## Description

The M16C/62 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/62 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 3K to 20K bytes
- Shortest instruction execution time ..... 62.5ns (f(XIN)=16MHz, VCC=5V)
  100ns (f(XIN)=10MHz, VCC=3V, with software one-wait) : Mask ROM version
  142.9ns (f(XIN)=7MHz, VCC=3V, with software one-wait) : One-time PROM version
- Supply voltage ..... 4.2 to 5.5V (f(XIN)=16MHz, without software wait) : Mask ROM version
  4.5 to 5.5V (f(XIN)=16MHz, without software wait) : One-time PROM version
  2.7 to 5.5V (f(XIN)=10MHz with software one-wait) : Mask ROM version
  2.7 to 5.5V (f(XIN)=7MHz with software one-wait) : One-time PROM version
- Low power consumption ..... 25.5mW ( f(XIN)=10MHz, with software one-wait, VCC = 3V)
- Interrupts ..... 25 internal and 8 external interrupt sources, 4 software interrupt sources; 7 levels (including key input interrupt)
- Multifunction 16-bit timer ..... 5 output timers + 6 input timers
- Serial I/O ..... 5 channels (3 for UART or clock synchronous, 2 for clock synchronous)
- DMAC ..... 2 channels (trigger: 24 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 1.2M bytes or 4M 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)

Specifications written in this manual are believed to be accurate, but are not guaranteed to be entirely free of error. Specifications in this manual may be changed for functional or performance improvements. Please make sure your manual is the latest edition.

## Applications

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

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

## Pin Configuration

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

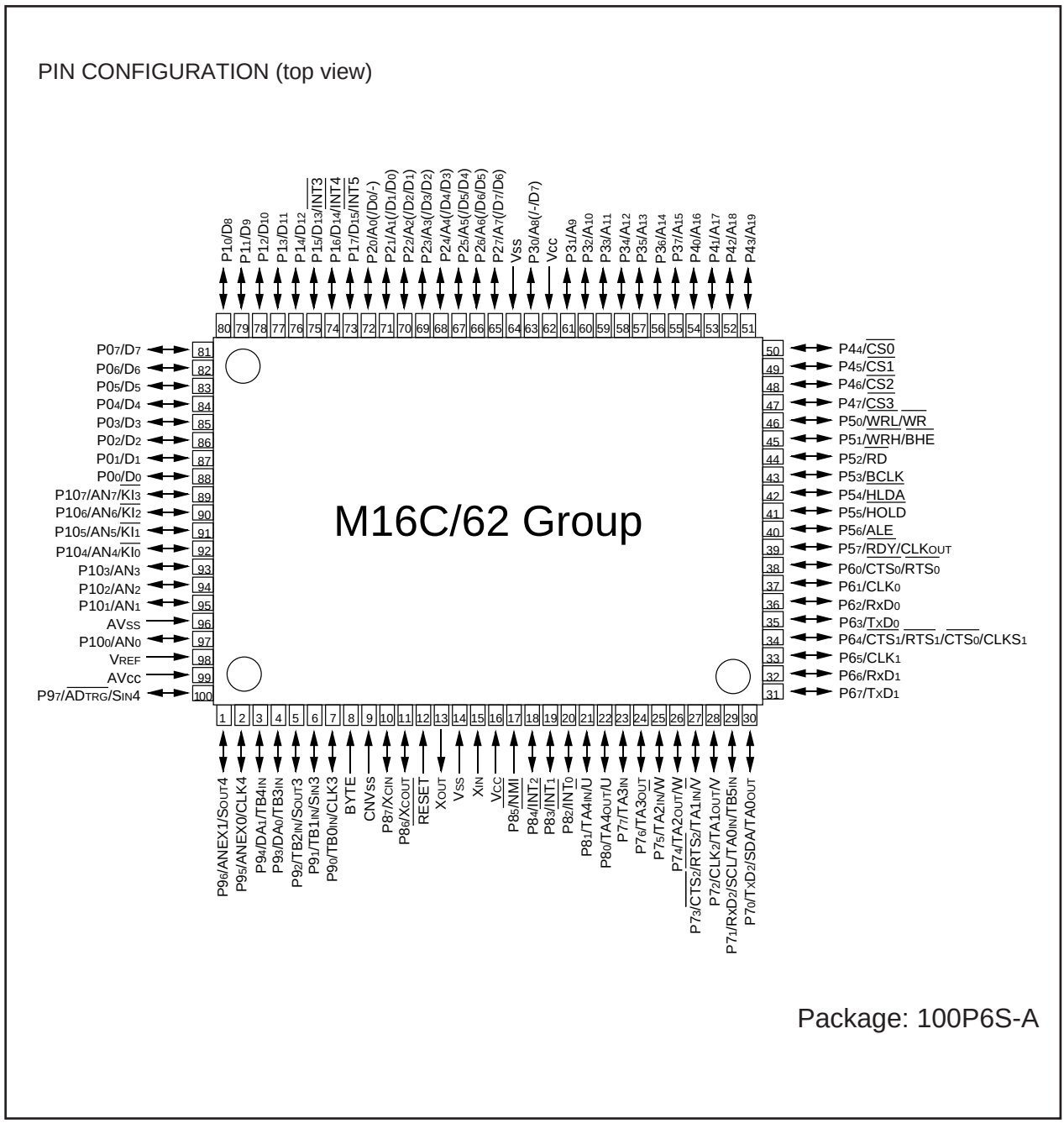



**Figure 1.1.1. Pin configuration (top view)**

## Description

PIN CONFIGURATION (top view)

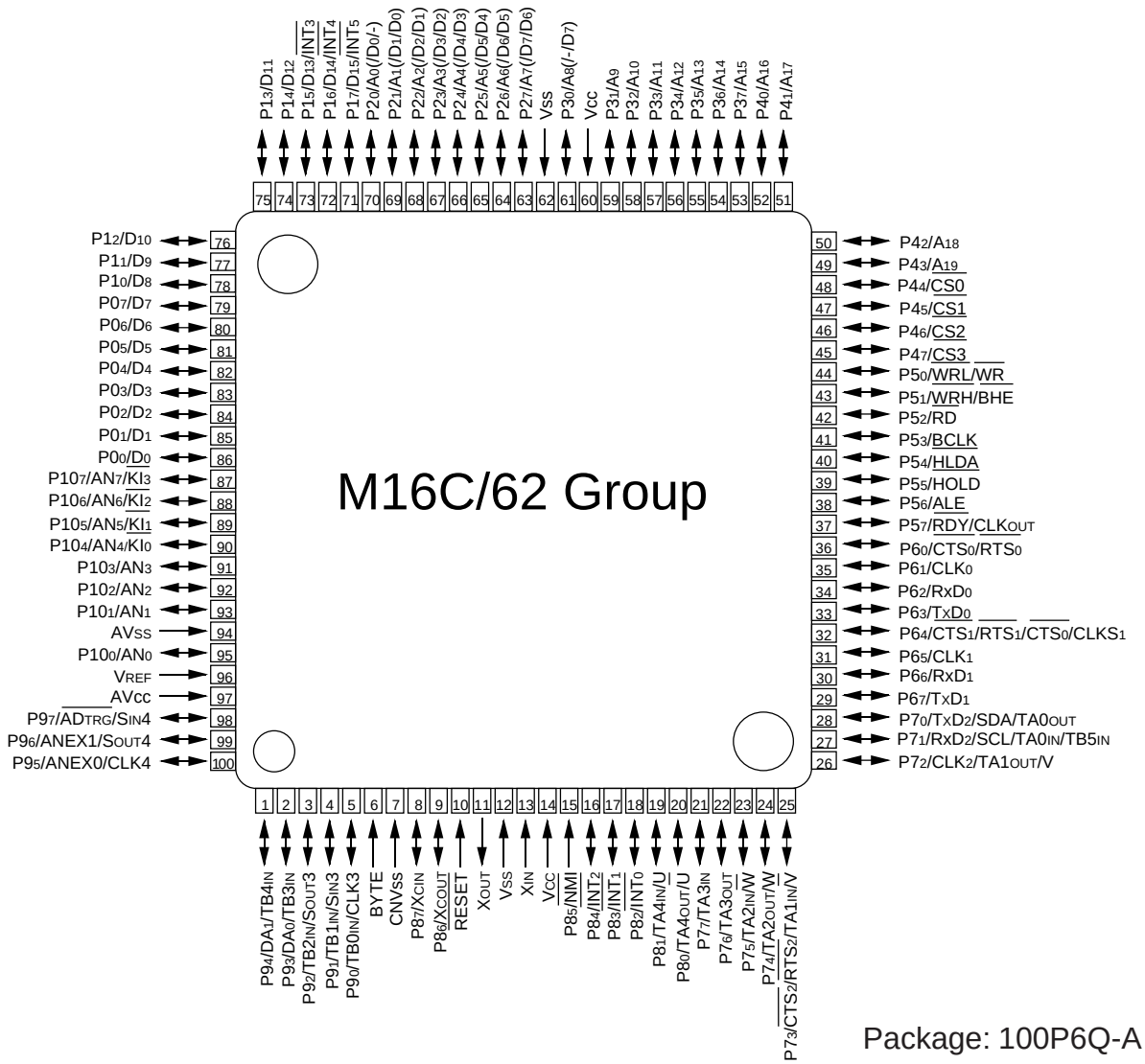
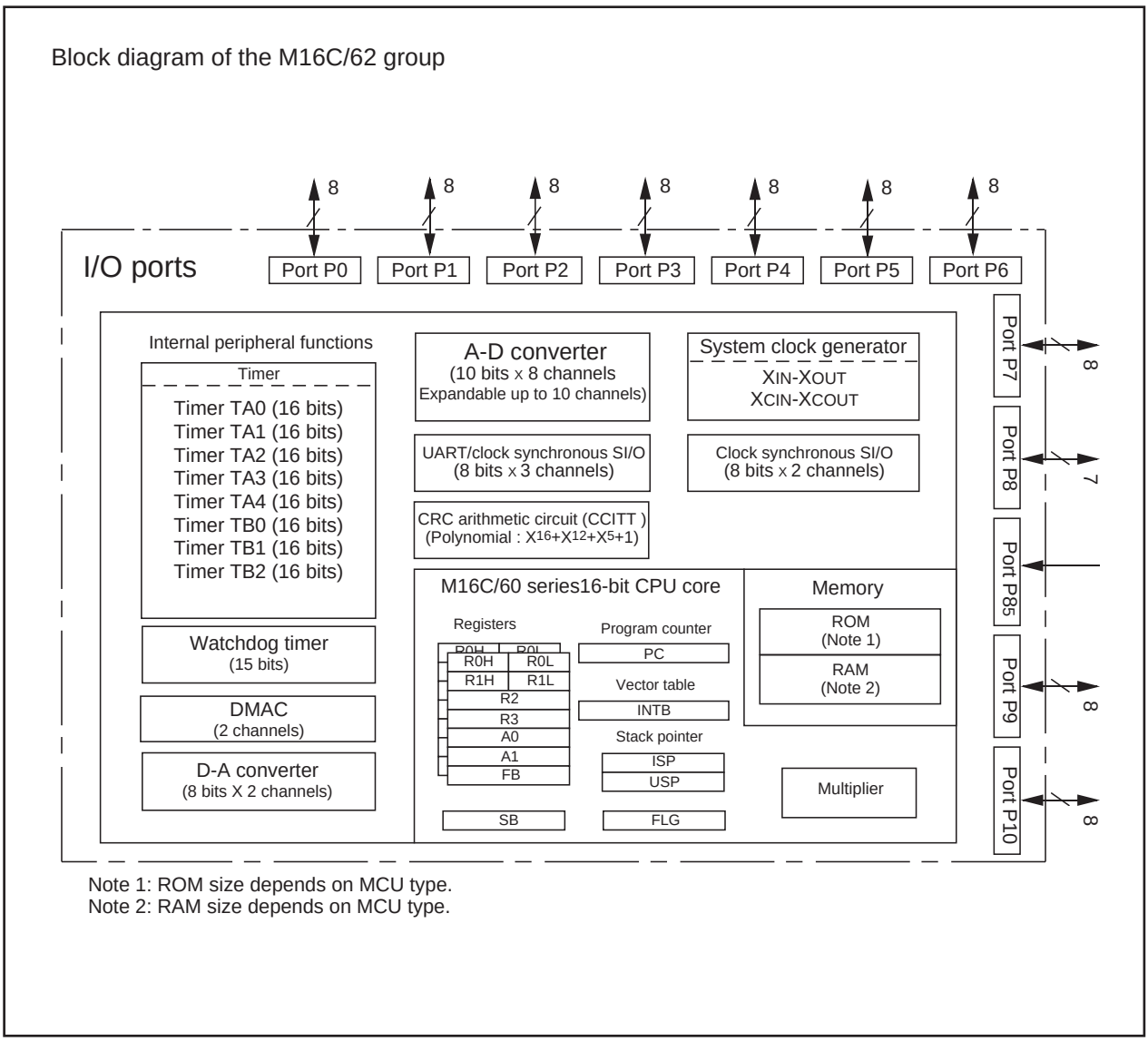



**Figure 1.1.2. Pin configuration (top view)**

## Block Diagram

Figure 1.1.3 is a block diagram of the M16C/62 group.




Note 1: ROM size depends on MCU type.
Note 2: RAM size depends on MCU type.

**Figure 1.1.3. Block diagram of M16C/62 group**

## Performance Outline

Table 1.1.1 is a performance outline of M16C/62 group.

**Table 1.1.1. Performance outline of M16C/62 group**

| Item | | Performance |
|---|---|---|
| Number of basic instructions | | 91 instructions |
| Shortest instruction execution time | | 62.5ns(f(XIN)=16MHz, VCC=5V)<br>100ns (f(XIN)=10MHz, VCC=3V, with software one-wait)<br>: Mask ROM version<br>142.9ns (f(XIN)=7MHz, VCC=3V, with software one-wait)<br>: One-time PROM version |
| Memory capacity | ROM | (See the figure 4. ROM Expansion) |
| | RAM | 3K to 20K 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, TB3, TB4, TB5 | 16 bits x 6 |
| Serial I/O | UART0, UART1, UART2 | (UART or clock synchronous) x 3 |
| | SI/O3, SI/O4 | (Clock synchronous) x 2 |
| A-D converter | | 10 bits x (8 + 2) channels |
| D-A converter | | 8 bits x 2 |
| DMAC | | 2 channels (trigger: 24 sources) |
| CRC calculation circuit | | CRC-CCITT |
| Watchdog timer | | 15 bits x 1 (with prescaler) |
| Interrupt | | 25 internal and 8 external sources, 4 software sources, 7 levels |
| Clock generating circuit | | 2 built-in clock generation circuits<br>(built-in feedback resistor, and external ceramic or quartz oscillator) |
| Supply voltage | | 4.2 to 5.5V (f(XIN)=16MHz, without software wait)<br>: Mask ROM version<br>4.5 to 5.5V (f(XIN)=16MHz, without software wait)<br>: One-time PROM version<br>2.7 to 5.5V (f(XIN)=10MHz with software one-wait)<br>: Mask ROM version<br>2.7 to 5.5V (f(XIN)=7MHz with software one-wait)<br>: One-time PROM version |
| Power consumption | | 25.5mW (f(XIN) = 10MHz, VCC=3V with software one-wait) |
| I/O characteristics | I/O withstand voltage | 5V |
| | Output current | 5mA |
| Memory expansion | | Available (to 1.2M bytes or 4M bytes) |
| Device configuration | | CMOS high performance silicon gate |
| Package | | 100-pin plastic mold QFP |

# Description

Mitsubishi plans to release the following products in the M16C/62 group:

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

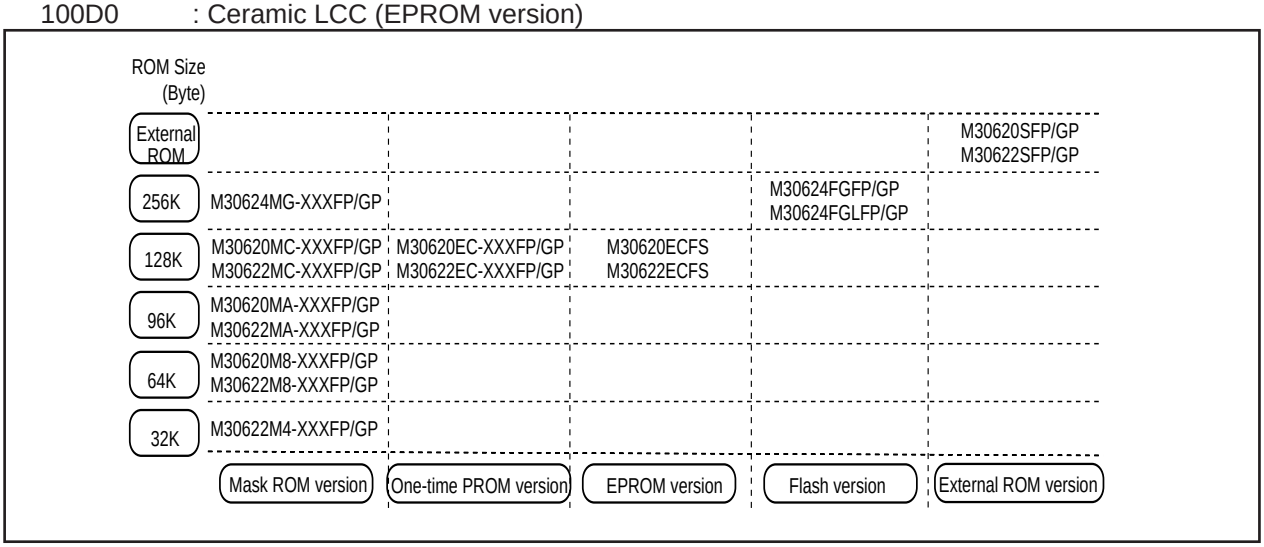



**Figure 1.1.4. ROM expansion**

The M16C/62 group products currently supported are listed in Table 1.1.2.

**Table 1.1.2. M16C/62 group**

Apr. 1998

| Type No | ROM capacity | RAM capacity | Package type | Remarks |
|---|---|---|---|---|
| M30622M4-XXXFP ** | 32K byte | 3K byte | 100P6S-A | mask ROM version |
| M30622M4-XXXGP ** | | | 100P6Q-A | |
| M30620M8-XXXFP ** | 64K byte | 10K byte | 100P6S-A | |
| M30620M8-XXXGP ** | | | 100P6Q-A | |
| M30622M8-XXXFP ** | | 4K byte | 100P6S-A | |
| M30622M8-XXXGP ** | | | 100P6Q-A | |
| M30620MA-XXXFP ** | 96K byte | 10K byte | 100P6S-A | |
| M30620MA-XXXGP ** | | | 100P6Q-A | |
| M30622MA-XXXFP ** | | 5K byte | 100P6S-A | |
| M30622MA-XXXGP ** | | | 100P6Q-A | |
| M30620MC-XXXFP ** | 128K byte | 10K byte | 100P6S-A | |
| M30620MC-XXXGP ** | | | 100P6Q-A | |
| M30622MC-XXXFP ** | | 5K byte | 100P6S-A | |
| M30622MC-XXXGP ** | | | 100P6Q-A | |
| M30624MG-XXXFP ** | 256K byte | 20K byte | 100P6S-A | |
| M30624MG-XXXGP ** | | | 100P6Q-A | |
| M30620EC-XXXFP ** | 128K byte | 10K byte | 100P6S-A | One-time PROM version |
| M30620EC-XXXGP ** | | | 100P6Q-A | |
| M30622EC-XXXFP ** | | 5K byte | 100P6S-A | |
| M30622EC-XXXGP ** | | | 100P6Q-A | |
| M30620ECFS ** | 128K byte | 10K byte | 100D0 | EPROM version (Note) |
| M30622ECFS ** | | 5K byte | 100D0 | |
| M30624FGFP ** | 256K byte | 20K byte | 100P6S-A | Flash 5V version |
| M30624FGGP ** | | | 100P6Q-A | |
| M30624FGLFP ** | 256K byte | 20K byte | 100P6S-A | Flash 3.3V version |
| M30624FGLGP ** | | | 100P6Q-A | |
| M30620SFP ** | ——— | 10K byte | 100P6S-A | External ROM version |
| M30620SGP ** | | | 100P6Q-A | |
| M30622SFP ** | | 3K byte | 100P6S-A | |
| M30622SGP ** | | | 100P6Q-A | |

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

**: Under development

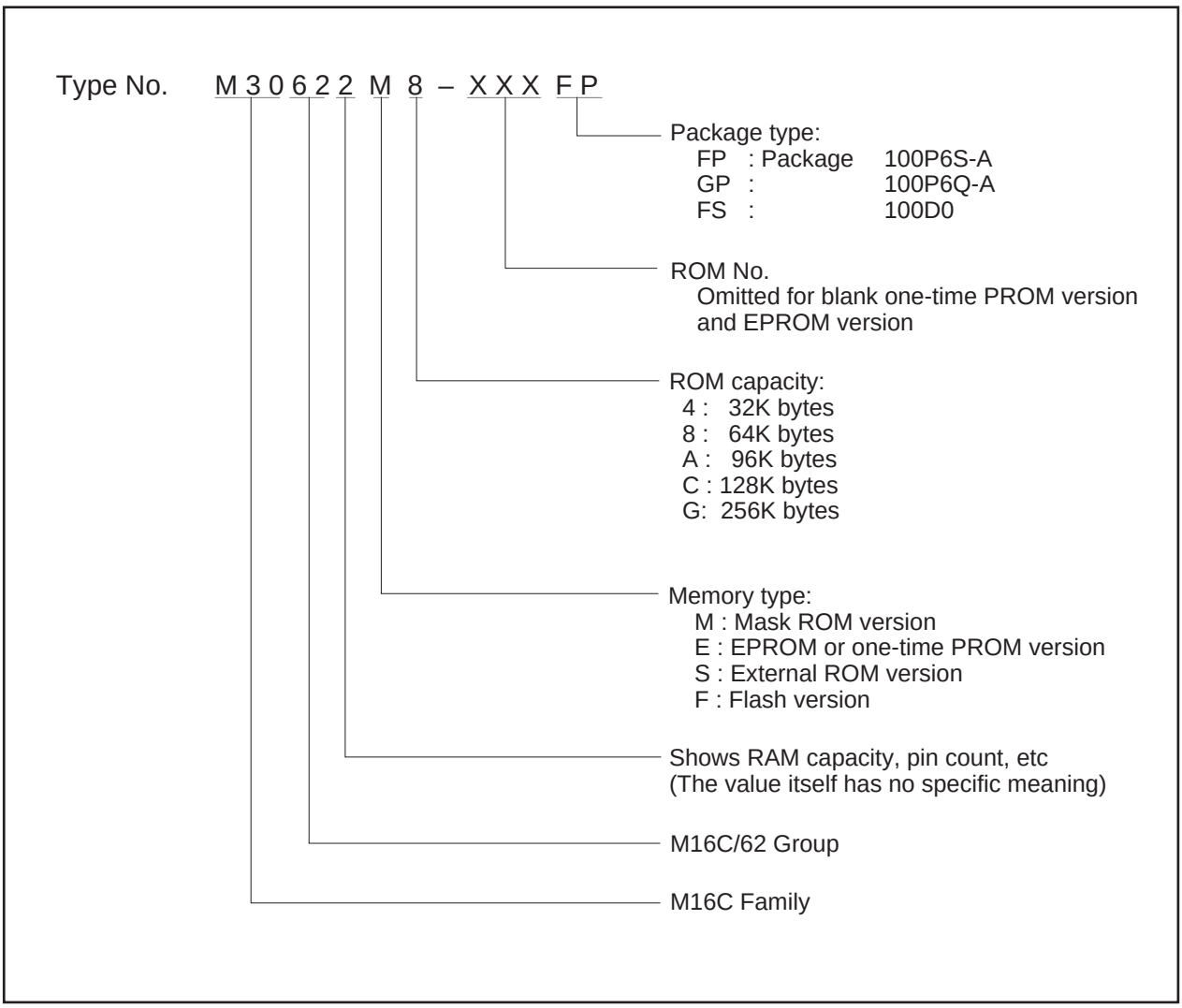



**Figure 1.1.5. Type No., memory size, and package**

### Pin Description

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| $V_{CC}$, $V_{SS}$ | Power supply input | | Supply 2.7 to 5.5 V to the $V_{CC}$ pin. Supply 0 V to the $V_{SS}$ pin. |
| CNV$_{SS}$ | CNV$_{SS}$ | Input | This pin switches between processor modes. Connect it to the $V_{SS}$ pin when operating in single-chip or memory expansion mode. Connect it to the $V_{CC}$ pin when in microprocessor mode. |
| $\overline{\text{RESET}}$ | Reset input | Input | A “L” on this input resets the microcomputer. |
| $X_{IN}$<br>$X_{OUT}$ | Clock input<br>Clock output | Input<br>Output | These pins are provided for the main clock generating circuit.Connect a ceramic resonator or crystal between the $X_{IN}$ and the $X_{OUT}$ pins. To use an externally derived clock, input it to the $X_{IN}$ pin and leave the $X_{OUT}$ 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 $V_{SS}$. |
| AV$_{CC}$ | Analog power supply input | | This pin is a power supply input for the A-D converter. Connect this pin to $V_{CC}$. |
| AV$_{SS}$ | Analog power supply input | | This pin is a power supply input for the A-D converter. Connect this pin to $V_{SS}$. |
| $V_{REF}$ | Reference voltage input | Input | This pin is a reference voltage input for the A-D converter. |
| $P0_0$ to $P0_7$ | 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 set for input, the user can specify in units of four bits via software whether or not they are tied to a pull-up resistor. |
| $D_0$ to $D_7$ | | Input/output | When set as a separate bus, these pins input and output data ($D_0$–$D_7$). |
| $P1_0$ to $P1_7$ | I/O port P1 | Input/output | This is an 8-bit I/O port equivalent to P0. Pins in this port also function as external interrupt pins as selected by software. |
| $D_8$ to $D_{15}$ | | Input/output | When set as a separate bus, these pins input and output data ($D_8$–$D_{15}$). |
| $P2_0$ to $P2_7$ | I/O port P2 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $A_0$ to $A_7$ | | Output | These pins output 8 low-order address bits ($A_0$–$A_7$). |
| $A_0$/$D_0$ to $A_7$/$D_7$ | | Input/output | If the external bus is set as an 8-bit wide multiplexed bus, these pins input and output data ($D_0$–$D_7$) and output 8 low-order address bits ($A_0$–$A_7$) separated in time by multiplexing. |
| $A_0$, $A_1$/$D_0$ to $A_7$/$D_6$ | | Output<br>Input/output | If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data ($D_0$–$D_6$) and output address ($A_1$–$A_7$) separated in time by multiplexing. They also output address ($A_0$). |
| $P3_0$ to $P3_7$ | I/O port P3 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $A_8$ to $A_{15}$ | | Output | These pins output 8 middle-order address bits ($A_8$–$A_{15}$). |
| $A_8$/$D_7$, $A_9$ to $A_{15}$ | | Input/output<br>Output | If the external bus is set as a 16-bit wide multiplexed bus, these pins input and output data ($D_7$) and output address ($A_8$) separated in time by multiplexing. They also output address ($A_9$–$A_{15}$). |
| $P4_0$ to $P4_7$ | I/O port P4 | Input/output | This is an 8-bit I/O port equivalent to P0. |
| $\overline{CS_0}$ to $\overline{CS_3}$, $A_{16}$ to $A_{19}$ | | Output<br>Output | These pins output $\overline{CS_0}$–$\overline{CS_3}$ signals and $A_{16}$–$A_{19}$. $\overline{CS_0}$–$\overline{CS_3}$ are chip select signals used to specify an access space. $A_{16}$–$A_{19}$ are 4 high-order address bits. |

**Pin Description**

| Pin name | Signal name | I/O type | Function |
|---|---|---|---|
| P5$_0$ to P5$_7$ | I/O port P5 | Input/output | This is an 8-bit I/O port equivalent to P0. In single-chip mode, P5$_7$ in this port outputs a divide-by-8 or divide-by-32 clock of $X_{IN}$ or a clock of the same frequency as $X_{CIN}$ as selected by software. |
| $\overline{WRL}$ / $\overline{WR}$,<br>WRH / $\overline{BHE}$,<br>$\overline{RD}$,<br>BCLK,<br>$\overline{HLDA}$,<br>$\overline{HOLD}$,<br><br>ALE,<br>$\overline{RDY}$ | | Output<br>Output<br>Output<br>Output<br>Output<br>Input<br><br>Output<br>Input | Output $\overline{WRL}$, $\overline{WRH}$ ($\overline{WR}$ and $\overline{BHE}$), $\overline{RD}$, BCLK, $\overline{HLDA}$, and ALE signals. $\overline{WRL}$ and $\overline{WRH}$, and $\overline{BHE}$ and $\overline{WR}$ can be switched using software control.<br>■ $\overline{WRL}$, $\overline{WRH}$, and $\overline{RD}$ selected<br>With a 16-bit external data bus, data is written to even addresses when the $\overline{WRL}$ signal is "L" and to the odd addresses when the $\overline{WRH}$ signal is "L". Data is read when $\overline{RD}$ is "L".<br>■ $\overline{WR}$, $\overline{BHE}$, and $\overline{RD}$ selected<br>Data is written when $\overline{WR}$ is "L". Data is read when $\overline{RD}$ is "L". Odd addresses are accessed when $\overline{BHE}$ is "L". Use this mode when using an 8-bit external data bus.<br>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. 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. |
| P6$_0$ to P6$_7$ | I/O port P6 | Input/output | This is an 8-bit I/O port equivalent to P0. Pins in this port also function as UART0 and UART1 I/O pins as selected by software. |
| P7$_0$ to P7$_7$ | I/O port P7 | Input/output | This is an 8-bit I/O port equivalent to P0 (P7$_0$ and P7$_1$ are N channel open-drain output). Pins in this port also function as timer $A_0$–$A_3$, timer B5 or UART2 I/O pins as selected by software. |
| P8$_0$ to P8$_4$,<br>P8$_6$,<br>P8$_7$,<br>P8$_5$ | I/O port P8<br><br><br>I/O port P8$_5$ | Input/output<br>Input/output<br>Input/output<br>Input | P8$_0$ to P8$_4$, P8$_6$, and P8$_7$ are I/O ports with the same functions as P0. Using software, they can be made to function as the I/O pins for timer A4 and the input pins for external interrupts. P8$_6$ and P8$_7$ 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 P8$_6$ ($X_{COUT}$ pin) and P8$_7$ ($X_{CIN}$ pin). P8$_5$ is an input-only port that also functions for $\overline{NMI}$. The $\overline{NMI}$ interrupt is generated when the input at this pin changes from "H" to "L". The $\overline{NMI}$ function cannot be cancelled using software. The pull-up cannot be set for this pin. |
| P9$_0$ to P9$_7$ | I/O port P9 | Input/output | This is an 8-bit I/O port equivalent to P0. Pins in this port also function as SI/O3, 4 I/O pins, Timer B0–B4 input pins, D-A converter output pins, A-D converter extended input pins, or A-D trigger input pins as selected by software. |
| P10$_0$ to P10$_7$ | I/O port P10 | Input/output | This is an 8-bit I/O port equivalent to P0. Pins in this port also function as A-D converter input pins. Furthermore, P10$_4$–P10$_7$ also function as input pins for the key input interrupt function. |

## Operation of Functional Blocks

The M16C/62 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/62 group. The address space extends the 1M bytes from address $00000_{16}$ to $FFFFF_{16}$. From $FFFFF_{16}$ down is ROM. For example, in the M30622MC-XXXFP, there is 128K bytes of internal ROM from $E0000_{16}$ to $FFFFF_{16}$. The vector table for fixed interrupts such as the reset and $\overline{NMI}$ are mapped to $FFFDC_{16}$ to $FFFFF_{16}$. 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.

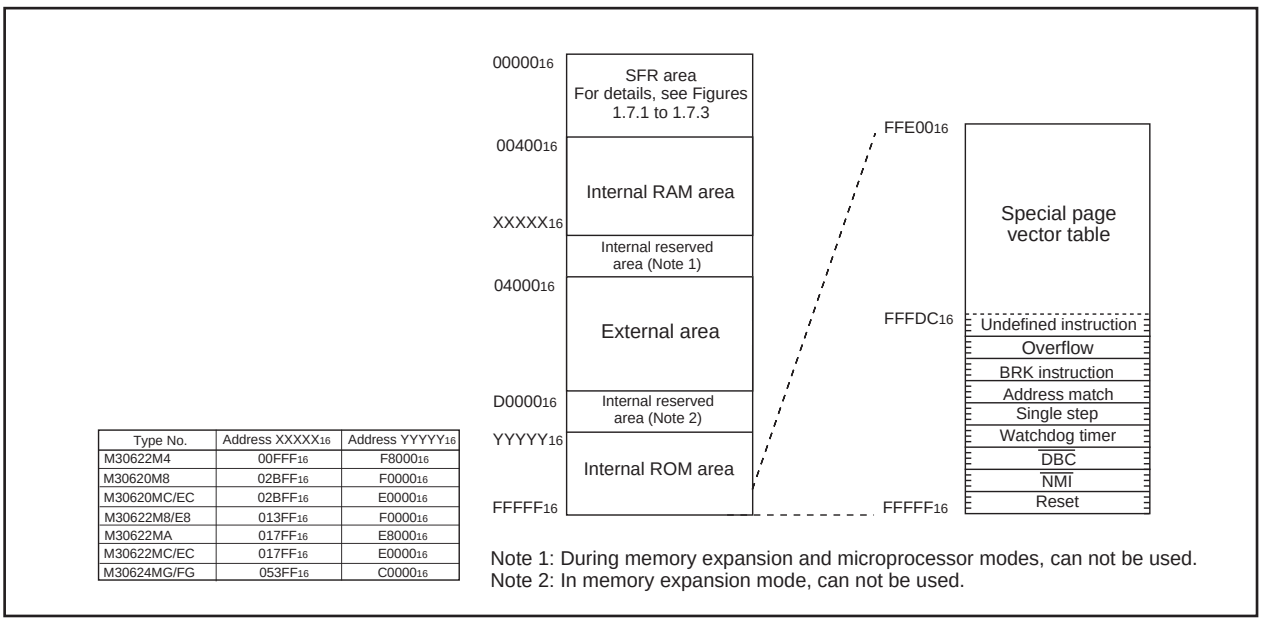
From $00400_{16}$ up is RAM. For example, in the M30622MC-XXXFP, 5K bytes of internal RAM is mapped to the space from $00400_{16}$ to $017FF_{16}$. In addition to storing data, the RAM also stores the stack used when calling subroutines and when interrupts are generated.

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. Figures 1.7.1 to 1.7.3 are location of peripheral unit control registers. 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 M30622MC-XXXFP, the following spaces cannot be used.

- The space between $01800_{16}$ and $03FFF_{16}$ (Memory expansion and microprocessor modes)
- The space between $D0000_{16}$ and $D7FFF_{16}$ (Memory expansion mode)



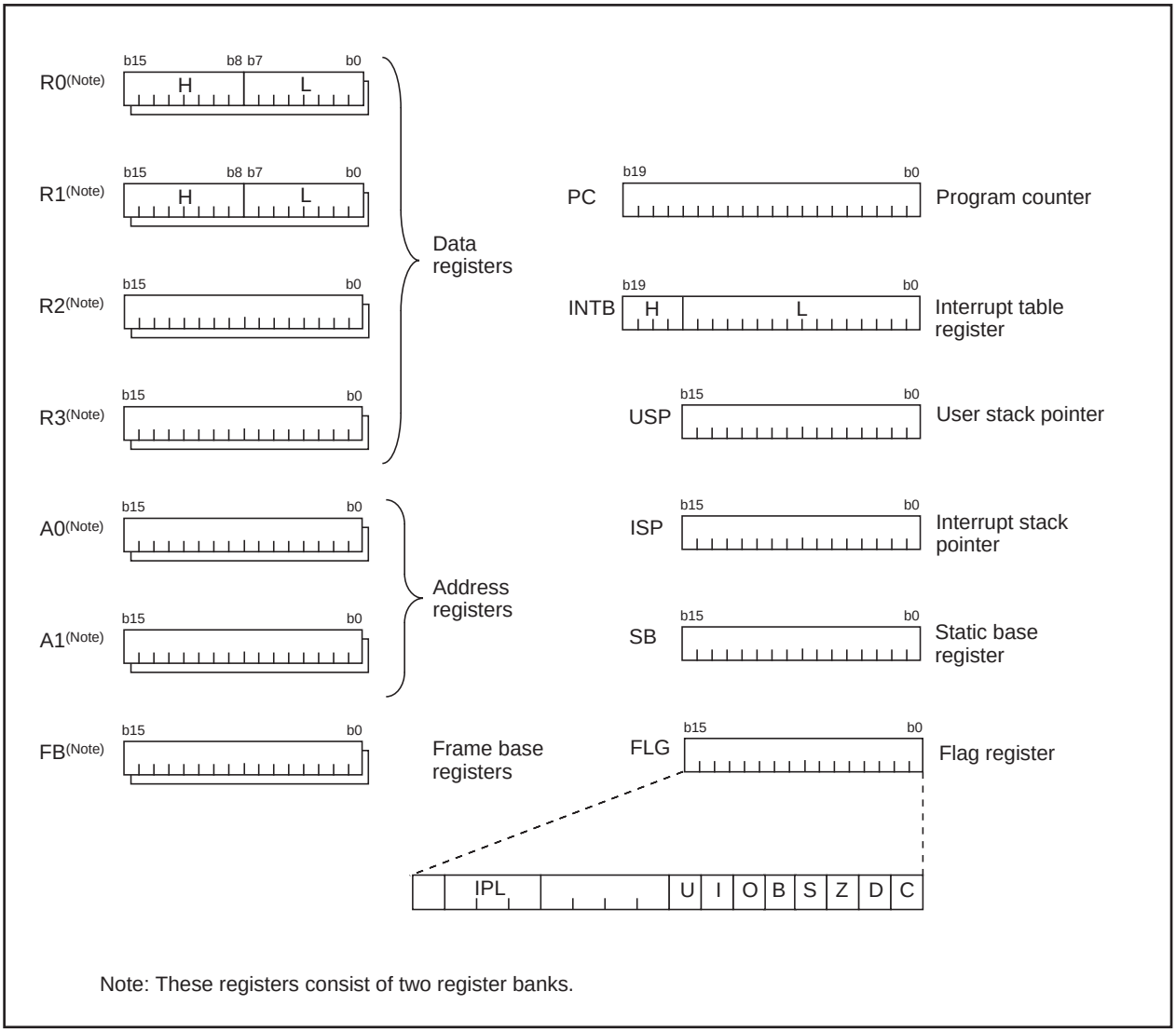

| Type No. | Address XXXXX$_{16}$ | Address YYYYY$_{16}$ |
|---|---|---|
| M30622M4 | 00FFF$_{16}$ | F8000$_{16}$ |
| M30620M8 | 02BFF$_{16}$ | F0000$_{16}$ |
| M30620MC/EC | 02BFF$_{16}$ | E0000$_{16}$ |
| M30622M8/E8 | 013FF$_{16}$ | F0000$_{16}$ |
| M30622MA | 017FF$_{16}$ | E8000$_{16}$ |
| M30622MC/EC | 017FF$_{16}$ | E0000$_{16}$ |
| M30624MG/FG | 053FF$_{16}$ | C0000$_{16}$ |

Note 1: During memory expansion and microprocessor modes, can not be used.
Note 2: In memory expansion mode, can not be used.

**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.




Note: These registers consist of 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.

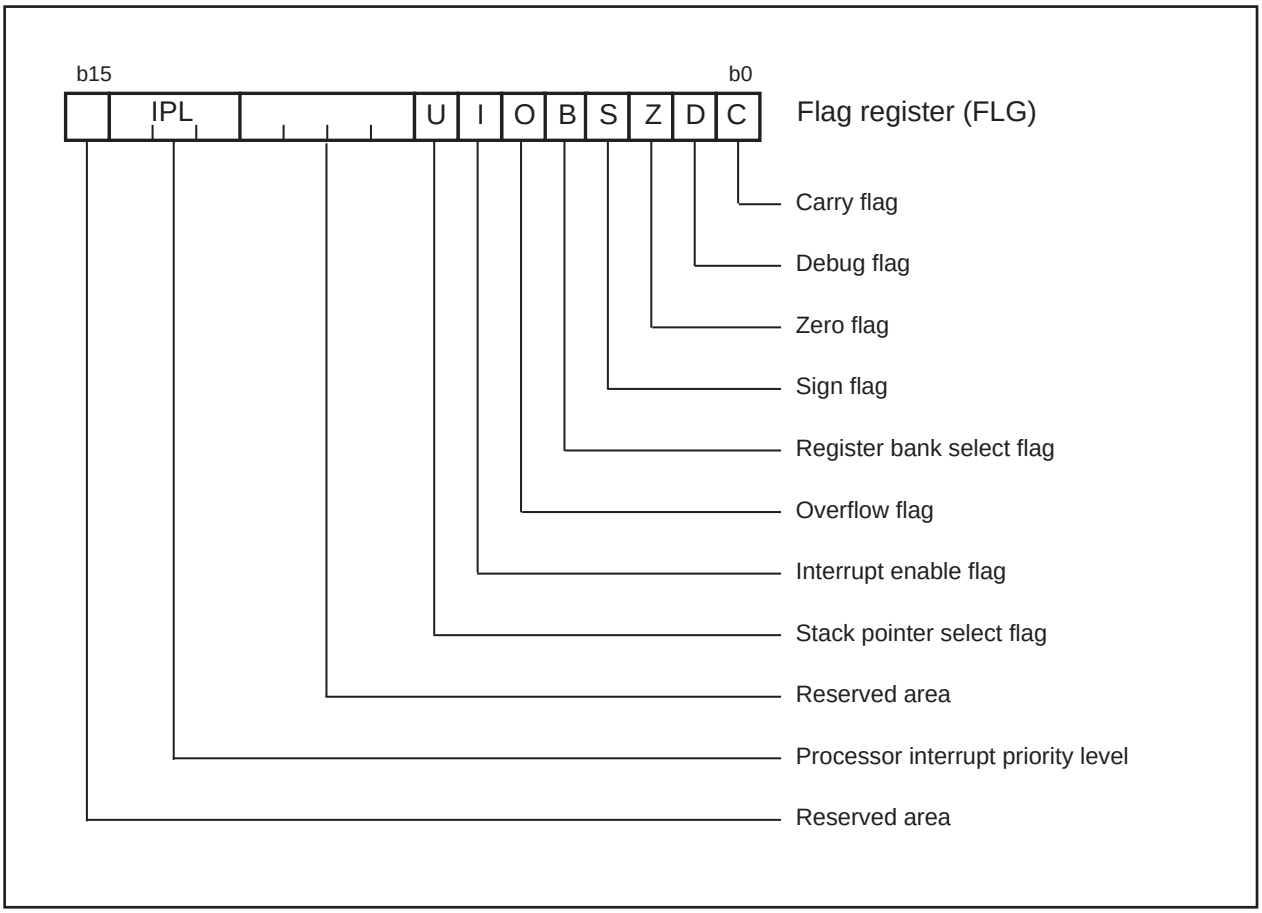



**Figure 1.5.2. Flag register (FLG)**

## Reset

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” (0.2Vcc 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.

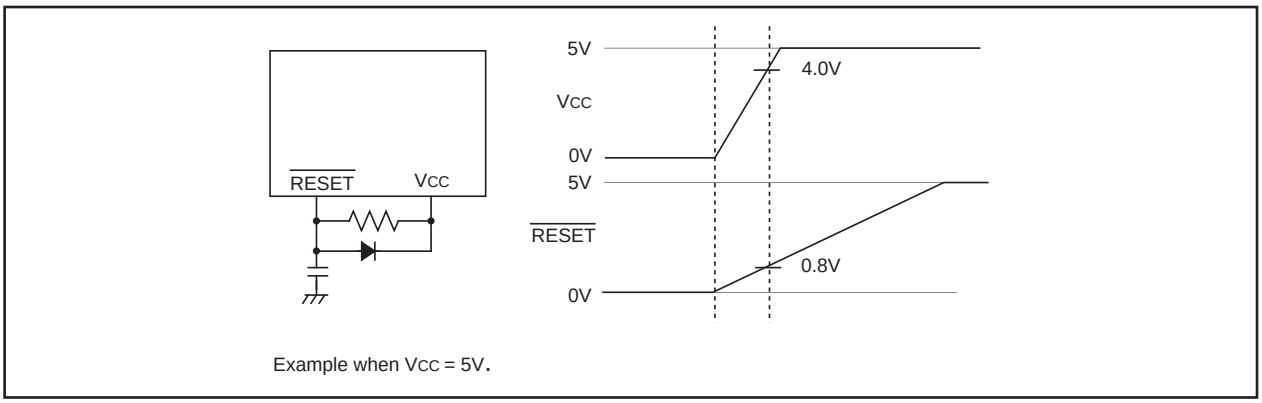



**Figure 1.6.1. Example reset circuit**

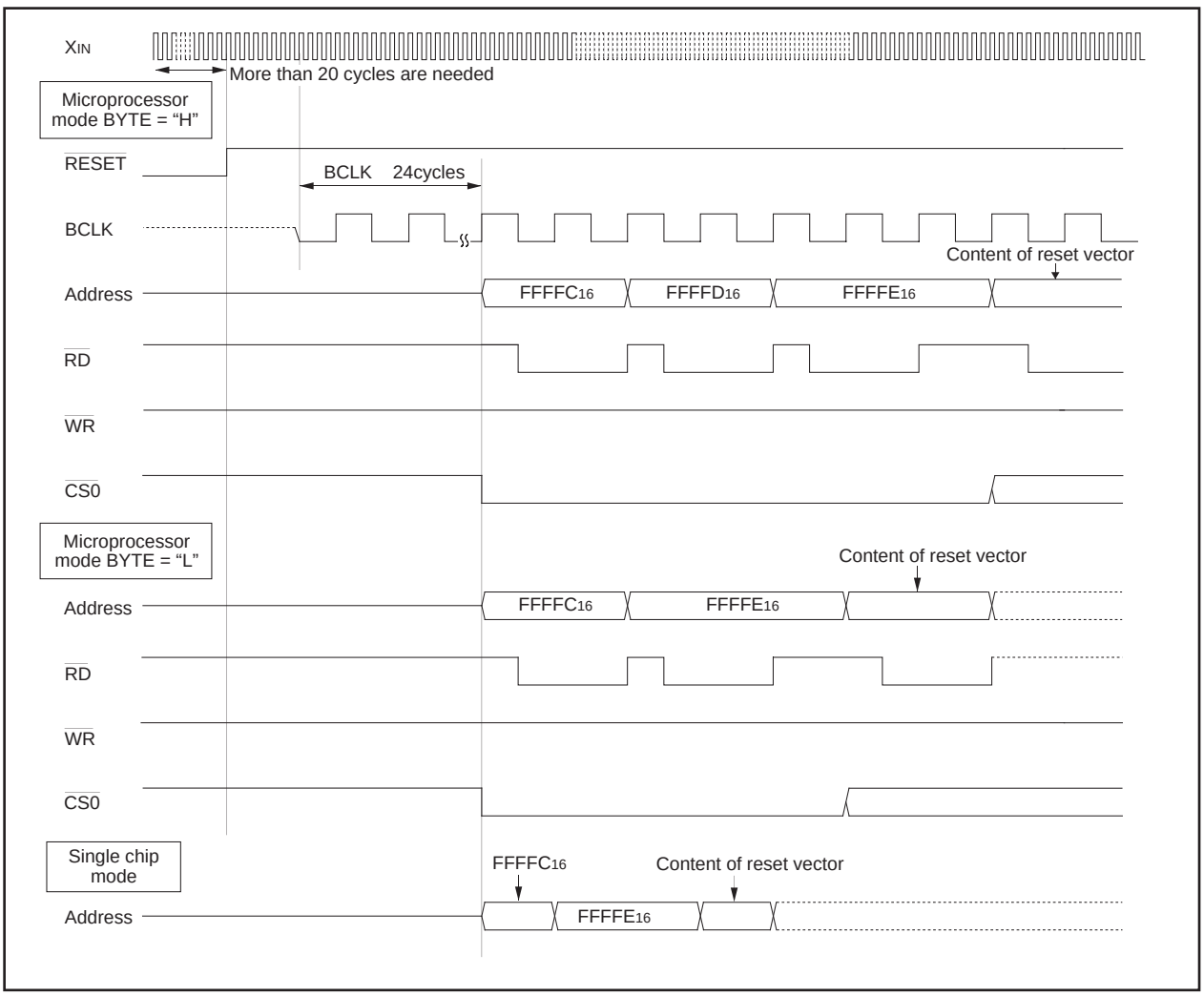



**Figure 1.6.2. Reset sequence**

Table 1.6.1 shows the statuses of the other pins while the $\overline{\text{RESET}}$ pin level is “L”. Figures 1.6.3 and 1.6.4 show 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 | | |
|---|---|---|---|
| | CNV$_{SS}$ = V$_{SS}$ | CNV$_{SS}$ = V$_{CC}$ | |
| | | BYTE = V$_{SS}$ | BYTE = V$_{CC}$ |
| P0 | Input port (floating) | Data input (floating) | Data input (floating) |
| P1 | Input port (floating) | Data input (floating) | Input port (floating) |
| P2, P3, P4$_0$ to P4$_3$ | Input port (floating) | Address output (undefined) | Address output (undefined) |
| P4$_4$ | Input port (floating) | $\overline{\text{CS0}}$ output (“H” level is output) | $\overline{\text{CS0}}$ output (“H” level is output) |
| P4$_5$ to P4$_7$ | Input port (floating) | Input port (floating)<br>(pull-up resistor is on) | Input port (floating)<br>(pull-up resistor is on) |
| P5$_0$ | Input port (floating) | $\overline{\text{WR}}$ output (“H” level is output) | $\overline{\text{WR}}$ output (“H” level is output) |
| P5$_1$ | Input port (floating) | $\overline{\text{BHE}}$ output (undefined) | $\overline{\text{BHE}}$ output (undefined) |
| P5$_2$ | Input port (floating) | $\overline{\text{RD}}$ output (“H” level is output) | $\overline{\text{RD}}$ output (“H” level is output) |
| P5$_3$ | Input port (floating) | BCLK output | BCLK output |
| P5$_4$ | Input port (floating) | $\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin) | $\overline{\text{HLDA}}$ output (The output value depends on the input to the $\overline{\text{HOLD}}$ pin) |
| P5$_5$ | Input port (floating) | $\overline{\text{HOLD}}$ input (floating) | $\overline{\text{HOLD}}$ input (floating) |
| P5$_6$ | Input port (floating) | ALE output (“L” level is output) | ALE output (“L” level is output) |
| P5$_7$ | Input port (floating) | $\overline{\text{RDY}}$ input (floating) | $\overline{\text{RDY}}$ input (floating) |
| P6, P7, P8$_0$ to P8$_4$, P8$_6$, P8$_7$, P9, P10 | Input port (floating) | Input port (floating) | Input port (floating) |

| | | | |
|---|---|---|---|
| (1) Processor mode register 0 (Note1) | (0004$_{16}$) 00$_{16}$ | (29) UART1 transmit interrupt control register | (0053$_{16}$) ? 0 0 0 |
| (2) Processor mode register 1 | (0005$_{16}$) 0 0 0 0 0 x x 0 | (30) UART1 receive interrupt control register | (0054$_{16}$) ? 0 0 0 |
| (3) System clock control register 0 | (0006$_{16}$) 0 1 0 0 1 0 0 0 | (31) Timer A0 interrupt control register | (0055$_{16}$) ? 0 0 0 |
| (4) System clock control register 1 | (0007$_{16}$) 0 0 1 0 0 0 0 0 | (32) Timer A1 interrupt control register | (0056$_{16}$) ? 0 0 0 |
| (5) Chip select control register | (0008$_{16}$) 0 0 0 0 0 0 0 1 | (33) Timer A2 interrupt control register | (0057$_{16}$) ? 0 0 0 |
| (6) Address match interrupt enable register | (0009$_{16}$) x x x x x x 0 0 | (34) Timer A3 interrupt control register | (0058$_{16}$) ? 0 0 0 |
| (7) Protect register | (000A$_{16}$) x x x x x 0 0 0 | (35) Timer A4 interrupt control register | (0059$_{16}$) ? 0 0 0 |
| (8) Data bank register | (000B$_{16}$) 00$_{16}$ | (36) Timer B0 interrupt control register | (005A$_{16}$) ? 0 0 0 |
| (9) Watchdog timer control register | (000F$_{16}$) 0 0 0 ? ? ? ? ? | (37) Timer B1 interrupt control register | (005B$_{16}$) ? 0 0 0 |
| (10) Address match interrupt register 0 | (0010$_{16}$) 00$_{16}$ | (38) Timer B2 interrupt control register | (005C$_{16}$) ? 0 0 0 |
| | (0011$_{16}$) 00$_{16}$ | (39) INT0 interrupt control register | (005D$_{16}$) x x 0 0 ? 0 0 0 |
| | (0012$_{16}$) x x x x 0 0 0 0 | (40) INT1 interrupt control register | (005E$_{16}$) x x 0 0 ? 0 0 0 |
| (11) Address match interrupt register 1 | (0014$_{16}$) 00$_{16}$ | (41) INT2 interrupt control register | (005F$_{16}$) x x 0 0 ? 0 0 0 |
| | (0015$_{16}$) 00$_{16}$ | (42) Timer B3,4,5 count start flag | (0340$_{16}$) 0 0 0 x x x x x |
| | (0016$_{16}$) x x x x 0 0 0 0 | (43) Three-phase PWM control register 0 | (0348$_{16}$) 00$_{16}$ |
| (12) DMA0 control register | (002C$_{16}$) 0 0 0 0 0 ? 0 0 | (44) Three-phase PWM control register 1 | (0349$_{16}$) 00$_{16}$ |
| (13) DMA1 control register | (003C$_{16}$) 0 0 0 0 0 ? 0 0 | (45) Three-phase output buffer register 0 | (034A$_{16}$) 00$_{16}$ |
| (14) INT3 interrupt control register | (0044$_{16}$) x x 0 0 ? 0 0 0 | (46) Three-phase output buffer register 1 | (034B$_{16}$) 00$_{16}$ |
| (15) Timer B5 interrupt control register | (0045$_{16}$) x x x x ? 0 0 0 | (47) Timer B3 mode register | (035B$_{16}$) 0 0 ? x 0 0 0 0 |
| (16) Timer B4 interrupt control register | (0046$_{16}$) x x x x ? 0 0 0 | (48) Timer B4 mode register | (035C$_{16}$) 0 0 ? x 0 0 0 0 |
| (17) Timer B3 interrupt control register | (0047$_{16}$) x x x x ? 0 0 0 | (49) Timer B5 mode register | (035D$_{16}$) 0 0 ? x 0 0 0 0 |
| (18) SI/O4 interrupt control register | (0048$_{16}$) x x 0 0 ? 0 0 0 | (50) Interrupt cause select register | (035F$_{16}$) 00$_{16}$ |
| (19) SI/O3 interrupt control register | (0049$_{16}$) x x 0 0 ? 0 0 0 | (51) SI/O3 control register | (0362$_{16}$) 40$_{16}$ |
| (20) Bus collision detection interrupt control register | (004A$_{16}$) x x x x ? 0 0 0 | (52) SI/O4 control register | (0366$_{16}$) 40$_{16}$ |
| (21) DMA0 interrupt control register | (004B$_{16}$) x x x x ? 0 0 0 | (53) UART2 special mode register 2 (Note2) | (0376$_{16}$) 00$_{16}$ |
| (22) DMA1 interrupt control register | (004C$_{16}$) x x x x ? 0 0 0 | (54) UART2 special mode register | (0377$_{16}$) 00$_{16}$ |
| (23) Key input interrupt control register | (004D$_{16}$) x x x x ? 0 0 0 | (55) UART2 transmit/receive mode register | (0378$_{16}$) 00$_{16}$ |
| (24) A-D conversion interrupt control register | (004E$_{16}$) x x x x ? 0 0 0 | (56) UART2 transmit/receive control register 0 | (037C$_{16}$) 0 0 0 0 1 0 0 0 |
| (25) UART2 transmit interrupt control register | (004F$_{16}$) x x x x ? 0 0 0 | (57) UART2 transmit/receive control register 1 | (037D$_{16}$) 0 0 0 0 0 0 1 0 |
| (26) UART2 receive interrupt control register | (0050$_{16}$) x x x x ? 0 0 0 | | |
| (27) UART0 transmit interrupt control register | (0051$_{16}$) x x x x ? 0 0 0 | | |
| (28) UART0 receive interrupt control register | (0052$_{16}$) x x x x ? 0 0 0 | | |

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.

Note1: When the $V_{CC}$ level is applied to the $CNV_{SS}$ pin, it is 03$_{16}$ at a reset.
Note2: This register is not exist in M30622EC.

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

| | | | |
|---|---|---|---|
| (58) Count start flag | (0380$_{16}$)··· 00$_{16}$ | (83) D-A control register | (03DC$_{16}$)··· 00$_{16}$ |
| (59) Clock prescaler reset flag | (0381$_{16}$)··· 0 x x x x x x x | (84) Port P0 direction register | (03E2$_{16}$)··· 00$_{16}$ |
| (60) One-shot start flag | (0382$_{16}$)··· 0 0 x 0 0 0 0 0 | (85) Port P1 direction register | (03E3$_{16}$)··· 00$_{16}$ |
| (61) Trigger select flag | (0383$_{16}$)··· 00$_{16}$ | (86) Port P2 direction register | (03E6$_{16}$)··· 00$_{16}$ |
| (62) Up-down flag | (0384$_{16}$)··· 00$_{16}$ | (87) Port P3 direction register | (03E7$_{16}$)··· 00$_{16}$ |
| (63) Timer A0 mode register | (0396$_{16}$)··· 00$_{16}$ | (88) Port P4 direction register | (03EA$_{16}$)··· 00$_{16}$ |
| (64) Timer A1 mode register | (0397$_{16}$)··· 00$_{16}$ | (89) Port P5 direction register | (03EB$_{16}$)··· 00$_{16}$ |
| (65) Timer A2 mode register | (0398$_{16}$)··· 00$_{16}$ | (90) Port P6 direction register | (03EE$_{16}$)··· 00$_{16}$ |
| (66) Timer A3 mode register | (0399$_{16}$)··· 00$_{16}$ | (91) Port P7 direction register | (03EF$_{16}$)··· 00$_{16}$ |
| (67) Timer A4 mode register | (039A$_{16}$)··· 00$_{16}$ | (92) Port P8 direction register | (03F2$_{16}$)··· 0 0 x 0 0 0 0 0 |
| (68) Timer B0 mode register | (039B$_{16}$)··· 0 0 ? x 0 0 0 0 | (93) Port P9 direction register | (03F3$_{16}$)··· 00$_{16}$ |
| (69) Timer B1 mode register | (039C$_{16}$)··· 0 0 ? x 0 0 0 0 | (94) Port P10 direction register | (03F6$_{16}$)··· 00$_{16}$ |
| (70) Timer B2 mode register | (039D$_{16}$)··· 0 0 ? x 0 0 0 0 | (95) Pull-up control register 0 | (03FC$_{16}$)··· 00$_{16}$ |
| (71) UART0 transmit/receive mode register | (03A0$_{16}$)··· 00$_{16}$ | (96) Pull-up control register 1(Note) | (03FD$_{16}$)··· 00$_{16}$ |
| (72) UART0 transmit/receive control register 0 | (03A4$_{16}$)··· 0 0 0 0 1 0 0 0 | (97) Pull-up control register 2 | (03FE$_{16}$)··· 00$_{16}$ |
| (73) UART0 transmit/receive control register 1 | (03A5$_{16}$)··· 0 0 0 0 0 0 1 0 | (98) Port control register | (03FF$_{16}$)··· 00$_{16}$ |
| (74) UART1 transmit/receive mode register | (03A8$_{16}$)··· 00$_{16}$ | (99) Data registers (R0/R1/R2/R3) | 0000$_{16}$ |
| (75) UART1 transmit/receive control register 0 | (03AC$_{16}$)··· 0 0 0 0 1 0 0 0 | (100) Address registers (A0/A1) | 0000$_{16}$ |
| (76) UART1 transmit/receive control register 1 | (03AD$_{16}$)··· 0 0 0 0 0 0 1 0 | (101) Frame base register (FB) | 0000$_{16}$ |
| (77) UART transmit/receive control register 2 | (03B0$_{16}$)··· x 0 0 0 0 0 0 0 | (102) Interrupt table register (INTB) | 00000$_{16}$ |
| (78) DMA0 cause select register | (03B8$_{16}$)··· 00$_{16}$ | (103) User stack pointer (USP) | 0000$_{16}$ |
| (79) DMA1 cause select register | (03BA$_{16}$)··· 00$_{16}$ | (104) Interrupt stack pointer (ISP) | 0000$_{16}$ |
| (80) A-D control register 2 | (03D4$_{16}$)··· 0 0 0 0 x x x 0 | (105) Static base register (SB) | 0000$_{16}$ |
| (81) A-D control register 0 | (03D6$_{16}$)··· 0 0 0 0 0 ? ? ? | (106) Flag register (FLG) | 0000$_{16}$ |
| (82) A-D control register 1 | (03D7$_{16}$)··· 00$_{16}$ | | |

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 $V_{CC}$ level is applied to the $CNV_{SS}$ pin, it is 02$_{16}$ at a reset.

**Figure 1.6.4. Device's internal status after a reset is cleared**

| | |
|---|---|
| 0000₁₆ | |
| 0001₁₆ | |
| 0002₁₆ | |
| 0003₁₆ | |
| 0004₁₆ | Processor mode register 0 (PM0) |
| 0005₁₆ | Processor mode register 1(PM1) |
| 0006₁₆ | System clock control register 0 (CM0) |
| 0007₁₆ | System clock control register 1 (CM1) |
| 0008₁₆ | Chip select control register (CSR) |
| 0009₁₆ | Address match interrupt enable register (AIER) |
| 000A₁₆ | Protect register (PRCR) |
| 000B₁₆ | Data bank register (DBR) |
| 000C₁₆ | |
| 000D₁₆ | |
| 000E₁₆ | Watchdog timer start register (WDTS) |
| 000F₁₆ | Watchdog timer control register (WDC) |
| 0010₁₆ | Address match interrupt register 0 (RMAD0) |
| 0011₁₆ | |
| 0012₁₆ | |
| 0013₁₆ | |
| 0014₁₆ | Address match interrupt register 1 (RMAD1) |
| 0015₁₆ | |
| 0016₁₆ | |
| 0017₁₆ | |
| 0018₁₆ | |
| 0019₁₆ | |
| 001A₁₆ | |
| 001B₁₆ | |
| 001C₁₆ | |
| 001D₁₆ | |
| 001E₁₆ | |
| 001F₁₆ | |
| 0020₁₆ | DMA0 source pointer (SAR0) |
| 0021₁₆ | |
| 0022₁₆ | |
| 0023₁₆ | |
| 0024₁₆ | DMA0 destination pointer (DAR0) |
| 0025₁₆ | |
| 0026₁₆ | |
| 0027₁₆ | |
| 0028₁₆ | DMA0 transfer counter (TCR0) |
| 0029₁₆ | |
| 002A₁₆ | |
| 002B₁₆ | |
| 002C₁₆ | DMA0 control register (DM0CON) |
| 002D₁₆ | |
| 002E₁₆ | |
| 002F₁₆ | |
| 0030₁₆ | DMA1 source pointer (SAR1) |
| 0031₁₆ | |
| 0032₁₆ | |
| 0033₁₆ | |
| 0034₁₆ | DMA1 destination pointer (DAR1) |
| 0035₁₆ | |
| 0036₁₆ | |
| 0037₁₆ | |
| 0038₁₆ | DMA1 transfer counter (TCR1) |
| 0039₁₆ | |
| 003A₁₆ | |
| 003B₁₆ | |
| 003C₁₆ | DMA1 control register (DM1CON) |
| 003D₁₆ | |
| 003E₁₆ | |
| 003F₁₆ | |

| | |
|---|---|
| 0040₁₆ | |
| 0041₁₆ | |
| 0042₁₆ | |
| 0043₁₆ | |
| 0044₁₆ | INT3 interrupt control register (INT3IC) |
| 0045₁₆ | Timer B5 interrupt control register (TB5IC) |
| 0046₁₆ | Timer B4 interrupt control register (TB4IC) |
| 0047₁₆ | Timer B3 interrupt control register (TB3IC) |
| 0048₁₆ | SI/O4 interrupt control register (S4IC)<br>INT5 interrupt control register (INT5IC) |
| 0049₁₆ | SI/O3 interrupt control register (S3IC)<br>INT4 interrupt control register (INT4IC) |
| 004A₁₆ | Bus collision detection interrupt control register (BCNIC) |
| 004B₁₆ | DMA0 interrupt control register (DM0IC) |
| 004C₁₆ | DMA1 interrupt control register (DM1IC) |
| 004D₁₆ | Key input interrupt control register (KUPIC) |
| 004E₁₆ | A-D conversion interrupt control register (ADIC) |
| 004F₁₆ | UART2 transmit interrupt control register (S2TIC) |
| 0050₁₆ | UART2 receive interrupt control register (S2RIC) |
| 0051₁₆ | UART0 transmit interrupt control register (S0TIC) |
| 0052₁₆ | UART0 receive interrupt control register (S0RIC) |
| 0053₁₆ | UART1 transmit interrupt control register (S1TIC) |
| 0054₁₆ | UART1 receive interrupt control register (S1RIC) |
| 0055₁₆ | Timer A0 interrupt control register (TA0IC) |
| 0056₁₆ | Timer A1 interrupt control register (TA1IC) |
| 0057₁₆ | Timer A2 interrupt control register (TA2IC) |
| 0058₁₆ | Timer A3 interrupt control register (TA3IC) |
| 0059₁₆ | Timer A4 interrupt control register (TA4IC) |
| 005A₁₆ | Timer B0 interrupt control register (TB0IC) |
| 005B₁₆ | Timer B1 interrupt control register (TB1IC) |
| 005C₁₆ | Timer B2 interrupt control register (TB2IC) |
| 005D₁₆ | INT0 interrupt control register (INT0IC) |
| 005E₁₆ | INT1 interrupt control register (INT1IC) |
| 005F₁₆ | INT2 interrupt control register (INT2IC) |
| 0060₁₆ | |
| 0061₁₆ | |
| 0062₁₆ | |
| 0063₁₆ | |
| 0064₁₆ | |
| 0065₁₆ | |
| ≈ | ≈ |
| 032A₁₆ | |
| 032B₁₆ | |
| 032C₁₆ | |
| 032D₁₆ | |
| 032E₁₆ | |
| 032F₁₆ | |
| 0330₁₆ | |
| 0331₁₆ | |
| 0332₁₆ | |
| 0333₁₆ | |
| 0334₁₆ | |
| 0335₁₆ | |
| 0336₁₆ | |
| 0337₁₆ | |
| 0338₁₆ | |
| 0339₁₆ | |
| 033A₁₆ | |
| 033B₁₆ | |
| 033C₁₆ | |
| 033D₁₆ | |
| 033E₁₆ | |
| 033F₁₆ | |

**Figure 1.7.1. Location of peripheral unit control registers (1)**

| | |
|---|---|
| 0340₁₆ | Timer B3, 4, 5 count start flag (TBSR) |
| 0341₁₆ | |
| 0342₁₆ | Timer A1-1 register (TA11) |
| 0343₁₆ | |
| 0344₁₆ | Timer A2-1 register (TA21) |
| 0345₁₆ | |
| 0346₁₆ | Timer A4-1 register (TA41) |
| 0347₁₆ | |
| 0348₁₆ | Three-phase PWM control register 0(INVC0) |
| 0349₁₆ | Three-phase PWM control register 1(INVC1) |
| 034A₁₆ | Thrree-phase output buffer register 0(IDB0) |
| 034B₁₆ | Thrree-phase output buffer register 1(IDB1) |
| 034C₁₆ | Dead time timer(DTT) |
| 034D₁₆ | Timer B2 interrupt occurrence frequency set counter(ICTB2) |
| 034E₁₆ | |
| 034F₁₆ | |
| 0350₁₆ | Timer B3 register (TB3) |
| 0351₁₆ | |
| 0352₁₆ | Timer B4 register (TB4) |
| 0353₁₆ | |
| 0354₁₆ | Timer B5 register (TB5) |
| 0355₁₆ | |
| 0356₁₆ | |
| 0357₁₆ | |
| 0358₁₆ | |
| 0359₁₆ | |
| 035A₁₆ | |
| 035B₁₆ | Timer B3 mode register (TB3MR) |
| 035C₁₆ | Timer B4 mode register (TB4MR) |
| 035D₁₆ | Timer B5 mode register (TB5MR) |
| 035E₁₆ | |
| 035F₁₆ | Interrupt cause select register (IFSR) |
| 0360₁₆ | SI/O3 transmit/receive register (S3TRR) |
| 0361₁₆ | |
| 0362₁₆ | SI/O3 control register (S3C) |
| 0363₁₆ | SI/O3 bit rate generator (S3BRG) |
| 0364₁₆ | SI/O4 transmit/receive register (S4TRR) |
| 0365₁₆ | |
| 0366₁₆ | SI/O4 control register (S4C) |
| 0367₁₆ | SI/O4 bit rate generator (S4BRG) |
| 0368₁₆ | |
| 0369₁₆ | |
| 036A₁₆ | |
| 036B₁₆ | |
| 036C₁₆ | |
| 036D₁₆ | |
| 036E₁₆ | |
| 036F₁₆ | |
| 0370₁₆ | |
| 0371₁₆ | |
| 0372₁₆ | |
| 0373₁₆ | |
| 0374₁₆ | |
| 0375₁₆ | |
| 0376₁₆ | UART2 special mode register 2(U2SMR2) (Note) |
| 0377₁₆ | UART2 special mode register (U2SMR) |
| 0378₁₆ | UART2 transmit/receive mode register (U2MR) |
| 0379₁₆ | UART2 bit rate generator (U2BRG) |
| 037A₁₆ | UART2 transmit buffer register (U2TB) |
| 037B₁₆ | |
| 037C₁₆ | UART2 transmit/receive control register 0 (U2C0) |
| 037D₁₆ | UART2 transmit/receive control register 1 (U2C1) |
| 037E₁₆ | UART2 receive buffer register (U2RB) |
| 037F₁₆ | |

| | |
|---|---|
| 0380₁₆ | Count start flag (TABSR) |
| 0381₁₆ | Clock prescaler reset flag (CPSRF) |
| 0382₁₆ | One-shot start flag (ONSF) |
| 0383₁₆ | Trigger select register (TRGSR) |
| 0384₁₆ | Up-down flag (UDF) |
| 0385₁₆ | |
| 0386₁₆ | Timer A0 (TA0) |
| 0387₁₆ | |
| 0388₁₆ | Timer A1 (TA1) |
| 0389₁₆ | |
| 038A₁₆ | Timer A2 (TA2) |
| 038B₁₆ | |
| 038C₁₆ | Timer A3 (TA3) |
| 038D₁₆ | |
| 038E₁₆ | Timer A4 (TA4) |
| 038F₁₆ | |
| 0390₁₆ | Timer B0 (TB0) |
| 0391₁₆ | |
| 0392₁₆ | Timer B1 (TB1) |
| 0393₁₆ | |
| 0394₁₆ | Timer B2 (TB2) |
| 0395₁₆ | |
| 0396₁₆ | Timer A0 mode register (TA0MR) |
| 0397₁₆ | Timer A1 mode register (TA1MR) |
| 0398₁₆ | Timer A2 mode register (TA2MR) |
| 0399₁₆ | Timer A3 mode register (TA3MR) |
| 039A₁₆ | Timer A4 mode register (TA4MR) |
| 039B₁₆ | Timer B0 mode register (TB0MR) |
| 039C₁₆ | Timer B1 mode register (TB1MR) |
| 039D₁₆ | Timer B2 mode register (TB2MR) |
| 039E₁₆ | |
| 039F₁₆ | |
| 03A0₁₆ | UART0 transmit/receive mode register (U0MR) |
| 03A1₁₆ | UART0 bit rate generator (U0BRG) |
| 03A2₁₆ | UART0 transmit buffer register (U0TB) |
| 03A3₁₆ | |
| 03A4₁₆ | UART0 transmit/receive control register 0 (U0C0) |
| 03A5₁₆ | UART0 transmit/receive control register 1 (U0C1) |
| 03A6₁₆ | UART0 receive buffer register (U0RB) |
| 03A7₁₆ | |
| 03A8₁₆ | UART1 transmit/receive mode register (U1MR) |
| 03A9₁₆ | UART1 bit rate generator (U1BRG) |
| 03AA₁₆ | UART1 transmit buffer register (U1TB) |
| 03AB₁₆ | |
| 03AC₁₆ | UART1 transmit/receive control register 0 (U1C0) |
| 03AD₁₆ | UART1 transmit/receive control register 1 (U1C1) |
| 03AE₁₆ | UART1 receive buffer register (U1RB) |
| 03AF₁₆ | |
| 03B0₁₆ | UART transmit/receive control register 2 (UCON) |
| 03B1₁₆ | |
| 03B2₁₆ | |
| 03B3₁₆ | |
| 03B4₁₆ | |
| 03B5₁₆ | |
| 03B6₁₆ | |
| 03B7₁₆ | |
| 03B8₁₆ | DMA0 request cause select register (DM0SL) |
| 03B9₁₆ | |
| 03BA₁₆ | DMA1 request cause select register (DM1SL) |
| 03BB₁₆ | |
| 03BC₁₆ | CRC data register (CRCD) |
| 03BD₁₆ | |
| 03BE₁₆ | CRC input register (CRCIN) |
| 03BF₁₆ | |

Note : This register is not exist in M30622EC.

