

## Description

### Description

The M16C/61 group of single-chip microcomputers are built using the high-performance silicon gate CMOS process using a M16C/60 Series CPU core and are packaged in a 100-pin plastic molded QFP. These single-chip microcomputers operate using sophisticated instructions featuring a high level of instruction efficiency. With 1M bytes of address space, they are capable of executing instructions at high speed. They also feature a built-in multiplier and DMAC, making them ideal for controlling office, communications, industrial equipment, and other high-speed processing applications.

The M16C/61 group includes a wide range of products with different internal memory types and sizes and various package types.

### Features

- Memory capacity ..... ROM (See Figure 1.1.4. ROM Expansion)  
RAM 4K to 10K bytes
- Shortest instruction execution time ..... 100ns (f(XIN)=10MHz)
- Supply voltage ..... 4.0 to 5.5V (f(XIN)=10MHz)  
2.7 to 5.5V (f(XIN)=7MHz with software one-wait)
- Low power consumption ..... 18mW ( f(XIN)=7MHz, with software one-wait, VCC = 3V)
- Interrupts ..... 20 internal and 5 external interrupt sources, 4 software  
interrupt sources; 7 levels (including key input interrupt)
- Multifunction 16-bit timer ..... 5 output timers + 3 input timers
- Serial I/O (UART or clock synchronous) ..... 3 channels
- DMAC ..... 2 channels (trigger: 16 sources)
- A-D converter ..... 10 bits X 8 channels  
(Expandable up to 10 channels)
- D-A converter ..... 8 bits X 2 channels
- CRC calculation circuit ..... 1 circuit
- Watchdog timer ..... 1 line
- Programmable I/O ..... 87 lines
- Input port ..... 1 line (P85 shared with  $\overline{\text{NMI}}$  pin)
- Memory expansion ..... Available (to a maximum of 1M bytes)
- Chip select output ..... 4 lines
- Clock generating circuit ..... 2 built-in clock generation circuits  
(built-in feedback resistor, and external ceramic or quartz oscillator)

### Applications

Audio, cameras, office equipment, communications equipment, portable equipment

## -----Table of Contents-----

Central Processing Unit (CPU) .....	11	Timer .....	70
Reset .....	14	Serial I/O .....	87
Processor Mode .....	19	A-D Converter .....	114
Clock Generating Circuit .....	30	D-A Converter .....	124
Protection .....	39	CRC Calculation Circuit .....	126
Interrupts .....	40	Programmable I/O Ports .....	128
Watchdog Timer .....	59	Electrical Characteristics .....	142
DMAC .....	61		

### Pin Configuration

Figures 1.1.1 and 1.1.2 show the pin configurations (top view).

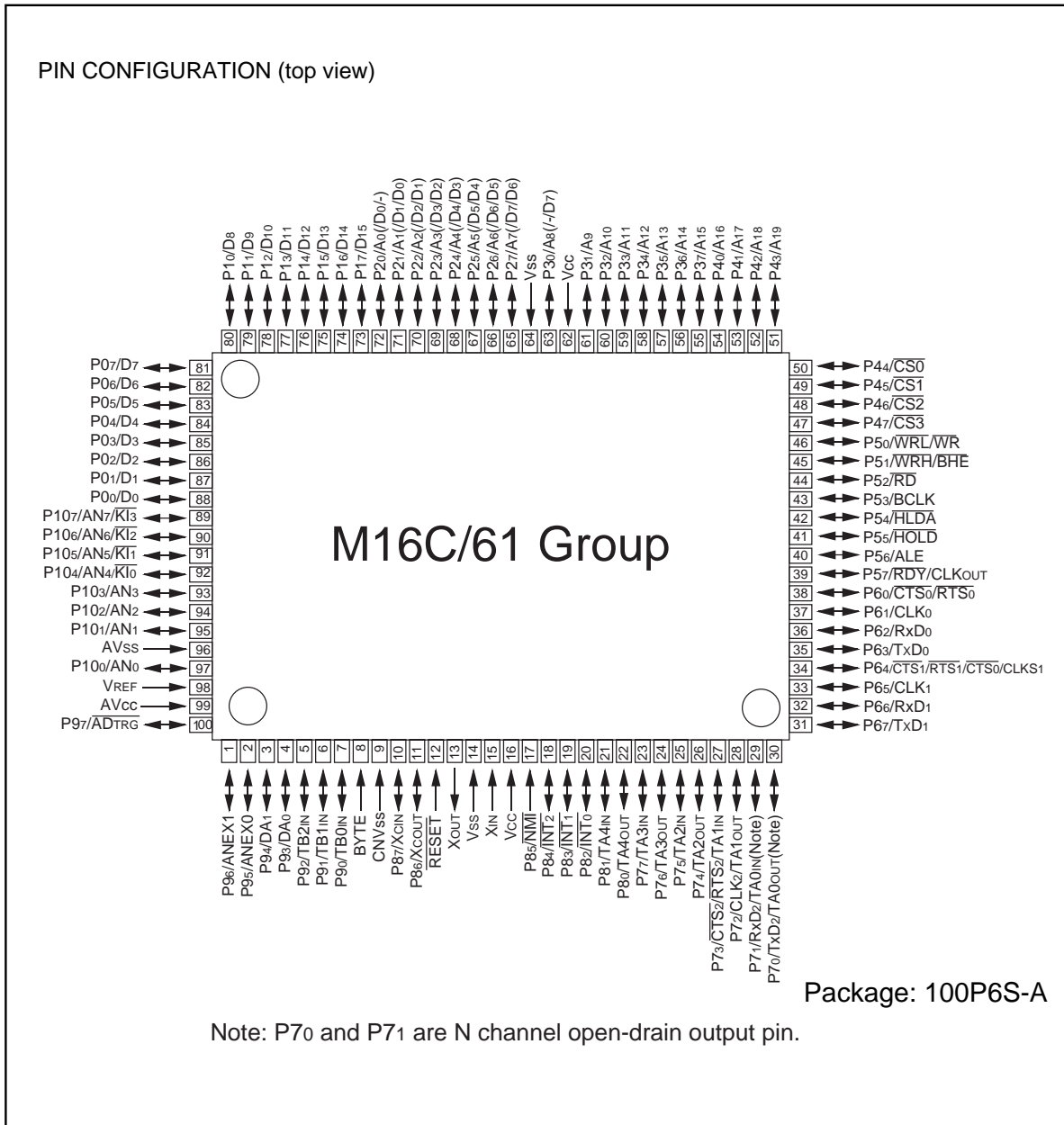


Figure 1.1.1. Pin configuration (top view)

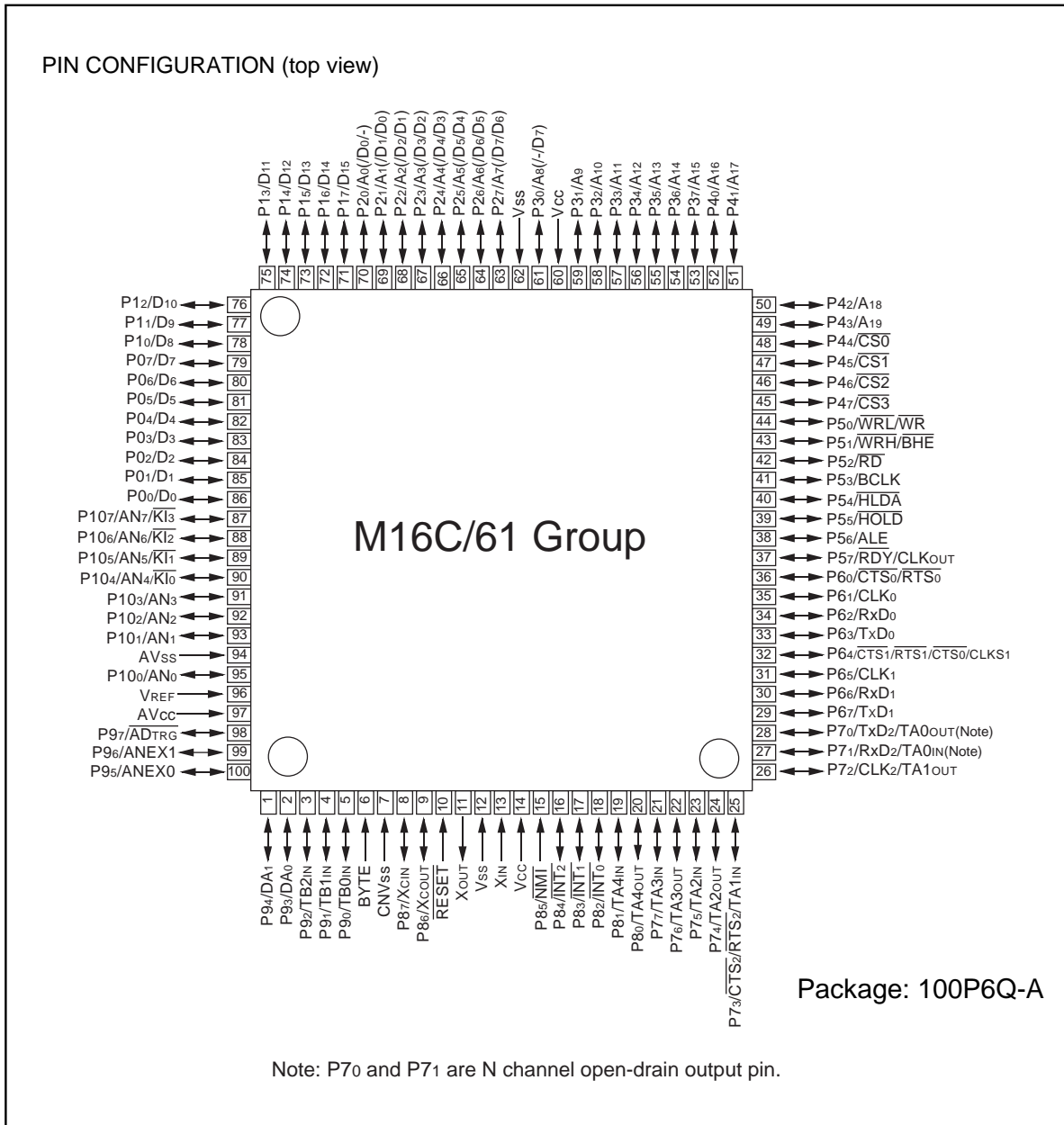


Figure 1.1.2. Pin configuration (top view)

Description

Block Diagram

Figure 1.1.3 is a block diagram of the M16C/61 group.

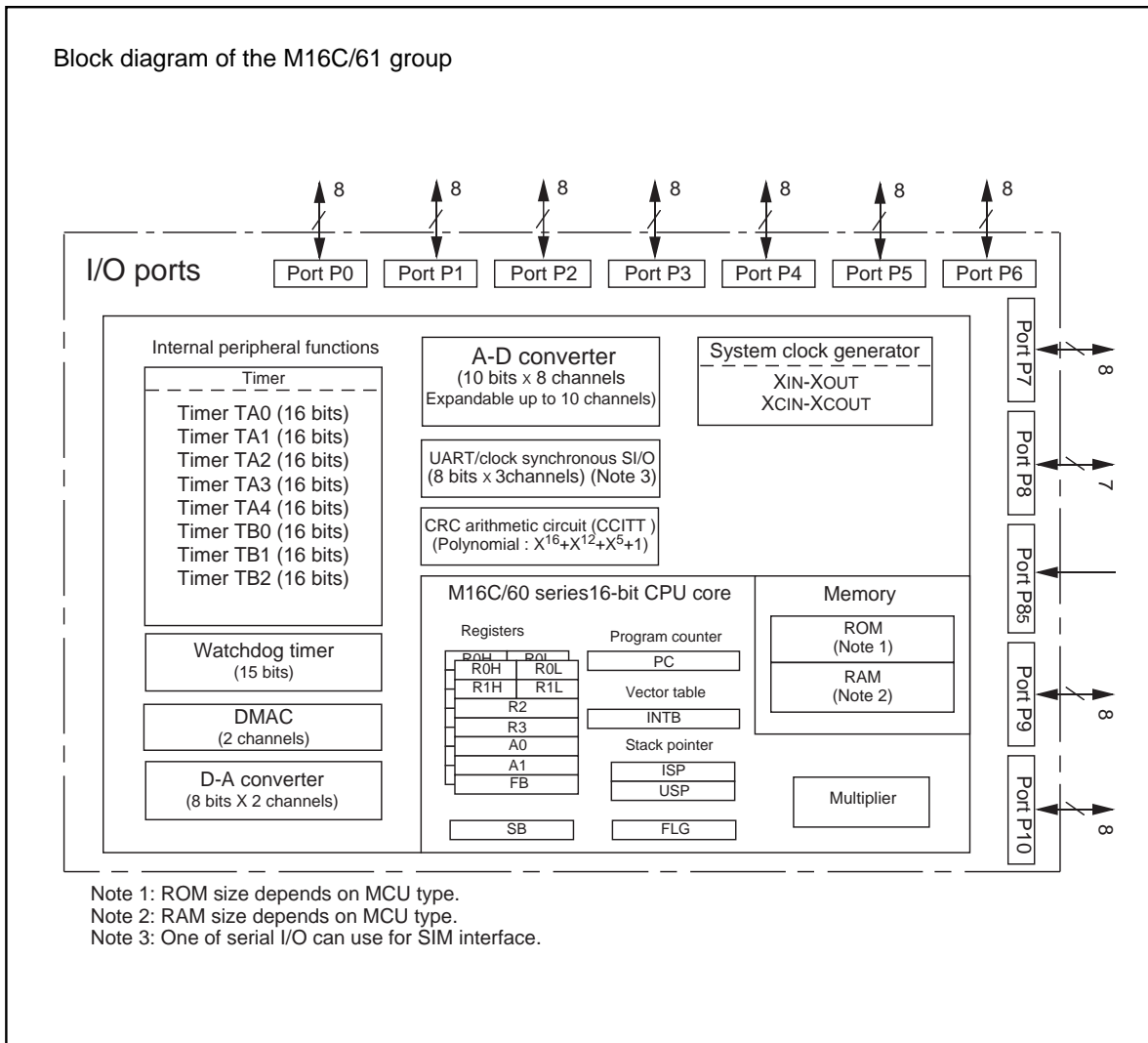


Figure 1.1.3. Block diagram of M16C/61 group

## Performance Outline

Table 1.1.1 is a performance outline of M16C/61 group.

**Table 1.1.1. Performance outline of M16C/61 group**

Item		Performance
Number of basic instructions		91 instructions
Shortest instruction execution time		100ns( $f(X_{IN})=10\text{MHz}$ )
Memory capacity	ROM	(See the Figure 4. ROM Expansion)
	RAM	4K to 10K bytes
I/O port	P0 to P10 (except P85)	8 bits x 10, 7 bits x 1
Input port	P85	1 bit x 1
Multifunction timer	TA0, TA1, TA2, TA3, TA4	16 bits x 5
	TB0, TB1, TB2	16 bits x 3
Serial I/O	UART0, UART1, UART2	(UART or clock synchronous) x 3
A-D converter		10 bits x (8 + 2) channels
D-A converter		8 bits x 2
DMAC		2 channels (trigger: 16 sources)
CRC calculation circuit		CRC - CCITT
Watchdog timer		15 bits x 1 (with prescaler)
Interrupt		20 internal and 5 external sources, 4 software sources, 7 levels
Clock generating circuit		2 built-in clock generation circuits (built-in feedback resistor, and external ceramic or quartz oscillator)
Supply voltage		4.0 to 5.5V ( $f(X_{IN}) = 10\text{MHz}$ ) 2.7 to 5.5V ( $f(X_{IN})=7\text{MHz}$ with software one-wait)
Power consumption		18mW ( $f(X_{IN}) = 7\text{MHz}$ with software one-wait, $V_{CC} = 3\text{V}$ )
I/O characteristics	I/O withstand voltage	5V
	Output current	5mA
Memory expansion		Available (to a maximum of 1M bytes)
Device configuration		CMOS silicon gate
Package		100-pin plastic mold QFP

## Description

Mitsubishi plans to release the following products in the M16C/61 group:

(1) Support for mask ROM version, external ROM version, one-time PROM version, and EPROM version

(2) ROM capacity

(3) Package

100P6S-A : Plastic molded QFP (mask ROM version and one-time PROM version)

100P6Q-A : Plastic molded QFP (mask ROM version and one-time PROM version)

100D0 : Ceramic LCC (EPROM version)

ROM Size(Byte)	Mask ROM version	One-time PROM version	EPROM version	External ROM version
External ROM				M30612SAFP/GP M30610SAFP/GP
128 K	M30610MCA-XXXFP/GP M30612MCA-XXXFP/GP	M30610ECFP/GP	M30610ECFS	
96 K	M30610MAA-XXXFP/GP M30612MAA-XXXFP/GP			
64 K	M30610M8A-XXXFP/GP M30612M8A-XXXFP/GP			
32 K	M30612M4A-XXXFP/GP	M30612E4FP/GP		

Figure 1.1.4. ROM expansion

The M16C/61 group products currently supported are listed in Table 2.

Table 1.1.2. M16C/61 group

Apr. 1999

Type No	ROM capacity	RAM capacity	Package type	Remarks
M30612M4A-XXXFP	32K byte	4K byte	100P6S-A	Mask ROM version
M30612M4A-XXXGP			100P6Q-A	
M30610M8A-XXXFP	64K byte	10K byte	100P6S-A	
M30610M8A-XXXGP			100P6Q-A	
M30612M8A-XXXFP		4K byte	100P6S-A	
M30612M8A-XXXGP			100P6Q-A	
M30610MAA-XXXFP	96K byte	10K byte	100P6S-A	
M30610MAA-XXXGP			100P6Q-A	
M30612MAA-XXXFP		4K byte	100P6S-A	
M30612MAA-XXXGP			100P6Q-A	
M30610MCA-XXXFP	128K byte	10K byte	100P6S-A	
M30610MCA-XXXGP			100P6Q-A	
M30612MCA-XXXFP		5K byte	100P6S-A	
M30612MCA-XXXGP			100P6Q-A	
M30612E4FP	32K byte	4K byte	100P6S-A	One-time PROM version
M30612E4GP			100P6Q-A	
M30610ECFP	128K byte	10K byte	100P6S-A	
M30610ECGP			100P6Q-A	
M30610ECFS	128K byte	10K byte	100D0	EPROM version (Note)
M30610SAFP	—	10K byte	100P6S-A	External ROM version
M30610SAGP			100P6Q-A	
M30612SAFP	—	4K byte	100P6S-A	
M30612SAGP			100P6Q-A	

Note: Do not use the EPROM version for mass production, because it is a tool for program development (for evaluation).

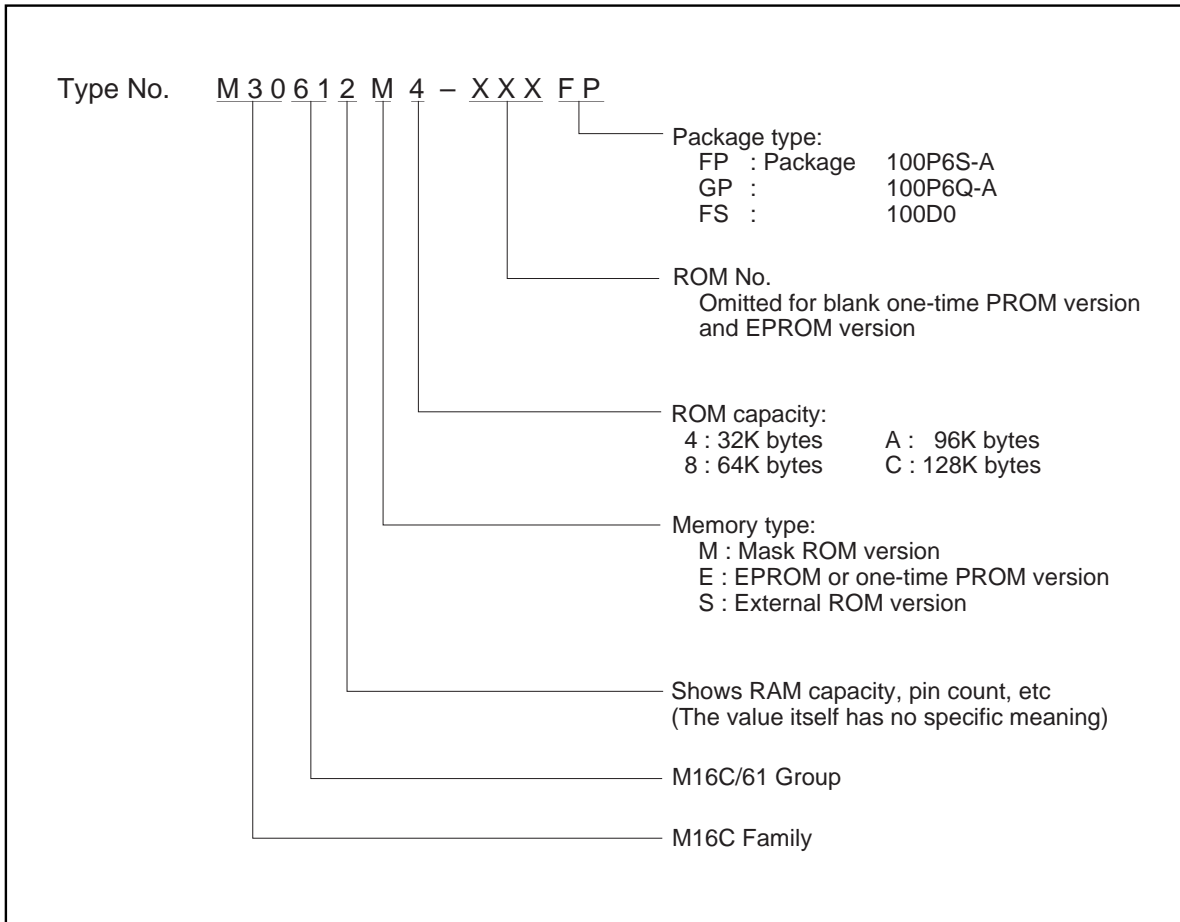


Figure 1.1.5. Type No., memory size, and package

## Pin Description

## Pin Description

Pin name	Signal name	I/O type	Function
VCC, VSS	Power supply input		Supply 2.7 to 5.5 V to the VCC pin. Supply 0 V to the VSS pin.
CNVSS	CNVSS	Input	This pin switches between processor modes. Connect it to the VSS pin when operating in single-chip or memory expansion mode. Connect it to the VCC pin when in microprocessor mode.
$\overline{\text{RESET}}$	Reset input	Input	A "L" on this input resets the microcomputer.
XIN XOUT	Clock input Clock output	Input Output	These pins are provided for the main clock generating circuit. Connect a ceramic resonator or crystal between the XIN and the XOUT pins. To use an externally derived clock, input it to the XIN pin and leave the XOUT pin open.
BYTE	External data bus width select input	Input	This pin selects the width of an external data bus. A 16-bit width is selected when this input is "L"; an 8-bit width is selected when this input is "H". This input must be fixed to either "H" or "L". When operating in single-chip mode, connect this pin to VSS.
AVCC	Analog power supply input		This pin is a power supply input for the A-D converter. Connect this pin to VCC.
AVSS	Analog power supply input		This pin is a power supply input for the A-D converter. Connect this pin to VSS.
VREF	Reference voltage input	Input	This pin is a reference voltage input for the A-D converter.
P00 to P07	I/O port P0	Input/output	This is an 8-bit CMOS I/O port. It has an input/output port direction register that allows the user to set each pin for input or output individually. When used for input in single-chip mode, the port can be set to have or not have a pull-up resistor in units of four bits by software. In memory expansion and microprocessor modes, selection of the internal pull-resistor is not available.
D0 to D7		Input/output	When set as a separate bus, these pins input and output data (D0–D7).
P10 to P17	I/O port P1	Input/output	This is an 8-bit I/O port equivalent to P0.
D8 to D15		Input/output	When set as a separate bus, these pins input and output data (D8–D15).
P20 to P27	I/O port P2	Input/output	This is an 8-bit I/O port equivalent to P0.
A0 to A7		Output	These pins output 8 low-order address bits (A0–A7).
A0/D0 to A7/D7		Input/output	If the external bus is set as an 8-bit wide multiplexed bus, these pins input and output data (D0–D7) and output 8 low-order address bits (A0–A7) separated in time by multiplexing.
A0, A1/D0 to A7/D6		Output Input/output	If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D0–D6) and output address (A1–A7) separated in time by multiplexing. They also output address (A0).
P30 to P37	I/O port P3	Input/output	This is an 8-bit I/O port equivalent to P0.
A8 to A15		Output	These pins output 8 middle-order address bits (A8–A15).
A8/D7, A9 to A15		Input/output Output	If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data (D7) and output address (A8) separated in time by multiplexing. They also output address (A9–A15).
P40 to P47	I/O port P4	Input/output	This is an 8-bit I/O port equivalent to P0.
CS0 to CS3, A16 to A19		Output Output	These pins output CS0–CS3 signals and A16–A19. CS0–CS3 are chip select signals used to specify an access space. A16–A19 are 4 high-order address bits.



## Pin Description

## Pin Description

Pin name	Signal name	I/O type	Function
P50 to P57	I/O port P5	Input/output	This is an 8-bit I/O port equivalent to P0. In single-chip mode, P57 in this port outputs a divide-by-8 or divide-by-32 clock of XIN or a clock of the same frequency as XCIN as selected by software.
$\overline{WRL}$ / $\overline{WR}$ , $\overline{WRH}$ / $\overline{BHE}$ , $\overline{RD}$ , $\overline{BCLK}$ , $\overline{HLDA}$ , $\overline{HOLD}$ ,  $\overline{ALE}$ , $\overline{RDY}$		Output Output Output Output Output Input  Output Input	Output $\overline{WRL}$ , $\overline{WRH}$ ( $\overline{WR}$ and $\overline{BHE}$ ), $\overline{RD}$ , $\overline{BCLK}$ , $\overline{HLDA}$ , and $\overline{ALE}$ signals. $\overline{WRL}$ and $\overline{WRH}$ , and $\overline{BHE}$ and $\overline{WR}$ can be switched using software control. <ul style="list-style-type: none"> <li>■ <math>\overline{WRL}</math>, <math>\overline{WRH}</math>, and <math>\overline{RD}</math> selected With a 16-bit external data bus, data is written to even addresses when the <math>\overline{WRL}</math> signal is "L" and to the odd addresses when the <math>\overline{WRH}</math> signal is "L". Data is read when <math>\overline{RD}</math> is "L".</li> <li>■ <math>\overline{WR}</math>, <math>\overline{BHE}</math>, and <math>\overline{RD}</math> selected Data is written when <math>\overline{WR}</math> is "L". Data is read when <math>\overline{RD}</math> is "L". Odd addresses are accessed when <math>\overline{BHE}</math> is "L". Use this mode when using an 8-bit external data bus.</li> </ul> While the input level at the $\overline{HOLD}$ pin is "L", the microcomputer is placed in the hold state. While in the hold state, $\overline{HLDA}$ outputs a "L" level. $\overline{ALE}$ is used to latch the address. While the input level of the $\overline{RDY}$ pin is "L", the microcomputer is in the ready state.
P60 to P67	I/O port P6	Input/output	This is an 8-bit I/O port equivalent to P0. When used for input in single-chip, memory expansion, and microprocessor modes, the port can be set to have or not have a pull-up resistor in units of four bits by software. Pins in this port also function as UART0 and UART1 I/O pins as selected by software.
P70 to P77	I/O port P7	Input/output	This is an 8-bit I/O port equivalent to P6 (P70 and P71 are N channel open-drain output). Pins in this port also function as timer A0–A3 or UART2 I/O pins as selected by software.
P80 to P84, P86, P87, P85	I/O port P8    I/O port P85	Input/output Input/output Input/output Input	P80 to P84, P86, and P87 are I/O ports with the same functions as P6. Using software, they can be made to function as the I/O pins for timer A4 and the input pins for external interrupts. P86 and P87 can be set using software to function as the I/O pins for a sub clock generation circuit. In this case, connect a quartz oscillator between P86 (XCOUT pin) and P87 (XCIN pin). P85 is an input-only port that also functions for NMI. The NMI interrupt is generated when the input at this pin changes from "H" to "L". The NMI function cannot be cancelled using software. The pull-up cannot be set for this pin.
P90 to P97	I/O port P9	Input/output	This is an 8-bit I/O port equivalent to P6. Pins in this port also function as Timer B0–B2 input pins, D-A converter output pins, A-D converter extended input pins, or A-D trigger input pins as selected by software.
P100 to P107	I/O port P10	Input/output	This is an 8-bit I/O port equivalent to P6. Pins in this port also function as A-D converter input pins. Furthermore, P104–P107 also function as input pins for the key input interrupt function.

## Operation of Functional Blocks

The M16C/61 group accommodates certain units in a single chip. These units include ROM and RAM to store instructions and data and the central processing unit (CPU) to execute arithmetic/logic operations. Also included are peripheral units such as timers, serial I/O, D-A converter, DMAC, CRC calculation circuit, A-D converter, and I/O ports.

The following explains each unit.

## Memory

Figure 1.4.1 is a memory map of the M16C/61 group. The address space extends the 1M bytes from address  $00000_{16}$  to  $FFFFFF_{16}$ . From  $FFFFFF_{16}$  down is ROM. For example, in the M30612M4A-XXXFP, there is 32K bytes of internal ROM from  $F8000_{16}$  to  $FFFFFF_{16}$ . The vector table for fixed interrupts such as the reset and  $\overline{\text{NMI}}$  are mapped to  $FFFDC_{16}$  to  $FFFFFF_{16}$ . The starting address of the interrupt routine is stored here. The address of the vector table for timer interrupts, etc., can be set as desired using the internal register (INTB). See the section on interrupts for details.

From  $00400_{16}$  up is RAM. For example, in the M30612M4A-XXXFP, 4K bytes of internal RAM is mapped to the space from  $00400_{16}$  to  $013FF_{16}$ . In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated.

The SFR area is mapped to  $00000_{16}$  to  $003FF_{16}$ . This area accommodates the control registers for peripheral devices such as I/O ports, A-D converter, serial I/O, and timers, etc. Any part of the SFR area that is not occupied is reserved and cannot be used for other purposes.

The special page vector table is mapped to  $FFE00_{16}$  to  $FFFDB_{16}$ . If the starting addresses of subroutines or the destination addresses of jumps are stored here, subroutine call instructions and jump instructions can be used as 2-byte instructions, reducing the number of program steps.

In memory expansion mode and microprocessor mode, a part of the spaces are reserved and cannot be used. For example, in the M30612M4A-XXXFP, the following spaces cannot be used.

- The space between  $01400_{16}$  and  $03FFF_{16}$
- The space between  $D0000_{16}$  and  $F7FFF_{16}$  (When external area do not expand in memory expansion mode)

Do not expand the external area in single chip mode. A part of internal memory cannot be used depending on MCU.

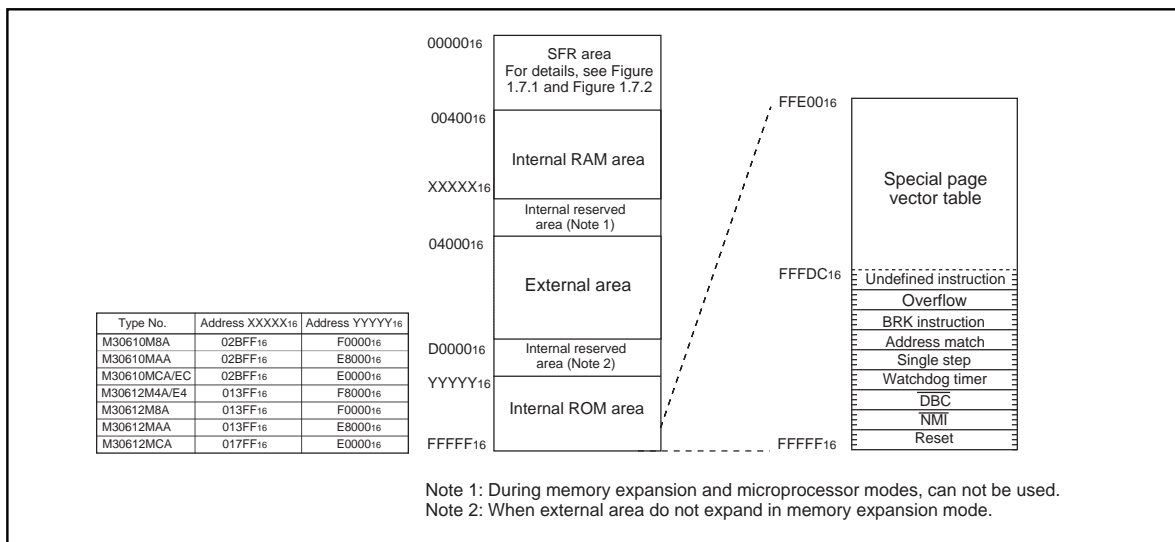


Figure 1.4.1. Memory map

## Central Processing Unit (CPU)

The CPU has a total of 13 registers shown in Figure 1.5.1. Seven of these registers (R0, R1, R2, R3, A0, A1, and FB) come in two sets; therefore, these have two register banks.

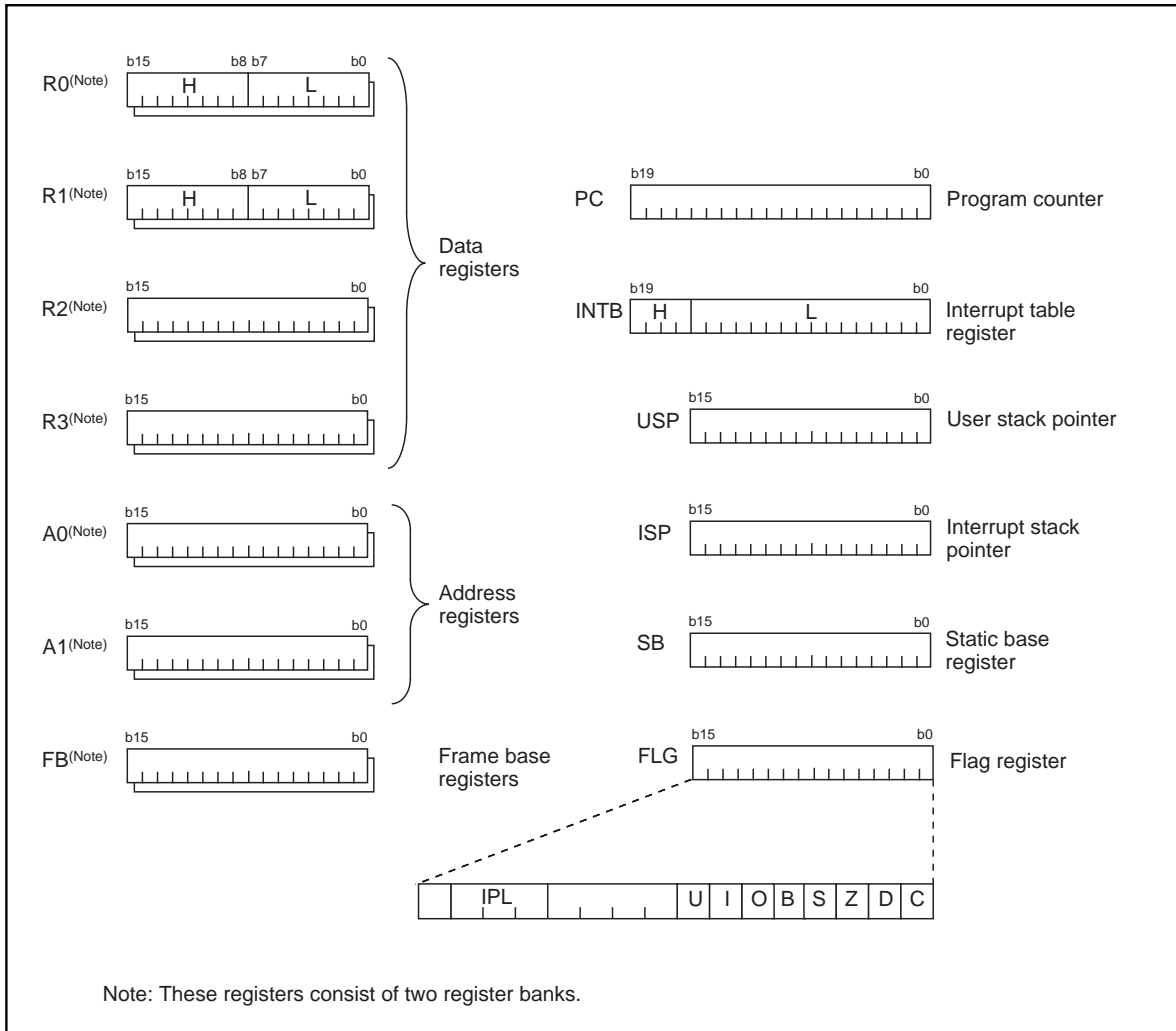


Figure 1.5.1. Central processing unit register

### (1) Data registers (R0, R0H, R0L, R1, R1H, R1L, R2, and R3)

Data registers (R0, R1, R2, and R3) are configured with 16 bits, and are used primarily for transfer and arithmetic/logic operations.

Registers R0 and R1 each can be used as separate 8-bit data registers, high-order bits as (R0H/R1H), and low-order bits as (R0L/R1L). In some instructions, registers R2 and R0, as well as R3 and R1 can use as 32-bit data registers (R2R0/R3R1).

### (2) Address registers (A0 and A1)

Address registers (A0 and A1) are configured with 16 bits, and have functions equivalent to those of data registers. These registers can also be used for address register indirect addressing and address register relative addressing.

In some instructions, registers A1 and A0 can be combined for use as a 32-bit address register (A1A0).

### (3) Frame base register (FB)

Frame base register (FB) is configured with 16 bits, and is used for FB relative addressing.

### (4) Program counter (PC)

Program counter (PC) is configured with 20 bits, indicating the address of an instruction to be executed.

### (5) Interrupt table register (INTB)

Interrupt table register (INTB) is configured with 20 bits, indicating the start address of an interrupt vector table.

### (6) Stack pointer (USP/ISP)

Stack pointer comes in two types: user stack pointer (USP) and interrupt stack pointer (ISP), each configured with 16 bits.

Your desired type of stack pointer (USP or ISP) can be selected by a stack pointer select flag (U flag).

This flag is located at the position of bit 7 in the flag register (FLG).

### (7) Static base register (SB)

Static base register (SB) is configured with 16 bits, and is used for SB relative addressing.

### (8) Flag register (FLG)

Flag register (FLG) is configured with 11 bits, each bit is used as a flag. Figure 1.5.2 shows the flag register (FLG). The following explains the function of each flag:

- **Bit 0: Carry flag (C flag)**

This flag retains a carry, borrow, or shift-out bit that has occurred in the arithmetic/logic unit.

- **Bit 1: Debug flag (D flag)**

This flag enables a single-step interrupt.

When this flag is "1", a single-step interrupt is generated after instruction execution. This flag is cleared to "0" when the interrupt is acknowledged.

- **Bit 2: Zero flag (Z flag)**

This flag is set to "1" when an arithmetic operation resulted in 0; otherwise, cleared to "0".

- **Bit 3: Sign flag (S flag)**

This flag is set to "1" when an arithmetic operation resulted in a negative value; otherwise, cleared to "0".

- **Bit 4: Register bank select flag (B flag)**

This flag chooses a register bank. Register bank 0 is selected when this flag is "0"; register bank 1 is selected when this flag is "1".

- **Bit 5: Overflow flag (O flag)**

This flag is set to "1" when an arithmetic operation resulted in overflow; otherwise, cleared to "0".

- **Bit 6: Interrupt enable flag (I flag)**

This flag enables a maskable interrupt.

An interrupt is disabled when this flag is "0", and is enabled when this flag is "1". This flag is cleared to "0" when the interrupt is acknowledged.

- **Bit 7: Stack pointer select flag (U flag)**

Interrupt stack pointer (ISP) is selected when this flag is “0” ; user stack pointer (USP) is selected when this flag is “1”.

This flag is cleared to “0” when a hardware interrupt is acknowledged or an INT instruction of software interrupt Nos. 0 to 31 is executed.

- **Bits 8 to 11: Reserved area**

- **Bits 12 to 14: Processor interrupt priority level (IPL)**

Processor interrupt priority level (IPL) is configured with three bits, for specification of up to eight processor interrupt priority levels from level 0 to level 7.

If a requested interrupt has priority greater than the processor interrupt priority level (IPL), the interrupt is enabled.

- **Bit 15: Reserved area**

The C, Z, S, and O flags are changed when instructions are executed. See the software manual for details.

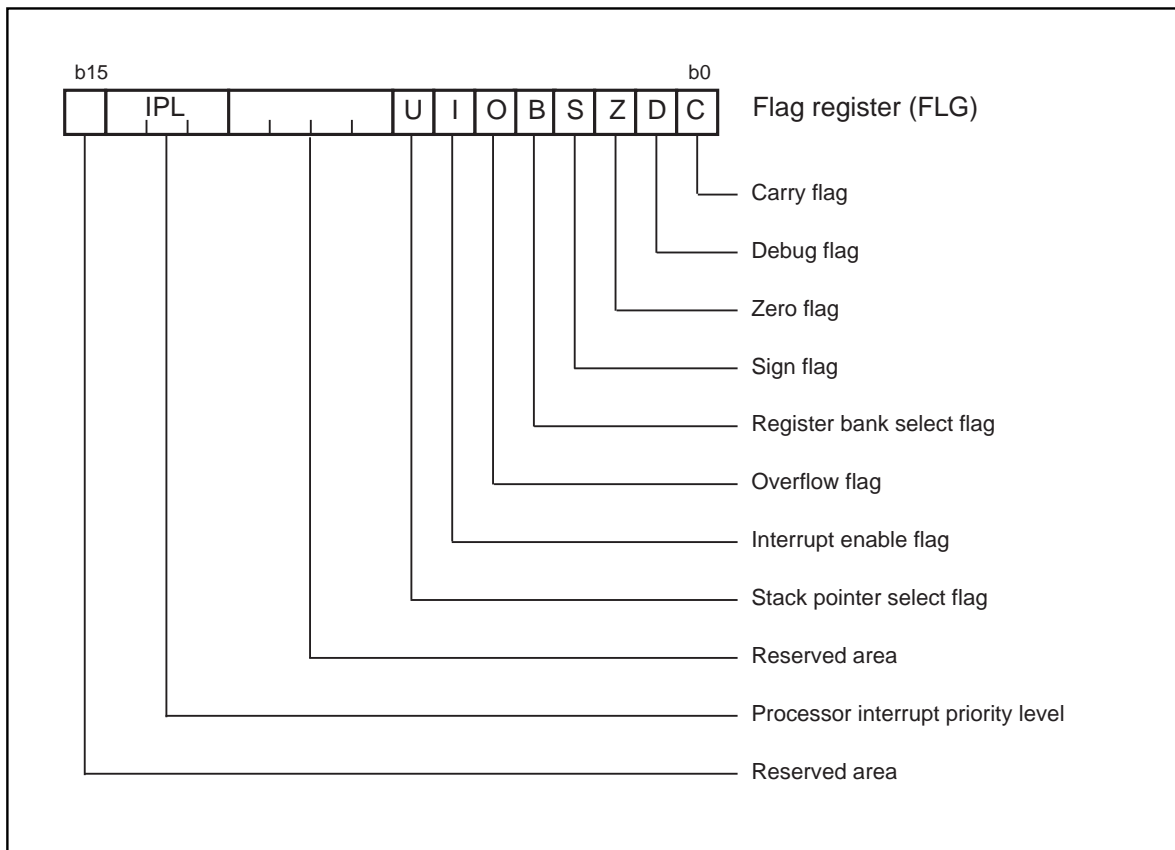


Figure 1.5.2. Flag register (FLG)

## Reset

## Reset

There are two kinds of resets; hardware and software. In both cases, operation is the same after the reset. (See "Software Reset" for details of software resets.) This section explains on hardware resets.

When the supply voltage is in the range where operation is guaranteed, a reset is effected by holding the reset pin level "L" (0.2V<sub>CC</sub> max.) for at least 20 cycles. When the reset pin level is then returned to the "H" level while main clock is stable, the reset status is cancelled and program execution resumes from the address in the reset vector table.

Figure 1.6.1 shows the example reset circuit. Figure 1.6.2 shows the reset sequence.

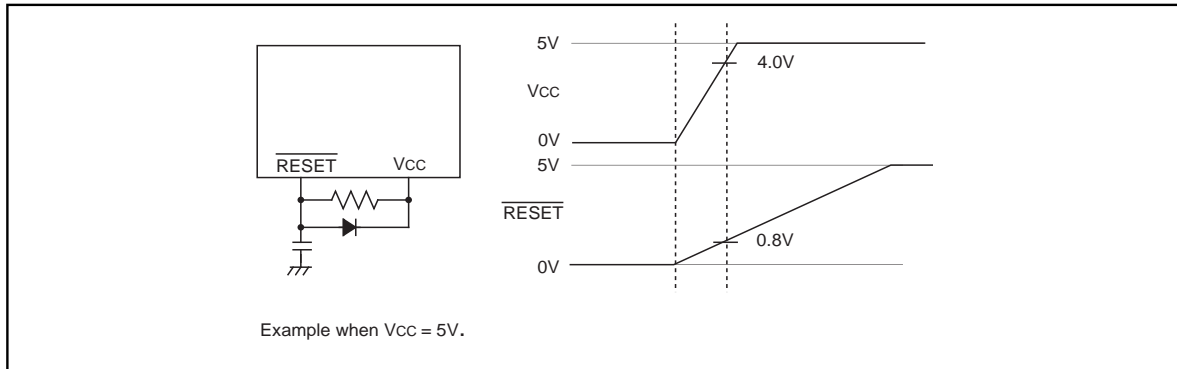


Figure 1.6.1. Example reset circuit

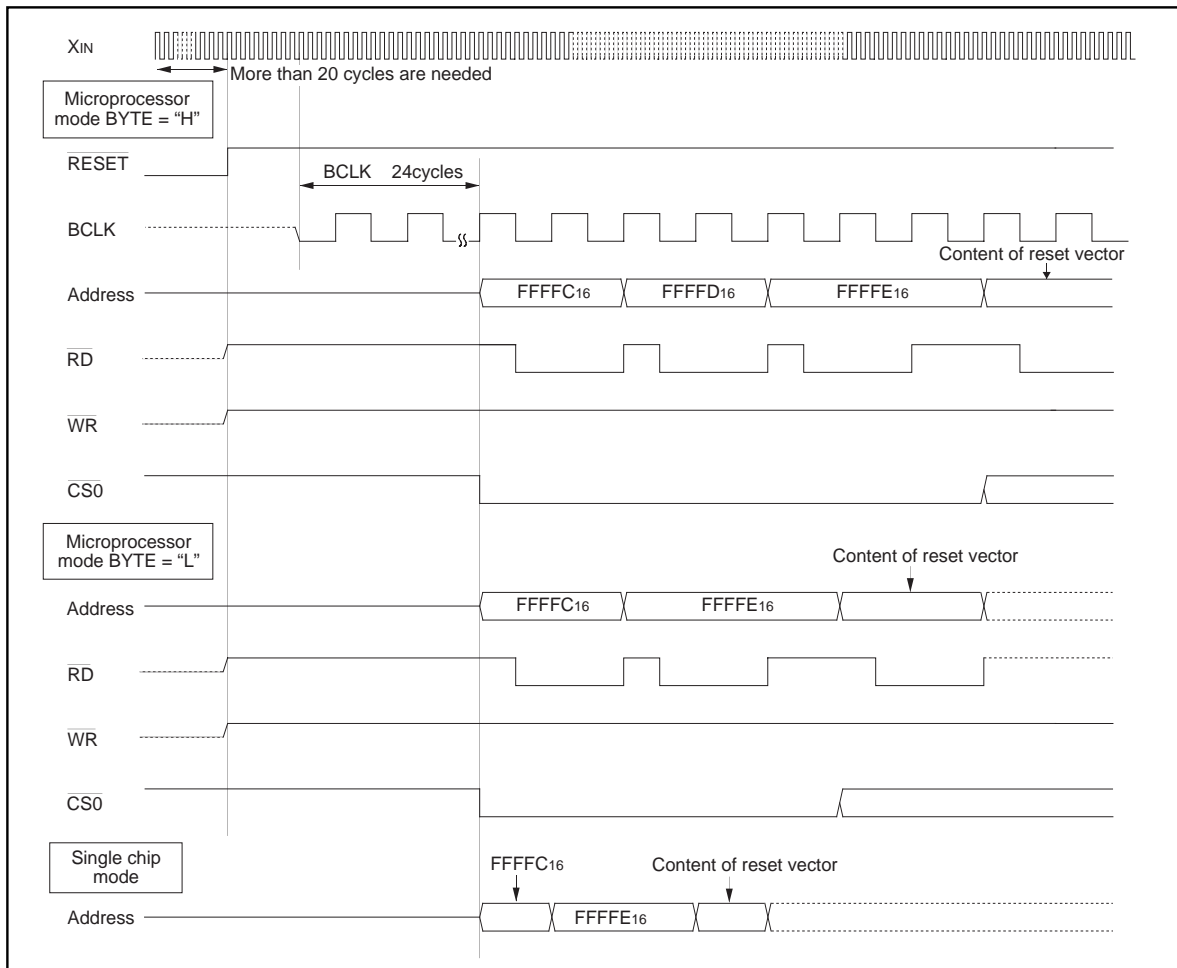


Figure 1.6.2. Reset sequence

Table 1.6.1 shows the statuses of the other pins while the  $\overline{\text{RESET}}$  pin level is "L". Figure 1.6.3 shows the internal status of the microcomputer immediately after the reset is cancelled.

**Table 1.6.1. Pin status when  $\overline{\text{RESET}}$  pin level is "L"**

Pin name	Status		
	CNVss = Vss	CNVss = Vcc	
		BYTE = Vss	BYTE = Vcc
P0	Input port (floating)	Data input (floating)	Data input (floating)
P1	Input port (floating)	Data input (floating)	Input port (floating)
P2, P3, P40 to P43	Input port (floating)	Address output (undefined)	Address output (undefined)
P44	Input port (floating)	$\overline{\text{CS0}}$ output ("H" level is output)	$\overline{\text{CS0}}$ output ("H" level is output)
P45 to P47	Input port (floating)	Input port (floating)	Input port (floating)
P50	Input port (floating)	$\overline{\text{WR}}$ output ("H" level is output)	$\overline{\text{WR}}$ output ("H" level is output)
P51	Input port (floating)	$\overline{\text{BHE}}$ output (undefined)	$\overline{\text{BHE}}$ output (undefined)
P52	Input port (floating)	$\overline{\text{RD}}$ output ("H" level is output)	$\overline{\text{RD}}$ output ("H" level is output)
P53	Input port (floating)	BCLK output	BCLK output
P54	Input port (floating)	$\overline{\text{HLDA}}$ output (The output value depends on the input to the HOLD pin)	$\overline{\text{HLDA}}$ output (The output value depends on the input to the HOLD pin)
P55	Input port (floating)	$\overline{\text{HOLD}}$ input (floating)	$\overline{\text{HOLD}}$ input (floating)
P56	Input port (floating)	ALE output ("L" level is output)	ALE output ("L" level is output)
P57	Input port (floating)	$\overline{\text{RDY}}$ input (floating)	$\overline{\text{RDY}}$ input (floating)
P6, P7, P80 to P84, P86, P87, P9, P10	Input port (floating)	Input port (floating)	Input port (floating)

(1) Processor mode register 0 (Note)	(00041e)...	001e	(43) Timer A0 mode register	(03961e)...	001e
(2) Processor mode register 1	(00051e)...	0 0 x x x x x 0	(44) Timer A1 mode register	(03971e)...	001e
(3) System clock control register 0	(00061e)...	0 1 0 0 1 0 0 0	(45) Timer A2 mode register	(03981e)...	001e
(4) System clock control register 1	(00071e)...	0 0 1 0 0 0 0 0	(46) Timer A3 mode register	(03991e)...	001e
(5) Chip select control register	(00081e)...	0 0 0 0 0 0 0 1	(47) Timer A4 mode register	(039A1e)...	001e
(6) Address match interrupt enable register	(00091e)...	x x x x x x 0 0	(48) Timer B0 mode register	(039B1e)...	0 0 ? x 0 0 0 0
(7) Protect register	(000A1e)...	x x x x x x 0 0 0	(49) Timer B1 mode register	(039C1e)...	0 0 ? x 0 0 0 0
(8) Watchdog timer control register	(000F1e)...	0 0 0 ? ? ? ? ?	(50) Timer B2 mode register	(039D1e)...	0 0 ? x 0 0 0 0
(9) Address match interrupt register 0	(00101e)...	001e	(51) UART0 transmit/receive mode register	(03A01e)...	001e
	(00111e)...	001e	(52) UART0 transmit/receive control register 0	(03A41e)...	0 0 0 0 1 0 0 0
	(00121e)...	x x x x 0 0 0 0	(53) UART0 transmit/receive control register 1	(03A51e)...	0 0 0 0 0 0 1 0
(10) Address match interrupt register 1	(00141e)...	001e	(54) UART1 transmit/receive mode register	(03A81e)...	001e
	(00151e)...	001e	(55) UART1 transmit/receive control register 0	(03AC1e)...	0 0 0 0 1 0 0 0
	(00161e)...	x x x x 0 0 0 0	(56) UART1 transmit/receive control register 1	(03AD1e)...	0 0 0 0 0 0 1 0
(11) DMA0 control register	(002C1e)...	0 0 0 0 0 ? 0 0	(57) UART transmit/receive control register 2	(03B01e)...	x 0 0 0 0 0 0 0
(12) DMA1 control register	(003C1e)...	0 0 0 0 0 ? 0 0	(58) DMA0 cause select register	(03B81e)...	001e
(13) Bus collision detection interrupt control register	(004A1e)...	x x x x ? 0 0 0	(59) DMA1 cause select register	(03BA1e)...	001e
(14) DMA0 interrupt control register	(004B1e)...	x x x x ? 0 0 0	(60) A-D control register 2	(03D41e)...	0 0 0 0 x x x 0
(15) DMA1 interrupt control register	(004C1e)...	x x x x ? 0 0 0	(61) A-D control register 0	(03D61e)...	0 0 0 0 0 ? ? ?
(16) Key input interrupt control register	(004D1e)...	x x x x ? 0 0 0	(62) A-D control register 1	(03D71e)...	001e
(17) A-D conversion interrupt control register	(004E1e)...	x x x x ? 0 0 0	(63) D-A control register	(03DC1e)...	001e
(18) UART2 transmit interrupt control register	(004F1e)...	x x x x ? 0 0 0	(64) Port P0 direction register	(03E21e)...	001e
(19) UART2 receive interrupt control register	(00501e)...	x x x x ? 0 0 0	(65) Port P1 direction register	(03E31e)...	001e
(20) UART0 transmit interrupt control register	(00511e)...	x x x x ? 0 0 0	(66) Port P2 direction register	(03E61e)...	001e
(21) UART0 receive interrupt control register	(00521e)...	x x x x ? 0 0 0	(67) Port P3 direction register	(03E71e)...	001e
(22) UART1 transmit interrupt control register	(00531e)...	x x x x ? 0 0 0	(68) Port P4 direction register	(03EA1e)...	001e
(23) UART1 receive interrupt control register	(00541e)...	x x x x ? 0 0 0	(69) Port P5 direction register	(03EB1e)...	001e
(24) Timer A0 interrupt control register	(00551e)...	x x x x ? 0 0 0	(70) Port P6 direction register	(03EE1e)...	001e
(25) Timer A1 interrupt control register	(00561e)...	x x x x ? 0 0 0	(71) Port P7 direction register	(03EF1e)...	001e
(26) Timer A2 interrupt control register	(00571e)...	x x x x ? 0 0 0	(72) Port P8 direction register	(03F21e)...	0 0 x 0 0 0 0 0
(27) Timer A3 interrupt control register	(00581e)...	x x x x ? 0 0 0	(73) Port P9 direction register	(03F31e)...	001e
(28) Timer A4 interrupt control register	(00591e)...	x x x x ? 0 0 0	(74) Port P10 direction register	(03F61e)...	001e
(29) Timer B0 interrupt control register	(005A1e)...	x x x x ? 0 0 0	(75) Pull-up control register 0	(03FC1e)...	001e
(30) Timer B1 interrupt control register	(005B1e)...	x x x x ? 0 0 0	(76) Pull-up control register 1	(03FD1e)...	001e
(31) Timer B2 interrupt control register	(005C1e)...	x x x x ? 0 0 0	(77) Pull-up control register 2	(03FE1e)...	001e
(32) INT0 interrupt control register	(005D1e)...	x x 0 0 ? 0 0 0	(78) Data registers (R0/R1/R2/R3)		00001e
(33) INT1 interrupt control register	(005E1e)...	x x 0 0 ? 0 0 0	(79) Address registers (A0/A1)		00001e
(34) INT2 interrupt control register	(005F1e)...	x x 0 0 ? 0 0 0	(80) Frame base register (FB)		00001e
(35) UART2 transmit/receive mode register	(03781e)...	001e	(81) Interrupt table register (INTB)		000001e
(36) UART2 transmit/receive control register 0	(037C1e)...	0 0 0 0 1 0 0 0	(82) User stack pointer (USP)		00001e
(37) UART2 transmit/receive control register 1	(037D1e)...	0 0 0 0 0 0 1 0	(83) Interrupt stack pointer (ISP)		00001e
(38) Count start flag	(03801e)...	001e	(84) Static base register (SB)		00001e
(39) Clock prescaler reset flag	(03811e)...	0 x x x x x x x	(85) Flag register (FLG)		00001e
(40) One-shot start flag	(03821e)...	0 0 x 0 0 0 0 0			
(41) Trigger select flag	(03831e)...	001e			
(42) Up-down flag	(03841e)...	001e			

x : Nothing is mapped to this bit  
? : Undefined

The content of other registers and RAM is undefined when the microcomputer is reset. The initial values must therefore be set.

Note: When the VCC level is applied to the CNVSS pin, it is 031e at a reset.

Figure 1.6.3. Device's internal status after a reset is cleared



0000 <sub>16</sub>		0040 <sub>16</sub>	
0001 <sub>16</sub>		0041 <sub>16</sub>	
0002 <sub>16</sub>		0042 <sub>16</sub>	
0003 <sub>16</sub>		0043 <sub>16</sub>	
0004 <sub>16</sub>	Processor mode register 0 (PM0)	0044 <sub>16</sub>	
0005 <sub>16</sub>	Processor mode register 1 (PM1)	0045 <sub>16</sub>	
0006 <sub>16</sub>	System clock control register 0 (CM0)	0046 <sub>16</sub>	
0007 <sub>16</sub>	System clock control register 1 (CM1)	0047 <sub>16</sub>	
0008 <sub>16</sub>	Chip select control register (CSR)	0048 <sub>16</sub>	
0009 <sub>16</sub>	Address match interrupt enable register (AIER)	0049 <sub>16</sub>	
000A <sub>16</sub>	Protect register (PRCR)	004A <sub>16</sub>	Bus collision detection interrupt control register (BCNIC)
000B <sub>16</sub>		004B <sub>16</sub>	DMA0 interrupt control register (DM0IC)
000C <sub>16</sub>		004C <sub>16</sub>	DMA1 interrupt control register (DM1IC)
000D <sub>16</sub>		004D <sub>16</sub>	Key input interrupt control register (KUPIC)
000E <sub>16</sub>	Watchdog timer start register (WDTS)	004E <sub>16</sub>	A-D conversion interrupt control register (ADIC)
000F <sub>16</sub>	Watchdog timer control register (WDC)	004F <sub>16</sub>	UART2 transmit interrupt control register (S2TIC)
0010 <sub>16</sub>		0050 <sub>16</sub>	UART2 receive interrupt control register (S2RIC)
0011 <sub>16</sub>	Address match interrupt register 0 (RMAD0)	0051 <sub>16</sub>	UART0 transmit interrupt control register (S0TIC)
0012 <sub>16</sub>		0052 <sub>16</sub>	UART0 receive interrupt control register (S0RIC)
0013 <sub>16</sub>		0053 <sub>16</sub>	UART1 transmit interrupt control register (S1TIC)
0014 <sub>16</sub>		0054 <sub>16</sub>	UART1 receive interrupt control register (S1RIC)
0015 <sub>16</sub>	Address match interrupt register 1 (RMAD1)	0055 <sub>16</sub>	Timer A0 interrupt control register (TA0IC)
0016 <sub>16</sub>		0056 <sub>16</sub>	Timer A1 interrupt control register (TA1IC)
0017 <sub>16</sub>		0057 <sub>16</sub>	Timer A2 interrupt control register (TA2IC)
0018 <sub>16</sub>		0058 <sub>16</sub>	Timer A3 interrupt control register (TA3IC)
0019 <sub>16</sub>		0059 <sub>16</sub>	Timer A4 interrupt control register (TA4IC)
001A <sub>16</sub>		005A <sub>16</sub>	Timer B0 interrupt control register (TB0IC)
001B <sub>16</sub>		005B <sub>16</sub>	Timer B1 interrupt control register (TB1IC)
001C <sub>16</sub>		005C <sub>16</sub>	Timer B2 interrupt control register (TB2IC)
001D <sub>16</sub>		005D <sub>16</sub>	INT0 interrupt control register (INT0IC)
001E <sub>16</sub>		005E <sub>16</sub>	INT1 interrupt control register (INT1IC)
001F <sub>16</sub>		005F <sub>16</sub>	INT2 interrupt control register (INT2IC)
0020 <sub>16</sub>		≈	≈
0021 <sub>16</sub>	DMA0 source pointer (SAR0)	0363 <sub>16</sub>	
0022 <sub>16</sub>		0364 <sub>16</sub>	
0023 <sub>16</sub>		0365 <sub>16</sub>	
0024 <sub>16</sub>		0366 <sub>16</sub>	
0025 <sub>16</sub>	DMA0 destination pointer (DAR0)	0367 <sub>16</sub>	
0026 <sub>16</sub>		0368 <sub>16</sub>	
0027 <sub>16</sub>		0369 <sub>16</sub>	
0028 <sub>16</sub>	DMA0 transfer counter (TCR0)	036A <sub>16</sub>	
0029 <sub>16</sub>		036B <sub>16</sub>	
002A <sub>16</sub>		036C <sub>16</sub>	
002B <sub>16</sub>		036D <sub>16</sub>	
002C <sub>16</sub>	DMA0 control register (DM0CON)	036E <sub>16</sub>	
002D <sub>16</sub>		036F <sub>16</sub>	
002E <sub>16</sub>		0370 <sub>16</sub>	
002F <sub>16</sub>		0371 <sub>16</sub>	
0030 <sub>16</sub>		0372 <sub>16</sub>	
0031 <sub>16</sub>	DMA1 source pointer (SAR1)	0373 <sub>16</sub>	
0032 <sub>16</sub>		0374 <sub>16</sub>	
0033 <sub>16</sub>		0375 <sub>16</sub>	
0034 <sub>16</sub>		0376 <sub>16</sub>	
0035 <sub>16</sub>	DMA1 destination pointer (DAR1)	0377 <sub>16</sub>	
0036 <sub>16</sub>		0378 <sub>16</sub>	UART2 transmit/receive mode register (U2MR)
0037 <sub>16</sub>		0379 <sub>16</sub>	UART2 bit rate generator (U2BRG)
0038 <sub>16</sub>	DMA1 transfer counter (TCR1)	037A <sub>16</sub>	
0039 <sub>16</sub>		037B <sub>16</sub>	UART2 transmit buffer register (U2TB)
003A <sub>16</sub>		037C <sub>16</sub>	UART2 transmit/receive control register 0 (U2C0)
003B <sub>16</sub>		037D <sub>16</sub>	UART2 transmit/receive control register 1 (U2C1)
003C <sub>16</sub>	DMA1 control register (DM1CON)	037E <sub>16</sub>	
003D <sub>16</sub>		037F <sub>16</sub>	UART2 receive buffer register (U2RB)
003E <sub>16</sub>			
003F <sub>16</sub>			

Figure 1.7.1. Location of peripheral unit control registers

0380 <sub>16</sub>	Count start flag (TABSR)	03C0 <sub>16</sub>	A-D register 0 (AD0)
0381 <sub>16</sub>	Clock prescaler reset flag (CPSRF)	03C1 <sub>16</sub>	
0382 <sub>16</sub>	One-shot start flag (ONSF)	03C2 <sub>16</sub>	A-D register 1 (AD1)
0383 <sub>16</sub>	Trigger select register (TRGSR)	03C3 <sub>16</sub>	
0384 <sub>16</sub>	Up-down flag (UDF)	03C4 <sub>16</sub>	A-D register 2 (AD2)
0385 <sub>16</sub>		03C5 <sub>16</sub>	
0386 <sub>16</sub>		03C6 <sub>16</sub>	A-D register 3 (AD3)
0387 <sub>16</sub>	Timer A0 (TA0)	03C7 <sub>16</sub>	
0388 <sub>16</sub>		03C8 <sub>16</sub>	A-D register 4 (AD4)
0389 <sub>16</sub>	Timer A1 (TA1)	03C9 <sub>16</sub>	
038A <sub>16</sub>		03CA <sub>16</sub>	A-D register 5 (AD5)
038B <sub>16</sub>	Timer A2 (TA2)	03CB <sub>16</sub>	
038C <sub>16</sub>		03CC <sub>16</sub>	A-D register 6 (AD6)
038D <sub>16</sub>	Timer A3 (TA3)	03CD <sub>16</sub>	
038E <sub>16</sub>		03CE <sub>16</sub>	A-D register 7 (AD7)
038F <sub>16</sub>	Timer A4 (TA4)	03CF <sub>16</sub>	
0390 <sub>16</sub>		03D0 <sub>16</sub>	
0391 <sub>16</sub>	Timer B0 (TB0)	03D1 <sub>16</sub>	
0392 <sub>16</sub>		03D2 <sub>16</sub>	
0393 <sub>16</sub>	Timer B1 (TB1)	03D3 <sub>16</sub>	
0394 <sub>16</sub>		03D4 <sub>16</sub>	A-D control register 2 (ADCON2)
0395 <sub>16</sub>	Timer B2 (TB2)	03D5 <sub>16</sub>	
0396 <sub>16</sub>	Timer A0 mode register (TA0MR)	03D6 <sub>16</sub>	A-D control register 0 (ADCON0)
0397 <sub>16</sub>	Timer A1 mode register (TA1MR)	03D7 <sub>16</sub>	A-D control register 1 (ADCON1)
0398 <sub>16</sub>	Timer A2 mode register (TA2MR)	03D8 <sub>16</sub>	D-A register 0 (DA0)
0399 <sub>16</sub>	Timer A3 mode register (TA3MR)	03D9 <sub>16</sub>	
039A <sub>16</sub>	Timer A4 mode register (TA4MR)	03DA <sub>16</sub>	D-A register 1 (DA1)
039B <sub>16</sub>	Timer B0 mode register (TB0MR)	03DB <sub>16</sub>	
039C <sub>16</sub>	Timer B1 mode register (TB1MR)	03DC <sub>16</sub>	D-A control register (DACON)
039D <sub>16</sub>	Timer B2 mode register (TB2MR)	03DD <sub>16</sub>	
039E <sub>16</sub>		03DE <sub>16</sub>	
039F <sub>16</sub>		03DF <sub>16</sub>	
03A0 <sub>16</sub>	UART0 transmit/receive mode register (U0MR)	03E0 <sub>16</sub>	Port P0 (P0)
03A1 <sub>16</sub>	UART0 bit rate generator (U0BRG)	03E1 <sub>16</sub>	Port P1 (P1)
03A2 <sub>16</sub>		03E2 <sub>16</sub>	Port P0 direction register (PD0)
03A3 <sub>16</sub>	UART0 transmit buffer register (U0TB)	03E3 <sub>16</sub>	Port P1 direction register (PD1)
03A4 <sub>16</sub>	UART0 transmit/receive control register 0 (U0C0)	03E4 <sub>16</sub>	Port P2 (P2)
03A5 <sub>16</sub>	UART0 transmit/receive control register 1 (U0C1)	03E5 <sub>16</sub>	Port P3 (P3)
03A6 <sub>16</sub>	UART0 receive buffer register (U0RB)	03E6 <sub>16</sub>	Port P2 direction register (PD2)
03A7 <sub>16</sub>		03E7 <sub>16</sub>	Port P3 direction register (PD3)
03A8 <sub>16</sub>	UART1 transmit/receive mode register (U1MR)	03E8 <sub>16</sub>	Port P4 (P4)
03A9 <sub>16</sub>	UART1 bit rate generator (U1BRG)	03E9 <sub>16</sub>	Port P5 (P5)
03AA <sub>16</sub>		03EA <sub>16</sub>	Port P4 direction register (PD4)
03AB <sub>16</sub>	UART1 transmit buffer register (U1TB)	03EB <sub>16</sub>	Port P5 direction register (PD5)
03AC <sub>16</sub>		03EC <sub>16</sub>	Port P6 (P6)
03AD <sub>16</sub>	UART1 transmit/receive control register 0 (U1C0)	03ED <sub>16</sub>	Port P7 (P7)
03AE <sub>16</sub>	UART1 transmit/receive control register 1 (U1C1)	03EE <sub>16</sub>	Port P6 direction register (PD6)
03AF <sub>16</sub>	UART1 receive buffer register (U1RB)	03EF <sub>16</sub>	Port P7 direction register (PD7)
03B0 <sub>16</sub>	UART transmit/receive control register 2 (UCON)	03F0 <sub>16</sub>	Port P8 (P8)
03B1 <sub>16</sub>		03F1 <sub>16</sub>	Port P9 (P9)
03B2 <sub>16</sub>		03F2 <sub>16</sub>	Port P8 direction register (PD8)
03B3 <sub>16</sub>		03F3 <sub>16</sub>	Port P9 direction register (PD9)
03B4 <sub>16</sub>		03F4 <sub>16</sub>	Port P10 (P10)
03B5 <sub>16</sub>		03F5 <sub>16</sub>	
03B6 <sub>16</sub>		03F6 <sub>16</sub>	Port P10 direction register (PD10)
03B7 <sub>16</sub>		03F7 <sub>16</sub>	
03B8 <sub>16</sub>	DMA0 cause select register (DM0SL)	03F8 <sub>16</sub>	
03B9 <sub>16</sub>		03F9 <sub>16</sub>	
03BA <sub>16</sub>	DMA1 cause select register (DM1SL)	03FA <sub>16</sub>	
03BB <sub>16</sub>		03FB <sub>16</sub>	
03BC <sub>16</sub>		03FC <sub>16</sub>	Pull-up control register 0 (PUR0)
03BD <sub>16</sub>	CRC data register (CRCD)	03FD <sub>16</sub>	Pull-up control register 1 (PUR1)
03BE <sub>16</sub>	CRC input register (CRCIN)	03FE <sub>16</sub>	Pull-up control register 2 (PUR2)
03BF <sub>16</sub>		03FF <sub>16</sub>	

Figure 1.7.2. Location of peripheral unit control registers

## Software Reset

Writing “1” to bit 3 of the processor mode register 0 (address 000416) applies a (software) reset to the microcomputer. A software reset has almost the same effect as a hardware reset. The contents of internal RAM are preserved.

## Processor Mode

### (1) Types of Processor Mode

One of three processor modes can be selected: single-chip mode, memory expansion mode, and microprocessor mode. The functions of some pins, the memory map, and the access space differ according to the selected processor mode.

- **Single-chip mode**

In single-chip mode, only internal memory space (SFR, internal RAM, and internal ROM) can be accessed. Ports P0 to P10 can be used as programmable I/O ports or as I/O ports for the internal peripheral functions.

- **Memory expansion mode**

In memory expansion mode, external memory can be accessed in addition to the internal memory space (SFR, internal RAM, and internal ROM).

In this mode, some of the pins function as the address bus, the data bus, and as control signals. The number of pins assigned to these functions depends on the bus and register settings. (See “Bus Settings” for details.)

- **Microprocessor mode**

In microprocessor mode, the SFR, internal RAM, and external memory space can be accessed. The internal ROM area cannot be accessed.

In this mode, some of the pins function as the address bus, the data bus, and as control signals. The number of pins assigned to these functions depends on the bus and register settings. (See “Bus Settings” for details.)

### (2) Setting Processor Modes

The processor mode is set using the CNVss pin and the processor mode bits (bits 1 and 0 at address 000416). Do not set the processor mode bits to “102”.

Regardless of the level of the CNVss pin, changing the processor mode bits selects the mode. Therefore, never change the processor mode bits when changing the contents of other bits. Also do not attempt to shift to or from the microprocessor mode within the program stored in the internal ROM area.

- **Applying Vss to CNVss pin**

The microcomputer begins operation in single-chip mode after being reset. Memory expansion mode is selected by writing “012” to the processor mode is selected bits.

- **Applying Vcc to CNVss pin**

The microcomputer starts to operate in microprocessor mode after being reset.

Figure 1.8.1 shows the processor mode register 0 and 1. Figure 1.9.1 shows the memory maps applicable for each of the modes.

## Processor Mode

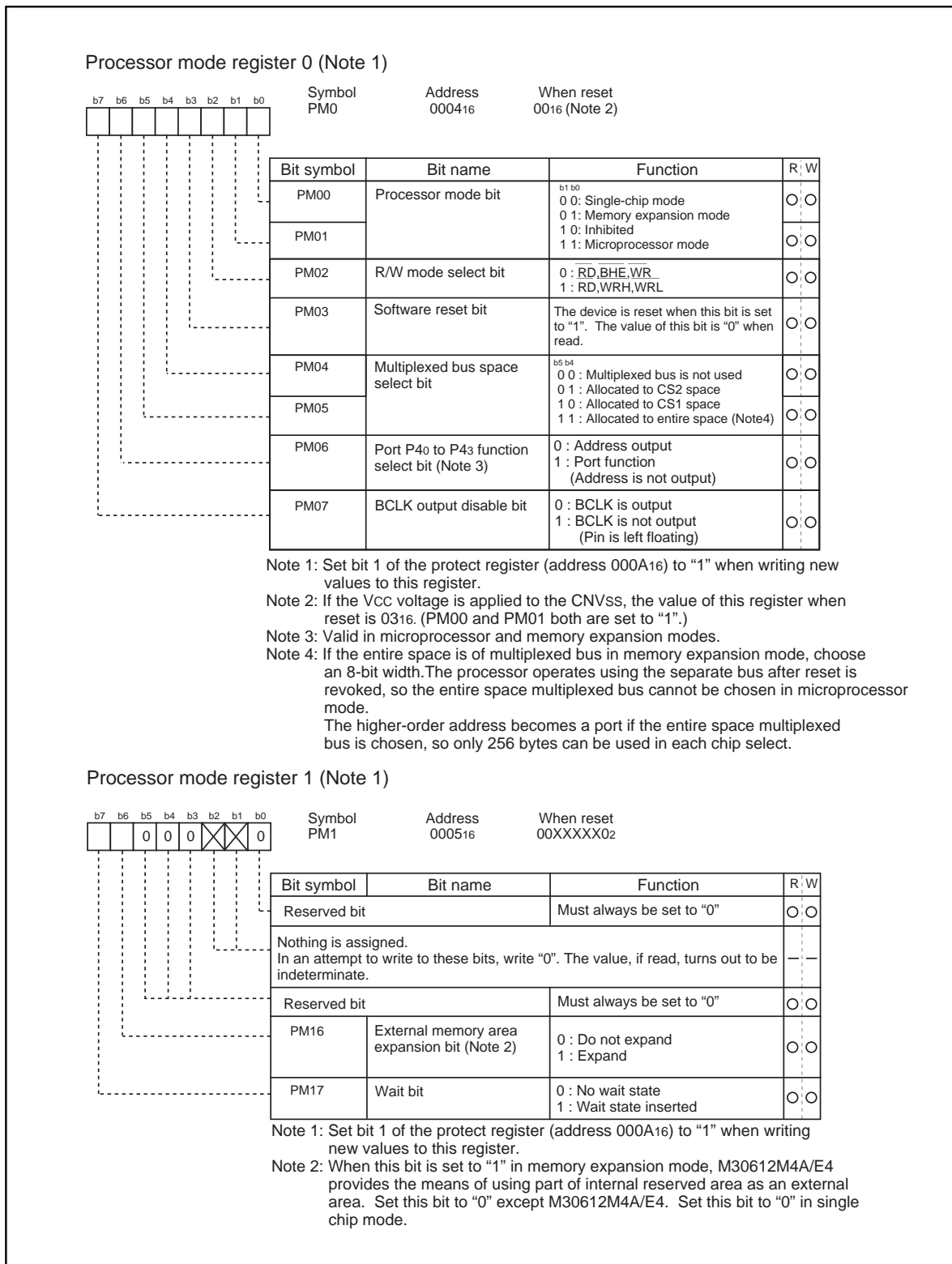


Figure 1.8.1. Processor mode register 0 and 1

## Processor Mode

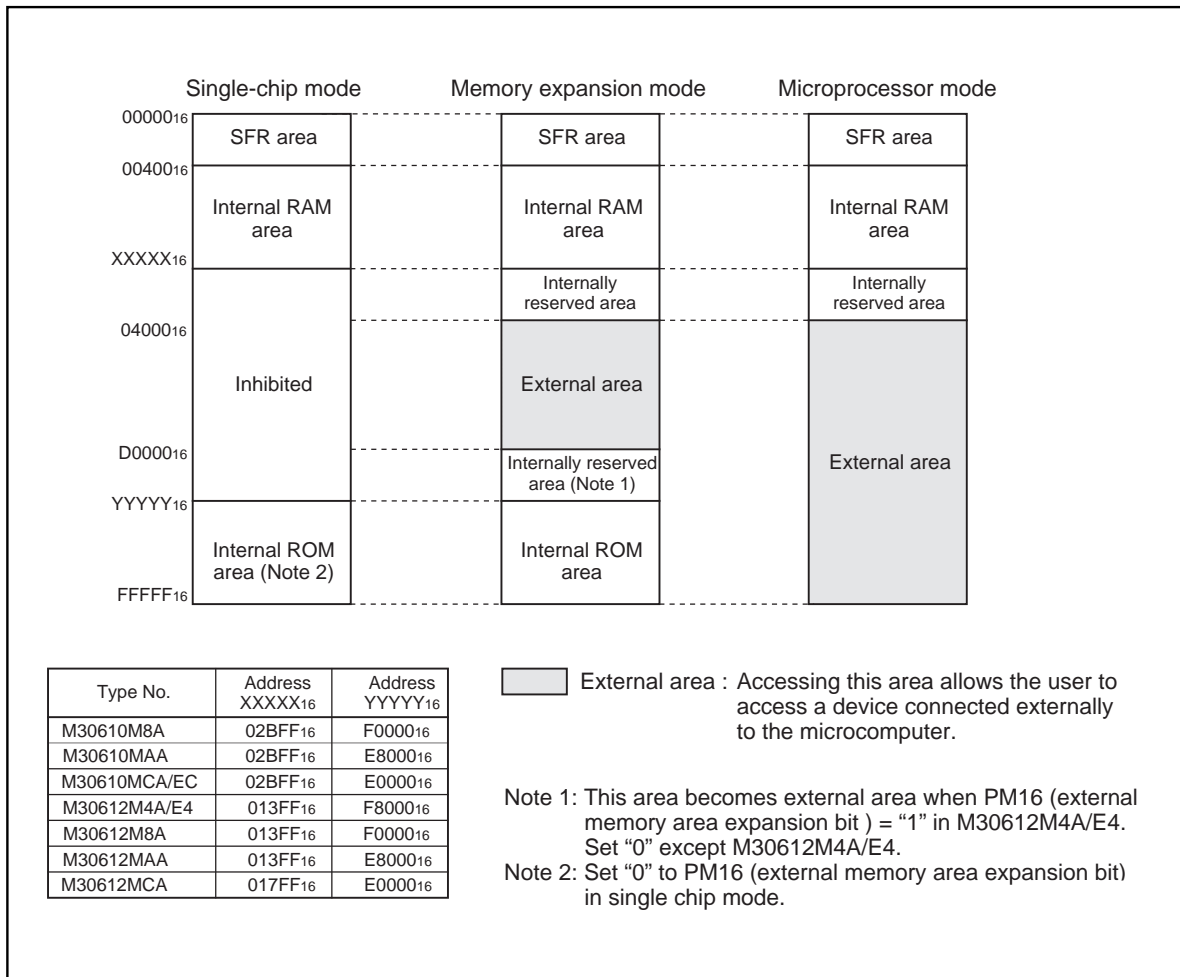


Figure 1.9.1. Memory maps in each processor mode

## Bus Settings

The BYTE pin and bits 4 to 6 of the processor mode register 0 (address 000416) are used to change the bus settings.

