8       NIBBLE8 0.20H
KEY_IN    MEM     0.73H            ; P0D port register

KEY_LOAD:
          SET2    LCDEN, KSEN      ; LCD segment and key source signal output
          MOV     DBF3, #0000B     ; Sets key source data
          MOV     DBF2, #0001B     ; Outputs low level from KS8
          MOV     DBF1, #0000B
          MOV     DBF0, #0000B
          MOV     IXM, #0000B
          MOV     IXL, #0000B
          MOV     RPH, #0000B
          MOV     RPL, #0000B
KSCAN:
          PUT     KSR, DBF         ; Outputs signal of key source data
LOOP:
          SKF1    KEYJ             ; Determines if key input is latched
          BR      KCHECK

          Processing A             ; Waits until key input is latched

          BR      LOOP
KCHECK:
          MOV     RPL#.DM.KEY_IN SHR 3 AND 0EH
          SET1    IXE
          ST      KS8, KEY_IN      ; Stores key input data to data memory
          CLR1    IXE
          MOV     RPL, #0000B
          INC     IX
          ADD     DBF2, DBF2       ; Updates value of key source data and
          ADD     DBF3, DBF3       ; scans key again
          SKT1    CY               ; Determines if all key source lines are input
          BR      KSCAN
 KEY_END:                          ; End of input
```

22.6.3 Inputting momentary switch by binary search

The key source controller/decoder requires 4 ms to input the key of one key source signal line.
To input the keys of 16 key source signals, therefore, it takes 64 ms.
It is therefore convenient if the binary search method explained in (1) and (2) below is used.

(1) Flowchart

When KS_7 through KS_0 are used as key source signals of momentary switch



Example of table data for binary search

Shift Address (RA)	Table Data (Key Source Data)															
	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
0000B									■	■	■	■	■	■	■	■
0001B									■	■	■	■				
0010B													■	■		
0011B											■	■				
0100B															■	
0101B													■			
0110B											■					
0111B									■							
1000B																■
1001B															■	
1010B														■		
1011B													■			
1100B												■				
1101B											■					
1110B										■						
1111B									■							

(2) Program example

```
RA      MEM   0.1AH                ; General-purpose work register
RB      MEM   0.1BH                ; General-purpose work register
RC      MEM   0.1CH                ; General-purpose work register

KEY_IN MEM    0.73H                ; P0D port register

KSDATA:
        ;     KKKKKKKKKKKKKKKK
        ;     SSSSSSSSSSSSSSSS
        ;     1111119876543210
        ;     543210
        DW    0000000011111111B    ; RA = 0
        DW    0000000011110000B    ; RA = 1
        DW    0000000000001100B    ; RA = 2
        DW    0000000000110000B    ; RA = 3
        DW    0000000000000010B    ; RA = 4
        DW    0000000000001000B    ; RA = 5
        DW    0000000000100000B    ; RA = 6
        DW    0000000010000000B    ; RA = 7
        DW    0000000000000001B    ; RA = 8
        DW    0000000000000010B    ; RA = 9
        DW    0000000000000100B    ; RA = 10
        DW    0000000000001000B    ; RA = 11
        DW    0000000000010000B    ; RA = 12
        DW    0000000000100000B    ; RA = 13
        DW    0000000001000000B    ; RA = 14
        DW    0000000010000000B    ; RA = 15
KEY_LOAD:
        SET2  LCDEN, KSEN          ; LCD segment and key source signal output
START:
        MOV   RA, #0000B
```

```
KSCAN:
        MOV   AR3, #.DL.KSDATA SHR 0CH AND 0FH
        MOV   AR2, #.DL.KSDATA SHR 8 AND 0FH
        MOV   AR1, #.DL.KSDATA SHR 4 AND 0FH
        MOV   AR0, #.DL.KSDATA AND 0FH
        MOV   RPL, #.DL.AR0 SHR 3 AND 0EH
        ADD   AR0, RA
        ADDC  AR1, #0
        ADDC  AR2, #0
        ADDC  AR3, #0
        MOV   RPL, #0
        MOVT  DBF, @AR              ; Reads table data

        PUT   KSR, DBF              ; Outputs signal of key source data
LOOP1:
        SKF1  KEYJ                  ; Determines if key input is latched
        BR    KCHECK

        Processing A                ; Waits until key input is latched

        BR    LOOP1
KCHECK:
        MOV   PRL, #.DM.RB SHR 3 AND 0EH
        LD    RB, KEY_IN            ; Stores key input data to RB
        SKNE  RA, #0000B            ; All keys are checked?
        SKE   RB, #0000B
        BR    Key input
        BR    START                 ; There is no key input
Key input:
        SKLT  RA, RA                ; Key sources are narrowed down to one?
        BR    LASTCHK
                                    ; If not, updates value of RA, and scans keys again
        ADD   RA, RA
        SKE   RB, #0000B
        ADD   RA, #0001B
        BR    KSCAN
LASTCHK:
        MOV   RPL, #0
        SKNE  RB, #0000B            ; Key input to one key source?
        BR    START                 ; If not, it is determined that chattering occurs

        Chattering wait
```

```
LOOP2:
       SKF1  KEYJ                  ; Determines if key input is latched
       BR    KEYDEC

       ┌──────────────┐
       │ Processing B │            ; Waits until key input is latched
       └──────────────┘

       BR    LOOP2
KEYDEC:
       MOV   RPL, #.DM.RC SHR 3 AND 0EH
       LD    RC, KEY_IN            ; Stores key input data to latch
       SET2  CAP, Z                ; Compares key input data after chattering with key input
       SUB   RC, RB                ; data before chattering wait
       SKT1  Z
       BR    START                 ; If data differ
KEY_END:
                                   ; Stores key source data to RA, key input data before
                                   ; chattering to RB, and key input data after chattering to RC
```

22.7 Status on Reset

22.7.1 On power-ON reset

The $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins are specified as the LCD segment signal output pins and output a low level (display OFF). A low level is output as the key source signal.

22.7.2 On execution of clock stop instruction

The $LCD_{29}/P0F_3$ through LCD_0/KS_0 pins are specified as the LCD segment signal output pins and output a low level (display OFF). A low level is output as the key source signal.

22.7.3 On CE reset

The output data is retained as is if the key source signal is being output.

22.7.4 In halt status

The output data is retained as is if the key source signal is being output.

If key input is specified as a halt status releasing condition, the halt status is released when a high level is input to the $P0D_3/ADC_5$ through $P0D_0/ADC_2$ pins.

If the key source controller is used, however, the halt status is released only by a high level that is input within 220 μs during which the key source data is output.

To release the halt status by key input by using the key source controller, do not use the $P0D_3/ADC_5$ through $P0D_0/ADC_2$ pins for the A/D converter.

For an explanation of how to release the halt status by key input, refer to **12.4 Halt Function**.