**Figure 1.7.2. Location of peripheral unit control registers (2)**

| | |
|---|---|
| 03C0₁₆ 03C1₁₆ | A-D register 0 (AD0) |
| 03C2₁₆ 03C3₁₆ | A-D register 1 (AD1) |
| 03C4₁₆ 03C5₁₆ | A-D register 2 (AD2) |
| 03C6₁₆ 03C7₁₆ | A-D register 3 (AD3) |
| 03C8₁₆ 03C9₁₆ | A-D register 4 (AD4) |
| 03CA₁₆ 03CB₁₆ | A-D register 5 (AD5) |
| 03CC₁₆ 03CD₁₆ | A-D register 6 (AD6) |
| 03CE₁₆ 03CF₁₆ | A-D register 7 (AD7) |
| 03D0₁₆ | |
| 03D1₁₆ | |
| 03D2₁₆ | |
| 03D3₁₆ | |
| 03D4₁₆ | A-D control register 2 (ADCON2) |
| 03D5₁₆ | |
| 03D6₁₆ | A-D control register 0 (ADCON0) |
| 03D7₁₆ | A-D control register 1 (ADCON1) |
| 03D8₁₆ | D-A register 0 (DA0) |
| 03D9₁₆ | |
| 03DA₁₆ | D-A register 1 (DA1) |
| 03DB₁₆ | |
| 03DC₁₆ | D-A control register (DACON) |
| 03DD₁₆ | |
| 03DE₁₆ | |
| 03DF₁₆ | |
| 03E0₁₆ | Port P0 (P0) |
| 03E1₁₆ | Port P1 (P1) |
| 03E2₁₆ | Port P0 direction register (PD0) |
| 03E3₁₆ | Port P1 direction register (PD1) |
| 03E4₁₆ | Port P2 (P2) |
| 03E5₁₆ | Port P3 (P3) |
| 03E6₁₆ | Port P2 direction register (PD2) |
| 03E7₁₆ | Port P3 direction register (PD3) |
| 03E8₁₆ | Port P4 (P4) |
| 03E9₁₆ | Port P5 (P5) |
| 03EA₁₆ | Port P4 direction register (PD4) |
| 03EB₁₆ | Port P5 direction register (PD5) |
| 03EC₁₆ | Port P6 (P6) |
| 03ED₁₆ | Port P7 (P7) |
| 03EE₁₆ | Port P6 direction register (PD6) |
| 03EF₁₆ | Port P7 direction register (PD7) |
| 03F0₁₆ | Port P8 (P8) |
| 03F1₁₆ | Port P9 (P9) |
| 03F2₁₆ | Port P8 direction register (PD8) |
| 03F3₁₆ | Port P9 direction register (PD9) |
| 03F4₁₆ | Port P10 (P10) |
| 03F5₁₆ | |
| 03F6₁₆ | Port P10 direction register (PD10) |
| 03F7₁₆ | |
| 03F8₁₆ | |
| 03F9₁₆ | |
| 03FA₁₆ | |
| 03FB₁₆ | |
| 03FC₁₆ | Pull-up control register 0 (PUR0) |
| 03FD₁₆ | Pull-up control register 1 (PUR1) |
| 03FE₁₆ | Pull-up control register 2 (PUR2) |
| 03FF₁₆ | Port control register (PCR) |

**Figure 1.7.3. Location of peripheral unit control registers (3)**

## Memory Space Expansion Features

Here follows the description of the memory space expansion function.

With the processor running in memory expansion mode or in microprocessor mode, the memory space expansion features provide the means of expanding the accessible space. The memory space expansion features run in one of the three modes given below.

(1) Normal mode (no expansion)

(2) Memory space expansion mode 1 (to be referred as expansion mode 1)

(3) Memory space expansion mode 2 (to be referred as expansion mode 2)

Use bits 5 and 4 (PM15, PM14) of processor mode register 1 to select a desired mode. The external memory area the chip select signal indicates is different in each mode so that the accessible memory space varies. Table 1.8.1 shows how to set individual modes and corresponding accessible memory spaces. For external memory area the chip select signal indicates, see Table 1.12.1 on page 33.

**Table 1.8.1. The way of setting memory space expansion modes and corresponding memory spaces**

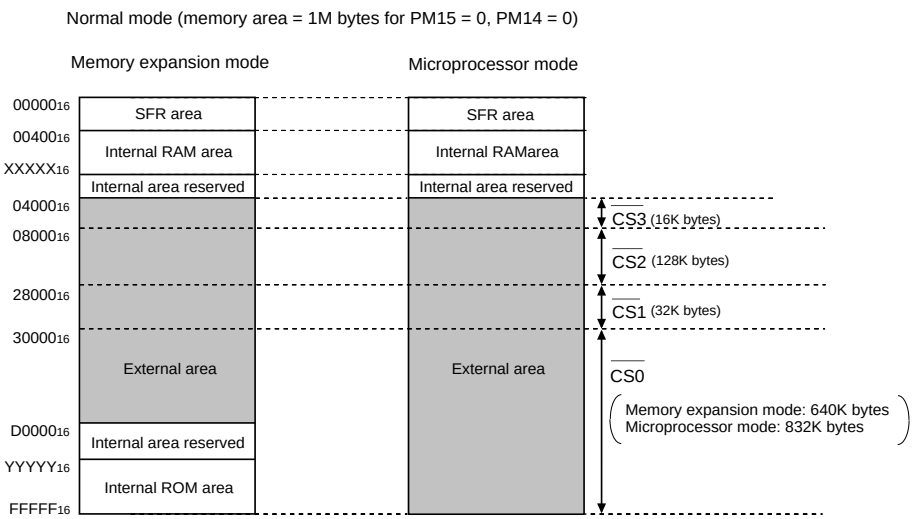
| Expansion mode | How to set PM15 and PM14 | Accessible memory space |
|---|---|---|
| Normal mode (no expansion) | 0, 0 | Up to 1M byte |
| Expansion mode 1 | 1, 0 | Up to 1.2M bytes |
| Expansion mode 2 | 1, 1 | Up to 4M bytes |

Here follows the description of individual modes.

### (1) Normal mode (a mode with memory not expanded)

'Normal mode' means a mode in which memory is not expanded.

Figure 1.8.1 shows the memory maps and the chip select areas in normal mode.




| Type No. | Address XXXXX$_{16}$ | Address YYYYY$_{16}$ |
|---|---|---|
| M30622M4 | 00FFF$_{16}$ | F8000$_{16}$ |
| M30620M8 | 02BFF$_{16}$ | F0000$_{16}$ |
| M30620MC/EC | 02BFF$_{16}$ | E0000$_{16}$ |
| M30622M8/E8 | 013FF$_{16}$ | F0000$_{16}$ |
| M30622MA | 017FF$_{16}$ | E8000$_{16}$ |
| M30622MC/EC | 017FF$_{16}$ | E0000$_{16}$ |
| M30624MG/FG | 053FF$_{16}$ | C0000$_{16}$ |

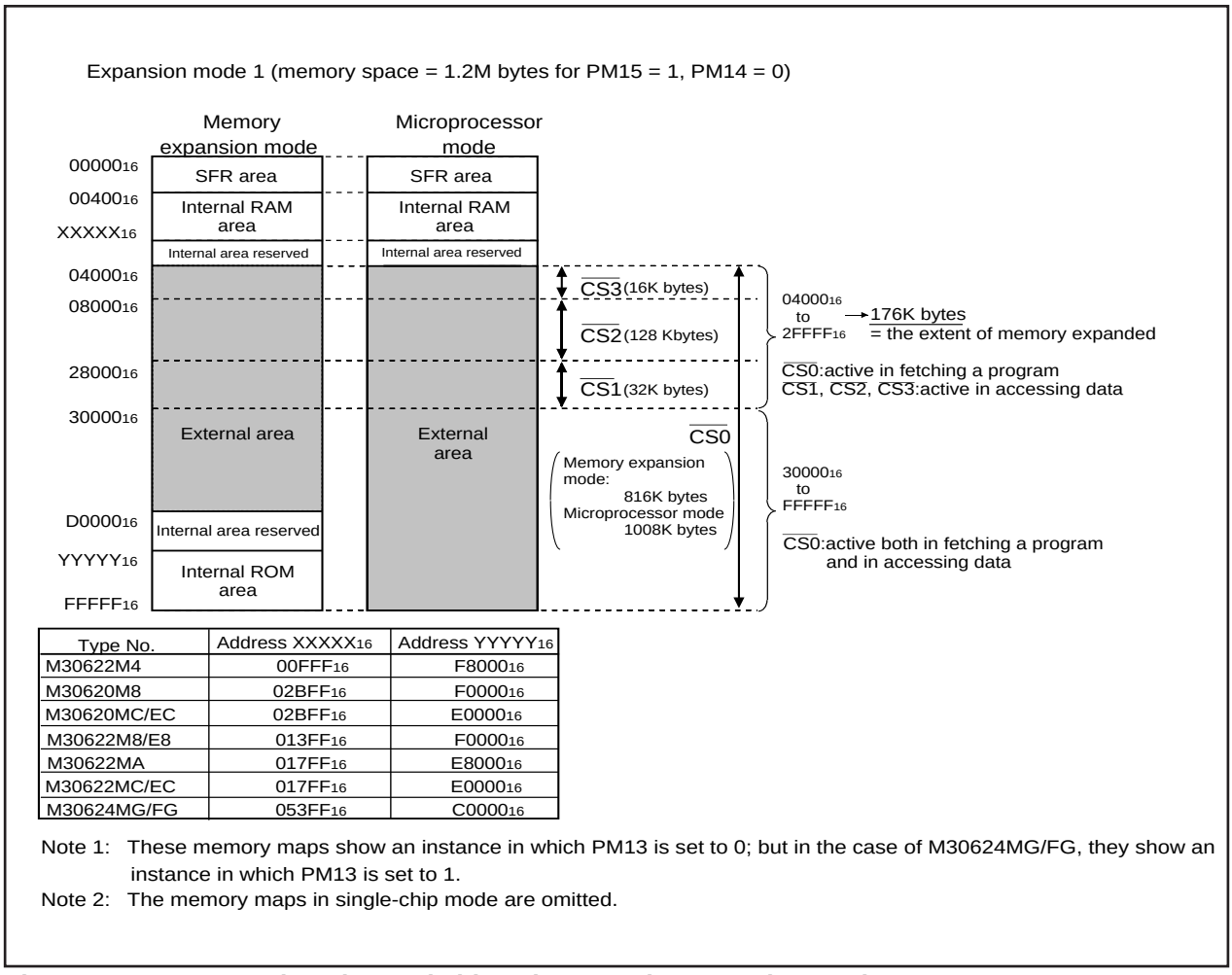
Note 1: These memory maps show an instance in which PM13 is set to 0; but in the case of M30624MG/FG, they show an instance in which PM13 is set to 1.

Note 2: The memory maps in single-chip mode are omitted.

**Figure 1.8.1. The memory maps and the chip select areas in normal mode**

## (2) Expansion mode 1

In this mode, the memory space can be expanded by 176K bytes in addition to that in normal mode.
Figure 1.8.2 shows the memory location and chip select area in expansion mode 1.
In accessing data in expansion mode 1, $\overline{CS3}$, $\overline{CS2}$, and $\overline{CS1}$ go active in the area from $04000_{16}$ through $2FFFF_{16}$; in fetching a program, $\overline{CS0}$ goes active. That is, the address space is expanded by using the area from $04000_{16}$ through $2FFFF_{16}$ (176K bytes) appropriately for accessing data ($\overline{CS3}$, $\overline{CS2}$, $\overline{CS1}$) and fetching a program ($\overline{CS0}$).




| Type No. | Address XXXXX$_{16}$ | Address YYYYY$_{16}$ |
|---|---|---|
| M30622M4 | 00FFF$_{16}$ | F8000$_{16}$ |
| M30620M8 | 02BFF$_{16}$ | F0000$_{16}$ |
| M30620MC/EC | 02BFF$_{16}$ | E0000$_{16}$ |
| M30622M8/E8 | 013FF$_{16}$ | F0000$_{16}$ |
| M30622MA | 017FF$_{16}$ | E8000$_{16}$ |
| M30622MC/EC | 017FF$_{16}$ | E0000$_{16}$ |
| M30624MG/FG | 053FF$_{16}$ | C0000$_{16}$ |

Note 1: These memory maps show an instance in which PM13 is set to 0; but in the case of M30624MG/FG, they show an instance in which PM13 is set to 1.
Note 2: The memory maps in single-chip mode are omitted.

**Figure 1.8.2. Memory location and chip select area in expansion mode 1**

**A connection example**

Figure 1.8.3 shows a connection example of the MCU with the external memories in expansion mode 1. In this example, $\overline{CS0}$ is connected with a 1-M byte flash ROM and $\overline{CS2}$ is connected with a 128-K byte SRAM.

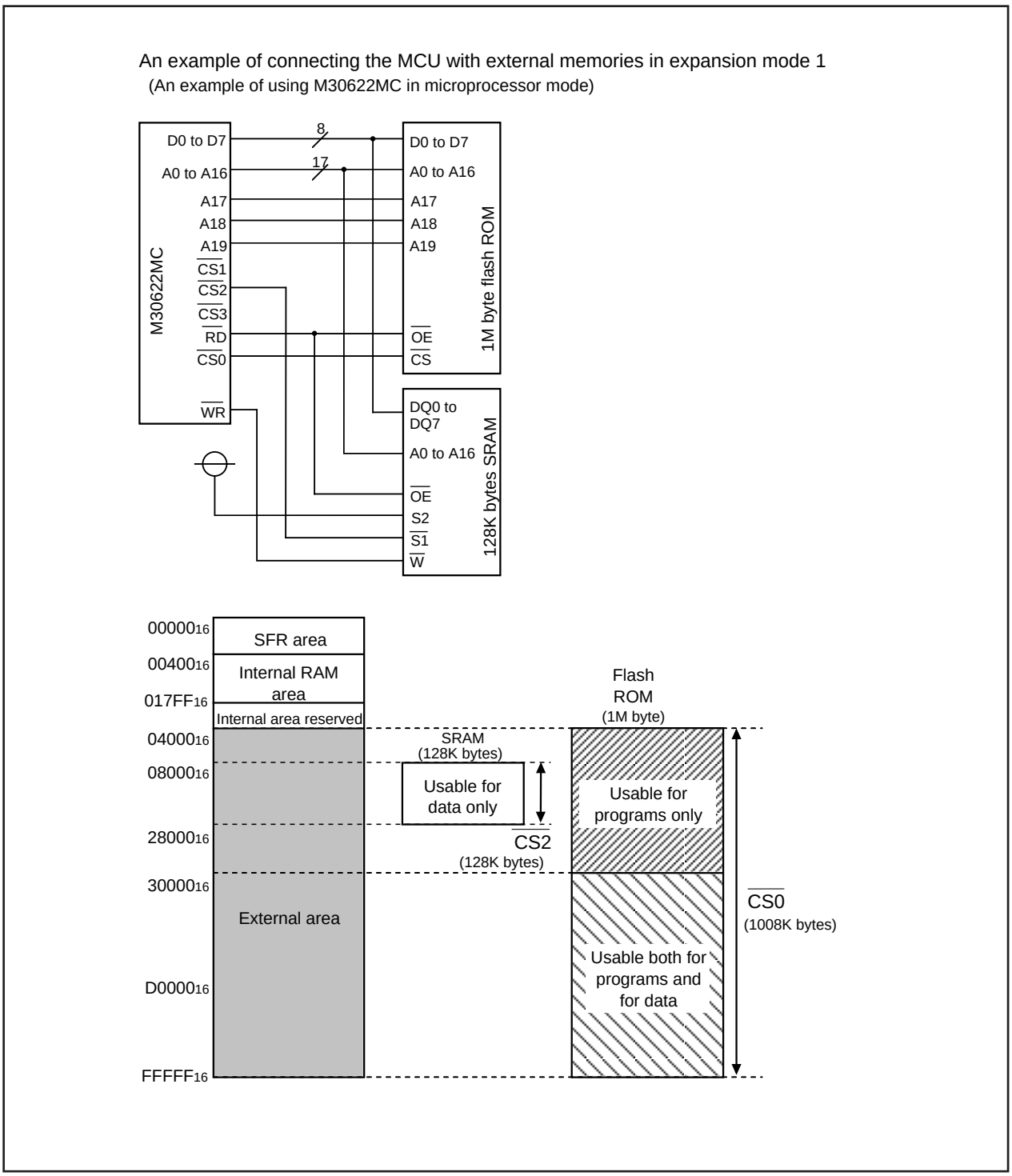



**Figure 1.8.3. External memory connect example in expansion mode 1**

### (3) Expansion mode 2

In expansion mode 2, the data bank register (0000B16) goes effective. Figure 1.8.4 shows the data bank register.

Data bank register

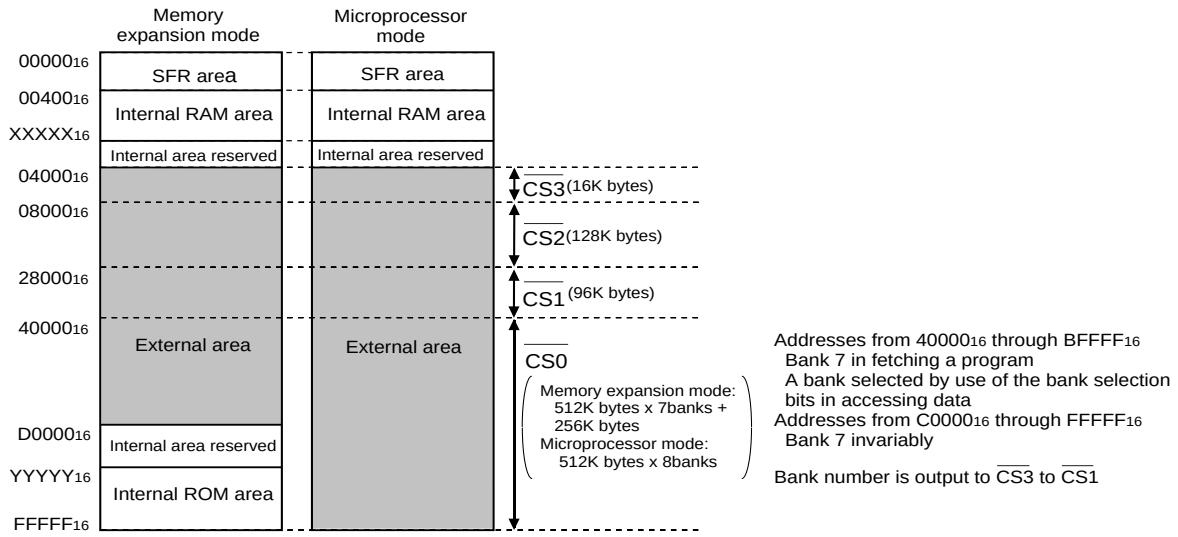
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| ✕ | ✕ | | | | | ✕ | ✕ |

| Symbol | Address | When reset |
|---|---|---|
| DBR | 000B16 | 0016 |

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | | — | — |
| OFS | Offset bit | 0: Not offset<br>1: Offset | ○ | ○ |
| BSR | Bank selection bits | b5 b4 b3<br>0 0 0: Bank 0<br>0 1 0: Bank 2<br>1 0 0: Bank 4<br>1 1 0: Bank 6<br>b5 b4 b3<br>0 0 1: Bank 1<br>0 1 1: Bank 3<br>1 0 1: Bank 5<br>1 1 1: Bank 7 | ○ | ○ |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | | — | — |

**Figure 1.8.4. Data bank register**

Expansion mode 2 (memory space = 4M bytes for PM15 = 1, PM14 = 1)




| Type No. | Address XXXXX16 | Address YYYYY16 |
|---|---|---|
| M30622M4 | 00FFF16 | F800016 |
| M30620M8 | 02BFF16 | F000016 |
| M30620MC/EC | 02BFF16 | E000016 |
| M30622M8/E8 | 013FF16 | F000016 |
| M30622MA | 017FF16 | E800016 |
| M30622MC/EC | 017FF16 | E000016 |
| M30624MG/FG | 053FF16 | C000016 |

Note 1: These memory maps show an instance in which PM13 is set to 0; but in the case of M30624MG/FG, they show an instance in which PM13 is set to 1.
Note 2: The memory maps in single-chip mode are omitted.

**Figure 1.8.5. Memory location and chip select area in expansion mode 2**

The data bank register is made up of the bank selection bits (bits 5 through 3) and the offset bit (bit 2). The bank selection bits are used to set a bank number for accessing data lying between $40000_{16}$ and $BFFFF_{16}$. Assigning 1 to the offset bit provides the means to set offsets covering $40000_{16}$.

Figure 1.8.5 shows the memory location and chip select areas in expansion mode 2.

The area relevant to $\overline{CS0}$ ranges from $40000_{16}$ through $FFFFF_{16}$. As for the area from $40000_{16}$ through $BFFFF_{16}$, the bank number set by use of the bank selection bits are output from the output terminals $\overline{CS3}$ - $\overline{CS1}$ only in accessing data. In fetching a program, bank 7 ($111_2$) is output from $\overline{CS3}$ - $\overline{CS1}$. As for the area from $C0000_{16}$ through $FFFFF_{16}$, bank 7 ($111_2$) is output from $\overline{CS3}$ - $\overline{CS1}$ without regard to accessing data or to fetching a program.

In accessing an area irrelevant to $\overline{CS0}$, a chip select signal $\overline{CS3}$ ($4000_{16}$ - $7FFF_{16}$), $\overline{CS2}$ ($8000_{16}$ - $27FFF_{16}$), and $\overline{CS1}$ ($28000_{16}$ - $3FFFF_{16}$) is output depending on the address as in the past.

Figure 1.8.6 shows an example of connecting the MCU with a 4-M byte ROM and to a 128-K byte SRAM. Connect the chip select of 4-M byte ROM with $\overline{CS0}$. Connect M16C's $\overline{CS3}$, $\overline{CS2}$, and $\overline{CS1}$ with address inputs A21, A20, and A19 respectively. Connect M16C's output A19 with address input A18. Figure 1.8.7 shows the relationship between addresses of the 4-M byte ROM and those of M16C.

With no offsets effected, banks switch from one 512-K byte segment to another 512-K byte segment. Bank selection bits need to be changed in dealing with data lying across the boundary between banks every time a bank switches to another. Assigning 1 to the offset bit brings about offsets covering $40000_{16}$ so that data can be accessed without changing the bank selection bits. For instance, accessing $80000_{16}$ of bank 0 with offsets effected causes the output bank number to turn to 1, and AD19 is inverted to be output; this results in accessing $40000_{16}$ of bank 1.

On the other hand, the SRAM's chip select assumes that $\overline{CS0}$=1 (not selected) and $\overline{CS2}$=0 (selected), so connect $\overline{CS0}$ with S2 and $\overline{CS2}$ with $\overline{S1}$. If the SRAM doesn't have a bipolar chip select input terminal, decode $\overline{CS0}$ and $\overline{CS2}$ externally.

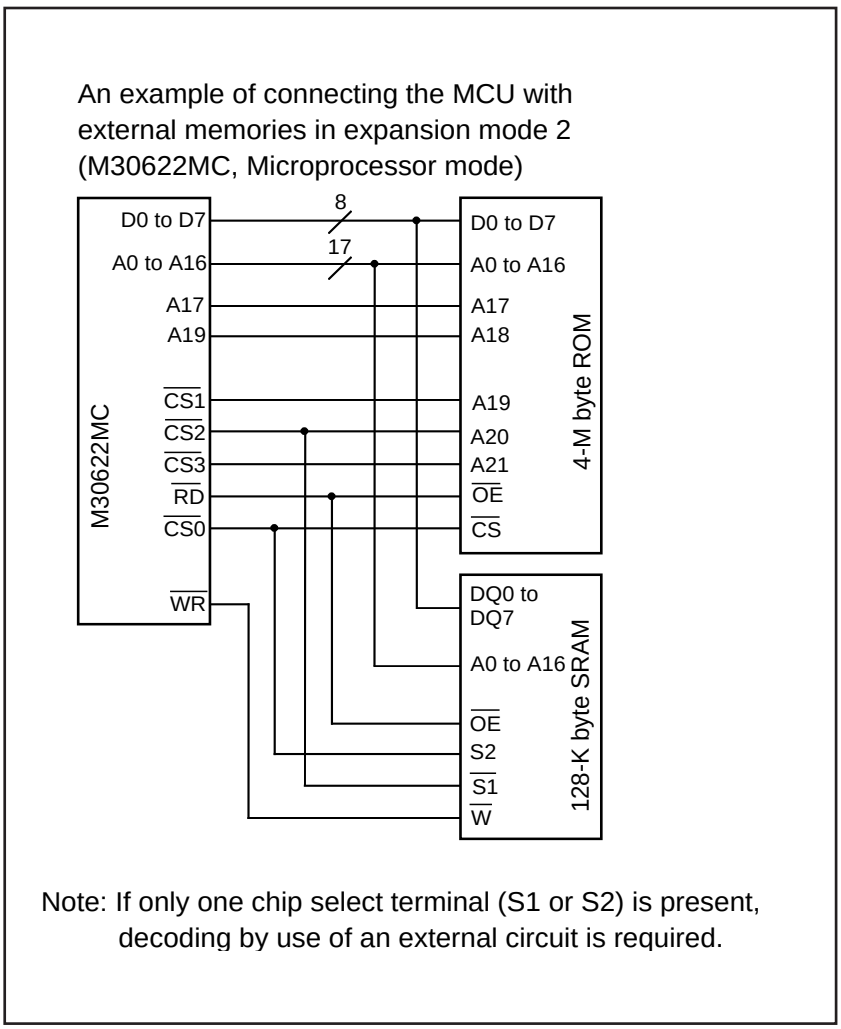



**Figure 1.8.6. An example of connecting the MCU with external memories in expansion mode 2**

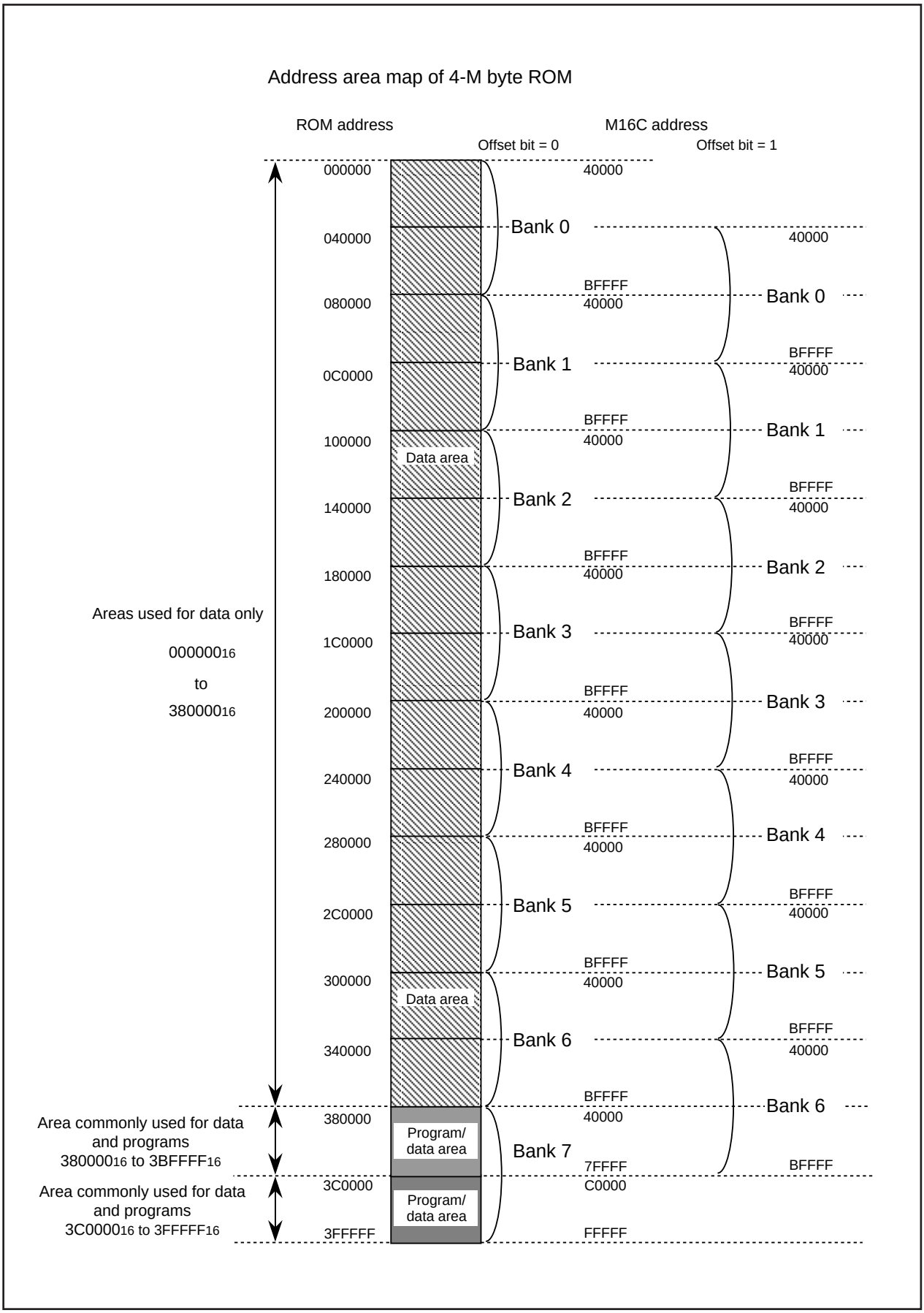



**Figure 1.8.7. Relationship between addresses on 4-M byte ROM and those on M16C**

## Software Reset

Writing “1” to bit 3 of the processor mode register 0 (address $0004_{16}$) 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 $CNV_{SS}$ pin and the processor mode bits (bits 1 and 0 at address $0004_{16}$). Do not set the processor mode bits to “$10_2$”.

Regardless of the level of the $CNV_{SS}$ 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 “$01_2$” to the processor mode is selected bits.

- **Applying Vcc to CNVss pin**

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

Figure 1.9.1 shows the processor mode register 0 and 1.

Figure 1.10.1 shows the memory maps applicable for each of the modes when memeory area dose not be expanded (normal mode).

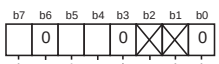
Processor mode register 0 (Note 1)


Symbol PM0 | Address 0004₁₆ | When reset 00₁₆ (Note 2)

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PM00 | Processor mode bit | b1 b0<br>0 0: Single-chip mode<br>0 1: Memory expansion mode<br>1 0: Inhibited<br>1 1: Microprocessor mode | ○ | ○ |
| PM01 | | | ○ | ○ |
| PM02 | R/W mode select bit | 0 : $\overline{RD}$,$\overline{BHE}$,$\overline{WR}$<br>1 : $\overline{RD}$,$\overline{WRH}$,$\overline{WRL}$ | ○ | ○ |
| PM03 | Software reset bit | The device is reset when this bit is set to "1". The value of this bit is "0" when read. | ○ | ○ |
| PM04 | Multiplexed bus space select bit | b5 b4<br>0 0 : Multiplexed bus is not used<br>0 1 : Allocated to CS2 space<br>1 0 : Allocated to CS1 space<br>1 1 : Allocated to entire space (Note4) | ○ | ○ |
| PM05 | | | ○ | ○ |
| PM06 | Port P4₀ to P4₃ function select bit (Note 3) | 0 : Address output<br>1 : Port function<br>(Address is not output) | ○ | ○ |
| PM07 | BCLK output disable bit | 0 : BCLK is output<br>1 : BCLK is not output<br>(Pin is left floating) | ○ | ○ |

Note 1: Set bit 1 of the protect register (address 000A₁₆) to "1" when writing new values to this register.
Note 2: If the Vcc voltage is applied to the CNVss, the value of this register when reset is 03₁₆. (PM00 and PM01 both are set to "1".)
Note 3: Valid in microprocessor and memory expansion modes.
Note 4: 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.

Processor mode register 1 (Note 1)



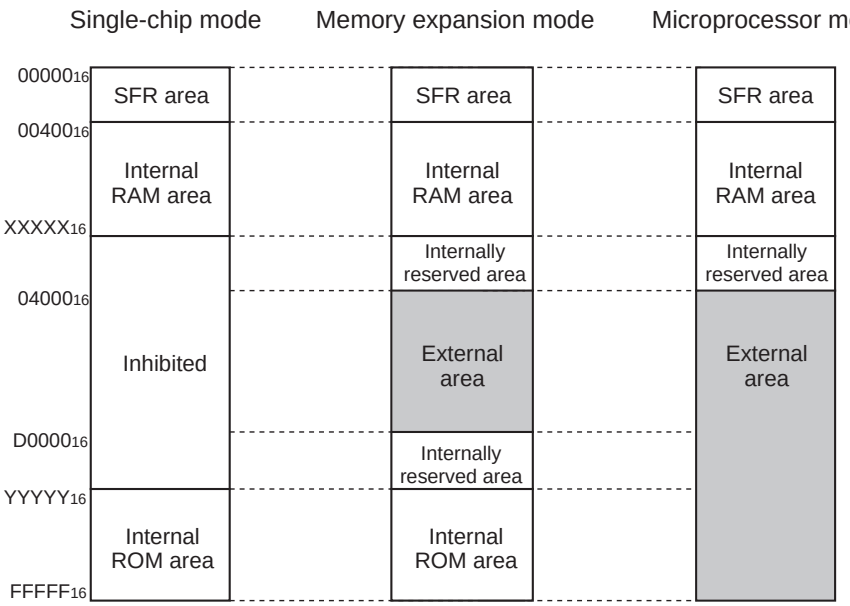

Symbol PM1 | Address 0005₁₆ | When reset 00000XX0₂

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Reserved bit | | Must always be set to "0" | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminate. | | | — | — |
| PM13 | Internal reserved area expansion bit (Note 2) | 0: The same internal reserved area as that of M16C/60 and M16C/61 group<br>1: Expands the internal RAM area and internal ROM area to 23 K bytes and to 256K bytes respectively. (Note 2) | ○ | ○ |
| PM14 | Memory area expansion bit (Note 3) | b5 b4<br>0 0 : Normal mode (Do not expand)<br>0 1 : Inhibited<br>1 0 : Memory area expansion mode 1<br>1 1 : Memory area expansion mode 2 | ○ | ○ |
| PM15 | | | | |
| Reserved bit | | Must always be set to "0" | ○ | ○ |
| PM17 | Wait bit | 0 : No wait state<br>1 : Wait state inserted | ○ | ○ |

Note 1: Set bit 1 of the protect register (address 000A₁₆) to "1" when writing new values to this register.
Note 2: Be sure to set this bit to 0 except products whose RAM size and ROM size exceed 15K bytes and 192K bytes respectively.
In using M30624MG/FG, a product having a RAM of more than 15K bytes and a ROM of more than 192K bytes, set this bit to 1 at the beginning of user program. Specify D000016 or a subsequent address, which becomes an internal ROM area if PM13 is set to 0 at the time reset is revoked, for the reset vector table of user program.
Note 3: With the processor running in memory expansion mode or in microprocessor mode, setting this bit provides the means of expanding the external memory area.
(Normal mode: up to 1M byte, expansion mode 1: up to 1.2 M bytes, expansion mode 2: up to 4M bytes)
For details, see "Memory space expansion functions".

**Figure 1.9.1. Processor mode register 0 and 1**

## Processor Mode




| Type No. | Address XXXXX$_{16}$ | Address YYYYY$_{16}$ |
|---|---|---|
| M30622M4 | 00FFF$_{16}$ | F8000$_{16}$ |
| M30620M8 | 02BFF$_{16}$ | F0000$_{16}$ |
| M30620MC/EC | 02BFF$_{16}$ | E0000$_{16}$ |
| M30622M8/E8 | 013FF$_{16}$ | F0000$_{16}$ |
| M30622MA | 017FF$_{16}$ | E8000$_{16}$ |
| M30622MC/EC | 017FF$_{16}$ | E0000$_{16}$ |
| M30624MG/FG | 053FF$_{16}$ | C0000$_{16}$ |

External area : Accessing this area allows the user to access a device connected externally to the microcomputer.

Note : These memory maps show an instance in which PM13 is set to 0; but in the case of M30624MG/FG, they show an instance in which PM13 is set to 1.

**Figure 1.10.1. Memory maps in each processor mode (without memeory area expansion, normal mode)**

Figure 1.10.2 shows the memory maps and the chip selection areas effected by PM13 (the internal reserved area expansion bit) in each processor mode for the product having an internal RAM of more than 15K bytes and a ROM of more than 192K bytes.

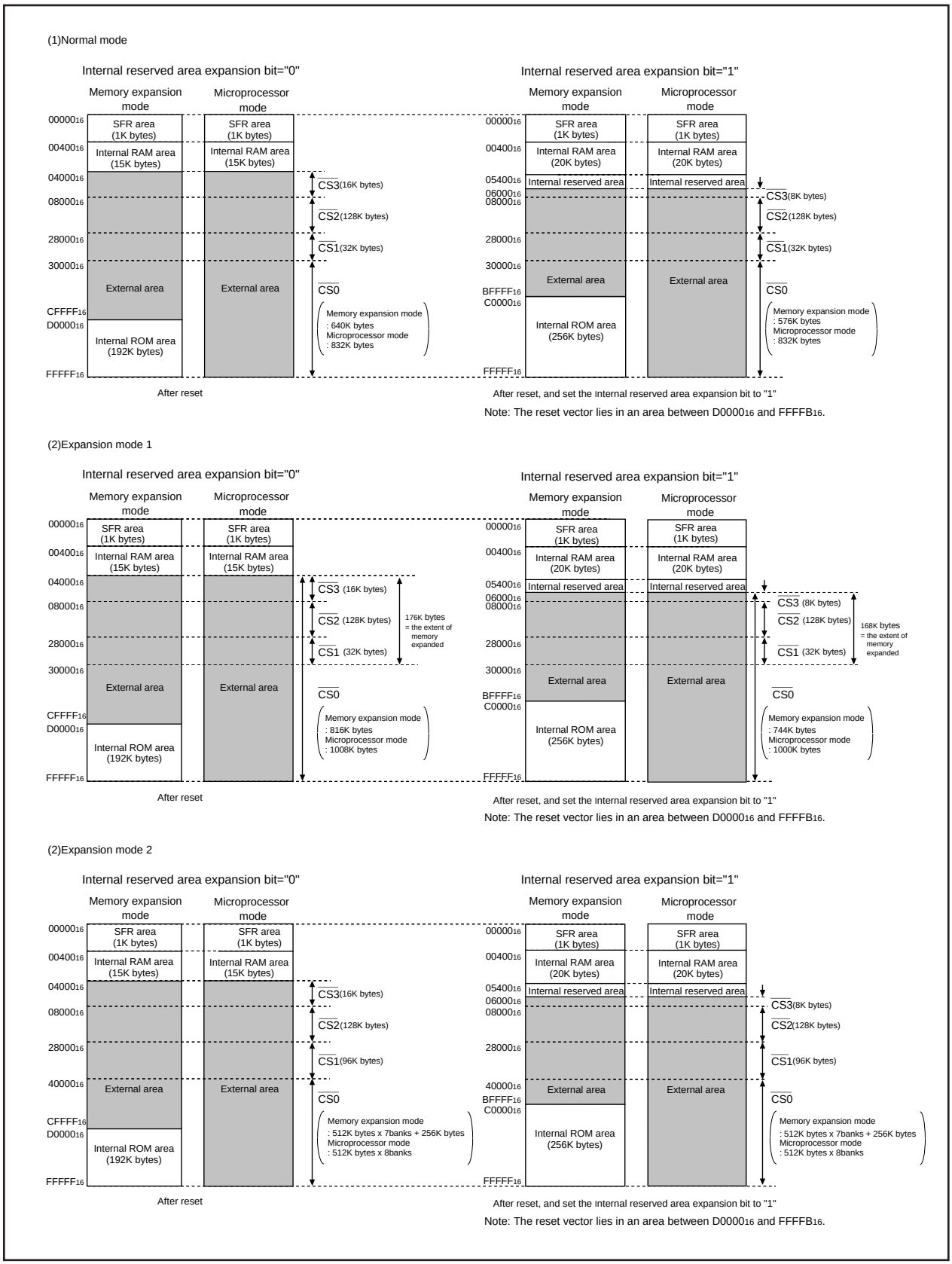



**Figure 1.10.2. Memory location and chip select area in each processor mode**

## Bus Settings

The BYTE pin and bits 4 to 6 of the processor mode register 0 (address $0004_{16}$) are used to change the bus settings. Table 1.11.1 shows the factors used to change the bus settings.

**Table 1.11.1. 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. $P4_0$ to $P4_3$ 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 $P4_0$ to $P4_3$ 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 $D_0$ to $D_7$ are multiplexed with $A_0$ to $A_7$.

  With a 16-bit data bus selected (BYTE pin = “L”), the 8 bits from $D_0$ to $D_7$ are multiplexed with $A_1$ to $A_8$. $D_8$ to $D_{15}$ 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 $P5_6$.

  Before accessing the multiplex bus, always set the CSi wait bit of the chip select control register to “0”.

  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.

**Table 1.11.2. 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" |
| $P0_0$ to $P0_7$ | I/O port | Data bus | Data bus | Data bus | Data bus | I/O port |
| $P1_0$ to $P1_7$ | I/O port | I/O port | Data bus | I/O port | Data bus | I/O port |
| $P2_0$ | I/O port | Address bus /data bus(Note 2) | Address bus | Address bus | Address bus | Address bus /data bus |
| $P2_1$ to $P2_7$ | I/O port | Address bus /data bus(Note 2) | Address bus /data bus(Note 2) | Address bus | Address bus | Address bus /data bus |
| $P3_0$ | I/O port | Address bus | Address bus /data bus(Note 2) | Address bus | Address bus | Address bus |
| $P3_1$ to $P3_7$ | I/O port | Address bus | Address bus | Address bus | Address bus | I/O port |
| $P4_0$ to $P4_3$ Port $P4_0$ to $P4_3$ function select bit = 1 | I/O port | I/O port | I/O port | /O port | I/O port | I/O port |
| $P4_0$ to $P4_3$ Port $P4_0$ to $P4_3$ function select bit = 0 | I/O port | Address bus | Address bus | Address bus | Address bus | I/O port |
| $P4_4$ to $P4_7$ | I/O port | $\overline{CS}$ (chip select) or programmable I/O port (For details, refer to "Bus control") | | | | |
| $P5_0$ to $P5_3$ | 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") | | | | |
| $P5_4$ | I/O port | $\overline{HLDA}$ | $\overline{HLDA}$ | $\overline{HLDA}$ | $\overline{HLDA}$ | $\overline{HLDA}$ |
| $P5_5$ | I/O port | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ | $\overline{HOLD}$ |
| $P5_6$ | I/O port | ALE | ALE | ALE | ALE | ALE |
| $P5_7$ | 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 $A_0$ to $A_{19}$ 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 $D_0$ to $D_7$ function as the data bus. When BYTE is "L", the 16 ports $D_0$ to $D_{15}$ function as the data bus.

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 $P4_4$ to $P4_7$. Bits 0 to 3 of the chip select control register (address $0008_{16}$) 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, $P4_4$ to $P4_7$ 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. Figure 1.12.1 shows the chip select control register.

The chip select signal can be used to split the external area into as many as four blocks. Tables 1.12.1 and 1.12.2 show the external memory areas specified using the chip select signal.

**Table 1.12.1. External areas specified by the chip select signals**

**(A product having an internal RAM equal to or less than 15K bytes and a ROM equal to or less than 192K bytes)(Note)**

| Memory space expansion mode | | Processor mode | Chip select signal | | | |
|---|---|---|---|---|---|---|
| | | | $\overline{CS0}$ | $\overline{CS1}$ | $\overline{CS2}$ | $\overline{CS3}$ |
| Specified address range | Normal mode (PM15,14=0,0) | Memory expansion mode | $30000_{16}$ to $CFFFF_{16}$ (640K bytes) | $28000_{16}$ to $2FFFF_{16}$ (32K bytes) | $08000_{16}$ to $27FFF_{16}$ (128K bytes) | $04000_{16}$ to $07FFF_{16}$ (16K bytes) |
| | | Microprocessor mode | $30000_{16}$ to $FFFFF_{16}$ (832K bytes) | | | |
| | Expansion mode 1 (PM15,14=1,0) | Memory expansion mode | $04000_{16}$ to $CFFFF_{16}$ (816K bytes) | | | |
| | | Microprocessor mode | $04000_{16}$ to $FFFFF_{16}$ (1008K bytes) | | | |
| | Expansion mode 2 (PM15,14=1,1) | Memory expansion mode | $40000_{16}$ to $BFFFF_{16}$ (512K bytes X 7 + 256K bytes) | $28000_{16}$ to $3FFFF_{16}$ (96K bytes) | | |
| | | Microprocessor mode | $40000_{16}$ to $FFFFF_{16}$ (512K bytes X 8) | | | |

Note :Be sure to set bit 3 (PM13) of processor mode register 1 to "0".

**Table 1.12.2. External areas specified by the chip select signals**

**(A product having an internal RAM of more than 15K bytes and a ROM of more than 192K bytes)**

| Memory space expansion mode | | Processor mode | Chip select signal | | | |
|---|---|---|---|---|---|---|
| | | | $\overline{CS0}$ | $\overline{CS1}$ | $\overline{CS2}$ | $\overline{CS3}$ |
| Specified address range | Normal mode (PM15,14=0,0) | Memory expansion mode | When PM13=0 $30000_{16}$ to $CFFFF_{16}$ (640K bytes) | $28000_{16}$ to $2FFFF_{16}$ (32K bytes) | $08000_{16}$ to $27FFF_{16}$ (128K bytes) | When PM13=0 $04000_{16}$ to $07FFF_{16}$ (16K bytes) When PM13=1 $06000_{16}$ to $07FFF_{16}$ (8K bytes) |
| | | | When PM13=1 $30000_{16}$ to $BFFFF_{16}$ (576K bytes) | | | |
| | | Microprocessor mode | $03000_{16}$ to $FFFFF_{16}$ (816K bytes) | | | |
| | Expansion mode 1 (PM15,14=1,0) | Memory expansion mode | When PM13=0 $04000_{16}$ to $CFFFF_{16}$ (816K bytes) | | | |
| | | | When PM13=1 $06000_{16}$ to $BFFFF_{16}$ (744K bytes) | | | |
| | | Microprocessor mode | When PM13=0 $04000_{16}$ to $FFFFF_{16}$ (1008K bytes) | | | |
| | | | When PM13=1 $06000_{16}$ to $FFFFF_{16}$ (1000K bytes) | | | |
| | Expansion mode 2 (PM15,14=1,1) | Memory expansion mode | $40000_{16}$ to $BFFFF_{16}$ (512K bytes X +256K bytes) | $28000_{16}$ to $3FFFF_{16}$ (96K bytes) | | |
| | | Microprocessor mode | $40000_{16}$ to $FFFFF_{16}$ (512K bytes X 8) | | | |

Chip select control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: CSR
Address: $0008_{16}$
When reset: $01_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CS0 | $\overline{CS0}$ output enable bit | 0 : Chip select output disabled (Normal port pin) 1 : Chip select output enabled | ○ | ○ |
| CS1 | $\overline{CS1}$ output enable bit | | ○ | ○ |
| CS2 | $\overline{CS2}$ output enable bit | | ○ | ○ |
| CS3 | $\overline{CS3}$ output enable bit | | ○ | ○ |
| CS0W | $\overline{CS0}$ wait bit | 0 : Wait state inserted 1 : No wait state | ○ | ○ |
| CS1W | $\overline{CS1}$ wait bit | | ○ | ○ |
| CS2W | $\overline{CS2}$ wait bit | | ○ | ○ |
| CS3W | $\overline{CS3}$ wait bit | | ○ | ○ |

**Figure 1.12.1. 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 $0004_{16}$) select the combinations of $\overline{RD}$, $\overline{BHE}$, and $\overline{WR}$ signals or $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ signals. With an 8-bit data bus (BYTE pin = "H"), use the combination of $\overline{RD}$, $\overline{WR}$, and $\overline{BHE}$ signals. (Set bit 2 of the processor mode register 0 (address $0004_{16}$) to "0".) Tables 1.12.3 and 1.12.4 show the operation of these signals.
After a reset has been cancelled, the combination of $\overline{RD}$, $\overline{WR}$, and $\overline{BHE}$ signals is automatically selected. When switching to the $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ combination, do not write to external memory until bit 2 of the processor mode register 0 (address $0004_{16}$) 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 $000A_{16}$) to "1".

**Table 1.12.3. Operation of $\overline{RD}$, $\overline{WRL}$, and $\overline{WRH}$ signals**

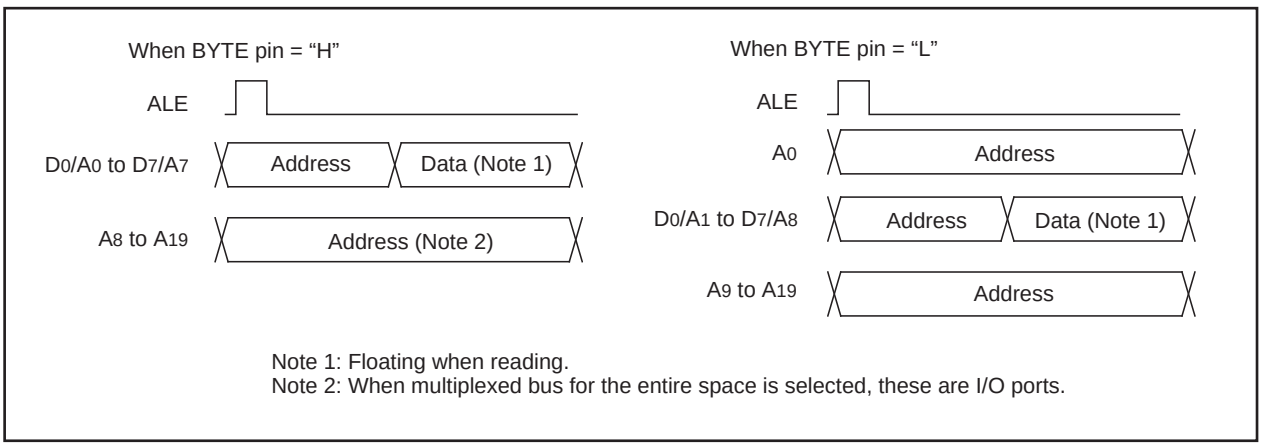
| Data bus width | $\overline{RD}$ | $\overline{WRL}$ | $\overline{WRH}$ | Status of external data bus |
|---|---|---|---|---|
| 16-bit (BYTE = "L") | L | H | H | Read data |
| | H | L | H | Write 1 byte of data to even address |
| | H | H | L | Write 1 byte of data to odd address |
| | H | L | L | Write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

**Table 1.12.4. Operation of $\overline{RD}$, $\overline{WR}$, and $\overline{BHE}$ signals**

| Data bus width | $\overline{RD}$ | $\overline{WR}$ | $\overline{BHE}$ | 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.




Note 1: Floating when reading.
Note 2: When multiplexed bus for the entire space is selected, these are I/O ports.

**Figure 1.12.2. 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 1.12.3, 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 1.12.5 shows the state of the microcomputer with the bus in the wait state, and Figure 1.12.3 shows an example in which the $\overline{RD}$ signal is prolonged by the $\overline{RDY}$ signal.

The $\overline{RDY}$ signal is useful in accessing an external area in a bus cycle with software wait operating. The $\overline{RDY}$ signal turns null if software wait is not operating, but the non-used pins should be properly treated in this instance too.

**Table 1.12.5. Microcomputer status in ready state (Note)**

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

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

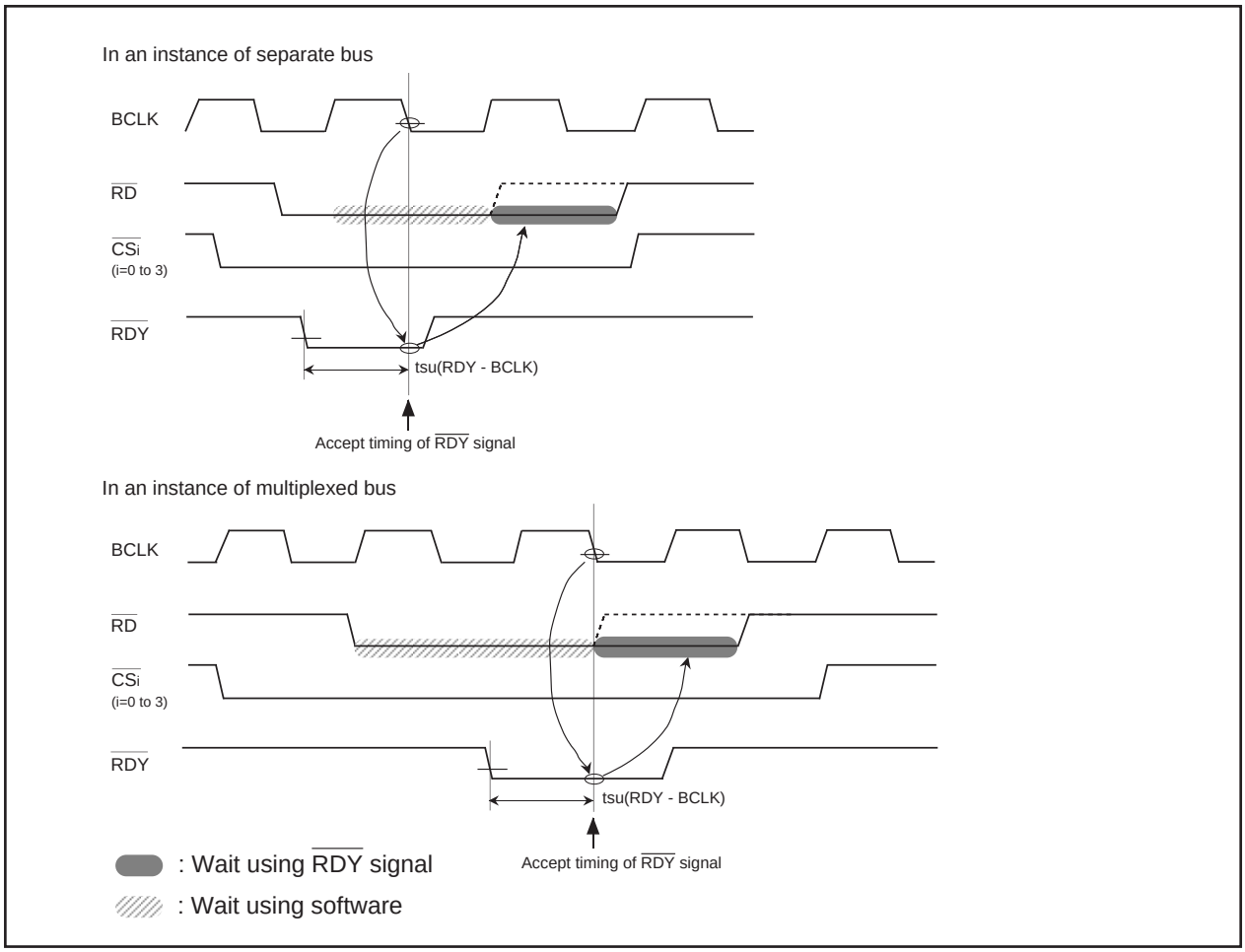



**Figure 1.12.3. Example of $\overline{RD}$ signal extended by $\overline{RDY}$ signal**

## (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 1.12.6 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.

$\overline{\text{HOLD}}$ > DMAC > CPU

**Figure 1.12.4. Bus-using priorities**

**Table 1.12.6. Microcomputer status in hold state**

| Item | | Status |
|---|---|---|
| Oscillation | | ON |
| R/$\overline{\text{W}}$ signal, address bus, data bus, $\overline{\text{CS}}$, $\overline{\text{BHE}}$ | | 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 1.12.7 shows the external bus status when the internal area is accessed.

**Table 1.12.7. 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 ($0004_{16}$) (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 $000A_{16}$) 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 $0005_{16}$) (Note) and bits 4 to 7 of the chip select control register (address $0008_{16}$).

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”, bits 4 to 7 of the chip select control register are invalid and a wait is applied to all external memory areas (two or three BCLK cycles). Set this bit after referring to the recommended operating conditions (main clock input oscillation frequency) of the electric characteristics.

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 $\overline{CS0}$ to $\overline{CS3}$. 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, the corresponding bits of the chip select control register must be set to “0” if using the multiplex bus to access the external memory area.

Table 1.12.8 shows the software wait and bus cycles. Figure 1.12.5 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 $000A_{16}$) to “1”.

**Table 1.12.8. 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 (Note) | 3 BCLK cycles |
| | Multiplex bus | 1 | 0 (Note) | 3 BCLK cycles |

Note: Always set to “0”.

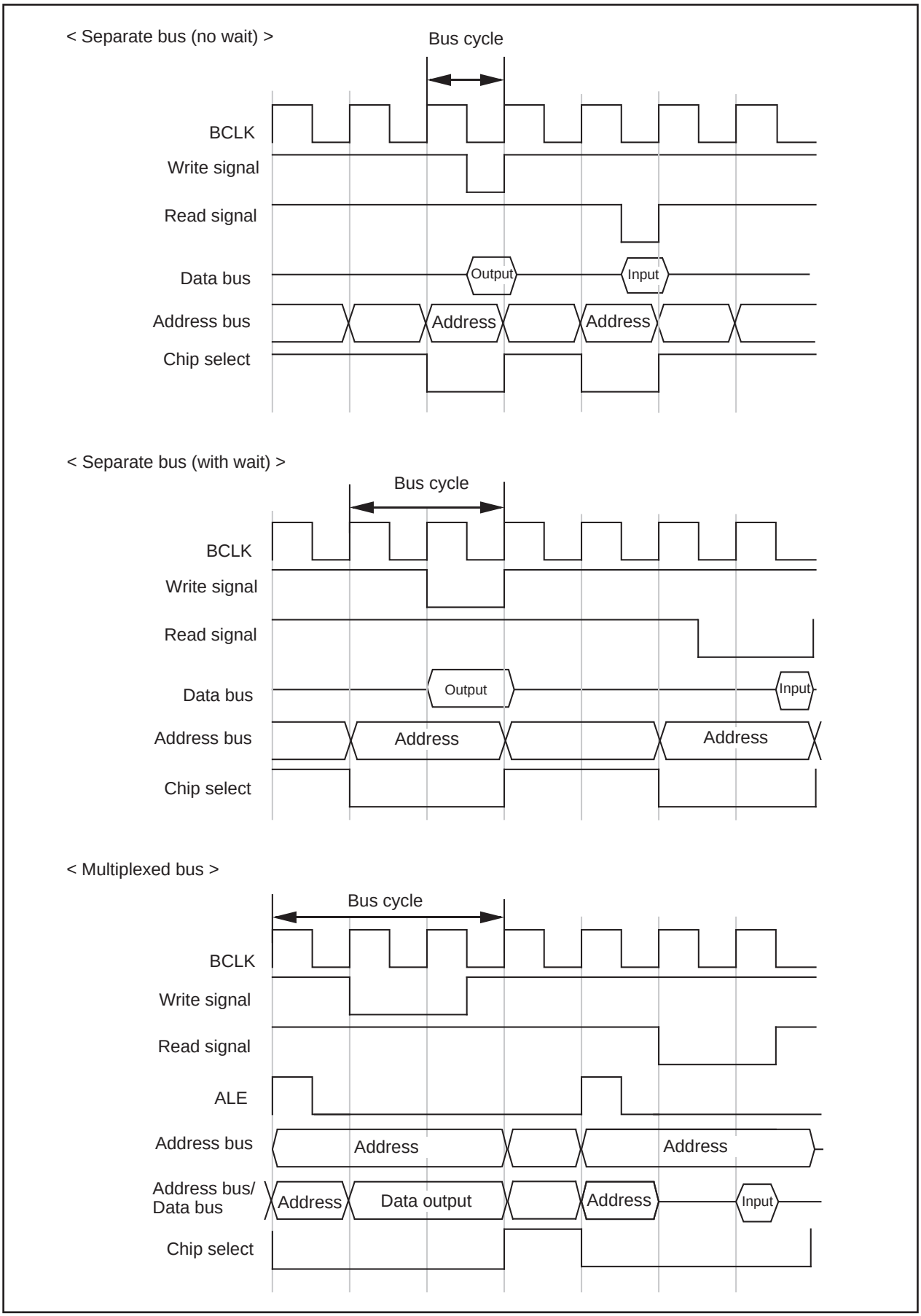



**Figure 1.12.5. Typical bus timings using software wait**

## Clock Generating Circuit

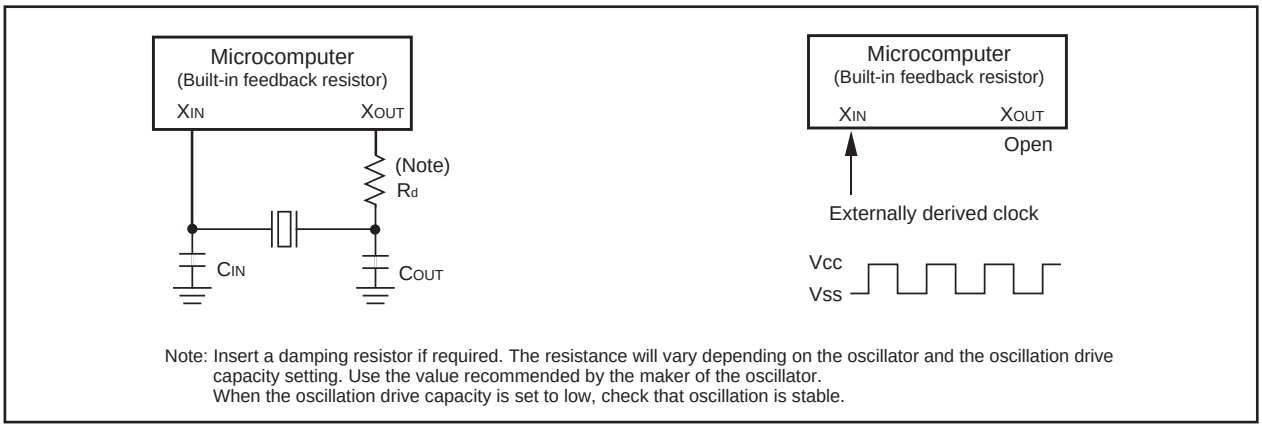
The clock generating circuit contains two oscillator circuits that supply the operating clock sources to the CPU and internal peripheral units.

**Table 1.13.1. Main clock and sub clock generating circuits**

| | Main clock generating circuit | Sub clock generating circuit |
|---|---|---|
| Use of clock | • CPU's operating clock source<br>• Internal peripheral units' operating clock source | • CPU's operating clock source<br>• Timer A/B's count clock source |
| 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 | |

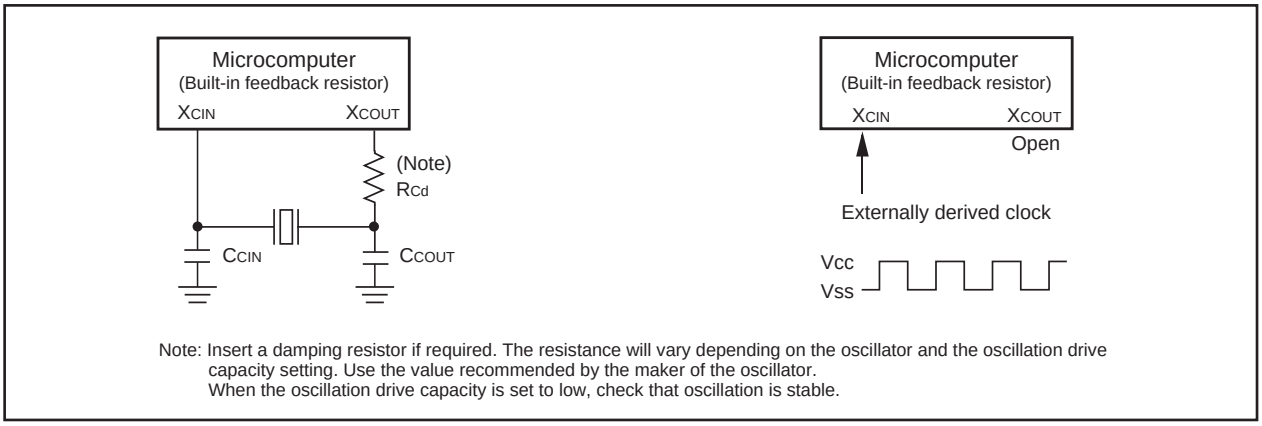
## Example of oscillator circuit

Figure 1.13.1 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 1.13.2 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 1.13.1 and 1.13.2 vary with each oscillator used. Use the values recommended by the manufacturer of your oscillator.




Note: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable.

**Figure 1.13.1. Examples of main clock**




Note: Insert a damping resistor if required. The resistance will vary depending on the oscillator and the oscillation drive capacity setting. Use the value recommended by the maker of the oscillator.
When the oscillation drive capacity is set to low, check that oscillation is stable.

**Figure 1.13.2. Examples of sub clock**

## Clock Control

Figure 1.13.3 shows the block diagram of the clock generating circuit.

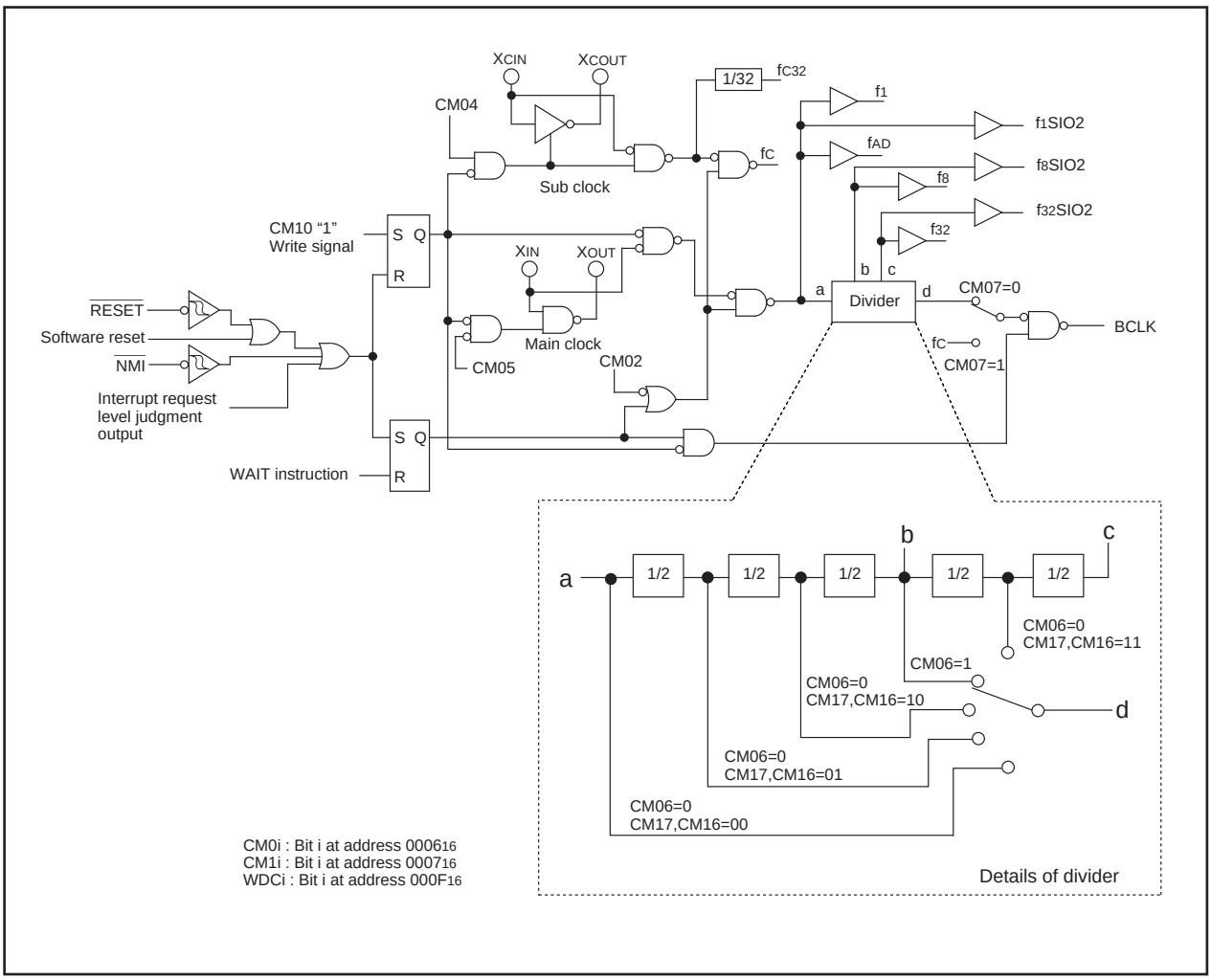



**Figure 1.13.3. Clock generating circuit**

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$_{16}$). Stopping the clock reduces the power dissipation.

After the oscillation of the main clock oscillation circuit has stabilized, the drive capacity of the XOUT pin can be reduced using the XIN-XOUT drive capacity select bit (bit 5 at address 0007$_{16}$). Reducing the drive capacity of the XOUT pin reduces the power dissipation. This bit defaults to “1” when shifting to stop mode and after a reset.

## (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$_{16}$), the sub clock can be selected as the BCLK by using the system clock select bit (bit 7 at address 0006$_{16}$). 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 XCOUT pin can be reduced using the XCIN-XCOUT drive capacity select bit (bit 3 at address 0006$_{16}$). Reducing the drive capacity of the XCOUT pin 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 either the main clock or fc or is derived by dividing the main clock by 2, 4, 8, or 16. The BCLK is derived by dividing the main clock by 8 after a reset.

When shifting to stop mode, the main clock division select bit (bit 6 at 0006$_{16}$) is set to “1”.

## (4) Peripheral function clock

- **$f_1$, $f_8$, $f_{32}$, $f_{1SIO2}$, $f_{8SIO2}$,$f_{32SIO2}$**

The clock for the peripheral devices is derived from the main clock or by dividing it by 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$_{16}$) to “1” and then executing a WAIT instruction.

- **$f_{AD}$**

This clock has the same frequency as the main clock and is used for A-D conversion.

## (5) $f_{C32}$

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

## (6) $f_C$

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

Figure 1.13.4 shows the system clock control registers 0 and 1.

System clock control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: CM0 — Address: 0006₁₆ — When reset: 48₁₆

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CM00 | Clock output function select bit | b1 b0<br>0 0 : I/O port P5₇<br>0 1 : fC output<br>1 0 : f8 output<br>1 1 : f32 output | O | O |
| CM01 | | | O | O |
| CM02 | WAIT peripheral function clock stop bit | 0 : Do not stop f1, f8, f32 in wait mode<br>1 : Stop f1, f8, f32 in wait mode | O | O |
| CM03 | XCIN-XCOUT drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | O | O |
| CM04 | Port XC select bit | 0 : I/O port<br>1 : XCIN-XCOUT generation | O | O |
| CM05 | Main clock (XIN-XOUT) stop bit (Note 4) (Note 5) | 0 : On<br>1 : Off | O | O |
| CM06 | Main clock division select bit 0 (Note 2) | 0 : CM16 and CM17 valid<br>1 : Division by 8 mode | O | O |
| CM07 | System clock select bit (Note 6) | 0 : XIN, XOUT<br>1 : XCIN, XCOUT | O | O |

Note 1: Set bit 0 of the protect register (address 000A16) to "1" before writing to this register.
Note 2: Changes to "1" when shiffing to stop mode.
Note 3: When entering power saving mode, main clock stops using this bit. When returning from stop mode and operating with XIN, set this bit to "0". When main clock oscillation is operating by itself, set system clock select bit (CM07) to "1" before setting this bit to "1".
Note 4: When inputting external clock, only clock oscillation buffer is stopped and clock input is acceptable.
Note 5: If this bit is set to "1", XOUT turns "H". The built-in feedback resistor remains ON, so XIN turns pulled up to XOUT ("H") via the feedback resistor.
Note 6: Set port Xc select bit (CM04) to "1" before writing to this bit. The both bits can not be written at the same time.

System clock control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | 0 | 0 | 0 | |

Symbol: CM1 — Address: 0007₁₆ — When reset: 20₁₆

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CM10 | All clock stop control bit (Note4) | 0 : Clock on<br>1 : All clocks off (stop mode) | O | O |
| Reserved bit | | Always set to "0" | O | O |
| Reserved bit | | Always set to "0" | O | O |
| Reserved bit | | Always set to "0" | O | O |
| Reserved bit | | Always set to "0" | O | O |
| CM15 | XIN-XOUT drive capacity select bit (Note 2) | 0 : LOW<br>1 : HIGH | O | O |
| CM16 | Main clock division select bit 1 (Note 3) | b7 b6<br>0 0 : No division mode<br>0 1 : Division by 2 mode<br>1 0 : Division by 4 mode<br>1 1 : Division by 16 mode | O | O |
| CM17 | | | | |

Note 1: Set bit 0 of the protect register (address 000A16) to "1" before writing to this register.
Note 2: Changes to "1" when shiffing to stop mode.
Note 3: Can be selected when bit 6 of the system clock control register 0 (address 0006₁₆) is "0". If "1", division mode is fixed at 8.
Note 4: If this bit is set to "1", XOUT turns "H", and the built-in feedback resistor turns null.

**Figure 1.13.4. Clock control registers 0 and 1**

## Clock Output

In single-chip mode, the clock output function select bits (bits 0 and 1 at address $0006_{16}$) enable $f_8$, $f_{32}$, or fc to be output from the $P5_7$/$CLK_{OUT}$ pin. When the WAIT peripheral function clock stop bit (bit 2 at address $0006_{16}$) is set to “1”, the output of $f_8$ and $f_{32}$ stops when a WAIT instruction is executed.

## Stop Mode

Writing “1” to the all-clock stop control bit (bit 0 at address $0007_{16}$) stops all oscillation and the microcomputer enters stop mode. In stop mode, the content of the internal RAM is retained provided that $V_{CC}$ remains above 2V.

Because the oscillation , BCLK, $f_1$ to $f_{32}$, $f_{1SIO2}$ to $f_{32SIO2}$, $f_C$, $f_{C32}$, and $f_{AD}$ 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 1.13.2 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 to stop mode, the main clock division select bit 0 (bit 6 at $0006_{16}$) is set to “1”.

**Table 1.13.2. Port status during stop mode**

| Pin | | Memory expansion mode<br>Microprocessor mode | Single-chip mode |
|---|---|---|---|
| Address bus, data bus, $\overline{CS0}$ to $\overline{CS3}$ | | Retains status before stop mode | |
| $\overline{RD}$, $\overline{WR}$, $\overline{BHE}$, $\overline{WRL}$, $\overline{WRH}$ | | “H” | |
| $\overline{HLDA}$, BCLK | | “H” | |
| ALE | | “H” | |
| Port | | Retains status before stop mode | Retains status before stop mode |
| $CLK_{OUT}$ | When fc selected | Valid only in single-chip mode | “H” |
| | When $f_8$, $f_{32}$ selected | Valid only in single-chip mode | Retains status before stop mode |

## Wait Mode

When a WAIT instruction is executed, the BCLK stops and the microcomputer enters the wait mode. In this mode, oscillation continues but the 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 1.13.3 shows the status of the ports in wait mode.

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

**Table 1.13.3. Port status during wait mode**

| Pin | | Memory expansion mode<br>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 |
| $CLK_{OUT}$ | When $f_C$ selected | Valid only in single-chip mode | Does not stop |
| | When $f_8$, $f_{32}$ selected | Valid only in single-chip mode | Does not stop when the WAIT peripheral function clock stop bit is “0”.<br>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.

After a reset, operation defaults to division by 8 mode. When shifting to stop mode, the main clock division select bit 0 (bit 6 at address $0006_{16}$) is set to “1”. 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. Note that oscillation of the main clock must have stabilized before transferring from this mode to another mode.

### (4) Division by 16 mode

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

### (5) No-division mode

The main clock is used as the BCLK.