Table 1.10.1 shows the factors used to change the bus settings.

**Table 1.10.1. Factors for switching bus settings**

Bus setting	Switching factor
Switching external address bus width	Bit 6 of processor mode register 0
Switching external data bus width	BYTE pin
Switching between separate and multiplex bus	Bits 4 and 5 of processor mode register 0

### (1) Selecting external address bus width

The address bus width for external output in the 1M bytes of address space can be set to 16 bits (64K bytes address space) or 20 bits (1M bytes address space). When bit 6 of the processor mode register 0 is set to "1", the external address bus width is set to 16 bits, and P2 and P3 become part of the address bus. P40 to P43 can be used as programmable I/O ports. When bit 6 of processor mode register 0 is set to "0", the external address bus width is set to 20 bits, and P2, P3, and P40 to P43 become part of the address bus.

### (2) Selecting external data bus width

The external data bus width can be set to 8 or 16 bits. (Note, however, that only the separate bus can be set.) When the BYTE pin is "L", the bus width is set to 16 bits; when "H", it is set to 8 bits. (The internal bus width is permanently set to 16 bits.) While operating, fix the BYTE pin either to "H" or to "L".

### (3) Selecting separate/multiplex bus

The bus format can be set to multiplex or separate bus using bits 4 and 5 of the processor mode register 0.

#### • Separate bus

In this mode, the data and address are input and output separately. The data bus can be set using the BYTE pin to be 8 or 16 bits. When the BYTE pin is "H", the data bus is set to 8 bits and P0 functions as the data bus and P1 as a programmable I/O port. When the BYTE pin is "L", the data bus is set to 16 bits and P0 and P1 are both used for the data bus.

When the separate bus is used for access, a software wait can be selected.

#### • Multiplex bus

In this mode, data and address I/O are time multiplexed. With an 8-bit data bus selected (BYTE pin = "H"), the 8 bits from D0 to D7 are multiplexed with A0 to A7.

With a 16-bit data bus selected (BYTE pin = "L"), the 8 bits from D0 to D7 are multiplexed with A1 to A8. D8 to D15 are not multiplexed. In this case, the external devices connected to the multiplexed bus are mapped to the microcomputer's even addresses (every 2nd address). To access these external devices, access the even addresses as bytes.

The ALE signal latches the address. It is output from P56.

Before using the multiplex bus for access, be sure to insert a software wait.

If the entire space is of multiplexed bus in memory expansion mode, choose an 8-bit width.

The processor operates using the separate bus after reset is revoked, so the entire space multiplexed bus cannot be chosen in microprocessor mode.

The higher-order address becomes a port if the entire space multiplexed bus is chosen, so only 256 bytes can be used in each chip select.

## Bus Settings

Table 1.10.2. Pin functions for each processor mode

Processor mode	Single-chip mode	Memory expansion mode/microprocessor modes				Memory expansion mode
Multiplexed bus space select bit		"01", "10" [ Either CS1 or CS2 is for multiplexed bus and others are for separate bus ]		"00" (separate bus)		"11" (Note 1) [ multiplexed bus for the entire space ]
Data bus width BYTE pin level		8 bits "H"	16 bits "L"	8 bits "H"	16 bits "L"	8 bit "H"
P00 to P07	I/O port	Data bus	Data bus	Data bus	Data bus	I/O port
P10 to P17	I/O port	I/O port	Data bus	I/O port	Data bus	I/O port
P20	I/O port	Address bus /data bus(Note 2)	Address bus	Address bus	Address bus	Address bus /data bus
P21 to P27	I/O port	Address bus /data bus(Note 2)	Address bus /data bus(Note 2)	Address bus	Address bus	Address bus /data bus
P30	I/O port	Address bus /data bus(Note 2)	Address bus	Address bus	Address bus	A8/D7
P31 to P37	I/O port	Address bus	Address bus	Address bus	Address bus	I/O port
P40 to P43 Port P40 to P43 function select bit = 1	I/O port	I/O port	I/O port	I/O port	I/O port	I/O port
P40 to P43 Port P40 to P43 function select bit = 0	I/O port	Address bus	Address bus	Address bus	Address bus	I/O port
P44 to P47	I/O port	$\overline{CS}$ (chip select) or programmable I/O port (For details, refer to "Bus control")				
P50 to P53	I/O port	Outputs $\overline{RD}$ , $\overline{WRL}$ , $\overline{WRH}$ , and BCLK or $\overline{RD}$ , $\overline{BHE}$ , $\overline{WR}$ , and BCLK (For details, refer to "Bus control")				
P54	I/O port	$\overline{HLDA}$	$\overline{HLDA}$	$\overline{HLDA}$	$\overline{HLDA}$	$\overline{HLDA}$
P55	I/O port	$\overline{HOLD}$	$\overline{HOLD}$	$\overline{HOLD}$	$\overline{HOLD}$	$\overline{HOLD}$
P56	I/O port	ALE	ALE	ALE	ALE	ALE
P57	I/O port	$\overline{RDY}$	$\overline{RDY}$	$\overline{RDY}$	$\overline{RDY}$	$\overline{RDY}$

Note 1: If the entire space is of multiplexed bus in memory expansion mode, choose an 8-bit width.

The processor operates using the separate bus after reset is revoked, so the entire space multiplexed bus cannot be chosen in microprocessor mode.

The higher-order address becomes a port if the entire space multiplexed bus is chosen, so only 256 bytes can be used in each chip select.

Note 2: Address bus when in separate bus mode.

## Bus Control

The following explains the signals required for accessing external devices and software waits. The signals required for accessing the external devices are valid when the processor mode is set to memory expansion mode and microprocessor mode. The software waits are valid in all processor modes.

### (1) Address bus/data bus

The address bus consists of the 20 pins A0 to A19 for accessing the 1M bytes of address space.

The data bus consists of the pins for data I/O. When the BYTE pin is "H", the 8 ports D0 to D7 function as the data bus. When BYTE is "L", the 16 ports D0 to D15 function as the data bus.

Both the address and data bus retain their previous states when internal ROM or RAM is accessed. Also, when a change is made from single-chip mode to memory expansion mode, the value of the address bus is undefined until external memory is accessed.

### (2) Chip select signal

The chip select signal is output using the same pins as P44 to P47. Bits 0 to 3 of the chip select control register (address 0008<sub>16</sub>) set each pin to function as a port or to output the chip select signal. The chip select control register is valid in memory expansion mode and microprocessor mode. In single-chip mode, P44 to P47 function as programmable I/O ports regardless of the value in the chip select control register.

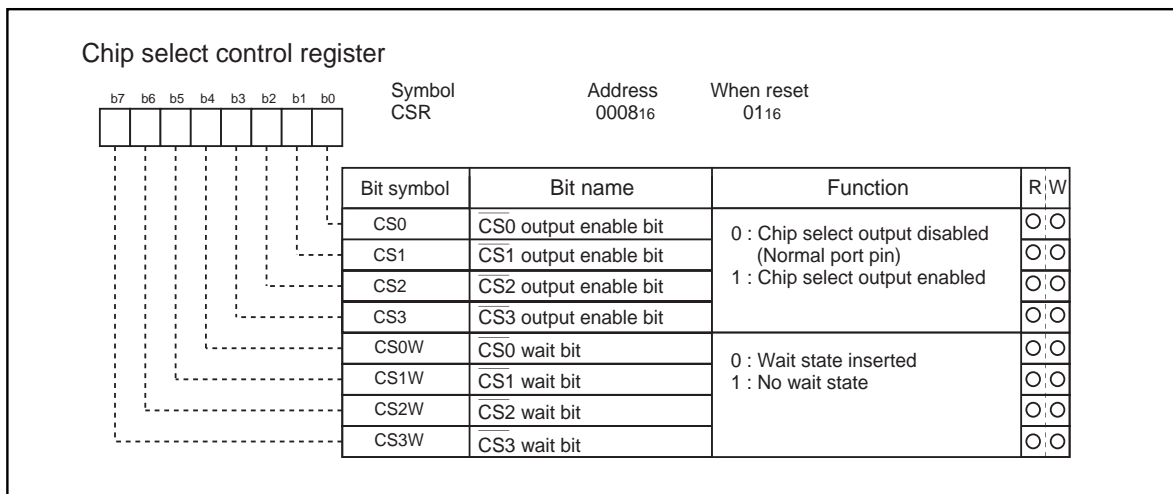
In microprocessor mode, only  $\overline{CS0}$  outputs the chip select signal after the reset state has been cancelled.  $\overline{CS1}$  to  $\overline{CS3}$  function as input ports. Therefore, when using  $\overline{CS1}$  to  $\overline{CS3}$ , external pull-up resistors are required. Figure 1.11.1 shows the chip select control register.

The chip select signal can be used to split the external area into as many as four blocks. Table 1.11.1 shows the external memory areas specified using the chip select signal.

**Table 1.11.1. External areas specified by the chip select signals**

Chip select	Specified address range	
	Memory expansion mode	Microprocessor mode
$\overline{CS0}$	30000 <sub>16</sub> to CFFFF <sub>16</sub> (640K) 30000 <sub>16</sub> to F7FFF <sub>16</sub> (800K) (Note)	30000 <sub>16</sub> to FFFFF <sub>16</sub> (832K)
$\overline{CS1}$	28000 <sub>16</sub> to 2FFFF <sub>16</sub> (32K)	28000 <sub>16</sub> to 2FFFF <sub>16</sub> (32K)
$\overline{CS2}$	08000 <sub>16</sub> to 27FFF <sub>16</sub> (128K)	08000 <sub>16</sub> to 27FFF <sub>16</sub> (128K)
$\overline{CS3}$	04000 <sub>16</sub> to 07FFF <sub>16</sub> (16K)	04000 <sub>16</sub> to 07FFF <sub>16</sub> (16K)

Note: When PM16 (External memory area expansion bit) = "1". (Only M30612M4A/E4 is valid.)



**Figure 1.11.1. Chip select control register**



**(3) Read/write signals**

With a 16-bit data bus (BYTE pin = "L"), bit 2 of the processor mode register 0 (address 000416) select the combinations of RD, BHE, and WR signals or RD, WRL, and WRH signals. With an 8-bit data bus (BYTE pin = "H"), use the combination of RD, WR, and BHE signals. (Set bit 2 of the processor mode register 0 (address 000416) to "0".) Tables 1.11.2 and 1.11.3 show the operation of these signals.

After a reset has been cancelled, the combination of RD, WR, and BHE signals is automatically selected. When switching to the RD, WRL, and WRH combination, do not write to external memory until bit 2 of the processor mode register 0 (address 000416) has been set (Note).

Note: Before attempting to change the contents of the processor mode register 0, set bit 1 of the protect register (address 000A16) to "1".

**Table 1.11.2. Operation of RD, WRL, and WRH signals**

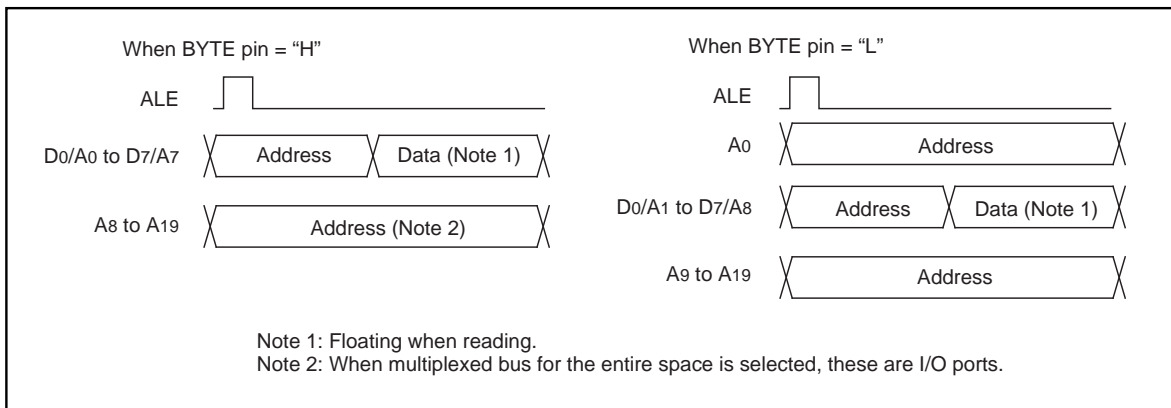
Data bus width	RD	WRL	WRH	Status of external data bus
16-bit (BYTE = "L")	L	H	H	Read data
	H	L	H	Write 1 byte of data to even address
	H	H	L	Write 1 byte of data to odd address
	H	L	L	Write data to both even and odd addresses

**Table 1.11.3. Operation of RD, WR, and BHE signals**

Data bus width	RD	WR	BHE	A0	Status of external data bus
16-bit (BYTE = "L")	H	L	L	H	Write 1 byte of data to odd address
	L	H	L	H	Read 1 byte of data from odd address
	H	L	H	L	Write 1 byte of data to even address
	L	H	H	L	Read 1 byte of data from even address
	H	L	L	L	Write data to both even and odd addresses
	L	H	L	L	Read data from both even and odd addresses
8-bit (BYTE = "H")	H	L	Not used	H / L	Write 1 byte of data
	L	H	Not used	H / L	Read 1 byte of data

**(4) ALE signal**

The ALE signal latches the address when accessing the multiplex bus space. Latch the address when the ALE signal falls.



**Figure 1.11.2. ALE signal and address/data bus**

**(5) The  $\overline{RDY}$  signal**

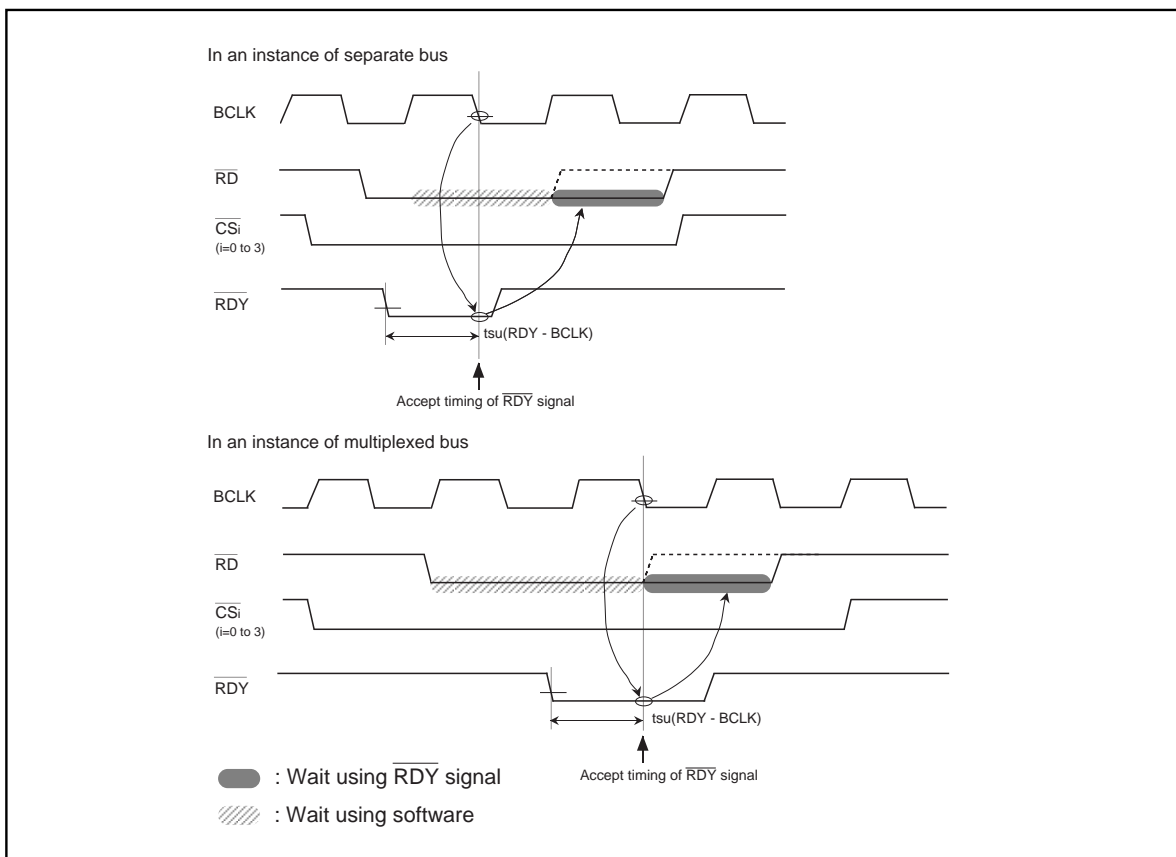
$\overline{RDY}$  is a signal that facilitates access to an external device that requires long access time. As shown in Figure 1.11.3, if an “L” is being input to the  $\overline{RDY}$  at the BCLK falling edge, the bus turns to the wait state. If an “H” is being input to the  $\overline{RDY}$  pin at the BCLK falling edge, the bus cancels the wait state. Table 1.11.4 shows the state of the microcomputer with the bus in the wait state, and Figure 1.11.3 shows an example in which the  $\overline{RD}$  signal is prolonged by the  $\overline{RDY}$  signal.

The  $\overline{RDY}$  signal is valid when accessing the external area during the bus cycle in which bits 4 to 7 of the chip select control register (address 000816) are set to “0”. The  $\overline{RDY}$  signal is invalid when setting “1” to all bits 4 to 7 of the chip select control register (address 000816), but the  $\overline{RDY}$  pin should be treated as properly as in non-using.

**Table 1.11.4. Microcomputer status in ready state (Note)**

Item	Status
Oscillation	On
R/W signal, address bus, data bus, $\overline{CS}$	Maintain status when $\overline{RDY}$ signal received
ALE signal, $\overline{HLDA}$ , programmable I/O ports	
Internal peripheral circuits	On

Note: The  $\overline{RDY}$  signal cannot be received immediately prior to a software wait.

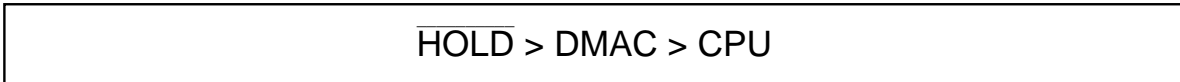


**Figure 1.11.3. Example of  $\overline{RD}$  signal extended by  $\overline{RDY}$  signal**

**(6) Hold signal**

The hold signal is used to transfer the bus privileges from the CPU to the external circuits. Inputting “L” to the  $\overline{\text{HOLD}}$  pin places the microcomputer in the hold state at the end of the current bus access. This status is maintained and “L” is output from the  $\overline{\text{HLDA}}$  pin as long as “L” is input to the  $\overline{\text{HOLD}}$  pin. Table 1.11.5 shows the microcomputer status in the hold state.

Bus-using priorities are given to  $\overline{\text{HOLD}}$ , DMAC, and CPU in order of decreasing precedence.



**Figure 1.11.4. Bus-using priorities**

**Table 1.11.5. Microcomputer status in hold state**

Item		Status
Oscillation		ON
R/ $\overline{\text{W}}$ signal, address bus, data bus, CS, BHE		Floating
Programmable I/O ports	P0, P1, P2, P3, P4, P5	Floating
	P6, P7, P8, P9, P10	Maintains status when hold signal is received
$\overline{\text{HLDA}}$		Output “L”
Internal peripheral circuits		ON (but watchdog timer stops)
ALE signal		Undefined

**(7) External bus status when the internal area is accessed**

Table 1.11.6 shows the external bus status when the internal area is accessed.

**Table 1.11.6. External bus status when the internal area is accessed**

Item		SFR accessed	Internal ROM/RAM accessed
Address bus		Address output	Maintain status before accessed address of external area
Data bus	When read	Floating	Floating
	When write	Output data	Undefined
$\overline{\text{RD}}, \overline{\text{WR}}, \overline{\text{WRL}}, \overline{\text{WRH}}$		$\overline{\text{RD}}, \overline{\text{WR}}, \overline{\text{WRL}}, \overline{\text{WRH}}$ output	Output “H”
$\overline{\text{BHE}}$		$\overline{\text{BHE}}$ output	Maintain status before accessed status of external area
$\overline{\text{CS}}$		Output “H”	Output “H”
ALE		Output “L”	Output “L”

**(8) BCLK output**

The user can choose the BCLK output by use of bit 7 of processor mode register 0 (000416) (Note). When set to “1”, the output floating.

Note: Before attempting to change the contents of the processor mode register 0, set bit 1 of the protect register (address 000A16) to “1”.

**(9) Software wait**

A software wait can be inserted by setting the wait bit (bit 7) of the processor mode register 1 (address 000516) (Note) and bits 4 to 7 of the chip select control register (address 000816).

A software wait is inserted in the internal ROM/RAM area and in the external memory area by setting the wait bit of the processor mode register 1. When set to “0”, each bus cycle is executed in one BCLK cycle. When set to “1”, each bus cycle is executed in two or three BCLK cycles. After the microcomputer has been reset, this bit defaults to “0”. When set to “1”, a wait is applied to all memory areas (two or three BCLK cycles), regardless of the contents of bits 4 to 7 of the chip select control register. Set this bit after referring to the recommended operating conditions (main clock input oscillation frequency) of the electric characteristics. However, when the user is using the  $\overline{\text{RDY}}$  signal, the relevant bit in the chip select control register’s bits 4 to 7 must be set to “0”.

When the wait bit of the processor mode register 1 is “0”, software waits can be set independently for each of the 4 areas selected using the chip select signal. Bits 4 to 7 of the chip select control register correspond to chip selects CS0 to CS3. When one of these bits is set to “1”, the bus cycle is executed in one BCLK cycle. When set to “0”, the bus cycle is executed in two or three BCLK cycles. These bits default to “0” after the microcomputer has been reset.

The SFR area is always accessed in two BCLK cycles regardless of the setting of these control bits. Also, insert a software wait if using the multiplex bus to access the external memory area.

Table 1.11.7 shows the software wait and bus cycles. Figure 1.11.5 shows example bus timing when using software waits.

Note: Before attempting to change the contents of the processor mode register 1, set bit 1 of the protect register (address 000A16) to “1”.

**Table 1.11.7. Software waits and bus cycles**

Area	Bus status	Wait bit	Bits 4 to 7 of chip select control register	Bus cycle
SFR	———	Invalid	Invalid	2 BCLK cycles
Internal ROM/RAM	———	0	Invalid	1 BCLK cycle
	———	1	Invalid	2 BCLK cycles
External memory area	Separate bus	0	1	1 BCLK cycle
	Separate bus	0	0	2 BCLK cycles
	Separate bus	1	0 (Note)	2 BCLK cycles
	Multiplex bus	0	0	3 BCLK cycles
	Multiplex bus	1	0 (Note)	3 BCLK cycles

Note: When using the  $\overline{\text{RDY}}$  signal, always set to “0”.

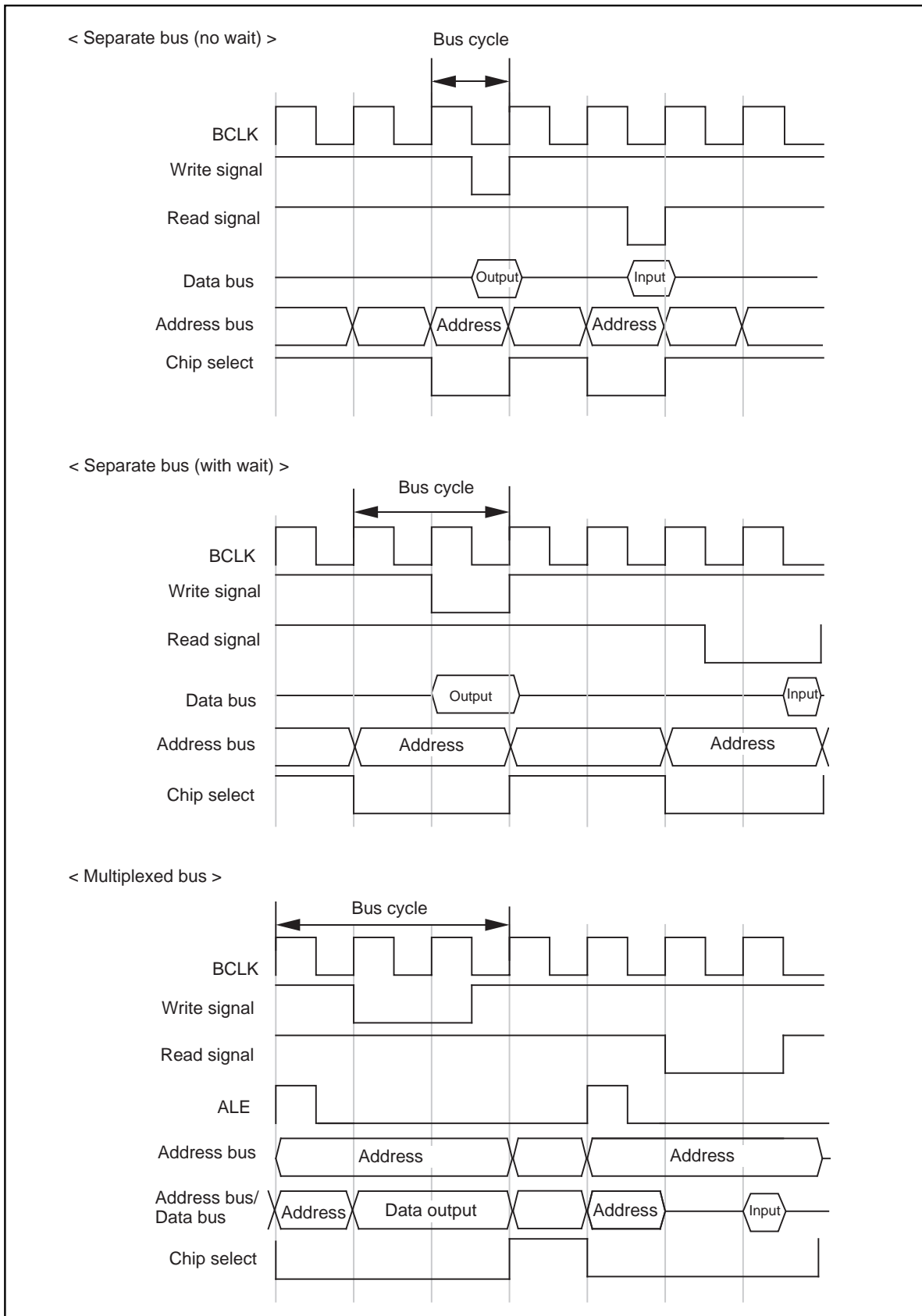


Figure 1.11.5. Typical bus timings using software wait

## Clock Generating Circuit

### Clock Generating Circuit

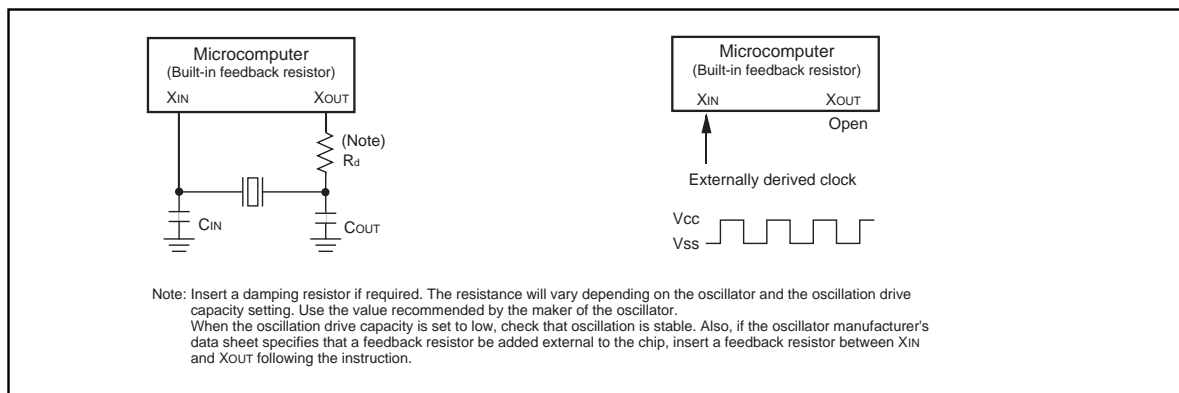
The clock generating circuit contains two oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units.

**Table 1.12.1. Main clock and sub-clock generating circuits**

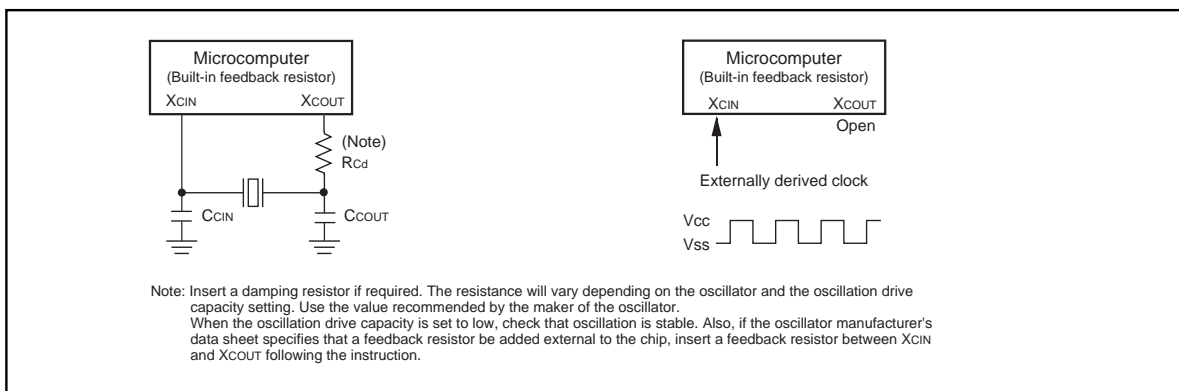
	Main clock generating circuit	Sub-clock generating circuit
Use of clock	<ul style="list-style-type: none"> <li>• CPU's operating clock source</li> <li>• Internal peripheral units' operating clock source</li> </ul>	<ul style="list-style-type: none"> <li>• CPU's operating clock source</li> <li>• Timer A/B's count clock source</li> </ul>
Usable oscillator	Ceramic or crystal oscillator	Crystal oscillator
Pins to connect oscillator	XIN, XOUT	XCIN, XCOUT
Oscillation stop/restart function	Available	Available
Oscillator status immediately after reset	Oscillating	Stopped
Other	Externally derived clock can be input	

### Example of oscillator circuit

Figure 1.12.1 shows some examples of the main clock circuit, one using an oscillator connected to the circuit, and the other one using an externally derived clock for input. Figure 1.12.2 shows some examples of sub-clock circuits, one using an oscillator connected to the circuit, and the other one using an externally derived clock for input. Circuit constants in Figures 1.12.1 and 1.12.2 vary with each oscillator used. Use the values recommended by the manufacturer of your oscillator.



**Figure 1.12.1. Examples of main clock**



**Figure 1.12.2. Examples of sub-clock**

# Clock Generating Circuit

## Clock Control

Figure 1.12.3 shows the block diagram of the clock generating circuit.

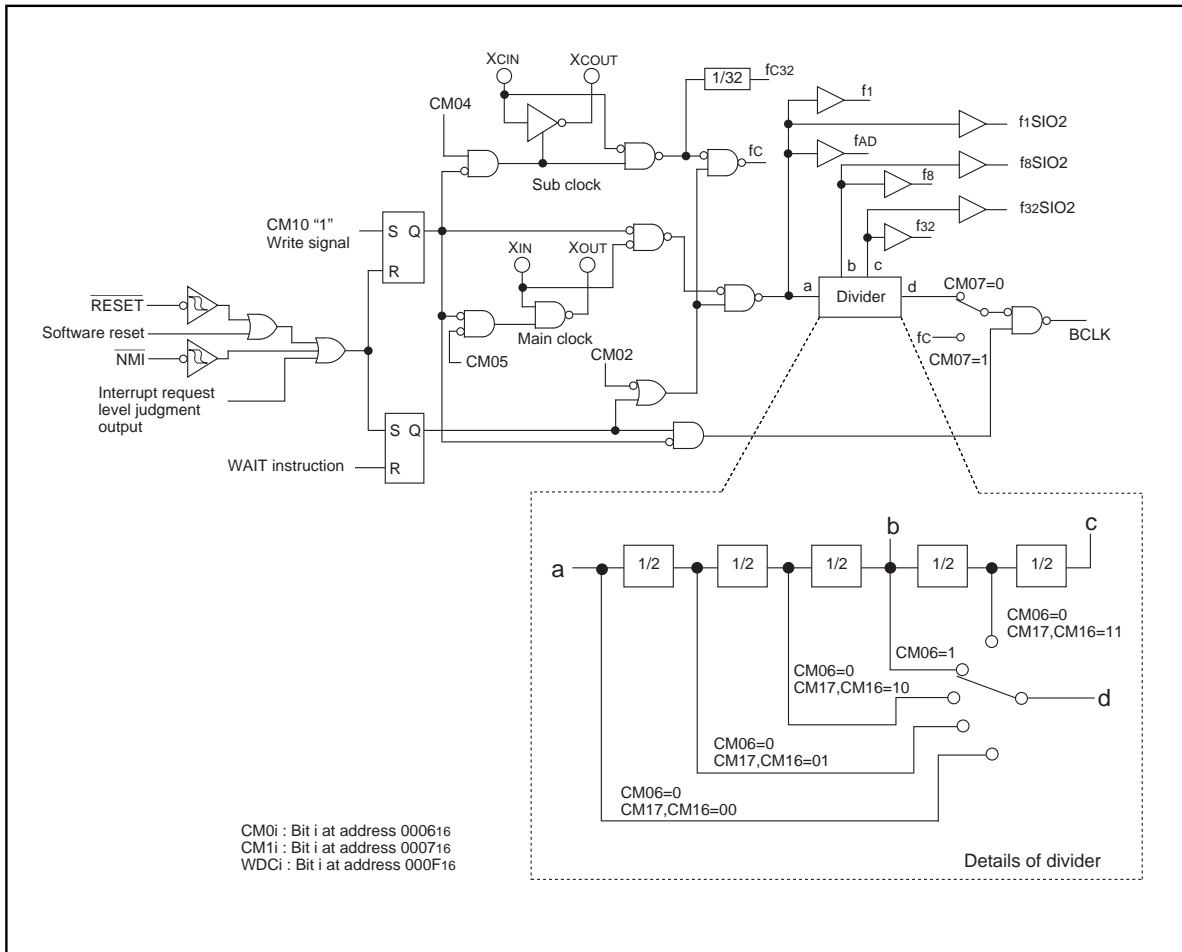


Figure 1.12.3. Clock generating circuit

The following paragraphs describes the clocks generated by the clock generating circuit.

### (1) Main clock

The main clock is generated by the main clock oscillation circuit. After a reset, the clock is divided by 8 to the BCLK. The clock can be stopped using the main clock stop bit (bit 5 at address 0006<sub>16</sub>). Stopping the clock, after switching the operating clock source of CPU to the sub-clock, reduces the power dissipation. After the oscillation of the main clock oscillation circuit has stabilized, the drive capacity of the main clock oscillation circuit can be reduced using the XIN-XOUT drive capacity select bit (bit 5 at address 0007<sub>16</sub>). Reducing the drive capacity of the main clock oscillation circuit reduces the power dissipation. This bit changes to "1" when shifting from high-speed/medium-speed mode to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

### (2) Sub-clock

The sub-clock is generated by the sub-clock oscillation circuit. No sub-clock is generated after a reset. After oscillation is started using the port Xc select bit (bit 4 at address 0006<sub>16</sub>), the sub-clock can be selected as the BCLK by using the system clock select bit (bit 7 at address 0006<sub>16</sub>). However, be sure that the sub-clock oscillation has fully stabilized before switching.

After the oscillation of the sub-clock oscillation circuit has stabilized, the drive capacity of the sub-clock oscillation circuit can be reduced using the XCIN-XCOUT drive capacity select bit (bit 3 at address 0006<sub>16</sub>). Reducing the drive capacity of the sub-clock oscillation circuit reduces the power dissipation. This bit changes to "1" when shifting to stop mode and at a reset.

### (3) BCLK

The BCLK is the clock that drives the CPU, and is fc or the clock is derived by dividing the main clock by 1, 2, 4, 8, or 16. The BCLK is derived by dividing the main clock by 8 after a reset. The BCLK signal can be output from BCLK pin by the BCLK output disable bit (bit 7 at address 0004<sub>16</sub>) in the memory expansion and the microprocessor modes.

The main clock division select bit 0 (bit 6 at address 0006<sub>16</sub>) changes to "1" when shifting from high-speed/medium-speed to stop mode and at reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

### (4) Peripheral function clock(f<sub>1</sub>, f<sub>8</sub>, f<sub>32</sub>, f<sub>1SIO2</sub>, f<sub>8SIO2</sub>, f<sub>32SIO2</sub>, f<sub>AD</sub>)

The clock for the peripheral devices is derived from the main clock or by dividing it by 1, 8, or 32. The peripheral function clock is stopped by stopping the main clock or by setting the WAIT peripheral function clock stop bit (bit 2 at 0006<sub>16</sub>) to "1" and then executing a WAIT instruction.

### (5) fc<sub>32</sub>

This clock is derived by dividing the sub-clock by 32. It is used for the timer A and timer B counts.

### (6) fc

This clock has the same frequency as the sub-clock. It is used for the BCLK and for the watchdog timer.



## Clock Generating Circuit

Figure 1.12.4 shows the system clock control registers 0 and 1.

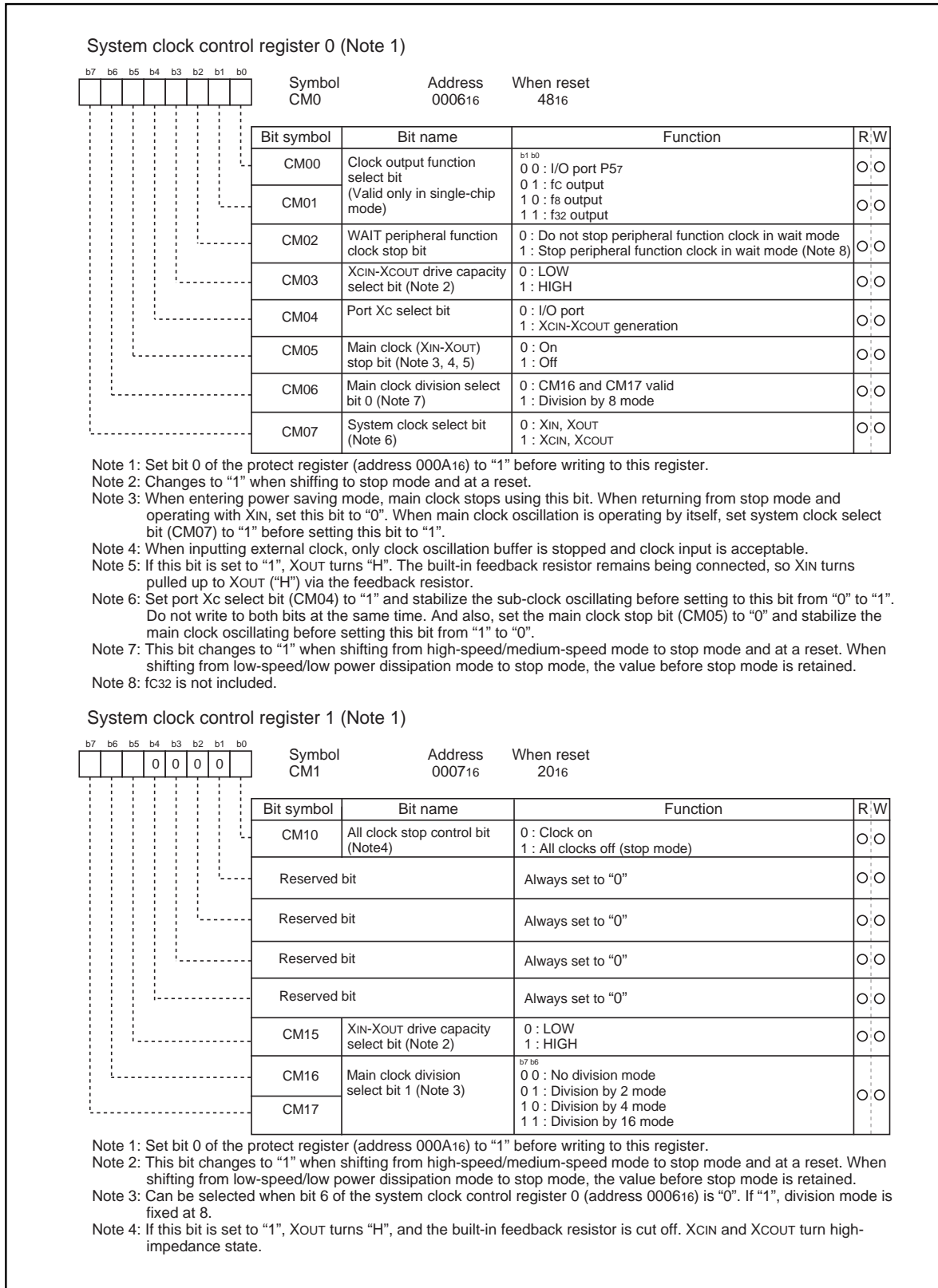


Figure 1.12.4. Clock control registers 0 and 1

**Clock Output**

In single-chip mode, the clock output function select bits (bits 0 and 1 at address 000616) enable f8, f32, or fc to be output from the P57/CLKOUT pin. When the WAIT peripheral function clock stop bit (bit 2 at address 000616) is set to “1”, the output of f8 and f32 stops when a WAIT instruction is executed.

**Stop Mode**

Writing “1” to the all-clock stop control bit (bit 0 at address 000716) stops all oscillation and the microcomputer enters stop mode. In stop mode, the content of the internal RAM is retained provided that VCC remains above 2V.

Because the oscillation, BCLK, f1 to f32, f1SIO2 to f32SIO2, fc, fc32, and fAD stops in stop mode, peripheral functions such as the A-D converter and watchdog timer do not function. However, timer A and timer B operate provided that the event counter mode is set to an external pulse, and UARTi(i = 0 to 2) functions provided an external clock is selected. Table 1.12.2 shows the status of the ports in stop mode.

Stop mode is cancelled by a hardware reset or interrupt. If an interrupt is to be used to cancel stop mode, that interrupt must first have been enabled.

When shifting from high-speed/medium-speed mode to stop mode and at a reset, the main clock division select bit 0 (bit 6 at address 000616) is set to “1”. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained.

**Table 1.12.2. Port status during stop mode**

Pin		Memory expansion mode Microprocessor mode	Single-chip mode
Address bus, data bus, CS0 to CS3		Retains status before stop mode	/
RD, WR, BHE, WRL, WRH		“H”	
HLDA, BCLK		“H”	
ALE		“H”	
Port		Retains status before stop mode	Retains status before stop mode
CLKOUT	When fc selected	Valid only in single-chip mode	“H”
	When f8, f32 selected	Valid only in single-chip mode	Retains status before stop mode

**Wait Mode**

When a WAIT instruction is executed, BCLK stops and the microcomputer enters the wait mode. In this mode, oscillation continues but BCLK and watchdog timer stop. Writing "1" to the WAIT peripheral function clock stop bit and executing a WAIT instruction stops the clock being supplied to the internal peripheral functions, allowing power dissipation to be reduced. Table 1.12.3 shows the status of the ports in wait mode.

Wait mode is cancelled by a hardware reset or an interrupt. If an interrupt is used to cancel wait mode, the microcomputer restarts from the interrupt routine using as BCLK, the clock that had been selected when the WAIT instruction was executed.

**Table 1.12.3. Port status during wait mode**

Pin		Memory expansion mode Microprocessor mode	Single-chip mode
Address bus, data bus, $\overline{CS0}$ to $\overline{CS3}$		Retains status before wait mode	/
$\overline{RD}$ , $\overline{WR}$ , $\overline{BHE}$ , $\overline{WRL}$ , $\overline{WRH}$		"H"	
$\overline{HLDA}$ , BCLK		"H"	
ALE		"H"	
Port		Retains status before wait mode	Retains status before wait mode
CLKOUT	When fc selected	Valid only in single-chip mode	Does not stop
	When f8, f32 selected	Valid only in single-chip mode	Does not stop when the WAIT peripheral function clock stop bit is "0". When the WAIT peripheral function clock stop bit is "1", the status immediately prior to entering wait mode is maintained.

## Status Transition Of BCLK

Power dissipation can be reduced and low-voltage operation achieved by changing the count source for BCLK. Table 1.13.4 shows the operating modes corresponding to the settings of system clock control registers 0 and 1.

When reset, the device starts in division by 8 mode. The main clock division select bit 0 (bit 6 at address 0006<sub>16</sub>) changes to "1" when shifting from high-speed/medium-speed to stop mode and at a reset. When shifting from low-speed/low power dissipation mode to stop mode, the value before stop mode is retained. The following shows the operational modes of BCLK.

### (1) Division by 2 mode

The main clock is divided by 2 to obtain the BCLK.

### (2) Division by 4 mode

The main clock is divided by 4 to obtain the BCLK.

### (3) Division by 8 mode

The main clock is divided by 8 to obtain the BCLK. When reset, the device starts operating from this mode. Before the user can go from this mode to no division mode, division by 2 mode, or division by 4 mode, the main clock must be oscillating stably. When going to low-speed or lower power consumption mode, make sure the sub-clock is oscillating stably.

### (4) Division by 16 mode

The main clock is divided by 16 to obtain the BCLK.

### (5) No-division mode

The main clock is divided by 1 to obtain the BCLK.

### (6) Low-speed mode

fc is used as the BCLK. Note that oscillation of both the main and sub-clocks must have stabilized before transferring from this mode to another or vice versa. At least 2 to 3 seconds are required after the sub-clock starts. Therefore, the program must be written to wait until this clock has stabilized immediately after powering up and after stop mode is cancelled.

### (7) Low power dissipation mode

fc is the BCLK and the main clock is stopped.

Note : Before the count source for BCLK can be changed from XIN to XCIN or vice versa, the clock to which the count source is going to be switched must be oscillating stably. Allow a wait time in software for the oscillation to stabilize before switching over the clock.

**Table 1.12.4. Operating modes dictated by settings of system clock control registers 0 and 1**

CM17	CM16	CM07	CM06	CM05	CM04	Operating mode of BCLK
0	1	0	0	0	Invalid	Division by 2 mode
1	0	0	0	0	Invalid	Division by 4 mode
Invalid	Invalid	0	1	0	Invalid	Division by 8 mode
1	1	0	0	0	Invalid	Division by 16 mode
0	0	0	0	0	Invalid	No-division mode
Invalid	Invalid	1	Invalid	0	1	Low-speed mode
Invalid	Invalid	1	Invalid	1	1	Low power dissipation mode

## Power control

The following is a description of the three available power control modes:

### Modes

Power control is available in three modes.

#### (a) Normal operation mode

- **High-speed mode**

Divide-by-1 frequency of the main clock becomes the BCLK. The CPU operates with the internal clock selected. Each peripheral function operates according to its assigned clock.

- **Medium-speed mode**

Divide-by-2, divide-by-4, divide-by-8, or divide-by-16 frequency of the main clock becomes the BCLK. The CPU operates according to the internal clock selected. Each peripheral function operates according to its assigned clock.

- **Low-speed mode**

fc becomes the BCLK. The CPU operates according to the fc clock. The fc clock is supplied by the secondary clock. Each peripheral function operates according to its assigned clock.

- **Low power consumption mode**

The main clock operating in low-speed mode is stopped. The CPU operates according to the fc clock. The fc clock is supplied by the secondary clock. The only peripheral functions that operate are those with the sub-clock selected as the count source.

#### (b) Wait mode

The CPU operation is stopped. The oscillators do not stop.

#### (c) Stop mode

All oscillators stop. The CPU and all built-in peripheral functions stop. This mode, among the three modes listed here, is the most effective in decreasing power consumption.

Figure 1.12.5 is the state transition diagram of the above modes.

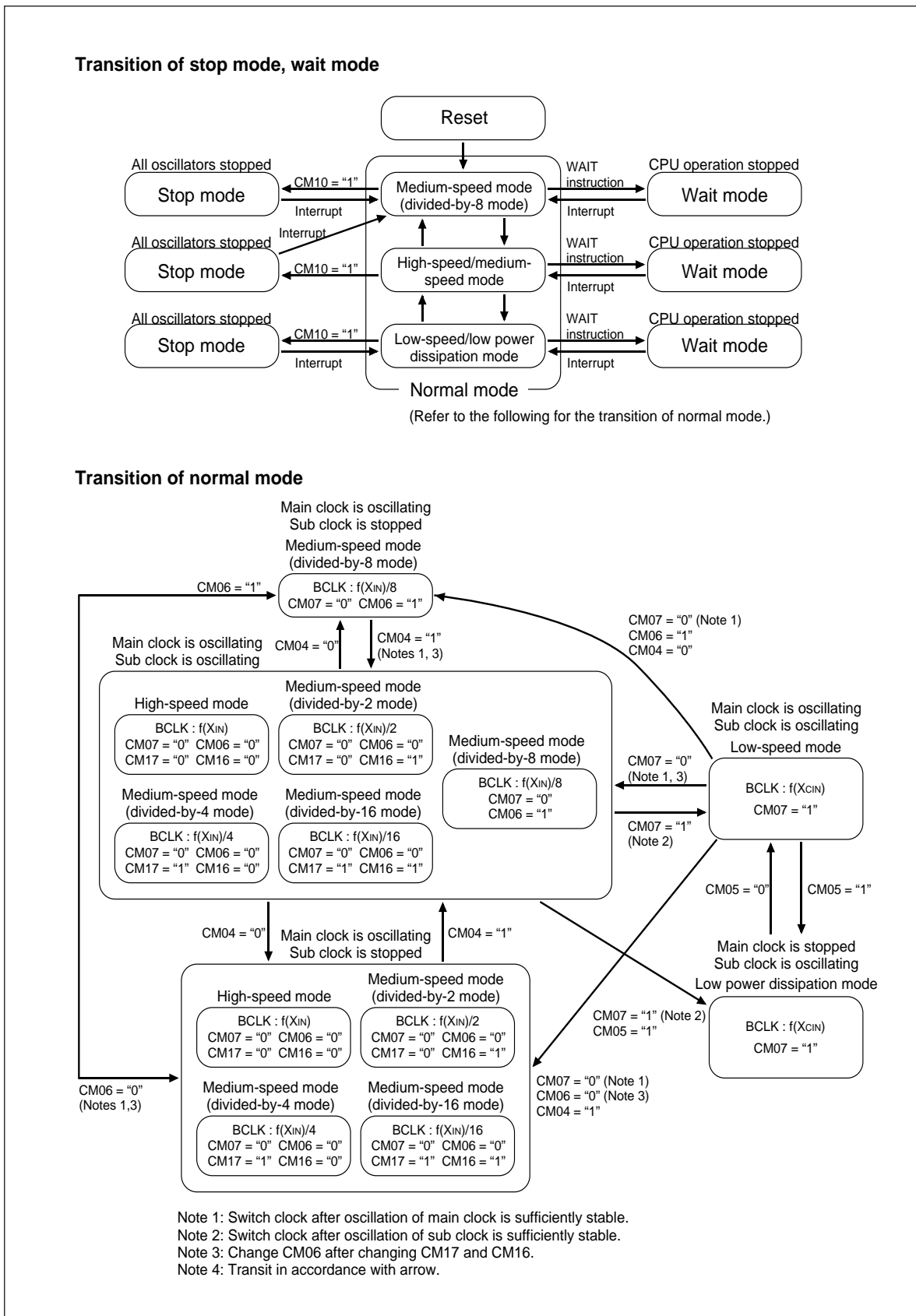


Figure 1.12.5. State transition diagram of Power control mode

## Protection

## Protection

The protection function is provided so that the values in important registers cannot be changed in the event that the program runs out of control. Figure 1.12.6 shows the protect register. The values in the processor mode register 0 (address 0004<sub>16</sub>), processor mode register 1 (address 0005<sub>16</sub>), system clock control register 0 (address 0006<sub>16</sub>), system clock control register 1 (address 0007<sub>16</sub>) and port P9 direction register (address 03F3<sub>16</sub>) can only be changed when the respective bit in the protect register is set to "1". Therefore, important outputs can be allocated to port P9.

If, after "1" (write-enabled) has been written to the port P9 direction register write-enable bit (bit 2 at address 000A<sub>16</sub>), a value is written to any address, the bit automatically reverts to "0" (write-inhibited). However, the system clock control registers 0 and 1 write-enable bit (bit 0 at 000A<sub>16</sub>) and processor mode register 0 and 1 write-enable bit (bit 1 at 000A<sub>16</sub>) do not automatically return to "0" after a value has been written to an address. The program must therefore be written to return these bits to "0".

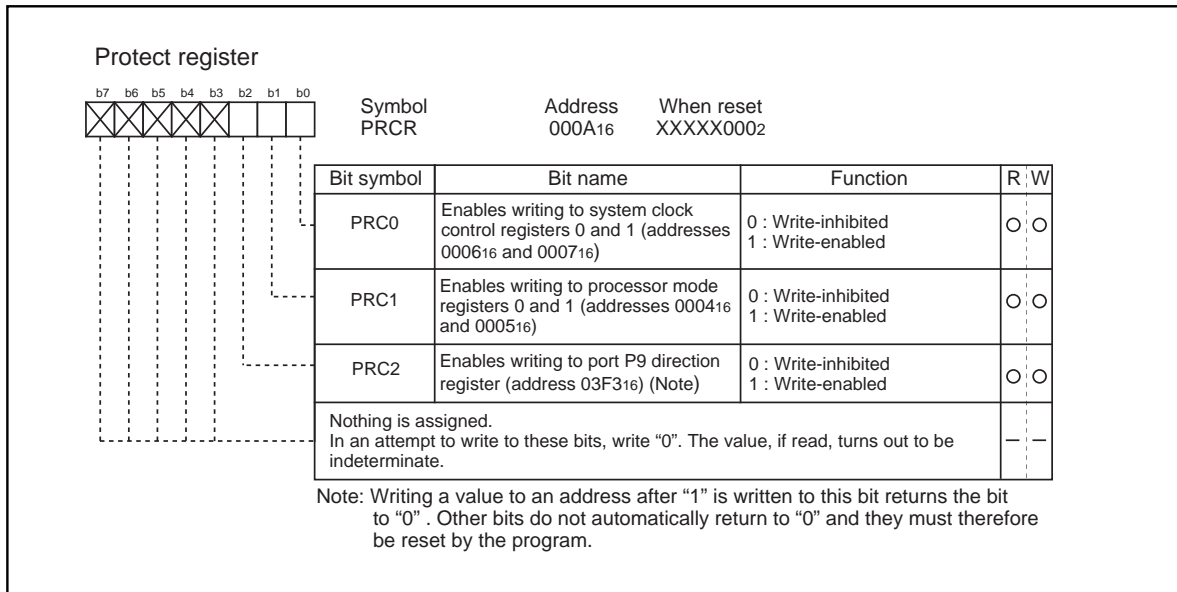


Figure 1.12.6. Protect register

## Overview of Interrupt

### Type of Interrupts

Figure 1.13.1 lists the types of interrupts.

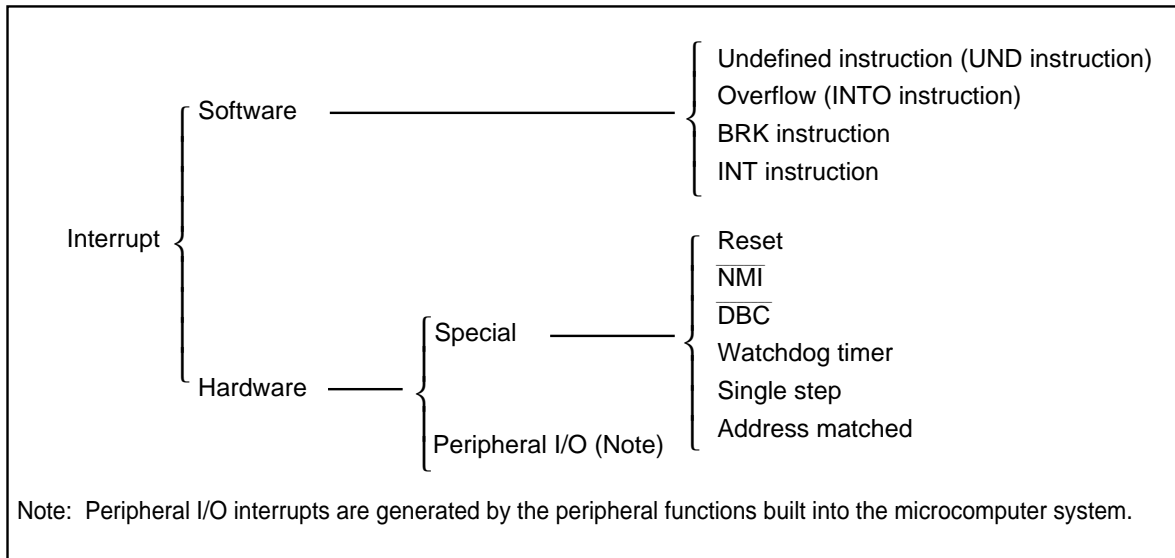


Figure 1.13.1. Classification of interrupts

- Maskable interrupt : An interrupt which can be enabled (disabled) by the interrupt enable flag (I flag) or whose interrupt priority **can be changed** by priority level.
- Non-maskable interrupt : An interrupt which cannot be enabled (disabled) by the interrupt enable flag (I flag) or whose interrupt priority **cannot be changed** by priority level.



---

## Software Interrupts

A software interrupt occurs when executing certain instructions. Software interrupts are non-maskable interrupts.

- **Undefined instruction interrupt**

An undefined instruction interrupt occurs when executing the UND instruction.

- **Overflow interrupt**

An overflow interrupt occurs when executing the INTO instruction with the overflow flag (O flag) set to "1". The following are instructions whose O flag changes by arithmetic:

ABS, ADC, ADCF, ADD, CMP, DIV, DIVU, DIVX, NEG, RMPA, SBB, SHA, SUB

- **BRK interrupt**

A BRK interrupt occurs when executing the BRK instruction.

- **INT interrupt**

An INT interrupt occurs when assigning one of software interrupt numbers 0 through 63 and executing the INT instruction. Software interrupt numbers 0 through 31 are assigned to peripheral I/O interrupts, so executing the INT instruction allows executing the same interrupt routine that a peripheral I/O interrupt does.

The stack pointer (SP) used for the INT interrupt is dependent on which software interrupt number is involved.

So far as software interrupt numbers 0 through 31 are concerned, the microcomputer saves the stack pointer assignment flag (U flag) when it accepts an interrupt request. It changes the U flag to "0" and selects the interrupt stack pointer (ISP), and then executes an interrupt sequence. When returning from the interrupt routine, the U flag is returned to the state it was before the acceptance of interrupt request. So far as software numbers 32 through 63 are concerned, the stack pointer does not make a shift.

## Hardware Interrupts

Hardware interrupts are classified into two types — special interrupts and peripheral I/O interrupts.

### (1) Special interrupts

Special interrupts are non-maskable interrupts.

- **Reset**

Reset occurs if an “L” is input to the  $\overline{\text{RESET}}$  pin.

- **$\overline{\text{NMI}}$  interrupt**

An  $\overline{\text{NMI}}$  interrupt occurs if an “L” is input to the  $\overline{\text{NMI}}$  pin.

- **$\overline{\text{DBC}}$  interrupt**

This interrupt is exclusively for the debugger, do not use it in other circumstances.

- **Watchdog timer interrupt**

Generated by the watchdog timer.

- **Single-step interrupt**

This interrupt is exclusively for the debugger, do not use it in other circumstances. With the debug flag (D flag) set to “1”, a single-step interrupt occurs after one instruction is executed.

- **Address match interrupt**

An address match interrupt occurs immediately before the instruction held in the address indicated by the address match interrupt register is executed with the address match interrupt enable bit set to “1”. If an address other than the first address of the instruction in the address match interrupt register is set, no address match interrupt occurs. For address match interrupt, see 2.11 Address match Interrupt.

### (2) Peripheral I/O interrupts

A peripheral I/O interrupt is generated by one of built-in peripheral functions. Built-in peripheral functions are dependent on classes of products, so the interrupt factors too are dependent on classes of products. The interrupt vector table is the same as the one for software interrupt numbers 0 through 31 the INT instruction uses. Peripheral I/O interrupts are maskable interrupts.

- **Bus collision detection interrupt**

This is an interrupt that the serial I/O bus collision detection generates.

- **DMA0 interrupt, DMA1 interrupt**

These are interrupts that DMA generates.

- **Key-input interrupt**

A key-input interrupt occurs if an “L” is input to the  $\overline{\text{KI}}$  pin.

- **A-D conversion interrupt**

This is an interrupt that the A-D converter generates.

- **UART0, UART1 and UART2 transmission interrupt**

These are interrupts that the serial I/O transmission generates.

- **UART0, UART1 and UART2 reception interrupt**

These are interrupts that the serial I/O reception generates.

- **Timer A0 interrupt through timer A4 interrupt**

These are interrupts that timer A generates

- **Timer B0 interrupt through timer B2 interrupt**

These are interrupts that timer B generates.

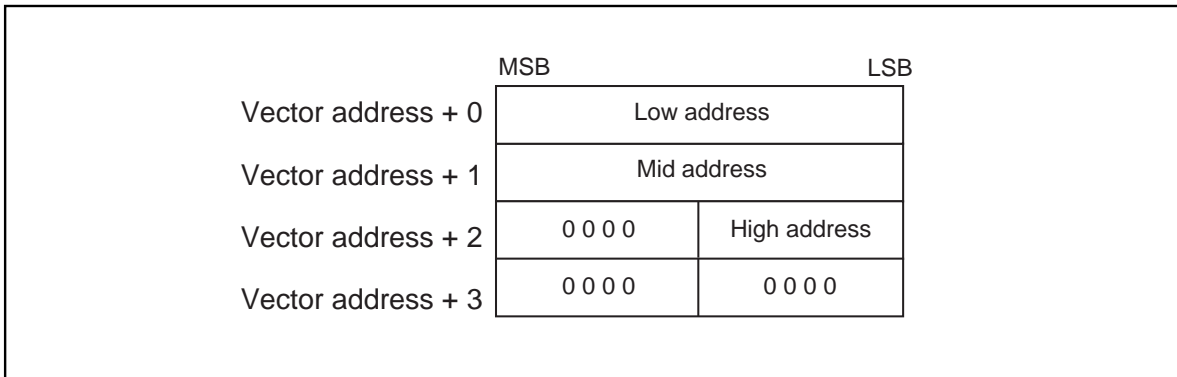
- **$\overline{\text{INT0}}$  interrupt through  $\overline{\text{INT2}}$  interrupt**

An  $\overline{\text{INT}}$  interrupt occurs if either a rising edge or a falling edge is input to the  $\overline{\text{INT}}$  pin.

### Interrupts and Interrupt Vector Tables

If an interrupt request is accepted, a program branches to the interrupt routine set in the interrupt vector table. Set the first address of the interrupt routine in each vector table. Figure 1.13.2 shows the format for specifying the address.

Two types of interrupt vector tables are available — fixed vector table in which addresses are fixed and variable vector table in which addresses can be varied by the setting.



**Figure 1.13.2. Format for specifying interrupt vector addresses**

- **Fixed vector tables**

The fixed vector table is a table in which addresses are fixed. The vector tables are located in an area extending from FFFDC<sub>16</sub> to FFFFF<sub>16</sub>. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 1.13.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

**Table 1.13.1. Interrupts assigned to the fixed vector tables and addresses of vector tables**

Interrupt source	Vector table addresses Address (L) to address (H)	Remarks
Undefined instruction	FFFD <sub>16</sub> to FFFD <sub>16</sub> F	Interrupt on UND instruction
Overflow	FFFE <sub>16</sub> 0 to FFFE <sub>16</sub> 3	Interrupt on INTO instruction
BRK instruction	FFFE <sub>16</sub> 4 to FFFE <sub>16</sub> 7	If the vector contains FF <sub>16</sub> , program execution starts from the address shown by the vector in the variable vector table
Address match	FFFE <sub>16</sub> 8 to FFFE <sub>16</sub> B	There is an address-matching interrupt enable bit
Single step (Note)	FFFE <sub>16</sub> C to FFFE <sub>16</sub> F	Do not use
Watchdog timer	FFFF <sub>16</sub> 0 to FFFF <sub>16</sub> 3	
DBC (Note)	FFFF <sub>16</sub> 4 to FFFF <sub>16</sub> 7	Do not use
NMI	FFFF <sub>16</sub> 8 to FFFF <sub>16</sub> B	External interrupt by input to NMI pin
Reset	FFFF <sub>16</sub> C to FFFF <sub>16</sub> F	

Note: Interrupts used for debugging purposes only.

• **Variable vector tables**

The addresses in the variable vector table can be modified, according to the user's settings. Indicate the first address using the interrupt table register (INTB). The 256-byte area subsequent to the address the INTB indicates becomes the area for the variable vector tables. One vector table comprises four bytes. Set the first address of the interrupt routine in each vector table. Table 1.13.2 shows the interrupts assigned to the variable vector tables and addresses of vector tables.

**Table 1.13.2. Interrupts assigned to the variable vector tables and addresses of vector tables**

Software interrupt number	Vector table address Address (L) to address (H)	Interrupt source	Remarks
Software interrupt number 0	+0 to +3 (Note)	BRK instruction	Cannot be masked I flag
Software interrupt number 10	+40 to +43 (Note)	Bus collision detection	
Software interrupt number 11	+44 to +47 (Note)	DMA0	
Software interrupt number 12	+48 to +51 (Note)	DMA1	
Software interrupt number 13	+52 to +55 (Note)	Key input interrupt	
Software interrupt number 14	+56 to +59 (Note)	A-D	
Software interrupt number 15	+60 to +63 (Note)	UART2 transmit	
Software interrupt number 16	+64 to +67 (Note)	UART2 receive	
Software interrupt number 17	+68 to +71 (Note)	UART0 transmit	
Software interrupt number 18	+72 to +75 (Note)	UART0 receive	
Software interrupt number 19	+76 to +79 (Note)	UART1 transmit	
Software interrupt number 20	+80 to +83 (Note)	UART1 receive	
Software interrupt number 21	+84 to +87 (Note)	Timer A0	
Software interrupt number 22	+88 to +91 (Note)	Timer A1	
Software interrupt number 23	+92 to +95 (Note)	Timer A2	
Software interrupt number 24	+96 to +99 (Note)	Timer A3	
Software interrupt number 25	+100 to +103 (Note)	Timer A4	
Software interrupt number 26	+104 to +107 (Note)	Timer B0	
Software interrupt number 27	+108 to +111 (Note)	Timer B1	
Software interrupt number 28	+112 to +115 (Note)	Timer B2	
Software interrupt number 29	+116 to +119 (Note)	$\overline{\text{INT0}}$	
Software interrupt number 30	+120 to +123 (Note)	$\overline{\text{INT1}}$	
Software interrupt number 31	+124 to +127 (Note)	$\overline{\text{INT2}}$	
Software interrupt number 32 to Software interrupt number 63	+128 to +131 (Note) to +252 to +255 (Note)	Software interrupt	Cannot be masked I flag

Note: Address relative to address in interrupt table register (INTB)

---

## Interrupt Control

Descriptions are given here regarding how to enable or disable maskable interrupts and how to set the priority to be accepted. What is described here does not apply to non-maskable interrupts.

Enable or disable a non-maskable interrupt using the interrupt enable flag (I flag), interrupt priority level selection bit, or processor interrupt priority level (IPL). Whether an interrupt request is present or absent is indicated by the interrupt request bit. The interrupt request bit and the interrupt priority level selection bit are located in the interrupt control register of each interrupt. Also, the interrupt enable flag (I flag) and the IPL are located in the flag register (FLG).

Figure 1.13.3 shows the memory map of the interrupt control registers.

Interrupt

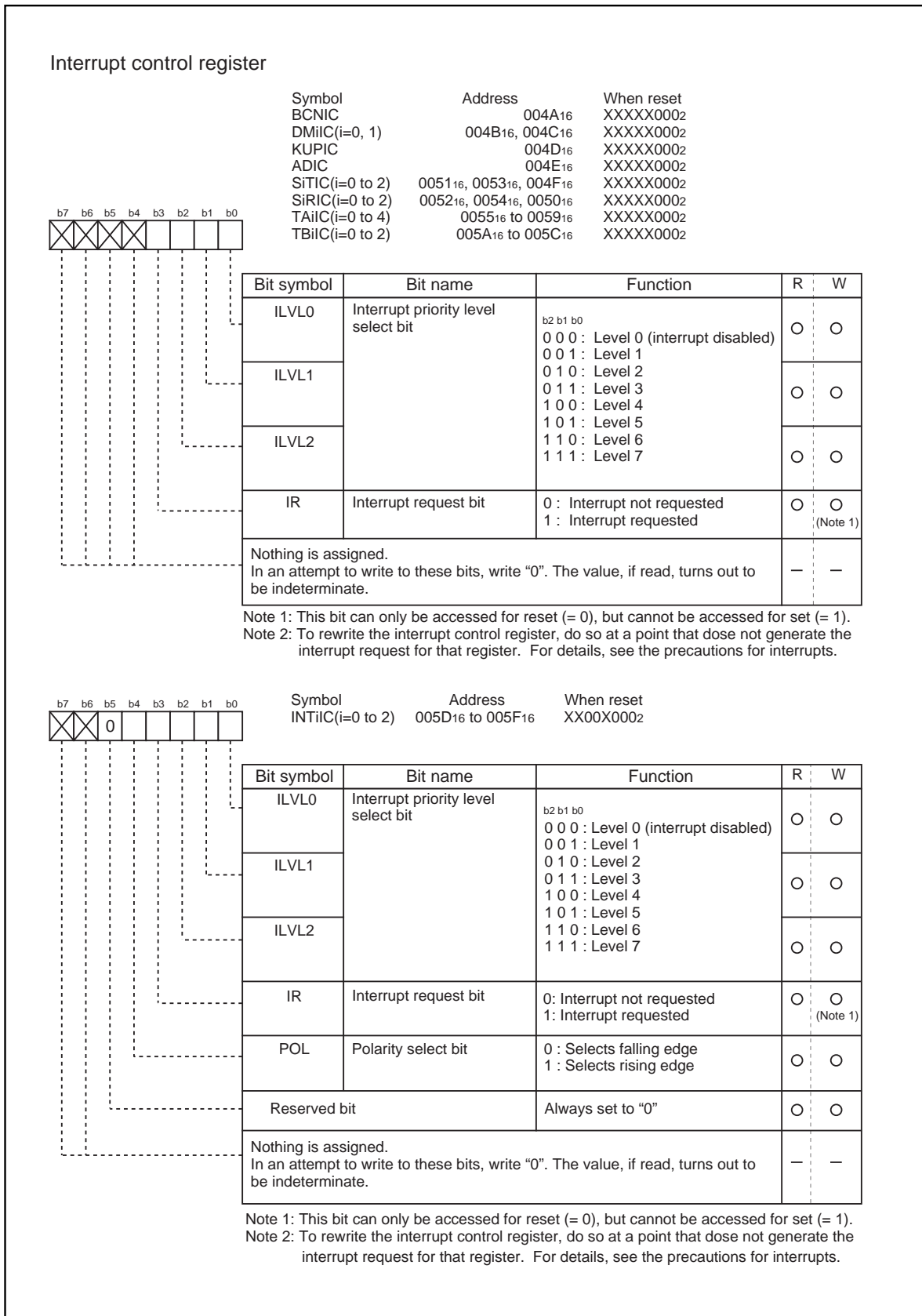


Figure 1.13.3. Interrupt control registers

### Interrupt Enable Flag (I flag)

The interrupt enable flag (I flag) controls the enabling and disabling of maskable interrupts. Setting this flag to "1" enables all maskable interrupts; setting it to "0" disables all maskable interrupts. This flag is set to "0" after reset.

### Interrupt Request Bit

The interrupt request bit is set to "1" by hardware when an interrupt is requested. After the interrupt is accepted and jumps to the corresponding interrupt vector, the request bit is set to "0" by hardware. The interrupt request bit can also be set to "0" by software. (Do not set this bit to "1").

### Interrupt Priority Level Select Bit and Processor Interrupt Priority Level (IPL)

Set the interrupt priority level using the interrupt priority level select bit, which is one of the component bits of the interrupt control register. When an interrupt request occurs, the interrupt priority level is compared with the IPL. The interrupt is enabled only when the priority level of the interrupt is higher than the IPL. Therefore, setting the interrupt priority level to "0" disables the interrupt.

Table 1.13.3 shows the settings of interrupt priority levels and Table 1.13.4 shows the interrupt levels enabled, according to the consist of the IPL.

The following are conditions under which an interrupt is accepted:

- interrupt enable flag (I flag) = 1
- interrupt request bit = 1
- interrupt priority level > IPL

The interrupt enable flag (I flag), the interrupt request bit, the interrupt priority select bit, and the IPL are independent, and they are not affected by one another.

**Table 1.13.3. Settings of interrupt priority levels**

Interrupt priority level select bit	Interrupt priority level	Priority order
b2 b1 b0 0 0 0	Level 0 (interrupt disabled)	———
0 0 1	Level 1	Low ↓ High
0 1 0	Level 2	
0 1 1	Level 3	
1 0 0	Level 4	
1 0 1	Level 5	
1 1 0	Level 6	
1 1 1	Level 7	

**Table 1.13.4. Interrupt levels enabled according to the contents of the IPL**

IPL	Enabled interrupt priority levels
IPL <sub>2</sub> IPL <sub>1</sub> IPL <sub>0</sub> 0 0 0	Interrupt levels 1 and above are enabled
0 0 1	Interrupt levels 2 and above are enabled
0 1 0	Interrupt levels 3 and above are enabled
0 1 1	Interrupt levels 4 and above are enabled
1 0 0	Interrupt levels 5 and above are enabled
1 0 1	Interrupt levels 6 and above are enabled
1 1 0	Interrupt levels 7 and above are enabled
1 1 1	All maskable interrupts are disabled

**Rewrite the interrupt control register**

To rewrite the interrupt control register, do so at a point that does not generate the interrupt request for that register. If there is possibility of the interrupt request occur, rewrite the interrupt control register after the interrupt is disabled. The program examples are described as follow:

**Example 1:**

```
INT_SWITCH1:
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  NOP                               ; Four NOP instructions are required when using HOLD function.
  NOP
  FSET  I           ; Enable interrupts.
```

**Example 2:**

```
INT_SWITCH2:
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  MOV.W MEM, R0     ; Dummy read.
  FSET  I           ; Enable interrupts.
```

**Example 3:**

```
INT_SWITCH3:
  PUSHC FLG         ; Push Flag register onto stack
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  POPC  FLG         ; Enable interrupts.
```

The reason why two NOP instructions (four when using the HOLD function) or dummy read are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

When a instruction to rewrite the interrupt control register is executed but the interrupt is disabled, the interrupt request bit is not set sometimes even if the interrupt request for that register has been generated. This will depend on the instruction. If this creates problems, use the below instructions to change the register.

Instructions : AND, OR, BCLR, BSET



## Interrupt Sequence

An interrupt sequence — what are performed over a period from the instant an interrupt is accepted to the instant the interrupt routine is executed — is described here.

If an interrupt occurs during execution of an instruction, the processor determines its priority when the execution of the instruction is completed, and transfers control to the interrupt sequence from the next cycle. If an interrupt occurs during execution of either the SMOVB, SMOVF, SSTR or RMPA instruction, the processor temporarily suspends the instruction being executed, and transfers control to the interrupt sequence.

In the interrupt sequence, the processor carries out the following in sequence given:

- (1) CPU gets the interrupt information (the interrupt number and interrupt request level) by reading address  $00000_{16}$ .
- (2) Saves the content of the flag register (FLG) as it was immediately before the start of interrupt sequence in the temporary register (Note) within the CPU.
- (3) Sets the interrupt enable flag (I flag), the debug flag (D flag), and the stack pointer select flag (U flag) to "0" (the U flag, however does not change if the INT instruction, in software interrupt numbers 32 through 63, is executed)
- (4) Saves the content of the temporary register (Note 1) within the CPU in the stack area.
- (5) Saves the content of the program counter (PC) in the stack area.
- (6) Sets the interrupt priority level of the accepted instruction in the IPL.