23. INSTRUCTION SET

23.1 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 INC RORC MOVT PUSH POP GET PUT PEEK POKE BR CALL RET RETSK RETI EI DI STOP HALT NOP	AR IX r DBF, @AR AR AR DBF, p p, DBF WR, rf rf, WR @AR @AR s h		
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, #n
1111	F	BR	addr (page 3)	SKF	m, #n

23.2 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 (bit 11 of program counter: μPD17016) (bits 11 and 12 of program counter: μPD17017)
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
r_{fR}	:	register file address (high-order 3 bits)
n_{fC}	:	register file address (low-order 4 bits)
SP	:	stack pointer
s	:	stop release condition
WR	:	window register
(×)	:	contents addressed by ×

23.3 Instruction Set List

Instructions	Mnemonic	Operand	Operation	Instruction Code			
				op Code	Operand		
Add	ADD	r, m	(r) ← (r) + (m)	00000	m_R	m_C	r
		m, #n4	(m) ← (m) + n4	10000	m_R	m_C	n4
	ADDC	r, m	(r) ← (r) + (m) + CY	00010	m_R	m_C	r
		m, #n4	(m) ← (m) + n4 + CY	10010	m_R	m_C	n4
	INC	AR	AR ← AR + 1	00111	000	1001	0000
		IX	IX ← IX + 1	00111	000	1000	0000
Subtract	SUB	r, m	(r) ← (r) – (m)	00001	m_R	m_C	r
		m, #n4	(m) ← (m) – n4	10001	m_R	m_C	n4
	SUBC	r, m	(r) ← (r) – (m) – CY	00011	m_R	m_C	r
		m, #n4	(m) ← (m) – n4 – CY	10011	m_R	m_C	n4
Logical operation	OR	r, m	(r) ← (r) ∨ (m)	00110	m_R	m_C	r
		m, #n4	(m) ← (m) ∨ n4	10110	m_R	m_C	n4
	AND	r, m	(r) ← (r) ∧ (m)	00100	m_R	m_C	r
		m, #n4	(m) ← (m) ∧ n4	10100	m_R	m_C	n4
	XOR	r, m	(r) ← (r) ∀ (m)	00101	m_R	m_C	r
		m, #n4	(m) ← (m) ∀ n4	10101	m_R	m_C	n4
Judge	SKT	m, #n	CMP ← 0, if (m) ∧ n = n, then skip	11110	m_R	m_C	n
	SKF	m, #n	CMP ← 0, if (m) ∧ 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	CY → (r)$_{b3}$ → (r)$_{b2}$ → (r)$_{b1}$ → (r)$_{b0}$	00111	000	0111	r
Transfer	LD	r, m	(r) ← (m)	01000	m_R	m_C	r
	ST	m, r	(m) ← (r)	11000	m_R	m_C	r
	MOV	@r, m	if MPE = 1: (MP, (r)) ← (m) if MPE = 0: (BANK, m_R, (r)) ← (m)	01010	m_R	m_C	r
		m, @r	if MPE = 1: (m) ← (MP, (r)) if MPE = 0: (m) ← (BANK, m_R, (r))	11010	m_R	m_C	r
		m, #n4	(m) ← n4	11101	m_R	m_C	n4
	MOVT	DBF, @AR	SP ← SP – 1, ASR ← PC, PC ← AR, DBF ← (PC), PC ← ASR, SP ← SP + 1	00111	000	0001	0000
	PUSH	AR	SP ← SP – 1, ASR ← AR	00111	000	1101	0000
	POP	AR	AR ← ASR, SP ← SP + 1	00111	000	1100	0000
	PEEK	WR, rf	WR ← (rf)	00111	rf_R	0011	rf_C

Caution The program memory addresses that can be specified by the address register range as follows:

μPD17016: 0000H through 00FFH
μPD17017: 0000H through 1EFBH

Instructions	Mnemonic	Operand	Operation	Instruction Code			
				op Code	Operand		
Transfer	POKE	rf, WR	(rf) ← WR	00111	rf_R	0010	rf_C
	GET	DBF, p	DBF ← (p)	00111	p_H	1011	p_L
	PUT	p, DBF	(p) ← DBF	00111	p_H	1010	p_L
Branch	BR	addr	**Note**	**Note**	addr		
		@AR	PC ← AR	00111	000	0100	0000
Subroutine	CALL	addr	SP ← SP – 1, ASR ← PC PC_{11} ← 0, PC_{10-0} ← addr	11100	addr		
		@AR	SP ← SP – 1, ASR ← PC PC ← AR	00111	000	0101	000
	RET		PC ← ASR, SP ← SP + 1	00111	000	1110	0000
	RETSK		PC ← ASR, SP ← SP + 1 and skip	00111	001	1110	0000
	RETI		PC ← ASR, INTR ← INTSK, SP ← SP + 1	00111	010	1110	0000
Interrupt	EI		INTEF ← 1	00111	000	1111	0000
	DI		INTEF ← 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

Note The operation and op code of “BR addr” of the μPD17016 and μPD17017 are as follows.

(a) μPD17016

Mnemonic	Operand	Operation	op Code
BR	addr	PC_{10-0} ← addr, PAGE ← 0	01100
		PC_{10-0} ← addr, PAGE ← 1	01101

(b) μPD17017

Mnemonic	Operand	Operation	op Code
BR	addr	PC_{10-0} ← addr, PAGE ← 0	01100
		PC_{10-0} ← addr, PAGE ← 1	01101
		PC_{10-0} ← addr, PAGE ← 2	01110
		PC_{10-0} ← addr, PAGE ← 3	01111

Caution The program memory addresses that can be specified by the address register range as follows:

μPD17016: 0000H through 00FFH
μPD17017: 0000H through 1EFBH

23.4 Assembler (AS17K) Embedded Macroinstruction

Legend

flag n: FLG symbol
n : bit number
< > : can be omitted

	Mnemonic	Operand	Operation	n
Embedded macro	SKTn	flag 1, ... flag n	if (flag 1) to (flag n) = all “1”, then skip	1 ≤ n ≤ 4
	SKFn	flag 1, ... flag n	if (flag 1) to (flag n) = all “0”, then skip	1 ≤ n ≤ 4
	SETn	flag 1, ... flag n	(flag 1) to (flag n) ← 1	1 ≤ n ≤ 4
	CLRn	flag 1, ... flag n	(flag 1) to (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	**Note**

Note 0 ≤ n ≤ 2: μPD17016
0 ≤ n ≤ 3: μPD17017

23.5 Software Macroinstructions

23.5.1 BEEP output software macro

Item	Macro format	Function
Controls BEEP output	BEEPON200	Outputs 200 Hz
	BEEPON1K	Outputs 1 kHz
	BEEPON3K	Outputs 3 kHz
	BEEPOFF	Stops BEEP output

Caution If the above macros are used, the contents of the window register are destroyed.
If an embedded macro instruction is used immediately before the above macro instructions, an "object error" occurs when the source file is assembled and then loaded to the in-circuit emulator. To use an embedded macro instruction immediately before the above macro instructions, insert a comment statement between them.

23.5.2 Flag manipulation software macro

Item	Macro format	Function
Detects INT_0 pin status	SKT1_INT0	Skips next instruction if high level is input to INT_0 pin
	SKF1_INT0	Skips next instruction if low level is input to INT_0 pin
Sets interrupt enable flag	SET1_IPTM	Enables timer interrupt
	SET1_IP0	Enables INT_0 pin interrupt
	SET2_IPTM_IP0	Enables timer interrupt and INT_0 pin interrupt
	CLR1_IPTM	Disables timer interrupt
	CLR1_IP0	Disables INT_0 pin interrupt
	CLR2_IPTM_IP0	Disables timer interrupt and INT_0 pin interrupt
	INIT_NOT_IPTM_IP0	Disables timer interrupt and enables INT_0 pin interrupt
	INIT_IPTM_NOT_IP0	Enables timer interrupt and disables INT_0 pin interrupt
Resets interrupt request flag	CLR1_IRQTM[Note]	Resets timer interrupt request
	CLR1_IRQ0[Note]	Resets INT_0 pin interrupt request

Note While any of these macros is executed, any other interrupt request cannot be acknowledged. For details, refer to **10.11 Notes on Using Interrupt**.

Caution If the above macros are used, the contents of the window register are destroyed.