### (6) Low-speed mode

$f_C$ 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

$f_C$ is the BCLK and the main clock is stopped.

**Table 1.13.4. 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 1.13.5 is the state transition diagram of the above modes.

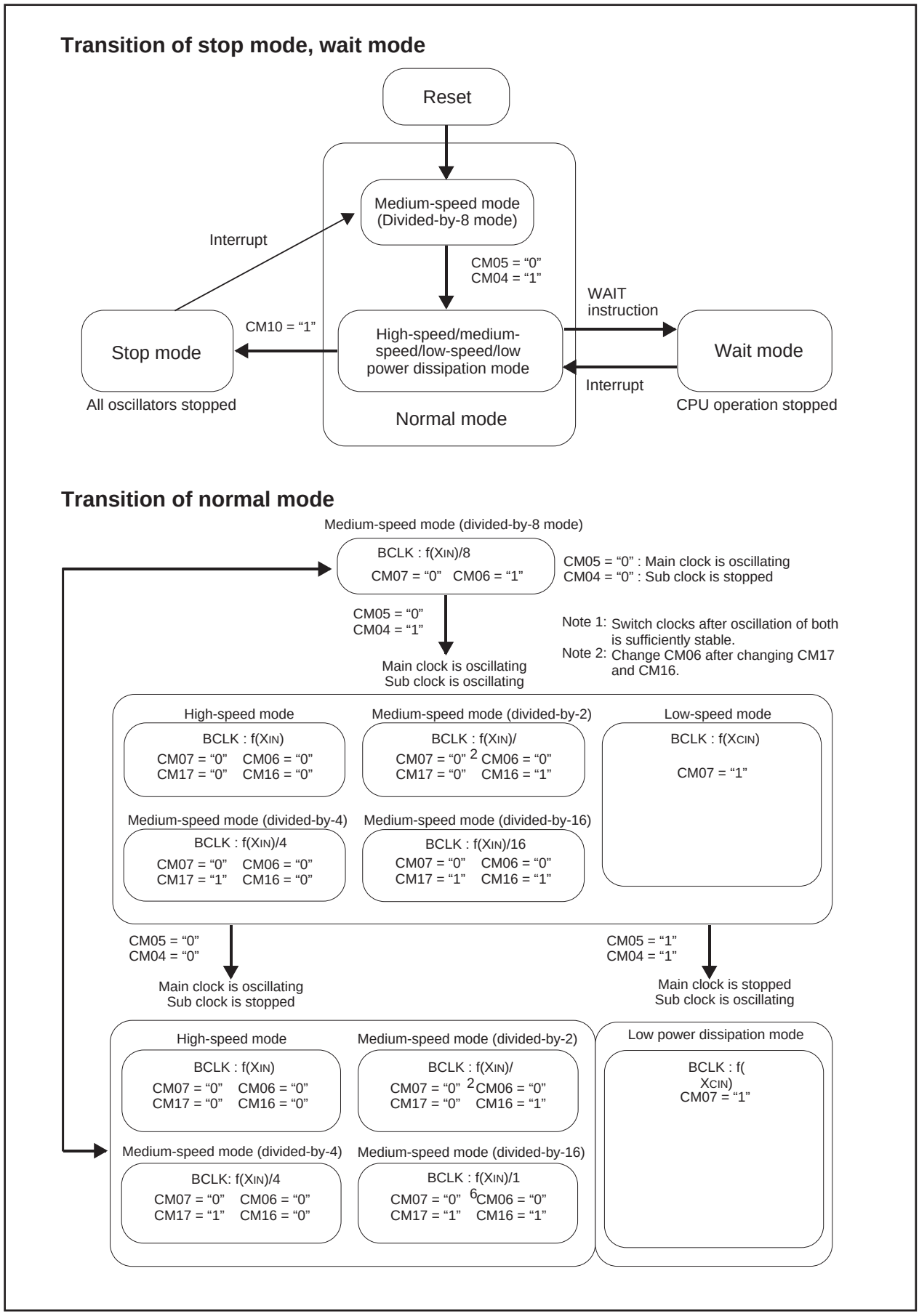


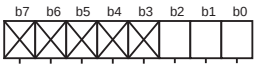

**Figure 1.13.5. State transition diagram of Power control mode**

## 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 1.13.6 shows the protect register. The values in the processor mode register 0 (address $0004_{16}$), processor mode register 1 (address $0005_{16}$), system clock control register 0 (address $0006_{16}$), system clock control register 1 (address $0007_{16}$), port P9 direction register (address $03F3_{16}$) , SI/O3 control register (address $0362_{16}$) and SI/O4 control register (address $0366_{16}$) 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 and SI/Oi control register (i=3,4) write-enable bit (bit 2 at address $000A_{16}$), 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 $000A_{16}$) and processor mode register 0 and 1 write-enable bit (bit 1 at $000A_{16}$) 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”.

Protect register




Symbol PRCR — Address $000A_{16}$ — When reset $XXXXX000_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PRC0 | Enables writing to system clock control registers 0 and 1 (addresses $0006_{16}$ and $0007_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC1 | Enables writing to processor mode registers 0 and 1 (addresses $0004_{16}$ and $0005_{16}$) | 0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| PRC2 | Enables writing to port P9 direction register (address $03F3_{16}$) (Note) | 0 : Write-inhibited<br>1 : Write-enabled | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminate. | | | — | — |

Note: Writing a value to an address after “1” is written to this bit returns the bit to “0” . Other bits do not automatically return to “0” and they must therefore be reset by the program.

**Figure 1.13.6. Protect register**

## Overview of Interrupt

### Type of Interrupts

Figure 1.14.1 lists the types of interrupts.

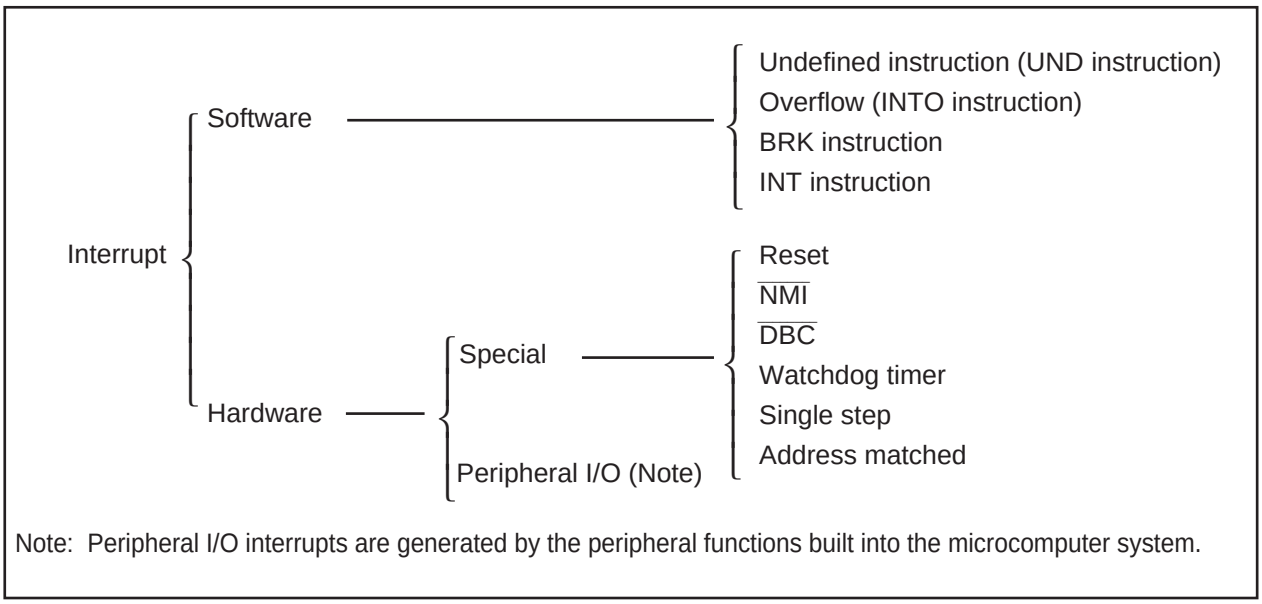



**Figure 1.14.1. Classification of interrupts**

- 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 assiging 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. If change the U flag to "0" and select the interrupt stack pointer (ISP), and then execute 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, UART2/NACK, SI/O3 and SI/O4 transmission interrupt**

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

- **UART0, UART1, UART2/ACK, SI/O3 and SI/O4 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 B5 interrupt**

  These are interrupts that timer B generates.

- **$\overline{\text{INT0}}$ interrupt through $\overline{\text{INT5}}$ interrupt**

  An $\overline{\text{INT}}$ interrupt occurs if either a rising edge or a falling edge or a both 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 1.14.2 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.

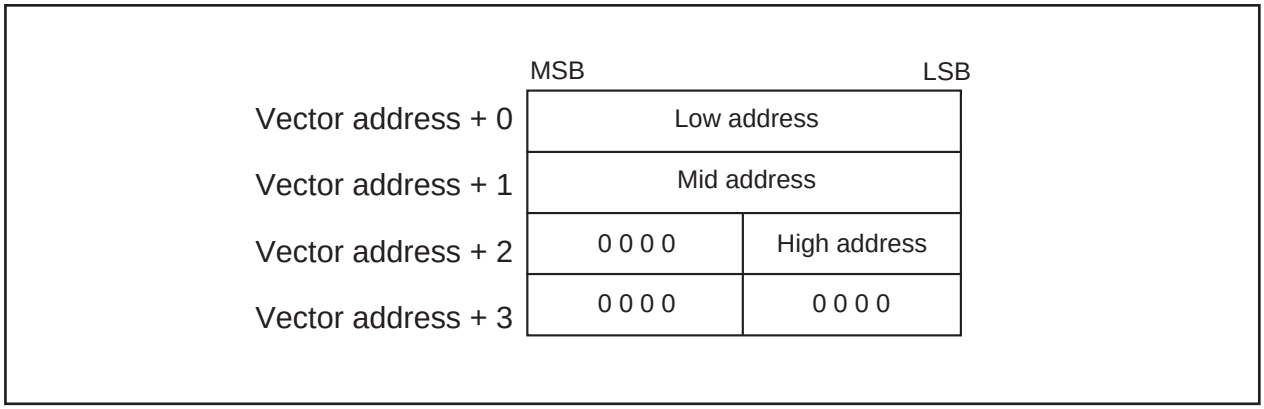



**Figure 1.14.2. Format for specifying interrupt vector addresses**

**• 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 FFFDC16 to FFFFF16. One vector table comprises four bytes. Set the first address of interrupt routine in each vector table. Table 1.14.1 shows the interrupts assigned to the fixed vector tables and addresses of vector tables.

**Table 1.14.1. Interrupts assigned to the fixed vector tables and addresses of vector tables**

| Interrupt source | Vector table addresses<br>Address (L) to address (H) | Remarks |
|---|---|---|
| Undefined instruction | FFFDC16 to FFFDF16 | Interrupt on UND instruction |
| Overflow | FFFE016 to FFFE316 | Interrupt on INTO instruction |
| BRK instruction | FFFE416 to FFFE716 | If the vector contains FF16, program execution starts from the address shown by the vector in the variable vector table |
| Address match | FFFE816 to FFFEB16 | There is an address-matching interrupt enable bit |
| Single step (Note) | FFFEC16 to FFFEF16 | Do not use |
| Watchdog timer | FFFF016 to FFFF316 | |
| $\overline{\text{DBC}}$ (Note) | FFFF416 to FFFF716 | Do not use |
| NMI | FFFF816 to FFFFB16 | External interrupt by input to $\overline{\text{NMI}}$ pin |
| Reset | FFFFC16 to FFFFF16 | |

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 1.14.2 shows the interrupts assigned to the variable vector tables and addresses of vector tables.

**Table 1.14.2. Interrupts assigned to the variable vector tables and addresses of vector tables**

| Software interrupt number | Vector table address<br>Address (L) to address (H) | Interrupt source | Remarks |
|---|---|---|---|
| Software interrupt number 0 | +0 to +3 (Note 1) | BRK instruction | Cannot be masked I flag |
| | | | |
| Software interrupt number 4 | +16 to +19 (Note 1) | $\overline{\text{INT3}}$ | |
| Software interrupt number 5 | +20 to +23 (Note 1) | Timer B5 | |
| Software interrupt number 6 | +24 to +27 (Note 1) | Timer B4 | |
| Software interrupt number 7 | +28 to +31 (Note 1) | Timer B3 | |
| Software interrupt number 8 | +32 to +35 (Note 1) | SI/O4/$\overline{\text{INT5}}$ (Note 2) | |
| Software interrupt number 9 | +36 to +39 (Note 1) | SI/O3/$\overline{\text{INT4}}$ (Note 2) | |
| Software interrupt number 10 | +40 to +43 (Note 1) | Bus collision detection | |
| Software interrupt number 11 | +44 to +47 (Note 1) | DMA0 | |
| Software interrupt number 12 | +48 to +51 (Note 1) | DMA1 | |
| Software interrupt number 13 | +52 to +55 (Note 1) | Key input interrupt | |
| Software interrupt number 14 | +56 to +59 (Note 1) | A-D | |
| Software interrupt number 15 | +60 to +63 (Note 1) | UART2 transmit/NACK (Note 3) | |
| Software interrupt number 16 | +64 to +67 (Note 1) | UART2 receive/ACK (Note 3) | |
| Software interrupt number 17 | +68 to +71 (Note 1) | UART0 transmit | |
| Software interrupt number 18 | +72 to +75 (Note 1) | UART0 receive | |
| Software interrupt number 19 | +76 to +79 (Note 1) | UART1 transmit | |
| Software interrupt number 20 | +80 to +83 (Note 1) | UART1 receive | |
| Software interrupt number 21 | +84 to +87 (Note 1) | Timer A0 | |
| Software interrupt number 22 | +88 to +91 (Note 1) | Timer A1 | |
| Software interrupt number 23 | +92 to +95 (Note 1) | Timer A2 | |
| Software interrupt number 24 | +96 to +99 (Note 1) | Timer A3 | |
| Software interrupt number 25 | +100 to +103 (Note 1) | Timer A4 | |
| Software interrupt number 26 | +104 to +107 (Note 1) | Timer B0 | |
| Software interrupt number 27 | +108 to +111 (Note 1) | Timer B1 | |
| Software interrupt number 28 | +112 to +115 (Note 1) | Timer B2 | |
| Software interrupt number 29 | +116 to +119 (Note 1) | $\overline{\text{INT0}}$ | |
| Software interrupt number 30 | +120 to +123 (Note 1) | $\overline{\text{INT1}}$ | |
| Software interrupt number 31 | +124 to +127 (Note 1) | $\overline{\text{INT2}}$ | |
| Software interrupt number 32<br>to<br>Software interrupt number 63 | +128 to +131 (Note 1)<br>to<br>+252 to +255 (Note 1) | Software interrupt | Cannot be masked I flag |

Note 1: Address relative to address in interrupt table register (INTB).
Note 2: It is selected by interrupt request cause bit (bit 6, 7 in address 035F$_{16}$ ).
Note 3: When IIC mode is selected, NACK and ACK interrupts are selected.

## Interrupt Control

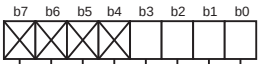
Descriptions are given here regarding how to enable or disable maskable interrupts and how to set the priority to be accepted. What is described here does not apply to non-maskable interrupts.

Enable or disable a 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 1.14.3 shows the memory map of the interrupt control registers.

Interrupt control register

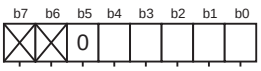
| Symbol | Address | When reset |
|---|---|---|
| TBiIC(i=3 to 5) | $0045_{16}$ to $0047_{16}$ | $XXXXX000_2$ |
| BCNIC | $004A_{16}$ | $XXXXX000_2$ |
| DMiIC(i=0, 1) | $004B_{16}$, $004C_{16}$ | $XXXXX000_2$ |
| KUPIC | $004D_{16}$ | $XXXXX000_2$ |
| ADIC | $004E_{16}$ | $XXXXX000_2$ |
| SiTIC(i=0 to 2) | $0051_{16}$, $0053_{16}$, $004F_{16}$ | $XXXXX000_2$ |
| SiRIC(i=0 to 2) | $0052_{16}$, $0054_{16}$, $0050_{16}$ | $XXXXX000_2$ |
| TAiIC(i=0 to 4) | $0055_{16}$ to $0059_{16}$ | $XXXXX000_2$ |
| TBiIC(i=0 to 2) | $005A_{16}$ to $005C_{16}$ | $XXXXX000_2$ |




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0 : Level 0 (interrupt disabled)<br>0 0 1 : Level 1<br>0 1 0 : Level 2<br>0 1 1 : Level 3<br>1 0 0 : Level 4<br>1 0 1 : Level 5<br>1 1 0 : Level 6<br>1 1 1 : Level 7 | ○ | ○ |
| ILVL1 | | | ○ | ○ |
| ILVL2 | | | ○ | ○ |
| IR | Interrupt request bit | 0 : Interrupt not requested<br>1 : Interrupt requested | ○ | ○ (Note 1) |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | – | – |

Note 1: This bit can only be accessed for reset (= 0), but cannot be accessed for set (= 1).
Note 2: To rewrite the interrupt control register, do so at a point that dose not generate the interrupt request for that register. For details, see the precautions for interrupts.

| Symbol | Address | When reset |
|---|---|---|
| INTiIC(i=3) | $0044_{16}$ | $XX00X000_2$ |
| SiIC/INTjIC (i=4, 3) (j=4, 5) | $0048_{16}$, $0049_{16}$ | $XX00X000_2$ |
| INTiIC(i=0 to 2) | $005D_{16}$ to $005F_{16}$ | $XX00X000_2$ |




| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| ILVL0 | Interrupt priority level select bit | b2 b1 b0<br>0 0 0 : Level 0 (interrupt disabled)<br>0 0 1 : Level 1<br>0 1 0 : Level 2<br>0 1 1 : Level 3<br>1 0 0 : Level 4<br>1 0 1 : Level 5<br>1 1 0 : Level 6<br>1 1 1 : Level 7 | ○ | ○ |
| ILVL1 | | | ○ | ○ |
| ILVL2 | | | ○ | ○ |
| IR | Interrupt request bit | 0: Interrupt not requested<br>1: Interrupt requested | ○ | ○ (Note 1) |
| POL | Polarity select bit | 0 : Selects falling edge<br>1 : Selects rising edge | ○ | ○ |
| Reserved bit | | Always set to “0” | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | – | – |

Note 1: This bit can only be accessed for reset (= 0), but cannot be accessed for set (= 1).
Note 2: To rewrite the interrupt control register, do so at a point that dose not generate the interrupt request for that register. For details, see the precautions for interrupts.

**Figure 1.14.3. 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 1.14.3 shows the settings of interrupt priority levels and Table 1.14.4 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 1.14.3. Settings of interrupt priority levels**

| Interrupt priority level select bit (b2 b1 b0) | Interrupt priority level | Priority order |
|---|---|---|
| 0 0 0 | Level 0 (interrupt disabled) | ——— |
| 0 0 1 | Level 1 | Low ↓ |
| 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 | High |

**Table 1.14.4. Interrupt levels enabled according to the contents of the IPL**

| IPL ($IPL_2$ $IPL_1$ $IPL_0$) | Enabled interrupt priority levels |
|---|---|
| 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) 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 1.14.4 shows the interrupt response time.

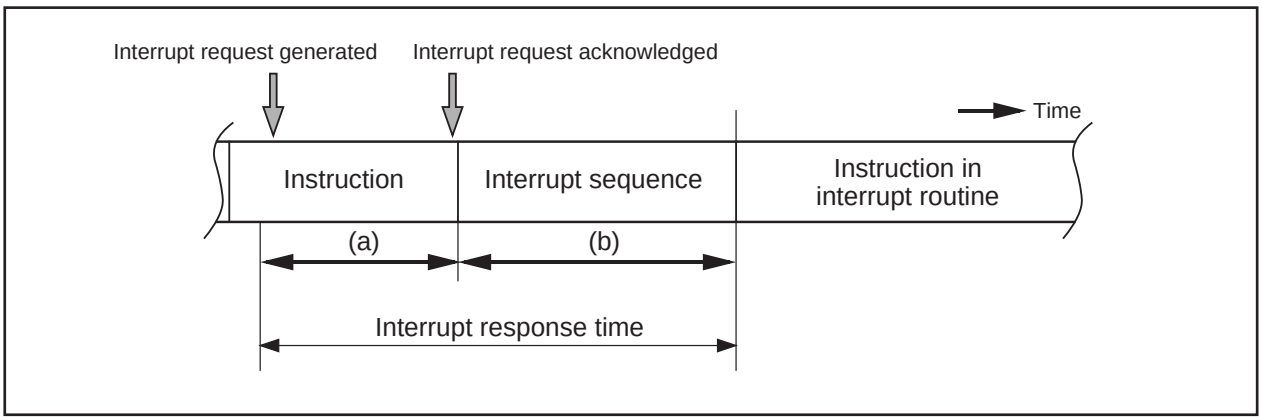



**Figure 1.14.4. Interrupt response time**

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 1.14.5.

**Table 1.14.5. 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 $\overline{DBC}$ 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.

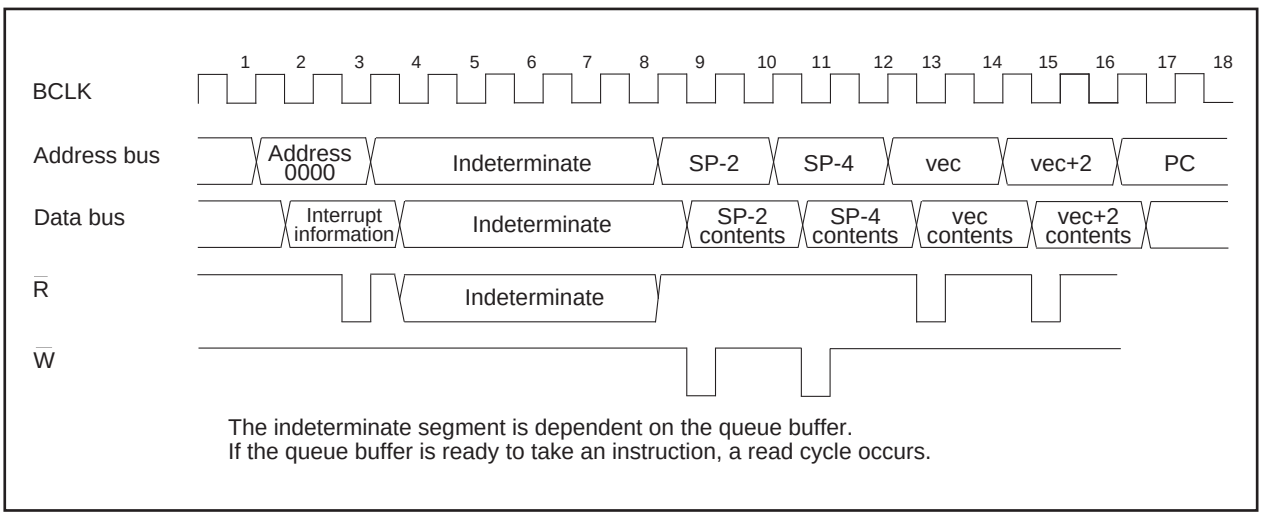



**Figure 1.14.5. Time required for executing the interrupt sequence**

## 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 1.14.6 is set in the IPL.

**Table 1.14.6. Relationship between interrupts without interrupt priority levels and IPL**

| Interrupt sources without priority levels | Value set in the IPL |
|---|---|
| Watchdog timer, $\overline{NMI}$ | 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 1.14.6 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).

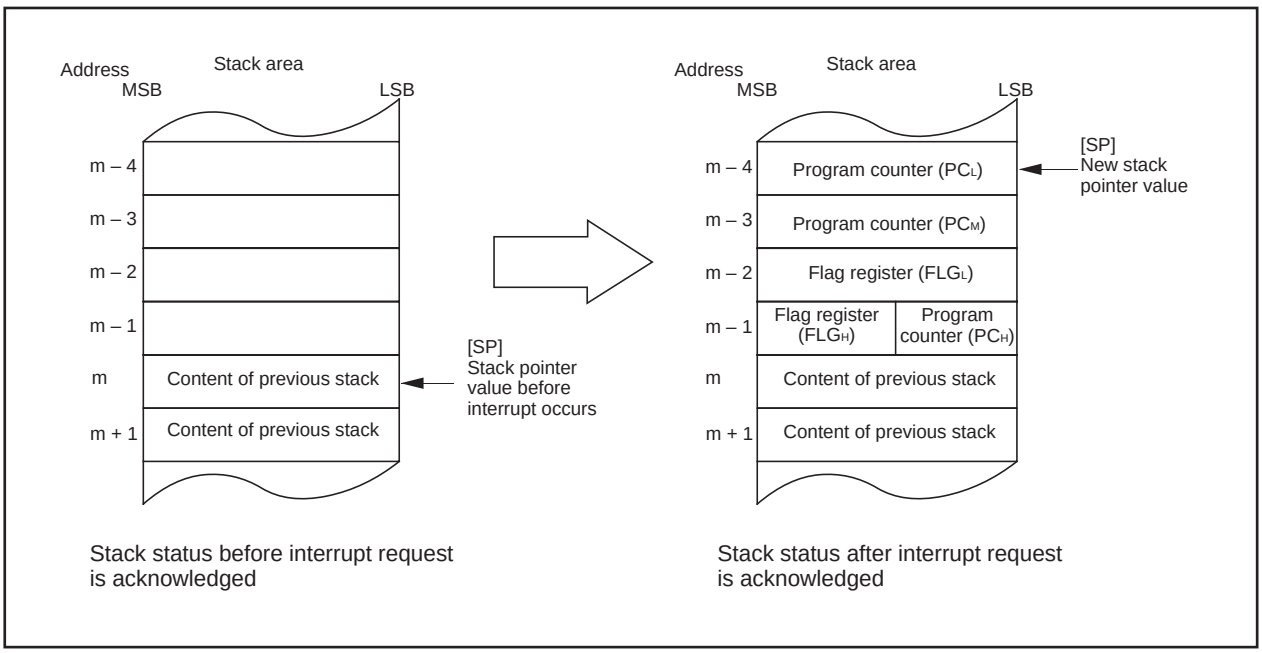
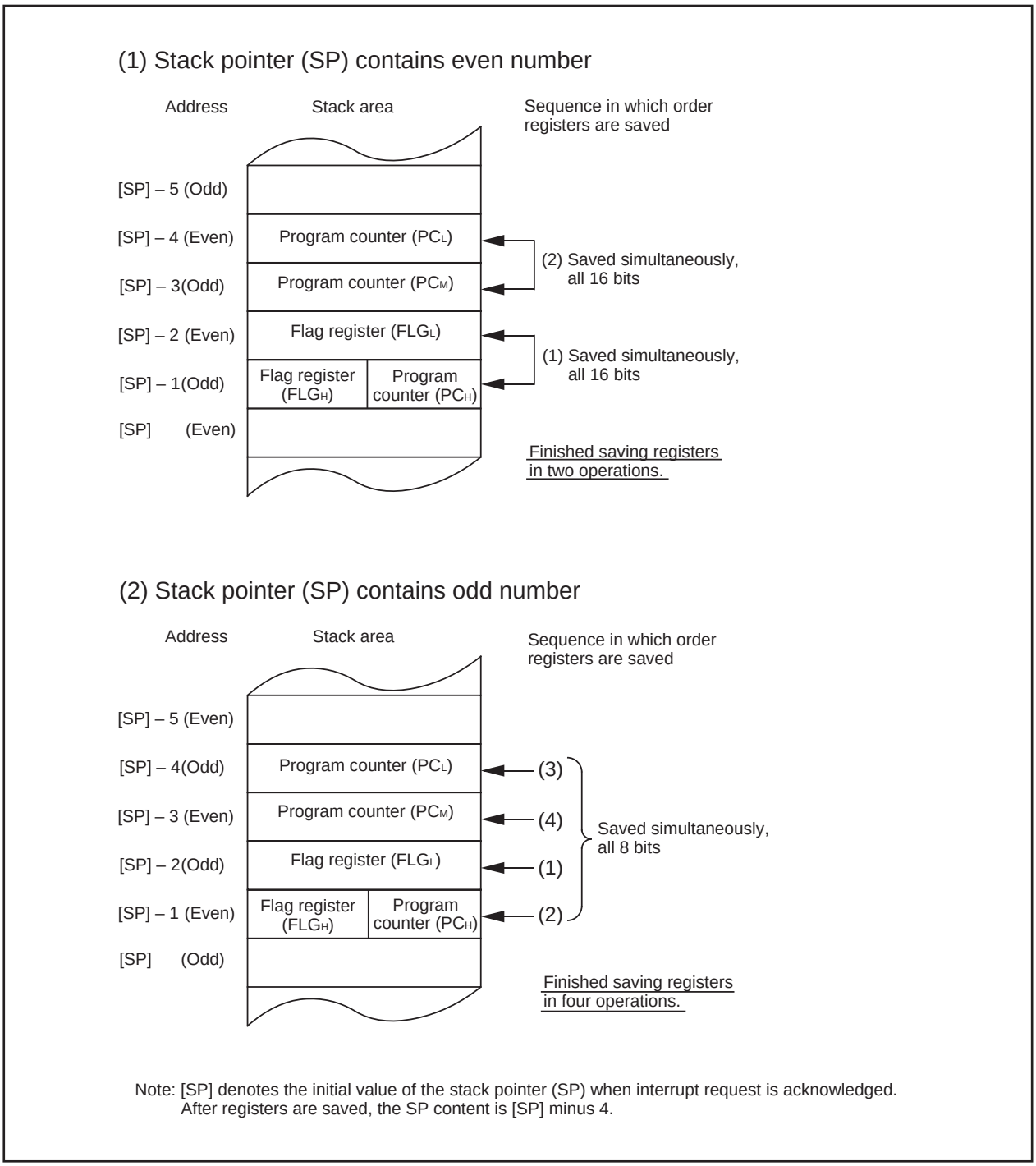



**Figure 1.14.6. State of stack before and after acceptance of interrupt request**

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 1.14.7 shows the operation of the saving registers.

Note: Stack pointer indicated by U flag.




Note: [SP] denotes the initial value of the stack pointer (SP) when interrupt request is acknowledged.
After registers are saved, the SP content is [SP] minus 4.

**Figure 1.14.7. 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 1.14.8 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 1.14.8. 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 1.14.9 shows the circuit that judges the interrupt priority level.

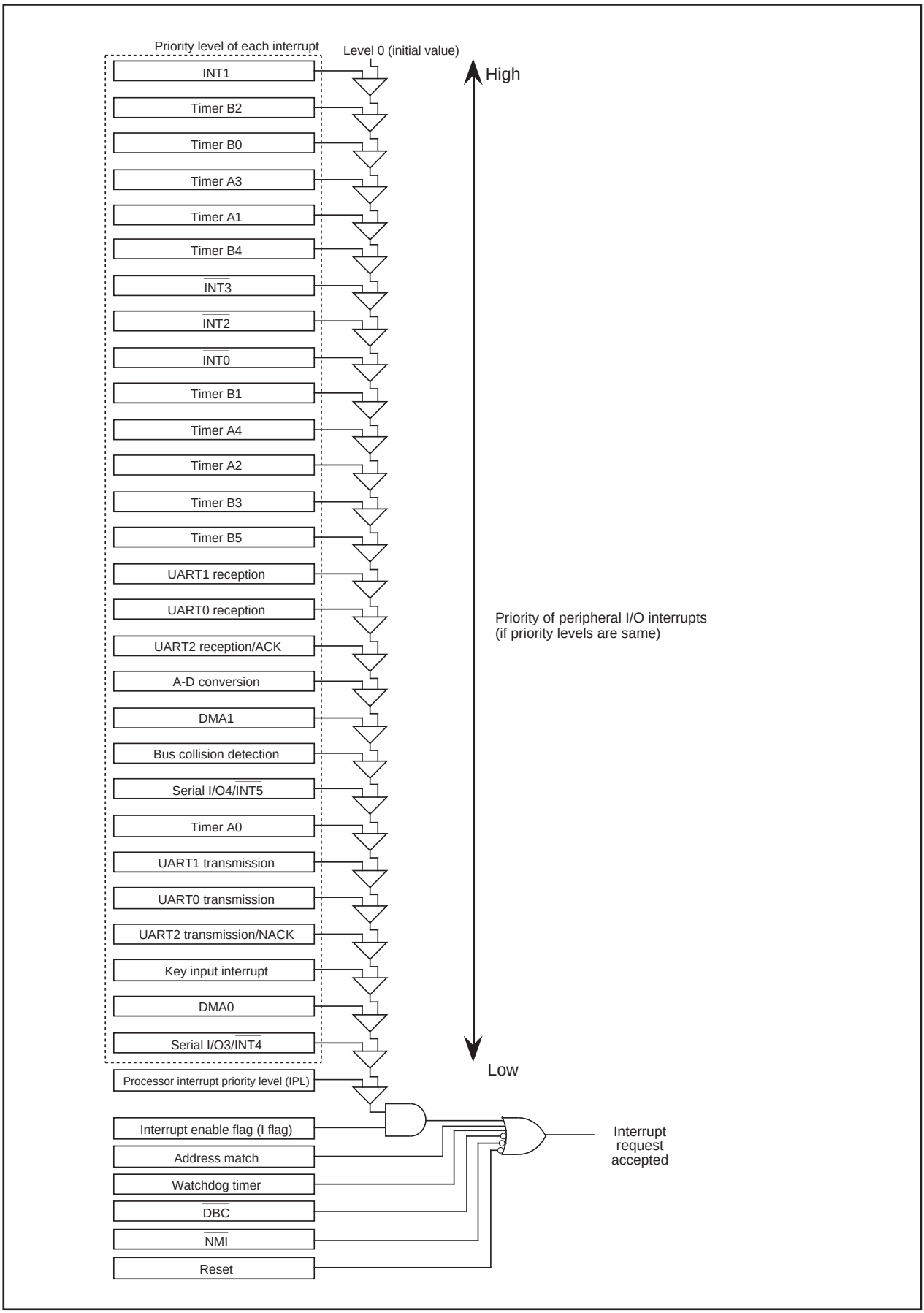



**Figure 1.14.9. Maskable interrupts priorities (peripheral I/O interrupts)**

## INT Interrupt

INT0 to INT5 are triggered by the edges of external inputs. The edge polarity is selected using the polarity select bit.

Of interrupt control registers, 0048$_{16}$ is used both as serial I/O4 and external interrupt INT5 input control register, and 0049$_{16}$ is used both as serial I/O3 and as external interrupt INT4 input control register. Use the interrupt request cause select bits - bits 6 and 7 of the interrupt request cause select register (035F$_{16}$) - to specify which interrupt request cause to select. After having set an interrupt request cause, be sure to clear the corresponding interrupt request bit before enabling an interrupt.

Either of the interrupt control registers - 0048$_{16}$, 0049$_{16}$ - has the polarity-switching bit. Be sure to set this bit to “0” to select an serial I/O as the interrupt request cause.

As for external interrupt input, an interrupt can be generated both at the rising edge and at the falling edge by setting “1” in the INTi interrupt polarity switching bit of the interrupt request cause select register (035F$_{16}$). To select both edges, set the polarity switching bit of the corresponding interrupt control register to ‘falling edge’ (“0”).

Figure 1.14.10 shows the Interrupt request cause select register.

Interrupt request cause select register


Symbol: IFSR
Address: 035F$_{16}$
When reset: 00$_{16}$

| Bit symbol | Bit name | Fumction | R | W |
|---|---|---|---|---|
| IFSR0 | INT0 interrupt polarity swiching bit | 0 : One edge<br>1 : Two edges | ○ | ○ |
| IFSR1 | INT1 interrupt polarity swiching bit | 0 : One edge<br>1 : Two edges | ○ | ○ |
| IFSR2 | INT2 interrupt polarity swiching bit | 0 : One edge<br>1 : Two edges | ○ | ○ |
| IFSR3 | INT3 interrupt polarity swiching bit | 0 : One edge<br>1 : Two edges | ○ | ○ |
| IFSR4 | INT4 interrupt polarity swiching bit | 0 : One edge<br>1 : Two edges | ○ | ○ |
| IFSR5 | INT5 interrupt polarity swiching bit | 0 : One edge<br>1 : Two edges | ○ | ○ |
| IFSR6 | Interrupt request cause select bit | 0 : SIO3<br>1 : INT4 | ○ | ○ |
| IFSR7 | Interrupt request cause select bit | 0 : SIO4<br>1 : INT5 | ○ | ○ |

**Figure 1.14.10. Interrupt request cause select register**

## NMI Interrupt

An NMI interrupt is generated when the input to the P85/NMI pin changes from “H” to “L”. The NMI interrupt is a non-maskable external interrupt. The pin level can be checked in the port P85 register (bit 5 at address 03F016).

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 cancelling 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 1.14.11 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.

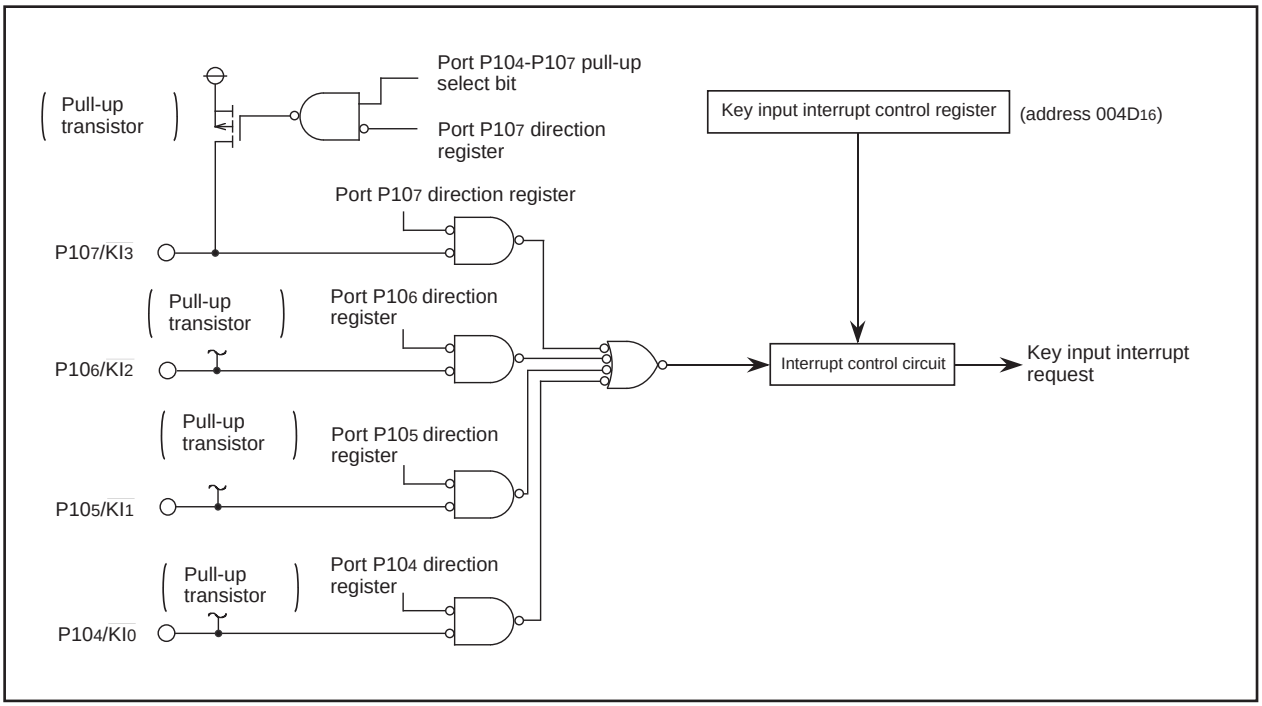



**Figure 1.14.11. Block diagram of key input interrupt**

## 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 1.14.12 shows the address match interrupt-related registers.

Address match interrupt enable register

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| AIER | $0009_{16}$ | $XXXXXX00_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| AIER0 | Address match interrupt 0 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | ○ | ○ |
| AIER1 | Address match interrupt 1 enable bit | 0 : Interrupt disabled<br>1 : Interrupt enabled | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | | — | — |

Address match interrupt register i (i = 0, 1)

(b23) b7 (b19) b3 (b16)(b15) b0 b7 (b8) b0 b7 b0

| Symbol | Address | When reset |
|---|---|---|
| RMAD0 | $0012_{16}$ to $0010_{16}$ | $X00000_{16}$ |
| RMAD1 | $0016_{16}$ to $0014_{16}$ | $X00000_{16}$ |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Address setting register for address match interrupt | $00000_{16}$ to $FFFFF_{16}$ | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be indeterminated. | | — | — |

**Figure 1.14.12. Address match interrupt-related registers**

## Precautions for Interrupts

### (1) Reading address 00000$_{16}$

- 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$_{16}$ will then be set to “0”.
  Reading address 00000$_{16}$ 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$_{16}$ by software.

### (2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to 0000$_{16}$. 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.

### (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 P8$_5$, 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 $\overline{\text{INT}}_5$ regardless of the CPU operation clock.
- When the polarity of the $\overline{\text{INT}}_0$ to $\overline{\text{INT}}_5$ pins is changed, the interrupt request bit is sometimes set to "1". After changing the polarity, set the interrupt request bit to "0". Figure 1.14.13 shows the procedure for changing the $\overline{\text{INT}}$ interrupt generate factor.

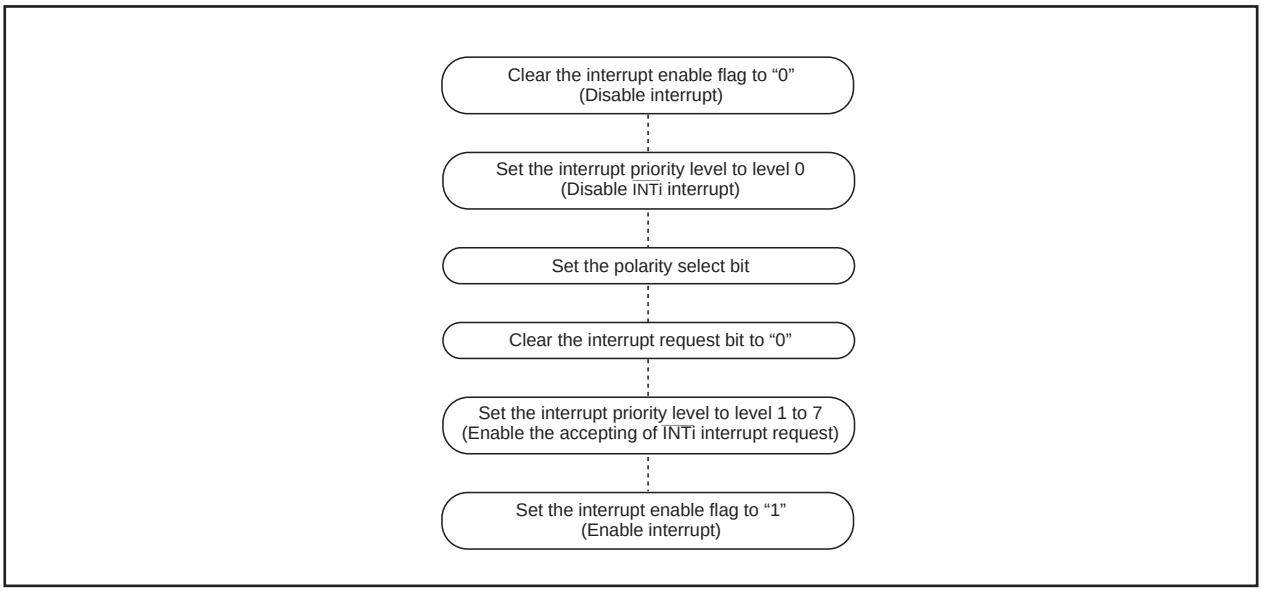



**Figure 1.14.13. Switching condition of $\overline{\text{INT}}$ interrupt request**

## (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

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 $X_{IN}$ is selected for the BCLK, bit 7 of the watchdog timer control register (address $000F_{16}$) selects the prescaler division ratio (by 16 or by 128). When $X_{CIN}$ is selected as the BCLK, the prescaler is set for division by 2 regardless of bit 7 of the watchdog timer control register (address $000F_{16}$). 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 $X_{IN}$ chosen for BCLK**

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

**With $X_{CIN}$ chosen for BCLK**

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

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 $000E_{16}$) 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 $000E_{16}$).
Figure 1.15.1 shows the block diagram of the watchdog timer. Figure 1.15.2 shows the watchdog timer-related registers.

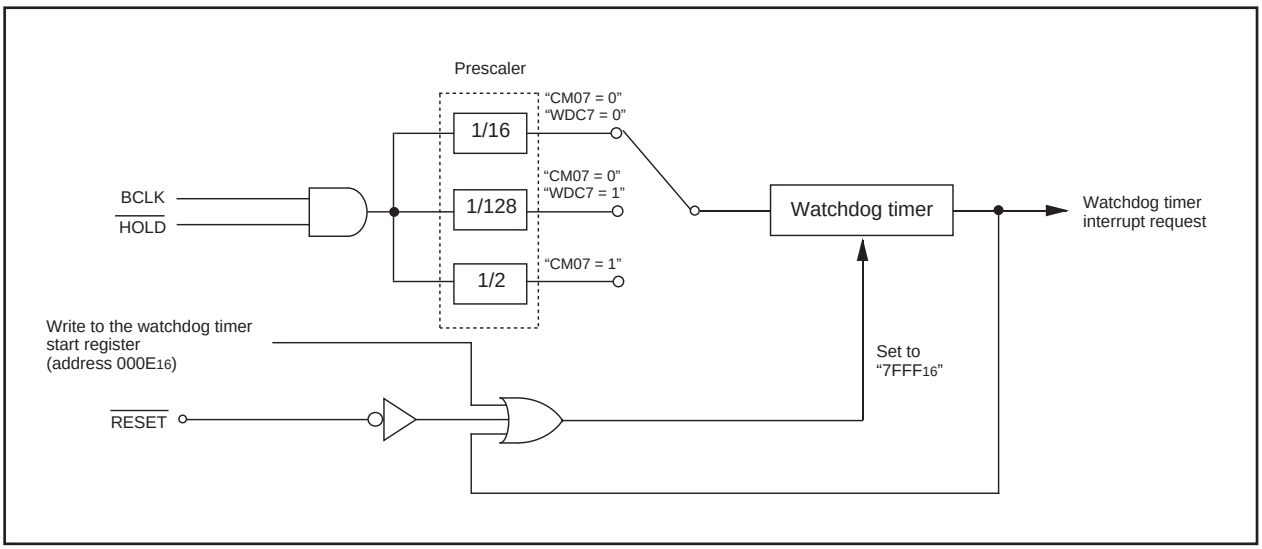



**Figure 1.15.1. Block diagram of watchdog timer**

Watchdog timer control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | 0 | 0 | | | | | |

Symbol: WDC
Address: $000F_{16}$
When reset: $000XXXXX_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| High-order bit of watchdog timer | | | ○ | × |
| Reserved bit | | Must always be set to “0” | ○ | ○ |
| Reserved bit | | Must always be set to “0” | ○ | ○ |
| WDC7 | Prescaler select bit | 0 : Divided by 16<br>1 : Divided by 128 | ○ | ○ |

Watchdog timer start register

| b7 | b0 |
|---|---|
| | |

Symbol: WDTS
Address: $000E_{16}$
When reset: Indeterminate

| Function | R | W |
|---|---|---|
| The watchdog timer is initialized and starts counting after a write instruction to this register. The watchdog timer value is always initialized to “$7FFF_{16}$” regardless of whatever value is written. | × | ○ |

**Figure 1.15.2. 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 1.16.1 shows the block diagram of the DMAC. Table 1.16.1 shows the DMAC specifications. Figures 1.16.2 to 1.16.4 show the registers used by the DMAC.

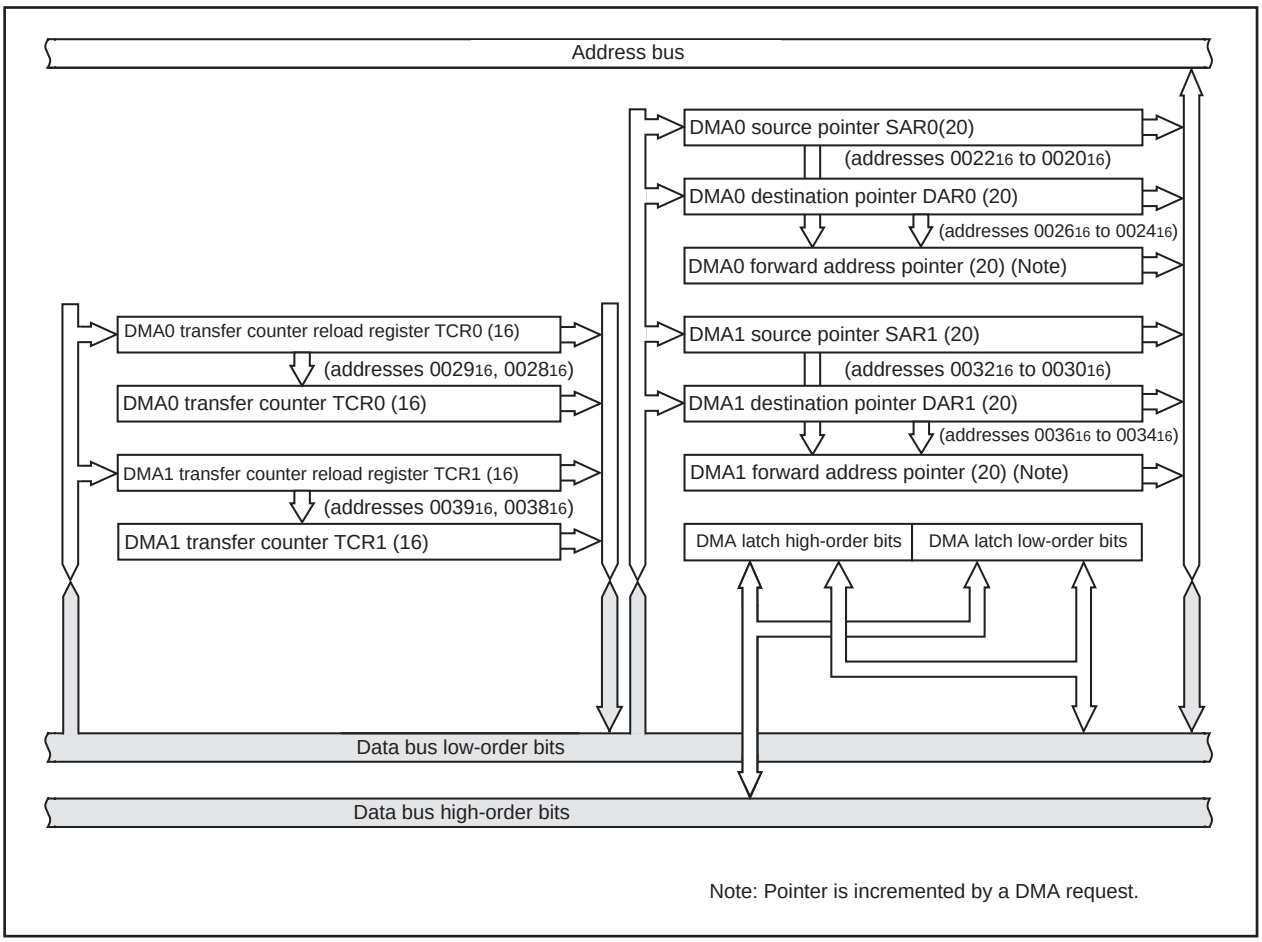



**Figure 1.16.1. Block diagram of DMAC**

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.

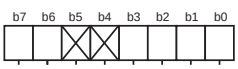
If the DMAC is active (the DMA enable bit is set to 1), data transfer starts every time a DMA transfer request signal occurs. If the cycle of the occurrences of DMA transfer request signals is higher than the DMA transfer cycle, there can be instances in which the number of transfer requests doesn't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

**Table 1.16.1. DMAC specifications**

| Item | Specification |
|---|---|
| No. of channels | 2 (cycle steal method) |
| Transfer memory space | • From any address in the 1M bytes space to a fixed address<br>• From a fixed address to any address in the 1M bytes space<br>• From a fixed address to a fixed address<br>(Note that DMA-related registers [0020$_{16}$ to 003F$_{16}$] 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 $\overline{INT0}$ or $\overline{INT1}$ ($\overline{INT0}$ can be selected by DMA0, $\overline{INT1}$ by DMA1) or both edge<br>Timer A0 to timer A4 interrupt requests<br>Timer B0 to timer B5 interrupt requests<br>UART0 transfer and reception interrupt requests<br>UART1 transfer and reception interrupt requests<br>UART2 transfer and reception interrupt requests<br>Serial I/O3, 4 interrpt requests<br>A-D conversion interrupt requests<br>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 | • Single transfer mode<br>After the transfer counter underflows, the DMA enable bit turns to "0", and the DMAC turns inactive<br>• Repeat transfer mode<br>After the transfer counter underflows, the value of the transfer counter reload register is reloaded to the transfer counter.<br>The DMAC remains active unless a "0" is written to the DMA enable bit. |
| 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.<br>When the DMAC is active, data transfer starts every time a DMA transfer request signal occurs. |
| Inactive | • When the DMA enable bit is set to "0", the DMAC is inactive.<br>• After the transfer counter underflows in single transfer mode |
| Forward address pointer and reload 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.<br>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.<br>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.

DMA0 request cause select register




Symbol: DM0SL
Address: 03B8$_{16}$
When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSEL0 | DMA request cause select bit | b3 b2 b1 b0<br>0 0 0 0 : Falling edge of INT0 pin<br>0 0 0 1 : Software trigger<br>0 0 1 0 : Timer A0<br>0 0 1 1 : Timer A1<br>0 1 0 0 : Timer A2<br>0 1 0 1 : Timer A3<br>0 1 1 0 : Timer A4 (DMS=0)<br>/two edges of INT0 pin (DMS=1)<br>0 1 1 1 : Timer B0 (DMS=0)<br>Timer B3 (DMS=1)<br>1 0 0 0 : Timer B1 (DMS=0)<br>Timer B4 (DMS=1)<br>1 0 0 1 : Timer B2 (DMS=0)<br>Timer B5 (DMS=1)<br>1 0 1 0 : UART0 transmit<br>1 0 1 1 : UART0 receive<br>1 1 0 0 : UART2 transmit<br>1 1 0 1 : UART2 receive<br>1 1 1 0 : A-D conversion<br>1 1 1 1 : UART1 transmit | ○ | ○ |
| DSEL1 | | | ○ | ○ |
| DSEL2 | | | ○ | ○ |
| DSEL3 | | | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | – | – |
| DMS | DMA request cause expansion bit | 0 : Normal<br>1 : Expanded cause | ○ | ○ |
| DSR | Software DMA request bit | If software trigger is selected, a DMA request is generated by setting this bit to “1” (When read, the value of this bit is always “0”) | ○ | ○ |

**Figure 1.16.2. DMAC register (1)**

DMA1 request cause select register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | X | X | | | | |

Symbol: DM1SL | Address: 03BA16 | When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DSEL0 | DMA request cause select bit | b3 b2 b1 b0<br>0 0 0 0 : Falling edge of INT1 pin<br>0 0 0 1 : Software trigger<br>0 0 1 0 : Timer A0<br>0 0 1 1 : Timer A1<br>0 1 0 0 : Timer A2<br>0 1 0 1 : Timer A3(DMS=0)<br>/serial I/O3 (DMS=1)<br>0 1 1 0 : Timer A4 (DMS=0)<br>/serial I/O4 (DMS=1)<br>0 1 1 1 : Timer B0 (DMS=0)<br>/two edges of INT1 (DMS=1)<br>1 0 0 0 : Timer B1<br>1 0 0 1 : Timer B2<br>1 0 1 0 : UART0 transmit<br>1 0 1 1 : UART0 receive<br>1 1 0 0 : UART2 transmit<br>1 1 0 1 : UART2 receive<br>1 1 1 0 : A-D conversion<br>1 1 1 1 : UART1 receive | ○ | ○ |
| DSEL1 | | | ○ | ○ |
| DSEL2 | | | ○ | ○ |
| DSEL3 | | | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | – | – |
| DMS | DMA request cause expansion bit | 0 : Normal<br>1 : Expanded cause | ○ | ○ |
| DSR | Software DMA request bit | If software trigger is selected, a DMA request is generated by setting this bit to “1” (When read, the value of this bit is always “0”) | ○ | ○ |

DMAi control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | X | | | | | | |

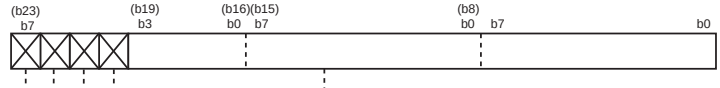
Symbol: DMiCON(i=0,1) | Address: 002C16, 003C16 | When reset: 00000X002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DMBIT | Transfer unit bit select bit | 0 : 16 bits<br>1 : 8 bits | ○ | ○ |
| DMASL | Repeat transfer mode select bit | 0 : Single transfer<br>1 : Repeat transfer | ○ | ○ |
| DMAS | DMA request bit (Note 1) | 0 : DMA not requested<br>1 : DMA requested | ○ | ○ (Note 2) |
| DMAE | DMA enable bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| DSD | Source address direction select bit (Note 3) | 0 : Fixed<br>1 : Forward | ○ | ○ |
| DAD | Destination address direction select bit (Note 3) | 0 : Fixed<br>1 : Forward | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | – | – |

Note 1: DMA request can be cleared by resetting the bit.
Note 2: This bit can only be set to “0”.
Note 3: Source address direction select bit and destination address direction select bit cannot be set to “1” simultaneously.

**Figure 1.16.3. DMAC register (2)**

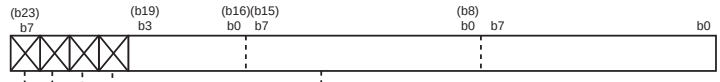
DMAi source pointer (i = 0, 1)




| Symbol | Address | When reset |
|---|---|---|
| SAR0 | $0022_{16}$ to $0020_{16}$ | Indeterminate |
| SAR1 | $0032_{16}$ to $0030_{16}$ | Indeterminate |

| Function | Transfer count specification | R | W |
|---|---|---|---|
| • Source pointer<br>Stores the source address | $00000_{16}$ to $FFFFF_{16}$ | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | – | – |

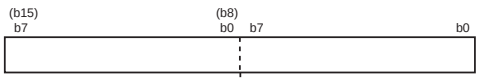
DMAi destination pointer (i = 0, 1)




| Symbol | Address | When reset |
|---|---|---|
| DAR0 | $0026_{16}$ to $0024_{16}$ | Indeterminate |
| DAR1 | $0036_{16}$ to $0034_{16}$ | Indeterminate |

| Function | Transfer count specification | R | W |
|---|---|---|---|
| • Destination pointer<br>Stores the destination address | $00000_{16}$ to $FFFFF_{16}$ | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | – | – |

DMAi transfer counter (i = 0, 1)




| Symbol | Address | When reset |
|---|---|---|
| TCR0 | $0029_{16}$, $0028_{16}$ | Indeterminate |
| TCR1 | $0039_{16}$, $0038_{16}$ | Indeterminate |

| Function | Transfer count specification | R | W |
|---|---|---|---|
| • Transfer counter<br>Set a value one less than the transfer count | $0000_{16}$ to $FFFF_{16}$ | ○ | ○ |

**Figure 1.16.4. DMAC register (3)**

## (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**

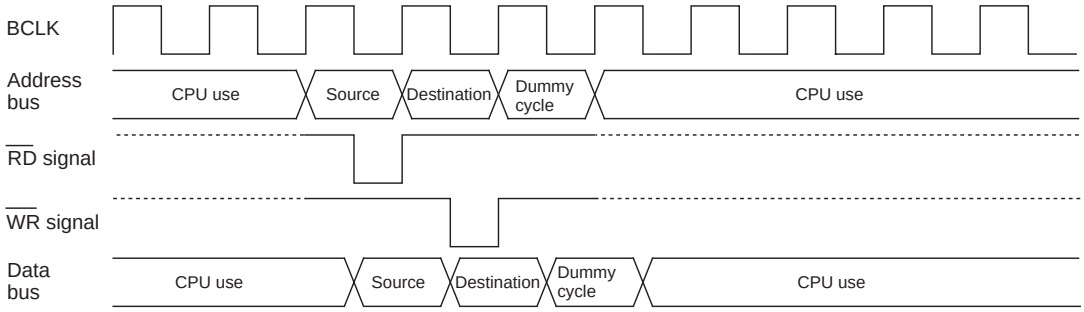
When transferring 16-bit data over an 8-bit data bus (BYTE pin = “H”) in memory expansion mode and microprocessor mode, the 16 bits of data are sent in two 8-bit blocks. Therefore, two bus cycles are required for reading the data and two are required for writing the data. Also, in contrast to when the 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**

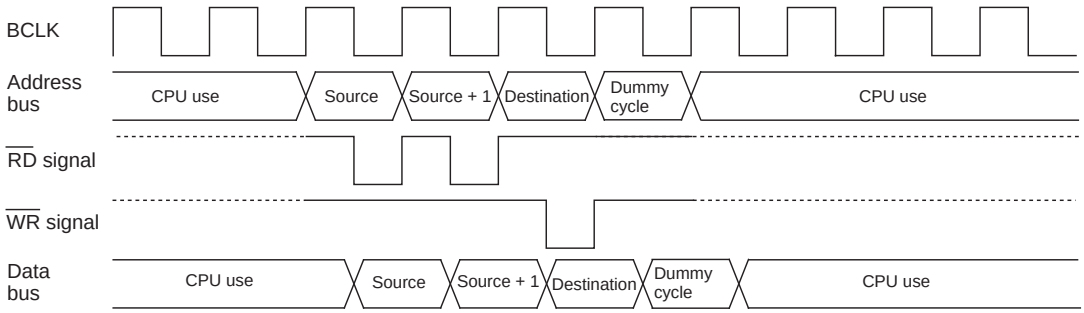
When the SFR area or a memory area with a software wait is accessed, the number of cycles is increased for the wait by 1 bus cycle. The length of the cycle is determined by BCLK.

Figure 1.16.5 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 1.16.5, 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.

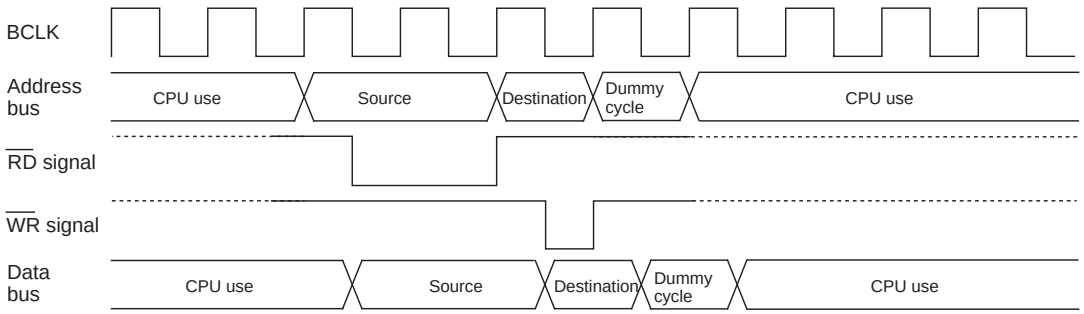
**(1) 8-bit transfers**
**16-bit transfers from even address and the source address is even.**



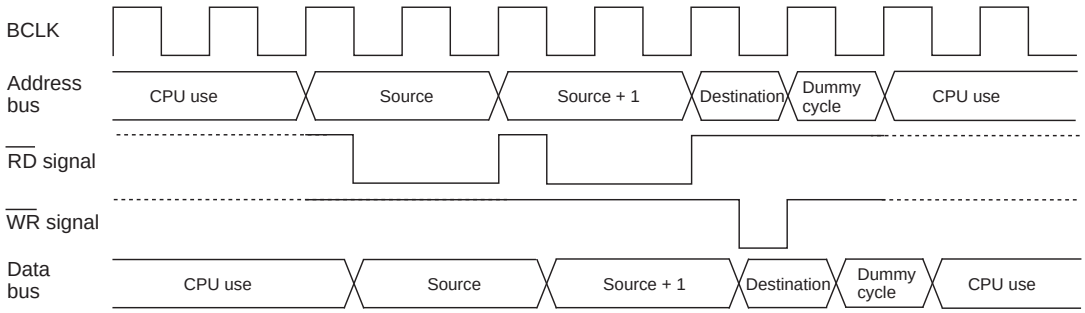

**(2) 16-bit transfers and the source address is odd**
**Transferring 16-bit data on an 8-bit data bus (In this case, there are also two destination write cycles).**




**(3) One wait is inserted into the source read under the conditions in (1)**




**(4) One wait is inserted into the source read under the conditions in (2)**
**(When 16-bit data is transferred on an 8-bit data bus, there are two destination write cycles).**




Note: The same timing changes occur with the respective conditions at the destination as at the source.

**Figure 1.16.5. 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 1.16.2 shows the number of DMAC transfer cycles.

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

No. of transfer cycles per transfer unit = No. of read cycles x j + No. of write cycles x k

**Table 1.16.2. 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 DMAi 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 INTi pin (i depends on which DMAC channel is used).

Selecting the INTi 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 INTi 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 1.16.6 An example of DMA transfer effected by external factors.

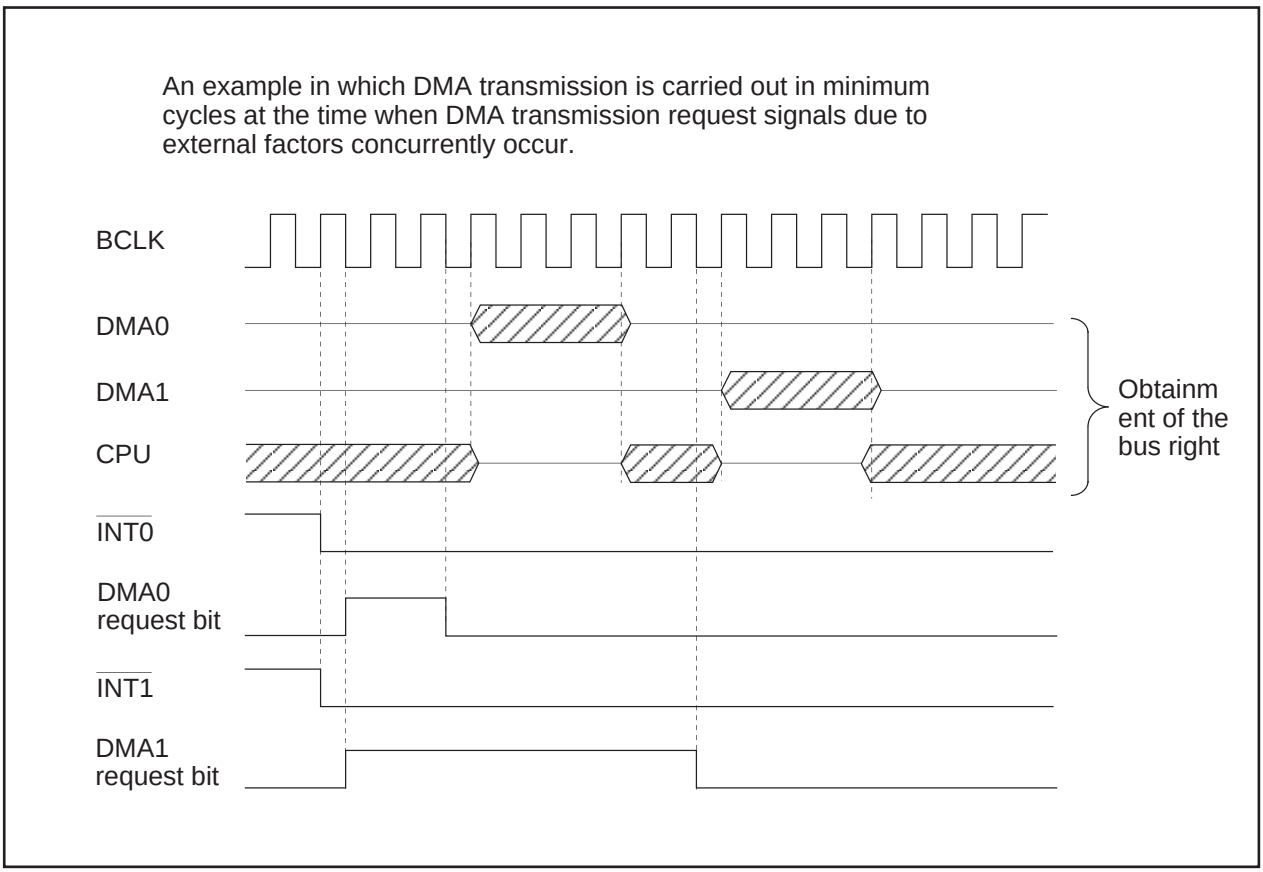
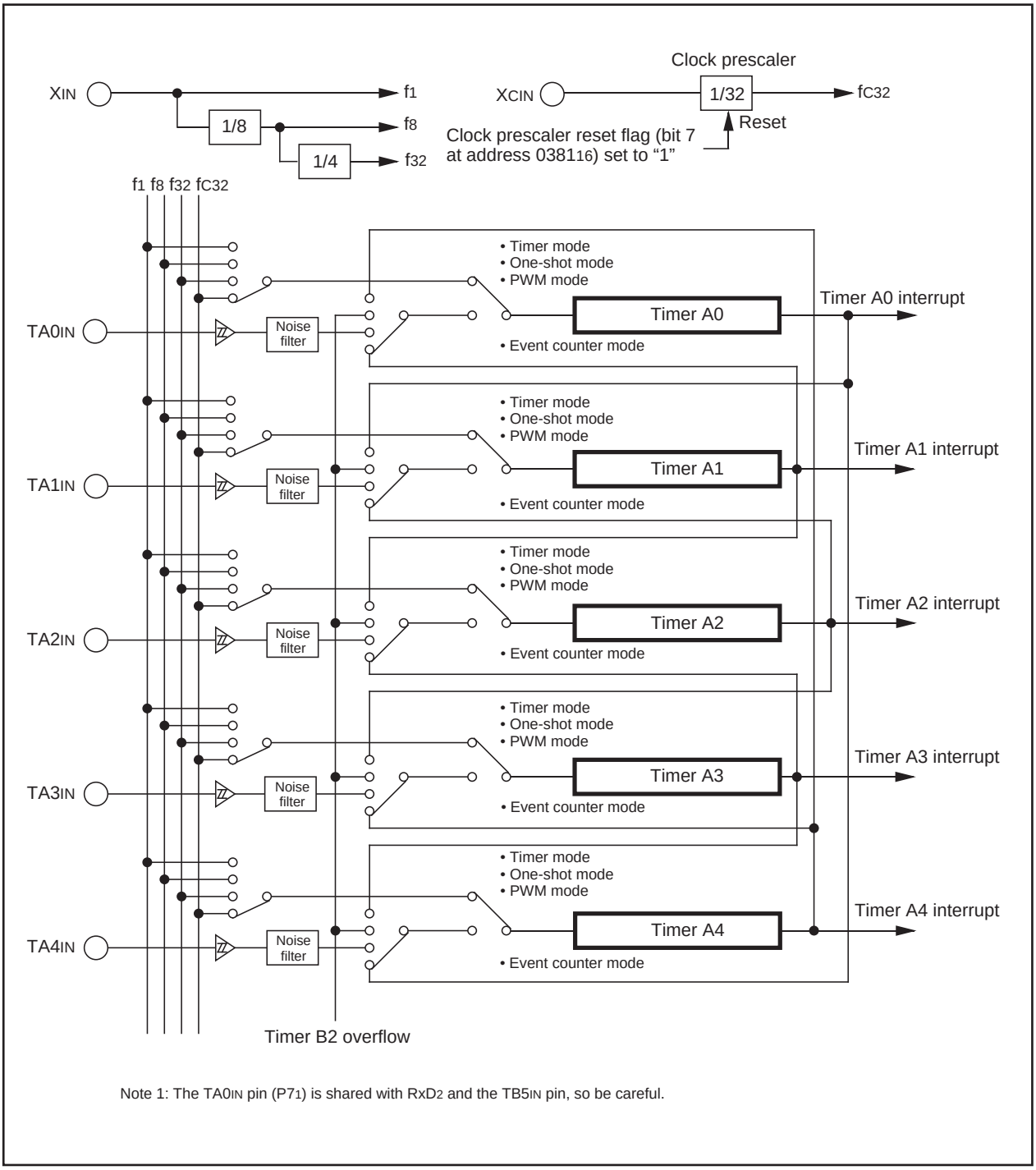



**Figure 1.16.6. An example of DMA transfer effected by external factors**

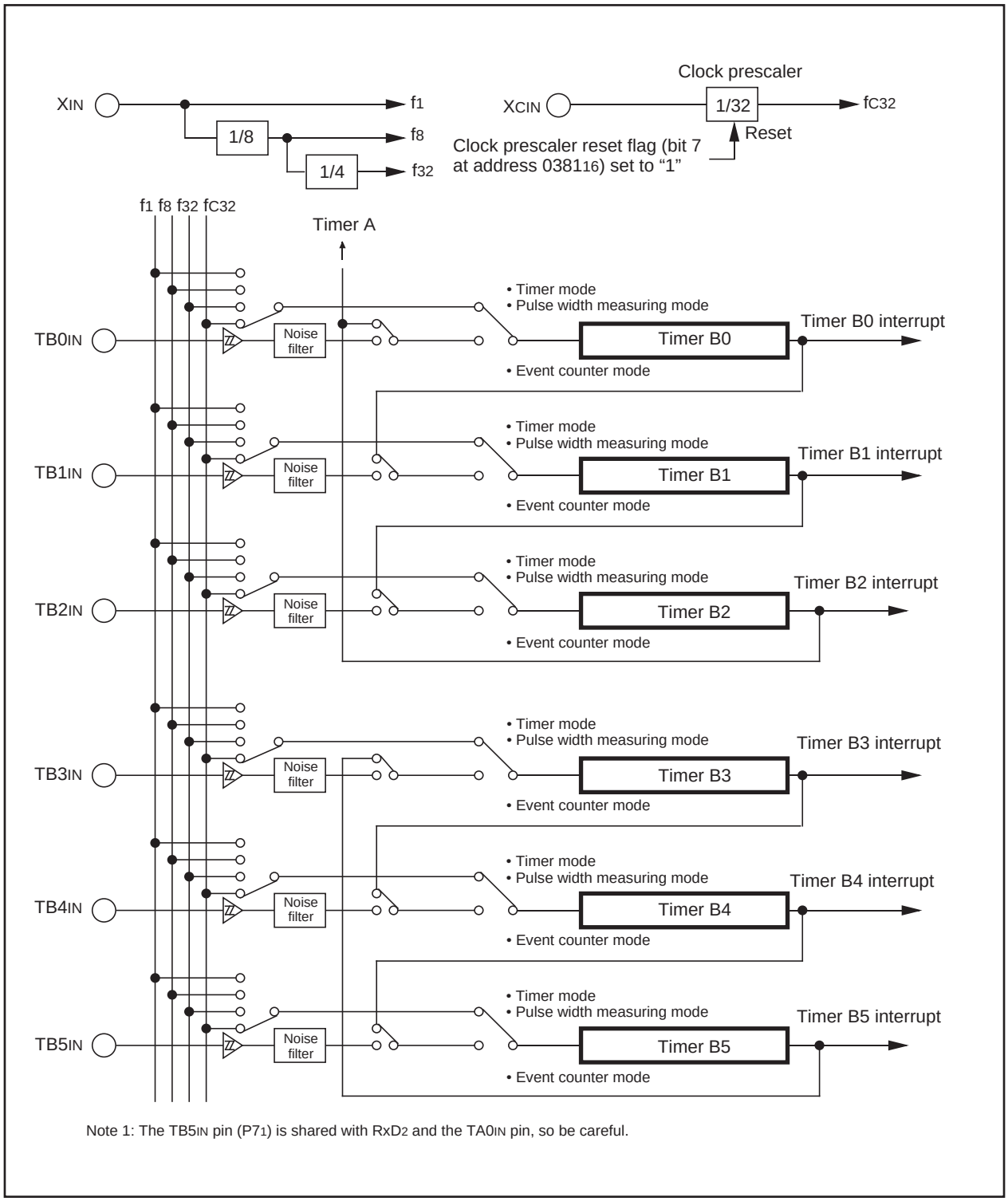
## Timer

There are eleven 16-bit timers. These timers can be classified by function into timers A (five) and timers B (six). All these timers function independently. Figures 1.17.1 and 1.17.2 show the block diagram of timers.