After the interrupt sequence is completed, the processor resumes executing instructions from the first address of the interrupt routine.

Note: This register cannot be utilized by the user.

## Interrupt Response Time

'Interrupt response time' is the period between the instant an interrupt occurs and the instant the first instruction within the interrupt routine has been executed. This time comprises the period from the occurrence of an interrupt to the completion of the instruction under execution at that moment (a) and the time required for executing the interrupt sequence (b). Figure 1.13.4 shows the interrupt response time.

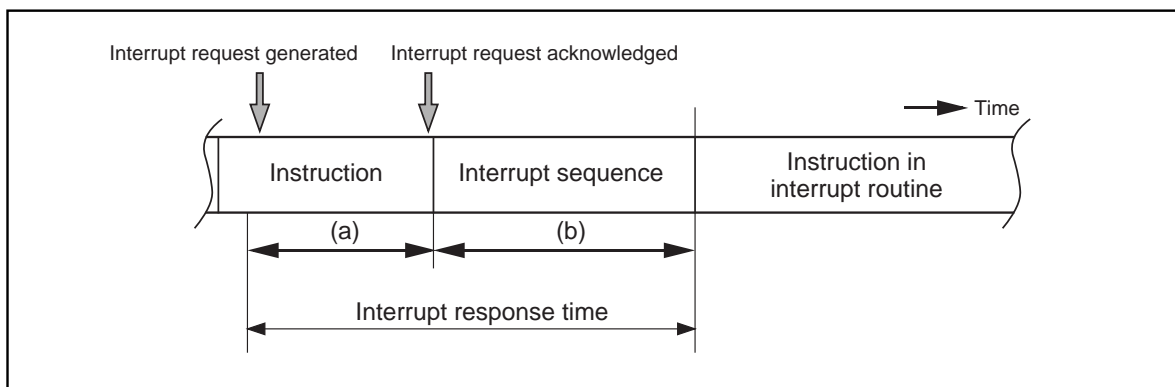


Figure 1.13.4. Interrupt response time

## Interrupt

Time (a) is dependent on the instruction under execution. Thirty cycles is the maximum required for the DIVX instruction (without wait).

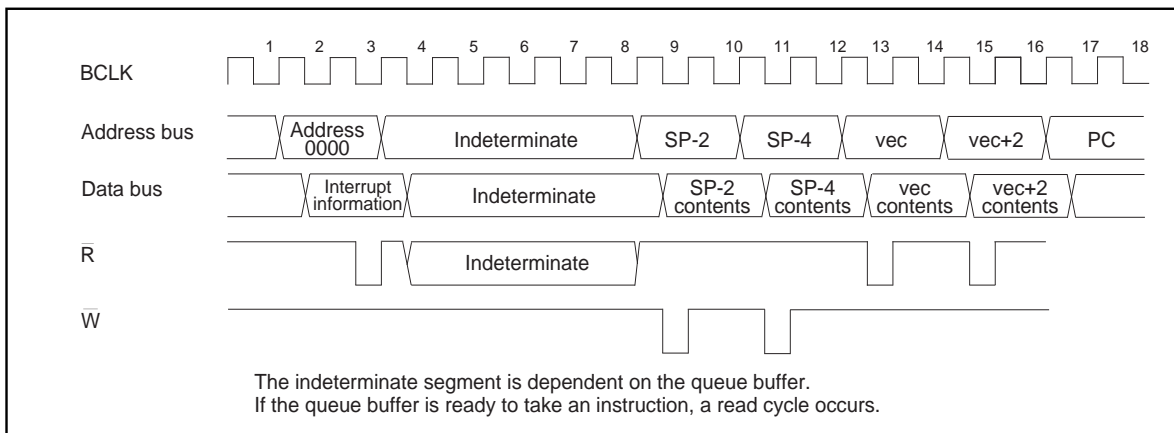
Time (b) is as shown in Table 1.13.5.

**Table 1.13.5. Time required for executing the interrupt sequence**

Interrupt vector address	Stack pointer (SP) value	16-Bit bus, without wait	8-Bit bus, without wait
Even	Even	18 cycles (Note 1)	20 cycles (Note 1)
Even	Odd	19 cycles (Note 1)	20 cycles (Note 1)
Odd (Note 2)	Even	19 cycles (Note 1)	20 cycles (Note 1)
Odd (Note 2)	Odd	20 cycles (Note 1)	20 cycles (Note 1)

Note 1: Add 2 cycles in the case of a DBC interrupt; add 1 cycle in the case either of an address coincidence interrupt or of a single-step interrupt.

Note 2: Locate an interrupt vector address in an even address, if possible.



**Figure 1.13.5. Time required for executing the interrupt sequence**

### Variation of IPL when Interrupt Request is Accepted

If an interrupt request is accepted, the interrupt priority level of the accepted interrupt is set in the IPL.

If an interrupt request, that does not have an interrupt priority level, is accepted, one of the values shown in Table 1.13.6 is set in the IPL.

**Table 1.13.6. Relationship between interrupts without interrupt priority levels and IPL**

Interrupt sources without priority levels	Value set in the IPL
Watchdog timer, NMI	7
Reset	0
Other	Not changed

### Saving Registers

In the interrupt sequence, only the contents of the flag register (FLG) and that of the program counter (PC) are saved in the stack area.

First, the processor saves the four higher-order bits of the program counter, and 4 upper-order bits and 8 lower-order bits of the FLG register, 16 bits in total, in the stack area, then saves 16 lower-order bits of the program counter. Figure 1.13.6 shows the state of the stack as it was before the acceptance of the interrupt request, and the state the stack after the acceptance of the interrupt request.

Save other necessary registers at the beginning of the interrupt routine using software. Using the PUSHM instruction alone can save all the registers except the stack pointer (SP).

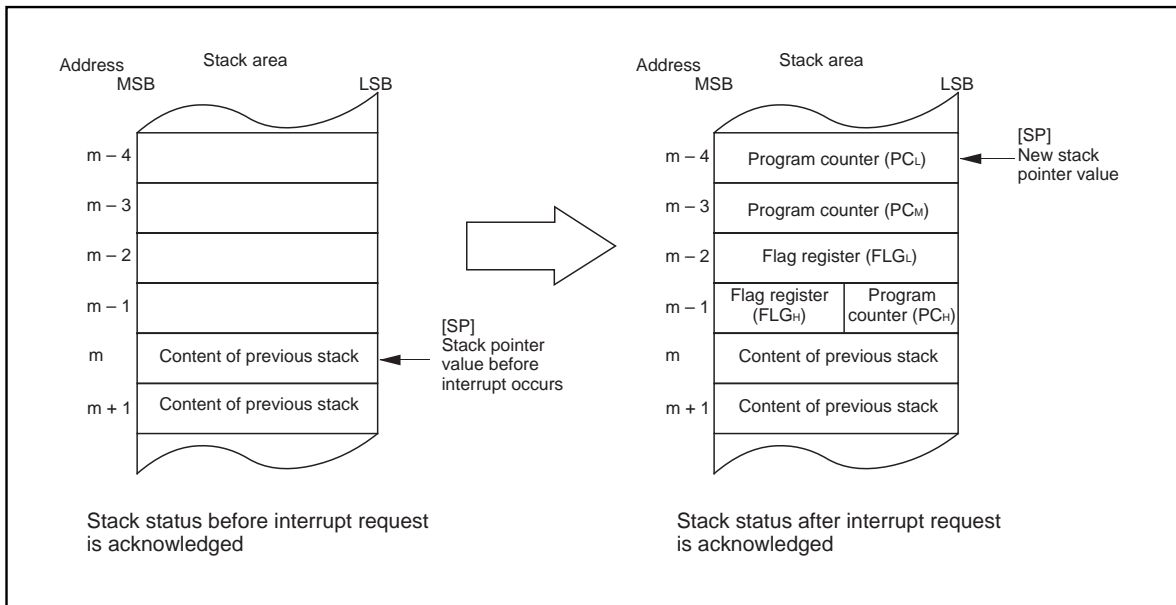


Figure 1.13.6. State of stack before and after acceptance of interrupt request

## Interrupt

The operation of saving registers carried out in the interrupt sequence is dependent on whether the content of the stack pointer, at the time of acceptance of an interrupt request, is even or odd. If the content of the stack pointer (Note) is even, the content of the flag register (FLG) and the content of the program counter (PC) are saved, 16 bits at a time. If odd, their contents are saved in two steps, 8 bits at a time. Figure 1.13.7 shows the operation of the saving registers.

Note: Stack pointer indicated by U flag.

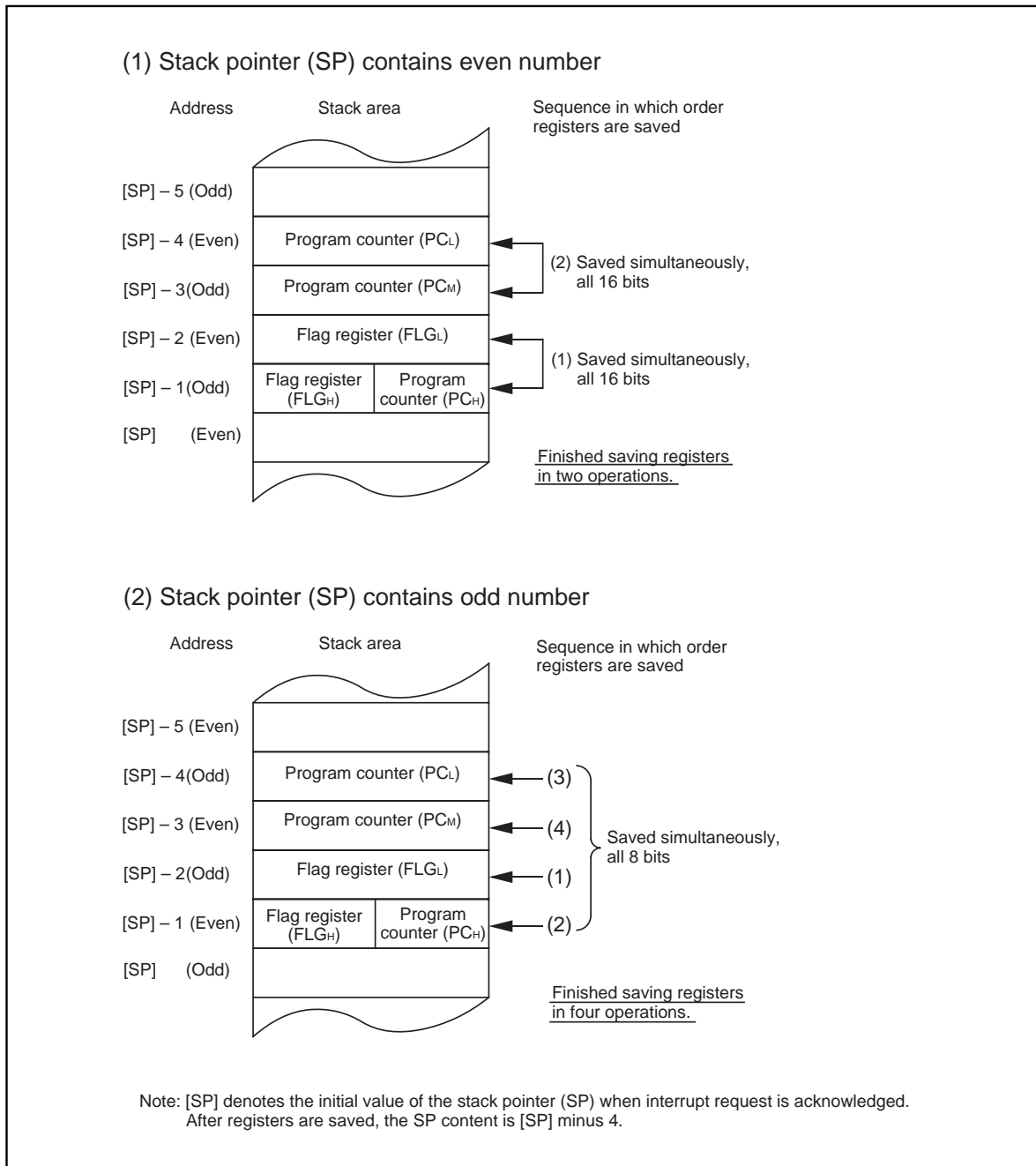


Figure 1.13.7. Operation of saving registers

### Returning from an Interrupt Routine

Executing the REIT instruction at the end of an interrupt routine returns the contents of the flag register (FLG) as it was immediately before the start of interrupt sequence and the contents of the program counter (PC), both of which have been saved in the stack area. Then control returns to the program that was being executed before the acceptance of the interrupt request, so that the suspended process resumes.

Return the other registers saved by software within the interrupt routine using the POPM or similar instruction before executing the REIT instruction.

### Interrupt Priority

If there are two or more interrupt requests occurring at a point in time within a single sampling (checking whether interrupt requests are made), the interrupt assigned a higher priority is accepted.

Assign an arbitrary priority to maskable interrupts (peripheral I/O interrupts) using the interrupt priority level select bit. If the same interrupt priority level is assigned, however, the interrupt assigned a higher hardware priority is accepted.

Priorities of the special interrupts, such as Reset (dealt with as an interrupt assigned the highest priority), watchdog timer interrupt, etc. are regulated by hardware.

Figure 1.13.8 shows the priorities of hardware interrupts.

Software interrupts are not affected by the interrupt priority. If an instruction is executed, control branches invariably to the interrupt routine.

Reset >  $\overline{\text{NMI}}$  >  $\overline{\text{DBC}}$  > Watchdog timer > Peripheral I/O > Single step > Address match

Figure 1.13.8. Hardware interrupts priorities

### Interrupt resolution circuit

When two or more interrupts are generated simultaneously, this circuit selects the interrupt with the highest priority level. Figure 1.13.9 shows the circuit that judges the interrupt priority level.

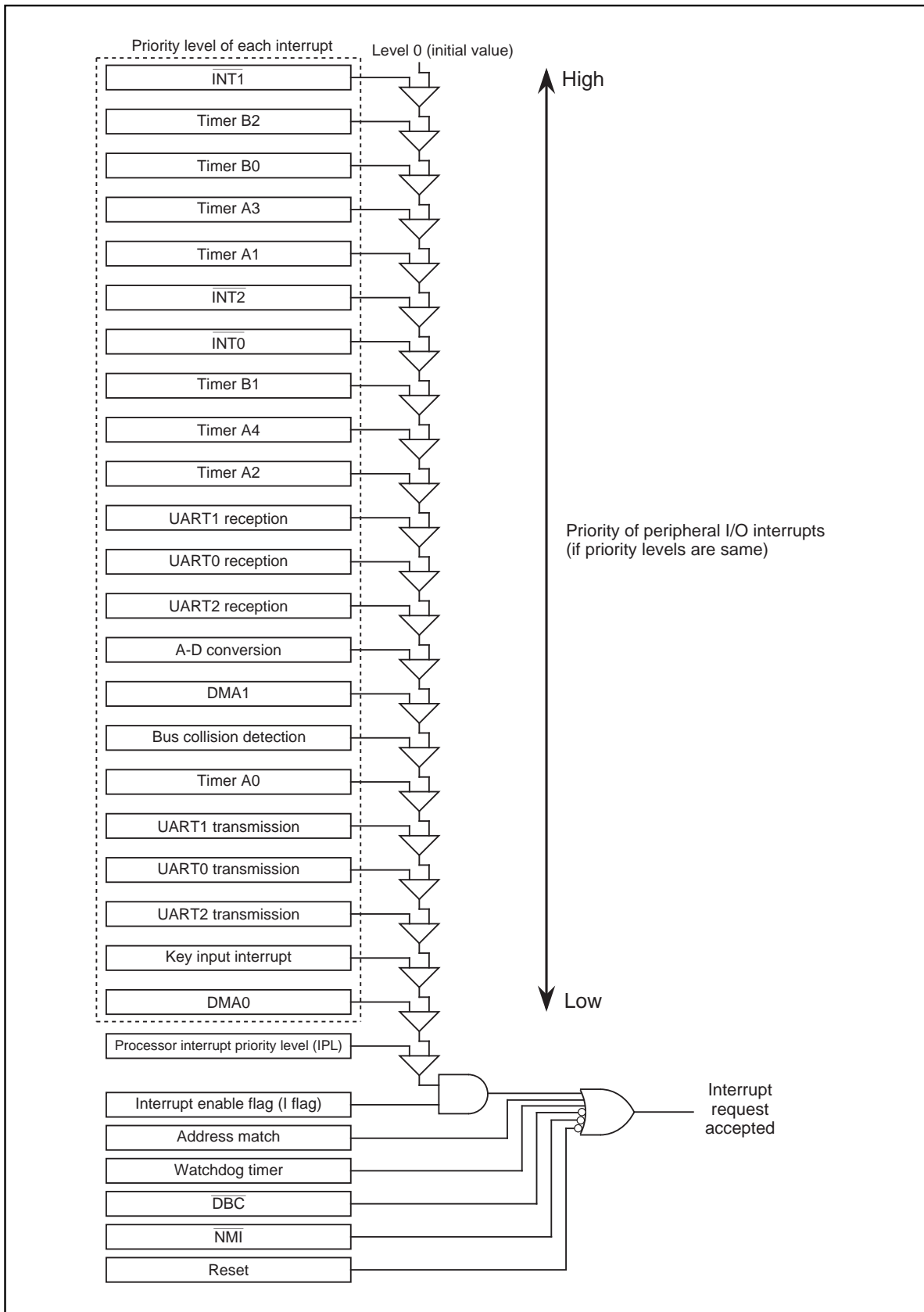


Figure 1.13.9. Maskable interrupts priorities (peripheral I/O interrupts)

## NMI Interrupt

### INT Interrupt

$\overline{\text{INT0}}$  to  $\overline{\text{INT2}}$  are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit.

### NMI Interrupt

An  $\overline{\text{NMI}}$  interrupt is generated when the input to the P85/ $\overline{\text{NMI}}$  pin changes from "H" to "L". The  $\overline{\text{NMI}}$  interrupt is a non-maskable external interrupt. The pin level can be checked in the port P85 register (bit 5 at address 03F0<sub>16</sub>).

This pin cannot be used as a normal port input.

### Key Input Interrupt

If the direction register of any of P104 to P107 is set for input and a falling edge is input to that port, a key input interrupt is generated. A key input interrupt can also be used as a key-on wakeup function for canceling the wait mode or stop mode. However, if you intend to use the key input interrupt, do not use P104 to P107 as A-D input ports. Figure 1.13.10 shows the block diagram of the key input interrupt. Note that if an "L" level is input to any pin that has not been disabled for input, inputs to the other pins are not detected as an interrupt.

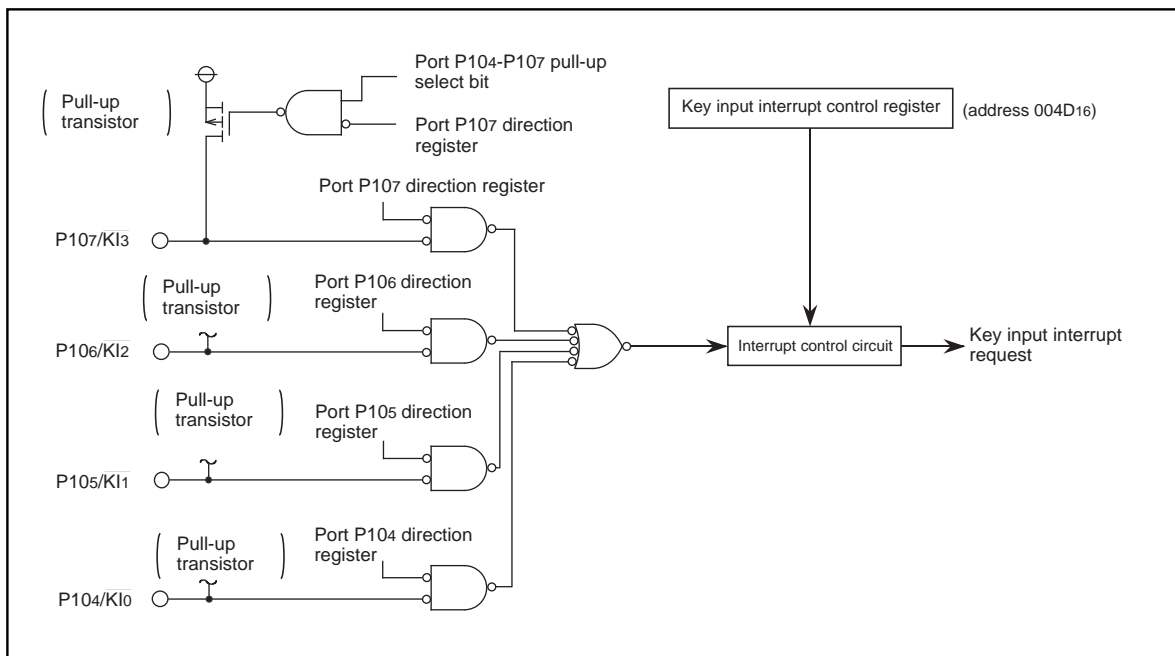


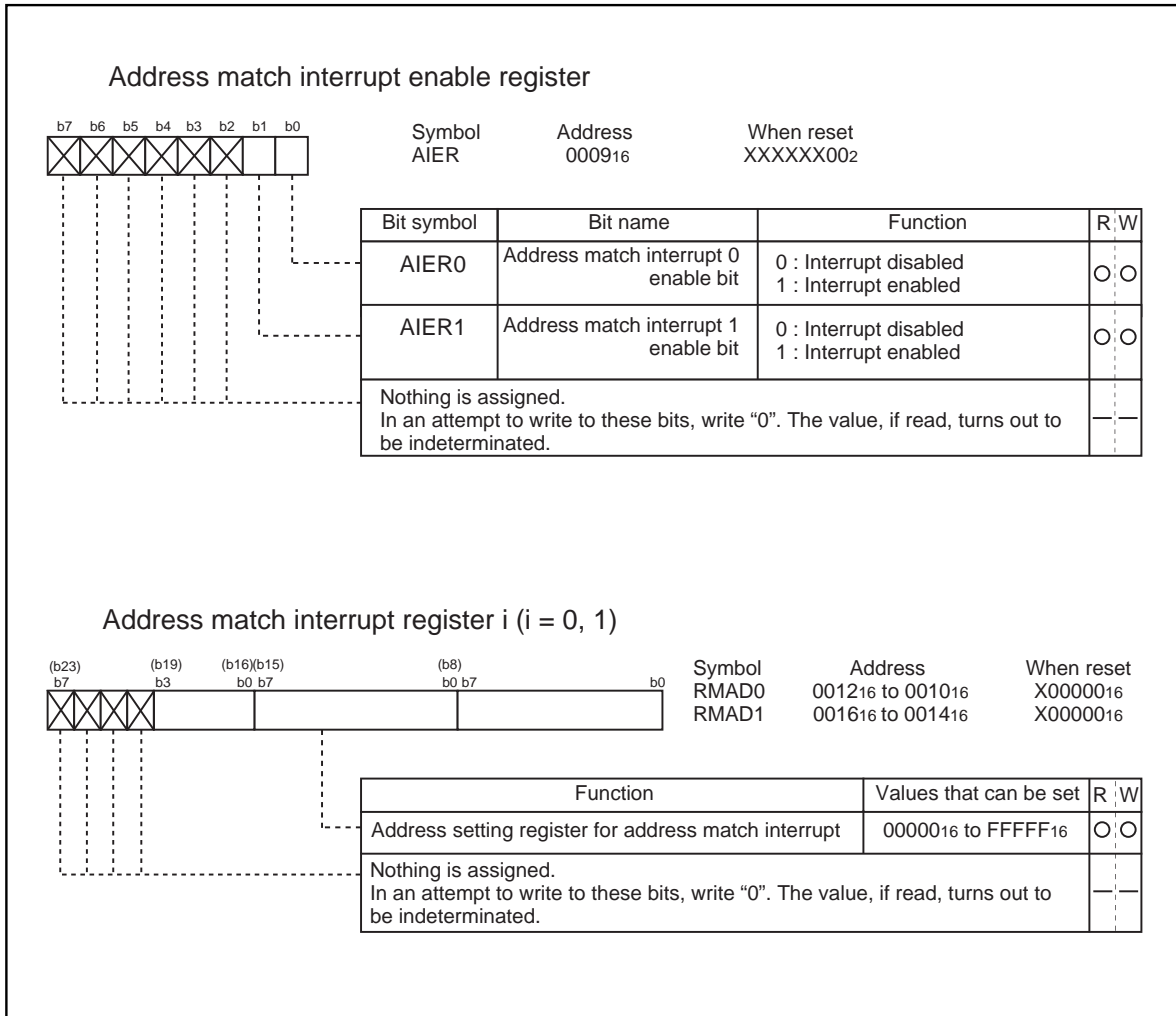
Figure 1.13.10. Block diagram of key input interrupt

Address Match Interrupt

**Address Match Interrupt**

An address match interrupt is generated when the address match interrupt address register contents match the program counter value. Two address match interrupts can be set, each of which can be enabled and disabled by an address match interrupt enable bit. Address match interrupts are not affected by the interrupt enable flag (I flag) and processor interrupt priority level (IPL). The value of the program counter (PC) for an address match interrupt varies depending on the instruction being executed.

Figure 1.13.11 shows the address match interrupt-related registers.



**Figure 1.13.11. Address match interrupt-related registers**



---

## Precautions for Interrupts

### (1) Reading address 00000<sub>16</sub>

- When maskable interrupt is occurred, CPU read the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence.

The interrupt request bit of the certain interrupt written in address 00000<sub>16</sub> will then be set to "0".

Reading address 00000<sub>16</sub> by software sets enabled highest priority interrupt source request bit to "0".

Though the interrupt is generated, the interrupt routine may not be executed.

Do not read address 00000<sub>16</sub> by software.

### (2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000<sub>16</sub>. Accepting an interrupt before setting a value in the stack pointer may become a factor of runaway. Be sure to set a value in the stack pointer before accepting an interrupt. When using the NMI interrupt, initialize the stack point at the beginning of a program. Concerning the first instruction immediately after reset, generating any interrupts including the  $\overline{\text{NMI}}$  interrupt is prohibited.

### (3) The $\overline{\text{NMI}}$ interrupt

- As for the  $\overline{\text{NMI}}$  interrupt pin, an interrupt cannot be disabled. Connect it to the VCC pin via a resistor (pull-up) if unused. Be sure to work on it.
- The  $\overline{\text{NMI}}$  pin also serves as P85, which is exclusively input. Reading the contents of the P8 register allows reading the pin value. Use the reading of this pin only for establishing the pin level at the time when the  $\overline{\text{NMI}}$  interrupt is input.
- Do not reset the CPU with the input to the  $\overline{\text{NMI}}$  pin being in the "L" state.
- Do not attempt to go into stop mode with the input to the  $\overline{\text{NMI}}$  pin being in the "L" state. With the input to the  $\overline{\text{NMI}}$  being in the "L" state, the CM10 is fixed to "0", so attempting to go into stop mode is turned down.
- Do not attempt to go into wait mode with the input to the  $\overline{\text{NMI}}$  pin being in the "L" state. With the input to the  $\overline{\text{NMI}}$  pin being in the "L" state, the CPU stops but the oscillation does not stop, so no power is saved. In this instance, the CPU is returned to the normal state by a later interrupt.
- Signals input to the  $\overline{\text{NMI}}$  pin require an "L" level of 1 clock or more, from the operation clock of the CPU.

### (4) External interrupt

- Either an "L" level or an "H" level of at least 250 ns width is necessary for the signal input to pins  $\overline{\text{INT}}_0$  through  $\overline{\text{INT}}_2$  regardless of the CPU operation clock.
- When the polarity of the  $\overline{\text{INT}}_0$  to  $\overline{\text{INT}}_2$  pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 1.13.12 shows the procedure for changing the  $\overline{\text{INT}}$  interrupt generate factor.

## Precautions for Interrupts

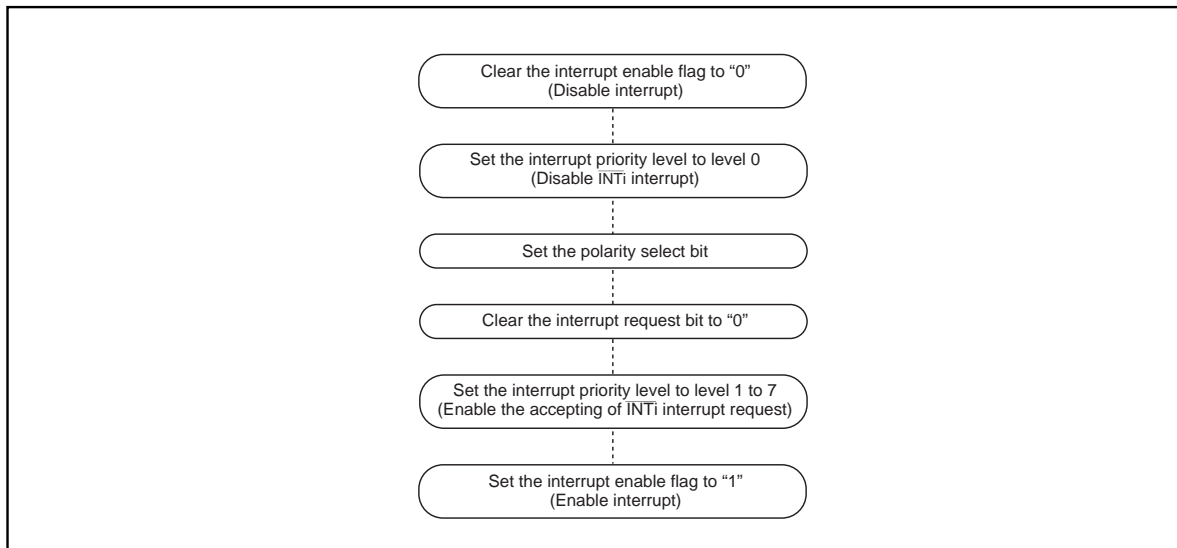


Figure 1.13.12. Switching condition of INT interrupt request

**(5) Rewrite the interrupt control register**

- To rewrite the interrupt control register, do so at a point that does not generate the interrupt request for that register. If there is possibility of the interrupt request occur, rewrite the interrupt control register after the interrupt is disabled. The program examples are described as follow:

**Example 1:**

```

INT_SWITCH1:
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  NOP                               ; Four NOP instructions are required when using HOLD function.
  NOP
  FSET  I           ; Enable interrupts.

```

**Example 2:**

```

INT_SWITCH2:
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  MOV.W MEM, R0    ; Dummy read.
  FSET  I           ; Enable interrupts.

```

**Example 3:**

```

INT_SWITCH3:
  PUSHC FLG        ; Push Flag register onto stack
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  POPC  FLG        ; Enable interrupts.

```

The reason why two NOP instructions (four when using the HOLD function) or dummy read are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

- When a instruction to rewrite the interrupt control register is executed but the interrupt is disabled, the interrupt request bit is not set sometimes even if the interrupt request for that register has been generated. This will depend on the instruction. If this creates problems, use the below instructions to change the register.

Instructions : AND, OR, BCLR, BSET

## Watchdog Timer

### Watchdog Timer

The watchdog timer has the function of detecting when the program is out of control. The watchdog timer is a 15-bit counter which down-counts the clock derived by dividing the BCLK using the prescaler. A watchdog timer interrupt is generated when an underflow occurs in the watchdog timer. When XIN is selected for the BCLK, bit 7 of the watchdog timer control register (address 000F16) selects the prescaler division ratio (by 16 or by 128). When XCIN is selected as the BCLK, the prescaler is set for division by 2 regardless of bit 7 of the watchdog timer control register (address 000F16). Thus the watchdog timer's period can be calculated as given below. The watchdog timer's period is, however, subject to an error due to the pre-scaler.

#### With XIN chosen for BCLK

$$\text{Watchdog timer period} = \frac{\text{pre-scaler dividing ratio (16 or 128)} \times \text{watchdog timer count (32768)}}{\text{BCLK}}$$

#### With XCIN chosen for BCLK

$$\text{Watchdog timer period} = \frac{\text{pre-scaler dividing ratio (2)} \times \text{watchdog timer count (32768)}}{\text{BCLK}}$$

For example, suppose that BCLK runs at 10 MHz and that 16 has been chosen for the dividing ratio of the pre-scaler, then the watchdog timer's period becomes approximately 52.4 ms.

The watchdog timer is initialized by writing to the watchdog timer start register (address 000E16) and when a watchdog timer interrupt request is generated. The prescaler is initialized only when the microcomputer is reset. After a reset is cancelled, the watchdog timer and prescaler are both stopped. The count is started by writing to the watchdog timer start register (address 000E16).

Figure 1.14.1 shows the block diagram of the watchdog timer. Figure 1.14.2 shows the watchdog timer-related registers.

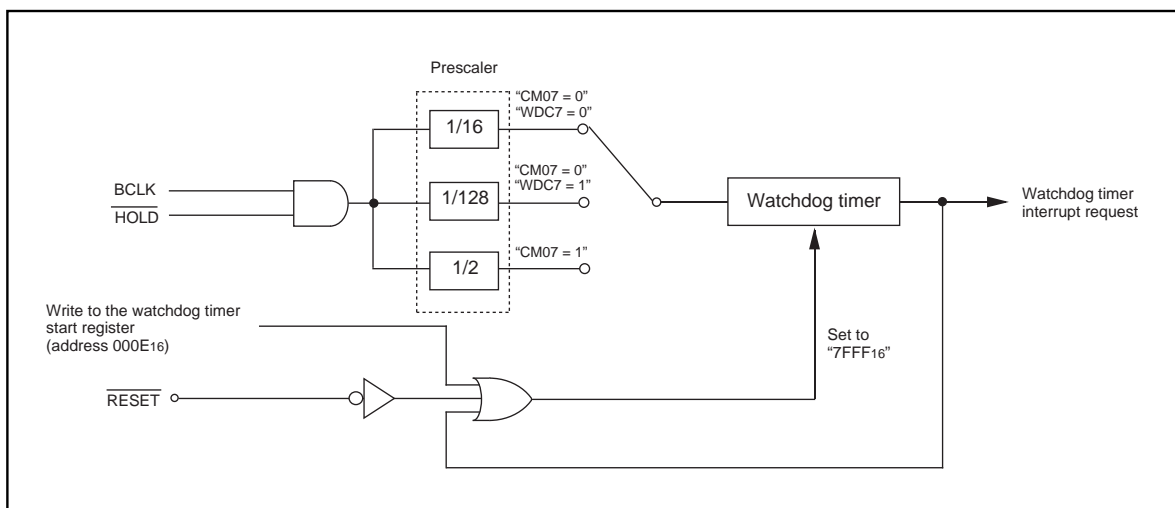


Figure 1.14.1. Block diagram of watchdog timer

Watchdog Timer

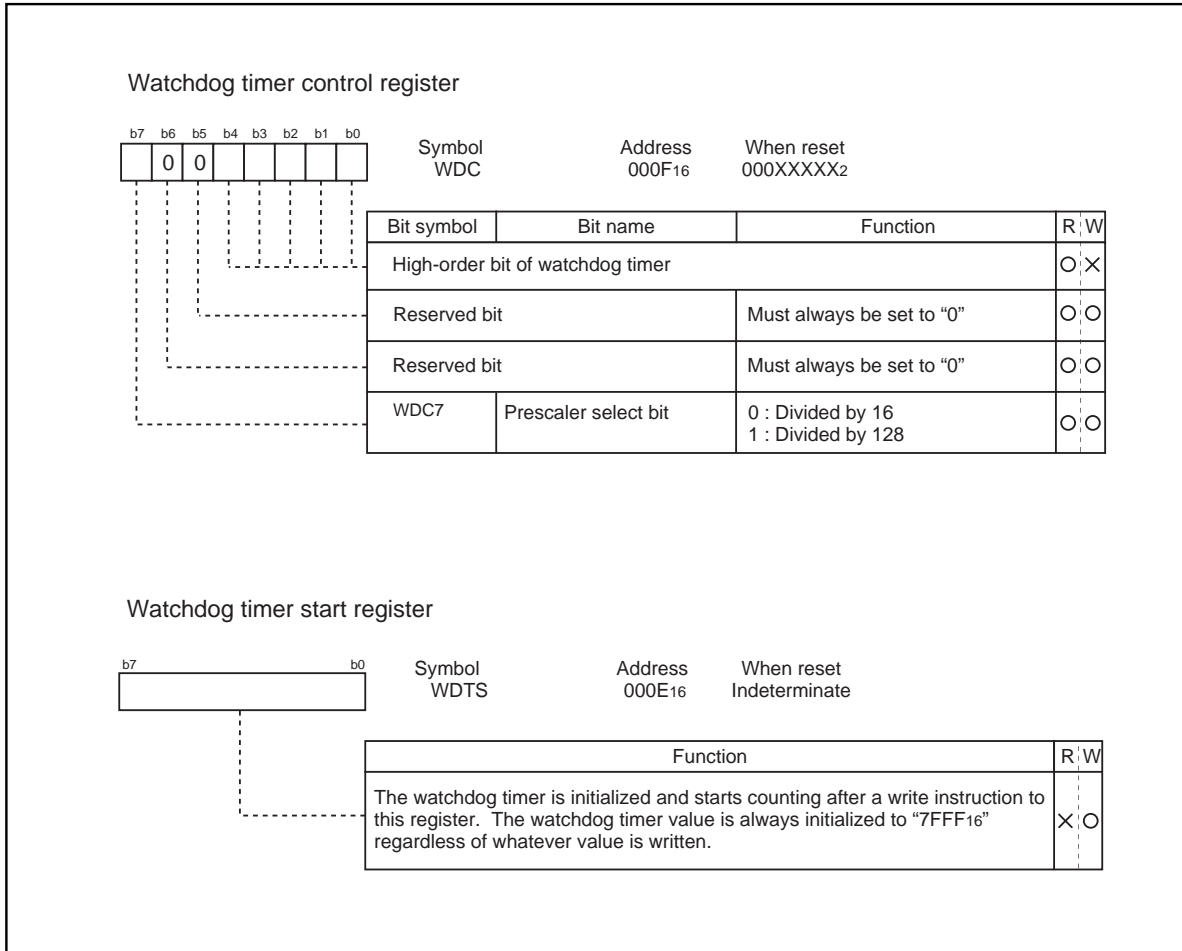
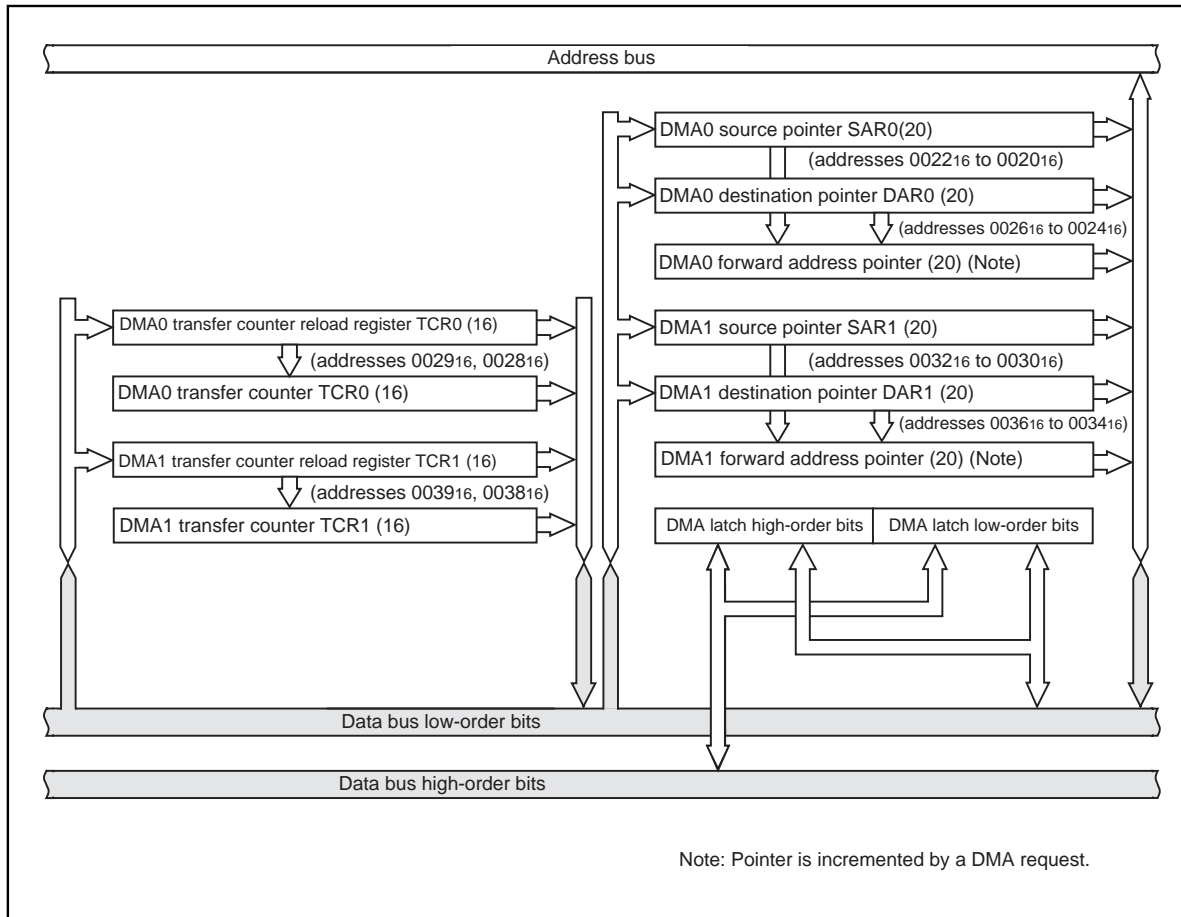


Figure 1.14.2. Watchdog timer control and start registers

**DMAC**

This microcomputer has two DMAC (direct memory access controller) channels that allow data to be sent to memory without using the CPU. DMAC shares the same data bus with the CPU. The DMAC is given a higher right of using the bus than the CPU, which leads to working the cycle stealing method. On this account, the operation from the occurrence of DMA transfer request signal to the completion of 1-word (16-bit) or 1-byte (8-bit) data transfer can be performed at high speed. Figure 1.15.1 shows the block diagram of the DMAC. Table 1.15.1 shows the DMAC specifications. Figure 1.15.2 to Figure 1.15.3 show the registers used by the DMAC.



**Figure 1.15.1. Block diagram of DMAC**

Either a write signal to the software DMA request bit or an interrupt request signal is used as a DMA transfer request signal. But the DMA transfer is affected neither by the interrupt enable flag (I flag) nor by the interrupt priority level. The DMA transfer doesn't affect any interrupts either.

If the DMAC is active (the DMA enable bit is set to 1), data transfer starts every time a DMA transfer request signal occurs. If the cycle of the occurrences of DMA transfer request signals is higher than the DMA transfer cycle, there can be instances in which the number of transfer requests doesn't agree with the number of transfers. For details, see the description of the DMA request bit.

**Table 1.15.1. DMAC specifications**

Item	Specification
No. of channels	2 (cycle steal method)
Transfer memory space	<ul style="list-style-type: none"> <li>• From any address in the 1M bytes space to a fixed address</li> <li>• From a fixed address to any address in the 1M bytes space</li> <li>• From a fixed address to a fixed address</li> </ul> (Note that DMA-related registers [0020 <sub>16</sub> to 003F <sub>16</sub> ] cannot be accessed)
Maximum No. of bytes transferred	128K bytes (with 16-bit transfers) or 64K bytes (with 8-bit transfers)
DMA request factors (Note)	Falling edge of INT0 or INT1 (INT0 can be selected by DMA0, INT1 by DMA1) Timer A0 to timer A4 interrupt requests Timer B0 to timer B2 interrupt requests UART0 transmission and reception interrupt requests UART1 transmission and reception interrupt requests (UART1 transmission can be selected by DMA0, UART1 reception by DMA1) UART2 transmission and reception interrupt requests A-D conversion interrupt requests Software triggers
Channel priority	DMA0 takes precedence if DMA0 and DMA1 requests are generated simultaneously
Transfer unit	8 bits or 16 bits
Transfer address direction	forward/fixed (forward direction cannot be specified for both source and destination simultaneously)
Transfer mode	<ul style="list-style-type: none"> <li>• Single transfer mode After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive</li> <li>• Repeat transfer mode After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter. The DMAC remains active unless a "0" is written to the DMA enable bit.</li> </ul>
DMA interrupt request generation timing	When an underflow occurs in the transfer counter
Active	When the DMA enable bit is set to "1", the DMAC is active. When the DMAC is active, data transfer starts every time a DMA transfer request signal occurs.
Inactive	<ul style="list-style-type: none"> <li>• When the DMA enable bit is set to "0", the DMAC is inactive.</li> <li>• After the transfer counter underflows in single transfer mode</li> </ul>
Forward address pointer and load timing for transfer counter	At the time of starting data transfer immediately after turning the DMAC active, the value of one of source pointer and destination pointer - the one specified for the forward direction - is reloaded to the forward direction address pointer, and the value of the transfer counter reload register is reloaded to the transfer counter.
Writing to register	Registers specified for forward direction transfer are always write enabled. Registers specified for fixed address transfer are write-enabled when the DMA enable bit is "0".
Reading the register	Can be read at any time. However, when the DMA enable bit is "1", reading the register set up as the forward register is the same as reading the value of the forward address pointer.

Note: DMA transfer is not effective to any interrupt. DMA transfer is affected neither by the interrupt enable flag (I flag) nor by the interrupt priority level.

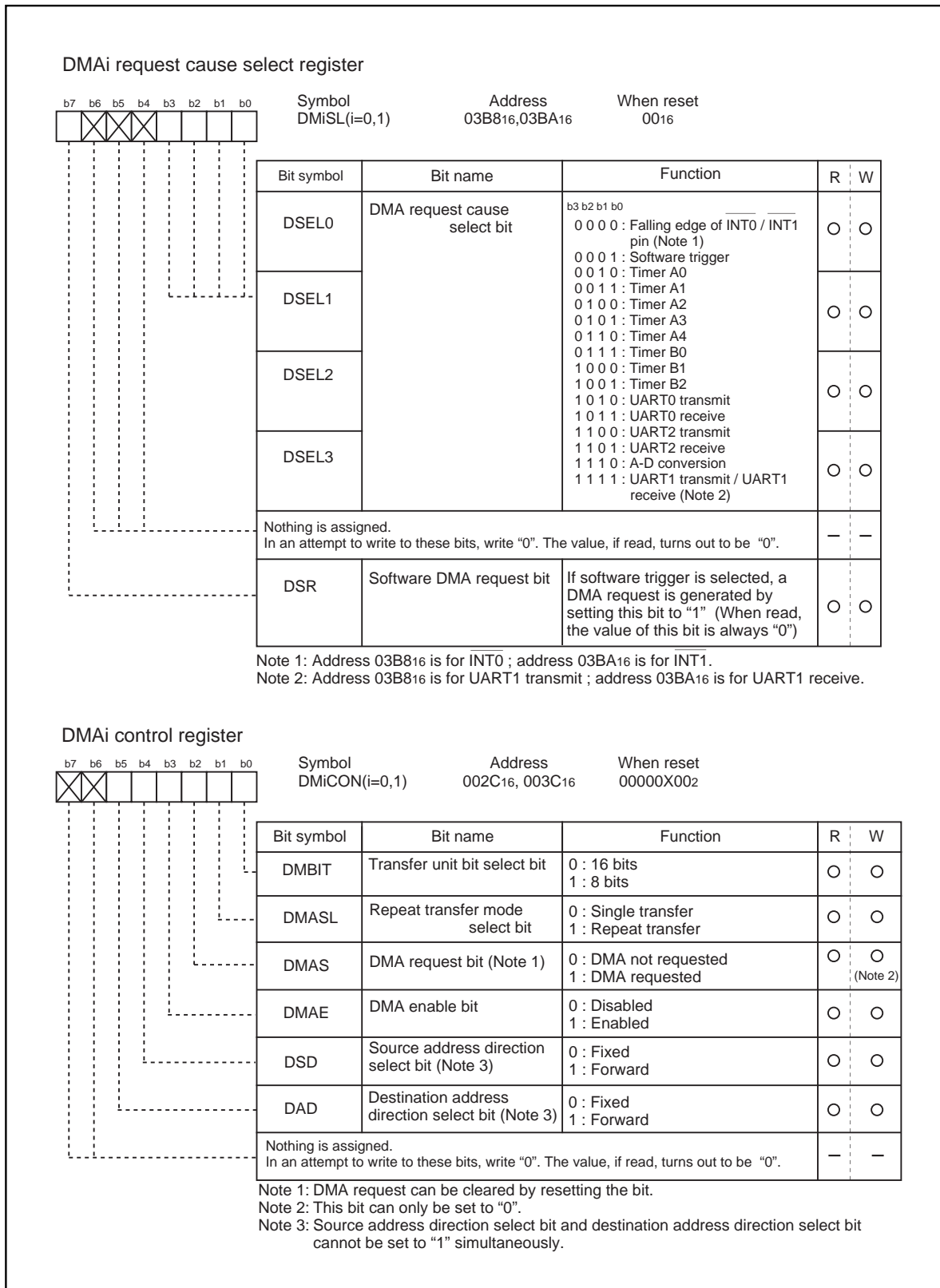


Figure 1.15.2. DMAC register (1)

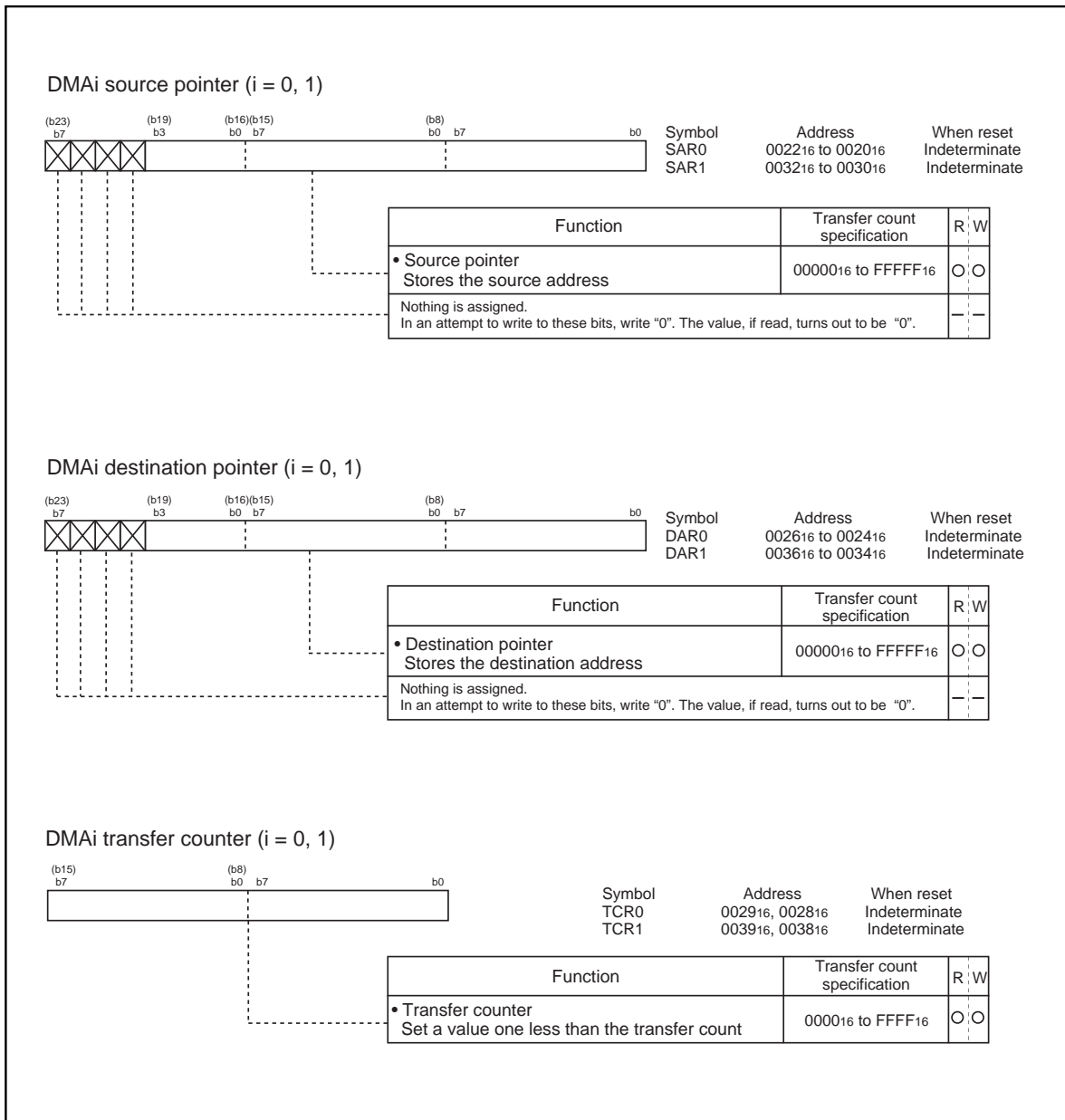


Figure 1.15.3. DMAC register (2)



## (1) Transfer cycle

The transfer cycle consists of the bus cycle in which data is read from memory or from the SFR area (source read) and the bus cycle in which the data is written to memory or to the SFR area (destination write). The number of read and write bus cycles depends on the source and destination addresses. In memory expansion mode and microprocessor mode, the number of read and write bus cycles also depends on the level of the BYTE pin. Also, the bus cycle itself is longer when software waits are inserted.

### (a) Effect of source and destination addresses

When 16-bit data is transferred on a 16-bit data bus, and the source and destination both start at odd addresses, there are one more source read cycle and destination write cycle than when the source and destination both start at even addresses.

### (b) Effect of BYTE pin level

When transferring 16-bit data over an 8-bit data bus (BYTE pin = "H") in memory expansion mode and microprocessor mode, the 16 bits of data are sent in two 8-bit blocks. Therefore, two bus cycles are required for reading the data and two are required for writing the data. Also, in contrast to when the CPU accesses internal memory, when the DMAC accesses internal memory (internal ROM, internal RAM, and SFR), these areas are accessed using the data size selected by the BYTE pin.

### (c) Effect of software wait

When the SFR area or a memory area with a software wait is accessed, the number of cycles is increased for the wait by 1 bus cycle. The length of the cycle is determined by BCLK.

Figure 1.15.4 shows the example of the transfer cycles for a source read. For convenience, the destination write cycle is shown as one cycle and the source read cycles for the different conditions are shown. In reality, the destination write cycle is subject to the same conditions as the source read cycle, with the transfer cycle changing accordingly. When calculating the transfer cycle, remember to apply the respective conditions to both the destination write cycle and the source read cycle. For example (2) in Figure 36, if data is being transferred in 16-bit units on an 8-bit bus, two bus cycles are required for both the source read cycle and the destination write cycle.

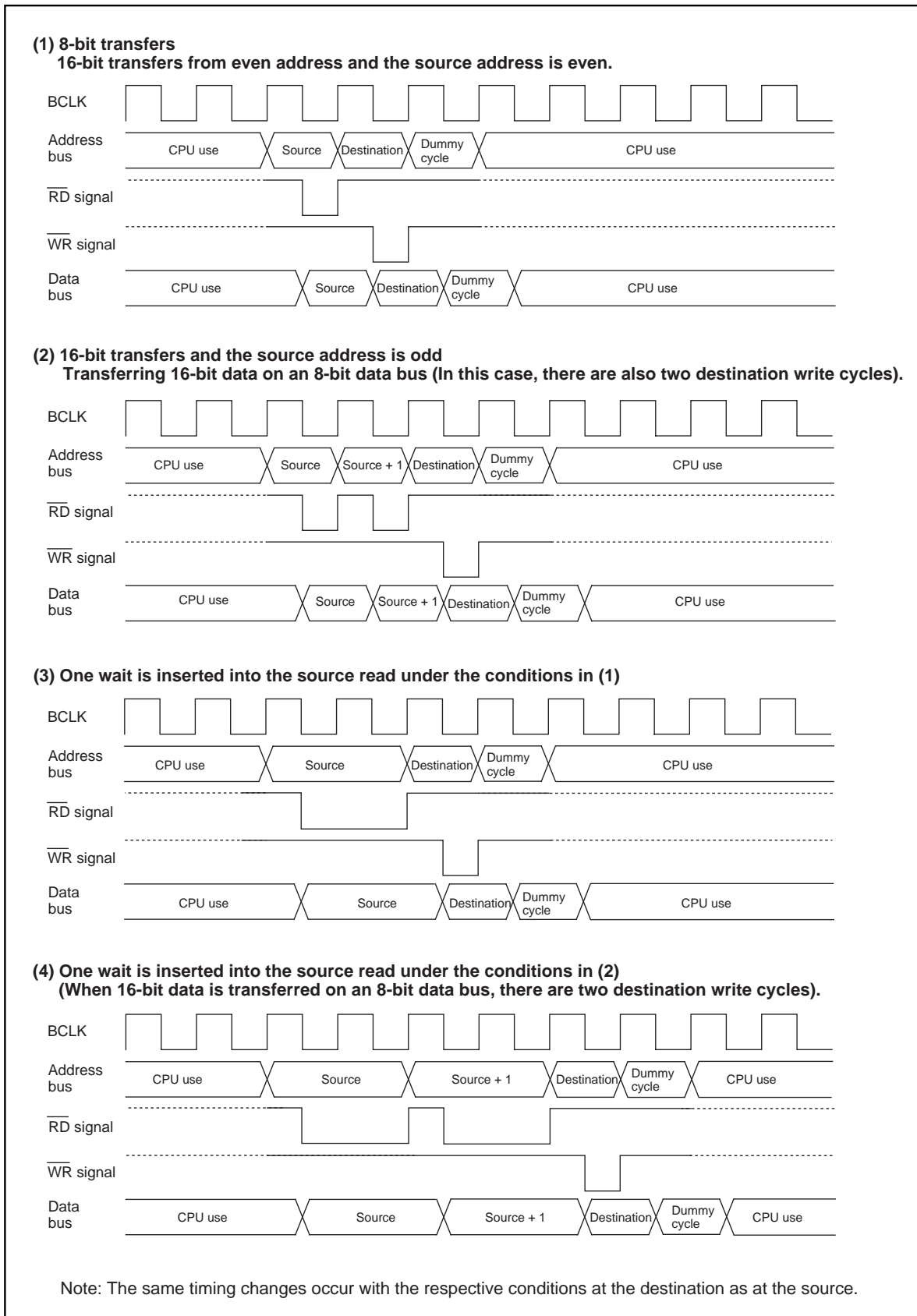


Figure 1.15.4. Example of the transfer cycles for a source read

**(2) DMAC transfer cycles**

Any combination of even or odd transfer read and write addresses is possible. Table 1.15.2 shows the number of DMAC transfer cycles.

The number of DMAC transfer cycles can be calculated as follows:

$$\text{No. of transfer cycles per transfer unit} = \text{No. of read cycles} \times j + \text{No. of write cycles} \times k$$

**Table 1.15.2. No. of DMAC transfer cycles**

Transfer unit	Bus width	Access address	Single-chip mode		Memory expansion mode Microprocessor mode	
			No. of read cycles	No. of write cycles	No. of read cycles	No. of write cycles
8-bit transfers (DMBIT="1")	16-bit (BYTE="L")	Even	1	1	1	1
		Odd	1	1	1	1
	8-bit (BYTE="H")	Even	—	—	1	1
		Odd	—	—	1	1
16-bit transfers (DMBIT="0")	16-bit (BYTE="L")	Even	1	1	1	1
		Odd	2	2	2	2
	8-bit (BYTE="H")	Even	—	—	2	2
		Odd	—	—	2	2

**Coefficient j, k**

Internal memory			External memory		
Internal ROM/RAM No wait	Internal ROM/RAM With wait	SFR area	Separate bus No wait	Separate bus With wait	Multiplex bus
1	2	2	1	2	3

### DMA enable bit

Setting the DMA enable bit to 1 makes the DMAC active. The DMAC carries out the following operations at the time data transfer starts immediately after DMAC is turned active.

- (1) Reloads the value of one of the source pointer and the destination pointer - the one specified for the forward direction - to the forward direction address pointer.
- (2) Reloads the value of the transfer counter reload register to the transfer counter.

Thus overwriting 1 to the DMA enable bit with the DMAC being active carries out the operations given above, so the DMAC operates again from the initial state at the instant 1 is overwritten to the DMA enable bit.

### DMA request bit

The DMAC can generate a DMA transfer request signal triggered by a factor chosen in advance out of DMA request factors for each channel.

DMA request factors include the following.

- \* Factors effected by using the interrupt request signals from the built-in peripheral functions and software DMA factors (internal factors) effected by a program.
- \* External factors effected by utilizing the input from external interrupt signals.

For the selection of DMA request factors, see the descriptions of the DMA<sub>i</sub> factor selection register.

The DMA request bit turns to 1 if the DMA transfer request signal occurs regardless of the DMAC's state (regardless of whether the DMA enable bit is set 1 or to 0). It turns to 0 immediately before data transfer starts.

In addition, it can be set to 0 by use of a program, but cannot be set to 1.

There can be instances in which a change in DMA request factor selection bit causes the DMA request bit to turn to 1. So be sure to set the DMA request bit to 0 after the DMA request factor selection bit is changed. The DMA request bit turns to 1 if a DMA transfer request signal occurs, and turns to 0 immediately before data transfer starts. If the DMAC is active, data transfer starts immediately, so the value of the DMA request bit, if read by use of a program, turns out to be 0 in most cases. To examine whether the DMAC is active, read the DMA enable bit.

Here follows the timing of changes in the DMA request bit.

#### (1) Internal factors

Except the DMA request factors triggered by software, the timing for the DMA request bit to turn to 1 due to an internal factor is the same as the timing for the interrupt request bit of the interrupt control register to turn to 1 due to several factors.

Turning the DMA request bit to 1 due to an internal factor is timed to be effected immediately before the transfer starts.

#### (2) External factors

An external factor is a factor caused to occur by the leading edge of input from the INT<sub>i</sub> pin (i depends on which DMAC channel is used).

Selecting the INT<sub>i</sub> pins as external factors using the DMA request factor selection bit causes input from these pins to become the DMA transfer request signals.

The timing for the DMA request bit to turn to 1 when an external factor is selected synchronizes with the signal's edge applicable to the function specified by the DMA request factor selection bit (synchronizes with the trailing edge of the input signal to each INT<sub>i</sub> pin, for example).

With an external factor selected, the DMA request bit is timed to turn to 0 immediately before data transfer starts similarly to the state in which an internal factor is selected.

**(3) The priorities of channels and DMA transfer timing**

If a DMA transfer request signal falls on a single sampling cycle (a sampling cycle means one period from the leading edge to the trailing edge of BCLK), the DMA request bits of applicable channels concurrently turn to 1. If the channels are active at that moment, DMA0 is given a high priority to start data transfer. When DMA0 finishes data transfer, it gives the bus right to the CPU. When the CPU finishes single bus access, then DMA1 starts data transfer and gives the bus right to the CPU.

An example in which DMA transfer is carried out in minimum cycles at the time when DMA transfer request signals due to external factors concurrently occur.

Figure 1.15.5 An example of DMA transfer effected by external factors.

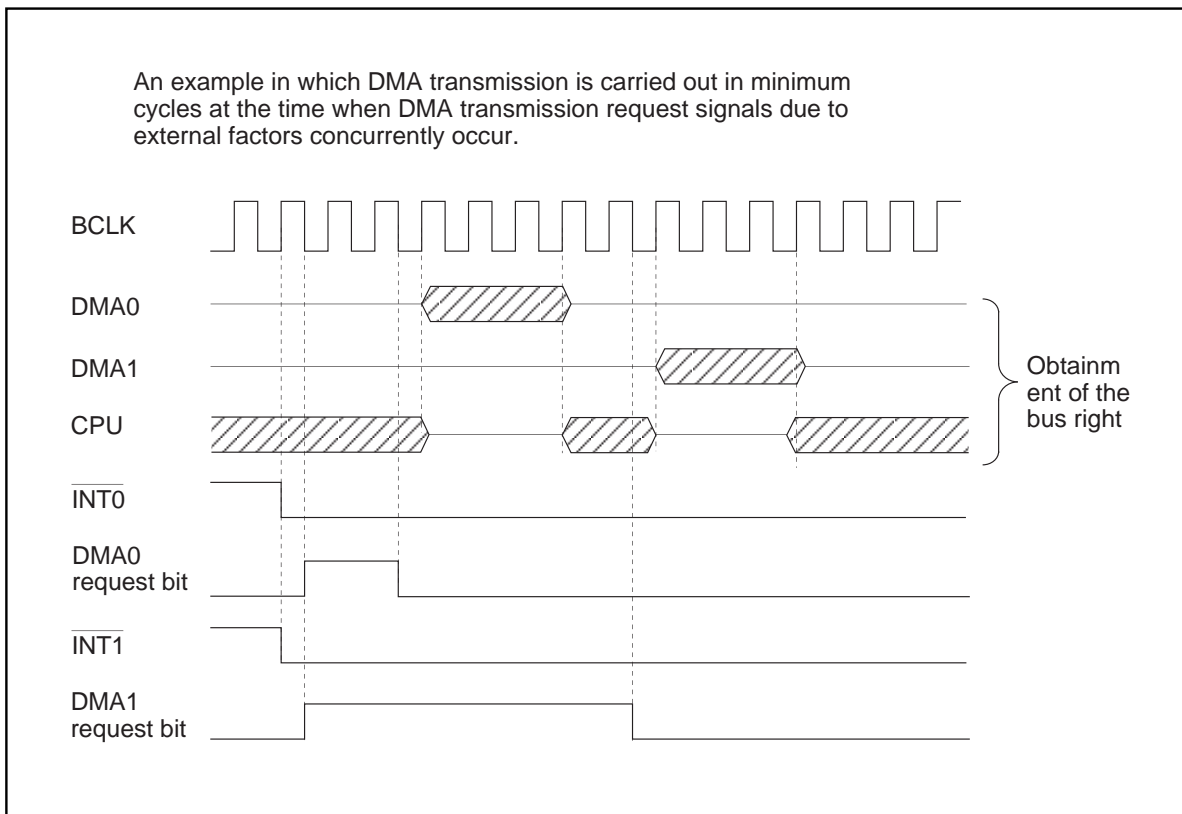


Figure 1.15.5. An example of DMA transfer effected by external factors

# Timer

## Timer

There are eight 16-bit timers. These timers can be classified by function into timers A (five) and timers B (three). All these timers function independently. Figure 1.16.1 shows the block diagram of timers.

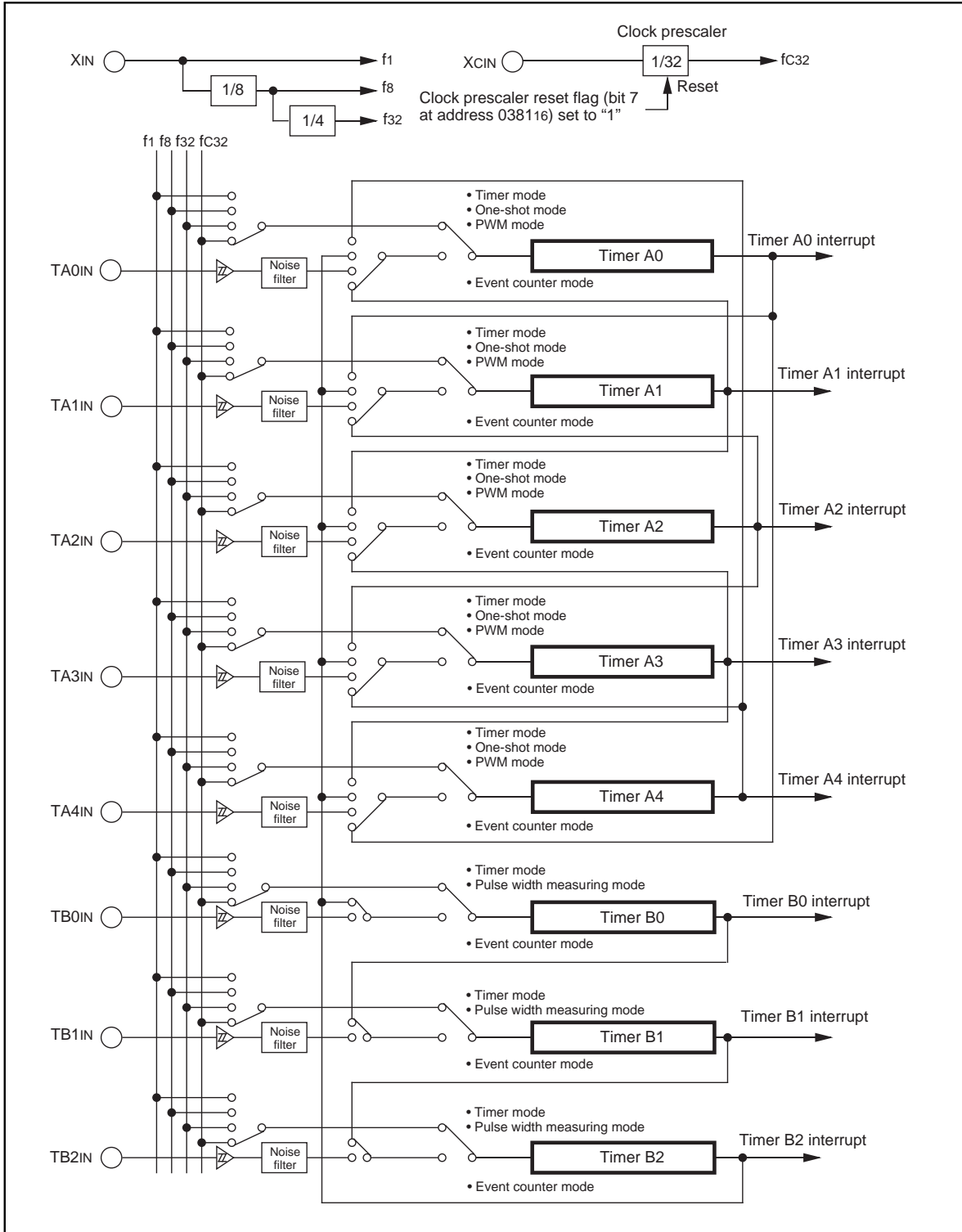


Figure 1.16.1. Block diagram of timer

## Timer A

## Timer A

Figure 1.16.2 shows the block diagram of timer A. Figures 1.16.3 to 1.16.5 show the timer A-related registers.

Except in event counter mode, timers A0 through A4 all have the same function. Use the timer Ai mode register (i = 0 to 4) bits 0 and 1 to choose the desired mode.

Timer A has the four operation modes listed as follows:

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external source or a timer over flow.
- One-shot timer mode: The timer stops counting when the count reaches "0000<sub>16</sub>".
- Pulse width modulation (PWM) mode: The timer outputs pulses of a given width.