If an embedded macro instruction is used immediately before the above macro instructions, an "object error" occurs when the source file is assembled and then loaded to the in-circuit emulator. To use an embedded macro instruction immediately before the above macro instructions, insert a comment statement between them.

Item	Macro format	Function
Sets general-purpose output port/LCD segment signal output (P0E, P0F)	SET1_P0EON	Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as general-purpose output port
	SET1_P0FON	Uses $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port
	SET2_P0EON_P0FON	Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port
	CLR1_P0EON	Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as LCD segment signal output pins
	CLR1_P0FON	Uses $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins
	CLR2_P0EON_P0FON	Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins
	INIT_NOT_P0EON_P0FON	Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as LCD segment signal output pins and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as general-purpose output port
	INIT_P0EON_NOT_P0FON	Uses $LCD_{22}/P0E_0$-$LCD_{25}/P0E_3$ pins as general-purpose output port and $LCD_{26}/P0F_0$-$LCD_{29}/P0F_3$ pins as LCD segment signal output pins
Sets general-purpose output port/PWM pin signal ($P1B_1$, $P1B_2$)	SET1_PWM0ON	Uses $P1B_1/PWM_0$ pin as PWM pin
	SET1_PWM1ON	Uses $P1B_2/PWM_1$ pin as PWM pin
	SET2_PWM1ON_PWM0ON	Uses $P1B_1/PWM_0$ and $P1B_2/PWM_1$ pins as PWM pins
	CLR1_PWM0ON	Uses $P1B_1/PWM_0$ pin as general-purpose output port pin
	CLR1_PWM1ON	Uses $P1B_2/PWM_1$ pin as general-purpose output port pin
	CLR2_PWM1ON_PWM0ON	Uses $P1B_1/PWM_0$ and $P1B_2/PWM_1$ pins as general-purpose output port pins
	INIT_NOT_PWM1ON_PWM0ON	Uses $P1B_1/PWM_0$ pin as PWM pin and $P1B_2/PWM_1$ pin as general-purpose output port pin
	INIT_PWM1ON_NOT_PWM0ON	Uses $P1B_1/PWM_0$ pin as general-purpose output port pin and $P1B_2/PWM_1$ pin as PWM pin

Caution If the above macros are used, the contents of the window register are destroyed.
If an embedded macro instruction is used immediately before the above macro instructions, an "object error" occurs when the source file is assembled and then loaded to the in-circuit emulator. To use an embedded macro instruction immediately before the above macro instructions, insert a comment statement between them.

24. RESERVED SYMBOL

24.1 Data Buffer (DBF)

Symbol Name	Attribute	Value	R/W	Description
DBF3	MEM	0.0CH	R/W	Bits 15 through 12 of DBF
DBF2	MEM	0.0DH	R/W	Bits 11 through 8 of DBF
DBF1	MEM	0.0EH	R/W	Bits 7 through 4 of DBF
DBF0	MEM	0.0FH	R/W	Bits 3 through 0 of DBF

24.2 System Register (SYSREG)

Symbol Name	Attribute	Value	R/W	Description
AR3	MEM	0.74H	R	Bits 15 through 12 of address register[Note 1]
★ AR2	MEM	0.75H	R	Bits 11 through 8 of address register[Note 2]
AR1	MEM	0.76H	R/W	Bits 7 through 4 of address register
AR0	MEM	0.77H	R/W	Bits 3 through 0 of address register
WR	MEM	0.78H	R/W	Window register
BANK	MEM	0.79H	R/W	Bank register
IXH	MEM	0.7AH	R/W	Index register, high
MPH	MEM	0.7AH	R/W	Memory pointer, high
MPE	FLG	0.7AH.3	R/W	Memory pointer enable flag
IXM	MEM	0.7BH	R/W	Index register, middle
MPL	MEM	0.7BH	R/W	Memory pointer, low
IXL	MEM	0.7CH	R/W	Index register, low
RPH	MEM	0.7DH	R/W	General register pointer, high
RPL	MEM	0.7EH	R/W	General register pointer, low
PSW	MEM	0.7FH	R/W	Program status word
BCD	FLG	0.7EH.0	R/W	BCD operation flag
CMP	FLG	0.7FH.3	R/W	Compare flag
CY	FLG	0.7FH.2	R/W	Carry flag
Z	FLG	0.7FH.1	R/W	Zero flag
IXE	FLG	0.7FH.0	R/W	Index enable flag

Notes 1. All the 4 bits are fixed to "0" with the μPD17016. The high-order 3 bits are fixed to "0" with the μPD17017.

2. The 4 bits are always fixed to "0" with the μPD17016.

24.3 LCD Segment Register

Symbol Name	Attribute	Value	R/W	Description
LCDD0	MEM	0.60H	R/W	LCD segment register
LCDD1	MEM	0.61H	R/W	LCD segment register
LCDD2	MEM	0.62H	R/W	LCD segment register
LCDD3	MEM	0.63H	R/W	LCD segment register
LCDD4	MEM	0.64H	R/W	LCD segment register
LCDD5	MEM	0.65H	R/W	LCD segment register
LCDD6	MEM	0.66H	R/W	LCD segment register
LCDD7	MEM	0.67H	R/W	LCD segment register
LCDD8	MEM	0.68H	R/W	LCD segment register
LCDD9	MEM	0.69H	R/W	LCD segment register
LCDD10	MEM	0.6AH	R/W	LCD segment register
LCDD11	MEM	0.6BH	R/W	LCD segment register
LCDD12	MEM	0.6CH	R/W	LCD segment register
LCDD13	MEM	0.6DH	R/W	LCD segment register
LCDD14	MEM	0.6EH	R/W	LCD segment register

24.4 Port Register

Symbol Name	Attribute	Value	R/W	Description
P0A3	FLG	0.70H.3	R/W	Bit 3 of port 0A
P0A2	FLG	0.70H.2	R/W	Bit 2 of port 0A
P0A1	FLG	0.70H.1	R/W	Bit 1 of port 0A
P0A0	FLG	0.70H.0	R/W	Bit 0 of port 0A
P0B3	FLG	0.71H.3	R/W	Bit 3 of port 0B
P0B2	FLG	0.71H.2	R/W	Bit 2 of port 0B
P0B1	FLG	0.71H.1	R/W	Bit 1 of port 0B
P0B0	FLG	0.71H.0	R/W	Bit 0 of port 0B
P0C3	FLG	0.72H.3	R/W	Bit 3 of port 0C
P0C2	FLG	0.72H.2	R/W	Bit 2 of port 0C
P0C1	FLG	0.72H.1	R/W	Bit 1 of port 0C
P0C0	FLG	0.72H.0	R/W	Bit 0 of port 0C
P0D3	FLG	0.73H.3	R[Note]	Bit 3 of port 0D
P0D2	FLG	0.73H.2	R[Note]	Bit 2 of port 0D
P0D1	FLG	0.73H.1	R[Note]	Bit 1 of port 0D
P0D0	FLG	0.73H.0	R[Note]	Bit 0 of port 0D

Note This port is an input port, but the assembler or in-circuit emulator does not output an error message even if an instruction that writes data to this port is described. Moreover, the operation is not affected even if such a program is actually executed on the device.