Note 1: The TA0IN pin (P71) is shared with RxD2 and the TB5IN pin, so be careful.

**Figure 1.17.1. Timer A block diagram**




Note 1: The TB5IN pin (P71) is shared with RxD2 and the TA0IN pin, so be careful.

**Figure 1.17.2. Timer B block diagram**

## Timer A

Figure 1.17.3 shows the block diagram of timer A. Figures 1.17.4 to 1.17.6 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 "$0000_{16}$".
- Pulse width modulation (PWM) mode: The timer outputs pulses of a given width.

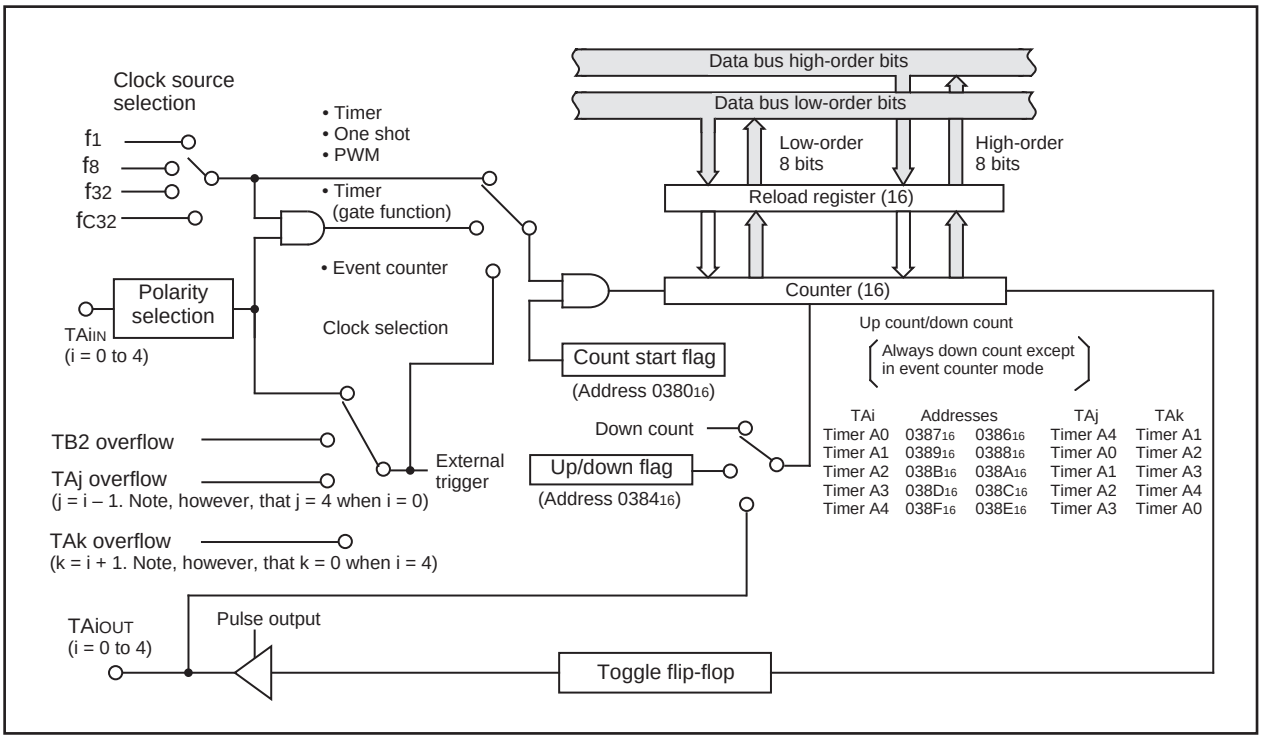



**Figure 1.17.3. Block diagram of timer A**

Timer Ai mode register

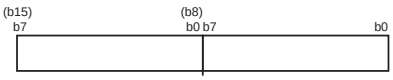
| Symbol | Address | When reset |
|---|---|---|
| TAiMR(i=0 to 4) | $0396_{16}$ to $039A_{16}$ | $00_{16}$ |

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode<br>0 1 : Event counter mode<br>1 0 : One-shot timer mode<br>1 1 : Pulse width modulation (PWM) mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Function varies with each operation mode | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | | | ○ | ○ |
| MR3 | | | ○ | ○ |
| TCK0 | Count source select bit (Function varies with each operation mode) | | ○ | ○ |
| TCK1 | | | ○ | ○ |

**Figure 1.17.4. Timer A-related registers (1)**

Timer Ai register (Note)

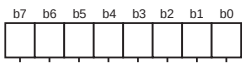



| Symbol | Address | When reset |
|---|---|---|
| TA0 | 0387$_{16}$,0386$_{16}$ | Indeterminate |
| TA1 | 0389$_{16}$,0388$_{16}$ | Indeterminate |
| TA2 | 038B$_{16}$,038A$_{16}$ | Indeterminate |
| TA3 | 038D$_{16}$,038C$_{16}$ | Indeterminate |
| TA4 | 038F$_{16}$,038E$_{16}$ | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| • Timer mode<br>Counts an internal count source | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |
| • Event counter mode<br>Counts pulses from an external source or timer overflow | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |
| • One-shot timer mode<br>Counts a one shot width | 0000$_{16}$ to FFFF$_{16}$ | × | ○ |
| • Pulse width modulation mode (16-bit PWM)<br>Functions as a 16-bit pulse width modulator | 0000$_{16}$ to FFFE$_{16}$ | × | ○ |
| • Pulse width modulation mode (8-bit PWM)<br>Timer low-order address functions as an 8-bit prescaler and high-order address functions as an 8-bit pulse width modulator | 00$_{16}$ to FE$_{16}$<br>(Both high-order and low-order addresses) | × | ○ |

Note: Read and write data in 16-bit units.

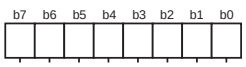
Count start flag




| Symbol | Address | When reset |
|---|---|---|
| TABSR | 0380$_{16}$ | 00$_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0S | Timer A0 count start flag | 0 : Stops counting<br>1 : Starts counting | ○ | ○ |
| TA1S | Timer A1 count start flag | | ○ | ○ |
| TA2S | Timer A2 count start flag | | ○ | ○ |
| TA3S | Timer A3 count start flag | | ○ | ○ |
| TA4S | Timer A4 count start flag | | ○ | ○ |
| TB0S | Timer B0 count start flag | | ○ | ○ |
| TB1S | Timer B1 count start flag | | ○ | ○ |
| TB2S | Timer B2 count start flag | | ○ | ○ |

Up/down flag




| Symbol | Address | When reset |
|---|---|---|
| UDF | 0384$_{16}$ | 00$_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0UD | Timer A0 up/down flag | 0 : Down count<br>1 : Up count<br><br>This specification becomes valid when the up/down flag content is selected for up/down switching cause | ○ | ○ |
| TA1UD | Timer A1 up/down flag | | ○ | ○ |
| TA2UD | Timer A2 up/down flag | | ○ | ○ |
| TA3UD | Timer A3 up/down flag | | ○ | ○ |
| TA4UD | Timer A4 up/down flag | | ○ | ○ |
| TA2P | Timer A2 two-phase pulse signal processing select bit | 0 : two-phase pulse signal processing disabled<br>1 : two-phase pulse signal processing enabled<br><br>When not using the two-phase pulse signal processing function, set the select bit to “0” | × | ○ |
| TA3P | Timer A3 two-phase pulse signal processing select bit | | × | ○ |
| TA4P | Timer A4 two-phase pulse signal processing select bit | | × | ○ |

**Figure 1.17.5. Timer A-related registers (2)**

### One-shot start flag

| Symbol | Address | When reset |
|---|---|---|
| ONSF | $0382_{16}$ | $00X00000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0OS | Timer A0 one-shot start flag | 1 : Timer start<br>When read, the value is "0" | ○ | ○ |
| TA1OS | Timer A1 one-shot start flag | | ○ | ○ |
| TA2OS | Timer A2 one-shot start flag | | ○ | ○ |
| TA3OS | Timer A3 one-shot start flag | | ○ | ○ |
| TA4OS | Timer A4 one-shot start flag | | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate. | | | — | — |
| TA0TGL | Timer A0 event/trigger select bit | b7 b6<br>0 0 : Input on $TA0_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA4 overflow is selected<br>1 1 : TA1 overflow is selected | ○ | ○ |
| TA0TGH | | | ○ | ○ |

Note: Set the corresponding port direction register to "0".

### Trigger select register

| Symbol | Address | When reset |
|---|---|---|
| TRGSR | $0383_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA1TGL | Timer A1 event/trigger select bit | b1 b0<br>0 0 : Input on $TA1_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA0 overflow is selected<br>1 1 : TA2 overflow is selected | ○ | ○ |
| TA1TGH | | | ○ | ○ |
| TA2TGL | Timer A2 event/trigger select bit | b3 b2<br>0 0 : Input on $TA2_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA1 overflow is selected<br>1 1 : TA3 overflow is selected | ○ | ○ |
| TA2TGH | | | ○ | ○ |
| TA3TGL | Timer A3 event/trigger select bit | b5 b4<br>0 0 : Input on $TA3_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA2 overflow is selected<br>1 1 : TA4 overflow is selected | ○ | ○ |
| TA3TGH | | | ○ | ○ |
| TA4TGL | Timer A4 event/trigger select bit | b7 b6<br>0 0 : Input on $TA4_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA3 overflow is selected<br>1 1 : TA0 overflow is selected | ○ | ○ |
| TA4TGH | | | ○ | ○ |

Note: Set the corresponding port direction register to "0".

### Clock prescaler reset flag

| Symbol | Address | When reset |
|---|---|---|
| CPSRF | $0381_{16}$ | $0XXXXXXX_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>In an attempt to write to these bits, write "0". The value, if read, turns out to be indeterminate. | | | — | — |
| CPSR | Clock prescaler reset flag | 0 : No effect<br>1 : Prescaler is reset<br>(When read, the value is "0") | ○ | ○ |

**Figure 1.17.6. Timer A-related registers (3)**

## (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.17.1.) Figure 1.17.7 shows the timer Ai mode register in timer mode.

**Table 1.17.1. Specifications of timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Down count<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| 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 | • When counting stopped<br>When a value is written to timer Ai register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |
| Select function | • Gate function<br>Counting can be started and stopped by the TAiIN pin's input signal<br>• Pulse output function<br>Each time the timer underflows, the TAiOUT pin's polarity is reversed |

Timer Ai mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 0 | 0 |

Symbol: TAiMR(i=0 to 4) Address: $039_{16}$ to $039A_{16}$ When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin is a normal port pin)<br>1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin) | ○ | ○ |
| MR1 | Gate function select bit | b4 b3<br>0 X (Note 2): Gate function not available (TAiIN pin is a normal port pin)<br>1 0 : Timer counts only when TAiIN pin is held "L" (Note 3)<br>1 1 : Timer counts only when TAiIN pin is held "H" (Note 3) | ○ | ○ |
| MR2 | | | ○ | ○ |
| MR3 | 0 (Must always be fixed to "0" in timer mode) | | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: The bit can be "0" or "1".
Note 3: Set the corresponding port direction register to "0".

**Figure 1.17.7. Timer Ai mode register in timer mode**

## (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 1.17.2 lists timer specifications when counting a single-phase external signal. Figure 1.17.8 shows the timer Ai mode register in event counter mode.

Table 1.17.3 lists timer specifications when counting a two-phase external signal. Figure 1.17.9 shows the timer Ai mode register in event counter mode.

**Table 1.17.2. Timer specifications in event counter mode (when not processing two-phase pulse signal)**

| Item | Specification |
|---|---|
| Count source | • External signals input to TAiIN pin (effective edge can be selected by software)<br>• TB2 overflow, TAj overflow |
| Count operation | • Up count or down count can be selected by external signal or software<br>• When the timer overflows or underflows, it reloads the reload register contents before continuing counting (Note) |
| Divide ratio | 1/ (FFFF16 - n + 1) for up count<br>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 | • When counting stopped<br>When a value is written to timer Ai register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |
| Select function | • Free-run count function<br>Even when the timer overflows or underflows, the reload register content is not reloaded to it<br>• Pulse output function<br>Each time the timer overflows or underflows, the TAiOUT pin's polarity is reversed |

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

Timer Ai mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 0 | 1 |

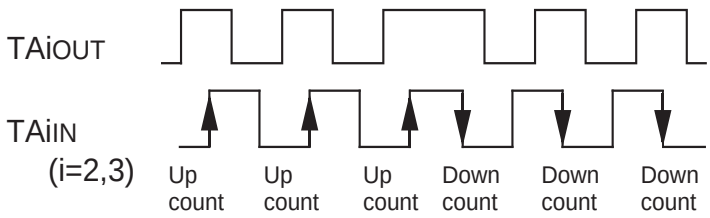
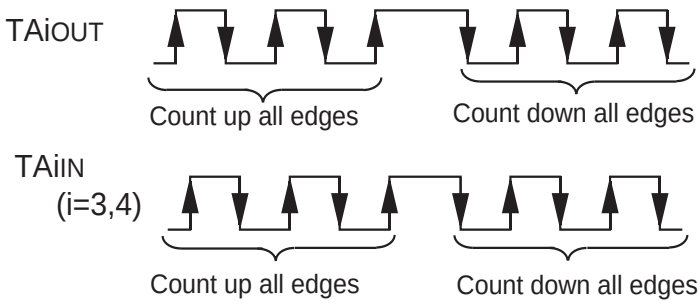
Symbol: TAiMR(i = 0, 1) Address: 039616, 039716 When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode (Note 1) | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin is a normal port pin)<br>1 : Pulse is output (Note 2) (TAiOUT pin is a pulse output pin) | ○ | ○ |
| MR1 | Count polarity select bit (Note 3) | 0 : Counts external signal's falling edge<br>1 : Counts external signal's rising edge | ○ | ○ |
| MR2 | Up/down switching cause select bit | 0 : Up/down flag's content<br>1 : TAiOUT pin's input signal (Note 4) | ○ | ○ |
| MR3 | 0 (Must always be fixed to "0" in event counter mode) | | ○ | ○ |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type | ○ | ○ |
| TCK1 | Invalid in event counter mode<br>Can be "0" or "1" | | ○ | ○ |

Note 1: In event counter mode, the count source is selected by the event / trigger select bit (addresses 038216 and 038316).
Note 2: The settings of the corresponding port register and port direction register are invalid.
Note 3: Valid only when counting an external signal.
Note 4: When an "L" signal is input to the TAiOUT pin, the downcount is activated. When "H", the upcount is activated. Set the corresponding port direction register to "0".

**Figure 1.17.8. Timer Ai mode register in event counter mode**

**Table 1.17.3. Timer specifications in event counter mode (when processing two-phase pulse signal with timers A2, A3, and A4)**

| Item | Specification |
|---|---|
| Count source | • Two-phase pulse signals input to TAiIN or TAiOUT pin |
| Count operation | • Up count or down count can be selected by two-phase pulse signal<br>• When the timer overflows or underflows, the reload register content is reloaded and the timer starts over again (Note) |
| Divide ratio | 1/ ($FFFF_{16}$ - n + 1) for up count<br>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 | • When counting stopped<br>When a value is written to timer A2, A3, or A4 register, it is written to both reload register and counter<br>• When counting in progress<br>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.) |
| Select function | • Normal processing operation<br>The timer counts up rising edges or counts down falling edges on the TAiIN pin when input signal on the TAiOUT pin is "H"<br>  <br>• Multiply-by-4 processing operation<br>If the phase relationship is such that the TAiIN pin goes "H" when the input signal on the TAiOUT pin is "H", the timer counts up rising and falling edges on the TAiOUT and TAiIN pins. If the phase relationship is such that the TAiIN pin goes "L" when the input signal on the TAiOUT pin is "H", the timer counts down rising and falling edges on the TAiOUT and TAiIN pins.<br>  |

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

Timer Ai mode register
(When not using two-phase pulse signal processing)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 0 | 1 |

Symbol: TAiMR(i = 2 to 4) — Address: 039816 to 039A16 — When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 : Pulse is not output (TAiOUT pin is a normal port pin)<br>1 : Pulse is output (Note 1) (TAiOUT pin is a pulse output pin) | ○ | ○ |
| MR1 | Count polarity select bit (Note 2) | 0 : Counts external signal's falling edges<br>1 : Counts external signal's rising edges | ○ | ○ |
| MR2 | Up/down switching cause select bit | 0 : Up/down flag's content<br>1 : TAiOUT pin's input signal (Note 3) | ○ | ○ |
| MR3 | 0 : (Must always be “0” in event counter mode) | | ○ | ○ |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type | ○ | ○ |
| TCK1 | Two-phase pulse signal processing operation select bit (Note 4)(Note 5) | 0 : Normal processing operation<br>1 : Multiply-by-4 processing operation | ○ | ○ |

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: This bit is valid when only counting an external signal.
Note 3: Set the corresponding port direction register to “0”.
Note 4: This bit is valid for the timer A3 mode register.
For timer A2 and A4 mode registers, this bit can be “0 ”or “1”.
Note 5: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 038416) is set to “1”. Also, always be sure to set the event/trigger select bit (addresses 038216 and 038316) to “00”.

Timer Ai mode register
(When using two-phase pulse signal processing)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 0 | 0 | 0 | 1 |

Symbol: TAiMR(i = 2 to 4) — Address: 039816 to 039A16 — When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | 0 (Must always be “0” when using two-phase pulse signal processing) | | ○ | ○ |
| MR1 | 0 (Must always be “0” when using two-phase pulse signal processing) | | ○ | ○ |
| MR2 | 1 (Must always be “1” when using two-phase pulse signal processing) | | ○ | ○ |
| MR3 | 0 (Must always be “0” when using two-phase pulse signal processing) | | ○ | ○ |
| TCK0 | Count operation type select bit | 0 : Reload type<br>1 : Free-run type | ○ | ○ |
| TCK1 | Two-phase pulse processing operation select bit (Note 1)(Note 2) | 0 : Normal processing operation<br>1 : Multiply-by-4 processing operation | ○ | ○ |

Note 1: This bit is valid for timer A3 mode register.
For timer A2 and A4 mode registers, this bit can be “0” or “1”.
Note 2: When performing two-phase pulse signal processing, make sure the two-phase pulse signal processing operation select bit (address 038416) is set to “1”. Also, always be sure to set the event/trigger select bit (addresses 038216 and 038316) to “00”.

**Figure 1.17.9. Timer Ai mode register in event counter mode**

### (3) One-shot timer mode

In this mode, the timer operates only once. (See Table 1.17.4.) When a trigger occurs, the timer starts up and continues operating for a given period. Figure 1.17.10 shows the timer Ai mode register in one-shot timer mode.

**Table1.17.4. Timer specifications in one-shot timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • The timer counts down<br>• When the count reaches $0000_{16}$, the timer stops counting after reloading a new count<br>• If a trigger occurs when counting, the timer reloads a new count and restarts counting |
| Divide ratio | 1/n n : Set value |
| Count start condition | • An external trigger is input<br>• The timer overflows<br>• The one-shot start flag is set (= 1) |
| Count stop condition | • A new count is reloaded after the count has reached $0000_{16}$<br>• The count start flag is reset (= 0) |
| Interrupt request generation timing | The count reaches $0000_{16}$ |
| $TAi_{IN}$ pin function | Programmable I/O port or trigger input |
| $TAi_{OUT}$ 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 | • When counting stopped<br>When a value is written to timer Ai register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |

Timer Ai mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | | | | 1 | 0 |

Symbol: TAiMR(i = 0 to 4) Address: $0396_{16}$ to $039A_{16}$ When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 : Pulse is not output ($TAi_{OUT}$ pin is a normal port pin)<br>1 : Pulse is output (Note 1) ($TAi_{OUT}$ pin is a pulse output pin) | ○ | ○ |
| MR1 | External trigger select bit (Note 2) | 0 : Falling edge of $TAi_{IN}$ pin's input signal (Note 3)<br>1 : Rising edge of $TAi_{IN}$ pin's input signal (Note 3) | ○ | ○ |
| MR2 | Trigger select bit | 0 : One-shot start flag is valid<br>1 : Selected by event/trigger select register | ○ | ○ |
| MR3 | 0 (Must always be "0" in one-shot timer mode) | | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: The settings of the corresponding port register and port direction register are invalid.
Note 2: Valid only when the $TAi_{IN}$ pin is selected by the event/trigger select bit (addresses $0382_{16}$ and $0383_{16}$). If timer overflow is selected, this bit can be "1" or "0".
Note 3: Set the corresponding port direction register to "0".

**Figure 1.17.10. Timer Ai mode register in one-shot timer mode**

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

In this mode, the timer outputs pulses of a given width in succession. (See Table 1.17.5.) In this mode, the counter functions as either a 16-bit pulse width modulator or an 8-bit pulse width modulator. Figure 1.17.11 shows the timer Ai mode register in pulse width modulation mode. Figure 1.17.12 shows the example of how a 16-bit pulse width modulator operates. Figure 1.17.13 shows the example of how an 8-bit pulse width modulator operates.

**Table 1.17.5. Timer specifications in pulse width modulation mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • The timer counts down (operating as an 8-bit or a 16-bit pulse width modulator)<br>• The timer reloads a new count at a rising edge of PWM pulse and continues counting<br>• The timer is not affected by a trigger that occurs when counting |
| 16-bit PWM | • High level width n / fi n : Set value<br>• Cycle time $(2^{16}-1)$ / fi fixed |
| 8-bit PWM | • High level width n×(m+1) / fi n : values set to timer Ai register's high-order address<br>• Cycle time $(2^{8}-1)\times$ (m+1) / fi m : values set to timer Ai register's low-order address |
| Count start condition | • External trigger is input<br>• The timer overflows<br>• The count start flag is set (= 1) |
| Count stop condition | • The count start flag is reset (= 0) |
| 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 | • When counting stopped<br>When a value is written to timer Ai register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Ai register, it is written to only reload register (Transferred to counter at next reload time) |

Timer Ai mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | 1 | 1 | 1 |

Symbol: TAiMR(i=0 to 4) Address: $039_{16}$ to $039A_{16}$ When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 1 : PWM mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | 1 (Must always be "1" in PWM mode) | | ○ | ○ |
| MR1 | External trigger select bit (Note 1) | 0: Falling edge of TAiIN pin's input signal (Note 2)<br>1: Rising edge of TAiIN pin's input signal (Note 2) | ○ | ○ |
| MR2 | Trigger select bit | 0: Count start flag is valid<br>1: Selected by event/trigger select register | ○ | ○ |
| MR3 | 16/8-bit PWM mode select bit | 0: Functions as a 16-bit pulse width modulator<br>1: Functions as an 8-bit pulse width modulator | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Valid only when the TAiIN pin is selected by the event/trigger select bit (addresses $0382_{16}$ and $0383_{16}$). If timer overflow is selected, this bit can be "1" or "0".
Note 2: Set the corresponding port direction register to "0".

**Figure 1.17.11. Timer Ai mode register in pulse width modulation mode**

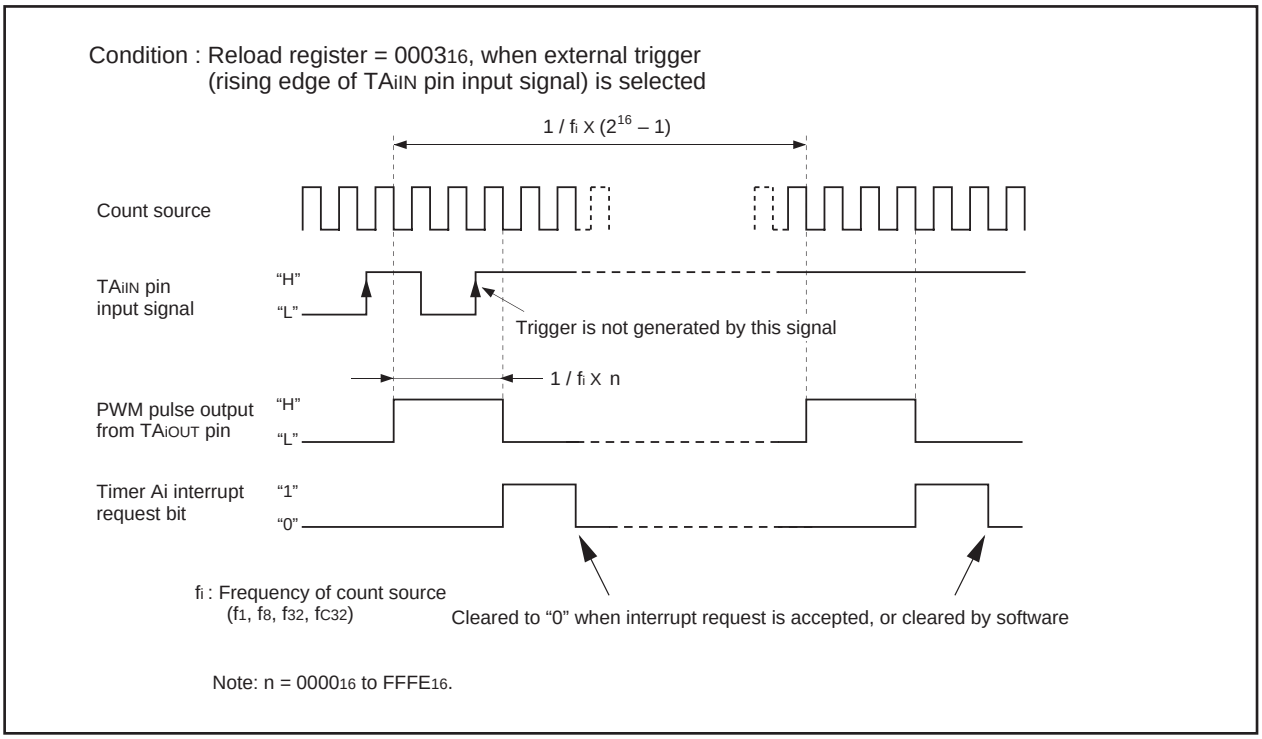



**Figure 1.17.12. Example of how a 16-bit pulse width modulator operates**

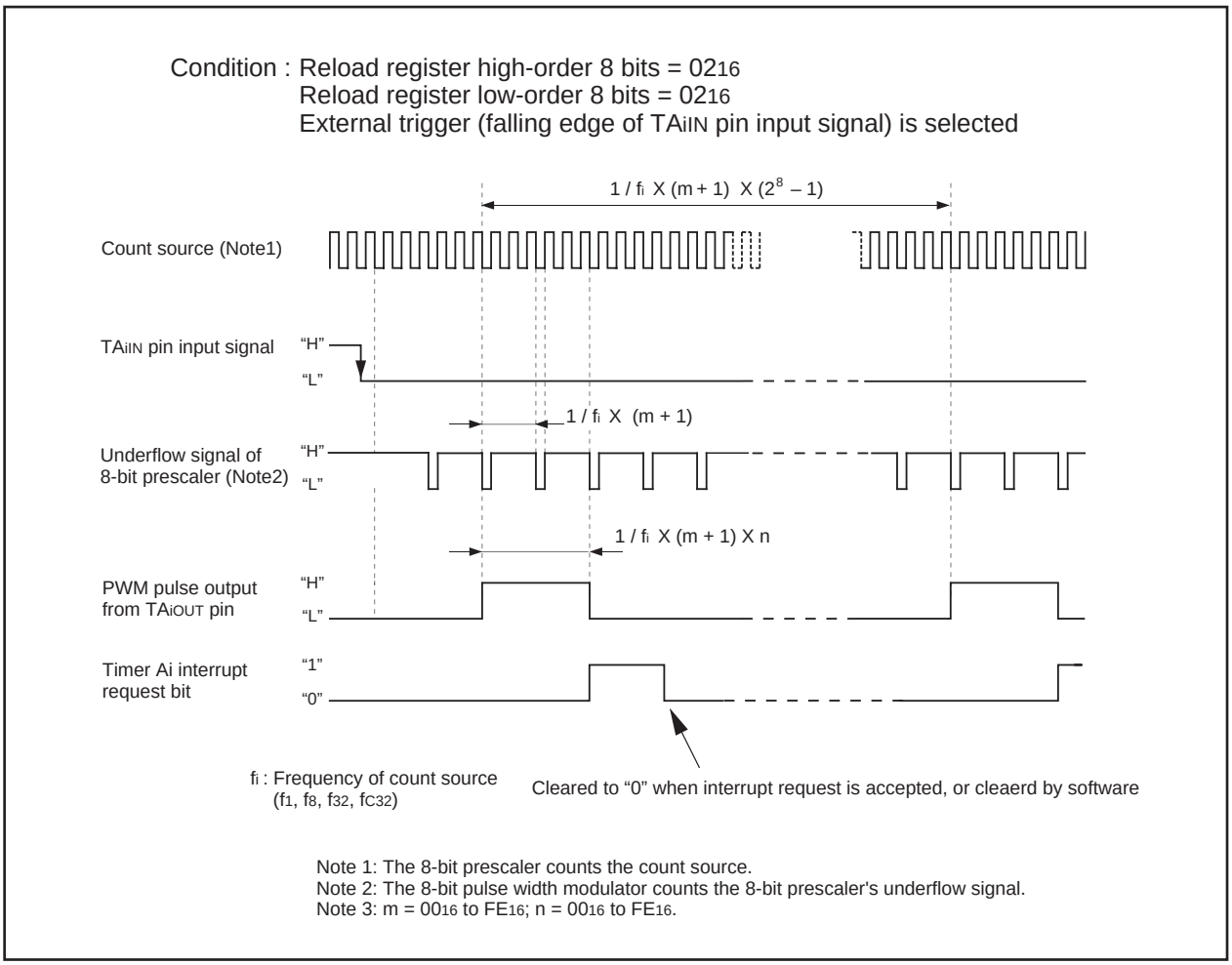



**Figure 1.17.13. Example of how an 8-bit pulse width modulator operates**

## Timer B

Figure 1.17.14 shows the block diagram of timer B. Figures 1.17.15 and 1.17.16 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.

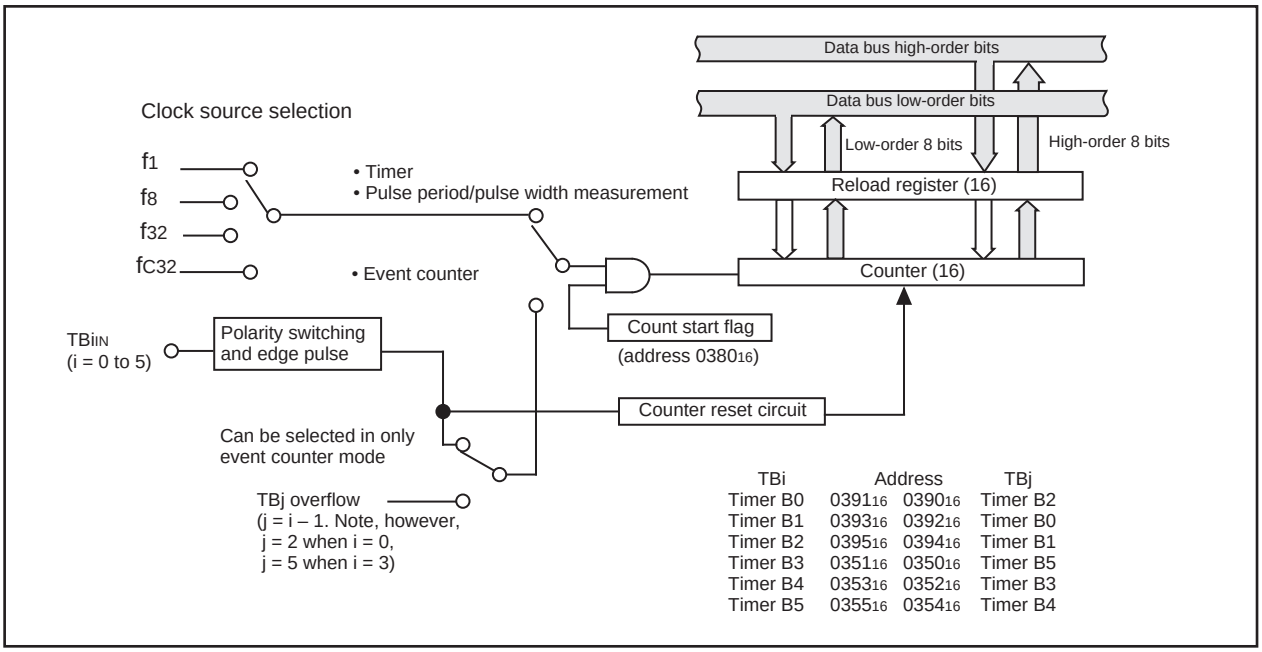



**Figure 1.17.14. Block diagram of timer B**

Timer Bi mode register

| Symbol | Address | When reset |
|---|---|---|
| TBiMR(i = 0 to 5) | 039B16 to 039D16 | 00XX00002 |
| | 035B16 to 035D16 | 00XX00002 |

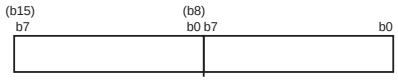
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode<br>0 1 : Event counter mode<br>1 0 : Pulse period/pulse width measurement mode<br>1 1 : Inhibited | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Function varies with each operation mode | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | | | ○ (Note 1) | ○ |
| | | | × (Note 2) | × |
| MR3 | | | ○ | × |
| TCK0 | Count source select bit (Function varies with each operation mode) | | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Timer B0, timer B3.
Note 2: Timer B1, timer B2, timer B4, timer B5.

**Figure 1.17.15. Timer B-related registers (1)**

### Timer Bi register (Note)




| Symbol | Address | When reset |
|---|---|---|
| TB0 | 039116, 039016 | Indeterminate |
| TB1 | 039316, 039216 | Indeterminate |
| TB2 | 039516, 039416 | Indeterminate |
| TB3 | 035116, 035016 | Indeterminate |
| TB4 | 035316, 035216 | Indeterminate |
| TB5 | 035516, 035416 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| • Timer mode<br>Counts the timer's period | 000016 to FFFF16 | O | O |
| • Event counter mode<br>Counts external pulses input or a timer overflow | 000016 to FFFF16 | O | O |
| • Pulse period / pulse width measurement mode<br>Measures a pulse period or width | —— | O | X |

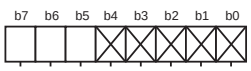
Note: Read and write data in 16-bit units.

### Count start flag


Symbol: TABSR
Address: 038016
When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0S | Timer A0 count start flag | 0 : Stops counting<br>1 : Starts counting | O | O |
| TA1S | Timer A1 count start flag | | O | O |
| TA2S | Timer A2 count start flag | | O | O |
| TA3S | Timer A3 count start flag | | O | O |
| TA4S | Timer A4 count start flag | | O | O |
| TB0S | Timer B0 count start flag | | O | O |
| TB1S | Timer B1 count start flag | | O | O |
| TB2S | Timer B2 count start flag | | O | O |

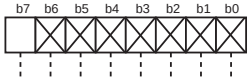
### Timer B3, 4, 5 count start flag




Symbol: TBSR
Address: 034016
When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | — | — |
| TB3S | Timer B3 count start flag | 0 : Stops counting<br>1 : Starts counting | O | O |
| TB4S | Timer B4 count start flag | | O | O |
| TB5S | Timer B5 count start flag | | O | O |

### Clock prescaler reset flag




Symbol: CPSRF
Address: 038116
When reset: 0XXXXXXX2

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | — | — |
| CPSR | Clock prescaler reset flag | 0 : No effect<br>1 : Prescaler is reset<br>(When read, the value is “0”) | O | O |

**Figure 1.17.16. Timer B-related registers (2)**

### (1) Timer mode

In this mode, the timer counts an internally generated count source. (See Table 1.17.6.) Figure 1.17.17 shows the timer Bi mode register in timer mode.

**Table 1.17.6. Timer specifications in timer mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Counts down<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| 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 |
| $TBi_{IN}$ pin function | Programmable I/O port |
| Read from timer | Count value is read out by reading timer Bi register |
| Write to timer | • When counting stopped<br>When a value is written to timer Bi register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time) |

Timer Bi mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 0 | 0 |

| Symbol | Address | When reset |
|---|---|---|
| TBiMR(i=0 to 5) | $039B_{16}$ to $039D_{16}$ | $00XX0000_2$ |
| | $035B_{16}$ to $035D_{16}$ | $00XX0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Invalid in timer mode<br>Can be “0” or “1” | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Fixed to “0” in timer mode ; i = 0, 3) | | ○ (Note 1) | ○ |
| | Nothing is assiigned (i = 1, 2, 4, 5).<br>In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate. | | × (Note 2) | × |
| MR3 | Invalid in timer mode.<br>In an attempt to write to this bit, write “0”. The value, if read in timer mode, turns out to be indeterminate. | | ○ | × |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 1: Timer B0, timer B3.
Note 2: Timer B1, timer B2, timer B4, timer B5.

**Figure 1.17.17. Timer Bi mode register in timer mode**

## (2) Event counter mode

In this mode, the timer counts an external signal or an internal timer's overflow. (See Table 1.17.7.) Figure 1.17.18 shows the timer Bi mode register in event counter mode.

**Table 1.17.7. Timer specifications in event counter mode**

| Item | Specification |
|---|---|
| Count source | • External signals input to TBiIN pin<br>• Effective edge of count source can be a rising edge, a falling edge, or falling and rising edges as selected by software |
| Count operation | • Counts down<br>• When the timer underflows, it reloads the reload register contents before continuing counting |
| 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 | • When counting stopped<br>When a value is written to timer Bi register, it is written to both reload register and counter<br>• When counting in progress<br>When a value is written to timer Bi register, it is written to only reload register (Transferred to counter at next reload time) |

Timer Bi mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 0 | 1 |

Symbol: TBiMR(i=0 to 5)
Address: 039B16 to 039D16, 035B16 to 035D16
When reset: 00XX00002, 00XX00002

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 1 : Event counter mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Count polarity select bit (Note 1) | b3 b2<br>0 0 : Counts external signal's falling edges<br>0 1 : Counts external signal's rising edges<br>1 0 : Counts external signal's falling and rising edges<br>1 1 : Inhibited | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Fixed to "0" in event counter mode; i = 0, 3) | | ○ (Note 2) | ○ |
| | Nothing is assigned (i = 1, 2, 4, 5). In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate. | | × (Note 3) | × |
| MR3 | Invalid in event counter mode. In an attempt to write to this bit, write "0". The value, if read in event counter mode, turns out to be indeterminate. | | ○ | × |
| TCK0 | Invalid in event counter mode. Can be "0" or "1". | | ○ | ○ |
| TCK1 | Event clock select | 0 : Input from TBiIN pin (Note 4)<br>1 : TBj overflow<br>(j = i – 1; however, j = 2 when i = 0, j = 5 when i = 3) | ○ | ○ |

Note 1: Valid only when input from the TBiIN pin is selected as the event clock. If timer's overflow is selected, this bit can be "0" or "1".
Note 2: Timer B0, timer B3.
Note 3: Timer B1, timer B2, timer B4, timer B5.
Note 4: Set the corresponding port direction register to "0".

**Figure 1.17.18. Timer Bi mode register in event counter mode**

## (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 1.17.8.) Figure 1.17.19 shows the timer Bi mode register in pulse period/pulse width measurement mode. Figure 1.17.20 shows the operation timing when measuring a pulse period. Figure 1.17.21 shows the operation timing when measuring a pulse width.

**Table 1.17.8. Timer specifications in pulse period/pulse width measurement mode**

| Item | Specification |
|---|---|
| Count source | $f_1$, $f_8$, $f_{32}$, $f_{C32}$ |
| Count operation | • Up count<br>• Counter value "$0000_{16}$" is transferred to reload register at measurement pulse's effective edge and the timer continues counting |
| Count start condition | Count start flag is set (= 1) |
| Count stop condition | Count start flag is reset (= 0) |
| Interrupt request generation timing | • When measurement pulse's effective edge is input (Note 1)<br>• 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.) |
| $TBi_{IN}$ 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.

Timer Bi mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | 1 | 0 |

Symbol: TBiMR(i=0 to 5) — Address: $039B_{16}$ to $039D_{16}$, $035B_{16}$ to $035D_{16}$ — When reset: $00XX0000_2$, $00XX0000_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : Pulse period / pulse width measurement mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Measurement mode select bit | b3 b2<br>0 0 : Pulse period measurement (Interval between measurement pulse's falling edge to falling edge)<br>0 1 : Pulse period measurement (Interval between measurement pulse's rising edge to rising edge)<br>1 0 : Pulse width measurement (Interval between measurement pulse's falling edge to rising edge, and between rising edge to falling edge)<br>1 1 : Inhibited | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Fixed to "0" in pulse period/pulse width measurement mode; i = 0, 3) | | ○ (Note 2) | ○ |
| | Nothing is assigned (i = 1, 2, 4, 5). In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate. | | × (Note 3) | × |
| MR3 | Timer Bi overflow flag ( Note 1) | 0 : Timer did not overflow<br>1 : Timer has overflowed | ○ | × |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $f_{C32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Note 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. This flag cannot be set to "1" by software.
Note 2: Timer B0, timer B3.
Note 3: Timer B1, timer B2, timer B4, timer B5.

**Figure 1.17.19. Timer Bi mode register in pulse period/pulse width measurement mode**

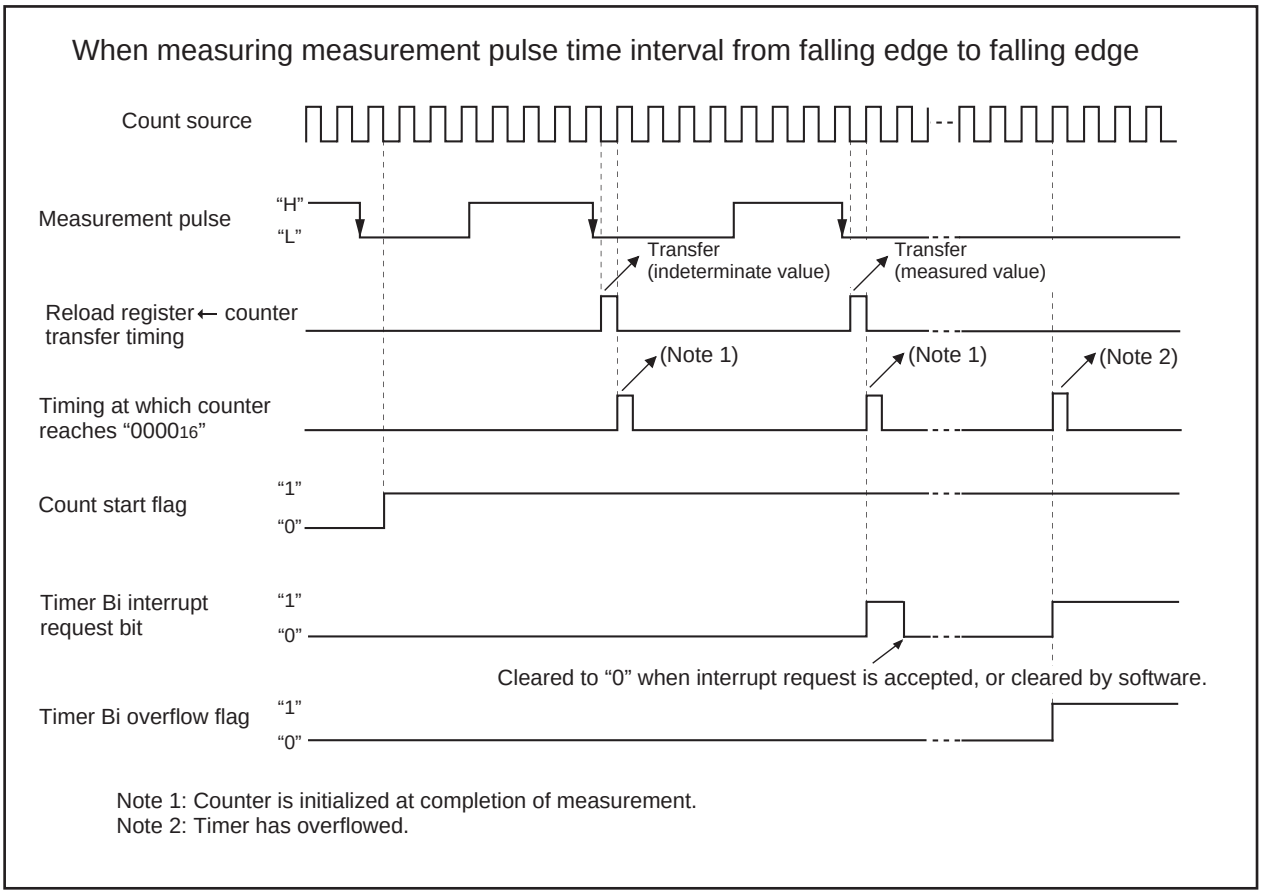



**Figure 1.17.20. Operation timing when measuring a pulse period**

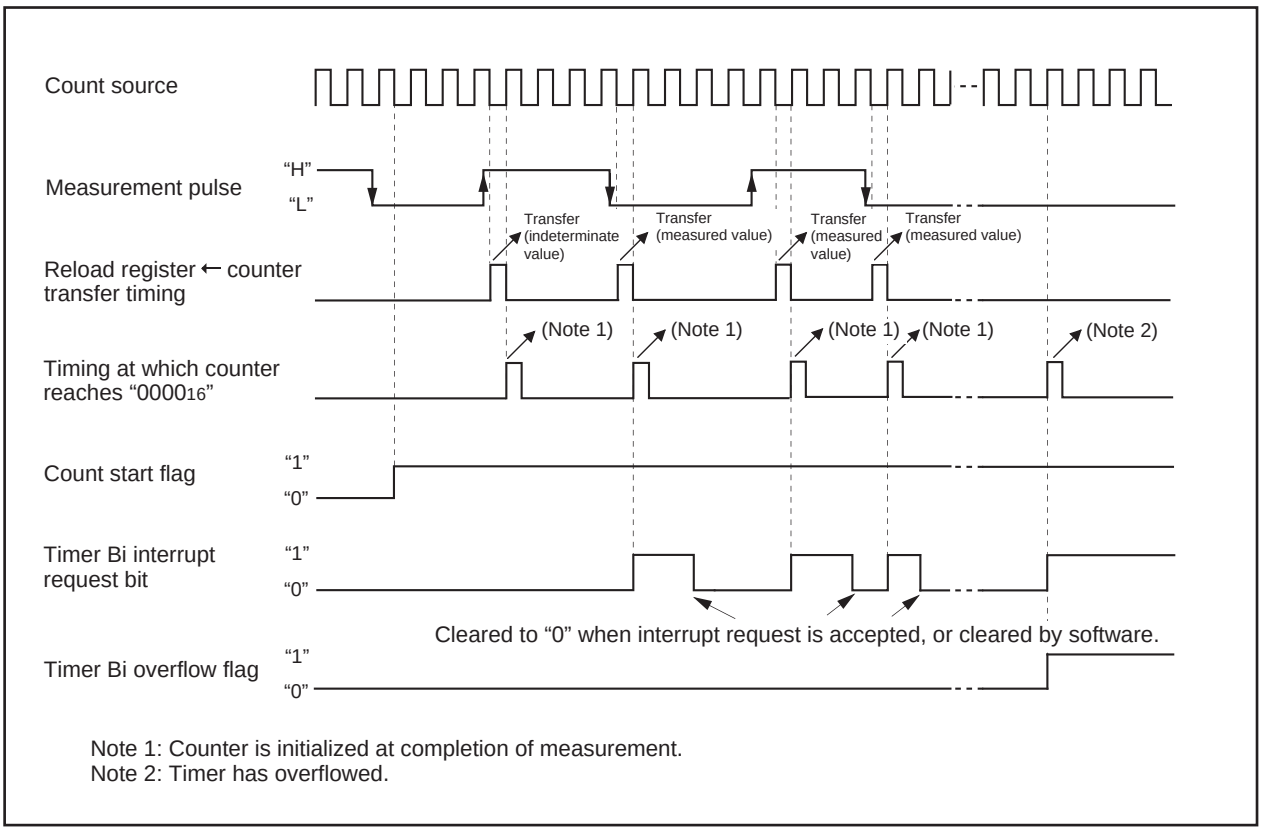



**Figure 1.17.21. Operation timing when measuring a pulse width**

## Timers' functions for three-phase motor control

Use of more than one built-in timer A and timer B provides the means of outputting three-phase motor driving waveforms.

Figures 1.18.1 to 1.18.3 show registers related to timers for three-phase motor control.

Three-phase PWM control register 0

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: INVC0 — Address: 0348$_{16}$ — When reset: 00$_{16}$

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| INV00 | Effective interrupt output polarity select bit (Note4) | 0: A timer B2 interrupt occurs when the timer A1 reload control signal is "1".<br>1: A timer B2 interrupt occurs when the timer A1 reload control signal is "0".<br>**Effective only in three-phase mode 1** | ○ | ○ |
| INV01 | Effective interrupt output specification bit (Note4) | 0: Not specified.<br>1: Selected by the effective interrupt output polarity selection bit.<br>**Effective only in three-phase mode 1** | ○ | ○ |
| INV02 | Mode select bit (Note 2) | 0: Normal mode<br>1: Three-phase PWM output mode | ○ | ○ |
| INV03 | Output control bit | 0: Output disabled<br>1: Output enabled | ○ | ○ |
| INV04 | Positive and negative phases concurrent L output disable function enable bit | 0: Feature disabled<br>1: Feature enabled | ○ | ○ |
| INV05 | Positive and negative phases concurrent L output detect flag | 0: Not detected yet<br>1: Already detected | ○ | ○ (Note 1) |
| INV06 | Modulation mode select bit (Note 3) | 0: Triangular wave modulation mode<br>1: Sawtooth wave modulation mode | ○ | ○ |
| INV07 | Software trigger bit | 1: Trigger generated<br>The value, when read, is "0". | ○ | ○ |

Note 1: No value other than "0" can be written.

Note 2: Selecting three-phase PWM output mode causes P8$_0$, P8$_1$, and P7$_2$ through P7$_5$ to output U, $\overline{U}$, V, $\overline{V}$, W, and $\overline{W}$, and works the timer for setting short circuit prevention time, the U, V, W phase output control circuits, and the circuit for setting timer B2 interrupt frequency.

Note 3: **In triangular wave modulation mode:**
The short circuit prevention timer starts in synchronization with the falling edge of timer Ai output.
The data transfer from the three-phase buffer register to the three-phase output shift register is made only once in synchronization with the transfer trigger signal after writing to the three-phase output buffer register.
**In sawtooth wave modulation mode:**
The short circuit prevention timer starts in synchronization with the falling edge of timer A output and with the transfer trigger signal.
The data transfer from the three-phase output buffer register to the three-phase output shift register is made with respect to every transfer trigger.

Note 4: To write "1" both to bit 0 (INV00) and bit 1 (INV01) of the three-phase PWM control register, set in advance the content of the timer B2 interrupt occurrences frequency set counter.

Three-phase PWM control register 1

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: INVC1 — Address: 0349$_{16}$ — When reset: 00$_{16}$

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| INV10 | Timer Ai start trigger signal select bit | 0: Timer B2 overflow signal<br>1: Timer B2 overflow signal, signal for writing to timer B2 | ○ | ○ |
| INV11 | Timer A1-1, A2-1, A4-1 control bit | 0: Three-phase mode 0<br>1: Three-phase mode 1 | ○ | ○ |
| INV12 | Short circuit timer count source select bit | 0 : Not to be used<br>1 : f$_1$/2 | ○ | ○ |
| Noting is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be "0". | | | — | — |
| Reserved bit | | Always set to "0" | ○ | ○ |
| Noting is assigned.<br>In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | | — | — |

Note 1: To use three-phase PWM output mode, write "1" to INV12.

**Figure1.18.1. Registers related to timers for three-phase motor control**

### Three-phase output buffer register 0

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| IDB0 | 034A16 | 0016 |

| Bit Symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DU0 | U phase output buffer 0 | Setting in U phase output buffer 0 | ○ | ○ |
| DUB0 | $\overline{U}$ phase output buffer 0 | Setting in $\overline{U}$ phase output buffer 0 | ○ | ○ |
| DV0 | V phase output buffer 0 | Setting in V phase output buffer 0 | ○ | ○ |
| DVB0 | $\overline{V}$ phase output buffer 0 | Setting in $\overline{V}$ phase output buffer 0 | ○ | ○ |
| DW0 | W phase output buffer 0 | Setting in W phase output buffer 0 | ○ | ○ |
| DWB0 | $\overline{W}$ phase output buffer 0 | Setting in $\overline{W}$ phase output buffer 0 | ○ | ○ |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | | — | — |

Note: When executing read instruction of this register, the contents of three-phase shift register is read out.

### Three-phase output buffer register 1

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| IDB1 | 034B16 | 0016 |

| Bit Symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DU1 | U phase output buffer 1 | Setting in U phase output buffer 1 | ○ | ○ |
| DUB1 | $\overline{U}$ phase output buffer 1 | Setting in $\overline{U}$ phase output buffer 1 | ○ | ○ |
| DV1 | V phase output buffer 1 | Setting in V phase output buffer 1 | ○ | ○ |
| DVB1 | $\overline{V}$ phase output buffer 1 | Setting in $\overline{V}$ phase output buffer 1 | ○ | ○ |
| DW1 | W phase output buffer 1 | Setting in W phase output buffer 1 | ○ | ○ |
| DWB1 | $\overline{W}$ phase output buffer 1 | Setting in $\overline{W}$ phase output buffer 1 | ○ | ○ |
| Nothing is assigned. In an attempt to write to these bits, write "0". The value, if read, turns out to be "0". | | | — | — |

Note: When executing read instruction of this register, the contents of three-phase shift register is read out.

### Dead time timer

b7 b0

| Symbol | Address | When reset |
|---|---|---|
| DOT | 034C16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Set dead time timer | 1 to 255 | — | ○ |

### Timer B2 interrupt occurrences frequency set counter

b3 b0

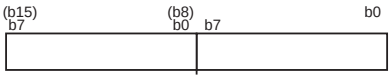
| Symbol | Address | When reset |
|---|---|---|
| ICTB2 | 034D16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Set occurrence frequency of timer B2 interrupt request | 1 to 15 | — | ○ |

Note1: In setting 1 to bit 1 (INV01) - the effective interrupt output specification bit - of three-phase PWM control register 0, do not change the B2 interrupt occurrences frequency set counter to deal with the timer function for three-phase motor control.
Note2: Do not write at the timing of an overflow occurrence in timer B2.

**Figure 1.18.2. Registers related to timers for three-phase motor control**

### Timer Ai register (Note)

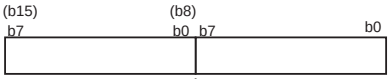



| Symbol | Address | When reset |
|---|---|---|
| TA1 | 0389$_{16}$,0388$_{16}$ | Indeterminate |
| TA2 | 038B$_{16}$,038A$_{16}$ | Indeterminate |
| TA4 | 038F$_{16}$,038E$_{16}$ | Indeterminate |
| TB2 | 0395$_{16}$,0394$_{16}$ | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| • Timer mode<br>Counts an internal count source | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |
| • One-shot timer mode<br>Counts a one shot width | 0000$_{16}$ to FFFF$_{16}$ | × | ○ |

Note: Read and write data in 16-bit units.

### Timer Ai-1 register (Note)

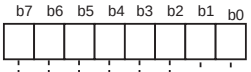



| Symbol | Address | When reset |
|---|---|---|
| TA11 | 0343$_{16}$,0342$_{16}$ | Indeterminate |
| TA21 | 0345$_{16}$,0344$_{16}$ | Indeterminate |
| TA41 | 0347$_{16}$,0346$_{16}$ | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Counts an internal count source | 0000$_{16}$ to FFFF$_{16}$ | ○ | ○ |

Note: Read and write data in 16-bit units.

### Trigger select register

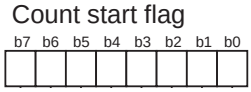



Symbol: TRGSR — Address: 0383$_{16}$ — When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA1TGL | Timer A1 event/trigger select bit | b1 b0<br>0 0 : Input on TA1$_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA0 overflow is selected<br>1 1 : TA2 overflow is selected | ○ | ○ |
| TA1TGH | | | ○ | ○ |
| TA2TGL | Timer A2 event/trigger select bit | b3 b2<br>0 0 : Input on TA2$_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA1 overflow is selected<br>1 1 : TA3 overflow is selected | ○ | ○ |
| TA2TGH | | | ○ | ○ |
| TA3TGL | Timer A3 event/trigger select bit | b5 b4<br>0 0 : Input on TA3$_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA2 overflow is selected<br>1 1 : TA4 overflow is selected | ○ | ○ |
| TA3TGH | | | ○ | ○ |
| TA4TGL | Timer A4 event/trigger select bit | b7 b6<br>0 0 : Input on TA4$_{IN}$ is selected (Note)<br>0 1 : TB2 overflow is selected<br>1 0 : TA3 overflow is selected<br>1 1 : TA0 overflow is selected | ○ | ○ |
| TA4TGH | | | ○ | ○ |

Note: Set the corresponding port direction register to "0".

### Count start flag




Symbol: TABSR — Address: 0380$_{16}$ — When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TA0S | Timer A0 count start flag | 0 : Stops counting<br>1 : Starts counting | ○ | ○ |
| TA1S | Timer A1 count start flag | | ○ | ○ |
| TA2S | Timer A2 count start flag | | ○ | ○ |
| TA3S | Timer A3 count start flag | | ○ | ○ |
| TA4S | Timer A4 count start flag | | ○ | ○ |
| TB0S | Timer B0 count start flag | | ○ | ○ |
| TB1S | Timer B1 count start flag | | ○ | ○ |
| TB2S | Timer B2 count start flag | | ○ | ○ |

**Figure 1.18.3. Registers related to timers for three-phase motor control**

## Three-phase motor driving waveform output mode (three-phase waveform mode)

Setting "1" in the mode select bit (bit 2 at $0348_{16}$) shown in Figure 1.18.1 - causes three-phase waveform mode that uses four timers A1, A2, A4, and B2 to be selected. As shown in Figure 1.18.4, set timers A1, A2, and A4 in one-shot timer mode, set the trigger in timer B2, and set timer B2 in timer mode using the respective timer mode registers.

Timer Ai mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 0 | 1 | | | 1 | 0 |

| Symbol | Address | When reset |
|---|---|---|
| TA1MR | $0397_{16}$ | $00_{16}$ |
| TA2MR | $0398_{16}$ | $00_{16}$ |
| TA3MR | $039A_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>1 0 : One-shot timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Pulse output function select bit | 0 (Must always be "0" in three-phase PWM output mode) | ○ | ○ |
| MR1 | External trigger select bit | Invalid in three-phase PWM output mode<br>Can be "0" or "1" | ○ | ○ |
| MR2 | Trigger select bit | 1 : Selected by event/trigger select register | ○ | ○ |
| MR3 | 0 (Must always be "0" in one-shot timer mode) | | ○ | ○ |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $fc_{32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

Timer B2 mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | | | 0 | 0 |

| Symbol | Address | When reset |
|---|---|---|
| TB2MR | $039D_{16}$ | $00XX0000_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| TMOD0 | Operation mode select bit | b1 b0<br>0 0 : Timer mode | ○ | ○ |
| TMOD1 | | | ○ | ○ |
| MR0 | Invalid in timer mode<br>Can be "0" or "1" | | ○ | ○ |
| MR1 | | | ○ | ○ |
| MR2 | 0 (Fixed to "0" in timer mode ; i = 0) | | ○ | ○ |
| MR3 | Invalid in timer mode.<br>This bit can neither be set nor reset. When read in timer mode, its content is indeterminate. | | ○ | × |
| TCK0 | Count source select bit | b7 b6<br>0 0 : $f_1$<br>0 1 : $f_8$<br>1 0 : $f_{32}$<br>1 1 : $fc_{32}$ | ○ | ○ |
| TCK1 | | | ○ | ○ |

**Figure 1.18.4. Timer mode registers in three-phase waveform mode**

Figure 1.18.5 shows the block diagram for three-phase waveform mode. In three-phase waveform mode, the positive-phase waveforms (U phase, V phase, and W phase) and negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase), six waveforms in total, are output from $P8_0$,$P8_1$, $P7_2$, $P7_3$, $P7_4$, and $P7_5$ as active on the "L" level. Of the timers used in this mode, timer A4 controls the U phase and $\overline{U}$ phase, timer A1 controls the V phase and $\overline{V}$ phase, and timer A2 controls the W phase and $\overline{W}$ phase respectively; timer B2 controls the periods of one-shot pulse output from timers A4, A1, and A2.

In outputting a waveform, dead time can be set so as to cause the "L" level of the positive waveform output (U phase, V phase, and W phase) not to lap over the "L" level of the negative waveform output ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase).

To set short circuit time, use three 8-bit timers sharing the reload register for setting dead time. A value from 1 through 255 can be set as the count of the timer for setting dead time. The timer for setting dead time works as a one-shot timer. If a value is written to the dead timer ($034C_{16}$), the value is written to the reload register shared by the three timers for setting dead time.

Any of the timers for setting dead time takes the value of the reload register into its counter, if a start trigger comes from its corresponding timer, and performs a down count in line with the clock source selected by the dead time timer count source select bit (bit 2 at $0349_{16}$). The timer can receive another trigger again before the workings due to the previous trigger are completed. In this instance, the timer performs a down count from the reload register's content after its transfer, provoked by the trigger, to the timer for setting dead time.

Since the timer for setting dead time works as a one-shot timer, it starts outputting pulses if a trigger comes; it stops outputting pulses as soon as its content becomes $00_{16}$, and waits for the next trigger to come.

The positive waveforms (U phase, V phase, and W phase) and the negative waveforms ($\overline{U}$ phase, $\overline{V}$ phase, and $\overline{W}$ phase) in three-phase waveform mode are output from respective ports by means of setting "1" in the output control bit (bit 3 at $0348_{16}$). Setting "0" in this bit causes the ports to be the state of set by port direction register. This bit can be set to "0" not only by use of the applicable instruction, but by entering a falling edge in the $\overline{NMI}$ terminal or by resetting. Also, if "1" is set in the positive and negative phases concurrent L output disable function enable bit (bit 4 at $0348_{16}$) causes one of the pairs of U phase and $\overline{U}$ phase, V phase and $\overline{V}$ phase, and W phase and $\overline{W}$ phase concurrently go to "L", as a result, the port become the state of set by port direction register.

## Timers' functions for three-phase motor control

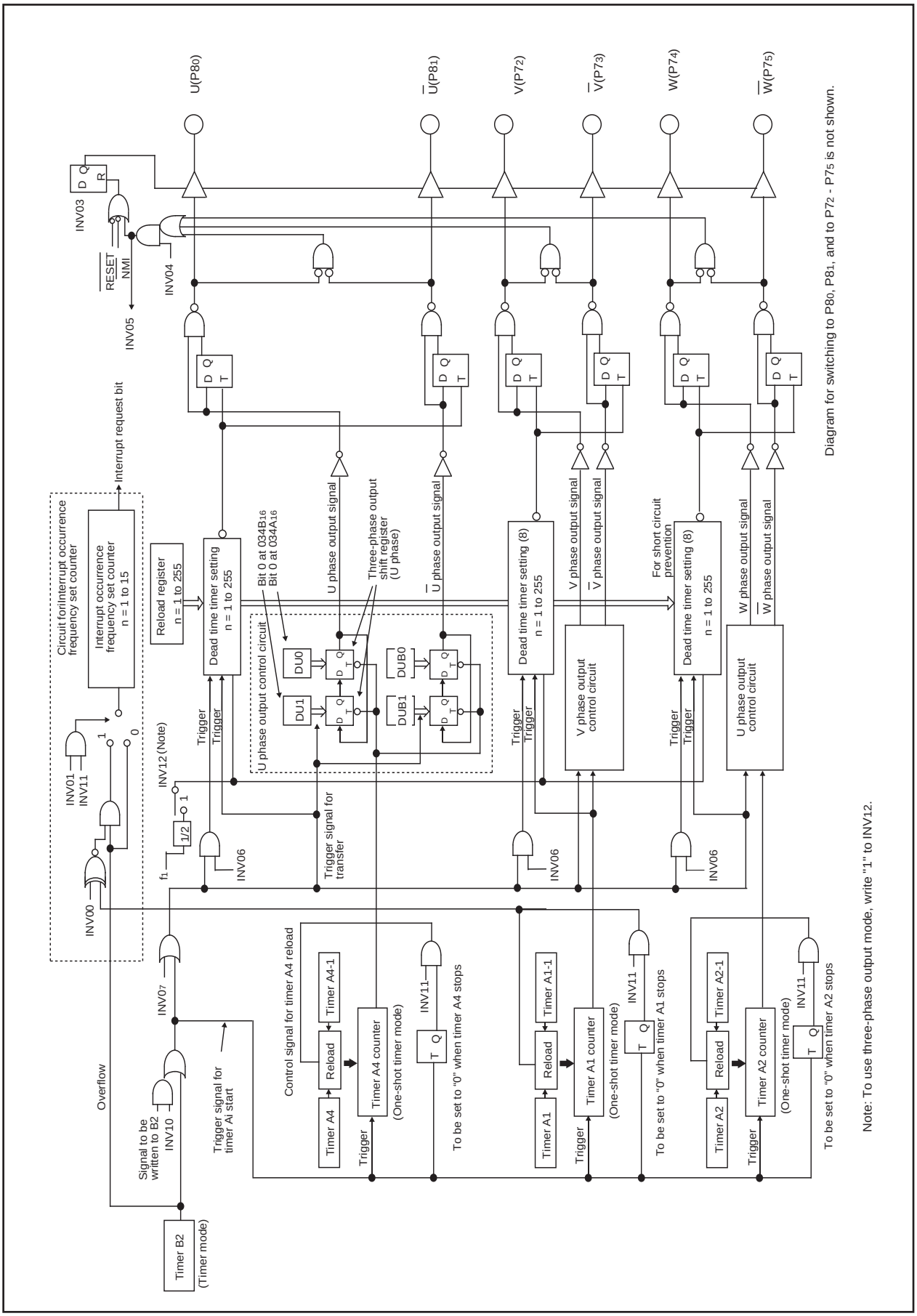



**Figure 1.18.5. Block diagram for three-phase waveform mode**

## Triangular wave modulation

To generate a PWM waveform of triangular wave modulation, set "0" in the modulation mode select bit (bit 6 at $0348_{16}$). Also, set "1" in the timers A4-1, A1-1, A2-1 control bit (bit 1 at $0349_{16}$). In this mode, each of timers A4, A1, and A2 has two timer registers, and alternately reloads the timer register's content to the counter every time timer B2 counter's content becomes $0000_{16}$. If "1" is set to the effective interrupt output specification bit (bit 1 at $0348_{16}$), the frequency of interrupt requests that occur every time the timer B2 counter's value becomes $0000_{16}$ can be set by use of the timer B2 counter ($034D_{16}$) for setting the frequency of interrupt occurrences. The frequency of occurrences is given by (setting; setting $\neq$ 0).

Setting "1" in the effective interrupt output specification bit (bit 1 at $0348_{16}$) provides the means to choose which value of the timer A1 reload control signal to use, "0" or "1", to cause timer B2's interrupt request to occur. To make this selection, use the effective interrupt output polarity selection bit (bit 0 at $0348_{16}$).

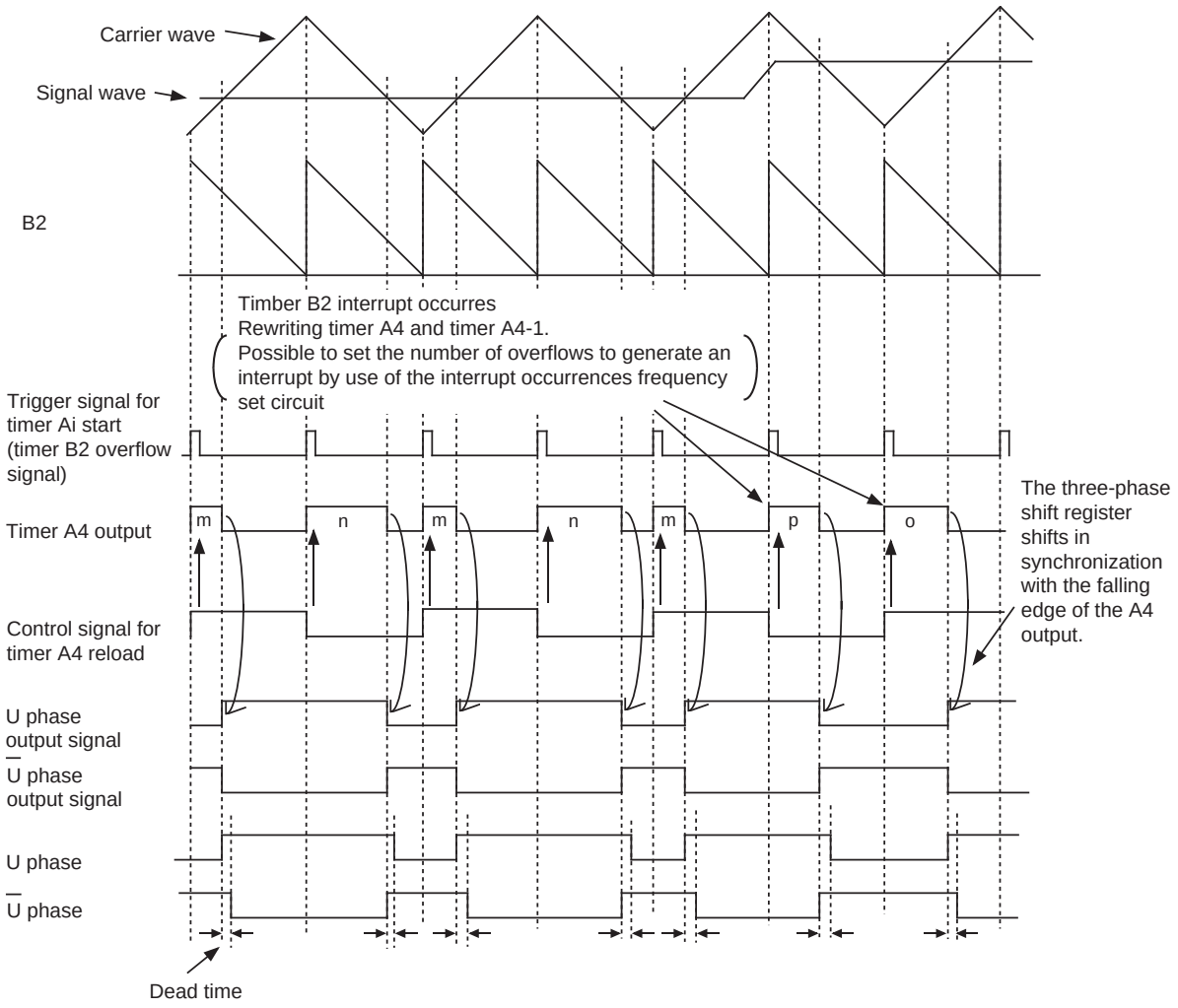
An example of U phase waveform is shown in Figure 1.18.6, and the description of waveform output workings is given below. Set "1" in DU0 (bit 0 at $034A_{16}$). And set "0" in DUB0 (bit 1 at $034A_{16}$). In addition, set "0" in DU1 (bit 0 at $034B_{16}$) and set "1" in DUB1 (bit 1 at $034B_{16}$). Also, set "0" in the effective interrupt output specification bit (bit 1 at $0348_{16}$) to set a value in the timer B2 interrupt occurrence frequency set counter. By this setting, a timer B2 interrupt occurs when the timer B2 counter's content becomes $0000_{16}$ as many as (setting) times. Furthermore, set "1" in the effective interrupt output specification bit (bit 1 at $0348_{16}$), set in the effective interrupt polarity select bit (bit 0 at $0348_{16}$) and set "1" in the interrupt occurrence frequency set counter($034D_{16}$). These settings cause a timer B2 interrupt to occur every other interval when the U phase output goes to "H".

When the timer B2 counter's content becomes $0000_{16}$, timer A4 starts outputting one-shot pulses. In this instance, the content of DU1 (bit 0 at $034B_{16}$) and that of DU0 (bit 0 at $034A_{16}$) are set in the three-phase output shift register (U phase), the content of DUB1 (bit 1 at $034B_{16}$) and that of DUB0 (bit 1 at $034A_{16}$) are set in the three-phase shift register ($\overline{U}$ phase). After triangular wave modulation mode is selected, however, no setting is made in the shift register even though the timer B2 counter's content becomes $0000_{16}$.