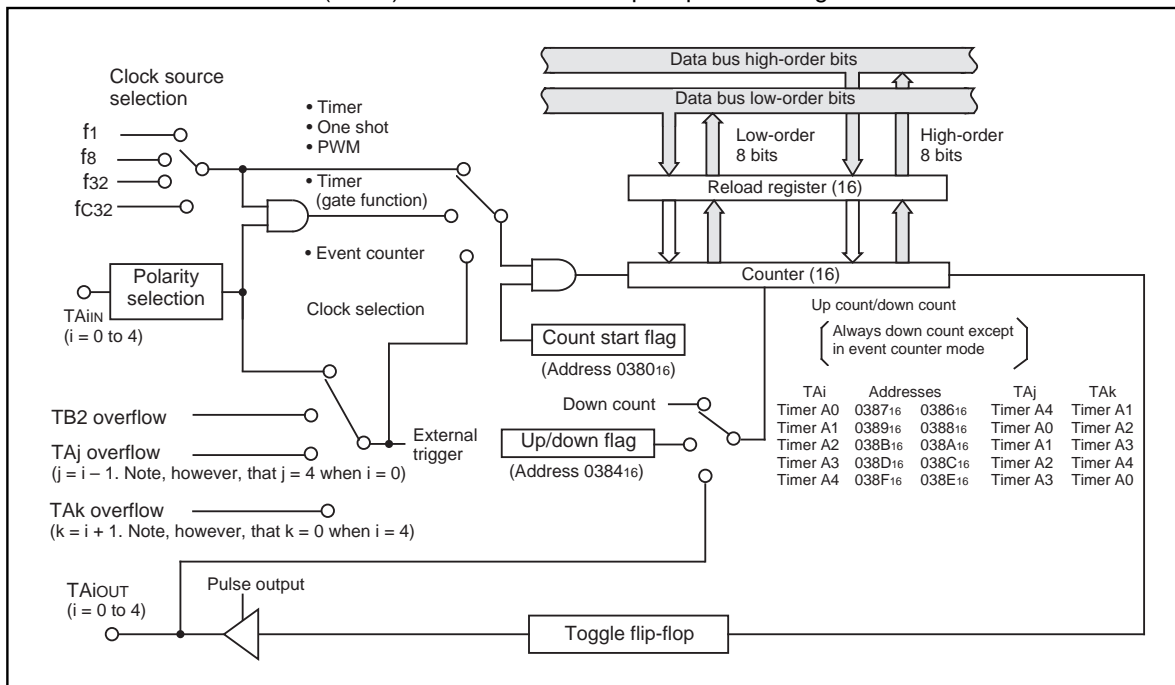


Figure 1.16.2. Block diagram of timer A

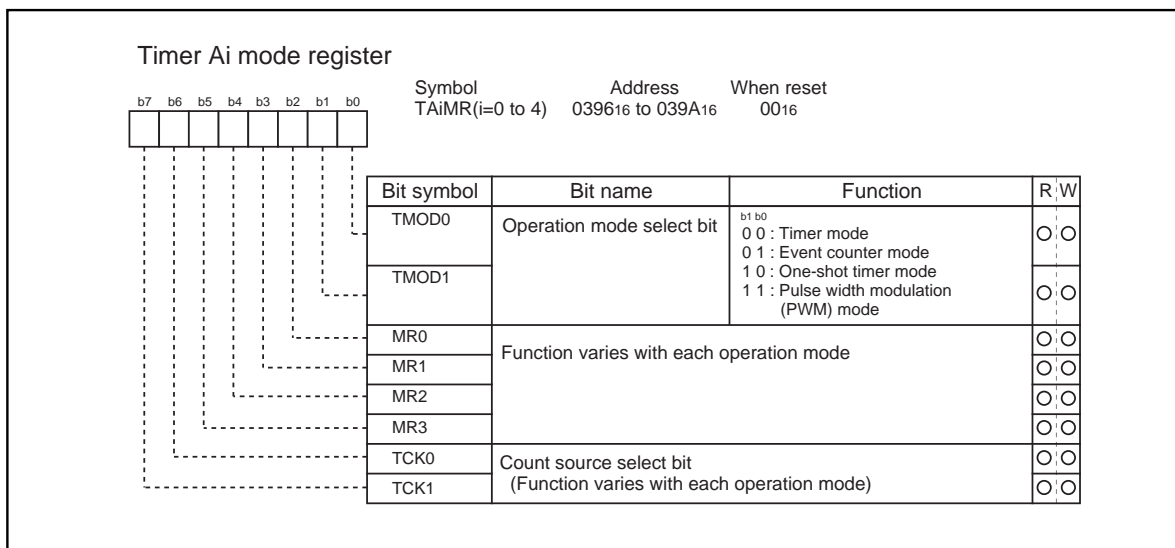


Figure 1.16.3. Timer A-related registers (1)

## Timer A

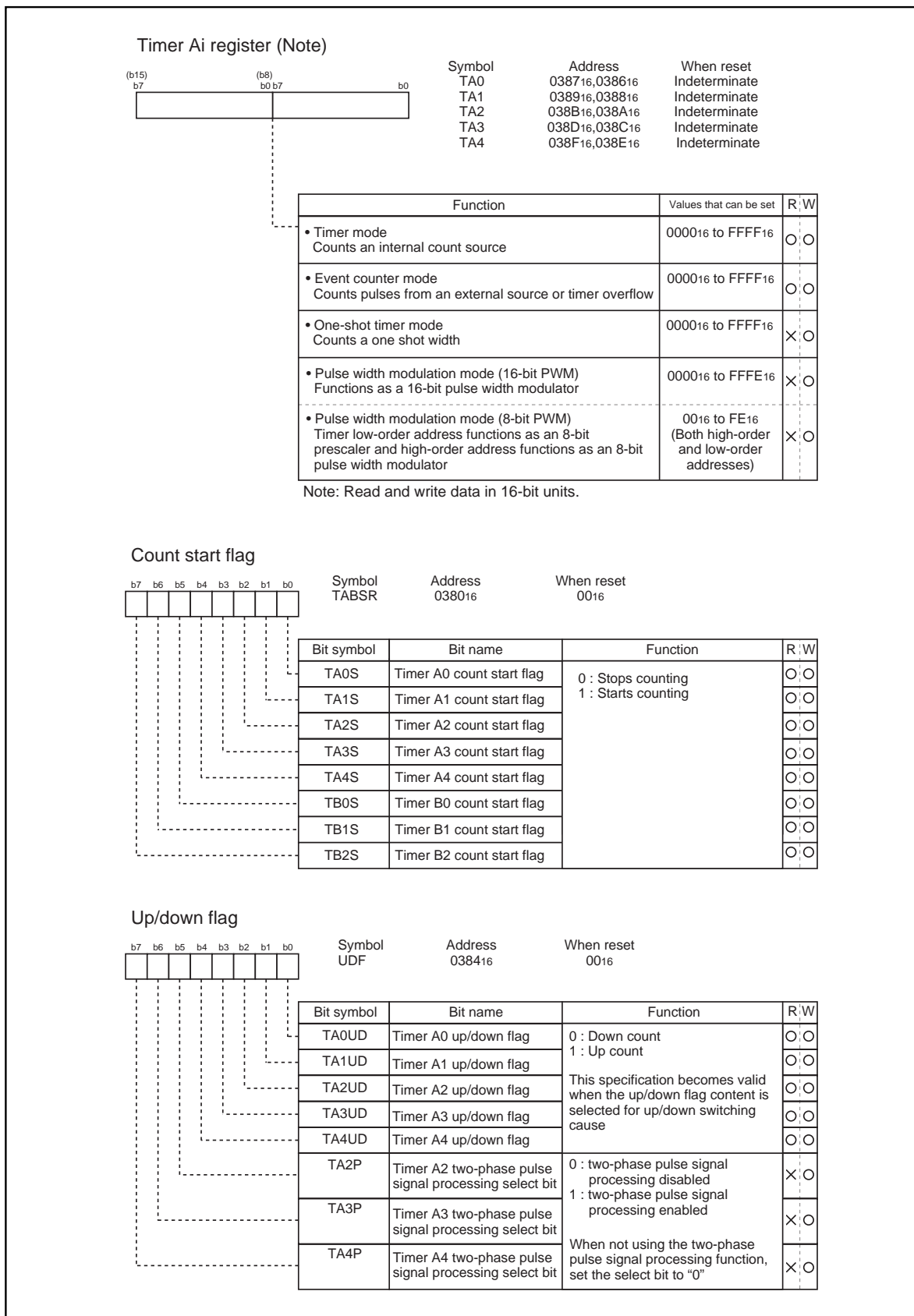


Figure 1.16.4. Timer A-related registers (2)



## Timer A

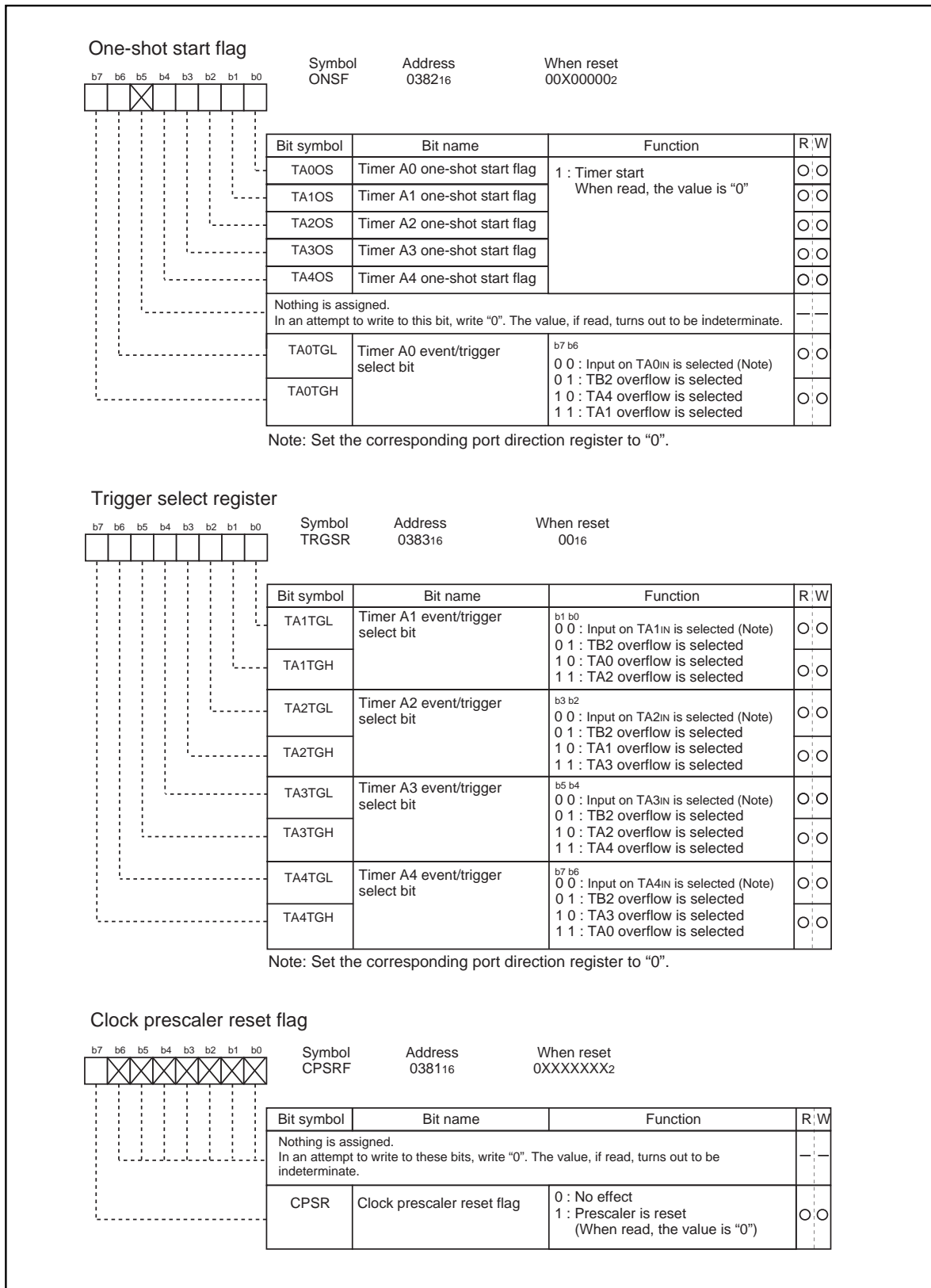


Figure 1.16.5. Timer A-related registers (3)

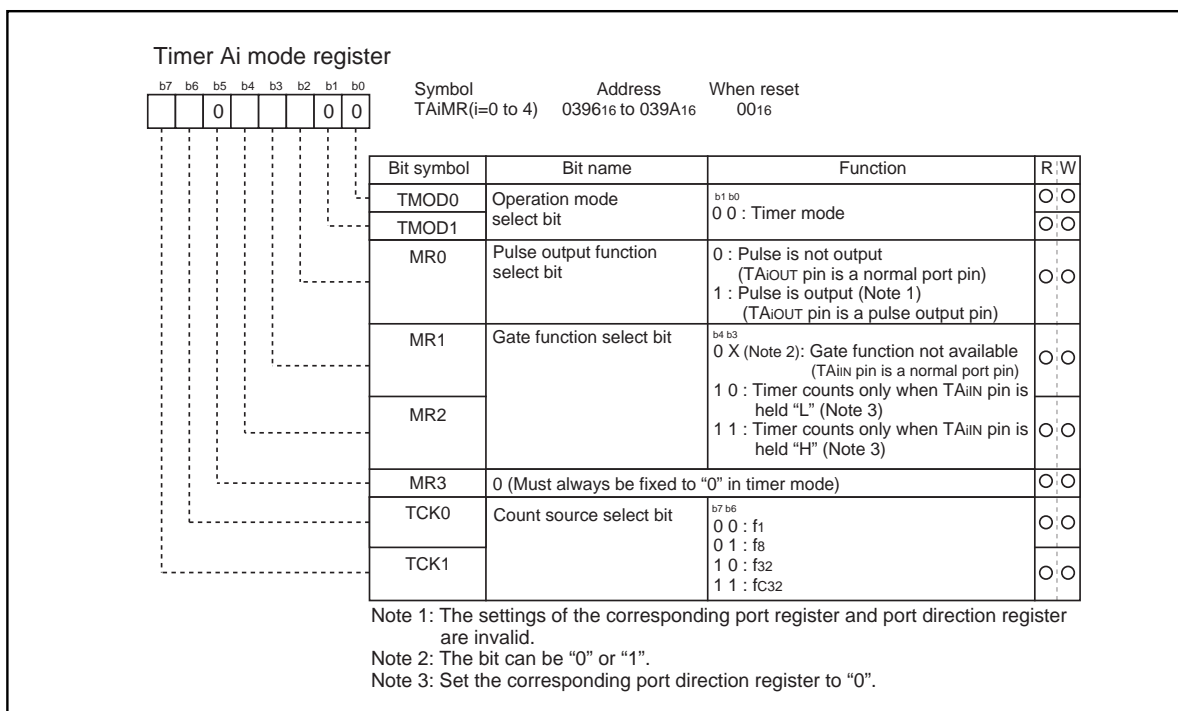
## Timer A

**(1) Timer mode**

In this mode, the timer counts an internally generated count source. (See Table 1.16.1.) Figure 1.16.6 shows the timer Ai mode register in timer mode.

**Table 1.16.1. Specifications of timer mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>Down count</li> <li>When the timer underflows, it reloads the reload register contents before continuing counting</li> </ul>
Divide ratio	1/(n+1) n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	When the timer underflows
TAiIN pin function	Programmable I/O port or gate input
TAiOUT pin function	Programmable I/O port or pulse output
Read from timer	Count value can be read out by reading timer Ai register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>Gate function Counting can be started and stopped by the TAiIN pin's input signal</li> <li>Pulse output function Each time the timer underflows, the TAiOUT pin's polarity is reversed</li> </ul>

**Figure 1.16.6. Timer Ai mode register in timer mode**

## Timer A

**(2) Event counter mode**

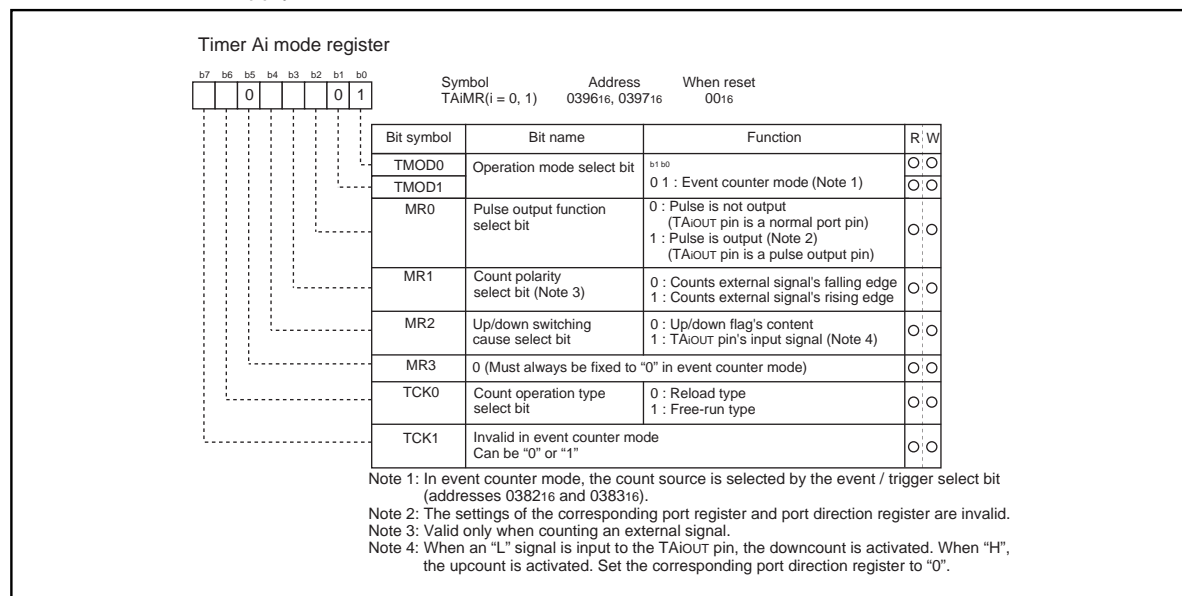
In this mode, the timer counts an external signal or an internal timer's overflow. Timers A0 and A1 can count a single-phase external signal. Timers A2, A3, and A4 can count a single-phase and a two-phase external signal. Table 1.16.2 lists timer specifications when counting a single-phase external signal. Figure 1.16.7 shows the timer Ai mode register in event counter mode.

Table 1.16.3 lists timer specifications when counting a two-phase external signal. Figure 1.16.8 shows the timer Ai mode register in event counter mode.

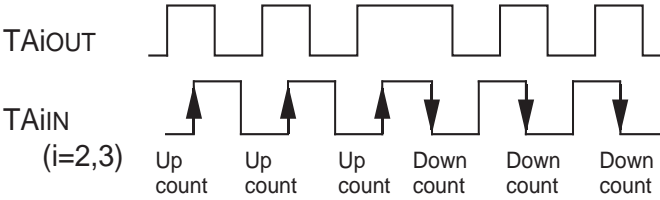
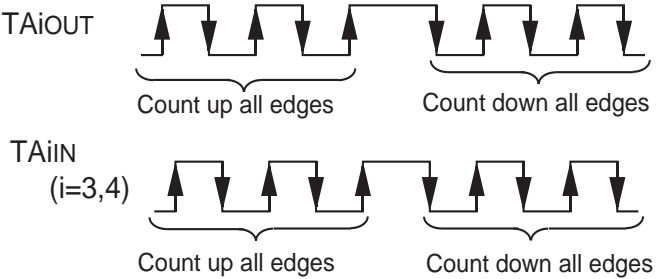
**Table 1.16.2. Timer specifications in event counter mode (when not processing two-phase pulse signal)**

Item	Specification
Count source	<ul style="list-style-type: none"> <li>External signals input to TAIiN pin (effective edge can be selected by software)</li> <li>TB2 overflow, TAJ overflow</li> </ul>
Count operation	<ul style="list-style-type: none"> <li>Up count or down count can be selected by external signal or software</li> <li>When the timer overflows or underflows, it reloads the reload register contents before continuing counting (Note)</li> </ul>
Divide ratio	$1 / (FFFF_{16} - n + 1)$ for up count $1 / (n + 1)$ for down count      n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer overflows or underflows
TAiIN pin function	Programmable I/O port or count source input
TAiOUT pin function	Programmable I/O port, pulse output, or up/down count select input
Read from timer	Count value can be read out by reading timer Ai register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>Free-run count function Even when the timer overflows or underflows, the reload register content is not reloaded to it</li> <li>Pulse output function Each time the timer overflows or underflows, the TAIOUT pin's polarity is reversed</li> </ul>

Note: This does not apply when the free-run function is selected.

**Figure 1.16.7. Timer Ai mode register in event counter mode**

**Table 1.16.3. Timer specifications in event counter mode (when processing two-phase pulse signal with timers A2, A3, and A4)**

Item	Specification
Count source	<ul style="list-style-type: none"> <li>Two-phase pulse signals input to TAIiN or TAIiOUT pin</li> </ul>
Count operation	<ul style="list-style-type: none"> <li>Up count or down count can be selected by two-phase pulse signal</li> <li>When the timer overflows or underflows, the reload register content is reloaded and the timer starts over again (Note)</li> </ul>
Divide ratio	$1 / (FFFF_{16} - n + 1)$ for up count $1 / (n + 1)$ for down count      n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	Timer overflows or underflows
TAiIN pin function	Two-phase pulse input
TAiOUT pin function	Two-phase pulse input
Read from timer	Count value can be read out by reading timer A2, A3, or A4 register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer A2, A3, or A4 register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer A2, A3, or A4 register, it is written to only reload register. (Transferred to counter at next reload time.)</li> </ul>
Select function	<ul style="list-style-type: none"> <li>Normal processing operation The timer counts up rising edges or counts down falling edges on the TAIiN pin when input signal on the TAIiOUT pin is "H"</li> </ul>  <ul style="list-style-type: none"> <li>Multiply-by-4 processing operation If the phase relationship is such that the TAIiN pin goes "H" when the input signal on the TAIiOUT pin is "H", the timer counts up rising and falling edges on the TAIiOUT and TAIiN pins. If the phase relationship is such that the TAIiN pin goes "L" when the input signal on the TAIiOUT pin is "H", the timer counts down rising and falling edges on the TAIiOUT and TAIiN pins.</li> </ul> 

Note: This does not apply when the free-run function is selected.

## Timer A

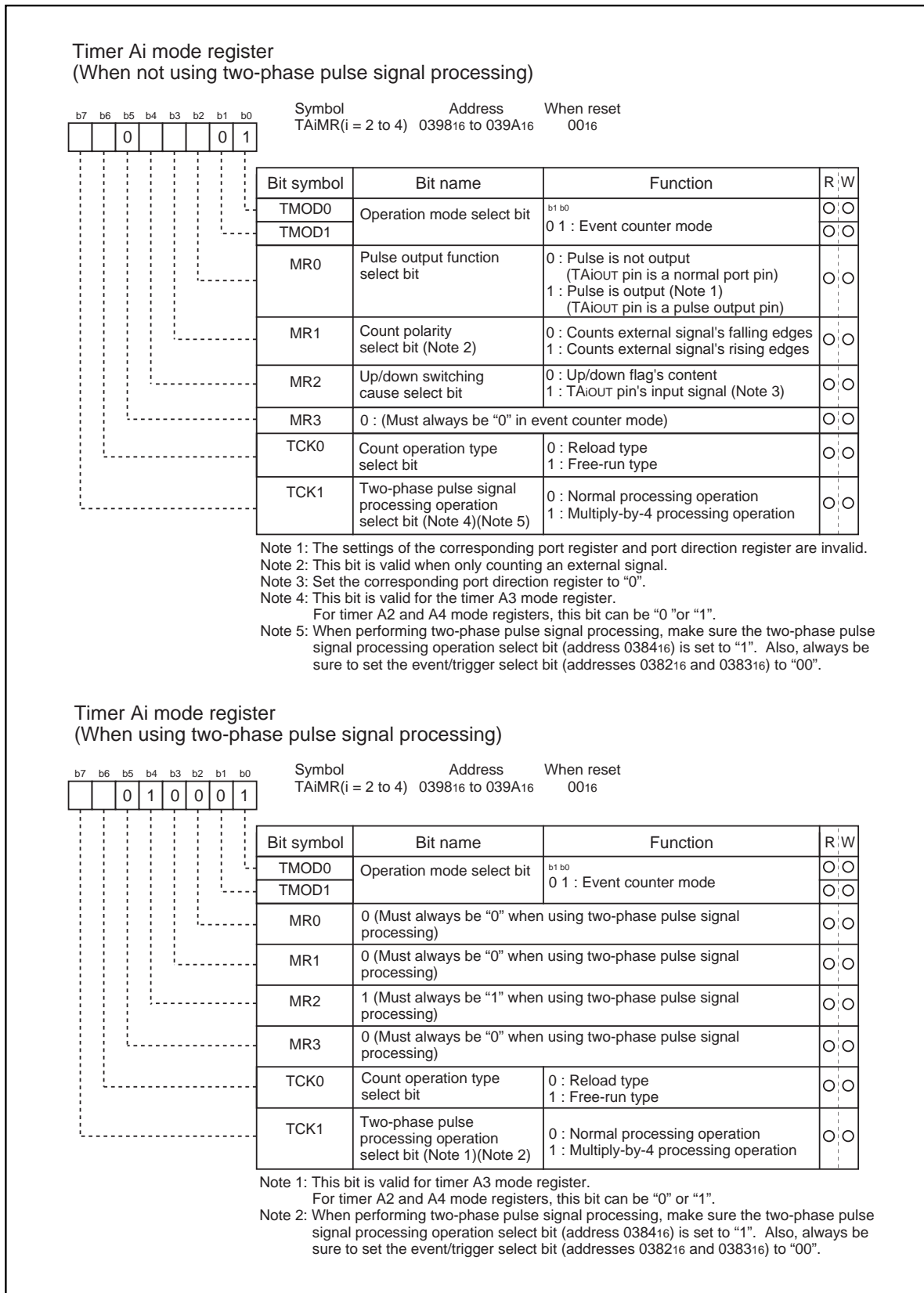


Figure 1.16.8. Timer Ai mode register in event counter mode

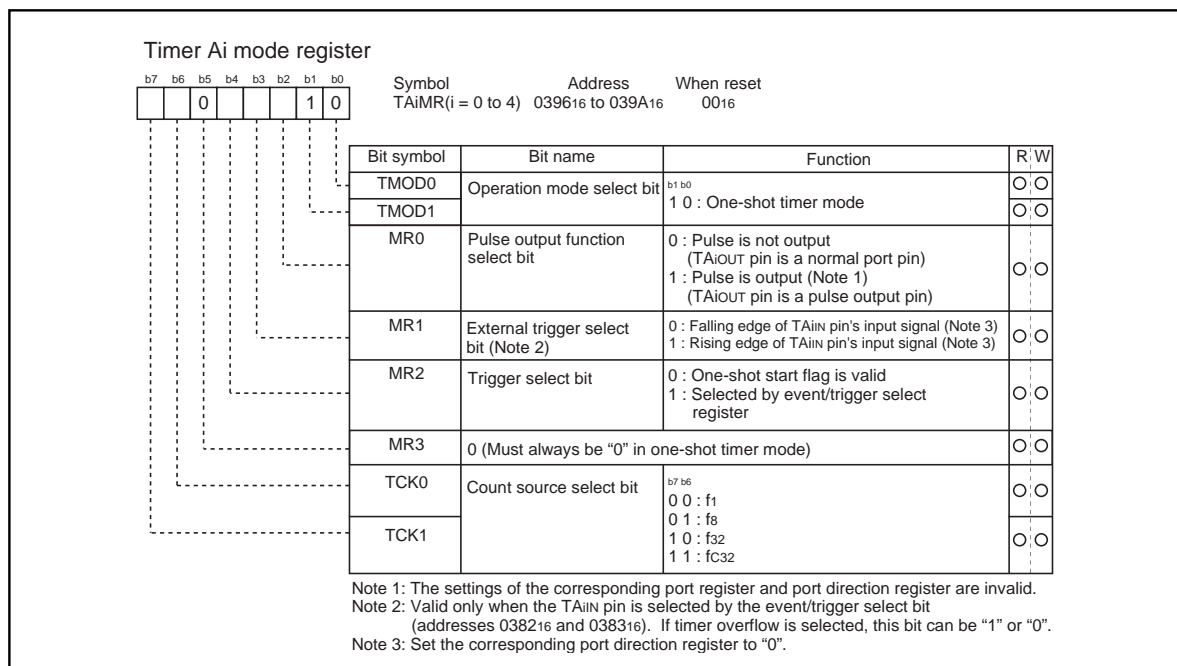
## Timer A

**(3) One-shot timer mode**

In this mode, the timer operates only once. (See Table 1.16.4.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 1.16.9 shows the timer Ai mode register in one-shot timer mode.

**Table 1.16.4. Timer specifications in one-shot timer mode**

Item	Specification
Count source	f1, f8, f32, fC32
Count operation	<ul style="list-style-type: none"> <li>The timer counts down</li> <li>When the count reaches 0000<sub>16</sub>, the timer stops counting after reloading a new count</li> <li>If a trigger occurs when counting, the timer reloads a new count and restarts counting</li> </ul>
Divide ratio	1/n    n : Set value
Count start condition	<ul style="list-style-type: none"> <li>An external trigger is input</li> <li>The timer overflows</li> <li>The one-shot start flag is set (= 1)</li> </ul>
Count stop condition	<ul style="list-style-type: none"> <li>A new count is reloaded after the count has reached 0000<sub>16</sub></li> <li>The count start flag is reset (= 0)</li> </ul>
Interrupt request generation timing	The count reaches 0000 <sub>16</sub>
TAiIN pin function	Programmable I/O port or trigger input
TAiOUT pin function	Programmable I/O port or pulse output
Read from timer	When timer Ai register is read, it indicates an indeterminate value
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>

**Figure 1.16.9. Timer Ai mode register in one-shot timer mode**

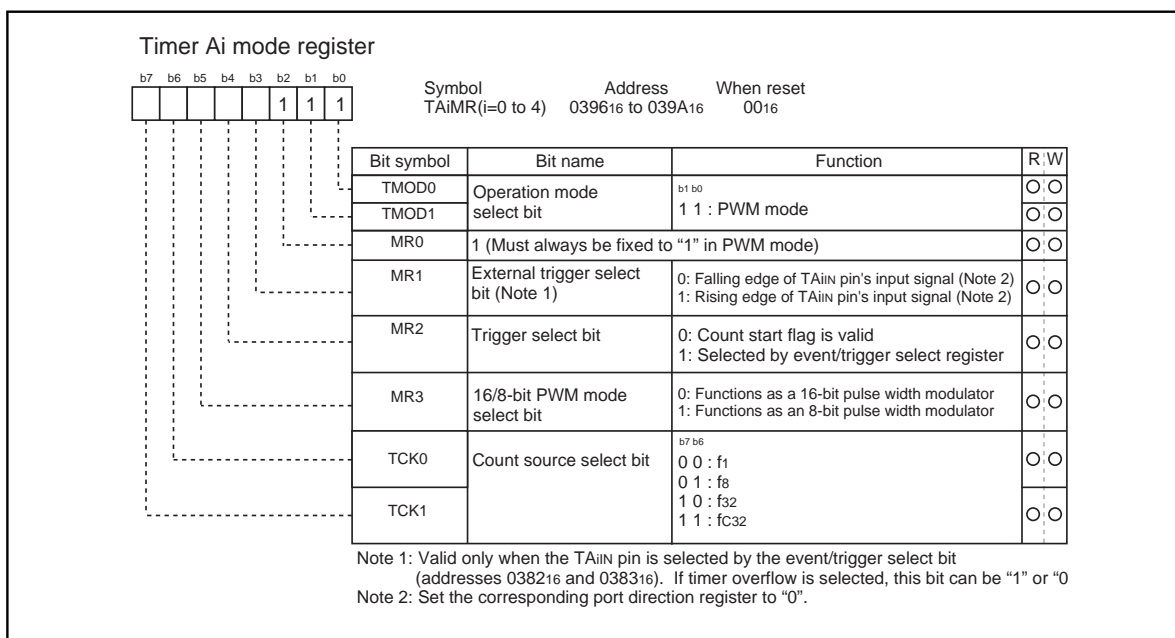
## Timer A

**(4) Pulse width modulation (PWM) mode**

In this mode, the timer outputs pulses of a given width in succession. (See Table 1.16.5.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 1.16.10 shows the configuration of the timer Ai mode register in pulse width modulation mode. Figure 1.16.11 shows an example of how a 16-bit pulse width modulator operates. Figure 1.16.12 shows an example of how an 8-bit pulse width modulator operates.

**Table 1.16.5. Timer specifications in pulse width modulation mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>The timer counts down (operating as an 8-bit or a 16-bit pulse width modulator)</li> <li>The timer reloads a new count at a rising edge of PWM pulse and continues counting</li> <li>The timer is not affected by a trigger that occurs when counting</li> </ul>
16-bit PWM	<ul style="list-style-type: none"> <li>High level width <math>n / f_i</math> : Set value</li> <li>Cycle time <math>(2^{16} - 1) / f_i</math> : fixed</li> </ul>
8-bit PWM	<ul style="list-style-type: none"> <li>High level width <math>n \times (m+1) / f_i</math> : values set to timer Ai register's high-order address</li> <li>Cycle time <math>(2^8 - 1) \times (m+1) / f_{im}</math> : values set to timer Ai register's low-order address</li> </ul>
Count start condition	<ul style="list-style-type: none"> <li>External trigger is input</li> <li>The timer overflows</li> <li>The count start flag is set (= 1)</li> </ul>
Count stop condition	<ul style="list-style-type: none"> <li>The count start flag is reset (= 0)</li> </ul>
Interrupt request generation timing	PWM pulse goes "L"
TAiIN pin function	Programmable I/O port or trigger input
TAiOUT pin function	Pulse output
Read from timer	When timer Ai register is read, it indicates an indeterminate value
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Ai register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>

**Figure 1.16.10. Configuration of timer Ai mode register in pulse width modulation mode**

Timer A

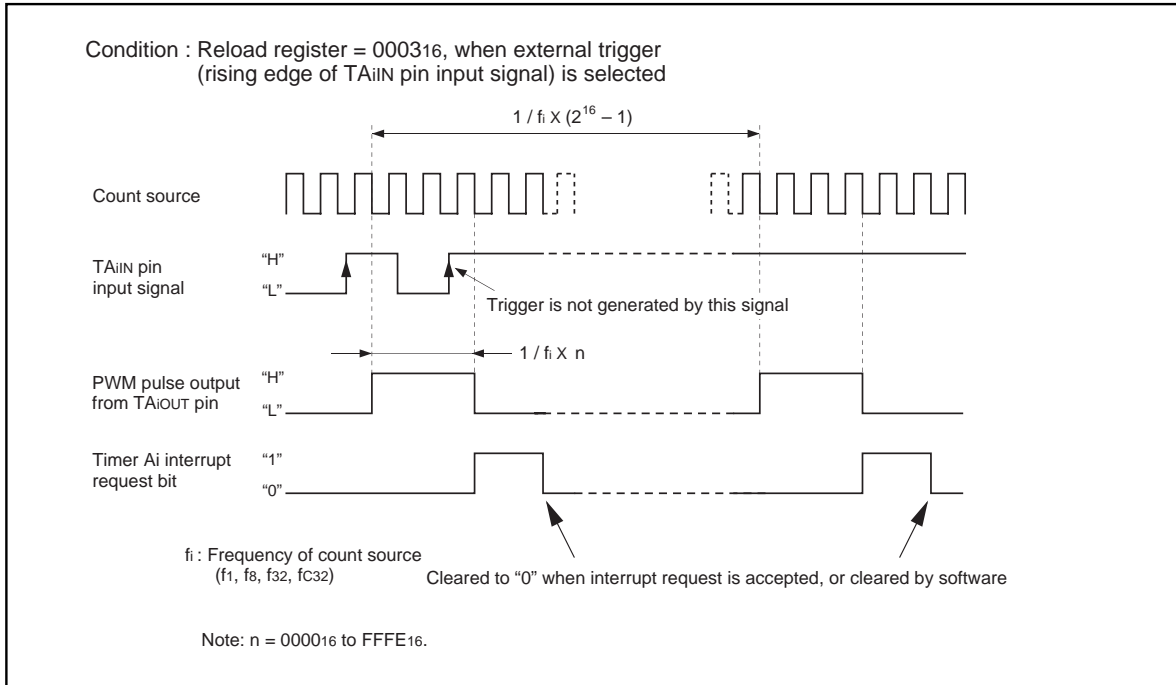


Figure 1.16.11. Example of how a 16-bit pulse width modulator operates

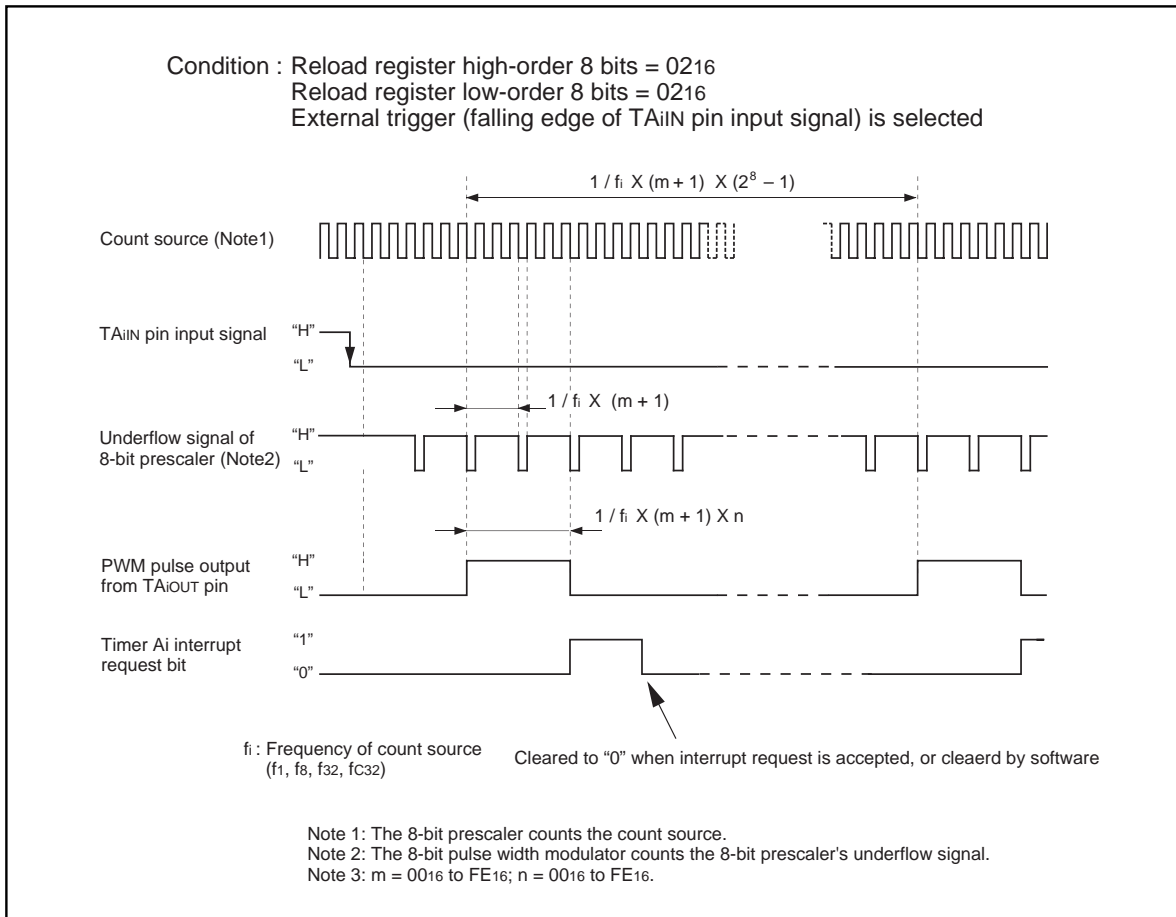


Figure 1.16.12. Example of how an 8-bit pulse width modulator operates



Timer B

Timer B

Figure 1.16.13 shows the block diagram of timer B. Figures 1.16.14 and 1.16.15 show the timer B-related registers.

Use the timer Bi mode register (i = 0 to 2) bits 0 and 1 to choose the desired mode.

Timer B has three operation modes listed as follows:

- Timer mode: The timer counts an internal count source.
- Event counter mode: The timer counts pulses from an external source or a timer overflow.
- Pulse period/pulse width measuring mode: The timer measures an external signal's pulse period or pulse width.

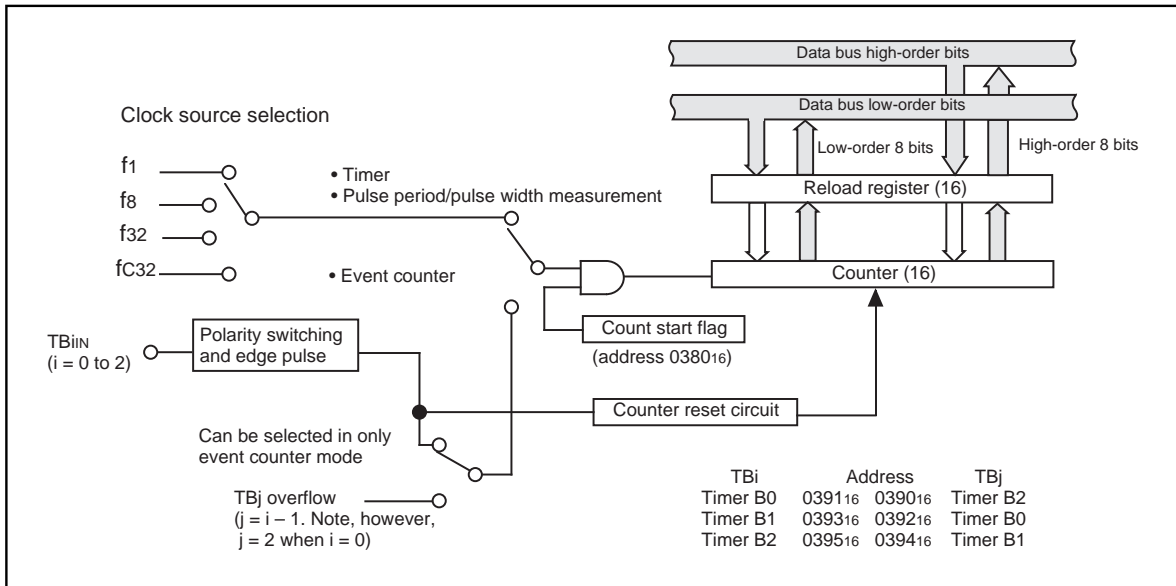


Figure 1.16.13. Block diagram of timer B

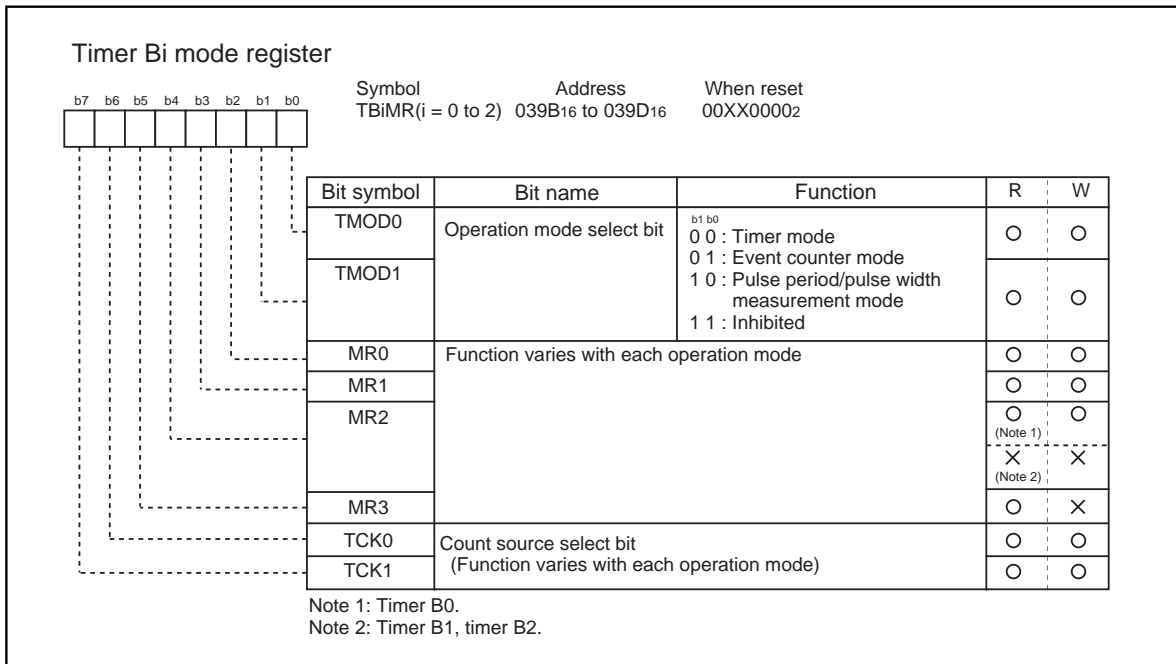


Figure 1.16.14. Timer B-related registers (1)

Timer B

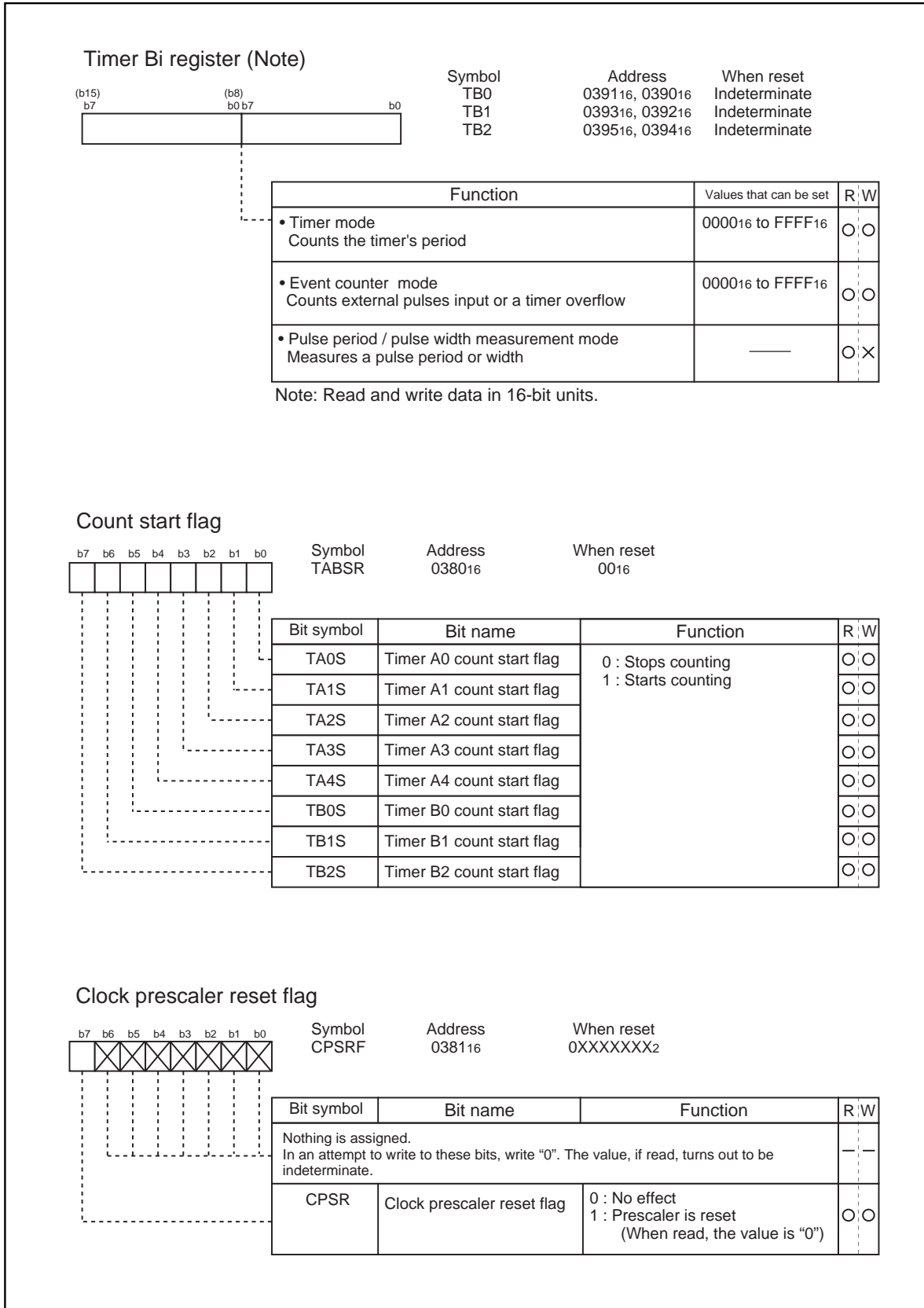


Figure 1.16.15. Timer B-related registers (2)

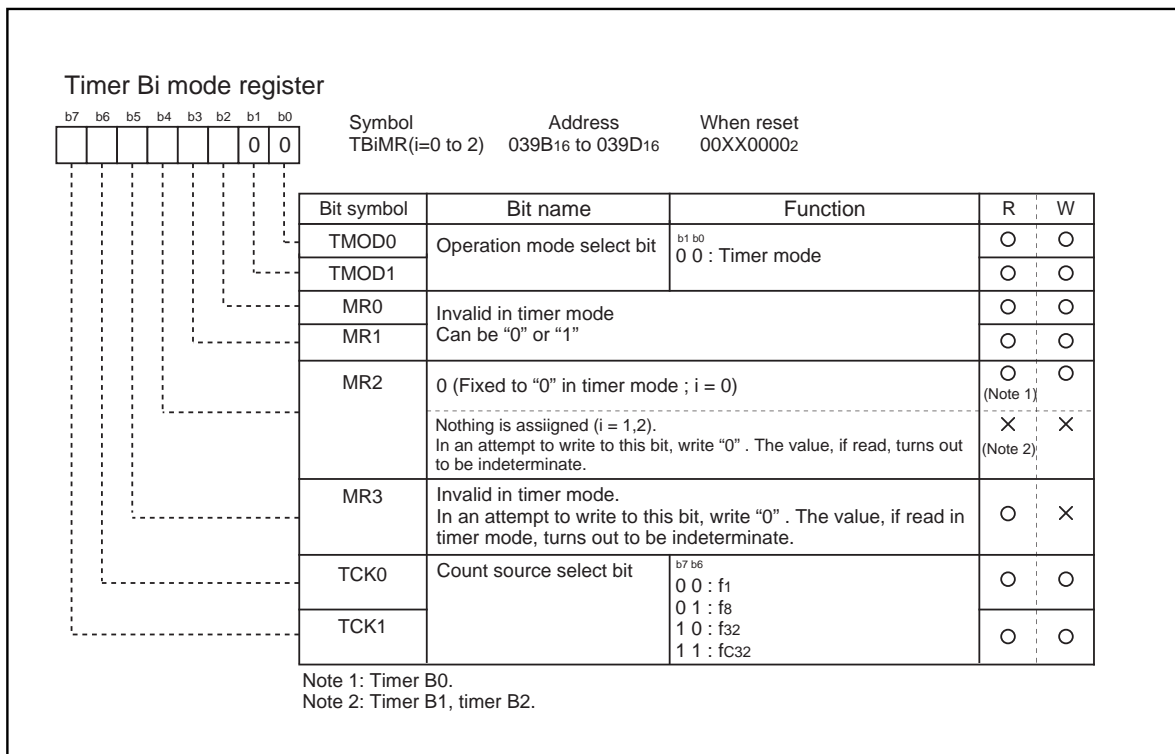
## Timer B

**(1) Timer mode**

In this mode, the timer counts an internally generated count source. (See Table 1.16.6.) Figure 1.16.16 shows the timer Bi mode register in timer mode.

**Table 1.16.6. Timer specifications in timer mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>Counts down</li> <li>When the timer underflows, it reloads the reload register contents before continuing counting</li> </ul>
Divide ratio	1/(n+1) n : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer underflows
TBiIN pin function	Programmable I/O port
Read from timer	Count value is read out by reading timer Bi register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>

**Figure 1.16.16. Timer Bi mode register in timer mode**

## Timer B

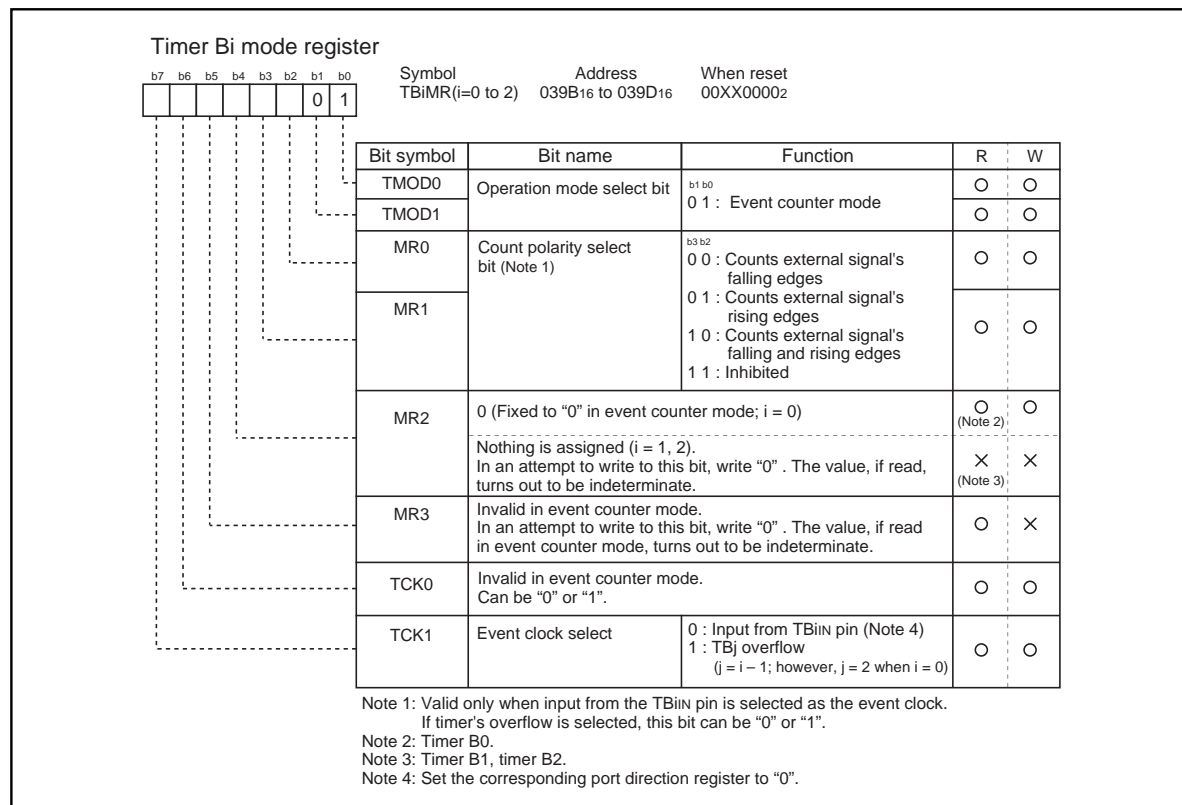
**(2) Event counter mode**

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 1.16.7.)

Figure 1.16.17 shows the timer Bi mode register in event counter mode.

**Table 1.16.7. Timer specifications in event counter mode**

Item	Specification
Count source	<ul style="list-style-type: none"> <li>External signals input to TBiIn pin</li> <li>Effective edge of count source can be a rising edge, a falling edge, or falling and rising edges as selected by software</li> </ul>
Count operation	<ul style="list-style-type: none"> <li>Counts down</li> <li>When the timer underflows, it reloads the reload register contents before continuing counting</li> </ul>
Divide ratio	$1/(n+1)$ $n$ : Set value
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	The timer underflows
TBiIn pin function	Count source input
Read from timer	Count value can be read out by reading timer Bi register
Write to timer	<ul style="list-style-type: none"> <li>When counting stopped When a value is written to timer Bi register, it is written to both reload register and counter</li> <li>When counting in progress When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time)</li> </ul>



**Figure 1.16.17. Timer Bi mode register in event counter mode**

## Timer B

**(3) Pulse period/pulse width measurement mode**

In this mode, the timer measures the pulse period or pulse width of an external signal. (See Table 1.16.8.)

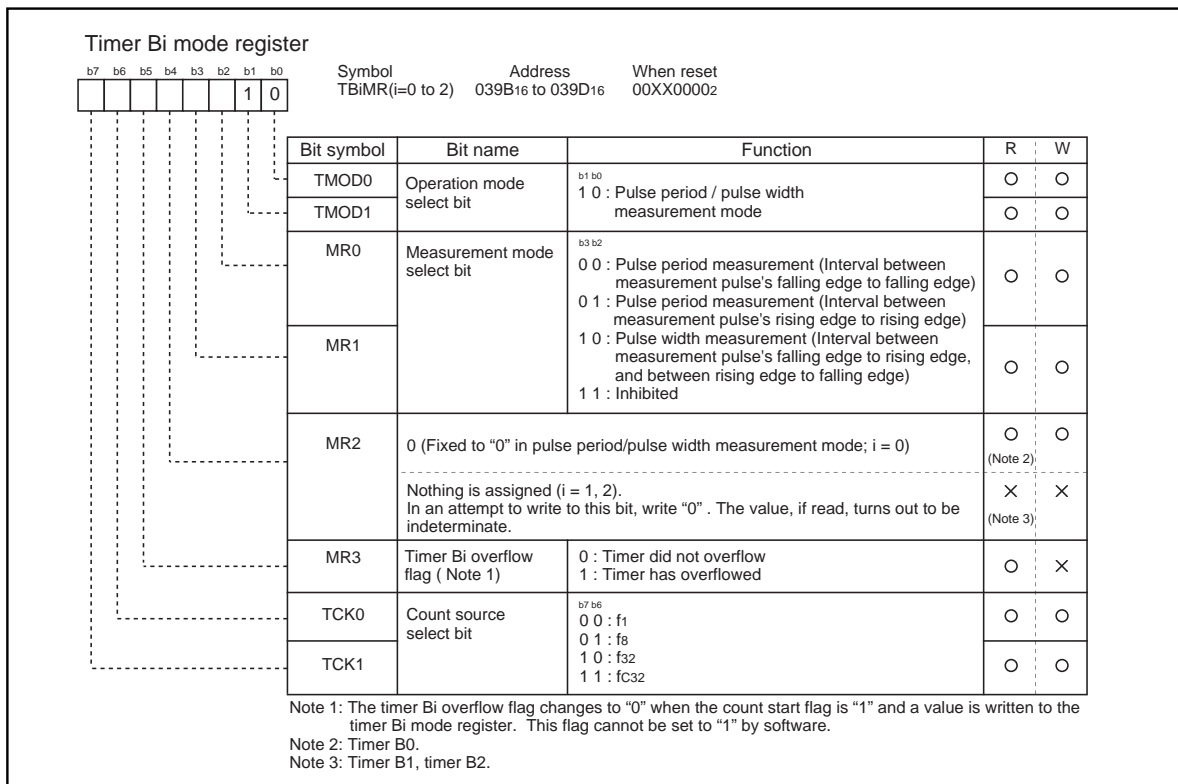
Figure 1.16.18 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 1.16.19 shows the operation timing when measuring a pulse period. Figure 1.16.20 shows the operation timing when measuring a pulse width.

**Table 1.16.8. Timer specifications in pulse period/pulse width measurement mode**

Item	Specification
Count source	f1, f8, f32, fc32
Count operation	<ul style="list-style-type: none"> <li>• Up count</li> <li>• Counter value "0000<sub>16</sub>" is transferred to reload register at measurement pulse's effective edge and the timer continues counting</li> </ul>
Count start condition	Count start flag is set (= 1)
Count stop condition	Count start flag is reset (= 0)
Interrupt request generation timing	<ul style="list-style-type: none"> <li>• When measurement pulse's effective edge is input (Note 1)</li> <li>• When an overflow occurs. (Simultaneously, the timer Bi overflow flag changes to "1". The timer Bi overflow flag changes to "0" when the count start flag is "1" and a value is written to the timer Bi mode register.)</li> </ul>
TBiIn pin function	Measurement pulse input
Read from timer	When timer Bi register is read, it indicates the reload register's content (measurement result) (Note 2)
Write to timer	Cannot be written to

Note 1: An interrupt request is not generated when the first effective edge is input after the timer has started counting.

Note 2: The value read out from the timer Bi register is indeterminate until the second effective edge is input after the timer.

**Figure 1.16.18. Timer Bi mode register in pulse period/pulse width measurement mode**

Timer B

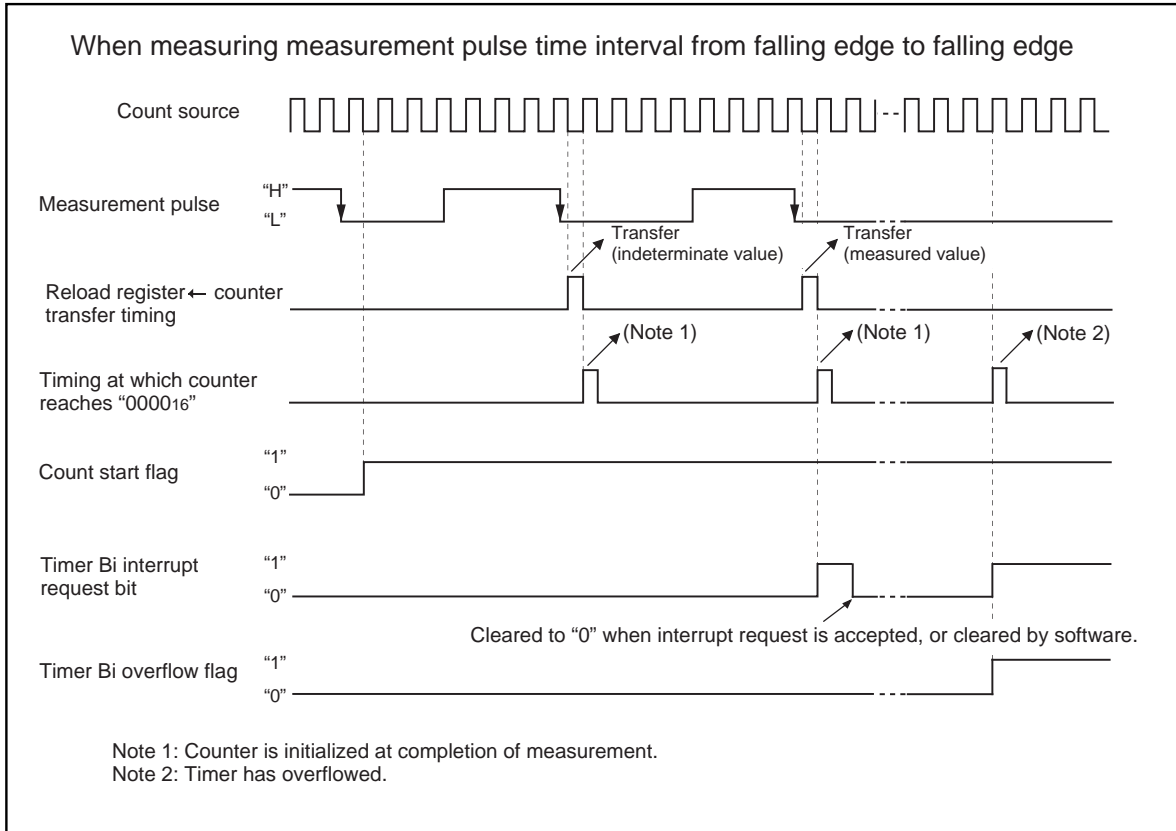


Figure 1.16.19. Operation timing when measuring a pulse period

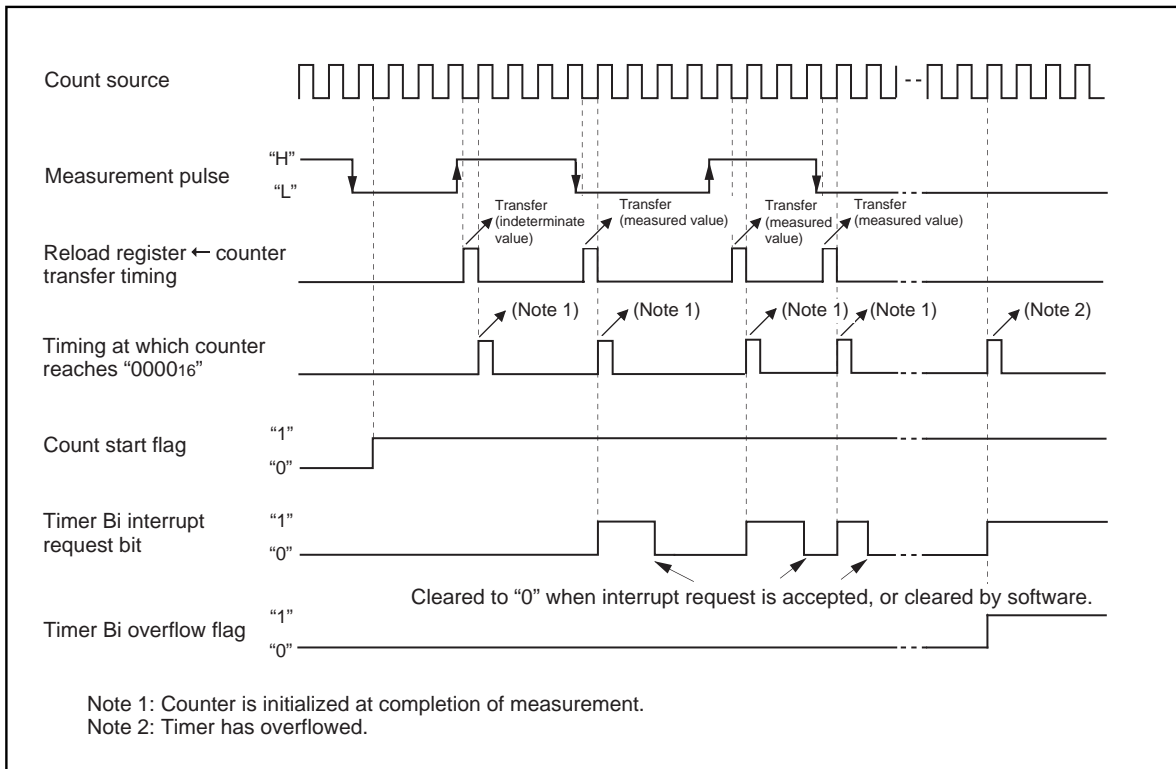


Figure 1.16.20. Operation timing when measuring a pulse width

## Serial I/O

Serial I/O is configured as three channels: UART0, UART1 and UART2. UART0, UART1 and UART2 each have an exclusive timer to generate a transfer clock, so they operate independently of each other.

Figure 1.17.1 shows the block diagram of UART0, UART1 and UART2. Figure 1.17.2 and figure 1.17.3 show the block diagram of the transmit/receive unit.

UARTi (i = 0 to 2) has two operation modes: a clock synchronous serial I/O mode and a clock asynchronous serial I/O mode (UART mode). The contents of the serial I/O mode select bits (bits 0 to 2 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub> and 0378<sub>16</sub>) determine whether UARTi is used as a clock synchronous serial I/O or as a UART.

UART0 through UART2 are almost equal in their functions with minor exceptions. UART2, in particular, is compliant with the SIM interface with some extra settings added in clock-asynchronous serial I/O mode (Note). It also has the bus collision detection function that generates an interrupt request if the TxD pin and the RxD pin are different in level.

Note: SIM : Subscriber Identity Module

Table 1.17.1 shows the comparison of functions of UART0 through UART2, and Figures 1.17.4 through 1.17.8 show the registers related to UARTi.

**Table 1.17.1. Comparison of functions of UART0 through UART2**

Function	UART0	UART1	UART2
CLK polarity selection	Possible (Note 1)	Possible (Note 1)	Possible (Note 1)
LSB first / MSB first selection	Possible (Note 1)	Possible (Note 1)	Possible (Note 2)
Continuous receive mode selection	Possible (Note 1)	Possible (Note 1)	Possible (Note 1)
Transfer clock output from multiple pins selection	Impossible	Possible (Note 1)	Impossible
Separate $\overline{\text{CTS}}/\overline{\text{RTS}}$ pins	Possible	Impossible	Impossible
Serial data logic switch	Impossible	Impossible	Possible (Note 4)
Sleep mode selection	Possible (Note 3)	Possible (Note 3)	Impossible
TxD, RxD I/O polarity switch	Impossible	Impossible	Possible
TxD, RxD port output format	CMOS output	CMOS output	N-channel open-drain output
Parity error signal output	Impossible	Impossible	Possible (Note 4)
Bus collision detection	Impossible	Impossible	Possible

Note 1: Only when clock synchronous serial I/O mode.

Note 2: Only when clock synchronous serial I/O mode and 8-bit UART mode.

Note 3: Only when UART mode.

Note 4: Using for SIM interface.

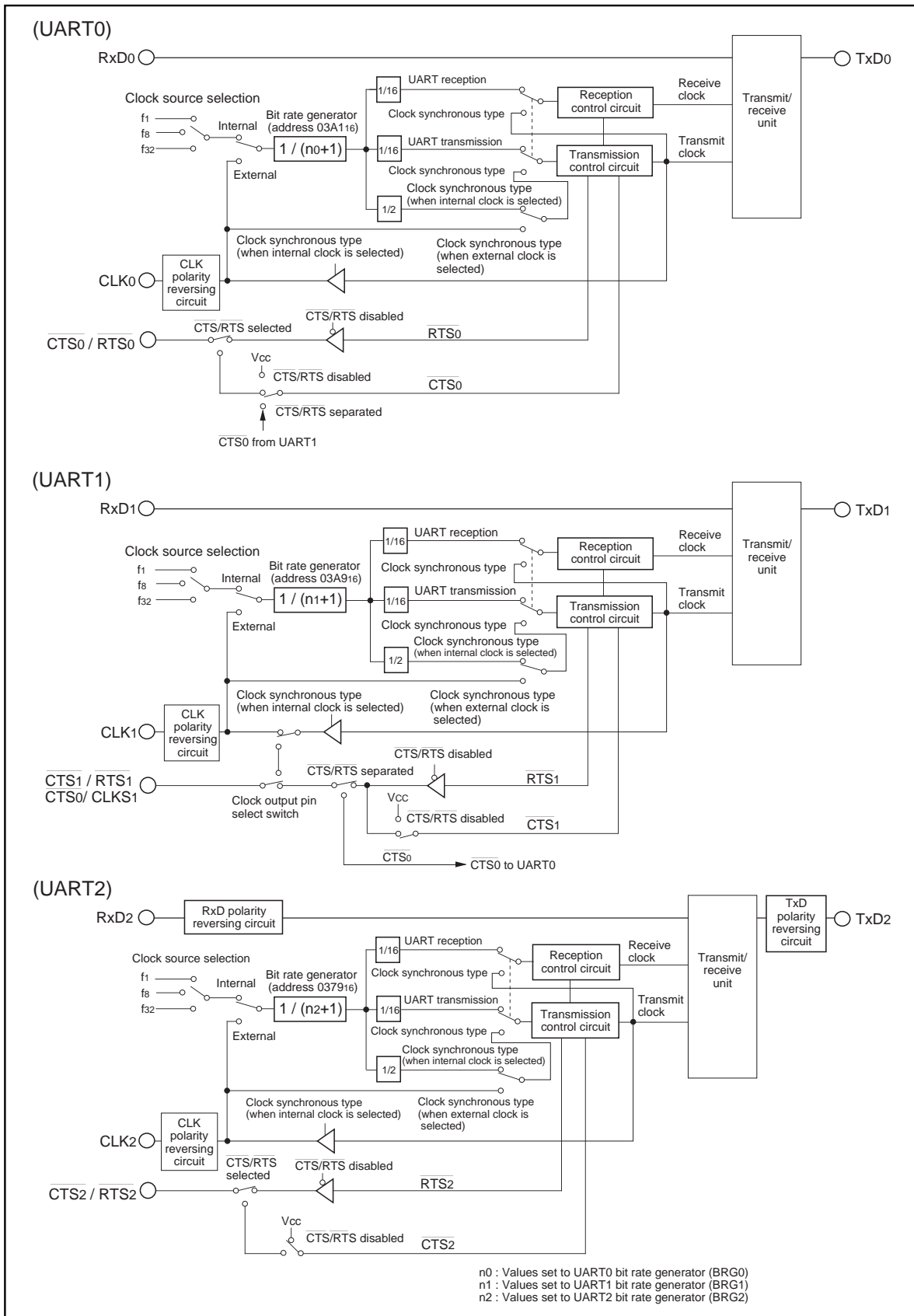


Figure 1.17.1. Block diagram of UARTi (i = 0 to 2)



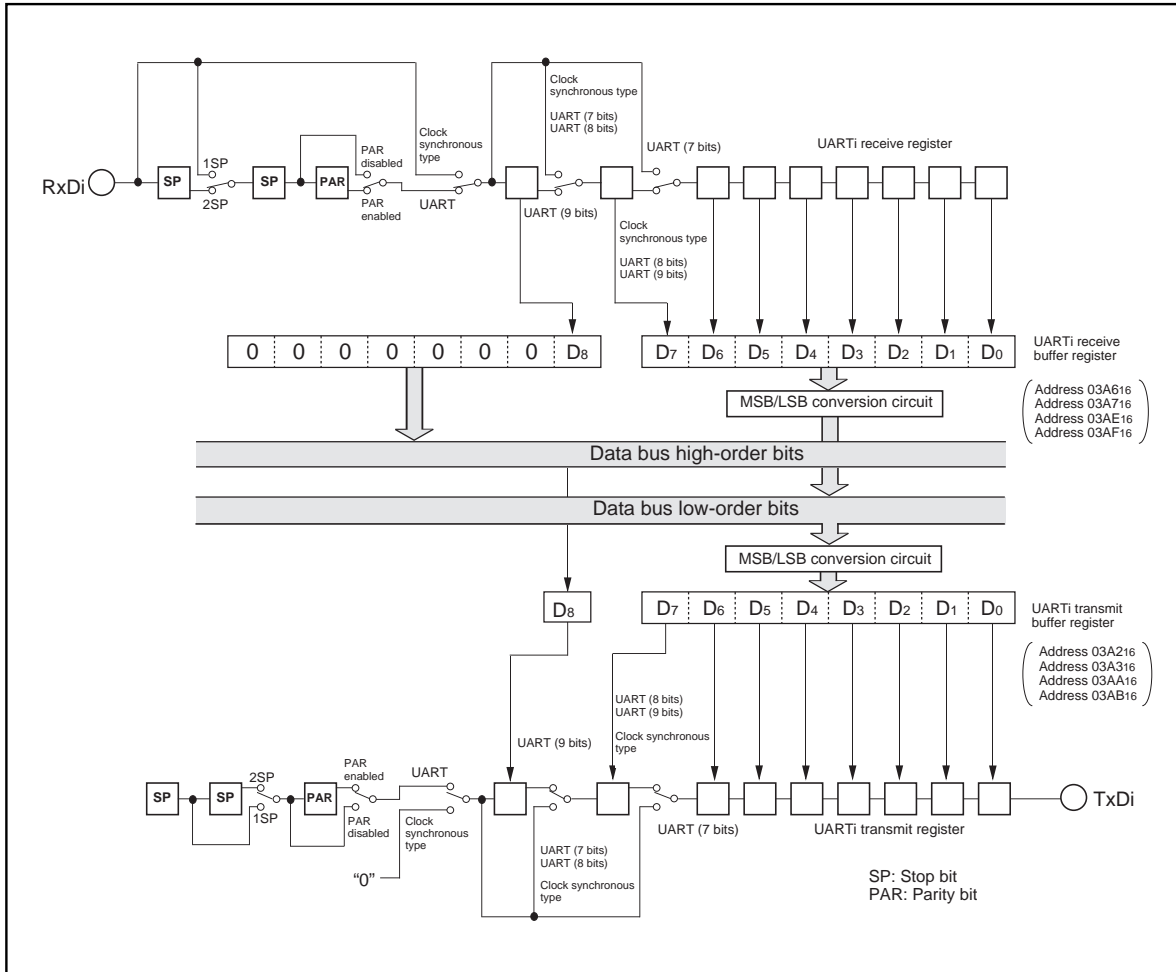


Figure 1.17.2. Block diagram of UARTi (i = 0, 1) transmit/receive unit

Serial I/O

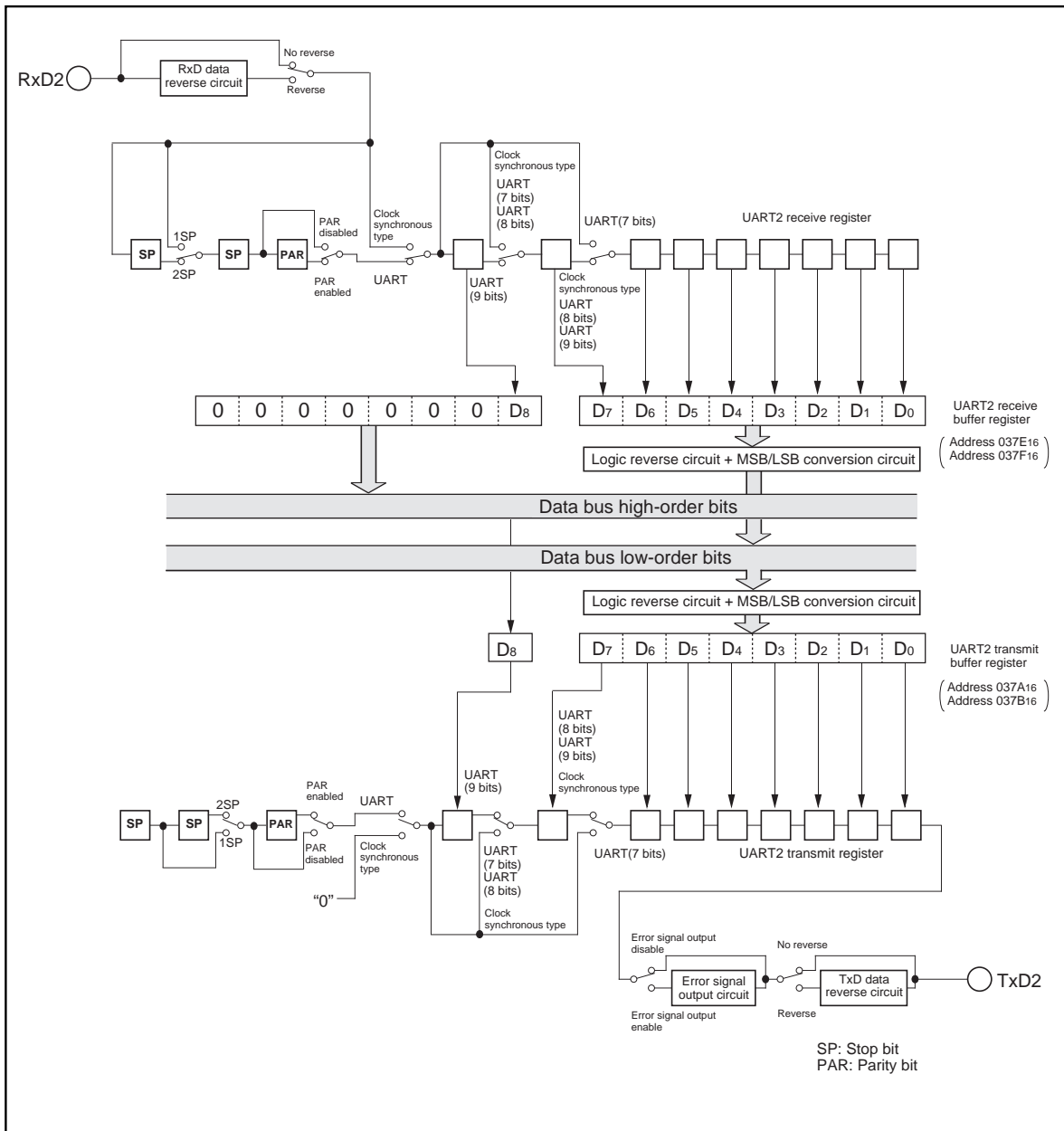


Figure 1.17.3. Block diagram of UART2 transmit/receive unit

Serial I/O

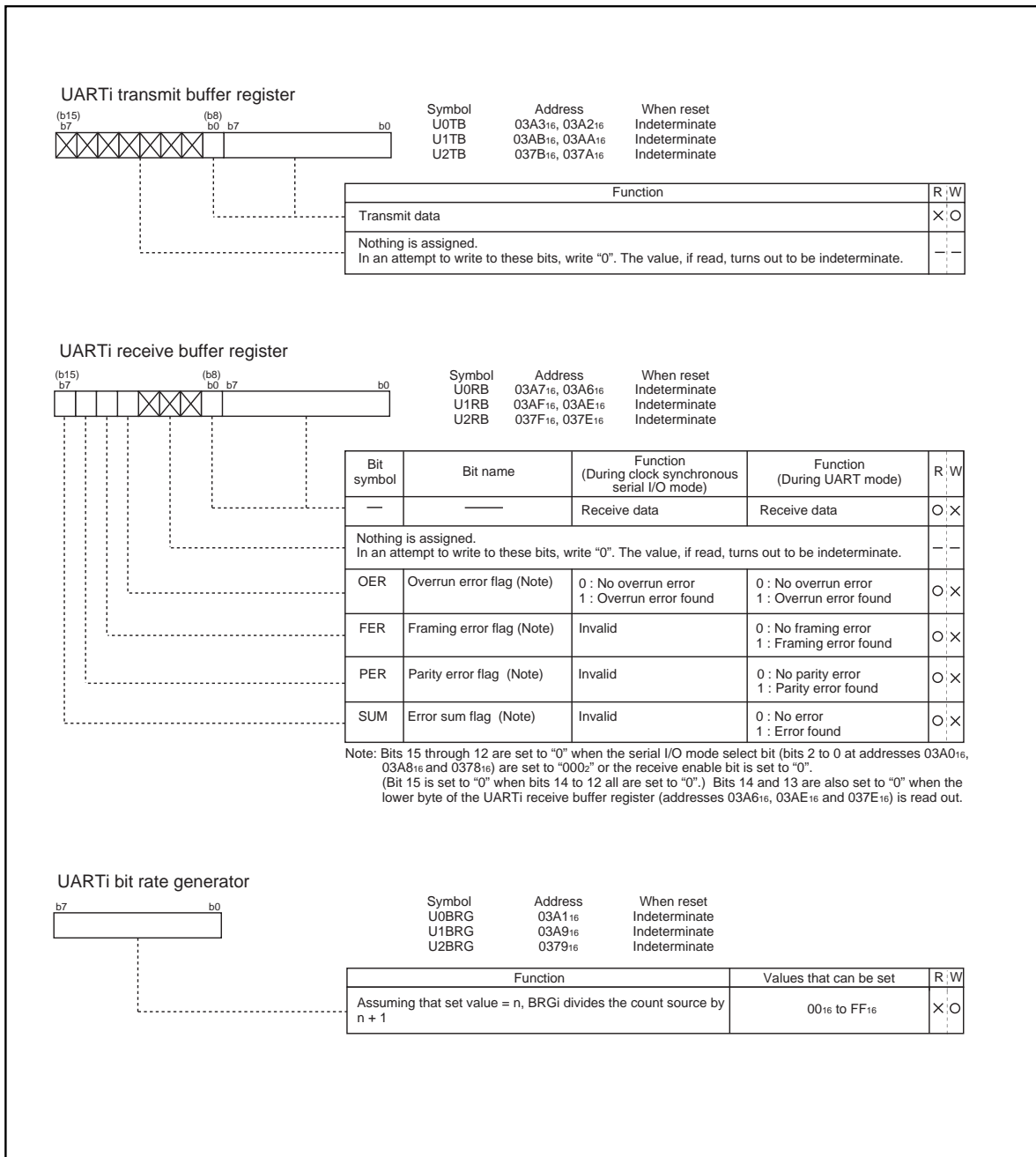


Figure 1.17.4. Serial I/O-related registers (1)