24.4 Port Register

Symbol Name	Attribute	Value	R/W	Description
P0E3	FLG	0.6BH.3	R/W	Bit 3 of port 0E
P0E2	FLG	0.6BH.2	R/W	Bit 2 of port 0E
P0E1	FLG	0.6BH.1	R/W	Bit 1 of port 0E
P0E0	FLG	0.6BH.0	R/W	Bit 0 of port 0E
P0F3	FLG	0.6DH.3	R/W	Bit 3 of port 0F
P0F2	FLG	0.6DH.2	R/W	Bit 2 of port 0F
P0F1	FLG	0.6DH.1	R/W	Bit 1 of port 0F
P0F0	FLG	0.6DH.0	R/W	Bit 0 of port 0F
P1A3	FLG	1.70H.3	R/W	Bit 3 of port 1A
P1A2	FLG	1.70H.2	R/W	Bit 2 of port 1A
P1A1	FLG	1.70H.1	R/W	Bit 1 of port 1A
P1A0	FLG	1.70H.0	R/W	Bit 0 of port 1A
P1B3	FLG	1.71H.3	R/W	Bit 3 of port 1B
P1B2	FLG	1.71H.2	R/W	Bit 2 of port 1B
P1B1	FLG	1.71H.1	R/W	Bit 1 of port 1B
P1B0	FLG	1.71H.0	R/W	Bit 0 of port 1B
P1C3	FLG	1.72H.3	R/W	Bit 3 of port 1C
P1C2	FLG	1.72H.2	R/W	Bit 2 of port 1C
P1C1	FLG	1.72H.1	R/W	Bit 1 of port 1C
P1C0	FLG	1.72H.0	R/W	Bit 0 of port 1C
P1D3	FLG	1.73H.3	R[Note]	Bit 3 of port 1D
P1D2	FLG	1.73H.2	R[Note]	Bit 2 of port 1D
P1D1	FLG	1.73H.1	R[Note]	Bit 1 of port 1D
P1D0	FLG	1.73H.0	R[Note]	Bit 0 of port 1D
P2A0	FLG	2.70H.0	R/W	Bit 0 of port 2A

Note This port is an input port, but the assembler or in-circuit emulator does not output an error message even if an instruction that writes data to this port is described. Moreover, the operation is not affected even if such a program is actually executed on the device.

24.5 Register File (control register)

(1/2)

Symbol Name	Attribute	Value	R/W	Description
SP	MEM	0.81H	R/W	Stack pointer
SIO1TS	FLG	0.82H.3	R/W	Serial interface transmit/receive start flag
SIO1HIZ	FLG	0.82H.2	R/W	Serial interface/general-purpose port select flag
SIO1CK1	FLG	0.82H.1	R/W	Serial interface I/O clock setting flag
SIO1CK0	FLG	0.82H.0	R/W	Serial interface I/O clock setting flag
IFCGOSTT	FLG	0.84H.0	R	IF counter gate status detection flag (1: open, 0: close)
PLLUL	FLG	0.85H.0	R	PLL unlock FF flag
ADCCMP	FLG	0.86H.0	R	A/D converter compare result detection flag
CE	FLG	0.87H.0	R	CE pin status detection flag
BTM1CK1	FLG	0.89H.3	R/W	Timer interrupt mode select flag
BTM1CK0	FLG	0.89H.2	R/W	Timer interrupt mode select flag
BTM0CK1	FLG	0.89H.1	R/W	Timer carry FF mode select flag
BTM0CK0	FLG	0.89H.0	R/W	Timer carry FF mode select flag
KSEN	FLG	0.90H.1	R/W	Key source signal output start flag
LCDEN	FLG	0.90H.0	R/W	LCD driver display start flag
IFCMD1	FLG	0.92H.3	R/W	IF counter mode select flag (10: AMIF, 11: setting prohibited)
IFCMD0	FLG	0.92H.2	R/W	IF counter mode select flag (00: BEEP, 01: FMIF)
IFCCK1	FLG	0.92H.1	R/W	IF counter clock select flag
IFCCK0	FLG	0.92H.0	R/W	IF counter clock select flag
ADCCH3	FLG	0.94H.3	R/W	A/D converter channel select flag (dummy)
ADCCH2	FLG	0.94H.2	R/W	A/D converter channel select flag
ADCCH1	FLG	0.94H.1	R/W	A/D converter channel select flag
ADCCH0	FLG	0.94H.0	R/W	A/D converter channel select flag
KEYJ	FLG	0.96H.0	R	Key input judge flag
BTM0CY	FLG	0.97H.0	R	Timer carry FF status detection flag
PLLMD3	FLG	0.0A1H.3	R/W	PLL mode select flag (dummy: 0)
PLLMD2	FLG	0.0A1H.2	R/W	PLL mode select flag (dummy: 0)
PLLMD1	FLG	0.0A1H.1	R/W	PLL mode select flag
PLLMD0	FLG	0.0A1H.0	R/W	PLL mode select flag
IFCSTRT	FLG	0.0A3H.1	W	IF counter count start flag
IFCRES	FLG	0.0A3H.0	W	IF counter reset flag
P0CGIO	FLG	0.0A7H.0	R/W	P0C input/output select flag
PLLRFCK3	FLG	0.0B1H.3	R/W	PLL reference frequency select flag
PLLRFCK2	FLG	0.0B1H.2	R/W	PLL reference frequency select flag
PLLRFCK1	FLG	0.0B1H.1	R/W	PLL reference frequency select flag
PLLRFCK0	FLG	0.0B1H.0	R/W	PLL reference frequency select flag

24.5 Register File (control register)

(2/2)

Symbol Name	Attribute	Value	R/W	Description
P1ABIO3	FLG	0.0B5H.3	R/W	$P1A_3$ input/output select flag
P1ABIO2	FLG	0.0B5H.2	R/W	$P1A_2$ input/output select flag
P1ABIO1	FLG	0.0B5H.1	R/W	$P1A_1$ input/output select flag
P1ABIO0	FLG	0.0B5H.0	R/W	$P1A_0$ input/output select flag
P0BBIO3	FLG	0.0B6H.3	R/W	$P0B_3$ input/output select flag
P0BBIO2	FLG	0.0B6H.2	R/W	$P0B_2$ input/output select flag
P0BBIO1	FLG	0.0B6H.1	R/W	$P0B_1$ input/output select flag
P0BBIO0	FLG	0.0B6H.0	R/W	$P0B_0$ input/output select flag
P0ABIO3	FLG	0.0B7H.3	R/W	$P0A_3$ input/output select flag
P0ABIO2	FLG	0.0B7H.2	R/W	$P0A_2$ input/output select flag
P0ABIO1	FLG	0.0B7H.1	R/W	$P0A_1$ input/output select flag
P0ABIO0	FLG	0.0B7H.0	R/W	$P0A_0$ input/output select flag

24.6 Peripheral Hardware Register

Symbol Name	Attribute	Value	R/W	Description
ADCR	DAT	02H	R/W	A/D converter reference voltage setting register
SIO1SFR	DAT	03H	R/W	Serial interface presettable shift register
PWMR0	DAT	05H	R/W	PWM_0 data register
PWMR1	DAT	06H	R/W	PWM_1 data register
AR	DAT	40H	R/W	Address register
PLLR	DAT	41H	R/W	PLL data register
KSR	DAT	42H	R/W	Key source data register
IFC	DAT	43H	R	IF counter data register

24.7 Others

Symbol Name	Attribute	Value	Description
DBF	DAT	0FH	Fixed operand value of PUT, GET, and MOVT instructions
IX	DAT	01H	Fixed operand value of INC instruction

25. ELECTRICAL SPECIFICATIONS

Absolute Maximum Ratings (T_A = 25 ± 2 °C)

Parameter	Symbol	Condition	Rating	Unit
Supply voltage	V_{DD}		–0.3 to +6.0	V
Input voltage	V_I		–0.3 to V_{DD} + 0.3	V
Output voltage	V_O	Except $P1B_1$ through $P1B_3$, $P0A_2$, $P0A_3$	–0.3 to V_{DD} + 0.3	V
Output breakdown voltage	V_{BDS1}	$P1B_1$-$P1B_3$	18.0	V
	V_{BDS2}	$P0A_2$, $P0A_3$	V_{DD} + 0.3	V
High-level output current	I_{OH}	1 pin	–12	mA
		Total of all pins	–20	mA
Low-level output current	I_{OL}	1 pin	12	mA
		Total of all pins	20	mA
Operating ambient temperature	T_A		–40 to +85	°C
Storage temperature	T_{stg}		–55 to +125	°C

Caution If the maximum absolute rating of even one of the above parameters is exceeded even momentarily, the quality of the product may degrade. The maximum absolute ratings, therefore, specify values exceeding which the product may be physically damaged. Never exceed these ratings when using the product.