The value of DU0 and that of DUB0 are output to the U terminal ($P8_0$) and to the $\overline{U}$ terminal ($P8_1$) respectively. When the timer A4 counter counts the value written to timer A4 ($038F_{16}$, $038E_{16}$) and when timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to $\overline{U}$ phase output signal respectively. At this time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the "L" level of the U phase waveform does not lap over the "L" level of the $\overline{U}$ phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the "H" level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register's content changes from "1" to "0" by the effect of the one-shot pulses. When the timer for setting dead time finishes outputting one-shot pulses, "0" already shifted in the three-phase shift register goes effective, and the U phase waveform changes to the "L" level. When the timer B2 counter's content becomes $0000_{16}$, the timer A4 counter starts counting the value written to timer A4-1 ($0347_{16}$, $0346_{16}$), and starts outputting one-shot pulses. When timer A4 finishes outputting one-shot pulses, the three-phase shift register's content is shifted one position, but if the three-phase output shift register's content changes from "0" to "1" as a result of the shift, the output level changes from "L" to "H" without waiting for the timer for setting dead time to finish outputting one-shot pulses. A U phase waveform is generated by these workings repeatedly. With the exception that the three-phase output shift register on the U phase side is used, the workings in generating a U phase waveform, which has the opposite phase of the U phase waveform, are the same as in generating a U

phase waveform. In this way, a waveform can be picked up from the applicable terminal in a manner in which the "L" level of the U phase waveform doesn't lap over that of the U phase waveform, which has the opposite phase of the U phase waveform. The width of the "L" level too can be adjusted by varying the values of timer B2, timer A4, and timer A4-1. In dealing with the V and W phases, and $\overline{V}$ and $\overline{W}$ phases, the latter are of opposite phase of the former, have the corresponding timers work similarly to dealing with the U and $\overline{U}$ phases to generate an intended waveform.

**A carrier wave of triangular waveform**




Note: Set to triangular wave modulation mode and to three-phase mode 1.

**Figure 1.18.6. Timing chart of operation (1)**

Assigning certain values to DU0 (bit 0 at 034A16) and DUB0 (bit 1 at 034A16), and to DU1 (bit 0 at 034B16) and DUB1 (bit 1 at 034B16) allows the user to output the waveforms as shown in Figure 1.18.7, that is, to output the U phase alone, to fix $\overline{U}$ phase to "H", to fix the U phase to "H," or to output the $\overline{U}$ phase alone.

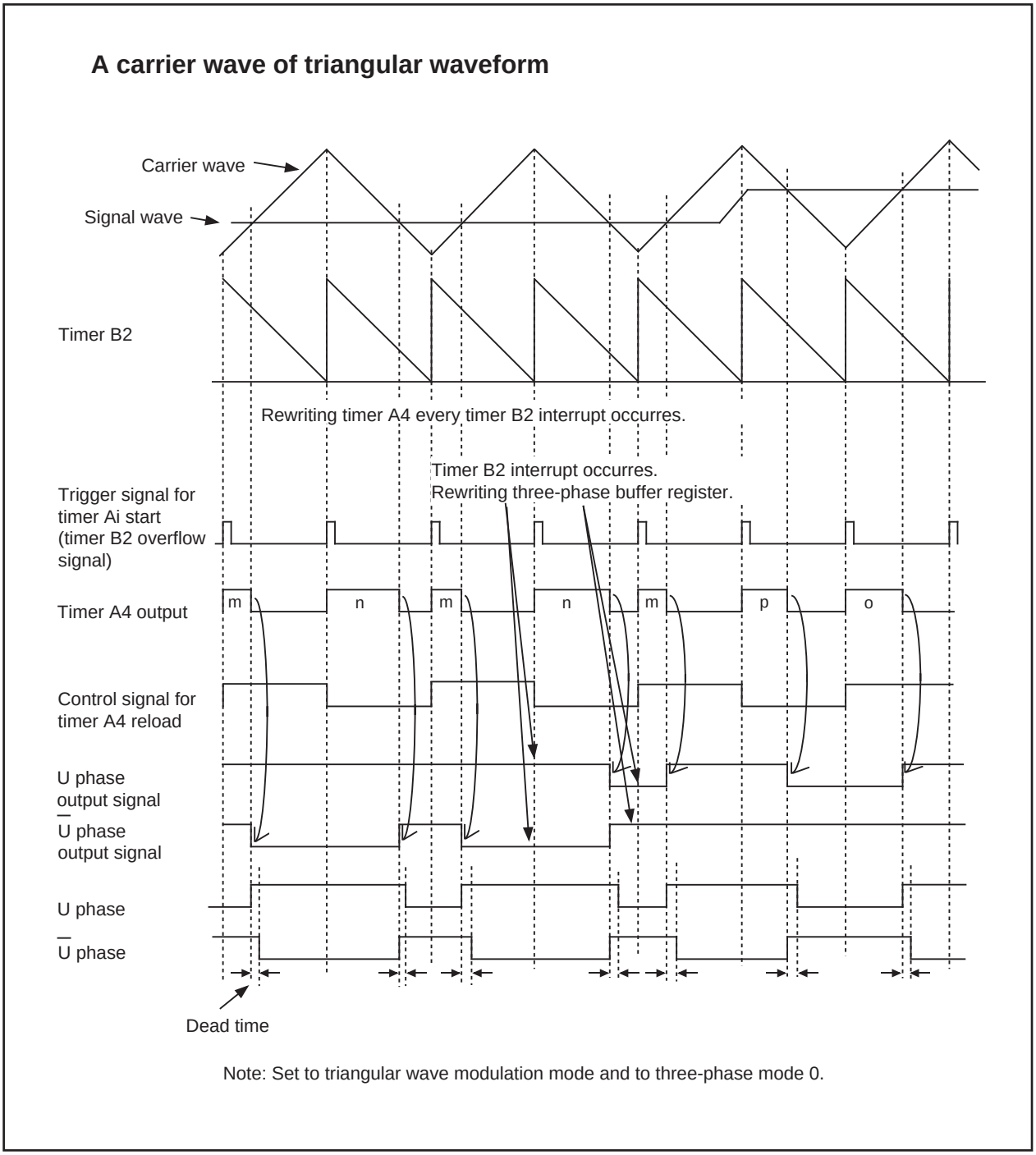



**Figure 1.18.7. Timing chart of operation (2)**

## Sawtooth modulation

To generate a PWM waveform of sawtooth wave modulation, set "1" in the modulation mode select bit (bit 6 at $0348_{16}$). Also, set "0" in the timers A4-1, A1-1, and A2-1 control bit (bit 1 at $0349_{16}$). In this mode, the timer registers of timers A4, A1, and A2 comprise conventional timers A4, A1, and A2 alone, and reload the corresponding timer register's content to the counter every time the timer B2 counter's content becomes $0000_{16}$. The effective interrupt output specification bit (bit 1 at $0348_{16}$) and the effective interrupt output polarity select bit (bit 0 at $0348_{16}$) go nullified.

An example of U phase waveform is shown in Figure 75, and the description of waveform output workings is given below. Set "1" in DU0 (bit 0 at $034A_{16}$), and set "0" in DUB0 (bit 1 at $034A_{16}$). In addition, set "0" in DU1 (bit 0 at $034A_{16}$) and set "1" in DUB1 (bit 1 at $034A_{16}$).

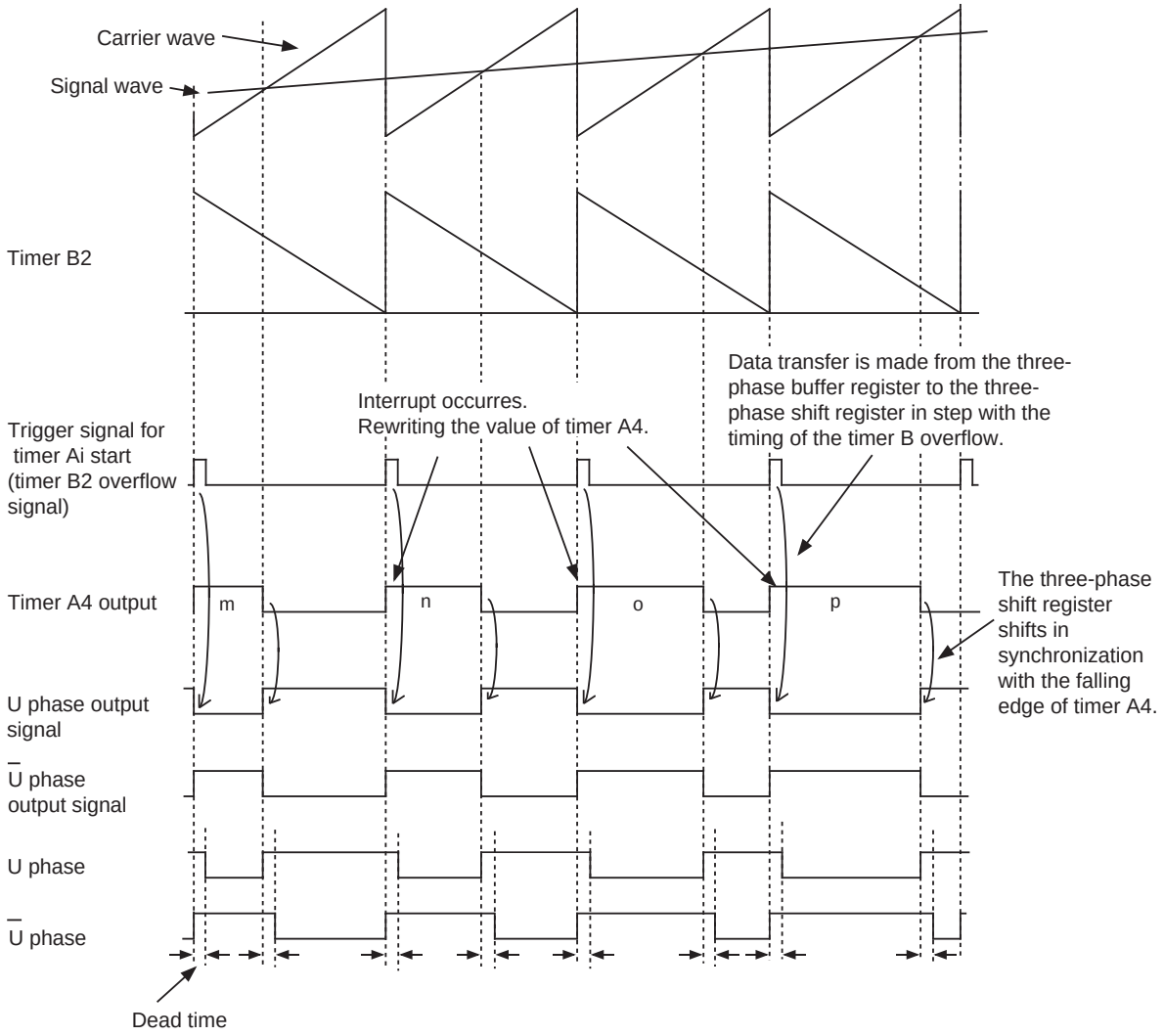
When the timber B2 counter's content becomes $0000_{16}$, timer B2 generates an interrupt, and timer A4 starts outputting one-shot pulses at the same time. In this instance, the contents of the three-phase buffer registers DU1 and DU0 are set in the three-phase output shift register (U phase), and the contents of DUB1 and DUB0 are set in the three-phase output register (U phase). After this, the three-phase buffer register's content is set in the three-phase shift register every time the timer B2 counter's content becomes $0000_{16}$.

The value of DU0 and that of DUB0 are output to the U terminal ($P8_0$) and to the $\overline{U}$ terminal ($P8_1$) respectively. When the timer A4 counter counts the value written to timer A4 ($038F_{16}$, $038E_{16}$) and when timer A4 finishes outputting one-shot pulses, the three-phase output shift register's content is shifted one position, and the value of DU1 and that of DUB1 are output to the U phase output signal and to the $\overline{U}$ output signal respectively. At this time, one-shot pulses are output from the timer for setting dead time used for setting the time over which the "L" level of the U phase waveform doesn't lap over the "L" level of the $\overline{U}$ phase waveform, which has the opposite phase of the former. The U phase waveform output that started from the "H" level keeps its level until the timer for setting dead time finishes outputting one-shot pulses even though the three-phase output shift register's content changes from "1" to "0 "by the effect of the one-shot pulses. When the timer for setting dead time finishes outputting one-shot pulses, 0 already shifted in the three-phase shift register goes effective, and the U phase waveform changes to the "L" level. When the timer B2 counter's content becomes $0000_{16}$, the contents of the three-phase buffer registers DU1 and DU0 are set in the three-phase shift register (U phase), and the contents of DUB1 and DUB0 are set in the three-phase shift register ($\overline{U}$ phase) again.

A U phase waveform is generated by these workings repeatedly. With the exception that the three-phase output shift register on the $\overline{U}$ phase side is used, the workings in generating a $\overline{U}$ phase waveform, which has the opposite phase of the U phase waveform, are the same as in generating a U phase waveform. In this way, a waveform can be picked up from the applicable terminal in a manner in which the "L" level of the U phase waveform doesn't lap over that of the U phase waveform, which has the opposite phase of the U phase waveform. The width of the "L" level too can be adjusted by varying the values of timer B2 and timer A4. In dealing with the V and W phases, and $\overline{V}$ and $\overline{W}$ phases, the latter are of opposite phase of the former, have the corresponding timers work similarly to dealing with the U and $\overline{U}$ phases to generate an intended waveform.

Setting "1" both in DUB0 and in DUB1 provides a means to output the U phase alone and to fix the $\overline{U}$ phase output to "H" as shown in Figure 1.18.8.

**A carrier wave of sawtooth waveform**




Note: Set to sawtooth modulation mode and to three-phase mode 0.

**Figure 1.18.8. Timing chart of operation (3)**

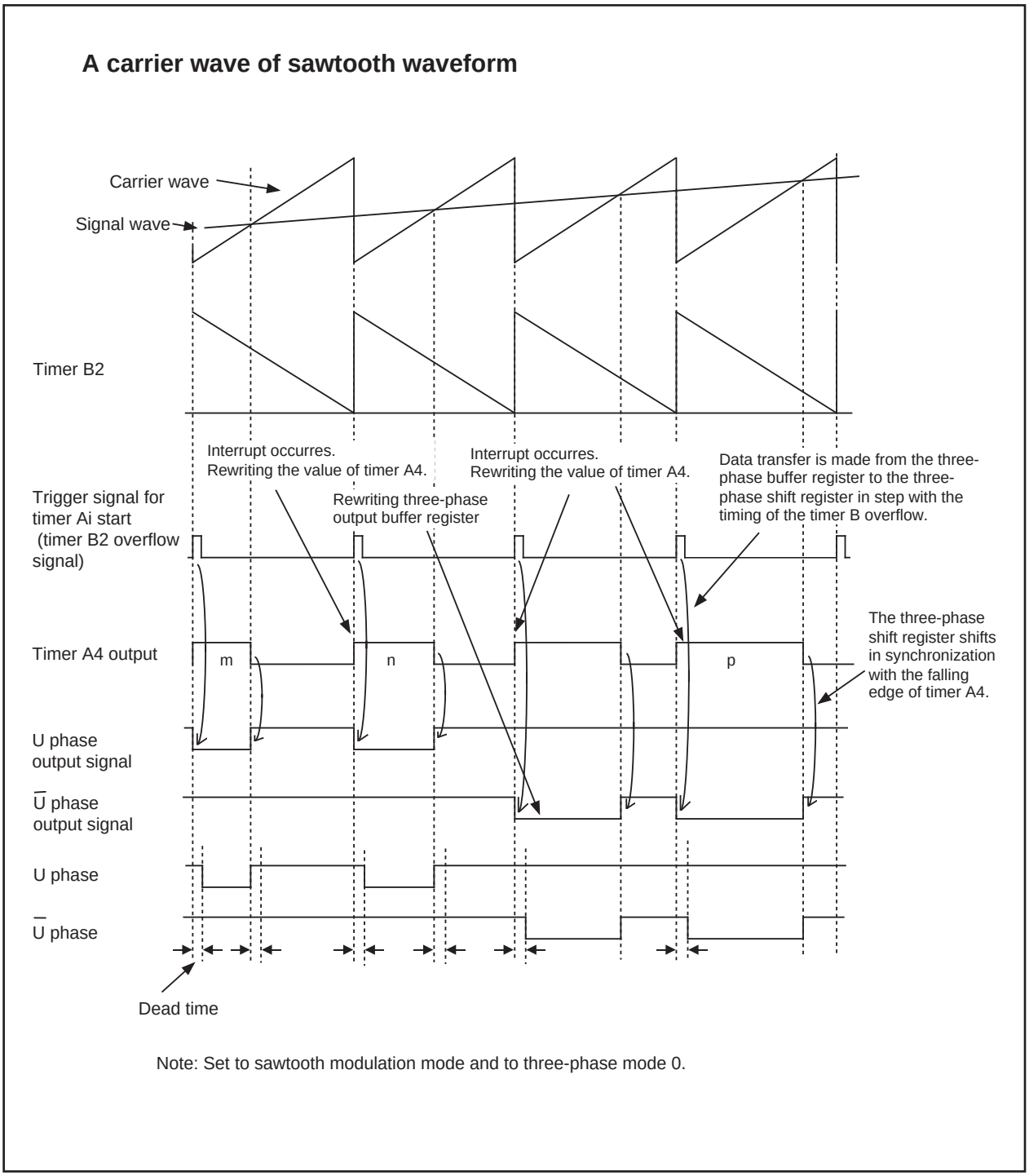



**Figure 1.18.9. Timing chart of operation (4)**

## Serial I/O

Serial I/O is configured as five channels: UART0, UART1, UART2, S I/O3 and S I/O4.

## UART0 to 2

UART0, UART1 and UART2 each have an exclusive timer to generate a transfer clock, so they operate independently of each other.

Figure 1.19.1 shows the block diagram of UART0, UART1 and UART2. Figures 1.19.2 and 1.19.3 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 $03A0_{16}$, $03A8_{16}$ and $0378_{16}$) determine whether UARTi is used as a clock synchronous serial I/O or as a UART. Although a few functions are different, UART0, UART1 and UART2 have almost the same functions. 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.

Table 1.19.1 shows the comparison of functions of UART0 through UART2, and Figures 1.19.4 to 1.19.8 show the registers related to UARTi.

Note: SIM : Subscriber Identity Module

**Table 1.19.1. 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 $\overline{CTS}$/$\overline{RTS}$ 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.

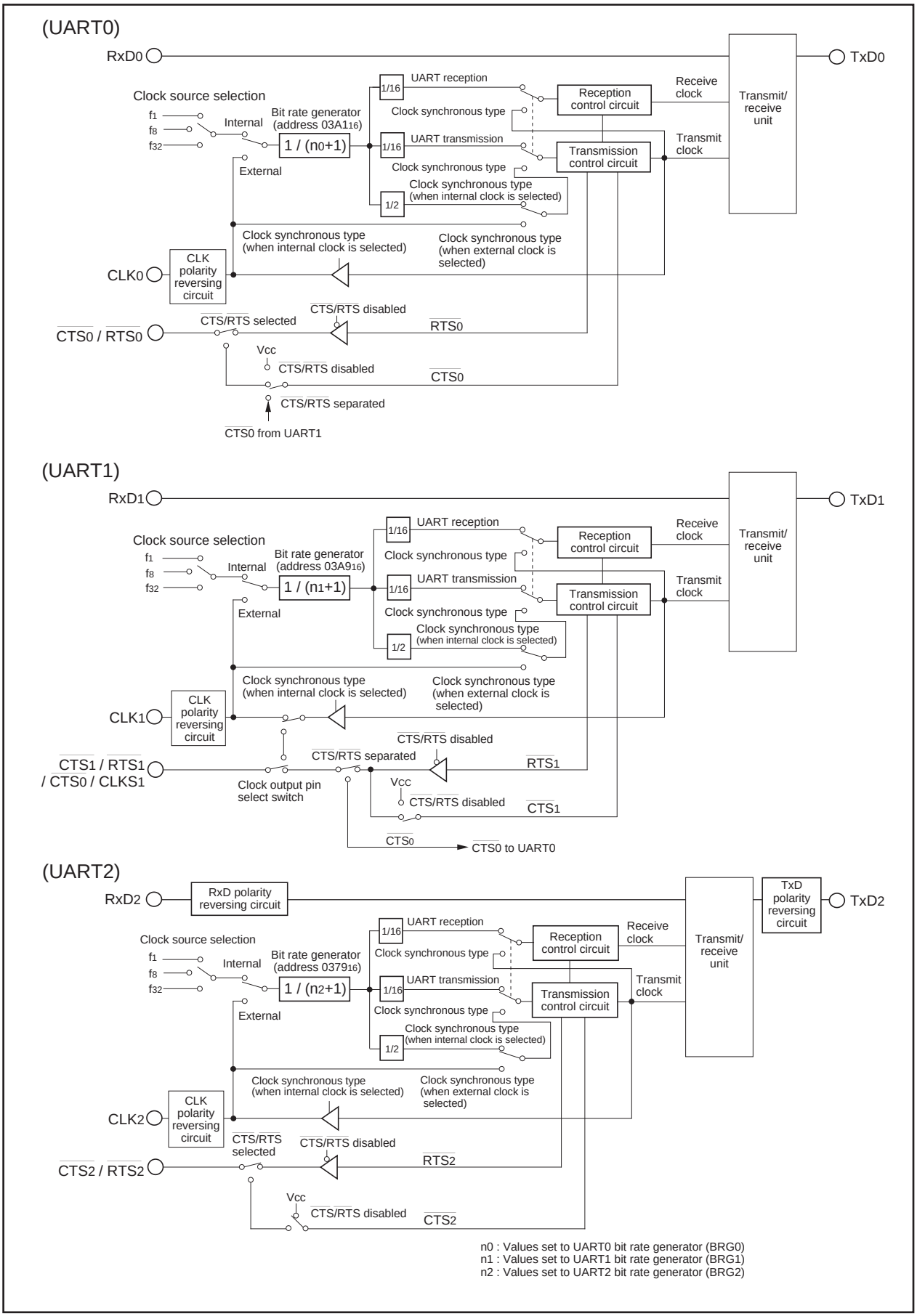



**Figure 1.19.1. Block diagram of UARTi (i = 0 to 2)**

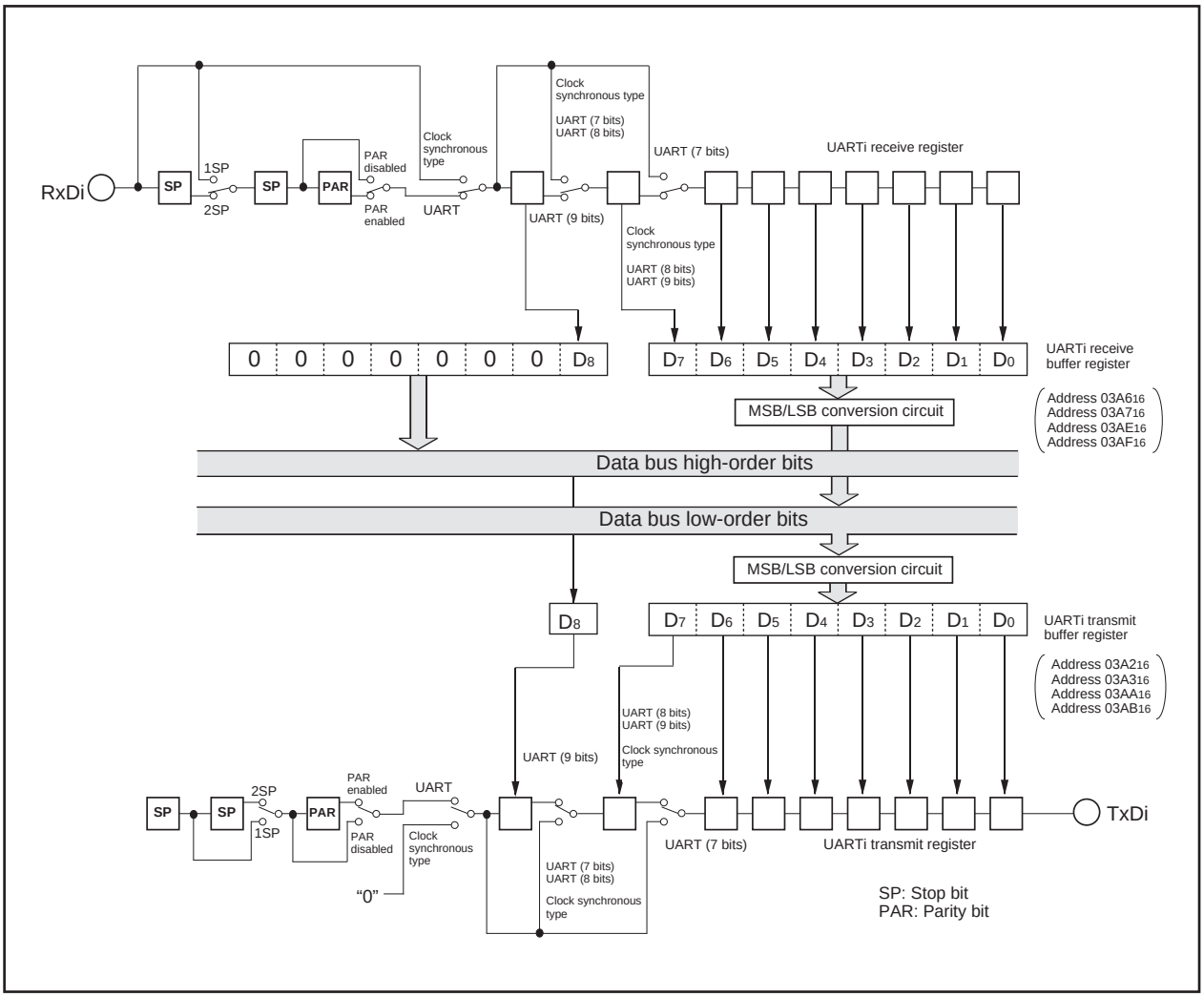



**Figure 1.19.2. Block diagram of UARTi (i = 0, 1) transmit/receive unit**

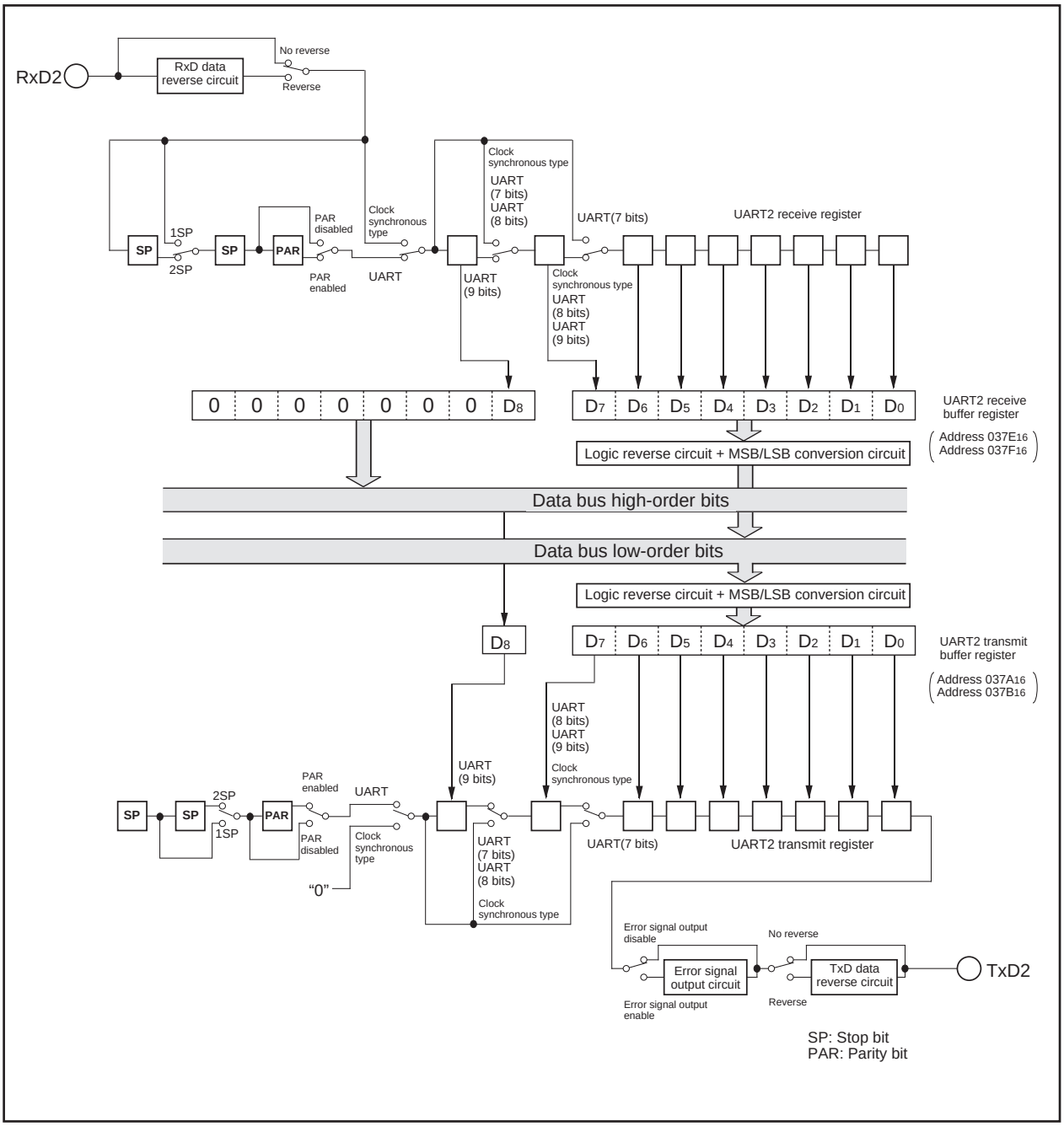
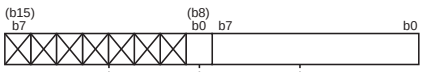



**Figure 1.19.3. Block diagram of UART2 transmit/receive unit**

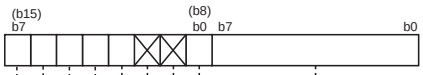
UARTi transmit buffer register




| Symbol | Address | When reset |
|---|---|---|
| U0TB | $03A3_{16}$, $03A2_{16}$ | Indeterminate |
| U1TB | $03AB_{16}$, $03AA_{16}$ | Indeterminate |
| U2TB | $037B_{16}$, $037A_{16}$ | Indeterminate |

| Function | R | W |
|---|---|---|
| Transmit data | × | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turn out to be indeterminate. | — | — |

UARTi receive buffer register

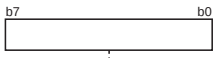



| Symbol | Address | When reset |
|---|---|---|
| U0RB | $03A7_{16}$, $03A6_{16}$ | Indeterminate |
| U1RB | $03AF_{16}$, $03AE_{16}$ | Indeterminate |
| U2RB | $037F_{16}$, $037E_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| — | ——— | Receive data | Receive data | ○ | × |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | | — | — |
| ABT | Arbitration lost detecting flag (Note 2) | 0 : Not detected<br>1 : Detected | Invalid | ○ | ○ |
| OER | Overrun error flag (Note 1) | 0 : No overrun error<br>1 : Overrun error found | 0 : No overrun error<br>1 : Overrun error found | ○ | × |
| FER | Framing error flag (Note 1) | Invalid | 0 : No framing error<br>1 : Framing error found | ○ | × |
| PER | Parity error flag (Note 1) | Invalid | 0 : No parity error<br>1 : Parity error found | ○ | × |
| SUM | Error sum flag (Note 1) | Invalid | 0 : No error<br>1 : Error found | ○ | × |

Note 1: Bits 15 through 12 are set to “0” when the serial I/O mode select bit (bits 2 to 0 at addresses $03A0_{16}$, $03A8_{16}$ and $0378_{16}$) are set to $000_2$ or the receive enable bit is set to “0”.
(Bit 15 is set to “0” when bits 14 to 12 all are set to “0”.) Bits 14 and 13 are also set to “0” when the lower byte of the UARTi receive buffer register (addresses $03A6_{16}$, $03AE_{16}$ and $037E_{16}$) is read out.
Note 2: Arbitration lost detecting flag is allocated to U2RB and noting but “0” may be written. Nothing is assigned in bit 11 of U0RB and U1RB. These bits can neither be set or reset. When read, the value of this bit is “0”.

UARTi bit rate generator




| Symbol | Address | When reset |
|---|---|---|
| U0BRG | $03A1_{16}$ | Indeterminate |
| U1BRG | $03A9_{16}$ | Indeterminate |
| U2BRG | $0379_{16}$ | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Assuming that set value = n, BRGi divides the count source by n + 1 | $00_{16}$ to $FF_{16}$ | × | ○ |

**Figure 1.19.4. Serial I/O-related registers (1)**

UARTi transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: UiMR(i=0,1) | Address: 03A0$_{16}$, 03A8$_{16}$ | When reset: 00$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | Must be fixed to 001 | b2 b1 b0<br>1 0 0 : Transfer data 7 bits long<br>1 0 1 : Transfer data 8 bits long<br>1 1 0 : Transfer data 9 bits long<br>0 0 0 : Serial I/O invalid<br>0 1 0 : Inhibited<br>0 1 1 : Inhibited<br>1 1 1 : Inhibited | ○ | ○ |
| SMD1 | | b2 b1 b0<br>0 0 0 : Serial I/O invalid<br>0 1 0 : Inhibited<br>0 1 1 : Inhibited<br>1 1 1 : Inhibited | | ○ | ○ |
| SMD2 | | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock | 0 : Internal clock<br>1 : External clock | ○ | ○ |
| STPS | Stop bit length select bit | Invalid | 0 : One stop bit<br>1 : Two stop bits | ○ | ○ |
| PRY | Odd/even parity select bit | Invalid | Valid when bit 6 = “1”<br>0 : Odd parity<br>1 : Even parity | ○ | ○ |
| PRYE | Parity enable bit | Invalid | 0 : Parity disabled<br>1 : Parity enabled | ○ | ○ |
| SLEP | Sleep select bit | Must always be “0” | 0 : Sleep mode deselected<br>1 : Sleep mode selected | ○ | ○ |

UART2 transmit/receive mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: U2MR | Address: 0378$_{16}$ | When reset: 00$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | Must be fixed to 001 | b2 b1 b0<br>1 0 0 : Transfer data 7 bits long<br>1 0 1 : Transfer data 8 bits long<br>1 1 0 : Transfer data 9 bits long<br>0 0 0 : Serial I/O invalid<br>0 1 0 : Inhibited<br>0 1 1 : Inhibited<br>1 1 1 : Inhibited | ○ | ○ |
| SMD1 | | b2 b1 b0<br>0 0 0 : Serial I/O invalid<br>0 1 0 : (Note)<br>0 1 1 : Inhibited<br>1 1 1 : Inhibited | | ○ | ○ |
| SMD2 | | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock | 0 : Internal clock<br>1 : External clock | ○ | ○ |
| STPS | Stop bit length select bit | Invalid | 0 : One stop bit<br>1 : Two stop bits | ○ | ○ |
| PRY | Odd/even parity select bit | Invalid | Valid when bit 6 = “1”<br>0 : Odd parity<br>1 : Even parity | ○ | ○ |
| PRYE | Parity enable bit | Invalid | 0 : Parity disabled<br>1 : Parity enabled | ○ | ○ |
| IOPOL | TxD, RxD I/O polarity reverse bit | 0 : No reverse<br>1 : Reverse<br>Usually set to “0” | 0 : No reverse<br>1 : Reverse<br>Usually set to “0” | ○ | ○ |

Note: Bit 2 to bit 0 are set to “010$_2$” when $I^2C$ mode is used.

**Figure 1.19.5. Serial I/O-related registers (2)**

UARTi transmit/receive control register 0

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: UiC0(i=0,1) — Address: 03A4$_{16}$, 03AC$_{16}$ — When reset: 08$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}/\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | × |
| CRD | $\overline{CTS}/\overline{RTS}$ disable bit | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled ($P6_0$ and $P6_4$ function as programmable I/O port) | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : $\overline{CTS}/\overline{RTS}$ function disabled ($P6_0$ and $P6_4$ function as programmable I/O port) | ○ | ○ |
| NCH | Data output select bit | 0 : TXDi pin is CMOS output<br>1 : TXDi pin is N-channel open-drain output | 0: TXDi pin is CMOS output<br>1: TXDi pin is N-channel open-drain output | ○ | ○ |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | Must always be "0" | ○ | ○ |
| UFORM | Transfer format select bit | 0 : LSB first<br>1 : MSB first | Must always be "0" | ○ | ○ |

Note 1: Set the corresponding port direction register to "0".
Note 2: The settings of the corresponding port register and port direction register are invalid.

UART2 transmit/receive control register 0

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | ⊠ | | | | | |

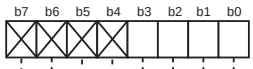
Symbol: U2C0 — Address: 037C$_{16}$ — When reset: 08$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| CLK0 | BRG count source select bit | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | b1 b0<br>0 0 : $f_1$ is selected<br>0 1 : $f_8$ is selected<br>1 0 : $f_{32}$ is selected<br>1 1 : Inhibited | ○ | ○ |
| CLK1 | | | | ○ | ○ |
| CRS | $\overline{CTS}/\overline{RTS}$ function select bit | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | Valid when bit 4 = "0"<br>0 : $\overline{CTS}$ function is selected (Note 1)<br>1 : $\overline{RTS}$ function is selected (Note 2) | ○ | ○ |
| TXEPT | Transmit register empty flag | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | 0 : Data present in transmit register (during transmission)<br>1 : No data present in transmit register (transmission completed) | ○ | × |
| CRD | $\overline{CTS}/\overline{RTS}$ disable bit | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : CTS/RTS function disabled ($P7_3$ functions programmable I/O port) | 0 : $\overline{CTS}/\overline{RTS}$ function enabled<br>1 : CTS/RTS function disabled ($P7_3$ functions programmable I/O port) | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be "0". | | | | — | — |
| CKPOL | CLK polarity select bit | 0 : Transmit data is output at falling edge of transfer clock and receive data is input at rising edge<br>1 : Transmit data is output at rising edge of transfer clock and receive data is input at falling edge | Must always be "0" | ○ | ○ |
| UFORM | Transfer format select bit (Note 3) | 0 : LSB first<br>1 : MSB first | 0 : LSB first<br>1 : MSB first | ○ | ○ |

Note 1: Set the corresponding port direction register to "0".
Note 2: The settings of the corresponding port register and port direction register are invalid.
Note 3: Only clock synchronous serial I/O mode and 8-bit UART mode are valid.

**Figure 1.19.6. Serial I/O-related registers (3)**

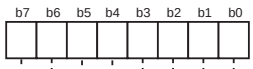
UARTi transmit/receive control register 1




Symbol: UiC1(i=0,1) | Address: 03A5$_{16}$,03AD$_{16}$ | When reset: 02$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | 0 : Transmission disabled<br>1 : Transmission enabled | ○ | ○ |
| TI | Transmit buffer empty flag | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | ○ | × |
| RE | Receive enable bit | 0 : Reception disabled<br>1 : Reception enabled | 0 : Reception disabled<br>1 : Reception enabled | ○ | ○ |
| RI | Receive complete flag | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | ○ | × |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | | – | – |

UART2 transmit/receive control register 1




Symbol: U2C1 | Address: 037D$_{16}$ | When reset: 02$_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| TE | Transmit enable bit | 0 : Transmission disabled<br>1 : Transmission enabled | 0 : Transmission disabled<br>1 : Transmission enabled | ○ | ○ |
| TI | Transmit buffer empty flag | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | 0 : Data present in transmit buffer register<br>1 : No data present in transmit buffer register | ○ | × |
| RE | Receive enable bit | 0 : Reception disabled<br>1 : Reception enabled | 0 : Reception disabled<br>1 : Reception enabled | ○ | ○ |
| RI | Receive complete flag | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | 0 : No data present in receive buffer register<br>1 : Data present in receive buffer register | ○ | × |
| U2IRS | UART2 transmit interrupt cause select bit | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmit is completed (TXEPT = 1) | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmit is completed (TXEPT = 1) | ○ | ○ |
| U2RRM | UART2 continuous receive mode enable bit | 0 : Continuous receive mode disabled<br>1 : Continuous receive mode enabled | Invalid | ○ | ○ |
| U2LCH | Data logic select bit | 0 : No reverse<br>1 : Reverse | 0 : No reverse<br>1 : Reverse | ○ | ○ |
| U2ERE | Error signal output enable bit | Must be fixed to “0” | 0 : Output disabled<br>1 : Output enabled | ○ | ○ |

**Figure 1.19.7. Serial I/O-related registers (4)**

UART transmit/receive control register 2

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| ☒ | | | | | | | |

Symbol: UCON — Address: $03B0_{16}$ — When reset: $X0000000_2$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| U0IRS | UART0 transmit interrupt cause select bit | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmission completed (TXEPT = 1) | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmission completed (TXEPT = 1) | ○ | ○ |
| U1IRS | UART1 transmit interrupt cause select bit | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmission completed (TXEPT = 1) | 0 : Transmit buffer empty (TI = 1)<br>1 : Transmission completed (TXEPT = 1) | ○ | ○ |
| U0RRM | UART0 continuous receive mode enable bit | 0 : Continuous receive mode disabled<br>1 : Continuous receive mode enable | Invalid | ○ | ○ |
| U1RRM | UART1 continuous receive mode enable bit | 0 : Continuous receive mode disabled<br>1 : Continuous receive mode enabled | Invalid | ○ | ○ |
| CLKMD0 | CLK/CLKS select bit 0 | Valid when bit 5 = "1"<br>0 : Clock output to CLK1<br>1 : Clock output to CLKS1 | Invalid | ○ | ○ |
| CLKMD1 | CLK/CLKS select bit 1 (Note) | 0 : Normal mode (CLK output is CLK1 only)<br>1 : Transfer clock output from multiple pins function selected | Must always be "0" | ○ | ○ |
| RCSP | Separate $\overline{CTS}/\overline{RTS}$ bit | 0 : $\overline{CTS}/\overline{RTS}$ shared pin<br>1 : CTS/RTS separated | 0 : $\overline{CTS}/\overline{RTS}$ shared pin<br>1 : $\overline{CTS}/\overline{RTS}$ separated | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be indeterminate. | | | | — | — |

Note: When using multiple pins to output the transfer clock, the following requirements must be met:
• UART1 internal/external clock select bit (bit 3 at address $03A8_{16}$) = "0".

UART2 special mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| ☒ | | | | | | | |

Symbol: U2SMR — Address: $0377_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | IIC mode selection bit | 0 : Normal mode<br>1 : IIC mode | Must always be "0" | ○ | ○ |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | Must always be "0" | ○ | ○ |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected | Must always be "0" | ○ | ○ (Note) |
| LSYN | SCLL sync output enable bit | 0 : Disabled<br>1 : Enabled | Must always be "0" | ○ | ○ |
| ABSCS | Bus collision detect sampling clock select bit | Must always be "0" | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer A0 | ○ | ○ |
| ACSE | Auto clear function select bit of transmit enable bit | Must always be "0" | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | ○ | ○ |
| SSS | Transmit start condition select bit | Must always be "0" | 0 : Ordinary<br>1 : Falling edge of RxD2 | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be "0". | | | | — | — |

Note: Nothing but "0" may be written.

**Figure 1.19.8. Serial I/O-related registers (5)**

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

The clock synchronous serial I/O mode uses a transfer clock to transmit and receive data. Tables 1.19.2 and 1.19.3 list the specifications of the clock synchronous serial I/O mode. Figure 1.19.9 shows the UARTi transmit/receive mode register.

**Table 1.19.2. Specifications of clock synchronous serial I/O mode (1)**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • When internal clock is selected (bit 3 at addresses 03A0$_{16}$, 03A8$_{16}$, 037816 = “0”) : fi/ 2(n+1) (Note 1) fi = f1, f8, f32<br>• When external clock is selected (bit 3 at addresses 03A0$_{16}$, 03A8$_{16}$, 037816 = “1”) : Input from CLKi pin |
| Transmission/reception control | • $\overline{\text{CTS}}$ function/$\overline{\text{RTS}}$ function/$\overline{\text{CTS}}$, $\overline{\text{RTS}}$ function chosen to be invalid |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>– Transmit enable bit (bit 0 at addresses 03A5$_{16}$, 03AD$_{16}$, 037D$_{16}$) = “1”<br>– Transmit buffer empty flag (bit 1 at addresses 03A5$_{16}$, 03AD$_{16}$, 037D$_{16}$) = “0”<br>– When $\overline{\text{CTS}}$ function selected, $\overline{\text{CTS}}$ input level = “L”<br>• Furthermore, if external clock is selected, the following requirements must also be met:<br>– CLKi polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”: CLKi input level = “H”<br>– CLKi polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “1”: CLKi input level = “L” |
| Reception start condition | • To start reception, the following requirements must be met:<br>– Receive enable bit (bit 2 at addresses 03A5$_{16}$, 03AD$_{16}$, 037D$_{16}$) = “1”<br>– Transmit enable bit (bit 0 at addresses 03A5$_{16}$, 03AD$_{16}$, 037D$_{16}$) = “1”<br>– Transmit buffer empty flag (bit 1 at addresses 03A5$_{16}$, 03AD$_{16}$, 037D$_{16}$) = “0”<br>• Furthermore, if external clock is selected, the following requirements must also be met:<br>– CLKi polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”: CLKi input level = “H”<br>– CLKi polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “1”: CLKi input level = “L” |
| Interrupt request generation timing | • When transmitting<br>– Transmit interrupt cause select bit (bits 0, 1 at address 03B0$_{16}$, bit 4 at address 037D$_{16}$) = “0”: Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>– Transmit interrupt cause select bit (bits 0, 1 at address 03B0$_{16}$, bit 4 at address 037D$_{16}$) = “1”: Interrupts requested when data transmission from UARTi transfer register is completed<br>• When receiving<br>– Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed |
| Error detection | • Overrun error (Note 2)<br>This error occurs when the next data is ready before contents of UARTi receive buffer register are read out |

Note 1: “n” denotes the value 00$_{16}$ to FF$_{16}$ 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 1.19.4. Specifications of clock synchronous serial I/O mode (2)**

| Item | Specification |
|---|---|
| Select function | • CLK polarity selection<br>Whether transmit data is output/input at the rising edge or falling edge of the transfer clock can be selected<br>• LSB first/MSB first selection<br>Whether transmission/reception begins with bit 0 or bit 7 can be selected<br>• Continuous receive mode selection<br>Reception is enabled simultaneously by a read from the receive buffer register<br>• Transfer clock output from multiple pins selection (UART1) (Note)<br>UART1 transfer clock can be chosen by software to be output from one of the two pins set<br>• Separate $\overline{CTS}$/$\overline{RTS}$ pins (UART0) (Note)<br>UART0 $\overline{CTS}$ and $\overline{RTS}$ pins each can be assigned to separate pins<br>• Switching serial data logic (UART2)<br>Whether to reverse data in writing to the transmission buffer register or reading the reception buffer register can be selected.<br>• TxD, RxD I/O polarity reverse (UART2)<br>This function is reversing TxD port output and RxD port input. All I/O data level is reversed. |

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

UARTi transmit/receive mode registers

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | | | | | 0 | 0 | 1 |

Symbol: UiMR(i=0,1) — Address: $03A0_{16}$, $03A8_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | b2 b1 b0<br>0 0 1 : Clock synchronous serial I/O mode | ○ | ○ |
| SMD1 | | | ○ | ○ |
| SMD2 | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock | ○ | ○ |
| STPS | Invalid in clock synchronous serial I/O mode | | ○ | ○ |
| PRY | | | ○ | ○ |
| PRYE | | | ○ | ○ |
| SLEP | 0 (Must always be “0” in clock synchronous serial I/O mode) | | ○ | ○ |

UART2 transmit/receive mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| 0 | | | | | 0 | 0 | 1 |

Symbol: U2MR — Address: $0378_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | b2 b1 b0<br>0 0 1 : Clock synchronous serial I/O mode | ○ | ○ |
| SMD1 | | | ○ | ○ |
| SMD2 | | | ○ | ○ |
| CKDIR | Internal/external clock select bit | 0 : Internal clock<br>1 : External clock | ○ | ○ |
| STPS | Invalid in clock synchronous serial I/O mode | | ○ | ○ |
| PRY | | | ○ | ○ |
| PRYE | | | ○ | ○ |
| IOPOL | TxD, RxD I/O polarity reverse bit (Note) | 0 : No reverse<br>1 : Reverse | ○ | ○ |

Note: Usually set to “0”.

**Figure 1.19.9. UARTi transmit/receive mode register in clock synchronous serial I/O mode**

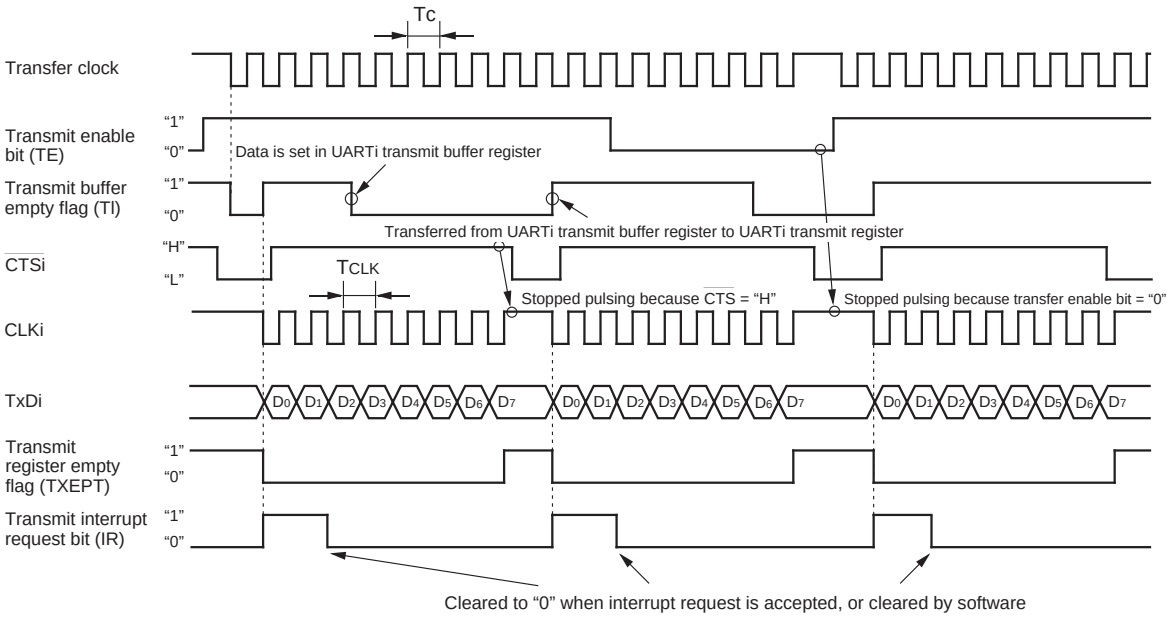
Table 1.19.4 lists the functions of the input/output pins during clock synchronous serial I/O mode. This table shows the pin functions when the transfer clock output from multiple pins and the separate $\overline{CTS}$/$\overline{RTS}$ pins functions are not selected. Note that for a period from when 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 1.19.4. Input/output pin functions in clock synchronous serial I/O mode**

| Pin name | Function | Method of selection |
|---|---|---|
| TxDi ($P6_3$, $P6_7$, $P7_0$) | Serial data output | (Outputs dummy data when performing reception only) |
| RxDi ($P6_2$, $P6_6$, $P7_1$) | Serial data input | Port $P6_2$, $P6_6$ and $P7_1$ direction register (bits 2 and 6 at address $03EE_{16}$, bit 1 at address $03EF_{16}$)= "0"<br>(Can be used as an input port when performing transmission only) |
| CLKi ($P6_1$, $P6_5$, $P7_2$) | Transfer clock output | Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$, $0378_{16}$) = "0" |
| | Transfer clock input | Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$, $0378_{16}$) = "1"<br>Port $P6_1$, $P6_5$ and $P7_2$ direction register (bits 1 and 5 at address $03EE_{16}$, bit 2 at address $03EF_{16}$) = "0" |
| $\overline{CTSi}$/$\overline{RTSi}$ ($P6_0$, $P6_4$, $P7_3$) | $\overline{CTS}$ input | $\overline{CTS}$/$\overline{RTS}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) ="0"<br>$\overline{CTS}$/$\overline{RTS}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "0"<br>Port $P6_0$, $P6_4$ and $P7_3$ direction register (bits 0 and 4 at address $03EE_{16}$, bit 3 at address $03EF_{16}$) = "0" |
| | $\overline{RTS}$ output | $\overline{CTS}$/$\overline{RTS}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "0"<br>$\overline{CTS}$/$\overline{RTS}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "1" |
| | Programmable I/O port | $\overline{CTS}$/$\overline{RTS}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "1" |

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

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

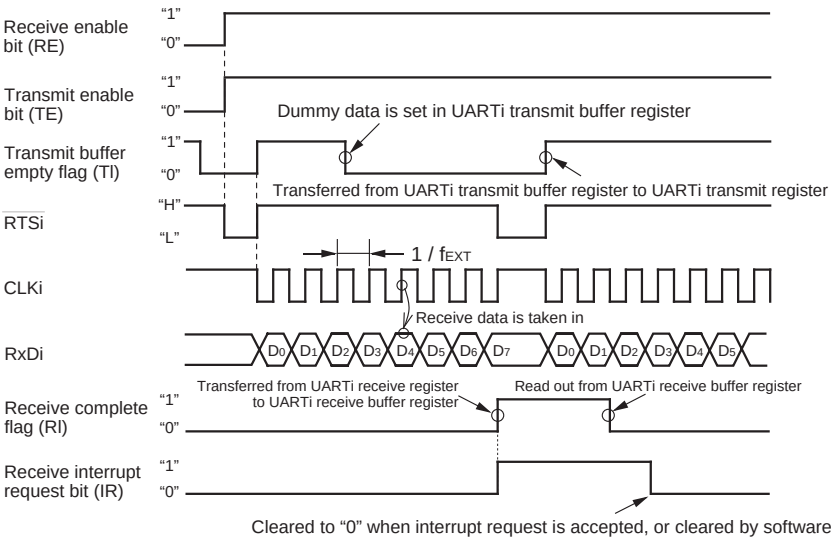



Shown in ( ) are bit symbols.

The above timing applies to the following settings:
- Internal clock is selected.
- CTS function is selected.
- CLK polarity select bit = "0".
- Transmit interrupt cause select bit = "0".

Tc = TCLK = 2(n + 1) / fi
fi: frequency of BRGi count source (f1, f8, f32)
n: value set to BRGi

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




Shown in ( ) are bit symbols.

The above timing applies to the following settings:
- External clock is selected.
- RTS function is selected.
- CLK polarity select bit = "0".

fEXT: frequency of external clock

Meet the following conditions are met when the CLK input before data reception = "H"
- Transmit enable bit → "1"
- Receive enable bit → "1"
- Dummy data write to UARTi transmit buffer register

**Figure 1.19.10. Typical transmit/receive timings in clock synchronous serial I/O mode**

**(a) Polarity select function**

As shown in Figure 1.19.11, the CLK polarity select bit (bit 6 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) allows selection of the polarity of the transfer clock.

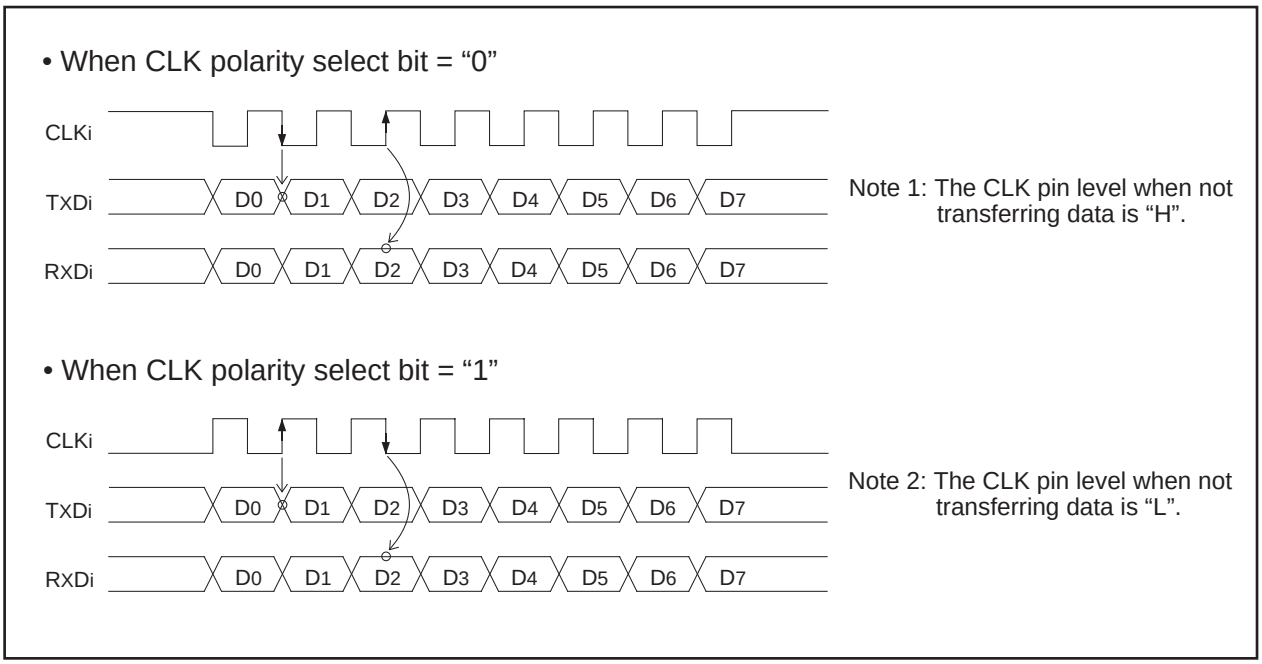



**Figure 1.19.11. Polarity of transfer clock**

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

As shown in Figure 1.19.12, when the transfer format select bit (bit 7 at addresses 03A4$_{16}$, 03AC$_{16}$, 037C$_{16}$) = “0”, the transfer format is “LSB first”; when the bit = “1”, the transfer format is “MSB first”.

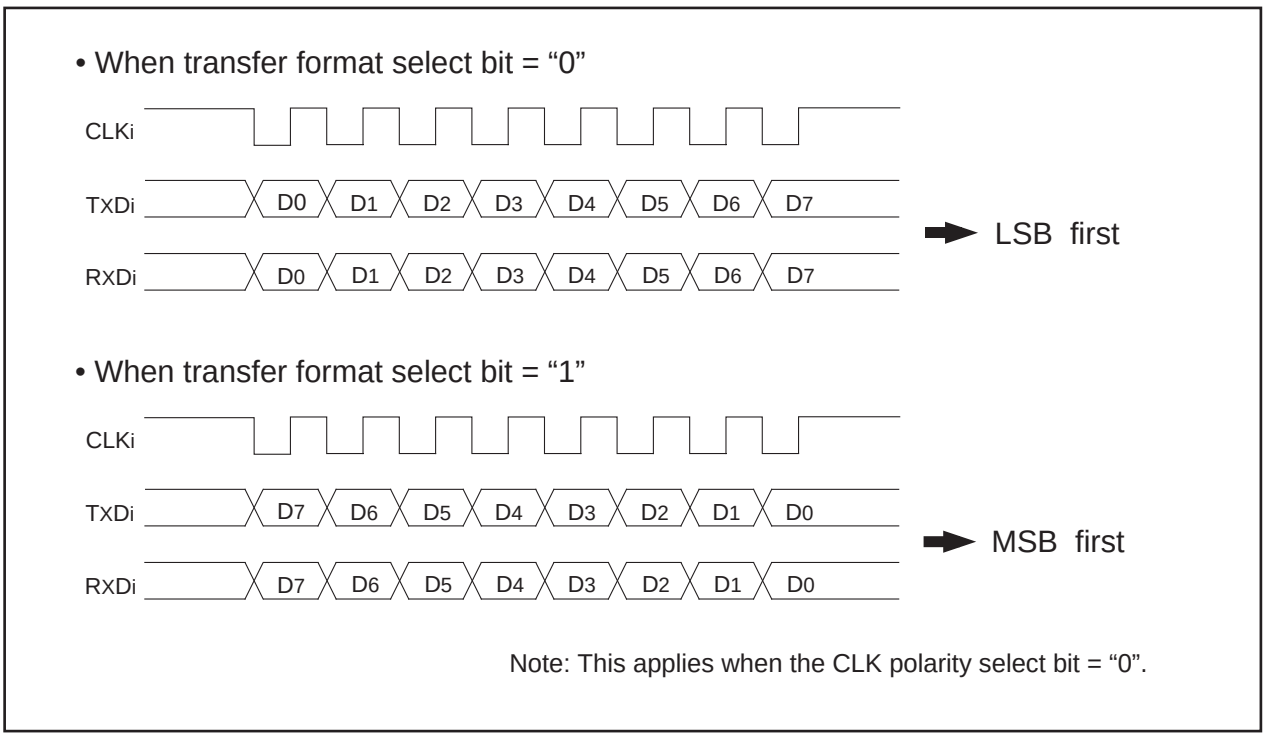



**Figure 1.19.12. Transfer format**

**(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 03B0$_{16}$). (See Figure 1.19.3.) The multiple pins function is valid only when the internal clock is selected for UART1. Note that when this function is selected, UART1 $\overline{CTS}$/$\overline{RTS}$ function cannot be used.

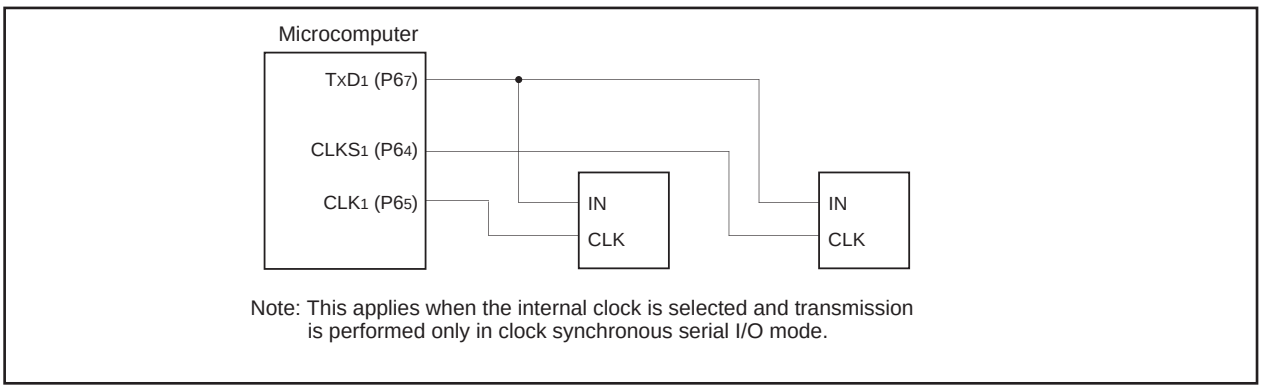



**Figure 1.19.13. The transfer clock output from the multiple pins function usage**

**(d) Continuous receive mode**

If the continuous receive mode enable bit (bits 2 and 3 at address 03B0$_{16}$, bit 5 at address 037D$_{16}$) 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 $\overline{CTS}$/$\overline{RTS}$ 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 037D$_{16}$) = "1", and writing to transmit buffer register or reading from receive buffer register, data is reversed. Figure 1.19.14 shows the example of serial data logic switch timing.

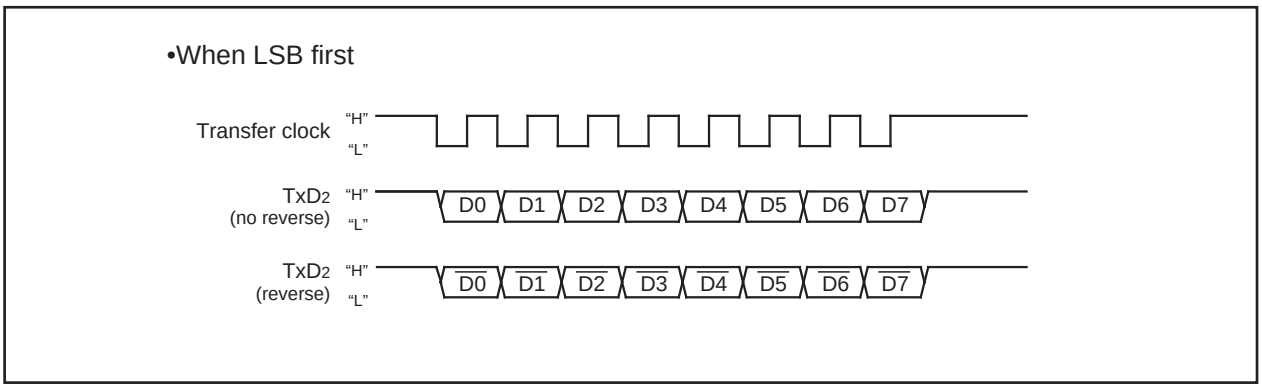



**Figure 1.19.14. Serial data logic switch timing**

## (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. Tables 1.19.5 and 1.19.6 list the specifications of the UART mode. Figure 1.19.15 shows the UARTi transmit/receive mode register.

**Table 1.19.5. Specifications of UART Mode (1)**

| Item | Specification |
|---|---|
| Transfer data format | • Character bit (transfer data): 7 bits, 8 bits, or 9 bits as selected<br>• Start bit: 1 bit<br>• Parity bit: Odd, even, or nothing as selected<br>• Stop bit: 1 bit or 2 bits as selected |
| Transfer clock | • When internal clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $0378_{16}$ = "0") :<br>fi/16(n+1) (Note 1) fi = $f_1$, $f_8$, $f_{32}$<br>• When external clock is selected (bit 3 at addresses $03A0_{16}$, $03A8_{16}$, $0378_{16}$ ="1") :<br>$f_{EXT}$/16(n+1)(Note 1) (Note 2) |
| Transmission/reception control | • $\overline{CTS}$ function/$\overline{RTS}$ function/$\overline{CTS}$, $\overline{RTS}$ function chosen to be invalid |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>- Transmit enable bit (bit 0 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>- Transmit buffer empty flag (bit 1 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "0"<br>- When $\overline{CTS}$ function selected, $\overline{CTS}$ input level = "L" |
| Reception start condition | • To start reception, the following requirements must be met:<br>- Receive enable bit (bit 2 at addresses $03A5_{16}$, $03AD_{16}$, $037D_{16}$) = "1"<br>- Start bit detection |
| Interrupt request generation timing | • When transmitting<br>- Transmit interrupt cause select bits (bits 0,1 at address $03B0_{16}$, bit4 at address $037D_{16}$) = "0": Interrupts requested when data transfer from UARTi transfer buffer register to UARTi transmit register is completed<br>- Transmit interrupt cause select bits (bits 0, 1 at address $03B0_{16}$, bit4 at address $037D_{16}$) = "1": Interrupts requested when data transmission from UARTi transfer register is completed<br>• When receiving<br>- Interrupts requested when data transfer from UARTi receive register to UARTi receive buffer register is completed |
| Error detection | • Overrun error (Note 3)<br>This error occurs when the next data is ready before contents of UARTi receive buffer register are read out<br>• Framing error<br>This error occurs when the number of stop bits set is not detected<br>• Parity error<br>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<br>• Error sum flag<br>This flag is set (= 1) when any of the overrun, framing, and parity errors is encountered |

Note 1: 'n' denotes the value $00_{16}$ to $FF_{16}$ 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 1.19.6. Specifications of UART Mode (2)**

| Item | Specification |
|---|---|
| Select function | • Separate $\overline{CTS}$/$\overline{RTS}$ pins (UART0)<br>UART0 $\overline{CTS}$ and $\overline{RTS}$ pins each can be assigned to separate pins<br>• Sleep mode selection (UART0, UART1)<br>This mode is used to transfer data to and from one of multiple slave microcomputers<br>• Serial data logic switch (UART2)<br>This function is reversing logic value of transferring data. Start bit, parity bit and stop bit are not reversed.<br>• TxD, RxD I/O polarity switch<br>This function is reversing TxD port output and RxD port input. All I/O data level is reversed. |

UARTi transmit / receive mode registers

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: UiMR(i=0,1) — Address: $03A0_{16}$, $03A8_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | b2 b1 b0<br>1 0 0 : Transfer data 7 bits long<br>1 0 1 : Transfer data 8 bits long<br>1 1 0 : Transfer data 9 bits long | ○ | ○ |
| SMD1 | | | ○ | ○ |
| SMD2 | | | ○ | ○ |
| CKDIR | Internal / external clock select bit | 0 : Internal clock<br>1 : External clock | ○ | ○ |
| STPS | Stop bit length select bit | 0 : One stop bit<br>1 : Two stop bits | ○ | ○ |
| PRY | Odd / even parity select bit | Valid when bit 6 = “1”<br>0 : Odd parity<br>1 : Even parity | ○ | ○ |
| PRYE | Parity enable bit | 0 : Parity disabled<br>1 : Parity enabled | ○ | ○ |
| SLEP | Sleep select bit | 0 : Sleep mode deselected<br>1 : Sleep mode selected | ○ | ○ |

UART2 transmit / receive mode register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: U2MR — Address: $0378_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMD0 | Serial I/O mode select bit | b2 b1 b0<br>1 0 0 : Transfer data 7 bits long<br>1 0 1 : Transfer data 8 bits long<br>1 1 0 : Transfer data 9 bits long | ○ | ○ |
| SMD1 | | | ○ | ○ |
| SMD2 | | | ○ | ○ |
| CKDIR | Internal / external clock select bit | 0 : Internal clock<br>1 : External clock | ○ | ○ |
| STPS | Stop bit length select bit | 0 : One stop bit<br>1 : Two stop bits | ○ | ○ |
| PRY | Odd / even parity select bit | Valid when bit 6 = “1”<br>0 : Odd parity<br>1 : Even parity | ○ | ○ |
| PRYE | Parity enable bit | 0 : Parity disabled<br>1 : Parity enabled | ○ | ○ |
| IOPOL | TxD, RxD I/O polarity reverse bit (Note) | 0 : No reverse<br>1 : Reverse | ○ | ○ |

Note: Usually set to “0”.

**Figure 1.19.15. UARTi transmit/receive mode register in UART mode**

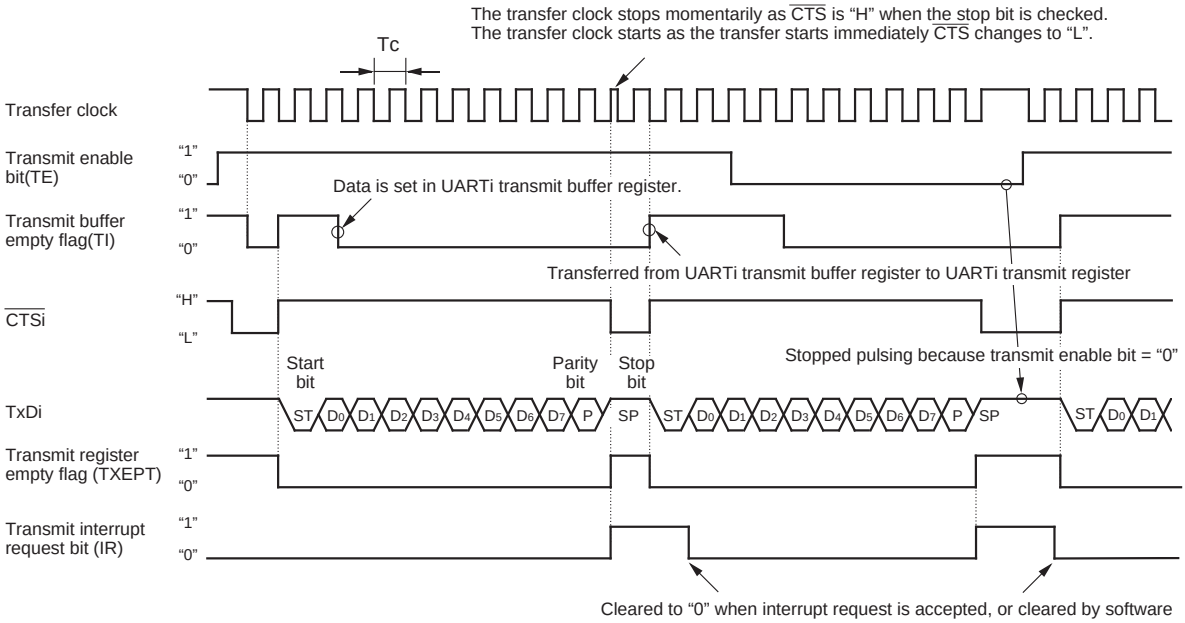
Table 1.19.7 lists the functions of the input/output pins during UART mode. This table shows the pin functions when the separate $\overline{CTS}/\overline{RTS}$ pins function is not selected. Note that for a period from when the UARTi operation mode is selected to when transfer starts, the TxDi pin outputs a "H". (If the N-channel open-drain is selected, this pin is in floating state.)

**Table 1.19.7. Input/output pin functions in UART mode**

| Pin name | Function | Method of selection |
|---|---|---|
| TxDi ($P6_3$, $P6_7$, $P7_0$) | Serial data output | |
| RxDi ($P6_2$, $P6_6$, $P7_1$) | Serial data input | Port $P6_2$, $P6_6$ and $P7_1$ direction register (bits 2 and 6 at address $03EE_{16}$, bit 1 at address $03EF_{16}$)= "0" (Can be used as an input port when performing transmission only) |
| CLKi ($P6_1$, $P6_5$, $P7_2$) | Programmable I/O port | Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$, $0378_{16}$) = "0" |
| | Transfer clock input | Internal/external clock select bit (bit 3 at address $03A0_{16}$, $03A8_{16}$, $0378_{16}$) = "1" Port $P6_1$, $P6_5$ and $P7_2$ direction register (bits 1 and 5 at address $03EE_{16}$, bit 2 at address $03EF_{16}$) = "0" |
| $\overline{CTS}i/\overline{RTS}i$ ($P6_0$, $P6_4$, $P7_3$) | $\overline{CTS}$ input | $\overline{CTS}/\overline{RTS}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) ="0" $\overline{CTS}/\overline{RTS}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "0" Port $P6_0$, $P6_4$ and $P7_3$ direction register (bits 0 and 4 at address $03EE_{16}$, bit 3 at address $03EF_{16}$) = "0" |
| | $\overline{RTS}$ output | $\overline{CTS}/\overline{RTS}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "0" $\overline{CTS}/\overline{RTS}$ function select bit (bit 2 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "1" |
| | Programmable I/O port | $\overline{CTS}/\overline{RTS}$ disable bit (bit 4 at address $03A4_{16}$, $03AC_{16}$, $037C_{16}$) = "1" |

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

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

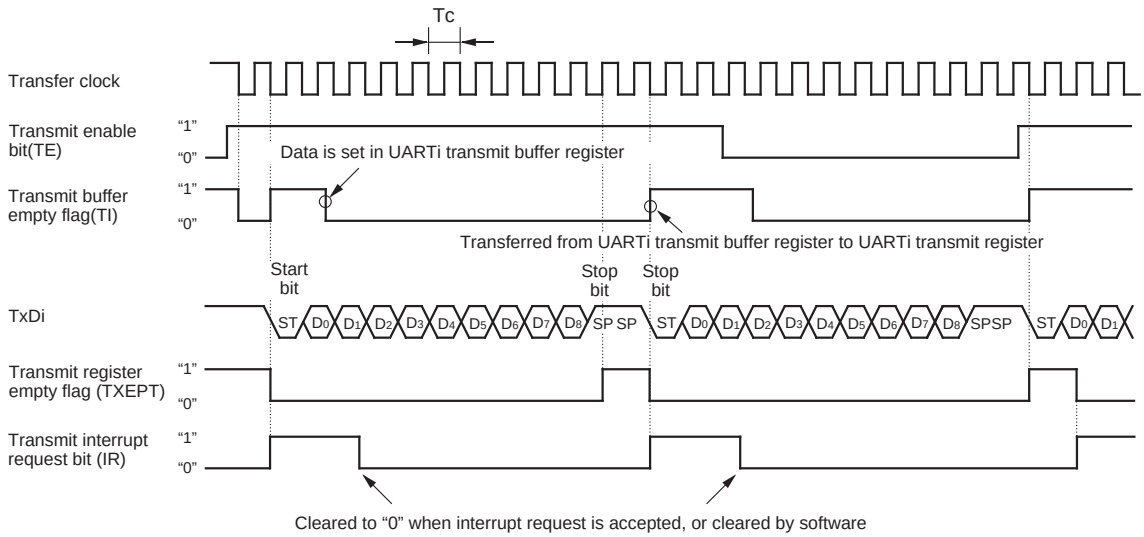



Shown in ( ) are bit symbols.