Serial I/O

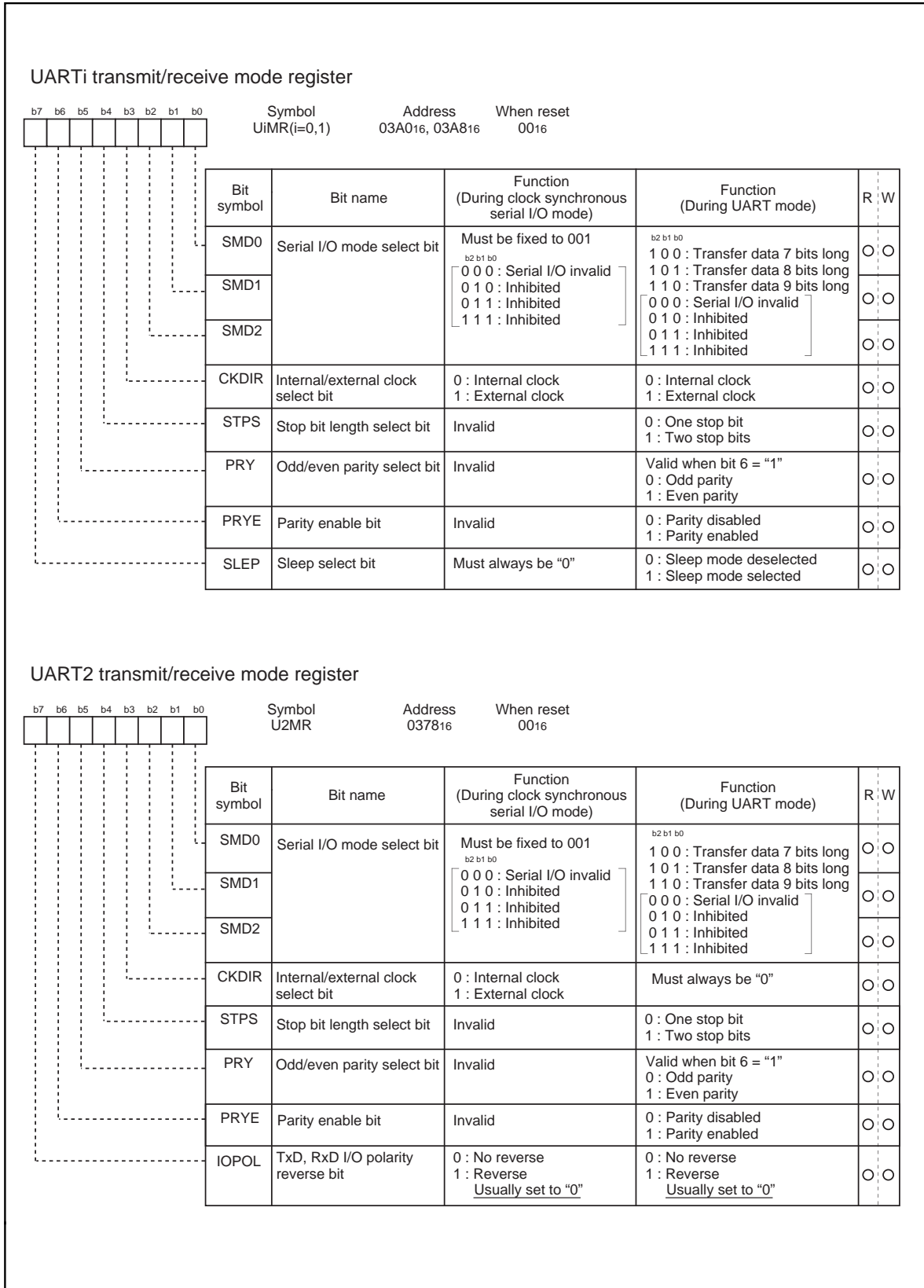


Figure 1.17.5. Serial I/O-related registers (2)

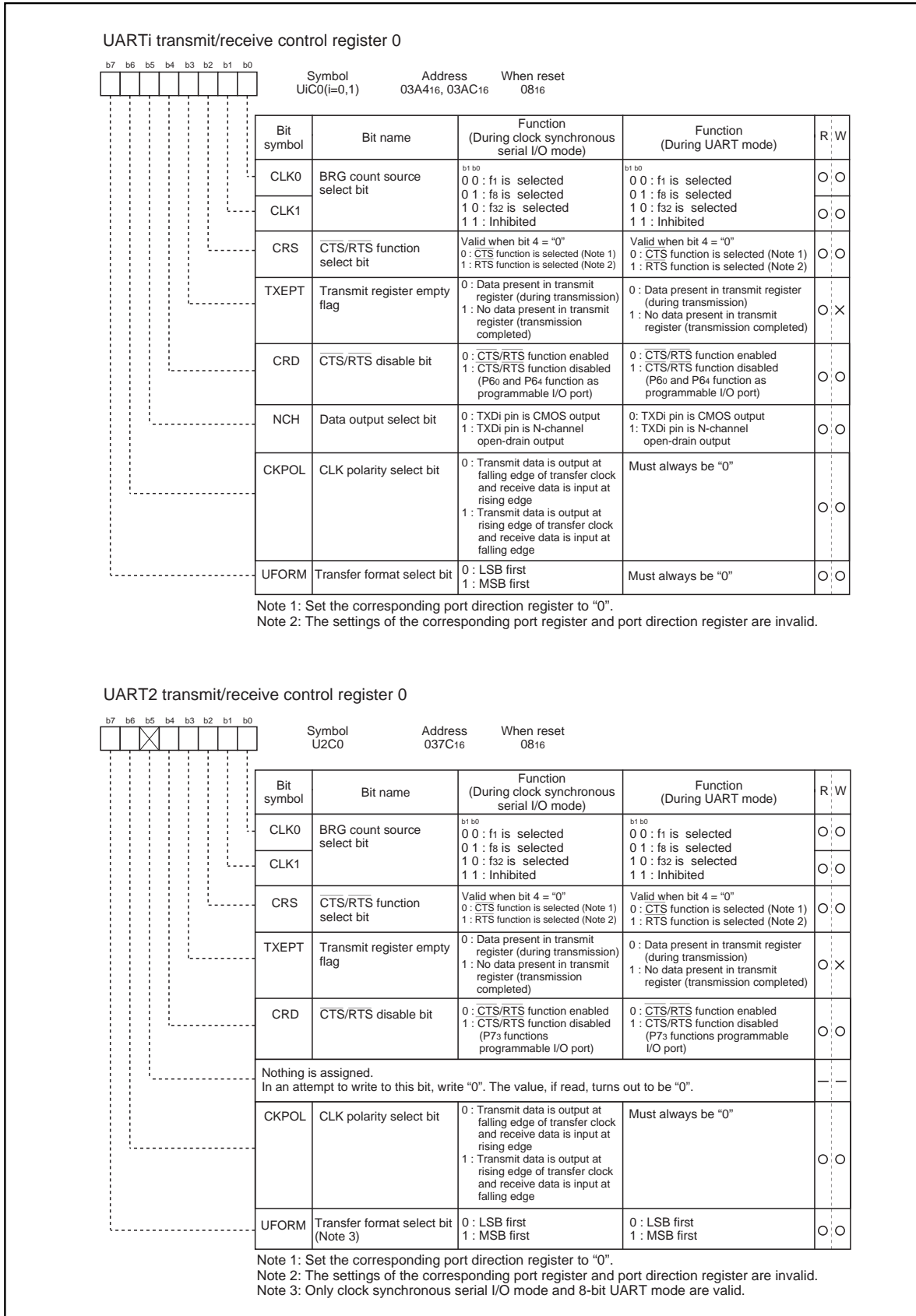


Figure 1.17.6. Serial I/O-related registers (3)

Serial I/O

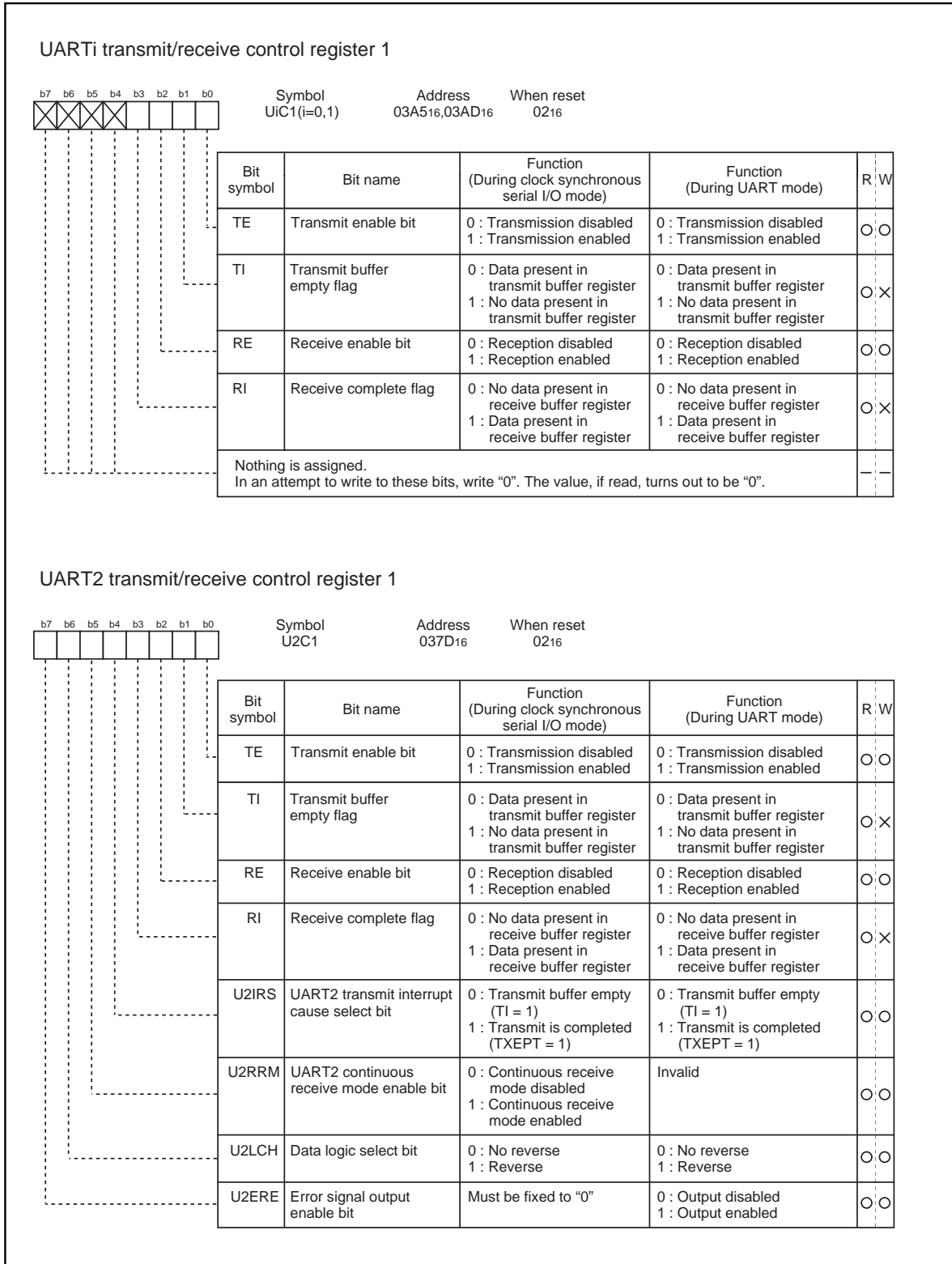
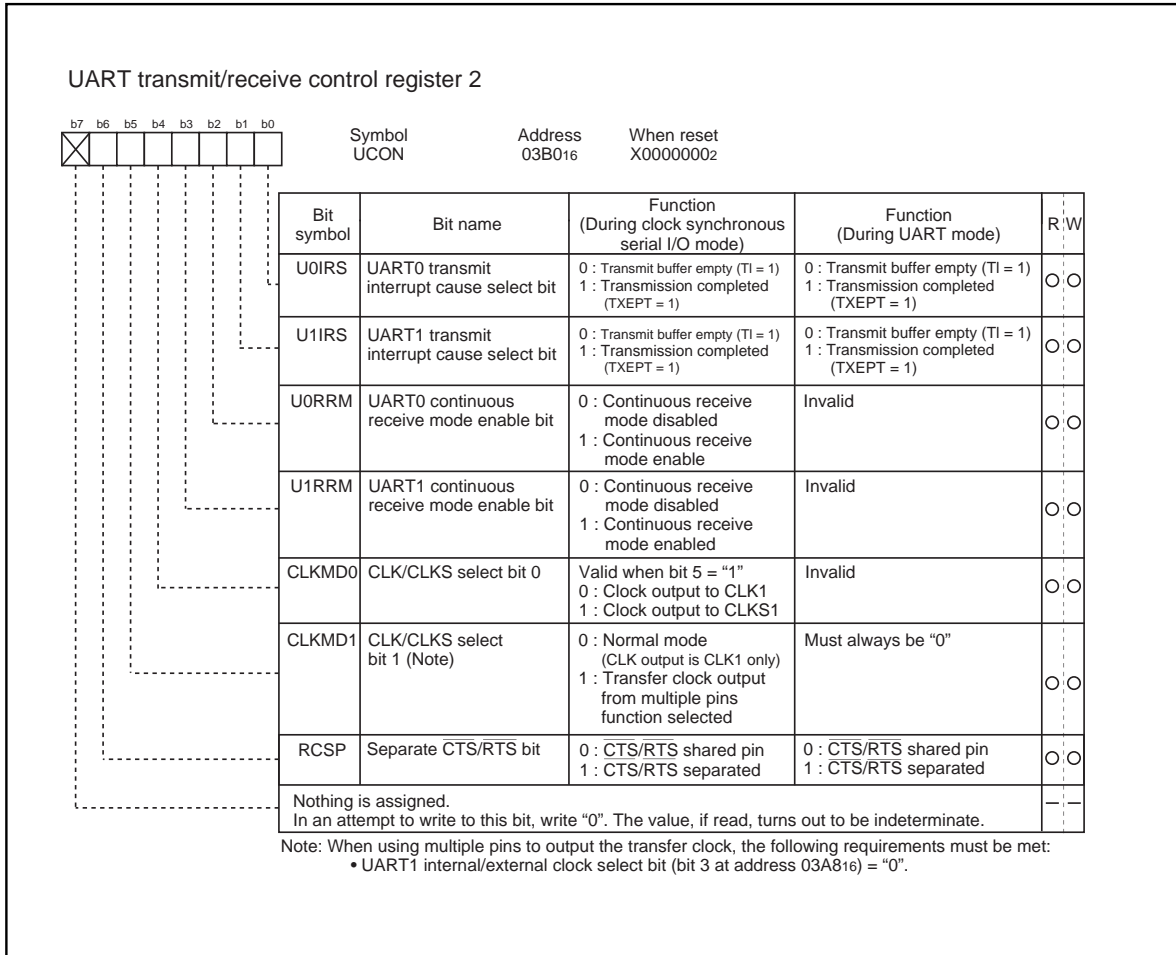


Figure 1.17.7. Serial I/O-related registers (4)



**Figure 1.17.8. Serial I/O-related registers (5)**

## Clock synchronous serial I/O mode

**(1) Clock synchronous serial I/O mode**

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Table 1.17.2 and table 1.17.3 list the specifications of the clock synchronous serial I/O mode. Figure 1.17.9 shows the UARTi transmit/receive mode register.

**Table 1.17.2. Specifications of clock synchronous serial I/O mode (1)**

Item	Specification
Transfer data format	<ul style="list-style-type: none"> <li>• Transfer data length: 8 bits</li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>• When internal clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub>, 0378<sub>16</sub> = "0") : <math>f_i / 2(n+1)</math> (Note 1) <math>f_i = f_1, f_8, f_{32}</math></li> <li>• When external clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub>, 0378<sub>16</sub> = "1") : Input from CLKi pin</li> </ul>
Transmission/reception control	<ul style="list-style-type: none"> <li>• CTS function/RTS function/CTS, RTS function chosen to be invalid</li> </ul>
Transmission start condition	<ul style="list-style-type: none"> <li>• To start transmission, the following requirements must be met: <ul style="list-style-type: none"> <li>– Transmit enable bit (bit 0 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "1"</li> <li>– Transmit buffer empty flag (bit 1 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "0"</li> <li>– When <math>\overline{\text{CTS}}</math> function selected, <math>\overline{\text{CTS}}</math> input level = "L"</li> </ul> </li> <li>• Furthermore, if external clock is selected, the following requirements must also be met: <ul style="list-style-type: none"> <li>– CLKi polarity select bit (bit 6 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>, 037C<sub>16</sub>) = "0": CLKi input level = "H"</li> <li>– CLKi polarity select bit (bit 6 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>, 037C<sub>16</sub>) = "1": CLKi input level = "L"</li> </ul> </li> </ul>
Reception start condition	<ul style="list-style-type: none"> <li>• To start reception, the following requirements must be met: <ul style="list-style-type: none"> <li>– Receive enable bit (bit 2 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "1"</li> <li>– Transmit enable bit (bit 0 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "1"</li> <li>– Transmit buffer empty flag (bit 1 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "0"</li> </ul> </li> <li>• Furthermore, if external clock is selected, the following requirements must also be met: <ul style="list-style-type: none"> <li>– CLKi polarity select bit (bit 6 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>, 037C<sub>16</sub>) = "0": CLKi input level = "H"</li> <li>– CLKi polarity select bit (bit 6 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>, 037C<sub>16</sub>) = "1": CLKi input level = "L"</li> </ul> </li> </ul>
Interrupt request generation timing	<ul style="list-style-type: none"> <li>• When transmitting <ul style="list-style-type: none"> <li>– Transmit interrupt cause select bit (bits 0, 1 at address 03B0<sub>16</sub>, bit 4 at address 037D<sub>16</sub>) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed</li> <li>– Transmit interrupt cause select bit (bits 0, 1 at address 03B0<sub>16</sub>, bit 4 at address 037D<sub>16</sub>) = "1": Interrupts requested when data transmission from UARTi transfer register is completed</li> </ul> </li> <li>• When receiving <ul style="list-style-type: none"> <li>– Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed</li> </ul> </li> </ul>
Error detection	<ul style="list-style-type: none"> <li>• Overrun error (Note 2) This error occurs when the next data is ready before contents of UARTi receive buffer register are read out</li> </ul>

Note 1: "n" denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UART bit rate generator.

Note 2: If an overrun error occurs, the UARTi receive buffer will have the next data written in. Note also that the UARTi receive interrupt request bit is not set to "1".



**Table 1.17.3. Specifications of clock synchronous serial I/O mode (2)**

Item	Specification
Select function	<ul style="list-style-type: none"> <li>• CLK polarity selection Whether transmit data is output/input at the rising edge or falling edge of the transfer clock can be selected</li> <li>• LSB first/MSB first selection Whether transmission/reception begins with bit 0 or bit 7 can be selected</li> <li>• Continuous receive mode selection Reception is enabled simultaneously by a read from the receive buffer register</li> <li>• Transfer clock output from multiple pins selection (UART1) (Note) UART1 transfer clock can be chosen by software to be output from one of the two pins set</li> <li>• Separate <math>\overline{\text{CTS}}</math>/<math>\overline{\text{RTS}}</math> pins (UART0) (Note) UART0 <math>\overline{\text{CTS}}</math> and <math>\overline{\text{RTS}}</math> pins each can be assigned to separate pins</li> <li>• Switching serial data logic (UART2) Whether to reverse data in writing to the transmission buffer register or reading the reception buffer register can be selected.</li> <li>• TxD, RxD I/O polarity reverse (UART2) This function is reversing TxD port output and RxD port input. All I/O data level is reversed.</li> </ul>

Note: The transfer clock output from multiple pins and the separate  $\overline{\text{CTS}}$ / $\overline{\text{RTS}}$  pins functions cannot be selected simultaneously.

Clock synchronous serial I/O mode

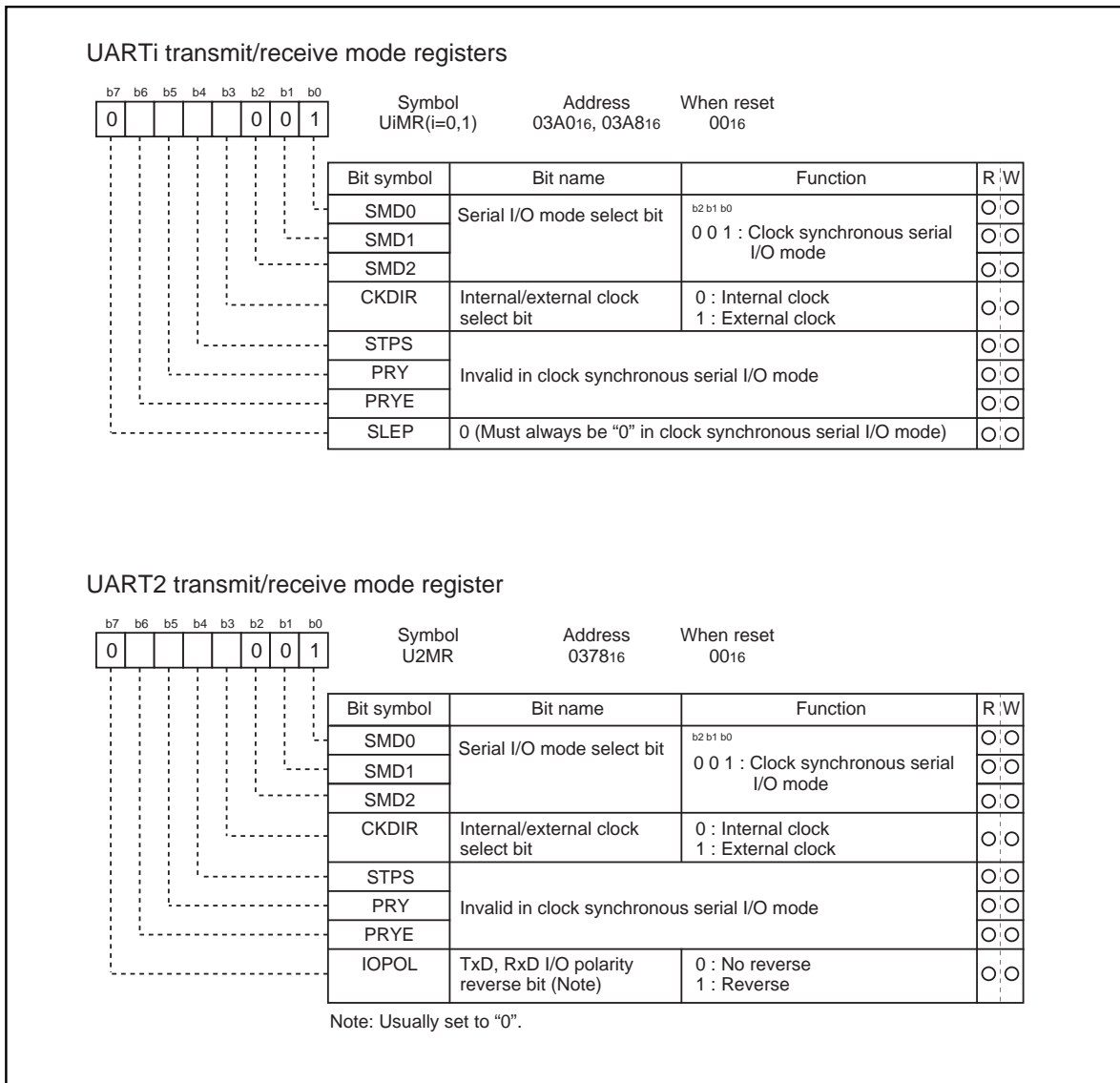


Figure 1.17.9. UARTi transmit/receive mode register in clock synchronous serial I/O mode

## Clock synchronous serial I/O mode

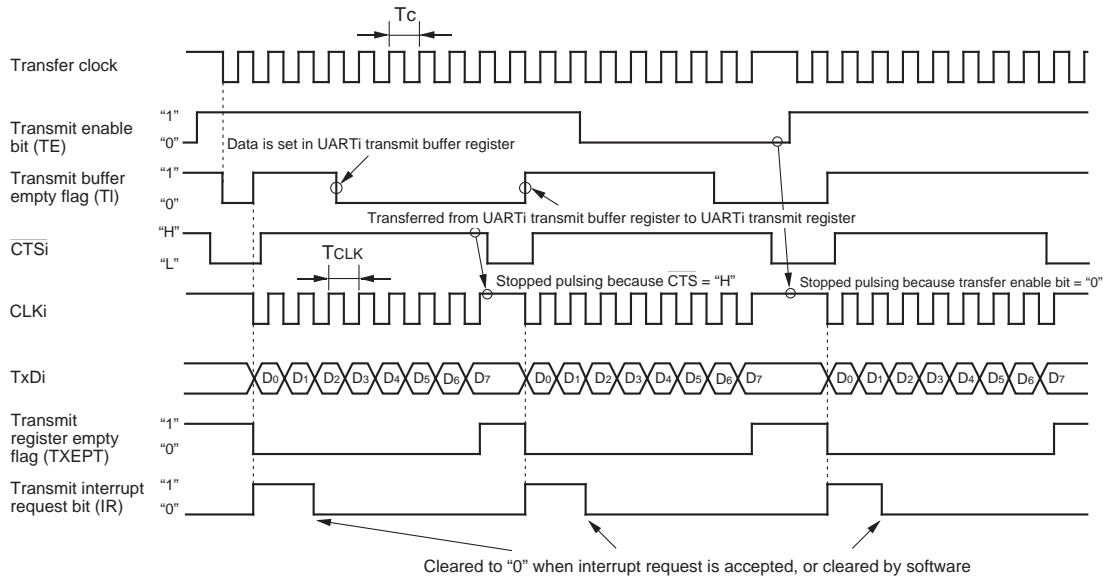
Table 1.17.4 lists the functions of the input/output pins during clock synchronous serial I/O mode. This table shows the pin functions when the transfer clock output from multiple pins and the separate  $\overline{\text{CTS}}$ / $\overline{\text{RTS}}$  pins functions are not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a "H". (If the N-channel open-drain is selected, this pin is in floating state.)

**Table 1.17.4. Input/output pin functions in clock synchronous serial I/O mode**

Pin name	Function	Method of selection
TxDi (P63, P67, P70)	Serial data output	(Outputs dummy data when performing reception only)
RxDi (P62, P66, P71)	Serial data input	Port P62, P66 and P71 direction register (bits 2 and 6 at address 03EE16, bit 1 at address 03EF16) = "0" (Can be used as an input port when performing transmission only)
CLKi (P61, P65, P72)	Transfer clock output	Internal/external clock select bit (bit 3 at address 03A016, 03A816, 037816) = "0"
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A016, 03A816, 037816) = "1" Port P61, P65 and P72 direction register (bits 1 and 5 at address 03EE16, bit 2 at address 03EF16) = "0"
$\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ (P60, P64, P73)	$\overline{\text{CTS}}$ input	$\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ disable bit (bit 4 at address 03A416, 03AC16, 037C16) = "0" $\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ function select bit (bit 2 at address 03A416, 03AC16, 037C16) = "0" Port P60, P64 and P73 direction register (bits 0 and 4 at address 03EE16, bit 3 at address 03EF16) = "0"
	$\overline{\text{RTS}}$ output	$\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ disable bit (bit 4 at address 03A416, 03AC16, 037C16) = "0" $\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ function select bit (bit 2 at address 03A416, 03AC16, 037C16) = "1"
	Programmable I/O port	$\overline{\text{CTS}}$ / $\overline{\text{RTS}}$ disable bit (bit 4 at address 03A416, 03AC16, 037C16) = "1"

(when transfer clock output from multiple pins and separate  $\overline{\text{CTS}}$ / $\overline{\text{RTS}}$  pins functions are not selected)

• Example of transmit timing (when internal clock is selected)



• Example of receive timing (when external clock is selected)

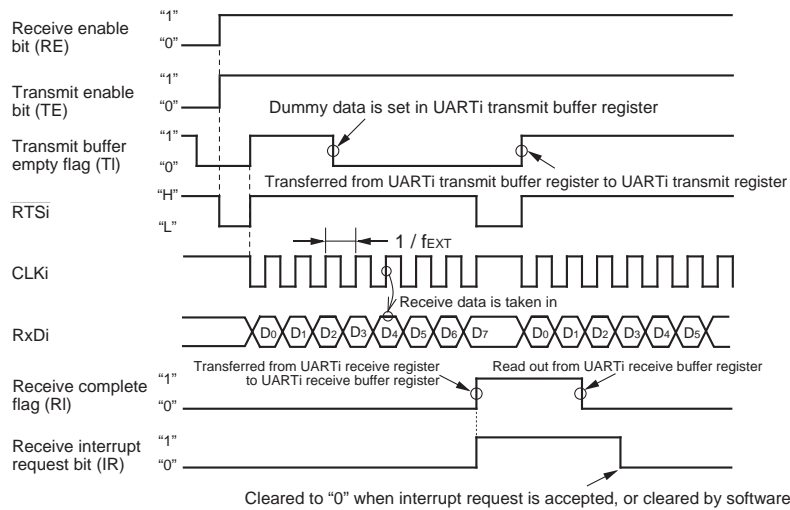


Figure 1.17.10. Typical transmit/receive timings in clock synchronous serial I/O mode

Clock synchronous serial I/O mode

**(a) Polarity select function**

As shown in Figure 1.17.11, the CLK polarity select bit (bit 6 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>, 037C<sub>16</sub>) allows selection of the polarity of the transfer clock.

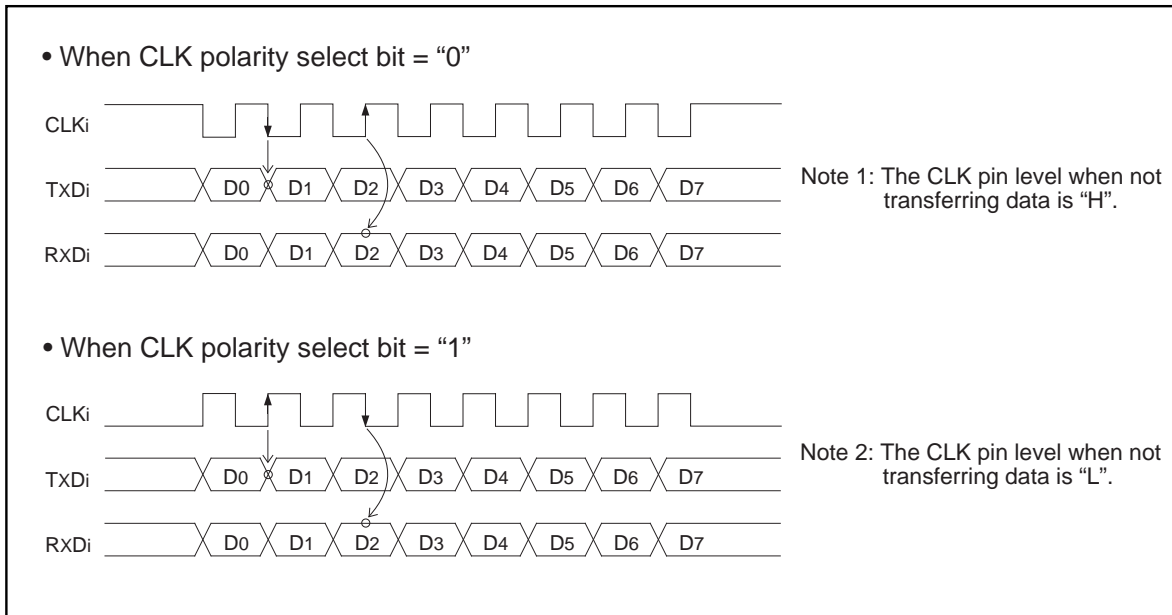


Figure 1.17.11. Polarity of transfer clock

**(b) LSB first/MSB first select function**

As shown in Figure 1.17.12, when the transfer format select bit (bit 7 at addresses 03A4<sub>16</sub>, 03AC<sub>16</sub>, 037C<sub>16</sub>) = "0", the transfer format is "LSB first"; when the bit = "1", the transfer format is "MSB first".

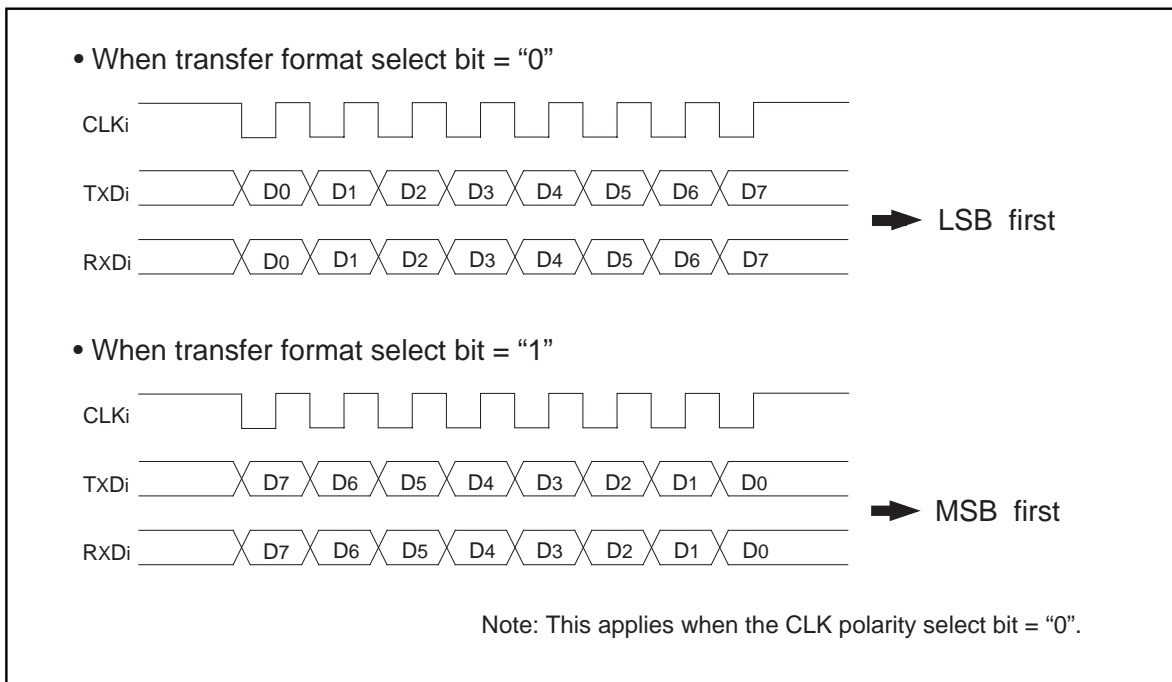


Figure 1.17.12. Transfer format

## Clock synchronous serial I/O mode

**(c) Transfer clock output from multiple pins function (UART1)**

This function allows the setting two transfer clock output pins and choosing one of the two to output a clock by using the CLK and CLKS select bit (bits 4 and 5 at address 03B016). (See Figure 1.17.13.)

The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected, UART1  $\overline{\text{CTS}}/\text{RTS}$  function cannot be used.

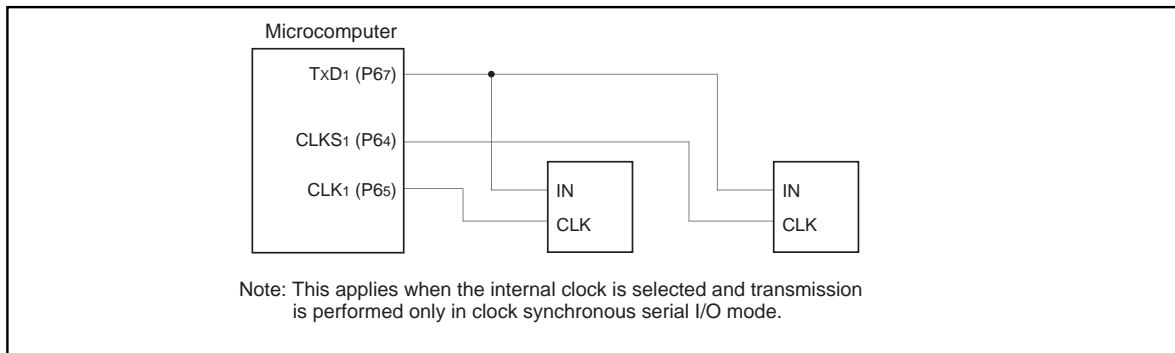


Figure 1.17.13. The transfer clock output from the multiple pins function usage

**(d) Continuous receive mode**

If the continuous receive mode enable bit (bits 2 and 3 at address 03B016, bit 5 at address 037D16) is set to "1", the unit is placed in continuous receive mode. In this mode, when the receive buffer register is read out, the unit simultaneously goes to a receive enable state without having to set dummy data to the transmit buffer register back again.

**(e) Separate  $\overline{\text{CTS}}/\text{RTS}$  pins function (UART0)**

This function works the same way as in the clock asynchronous serial I/O (UART) mode. The method of setting and the input/output pin functions are both the same, so refer to select function in the next section, "(2) Clock asynchronous serial I/O (UART) mode." Note that this function is invalid if the transfer clock output from the multiple pins function is selected.

**(f) Serial data logic switch function (UART2)**

When the data logic select bit (bit6 at address 037D16) = "1", and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 1.17.14 shows the example of serial data logic switch timing.

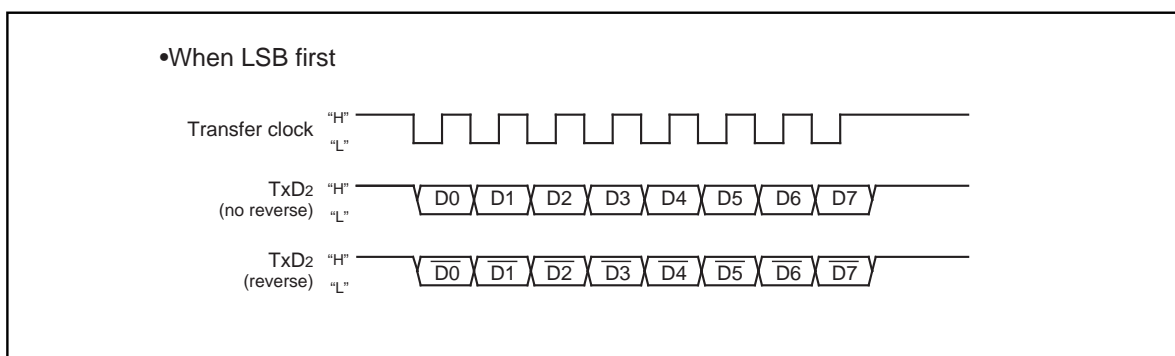


Figure 1.17.14. Serial data logic switch timing

## Clock asynchronous serial I/O (UART) mode

**(2) Clock asynchronous serial I/O (UART) mode**

The UART mode allows transmitting and receiving data after setting the desired transfer rate and transfer data format. Table 1.17.5 and table 1.17.6 list the specifications of the UART mode. Figure 1.17.15 shows the UARTi transmit/receive mode register.

**Table 1.17.5. Specifications of UART Mode (1)**

Item	Specification
Transfer data format	<ul style="list-style-type: none"> <li>• Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected</li> <li>• Start bit: 1 bit</li> <li>• Parity bit: Odd, even, or nothing as selected</li> <li>• Stop bit: 1 bit or 2 bits as selected</li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>• When internal clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub>, 0378<sub>16</sub> = "0") : <math>f_i/16(n+1)</math> (Note 1) <math>f_i = f_1, f_8, f_{32}</math></li> <li>• When external clock is selected (bit 3 at addresses 03A0<sub>16</sub>, 03A8<sub>16</sub> = "1") : <math>f_{EXT}/16(n+1)</math> (Note 1) (Note 2) (Do not set external clock for UART2)</li> </ul>
Transmission/reception control	<ul style="list-style-type: none"> <li>• <math>\overline{CTS}</math> function/<math>\overline{RTS}</math> function/<math>\overline{CTS}</math>, <math>\overline{RTS}</math> function chosen to be invalid</li> </ul>
Transmission start condition	<ul style="list-style-type: none"> <li>• To start transmission, the following requirements must be met: <ul style="list-style-type: none"> <li>- Transmit enable bit (bit 0 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "1"</li> <li>- Transmit buffer empty flag (bit 1 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "0"</li> <li>- When <math>\overline{CTS}</math> function selected, <math>\overline{CTS}</math> input level = "L"</li> </ul> </li> </ul>
Reception start condition	<ul style="list-style-type: none"> <li>• To start reception, the following requirements must be met: <ul style="list-style-type: none"> <li>- Receive enable bit (bit 2 at addresses 03A5<sub>16</sub>, 03AD<sub>16</sub>, 037D<sub>16</sub>) = "1"</li> <li>- Start bit detection</li> </ul> </li> </ul>
Interrupt request generation timing	<ul style="list-style-type: none"> <li>• When transmitting <ul style="list-style-type: none"> <li>- Transmit interrupt cause select bits (bits 0,1 at address 03B0<sub>16</sub>, bit4 at address 037D<sub>16</sub>) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed</li> <li>- Transmit interrupt cause select bits (bits 0, 1 at address 03B0<sub>16</sub>, bit4 at address 037D<sub>16</sub>) = "1": Interrupts requested when data transmission from UARTi transfer register is completed</li> </ul> </li> <li>• When receiving <ul style="list-style-type: none"> <li>- Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed</li> </ul> </li> </ul>
Error detection	<ul style="list-style-type: none"> <li>• Overrun error (Note 3) <ul style="list-style-type: none"> <li>This error occurs when the next data is ready before contents of UARTi receive buffer register are read out</li> </ul> </li> <li>• Framing error <ul style="list-style-type: none"> <li>This error occurs when the number of stop bits set is not detected</li> </ul> </li> <li>• Parity error <ul style="list-style-type: none"> <li>This error occurs when if parity is enabled, the number of 1's in parity and character bits does not match the number of 1's set</li> </ul> </li> <li>• Error sum flag <ul style="list-style-type: none"> <li>This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered</li> </ul> </li> </ul>

Note 1: 'n' denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UARTi bit rate generator.

Note 2:  $f_{EXT}$  is input from the CLKi pin.

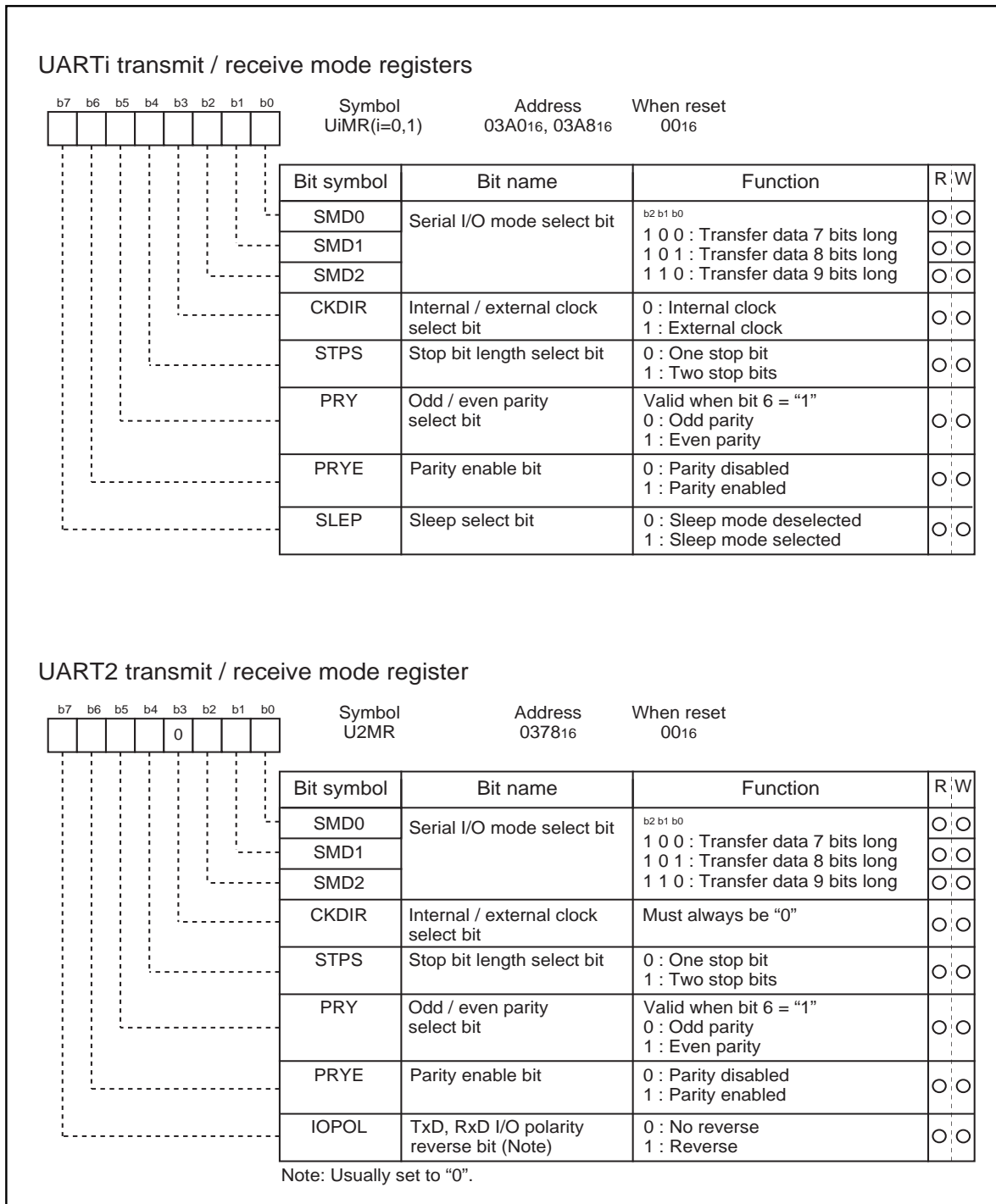
Note 3: If an overrun error occurs, the UARTi receive buffer will have the next data written in. Note also that the UARTi receive interrupt request bit is not set to "1".

**Table 1.17.6. Specifications of UART Mode (2)**

Item	Specification
Select function	<ul style="list-style-type: none"><li data-bbox="540 321 1395 390">• Separate CTS/RTS pins (UART0) UART0 <math>\overline{\text{CTS}}</math> and <math>\overline{\text{RTS}}</math> pins each can be assigned to separate pins</li><li data-bbox="540 396 1395 499">• Sleep mode selection (UART0, UART1) This mode is used to transfer data to and from one of multiple slave micro-computers</li><li data-bbox="540 506 1395 609">• Serial data logic switch (UART2) This function is reversing logic value of transferring data. Start bit, parity bit and stop bit are not reversed.</li><li data-bbox="540 615 1395 718">• TxD, RxD I/O polarity switch This function is reversing TxD port output and RxD port input. All I/O data level is reversed.</li></ul>



## Clock asynchronous serial I/O (UART) mode

Figure 1.17.15. UART<sub>i</sub> transmit/receive mode register in UART mode

## Clock asynchronous serial I/O (UART) mode

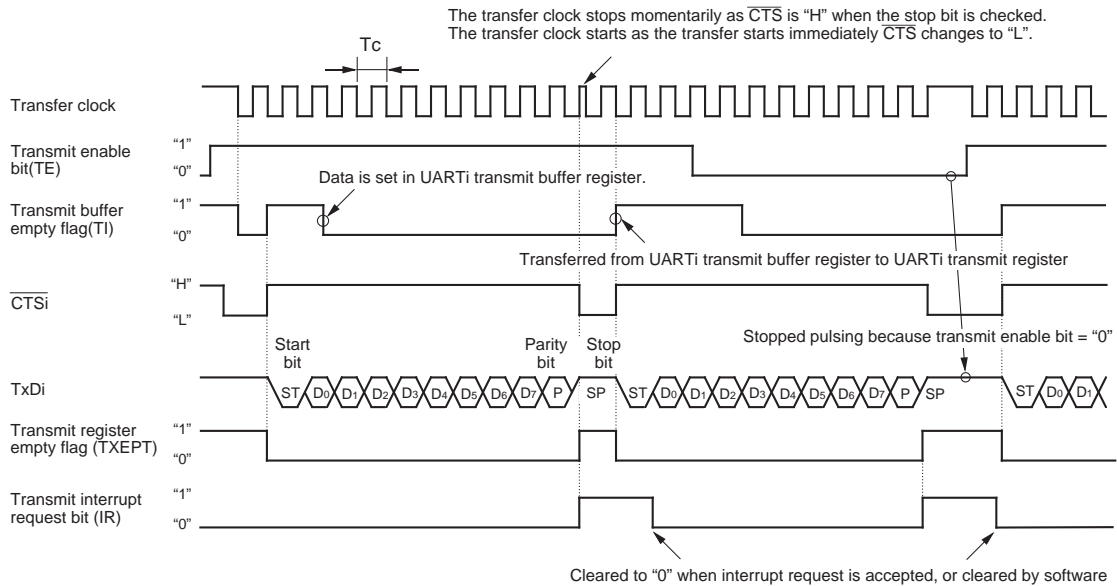
Table 1.17.7 lists the functions of the input/output pins during UART mode. This table shows the pin functions when the separate  $\overline{\text{CTS}}/\overline{\text{RTS}}$  pins function is not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a "H". (If the N-channel open-drain is selected, this pin is in floating state.)

**Table 1.17.7. Input/output pin functions in UART mode**

Pin name	Function	Method of selection
TxDi (P63, P67, P70)	Serial data output	
RxDi (P62, P66, P71)	Serial data input	Port P62, P66 and P71 direction register (bits 2 and 6 at address 03EE16, bit 1 at address 03EF16) = "0" (Can be used as an input port when performing transmission only)
CLKi (P61, P65, P72)	Programmable I/O port	Internal/external clock select bit (bit 3 at address 03A016, 03A816, 037816) = "0"
	Transfer clock input	Internal/external clock select bit (bit 3 at address 03A016, 03A816) = "1" Port P61, P65 direction register (bits 1 and 5 at address 03EE16) = "0" (Do not set external clock for UART2)
$\overline{\text{CTS}}/\overline{\text{RTS}}_i$ (P60, P64, P73)	$\overline{\text{CTS}}$ input	$\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A416, 03AC16, 037C16) = "0" $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A416, 03AC16, 037C16) = "0" Port P60, P64 and P73 direction register (bits 0 and 4 at address 03EE16, bit 3 at address 03EF16) = "0"
	$\overline{\text{RTS}}$ output	$\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A416, 03AC16, 037C16) = "0" $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 at address 03A416, 03AC16, 037C16) = "1"
	Programmable I/O port	$\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 at address 03A416, 03AC16, 037C16) = "1"

(when separate  $\overline{\text{CTS}}/\overline{\text{RTS}}$  pins function is not selected)

• Example of transmit timing when transfer data is 8 bits long (parity enabled, one stop bit)



• Example of transmit timing when transfer data is 9 bits long (parity disabled, two stop bits)

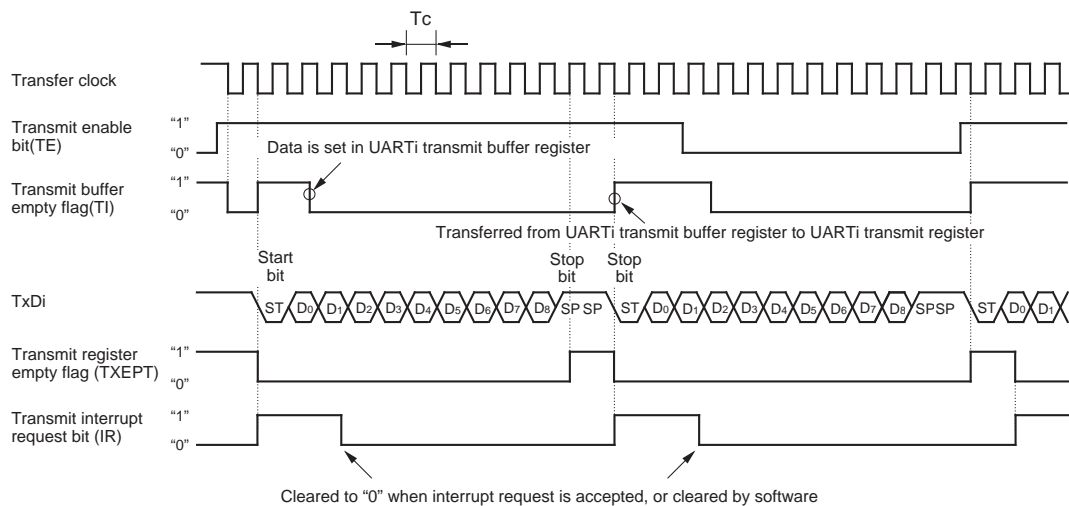


Figure 1.17.16. Typical transmit timings in UART mode (UART0, UART1)

Clock asynchronous serial I/O (UART) mode

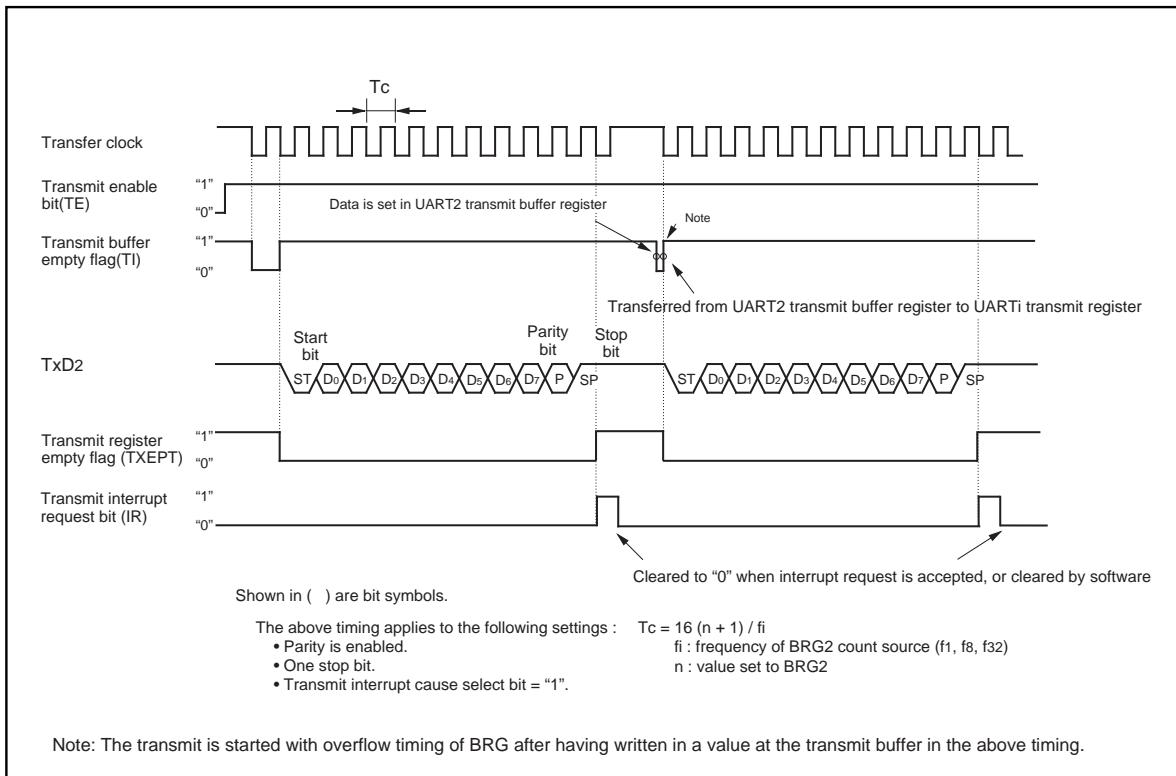


Figure 1.17.17. Typical transmit timings in UART mode (UART2)

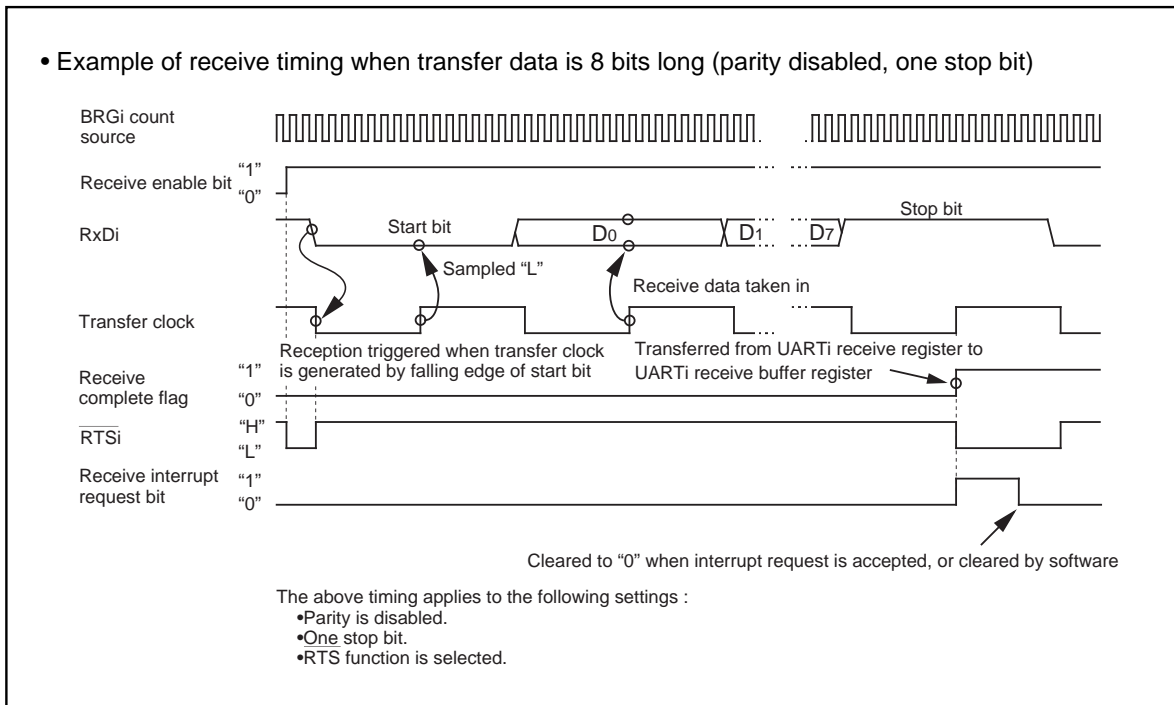


Figure 1.17.18. Typical receive timing in UART mode

**(a) Separate CTS/RTS pins function (UART0)**

Setting the CTS/RTS separate bit (bit 6 of address 03B016) to "1" inputs/outputs the CTS signal and RTS signal from different pins. Choose which to use, CTS or RTS, by use of the CTS/RTS function select bit (bit 2 of address 03A416). This function is effective in UART0 only. With this function chosen, the user cannot use the CTS/RTS function. Set "0" both to the CTS/RTS function select bit (bit 2 of address 03AC16) and to the CTS/RTS disable bit (bit 4 of address 03AC16).

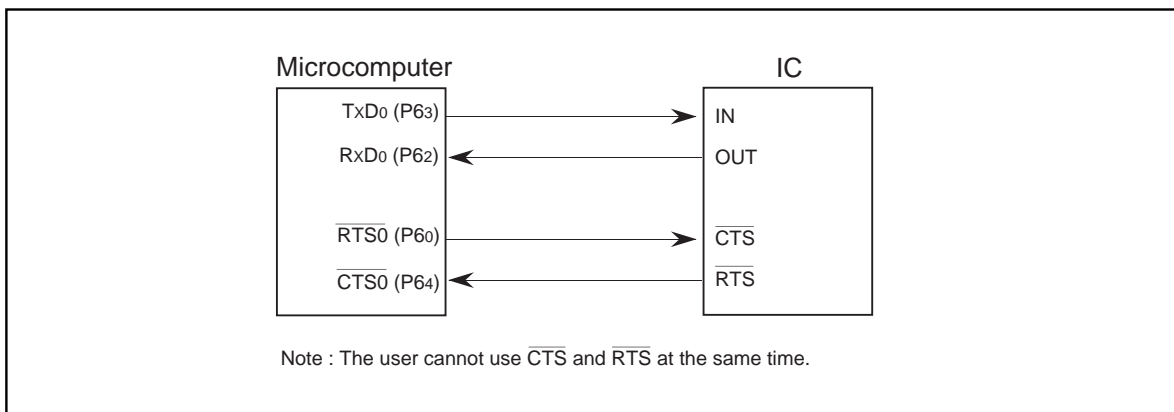


Figure 1.17.19. The separate CTS/RTS pins function usage

**(b) Sleep mode (UART0, UART1)**

This mode is used to transfer data between specific microcomputers among multiple microcomputers connected using UARTi. The sleep mode is selected when the sleep select bit (bit 7 at addresses 03A016, 03A816) is set to "1" during reception. In this mode, the unit performs receive operation when the MSB of the received data = "1" and does not perform receive operation when the MSB = "0".

## Clock asynchronous serial I/O (UART) mode

**(c) Function for switching serial data logic (UART2)**

When the data logic select bit (bit 6 of address 037D16) is assigned 1, data is inverted in writing to the transmission buffer register or reading the reception buffer register. Figure 1.17.20 shows the example of timing for switching serial data logic.

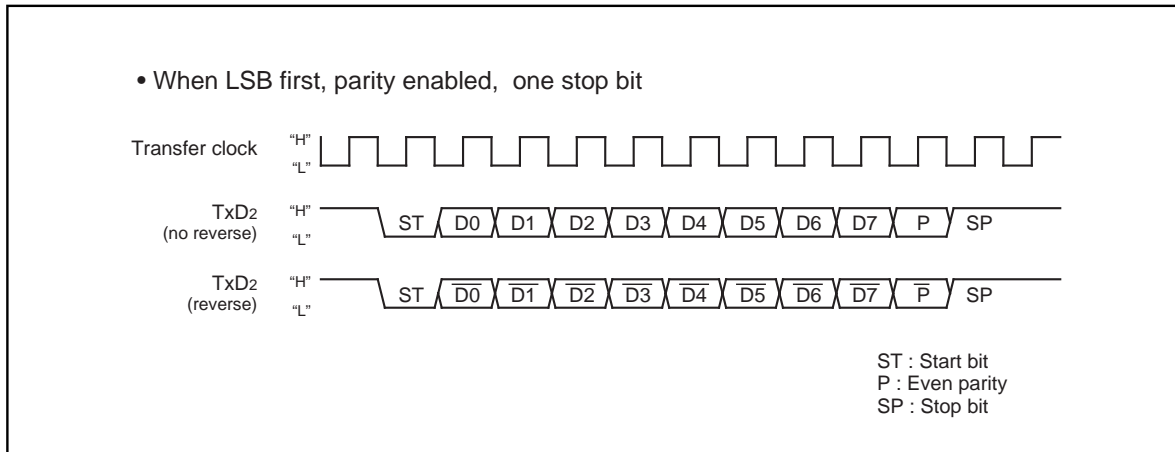


Figure 1.17.20. Timing for switching serial data logic

**(d) TxD, RxD I/O polarity reverse function (UART2)**

This function is to reverse TxD pin output and RxD pin input. The level of any data to be input or output (including the start bit, stop bit(s), and parity bit) is reversed. Set this function to "0" (not to reverse) for usual use.

**(e) Bus collision detection function (UART2)**

This function is to sample the output level of the TxD pin and the input level of the RxD pin at the rising edge of the transfer clock; if their values are different, then an interrupt request occurs. Figure 1.17.21 shows the example of detection timing of a buss collision (in UART mode).

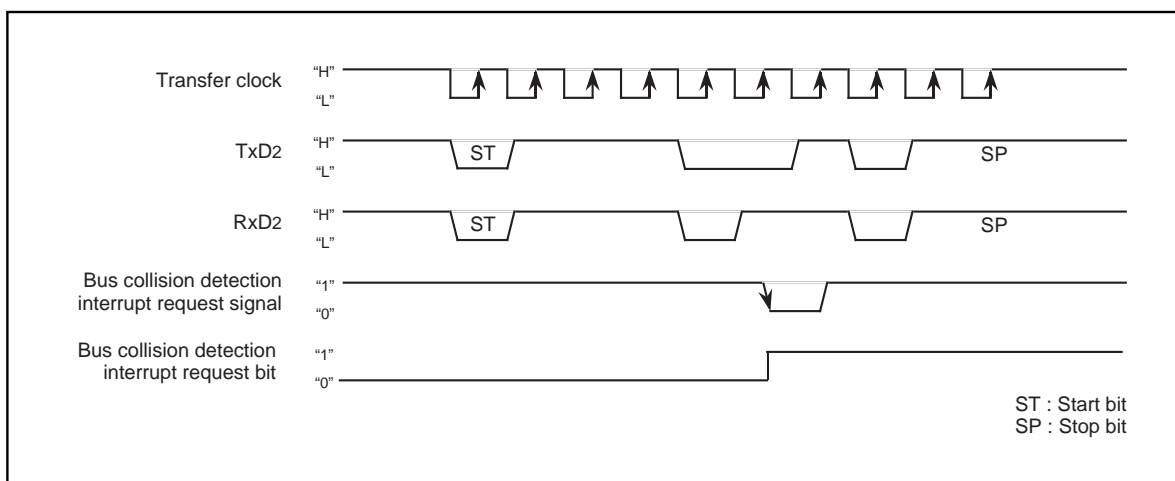


Figure 1.17.21. Detection timing of a bus collision (in UART mode)

## Clock asynchronous serial I/O (UART) mode

**(3) Clock-asynchronous serial I/O mode (compliant with the SIM interface)**

The SIM interface is used for connecting the microcomputer with a memory card or the like; adding some extra settings in UART2 clock-asynchronous serial I/O mode allows the user to effect this function. Table 1.17.8 shows the specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface).

**Table 1.17.8. Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface)**

Item	Specification
Transfer data format	<ul style="list-style-type: none"> <li>• Transfer data 8-bit UART mode (bit 2 through bit 0 of address 0378<sub>16</sub> = "1012")</li> <li>• One stop bit (bit 4 of address 0378<sub>16</sub> = "0")</li> <li>• With the direct format chosen               <ul style="list-style-type: none"> <li>Set parity to "even" (bit 5 and bit 6 of address 0378<sub>16</sub> = "1" and "1" respectively)</li> <li>Set data logic to "direct" (bit 6 of address 037D<sub>16</sub> = "0").</li> <li>Set transfer format to LSB (bit 7 of address 037C<sub>16</sub> = "0").</li> </ul> </li> <li>• With the inverse format chosen               <ul style="list-style-type: none"> <li>Set parity to "odd" (bit 5 and bit 6 of address 0378<sub>16</sub> = "0" and "1" respectively)</li> <li>Set data logic to "inverse" (bit 6 of address 037D<sub>16</sub> = "1")</li> <li>Set transfer format to MSB (bit 7 of address 037C<sub>16</sub> = "1")</li> </ul> </li> </ul>
Transfer clock	<ul style="list-style-type: none"> <li>• With the internal clock chosen (bit 3 of address 0378<sub>16</sub> = "0") : <math>f_i / 16 (n + 1)</math> (Note 1) : <math>f_i = f_1, f_8, f_{32}</math> (Do not set external clock)</li> </ul>
Transmission / reception control	<ul style="list-style-type: none"> <li>• Disable the CTS and RTS function (bit 4 of address 037C<sub>16</sub> = "1")</li> </ul>
Other settings	<ul style="list-style-type: none"> <li>• The sleep mode select function is not available for UART2</li> <li>• Set transmission interrupt factor to "transmission completed" (bit 4 of address 037D<sub>16</sub> = "1")</li> </ul>
Transmission start condition	<ul style="list-style-type: none"> <li>• To start transmission, the following requirements must be met:               <ul style="list-style-type: none"> <li>- Transmit enable bit (bit 0 of address 037D<sub>16</sub>) = "1"</li> <li>- Transmit buffer empty flag (bit 1 of address 037D<sub>16</sub>) = "0"</li> </ul> </li> </ul>
Reception start condition	<ul style="list-style-type: none"> <li>• To start reception, the following requirements must be met:               <ul style="list-style-type: none"> <li>- Reception enable bit (bit 2 of address 037D<sub>16</sub>) = "1"</li> <li>- Detection of a start bit</li> </ul> </li> </ul>
Interrupt request generation timing	<ul style="list-style-type: none"> <li>• When transmitting               <ul style="list-style-type: none"> <li>When data transmission from the UART2 transfer register is completed (bit 4 of address 037D<sub>16</sub> = "1")</li> </ul> </li> <li>• When receiving               <ul style="list-style-type: none"> <li>When data transfer from the UART2 receive register to the UART2 receive buffer register is completed</li> </ul> </li> </ul>
Error detection	<ul style="list-style-type: none"> <li>• Overrun error (see the specifications of clock-asynchronous serial I/O) (Note 3)</li> <li>• Framing error (see the specifications of clock-asynchronous serial I/O)</li> <li>• Parity error (see the specifications of clock-asynchronous serial I/O)               <ul style="list-style-type: none"> <li>- On the reception side, an "L" level is output from the TxD2 pin by use of the parity error signal output function (bit 7 of address 037D<sub>16</sub> = "1") when a parity error is detected</li> <li>- On the transmission side, a parity error is detected by the level of input to the RxD2 pin when a transmission interrupt occurs</li> </ul> </li> <li>• The error sum flag (see the specifications of clock-asynchronous serial I/O)</li> </ul>

Note 1: 'n' denotes the value 00<sub>16</sub> to FF<sub>16</sub> that is set to the UARTi bit rate generator.

Note 2: f<sub>EXT</sub> is input from the CLK2 pin.

Note 3: If an overrun error occurs, the UART2 receive buffer will have the next data written in. Note also that the UARTi receive interrupt request bit is not set to "1".

Clock asynchronous serial I/O (UART) mode

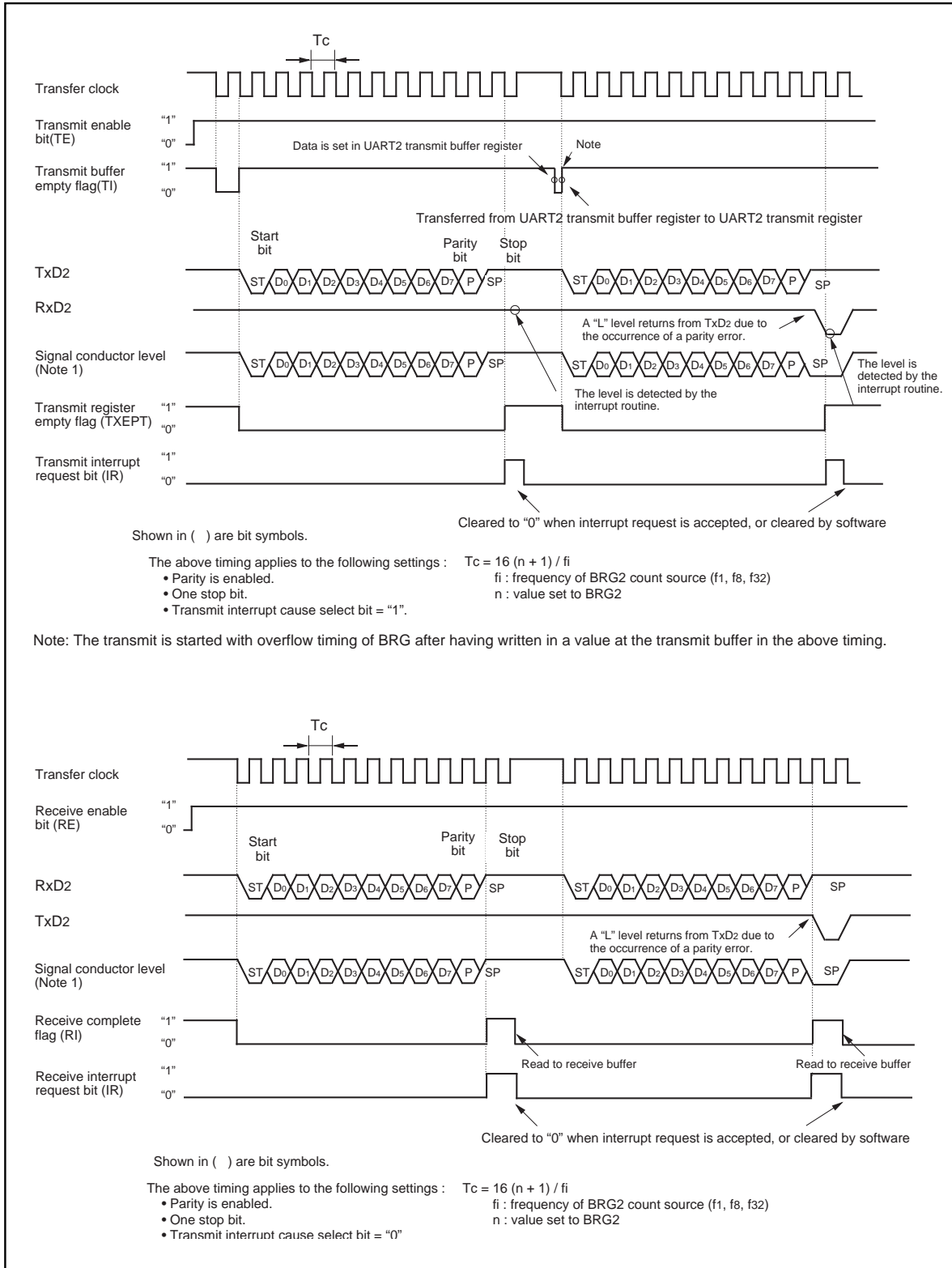
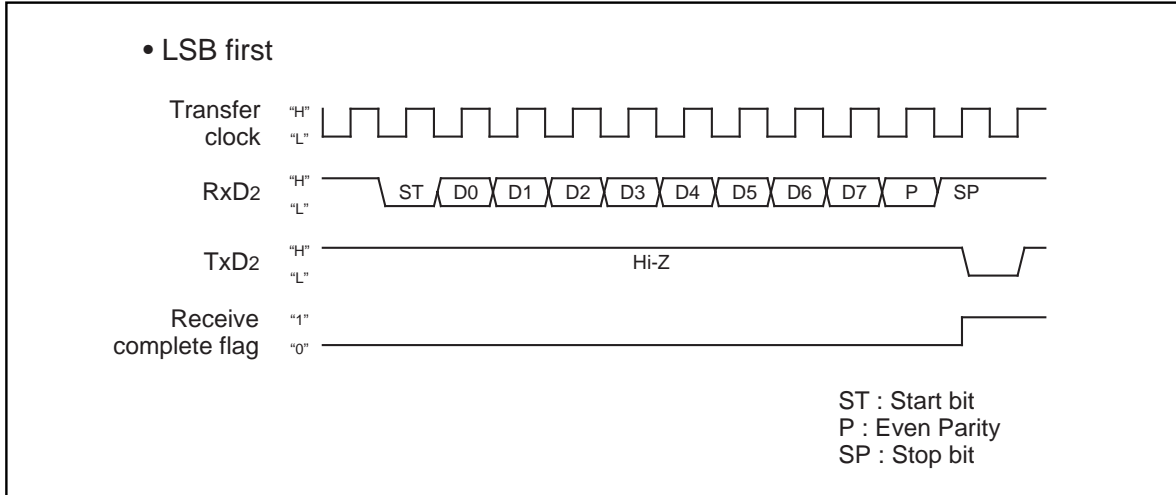


Figure 1.17.22. Typical transmit/receive timing in UART mode (compliant with the SIM interface)



**(a) Function for outputting a parity error signal**

With the error signal output enable bit (bit 7 of address 037D16) assigned “1”, you can output an “L” level from the TxD2 pin when a parity error is detected. In step with this function, the generation timing of a transmission completion interrupt changes to the detection timing of a parity error signal. Figure 1.17.23 shows the output timing of the parity error signal.

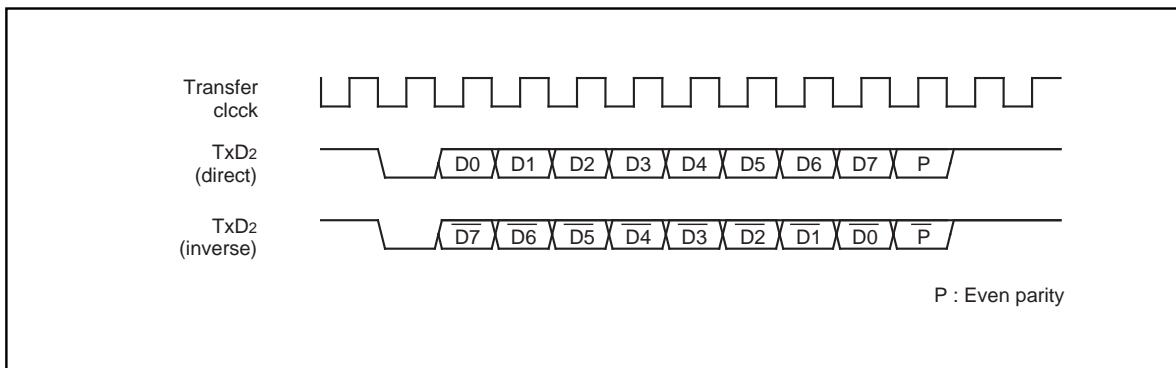


**Figure 1.17.23. Output timing of the parity error signal**

**(b) Direct format/inverse format**

Connecting the SIM card allows you to switch between direct format and inverse format. If you choose the direct format, D0 data is output from TxD2. If you choose the inverse format, D7 data is inverted and output from TxD2.

Figure 1.17.24 shows the SIM interface format.



**Figure 1.17.24. SIM interface format**

Figure 1.17.25 shows the example of connecting the SIM interface. Connect TxD2 and RxD2 and apply pull-up.

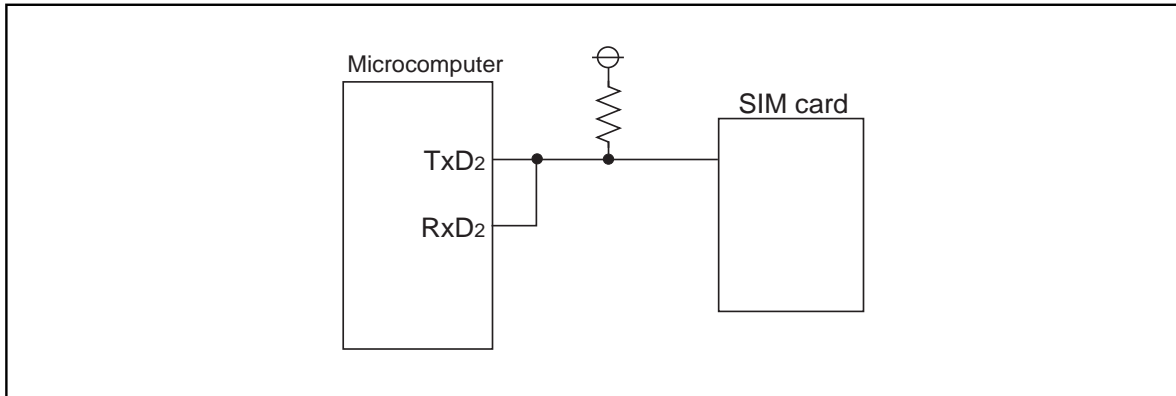


Figure 1.17.25. Connecting the SIM interface

## A-D Converter

The A-D converter consists of one 10-bit successive approximation A-D converter circuit with a capacitive coupling amplifier. Pins P10<sub>0</sub> to P10<sub>7</sub>, P9<sub>5</sub>, and P9<sub>6</sub> also function as the analog signal input pins. The direction registers of these pins for A-D conversion must therefore be set to input. The Vref connect bit (bit 5 at address 03D716) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin (VREF) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from VREF, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of 03D716 to connect VREF. The result of A-D conversion is stored in the A-D registers of the selected pins. When set to 10-bit precision, the low 8 bits are stored in the even addresses and the high 2 bits in the odd addresses. When set to 8-bit precision, the low 8 bits are stored in the even addresses.