Recommended Operating Conditions

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Supply voltage	V_{DD1}	When PLL and CPU operate	4.5	5.0	5.5	V
	V_{DD2}	When PLL stops and CPU operates	3.5	5.0	5.5	V
Data retention voltage	V_{DDR}	When crystal oscillation stops	2.2		5.5	V
Supply voltage rise time	t_{rise}	V_{DD} = 0 → 4.5 V			500	ms
Input amplitude	V_{IN1}	VCOL, VCOH	0.5		V_{DD}	V_{p-p}
	V_{IN2}	AMIFC, FMIFC	0.5		V_{DD}	V_{p-p}
Output breakdown voltage	V_{BDS}	$P1B_1$-$P1B_3$			16.0	V
Operating ambient temperature	T_A		–40		+85	°C

DC Characteristics (T_A = –40 to +85 °C, V_{DD} = 4.5 to 5.5 V)

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Supply voltage	V_{DD1}	When CPU and PLL operate	4.5	5.0	5.5	V
	V_{DD2}	When CPU operates and PLL stops	3.5	5.0	5.5	V
Supply current	I_{DD1}	When CPU and PLL operate X_{IN} pin Sine wave input (f_{IN} = 4.5 MHz, V_{IN} = V_{DD}), T_A = 25 °C		1.2	2.4	mA
	I_{DD2}	When CPU operates and PLL stops HALT instruction used (20 instructions executed every 1 ms) X_{IN} pin sine wave input (f_{IN} = 4.5 MHz, V_{IN} = V_{DD}), T_A = 25 °C		0.45	0.90	mA
Data retention voltage	V_{DDR1}	Power failure detection by timer FF. During crystal oscillation	3.5		5.5	V
	V_{DDR2}	Power failure detection by timer FF. When crystal oscillation stops	2.2		5.5	V
	V_{DDR3}	Retention of data memory (RAM)	2.0		5.5	V
Data retention current	I_{DDR1}	When crystal oscillation stops T_A = 25 °C		2	15	μA
	I_{DDR2}	When crystal oscillation stops V_{DD} = 5.0 V, T_A = 25 °C		2	10	μA
Intermediate-level output voltage	V_{OM1}	COM_0, COM_1 V_{DD} = 5 V	2.3	2.5	2.7	V
High-level input voltage	V_{IH1}	$P0A_0$-$P0A_3$, $P0B_0$-$P0B_3$, $P0C_0$-$P0C_3$, $P1A_0$-$P1A_3$, $P1D_0$-$P1D_3$ CE, INT_0	0.8 V_{DD}			V
	V_{IH2}	$P0D_0$-$P0D_3$	0.6 V_{DD}			V
Low-level input voltage	V_{IL}	$P0A_0$-$P0A_3$, $P0B_0$-$P0B_3$, $P0C_0$-$P0C_3$, $P0D_0$-$P0D_3$, $P1A_0$-$P1A_3$, $P1D_0$-$P1D_3$ CE, INT_0			0.2 V_{DD}	V
High-level output current	I_{OH1}	$P0A_0$, $P0A_1$, $P1A_0$-$P1A_3$, $P2A_0$ V_{OH} = V_{DD} – 2 V, V_{DD} = 5 V, T_A = 25 °C	–2.0	–10		mA
	I_{OH2}	$P0B_0$-$P0B_3$, $P0C_0$-$P0C_3$, $P1B_0$, $P1C_0$-$P1C_3$ V_{OH} = V_{DD} = –1 V	–1.0	–5.0		mA
	I_{OH3}	LCD_0-LCD_{29}, EO_0, EO_1 V_{OH} = V_{DD} – 1 V	–1.0	–4.0		mA
Low-level output current	I_{OL1}	$P0A_0$-$P0A_3$, $P1A_0$-$P1A_3$, $P2A_0$ V_{OL} = 2 V, V_{DD} = 5 V, T_A = 25 °C	5.0	15.0		mA
	I_{OL2}	$P0B_0$-$P0B_3$, $P0C_0$-$P0C_3$, $P1B_0$, $P1C_0$-$P1C_3$ V_{OL} = 1 V	1.0	7.0		mA
	I_{OL3}	LDC_0-LDC_{29}, EO_0, EO_1 V_{OL} = 1 V	1.0	3.5		mA
	I_{OL4}	$P1B_1$-$P1B_3$ V_{OL} = 1 V	1.0	2.0		mA
High-level input current	I_{IH1}	With VCOH pulled down V_{IH} = V_{DD}	0.1	0.8		mA
	I_{IH2}	With VCOL pulled down V_{IH} = V_{DD}	0.1	0.8		mA
	I_{IH3}	With X_{IN} pulled down V_{IH} = V_{DD}	0.1	1.3		mA
	I_{IH4}	With $P0D_0$ through $P0D_3$ pulled down V_{IH} = V_{DD}	0.05	0.13	0.30	mA
Output off leakage current	I_{L1}	$P0A_2$, $P0A_3$ V_{OH} = V_{DD}			500	nA
	I_{L2}	$P1B_1$-$P1B_3$ V_{OH} = 16 V			500	nA
	I_{L3}	EO_0, EO_1 V_{OH} = V_{DD}, V_{OL} = 0 V			±100	nA

AC Characteristics (T_A = –40 to +85 °C, V_{DD} = 4.5 to 5.5 V)

Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Operating frequency	f_{IN1}	VCOL MF mode, sine wave input V_{IN} = 0.2 V_{p-p}	0.5		30	MHz
	f_{IN2}	VCOL HF mode, sine wave input V_{IN} = 0.2 V_{p-p}	5		40	MHz
	f_{IN3}	VCOH sine wave input V_{IN} = 0.2 V_{p-p}	9		150	MHz
	f_{IN4}	AMIFC sine wave input V_{IN} = 0.3 V_{p-p}	0.1		1	MHz
	f_{IN5}	AMIFC sine wave input V_{IN} = 0.05 V_{p-p}	0.44		0.46	MHz
	f_{IN6}	FMIFC sine wave input V_{IN} = 0.3 V_{p-p}	5		15	MHz
	f_{IN7}	FMIFC sine wave input V_{IN} = 0.06 V_{p-p}	10.5		10.9	MHz
AD conversion resolution					6	bit
AD conversion total error		T_A = –10 to +50 °C		±1	±1.5	LSB

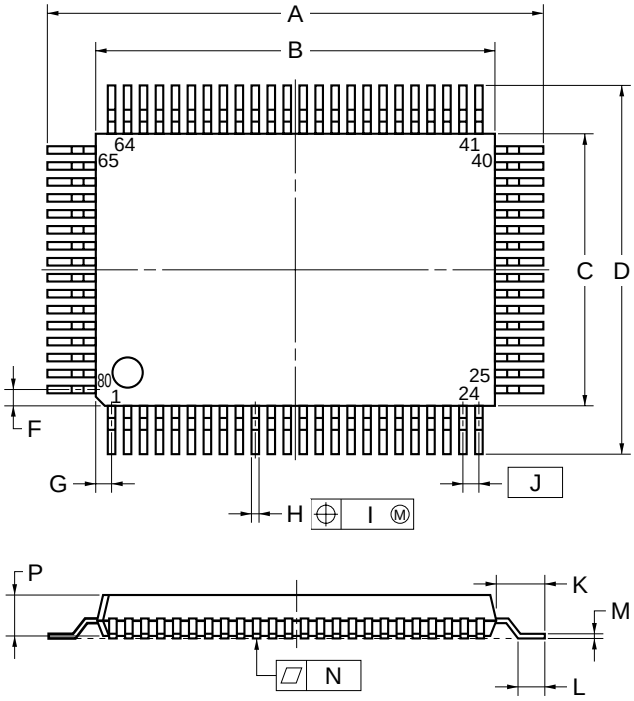
Reference Characteristics

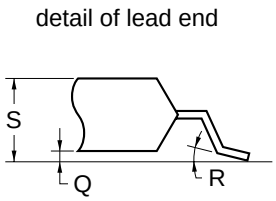
Parameter	Symbol	Condition	MIN.	TYP.	MAX.	Unit
Supply current	I_{DD3}	When CPU and PLL operate VCOH sine wave input f_{IN} = 150 MHz, V_{IN} = 0.3 V_{p-p} V_{DD} = 5 V, T_A = 25 °C		15		mA
High-level output current	I_{OH3}	COM_0, COM_1 V_{OH} = V_{DD} – 1 V		–0.2		mA
Intermediate-level output current	I_{OM1}	COM_0, COM_1 V_{OM} = V_{DD} – 1 V		–20		μA
	I_{OM2}	COM_0, COM_1 V_{OM} = 1 V		20		μA
Low-level output current	I_{OL5}	COM_0, COM_1 V_{OL} = 1 V		0.2		mA

26. PACKAGE

Package of mass-produced model

80 PIN PLASTIC QFP (14×20)





NOTE

Each lead centerline is located within 0.15 mm (0.006 inch) of its true position (T.P.) at maximum material condition.