The above timing applies to the following settings :
- Parity is enabled.
- One stop bit.
- CTS function is selected.
- Transmit interrupt cause select bit = "1".

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of BRGi count source (f1, f8, f32)
fEXT : frequency of BRGi count source (external clock)
n : value set to BRGi

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




Shown in ( ) are bit symbols.

The above timing applies to the following settings :
- Parity is disabled.
- Two stop bits.
- CTS function is disabled.
- Transmit interrupt cause select bit = "0".

Tc = 16 (n + 1) / fi or 16 (n + 1) / fEXT
fi : frequency of BRGi count source (f1, f8, f32)
fEXT : frequency of BRGi count source (external clock)
n : value set to BRGi

**Figure 1.19.16. Typical transmit timings in UART mode**

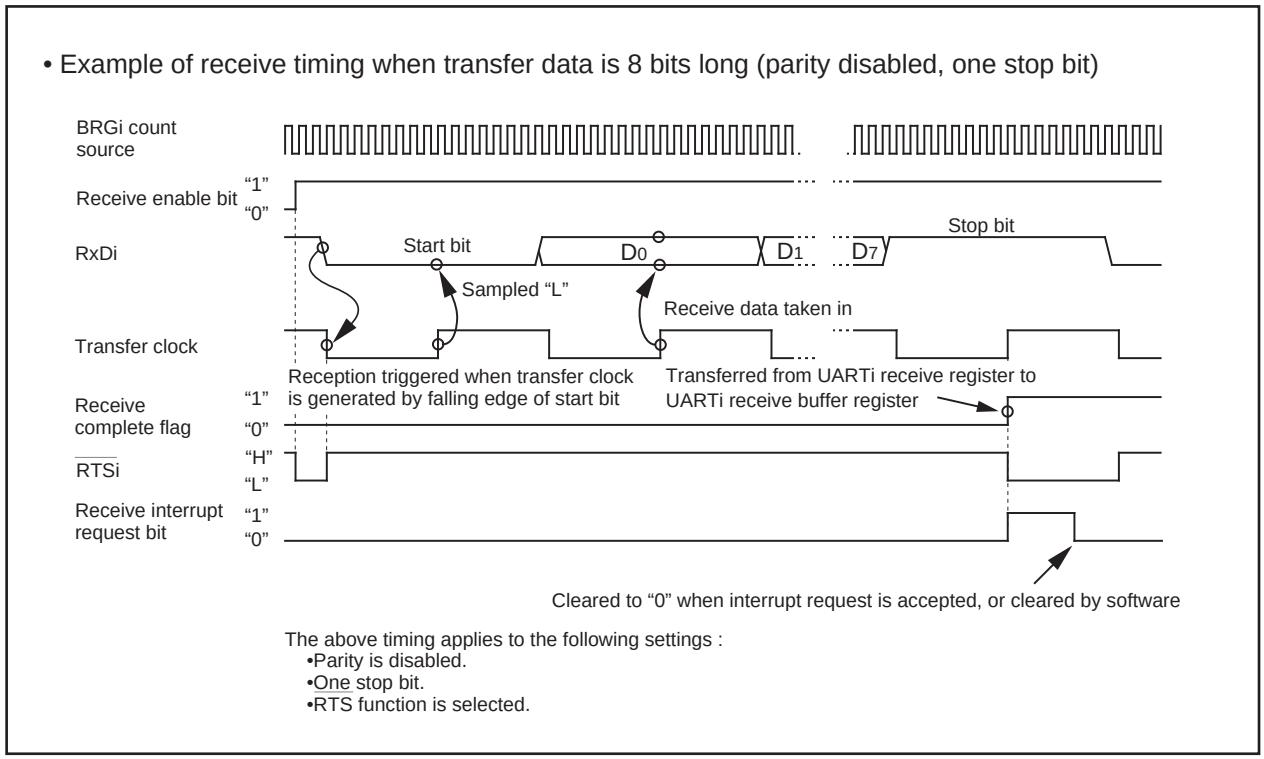



**Figure 1.19.17. Typical receive timing in UART mode**

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

Setting the $\overline{\text{CTS}}/\overline{\text{RTS}}$ separate bit (bit 6 of address $03B0_{16}$) to "1" inputs/outputs the $\overline{\text{CTS}}$ signal and $\overline{\text{RTS}}$ signal from different pins. Choose which to use, $\overline{\text{CTS}}$ or $\overline{\text{RTS}}$, by use of the $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 of address $03A4_{16}$). This function is effective in UART0 only. With this function chosen, the user cannot use the $\overline{\text{CTS}}/\overline{\text{RTS}}$ function. Set "0" both to the $\overline{\text{CTS}}/\overline{\text{RTS}}$ function select bit (bit 2 of address $03AC_{16}$) and to the $\overline{\text{CTS}}/\overline{\text{RTS}}$ disable bit (bit 4 of address $03AC_{16}$).

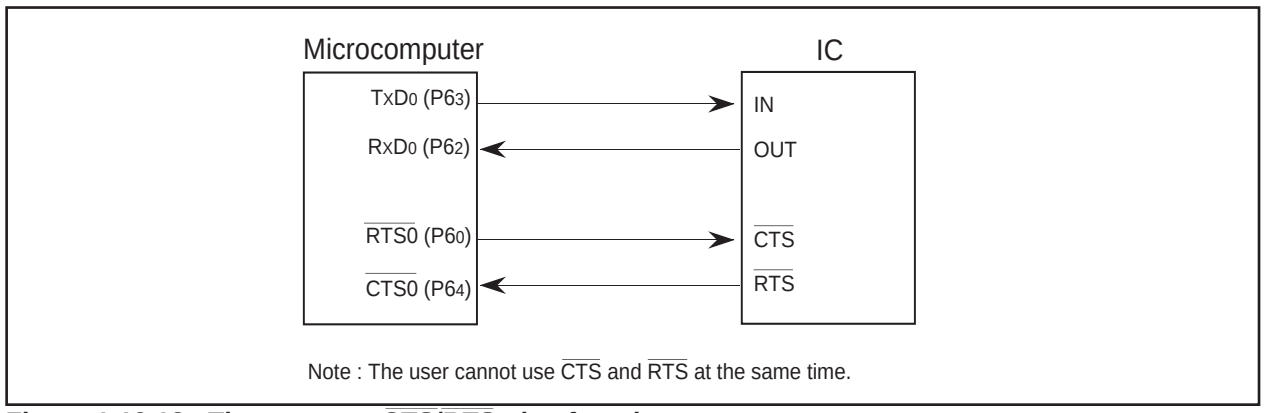



**Figure 1.19.18. The separate $\overline{\text{CTS}}/\overline{\text{RTS}}$ pins function usage**

**(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 $03A0_{16}$, $03A8_{16}$) 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".

**(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 1.19.19 shows the example of timing for switching serial data logic.

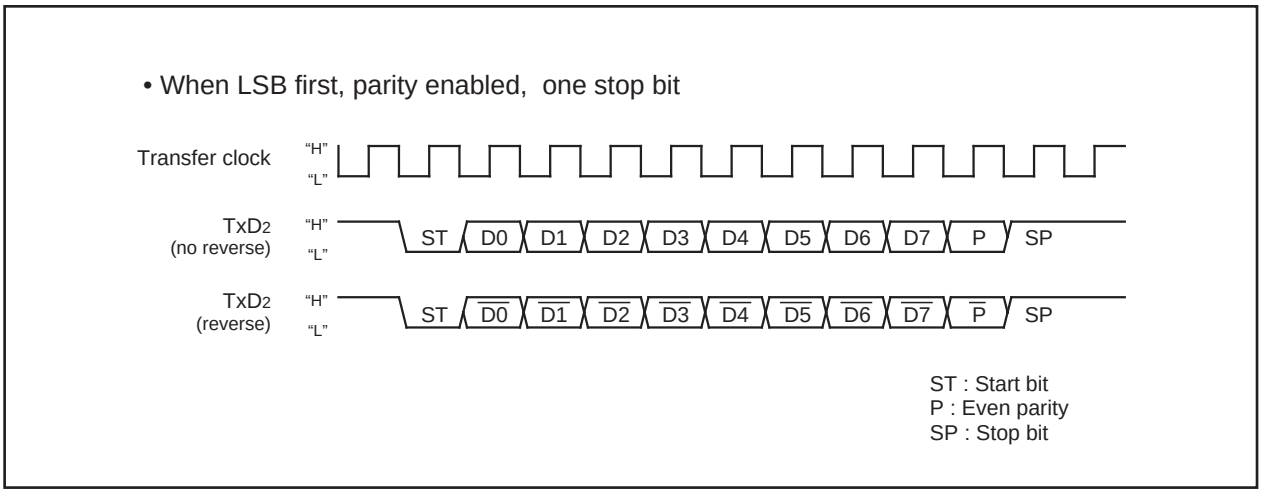



**Figure 1.19.19. Timing for switching serial data logic**

**(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 1.19.20 shows the example of detection timing of a buss collision (in UART mode).

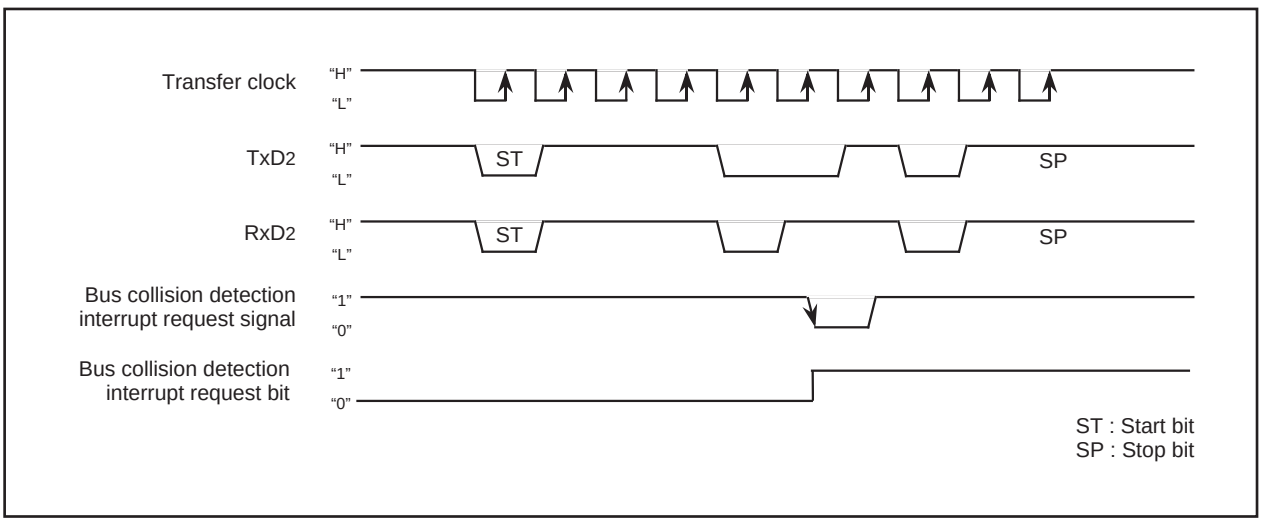



**Figure 1.19.20. Detection timing of a bus collision (in UART mode)**

## (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 1.19.8 shows the specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface).

**Table 1.19.8. Specifications of clock-asynchronous serial I/O mode (compliant with the SIM interface)**

| Item | Specification |
|---|---|
| Transfer data format | • Transfer data 8-bit UART mode (bit 2 through bit 0 of address $0378_{16}$ = "$101_2$")<br>• One stop bit (bit 4 of address $0378_{16}$ = "0")<br>• With the direct format chosen<br>Set parity to "even" (bit 5 and bit 6 of address $0378_{16}$ = "1" and "1" respectively)<br>Set data logic to "direct" (bit 6 of address $037D_{16}$ = "0").<br>Set transfer format to LSB (bit 7 of address $037C_{16}$ = "0").<br>• With the inverse format chosen<br>Set parity to "odd" (bit 5 and bit 6 of address $0378_{16}$ = "0" and "1" respectively)<br>Set data logic to "inverse" (bit 6 of address $037D_{16}$ = "1")<br>Set transfer format to MSB (bit 7 of address $037C_{16}$ = "1") |
| Transfer clock | • With the internal clock chosen (bit 3 of address $0378_{16}$ = "0") : fi / 16 (n + 1) (Note 1) : fi=$f_1$, $f_8$, $f_{32}$<br>• With an external clock chosen (bit 3 of address $0378_{16}$ = "1") : $f_{EXT}$ / 16 (n+1) (Note 1) (Note 2) |
| Transmission / reception control | • Disable the $\overline{CTS}$ and $\overline{RTS}$ function (bit 4 of address $037C_{16}$ = "1") |
| Other settings | • The sleep mode select function is not available for UART2<br>• Set transmission interrupt factor to "transmission completed" (bit 4 of address $037D_{16}$ = "1") |
| Transmission start condition | • To start transmission, the following requirements must be met:<br>- Transmit enable bit (bit 0 of address $037D_{16}$) = "1"<br>- Transmit buffer empty flag (bit 1 of address $037D_{16}$) = "0" |
| Reception start condition | • To start reception, the following requirements must be met:<br>- Reception enable bit (bit 2 of address $037D_{16}$) = "1"<br>- Detection of a start bit |
| Interrupt request generation timing | • When transmitting<br>When data transmission from the UART2 transfer register is completed (bit 4 of address $037D_{16}$ = "1")<br>• When receiving<br>When data transfer from the UART2 receive register to the UART2 receive buffer register is completed |
| Error detection | • Overrun error (see the specifications of clock-asynchronous serial I/O) (Note 3)<br>• Framing error (see the specifications of clock-asynchronous serial I/O)<br>• Parity error (see the specifications of clock-asynchronous serial I/O)<br>- On the reception side, an "L" level is output from the $TxD_2$ pin by use of the parity error signal output function (bit 7 of address $037D_{16}$ = "1") when a parity error is detected<br>- On the transmission side, a parity error is detected by the level of input to the $RxD_2$ pin when a transmission interrupt occurs<br>• The error sum flag (see the specifications of clock-asynchronous serial I/O) |

Note 1: 'n' denotes the value $00_{16}$ to $FF_{16}$ that is set to the UARTi bit rate generator.

Note 2: $f_{EXT}$ is input from the $CLK_2$ 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".

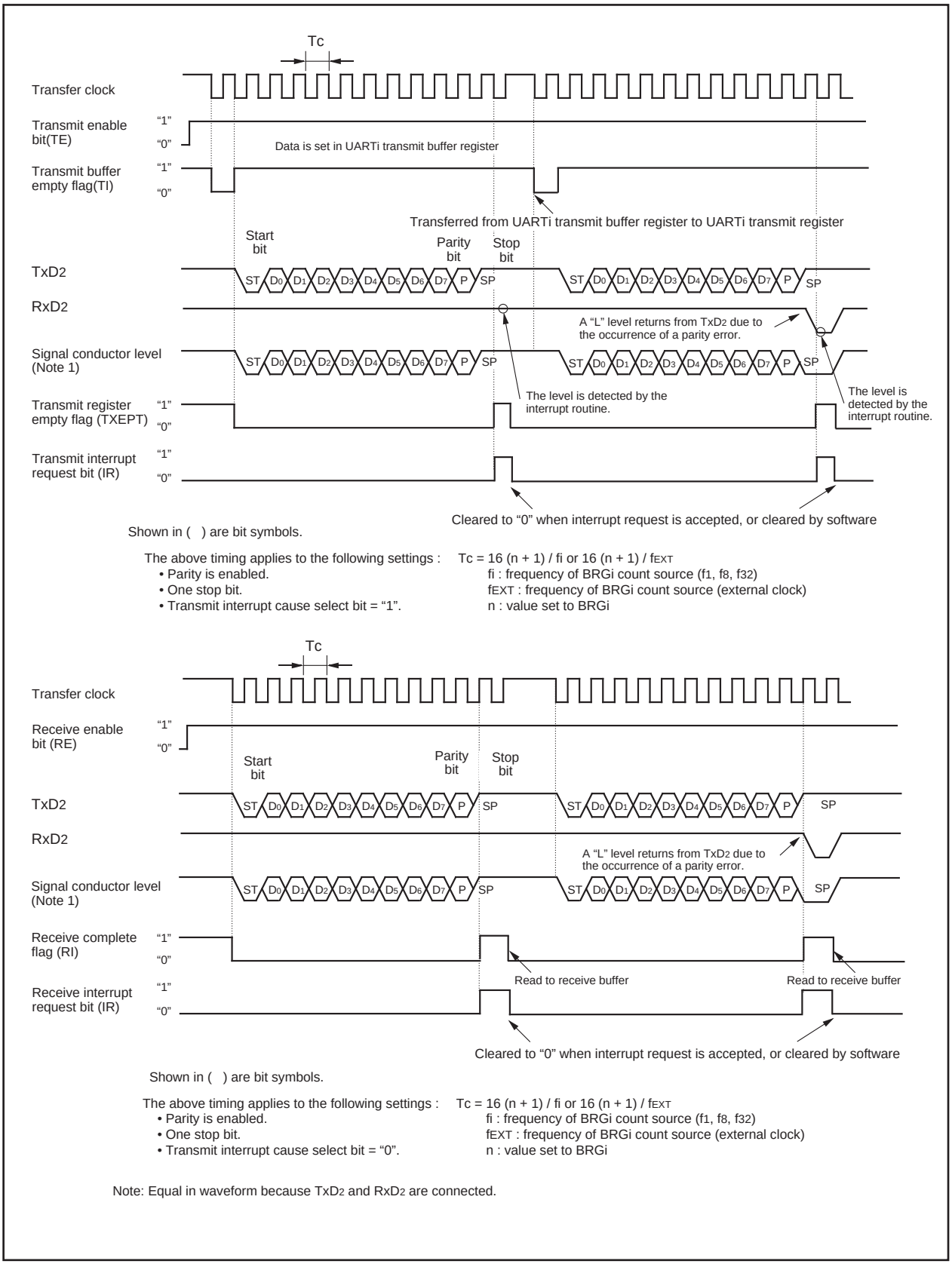



**Figure 1.19.21. Typical transmit/receive timing in UART mode (compliant with the SIM interface)**

**(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 1.19.22 shows the output timing of the parity error signal.

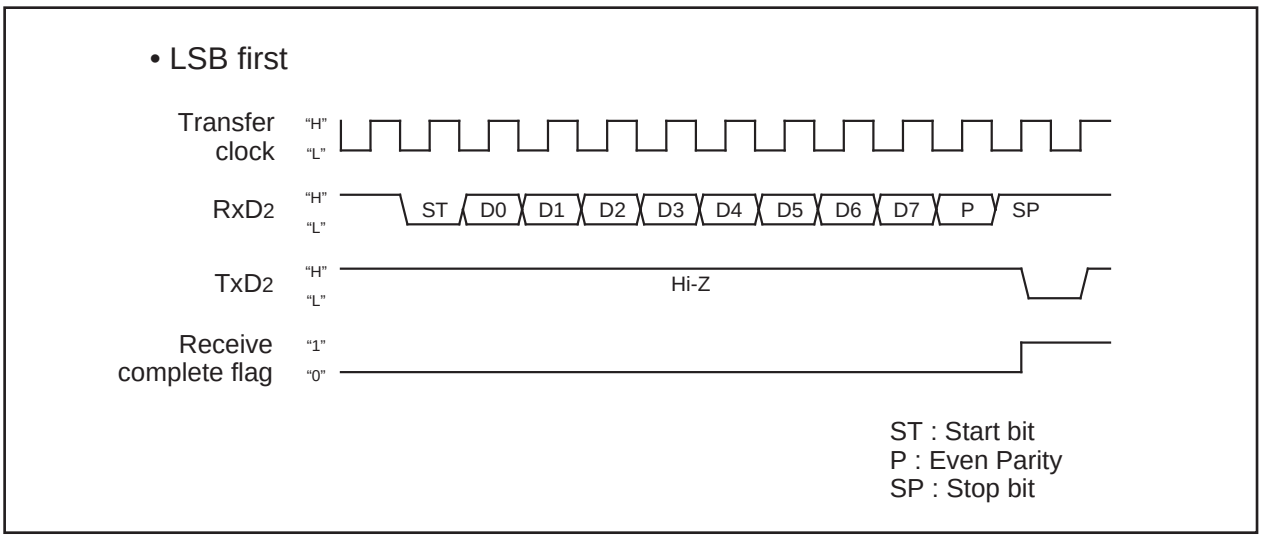



**Figure 1.19.22. Output timing of the parity error signal**

**(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 1.19.23 shows the SIM interface format.

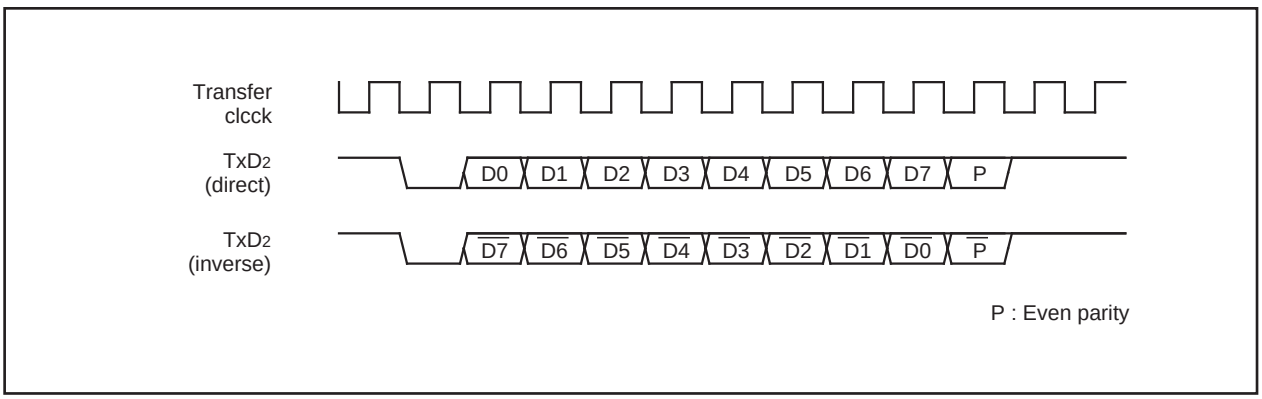



**Figure 1.19.23. SIM interface format**

Figure 1.19.24 shows the example of connecting the SIM interface. Connect $TxD_2$ and $RxD_2$ and apply pull-up.

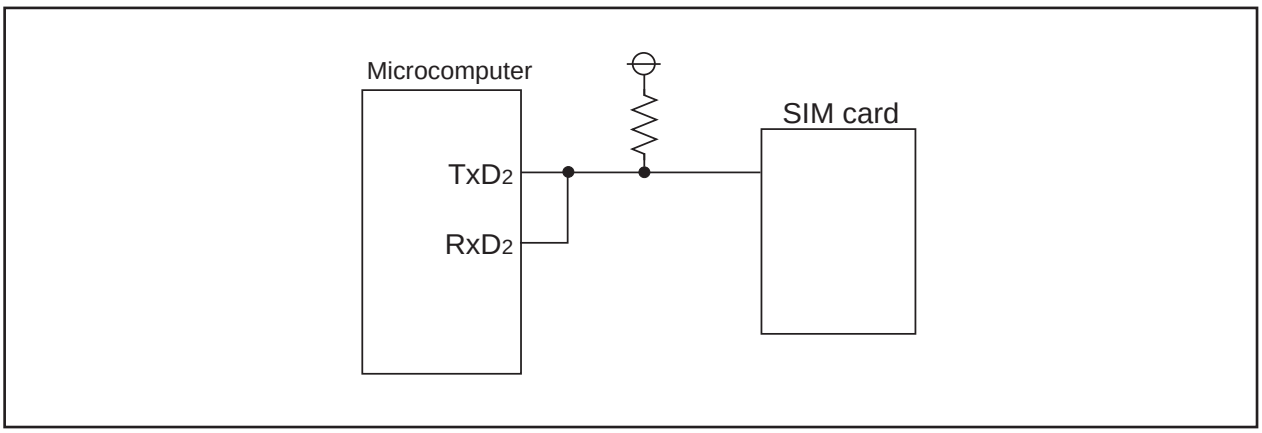



**Figure 1.19.24. Connecting the SIM interface**

## UART2 Special Mode Register

The UART2 special mode register (address $0377_{16}$) is used to control UART2 in various ways. Figure 1.19.25 shows the UART2 special mode register.

UART2 special mode register

b7 b6 b5 b4 b3 b2 b1 b0

Symbol: U2SMR | Address: $0377_{16}$ | When reset: $00_{16}$

| Bit symbol | Bit name | Function (During clock synchronous serial I/O mode) | Function (During UART mode) | R | W |
|---|---|---|---|---|---|
| IICM | $I^2C$ mode selection bit | 0 : Normal mode<br>1 : $I^2C$ mode | Must always be "0" | ○ | ○ |
| ABC | Arbitration lost detecting flag control bit | 0 : Update per bit<br>1 : Update per byte | Must always be "0" | ○ | ○ |
| BBS | Bus busy flag | 0 : STOP condition detected<br>1 : START condition detected | Must always be "0" | ○ | ○ (Note) |
| LSYN | SCLL sync output enable bit | 0 : Disabled<br>1 : Enabled | Must always be "0" | ○ | ○ |
| ABSCS | Bus collision detect sampling clock select bit | Must always be "0" | 0 : Rising edge of transfer clock<br>1 : Underflow signal of timer A0 | ○ | ○ |
| ACSE | Auto clear function select bit of transmit enable bit | Must always be "0" | 0 : No auto clear function<br>1 : Auto clear at occurrence of bus collision | ○ | ○ |
| SSS | Transmit start condition select bit | Must always be "0" | 0 : Ordinary<br>1 : Falling edge of RxD2 | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to this bit, write "0". The value, if read, turns out to be "0". | | | | — | — |

Note: Nothing but "0" may be written.

**Figure 1.19.25. UART2 special mode register**

**Table 1.19.8. Features in $I^2C$ mode**

| | Function | Normal mode | $I^2C$ mode (Note 1) |
|---|---|---|---|
| 1 | Factor of interrupt number 10 (Note 2) | Bus collision detection | Start condition detection or stop condition detection |
| 2 | Factor of interrupt number 15 (Note 2) | UART2 transmission | No acknowledgment detection (NACK) |
| 3 | Factor of interrupt number 16 (Note 2) | UART2 reception | Acknowledgment detection (ACK) |
| 4 | UART2 transmission output delay | Not delayed | Delayed |
| 5 | $P7_0$ at the time when UART2 is in use | $TxD_2$ (output) | SDA (input/output) (Note 3) |
| 6 | $P7_1$ at the time when UART2 is in use | $RxD_2$ (input) | SCL (input/output) |
| 7 | $P7_2$ at the time when UART2 is in use | $CLK_2$ | $P7_2$ |
| 8 | DMA1 factor at the time when 1 1 0 1 is assigned to the DMA request factor selection bits | UART2 reception | Acknowledgment detection (ACK) |
| 9 | Noise filter width | 15ns | 50ns |
| 10 | Reading $P7_1$ | Reading the terminal when 0 is assigned to the direction register | Reading the terminal regardless of the value of the direction register |
| 11 | Initial value of UART2 output | H level (when 0 is assigned to the CLK polarity select bit) | The value set in latch $P7_0$ when the port is selected |

Note 1: Make the settings given below when $I^2C$ mode is in use.
Set 0 1 0 in bits 2, 1, 0 of the UART2 transmission/reception mode register.
Disable the RTS/CTS function. Choose the LSB First function.
Note 2: Follow the steps given below to switch from a factor to another.
1. Disable the interrupt of the corresponding number.
2. Switch from a factor to another.
3. Reset the interrupt request flag of the corresponding number.
4. Set an interrupt level of the corresponding number.
Note 3: Set an initial value of SDA transmission output when serial I/O is invalid.

In the first place, the control bits related to the $I^2C$ bus(simplified $I^2C$ bus) interface are explained.
Bit 0 of the UART special mode register ($037_{16}$) is used as the $I^2C$ mode selection bit.
Setting "1" in the $I^2C$ mode select bit (bit 0) goes the circuit to achieve the $I^2C$ bus interface effective.
Table 1.19.9 shows the relation between the $I^2C$ mode select bit and respective control workings.
Since this function uses clock-synchronous serial I/O mode, set this bit to "0" in UART mode.

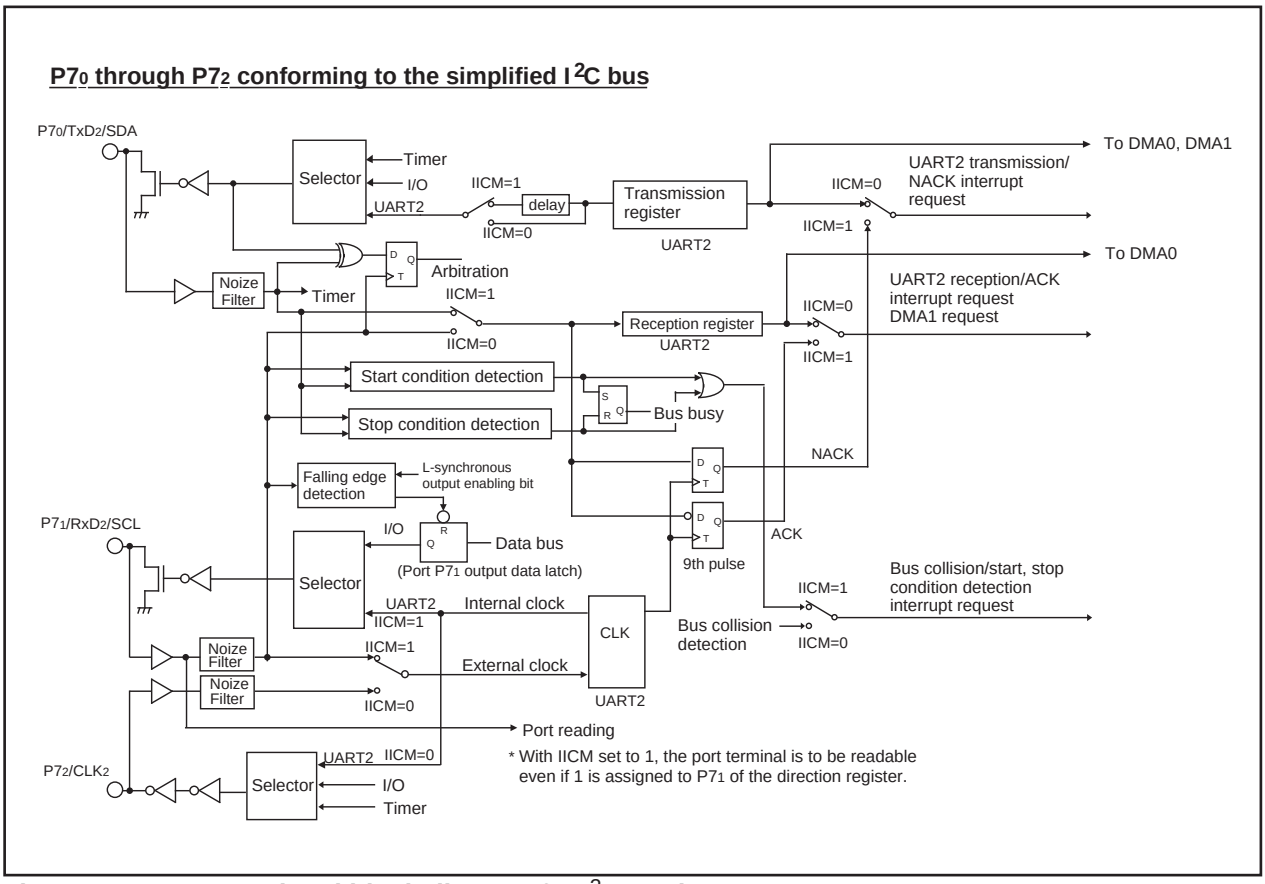



**Figure 1.19.26. Functional block diagram for $I^2C$ mode**

Figure 1.19.26 shows the functional block diagram for $I^2C$ mode. Setting "1" in the $I^2C$ mode selection bit (IICM) causes ports $P7_0$, $P7_1$, and $P7_2$ to work as data transmission-reception terminal SDA, clock input-output terminal SCL, and port $P7_2$ respectively. A delay circuit is added to the SDA transmission output, so the SDA output changes after SCL fully goes to "L". An attempt to read Port $P7_1$ (SCL) results in getting the terminal's level regardless of the content of the port direction register. The initial value of SDA transmission output in this mode goes to the value set in port $P7_0$. The interrupt factors of the bus collision detection interrupt, UART2 transmission interrupt, and of UART2 reception interrupt turn to the start/stop condition detection interrupt, acknowledgment non-detection interrupt, and acknowledgment detection interrupt respectively.
The start condition detection interrupt refers to the interrupt that occurs when the falling edge of the SDA terminal ($P7_0$) is detected with the SCL terminal ($P7_1$) staying "H". The stop condition detection interrupt refers to the interrupt that occurs when the rising edge of the SDA terminal ($P7_0$) is detected with the SCL terminal ($P7_1$) staying "H". The bus busy flag (bit 2 of the UART2 special mode register) is set to "1" by the start condition detection, and set to "0" by the stop condition detection.

The acknowledgment non-detection interrupt refers to the interrupt that occurs when the SDA terminal level is detected still staying “H” at the rising edge of the 9th transmission clock. The acknowledgment detection interrupt refers to the interrupt that occurs when SDA terminal’s level is detected already went to “L” at the 9th transmission clock. Also, assigning 1 1 0 1 (UART2 reception) to the DMA1 request factor select bits provides the means to start up the DMA transfer by the effect of acknowledgment detection.

Bit 1 of the UART2 special mode register ($0377_{16}$) is used as the arbitration loss detecting flag control bit. Arbitration means the act of detecting the nonconformity between transmission data and SDA terminal data at the timing of the SCL rising edge. This detecting flag is located at bit 3 of the UART2 reception buffer register ($037F_{16}$), and “1” is set in this flag when nonconformity is detected. Use the arbitration lost detecting flag control bit to choose which way to use to update the flag, bit by bit or byte by byte. When setting this bit to “1” and updated the flag byte by byte if nonconformity is detected, the arbitration lost detecting flag is set to “1” at the falling edge of the 9th transmission clock.

If update the flag byte by byte, must judge and clear (“0”) the arbitration lost detecting flag after completing the first byte acknowledge detect and before starting the next one byte transmission.

Bit 3 of the UART2 special mode register is used as SCL- and L-synchronous output enable bit. Setting this bit to “1” goes the $P7_1$ data register to “0” in synchronization with the SCL terminal level going to “L”.

Some other functions added are explained here. Figure 1.19.27 shows their workings.

Bit 4 of the UART2 special mode register is used as the bus collision detect sampling clock select bit. The bus collision detect interrupt occurs when the $RxD_2$ level and $TxD_2$ level do not match, but the nonconformity is detected in synchronization with the rising edge of the transfer clock signal if the bit is set to "0". If this bit is set to "1", the nonconformity is detected at the timing of the overflow of timer A0 rather than at the rising edge of the transfer clock.

Bit 5 of the UART2 special mode register is used as the auto clear function select bit of transmit enable bit. Setting this bit to "1" automatically resets the transmit enable bit to "0" when "1" is set in the bus collision detect interrupt request bit (nonconformity).

Bit 6 of the UART2 special mode register is used as the transmit start condition select bit. Setting this bit to "1" starts the TxD transmission in synchronization with the falling edge of the RxD terminal.

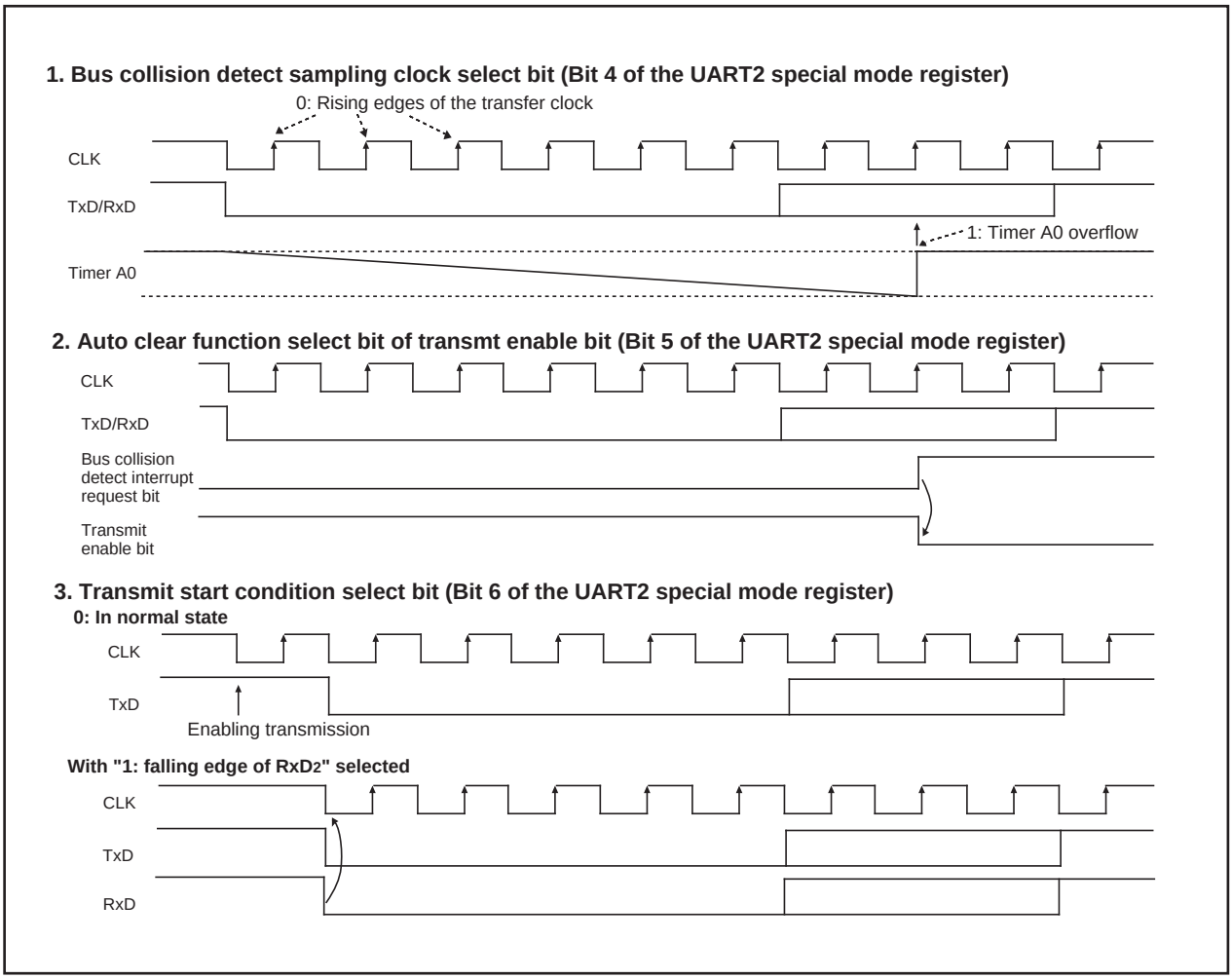



**Figure 1.19.27. Some other functions added**

## UART2 Special Mode Register 2

UART2 special mode register 2 (address $0376_{16}$) is used to further control UART2 in $I^2C$ mode, but is absent from M30622EC. Figure 1.19.28 shows the UART2 special mode register 2.

UART2 special mode register 2

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: U2SMR2 — Address: $0376_{16}$ — When reset: $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| IICM2 | $I^2C$ mode selection bit 2 | Refer to Table 1.19.10 | ○ | ○ |
| CSC | Clock-synchronous bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC | SCL wait output bit | 0 : Disabled<br>1 : Enabled | ○ | ○ (Note) |
| ASL | SDA output stop bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| STAC | UART2 initialization bit | 0 : Disabled<br>1 : Enabled | ○ | ○ |
| SWC2 | SCL wait output bit 2 | 0: UART2 clock<br>1: 0 output | ○ | ○ |
| SDHI | SDA output disable bit | 0: Enabled<br>1: Disabled (high impedance) | ○ | ○ |
| SHTC | Start/stop condition control bit | Set this bit to "1" in $I^2C$ mode (refer to Table 1.19.11) | ○ | ○ |

**Figure 1.19.28. UART2 special mode register 2**

## UART2 Special Mode Register 2

Bit 0 of the UART2 special mode register 2 (address $0376_{16}$) is used as the $I^2C$ mode selection bit 2. Table 1.19.10 shows the types of control to be changed by $I^2C$ mode selection bit 2 when the $I^2C$ mode selection bit is set to "1". Table 1.19.11 shows the timing characteristics of detecting the start condition and the stop condition. Set the start/stop condition control bit (bit 7 of UART2 special mode register 2) to "1" in $I^2C$ mode.

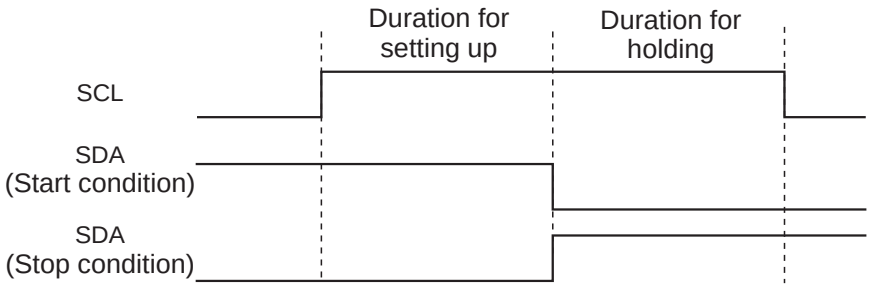
**Table 1.19.10. Functions changed by $I^2C$ mode selection bit 2**

| | Function | IICM2 = 0 | IICM2 = 1 |
|---|---|---|---|
| 1 | Factor of interrupt number 15 | No acknowledgment detection (NACK) | UART2 transmission (the rising edge of the final bit of the clock) |
| 2 | Factor of interrupt number 16 | Acknowledgment detection (ACK) | UART2 reception (the falling edge of the final bit of the clock) |
| 3 | DMA1 factor at the time when 1 1 0 1 is assigned to the DMA request factor selection bits | Acknowledgment detection (ACK) | UART2 reception (the falling edge of the final bit of the clock) |
| 4 | Timing for transferring data from the UART2 reception shift register to the reception buffer. | The rising edge of the final bit of the reception clock | The falling edge of the final bit of the reception clock |
| 5 | Timing for generating a UART2 reception/ACK interrupt request | The rising edge of the final bit of the reception clock | The falling edge of the final bit of the reception clock |

**Table 1.19.11. Timing characteristics of detecting the start condition and the stop condition**

| |
|---|
| 3 to 6 cycles < duration for setting-up (Note2) |
| 3 to 6 cycles < duration for holding (Note2) |

Note 1 : When the start/stop condition count bit is "1" .
Note 2 : "cycles" is in terms of the input oscillation frequency f($X_{IN}$) of the main clock.



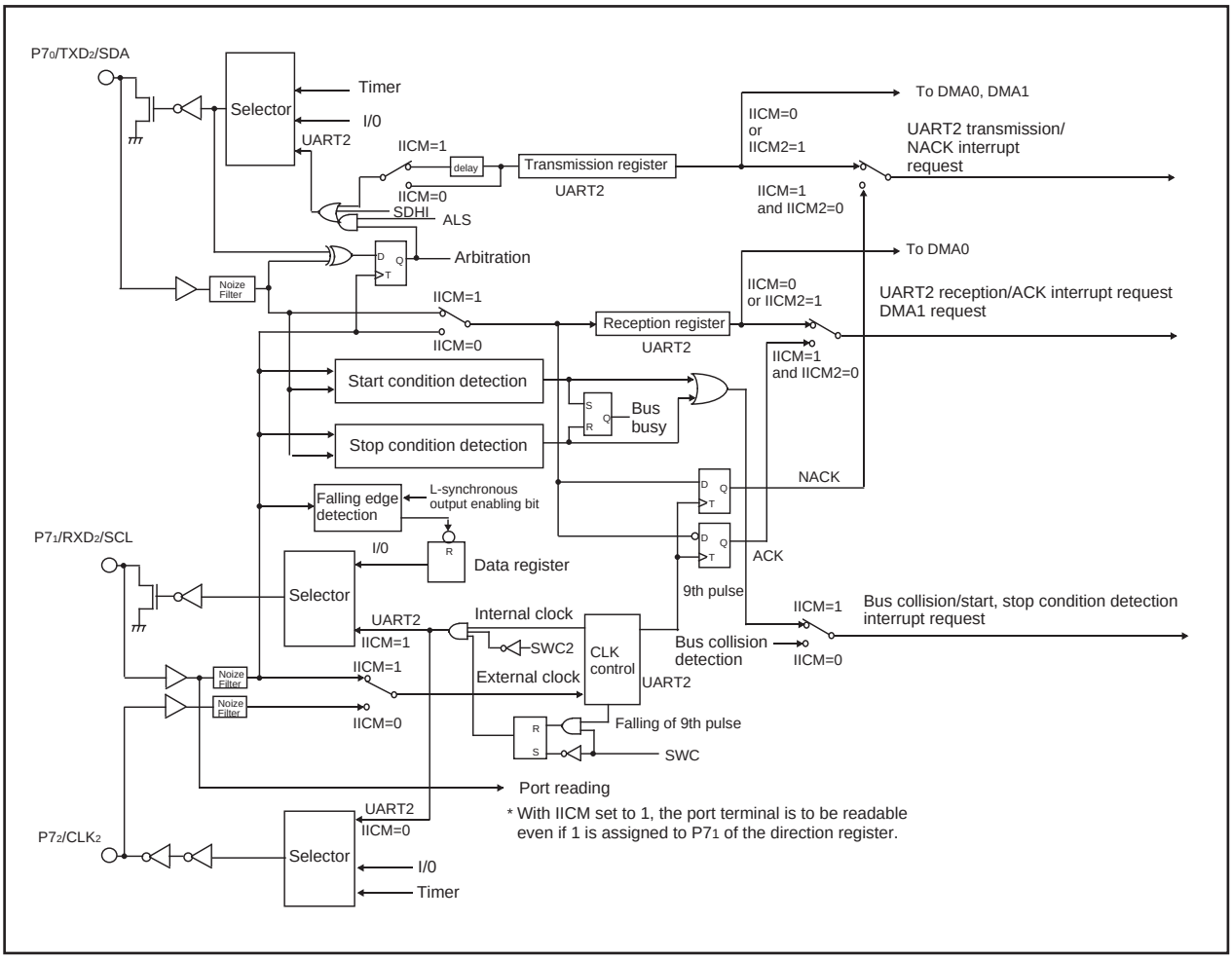



**Figure 1.19.29. Functional block diagram for I²C mode**

Functions available in I²C mode are shown in Figure 1.19.29 — a functional block diagram.

Bit 3 of the UART2 special mode register 2 (address 0376₁₆) is used as the SDA output stop bit. Setting this bit to "1" causes an arbitration loss to occur, and the SDA pin turns to high-impedance state the instant when the arbitration loss detection flag is set to "1".

Bit 1 of the UART2 special mode register 2 (address 0367₁₆) is used as the clock synchronization bit. With this bit set to "1" at the time when the internal SCL is set to "H", the internal SCL turns to "L" if the falling edge is found in the SCL pin; and the baud rate generator reloads the set value, and start counting within the "L" interval. When the internal SCL changes from "L" to "H" with the SCL pin set to "L", stops counting the baud rate generator, and starts counting it again when the SCL pin turns to "H". Due to this function, the UART2 transmission-reception clock becomes the logical product of the signal flowing through the internal SCL and that flowing through the SCL pin. This function operates over the period from the moment earlier by a half cycle than falling edge of the UART2 first clock to the rising edge of the ninth bit. To use this function, choose the internal clock for the transfer clock.

Bit 2 of the UART2 special mode register 2 (0376₁₆) is used as the SCL wait output bit. Setting this bit to "1" causes the SCL pin to be fixed to "L" at the falling edge of the ninth bit of the clock. Setting this bit to "0" frees the output fixed to "L".

## UART2 Special Mode Register 2

Bit 4 of the UART2 special mode register 2 (address $0376_{16}$) is used as the UART2 initialization bit. Setting this bit to "1", and when the start condition is detected, the microcomputer operates as follows.

(1) The transmission shift register is initialized, and the content of the transmission register is transferred to the transmission shift register. This starts transmission by dealing with the clock entered next as the first bit. The UART2 output value, however, doesn't change until the first bit data is output after the entrance of the clock, and remains unchanged from the value at the moment when the microcomputer detected the start condition.

(2) The reception shift register is initialized, and the microcomputer starts reception by dealing with the clock entered next as the first bit.

(3) The SCL wait output bit turns to "1". This turns the SCL pin to "L" at the falling edge of the ninth bit of the clock.

Starting to transmit/receive signals to/from UART2 using this function doesn't change the value of the transmission buffer empty flag. To use this function, choose the external clock for the transfer clock.

Bit 5 of the UART2 special mode register 2 ($0376_{16}$) is used as the SCL pin wait output bit 2. Setting this bit to "1" with the serial I/O specified allows the user to forcibly output an "L" from the SCL pin even if UART2 is in operation. Setting this bit to "0" frees the "L" output from the SCL pin, and the UART2 clock is input/output.

Bit 6 of the UART2 special mode register 2 ($0376_{16}$) is used as the SDA output enable bit. Setting this bit to "1" forces the SDA pin to turn to the high-impedance state. Refrain from changing the value of this bit at the rising edge of the UART2 transfer clock. There can be instances in which arbitration lost detection flag is turned on.

## S I/O3, 4

S I/O3 and S I/O4 are exclusive clock-synchronous serial I/Os.

Figure 1.19.30 shows the S I/O3, 4 block diagram, and Figure 1.19.31 shows the S I/O3, 4 control register.

Table 1.19.12 shows the specifications of S I/O3, 4.

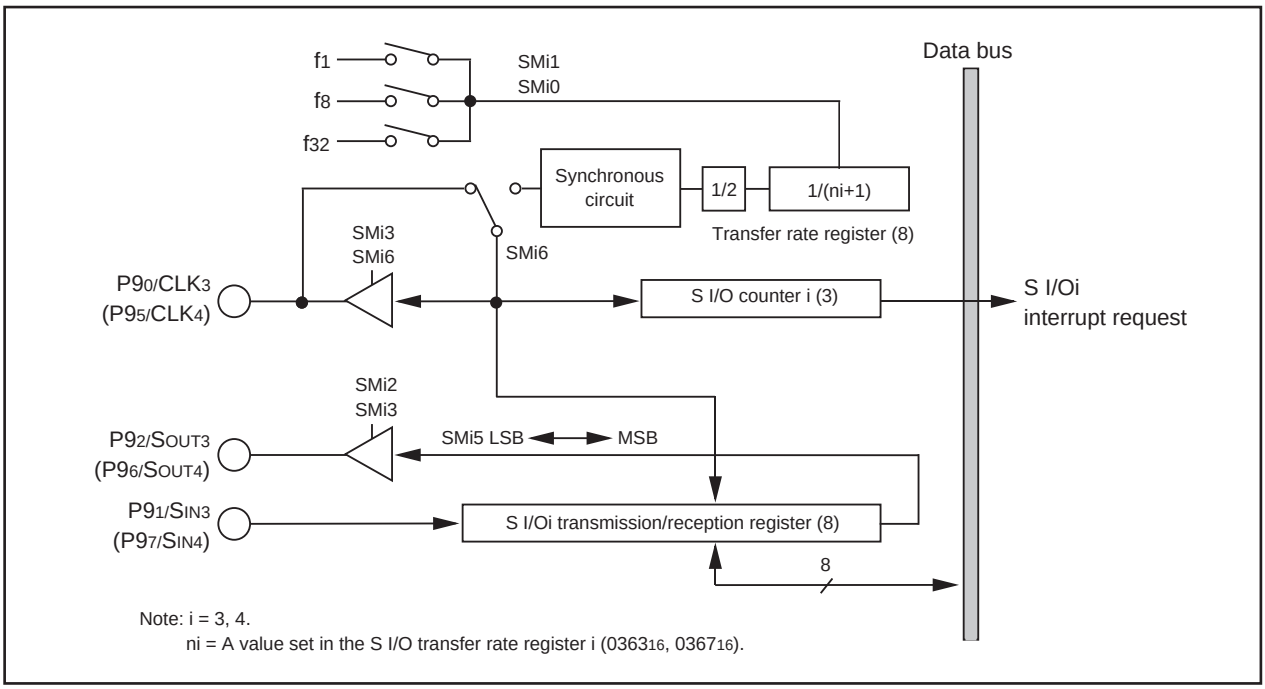



**Figure 1.19.30. S I/O3, 4 block diagram**

S I/Oi control register (i = 3, 4) (Note 1)

b7 b6 b5 b4 b3 b2 b1 b0

| Symbol | Address | When reset |
|---|---|---|
| SiC | 036216, 036616 | 4016 |

| Bit symbol | Bit name | Description | R | W |
|---|---|---|---|---|
| SMi0 | Internal synchronous clock select bit | b1 b0<br>0 0 : Selecting f1<br>0 1 : Selecting f8<br>1 0 : Selecting f32<br>1 1 : Not to be used | ○ | ○ |
| SMi1 | | | ○ | ○ |
| SMi2 | SOUTi high impedance control bit | 0 : SOUTi output<br>1 : SOUTi high impedance | ○ | ○ |
| SMi3 | S I/Oi port select bit (Note 2) | 0 : Input-output port<br>1 : SOUTi output, CLK function | ○ | ○ |
| Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be “0”. | | | — | — |
| SMi5 | Transfer direction lect bit | 0 : LSB first<br>1 : MSB first | ○ | ○ |
| SMi6 | Synchronous clock select bit (Note 2) | 0 : External clock<br>1 : Internal clock | ○ | ○ |
| SMi7 | SOUTi initial value set bit | Effective when SMi3 = 0<br>0 : L output<br>1 : H output | ○ | ○ |

Note 1: Set "1" in bit 2 of the protection register (000A16) in advance to write to the S I/Oi control register (i = 3, 4).

Note 2: When set “0” to SMi3 (i = 3, 4) and select input - output port, set “1” to SMi6 (i = 3, 4) and select internal clock, or input “H” to P90 and P95.

**Figure 1.19.31. S I/O3, 4 control register**

**Table 1.19.12. Specifications of S I/O3, 4**

| Item | Specifications |
|---|---|
| Transfer data format | • Transfer data length: 8 bits |
| Transfer clock | • With the internal clock selected (bit 6 of $0362_{16}$, $0366_{16}$ = "1"): f1/2(ni+1), f8/2(ni+1), f32/2(ni+1) (Note 1)Å@<br>• With the external clock selected (bit 6 of $0362_{16}$, $0366_{16}$ = 0):Input from the CLKi terminal (Note 2) |
| Conditions for transmission/ reception start | • To start transmit/reception, the following requirements must be met:<br>- Select the synchronous clock (use bit 6 of 036216, 036616).<br>Select a frequency dividing ratio if the internal clock has been selected (use bits 0 and 1 of $0362_{16}$, $0366_{16}$).<br>- SOUTi initial value set bit (use bit 7 of $0362_{16}$, $0366_{16}$)= 1.<br>- S I/Oi port select bit (bit 3 of $0362_{16}$, $0366_{16}$) = 1.<br>- Select the transfer direction (use bit 5 of $0362_{16}$, $0366_{16}$)<br>• To use S I/Oi interrupts, the following requirements must be met:<br>- S I/Oi interrupt request bit (bit 3 of $0049_{16}$, $0048_{16}$) = 0. |
| Interrupt request generation timing | • An interrupt occurs after counting eight transfer clock either in transmitting or receiving data. (Note 3)<br>- In transmitting: At the time data transfer from the S I/Oi transmission/reception register finishes.<br>- In receiving: At the time data reception to the S I/Oi transmission/reception register finishes. |
| Select function | • LSB first or MSB first selection<br>Whether transmission/reception begins with bit 0 or bit 7 can be selected. |

Note 1: n is a value from $00_{16}$ through $FF_{16}$ set in the S I/Oi transfer rate register (i = 3, 4).

Note 2: With the external clock selected:

- To write to the S I/Oi transmission/reception register ($0360_{16}$, $0364_{16}$), enter the "H" level to the CLKi terminal. Also, to write to the bit 7 (SOUTi initial value set bit) of SI/Oi control register ($0362_{16}$, $0366_{16}$), enter the "H" level to the CLKi terminal.
- The S I/Oi circuit keeps on with the shift operation as long as the synchronous clock is entered in it, so stop the synchronous clock at the instant when it counts to eight. The internal clock, if selected, automatically stops.

Note 3: If the internal clock is used for the synchronous clock, the transfer clock signal stops at the "H" state.

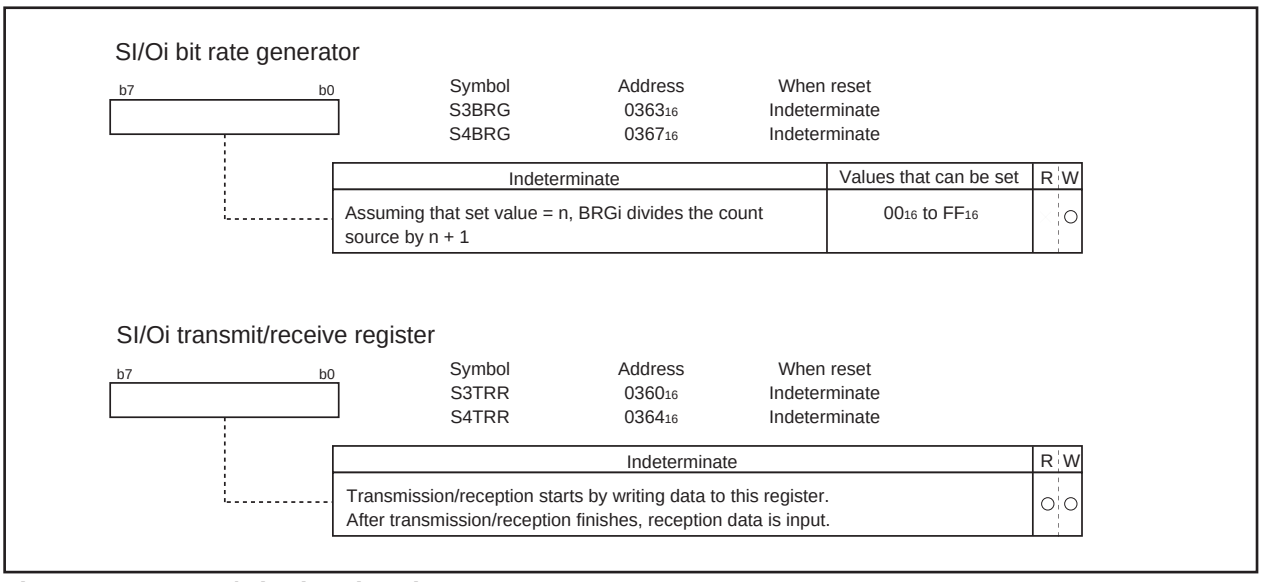
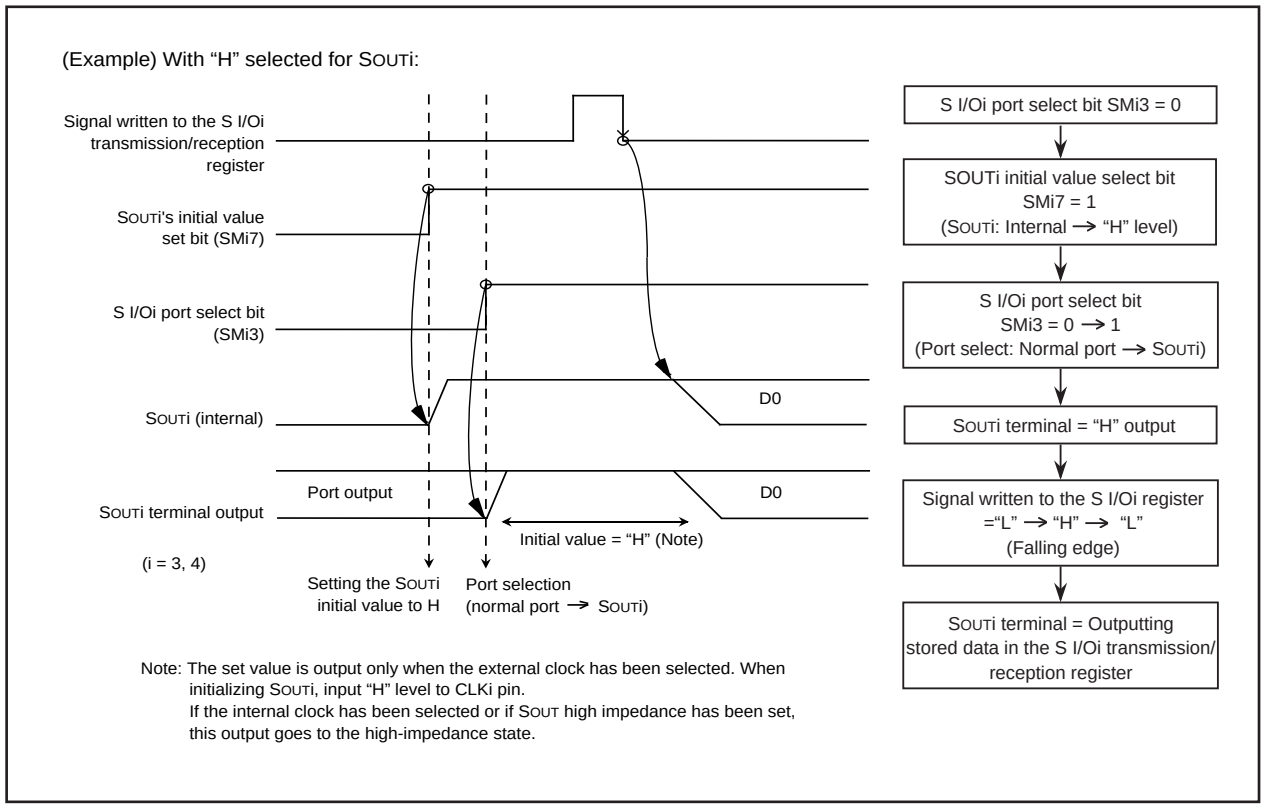



**Figure 1.19.32. SI/Oi related register**

### ■ Functions for setting an SOUTi initial value

In carrying out transmission, the output level of the SOUTi terminal as it is before transmitting 1-bit data can be set either to “H” or to “L”. Figure 1.19.33 shows the timing chart for setting an SOUTi initial value and how to set it.

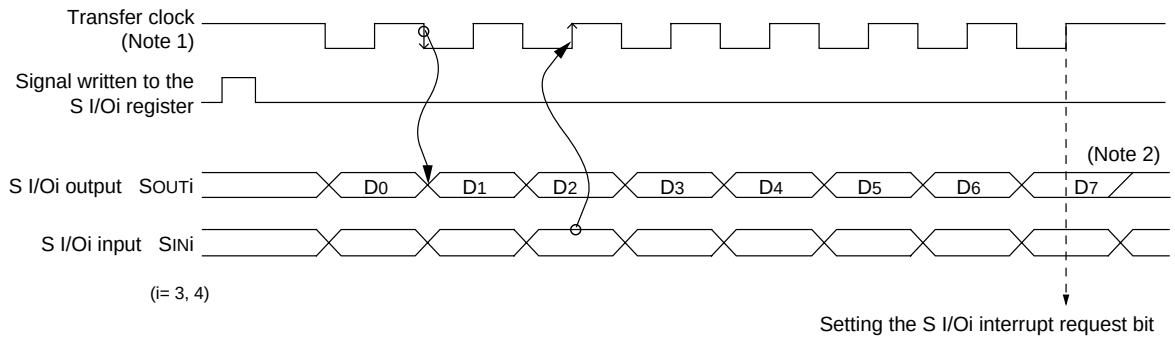



Note: The set value is output only when the external clock has been selected. When initializing SOUTi, input “H” level to CLKi pin.
If the internal clock has been selected or if SOUT high impedance has been set, this output goes to the high-impedance state.

**Figure 1.19.33. Timing chart for setting SOUTi’s initial value and how to set it**

### ■ S I/Oi operation timing

Figure 1.19.34 shows the S I/Oi operation timing




Note 1: With the internal clock selected for the transfer clock, the frequency dividing ratio can be selected using bits 0 and 1 of the S I/Oi control register. (No frequency division, 8-division frequency, 32-division frequency.)
Note 2: With the internal clock selected for the transfer clock, the SOUTi terminal becomes to the high-impedance state after the transfer finishes.

**Figure 1.19.34. S I/Oi operation timing chart**

## 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$_0$ to P10$_7$, P9$_5$, and P9$_6$ 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 03D7$_{16}$) can be used to isolate the resistance ladder of the A-D converter from the reference voltage input pin ($V_{REF}$) when the A-D converter is not used. Doing so stops any current flowing into the resistance ladder from $V_{REF}$, reducing the power dissipation. When using the A-D converter, start A-D conversion only after setting bit 5 of 03D7$_{16}$ to connect $V_{REF}$. 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 1.20.1 shows the performance of the A-D converter. Figure 1.20.1 shows the block diagram of the A-D converter, and Figures 1.20.2 and 1.20.3 show the A-D converter-related registers.

**Table 1.20.1. Performance of A-D converter**

| Item | Performance |
|---|---|
| Method of A-D conversion | Successive approximation (capacitive coupling amplifier) |
| Analog input voltage (Note 1) | 0V to AV$_{CC}$ ($V_{CC}$) |
| Operating clock $\phi$AD (Note 2) | $V_{CC}$ = 5V $f_{AD}$/divide-by-2 of $f_{AD}$/divide-by-4 of $f_{AD}$, $f_{AD}$=f($X_{IN}$)<br>$V_{CC}$ = 3V divide-by-2 of $f_{AD}$/divide-by-4 of $f_{AD}$, $f_{AD}$=f($X_{IN}$) |
| Resolution | 8-bit or 10-bit (selectable) |
| Absolute precision | $V_{CC}$ = 5V • Without sample and hold function<br>±3LSB<br>• With sample and hold function (8-bit resolution)<br>±2LSB<br>• With sample and hold function (10-bit resolution)<br>AN$_0$ to AN$_7$ input : ±3LSB<br>ANEX0 and ANEX1 input (including mode in which external operation amp is connected) : ±7LSB<br>$V_{CC}$ = 3V • Without sample and hold function (8-bit resolution)<br>±2LSB |
| Operating modes | One-shot mode, repeat mode, single sweep mode, repeat sweep mode 0, and repeat sweep mode 1 |
| Analog input pins | 8pins (AN$_0$ to AN$_7$) + 2pins (ANEX0 and ANEX1) |
| A-D conversion start condition | • Software trigger<br>A-D conversion starts when the A-D conversion start flag changes to “1”<br>• External trigger (can be retriggered)<br>A-D conversion starts when the A-D conversion start flag is “1” and the $\overline{ADTRG}$/P9$_7$ input changes from “H” to “L” |
| Conversion speed per pin | • Without sample and hold function<br>8-bit resolution: 49 $\phi$AD cycles, 10-bit resolution: 59 $\phi$AD cycles<br>• With sample and hold function<br>8-bit resolution: 28 $\phi$AD cycles, 10-bit resolution: 33 $\phi$AD cycles |

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

Note 2: Divide the frequency if f($X_{IN}$) exceeds 10MHz, and make $\phi$AD frequency equal to 10MHz.
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.

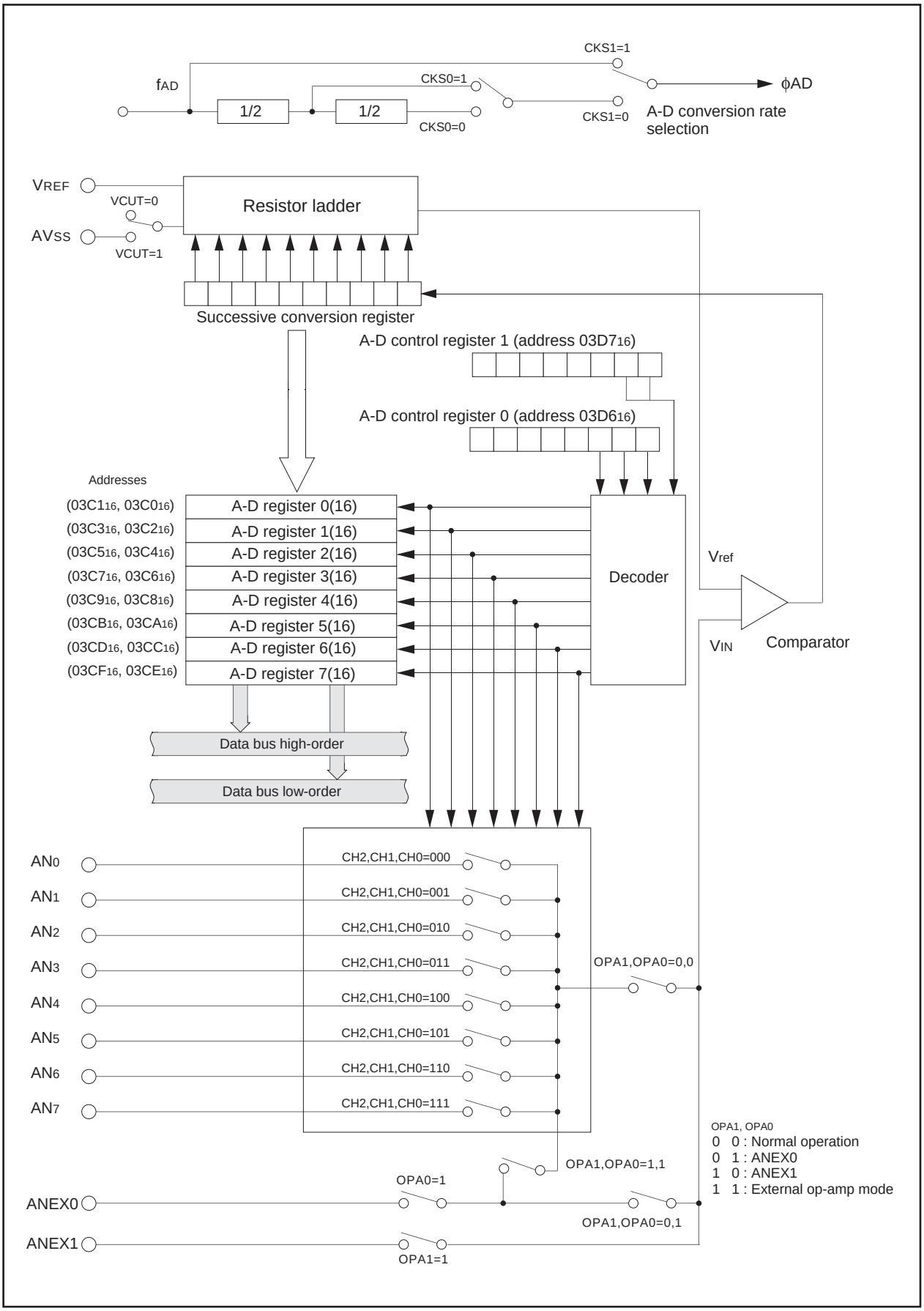



**Figure 1.20.1. Block diagram of A-D converter**

A-D control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | When reset |
|---|---|---|
| ADCON0 | $03D6_{16}$ | $00000XXX_2$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : $AN_0$ is selected<br>0 0 1 : $AN_1$ is selected<br>0 1 0 : $AN_2$ is selected<br>0 1 1 : $AN_3$ is selected<br>1 0 0 : $AN_4$ is selected<br>1 0 1 : $AN_5$ is selected<br>1 1 0 : $AN_6$ is selected<br>1 1 1 : $AN_7$ is selected (Note 2) | ○ | ○ |
| CH1 | | | ○ | ○ |
| CH2 | | | ○ | ○ |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>0 0 : One-shot mode<br>0 1 : Repeat mode<br>1 0 : Single sweep mode<br>1 1 : Repeat sweep mode 0<br>Repeat sweep mode 1 (Note 2) | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $\overline{AD_{TRG}}$ trigger | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | ○ | ○ |
| CKS0 | Frequency select bit 0 | 0 : $f_{AD}/4$ is selected<br>1 : $f_{AD}/2$ is selected | ○ | ○ |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: When changing A-D operation mode, set analog input pin again.

A-D control register 1 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | When reset |
|---|---|---|
| ADCON1 | $03D7_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When single sweep and repeat sweep mode 0 are selected<br>b1 b0<br>0 0 : $AN_0$, $AN_1$ (2 pins)<br>0 1 : $AN_0$ to $AN_3$ (4 pins)<br>1 0 : $AN_0$ to $AN_5$ (6 pins)<br>1 1 : $AN_0$ to $AN_7$ (8 pins) | ○ | ○ |
| SCAN1 | | When repeat sweep mode 1 is selected<br>b1 b0<br>0 0 : $AN_0$ (1 pin)<br>0 1 : $AN_0$, $AN_1$ (2 pins)<br>1 0 : $AN_0$ to $AN_2$ (3 pins)<br>1 1 : $AN_0$ to $AN_3$ (4 pins) | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1<br>1 : Repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit 1 | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}$ is selected | ○ | ○ |
| VCUT | Vref connect bit | 0 : Vref not connected<br>1 : Vref connected | ○ | ○ |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | ○ | ○ |
| OPA1 | | | ○ | ○ |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

**Figure 1.20.2. A-D converter-related registers (1)**

A-D control register 2 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | X | X | X | 0 | 0 | 0 | |

Symbol: ADCON2 | Address: $03D4_{16}$ | When reset: $0000XXX0_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SMP | A-D conversion method select bit | 0 : Without sample and hold<br>1 : With sample and hold | ○ | ○ |
| Reserved bit | | Always set to “0” | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | — | — |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

A-D register i

Symbol: ADi(i=0 to 7) | Address: $03C0_{16}$ to $03CF_{16}$ | When reset: Indeterminate

(b15) b7 (b8) b0 b7 b0

| Function | R | W |
|---|---|---|
| Eight low-order bits of A-D conversion result | ○ | × |
| • During 10-bit mode<br>Two high-order bits of A-D conversion result | ○ | × |
| • During 8-bit mode<br>When read, the content is indeterminate | × | × |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | — | — |

**Figure 1.20.3. 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 1.20.2 shows the specifications of one-shot mode. Figure 1.20.4 shows the A-D control register in one-shot mode.