Table 1.18.1 shows the performance of the A-D converter. Figure 1.18.1 shows the block diagram of the A-D converter, and Figures 1.18.2 and 1.18.3 show the A-D converter-related registers.

**Table 1.18.1. Performance of A-D converter**

Item	Performance
Method of A-D conversion	Successive approximation (capacitive coupling amplifier)
Analog input voltage (Note 1)	0V to AVCC (VCC)
Operating clock $\phi_{AD}$ (Note 2)	VCC = 5V fAD/divide-by-2 of fAD/divide-by-4 of fAD, fAD=f(XIN) VCC = 3V divide-by-2 of fAD/divide-by-4 of fAD, fAD=f(XIN)
Resolution	8-bit or 10-bit (selectable)
Absolute precision	VCC = 5V <ul style="list-style-type: none"> <li>• Without sample and hold function <math>\pm 3\text{LSB}</math></li> <li>• With sample and hold function (8-bit resolution) <math>\pm 2\text{LSB}</math></li> <li>• With sample and hold function (10-bit resolution) AN<sub>0</sub> to AN<sub>7</sub> input : <math>\pm 3\text{LSB}</math> ANEX<sub>0</sub> and ANEX<sub>1</sub> input (including mode in which external operation amp is connected) : <math>\pm 7\text{LSB}</math></li> </ul> VCC = 3V <ul style="list-style-type: none"> <li>• Without sample and hold function (8-bit resolution) <math>\pm 2\text{LSB}</math></li> </ul>
Operating modes	One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1
Analog input pins	8pins (AN <sub>0</sub> to AN <sub>7</sub> ) + 2pins (ANEX <sub>0</sub> and ANEX <sub>1</sub> )
A-D conversion start condition	<ul style="list-style-type: none"> <li>• Software trigger A-D conversion starts when the A-D conversion start flag changes to "1"</li> <li>• External trigger (can be retriggered) A-D conversion starts when the A-D conversion start flag is "1" and the <math>\overline{\text{ADTRG}}/\text{P97}</math> input changes from "H" to "L"</li> </ul>
Conversion speed per pin	<ul style="list-style-type: none"> <li>• Without sample and hold function 8-bit resolution: 49 <math>\phi_{AD}</math> cycles, 10-bit resolution: 59 <math>\phi_{AD}</math> cycles</li> <li>• With sample and hold function 8-bit resolution: 28 <math>\phi_{AD}</math> cycles, 10-bit resolution: 33 <math>\phi_{AD}</math> cycles</li> </ul>

Note 1: Does not depend on use of sample and hold function.

Note 2: Without sample and hold function, set the  $\phi_{AD}$  frequency to 250kHz min.

With the sample and hold function, set the  $\phi_{AD}$  frequency to 1MHz min.

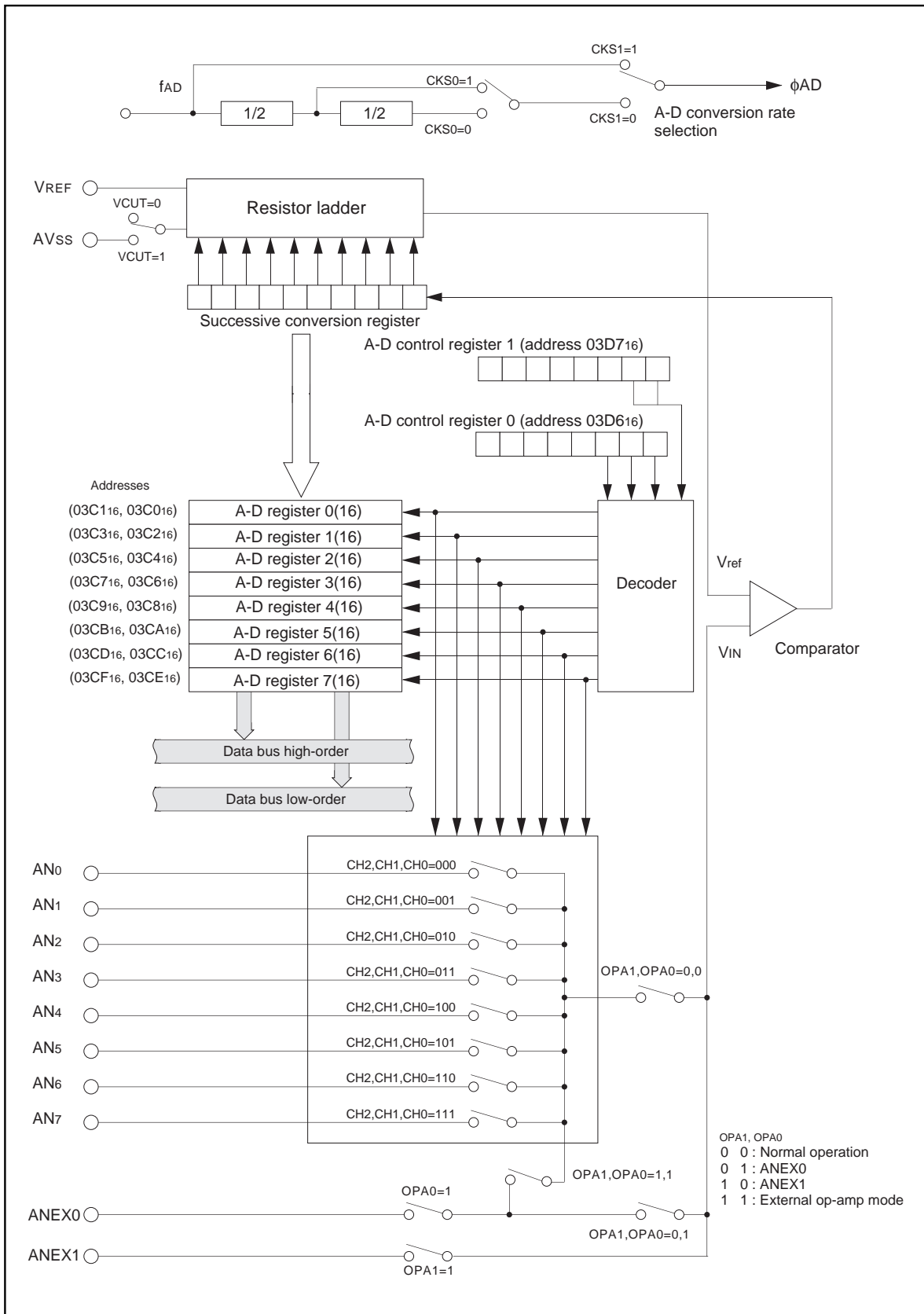


Figure 1.18.1. Block diagram of A-D converter

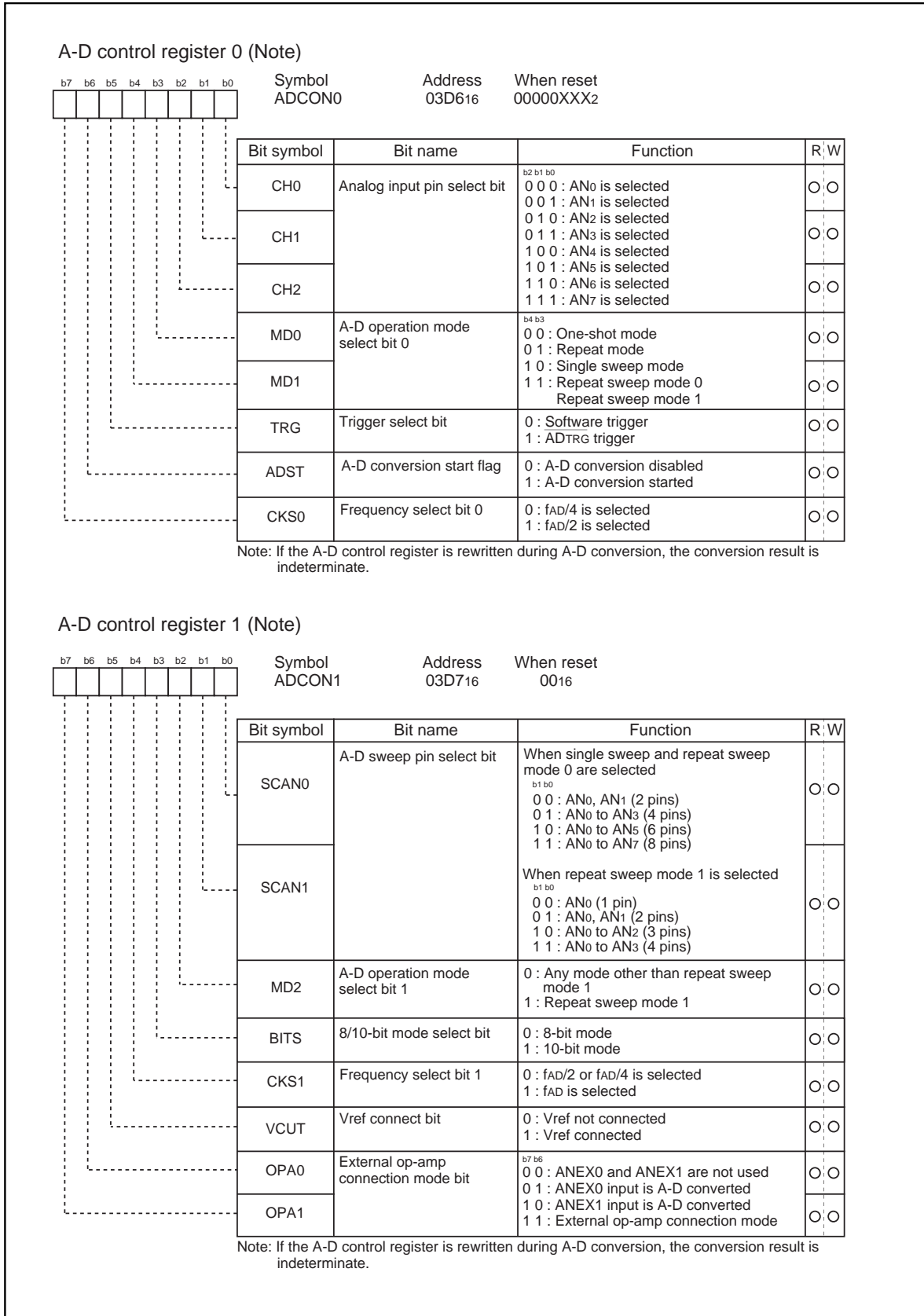


Figure 1.18.2. A-D converter-related registers (1)

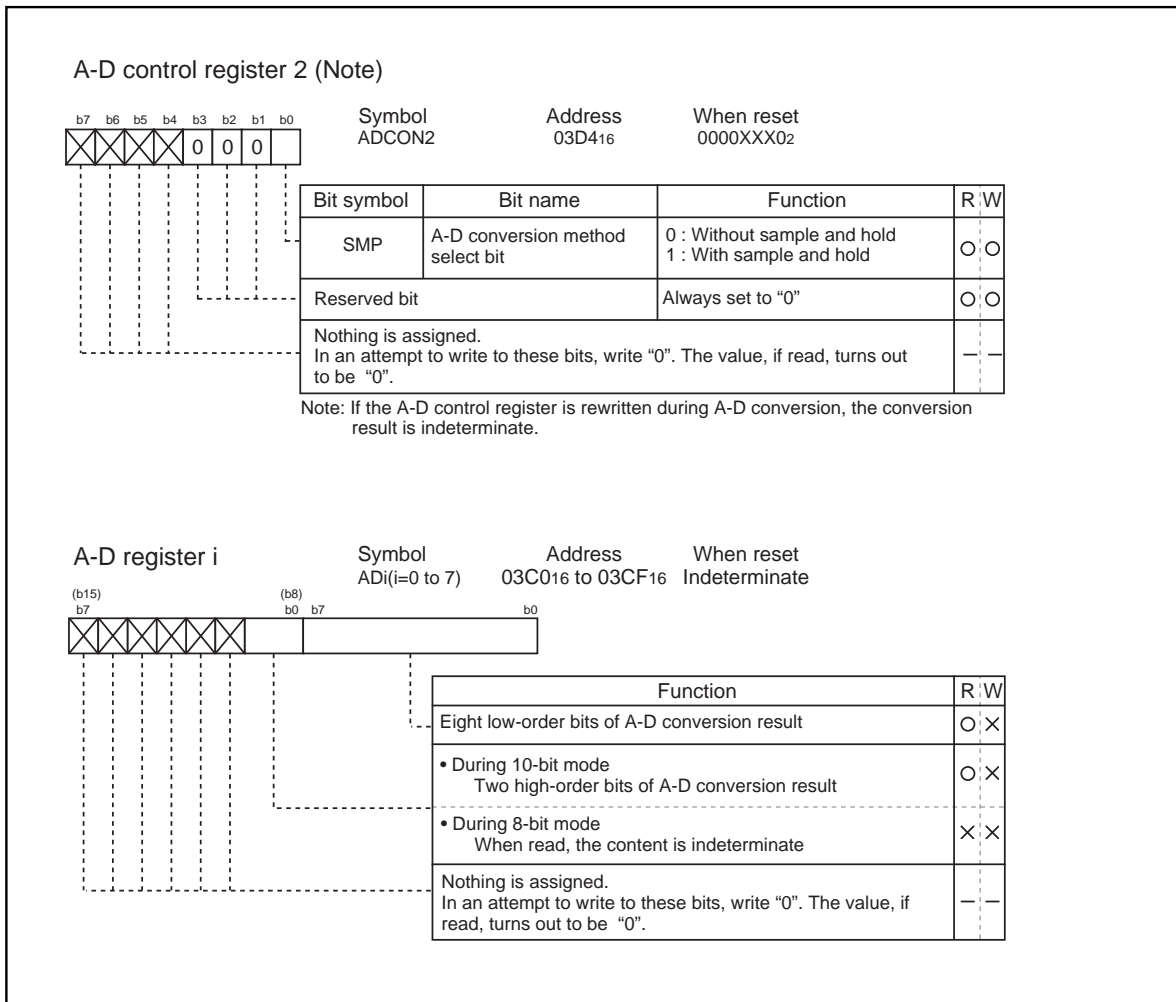


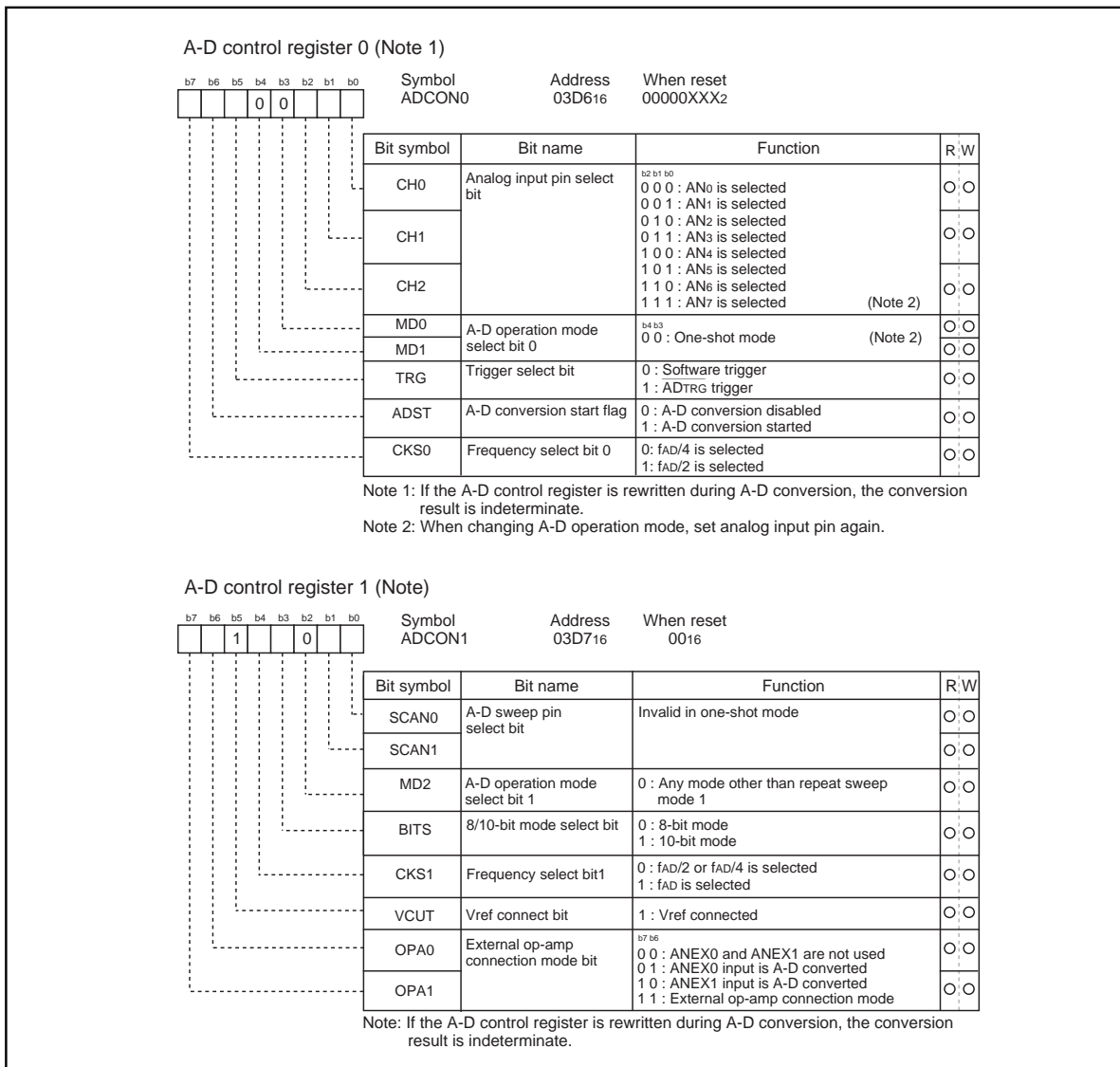
Figure 1.18.3. A-D converter-related registers (2)

**(1) One-shot mode**

In one-shot mode, the pin selected using the analog input pin select bit is used for one-shot A-D conversion. Table 1.18.2 shows the specifications of one-shot mode. Figure 1.18.4 shows the A-D control register in one-shot mode.

**Table 1.18.2. One-shot mode specifications**

Item	Specification
Function	The pin selected by the analog input pin select bit is used for one A-D conversion
Start condition	Writing "1" to A-D conversion start flag
Stop condition	<ul style="list-style-type: none"> <li>End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)</li> <li>Writing "0" to A-D conversion start flag</li> </ul>
Interrupt request generation timing	End of A-D conversion
Input pin	One of AN0 to AN7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin



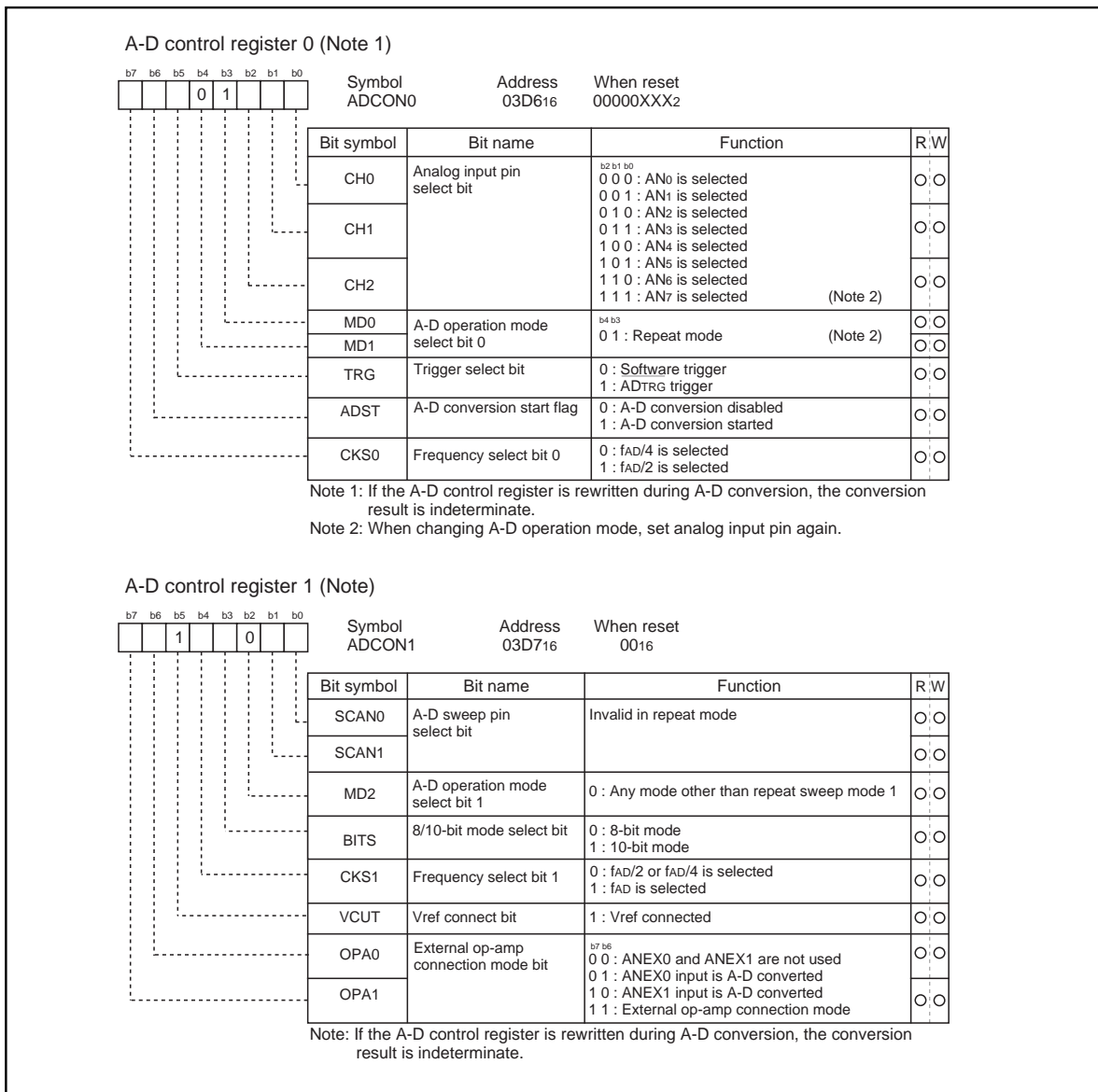
**Figure 1.18.4. A-D conversion register in one-shot mode**

**(2) Repeat mode**

In repeat mode, the pin selected using the analog input pin select bit is used for repeated A-D conversion. Table 1.18.3 shows the specifications of repeat mode. Figure 1.18.5 shows the A-D control register in repeat mode.

**Table 1.18.3. Repeat mode specifications**

Item	Specification
Function	The pin selected by the analog input pin select bit is used for repeated A-D conversion
Start condition	Writing "1" to A-D conversion start flag
Stop condition	Writing "0" to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	One of AN0 to AN7, as selected
Reading of result of A-D converter	Read A-D register corresponding to selected pin



**Figure 1.18.5. A-D conversion register in repeat mode**

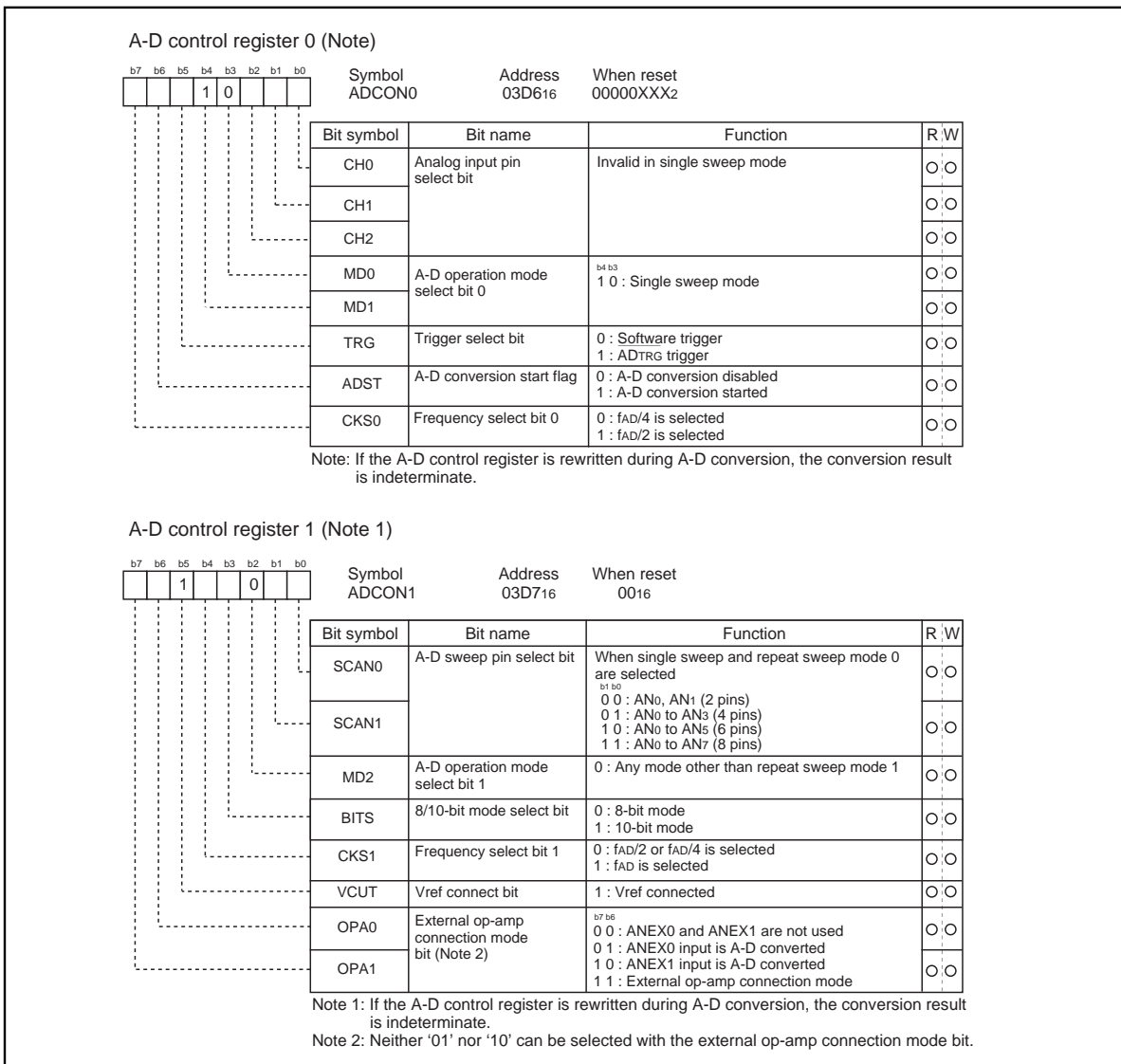


**(3) Single sweep mode**

In single sweep mode, the pins selected using the A-D sweep pin select bit are used for one-by-one A-D conversion. Table 1.18.4 shows the specifications of single sweep mode. Figure 1.18.6 shows the A-D control register in single sweep mode.

**Table 1.18.4. Single sweep mode specifications**

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for one-by-one A-D conversion
Start condition	Writing "1" to A-D converter start flag
Stop condition	<ul style="list-style-type: none"> <li>• End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)</li> <li>• Writing "0" to A-D conversion start flag</li> </ul>
Interrupt request generation timing	End of A-D conversion
Input pin	AN0 and AN1 (2 pins), AN0 to AN3 (4 pins), AN0 to AN5 (6 pins), or AN0 to AN7 (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin



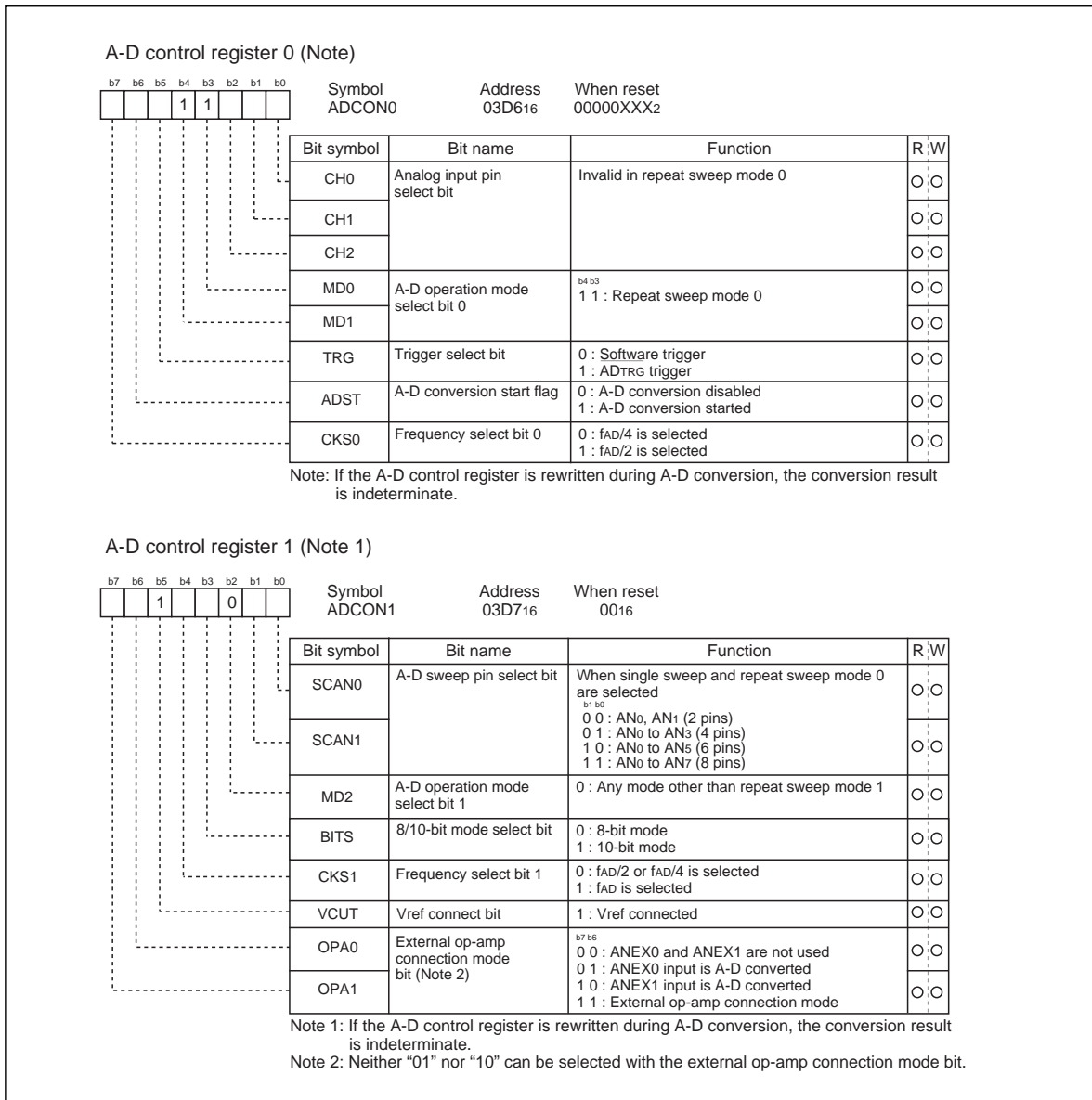
**Figure 1.18.6. A-D conversion register in single sweep mode**

**(4) Repeat sweep mode 0**

In repeat sweep mode 0, the pins selected using the A-D sweep pin select bit are used for repeat sweep A-D conversion. Table 1.18.5 shows the specifications of repeat sweep mode 0. Figure 1.18.7 shows the A-D control register in repeat sweep mode 0.

**Table 1.18.5. Repeat sweep mode 0 specifications**

Item	Specification
Function	The pins selected by the A-D sweep pin select bit are used for repeat sweep A-D conversion
Start condition	Writing "1" to A-D conversion start flag
Stop condition	Writing "0" to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN0 and AN1 (2 pins), AN0 to AN3 (4 pins), AN0 to AN5 (6 pins), or AN0 to AN7 (8 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)



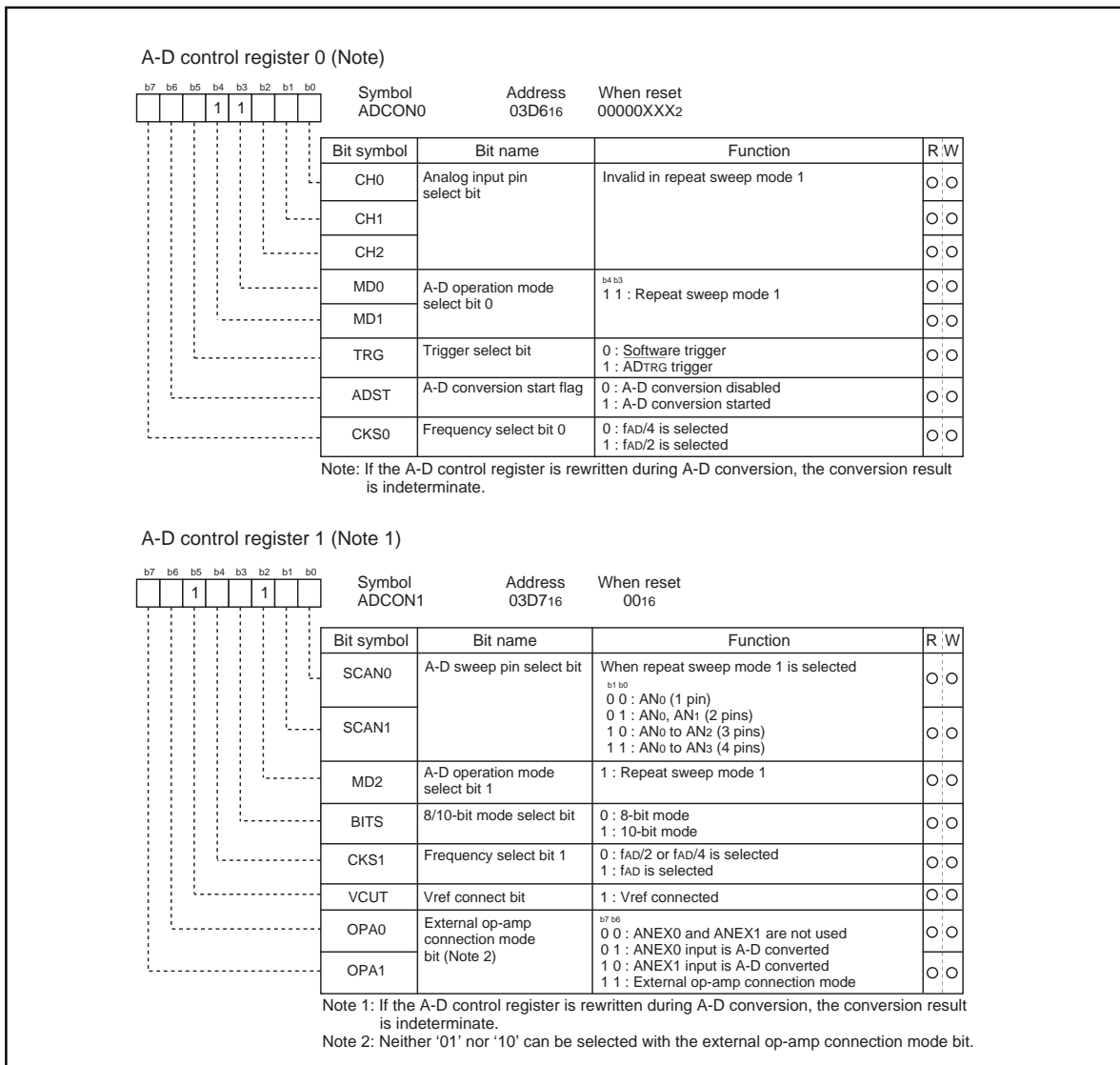
**Figure 1.18.7. A-D conversion register in repeat sweep mode 0**

**(5) Repeat sweep mode 1**

In repeat sweep mode 1, all pins are used for A-D conversion with emphasis on the pin or pins selected using the A-D sweep pin select bit. Table 1.18.6 shows the specifications of repeat sweep mode 1. Figure 1.18.8 shows the A-D control register in repeat sweep mode 1.

**Table 1.18.6. Repeat sweep mode 1 specifications**

Item	Specification
Function	All pins perform repeat sweep A-D conversion, with emphasis on the pin or pins selected by the A-D sweep pin select bit Example : AN <sub>0</sub> selected AN <sub>0</sub> → AN <sub>1</sub> → AN <sub>0</sub> → AN <sub>2</sub> → AN <sub>0</sub> → AN <sub>3</sub> , etc
Start condition	Writing "1" to A-D conversion start flag
Stop condition	Writing "0" to A-D conversion start flag
Interrupt request generation timing	None generated
Input pin	AN <sub>0</sub> (1 pin), AN <sub>0</sub> and AN <sub>1</sub> (2 pins), AN <sub>0</sub> to AN <sub>2</sub> (3 pins), AN <sub>0</sub> to AN <sub>3</sub> (4 pins)
Reading of result of A-D converter	Read A-D register corresponding to selected pin (at any time)



**Figure 1.18.8. A-D conversion register in repeat sweep mode 1**

**(a) Sample and hold**

Sample and hold is selected by setting bit 0 of the A-D control register 2 (address 03D416) to "1". When sample and hold is selected, the rate of conversion of each pin increases. As a result, a  $28 \phi_{AD}$  cycle is achieved with 8-bit resolution and  $33 \phi_{AD}$  with 10-bit resolution. Sample and hold can be selected in all modes. However, in all modes, be sure to specify before starting A-D conversion whether sample and hold is to be used.

**(b) Extended analog input pins**

In one-shot mode and repeat mode, the input via the extended analog input pins ANEX0 and ANEX1 can also be converted from analog to digital.

When bit 6 of the A-D control register 1 (address 03D716) is "1" and bit 7 is "0", input via ANEX0 is converted from analog to digital. The result of conversion is stored in A-D register 0.

When bit 6 of the A-D control register 1 (address 03D716) is "0" and bit 7 is "1", input via ANEX1 is converted from analog to digital. The result of conversion is stored in A-D register 1.

**(c) External operation amp connection mode**

In this mode, multiple external analog inputs via the extended analog input pins, ANEX0 and ANEX1, can be amplified together by just one operation amp and used as the input for A-D conversion.

When bit 6 of the A-D control register 1 (address 03D716) is "1" and bit 7 is "1", input via AN0 to AN7 is output from ANEX0. The input from ANEX1 is converted from analog to digital and the result stored in the corresponding A-D register. The speed of A-D conversion depends on the response of the external operation amp. Do not connect the ANEX0 and ANEX1 pins directly. Figure 1.18.9 is an example of how to connect the pins in external operation amp mode.

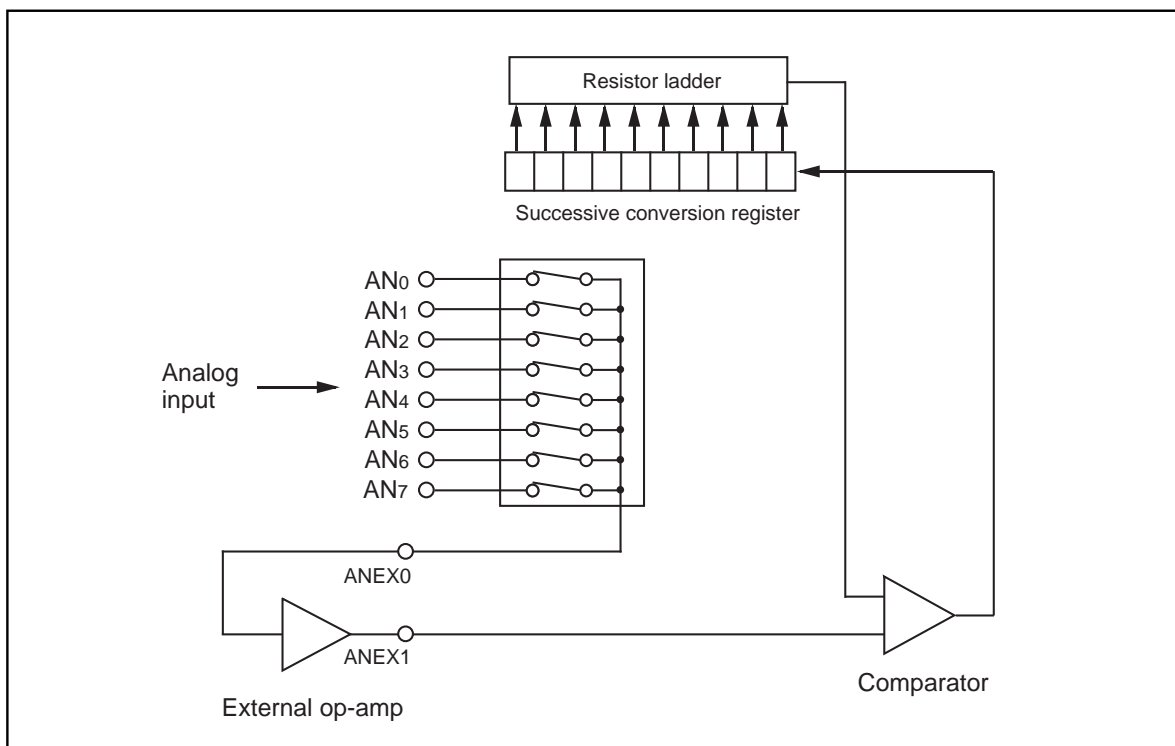


Figure 1.18.9. Example of external op-amp connection mode

## D-A Converter

**D-A Converter**

This is an 8-bit, R-2R type D-A converter. The microcomputer contains two independent D-A converters of this type.

D-A conversion is performed when a value is written to the corresponding D-A register. Bits 0 and 1 (D-A output enable bits) of the D-A control register decide if the result of conversion is to be output. Do not set the target port to output mode if D-A conversion is to be performed.

Output analog voltage (V) is determined by a set value (n : decimal) in the D-A register.

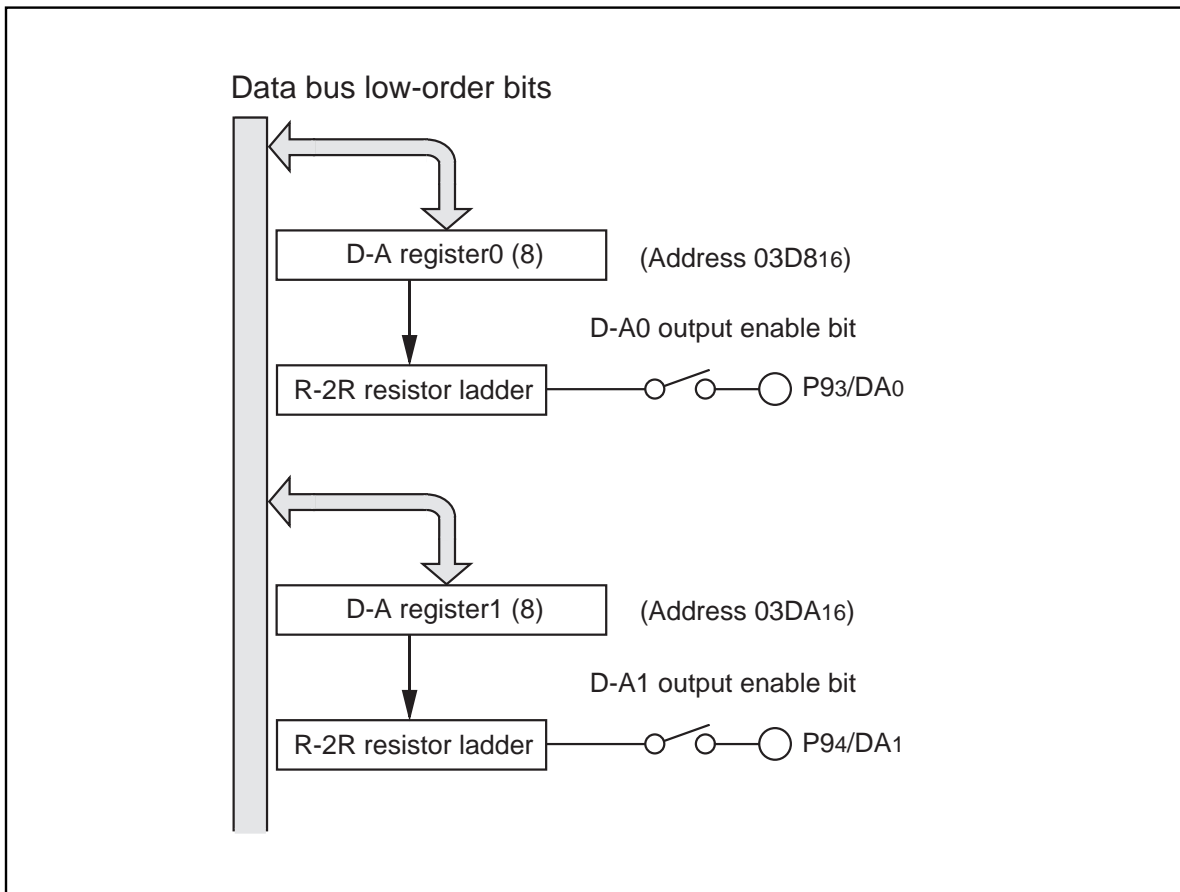
$$V = V_{REF} \times n / 256 \quad (n = 0 \text{ to } 255)$$

$V_{REF}$  : reference voltage

Table 1.19.1 lists the performance of the D-A converter. Figure 1.19.1 shows the block diagram of the D-A converter. Figure 1.19.2 shows the D-A control register. Figure 1.19.3 shows the D-A converter equivalent circuit.

**Table 1.19.1. Performance of D-A converter**

Item	Performance
Conversion method	R-2R method
Resolution	8 bits
Analog output pin	2 channels



**Figure 1.19.1. Block diagram of D-A converter**

D-A Converter

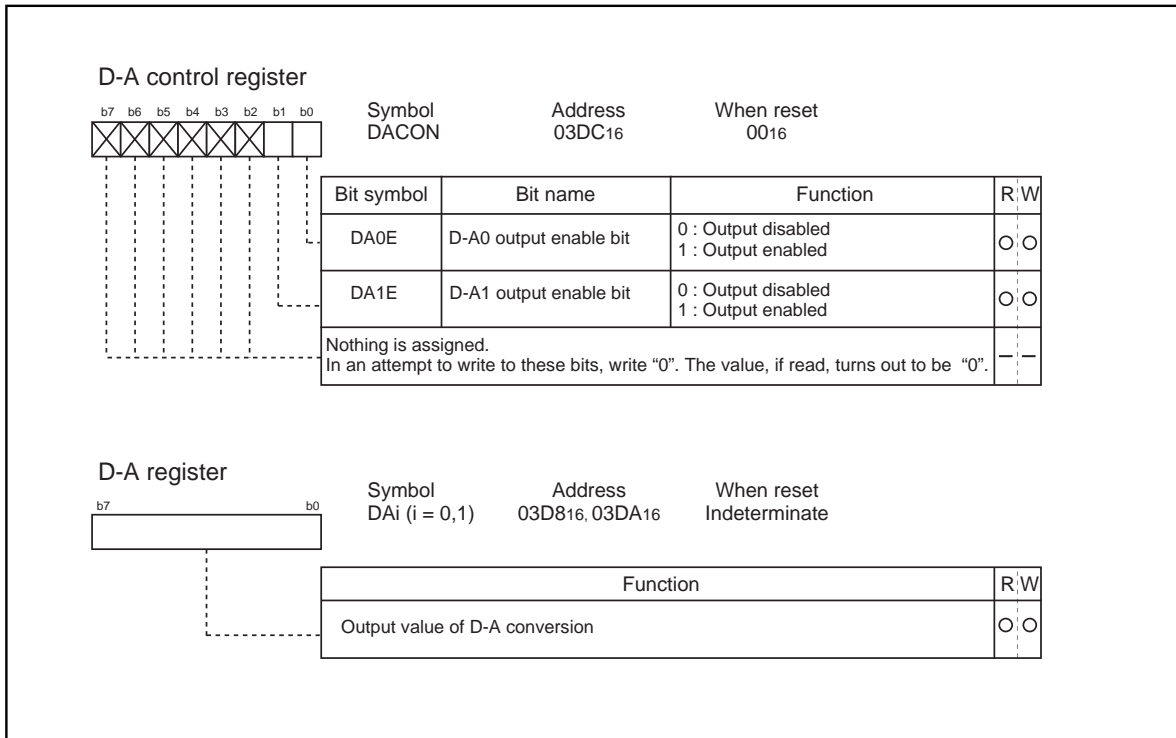


Figure 1.19.2. D-A control register

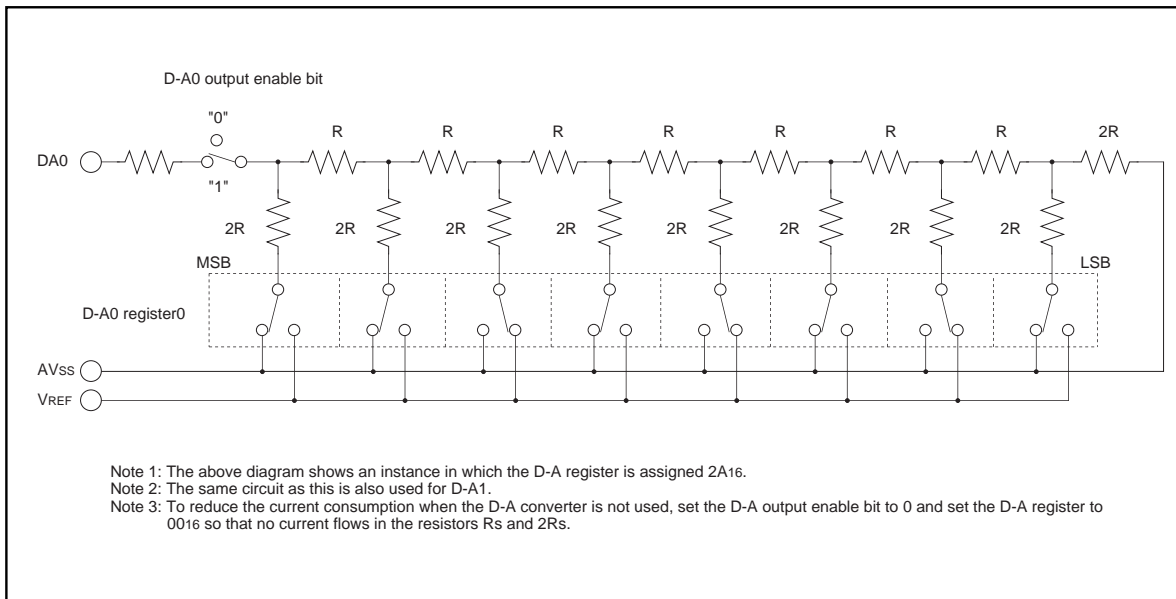


Figure 1.19.3. D-A converter equivalent circuit

### CRC Calculation Circuit

The Cyclic Redundancy Check (CRC) calculation circuit detects an error in data blocks. The microcomputer uses a generator polynomial of CRC\_CCITT ( $X^{16} + X^{12} + X^5 + 1$ ) to generate CRC code. The CRC code is a 16-bit code generated for a block of a given data length in multiples of 8 bits. The CRC code is set in a CRC data register each time one byte of data is transferred to a CRC input register after writing an initial value into the CRC data register. Generation of CRC code for one byte of data is completed in two machine cycles.

Figure 1.20.1 shows the block diagram of the CRC circuit. Figure 1.20.2 shows the CRC-related registers. Figure 1.20.3 shows the calculation example using the CRC calculation circuit

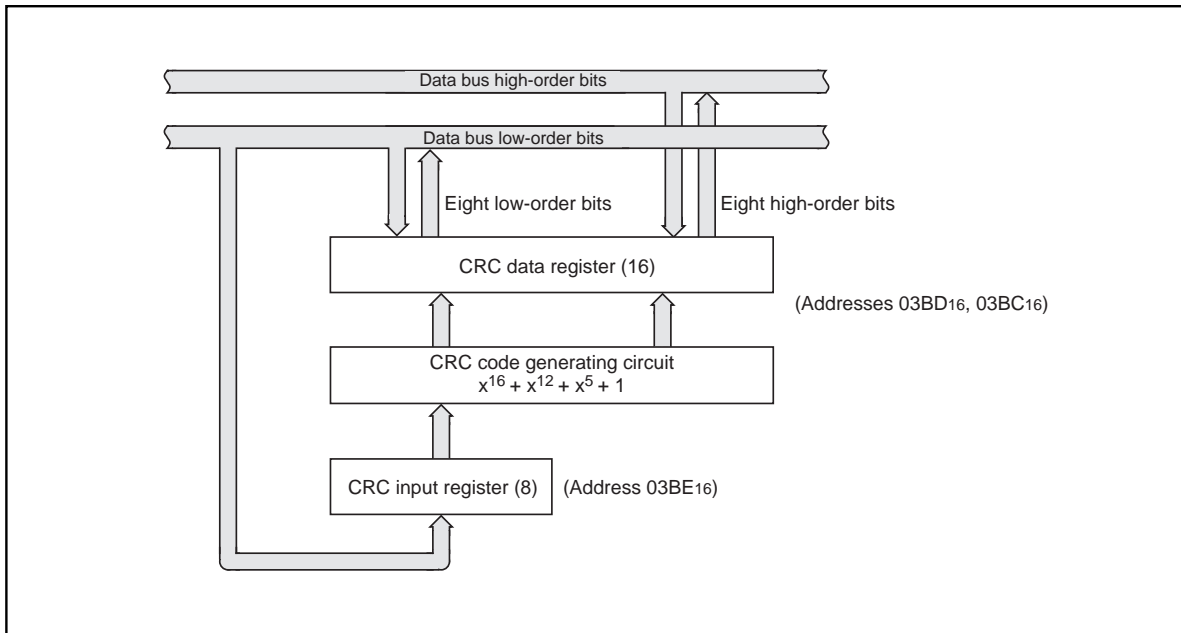


Figure 1.20.1. Block diagram of CRC circuit

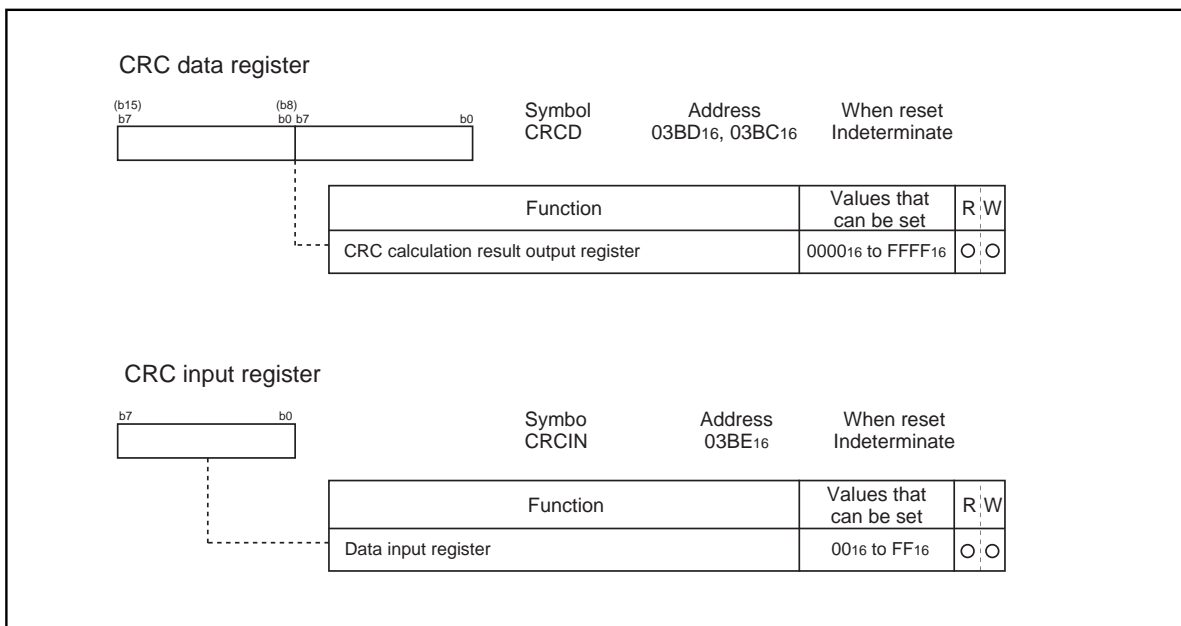


Figure 1.20.2. CRC-related registers

CRC

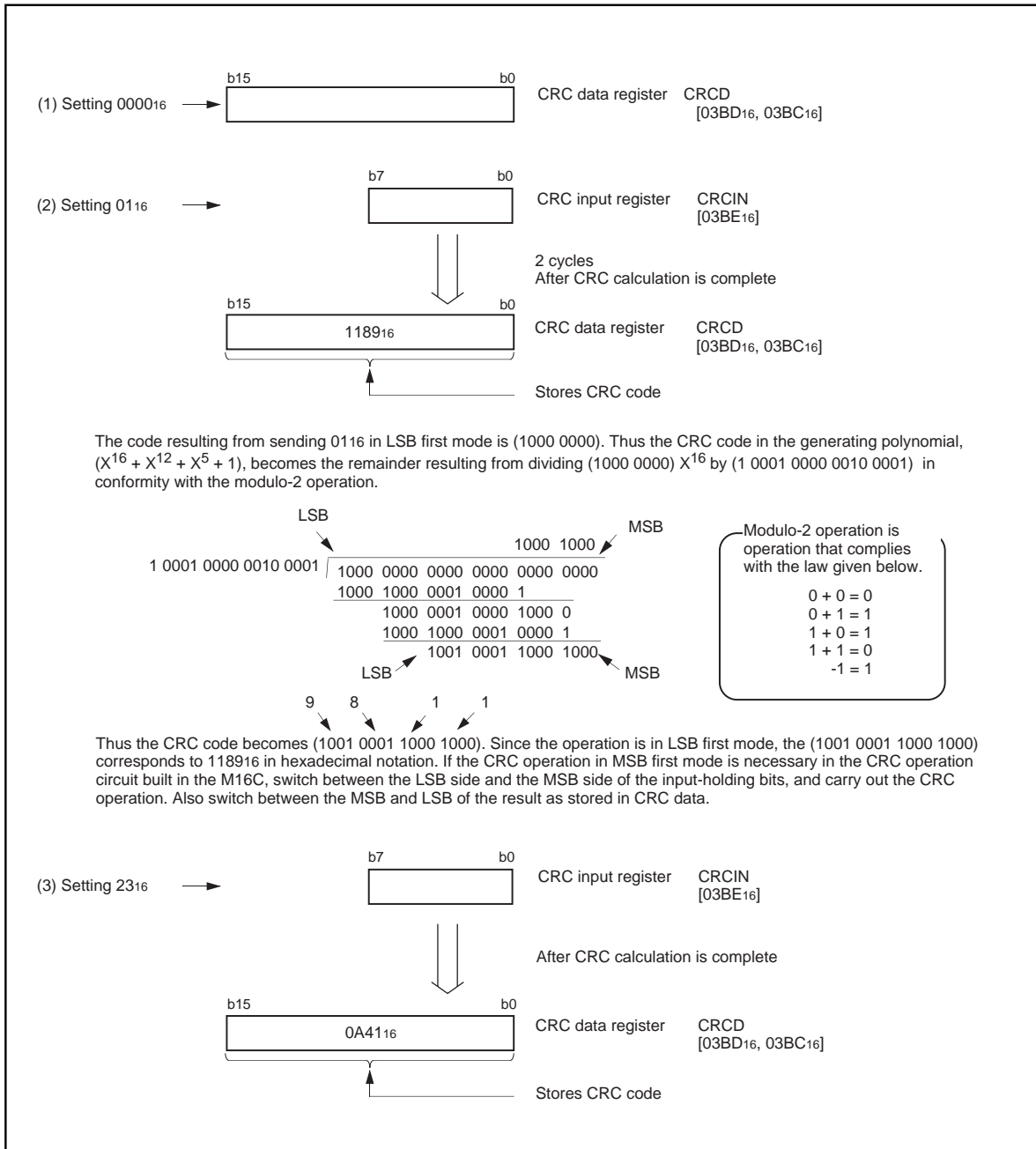


Figure 1.20.3. Calculation example using the CRC calculation circuit



## Programmable I/O Ports

There are 87 programmable I/O ports: P0 to P10 (excluding P85). Each port can be set independently for input or output using the direction register. A pull-up resistance for each block of 4 ports can be set. P85 is an input-only port and has no built-in pull-up resistance.

Figures 1.21.1 to 1.21.4 show the programmable I/O ports. Figure 1.21.5 shows the I/O pins.

Each pin functions as a programmable I/O port and as the I/O for the built-in peripheral devices.

To use the pins as the inputs for the built-in peripheral devices, set the direction register of each pin to input mode. When the pins are used as the outputs for the built-in peripheral devices (other than the D-A converter), they function as outputs regardless of the contents of the direction registers. When pins are to be used as the outputs for the D-A converter, do not set the direction registers to output mode. See the descriptions of the respective functions for how to set up the built-in peripheral devices.

### (1) Direction registers

Figure 1.21.6 shows the direction registers.

These registers are used to choose the direction of the programmable I/O ports. Each bit in these registers corresponds one for one to each I/O pin.

Note: There is no direction register bit for P85.

### (2) Port registers

Figure 1.21.7 shows the port registers.

These registers are used to write and read data for input and output to and from an external device. A port register consists of a port latch to hold output data and a circuit to read the status of a pin. Each bit in port registers corresponds one for one to each I/O pin.

### (3) Pull-up control registers

Figure 1.21.8 shows the pull-up control registers.

The pull-up control register can be set to apply a pull-up resistance to each block of 4 ports. When ports are set to have a pull-up resistance, the pull-up resistance is connected only when the direction register is set for input.

However, in memory expansion mode and microprocessor mode, the pull-up control register of P0 to P5 is invalid.

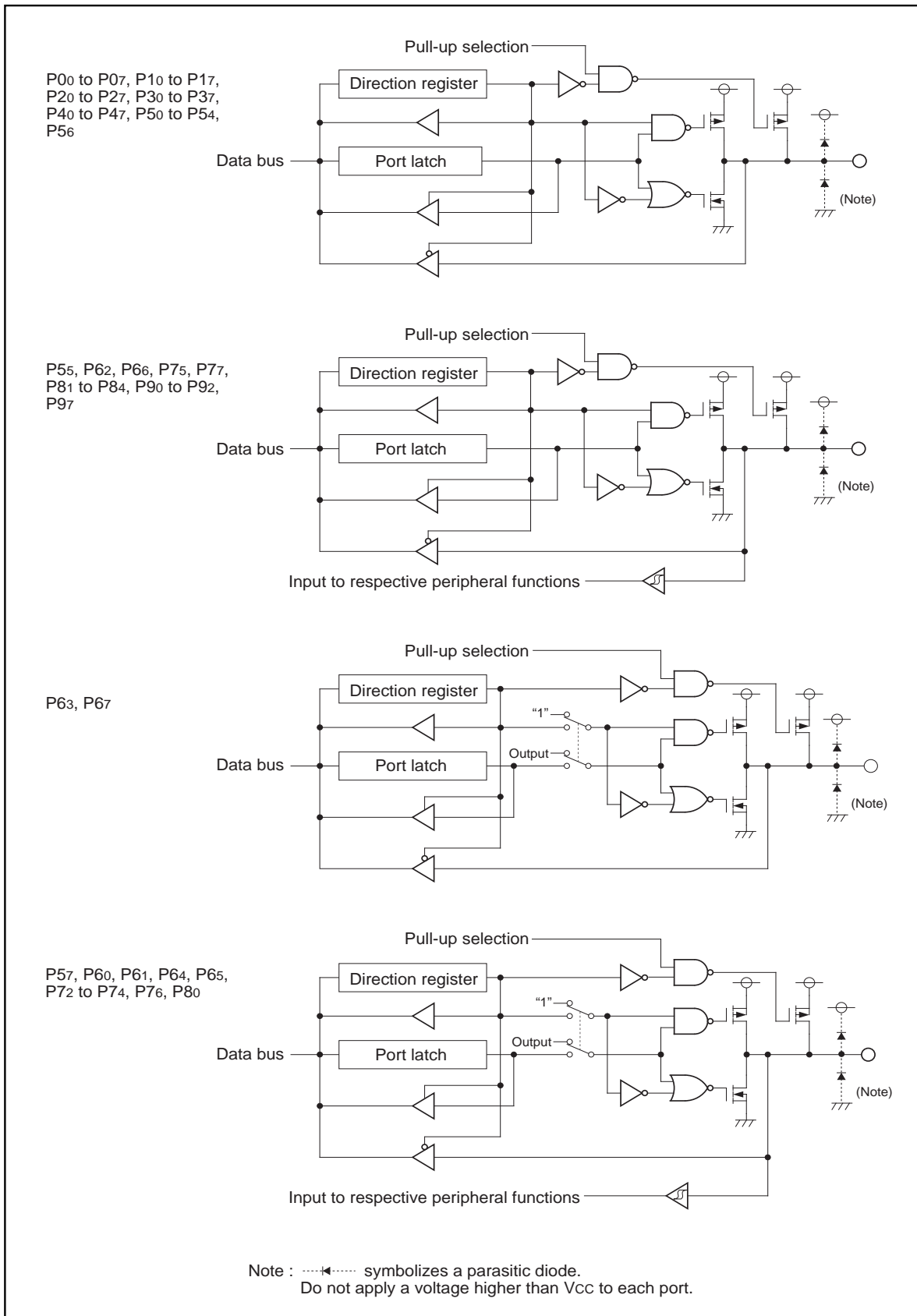


Figure 1.21.1. Programmable I/O ports (1)

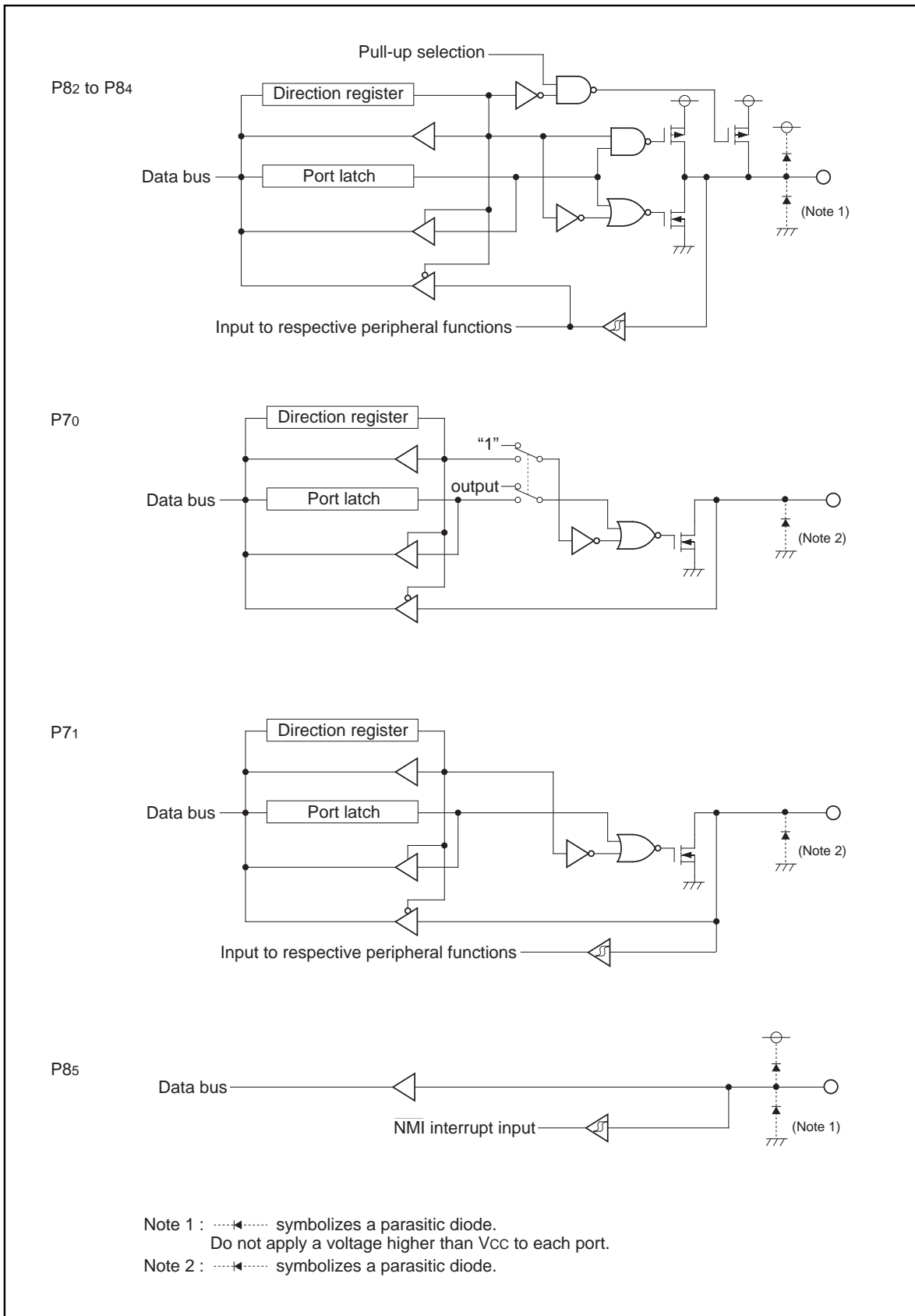


Figure 1.21.2. Programmable I/O ports (2)

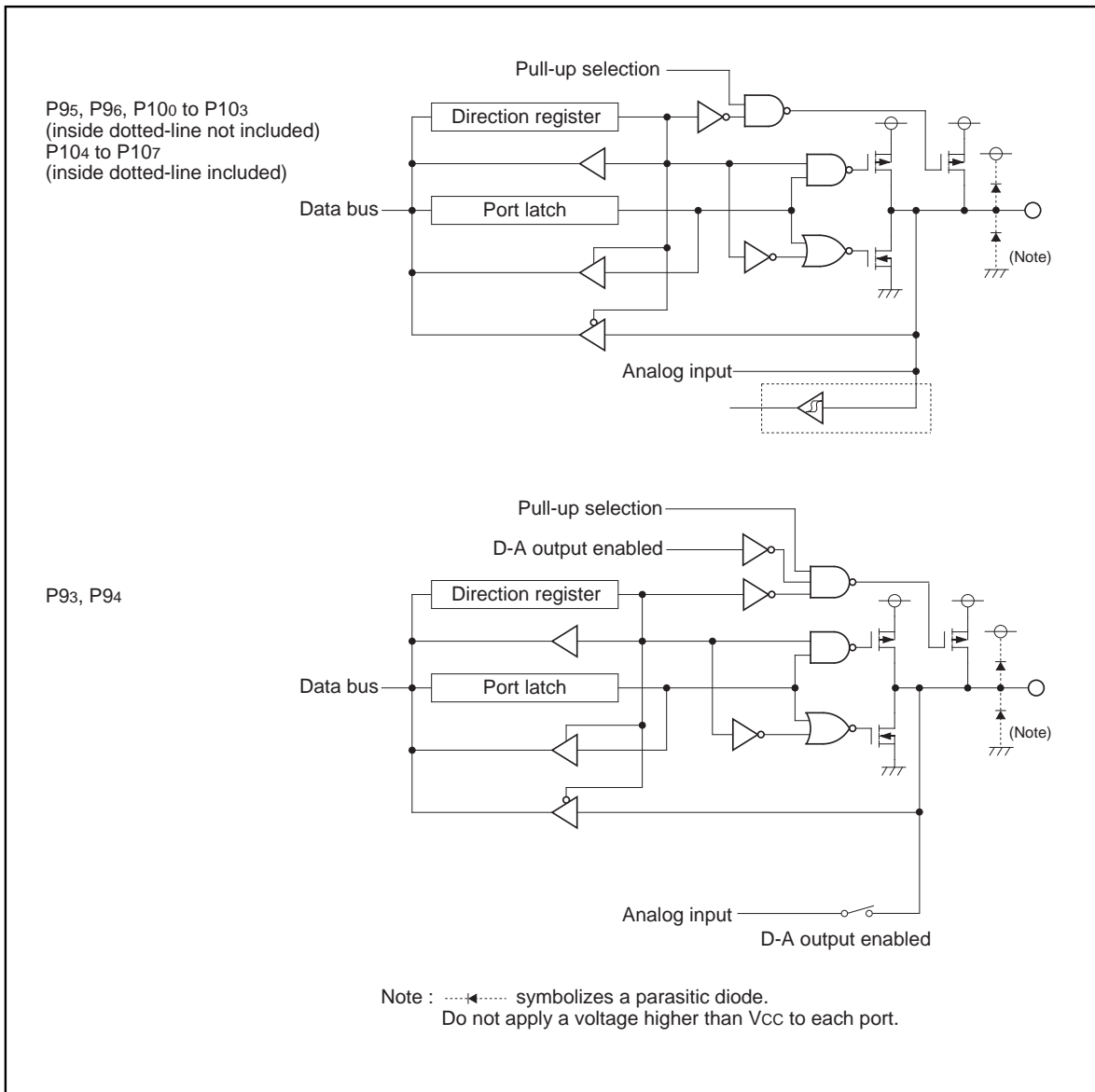


Figure 1.21.3. Programmable I/O ports (3)

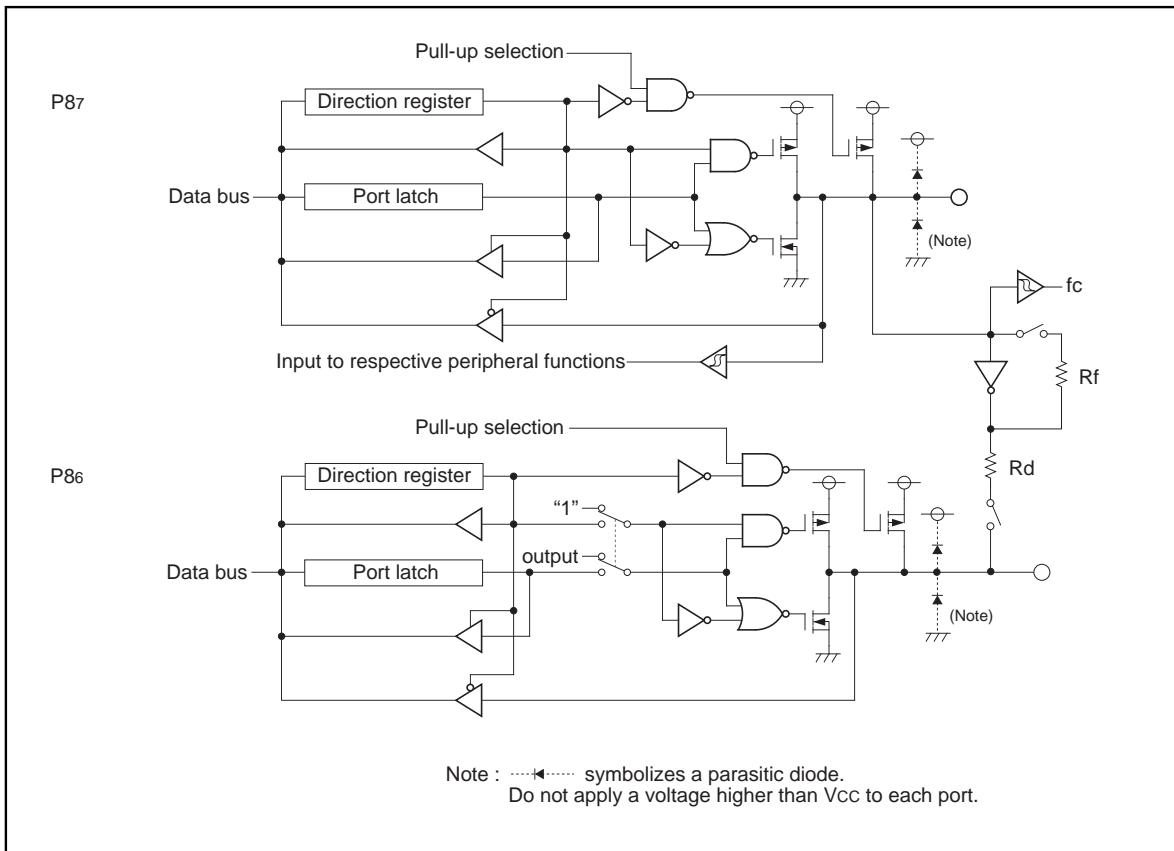


Figure 1.21.4. Programmable I/O ports (4)