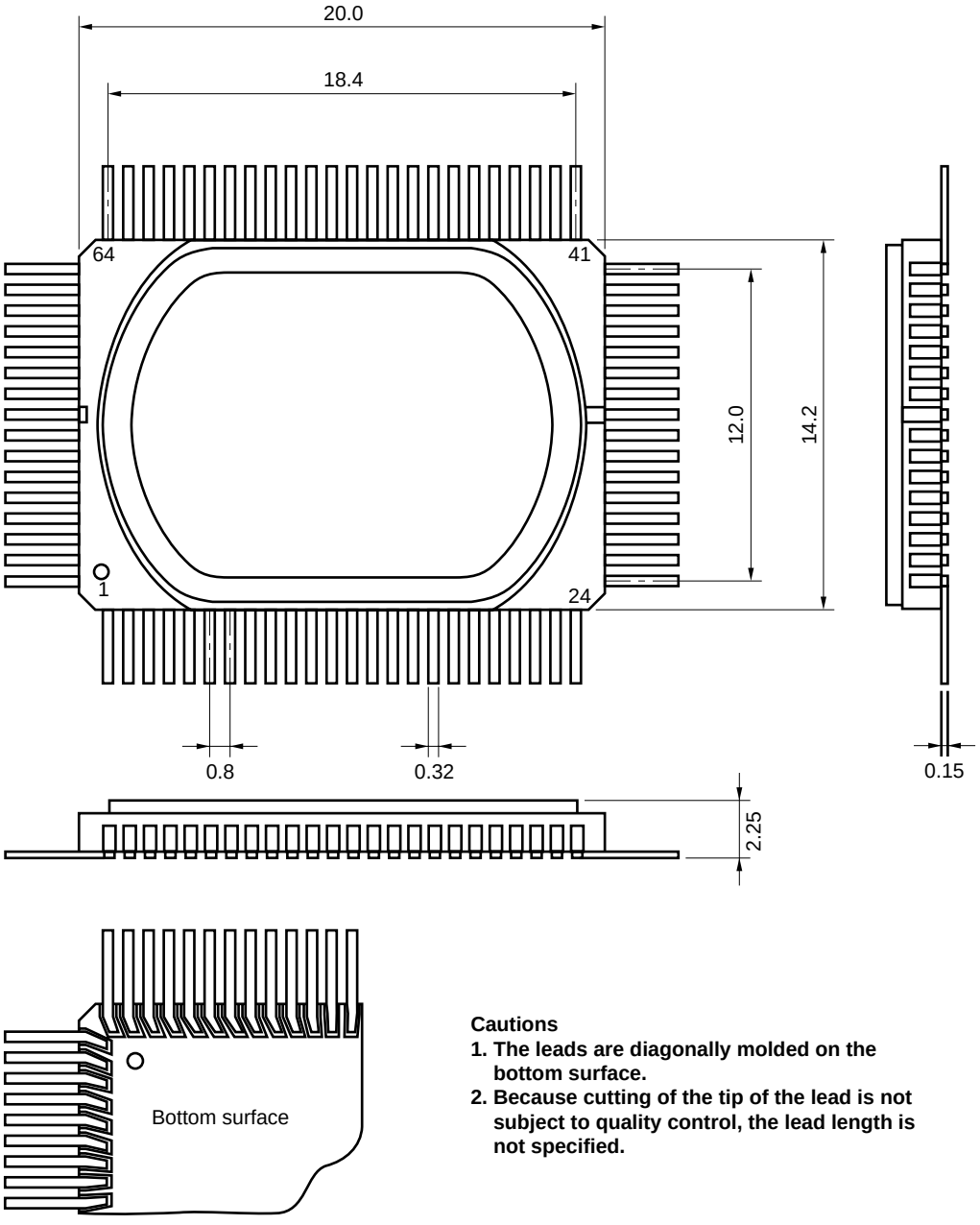
ITEM	MILLIMETERS	INCHES
A	23.2±0.2	$0.913^{+0.009}_{-0.008}$
B	20.0±0.2	$0.787^{+0.009}_{-0.008}$
C	14.0±0.2	$0.551^{+0.009}_{-0.008}$
D	17.2±0.2	0.677±0.008
F	1.0	0.039
G	1.8	0.031
H	0.35±0.10	$0.014^{+0.004}_{-0.005}$
I	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
K	1.6±0.2	0.063±0.008
L	0.8±0.2	$0.031^{+0.009}_{-0.008}$
M	$0.15^{+0.10}_{-0.05}$	$0.006^{+0.004}_{-0.003}$
N	0.10	0.004
P	2.7	0.106
Q	0.125±0.075	0.005±0.003
R	5°±5°	5°±5°
S	3.0 MAX.	0.119 MAX.

S80GF-80-3B9-3

Caution The ES model differs from the mass-produced model in package and materials. Refer to the package drawing of the ES model.

Package of ES model

ES 80-PIN CERAMIC QFP (REFERENCE) (UNIT: mm)



Cautions

1. **The leads are diagonally molded on the bottom surface.**
2. **Because cutting of the tip of the lead is not subject to quality control, the lead length is not specified.**

27. RECOMMENDED SOLDERING CONDITIONS

Solder the μPD17016 and 17017 under the following recommended conditions.

For details of the recommended soldering conditions, refer to Information document **Semiconductor Device Mounting Technology Manual (C10535E)**.

For the soldering methods and conditions other than those recommended, consult NEC.