**Table 1.20.2. 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 | • End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)<br>• Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | One of $AN_0$ to $AN_7$, as selected |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

A-D control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | 0 | | | |

Symbol ADCON0 — Address $03D6_{16}$ — When reset $00000XXX_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : $AN_0$ is selected<br>0 0 1 : $AN_1$ is selected<br>0 1 0 : $AN_2$ is selected<br>0 1 1 : $AN_3$ is selected<br>1 0 0 : $AN_4$ is selected<br>1 0 1 : $AN_5$ is selected<br>1 1 0 : $AN_6$ is selected<br>1 1 1 : $AN_7$ is selected (Note 2) | ○ | ○ |
| CH1 | | | ○ | ○ |
| CH2 | | | ○ | ○ |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>0 0 : One-shot mode (Note 2) | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $\overline{AD_{TRG}}$ trigger | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | ○ | ○ |
| CKS0 | Frequency select bit 0 | 0: $f_{AD}/4$ is selected<br>1: $f_{AD}/2$ is selected | ○ | ○ |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: When changing A-D operation mode, set analog input pin again.

A-D control register 1 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | | 0 | | |

Symbol ADCON1 — Address $03D7_{16}$ — When reset $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | Invalid in one-shot mode | ○ | ○ |
| SCAN1 | | | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit1 | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}$ is selected | ○ | ○ |
| VCUT | Vref connect bit | 1 : Vref connected | ○ | ○ |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | ○ | ○ |
| OPA1 | | | ○ | ○ |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

**Figure 1.20.4. 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 1.20.3 shows the specifications of repeat mode. Figure 1.20.5 shows the A-D control register in repeat mode.

**Table 1.20.3. Repeat mode specifications**

| Item | Specification |
|---|---|
| Function | The pin selected by the analog input pin select bit is used for repeated A-D conversion |
| Star 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 $AN_0$ to $AN_7$, as selected |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

A-D control register 0 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 0 | 1 | | | |

Symbol ADCON0 — Address $03D6_{16}$ — When reset $00000XXX_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | b2 b1 b0<br>0 0 0 : $AN_0$ is selected<br>0 0 1 : $AN_1$ is selected<br>0 1 0 : $AN_2$ is selected<br>0 1 1 : $AN_3$ is selected<br>1 0 0 : $AN_4$ is selected<br>1 0 1 : $AN_5$ is selected<br>1 1 0 : $AN_6$ is selected<br>1 1 1 : $AN_7$ is selected (Note 2) | ○ | ○ |
| CH1 | | | ○ | ○ |
| CH2 | | | ○ | ○ |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>0 1 : Repeat mode (Note 2) | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $AD_{TRG}$ trigger | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | ○ | ○ |
| CKS0 | Frequency select bit 0 | 0 : $f_{AD}/4$ is selected<br>1 : $f_{AD}/2$ is selected | ○ | ○ |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: When changing A-D operation mode, set analog input pin again.

A-D control register 1 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | | 0 | | |

Symbol ADCON1 — Address $03D7_{16}$ — When reset $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | Invalid in repeat mode | ○ | ○ |
| SCAN1 | | | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit 1 | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}$ is selected | ○ | ○ |
| VCUT | Vref connect bit | 1 : Vref connected | ○ | ○ |
| OPA0 | External op-amp connection mode bit | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | ○ | ○ |
| OPA1 | | | ○ | ○ |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

**Figure 1.20.5. 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 1.20.4 shows the specifications of single sweep mode. Figure 1.20.6 shows the A-D control register in single sweep mode.

**Table 1.20.4. 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 | • End of A-D conversion (A-D conversion start flag changes to "0", except when external trigger is selected)<br>• Writing "0" to A-D conversion start flag |
| Interrupt request generation timing | End of A-D conversion |
| Input pin | $AN_0$ and $AN_1$ (2 pins), $AN_0$ to $AN_3$ (4 pins), $AN_0$ to $AN_5$ (6 pins), or $AN_0$ to $AN_7$ (8 pins) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin |

A-D control register 0 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 1 | 0 | | | |

Symbol ADCON0 — Address $03D6_{16}$ — When reset $00000XXX_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | Invalid in single sweep mode | O | O |
| CH1 | | | O | O |
| CH2 | | | O | O |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>1 0 : Single sweep mode | O | O |
| MD1 | | | O | O |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $AD_{TRG}$ trigger | O | O |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | O | O |
| CKS0 | Frequency select bit 0 | 0 : $f_{AD}/4$ is selected<br>1 : $f_{AD}/2$ is selected | O | O |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

A-D control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | | 0 | | |

Symbol ADCON1 — Address $03D7_{16}$ — When reset $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When single sweep and repeat sweep mode 0 are selected<br>b1 b0<br>0 0 : $AN_0$, $AN_1$ (2 pins)<br>0 1 : $AN_0$ to $AN_3$ (4 pins)<br>1 0 : $AN_0$ to $AN_5$ (6 pins)<br>1 1 : $AN_0$ to $AN_7$ (8 pins) | O | O |
| SCAN1 | | | O | O |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1 | O | O |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | O | O |
| CKS1 | Frequency select bit 1 | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}$ is selected | O | O |
| VCUT | Vref connect bit | 1 : Vref connected | O | O |
| OPA0 | External op-amp connection mode bit (Note 2) | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | O | O |
| OPA1 | | | O | O |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: Neither '01' nor '10' can be selected with the external op-amp connection mode bit.

**Figure 1.20.6. 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 1.20.5 shows the specifications of repeat sweep mode 0. Figure 1.20.7 shows the A-D control register in repeat sweep mode 0.

**Table 1.20.5. 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 | $AN_0$ and $AN_1$ (2 pins), $AN_0$ to $AN_3$ (4 pins), $AN_0$ to $AN_5$ (6 pins), or $AN_0$ to $AN_7$ (8 pins) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin (at any time) |

A-D control register 0 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 1 | 1 | | | |

Symbol ADCON0 — Address $03D6_{16}$ — When reset $00000XXX_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | Invalid in repeat sweep mode 0 | ○ | ○ |
| CH1 | | | ○ | ○ |
| CH2 | | | ○ | ○ |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>1 1 : Repeat sweep mode 0 | ○ | ○ |
| MD1 | | | ○ | ○ |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $AD_{TRG}$ trigger | ○ | ○ |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | ○ | ○ |
| CKS0 | Frequency select bit 0 | 0 : $f_{AD}/4$ is selected<br>1 : $f_{AD}/2$ is selected | ○ | ○ |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

A-D control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | | 0 | | |

Symbol ADCON1 — Address $03D7_{16}$ — When reset $00_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When single sweep and repeat sweep mode 0 are selected<br>b1 b0<br>0 0 : $AN_0$, $AN_1$ (2 pins)<br>0 1 : $AN_0$ to $AN_3$ (4 pins)<br>1 0 : $AN_0$ to $AN_5$ (6 pins)<br>1 1 : $AN_0$ to $AN_7$ (8 pins) | ○ | ○ |
| SCAN1 | | | ○ | ○ |
| MD2 | A-D operation mode select bit 1 | 0 : Any mode other than repeat sweep mode 1 | ○ | ○ |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | ○ | ○ |
| CKS1 | Frequency select bit 1 | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}$ is selected | ○ | ○ |
| VCUT | Vref connect bit | 1 : Vref connected | ○ | ○ |
| OPA0 | External op-amp connection mode bit (Note 2) | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | ○ | ○ |
| OPA1 | | | ○ | ○ |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: Neither "01" nor "10" can be selected with the external op-amp connection mode bit.

**Figure 1.20.7. 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 1.20.6 shows the specifications of repeat sweep mode 1. Figure 1.20.8 shows the A-D control register in repeat sweep mode 1.

**Table 1.20.6. 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<br>Example : $AN_0$ selected $AN_0 \rightarrow AN_1 \rightarrow AN_0 \rightarrow AN_2 \rightarrow AN_0 \rightarrow AN_3$, 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 | $AN_0$ (1 pin), $AN_0$ and $AN_1$ (2 pins), $AN_0$ to $AN_2$ (3 pins), $AN_0$ to $AN_3$ (4 pins) |
| Reading of result of A-D converter | Read A-D register corresponding to selected pin (at any time) |

A-D control register 0 (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | 1 | 1 | | | |

Symbol ADCON0 — Address 03D6₁₆ — When reset 00000XXX₂

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| CH0 | Analog input pin select bit | Invalid in repeat sweep mode 1 | O | O |
| CH1 | | | O | O |
| CH2 | | | O | O |
| MD0 | A-D operation mode select bit 0 | b4 b3<br>1 1 : Repeat sweep mode 1 | O | O |
| MD1 | | | O | O |
| TRG | Trigger select bit | 0 : Software trigger<br>1 : $AD_{TRG}$ trigger | O | O |
| ADST | A-D conversion start flag | 0 : A-D conversion disabled<br>1 : A-D conversion started | O | O |
| CKS0 | Frequency select bit 0 | 0 : $f_{AD}/4$ is selected<br>1 : $f_{AD}/2$ is selected | O | O |

Note: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.

A-D control register 1 (Note 1)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | 1 | | | 1 | | |

Symbol ADCON1 — Address 03D7₁₆ — When reset 00₁₆

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| SCAN0 | A-D sweep pin select bit | When repeat sweep mode 1 is selected<br>b1 b0<br>0 0 : $AN_0$ (1 pin)<br>0 1 : $AN_0$, $AN_1$ (2 pins)<br>1 0 : $AN_0$ to $AN_2$ (3 pins)<br>1 1 : $AN_0$ to $AN_3$ (4 pins) | O | O |
| SCAN1 | | | O | O |
| MD2 | A-D operation mode select bit 1 | 1 : Repeat sweep mode 1 | O | O |
| BITS | 8/10-bit mode select bit | 0 : 8-bit mode<br>1 : 10-bit mode | O | O |
| CKS1 | Frequency select bit 1 | 0 : $f_{AD}/2$ or $f_{AD}/4$ is selected<br>1 : $f_{AD}$ is selected | O | O |
| VCUT | Vref connect bit | 1 : Vref connected | O | O |
| OPA0 | External op-amp connection mode bit (Note 2) | b7 b6<br>0 0 : ANEX0 and ANEX1 are not used<br>0 1 : ANEX0 input is A-D converted<br>1 0 : ANEX1 input is A-D converted<br>1 1 : External op-amp connection mode | O | O |
| OPA1 | | | O | O |

Note 1: If the A-D control register is rewritten during A-D conversion, the conversion result is indeterminate.
Note 2: Neither ‘01’ nor ‘10’ can be selected with the external op-amp connection mode bit.

**Figure 1.20.8. 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 03D4$_{16}$) to “1”. When sample and hold is selected, the rate of conversion of each pin increases. As a result, a 28 f AD cycle is achieved with 8-bit resolution and 33 f 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 03D7$_{16}$) 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 03D7$_{16}$) 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 03D7$_{16}$) is “1” and bit 7 is “1”, input via AN$_0$ to AN$_7$ 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 1.20.9 is an example of how to connect the pins in external operation amp mode.

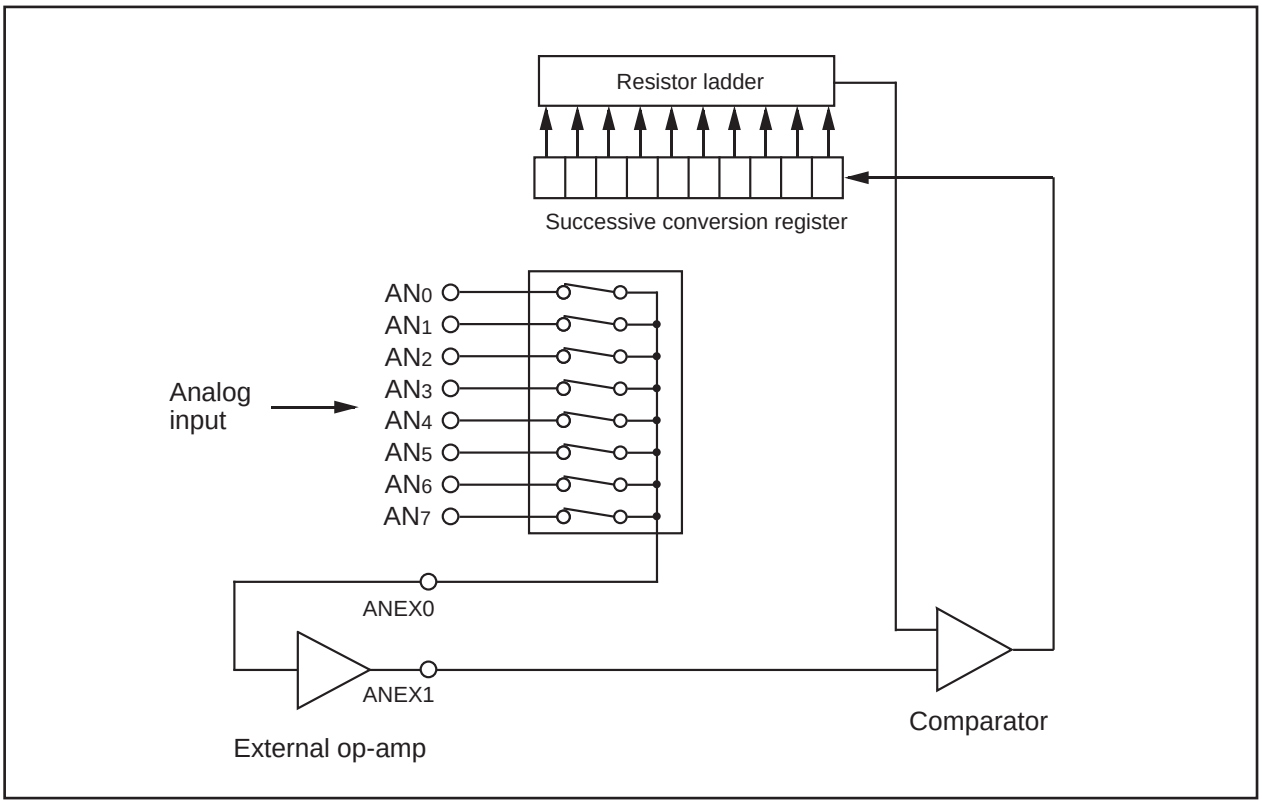



**Figure 1.20.9. Example of external op-amp connection mode**

## 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 1.21.1 lists the performance of the D-A converter. Figure 1.21.1 shows the block diagram of the D-A converter. Figure 1.21.2 shows the D-A control register. Figure J1.21.3 shows the D-A converter equivalent circuit.

**Table 1.21.1. Performance of D-A converter**

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

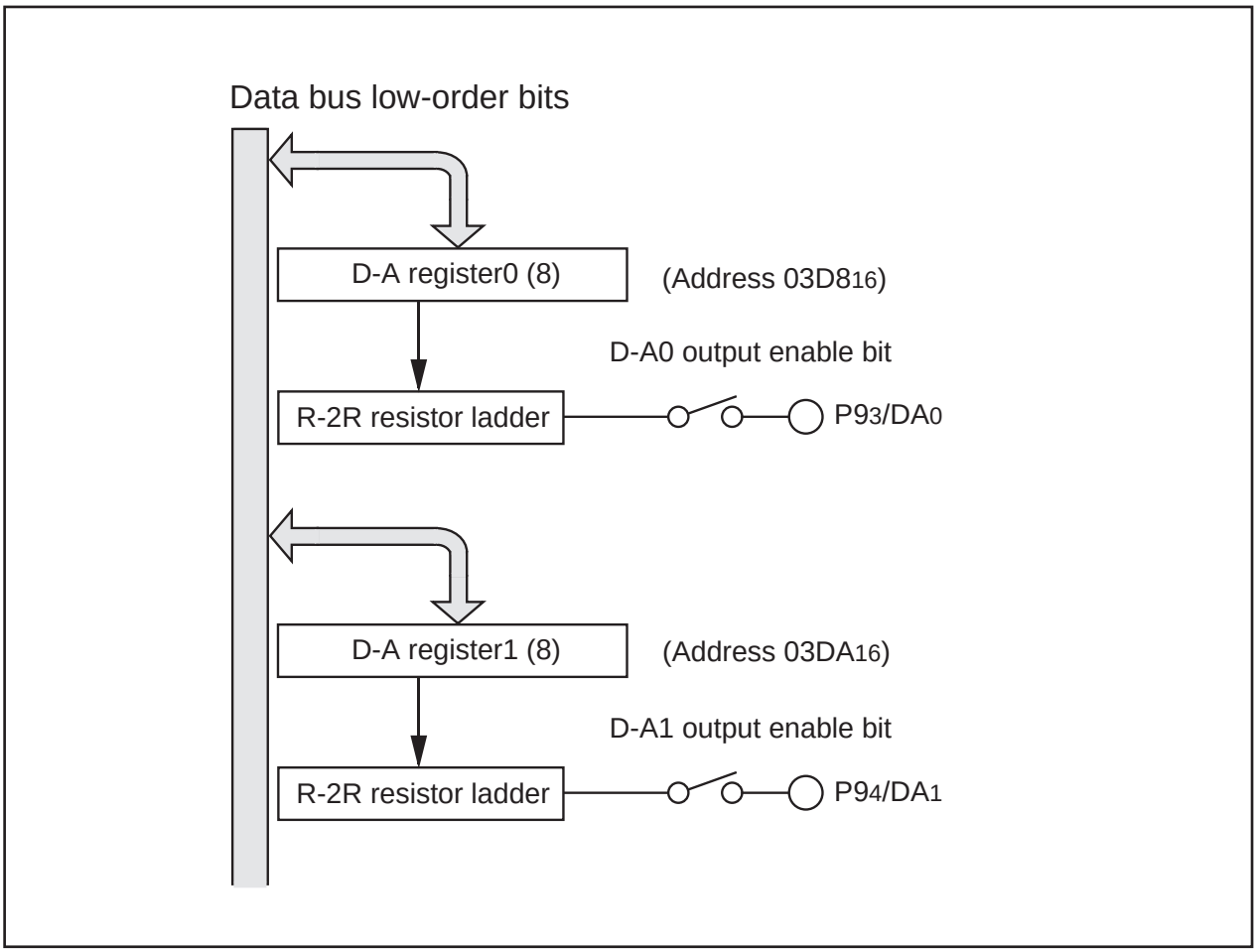



**Figure 1.21.1. Block diagram of D-A converter**

D-A control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| X | X | X | X | X | X | | |

| Symbol | Address | When reset |
|---|---|---|
| DACON | 03DC16 | 0016 |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| DA0E | D-A0 output enable bit | 0 : Output disabled<br>1 : Output enabled | O | O |
| DA1E | D-A1 output enable bit | 0 : Output disabled<br>1 : Output enabled | O | O |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0” | | | – | – |

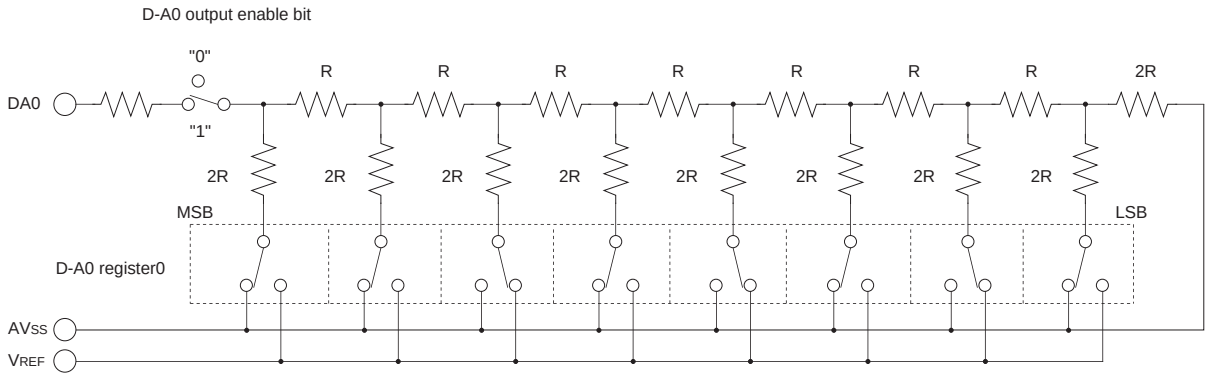
D-A register

| b7 | b0 |
|---|---|
| | |

| Symbol | Address | When reset |
|---|---|---|
| DAi (i = 0,1) | 03D816, 03DA16 | Indeterminate |

| Function | R | W |
|---|---|---|
| Output value of D-A conversion | O | O |

**Figure 1.21.2. D-A control register**




Note 1: The above diagram shows an instance in which the D-A register is assigned 2A16.
Note 2: The same circuit as this is also used for D-A1.
Note 3: To reduce the current consumption when the D-A converter is not used, set the D-A output enable bit to 0 and set the D-A register to 0016 so that no current flows in the resistors Rs and 2Rs.

**Figure 1.21.3. 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 1.22.1 shows the block diagram of the CRC circuit. Figure 1.22.2 shows the CRC-related registers. Figure 1.22.3 shows the calculation example using the CRC calculation circuit

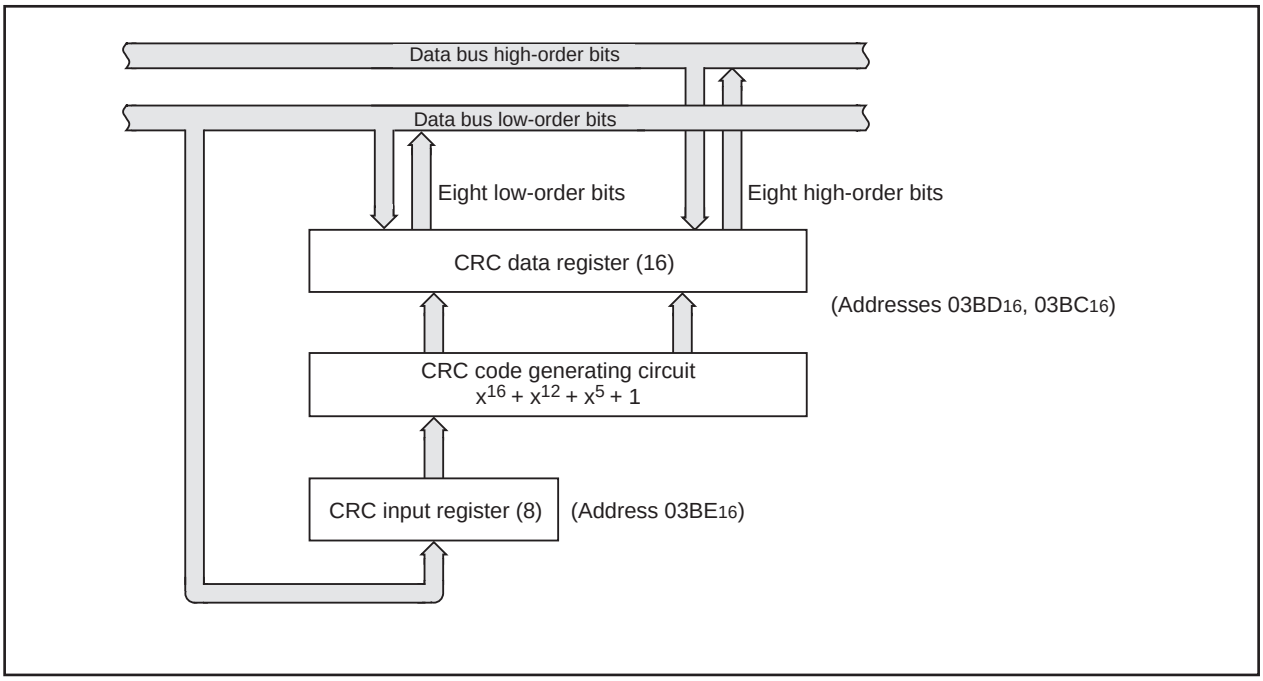
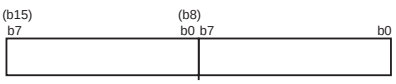



**Figure 1.22.1. Block diagram of CRC circuit**

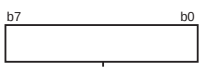
CRC data register




| Symbol | Address | When reset |
|---|---|---|
| CRCD | 03BD16, 03BC16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| CRC calculation result output register | 000016 to FFFF16 | ○ | ○ |

CRC input register




| Symbo | Address | When reset |
|---|---|---|
| CRCIN | 03BE16 | Indeterminate |

| Function | Values that can be set | R | W |
|---|---|---|---|
| Data input register | 0016 to FF16 | ○ | ○ |

**Figure 1.22.2. CRC-related registers**

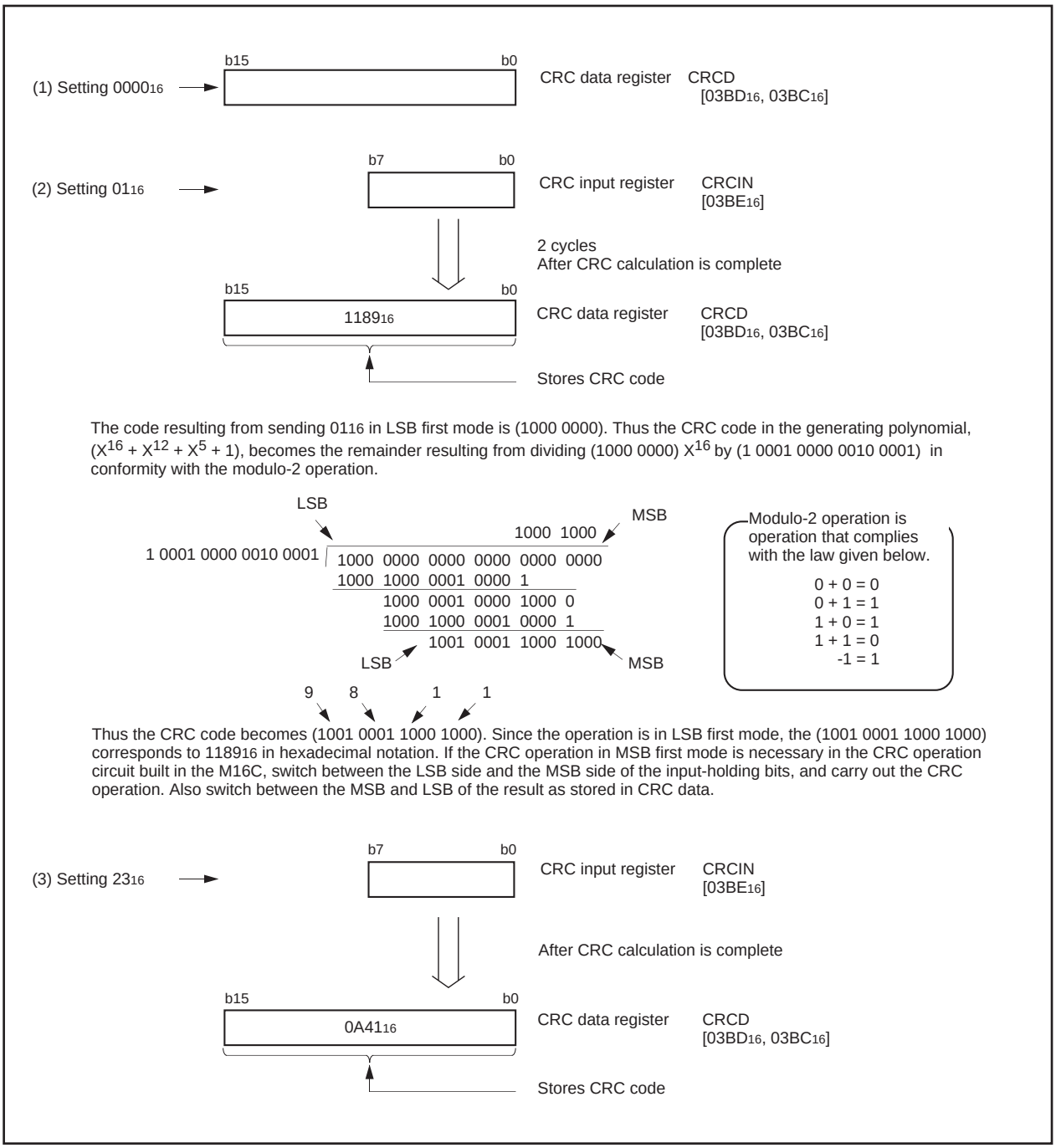



**Figure 1.22.3. Calculation example using the CRC calculation circuit**

## Programmable I/O Ports

There are 87 programmable I/O ports: P0 to P10 (excluding P8$_5$). 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. P8$_5$ is an input-only port and has no built-in pull-up resistance.

Figures 1.23.1 to 1.23.4 show the programmable I/O ports. Figure 1.23.5 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 1.23.6 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 P8$_5$.

### (2) Port registers

Figure 1.23.7 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 1.23.8 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, P0 to P5 operate as the bus and the pull-up control register setting is invalid.

### (4) Port control register

Figure 1.23.9 shows the port control register.

The bit 0 of port control resister is used to read port P1 as follows:

0 : When port P1 is input port, port input level is read.
When port P1 is output port , the contents of port P1 register is read.

1 : The contents of port P1 register is read always.

This register is valid in the following:

- External bus width is 8 bits in microprocessor mode or memory expansion mode.
- Port P1 can be used as a port in multiplexed bus for the entire space.

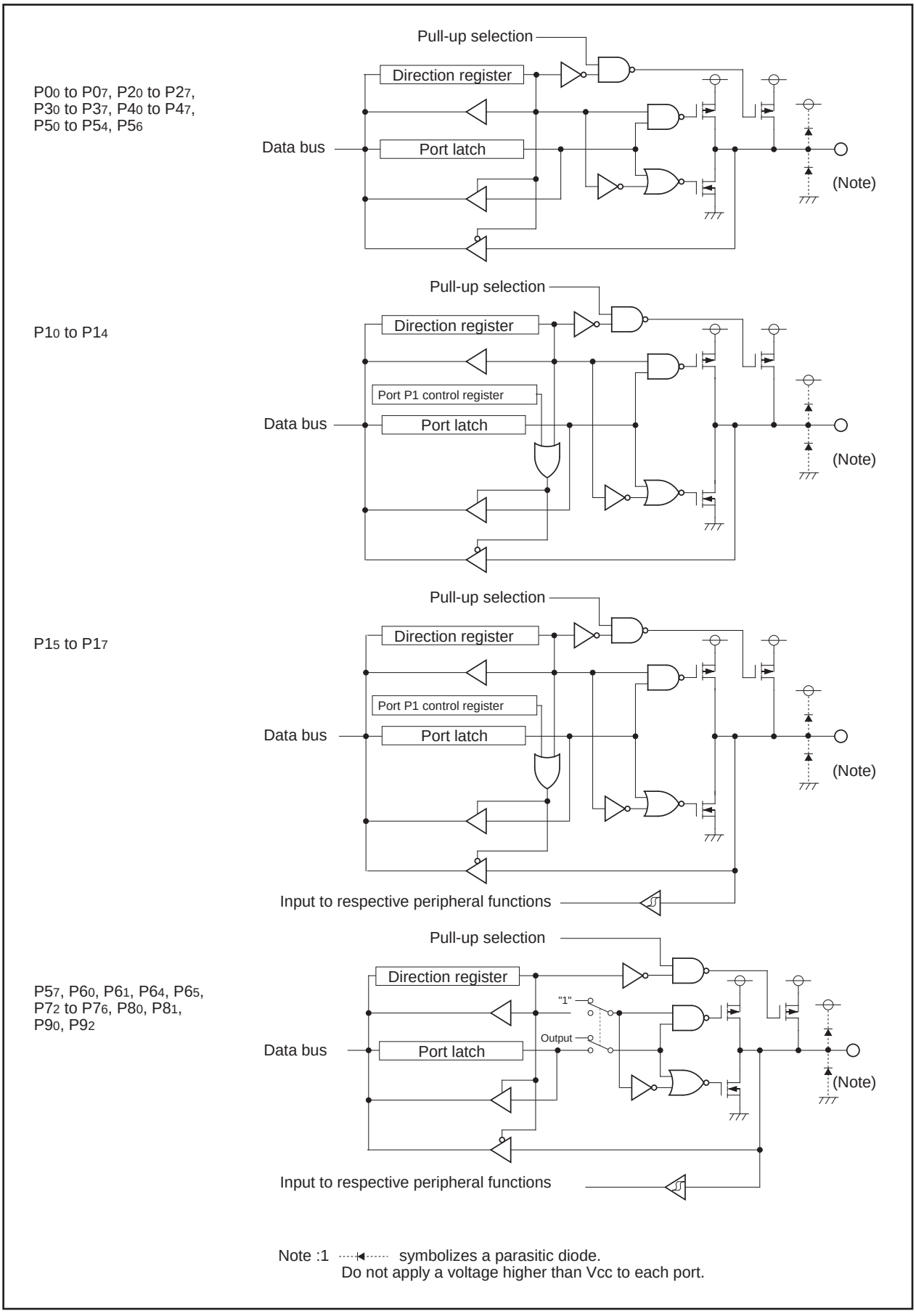



**Figure 1.23.1. Programmable I/O ports (1)**

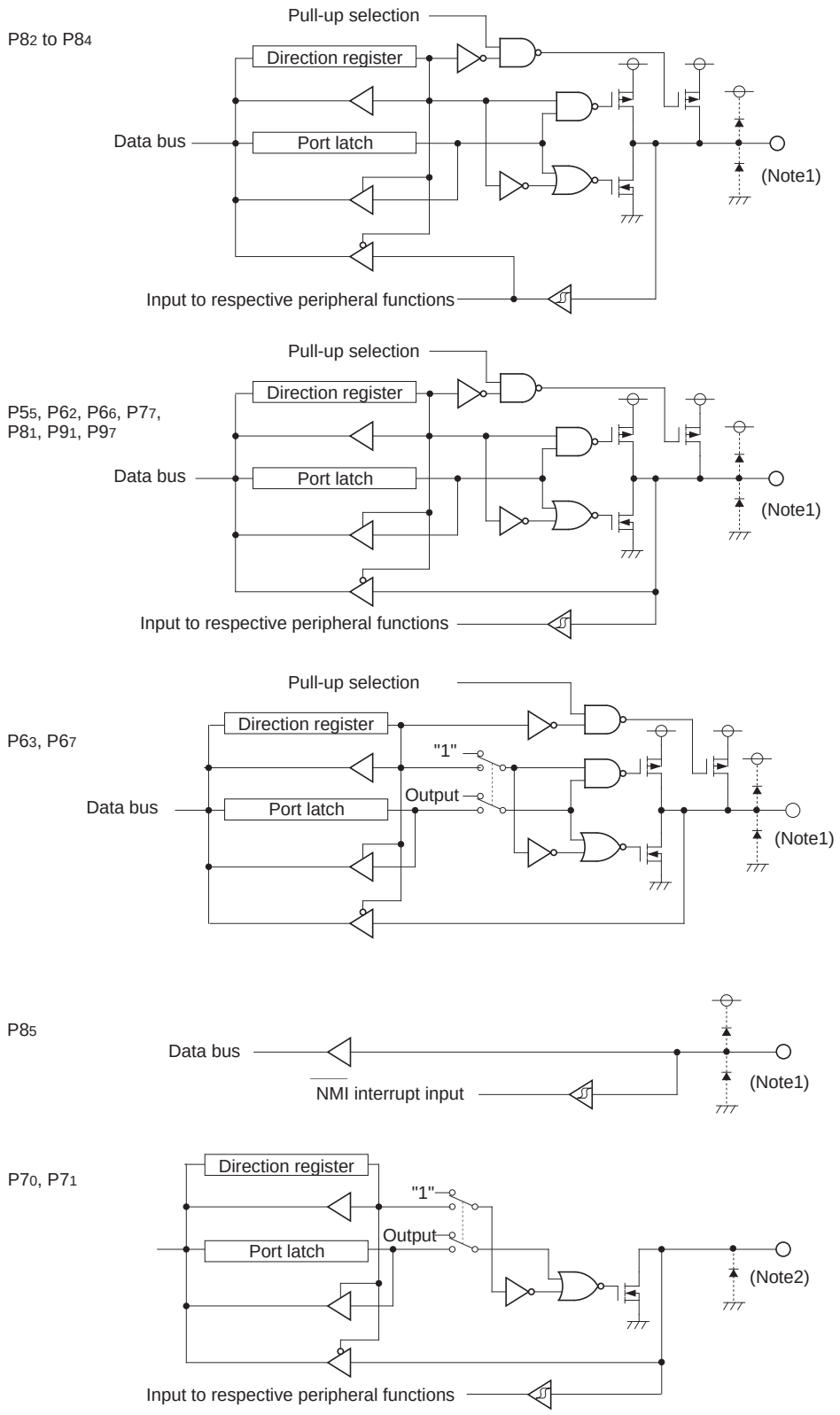



**Figure 1.23.2. Programmable I/O ports (2)**

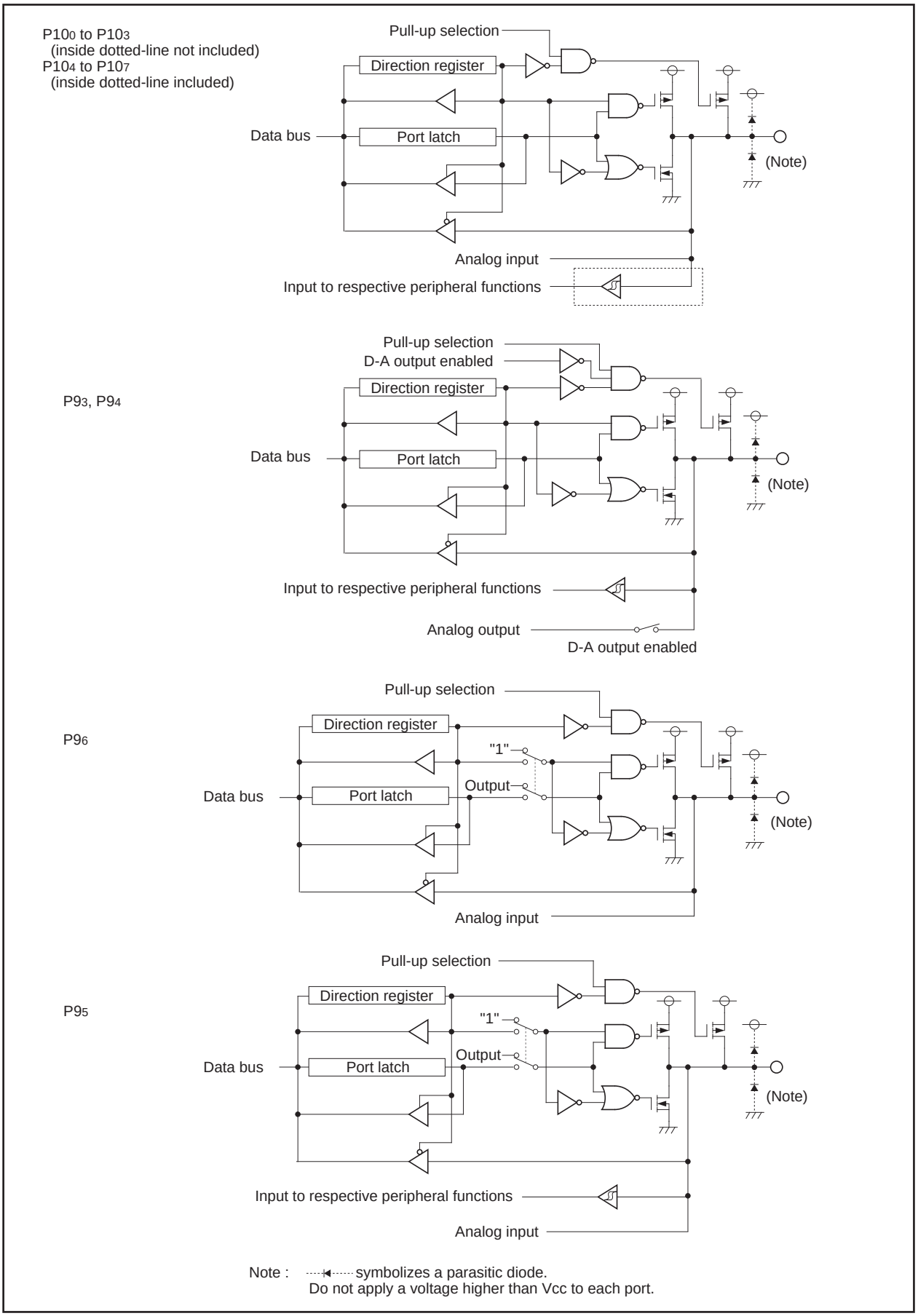



**Figure 1.23.3. Programmable I/O ports (3)**

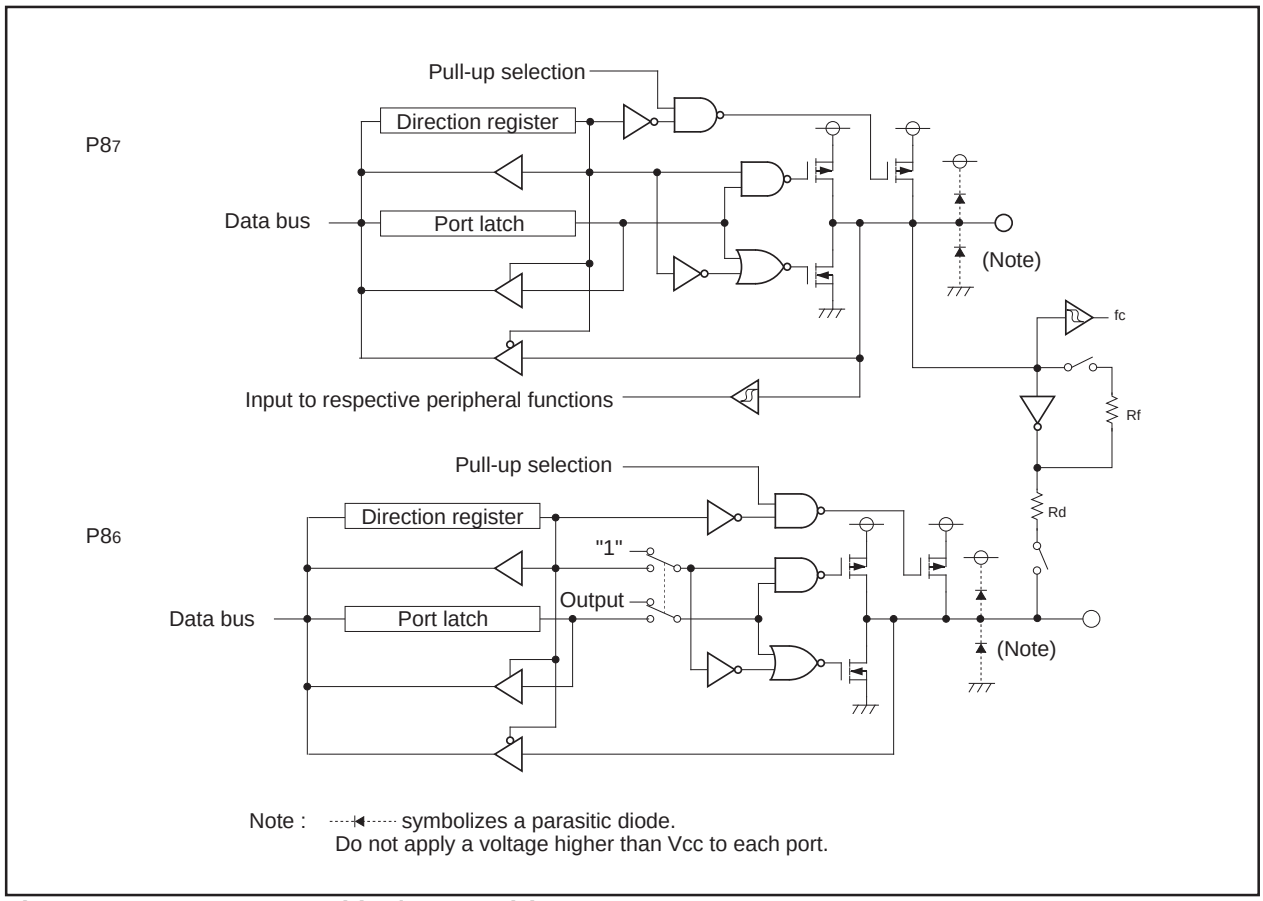



**Figure 1.23.4. Programmable I/O ports (4)**

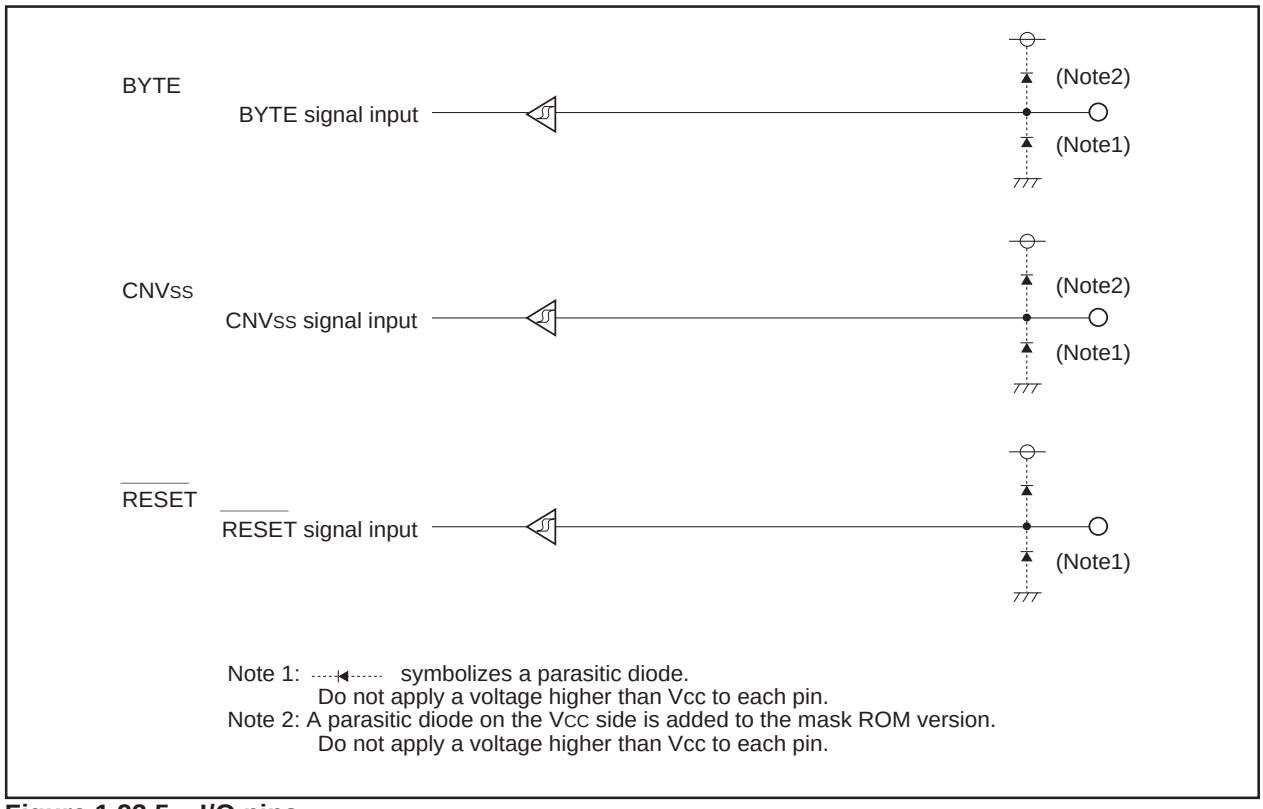



**Figure 1.23.5. I/O pins**

Port Pi direction register (Note)

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

Symbol: PDi (i = 0 to 10, except 8)
Address: 03E2$_{16}$, 03E3$_{16}$, 03E6$_{16}$, 03E7$_{16}$, 03EA$_{16}$, 03EB$_{16}$, 03EE$_{16}$, 03EF$_{16}$, 03F3$_{16}$, 03F6$_{16}$
When reset: 00$_{16}$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PDi_0 | Port Pi$_0$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port)<br>(i = 0 to 10 except 8) | O | O |
| PDi_1 | Port Pi$_1$ direction register | | O | O |
| PDi_2 | Port Pi$_2$ direction register | | O | O |
| PDi_3 | Port Pi$_3$ direction register | | O | O |
| PDi_4 | Port Pi$_4$ direction register | | O | O |
| PDi_5 | Port Pi$_5$ direction register | | O | O |
| PDi_6 | Port Pi$_6$ direction register | | O | O |
| PDi_7 | Port Pi$_7$ direction register | | O | O |

Note: Set bit 2 of protect register (address 000A$_{16}$) to “1” before rewriting to the port P9 direction register.

Port P8 direction register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | ☒ | | | | | |

Symbol: PD8
Address: 03F2$_{16}$
When reset: 00X00000$_2$

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PD8_0 | Port P8$_0$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | O | O |
| PD8_1 | Port P8$_1$ direction register | | O | O |
| PD8_2 | Port P8$_2$ direction register | | O | O |
| PD8_3 | Port P8$_3$ direction register | | O | O |
| PD8_4 | Port P8$_4$ direction register | | O | O |
| Nothing is assigned. In an attempt to write to this bit, write “0”. The value, if read, turns out to be indeterminate. | | | — | — |
| PD8_6 | Port P8$_6$ direction register | 0 : Input mode (Functions as an input port)<br>1 : Output mode (Functions as an output port) | O | O |
| PD8_7 | Port P8$_7$ direction register | | O | O |

**Figure 1.23.6. Direction register**

Port Pi register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | When reset |
|---|---|---|
| Pi (i = 0 to 10, except 8) | 03E0$_{16}$, 03E1$_{16}$, 03E4$_{16}$, 03E5$_{16}$, 03E8$_{16}$ | Indeterminate |
| | 03E9$_{16}$, 03EC$_{16}$, 03ED$_{16}$, 03F1$_{16}$, 03F4$_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PDi_0 | Port Pi$_0$ register | Data is input and output to and from each pin by reading and writing to and from each corresponding bit<br>0 : "L" level data<br>1 : "H" level data (Note)<br><br>(i = 0 to 10 except 8) | ○ | ○ |
| PDi_1 | Port Pi$_1$ register | | ○ | ○ |
| PDi_2 | Port Pi$_2$ register | | ○ | ○ |
| PDi_3 | Port Pi$_3$ register | | ○ | ○ |
| PDi_4 | Port Pi$_4$ register | | ○ | ○ |
| PDi_5 | Port Pi$_5$ register | | ○ | ○ |
| PDi_6 | Port Pi$_6$ register | | ○ | ○ |
| PDi_7 | Port Pi$_7$ register | | ○ | ○ |

Note : Since P7$_0$ and P7$_1$ are N-channel open drain ports, the data is high-impedance.

Port P8 register

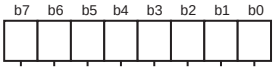
| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbol | Address | When reset |
|---|---|---|
| P8 | 03F0$_{16}$ | Indeterminate |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PD8_0 | Port P8$_0$ register | Data is input and output to and from each pin by reading and writing to and from each corresponding bit (except for P8$_5$)<br>0 : "L" level data<br>1 : "H" level data | ○ | ○ |
| PD8_1 | Port P8$_1$ register | | ○ | ○ |
| PD8_2 | Port P8$_2$ register | | ○ | ○ |
| PD8_3 | Port P8$_3$ register | | ○ | ○ |
| PD8_4 | Port P8$_4$ register | | ○ | ○ |
| PD8_5 | Port P8$_5$ register | | ○ | × |
| PD8_6 | Port P8$_6$ register | | ○ | ○ |
| PD8_7 | Port P8$_7$ register | | ○ | ○ |

**Figure 1.23.7. Port register**

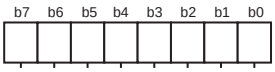
### Pull-up control register 0




Symbol: PUR0 | Address: 03FC16 | When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU00 | P00 to P03 pull-up | The corresponding port is pulled high with a pull-up resistor<br>0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU01 | P04 to P07 pull-up | | ○ | ○ |
| PU02 | P10 to P13 pull-up | | ○ | ○ |
| PU03 | P14 to P17 pull-up | | ○ | ○ |
| PU04 | P20 to P23 pull-up | | ○ | ○ |
| PU05 | P24 to P27 pull-up | | ○ | ○ |
| PU06 | P30 to P33 pull-up | | ○ | ○ |
| PU07 | P34 to P37 pull-up | | ○ | ○ |

### Pull-up control register 1

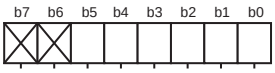



Symbol: PUR1 | Address: 03FD16 | When reset: 0016 (Note 2)

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU10 | P40 to P43 pull-up | The corresponding port is pulled high with a pull-up resistor<br>0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU11 | P44 to P47 pull-up | | ○ | ○ |
| PU12 | P50 to P53 pull-up | | ○ | ○ |
| PU13 | P54 to P57 pull-up | | ○ | ○ |
| PU14 | P60 to P63 pull-up | | ○ | ○ |
| PU15 | P64 to P67 pull-up | | ○ | ○ |
| PU16 | P70 to P73 pull-up (Note 1) | | ○ | ○ |
| PU17 | P74 to P77 pull-up | | ○ | ○ |

Note 1: Since P70 and P71 are N-channel open drain ports, pull-up is not available for them.
Note 2: When the VCC level is being impressed to the CNVSS terminal, this register becomes to 0216 when reset (PU11 becomes to “1”).

### Pull-up control register 2




Symbol: PUR2 | Address: 03FE16 | When reset: 0016

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PU20 | P80 to P83 pull-up | The corresponding port is pulled high with a pull-up resistor<br>0 : Not pulled high<br>1 : Pulled high | ○ | ○ |
| PU21 | P84 to P87 pull-up (Except P85) | | ○ | ○ |
| PU22 | P90 to P93 pull-up | | ○ | ○ |
| PU23 | P94 to P97 pull-up | | ○ | ○ |
| PU24 | P100 to P103 pull-up | | ○ | ○ |
| PU25 | P104 to P107 pull-up | | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | — | — |

**Figure 1.23.8. Pull-up control register**

Port control register

| b7 | b6 | b5 | b4 | b3 | b2 | b1 | b0 |
|---|---|---|---|---|---|---|---|
| | | | | | | | |

| Symbpl | Address | When reset |
|---|---|---|
| PCR | $03FF_{16}$ | $00_{16}$ |

| Bit symbol | Bit name | Function | R | W |
|---|---|---|---|---|
| PCR0 | Port P1 control register | 0 : When input port, read port input level. When output port, read the contents of port P1 register.<br>1 : Read the contents of port P1 register though input/output port. | ○ | ○ |
| Nothing is assigned.<br>In an attempt to write to these bits, write “0”. The value, if read, turns out to be “0”. | | | — | — |

**Figure 1.23.9. Port control register**

**Table 1.23.1. Example connection of unused pins in single-chip mode**

| Pin name | Connection |
|---|---|
| Ports P0 to P10 (excluding P8$_5$) | After setting for input mode, connect every pin to $V_{SS}$ or $V_{CC}$ via a resistor; or after setting for output mode, leave these pins open. |
| $X_{OUT}$ (Note) | Open |
| $\overline{NMI}$ | Connect via resistor to $V_{CC}$ (pull-up) |
| $AV_{CC}$ | Connect to $V_{CC}$ |
| $AV_{SS}$, $V_{REF}$, BYTE | Connect to $V_{SS}$ |

Note: With external clock input to $X_{IN}$ pin.

**Table 1.23.2. Example connection of unused pins in memory expansion mode and microprocessor mode**

| Pin name | Connection |
|---|---|
| Ports P6 to P10 (excluding P8$_5$) | After setting for input mode, connect every pin to $V_{SS}$ or $V_{CC}$ via a resistor; or after setting for output mode, leave these pins open. |
| P4$_5$ / $\overline{CS1}$ to P4$_7$ / $\overline{CS3}$ | Sets ports to input mode, sets bits $\overline{CS1}$ through $\overline{CS3}$ to 0, and connects to Vcc via resistors (pull-up). |
| $\overline{BHE}$, ALE, $\overline{HLDA}$, $X_{OUT}$(Note), BCLK | Open |
| $\overline{HOLD}$, $\overline{RDY}$, $\overline{NMI}$ | Connect via resistor to $V_{CC}$ (pull-up) |
| $AV_{CC}$ | Connect to $V_{CC}$ |
| $AV_{SS}$, $V_{REF}$ | Connect to $V_{SS}$ |

Note: With external clock input to $X_{IN}$ pin.

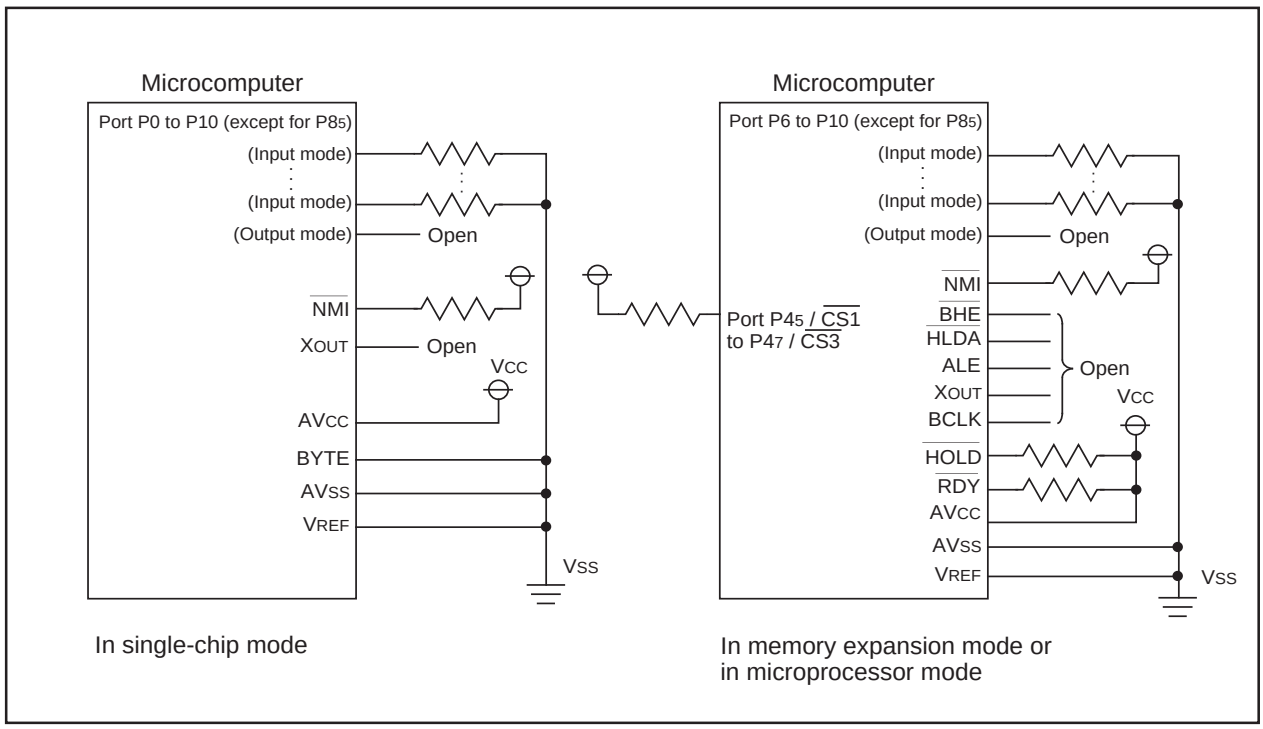



**Figure 1.23.10. Example connection of unused pins**

## 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 "FFFF16". 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 "FFFF16" by underflow or "000016" 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 "FFFF16". 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 μs 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 $00000_{16}$

- 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 $00000_{16}$ will then be set to “0”.
  Reading address $00000_{16}$ 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 $00000_{16}$ by software.

(2) Setting the stack pointer

- The value of the stack pointer immediately after reset is initialized to $0000_{16}$. 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.

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

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

(4) External interrupt

• When the polarity of the $\overline{INT0}$ to $\overline{INT5}$ 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 "11$_2$"

## 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 (M30622ECFP, M30622ECGP, M30620ECFP, M30620ECGP), 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. To improve their reliability after programming, we recommend to program and test as flow shown in Figure 1.24.1 before use.

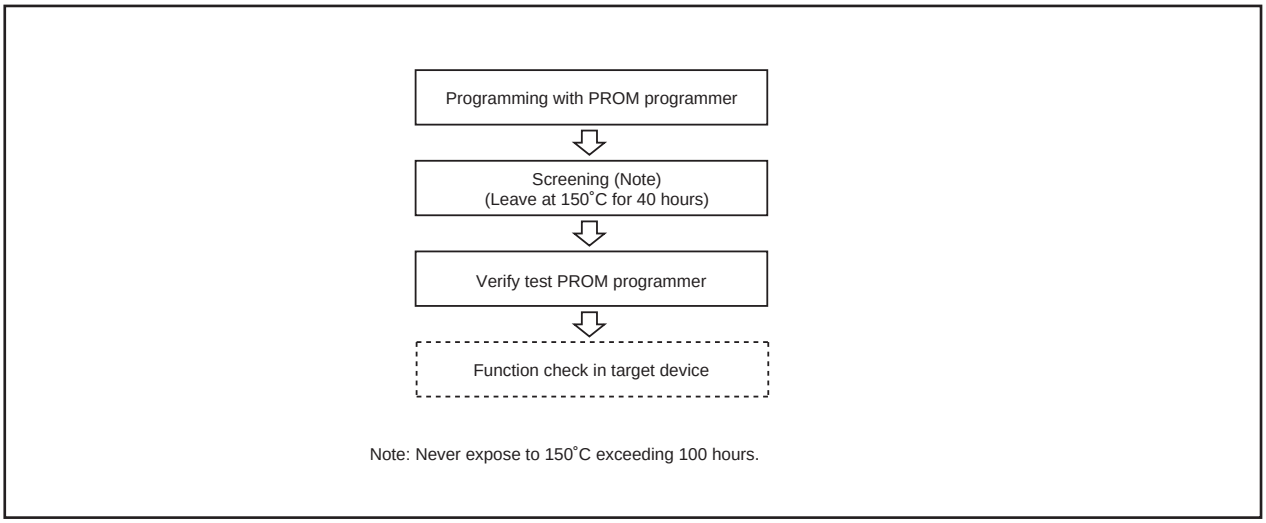



**Figure 1.24.1. Programming and test flow for One Time PROM version**

(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) Mark 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.

## Items to be submitted when ordering data to be written to ROM

Please submit the following when ordering data to be written to one-time PROM products at the factory:

(1) ROM writing order form

(2) Mark 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 1.26.1. Absolute maximum ratings**

| Symbol | Parameter | | Condition | Rated value | Unit |
|---|---|---|---|---|---|
| $V_{CC}$ | Supply voltage | | $V_{CC}$=AV$_{CC}$ | -0.3 to 6.5 | V |
| AVcc | Analog supply voltage | | $V_{CC}$=AV$_{CC}$ | -0.3 to 6.5 | V |
| $V_I$ | Input voltage | $\overline{RESET}$, (maskROM : CNV$_{SS}$, BYTE), P0$_0$ to P0$_7$, P1$_0$ to P1$_7$, P2$_0$ to P2$_7$, P3$_0$ to P3$_7$, P4$_0$ to P4$_7$, P5$_0$ to P5$_7$, P6$_0$ to P6$_7$, P7$_2$ to P7$_7$, P8$_0$ to P8$_7$, P9$_0$ to P9$_7$, P10$_0$ to P10$_7$, $V_{REF}$, $X_{IN}$ | | -0.3 to Vcc+0.3 | V |
| | | P7$_0$, P7$_1$,(EPROM : CNV$_{SS}$, BYTE) | | -0.3 to 6.5(Note 1) | V |
| $V_O$ | Output voltage | P0$_0$ to P0$_7$, P1$_0$ to P1$_7$, P2$_0$ to P2$_7$, P3$_0$ to P3$_7$,P4$_0$ to P4$_7$, P5$_0$ to P5$_7$, P6$_0$ to P6$_7$,P7$_2$ to P7$_7$, P8$_0$ to P8$_4$, P8$_6$, P8$_7$, P9$_0$ to P9$_7$, P10$_0$ to P10$_7$, $X_{OUT}$ | | -0.3 to Vcc+0.3 | V |
| | | P7$_0$, P7$_1$, | | -0.3 to 6.5 | V |
| $P_d$ | Power dissipation | | Ta=25°C | 300 | mW |
| $T_{opr}$ | Operating ambient temperature | | | -20 to 85 / -40 to 85(Note 2) | °C |
| $T_{stg}$ | Storage temperature | | | -65 to 150 | °C |

Note 1: When writing to EPROM ,only CNVss is –0.3 to 13 (V) .

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

**Table 1.26.2. Recommended operating conditions (referenced to Vcc = 2.7V to 5.5V at Ta = – 20 to 85ºC / – 40 to 85ºC(Note3) unless otherwise specified)**

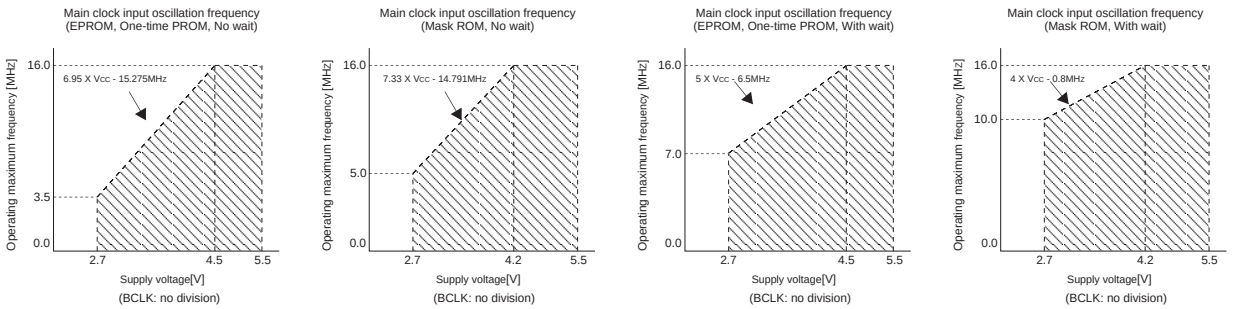
| Symbol | Parameter | | | | Standard Min | Standard Typ. | Standard Max. | Unit |
|---|---|---|---|---|---|---|---|---|
| Vcc | Supply voltage | | | | 2.7 | 5.0 | 5.5 | V |
| AVcc | Analog supply voltage | | | | | Vcc | | V |
| Vss | Supply voltage | | | | | 0 | | V |
| AVss | Analog supply voltage | | | | | 0 | | V |
| $V_{IH}$ | HIGH input voltage | $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $X_{IN}$, RESET, $CNV_{SS}$, BYTE | | | 0.8Vcc | | Vcc | V |
| | | $P7_0$, $P7_1$ | | | 0.8Vcc | | 6.5 | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (during single-chip mode) | | | 0.8Vcc | | Vcc | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (data input function during memory expansion and microprocessor modes) | | | 0.5Vcc | | Vcc | V |
| $V_{IL}$ | LOW input voltage | $P3_1$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, $X_{IN}$, RESET, $CNV_{SS}$, BYTE | | | 0 | | 0.2Vcc | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (during single-chip mode) | | | 0 | | 0.2Vcc | V |
| | | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ (data input function during memory expansion and microprocessor modes) | | | 0 | | 0.16Vcc | V |
| $I_{OH (peak)}$ | HIGH peak output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | | -10.0 | mA |
| $I_{OH (avg)}$ | HIGH average output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | | -5.0 | mA |
| $I_{OL (peak)}$ | LOW peak output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | | 10.0 | mA |
| $I_{OL (avg)}$ | LOW average output current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | | | | | 5.0 | mA |
| f ($X_{IN}$) | Main clock input oscillation frequency | No wait | EPROM, One time PROM | Vcc=4.5V to 5.5V | 0 | | 16 | MHz |
| | | | | Vcc=2.7V to 4.5V | 0 | | 6.95 X Vcc -15.275 | MHz |
| | | | Mask ROM | Vcc=4.2V to 5.5V | 0 | | 16 | MHz |
| | | | | Vcc=2.7V to 4.2V | 0 | | 7.33 X Vcc -14.791 | MHz |
| | | with wait | EPROM, One time PROM | Vcc=4.5V to 5.5V | 0 | | 16 | MHz |
| | | | | Vcc=2.7V to 4.5V | 0 | | 5X Vcc -6.5 | MHz |
| | | | Mask ROM | Vcc=4.2V to 5.5V | 0 | | 16 | MHz |
| | | | | Vcc=2.7V to 4.2V | 0 | | 4 X Vcc -0.8 | MHz |
| f ($X_{CIN}$) | Subclock oscillation frequency | | | | | 32.768 | 50 | kHz |

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

Note 2: The total $I_{OL}$ (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, and P10 must be 80mA max. The total $I_{OH}$ (peak) for ports P0, P1, P2, $P8_6$, $P8_7$, P9, and P10 must be 80mA max. The total $I_{OL}$ (peak) for ports P3, P4, P5, P6, P7, and $P8_0$ to $P8_4$ must be 80mA max. The total $I_{OH}$ (peak) for ports P3, P4, P5, P6, $P7_2$ to $P7_7$, and $P8_0$ to $P8_4$ must be 80mA max.

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

Note 4: Relationship between main clock oscillation frequency and supply voltage.



# Electrical characteristics (Vcc = 5V)

## Vcc = 5V

**Table 1.26.3. Electrical characteristics (referenced to Vcc = 5V, Vss = 0V at Ta = 25°C, f(XIN) = 16MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring condition | | Standard Min | Standard Typ. | Standard Max. | Unit |
|---|---|---|---|---|---|---|---|---|
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | $I_{OH}$=-5mA | | 3.0 | | | V |
| $V_{OH}$ | HIGH output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | $I_{OH}$=-200µA | | 4.7 | | | V |
| $V_{OH}$ | HIGH output voltage | $X_{OUT}$ | HIGHPOWER | $I_{OH}$=-1mA | 3.0 | | | V |
| | | | LOWPOWER | $I_{OH}$=-0.5mA | 3.0 | | | |
| | HIGH output voltage | $X_{COUT}$ | HIGHPOWER | With no load applied | | 3.0 | | V |
| | | | LOWPOWER | With no load applied | | 1.6 | | |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | $I_{OL}$=5mA | | | | 2.0 | V |
| $V_{OL}$ | LOW output voltage | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | $I_{OL}$=200µA | | | | 0.45 | V |
| $V_{OL}$ | LOW output voltage | $X_{OUT}$ | HIGHPOWER | $I_{OL}$=1mA | | | 2.0 | V |
| | | | LOWPOWER | $I_{OL}$=0.5mA | | | 2.0 | |
| | LOW output voltage | $X_{COUT}$ | HIGHPOWER | With no load applied | | 0 | | V |
| | | | LOWPOWER | With no load applied | | 0 | | |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{HOLD}$, $\overline{RDY}$, TA0IN to TA4IN, TB0IN to TB2IN, $\overline{INT0}$ to $\overline{INT5}$, $\overline{ADTRG}$, $\overline{CTS0}$, $\overline{CTS1}$, CLK0, CLK1, TA2OUT to TA4OUT, NMI, KI0 to KI3 | | | 0.2 | | 0.8 | V |
| $V_{T+}$-$V_{T-}$ | Hysteresis | $\overline{RESET}$ | | | 0.2 | | 1.8 | V |
| $I_{IH}$ | HIGH input current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, XIN, $\overline{RESET}$, CNVss, BYTE | $V_I$=5V | | | | 5.0 | µA |
| $I_{IL}$ | LOW input current | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_0$ to $P7_7$, $P8_0$ to $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$, XIN, $\overline{RESET}$, CNVss, BYTE | $V_I$=0V | | | | -5.0 | µA |
| $R_{PULLUP}$ | Pull-up resistance | $P0_0$ to $P0_7$, $P1_0$ to $P1_7$, $P2_0$ to $P2_7$, $P3_0$ to $P3_7$, $P4_0$ to $P4_7$, $P5_0$ to $P5_7$, $P6_0$ to $P6_7$, $P7_2$ to $P7_7$, $P8_0$ to $P8_4$, $P8_6$, $P8_7$, $P9_0$ to $P9_7$, $P10_0$ to $P10_7$ | $V_I$=0V | | 30.0 | 50.0 | 167.0 | kΩ |
| $R_{fXIN}$ | Feedback resistance XIN | | | | | 1.0 | | MΩ |
| $R_{fCXIN}$ | Feedback resistance XCIN | | | | | 6.0 | | MΩ |
| $V_{RAM}$ | RAM retention voltage | | When clock is stopped | | 2.0 | | | V |
| Icc | Power supply current | | In single-chip mode, the output pins are open and other pins are Vss | f(XIN)=16MHz Square wave, no division | | 30.0 | 50.0 | mA |
| | | | | f(XCIN)=32kHz Square wave | | 90.0 | | µA |
| | | | | f(XCIN)=32kHz When a WAIT instruction is executed | | 4.0 | | µA |
| | | | | Ta=25°C when clock is stopped | | | 1.0 | µA |
| | | | | Ta=85°C when clock is stopped | | | 20.0 | |

## Electrical characteristics (Vcc = 5V)

### Vcc = 5V

**Table 1.26.4. A-D conversion characteristics (referenced to Vcc = AVcc = VREF = 5V, Vss = AVss = 0V at Ta = 25°C, f(XIN) = 16MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring condition | | Standard | | | Unit |
|---|---|---|---|---|---|---|---|---|
| | | | | | Min. | Typ. | Max. | |
| – | Resolution | | VREF = Vcc | | | | 10 | Bits |
| – | Absolute accuracy | Sample & hold function not available | VREF = Vcc = 5V | | | | ±3 | LSB |
| | | Sample & hold function available(10bit) | VREF =Vcc = 5V | AN0 to AN7 input | | | ±3 | LSB |
| | | | | ANEX0, ANEX1 input, External op-amp connection mode | | | ±7 | LSB |
| | | Sample & hold function available(8bit) | VREF = Vcc = 5V | | | | ±2 | LSB |
| RLADDER | Ladder resistance | | VREF = Vcc | | 10 | | 40 | kΩ |
| tCONV | Conversion time(10bit) | | | | 3.3 | | | µs |
| tCONV | Conversion time(8bit) | | | | 2.8 | | | µs |
| tSAMP | Sampling time | | | | 0.3 | | | µs |
| VREF | Reference voltage | | | | 2 | | Vcc | V |
| VIA | Analog input voltage | | | | 0 | | VREF | V |

Note: Divide the frequency if f(XIN) exceeds 10 MHz, and make ØAD equal to or lower than 10 MHz.