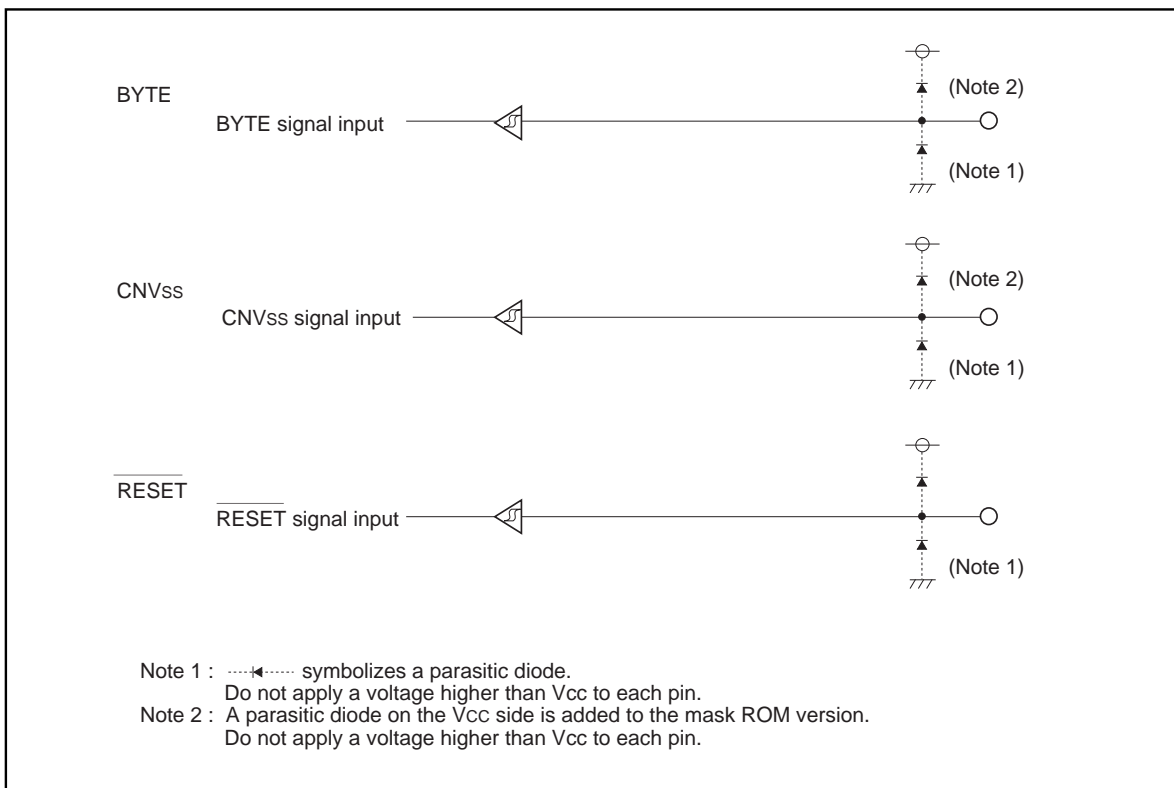


Figure 1.21.5. I/O pins

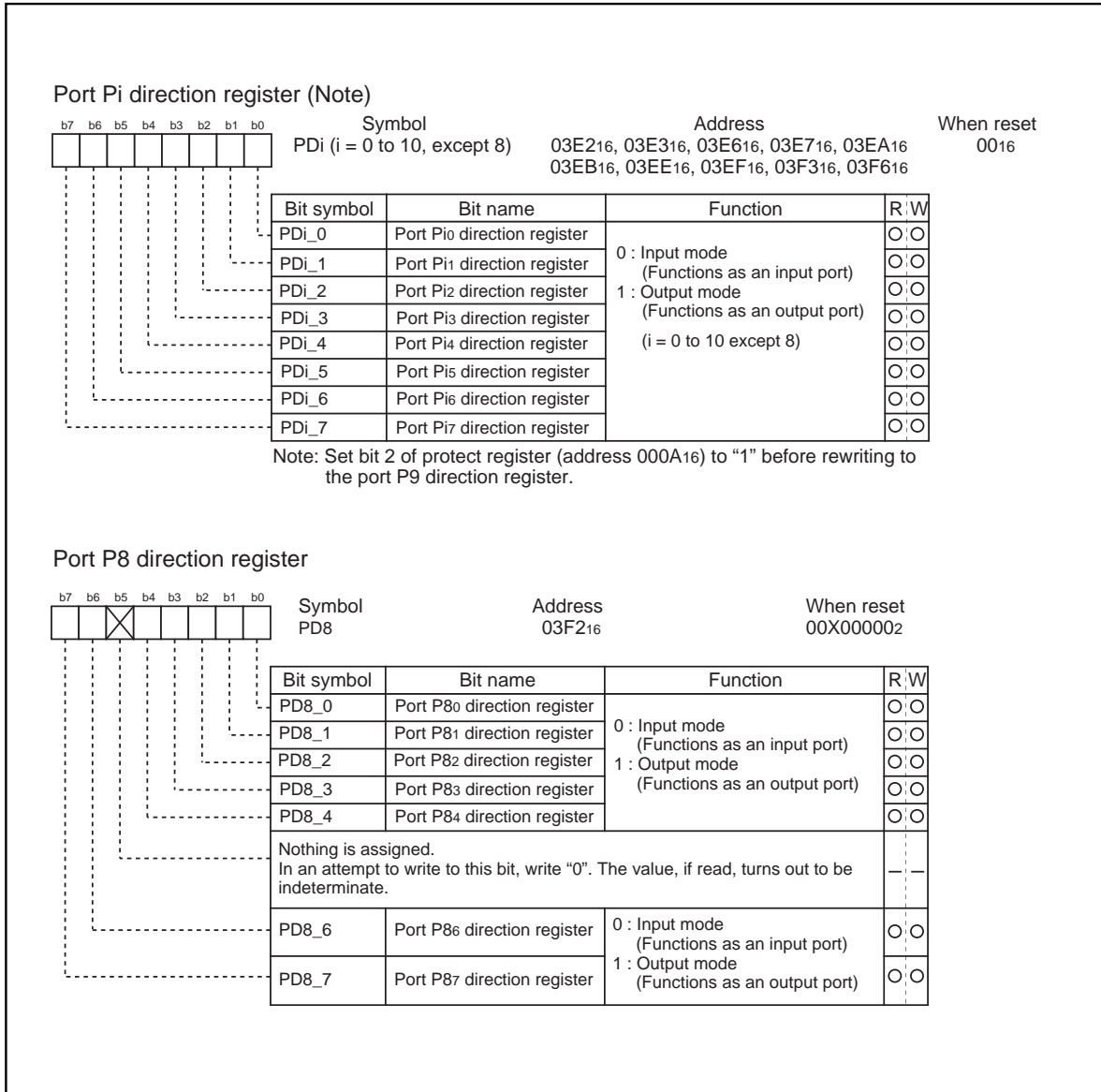


Figure 1.21.6. Direction register

Programmable I/O Port

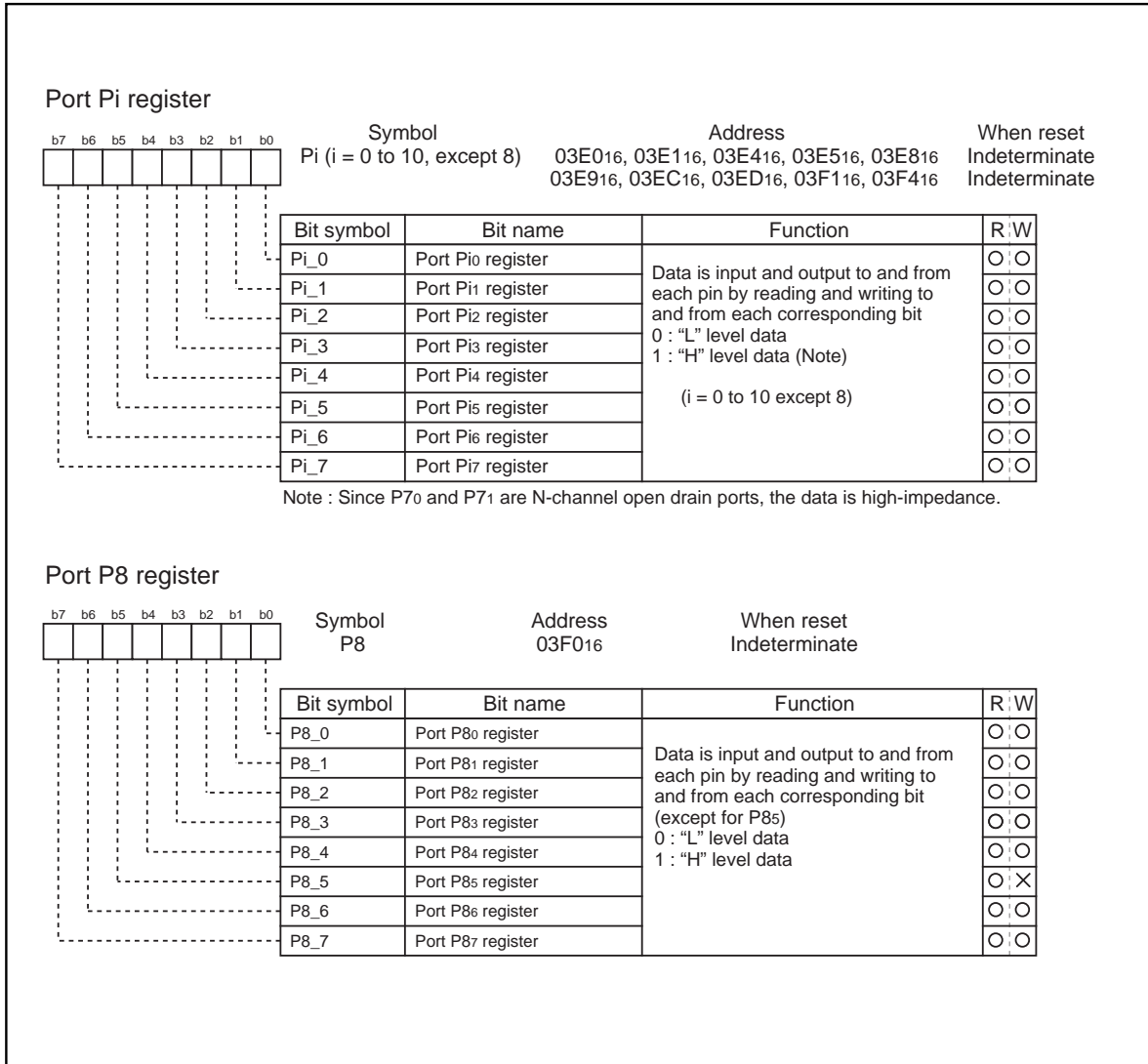


Figure 1.21.7. Port register

## Programmable I/O Port

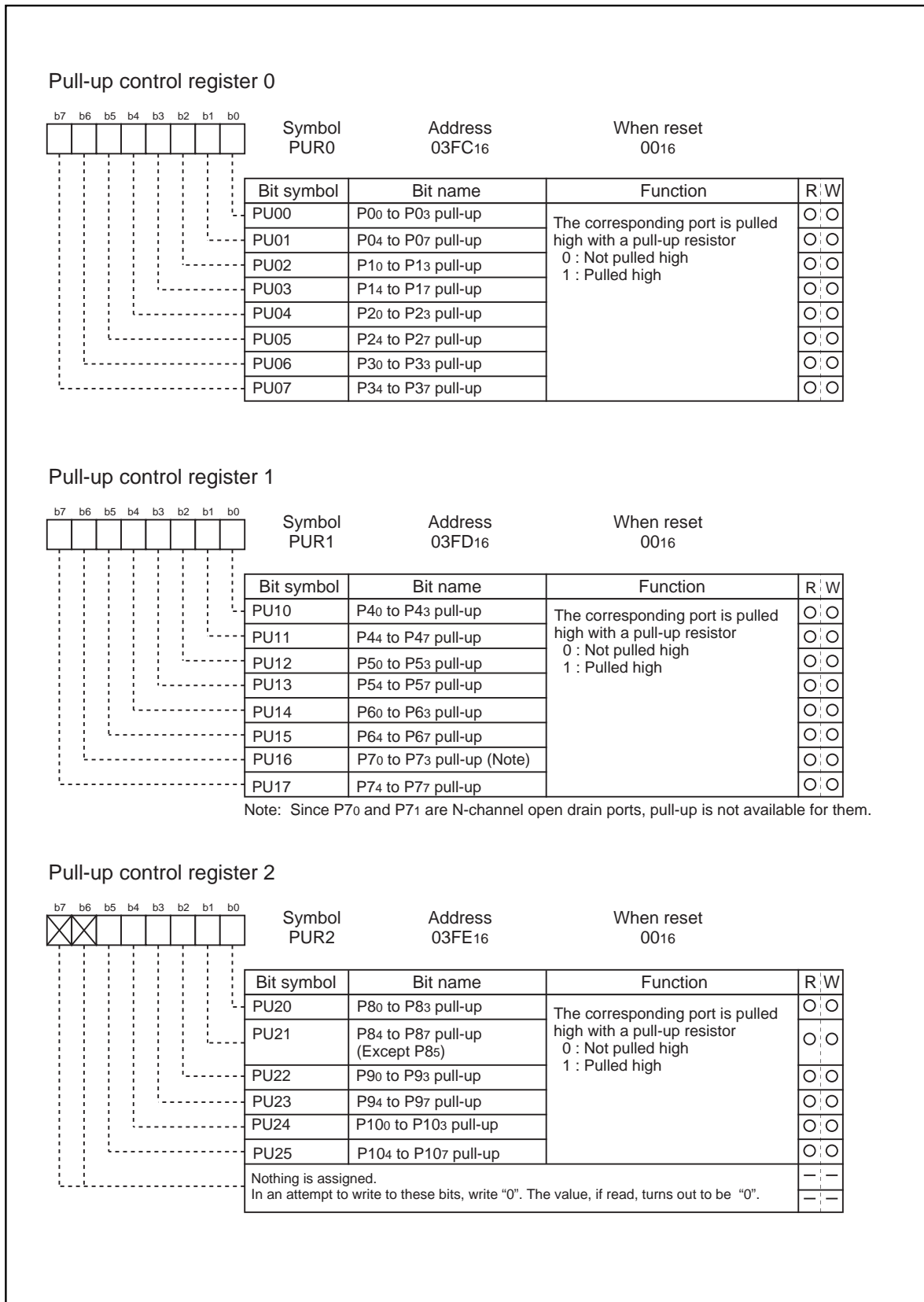


Figure 1.21.8. Pull-up control register



**Table 1.21.1. Example connection of unused pins in single-chip mode**

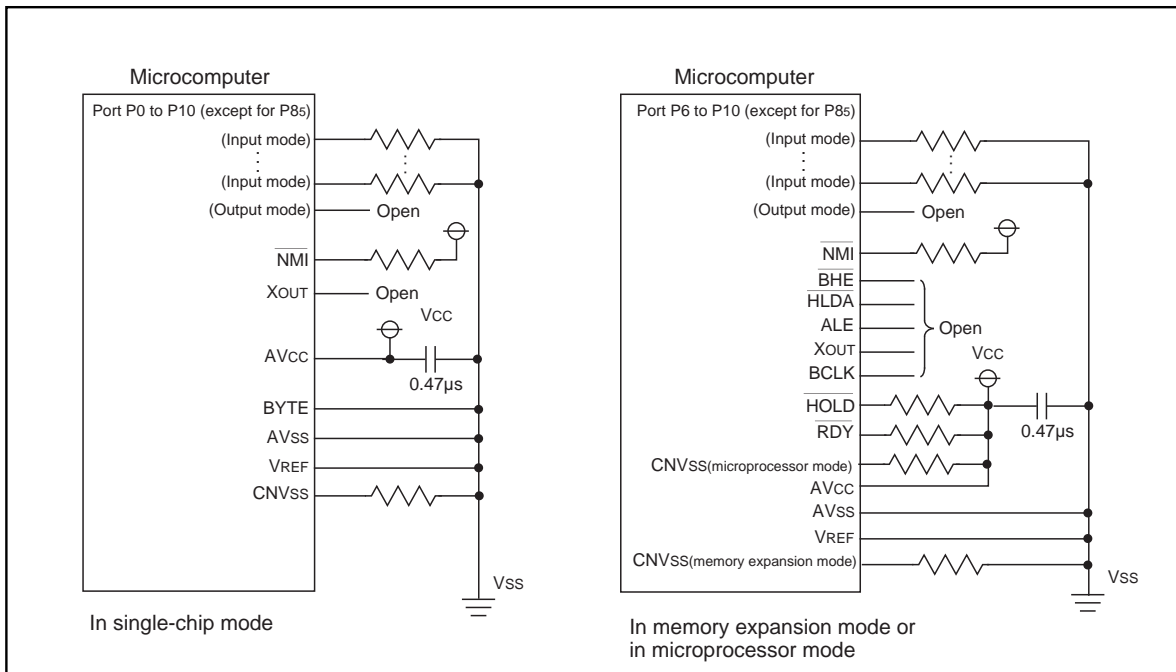
Pin name	Connection
Ports P0 to P10 (excluding P8s)	After setting for input mode, connect every pin to VSS via a resistor; or after setting for output mode, leave these pins open.
XOUT (Note)	Open
NMI	Connect via resistor to VCC (pull-up)
AVCC	Connect to VCC
AVSS, VREF, BYTE	Connect to VSS
CNVSS	Connect via resistor to VSS (pull-down)

Note: With external clock input to XIN pin.

**Table 1.21.2. Example connection of unused pins in memory expansion mode and microprocessor mode**

Pin name	Connection
Ports P6 to P10 (excluding P8s)	After setting for input mode, connect every pin to VSS or VCC via a resistor; or after setting for output mode, leave these pins open.
BHE, ALE, HLDA, XOUT(Note), BCLK	Open
HOLD, RDY, NMI	Connect via resistor to VCC (pull-up)
AVCC	Connect to VCC
AVSS, VREF	Connect to VSS
CNVSS	Connect via resistor to VSS (pull-down) in the memory expansion mode Connect via resistor to VCC (pull-up) in the microprocessor mode

Note: With external clock input to XIN pin.



**Figure 1.21.9. Example connection of unused pins**

## Usage Precaution

### Timer A (timer mode)

- (1) Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing gets "FFFF<sub>16</sub>". Reading the timer Ai register after setting a value in the timer Ai register with a count halted but before the counter starts counting gets a proper value.

### Timer A (event counter mode)

- (1) Reading the timer Ai register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Ai register with the reload timing gets "FFFF<sub>16</sub>" by underflow or "0000<sub>16</sub>" by overflow. Reading the timer Ai register after setting a value in the timer Ai register with a count halted but before the counter starts counting gets a proper value.
- (2) When stop counting in free run type, set timer again.

### Timer A (one-shot timer mode)

- (1) Setting the count start flag to "0" while a count is in progress causes as follows:
  - The counter stops counting and a content of reload register is reloaded.
  - The TAIOUT pin outputs "L" level.
  - The interrupt request generated and the timer Ai interrupt request bit goes to "1".
- (2) The timer Ai interrupt request bit goes to "1" if the timer's operation mode is set using any of the following procedures:
  - Selecting one-shot timer mode after reset.
  - Changing operation mode from timer mode to one-shot timer mode.
  - Changing operation mode from event counter mode to one-shot timer mode.Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to "0" after the above listed changes have been made.

### Timer A (pulse width modulation mode)

- (1) The timer Ai interrupt request bit becomes "1" if setting operation mode of the timer in compliance with any of the following procedures:
  - Selecting PWM mode after reset.
  - Changing operation mode from timer mode to PWM mode.
  - Changing operation mode from event counter mode to PWM mode.Therefore, to use timer Ai interrupt (interrupt request bit), set timer Ai interrupt request bit to "0" after the above listed changes have been made.
- (2) Setting the count start flag to "0" while PWM pulses are being output causes the counter to stop counting. If the TAIOUT pin is outputting an "H" level in this instance, the output level goes to "L", and the timer Ai interrupt request bit goes to "1". If the TAIOUT pin is outputting an "L" level in this instance, the level does not change, and the timer Ai interrupt request bit does not becomes "1".

### Timer B (timer mode, event counter mode)

- (1) Reading the timer Bi register while a count is in progress allows reading, with arbitrary timing, the value of the counter. Reading the timer Bi register with the reload timing gets "FFFF<sub>16</sub>". Reading the timer Bi register after setting a value in the timer Bi register with a count halted but before the counter starts counting gets a proper value.

### Timer B (pulse period/pulse width measurement mode)

- (1) If changing the measurement mode select bit is set after a count is started, the timer Bi interrupt request bit goes to "1".
- (2) When the first effective edge is input after a count is started, an indeterminate value is transferred to the reload register. At this time, timer Bi interrupt request is not generated.

### A-D Converter

- (1) Write to each bit (except bit 6) of A-D control register 0, to each bit of A-D control register 1, and to bit 0 of A-D control register 2 when A-D conversion is stopped (before a trigger occurs).  
In particular, when the Vref connection bit is changed from "0" to "1", start A-D conversion after an elapse of 1  $\mu$ s or longer.
- (2) When changing A-D operation mode, select analog input pin again.
- (3) Using one-shot mode or single sweep mode  
Read the correspondence A-D register after confirming A-D conversion is finished. (It is known by A-D conversion interrupt request bit.)
- (4) Using repeat mode, repeat sweep mode 0 or repeat sweep mode 1  
Use the undivided main clock as the internal CPU clock.

### Stop Mode and Wait Mode

- (1) When returning from stop mode by hardware reset,  $\overline{\text{RESET}}$  pin must be set to "L" level until main clock oscillation is stabilized.
- (2) When switching to either wait mode or stop mode, instructions occupying four bytes either from the WAIT instruction or from the instruction that sets the every-clock stop bit to "1" within the instruction queue are prefetched and then the program stops. So put at least four NOPs in succession either to the WAIT instruction or to the instruction that sets the every-clock stop bit to 1.

### Interrupts

- (1) Reading address 00000<sub>16</sub>
  - When maskable interrupt is occurred, CPU read the interrupt information (the interrupt number and interrupt request level) in the interrupt sequence.  
The interrupt request bit of the certain interrupt written in address 00000<sub>16</sub> will then be set to "0".  
Reading address 00000<sub>16</sub> by software sets enabled highest priority interrupt source request bit to "0".  
Though the interrupt is generated, the interrupt routine may not be executed.  
Do not read address 00000<sub>16</sub> by software.
- (2) Setting the stack pointer
  - The value of the stack pointer immediately after reset is initialized to 0000<sub>16</sub>. Accepting an interrupt before setting a value in the stack pointer may become a factor of runaway. Be sure to set a value in the stack pointer before accepting an interrupt.  
When using the  $\overline{\text{NMI}}$  interrupt, initialize the stack point at the beginning of a program. Concerning the first instruction immediately after reset, generating any interrupts including the  $\overline{\text{NMI}}$  interrupt is prohibited.

## Usage precaution

(3) The  $\overline{\text{NMI}}$  interrupt

- As for the  $\overline{\text{NMI}}$  interrupt pin, an interrupt cannot be disabled. Connect it to the VCC pin via a resistor (pull-up) if unused. Be sure to work on it.
- Do not get either into stop mode or into wait mode with the  $\overline{\text{NMI}}$  pin set to "L".

## (4) External interrupt

- When the polarity of the  $\overline{\text{INT0}}$  to  $\overline{\text{INT2}}$  pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0".

## (5) Rewrite the interrupt control register

- To rewrite the interrupt control register, do so at a point that does not generate the interrupt request for that register. If there is possibility of the interrupt request occur, rewrite the interrupt control register after the interrupt is disabled. The program examples are described as follow:

**Example 1:**

```

INT_SWITCH1:
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  NOP                               ; Four NOP instructions are required when using HOLD function.
  NOP
  FSET  I           ; Enable interrupts.

```

**Example 2:**

```

INT_SWITCH2:
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  MOV.W MEM, R0    ; Dummy read.
  FSET  I           ; Enable interrupts.

```

**Example 3:**

```

INT_SWITCH3:
  PUSHC FLG        ; Push Flag register onto stack
  FCLR  I           ; Disable interrupts.
  AND.B #00h, 0055h ; Clear TA0IC int. priority level and int. request bit.
  POPC  FLG        ; Enable interrupts.

```

The reason why two NOP instructions (four when using the HOLD function) or dummy read are inserted before FSET I in Examples 1 and 2 is to prevent the interrupt enable flag I from being set before the interrupt control register is rewritten due to effects of the instruction queue.

- When a instruction to rewrite the interrupt control register is executed but the interrupt is disabled, the interrupt request bit is not set sometimes even if the interrupt request for that register has been generated. This will depend on the instruction. If this creates problems, use the below instructions to change the register.

Instructions : AND, OR, BCLR, BSET

### External ROM version

The external ROM version is operated only in microprocessor mode, so be sure to perform the following:

- Connect CNVSS pin to VCC.
- Fix the processor mode bit to “112”

### Built-in PROM version

#### (1) All built-in PROM versions

High voltage is required to program to the built-in PROM. Be careful not to apply excessive voltage. Be especially careful during power-on.

#### (2) One Time PROM version

One Time PROM versions shipped in blank (M30612E4FP, M30612E4GP, M30610ECFP, M30610ECGP), of which built-in PROMs are programmed by users, are also provided. For these microcomputers, a programming test and screening are not performed in the assembly process and the following processes. Therefore ROM write defectiveness occurs around 5 %. To improve their reliability after programming, we recommend to program and test as flow shown in Figure 1.22.1 before use.

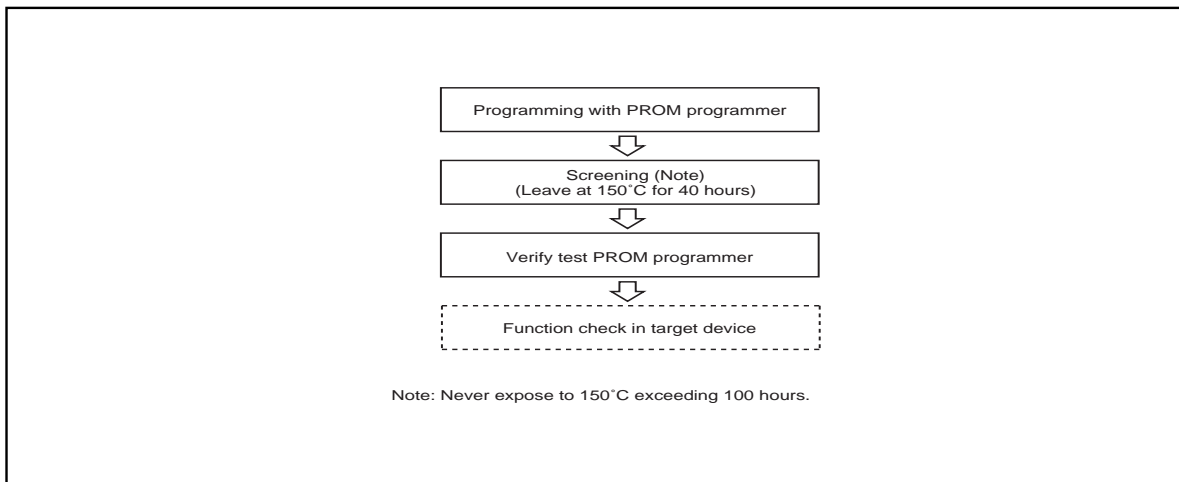


Figure 1.22.1. Programming and test flow for One Time PROM version

#### (3) EPROM version

- Cover the transparent glass window with a shield or others during the read mode because exposing to sun light or fluorescent lamp can cause erasing the information.  
A shield to cover the transparent window is available from Mitsubishi Electric Corp. Be careful that the shield does not touch the EPROM lead pins.
- Clean the transparent glass before erasing. Fingers' flat and paste disturb the passage of ultraviolet rays and may affect badly the erasure capability.
- The EPROM version is a tool only for program development (for evaluation), and do not use it for the mass product run.

### **Items to be submitted when ordering masked ROM version**

Please submit the following when ordering masked ROM products:

- (1) Mask ROM confirmation form
- (2) Mask specification sheet
- (3) ROM data : EPROMs or floppy disks

\*: In the case of EPROMs, there sets of EPROMs are required per pattern.

\*: In the case of floppy disks, 3.5-inch double-sided high-density disk (IBM format) is required per pattern.

**Table 1.24.1. Absolute maximum ratings**

Symbol	Parameter	Condition	Rated value	Unit
V <sub>cc</sub>	Supply voltage	V <sub>cc</sub> = AV <sub>cc</sub>	-0.3 to 6.5 (Note 3)	V
AV <sub>cc</sub>	Analog supply voltage	V <sub>cc</sub> = AV <sub>cc</sub>	-0.3 to 6.5 (Note 3)	V
V <sub>I</sub>	Input voltage P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>2</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> , RESET, VREF, XIN		-0.3 to V <sub>cc</sub> +0.3	V
V <sub>I</sub>	Input voltage P7 <sub>0</sub> , P7 <sub>1</sub> , CNV <sub>ss</sub> , BYTE		-0.3 to 6.5 (Note 1, Note 3)	V
V <sub>O</sub>	Output voltage P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>2</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>4</sub> , P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> , XOUT		-0.3 to V <sub>cc</sub> +0.3	V
V <sub>O</sub>	Output voltage P7 <sub>0</sub> , P7 <sub>1</sub>		-0.3 to 6.5 (Note 3)	V
P <sub>d</sub>	Power dissipation	T <sub>a</sub> =25 °C	300	mW
T <sub>opr</sub>	Operating ambient temperature		-20 to 85 / -40 to 85 (Note 2)	°C
T <sub>stg</sub>	Storage temperature		-65 to 150	°C

Note 1: When writing to EPROM, only CNV<sub>ss</sub> is -0.3 to 13 (V) .

Note 2: Specify a product of -40 to 85°C to use it.

Note 3: -0.3V to 6.5V for M30610M8A, M30610MAA, M30610MCA, M30612M4A, M30612M8A, M30612MAA, M30612MCA, M30610SA and M30612SA.  
Otherwise, -0.3V to 7.0V is used.

**Table 1.24.2. Recommended operating conditions (referenced to Vcc = 2.7V to 5.5V at Ta = -20 to 85°C / -40 to 85°C (Note 3) unless otherwise specified)**

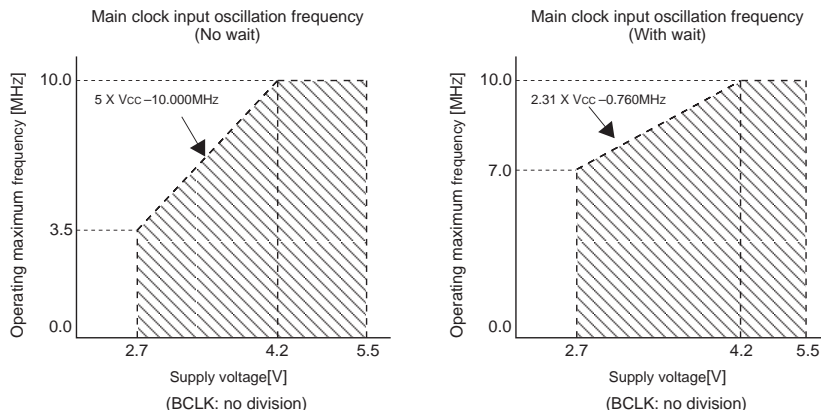
Symbol	Parameter		Standard			Unit
			Min	Typ.	Max.	
Vcc	Supply voltage		2.7	5.0	5.5	V
AVcc	Analog supply voltage			Vcc		V
Vss	Supply voltage			0		V
AVss	Analog supply voltage			0		V
VIH	HIGH input voltage	P31 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P87, P90 to P97, P100 to P107, X IN, RESET, CNVss, BYTE	0.8Vcc		Vcc	V
		P70, P71	0.8Vcc		6.5	V
		P00 to P07, P10 to P17, P20 to P27, P30 (during single-chip mode)	0.8Vcc		Vcc	V
		P00 to P07, P10 to P17, P20 to P27, P30 (data input function during memory expansion and microprocessor modes)	0.5Vcc		Vcc	V
VIL	LOW input voltage	P31 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, P90 to P97, P100 to P107, X IN, RESET, CNVss, BYTE	0		0.2Vcc	V
		P00 to P07, P10 to P17, P20 to P27, P30 (during single-chip mode)	0		0.2Vcc	V
		P00 to P07, P10 to P17, P20 to P27, P30 (data input function during memory expansion and microprocessor modes)	0		0.16Vcc	V
IOH (peak)	HIGH peak output current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107			-10.0	mA
IOH (avg)	HIGH average output current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107			-5.0	mA
IOL (peak)	LOW peak output current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107			10.0	mA
IOL (avg)	LOW average output current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107			5.0	mA
f (XIN)	Main clock input oscillation frequency	No wait	Vcc = 4.0V to 5.5V	0	10	MHz
			Vcc = 2.7V to 4.0V	0	5 X Vcc - 10.000	MHz
		With wait	Vcc = 4.0V to 5.5V	0	10	MHz
			Vcc = 2.7V to 4.0V	0	2.31 X Vcc + 0.760	MHz
f (XCIN)	Subclock oscillation frequency			32.768	50	kHz

Note 1: The mean output current is the mean value within 100ms.

Note 2: The total IOH (peak) for ports P0, P1, P2, P86, P87, P9, and P10 must be 80mA max. The total IOL (peak) for ports P0, P1, P2, P86, P87, P9, and P10 must be 80mA max. The total IOL (peak) for ports P3, P4, P5, P6, P7, and P80 to P84 must be 80mA max. The total IOH (peak) for ports P3, P4, P5, P6, P72 to P77, and P80 to P84 must be 80mA max.

Note 3: Specify a product of -40 to 85°C to use it.

Note 4: The relationship between main clock input frequency and power supply voltage is as below.





VCC = 5V

**Table 1.24.3. Electrical characteristics (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C, f(XIN) = 10MHz unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit	
				Min	Typ.	Max.		
VOH	HIGH output voltage	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107	IOH=-5mA	3.0			V	
VOH	HIGH output voltage	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107	IOH=-200μA	4.7			V	
VOH	HIGH output voltage	XOUT	HIGHPOWER	3.0			V	
			LOWPOWER	3.0			V	
	HIGH output voltage	XCOUT	HIGHPOWER		3.0		V	
			LOWPOWER		1.6		V	
VOL	LOW output voltage	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107	IOL=5mA			2.0	V	
VOL	LOW output voltage	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107	IOL=200μA			0.45	V	
VOL	LOW output voltage	XOUT	HIGHPOWER			2.0	V	
			LOWPOWER			2.0	V	
	LOW output voltage	XCOUT	HIGHPOWER	With no load applied		0	V	
			LOWPOWER	With no load applied		0	V	
VT+-VT-	Hysteresis	HOLD, RDY, TA0IN to TA4IN, TB0IN to TB2IN, INT0 to INT2, ADTRG, CTS0 to CTS2, CLK0 to CLK2, TA2OUT to TA4OUT, NMI, Kl0 to Kl3, RxD0 to RxD2		0.2		0.8	V	
VT+-VT-	Hysteresis	RESET		0.2		1.8	V	
I <sub>IH</sub>	HIGH input current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, P90 to P97, P100 to P107, XIN, RESET, CNVss, BYTE	VI=5V			5.0	μA	
I <sub>IL</sub>	LOW input current	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P70 to P77, P80 to P87, P90 to P97, P100 to P107, XIN, RESET, CNVss, BYTE	VI=0V			-5.0	μA	
R <sub>PULLUP</sub>	Pull-up resistance	P00 to P07, P10 to P17, P20 to P27, P30 to P37, P40 to P47, P50 to P57, P60 to P67, P72 to P77, P80 to P84, P86, P87, P90 to P97, P100 to P107	VI=0V	30.0	50.0	167.0	kΩ	
R <sub>I<sub>XIN</sub></sub>	Feedback resistance	XIN			1.0		MΩ	
R <sub>I<sub>XCIN</sub></sub>	Feedback resistance	XCIN			6.0		MΩ	
V <sub>RAM</sub>	RAM retention voltage		When clock is stopped	2.0			V	
I <sub>CC</sub>	Power supply current		In single-chip mode, the output pins are open and other pins are Vss	f(XIN)=10MHz Square wave, no division	19.0	38.0	mA	
				f(XCIN)=32kHz Square wave		90.0	μA	
				f(XCIN)=32kHz When a WAIT instruction is executed(Note)		4.0	μA	
				Ta=25°C when clock is stopped			1.0	μA
				Ta=85°C when clock is stopped			20.0	μA

Note: With one timer operated using fc32.

V<sub>CC</sub> = 5V**Table 1.24.4. A-D conversion characteristics (referenced to V<sub>CC</sub> = AV<sub>CC</sub> = V<sub>REF</sub> = 5V, V<sub>SS</sub> = AV<sub>SS</sub> = 0V at Ta = 25°C, f(X<sub>IN</sub>) = 10MHz unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit	
				Min.	Typ.	Max.		
-	Resolution		V <sub>REF</sub> = V <sub>CC</sub>			10	Bits	
-	Absolute accuracy	Sample & hold function not available	V <sub>REF</sub> = V <sub>CC</sub> = 5V			±3	LSB	
		Sample & hold function available(10bit)	V <sub>REF</sub> = V <sub>CC</sub> = 5V	AN <sub>0</sub> to AN <sub>7</sub> input			±3	LSB
				ANEX <sub>0</sub> , ANEX <sub>1</sub> input, External op-amp connection mode			±7	LSB
	Sample & hold function available(8bit)	V <sub>REF</sub> = V <sub>CC</sub> = 5V			±2	LSB		
RLADDER	Ladder resistance		V <sub>REF</sub> = V <sub>CC</sub>	10		40	kΩ	
t <sub>CONV</sub>	Conversion time(10bit)			3.3			μs	
t <sub>CONV</sub>	Conversion time(8bit)			2.8			μs	
t <sub>SAMP</sub>	Sampling time			0.3			μs	
V <sub>REF</sub>	Reference voltage			2		V <sub>CC</sub>	V	
V <sub>IA</sub>	Analog input voltage			0		V <sub>REF</sub>	V	

**Table 1.24.5. D-A conversion characteristics (referenced to V<sub>CC</sub> = 5V, V<sub>SS</sub> = AV<sub>SS</sub> = 0V, V<sub>REF</sub> = 5V at Ta = 25°C, f(X<sub>IN</sub>) = 10MHz unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max.	
-	Resolution					8	Bits
-	Absolute accuracy					1.0	%
t <sub>su</sub>	Setup time					3	μs
R <sub>O</sub>	Output resistance			4	10	20	kΩ
I <sub>VREF</sub>	Reference power supply input current		(Note)			1.5	mA

Note: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "0016".

The A-D converter's ladder resistance is not included.

Also, when the V<sub>ref</sub> is unconnected at the A-D control register, I<sub>VREF</sub> is sent.

**V<sub>CC</sub> = 5V****Timing requirements (referenced to V<sub>CC</sub> = 5V, V<sub>SS</sub> = 0V at Ta = 25°C unless otherwise specified)****Table 1.24.6. External clock input**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub>	External clock input cycle time	100		ns
t <sub>w(H)</sub>	External clock input HIGH pulse width	40		ns
t <sub>w(L)</sub>	External clock input LOW pulse width	40		ns
t <sub>r</sub>	External clock rise time		15	ns
t <sub>f</sub>	External clock fall time		15	ns

**Table 1.24.7. Memory expansion and microprocessor modes**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>ac1</sub> (RD-DB)	Data input access time (no wait)		(Note)	ns
t <sub>ac2</sub> (RD-DB)	Data input access time (with wait)		(Note)	ns
t <sub>ac3</sub> (RD-DB)	Data input access time (when accessing multiplex bus area)		(Note)	ns
t <sub>su</sub> (DB-RD)	Data input setup time	40		ns
t <sub>su</sub> (RDY-BCLK)	RDY input setup time	30		ns
t <sub>su</sub> (HOLD-BCLK)	HOLD input setup time	40		ns
t <sub>h</sub> (RD-DB)	Data input hold time	0		ns
t <sub>h</sub> (BCLK-RDY)	RDY input hold time	0		ns
t <sub>h</sub> (BCLK-HOLD)	HOLD input hold time	0		ns
t <sub>d</sub> (BCLK-HLDA)	HLDA output delay time		40	ns

Note: Calculated according to the BCLK frequency as follows:

$$t_{ac1}(\text{RD} - \text{DB}) = \frac{10^9}{f(\text{BCLK}) \times 2} - 45 \quad [\text{ns}]$$

$$t_{ac2}(\text{RD} - \text{DB}) = \frac{3 \times 10^9}{f(\text{BCLK}) \times 2} - 45 \quad [\text{ns}]$$

$$t_{ac3}(\text{RD} - \text{DB}) = \frac{3 \times 10^9}{f(\text{BCLK}) \times 2} - 45 \quad [\text{ns}]$$

**V<sub>CC</sub> = 5V****Timing requirements (referenced to V<sub>CC</sub> = 5V, V<sub>SS</sub> = 0V at Ta = 25°C unless otherwise specified)****Table 1.24.8. Timer A input (counter input in event counter mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TA)	TAiIN input cycle time	100		ns
t <sub>w</sub> (TAH)	TAiIN input HIGH pulse width	40		ns
t <sub>w</sub> (TAL)	TAiIN input LOW pulse width	40		ns

**Table 1.24.9. Timer A input (gating input in timer mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TA)	TAiIN input cycle time	400		ns
t <sub>w</sub> (TAH)	TAiIN input HIGH pulse width	200		ns
t <sub>w</sub> (TAL)	TAiIN input LOW pulse width	200		ns

**Table 1.24.10. Timer A input (external trigger input in one-shot timer mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TA)	TAiIN input cycle time	200		ns
t <sub>w</sub> (TAH)	TAiIN input HIGH pulse width	100		ns
t <sub>w</sub> (TAL)	TAiIN input LOW pulse width	100		ns

**Table 1.24.11. Timer A input (external trigger input in pulse width modulation mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>w</sub> (TAH)	TAiIN input HIGH pulse width	100		ns
t <sub>w</sub> (TAL)	TAiIN input LOW pulse width	100		ns

**Table 1.24.12. Timer A input (up/down input in event counter mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (UP)	TAiOUT input cycle time	2000		ns
t <sub>w</sub> (UPH)	TAiOUT input HIGH pulse width	1000		ns
t <sub>w</sub> (UPL)	TAiOUT input LOW pulse width	1000		ns
t <sub>su</sub> (UP-TiN)	TAiOUT input setup time	400		ns
t <sub>h</sub> (TiN-UP)	TAiOUT input hold time	400		ns

VCC = 5V

Timing requirements (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C unless otherwise specified)

Table 1.24.13. Timer B input (counter input in event counter mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TB)	TBiIN input cycle time (counted on one edge)	100		ns
t <sub>w</sub> (TBH)	TBiIN input HIGH pulse width (counted on one edge)	40		ns
t <sub>w</sub> (TBL)	TBiIN input LOW pulse width (counted on one edge)	40		ns
t <sub>c</sub> (TB)	TBiIN input cycle time (counted on both edges)	200		ns
t <sub>w</sub> (TBH)	TBiIN input HIGH pulse width (counted on both edges)	80		ns
t <sub>w</sub> (TBL)	TBiIN input LOW pulse width (counted on both edges)	80		ns

Table 1.24.14. Timer B input (pulse period measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TB)	TBiIN input cycle time	400		ns
t <sub>w</sub> (TBH)	TBiIN input HIGH pulse width	200		ns
t <sub>w</sub> (TBL)	TBiIN input LOW pulse width	200		ns

Table 1.24.15. Timer B input (pulse width measurement mode)

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TB)	TBiIN input cycle time	400		ns
t <sub>w</sub> (TBH)	TBiIN input HIGH pulse width	200		ns
t <sub>w</sub> (TBL)	TBiIN input LOW pulse width	200		ns

Table 1.24.16. A-D trigger input

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (AD)	ADTRG input cycle time (trigger able minimum)	1000		ns
t <sub>w</sub> (ADL)	ADTRG input LOW pulse width	125		ns

Table 1.24.17. Serial I/O

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (CK)	CLKi input cycle time	200		ns
t <sub>w</sub> (CKH)	CLKi input HIGH pulse width	100		ns
t <sub>w</sub> (CKL)	CLKi input LOW pulse width	100		ns
t <sub>d</sub> (C-Q)	TxDi output delay time		80	ns
t <sub>h</sub> (C-Q)	TxDi hold time	0		ns
t <sub>su</sub> (D-C)	RxDi input setup time	30		ns
t <sub>h</sub> (C-D)	RxDi input hold time	90		ns

Table 1.24.18. External interrupt INTi inputs

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>w</sub> (INH)	INTi input HIGH pulse width	250		ns
t <sub>w</sub> (INL)	INTi input LOW pulse width	250		ns

Vcc = 5V

Switching characteristics (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)

Table 1.24.19. Memory expansion mode and microprocessor mode (no wait)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 1.24.1		25	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		0		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		0		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			25	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>d</sub> (BCLK-ALE)	ALE signal output delay time			25	ns
t <sub>h</sub> (BCLK-ALE)	ALE signal output hold time		-4		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			25	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			25	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			40	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note1)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)(Note2)		0		ns

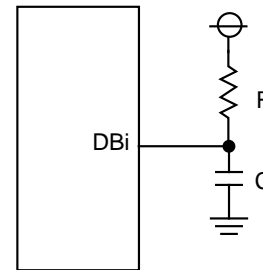
Note 1: Calculated according to the BCLK frequency as follows:

$$t_d(\text{DB} - \text{WR}) = \frac{10^9}{f(\text{BCLK}) \times 2} - 40 \quad [\text{ns}]$$

Note 2: This is standard value shows the timing when the output is off, and doesn't show hold time of data bus.  
Hold time of data bus is different by capacitor volume and pull-up (pull-down) resistance value.  
Hold time of data bus is expressed in  
 $t = -CR \times \ln(1 - V_{OL} / V_{CC})$   
by a circuit of the right figure.

For example, when V<sub>OL</sub> = 0.2V<sub>CC</sub>, C = 30pF, R = 1kΩ, hold time of output "L" level is

$$t = -30\text{pF} \times 1\text{k}\Omega \times \ln(1 - 0.2V_{CC} / V_{CC}) = 6.7\text{ns}.$$



VCC = 5V

Switching characteristics (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)

Table 1.24.20. Memory expansion mode and microprocessor mode  
(with wait, accessing external memory)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 1.24.1		25	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		0		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		0		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			25	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>d</sub> (BCLK-ALE)	ALE signal output delay time			25	ns
t <sub>h</sub> (BCLK-ALE)	ALE signal output hold time		-4		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			25	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			25	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			40	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note1)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)(Note2)		0		ns

Note 1: Calculated according to the BCLK frequency as follows:

$$t_d(\text{DB} - \text{WR}) = \frac{10^9}{f(\text{BCLK})} - 40 \quad [\text{ns}]$$

Note 2: This is standard value shows the timing when the output is off, and doesn't show hold time of data bus.  
Hold time of data bus is different by capacitor volume and pull-up (pull-down) resistance value.

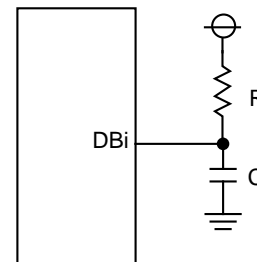
Hold time of data bus is expressed in

$$t = -CR \times \ln(1 - V_{OL} / V_{CC})$$

by a circuit of the right figure.

For example, when  $V_{OL} = 0.2V_{CC}$ ,  $C = 30\text{pF}$ ,  $R = 1\text{k}\Omega$ , hold time of output "L" level is

$$t = -30\text{pF} \times 1\text{k}\Omega \times \ln(1 - 0.2V_{CC} / V_{CC}) \\ = 6.7\text{ns}.$$



VCC = 5V

**Switching characteristics (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)**

**Table 1.24.21. Memory expansion mode and microprocessor mode  
(with wait, accessing external memory, multiplex bus area selected)**

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 1.24.1		25	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		(Note)		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		(Note)		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			25	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-CS)	Chip select output hold time (RD standard)		(Note)		ns
t <sub>h</sub> (WR-CS)	Chip select output hold time (WR standard)		(Note)		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			25	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			25	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			40	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)		(Note)		ns
t <sub>d</sub> (BCLK-ALE)	ALE signal output delay time (BCLK standard)			25	ns
t <sub>h</sub> (BCLK-ALE)	ALE signal output hold time (BCLK standard)		- 4		ns
t <sub>d</sub> (AD-ALE)	ALE signal output delay time (Address standard)		(Note)		ns
t <sub>h</sub> (ALE-AD)	ALE signal output hold time (Address standard)		50		ns
t <sub>d</sub> (AD-RD)	Post-address RD signal output delay time	0		ns	
t <sub>d</sub> (AD-WR)	Post-address WR signal output delay time	0		ns	
t <sub>dZ</sub> (RD-AD)	Address output floating start time		8	ns	

Note: Calculated according to the BCLK frequency as follows:

$$t_h(\text{RD} - \text{AD}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_h(\text{WR} - \text{AD}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_h(\text{RD} - \text{CS}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_h(\text{WR} - \text{CS}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_d(\text{DB} - \text{WR}) = \frac{10^9 \times 3}{f(\text{BCLK}) \times 2} - 40 \quad [\text{ns}]$$

$$t_h(\text{WR} - \text{DB}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_d(\text{AD} - \text{ALE}) = \frac{10^9}{f(\text{BCLK}) \times 2} - 25 \quad [\text{ns}]$$



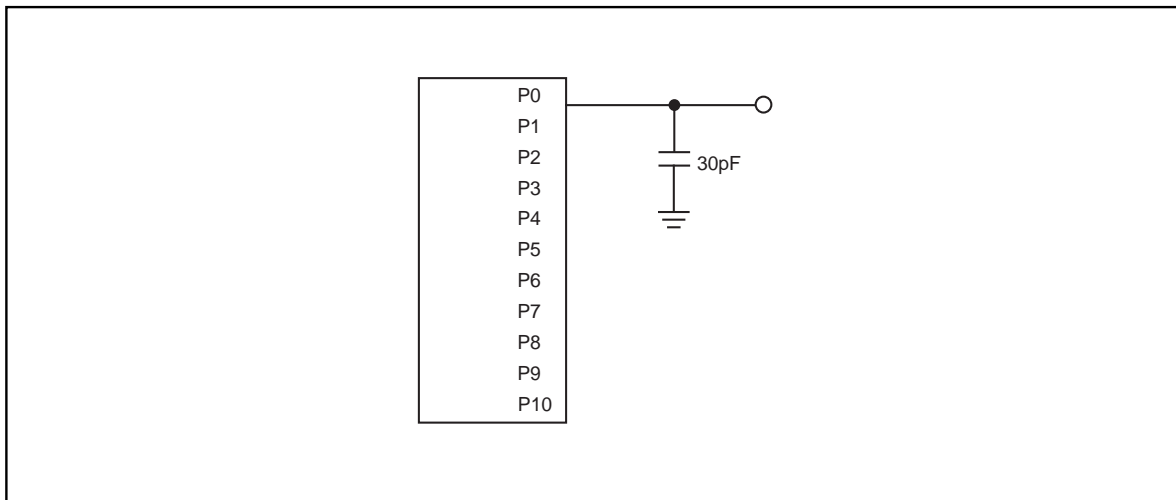


Figure 1.24.1. Port P0 to P10 measurement circuit

Timing ( $V_{CC} = 5V$ )

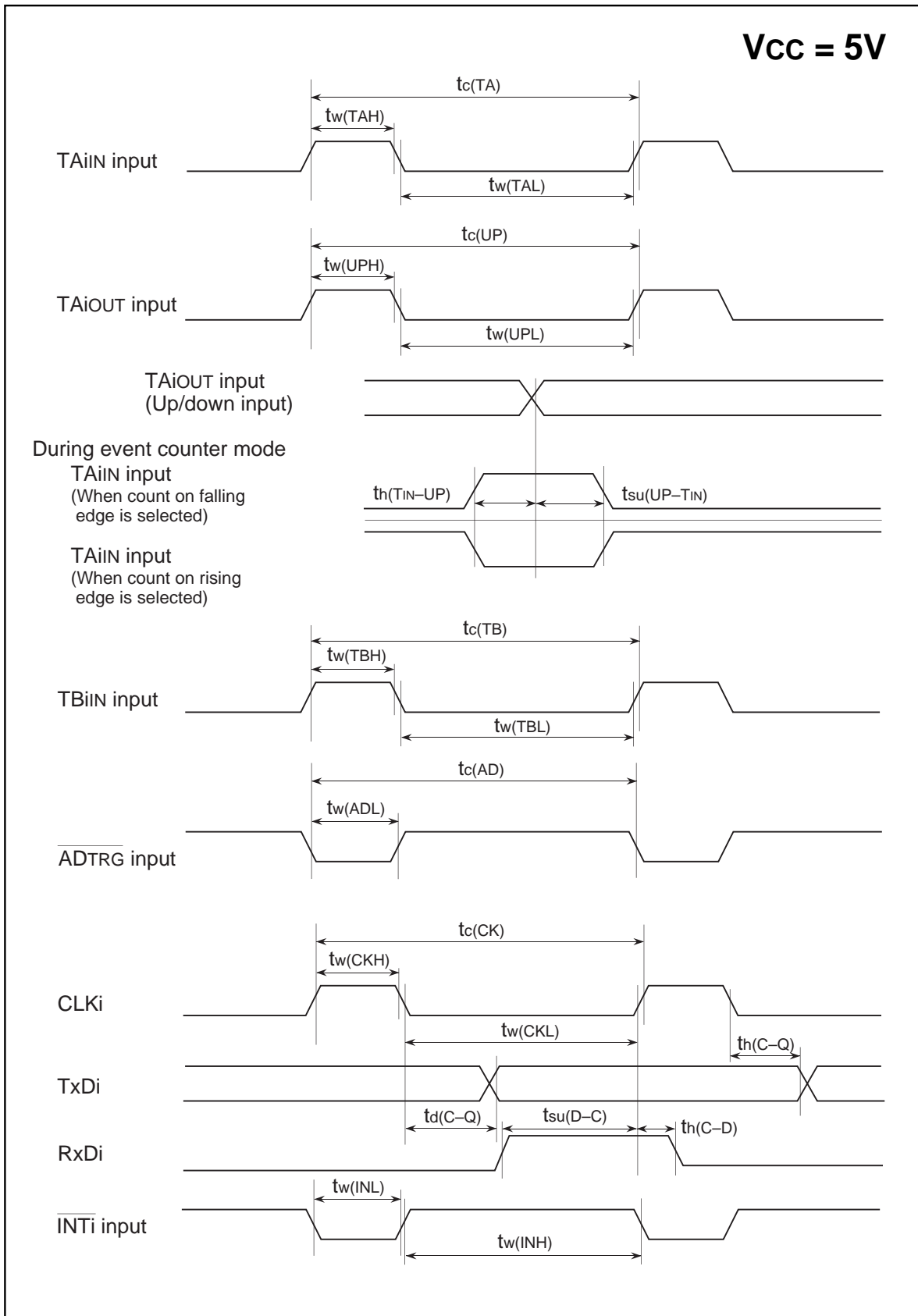


Figure 1.24.2.  $V_{CC}=5V$  timing diagram (1)

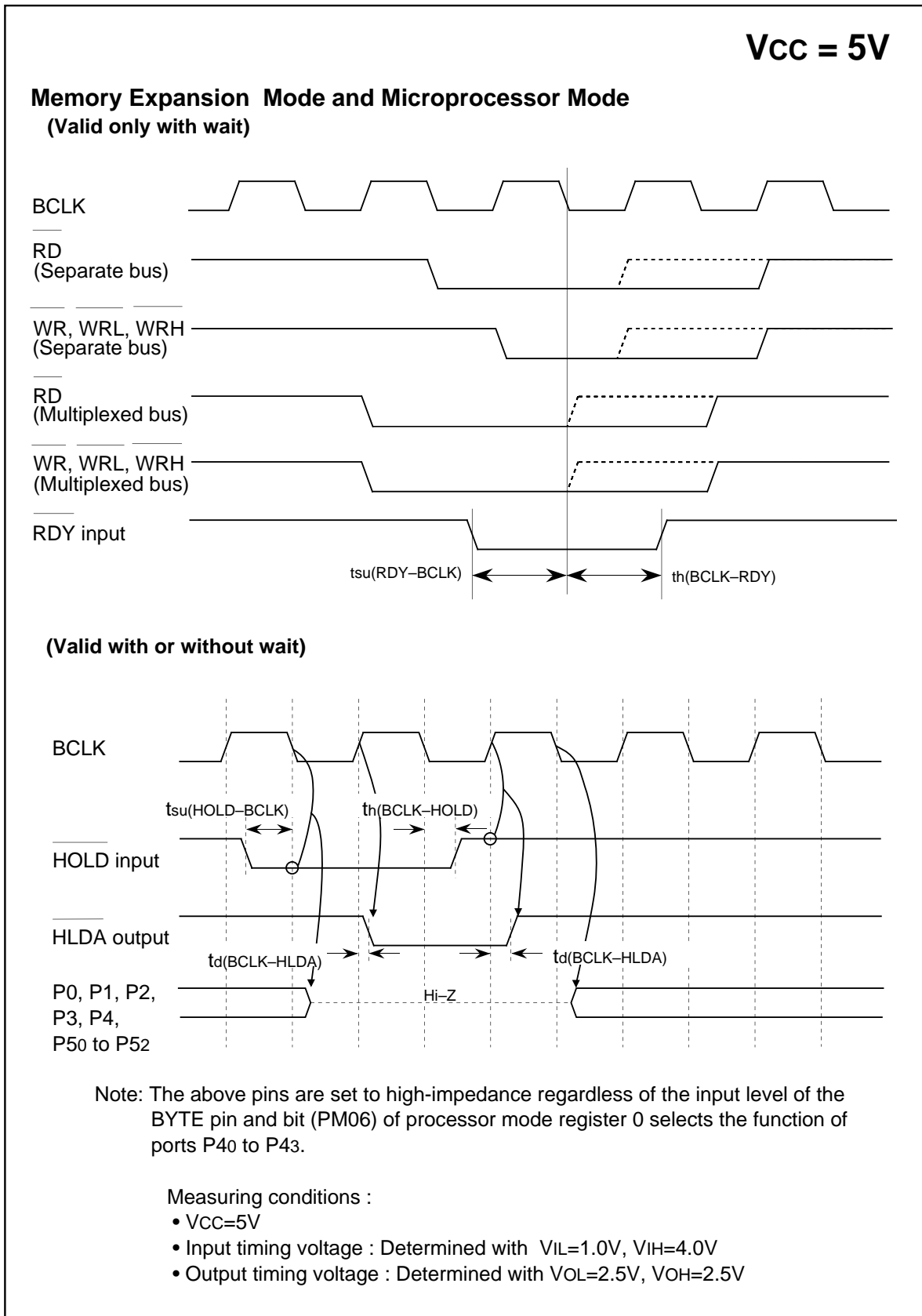
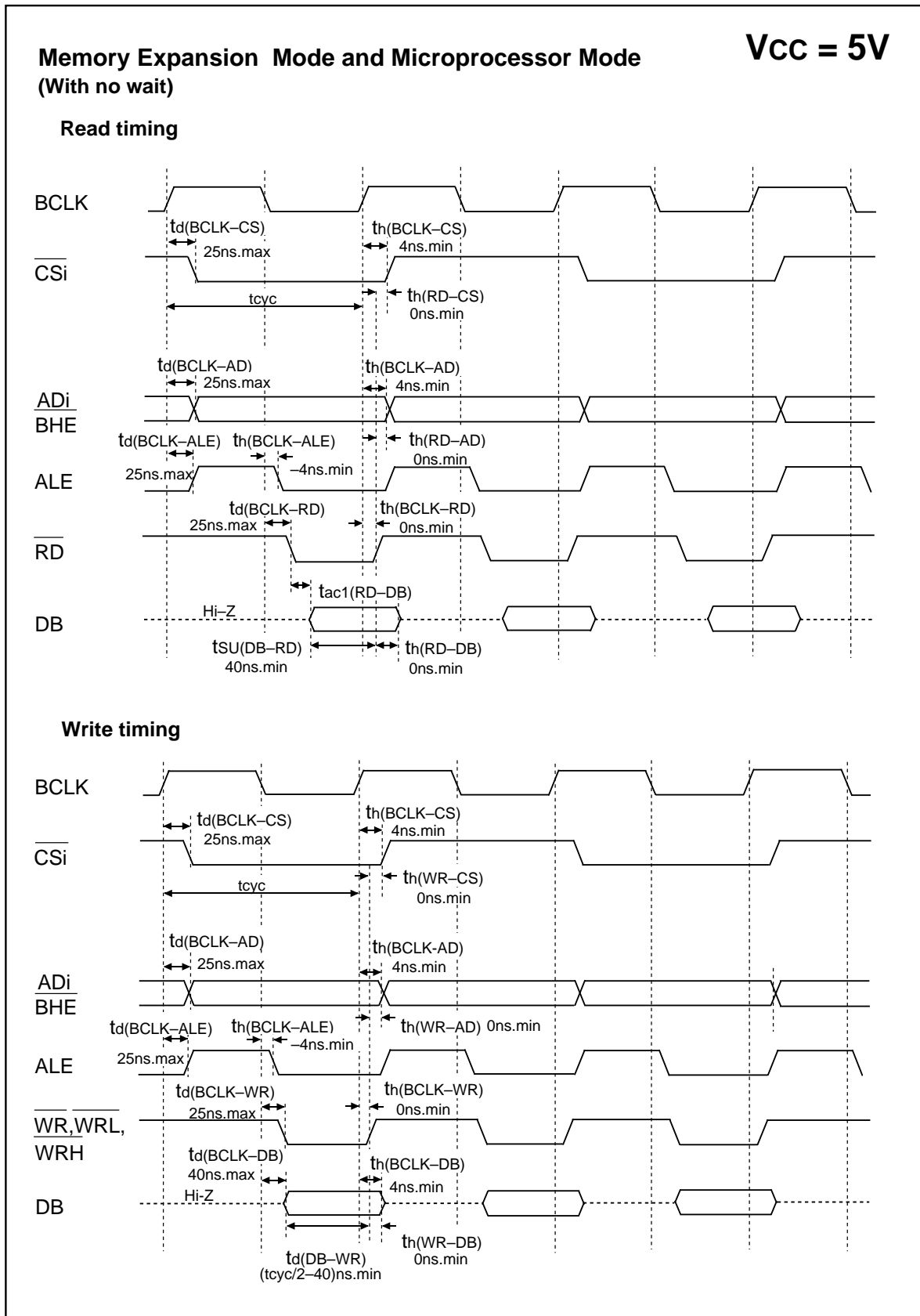
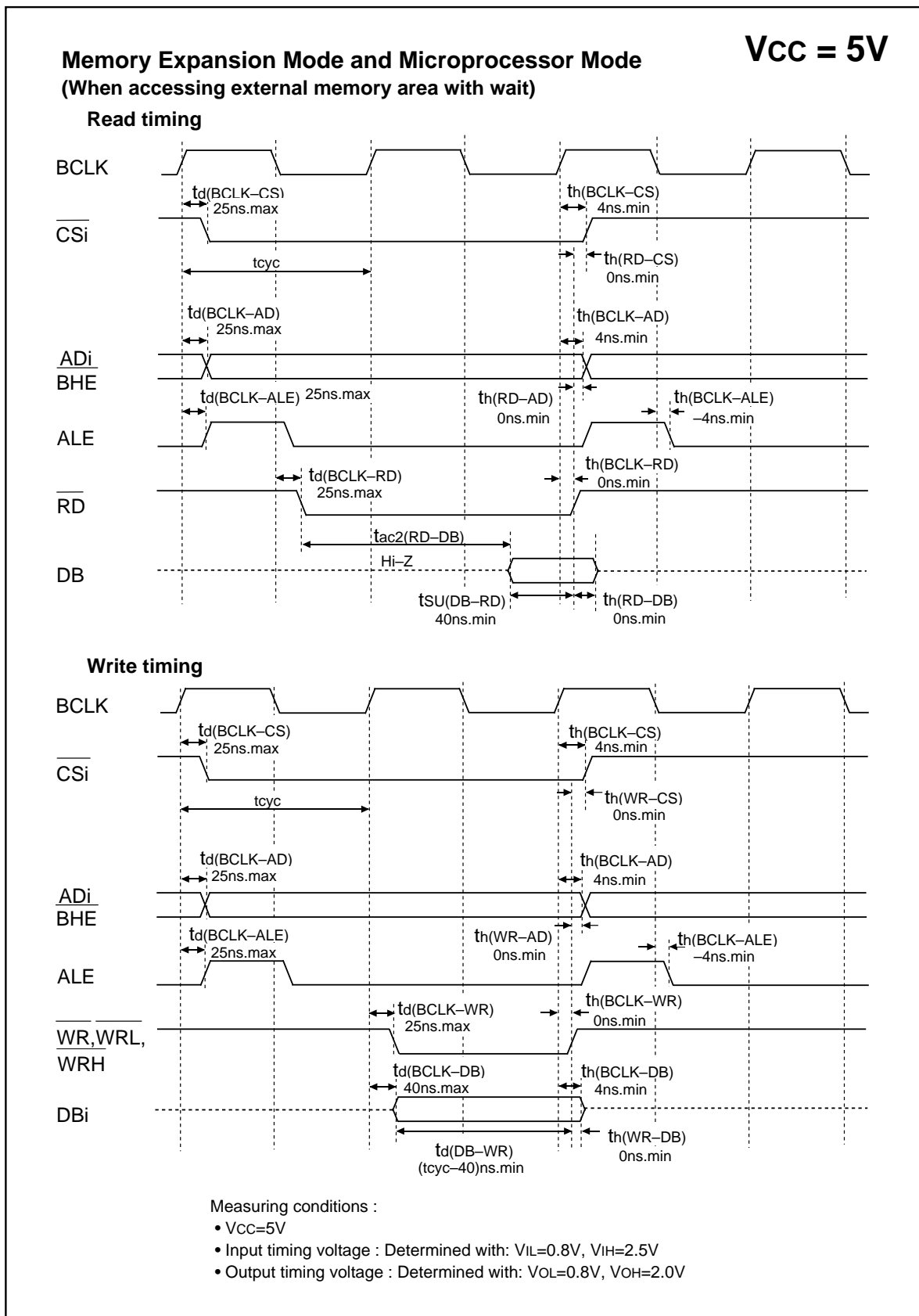
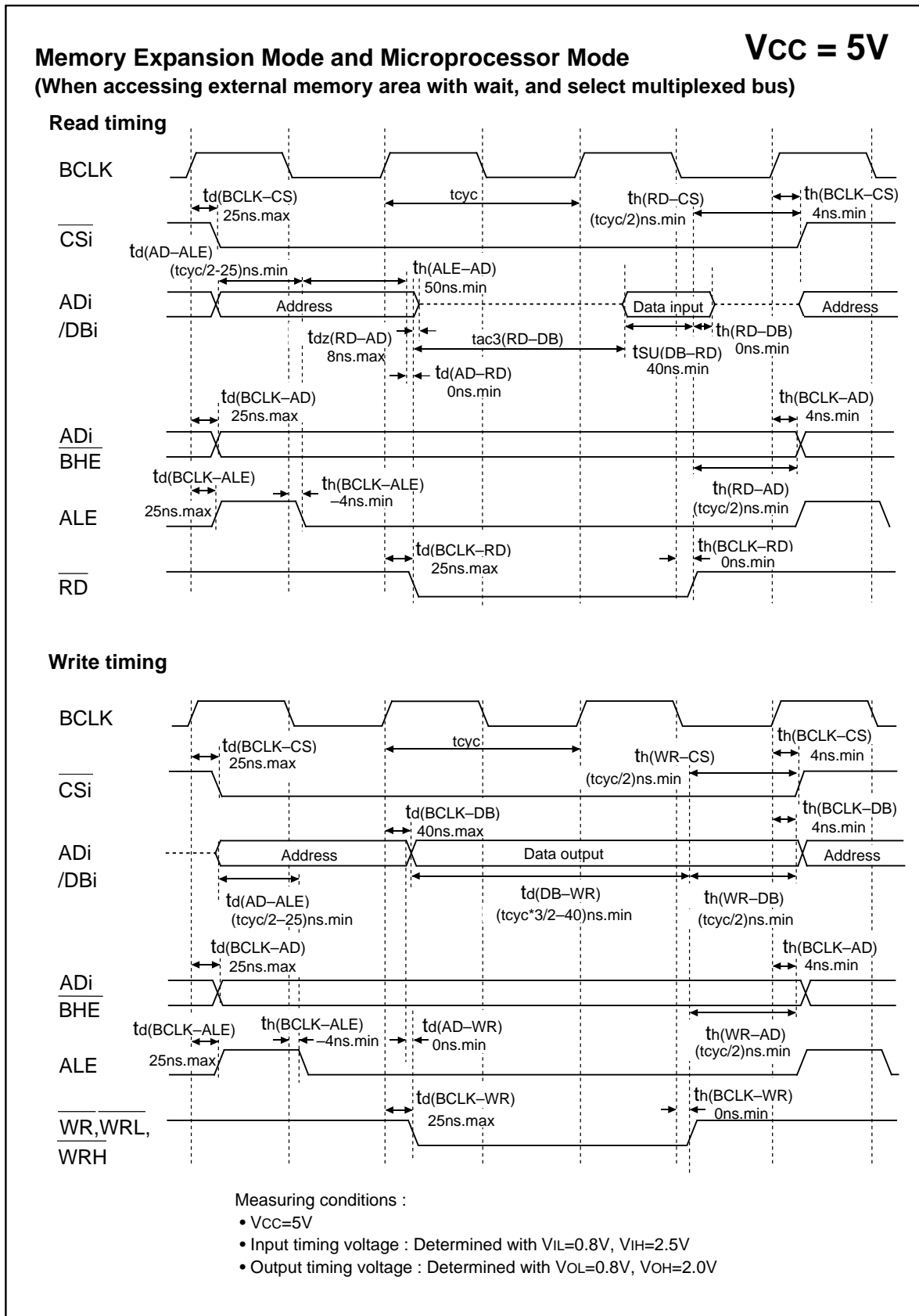


Figure 1.24.3. Vcc=5V timing diagram (2)

Figure 1.24.4.  $V_{CC}=5V$  timing diagram (3)

Timing ( $V_{CC} = 5V$ )Figure 1.24.5.  $V_{CC}=5V$  timing diagram (4)

Timing ( $V_{CC} = 5V$ )Figure 1.24.6.  $V_{CC}=5V$  timing diagram (5)

V<sub>CC</sub> = 3V**Table 1.24.22. Electrical characteristics (referenced to V<sub>CC</sub> = 3V, V<sub>SS</sub> = 0V at Ta = 25°C, f(X<sub>IN</sub>) = 7MHz, with wait)**

Symbol	Parameter		Measuring condition	Standard			Unit	
				Min	Typ.	Max.		
V <sub>OH</sub>	HIGH output voltage	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>2</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>4</sub> , P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	I <sub>OH</sub> =-1mA	2.5			V	
V <sub>OH</sub>	HIGH output voltage	X <sub>OUT</sub>	HIGHPOWER	I <sub>OH</sub> =-0.1mA	2.5		V	
			LOWPOWER	I <sub>OH</sub> =-50μA	2.5			
V <sub>OH</sub>	HIGH output voltage	X <sub>COUT</sub>	HIGHPOWER	With no load applied		3.0	V	
			LOWPOWER	With no load applied		1.6		
V <sub>OL</sub>	LOW output voltage	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>4</sub> , P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	I <sub>OL</sub> =1mA			0.5	V	
V <sub>OL</sub>	LOW output voltage	X <sub>OUT</sub>	HIGHPOWER	I <sub>OL</sub> =0.1mA		0.5	V	
			LOWPOWER	I <sub>OL</sub> =50μA		0.5		
V <sub>OL</sub>	LOW output voltage	X <sub>COUT</sub>	HIGHPOWER	With no load applied		0	V	
			LOWPOWER	With no load applied		0		
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis	HOLD, RDY, TA0 <sub>IN</sub> to TA4 <sub>IN</sub> , TB0 <sub>IN</sub> to TB2 <sub>IN</sub> , INT0 to INT2, ADTRG, CTS0 to CTS2, CLK0 to CLK2, TA2 <sub>OUT</sub> to TA4 <sub>OUT</sub> , NMI, K10 to K13, RxD0 to RxD2		0.2		0.8	V	
V <sub>T+</sub> -V <sub>T-</sub>	Hysteresis	RESET		0.2		1.8	V	
I <sub>IH</sub>	HIGH input current	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> . X <sub>IN</sub> , RESET, CNV <sub>SS</sub> , BYTE	V <sub>I</sub> =3V			4.0	μA	
I <sub>IL</sub>	LOW input current	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>0</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub> . X <sub>IN</sub> , RESET, CNV <sub>SS</sub> , BYTE	V <sub>I</sub> =0V			-4.0	μA	
R <sub>PULLUP</sub>	Pull-up resistance	P0 <sub>0</sub> to P0 <sub>7</sub> , P1 <sub>0</sub> to P1 <sub>7</sub> , P2 <sub>0</sub> to P2 <sub>7</sub> , P3 <sub>0</sub> to P3 <sub>7</sub> , P4 <sub>0</sub> to P4 <sub>7</sub> , P5 <sub>0</sub> to P5 <sub>7</sub> , P6 <sub>0</sub> to P6 <sub>7</sub> , P7 <sub>2</sub> to P7 <sub>7</sub> , P8 <sub>0</sub> to P8 <sub>4</sub> , P8 <sub>6</sub> , P8 <sub>7</sub> , P9 <sub>0</sub> to P9 <sub>7</sub> , P10 <sub>0</sub> to P10 <sub>7</sub>	V <sub>I</sub> =0V	66.0	120.0	500.0	kΩ	
R <sub>FXIN</sub>	Feedback resistance	X <sub>IN</sub>			3.0		MΩ	
R <sub>FXCIN</sub>	Feedback resistance	X <sub>CIN</sub>			10.0		MΩ	
V <sub>RAM</sub>	RAM retention voltage		When clock is stopped	2.0			V	
I <sub>CC</sub>	Power supply current		In single-chip mode, the output pins are open and other pins are V <sub>SS</sub>	f(X <sub>IN</sub> )=7MHz Square wave, no division		6.0	15.0	mA
				f(X <sub>CIN</sub> )=32kHz Square wave		40.0		
				f(X <sub>CIN</sub> )=32kHz When a WAIT instruction is executed. Oscillation capacity High (Note)		2.8		μA
				f(X <sub>CIN</sub> )=32kHz When a WAIT instruction is executed. Oscillation capacity Low (Note)		0.9		
				Ta=25°C when clock is stopped			1.0	μA
				Ta=85°C when clock is stopped			20.0	

Note: With one timer operated using fc32.

**V<sub>CC</sub> = 3V****Table 1.24.23. A-D conversion characteristics (referenced to V<sub>CC</sub> = AV<sub>CC</sub> = V<sub>REF</sub> = 3V, V<sub>SS</sub> = AV<sub>SS</sub> = 0V at Ta = 25°C, f(X<sub>IN</sub>) = 7MHz unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max	
-	Resolution		V <sub>REF</sub> = V <sub>CC</sub>			10	Bits
-	Absolute accuracy	Sample & hold function not available (8 bit)	V <sub>REF</sub> = V <sub>CC</sub> = 3V, φ <sub>AD</sub> = f(X <sub>IN</sub> )/2			±2	LSB
R <sub>LADDER</sub>	Ladder resistance		V <sub>REF</sub> = V <sub>CC</sub>	10		40	kΩ
t <sub>CONV</sub>	Conversion time(8bit)			14.0			μs
V <sub>REF</sub>	Reference voltage			2.7		V <sub>CC</sub>	V
V <sub>IA</sub>	Analog input voltage			0		V <sub>REF</sub>	V

**Table 1.24.24. D-A conversion characteristics (referenced to V<sub>CC</sub> = 3V, V<sub>SS</sub> = AV<sub>SS</sub> = 0V, V<sub>REF</sub> = 3V at Ta = 25°C, f(X<sub>IN</sub>) = 7MHz unless otherwise specified)**

Symbol	Parameter		Measuring condition	Standard			Unit
				Min.	Typ.	Max	
-	Resolution					8	Bits
-	Absolute accuracy					1.0	%
t <sub>su</sub>	Setup time					3	μs
R <sub>O</sub>	Output resistance			4	10	20	kΩ
I <sub>VREF</sub>	Reference power supply input current		(Note)			1.0	mA

Note: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "0016".  
The A-D converter's ladder resistance is not included.  
Also, when the Vref is unconnected at the A-D control register, I<sub>VREF</sub> is sent.