★ **Table 27-1. Soldering Conditions of Surface Mount Type**

μPD17016GF-xxx-3B9: 80-pin plastic QFP (14 × 20 mm)
μPD17017GF-xxx-3B9: 80-pin plastic QFP (14 × 20 mm)

Soldering Method	Soldering Condition	Symbol of Recommended Soldering Condition
Infrared reflow	Package peak temperature: 235 °C, Time: 30 seconds max. (210 °C min.), Number of times: 2 max., Number of days: 7Note (After this, prebaking at 125 °C for 20 hours is necessary.) <Precaution> Products other than those supplied in heat-resistance tray (magazine, taping, and heat-labile tray) cannot be baked in their packs.	IR35-207-2
VPS	Package peak temperature: 215 °C, Time: 40 seconds max. (200 °C min.), Number of times: 2 max., Number of days: 7Note (After this, prebaking at 125 °C for 20 hours is necessary.) <Precaution> Products other than those supplied in heat-resistance tray (magazine, taping, and heat-labile tray) cannot be baked in their packs.	VP15-207-2
Wave soldering	Solder bath temperature: 260 °C max., Time: 10 seconds max., Number of times: 1, Preheating temperature: 120 °C max. (package surface temperature), Number of days: 7Note (After this, prebaking at 125 °C for 20 hours is necessary.)	WS60-207-1
Partial heating	Pin temperature: 300 °C max., Time: 3 seconds max. (per side of device)	–

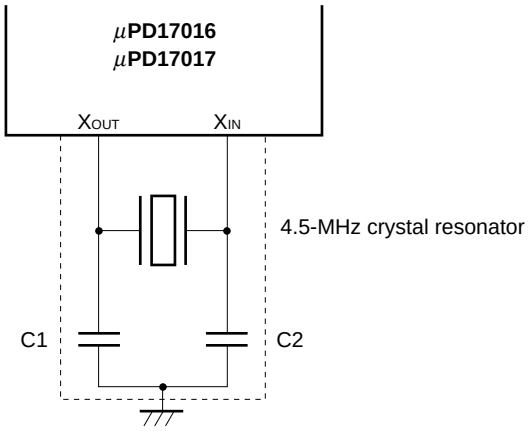
Note The number of days for which the device can be left at 25 °C, 65 % RH max. after the dry pack has been opened.

Caution Do not use two or more soldering methods in combination (except the partial heating method).

APPENDIX A. NOTE ON CONNECTING CRYSTAL RESONATOR

When using the system clock oscillation circuit, the following points must be noted in wiring the portion indicated by the dotted line in the figure below to prevent adverse influence from wiring capacitance.

- Keep the wiring length as short as possible.
- If capacitances C1 and C2 are too high, the oscillation start characteristic may be degraded or the current consumption may increase.
- Generally, connect the trimmer capacitor for adjusting the oscillation frequency to the X_{IN} pin. Depending on the crystal resonator used, however, the oscillation stabilization differs. Therefore, evaluate the crystal resonator actually to be used.
- The crystal frequency cannot be accurately adjusted because of probe capacitance if an emulation probe is connected to the X_{OUT} or X_{IN} pin. Adjust the frequency while measuring the LCD drive waveform (125 Hz) or VCO oscillation frequency.



APPENDIX B. DIFFERENCES AMONG μPD17016, 17017, 17003A, 17005, AND 17010

(1) Hardware

Item	μPD17010	μPD17003A	μPD17005
ROM	16 KB (7932 × 16 bits)	8 KB (3836 × 16 bits)	16 KB (7932 × 16 bits)
RAM	432 × 4 bits	320 × 4 bits	432 × 4 bits
Output port	13 pins (+30: LCD segment pins)	9 pins (+30: LCD segment pins)	
Control register	41 × 4 bits	33 × 4 bits	
General register pointer	5 bits	4 bits	
Stack level	9 levels	7 levels	
Serial interface	2 modes (3 channels): serial I/O mode, I^2C bus mode		
	• SIO0 clock 37.5, 75, 112.5, 225 kHz • SIO1 clock External, 37.5, 75, 450 kHz • Hysteresis characteristics for SCL, SDA, $\overline{SCK_0}$, $\overline{SCK_1}$, SI_0, and SI_1 pins	• SIO1 clock 75, 150, 225, 450 kHz • SIO2 clock External, 75, 150, 450 kHz	
D/A converter frequency	4394.5 Hz	878.9 Hz	
D/A converter output (PWM)	3 channels		
Interrupt	• 6 sources External: 1 source (INT_0 pin) Internal : 4 sources (TM, BTM1, SIO0, IFC) External/internal: 1 source (INT_1 pin or overflow of timer/counter) • Interrupt priority (vector address) 1. (6H) INT_0 pin 2. (5H) INT_1 pin (shared with overflow of timer/counter) 3. (4H) 12-bit timer 4. (3H) Basic timer 1 5. (2H) Serial interface 0 6. (1H) Frequency counter • Automatic saving of system register (3 levels) (WR, BANK, RP, PSWORD) • Address change of IRQxxx flag	• 5 sources External: 2 sources (INT_0 and INT_1 pins) Internal : 3 sources (TM, SIO1, IFC) • Interrupt priority (vector address) 1. (5H) INT_0 pin 2. (4H) INT_1 pin 3. (3H) Timer 4. (2H) Serial interface 1 5. (1H) Frequency counter • Automatic saving of system register (4 levels) (BANK, IXE)	
Timer	• Basic timer 0 carry (1, 5, 100, 250 ms) • Basic timer 1 interrupt (1, 5, 100, 250 ms) • 12-bit timer (10 μs to 4095 ms)	• Timer carry (1, 5, 100, 250 ms) • Timer interrupt (1, 5, 100, 250 ms)	
Frequency counter	Frequency measurement (IF measurement) and external gate width measurement functions selectable		
Clock generator port (CGP)	• CGP output: 1 line • SG (Signal Generator) function • VDP (Variable Duty Pulse) function		
PLL frequency synthesizer phase comparator	Provided (Unlock detection by program. Unlock FF delay time selectable)		
Operational amplifier for PLL frequency synthesizer lowpass filter	None	Provided	
Package	80-pin plastic QFP (14 × 20 mm)		

μPD17016	μPD17017
8 KB (3836 × 16 bits)	16 KB (7932 × 16 bits)
320 × 4 bits	432 × 4 bits
9 pins (+8: LCD segment pins)	
19 × 4 bits	
4 bits	
7 levels	
1 channel: serial I/O mode	
• SIO1 clock External, 75, 150, 450 kHz	
878.9 Hz	
2 channels	
• 2 sources External: 1 source (INT_0 pin) Internal : 1 source (timer) • Interrupt priority (vector address) 1. (5H) INT_0 pin 2. (3H) Timer • Automatic saving of system register (4 levels) (BANK, IXE)	
• Timer carry (1, 5, 100, 250 ms) • Timer interrupt (1, 5, 100, 250 ms)	
Frequency measurement function only	
None (BEEP output provided)	
Internal (without unlock FF delay time select function)	
None	
80-pin plastic QFP (14 × 20 mm)	

(2) Software

Item	μPD17010	μPD17003A	μPD17005	μPD17016	μPD17017
Setting of LCD segment output or output port of $LCD_{22}/P0E_0$ through $LCD_{29}/P0F_3$ pins	Write (POKE) instruction and flag manipulation embedded macroinstructions can be used.			Manipulated by software macro supplied with device file.	
Setting of output port or PWM output of $P1B_1/PWM_0$ and $P1B_2/PWM_1$ pin					
Interrupt request lag	Write (POKE) instruction and flag manipulation embedded macroinstructions can be used.	If write (POKE) instruction is used, error occurs during assembly. If flag manipulation embedded macroinstruction is used, error occurs during assembly. Subjected flags • IRQIFC • IRQSIO1 • IRQTM • IRQ1 • IRQ0 To write above flags, use macro library IRQ.MAC.			