**Table 1.26.5. D-A conversion characteristics (referenced to Vcc = 5V, Vss = AVss = 0V, VREF = 5V at Ta = 25°C, f(XIN) = 16MHz unless otherwise specified)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max. | |
| – | Resolution | | | | 8 | Bits |
| – | Absolute accuracy | | | | 1.0 | % |
| tsu | Setup time | | | | 3 | µs |
| RO | Output resistance | | 4 | 10 | 20 | kΩ |
| IVREF | 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 "00$_{16}$".
The A-D converter's ladder resistance is not included.
Also, when the Vref is unconnected at the A-D control register, IVREF is sent.

# Vcc = 5V

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

**Table 1.26.6. External clock input**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_c$ | External clock input cycle time | 62.5 | | ns |
| $t_{w(H)}$ | External clock input HIGH pulse width | 25 | | ns |
| $t_{w(L)}$ | External clock input LOW pulse width | 25 | | ns |
| $t_r$ | External clock rise time | | 15 | ns |
| $t_f$ | External clock fall time | | 15 | ns |

**Table 1.26.7. Memory expansion and microprocessor modes**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{ac1(RD-DB)}$ | Data input access time (no wait) | | (Note) | ns |
| $t_{ac2(RD-DB)}$ | Data input access time (with wait) | | (Note) | ns |
| $t_{ac3(RD-DB)}$ | Data input access time (when accessing multiplex bus area) | | (Note) | ns |
| $t_{su(DB-RD)}$ | Data input setup time | 40 | | ns |
| $t_{su(RDY-BCLK)}$ | $\overline{RDY}$ input setup time | 30 | | ns |
| $t_{su(HOLD-BCLK)}$ | $\overline{HOLD}$ input setup time | 40 | | ns |
| $t_{h(RD-DB)}$ | Data input hold time | 0 | | ns |
| $t_{h(BCLK-RDY)}$ | $\overline{RDY}$ input hold time | 0 | | ns |
| $t_{h(BCLK-HOLD)}$ | $\overline{HOLD}$ input hold time | 0 | | ns |
| $t_{d(BCLK-HLDA)}$ | $\overline{HLDA}$ output delay time | | 40 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

# Vcc = 5V

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

### Table 1.26.8. Timer A input (counter input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 100 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 40 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 40 | | ns |

### Table 1.26.9. Timer A input (gating input in timer mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 400 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 200 | | ns |

### Table 1.26.10. Timer A input (external trigger input in one-shot timer mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 200 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

### Table 1.26.11. Timer A input (external trigger input in pulse width modulation mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 100 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 100 | | ns |

### Table 1.26.12. Timer A input (up/down input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | TAiOUT input cycle time | 2000 | | ns |
| $t_{w(UPH)}$ | TAiOUT input HIGH pulse width | 1000 | | ns |
| $t_{w(UPL)}$ | TAiOUT input LOW pulse width | 1000 | | ns |
| $t_{su(UP-TIN)}$ | TAiOUT input setup time | 400 | | ns |
| $t_{h(TIN-UP)}$ | TAiOUT input hold time | 400 | | ns |

# Vcc = 5V

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

### Table 1.26.13. Timer B input (counter input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 100 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 40 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 40 | | ns |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on both edges) | 200 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on both edges) | 80 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 80 | | ns |

### Table 1.26.14. Timer B input (pulse period measurement mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

### Table 1.26.15. Timer B input (pulse width measurement mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 400 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 200 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 200 | | ns |

### Table 1.26.16. A-D trigger input

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(AD)}$ | $\overline{ADTRG}$ input cycle time (trigger able minimum) | 1000 | | ns |
| $t_{w(ADL)}$ | $\overline{ADTRG}$ input LOW pulse width | 125 | | ns |

### Table 1.26.17. Serial I/O

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

### Table 1.26.18. External interrupt $\overline{INTi}$ inputs

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(INH)}$ | $\overline{INTi}$ input HIGH pulse width | 250 | | ns |
| $t_{w(INL)}$ | $\overline{INTi}$ 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 1.26.19. Memory expansion mode and microprocessor mode (no wait)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.26.1 | | 25 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | 0 | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 25 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 25 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 4 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 25 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 25 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 40 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note1) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard)(Note2) | | 0 | | ns |

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

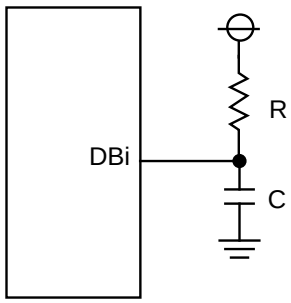
$$td(DB - WR) = \frac{10^9}{f(BCLK) \times 2} - 40 \quad [ns]$$

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 = 30pF, R = 1kΩ, hold time of output "L" level is

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



# VCC = 5V

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

**Table 1.26.20. Memory expansion mode and microprocessor mode (with wait, accessing external memory)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.26.1 | | 25 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | 0 | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 25 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 25 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 4 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 25 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 25 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 40 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note1) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard)(Note2) | | 0 | | ns |

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

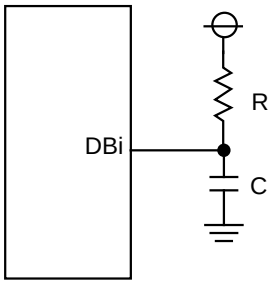
$$td(DB - WR) = \frac{10^9}{f(BCLK)} - 40 \quad [ns]$$

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 = 30pF, R = 1kΩ, hold time of output "L" level is

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



# Electrical characteristics (Vcc = 5V)

## Vcc = 5V

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

**Table 1.26.21. Memory expansion mode and microprocessor mode (with wait, accessing external memory, multiplex bus area selected)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.26.1 | | 25 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | (Note) | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 25 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-CS)}$ | Chip select output hold time (RD standard) | | (Note) | | ns |
| $t_{h(WR-CS)}$ | Chip select output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 25 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 25 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 40 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time (BCLK standard) | | | 25 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time (BCLK standard) | | – 4 | | ns |
| $t_{d(AD-ALE)}$ | ALE signal output delay time (Address standard) | | (Note) | | ns |
| $t_{h(ALE-AD)}$ | ALE signal output hold time (Adderss standard) | | 50 | | ns |
| $t_{d(AD-RD)}$ | Post-address RD signal output delay time | | 0 | | ns |
| $t_{d(AD-WR)}$ | Post-address WR signal output delay time | | 0 | | ns |
| $t_{dZ(RD-AD)}$ | Address output floating start time | | | 8 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

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

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

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

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

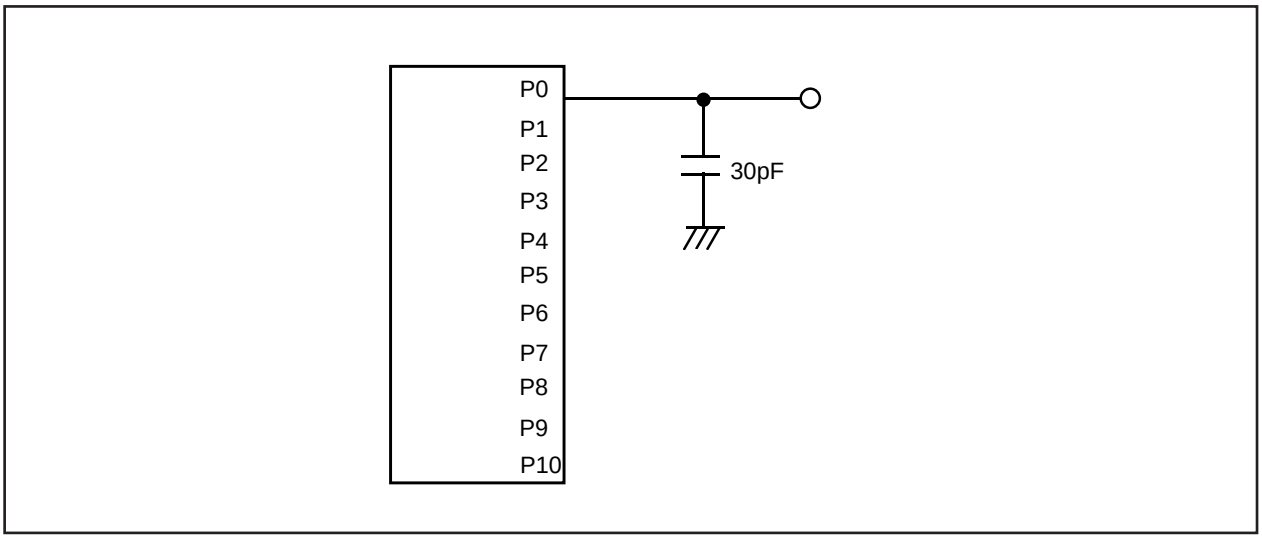
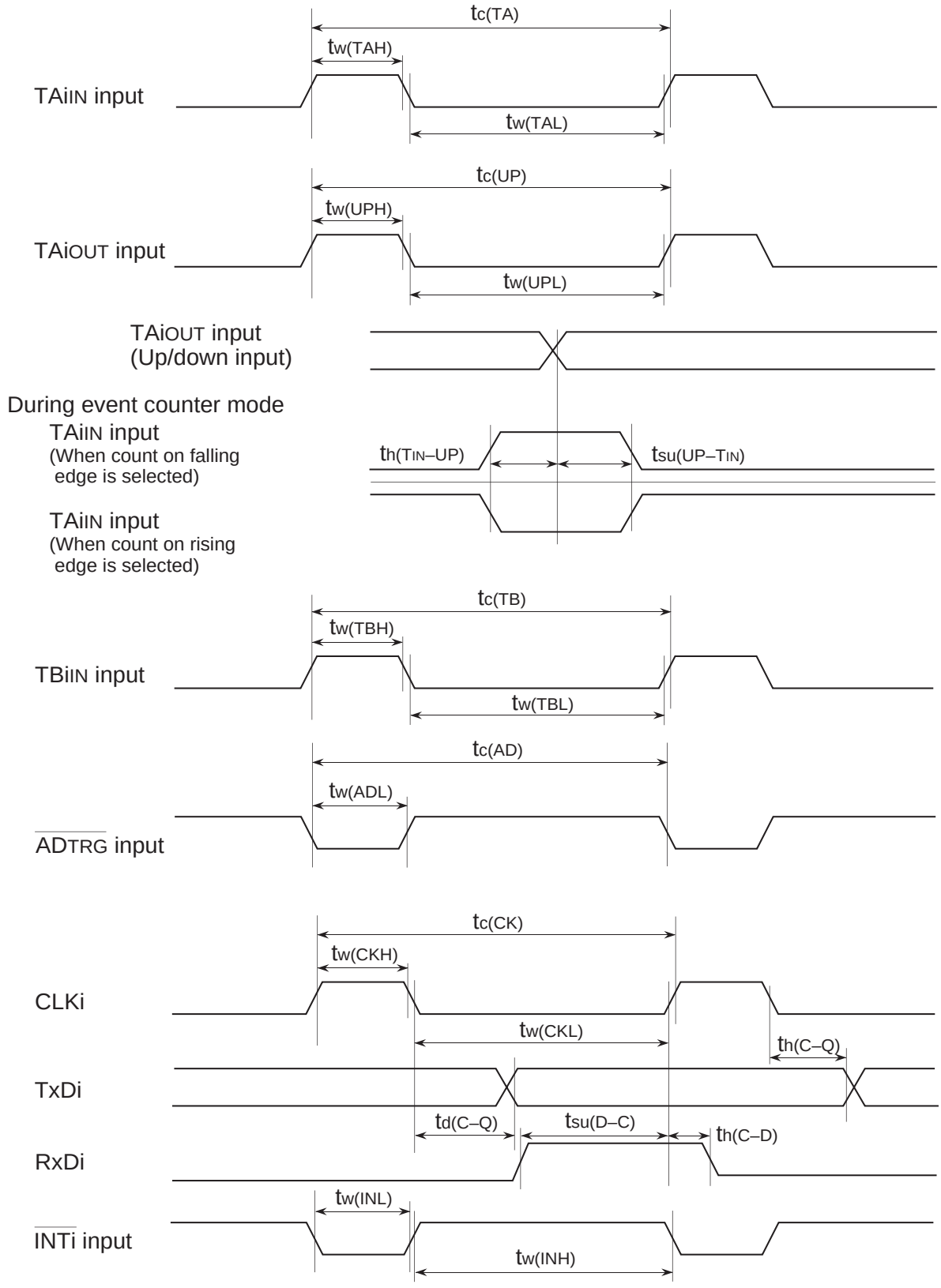



**Figure 1.26.1. Port P0 to P10 measurement circuit**

## Timing (Vcc = 5V)

### Vcc = 5V

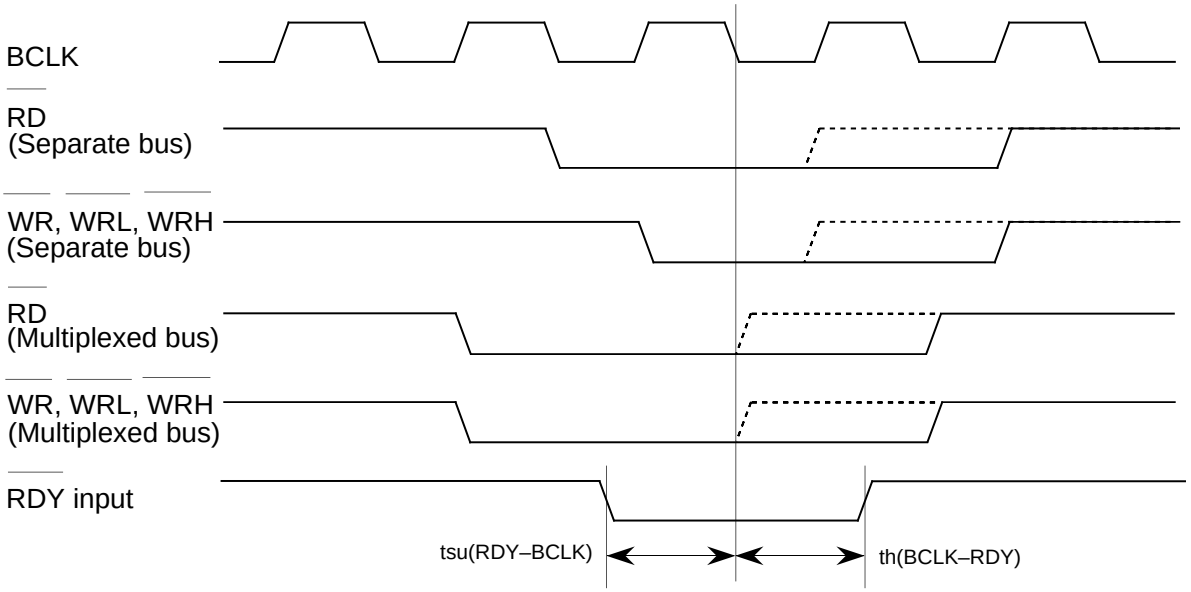


Timing (Vcc = 5V)

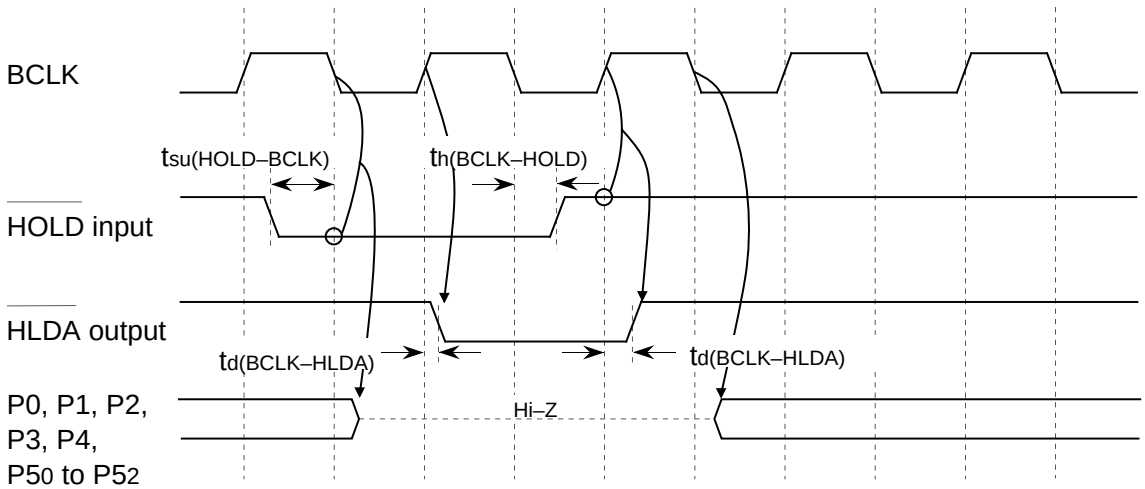
## Vcc = 5V

### Memory Expansion Mode and Microprocessor Mode

**(Valid only with wait)**




**(Valid with or without wait)**




Note: The above pins are set to high-impedance regardless of the input level of the BYTE pin and bit (PM06) of processor mode register 0 selects the function of ports P40 to P43.

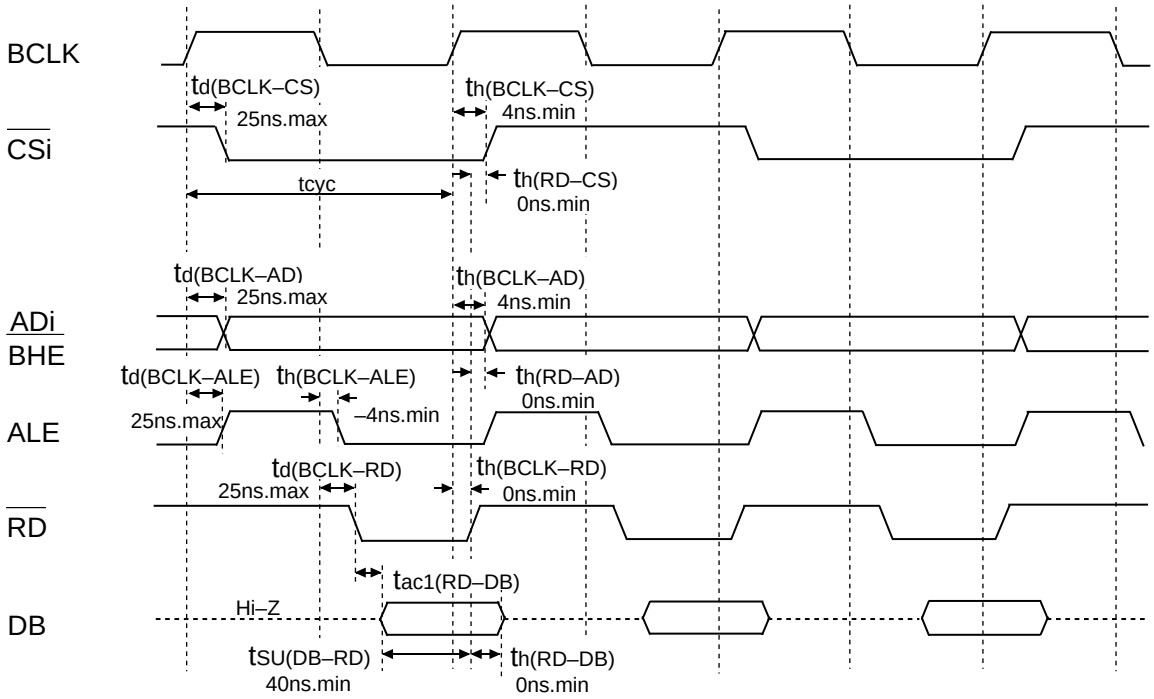
Measuring conditions :

- VCC=5V
- Input timing voltage : Determined with VIL=1.0V, VIH=4.0V
- Output timing voltage : Determined with VOL=2.5V, VOH=2.5V

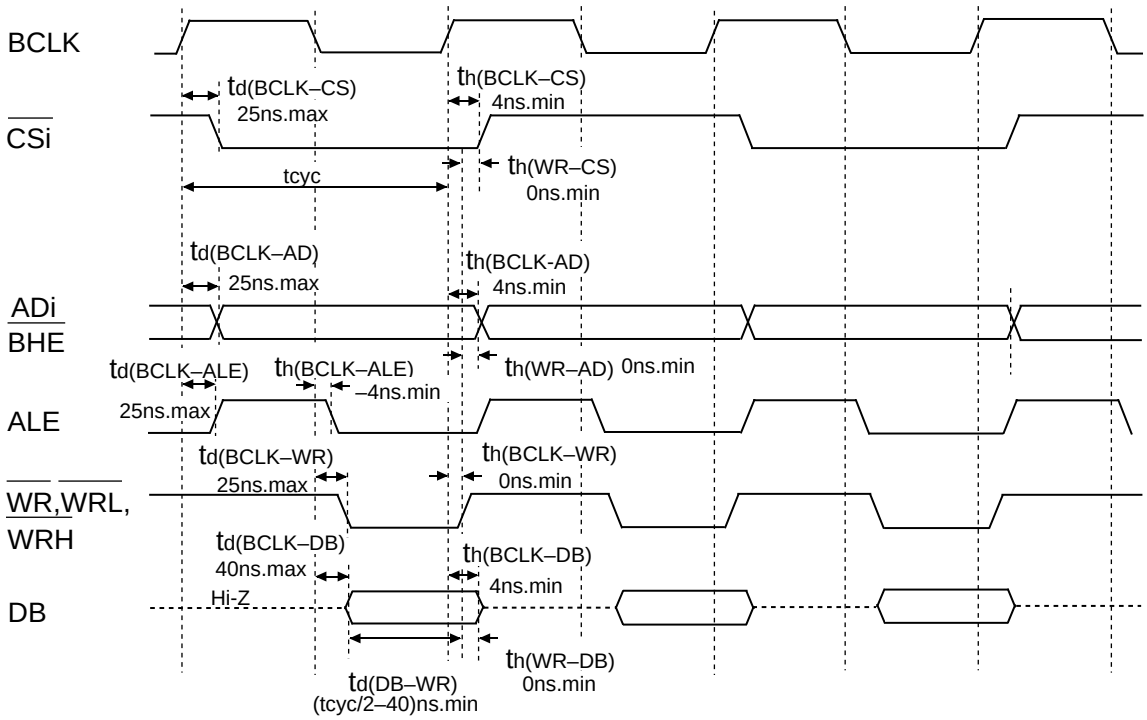
## Memory Expansion Mode and Microprocessor Mode (With no wait)

**Vcc = 5V**

### Read timing




### Write timing



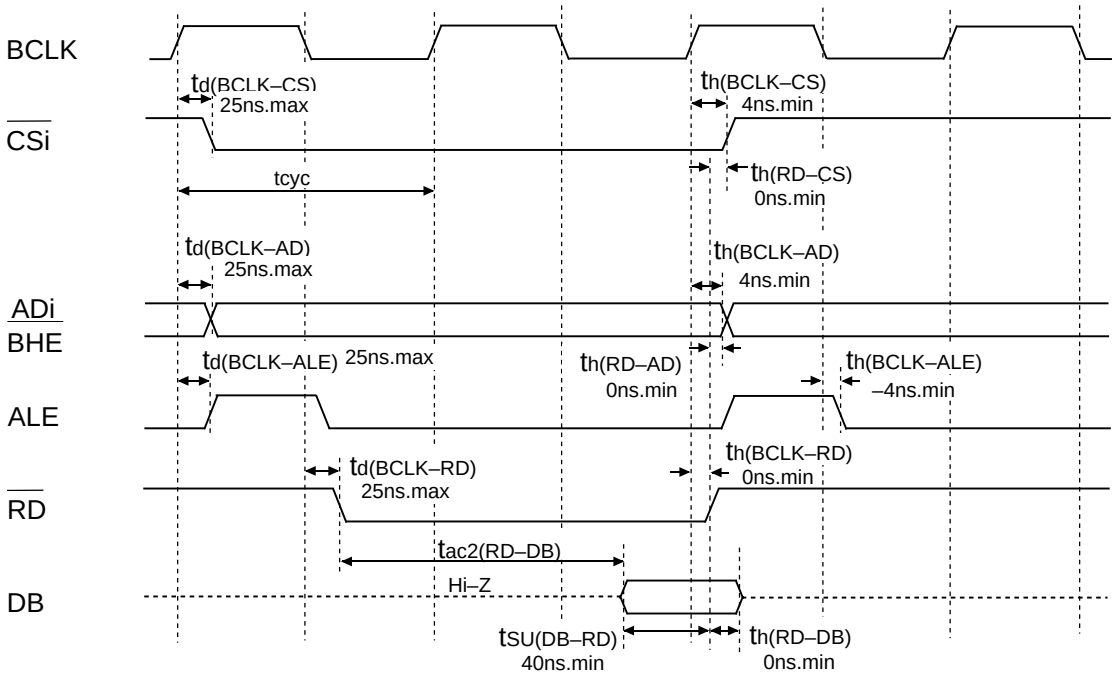
Timing (Vcc = 5V)

## Memory Expansion Mode and Microprocessor Mode

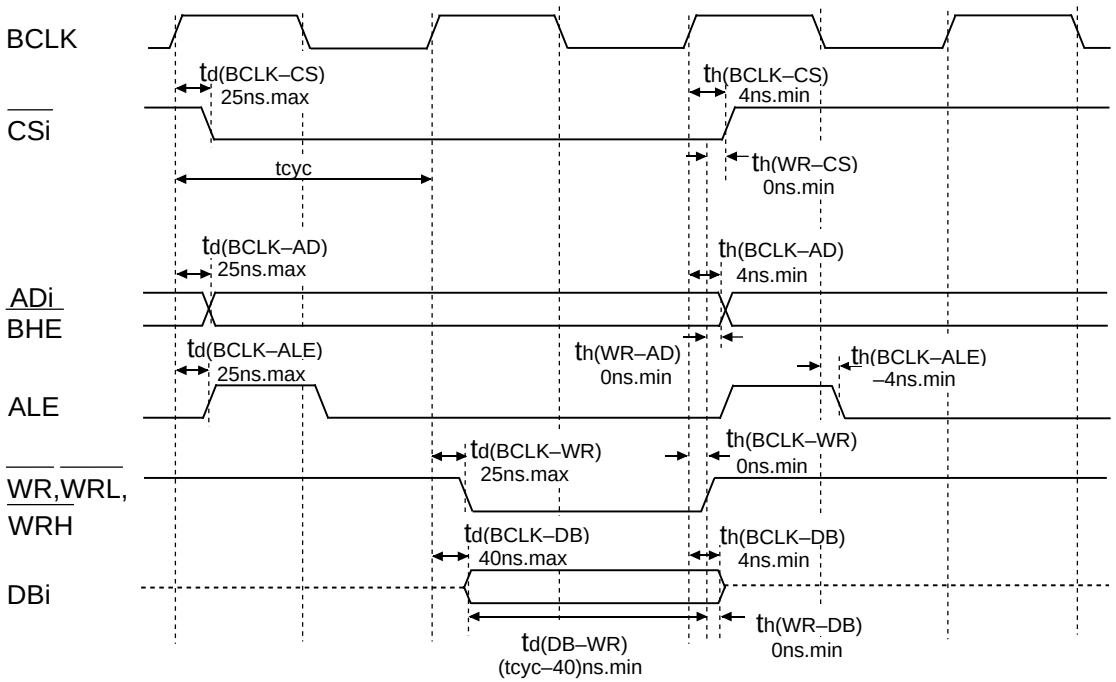
**VCC = 5V**

**(When accessing external memory area with wait)**

**Read timing**




**Write timing**




Measuring conditions :

- VCC=5V
- Input timing voltage : Determined with: VIL=0.8V, VIH=2.5V
- Output timing voltage : Determined with: VOL=0.8V, VOH=2.0V

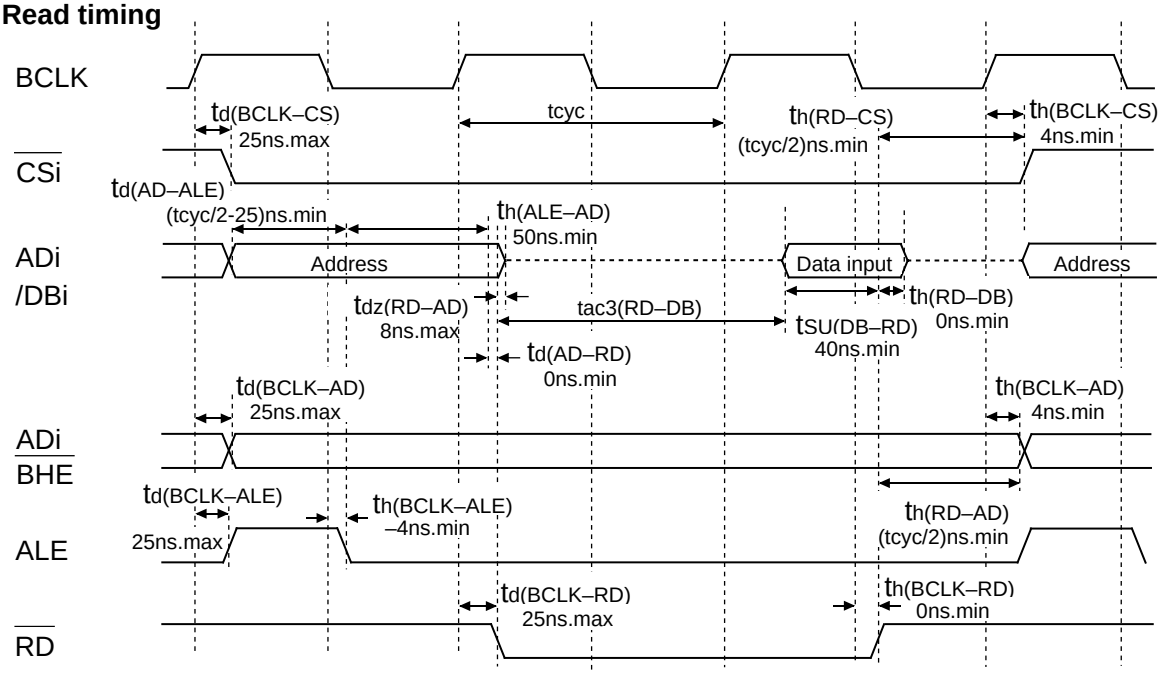
Timing (Vcc = 5V)

## Memory Expansion Mode and Microprocessor Mode

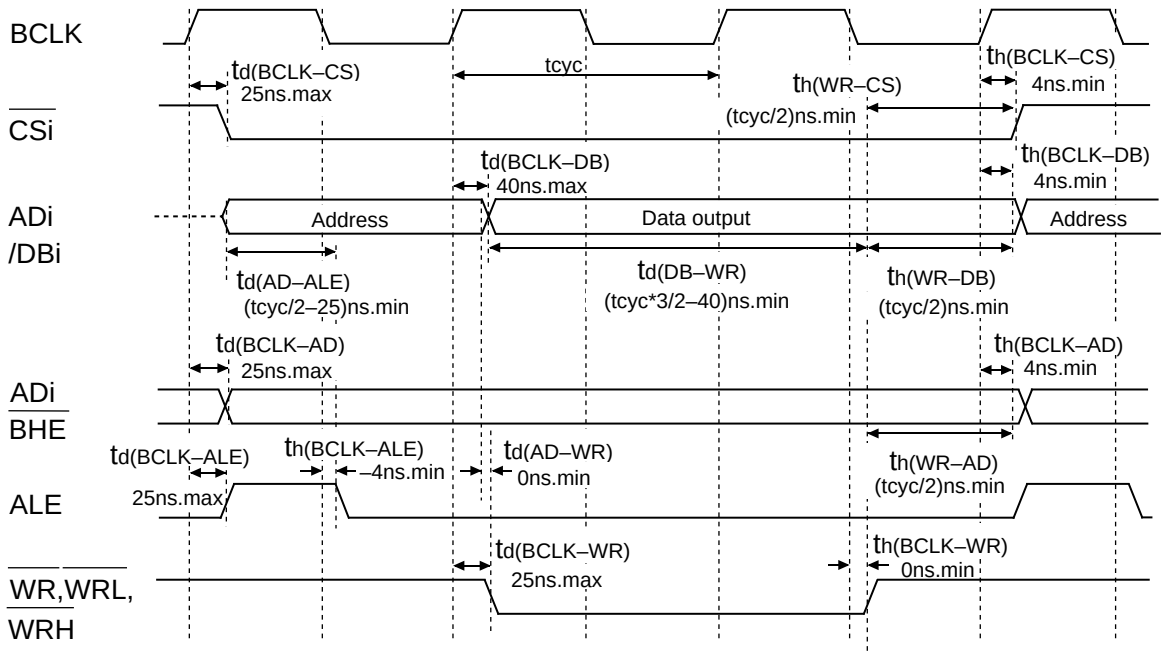
**VCC = 5V**

**(When accessing external memory area with wait, and select multiplexed bus)**

### Read timing




### Write timing




Measuring conditions :

- VCC=5V
- Input timing voltage : Determined with VIL=0.8V, VIH=2.5V
- Output timing voltage : Determined with VOL=0.8V, VOH=2.0V

# Electrical characteristics (Vcc = 3V)

## Vcc = 3V

**Table 1.26.22. Electrical characteristics (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C, f(XIN) = 7MHz(Note 1) with wait)**

| Symbol | Parameter | | Measuring condition | | Standard Min | Standard Typ. | Standard Max. | Unit |
|---|---|---|---|---|---|---|---|---|
| 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=-1mA | | 2.5 | | | V |
| VOH | HIGH output voltage XOUT | HIGHPOWER | IOH=-0.1mA | | 2.5 | | | V |
| | | LOWPOWER | IOH=-50µA | | 2.5 | | | |
| | HIGH output voltage XCOUT | HIGHPOWER | With no load applied | | | 3.0 | | V |
| | | LOWPOWER | With no load applied | | | 1.6 | | |
| 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=1mA | | | | 0.5 | V |
| VOL | LOW output voltage XOUT | HIGHPOWER | IOL=0.1mA | | | | 0.5 | V |
| | | LOWPOWER | IOL=50µA | | | | 0.5 | |
| | LOW output voltage XCOUT | HIGHPOWER | With no load applied | | | 0 | | V |
| | | LOWPOWER | With no load applied | | | 0 | | |
| VT+-VT- | Hysteresis HOLD, RDY, TA0IN to TA4IN, TB0IN to TB2IN, INT0 to INT5, ADTRG, CTS0, CTS1, CLK0, CLK1, TA2OUT to TA4OUT, NMI, KI0 to KI3 | | | | 0.2 | | 0.8 | V |
| VT+-VT- | Hysteresis RESET | | | | 0.2 | | 1.8 | V |
| IIH | 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=3V | | | | 4.0 | µA |
| IIL | 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 | | | | -4.0 | µA |
| RPULLUP | 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 | | 66.0 | 120.0 | 500.0 | kΩ |
| RfXIN | Feedback resistance XIN | | | | | 3.0 | | MΩ |
| RfCXIN | Feedback resistance XCIN | | | | | 10.0 | | MΩ |
| VRAM | RAM retention voltage | | When clock is stopped | | 2.0 | | | V |
| Icc | Power supply current | In single-chip mode, the output pins are open and other pins are Vss | EPROM One-time PROM | f(XIN)=7MHz Square wave, no division | | 6.0 | 15.0 | mA |
| | | | Mask ROM | f(XIN)=10MHz Square wave, no division | | 8.5 | 21.25 | mA |
| | | | | f(XCIN)=32kHz Square wave | | 40.0 | | µA |
| | | | | f(XCIN)=32kHz When a WAITinstruction is executed. Oscillation capacity High (Note2) | | 2.8 | | µA |
| | | | | f(XCIN)=32kHz When a WAIT instruction is executed. Oscillation capacity Low (Note2) | | 0.9 | | µA |
| | | | | Ta=25°C when clock is stopped | | | 1.0 | µA |
| | | | | Ta=85°C when clock is stopped | | | 20.0 | |

Note 1: 10 MHz for the mask ROM version.

Note 2: With one timer operated using fC32.

## Vcc = 3V

**Table 1.26.23. A-D conversion characteristics (referenced to Vcc = AVcc = VREF = 3V, Vss = AVss = 0V at Ta = 25°C, f(XIN) = 7MHz unless otherwise specified)**

| Symbol | Parameter | | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|---|
| | | | | Min. | Typ. | Max | |
| – | Resolution | | VREF = Vcc | | | 10 | Bits |
| – | Absolute accuracy | Sample & hold function not available (8 bit) | VREF = Vcc = 3V, φAD = f(XIN)/2 | | | ±2 | LSB |
| RLADDER | Ladder resistance | | VREF = Vcc | 10 | | 40 | kΩ |
| tCONV | Conversion time(8bit) | | | 14.0 | | | µs |
| VREF | Reference voltage | | | 2.7 | | Vcc | V |
| VIA | Analog input voltage | | | 0 | | VREF | V |

Note: 10 MHz for the mask ROM version.

**Table 1.26.24. D-A conversion characteristics (referenced to Vcc = 3V, Vss = AVss = 0V, VREF = 3V at Ta = 25°C, f(XIN) = 7MHz(Note2) unless otherwise specified)**

| Symbol | Parameter | Measuring condition | Standard | | | Unit |
|---|---|---|---|---|---|---|
| | | | Min. | Typ. | Max | |
| – | Resolution | | | | 8 | Bits |
| – | Absolute accuracy | | | | 1.0 | % |
| tsu | Setup time | | | | 3 | µs |
| RO | Output resistance | | 4 | 10 | 20 | kΩ |
| IVREF | Reference power supply input current | (Note1) | | | 1.0 | mA |

Note 1: This applies when using one D-A converter, with the D-A register for the unused D-A converter set to "$00_{16}$". The A-D converter's ladder resistance is not included.

Also, when the Vref is unconnected at the A-D control register, IVREF is sent.

Note 2: 10 MHz for the mask ROM version.

# Electrical characteristics (Vcc = 3V)

## Vcc = 3V

**Timing requirements (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C unless otherwise specified)**

**Table 1.26.25. External clock input**

| Symbol | Parameter | | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_c$ | External clock input cycle time | EPROM, One-time PROM | 143 | | ns |
| | | Mask ROM | 100 | | ns |
| $t_{w(H)}$ | External clock input HIGH pulse width | EPROM, One-time PROM | 60 | | ns |
| | | Mask ROM | 40 | | ns |
| $t_{w(L)}$ | External clock input LOW pulse width | EPROM, One-time PROM | 60 | | ns |
| | | Mask ROM | 40 | | ns |
| $t_r$ | External clock rise time | | | 18 | ns |
| $t_f$ | External clock fall time | | | 18 | ns |

**Table 1.26.26. Memory expansion and microprocessor modes**

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{ac1(RD-DB)}$ | Data input access time (no wait) | | (Note) | ns |
| $t_{ac2(RD-DB)}$ | Data input access time (with wait) | | (Note) | ns |
| $t_{ac3(RD-DB)}$ | Data input access time (when accessing multiplex bus area) | | (Note) | ns |
| $t_{su(DB-RD)}$ | Data input setup time | 80 | | ns |
| $t_{su(RDY-BCLK)}$ | $\overline{RDY}$ input setup time | 60 | | ns |
| $t_{su(HOLD-BCLK)}$ | $\overline{HOLD}$ input setup time | 80 | | ns |
| $t_{h(RD-DB)}$ | Data input hold time | 0 | | ns |
| $t_{h(BCLK-RDY)}$ | $\overline{RDY}$ input hold time | 0 | | ns |
| $t_{h(BCLK-HOLD)}$ | $\overline{HOLD}$ input hold time | 0 | | ns |
| $t_{d(BCLK-HLDA)}$ | $\overline{HLDA}$ output delay time | | 100 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

# VCC = 3V

**Timing requirements (referenced to VCC = 3V, VSS = 0V at Ta = 25°C unless otherwise specified)**

### Table 1.26.27. Timer A input (counter input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 150 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 60 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 60 | | ns |

### Table 1.26.28. Timer A input (gating input in timer mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 600 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 300 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 300 | | ns |

### Table 1.26.29. Timer A input (external trigger input in one-shot timer mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TA)}$ | TAiIN input cycle time | 300 | | ns |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 150 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 150 | | ns |

### Table 1.26.30. Timer A input (external trigger input in pulse width modulation mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(TAH)}$ | TAiIN input HIGH pulse width | 150 | | ns |
| $t_{w(TAL)}$ | TAiIN input LOW pulse width | 150 | | ns |

### Table 1.26.31. Timer A input (up/down input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(UP)}$ | TAiOUT input cycle time | 3000 | | ns |
| $t_{w(UPH)}$ | TAiOUT input HIGH pulse width | 1500 | | ns |
| $t_{w(UPL)}$ | TAiOUT input LOW pulse width | 1500 | | ns |
| $t_{su(UP-TIN)}$ | TAiOUT input setup time | 600 | | ns |
| $t_{h(TIN-UP)}$ | TAiOUT input hold time | 600 | | ns |

# Electrical characteristics (Vcc = 3V)

## Vcc = 3V

**Timing requirements (referenced to Vcc = 3V, Vss = 0V at Ta = 25°C unless otherwise specified)**

### Table 1.26.32. Timer B input (counter input in event counter mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on one edge) | 150 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on one edge) | 60 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on one edge) | 60 | | ns |
| $t_{c(TB)}$ | TBiIN input cycle time (counted on both edges) | 300 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width (counted on both edges) | 160 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width (counted on both edges) | 160 | | ns |

### Table 1.26.33. Timer B input (pulse period measurement mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 600 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 300 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 300 | | ns |

### Table 1.26.34. Timer B input (pulse width measurement mode)

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(TB)}$ | TBiIN input cycle time | 600 | | ns |
| $t_{w(TBH)}$ | TBiIN input HIGH pulse width | 300 | | ns |
| $t_{w(TBL)}$ | TBiIN input LOW pulse width | 300 | | ns |

### Table 1.26.35. A-D trigger input

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{c(AD)}$ | $\overline{ADTRG}$ input cycle time (trigger able minimum) | 1500 | | ns |
| $t_{w(ADL)}$ | $\overline{ADTRG}$ input LOW pulse width | 200 | | ns |

### Table 1.26.36. Serial I/O

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

### Table 1.26.37. External interrupt $\overline{INTi}$ inputs

| Symbol | Parameter | Standard | | Unit |
|---|---|---|---|---|
| | | Min. | Max. | |
| $t_{w(INH)}$ | $\overline{INTi}$ input HIGH pulse width | 380 | | ns |
| $t_{w(INL)}$ | $\overline{INTi}$ 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 1.26.38. Memory expansion and microprocessor modes (with no wait)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.26.1 | | 60 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | 0 | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 60 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 60 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 4 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 60 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 60 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 80 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note1) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard)(Note2) | | 0 | | ns |

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

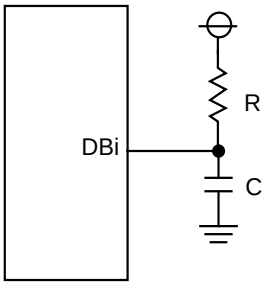
$$td(DB - WR) = \frac{10^9}{f(BCLK) \times 2} - 80 \quad [ns]$$

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 = 30pF, R = 1kΩ, hold time of output "L" level is

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



# VCC = 3V

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

**Table 1.26.39. Memory expansion and microprocessor modes (when accessing external memory area with wait)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.26.1 | | 60 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | 0 | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | 0 | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 60 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time | | | 60 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time | | – 4 | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 60 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 60 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 80 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note1) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard)(Note2) | | 0 | | ns |

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

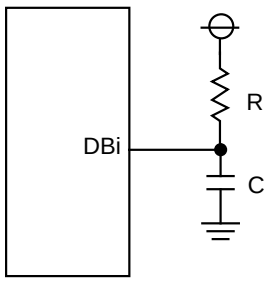
$$td(DB - WR) = \frac{10^9}{f(BCLK)} - 80 \quad [ns]$$

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 = 30pF, R = 1kΩ, hold time of output "L" level is

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



# VCC = 3V

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

**Table 1.26.40. Memory expansion and microprocessor modes (when accessing external memory area with wait, and select multiplexed bus)**

| Symbol | Parameter | Measuring condition | Standard | | Unit |
|---|---|---|---|---|---|
| | | | Min. | Max. | |
| $t_{d(BCLK-AD)}$ | Address output delay time | Figure 1.26.1 | | 60 | ns |
| $t_{h(BCLK-AD)}$ | Address output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-AD)}$ | Address output hold time (RD standard) | | (Note) | | ns |
| $t_{h(WR-AD)}$ | Address output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-CS)}$ | Chip select output delay time | | | 60 | ns |
| $t_{h(BCLK-CS)}$ | Chip select output hold time (BCLK standard) | | 4 | | ns |
| $t_{h(RD-CS)}$ | Chip select output hold time (RD standard) | | (Note) | | ns |
| $t_{h(WR-CS)}$ | Chip select output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-RD)}$ | RD signal output delay time | | | 60 | ns |
| $t_{h(BCLK-RD)}$ | RD signal output hold time | | 0 | | ns |
| $t_{d(BCLK-WR)}$ | WR signal output delay time | | | 60 | ns |
| $t_{h(BCLK-WR)}$ | WR signal output hold time | | 0 | | ns |
| $t_{d(BCLK-DB)}$ | Data output delay time (BCLK standard) | | | 80 | ns |
| $t_{h(BCLK-DB)}$ | Data output hold time (BCLK standard) | | 4 | | ns |
| $t_{d(DB-WR)}$ | Data output delay time (WR standard) | | (Note) | | ns |
| $t_{h(WR-DB)}$ | Data output hold time (WR standard) | | (Note) | | ns |
| $t_{d(BCLK-ALE)}$ | ALE signal output delay time (BCLK standard) | | | 60 | ns |
| $t_{h(BCLK-ALE)}$ | ALE signal output hold time (BCLK standard) | | – 4 | | ns |
| $t_{d(AD-ALE)}$ | ALE signal output delay time (Address standard) | | (Note) | | ns |
| $t_{h(ALE-AD)}$ | ALE signal output hold time(Address standard) | | 50 | | ns |
| $t_{d(AD-RD)}$ | Post-address RD signal output delay time | | 0 | | ns |
| $t_{d(AD-WR)}$ | Post-address WR signal output delay time | | 0 | | ns |
| $t_{dZ(RD-AD)}$ | Address output floating start time | | | 8 | ns |

Note: Calculated according to the BCLK frequency as follows:

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

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

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

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

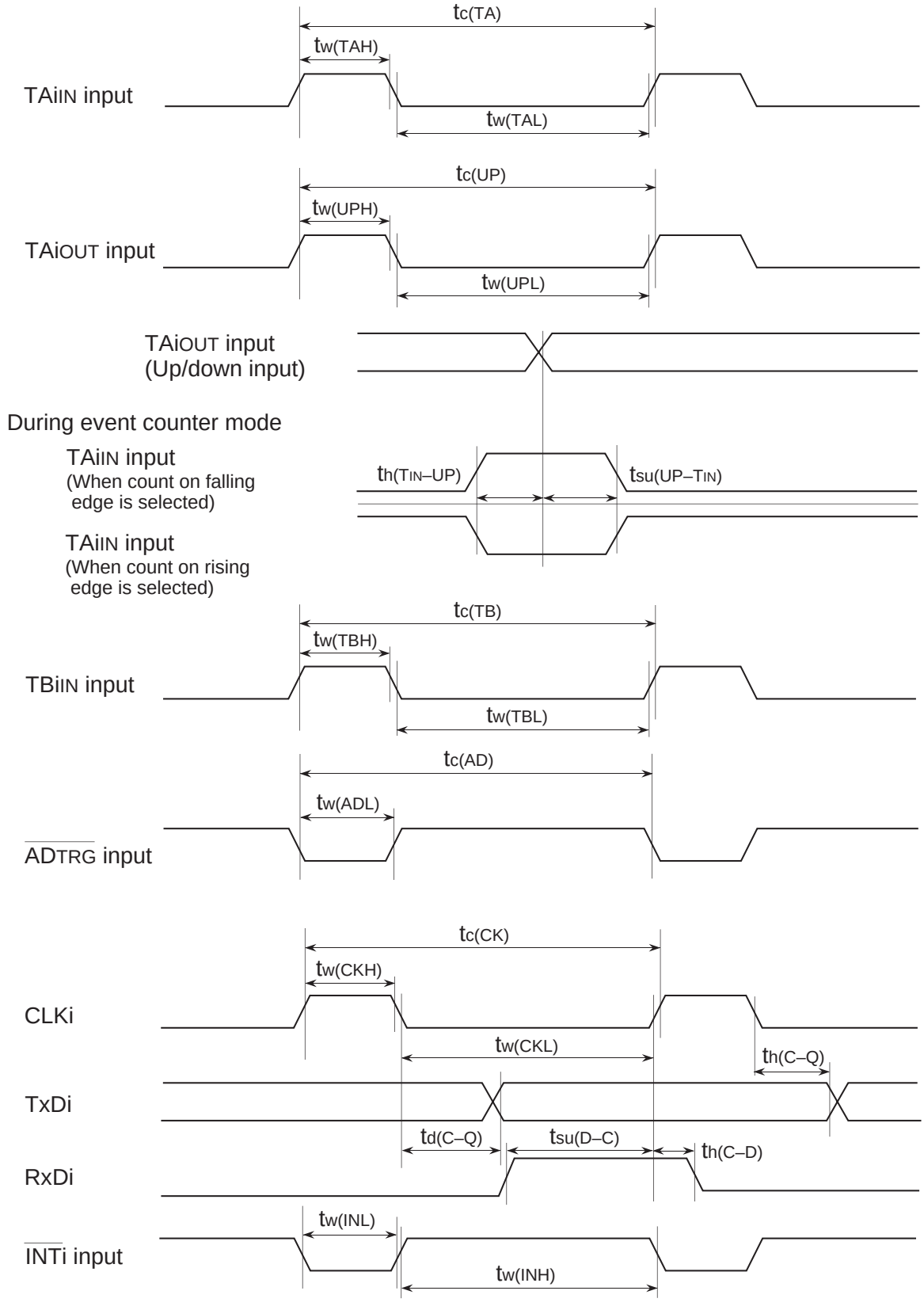
$$td(DB - WR) = \frac{10^9 \times 3}{f(BCLK) \times 2} - 80 \quad [ns]$$

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

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

## Timing (Vcc = 3V)

### Vcc = 3V

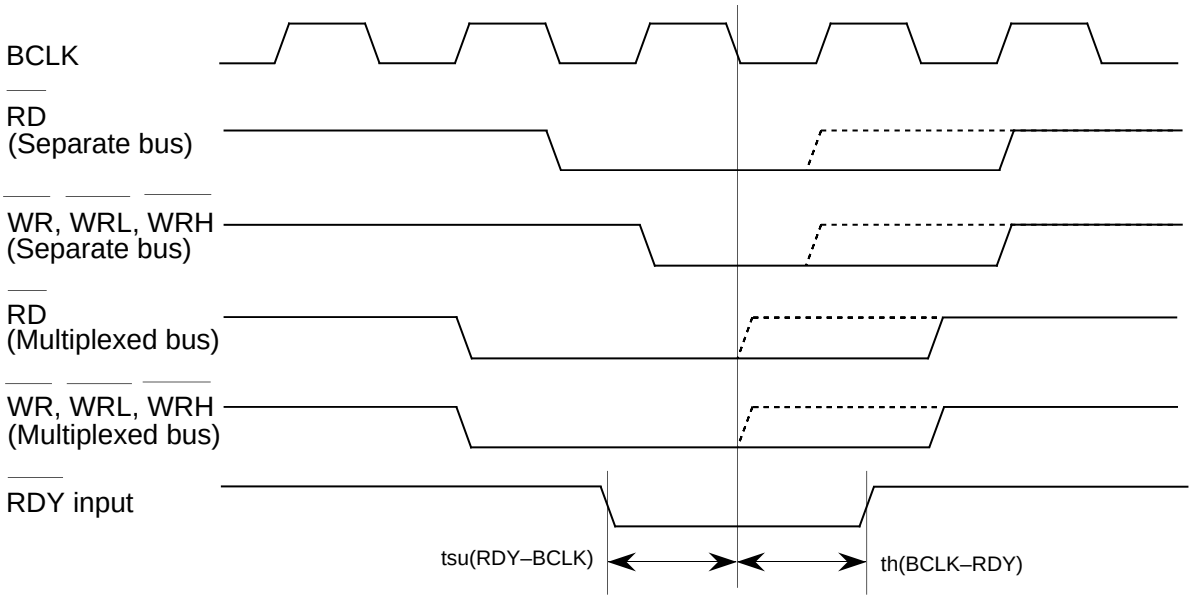


Timing (Vcc = 3V)

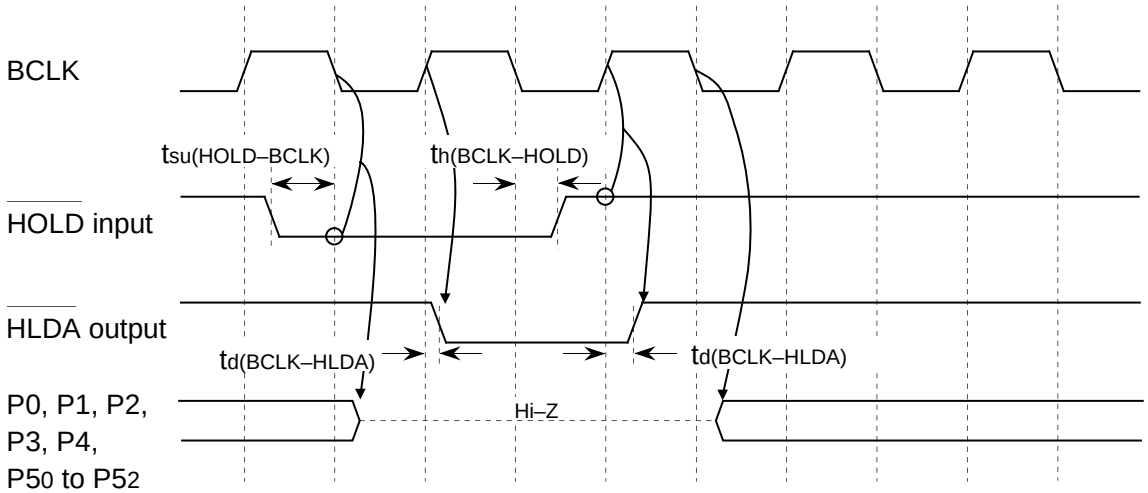
## Vcc = 3V

### Memory Expansion Mode and Microprocessor Mode

**(Valid only with wait)**




**(Valid with or without wait)**




Note: The above pins are set to high-impedance regardless of the input level of the BYTE pin and bit (PM06) of processor mode register 0 selects the function of ports P40 to P43.

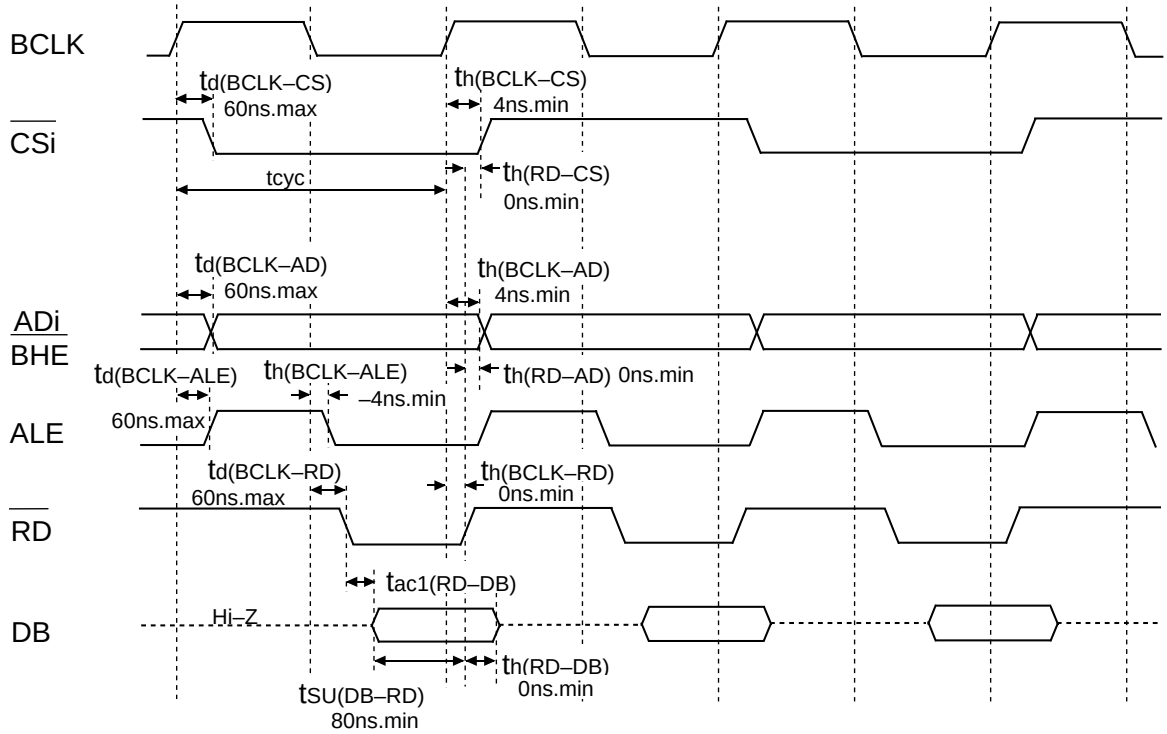
Measuring conditions :

- VCC=3V
- Input timing voltage : Determined with VIL=0.6V, VIH=2.4V
- Output timing voltage : Determined with VOL=1.5V, VOH=1.5V

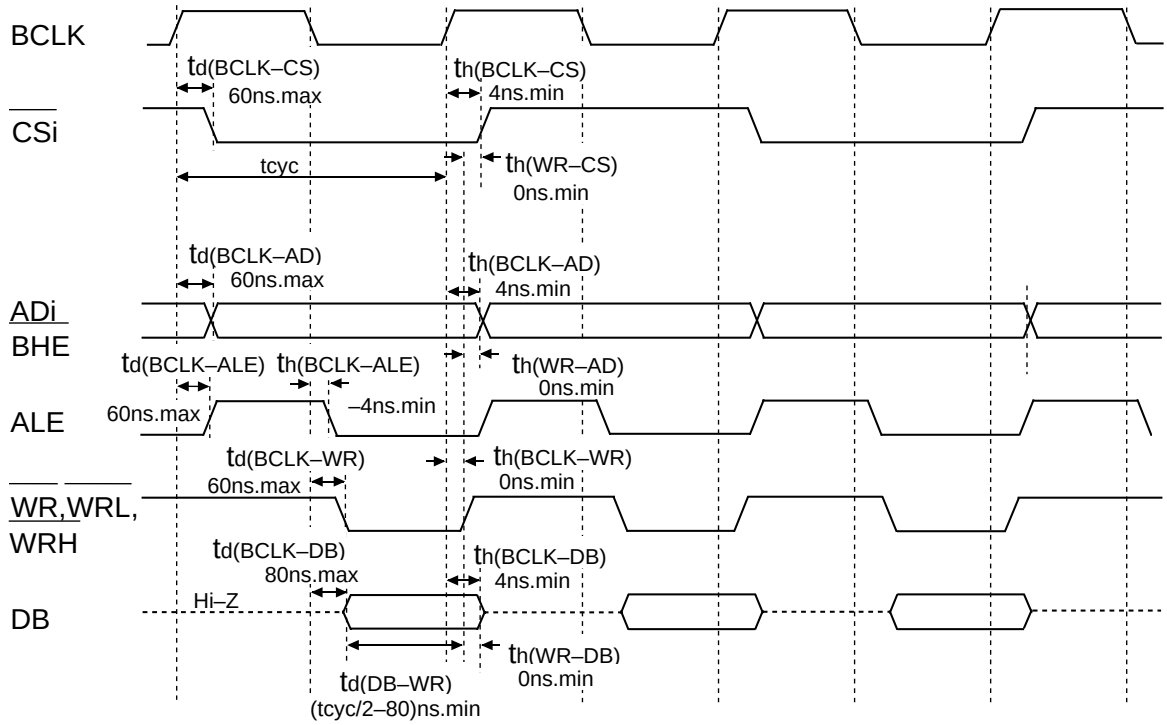
## Memory Expansion Mode and Microprocessor Mode
**(With no wait)**

**Vcc = 3V**

### Read timing




### Write timing



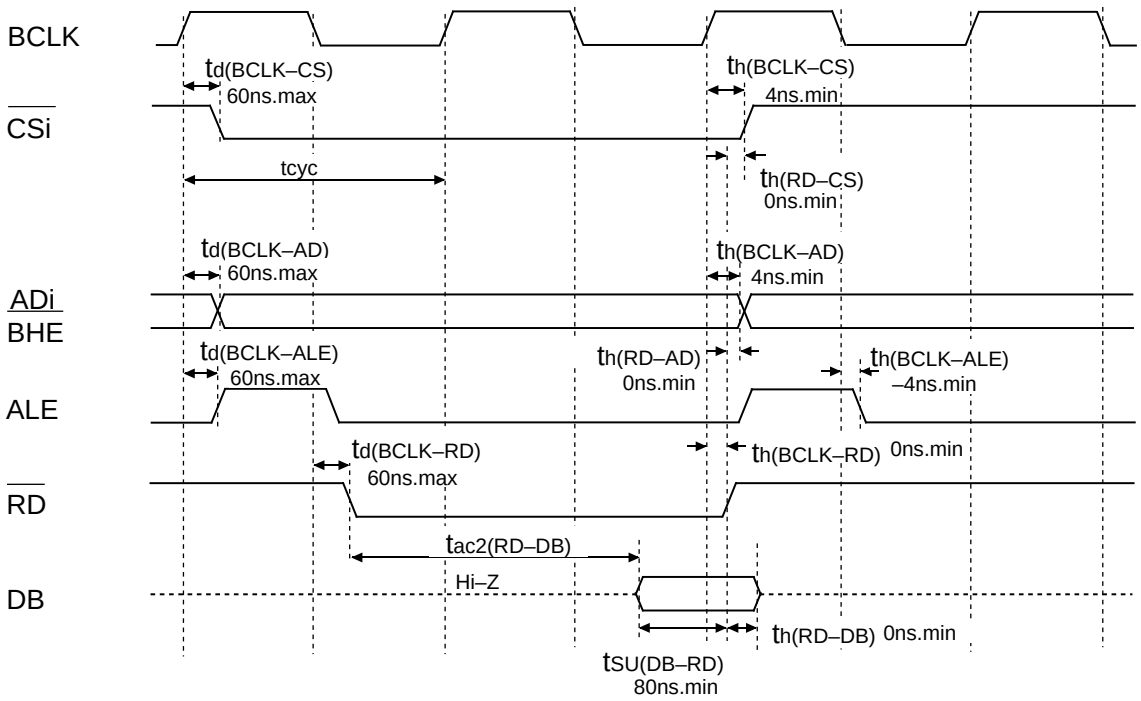
Timing (Vcc = 3V)

## Memory Expansion Mode and Microprocessor Mode

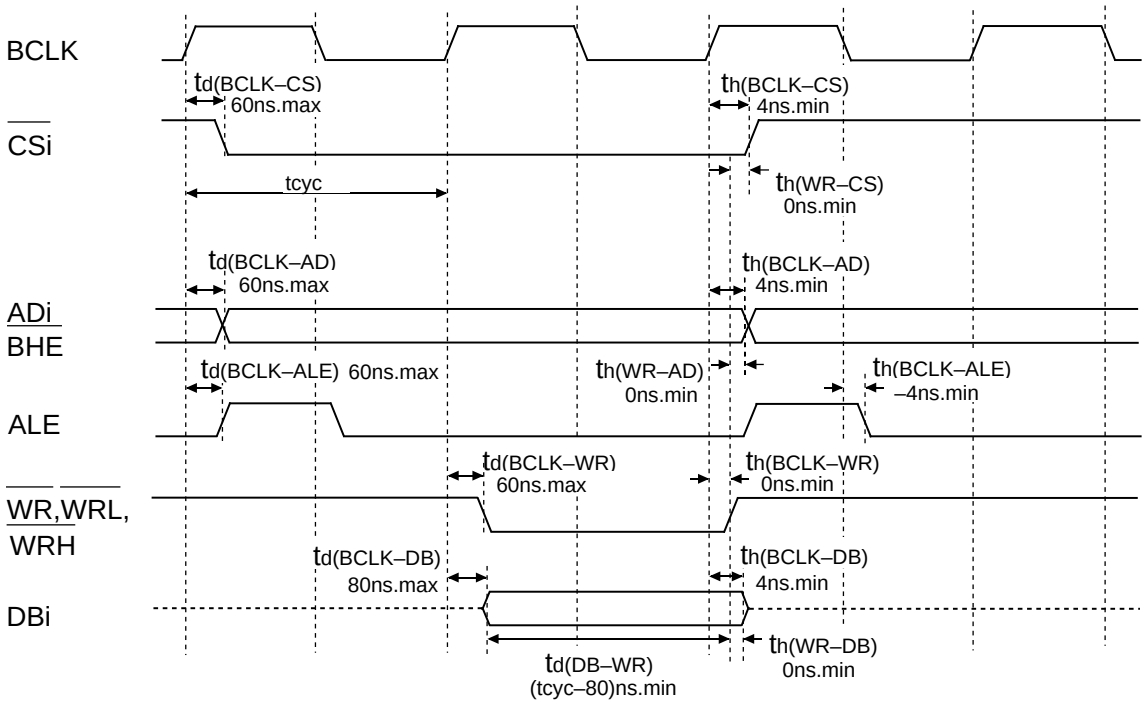
**Vcc = 3V**

**(When accessing external memory area with wait)**

**Read timing**




**Write timing**




Measuring conditions :

- Vcc=3V
- Input timing voltage : Determined with VIL=0.48V, VIH=1.5V
- Output timing voltage : Determined with VOL=1.5V, VOH=1.5V

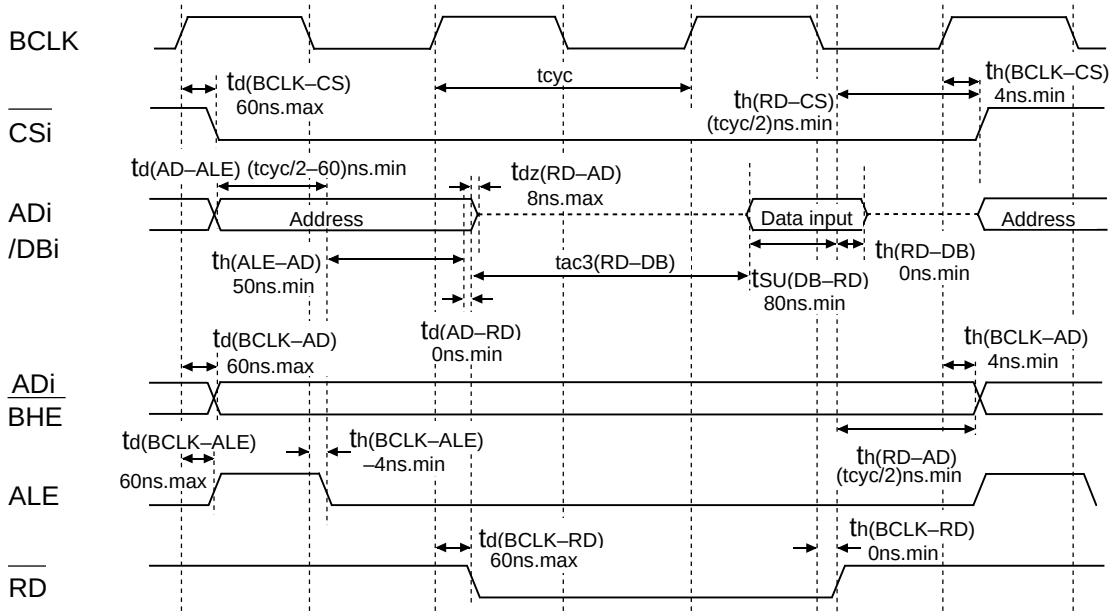
Timing (Vcc = 3V)

## Vcc = 3V

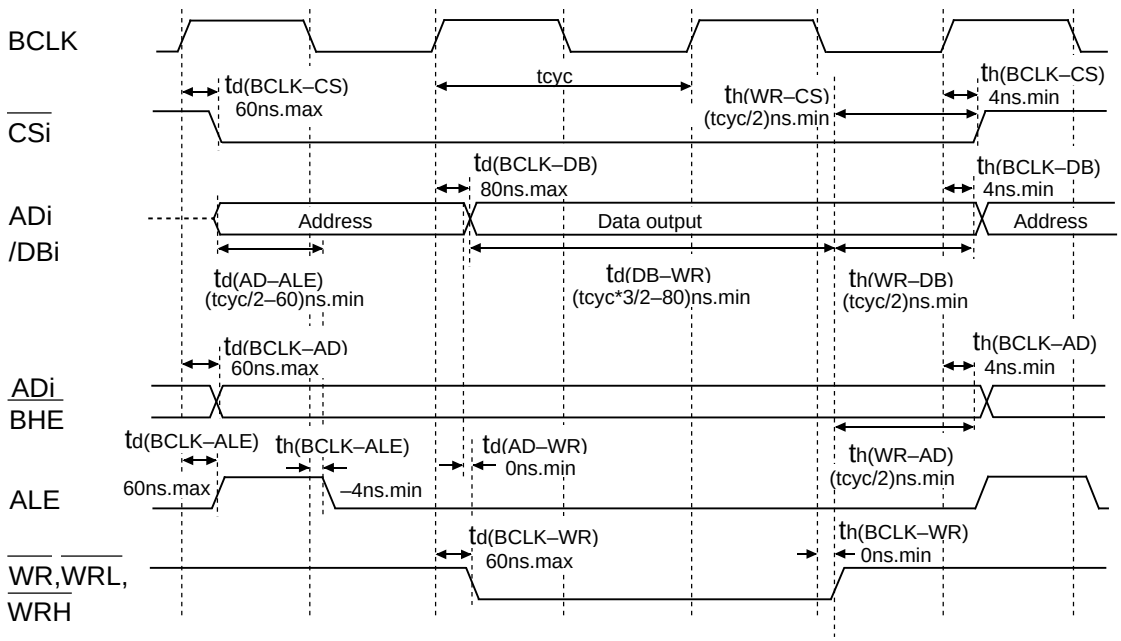
### Memory Expansion Mode and Microprocessor Mode

**(When accessing external memory area with wait, and select multiplexed bus)**

**Read timing**




**Write timing**




Measuring conditions :

- Vcc=3V
- Input timing voltage : Determined with VIL=0.48V,VIH=1.5V
- Output timing voltage : Determined with VOL=1.5V,VOH=1.5V

## Differences between M30622MC and M30612MC

| Type Name | M30622MC | M30612MC |
|---|---|---|
| Memory space | Memory expansion is possible<br>1.2M bytes mode<br>4M bytes mode | 1M byte fixed |
| Timer B | 6 channels | 3 channels |
| Serial I/O | UART/clocked SI/O · · · · · 3 channel<br>Clocked SI/O · · · · · · · · · · 2 channel | UART/clocked SI/O · · · · · 3 channels |
| IIC bus mode | UART2 used<br>IIC bus interface can be performed with software | Impossible |
| Port function | $P9_0$ · · · · · TB0IN/CLK3<br>$P9_1$ · · · · · TB1IN/SIN3<br>$P9_2$ · · · · · TB2IN/SOUT3<br>$P9_3$ · · · · · TB3IN/DA0<br>$P9_4$ · · · · · TB4IN/DA1<br>$P9_5$ · · · · · ANEX0/CLK4<br>$P9_6$ · · · · · ANEX1/SOUT4<br>$P9_7$ · · · · · ADTRG/SIN4<br>$P1_5$ · · · · · D13/INT3<br>$P1_6$ · · · · · D14/INT4<br>$P1_7$ · · · · · D15/INT5<br>$P7_1$ · · · · · RxD2/TA0IN/TB5IN | $P9_0$ · · · · · TB0IN<br>$P9_1$ · · · · · TB1IN<br>$P9_2$ · · · · · TB2IN<br>$P9_3$ · · · · · DA0<br>$P9_4$ · · · · · DA1<br>$P9_5$ · · · · · ANEX0<br>$P9_6$ · · · · · ANEX1<br>$P9_7$ · · · · · ADTRG<br>$P1_5$ · · · · · D13<br>$P1_6$ · · · · · D14<br>$P1_7$ · · · · · D15<br>$P7_1$ · · · · · RxD2/TA0IN |
| Interrupt cause | Internal 25 sources<br>External 8 sources<br>Software 4 sources<br>(Added two Serial I/O, three timers and 3external interrupts) | Internal 20 sources<br>External 5 sources<br>Software 4 sources |
| Chip select | M30612MC type and the type as below can be switched<br>(Besides 4M-byte mode is possible.)<br>CS0 : $04000_{16}$ to $3FFFF_{16}$ (fetch)<br>$40000_{16}$ to $FFFFF_{16}$ (data/facth)<br>CS1 : $28000_{16}$ to $2FFFF_{16}$ (data)<br>CS2 : $08000_{16}$ to $27FFF_{16}$ (data)<br>CS3 : $04000_{16}$ to $07FFF_{16}$ (data) | CS0 : $30000_{16}$ to $FFFFF_{16}$<br>CS1 : $28000_{16}$ to $2FFFF_{16}$<br>CS2 : $08000_{16}$ to $27FFF_{16}$<br>CS3 : $04000_{16}$ to $07FFF_{16}$ |
| Three-phase inverter control circuit | PWM output for three-phase inverter can be performed using timer A4, A1 and A2.<br>Output port is arranged to $P7_2$ to $P7_5$, $P8_0$ and $P8_1$. | Impossible |
| Read port P1 | By setting to register, the state of port register can be read always. | The state of port when input mode.<br>The state of port register when output mode. |
| $P4_4$/CS0 - $P4_7$/CS3 pin pull-up resistors | If a Vcc level is applied to the CNVss pin, bit 2 (PU11) of pull-up control register 1 turns to "1" when reset, and $P4_4$/ $\overline{CS0}$ - $P4_7$/ $\overline{CS3}$ turn involved in pull-up. | Bit 2 (PU11) of the pull-up control register 1 turns to "0" when reset, and $P4_4$/ $\overline{CS0}$ - $P4_7$/ $\overline{CS3}$ turn free from pull-up. |

MITSUBISHI SEMICONDUCTORS
M16C/62 Group Tentative Specification REV.C

Aug First Edition 1998

Editioned by
Committee of editing of Mitsubishi Semiconductor

Published by
Mitsubishi Electric Corp., Kitaitami Works