**V<sub>CC</sub> = 3V****Timing requirements (referenced to V<sub>CC</sub> = 3V, V<sub>SS</sub> = 0V at Ta = 25°C unless otherwise specified)****Table 1.24.25. External clock input**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub>	External clock input cycle time	143		ns
t <sub>w(H)</sub>	External clock input HIGH pulse width	60		ns
t <sub>w(L)</sub>	External clock input LOW pulse width	60		ns
t <sub>r</sub>	External clock rise time		18	ns
t <sub>f</sub>	External clock fall time		18	ns

**Table 1.24.26. Memory expansion and microprocessor modes**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>ac1</sub> (RD-DB)	Data input access time (no wait)		(Note)	ns
t <sub>ac2</sub> (RD-DB)	Data input access time (with wait)		(Note)	ns
t <sub>ac3</sub> (RD-DB)	Data input access time (when accessing multiplex bus area)		(Note)	ns
t <sub>su</sub> (DB-RD)	Data input setup time	80		ns
t <sub>su</sub> (RDY-BCLK)	RDY input setup time	60		ns
t <sub>su</sub> (HOLD-BCLK)	HOLD input setup time	80		ns
t <sub>h</sub> (RD-DB)	Data input hold time	0		ns
t <sub>h</sub> (BCLK-RDY)	RDY input hold time	0		ns
t <sub>h</sub> (BCLK-HOLD)	HOLD input hold time	0		ns
t <sub>d</sub> (BCLK-HLDA)	HLDA output delay time		100	ns

Note: Calculated according to the BCLK frequency as follows:

$$t_{ac1}(\text{RD} - \text{DB}) = \frac{10^9}{f(\text{BCLK}) \times 2} - 90 \quad [\text{ns}]$$

$$t_{ac2}(\text{RD} - \text{DB}) = \frac{3 \times 10^9}{f(\text{BCLK}) \times 2} - 90 \quad [\text{ns}]$$

$$t_{ac3}(\text{RD} - \text{DB}) = \frac{3 \times 10^9}{f(\text{BCLK}) \times 2} - 90 \quad [\text{ns}]$$

**V<sub>CC</sub> = 3V****Timing requirements (referenced to V<sub>CC</sub> = 3V, V<sub>SS</sub> = 0V at Ta = 25°C unless otherwise specified)****Table 1.24.27. Timer A input (counter input in event counter mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c(TA)</sub>	TAiIN input cycle time	150		ns
t <sub>w(TAH)</sub>	TAiIN input HIGH pulse width	60		ns
t <sub>w(TAL)</sub>	TAiIN input LOW pulse width	60		ns

**Table 1.24.28. Timer A input (gating input in timer mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c(TA)</sub>	TAiIN input cycle time	600		ns
t <sub>w(TAH)</sub>	TAiIN input HIGH pulse width	300		ns
t <sub>w(TAL)</sub>	TAiIN input LOW pulse width	300		ns

**Table 1.24.29. Timer A input (external trigger input in one-shot timer mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c(TA)</sub>	TAiIN input cycle time	300		ns
t <sub>w(TAH)</sub>	TAiIN input HIGH pulse width	150		ns
t <sub>w(TAL)</sub>	TAiIN input LOW pulse width	150		ns

**Table 1.24.30. Timer A input (external trigger input in pulse width modulation mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>w(TAH)</sub>	TAiIN input HIGH pulse width	150		ns
t <sub>w(TAL)</sub>	TAiIN input LOW pulse width	150		ns

**Table 1.24.31. Timer A input (up/down input in event counter mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c(UP)</sub>	TAiOUT input cycle time	3000		ns
t <sub>w(UPH)</sub>	TAiOUT input HIGH pulse width	1500		ns
t <sub>w(UPL)</sub>	TAiOUT input LOW pulse width	1500		ns
t <sub>su(UP-TIN)</sub>	TAiOUT input setup time	600		ns
t <sub>h(TIN-UP)</sub>	TAiOUT input hold time	600		ns

**Vcc = 3V****Timing requirements (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C unless otherwise specified)****Table 1.24.32. Timer B input (counter input in event counter mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TB)	TBiIn input cycle time (counted on one edge)	150		ns
t <sub>w</sub> (TBH)	TBiIn input HIGH pulse width (counted on one edge)	60		ns
t <sub>w</sub> (TBL)	TBiIn input LOW pulse width (counted on one edge)	60		ns
t <sub>c</sub> (TB)	TBiIn input cycle time (counted on both edges)	300		ns
t <sub>w</sub> (TBH)	TBiIn input HIGH pulse width (counted on both edges)	160		ns
t <sub>w</sub> (TBL)	TBiIn input LOW pulse width (counted on both edges)	160		ns

**Table 1.24.33. Timer B input (pulse period measurement mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TB)	TBiIn input cycle time	600		ns
t <sub>w</sub> (TBH)	TBiIn input HIGH pulse width	300		ns
t <sub>w</sub> (TBL)	TBiIn input LOW pulse width	300		ns

**Table 1.24.34. Timer B input (pulse width measurement mode)**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (TB)	TBiIn input cycle time	600		ns
t <sub>w</sub> (TBH)	TBiIn input HIGH pulse width	300		ns
t <sub>w</sub> (TBL)	TBiIn input LOW pulse width	300		ns

**Table 1.24.35. A-D trigger input**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (AD)	ADTRG input cycle time (trigger able minimum)	1500		ns
t <sub>w</sub> (ADL)	ADTRG input LOW pulse width	200		ns

**Table 1.24.36. Serial I/O**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>c</sub> (CK)	CLKi input cycle time	300		ns
t <sub>w</sub> (CKH)	CLKi input HIGH pulse width	150		ns
t <sub>w</sub> (CKL)	CLKi input LOW pulse width	150		ns
t <sub>d</sub> (C-Q)	TxDi output delay time		160	ns
t <sub>h</sub> (C-Q)	TxDi hold time	0		ns
t <sub>su</sub> (D-C)	RxDi input setup time	50		ns
t <sub>h</sub> (C-D)	RxDi input hold time	90		ns

**Table 1.24.37. External interrupt INTi inputs**

Symbol	Parameter	Standard		Unit
		Min.	Max.	
t <sub>w</sub> (INH)	INTi input HIGH pulse width	380		ns
t <sub>w</sub> (INL)	INTi input LOW pulse width	380		ns

VCC = 3V

**Switching characteristics (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C, CM15 = “1” unless otherwise specified)**

**Table 1.24.38. Memory expansion and microprocessor modes (with no wait)**

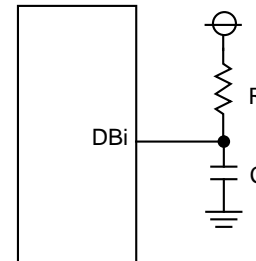
Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 1.24.1		60	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		0		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		0		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			60	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>d</sub> (BCLK-ALE)	ALE signal output delay time			60	ns
t <sub>h</sub> (BCLK-ALE)	ALE signal output hold time		-4		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			60	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			60	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			80	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note1)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)(Note2)		0		ns

Note 1: Calculated according to the BCLK frequency as follows:

$$t_d(\text{DB} - \text{WR}) = \frac{10^9}{f(\text{BCLK}) \times 2} - 80 \quad [\text{ns}]$$

Note 2: This is standard value shows the timing when the output is off, and doesn't show hold time of data bus.  
 Hold time of data bus is different by capacitor volume and pull-up (pull-down) resistance value.  
 Hold time of data bus is expressed in  
 $t = -CR \times \ln(1 - V_{OL} / V_{CC})$   
 by a circuit of the right figure.  
 For example, when  $V_{OL} = 0.2V_{CC}$ ,  $C = 30\text{pF}$ ,  $R = 1\text{k}\Omega$ , hold time of output “L” level is

$$t = -30\text{pF} \times 1\text{k}\Omega \times \ln(1 - 0.2V_{CC} / V_{CC}) = 6.7\text{ns}.$$



Vcc = 3V

Switching characteristics (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C, CM15 = “1” unless otherwise specified)

**Table 1.24.39. Memory expansion and microprocessor modes**  
 (when accessing external memory area with wait)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 1.24.1		60	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		0		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		0		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			60	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>d</sub> (BCLK-ALE)	ALE signal output delay time			60	ns
t <sub>h</sub> (BCLK-ALE)	ALE signal output hold time		-4		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			60	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			60	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			80	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note1)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)(Note2)		0		ns

Note 1: Calculated according to the BCLK frequency as follows:

$$t_d(\text{DB} - \text{WR}) = \frac{10^9}{f(\text{BCLK})} - 80 \quad [\text{ns}]$$

Note 2: This is standard value shows the timing when the output is off, and doesn't show hold time of data bus. Hold time of data bus is different by capacitor volume and pull-up (pull-down) resistance value.

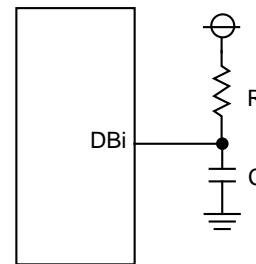
Hold time of data bus is expressed in

$$t = -CR \times \ln(1 - V_{OL} / V_{CC})$$

by a circuit of the right figure.

For example, when V<sub>OL</sub> = 0.2V<sub>CC</sub>, C = 30pF, R = 1kΩ, hold time of output “L” level is

$$t = -30\text{pF} \times 1\text{k}\Omega \times \ln(1 - 0.2V_{CC} / V_{CC}) = 6.7\text{ns}.$$



VCC = 3V

Switching characteristics (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C, CM15 = "1" unless otherwise specified)

**Table 1.24.40. Memory expansion and microprocessor modes**  
(when accessing external memory area with wait, and select multiplexed bus)

Symbol	Parameter	Measuring condition	Standard		Unit
			Min.	Max.	
t <sub>d</sub> (BCLK-AD)	Address output delay time	Figure 1.24.1		60	ns
t <sub>h</sub> (BCLK-AD)	Address output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-AD)	Address output hold time (RD standard)		(Note)		ns
t <sub>h</sub> (WR-AD)	Address output hold time (WR standard)		(Note)		ns
t <sub>d</sub> (BCLK-CS)	Chip select output delay time			60	ns
t <sub>h</sub> (BCLK-CS)	Chip select output hold time (BCLK standard)		4		ns
t <sub>h</sub> (RD-CS)	Chip select output hold time (RD standard)		(Note)		ns
t <sub>h</sub> (WR-CS)	Chip select output hold time (WR standard)		(Note)		ns
t <sub>d</sub> (BCLK-RD)	RD signal output delay time			60	ns
t <sub>h</sub> (BCLK-RD)	RD signal output hold time		0		ns
t <sub>d</sub> (BCLK-WR)	WR signal output delay time			60	ns
t <sub>h</sub> (BCLK-WR)	WR signal output hold time		0		ns
t <sub>d</sub> (BCLK-DB)	Data output delay time (BCLK standard)			80	ns
t <sub>h</sub> (BCLK-DB)	Data output hold time (BCLK standard)		4		ns
t <sub>d</sub> (DB-WR)	Data output delay time (WR standard)		(Note)		ns
t <sub>h</sub> (WR-DB)	Data output hold time (WR standard)		(Note)		ns
t <sub>d</sub> (BCLK-ALE)	ALE signal output delay time (BCLK standard)			60	ns
t <sub>h</sub> (BCLK-ALE)	ALE signal output hold time (BCLK standard)		-4		ns
t <sub>d</sub> (AD-ALE)	ALE signal output delay time (Address standard)		(Note)		ns
t <sub>h</sub> (ALE-AD)	ALE signal output hold time (Address standard)		50		ns
t <sub>d</sub> (AD-RD)	Post-address RD signal output delay time	0		ns	
t <sub>d</sub> (AD-WR)	Post-address WR signal output delay time	0		ns	
t <sub>d</sub> (RD-AD)	Address output floating start time		8	ns	

Note: Calculated according to the BCLK frequency as follows:

$$t_h(\text{RD} - \text{AD}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_h(\text{WR} - \text{AD}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

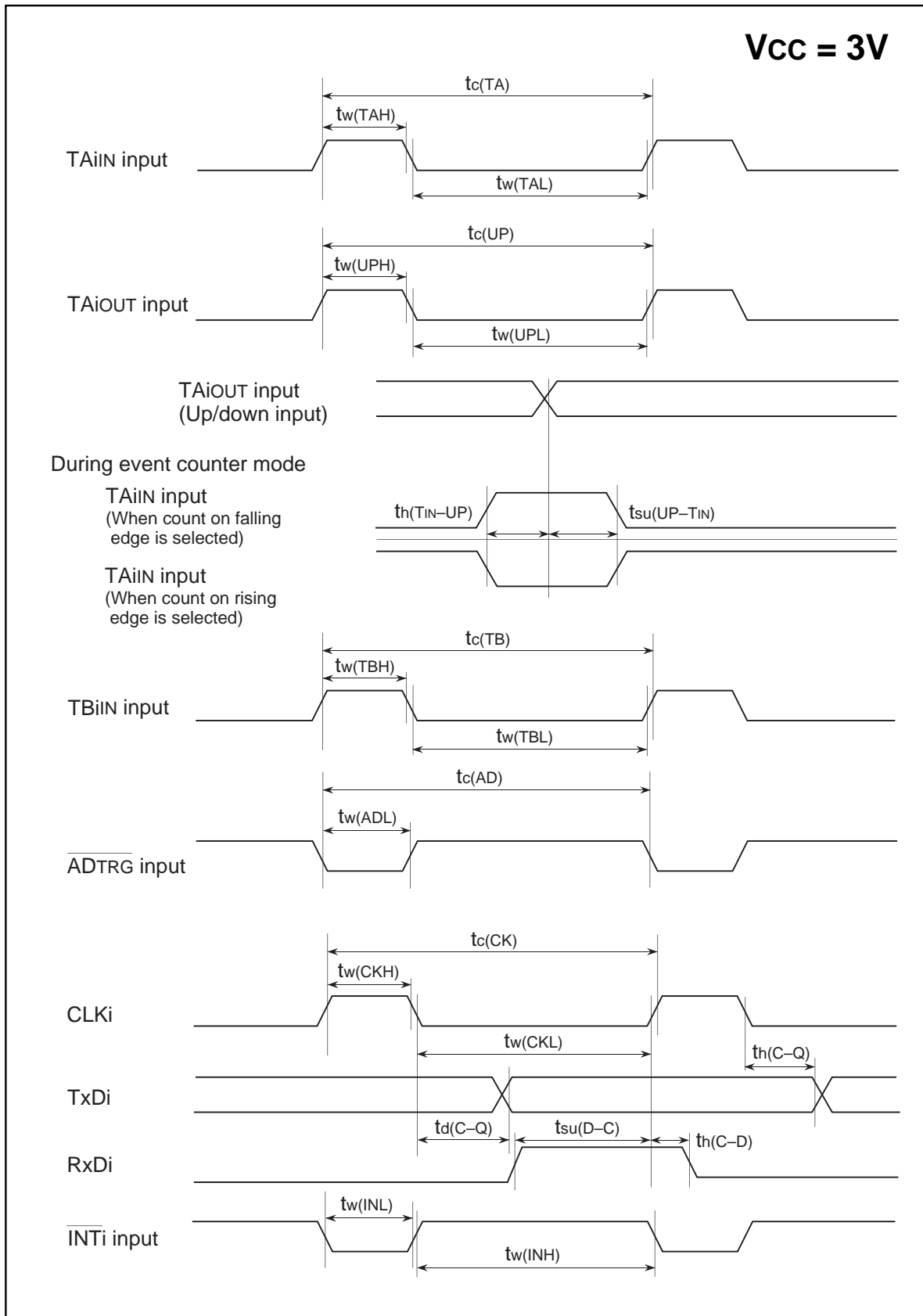
$$t_h(\text{RD} - \text{CS}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_h(\text{WR} - \text{CS}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_d(\text{DB} - \text{WR}) = \frac{10^9 \times 3}{f(\text{BCLK}) \times 2} - 80 \quad [\text{ns}]$$

$$t_h(\text{WR} - \text{DB}) = \frac{10^9}{f(\text{BCLK}) \times 2} \quad [\text{ns}]$$

$$t_d(\text{AD} - \text{ALE}) = \frac{10^9}{f(\text{BCLK}) \times 2} - 60 \quad [\text{ns}]$$

Timing ( $V_{CC} = 3V$ )Figure 1.24.7.  $V_{CC}=3V$  timing diagram (1)

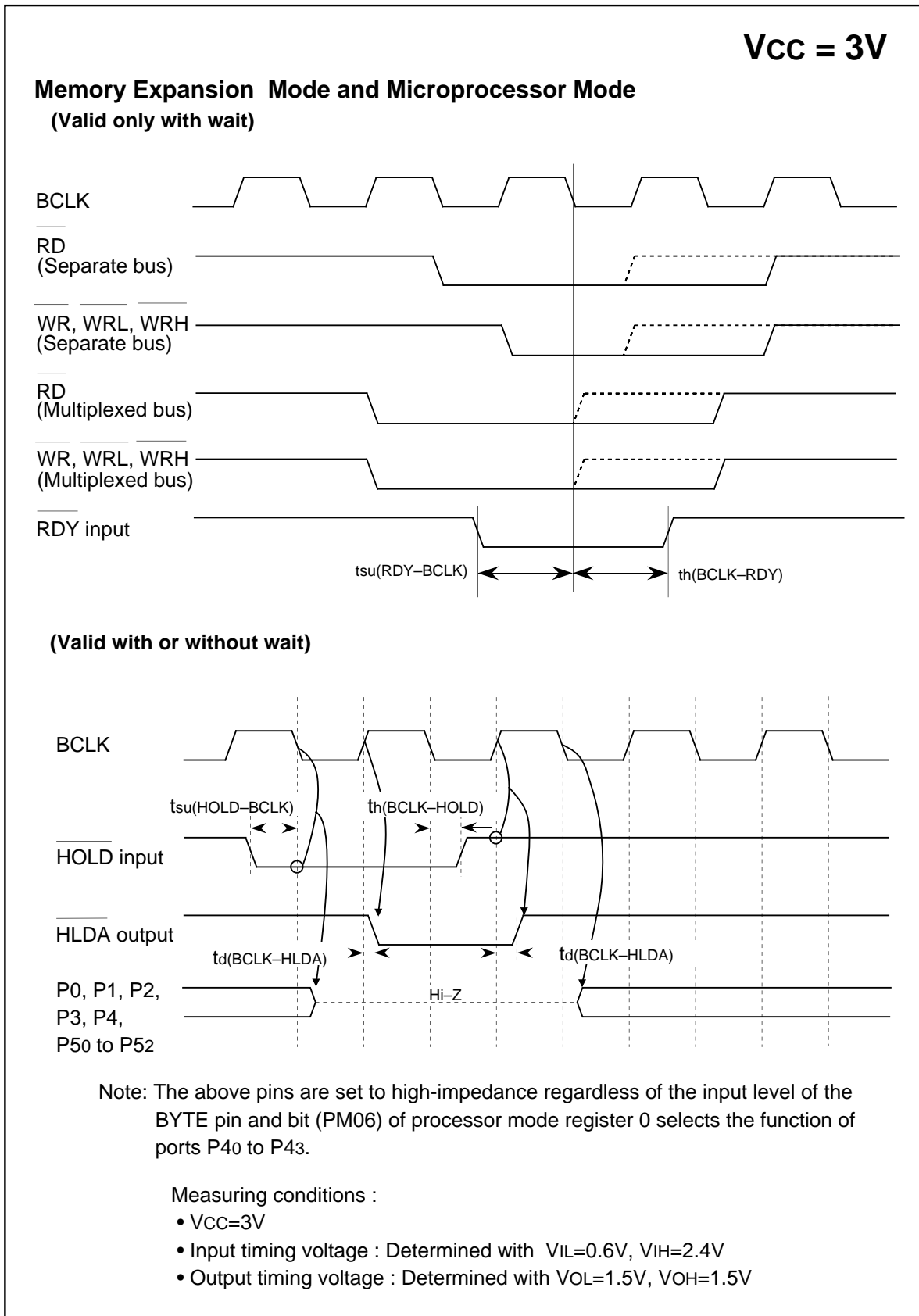
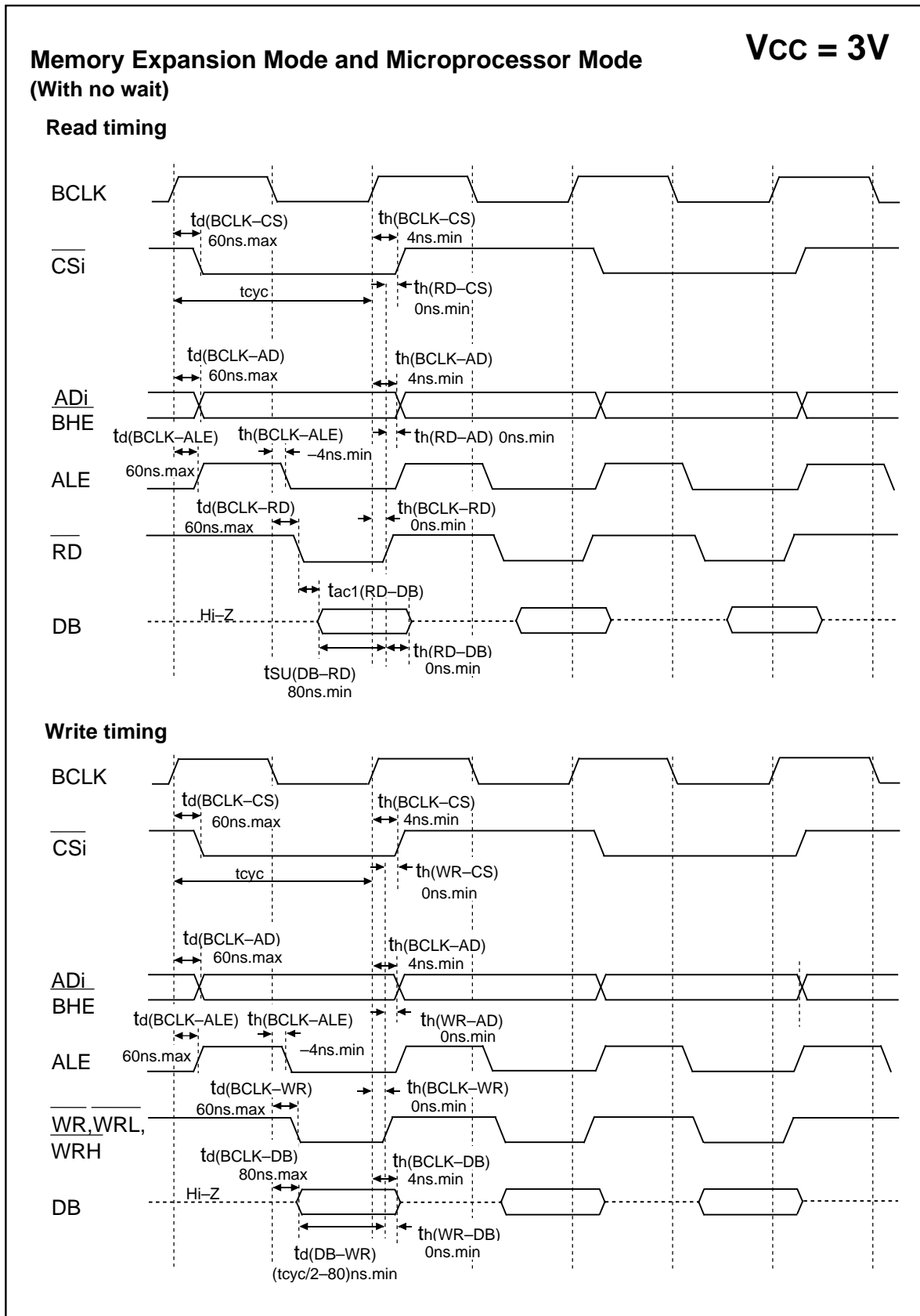


Figure 1.24.8. Vcc=3V timing diagram (2)



Timing ( $V_{CC} = 3V$ )Figure 1.24.9.  $V_{CC}=3V$  timing diagram (3)

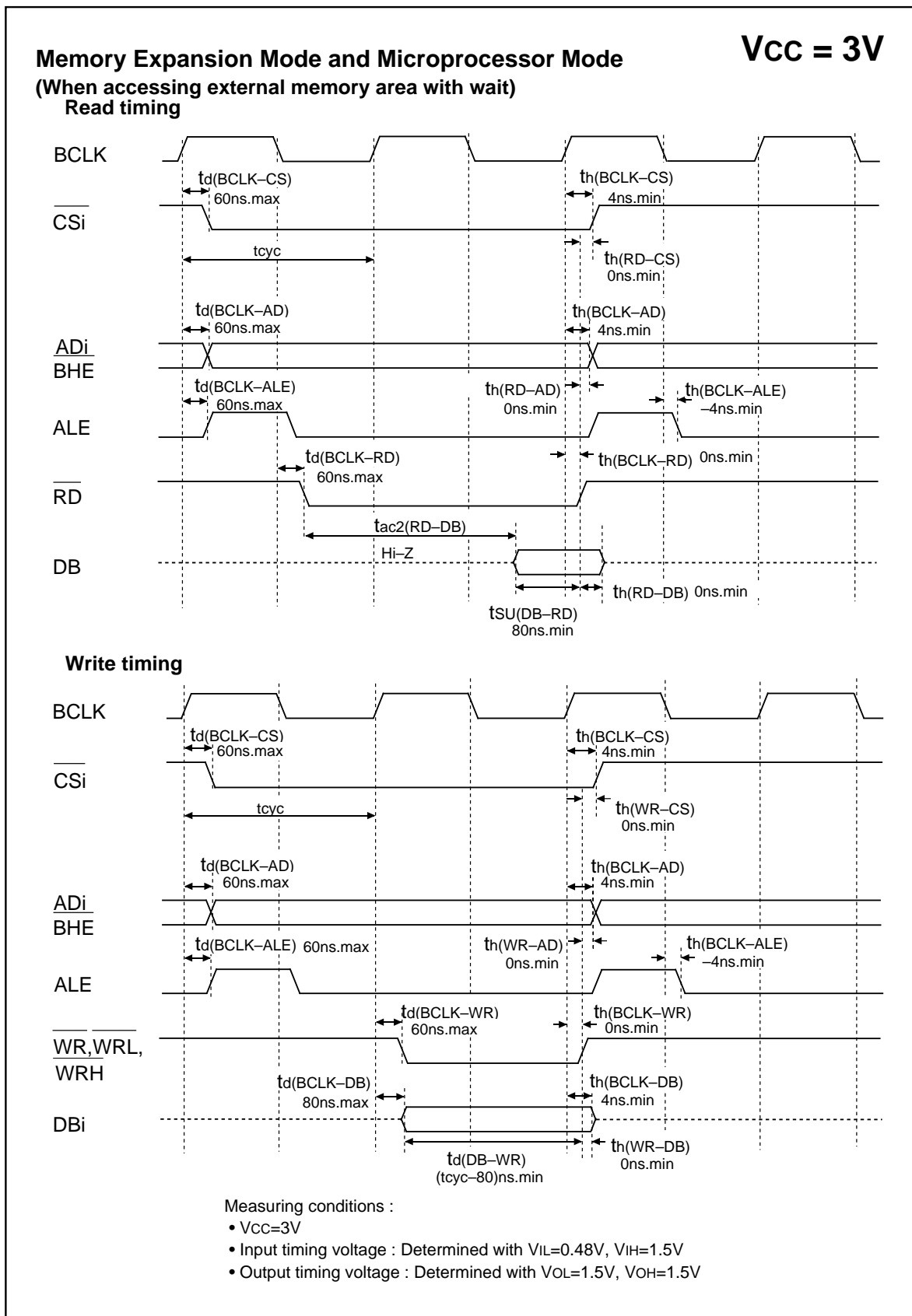
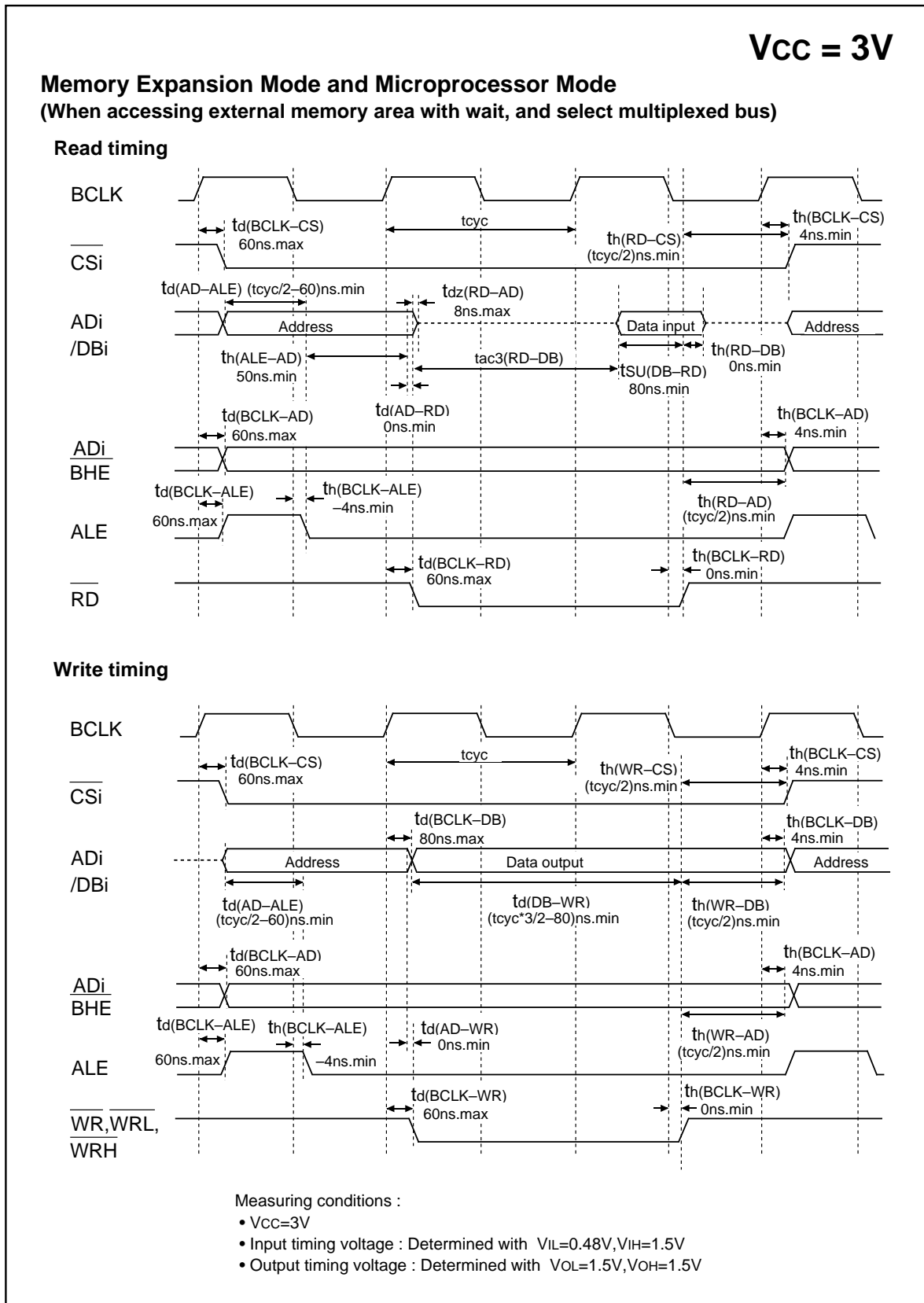


Figure 1.24.10. Vcc=3V timing diagram (4)

Timing ( $V_{CC} = 3V$ )Figure 1.24.11.  $V_{CC}=3V$  timing diagram (5)

GZZ—SH11—53B <71A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30610M8A-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product. Please carefully check the data on the EPROMs being submitted to Mitsubishi.

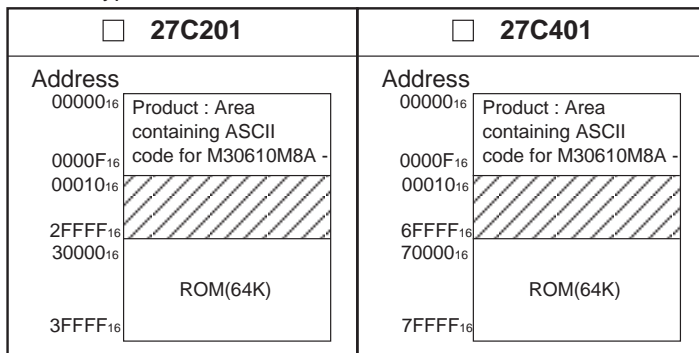
Microcomputer type No. :  M30610M8A-XXXFP  M30610M8A-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :



- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 00000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30610M8A-' is shown at right.  
 The data in this table must be written to address 00000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>Address</td><td></td><td></td></tr> <tr><td>00000<sub>16</sub></td><td>'M'</td><td>= 4D<sub>16</sub></td></tr> <tr><td>00001<sub>16</sub></td><td>'3'</td><td>= 33<sub>16</sub></td></tr> <tr><td>00002<sub>16</sub></td><td>'0'</td><td>= 30<sub>16</sub></td></tr> <tr><td>00003<sub>16</sub></td><td>'6'</td><td>= 36<sub>16</sub></td></tr> <tr><td>00004<sub>16</sub></td><td>'1'</td><td>= 31<sub>16</sub></td></tr> <tr><td>00005<sub>16</sub></td><td>'0'</td><td>= 30<sub>16</sub></td></tr> <tr><td>00006<sub>16</sub></td><td>'M'</td><td>= 4D<sub>16</sub></td></tr> <tr><td>00007<sub>16</sub></td><td>'8'</td><td>= 38<sub>16</sub></td></tr> </table>	Address			00000 <sub>16</sub>	'M'	= 4D <sub>16</sub>	00001 <sub>16</sub>	'3'	= 33 <sub>16</sub>	00002 <sub>16</sub>	'0'	= 30 <sub>16</sub>	00003 <sub>16</sub>	'6'	= 36 <sub>16</sub>	00004 <sub>16</sub>	'1'	= 31 <sub>16</sub>	00005 <sub>16</sub>	'0'	= 30 <sub>16</sub>	00006 <sub>16</sub>	'M'	= 4D <sub>16</sub>	00007 <sub>16</sub>	'8'	= 38 <sub>16</sub>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>Address</td><td></td><td></td></tr> <tr><td>00008<sub>16</sub></td><td>'A'</td><td>= 41<sub>16</sub></td></tr> <tr><td>00009<sub>16</sub></td><td>'—'</td><td>= 2D<sub>16</sub></td></tr> <tr><td>0000A<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>0000B<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>0000C<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>0000D<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>0000E<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>0000F<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> </table>	Address			00008 <sub>16</sub>	'A'	= 41 <sub>16</sub>	00009 <sub>16</sub>	'—'	= 2D <sub>16</sub>	0000A <sub>16</sub>		FF <sub>16</sub>	0000B <sub>16</sub>		FF <sub>16</sub>	0000C <sub>16</sub>		FF <sub>16</sub>	0000D <sub>16</sub>		FF <sub>16</sub>	0000E <sub>16</sub>		FF <sub>16</sub>	0000F <sub>16</sub>		FF <sub>16</sub>	
Address																																																								
00000 <sub>16</sub>	'M'	= 4D <sub>16</sub>																																																						
00001 <sub>16</sub>	'3'	= 33 <sub>16</sub>																																																						
00002 <sub>16</sub>	'0'	= 30 <sub>16</sub>																																																						
00003 <sub>16</sub>	'6'	= 36 <sub>16</sub>																																																						
00004 <sub>16</sub>	'1'	= 31 <sub>16</sub>																																																						
00005 <sub>16</sub>	'0'	= 30 <sub>16</sub>																																																						
00006 <sub>16</sub>	'M'	= 4D <sub>16</sub>																																																						
00007 <sub>16</sub>	'8'	= 38 <sub>16</sub>																																																						
Address																																																								
00008 <sub>16</sub>	'A'	= 41 <sub>16</sub>																																																						
00009 <sub>16</sub>	'—'	= 2D <sub>16</sub>																																																						
0000A <sub>16</sub>		FF <sub>16</sub>																																																						
0000B <sub>16</sub>		FF <sub>16</sub>																																																						
0000C <sub>16</sub>		FF <sub>16</sub>																																																						
0000D <sub>16</sub>		FF <sub>16</sub>																																																						
0000E <sub>16</sub>		FF <sub>16</sub>																																																						
0000F <sub>16</sub>		FF <sub>16</sub>																																																						

GZZ—SH11—53B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30610M8A-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

The ASCII code for the type No. can be written to EPROM addresses 0000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30610M8A- '	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30610M8A- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in. Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :       M30610M8A-XXXFP       M30610M8A-XXXGP

File code :                      

--	--	--	--	--	--	--	--

 (hex)

Mask file name :                

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30610M8A-XXXFP, submit the 100P6S mark specification sheet. For the M30610M8A-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of XIN-XOUT oscillation circuit is used?
- |   |   |
|---|---|
| <input type="checkbox"/> Ceramic resonator    | <input type="checkbox"/> Quartz-crystal oscillator      |
| <input type="checkbox"/> External clock input | <input type="checkbox"/> Other (                      ) |

What frequency do you use?  
 f(XIN) =  MHz

GZZ—SH11—53B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
MICROCOMPUTER M30610M8A-XXXFP/GP  
MASK ROM CONFIRMATION FORM**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?  
 Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                      )
- What frequency do you use?  
f(XCIN) =  kHz
- (3) Which operation mode do you use?  
 Single-chip mode       Memory expansion mode  
 Microprocessor mode
- (4) Which operating ambient temperature do you use?  
 -10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C
- (5) Which operating supply voltage do you use?  
 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

GZZ—SH11—52B <71A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30610MAA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product. Please carefully check the data on the EPROMs being submitted to Mitsubishi.

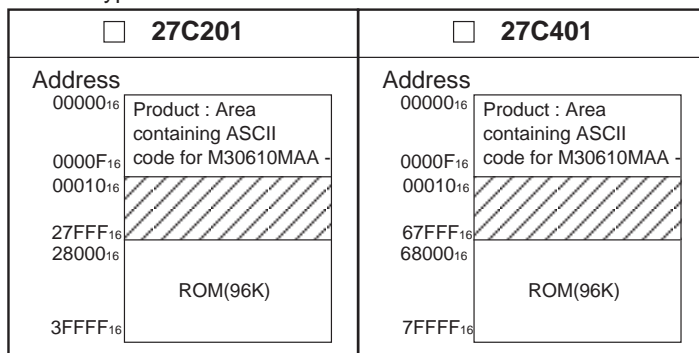
Microcomputer type No. :  M30610MAA-XXXFP  M30610MAA-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :



- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 0000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30610MAA-' is shown at right.  
 The data in this table must be written to address 0000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>Address</td><td></td><td></td></tr> <tr><td>0000<sub>16</sub></td><td>'M'</td><td>= 4D<sub>16</sub></td></tr> <tr><td>0001<sub>16</sub></td><td>'3'</td><td>= 33<sub>16</sub></td></tr> <tr><td>0002<sub>16</sub></td><td>'0'</td><td>= 30<sub>16</sub></td></tr> <tr><td>0003<sub>16</sub></td><td>'6'</td><td>= 36<sub>16</sub></td></tr> <tr><td>0004<sub>16</sub></td><td>'1'</td><td>= 31<sub>16</sub></td></tr> <tr><td>0005<sub>16</sub></td><td>'0'</td><td>= 30<sub>16</sub></td></tr> <tr><td>0006<sub>16</sub></td><td>'M'</td><td>= 4D<sub>16</sub></td></tr> <tr><td>0007<sub>16</sub></td><td>'A'</td><td>= 41<sub>16</sub></td></tr> </table>	Address			0000 <sub>16</sub>	'M'	= 4D <sub>16</sub>	0001 <sub>16</sub>	'3'	= 33 <sub>16</sub>	0002 <sub>16</sub>	'0'	= 30 <sub>16</sub>	0003 <sub>16</sub>	'6'	= 36 <sub>16</sub>	0004 <sub>16</sub>	'1'	= 31 <sub>16</sub>	0005 <sub>16</sub>	'0'	= 30 <sub>16</sub>	0006 <sub>16</sub>	'M'	= 4D <sub>16</sub>	0007 <sub>16</sub>	'A'	= 41 <sub>16</sub>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>Address</td><td></td><td></td></tr> <tr><td>0000<sub>16</sub></td><td>'A'</td><td>= 41<sub>16</sub></td></tr> <tr><td>0009<sub>16</sub></td><td>'—'</td><td>= 2D<sub>16</sub></td></tr> <tr><td>000A<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>000B<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>000C<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>000D<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>000E<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> <tr><td>000F<sub>16</sub></td><td></td><td>FF<sub>16</sub></td></tr> </table>	Address			0000 <sub>16</sub>	'A'	= 41 <sub>16</sub>	0009 <sub>16</sub>	'—'	= 2D <sub>16</sub>	000A <sub>16</sub>		FF <sub>16</sub>	000B <sub>16</sub>		FF <sub>16</sub>	000C <sub>16</sub>		FF <sub>16</sub>	000D <sub>16</sub>		FF <sub>16</sub>	000E <sub>16</sub>		FF <sub>16</sub>	000F <sub>16</sub>		FF <sub>16</sub>	
Address																																																								
0000 <sub>16</sub>	'M'	= 4D <sub>16</sub>																																																						
0001 <sub>16</sub>	'3'	= 33 <sub>16</sub>																																																						
0002 <sub>16</sub>	'0'	= 30 <sub>16</sub>																																																						
0003 <sub>16</sub>	'6'	= 36 <sub>16</sub>																																																						
0004 <sub>16</sub>	'1'	= 31 <sub>16</sub>																																																						
0005 <sub>16</sub>	'0'	= 30 <sub>16</sub>																																																						
0006 <sub>16</sub>	'M'	= 4D <sub>16</sub>																																																						
0007 <sub>16</sub>	'A'	= 41 <sub>16</sub>																																																						
Address																																																								
0000 <sub>16</sub>	'A'	= 41 <sub>16</sub>																																																						
0009 <sub>16</sub>	'—'	= 2D <sub>16</sub>																																																						
000A <sub>16</sub>		FF <sub>16</sub>																																																						
000B <sub>16</sub>		FF <sub>16</sub>																																																						
000C <sub>16</sub>		FF <sub>16</sub>																																																						
000D <sub>16</sub>		FF <sub>16</sub>																																																						
000E <sub>16</sub>		FF <sub>16</sub>																																																						
000F <sub>16</sub>		FF <sub>16</sub>																																																						

GZZ—SH11—52B <71A1>

**mitsubishi electric single-chip 16-bit  
 microcomputer M30610MAA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

The ASCII code for the type No. can be written to EPROM addresses 00000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30610MAA- '	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30610MAA- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in. Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :  M30610MAA-XXXFP  M30610MAA-XXXGP

File code : 

--	--	--	--	--	--	--	--

 (hex)

Mask file name : 

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30610MAA-XXXFP, submit the 100P6S mark specification sheet. For the M30610MAA-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of X<sub>IN</sub>-X<sub>OUT</sub> oscillation circuit is used?  
 Ceramic resonator  Quartz-crystal oscillator  
 External clock input  Other (                    )

What frequency do you use?  
 f(X<sub>IN</sub>) = 

--

 MHz



GZZ—SH11—52B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
MICROCOMPUTER M30610MAA-XXXFP/GP  
MASK ROM CONFIRMATION FORM**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?  
 Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                      )  
What frequency do you use?  
f(XCIN) =  kHz
- (3) Which operation mode do you use?  
 Single-chip mode       Memory expansion mode  
 Microprocessor mode
- (4) Which operating ambient temperature do you use?  
 -10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C
- (5) Which operating supply voltage do you use?  
 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

GZZ—SH11—51B <71A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30610MCA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product. Please carefully check the data on the EPROMs being submitted to Mitsubishi.

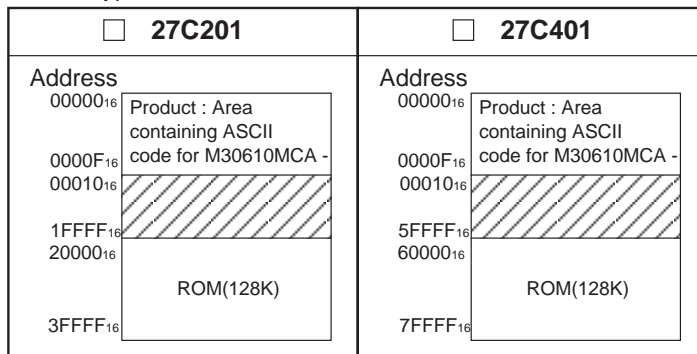
Microcomputer type No. :  M30610MCA-XXXFP  M30610MCA-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :



- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 00000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30610MCA-' is shown at right. The data in this table must be written to address 00000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="width: 20%; text-align: right;">00000<sub>16</sub></td><td>'M' = 4D<sub>16</sub></td></tr> <tr><td style="text-align: right;">00001<sub>16</sub></td><td>'3' = 33<sub>16</sub></td></tr> <tr><td style="text-align: right;">00002<sub>16</sub></td><td>'0' = 30<sub>16</sub></td></tr> <tr><td style="text-align: right;">00003<sub>16</sub></td><td>'6' = 36<sub>16</sub></td></tr> <tr><td style="text-align: right;">00004<sub>16</sub></td><td>'1' = 31<sub>16</sub></td></tr> <tr><td style="text-align: right;">00005<sub>16</sub></td><td>'0' = 30<sub>16</sub></td></tr> <tr><td style="text-align: right;">00006<sub>16</sub></td><td>'M' = 4D<sub>16</sub></td></tr> <tr><td style="text-align: right;">00007<sub>16</sub></td><td>'C' = 43<sub>16</sub></td></tr> </table>	00000 <sub>16</sub>	'M' = 4D <sub>16</sub>	00001 <sub>16</sub>	'3' = 33 <sub>16</sub>	00002 <sub>16</sub>	'0' = 30 <sub>16</sub>	00003 <sub>16</sub>	'6' = 36 <sub>16</sub>	00004 <sub>16</sub>	'1' = 31 <sub>16</sub>	00005 <sub>16</sub>	'0' = 30 <sub>16</sub>	00006 <sub>16</sub>	'M' = 4D <sub>16</sub>	00007 <sub>16</sub>	'C' = 43 <sub>16</sub>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td style="width: 20%; text-align: right;">00008<sub>16</sub></td><td>'A' = 41<sub>16</sub></td></tr> <tr><td style="text-align: right;">00009<sub>16</sub></td><td>'—' = 2D<sub>16</sub></td></tr> <tr><td style="text-align: right;">0000A<sub>16</sub></td><td>FF<sub>16</sub></td></tr> <tr><td style="text-align: right;">0000B<sub>16</sub></td><td>FF<sub>16</sub></td></tr> <tr><td style="text-align: right;">0000C<sub>16</sub></td><td>FF<sub>16</sub></td></tr> <tr><td style="text-align: right;">0000D<sub>16</sub></td><td>FF<sub>16</sub></td></tr> <tr><td style="text-align: right;">0000E<sub>16</sub></td><td>FF<sub>16</sub></td></tr> <tr><td style="text-align: right;">0000F<sub>16</sub></td><td>FF<sub>16</sub></td></tr> </table>	00008 <sub>16</sub>	'A' = 41 <sub>16</sub>	00009 <sub>16</sub>	'—' = 2D <sub>16</sub>	0000A <sub>16</sub>	FF <sub>16</sub>	0000B <sub>16</sub>	FF <sub>16</sub>	0000C <sub>16</sub>	FF <sub>16</sub>	0000D <sub>16</sub>	FF <sub>16</sub>	0000E <sub>16</sub>	FF <sub>16</sub>	0000F <sub>16</sub>	FF <sub>16</sub>
00000 <sub>16</sub>	'M' = 4D <sub>16</sub>																																
00001 <sub>16</sub>	'3' = 33 <sub>16</sub>																																
00002 <sub>16</sub>	'0' = 30 <sub>16</sub>																																
00003 <sub>16</sub>	'6' = 36 <sub>16</sub>																																
00004 <sub>16</sub>	'1' = 31 <sub>16</sub>																																
00005 <sub>16</sub>	'0' = 30 <sub>16</sub>																																
00006 <sub>16</sub>	'M' = 4D <sub>16</sub>																																
00007 <sub>16</sub>	'C' = 43 <sub>16</sub>																																
00008 <sub>16</sub>	'A' = 41 <sub>16</sub>																																
00009 <sub>16</sub>	'—' = 2D <sub>16</sub>																																
0000A <sub>16</sub>	FF <sub>16</sub>																																
0000B <sub>16</sub>	FF <sub>16</sub>																																
0000C <sub>16</sub>	FF <sub>16</sub>																																
0000D <sub>16</sub>	FF <sub>16</sub>																																
0000E <sub>16</sub>	FF <sub>16</sub>																																
0000F <sub>16</sub>	FF <sub>16</sub>																																

GZZ—SH11—51B <71A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30610MCA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

The ASCII code for the type No. can be written to EPROM addresses 00000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30610MCA- '	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30610MCA- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in.

Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :  M30610MCA-XXXFP  M30610MCA-XXXGP

File code : 

--	--	--	--	--	--	--	--

 (hex)

Mask file name : 

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30610MCA-XXXFP, submit the 100P6S mark specification sheet. For the M30610MCA-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of XIN-XOUT oscillation circuit is used?  
 Ceramic resonator  Quartz-crystal oscillator  
 External clock input  Other ( )

What frequency do you use?

f(XIN) =  MHz

GZZ—SH11—51B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
MICROCOMPUTER M30610MCA-XXXFP/GP  
MASK ROM CONFIRMATION FORM**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?  
 Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                      )
- What frequency do you use?  
f(XCIN) =  kHz
- (3) Which operation mode do you use?  
 Single-chip mode       Memory expansion mode  
 Microprocessor mode
- (4) Which operating ambient temperature do you use?  
 -10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C
- (5) Which operating supply voltage do you use?  
 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

GZZ—SH12—35B <79A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612M4A-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product. Please carefully check the data on the EPROMs being submitted to Mitsubishi.

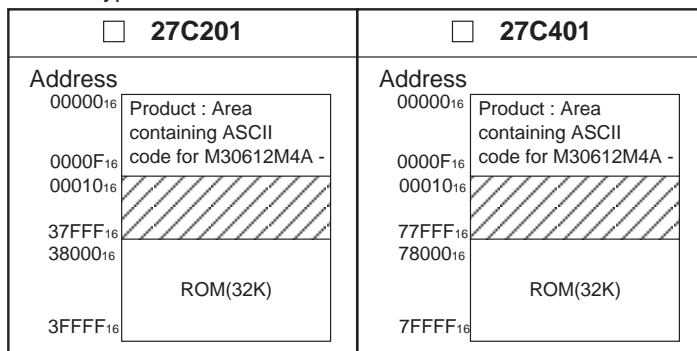
Microcomputer type No. :  M30612M4A-XXXFP  M30612M4A-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :



- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 00000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30612M4A-' is shown at right.  
 The data in this table must be written to address 00000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

Address	
00000 <sub>16</sub>	'M' = 4D <sub>16</sub>
00001 <sub>16</sub>	'3' = 33 <sub>16</sub>
00002 <sub>16</sub>	'0' = 30 <sub>16</sub>
00003 <sub>16</sub>	'6' = 36 <sub>16</sub>
00004 <sub>16</sub>	'1' = 31 <sub>16</sub>
00005 <sub>16</sub>	'2' = 32 <sub>16</sub>
00006 <sub>16</sub>	'M' = 4D <sub>16</sub>
00007 <sub>16</sub>	'4' = 34 <sub>16</sub>

Address	
00008 <sub>16</sub>	'A' = 41 <sub>16</sub>
00009 <sub>16</sub>	'_' = 2D <sub>16</sub>
0000A <sub>16</sub>	FF <sub>16</sub>
0000B <sub>16</sub>	FF <sub>16</sub>
0000C <sub>16</sub>	FF <sub>16</sub>
0000D <sub>16</sub>	FF <sub>16</sub>
0000E <sub>16</sub>	FF <sub>16</sub>
0000F <sub>16</sub>	FF <sub>16</sub>

GZZ— SH12— 35B <79A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612M4A-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

The ASCII code for the type No. can be written to EPROM addresses 00000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30612M4A- '	△ .SECTION△ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30612M4A- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in.  
 Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :  M30612M4A-XXXFP  M30612M4A-XXXGP

File code : 

--	--	--	--	--	--	--	--

 (hex)

Mask file name : 

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30612M4A-XXXFP, submit the 100P6S mark specification sheet. For the M30612M4A-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of X<sub>IN</sub>-X<sub>OUT</sub> oscillation circuit is used?  
 Ceramic resonator  Quartz-crystal oscillator  
 External clock input  Other ( )

What frequency do you use?

f(X<sub>IN</sub>) =  MHz

GZZ— SH12— 35B <79A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
MICROCOMPUTER M30612M4A-XXXFP/GP  
MASK ROM CONFIRMATION FORM**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?
- Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                    )

What frequency do you use?

f(XCIN) =  kHz

- (3) Which operation mode do you use?
- Single-chip mode       Memory expansion mode  
 Microprocessor mode

- (4) Which operating ambient temperature do you use?
- 10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C

- (5) Which operating supply voltage do you use?
- 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

GZZ—SH12—34B <79A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612M8A-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product.  
 Please carefully check the data on the EPROMs being submitted to Mitsubishi.

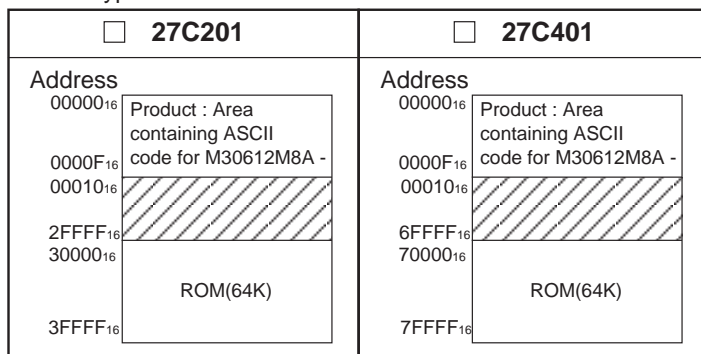
Microcomputer type No. :  M30612M8A-XXXFP  M30612M8A-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :



- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 0000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30612M8A-' is shown at right.  
 The data in this table must be written to address 00000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

Address	Address	Address
0000 <sub>16</sub> 'M' = 4D <sub>16</sub>	00008 <sub>16</sub> 'A' = 41 <sub>16</sub>	
00001 <sub>16</sub> '3' = 33 <sub>16</sub>	00009 <sub>16</sub> '-' = 2D <sub>16</sub>	
00002 <sub>16</sub> '0' = 30 <sub>16</sub>	0000A <sub>16</sub> FF <sub>16</sub>	
00003 <sub>16</sub> '6' = 36 <sub>16</sub>	0000B <sub>16</sub> FF <sub>16</sub>	
00004 <sub>16</sub> '1' = 31 <sub>16</sub>	0000C <sub>16</sub> FF <sub>16</sub>	
00005 <sub>16</sub> '2' = 32 <sub>16</sub>	0000D <sub>16</sub> FF <sub>16</sub>	
00006 <sub>16</sub> 'M' = 4D <sub>16</sub>	0000E <sub>16</sub> FF <sub>16</sub>	
00007 <sub>16</sub> '8' = 38 <sub>16</sub>	0000F <sub>16</sub> FF <sub>16</sub>	



GZZ—SH12—34B <79A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612M8A-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

The ASCII code for the type No. can be written to EPROM addresses 00000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30612M8A- '	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30612M8A- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in. Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :  M30612M8A-XXXFP  M30612M8A-XXXGP

File code : 

--	--	--	--	--	--	--	--

 (hex)

Mask file name : 

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30612M8A-XXXFP, submit the 100P6S mark specification sheet. For the M30612M8A-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of X<sub>IN</sub>-X<sub>OUT</sub> oscillation circuit is used?  
 Ceramic resonator  Quartz-crystal oscillator  
 External clock input  Other ( )

What frequency do you use?  
 f(X<sub>IN</sub>) = 

--

 MHz

GZZ—SH12—34B <79A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
MICROCOMPUTER M30612M8A-XXXFP/GP  
MASK ROM CONFIRMATION FORM**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?
- Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                    )

What frequency do you use?

f(XCIN) =  kHz

- (3) Which operation mode do you use?
- Single-chip mode       Memory expansion mode  
 Microprocessor mode

- (4) Which operating ambient temperature do you use?
- 10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C

- (5) Which operating supply voltage do you use?
- 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

GZZ—SH12—55B <71A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612MAA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product.  
 Please carefully check the data on the EPROMs being submitted to Mitsubishi.

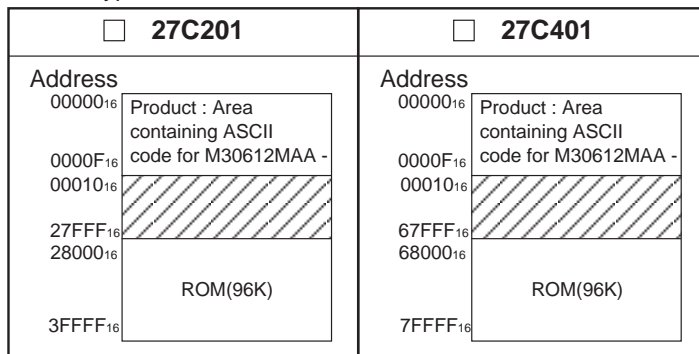
Microcomputer type No. :  M30612MAA-XXXFP  M30612MAA-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :



- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 00000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30612MAA-' is shown at right.  
 The data in this table must be written to address 00000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

Address	
0000 <sub>16</sub>	'M' = 4D <sub>16</sub>
00001 <sub>16</sub>	'3' = 33 <sub>16</sub>
00002 <sub>16</sub>	'0' = 30 <sub>16</sub>
00003 <sub>16</sub>	'6' = 36 <sub>16</sub>
00004 <sub>16</sub>	'1' = 31 <sub>16</sub>
00005 <sub>16</sub>	'2' = 32 <sub>16</sub>
00006 <sub>16</sub>	'M' = 4D <sub>16</sub>
00007 <sub>16</sub>	'A' = 41 <sub>16</sub>

Address	
00008 <sub>16</sub>	'A' = 41 <sub>16</sub>
00009 <sub>16</sub>	'—' = 2D <sub>16</sub>
0000A <sub>16</sub>	FF <sub>16</sub>
0000B <sub>16</sub>	FF <sub>16</sub>
0000C <sub>16</sub>	FF <sub>16</sub>
0000D <sub>16</sub>	FF <sub>16</sub>
0000E <sub>16</sub>	FF <sub>16</sub>
0000F <sub>16</sub>	FF <sub>16</sub>

GZZ—SH11—55B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612MAA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

The ASCII code for the type No. can be written to EPROM addresses 00000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30612MAA- '	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30612MAA- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in.

Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :  M30612MAA-XXXFP  M30612MAA-XXXGP

File code : 

--	--	--	--	--	--	--	--

 (hex)

Mask file name : 

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30612MAA-XXXFP, submit the 100P6S mark specification sheet. For the M30612MAA-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of XIN-XOUT oscillation circuit is used?  
 Ceramic resonator  Quartz-crystal oscillator  
 External clock input  Other (                      )

What frequency do you use?  
 f(XIN) = 

--

 MHz

GZZ—SH11—55B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
MICROCOMPUTER M30612MAA-XXXFP/GP  
MASK ROM CONFIRMATION FORM**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?  
 Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                      )

What frequency do you use?

f(XCIN) =  kHz

- (3) Which operation mode do you use?  
 Single-chip mode       Memory expansion mode  
 Microprocessor mode

- (4) Which operating ambient temperature do you use?  
 -10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C

- (5) Which operating supply voltage do you use?  
 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

※ 4. Special item (Indicate none if there is no specified item)

GZZ—SH11—54B <71A1>

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612MCA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

Mask ROM number	
-----------------	--

Receipt	Date :	
	Section head signature	Supervisor signature

Note : Please complete all items marked ※.

※ Customer	Company name	TEL ( )	Issuance signature	Submitted by	Supervisor
	Date issued	Date :			

※ 1. Check sheet

Name the product you order, and choose which to give in, EPROMs or floppy disks.  
 If you order by means of EPROMs, three sets of EPROMs are required per pattern. If you order by means of floppy disks, one floppy disk is required per pattern.

In the case of EPROMs

Mitsubishi will create the mask using the data on the EPROMs supplied, providing the data is the same on at least two of those sets. Mitsubishi will, therefore, only accept liability if there is any discrepancy between the data on the EPROM sets and the ROM data written to the product. Please carefully check the data on the EPROMs being submitted to Mitsubishi.

Microcomputer type No. :  M30612MCA-XXXFP  M30612MCA-XXXGP

Checksum code for total EPROM area : 

--	--	--	--

 (hex)

EPROM type :

<input type="checkbox"/> <b>27C201</b> Address 00000 <sub>16</sub> Product : Area containing ASCII code for M30612MCA - 0000F <sub>16</sub> 00010 <sub>16</sub> 1FFFF <sub>16</sub> 20000 <sub>16</sub> ROM(128K) 3FFFF <sub>16</sub>	<input type="checkbox"/> <b>27C401</b> Address 00000 <sub>16</sub> Product : Area containing ASCII code for M30612MCA - 0000F <sub>16</sub> 00010 <sub>16</sub> 5FFFF <sub>16</sub> 60000 <sub>16</sub> ROM(128K) 7FFFF <sub>16</sub>
--	--

- (1) Write "FF<sub>16</sub>" to the lined area.
- (2) The area from 00000<sub>16</sub> to 0000F<sub>16</sub> is for storing data on the product type name.  
 The ASCII code for 'M30612MCA-' is shown at right.  
 The data in this table must be written to address 00000<sub>16</sub> to 0000F<sub>16</sub>.  
 Both address and data are shown in hex.

Address	Address	Address
00000 <sub>16</sub> 'M' = 4D <sub>16</sub>	00008 <sub>16</sub> 'A' = 41 <sub>16</sub>	
00001 <sub>16</sub> '3' = 33 <sub>16</sub>	00009 <sub>16</sub> '—' = 2D <sub>16</sub>	
00002 <sub>16</sub> '0' = 30 <sub>16</sub>	0000A <sub>16</sub> FF <sub>16</sub>	
00003 <sub>16</sub> '6' = 36 <sub>16</sub>	0000B <sub>16</sub> FF <sub>16</sub>	
00004 <sub>16</sub> '1' = 31 <sub>16</sub>	0000C <sub>16</sub> FF <sub>16</sub>	
00005 <sub>16</sub> '2' = 32 <sub>16</sub>	0000D <sub>16</sub> FF <sub>16</sub>	
00006 <sub>16</sub> 'M' = 4D <sub>16</sub>	0000E <sub>16</sub> FF <sub>16</sub>	
00007 <sub>16</sub> 'C' = 43 <sub>16</sub>	0000F <sub>16</sub> FF <sub>16</sub>	

GZZ—SH11—54B <71A1>

Mask ROM number	
-----------------	--

**MITSUBISHI ELECTRIC SINGLE-CHIP 16-BIT  
 MICROCOMPUTER M30612MCA-XXXFP/GP  
 MASK ROM CONFIRMATION FORM**

The ASCII code for the type No. can be written to EPROM addresses 00000<sub>16</sub> to 0000F<sub>16</sub> by specifying the pseudo-instructions for the respective EPROM type shown in the following table at the beginning of the assembler source program.

EPROM type	27C201	27C401
Code entered in source program	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 0C0000H △ .BYTE △ ' M30612MCA- '	△ .SECTION△ ASCII CODE, ROM DATA △ .ORG △ 080000H △ .BYTE △ ' M30612MCA- '

Note: The ROM cannot be processed if the type No. written to the EPROM does not match the type No. in the check sheet.

In the case of floppy disks

Mitsubishi processes the mask files generated by the mask file generation utilities out of those held on the floppy disks you give in to us, and forms them into masks. Hence, we assume liability provided that there is any discrepancy between the contents of these mask files and the ROM data to be burned into products we produce. Check thoroughly the contents of the mask files you give in. Prepare 3.5 inches 2HD(IBM format) floppy disks. And store only one mask file in a floppy disk.

Microcomputer type No. :       M30612MCA-XXXFP       M30612MCA-XXXGP

File code :                      

--	--	--	--	--	--	--	--

 (hex)

Mask file name :                

--	--	--	--	--	--	--	--

 .MSK (alpha-numeric 8-digit)

※2. Mark specification

The mark specification differs according to the type of package. After entering the mark specification on the separate mark specification sheet (for each package), attach that sheet to this masking check sheet for submission to Mitsubishi.

For the M30612MCA-XXXFP, submit the 100P6S mark specification sheet. For the M30612MCA-XXXGP, submit the 100P6Q mark specification sheet.

※3. Usage Conditions

For our reference when of testing our products, please reply to the following questions about the usage of the products you ordered.

- (1) Which kind of X<sub>IN</sub>-X<sub>OUT</sub> oscillation circuit is used?  
 Ceramic resonator                       Quartz-crystal oscillator  
 External clock input                       Other (                      )

What frequency do you use?  
 f(X<sub>IN</sub>) =  MHz

GZZ—SH11—54B <71A1>

Mask ROM number	
-----------------	--

**mitsubishi electric single-chip 16-bit  
microcomputer M30612MCA-XXXFP/GP  
mask rom confirmation form**

- (2) Which kind of XCIN-XCOUT oscillation circuit is used?  
 Ceramic resonator       Quartz-crystal oscillator  
 External clock input       Other (                      )
- What frequency do you use?  
f(XCIN) =  kHz
- (3) Which operation mode do you use?  
 Single-chip mode       Memory expansion mode  
 Microprocessor mode
- (4) Which operating ambient temperature do you use?  
 -10 °C to 75 °C       -20 °C to 75 °C       -40 °C to 75 °C  
 -10 °C to 85 °C       -20 °C to 85 °C       -40 °C to 85 °C
- (5) Which operating supply voltage do you use?  
 2.7V to 3.2V       3.2V to 3.7V       3.7V to 4.2V  
 4.2V to 4.7V       4.7V to 5.2V       5.2V to 5.5V

Thank you cooperation.

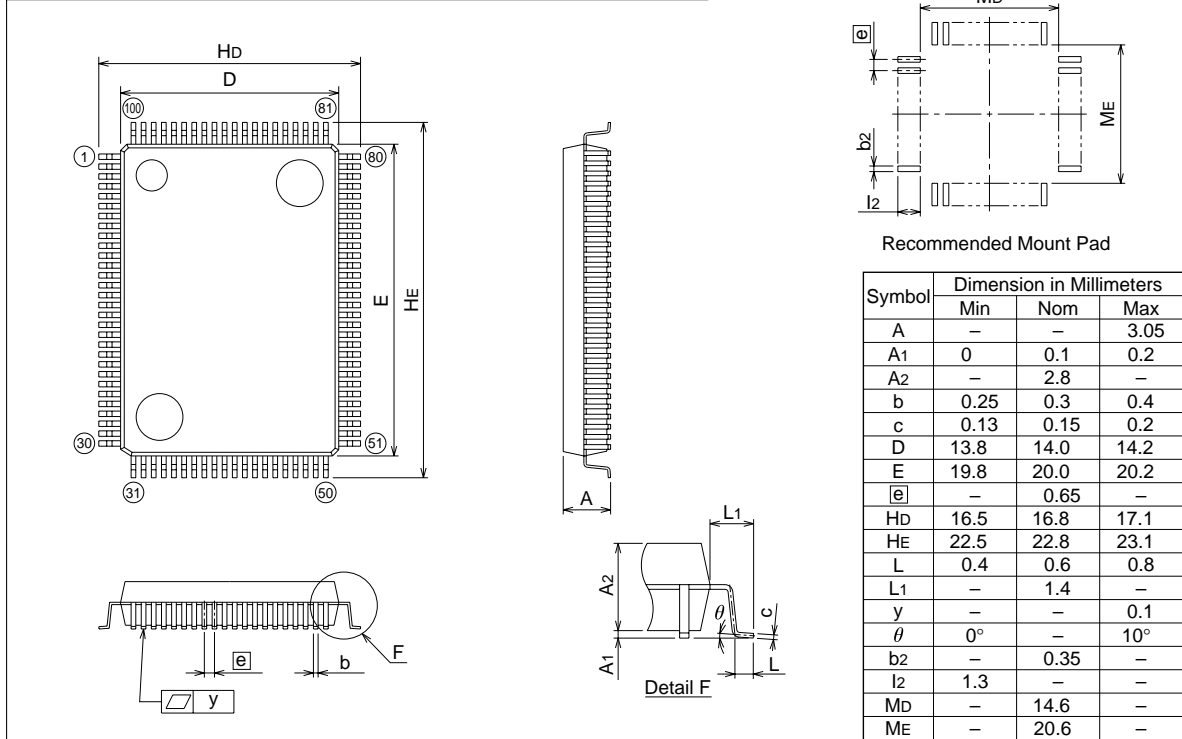
※ 4. Special item (Indicate none if there is no specified item)



**100P6S-A**

**Plastic 100pin 14x20mm body QFP**

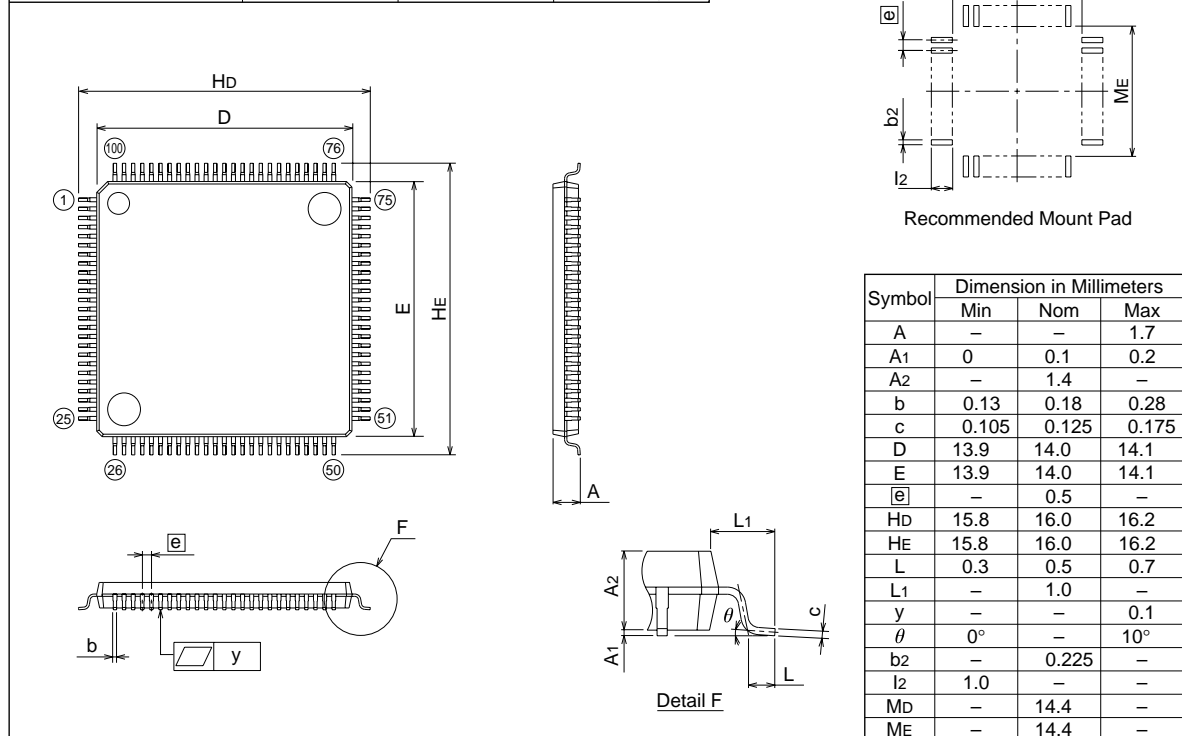
EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
QFP100-P-1420-0.65	-	1.58	Alloy 42




**100P6Q-A**

**Plastic 100pin 14x14mm body LQFP**

EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
LQFP100-P-1414-0.50	-	-	Cu Alloy

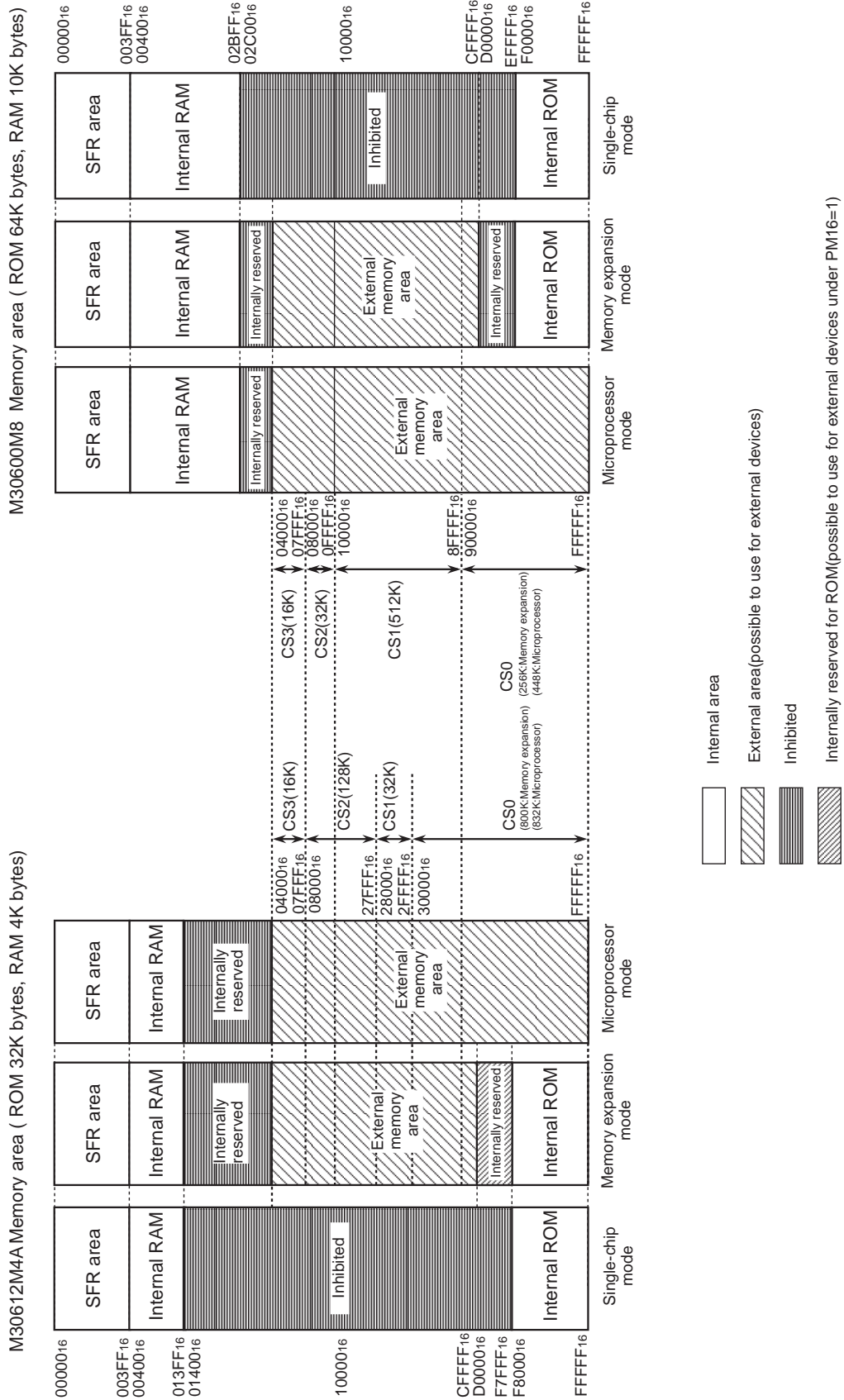


## Differences between M16C/61 group and M30600M8

Type name		M16C/61 group	M30600M8
Internal memory size	ROM	See Figure 4. ROM Expansion	64 K bytes
	RAM	4 K to 10 K bytes	10 K bytes
Chip select		CS0 30000 <sub>16</sub> to FFFFF <sub>16</sub> (besides internal area) CS1 28000 <sub>16</sub> to 2FFFF <sub>16</sub> CS2 08000 <sub>16</sub> to 27FFF <sub>16</sub> CS3 04000 <sub>16</sub> to 07FFF <sub>16</sub>	CS0 90000 <sub>16</sub> to FFFFF <sub>16</sub> (besides internal area) CS1 10000 <sub>16</sub> to 8FFFF <sub>16</sub> CS2 08000 <sub>16</sub> to 0FFFF <sub>16</sub> CS3 04000 <sub>16</sub> to 07FFF <sub>16</sub>
Internal area on memory expansion mode		SFR area 00000 <sub>16</sub> to 003FF <sub>16</sub> RAM area 00400 <sub>16</sub> to 03FFF <sub>16</sub> ROM area D0000 <sub>16</sub> to FFFFF <sub>16</sub> (PM16=0) ROM area F8000 <sub>16</sub> to FFFFF <sub>16</sub> (PM16=1) (Note)	SFR area 00000 <sub>16</sub> to 003FF <sub>16</sub> RAM area 00400 <sub>16</sub> to 03FFF <sub>16</sub> ROM area D0000 <sub>16</sub> to FFFFF <sub>16</sub> (FIX)
Serial I/O		3 channel (clocked SIO / UART) :2 channel (clocked SIO / UART / SIM)	2 channel (clocked SIO / UART)
Port P7 <sub>0</sub> to P7 <sub>3</sub> function		Port P7 <sub>0</sub> TXD <sub>2</sub> / TA0 <sub>OUT</sub> Port P7 <sub>1</sub> RXD <sub>2</sub> / TA0 <sub>IN</sub> Port P7 <sub>2</sub> CLK <sub>2</sub> / TA1 <sub>OUT</sub> Port P7 <sub>3</sub> CTS <sub>2</sub> / RTS <sub>2</sub> / TA1 <sub>IN</sub>	Port P7 <sub>0</sub> TA0 <sub>OUT</sub> Port P7 <sub>1</sub> TA0 <sub>IN</sub> Port P7 <sub>2</sub> TA1 <sub>OUT</sub> Port P7 <sub>3</sub> TA1 <sub>IN</sub>
Port output style		Port P7 <sub>0</sub> and Port P7 <sub>1</sub> are N-channel open drain Others are CMOS	All Ports are CMOS
Port P9 <sub>3</sub> and P9 <sub>4</sub> pull-up set up condition		All of the following: • Pull-up is selected. • DA output is enabled. • Input port is selected.	Both of the following: • Pull-up is selected. • Input port is selected.
Interrupt sources		 Internal 20 sources External 5 sources Software 4 sources  Add 3 sources -trans., recv. and arbit. for UART2	Internal 17 sources External 5 sources Software 4 sources
DMA request		DMA0 DMA1 1100 UART2 trans. UART2 trans. 1101 UART2 recv. UART2 recv. 1110 A-D A-D 1111 UART1 trans. UART1 recv.	DMA0 DMA1 1100 UART1 trans. UART1 trans. 1101 UART1 recv. UART1 recv. 1110 A-D A-D 1111 prohibited prohibited

Note: M30612M4A/E4 only.

**Memory map Comparison**



### Keep safety first in your circuit designs!

- Mitsubishi Electric Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of non-flammable material or (iii) prevention against any malfunction or mishap.

### Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Mitsubishi semiconductor product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Mitsubishi Electric Corporation or a third party.
- Mitsubishi Electric Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts, programs, algorithms, or circuit application examples contained in these materials.
- All information contained in these materials, including product data, diagrams, charts, programs and algorithms represents information on products at the time of publication of these materials, and are subject to change by Mitsubishi Electric Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for the latest product information before purchasing a product listed herein.  
The information described here may contain technical inaccuracies or typographical errors. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability, or other loss rising from these inaccuracies or errors.  
Please also pay attention to information published by Mitsubishi Electric Corporation by various means, including the Mitsubishi Semiconductor home page (<http://www.mitsubishichips.com>).
- When using any or all of the information contained in these materials, including product data, diagrams, charts, programs, and algorithms, please be sure to evaluate all information as a total system before making a final decision on the applicability of the information and products. Mitsubishi Electric Corporation assumes no responsibility for any damage, liability or other loss resulting from the information contained herein.
- Mitsubishi Electric Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.
- The prior written approval of Mitsubishi Electric Corporation is necessary to reprint or reproduce in whole or in part these materials.
- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination. Any diversion or reexport contrary to the export control laws and regulations of Japan and/or the country of destination is prohibited.
- Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for further details on these materials or the products contained therein.

MITSUBISHI SEMICONDUCTORS  
M16C/61 Group Specification REV.E

---

Apr. First Edition 1999

Edited by  
Committee of editing of Mitsubishi Semiconductor

Published by  
Mitsubishi Electric Corp., Kitaitami Works

---

This book, or parts thereof, may not be reproduced in any form without  
permission of Mitsubishi Electric Corporation.

©1999 MITSUBISHI ELECTRIC CORPORATION