Enabling interrupt	Write (POKE) instruction and flag manipulation embedded macroinstructions can be used.				
Level judge of INT_0 pin	Write (POKE) instruction and embedded macroinstructions SKTn and SKFn can be used.				
IF counter control register	Read (PEEK) instruction is used. At this time, "0" is always read. SETn instruction can be also used.	If read (PEEK) instruction is used, error occurs during assembly. Therefore, macroinstructions that creates PEEK instruction to object when they are expanded to SETn instruction must not be used. Subjected flags • IFCRES • IFCSTRT To read above flags, use macro library IFCSET.LIB.			
BEEP output setting	None			Manipulated by software macro supplied with device file.	

(3) Development tools

Item		μPD17010	μPD17003A	μPD17005	μPD17016	μPD17017
Hardware	SE board	SE-17010				
	Emulation probe	EP-17003GF				
Software	Device file	AS17010	AS17003	AS17005	AS17016	
	Macro library	None	• IFCSET.LIB • IRQ.MAC		• IFCSET.LIB • D17016.INC	• IFCSET.LIB • D17017.INC
One-time PROM		μPD17P010	μPD17P005			

(4) Notes on reserved word names

The reserved words of the control registers of the μPD17016 and 17017 are different from some reserved words of the μPD17003A and 17005.

The following table shows the differences in the reserved words between the μPD17016/17017 and μPD17003A/17005. Note that this table shows only the different reserved words.

Item	μPD17003A	μPD17005	μPD17016	μPD17017
Timer	TMMD3		BTM1CK1	
	TMMD2		BTM1CK0	
	TMMD1		BTM0CK1	
	TMMD0		BTM0CK0	
	TMCY		BTM0CY	
PLL frequency synthesizer	PLULDLY3		None	
	PLULDLY2			
	PLULDLY1			
	PLULDLY0			
	PLLRFMD3		PLLRFCK3	
	PLLRFMD2		PLLRFCK2	
	PLLRFMD1		PLLRFCK1	
	PLLRFMD0		PLLRFCK0	
D/A converter	PWM2ON		None	
	PWM1ON[Note]			
	PWM0ON[Note]			
	CGPON			
LCD controller/driver	P0YON		None	
	P0XON			
	P0EON[Note]			
	P0FON[Note]			
Frequency counter	IFCG		IFCGOSTT	
Serial interface	SIO2TS		SIO1TS	
	SIO2HIZ		SIO1HIZ	
	SIO2CK1		SIO1CK1	
	SIO2CK0		SIO1CK0	
	SIO1CH		None	
	SB			
	SIO1MS			
	SIO1TX			
	SBACK			

Note These reserved words are manipulated by software macros with the μPD17016 and 17017.

Item	μPD17003A	μPD17005	μPD17016	μPD17017
Serial interface	SIO1NWT		None	
	SIO1WRQ1			
	SIO1WRQ0			
	SIO1SF8			
	SIO1SF9			
	SBBSY			
	SBSTT			
	SIO1IMD3			
	SIO1IMD2			
	SIO1IMD1			
	SIO1IMD0			
	SIO1CK3			
	SIO1CK2			
	SIO1CK1			
	SIO1CK0			
Interrupt	INT1		None	
	INT0**Note**			
	IEG1			
	IEG0			
	IPIFC			
	IPSIO1			
	IPTM**Note**			
	IP1			
	IP0			
	IRQIFC			
	IRQSIO1			
	IRQTM**Note**			
	IRQ1			
	IRQ0**Note**			

Note These reserved words are manipulated by software macros with the μPD17016 and 17017.

APPENDIX C. DEVELOPMENT TOOLS

The following development tools are available to support development of the program of the μPD17016 and 17017.

Hardware

Name	Description
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 commonly used with any model in 17K Series. IE-17K and IE-17K-ET are connected to host machine, which is PC-9800 series or IBM PC/AT™, with RS-232-C. EMU-17K is mounted to expansion slot of PC-9800 series. When these in-circuit emulators are used in combination with system evaluation board (SE board) dedicated to each model, they operate as emulators corresponding to that model. When man-machine interface software *SIMPLEHOST*® is used, a more sophisticated debugging environment can be created. EMU-17K has a function for checking the contents of data memory real-time.
SE board (SE-17010)	SE-17010 is SE board for μPD17016 and 17017. This SE board alone can be used to evaluate system or in combination with in-circuit emulator for debugging.
Emulation probe (EP-17003GF)	EP-17003GF is emulation probe for μPD17016GF and 17017GF. It connects SE board and target system when used with EV-9200G-80[Note 3].
Conversion socket (EV-9200G-80[Note 3])	EV-9200G-80 is conversion socket for 80-pin plastic QFP (14 × 20 mm) and is used to connect EP-17003GF and target system.
PROM programmer (AF-9703[Note 4], AF-9704[Note 4], AF-9705[Note 4], AF-9706[Note 4])	AF-9703, AF-9704, AF-9705, and AF-9706 are PROM programmers corresponding to μPD17P005. They can program the μPD17P005 when connected to program adapter AF-9803.
Program adapter (AF-9803[Note 4])	AF-9803 is adapter for programming μPD17P005, and is used in combination with AF-9703, AF-9704, AF-9705, or AF-9706.

Notes 1. Low-cost model: external power supply type

2. This is a product by I.C Corp. For details, consult I.C Corp. (Tokyo, 03-3447-3793)

3. One EV-9200G-80 is provided with the EP-17003GF. Five EV-9200G-80s are also available as a set.

4. These are products of Ando Electric Co., Ltd. For details, consult Ando Electric Co, Ltd. (Tokyo, 03-3733-1163).

Software

Name	Description	Host Machine	OS		Supply Media	Order Code
17K series assembler (AS17K)	AS17K is an assembler that can be commonly used with any model of 17K series. To develop program of μPD17016 and 17017, this AS17K is used in combination with device file (AS17016).	PC-9800 series	MS-DOS™		5"2HD	μS5A10AS17K
					3.5"2HD	μS5A13AS17K
		IBM PC/AT	PC DOS™		5"2HC	μS7B10AS17K
					3.5"2HC	μS7B13AS17K
Device file (AS17016)	AS17016 contains device file for μPD17016 and 17017. It is used in combination with 17K series assembler (AS17K).	PC-9800 series	MS-DOS		5"2HD	μS5A10AS17016
					3.5"2HC	μS5A13AS17016
		IBM PC/AT	PC DOS		5"2HC	μS7B10AS17016
					3.5"2HC	μS7B13AS17016
Support software (*SIMPLEHOST*)	*SIMPLEHOST* is software that serves as man-machine interface on Windows™ when program is developed by using in-circuit emulator and personal computer.	PC-9800 series	MS-DOS	Windows	5"2HD	μS5A10IE17K
					3.5"2HD	μS5A13IE17K
		IBM PC/AT	PC DOS		5"2HC	μS7B10IE17K
					3.5"2HC	μS7B13IE17K

Remark The following versions of OSs are supported.

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]

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.)
Mountain View, California
Tel: 800-366-9782
Fax: 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.1.
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.
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: 8-63 80 820
Fax: 8-63 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: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Sao Paulo-SP, Brasil
Tel: 011-889-1680
Fax: 011-889-1689

J96. 3

***SIMPLEHOST* is a registered trademark of NEC Corporation.**

MS-DOS and Windows are trademarks of Microsoft Corporation.

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.

M4 96.5